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

A. AIM: Create an application that obtains four int values from the user and displays the product.

using System;

namespace ConsoleApp

{

class Program

{

static void Main(string[] args)

{

int num1, num2, num3, num4, prod;

Console.Write("Enter number 1: ");

num1 = int.Parse(Console.ReadLine());

Console.Write("Enter number 2: ");

num2 = int.Parse(Console.ReadLine());

Console.Write("Enter number 3: ");

num3 = int.Parse(Console.ReadLine());

Console.Write("Enter number 4: ");

num4 = int.Parse(Console.ReadLine());

prod = num1 \* num2 \* num3 \* num4;

Console.WriteLine(num1 + "\*" + num2 + "\*" + num3 + "\*" + num4 + "=" + prod);

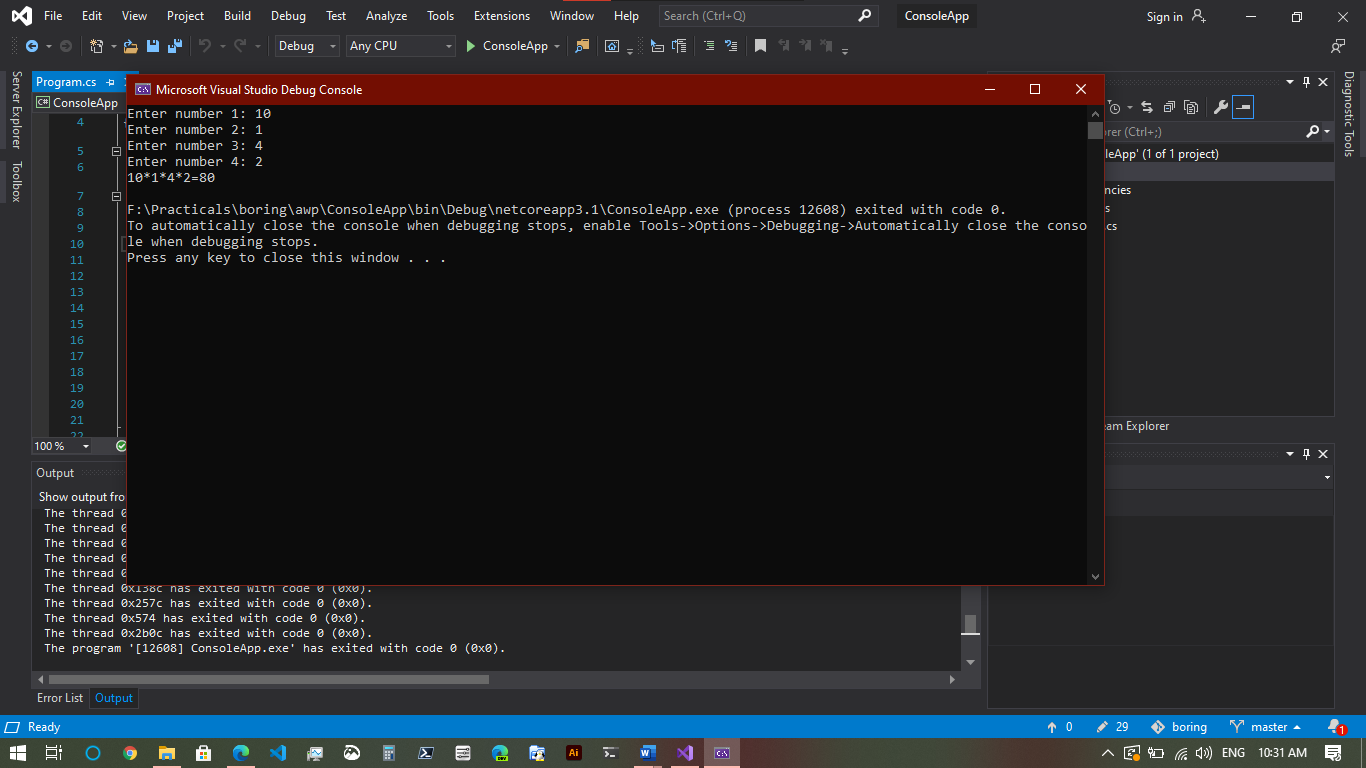
Console.ReadKey();

}

}

}

OUTPUT:



B. AIM: Create an application to demonstrate string operations.

using System;

using System.Text;

namespace StringOperate

{

class Program

{

static void Main(string[] args)

{

// Create a StringBuilder that expects to hold 50 characters.

// Initialize the StringBuilder with "ABC".

StringBuilder sb = new StringBuilder("ABC", 50);

// Append three characters (D, E, and F) to the end of the StringBuilder.

sb.Append(new char[] { 'D', 'E', 'F' });

// Append a format string to the end of the StringBuilder.

sb.AppendFormat("GHI{0}{1}", 'J', 'k');

// Display the number of characters in the StringBuilder and its string.

Console.WriteLine("{0} chars: {1}", sb.Length, sb.ToString());

// Insert a string at the beginning of the StringBuilder.

sb.Insert(0, "Alphabet: ");

// Replace all lowercase k's with uppercase K's.

sb.Replace('k', 'K');

// Display the number of characters in the StringBuilder and its string.

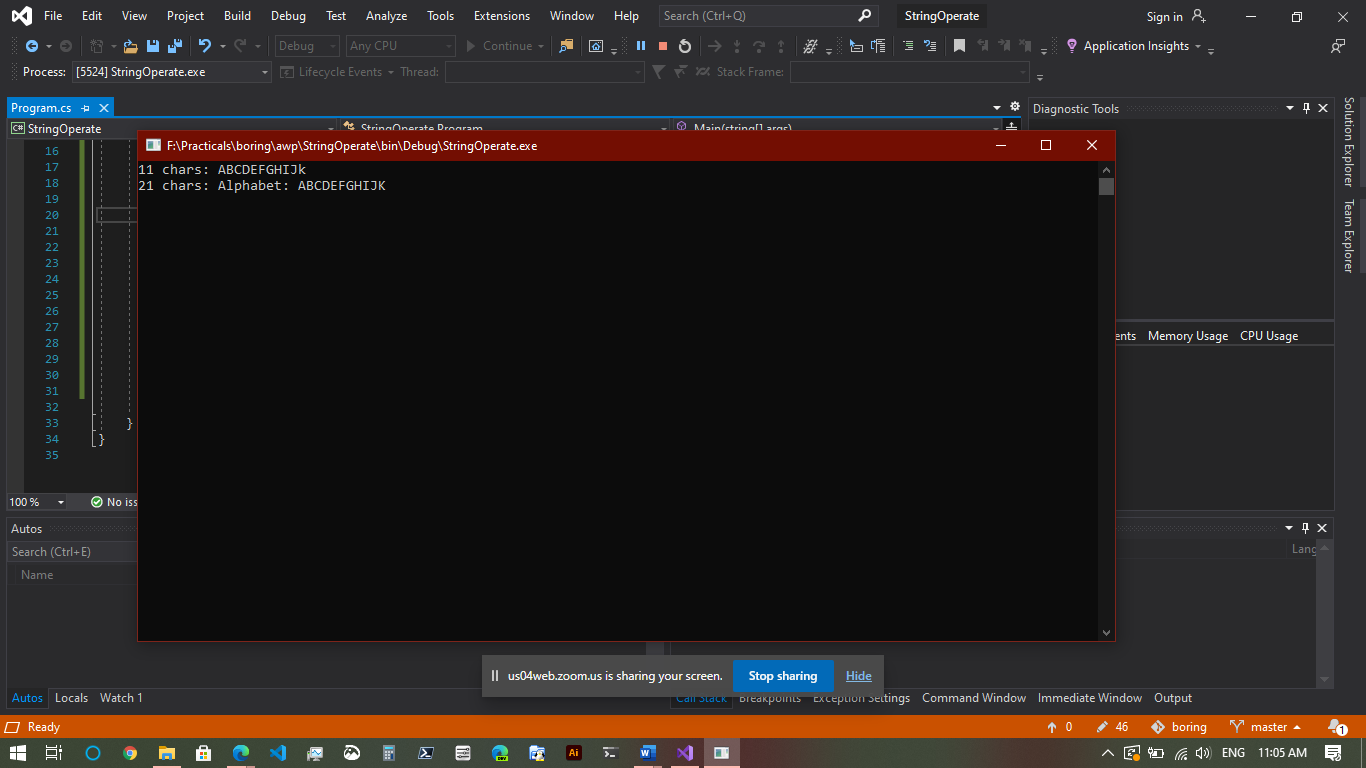
Console.WriteLine("{0} chars: {1}", sb.Length, sb.ToString());

Console.ReadKey();

}

}

}



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

using System;

namespace StudentData

{

class Program

{

struct Student

{

public string studid, name, cname;

public int day, month, year;

}

static void Main(string[] args)

{

int numberOfEntries = 2;

Student[] s = new Student[numberOfEntries];

int i;

for (i = 0; i < numberOfEntries; i++)

{

Console.Write("Enter Student Id:");

s[i].studid = Console.ReadLine();

Console.Write("Enter Student name : ");

s[i].name = Console.ReadLine();

Console.Write("Enter Course name : ");

s[i].cname = Console.ReadLine();

Console.Write("Enter date of birth\n Enter day(1-31):");

s[i].day = Convert.ToInt32(Console.ReadLine());

Console.Write("Enter month(1-12):");

s[i].month = Convert.ToInt32(Console.ReadLine());

Console.Write("Enter year:");

s[i].year = Convert.ToInt32(Console.ReadLine());

}

Console.WriteLine("\n\nStudent's List\n");

for (i = 0; i < numberOfEntries; i++)

{

Console.WriteLine("\nStudent ID : " + s[i].studid);

Console.WriteLine("\nStudent name : " + s[i].name);

Console.WriteLine("\nCourse name : " + s[i].cname);

Console.WriteLine("\nDate of birth(dd-mm-yy) : " + s[i].day + "-" + s[i].month +

"-" + s[i].year);

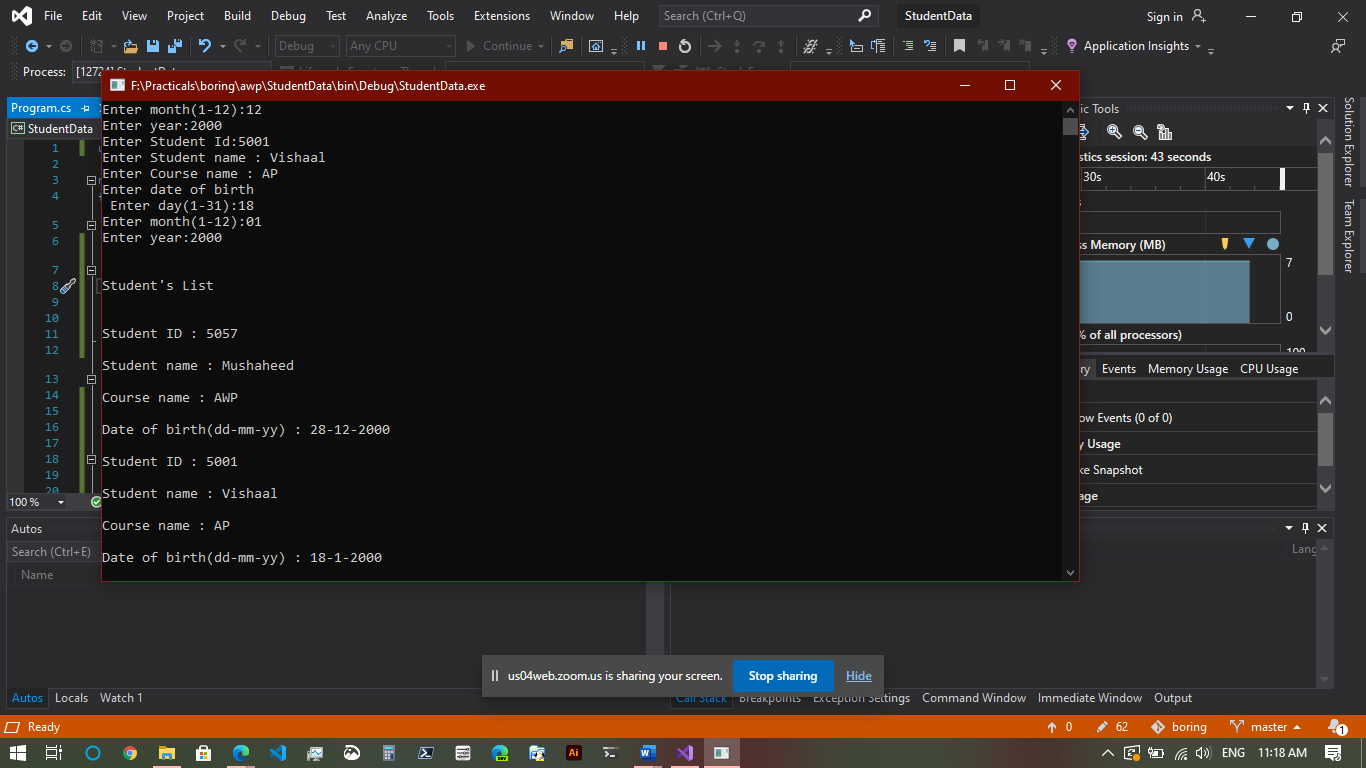
}

Console.ReadKey();

}

}

}



D. AIM: Create an application to demonstrate following operation:

i. Generate Fibonacci series, ii. Test for Prime numbers, iii. Test for vowels, iv. Use of foreach with arrays, v. Reverse a number & find sum of digits of a number.

using System;

namespace Operations1D

{

class Program

{

static void GenerateFibonacci()

{

Console.WriteLine("FIBONACCI SERIES");

int num1 = 0, num2 = 1, num3, limit, counter;

Console.Write("Upto how many number you want fibonacci series: ");

limit = int.Parse(Console.ReadLine());

counter = 3;

Console.Write(num1 + " " + num2);

while (counter <= limit)

{

num3 = num1 + num2;

if (counter >= limit)

break;

Console.Write(" " + num3);

num1 = num2;

num2 = num3;

counter++;

}

}

static void TestPrime()

{

Console.WriteLine("\n\nTest for Prime numbers");

int num, counter;

Console.Write("Enter number:");

num = int.Parse(Console.ReadLine());

for (counter = 2; counter <= num / 2; counter++)

{

if ((num % counter) == 0)

break;

}

if (num == 1)

Console.WriteLine(num + " is neither prime nor composite");

else if (counter < (num / 2))

Console.WriteLine(num + " is not prime number");

else

Console.WriteLine(num + " is prime number");

}

static void TestVowels()

{

Console.WriteLine("\nTest Vowels");

char ch;

Console.Write("Enter a character: ");

ch = (char) Console.ReadLine()[0];

switch (ch)

{

case 'a':

case 'A':

case 'e':

case 'E':

case 'i':

case 'I':

case 'o':

case 'O':

case 'u':

case 'U':

Console.WriteLine(ch + " is vowel");

break;

default:

Console.Write(ch + " is not a vowel");

break;

}

}

static void ForeachArrays()

{

Console.WriteLine("\nForeach Arrays");

string[] str = { "One", "Two", "Three" };

foreach (String s in str)

{

Console.WriteLine(s);

}

}

static void ReverseNSumNumber()

{

Console.WriteLine("\nReverse Number");

int num, actualnumber, revnum = 0, digit, sumDigits = 0;

Console.Write("Enter number: ");

num = int.Parse(Console.ReadLine());

actualnumber = num;

while (num > 0)

{

digit = num % 10;

revnum = revnum \* 10 + digit;

sumDigits = sumDigits + digit;

num = num / 10;

}

Console.WriteLine("Reverse of " + actualnumber + "=" + revnum);

Console.WriteLine("Sum of its digits: " + sumDigits);

}

static void Main(string[] args)

{

GenerateFibonacci();

TestPrime();

TestVowels();

ForeachArrays();

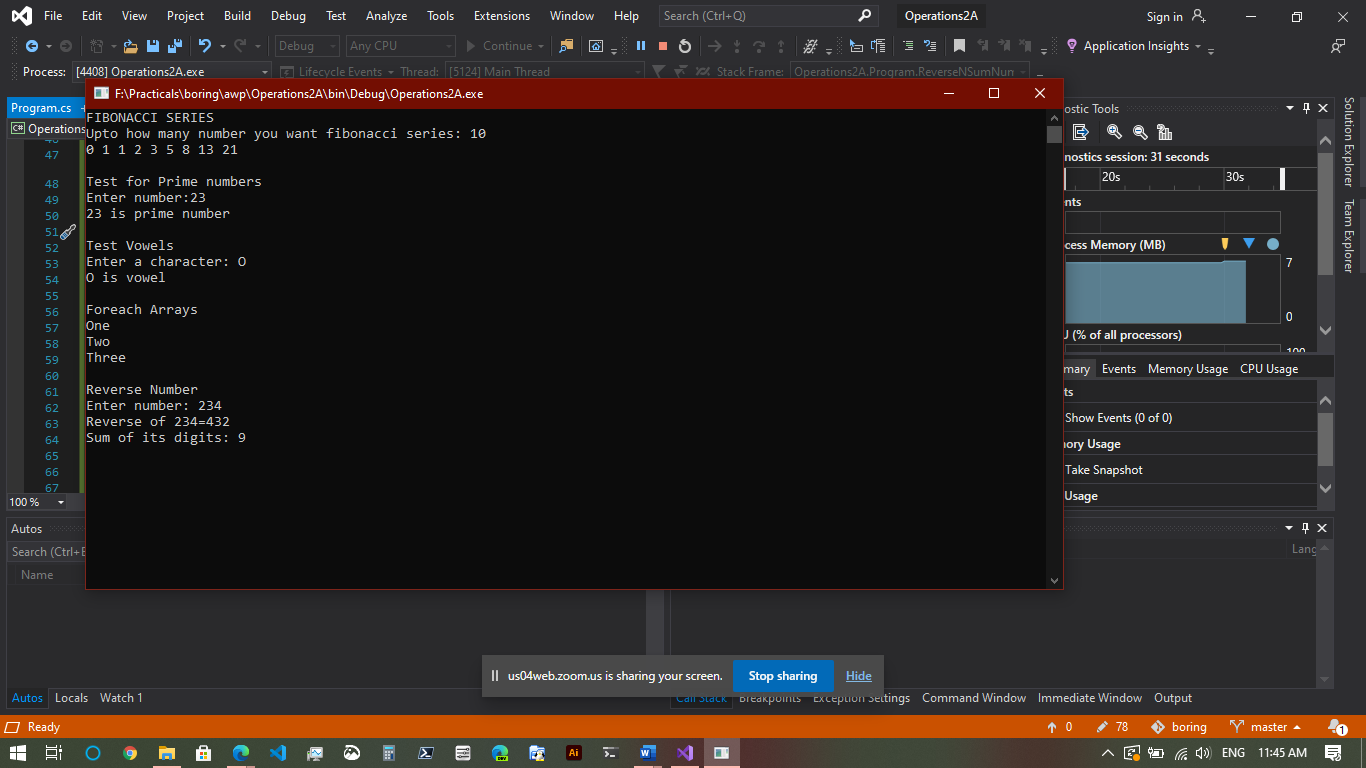
ReverseNSumNumber();

Console.ReadKey();

}

}

}



Practical 2: Working with Web Forms & Controls

A. AIM: Create simple application to perform following operations

i. Finding Factorial value, ii. Money conversion, iii. Quadratic Equation, iv. Temperature conversion

using System;

namespace Operations2A

{

class Quadraticroots

{

double a, b, c;

public void read()

{

Console.WriteLine("To find the roots of a quadratic equation of the form a\*x\*x + b\*x + c = 0");

Console.Write("Enter value for a: ");

a = double.Parse(Console.ReadLine());

Console.Write("Enter value for b: ");

b = double.Parse(Console.ReadLine());

Console.Write("Enter value for c: ");

c = double.Parse(Console.ReadLine());

}

public void compute()

{

int m;

double r1, r2, d1;

d1 = b \* b - 4 \* a \* c;

if (a == 0)

m = 1;

else if (d1 > 0)

m = 2;

else if (d1 == 0)

m = 3;

else

m = 4;

switch (m)

{

case 1:

Console.WriteLine("Not a Quadratic equation, Linear equation");

Console.ReadLine();

break;

case 2:

Console.WriteLine("Roots are Real and Distinct");

r1 = (-b + Math.Sqrt(d1)) / (2 \* a);

r2 = (-b - Math.Sqrt(d1)) / (2 \* a);

Console.WriteLine("First root is {0:#.##}", r1);

Console.WriteLine("Second root is {0:#.##}", r2);

Console.ReadLine();

break;

case 3:

Console.WriteLine("Roots are Real and Equal");

r1 = r2 = (-b) / (2 \* a);

Console.WriteLine("First root is {0:#.##}", r1);

Console.WriteLine("Second root is {0:#.##}", r2);

Console.ReadLine();

break;

case 4:

Console.WriteLine("\n Roots are Imaginary");

r1 = (-b) / (2 \* a);

r2 = Math.Sqrt(-d1) / (2 \* a);

Console.WriteLine("\n First root is {0:#.##} + i {1:#.##}", r1, r2);

Console.WriteLine("\n Second root is {0:#.##} - i {1:#.##}", r1, r2);

Console.ReadLine();

break;

}

}

}

class Program

{

static void findFactorial()

{

Console.WriteLine("Find factorial");

int i, number, fact;

Console.Write("Enter the Number: ");

number = int.Parse(Console.ReadLine());

fact = number;

for (i = number - 1; i >= 1; i--)

{

fact = fact \* i;

}

Console.WriteLine("\nFactorial of Given Number is: " + fact);

Console.ReadKey();

}

static void moneyConversion()

{

Console.WriteLine("\nMoney conversion: ");

int choice;

Console.Write("Choices:\n1 - Dollar to Rupee\n2 - Euro to Rupee\n3 - Malaysian Ringgit to Rupee\nEnter your Choice: ");

choice = int.Parse(Console.ReadLine());

switch (choice)

{

case 1:

Double dollar, rupee, val;

Console.Write("Enter the Dollar Amount: ");

dollar = Double.Parse(Console.ReadLine());

Console.Write("Enter the Dollar Value: ");

val = double.Parse(Console.ReadLine());

rupee = dollar \* val;

Console.WriteLine("{0} Dollar equals to {1} Rupees", dollar, rupee);

break;

case 2:

Double Euro, rupe, valu;

Console.Write("Enter the Euro Amount: ");

Euro = Double.Parse(Console.ReadLine());

Console.Write("Enter the Euro Value: ");

valu = double.Parse(Console.ReadLine());

rupe = Euro \* valu;

Console.WriteLine("{0} Euro equals to {1} Rupees ", Euro, rupe);

break;

case 3:

Double ringit, rup, value;

Console.Write("Enter the Ringgit Amount: ");

ringit = Double.Parse(Console.ReadLine());

Console.Write("Enter the Ringgit Value: ");

value = double.Parse(Console.ReadLine());

rup = ringit \* value;

Console.WriteLine("{0}Malaysian Ringgit equals to {1} Rupees", ringit, rup);

break;

}

}

static void quadraticRoots()

{

Console.WriteLine("\nQuadratic roots");

Quadraticroots qr = new Quadraticroots();

qr.read();

qr.compute();

}

static void temperatureConversion()

{

Console.WriteLine("\nTemperatue conversion");

int celsius, faren;

Console.Write("Enter the Temperature in Celsius(°C): ");

celsius = int.Parse(Console.ReadLine());

faren = (celsius \* 9) / 5 + 32;

Console.WriteLine("Temperature in Fahrenheit is(°F): " + faren);

}

static void Main(string[] args)

{

findFactorial();

moneyConversion();

quadraticRoots();

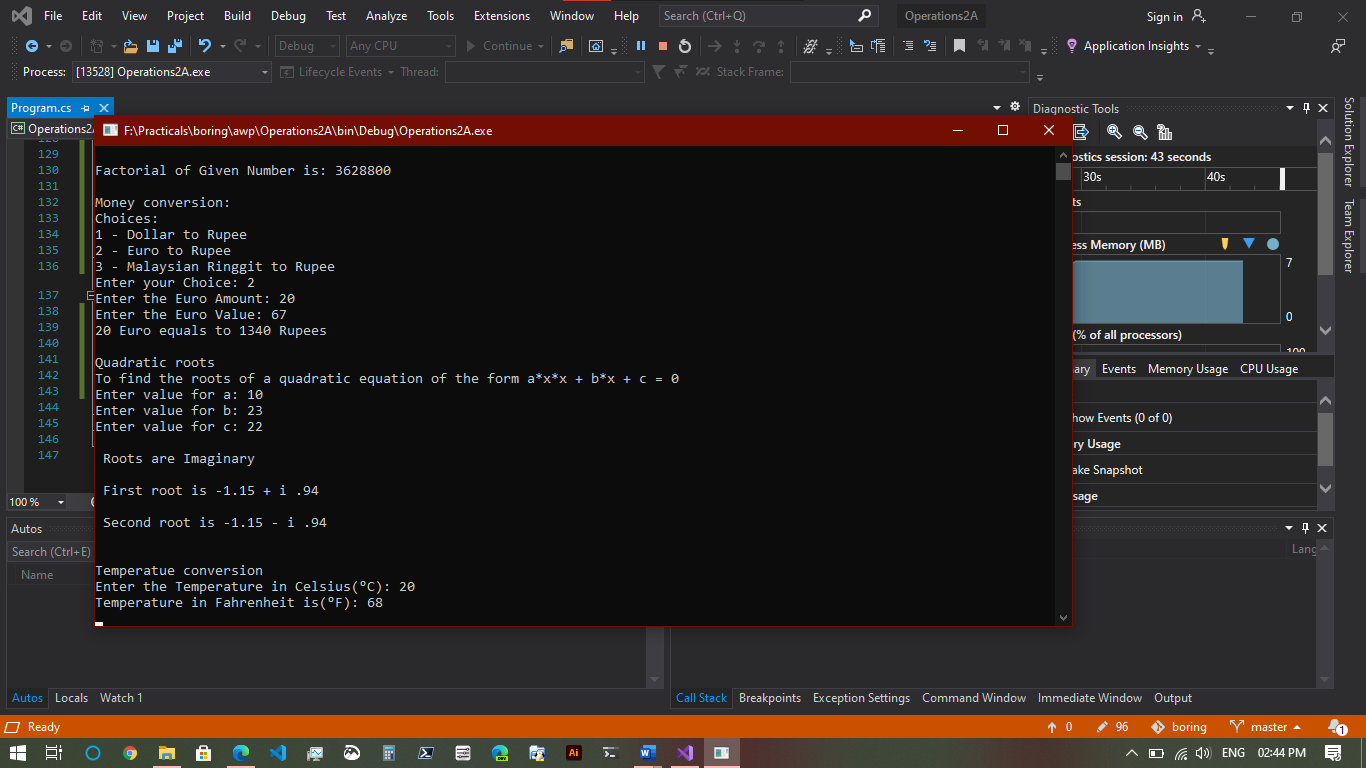
temperatureConversion();

Console.ReadKey();

}

}

}



B. AIM: Create simple application to demonstrate use of following concepts.

i. Function overloading, ii. Inheritance (All types), iii. Constructor overloading, iv. Interfaces

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Demonstrate1

{

class Overloading

{

public void swap(ref int n, ref int m)

{

int t;

t = n;

n = m;

m = t;

}

public void swap(ref float f1, ref float f2)

{

float f;

f = f1;

f1 = f2;

f2 = f;

}

}

class Furniture

{

string material;

float price;

public void getdata()

{

Console.Write("Enter material : ");

material = Console.ReadLine();

Console.Write("Enter price : ");

price = float.Parse(Console.ReadLine());

}

public void showdata()

{

Console.WriteLine("Material : " + material);

Console.WriteLine("Price : " + price);

}

}

class Table : Furniture

{

int height, surface\_area;

public void getdata()

{

base.getdata();

Console.Write("Enter height: ");

height = int.Parse(Console.ReadLine());

Console.Write("Enter surface area: ");

surface\_area = int.Parse(Console.ReadLine());

}

public void showdata()

{

base.showdata();

Console.WriteLine("Height : " + height);

Console.WriteLine("Surface Area : " + surface\_area);

}

}

class A

{

public void About()

{

Console.WriteLine("About method from class A");

}

}

class B : A

{

}

class C : B

{

}

interface D

{

void say();

}

class E : A, D

{

public void say()

{

Console.WriteLine("Inherited 'say' from an interface");

}

}

class Salary

{

int basic, ta, da, hra;

public Salary()

{

da = 9000;

hra = 6000;

}

public void getdata()

{

Console.Write("Enter basic salary : ");

basic = int.Parse(Console.ReadLine());

Console.Write("Enter travelling allowance : ");

ta = int.Parse(Console.ReadLine());

}

public void showdata()

{

Console.WriteLine("Basic salary : " + basic);

Console.WriteLine("Dearness allowence : " + da);

Console.WriteLine("Housing rent allowence : " + hra);

Console.WriteLine("Travelling allowence : " + ta);

Console.WriteLine("Gross Salary : " + (basic + da + hra + ta));

}

}

class Program

{

static void overloading()

{

Console.WriteLine("\nOverloading");

Overloading objOverloading = new Overloading();

int n = 10, m = 20;

objOverloading.swap(ref n, ref m);

Console.WriteLine("N=" + n + "\tM=" + m);

float f1 = 10.5f, f2 = 20.6f;

objOverloading.swap(ref f1, ref f2);

Console.WriteLine("F1=" + f1 + "\tF2=" + f2);

}

static void singleInheritance()

{

B obj = new B();

obj.About();

}

static void multipleInheritance()

{

E obj = new E();

obj.About();

obj.say();

}

static void multilevelInheritance()

{

C obj = new C();

obj.About();

}

static void hierarchicalInheritance()

{

B obj1 = new B();

E obj2 = new E();

obj1.About();

obj2.About();

}

static void inheritance()

{

Console.WriteLine("\nInheritance");

singleInheritance();

multipleInheritance();

multilevelInheritance();

hierarchicalInheritance();

}

static void constructorOverloading()

{

Console.WriteLine("\nConstructor overloading");

Salary s = new Salary();

s.getdata();

s.showdata();

}

static void interfaceDemo()

{

// uses interface

hierarchicalInheritance();

}

static void Main(string[] args)

{

overloading();

inheritance();

constructorOverloading();

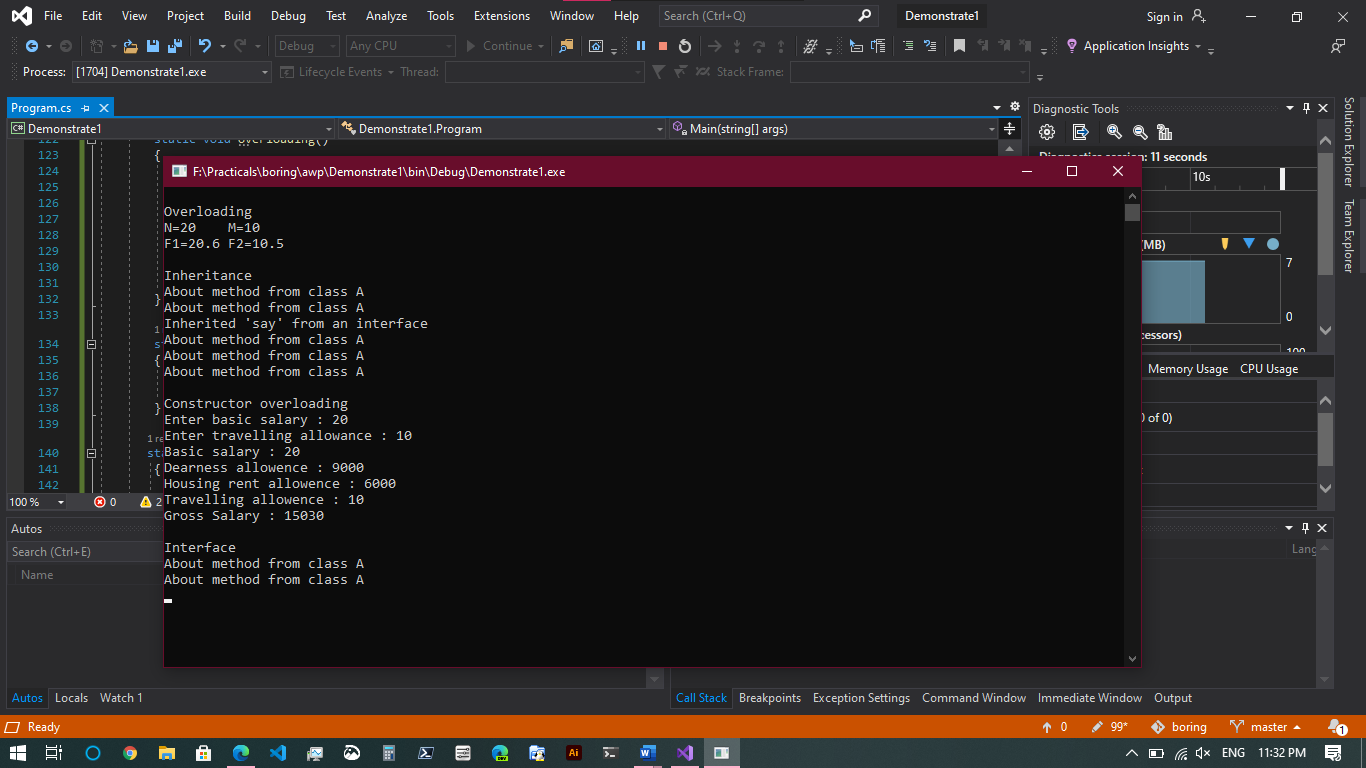
interfaceDemo();

Console.ReadKey();

}

}

}



C. AIM: Create simple application to demonstrate use of following concepts.

i. Delegates & Events, ii. Exception handling

using System;

namespace DelegateAndEventWithExceptionHandling

{

public delegate void TrafficDel();

class TrafficSignal

{

public static void Yellow()

{

Console.WriteLine("Yellow light signals to get ready");

}

public static void Green()

{

Console.WriteLine("Green light signals to go");

}

public static void Red()

{

Console.WriteLine("Red light signals to stop");

}

TrafficDel[] td = new TrafficDel[3];

public void IdentifySignal()

{

td[0] = new TrafficDel(Yellow);

td[1] = new TrafficDel(Green);

td[2] = new TrafficDel(Red);

}

public void display()

{

td[0]();

td[1]();

td[2]();

}

}

class NotEvenException : Exception

{

public NotEvenException(string msg)

: base(msg)

{

}

}

class Program

{

static void delegateExample()

{

Console.WriteLine("\nDelegate & Events example");

TrafficSignal ts = new TrafficSignal();

ts.IdentifySignal();

ts.display();

}

static void handlingExceptions()

{

Console.WriteLine("\nException handling");

int num;

try

{

Console.Write("Enter a number: ");

num = int.Parse(Console.ReadLine());

if ((num % 2) != 0) throw new NotEvenException("Not an even number ");

else

Console.WriteLine("Its even number ");

}

catch (NotEvenException e) { Console.WriteLine(e.Message); }

}

static void Main(string[] args)

{

delegateExample();

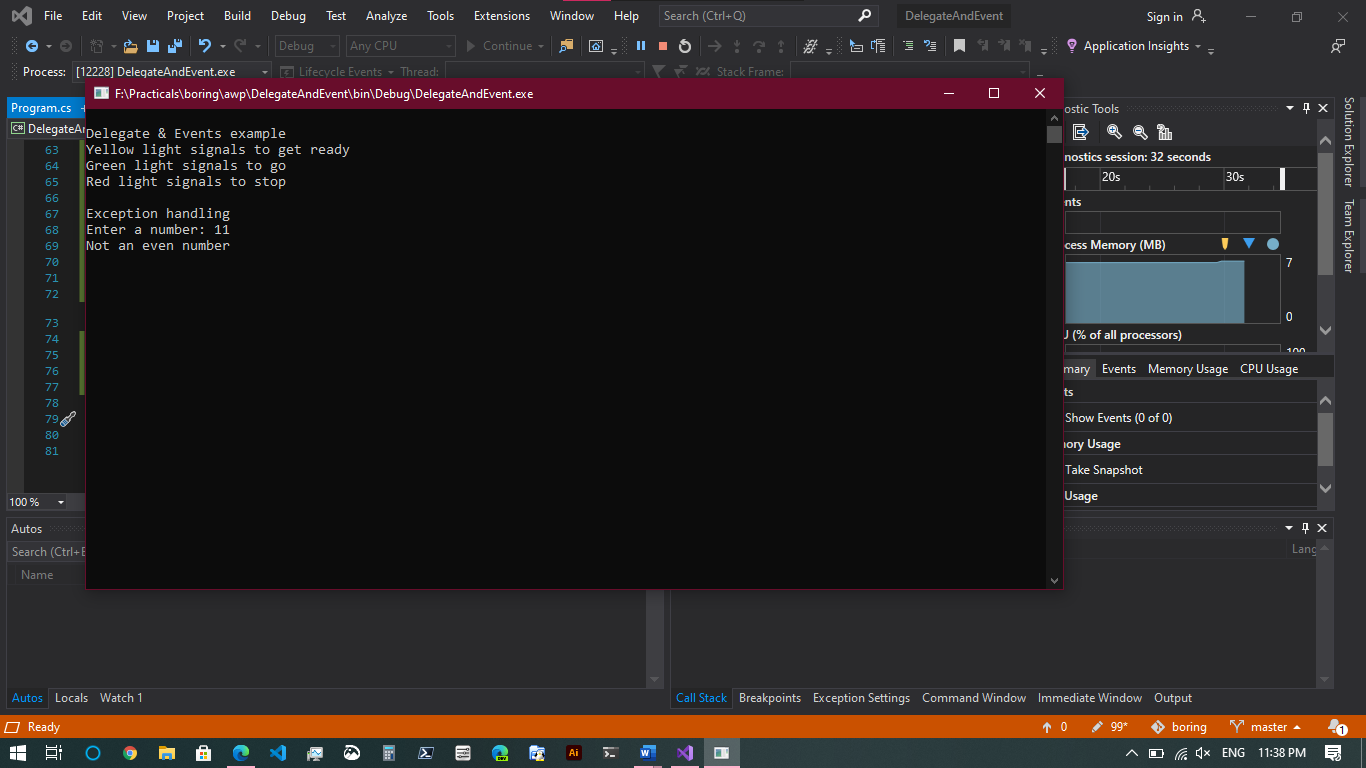
handlingExceptions();

Console.ReadKey();

}

}

}



Practical 3: Working with Web Forms & Controls

A. AIM: Create a simple web page with various server web controls to demonstrate settings and use of their properties (Example: AutoPostBack)

FILE> Default.aspx

<%@ Page Title="Home Page" Language="C#" MasterPageFile="~/Site.Master" AutoEventWireup="true" CodeBehind="Default.aspx.cs" Inherits="ServerControlWebApp.\_Default" %>

<asp:Content ID="BodyContent" ContentPlaceHolderID="MainContent" runat="server">

<h1>Select Fruit: </h1>

<div>

<asp:DropDownList ID="Fruits" runat="server"

OnSelectedIndexChanged="DropDownList1\_SelectedIndexChanged" Width="118px"

AutoPostBack="True">

<asp:ListItem Value="1">Mango</asp:ListItem>

<asp:ListItem Value="2">Apple</asp:ListItem>

<asp:ListItem Value="3">Banana</asp:ListItem>

<asp:ListItem Value="4">Kiwi</asp:ListItem>

</asp:DropDownList>

</div>

</asp:Content>

FILE> Default.aspx.cs

using System;

using System.Web.UI;

namespace ServerControlWebApp

{

public partial class \_Default : Page

{

protected void Page\_Load(object sender, EventArgs e)

{

}

protected void DropDownList1\_SelectedIndexChanged(object sender, EventArgs e)

{

string message = "You have selected : " + Fruits.SelectedItem.Text;

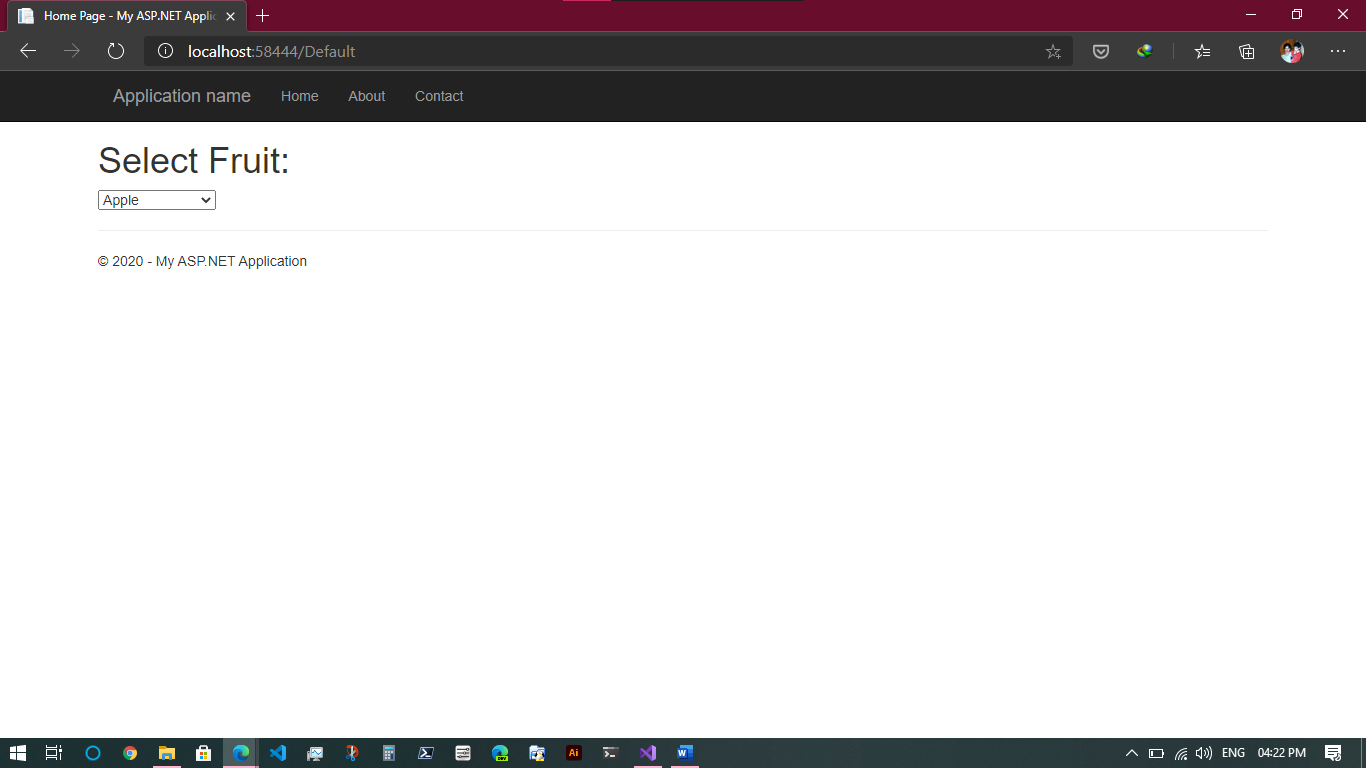
ClientScript.RegisterStartupScript(this.GetType(), "alert", "alert('" + message + "');",

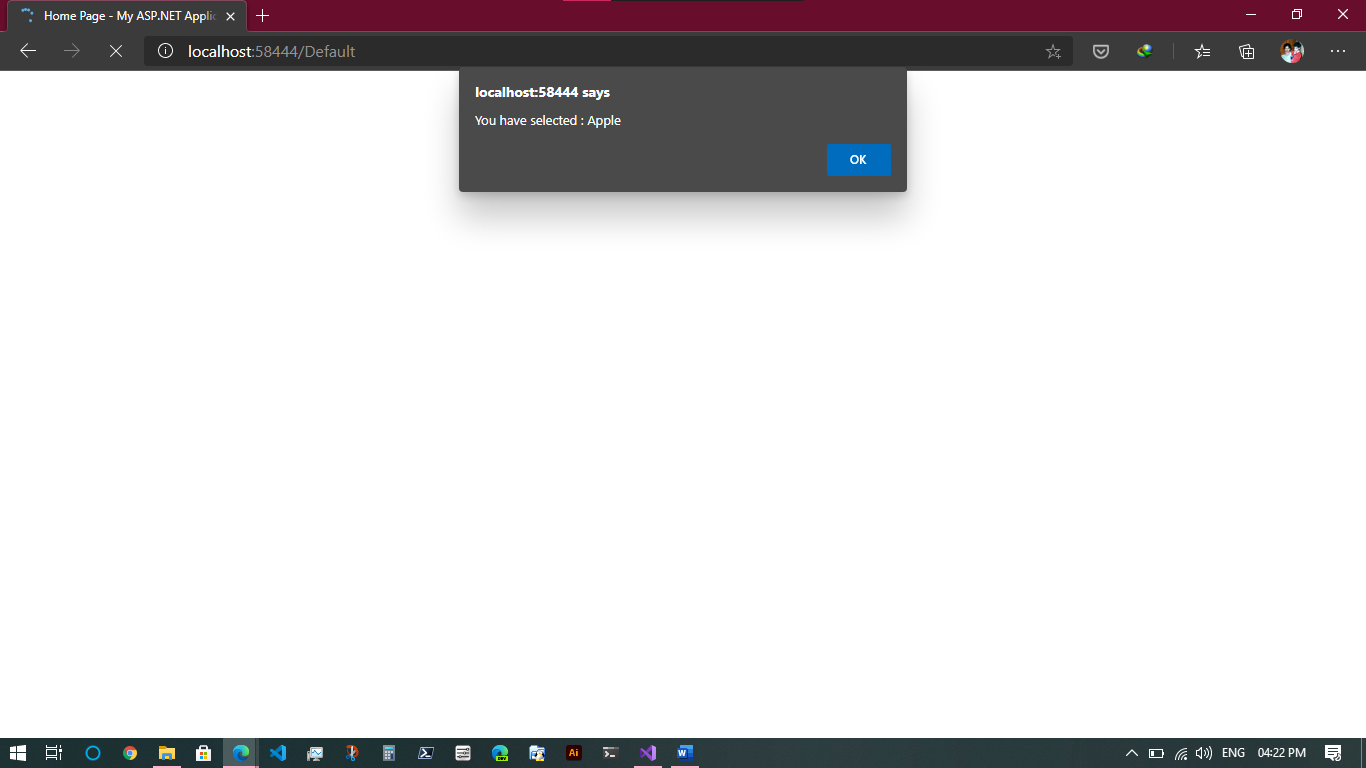
true);

}

}

}





B. AIM: Demonstrate the use of Calendar control to perform following operations.

i. Display messages in a calendar control, ii. Display vacation in a calendar control, iii. Selected day in a calendar control using style, iv. Difference between two calendar dates

FILE> Default.aspx

<%@ Page Title="Home Page" Language="C#" MasterPageFile="~/Site.Master" AutoEventWireup="true" CodeBehind="Default.aspx.cs" Inherits="CalendarControl.\_Default" %>

<asp:Content ID="BodyContent" ContentPlaceHolderID="MainContent" runat="server">

<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" OnSelectionChanged="Calendar1\_SelectionChanged" 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>

<br />

<asp:Button ID="Button1" runat="server" OnClick="Button1\_Click" Text="Submit" />

&nbsp;

<asp:Button ID="Button2" runat="server" OnClick="Button2\_Click" Text="Reset" Width="54px" />

&nbsp;

<br />

<br />

<asp:Label ID="Label0" runat="server" Text=""></asp:Label>

<br />

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

<br />

<asp:Label ID="Label2" runat="server" Text=""></asp:Label>

<br />

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

<br />

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

<br />

<asp:Label ID="Label5" runat="server" Text=""></asp:Label>

<br />

<asp:Label ID="Label6" runat="server" Text=""></asp:Label>

</asp:Content>

FILE> Default.aspx.cs

using System;

using System.Web.UI;

using System.Web.UI.WebControls;

namespace CalendarControl

{

public partial class \_Default : Page

{

protected void Calendar1\_DayRender(object sender,

DayRenderEventArgs e)

{

if (e.Day.Date.Day == 24 && e.Day.Date.Month == 12)

{

e.Cell.BackColor = System.Drawing.Color.Yellow;

Label lbl = new Label();

lbl.Text = "<br>Winter vacation Start: 24 - 12 - 2020";

e.Cell.Controls.Add(lbl);

Image g1 = new Image();

g1.ImageUrl = "https://cdn.travelpulse.com/images/99999999-9999-9999-9999-999999999999/5712ea02-89d7-e611-9aa9-0050568e420d/630x355.jpg";

g1.Height = 20;

g1.Width = 20;

e.Cell.Controls.Add(g1);

}

if (e.Day.Date.Day == 1 && e.Day.Date.Month == 1)

{

Calendar1.SelectedDate = new DateTime(2020, 1, 1);

Calendar1.SelectedDates.SelectRange(Calendar1.SelectedDate,

Calendar1.SelectedDate.AddDays(10));

Label lbl1 = new Label();

lbl1.Text = "<br>New Year";

e.Cell.Controls.Add(lbl1);

}

}

protected void Calendar1\_SelectionChanged(object sender, EventArgs e)

{

Label0.Text = "Your Selected Date:" + Calendar1.SelectedDate.Date.ToString();

}

protected void Page\_Load(object sender, EventArgs e)

{

}

protected void Button1\_Click(object sender, EventArgs e)

{

Calendar1.Caption = "SAMBARE";

Calendar1.FirstDayOfWeek = FirstDayOfWeek.Sunday;

Calendar1.NextPrevFormat = NextPrevFormat.ShortMonth;

Calendar1.TitleFormat = TitleFormat.Month;

Label1.Text = "Todays Date" + Calendar1.TodaysDate.ToShortDateString();

Label2.Text = "Winter vacation Start: 24-12-2020";

TimeSpan d = new DateTime(2020, 12, 24) - DateTime.Now;

Label3.Text = "Days Remaining for Teacher's day Vacation:" + d.Days.ToString();

TimeSpan d1 = new DateTime(2021, 1, 1) - DateTime.Now;

Label4.Text = "Days Remaining for New Year:" + d1.Days.ToString();

if (Calendar1.SelectedDate.ToShortDateString() == "24-12-2020")

Label5.Text = "<b>Winter vacation's Start</b>";

if (Calendar1.SelectedDate.ToShortDateString() == "1-1-2020")

Label6.Text = "<b>It's New Year</b>";

}

protected void Button2\_Click(object sender, EventArgs e)

{

Label0.Text = "";

Label1.Text = "";

Label2.Text = "";

Label3.Text = "";

Label4.Text = "";

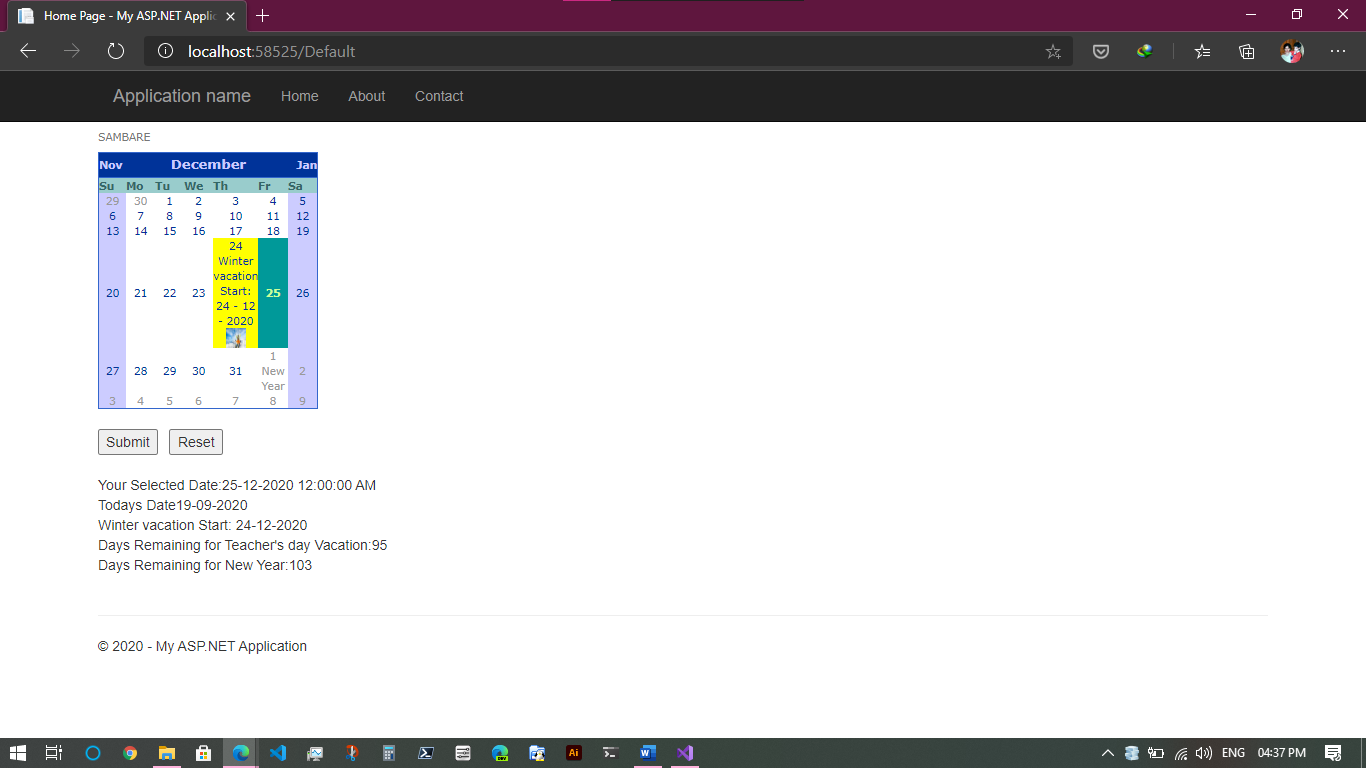
Label5.Text = "";

Label6.Text = "";

Calendar1.SelectedDates.Clear();

}

}

} 

C. AIM: Demonstrate the use of TreeView control. Perform the following operations.

i. TreeView Control & datalist, ii. Exception handling

FILE> Default.aspx

<%@ Page Title="Home Page" Language="C#" MasterPageFile="~/Site.Master" AutoEventWireup="true" CodeBehind="Default.aspx.cs" Inherits="TreeViewAndDataList.\_Default" %>

<asp:Content ID="BodyContent" ContentPlaceHolderID="MainContent" runat="server">

<asp:TreeView ID="TreeView1" runat="server" Width="171px" ImageSet="Arrows"

Height="127px">

<HoverNodeStyle Font-Underline="True" ForeColor="#5555DD" />

<Nodes>

<asp:TreeNode Text="Other Pages" Value="New Node">

<asp:TreeNode Text="Contact" Value="RED" NavigateUrl="Contact.aspx"></asp:TreeNode>

<asp:TreeNode Text="About" Value="GREEN"

NavigateUrl="About.aspx"></asp:TreeNode>

</asp:TreeNode>

</Nodes>

<NodeStyle Font-Names="Tahoma" Font-Size="10pt" ForeColor="Black"

HorizontalPadding="5px" NodeSpacing="0px" VerticalPadding="0px" />

<ParentNodeStyle Font-Bold="False" />

<SelectedNodeStyle Font-Underline="True" ForeColor="#5555DD"

HorizontalPadding="0px" VerticalPadding="0px" />

</asp:TreeView>

<br />

<asp:DataList ID="DataList1" runat="server">

<ItemTemplate>

<table class="table" border="1">

<tr>

<td>Roll Num : <%# Eval("sid") %><br />

Name : <%# Eval("sname") %><br />

Class : <%# Eval("sclass")%>

</td>

</tr>

</table>

</ItemTemplate>

</asp:DataList>

</asp:Content>

FILE> Default.aspx.cs

using System;

using System.Data;

using System.Web.UI;

namespace TreeViewAndDataList

{

public partial class \_Default : Page

{

protected void Page\_Load(object sender, EventArgs e)

{

if (!IsPostBack)

{

BindData();

}

}

protected void BindData()

{

DataSet ds = new DataSet();

ds.ReadXml(Server.MapPath("studentdetails.xml"));

if (ds != null && ds.HasChanges())

{

DataList1.DataSource = ds;

DataList1.DataBind();

}

else

{

DataList1.DataBind();

}

}

}

}

FILE> studentdetails.xml

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

<studentdetail>

<student>

<sid>1</sid>

<sname>Mushaheed</sname>

<sclass>TYIT</sclass>

</student>

<student>

<sid>2</sid>

<sname>Sonal</sname>

<sclass>TYIT</sclass>

</student>

<student>

<sid>3</sid>

<sname>Vishaal</sname>

<sclass>TYIT</sclass>

</student>

<student>

<sid>4</sid>

<sname>Amaan</sname>

<sclass>TYIT</sclass>

</student>

<student>

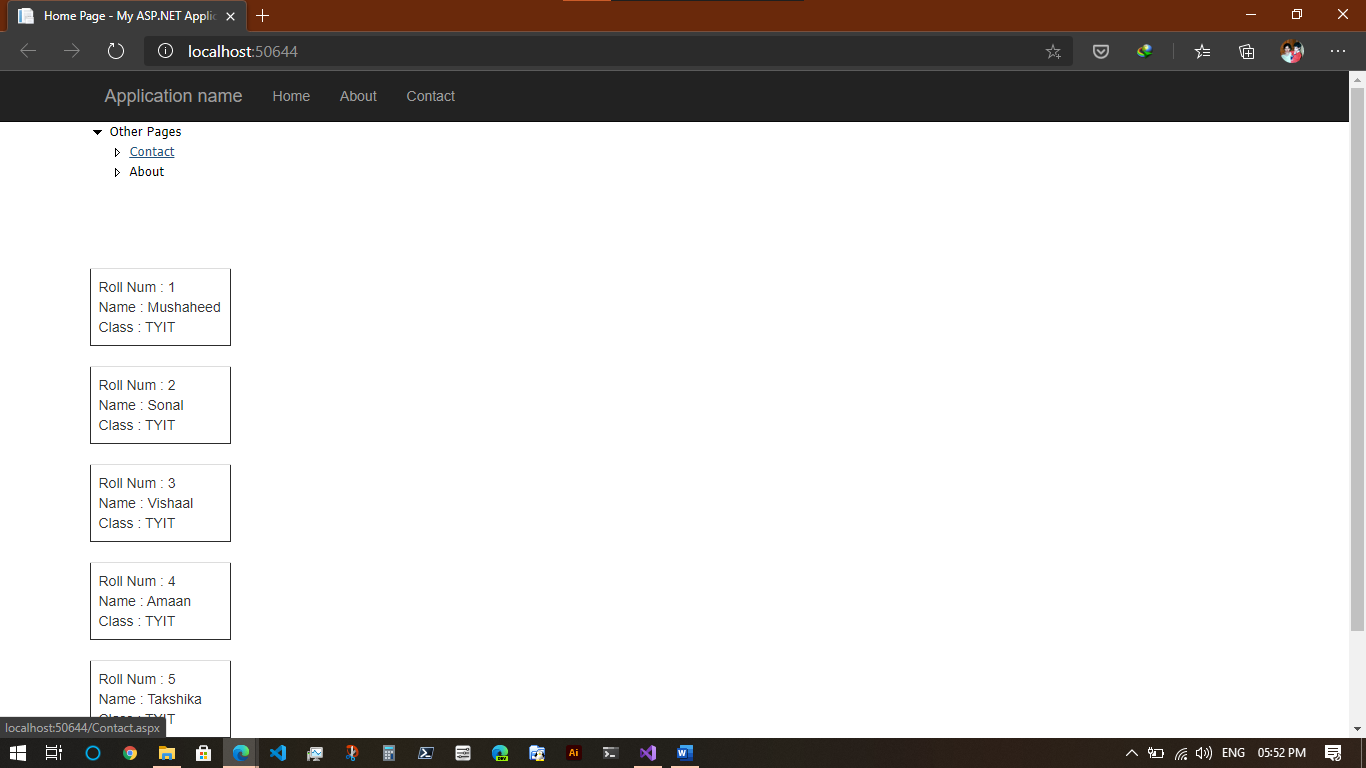
<sid>5</sid>

<sname>Takshika</sname>

<sclass>TYIT</sclass>

</student>

</studentdetail>



Practical 4: Working with Form controls

A. AIM: Create a registration form to demonstrate use of various Validation controls.

FILE> Default.aspx

<%@ Page Title="Home Page" Language="C#" MasterPageFile="~/Site.Master" AutoEventWireup="true" CodeBehind="Default.aspx.cs" Inherits="RegistrationValidation.\_Default" %>

<asp:Content ID="BodyContent" ContentPlaceHolderID="MainContent" runat="server">

<div style:"font-family:Verdana">

Your Name :

<asp:TextBox ID="txtName" runat="server"></asp:TextBox>

<asp:RequiredFieldValidator runat="server" ID="reqName" ForeColor="Red"

ControlToValidate="txtName" ErrorMessage="Please Enter Your Name!" />

<br />

Enter Your Age :

<asp:TextBox ID="txt1" runat="server"

Width="56px"></asp:TextBox>

<asp:RangeValidator ID="RangeValidator2" Type="Integer" runat="server"

ForeColor="Red" ControlToValidate="txt1" MinimumValue="18" MaximumValue="100"

ErrorMessage="Please Enter a Valid Age"></asp:RangeValidator>

<br />

Password :

<asp:TextBox ID="txt11" runat="server"></asp:TextBox>

<br />

Re-enter password :

<asp:TextBox ID="txt12" runat="server"></asp:TextBox>

<asp:CompareValidator ID="cmpNumbers" runat="server" ControlToCompare="txt11"

ControlToValidate="txt12" ErrorMessage="Password Do Not Match!"

ForeColor="Red"

Operator="Equal" Type="Integer">Password should match!</asp:CompareValidator>

<br />

<br />

Email :<asp:TextBox ID="txtNumber" runat="server" Width="207px"></asp:TextBox>

<asp:RegularExpressionValidator ID="rexNumber" runat="server"

ControlToValidate="txtNumber" ErrorMessage="Please enter a valid email address"

ForeColor="Red"

ValidationExpression="\w+([-+.']\w+)\*@\w+([-.]\w+)\*\.\w+([-.]\w+)\*">Please enter

valid email address!</asp:RegularExpressionValidator>

<br />

Custom text:<asp:TextBox ID="txtCustom" runat="server"></asp:TextBox>

<asp:CustomValidator ID="CustomValidator1" runat="server"

ClientValidationFunction="ServerValidation" ControlToValidate="txtCustom"

ErrorMessage="CustomValidator" ForeColor="Red"></asp:CustomValidator>

<br />

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

<br />

<asp:Button ID="Button1" runat="server" OnClick="Button1\_Click"

Text="Validate Form" />

<br />

<br />

<asp:ValidationSummary ID="ValidationSummary1" runat="server" Height="61px" />

</div>

</asp:Content>

FILE> Default.aspx.cs

using System;

using System.Web.UI;

using System.Web.UI.WebControls;

namespace RegistrationValidation

{

public partial class \_Default : Page

{

protected void Page\_Load(object sender, EventArgs e)

{

}

protected void Button1\_Click(object sender, EventArgs e)

{

if (Page.IsValid)

{

Label1.Text = "Thank you";

}

else

{

Label1.Text = "The text must be exactly 8 characters Long!";

}

}

void ServerValidation(object source, ServerValidateEventArgs e)

{

if (e.Value.Length == 8)

e.IsValid = true;

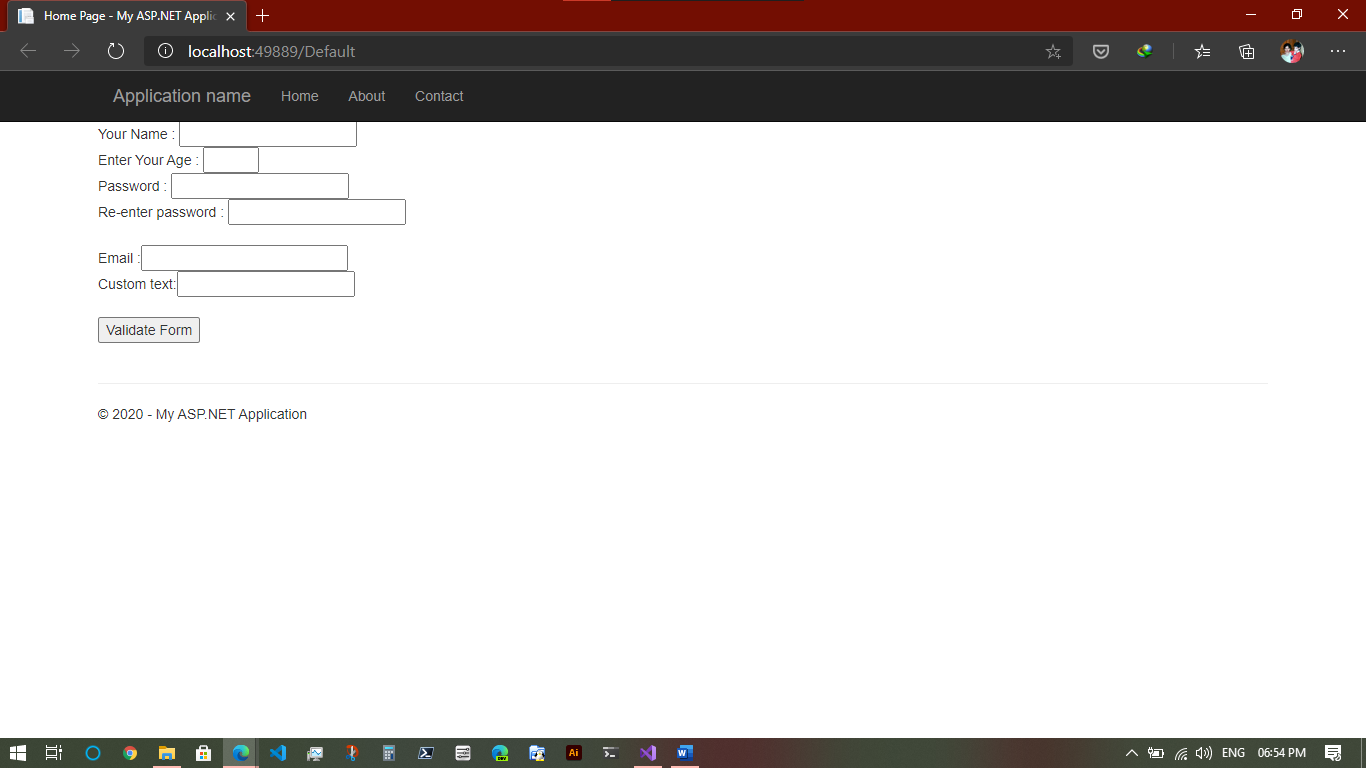
else

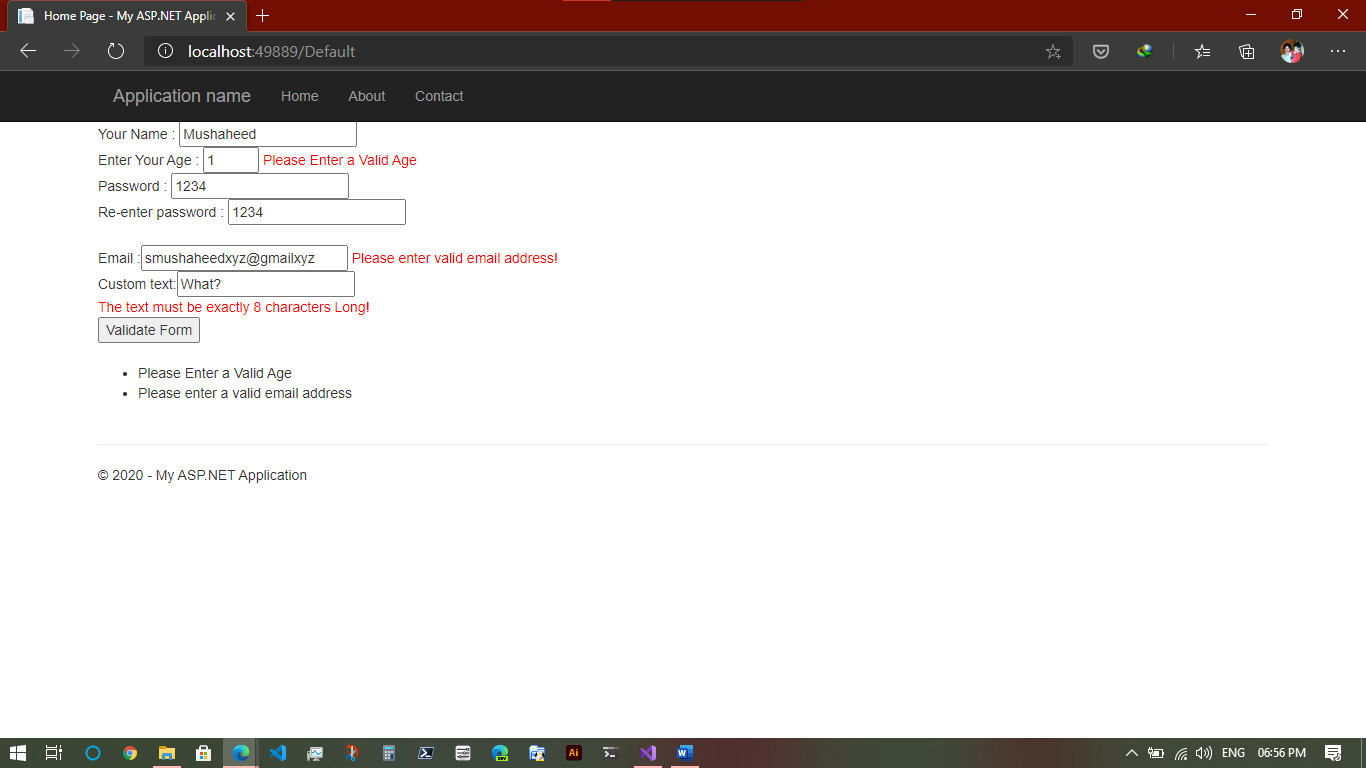
e.IsValid = false;

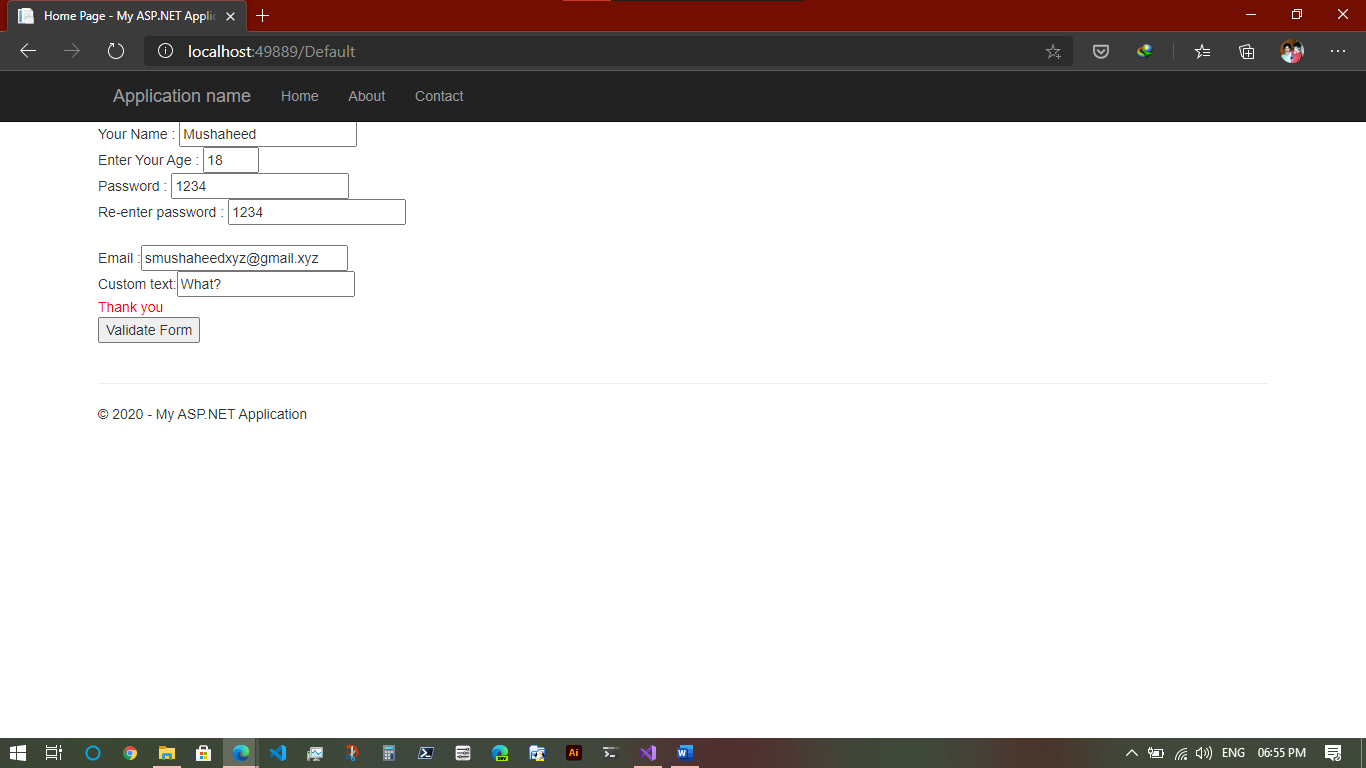
}

}

}







B. AIM: Create a web form to demonstrate use of Adrotator Control

FILE> Adfile.xml

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

<Advertisements>

<Ad>

<ImageUrl>https://i.pinimg.com/originals/ec/cf/1f/eccf1f835ea346d58d9de3a5c786c250.jpg</ImageUrl>

<NavigateUrl>http://www.1800flowers.com</NavigateUrl>

<AlternateText>

Order flowers, roses, gifts and more

</AlternateText>

<Impressions>20</Impressions>

<Keyword>flowers</Keyword>

</Ad>

<Ad>

<ImageUrl>https://64.media.tumblr.com/d13f2c15704f8dc283313c77777abdcb/597f4ceeb087f74d-2f/s500x750/1711dfcf19a689a494902e8b4498d9110719c083.jpg</ImageUrl>

<NavigateUrl>http://www.babybouquets.com.au</NavigateUrl>

<AlternateText>Order roses and flowers</AlternateText>

<Impressions>20</Impressions>

<Keyword>gifts</Keyword>

</Ad>

<Ad>

<ImageUrl>https://i.pinimg.com/originals/5b/21/bc/5b21bc6210e1fddcb1042dec93f30a4d.jpg</ImageUrl>

<NavigateUrl>http://www.flowers2moscow.com</NavigateUrl>

<AlternateText>Send flowers to Russia</AlternateText>

<Impressions>20</Impressions>

<Keyword>russia</Keyword>

</Ad>

</Advertisements>

FILE> Default.aspx

<%@ Page Title="Home Page" Language="C#" MasterPageFile="~/Site.Master" AutoEventWireup="true" CodeBehind="Default.aspx.cs" Inherits="AdRotatorControlDemo.\_Default" %>

<asp:Content ID="BodyContent" ContentPlaceHolderID="MainContent" runat="server">

<div>

<asp:AdRotator ID="AdRotator1" OnAdCreated="AdRotator1\_AdCreated" AdvertisementFile="~/Adfile.xml" runat="server" Target="\_blank" Height="200px" Width="200px" />

<br />

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

</div>

</asp:Content>

FILE> Default.aspx.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

namespace AdRotatorControlDemo

{

public partial class \_Default : Page

{

protected void Page\_Load(object sender, EventArgs e)

{

}

protected void AdRotator1\_AdCreated(object sender, AdCreatedEventArgs e)

{

Label1.Text = e.AdProperties["ImageUrl"].ToString();

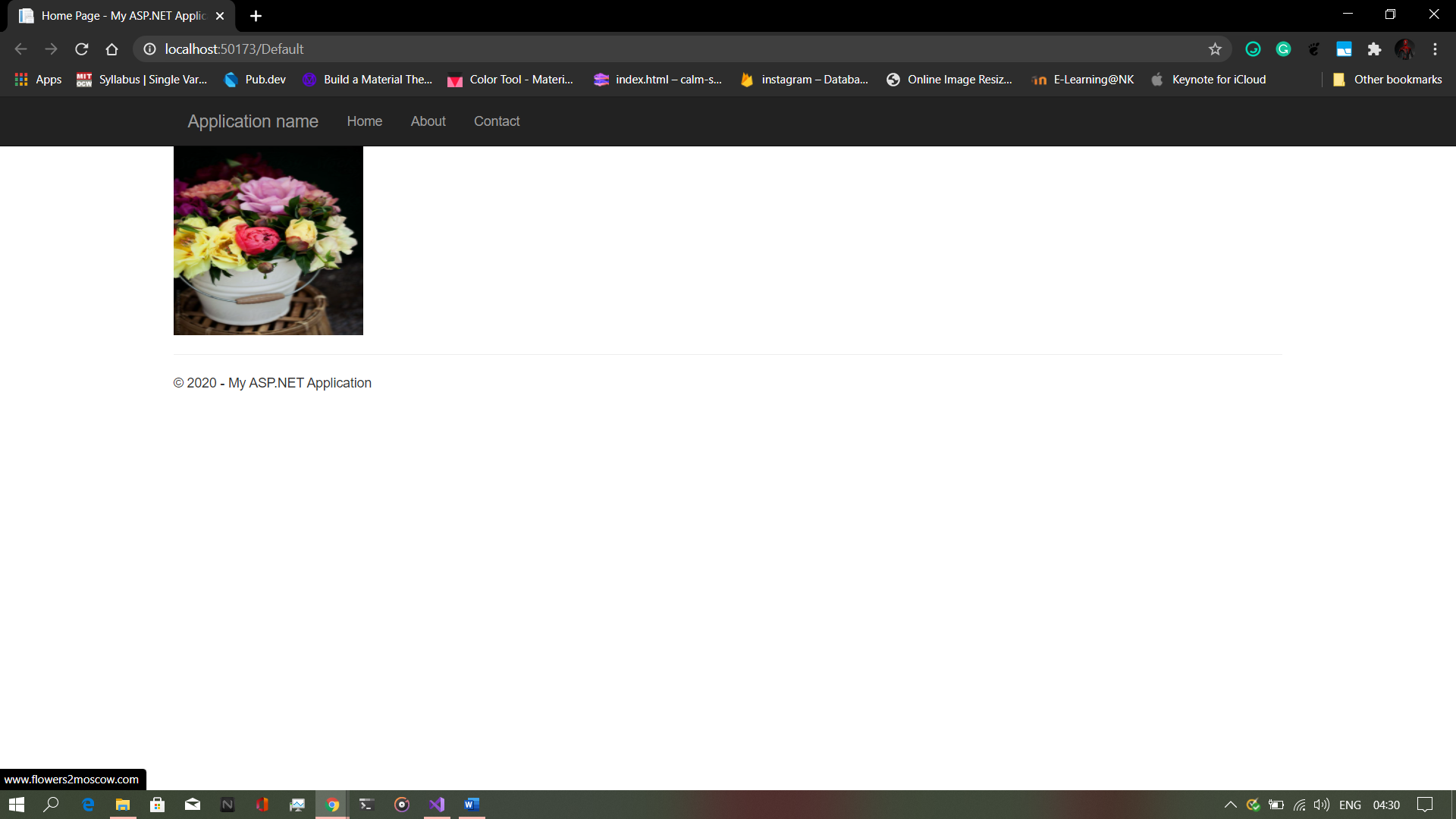
Label1.Visible = false;

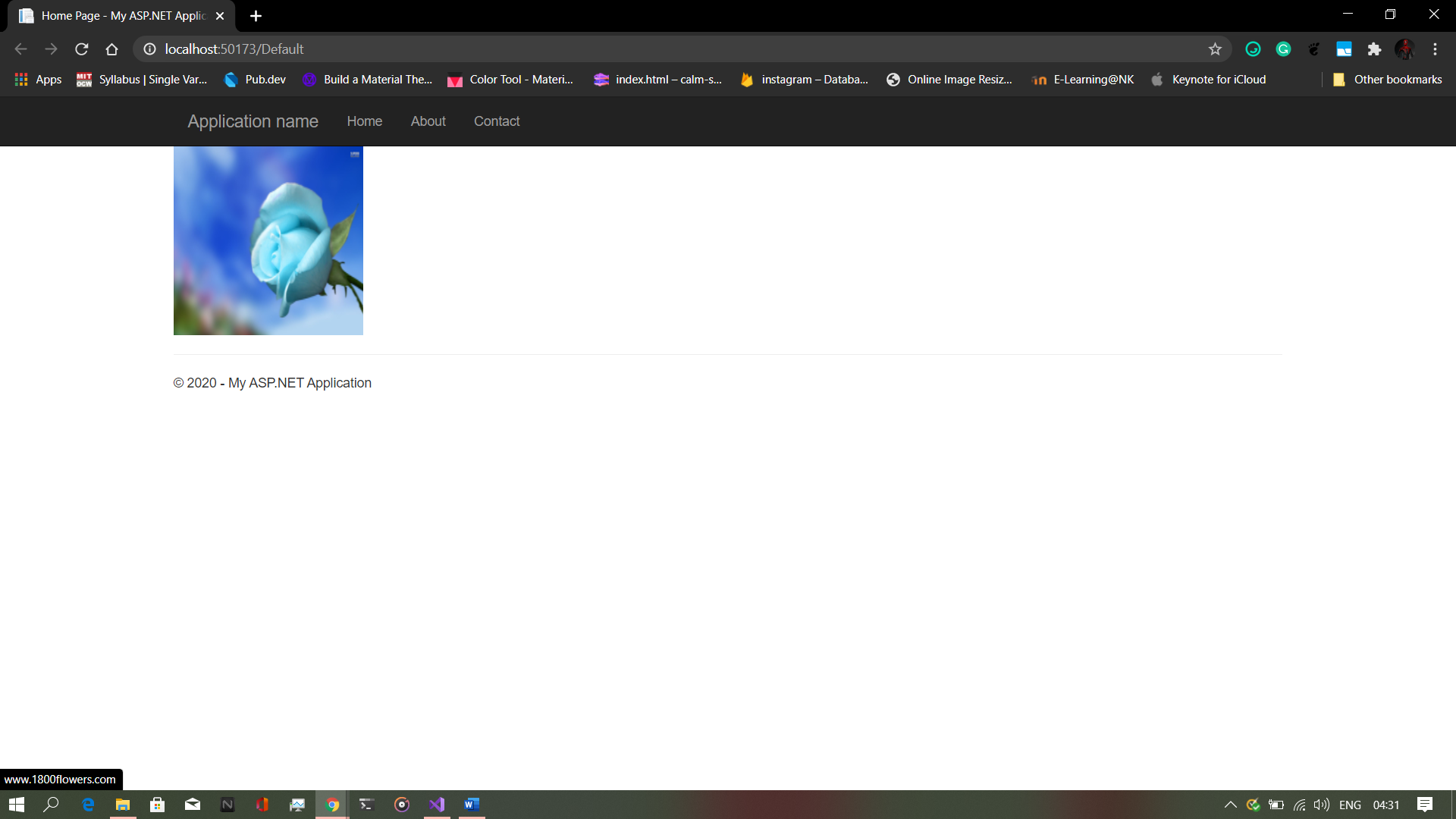
AdRotator1.Visible = true;

}

}

}





C. AIM: Create a web form to demonstrate use of User Controls

FILE> WebUserControl1.ascx

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

<h3>User Information</h3>

<table>

<tr>

<td>Name</td>

<td>

<asp:TextBox ID="txtName" runat="server"

OnTextChanged="txtName\_TextChanged"></asp:TextBox>

</td>

</tr>

<tr>

<td>Age</td>

<td>

<asp:TextBox ID="TextBox1" runat="server"

OnTextChanged="TextBox1\_TextChanged"></asp:TextBox>

</td>

</tr>

<tr>

<td>City</td>

<td>

<asp:TextBox ID="txtcity" runat="server"

OnTextChanged="txtcity\_TextChanged"></asp:TextBox></td>

</tr>

</table>

<br />

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

<asp:Label ID="Label2" runat="server" Text=" "></asp:Label><br />

<asp:Label ID="Label3" runat="server" Text=" "></asp:Label><br />

<asp:Button ID="txtSave" runat="server" Text="Save" OnClick="txtSave\_Click" />

FILE> WebUserControl1.ascx.cs

using System;

namespace UserCtrlDemo

{

public partial class WebUserControl1 : System.Web.UI.UserControl

{

protected void Page\_Load(object sender, EventArgs e)

{

}

protected void txtSave\_Click(object sender, EventArgs e)

{

{

Label1.Text = "Your Name is " + txtName.Text;

Label2.Text = "Your age is " + TextBox1.Text;

Label3.Text = "You are from " + txtcity.Text;

}

}

protected void txtName\_TextChanged(object sender, EventArgs e)

{

}

protected void TextBox1\_TextChanged(object sender, EventArgs e)

{

}

protected void txtcity\_TextChanged(object sender, EventArgs e)

{

}

}

}

FILE> Default.aspx

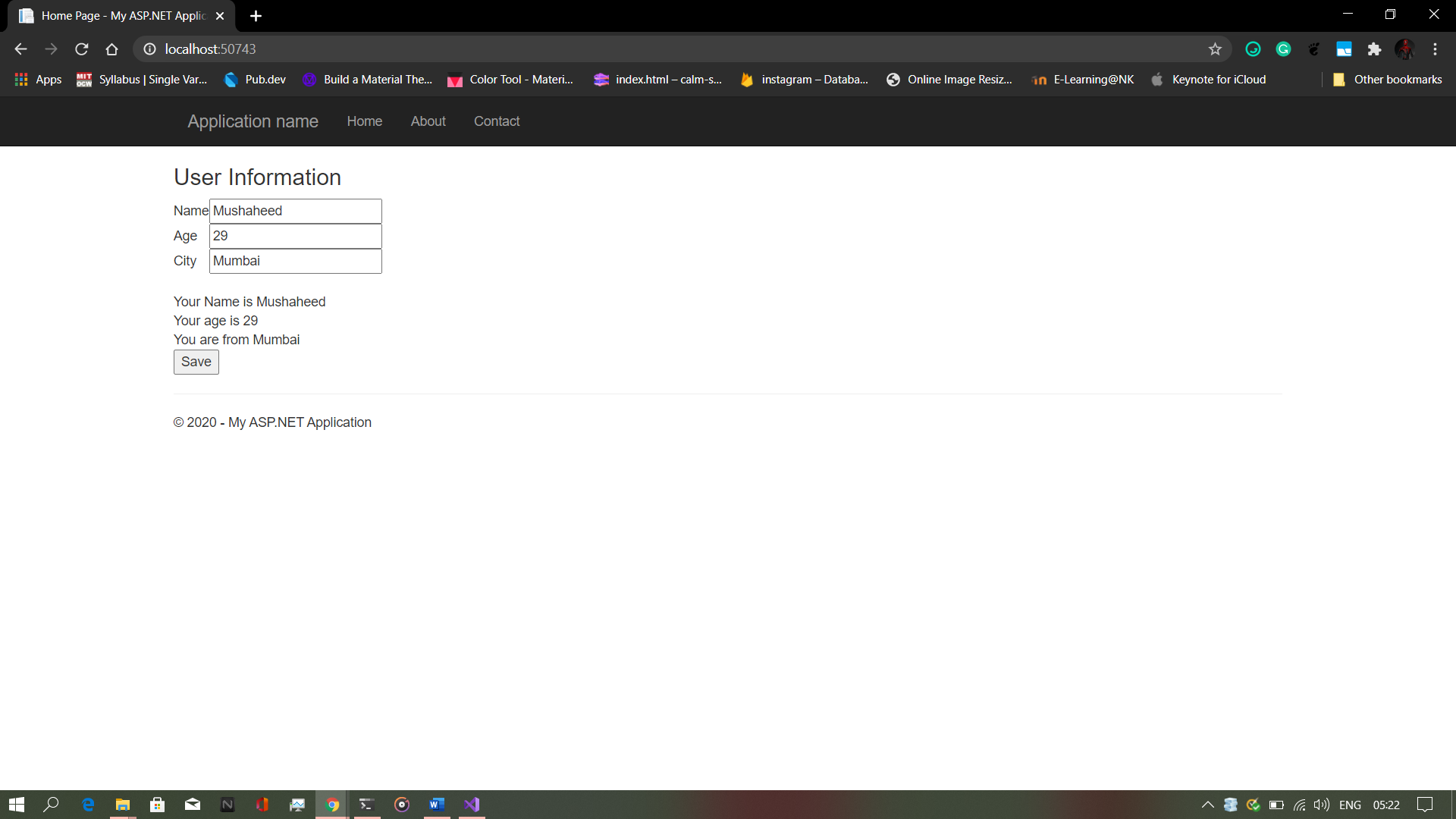
<%@ Page Title="Home Page" Language="C#" MasterPageFile="~/Site.Master" AutoEventWireup="true" CodeBehind="Default.aspx.cs" Inherits="UserCtrlDemo.\_Default" %>

<asp:Content ID="BodyContent" ContentPlaceHolderID="MainContent" runat="server">

<%@ Register Src="~/WebUserControl1.ascx" TagName="WebControl" TagPrefix="TWebControl" %>

<TWebControl:WebControl ID="Header" runat="server" />

</asp:Content>



Practical 5: Working with navigation, beautification & Master pages.

A. Create a Web Form to demonstrate use of Website Navigation controls & Site map.

FILE> Default.aspx

<%@ Page Title="Home Page" Language="C#" MasterPageFile="~/Site.Master" AutoEventWireup="true" CodeBehind="Default.aspx.cs" Inherits="WebNavControl.\_Default" %>

<asp:Content ID="BodyContent" ContentPlaceHolderID="MainContent" runat="server">

<h1>Home Page</h1>

<div>

<asp:SiteMapPath ID="SiteMapPath1" runat="server"></asp:SiteMapPath>

<asp:Menu ID="Menu1" runat="server" Orientation="Vertical">

<Items>

<asp:MenuItem NavigateUrl="~/About.aspx" Text="About"

Value="About"></asp:MenuItem>

<asp:MenuItem NavigateUrl="~/Contact.aspx" Text="Contact"

Value="Contact"></asp:MenuItem>

</Items>

</asp:Menu>

</div>

</asp:Content>

FILE> About.aspx

<%@ Page Title="About" Language="C#" MasterPageFile="~/Site.Master" AutoEventWireup="true" CodeBehind="About.aspx.cs" Inherits="WebNavControl.About" %>

<asp:Content ID="BodyContent" ContentPlaceHolderID="MainContent" runat="server">

<h1>About Page</h1>

<div>

<asp:SiteMapPath ID="SiteMapPath1" runat="server"></asp:SiteMapPath>

<asp:Menu ID="Menu1" runat="server" Orientation="Vertical">

<Items>

<asp:MenuItem NavigateUrl="~/Default.aspx" Text="Home"

Value="Home"></asp:MenuItem>

<asp:MenuItem NavigateUrl="~/Contact.aspx" Text="Contact"

Value="Contact"></asp:MenuItem>

</Items>

</asp:Menu>

</div>

</asp:Content>

FILE> Contact.aspx

<%@ Page Title="Contact" Language="C#" MasterPageFile="~/Site.Master" AutoEventWireup="true" CodeBehind="Contact.aspx.cs" Inherits="WebNavControl.Contact" %>

<asp:Content ID="BodyContent" ContentPlaceHolderID="MainContent" runat="server">

<h1>Contact Page</h1>

<div>

<asp:SiteMapPath ID="SiteMapPath1" runat="server"></asp:SiteMapPath>

<asp:Menu ID="Menu1" runat="server" Orientation="Vertical">

<Items>

<asp:MenuItem NavigateUrl="~/Default.aspx" Text="Home"

Value="Home"></asp:MenuItem>

<asp:MenuItem NavigateUrl="~/About.aspx" Text="About"

Value="About"></asp:MenuItem>

</Items>

</asp:Menu>

</div>

</asp:Content>

FILE> Web.sitemap

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

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

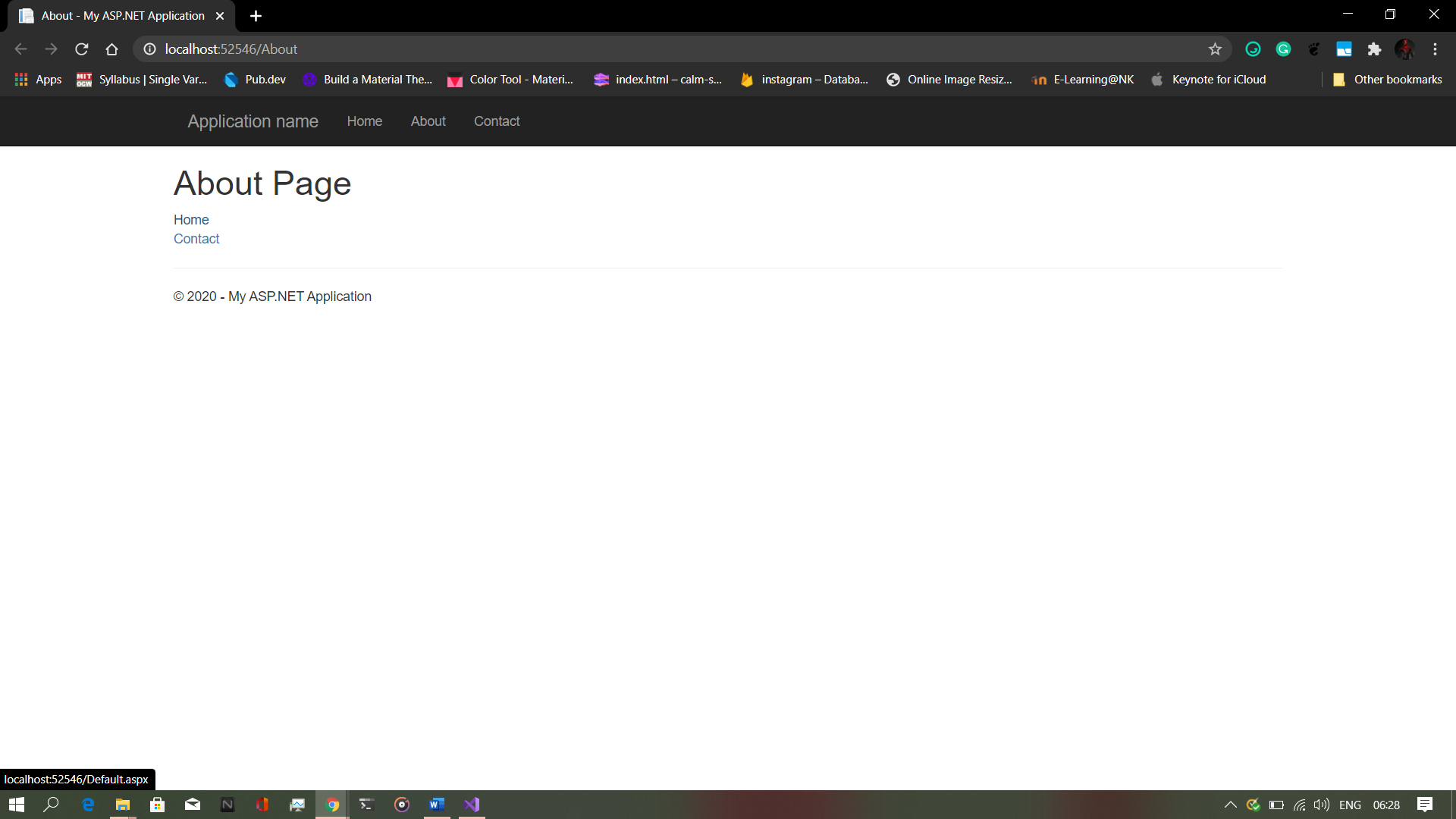
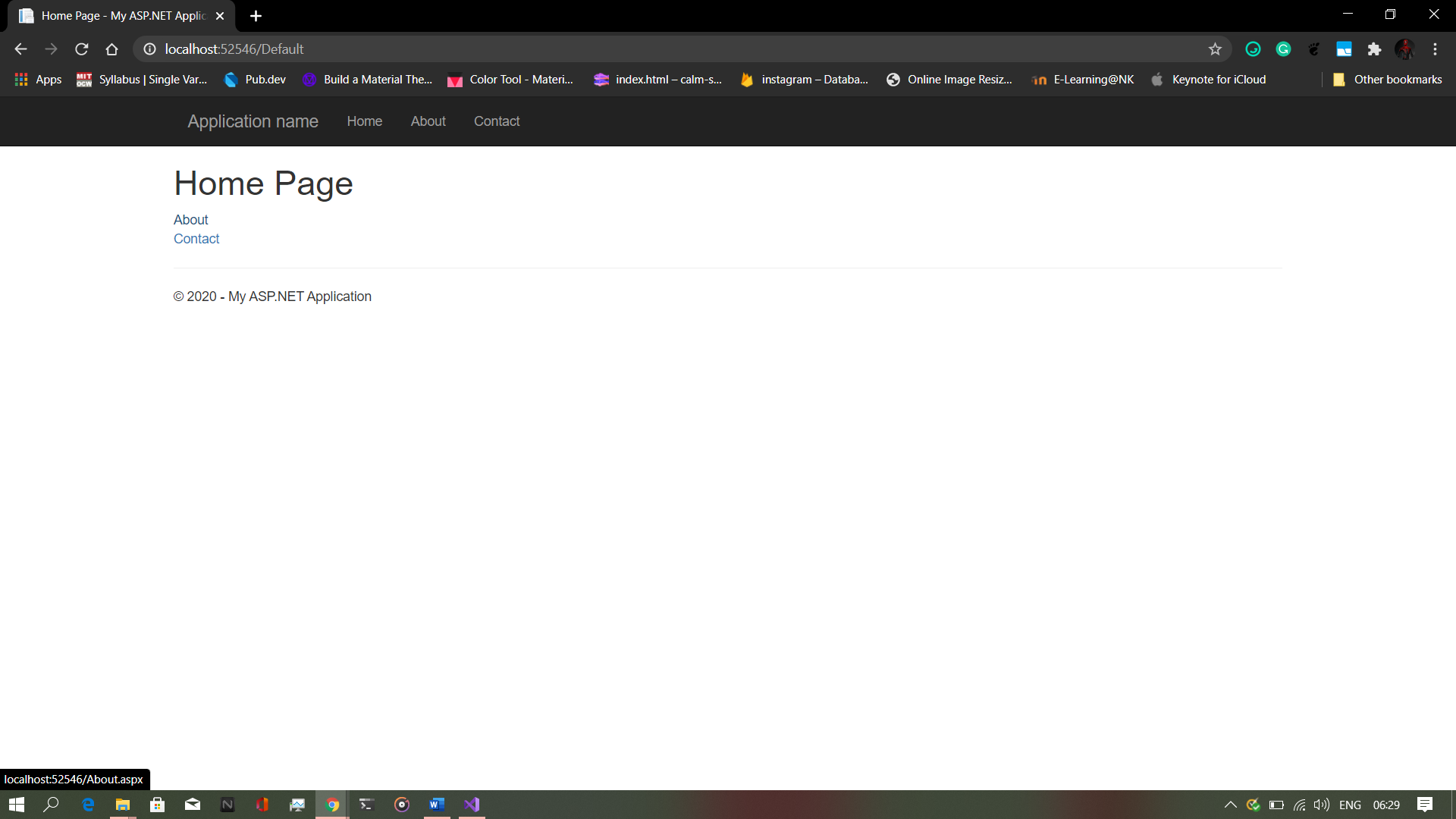
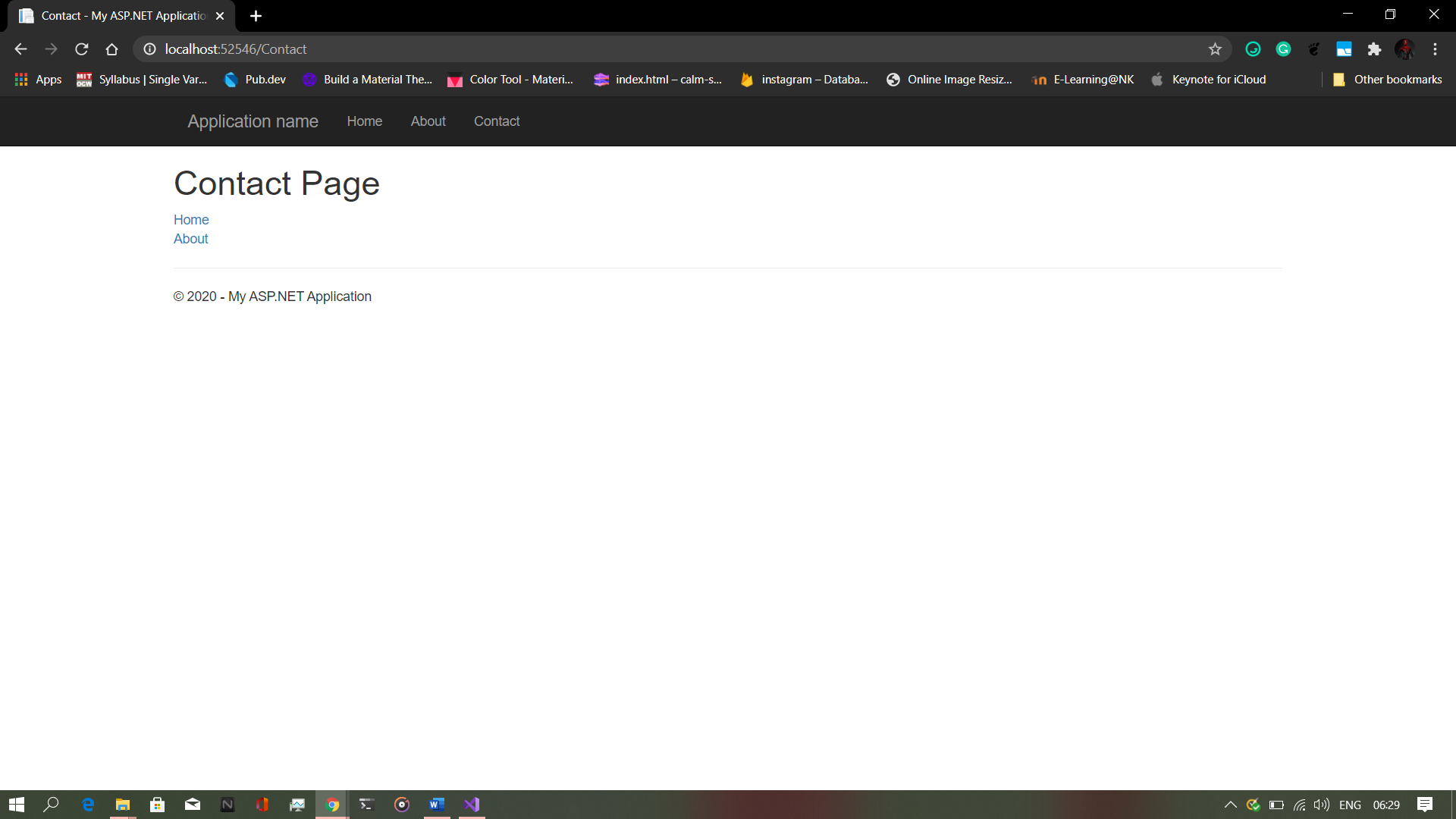
<siteMapNode url="Default.aspx" title="" description="Home">

<siteMapNode url="About.aspx" title="" description="About" />

<siteMapNode url="Contact.aspx" title="" description="Contact" />

</siteMapNode>

</siteMap>

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

FILE> Site.Master

<%@ Master Language="C#" AutoEventWireup="true" CodeBehind="Site.master.cs" Inherits="Beautified.SiteMaster" %>

<!DOCTYPE html>

<html lang="en">

<head runat="server">

<meta charset="utf-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title><%: Page.Title %> -Beauification Demo</title>

<asp:PlaceHolder runat="server">

<%: Scripts.Render("~/bundles/modernizr") %>

</asp:PlaceHolder>

<webopt:BundleReference runat="server" Path="~/Content/css" />

<link rel="stylesheet" type="text/css" href="StyleSheet.css">

<style type="text/css">

.auto-style1 {

position: absolute;

top: 373px;

left: 1028px;

bottom: 303px;

}

.auto-style2 {

position: absolute;

top: 537px;

left: 1016px;

z-index: 1;

}

</style>

</head>

<body>

<form runat="server">

<asp:ScriptManager runat="server">

</asp:ScriptManager>

<div class="navbar navbar-inverse navbar-fixed-top">

<div class="container">

<div class="navbar-header">

<button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar-collapse">

<span class="icon-bar"></span>

<span class="icon-bar"></span>

<span class="icon-bar"></span>

</button>

<a class="navbar-brand" runat="server" href="~/">Master page beautification</a>

</div>

<div class="navbar-collapse collapse">

<ul class="">

<li><a runat="server" href="~/">Home</a></li>

<li><a runat="server" href="~/About">About</a></li>

<li><a runat="server" href="~/Contact">Contact</a></li>

</ul>

</div>

</div>

</div>

<aside id="side">

<h1>Info</h1>

<a href="#">

<p>Your orders</p>

</a>

<a href="#">

<p>Cart</p>

</a>

<a href="#">

<p>

Wishlist

</p>

</a>

<asp:Button ID="Button2" runat="server" CssClass="auto-style1" Style="z-index: 1"

Text="Button" />

<asp:Button ID="Button1" runat="server" CssClass="auto-style2" Text="Button" />

</aside>

<div class="container body-content">

<asp:ContentPlaceHolder ID="MainContent" runat="server">

</asp:ContentPlaceHolder>

<hr />

<footer>

<p>Demonstraton of Master Page with applying Styles & Themes for page beautification.</p>

</footer>

</div>

</form>

</body>

</html>

FILE> StyleSheet.css

#header {

color: darkred;

text-align: center;

font-size: 20px;

}

#nav {

background-color: lightcyan;

padding: 5px;

}

ul {

list-style-type: none;

}

li a {

color: darkolivegreen;

font-size: 30px;

column-width: 5%;

}

li {

display: inline;

padding-left: 2px;

column-width: 20px;

}

a {

text-decoration: none;

margin-left: 20px

}

li a:hover {

background-color: aqua;

color: coral;

padding: 1%;

}

#side {

text-align: center;

float: right;

width: 15%;

padding-bottom: 79%;

background-color: #F1FAEE;

}

#article {

background-color: burlywood;

padding: 10px;

padding-bottom: 75%;

}

#footer {

background-color: #C7EFCF;

text-align: center;

padding-bottom: 5%;

font-size: 20px;

}

#con {

border: double;

border-color: burlywood;

}

FILE> Default.aspx

<%@ Page Title="Home Page" Language="C#" MasterPageFile="~/Site.Master" AutoEventWireup="true" CodeBehind="Default.aspx.cs" Inherits="Beautified.\_Default" %>

<asp:Content ID="BodyContent" ContentPlaceHolderID="MainContent" runat="server">

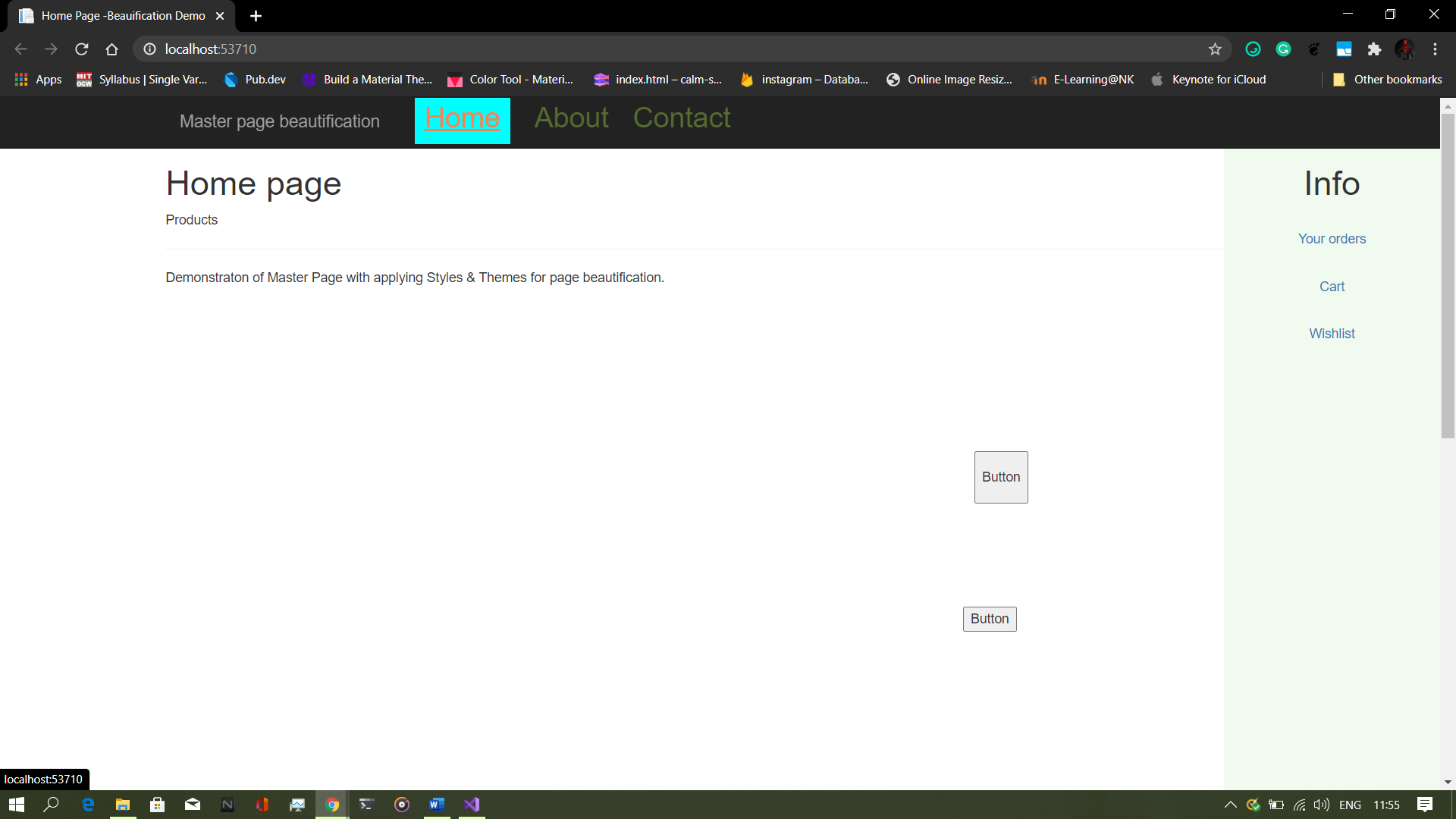
<h1>Home page</h1>

<p>

Products

</p>

</asp:Content>



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

FILE> Default.aspx

<%@ Page Title="Home Page" Language="C#" MasterPageFile="~/Site.Master" AutoEventWireup="true" CodeBehind="Default.aspx.cs" Inherits="ViewStateDemo.\_Default" %>

<asp:Content ID="BodyContent" ContentPlaceHolderID="MainContent" runat="server">

<div>

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

<br />

<asp:Button ID="Button1" runat="server" OnClick="Button1\_Click" Text="Get

Data" />

</div>

</asp:Content>

FILE> Default.aspx.cs

using System;

using System.Web.UI;

namespace ViewStateDemo

{

public partial class \_Default : Page

{

protected void Page\_Load(object sender, EventArgs e)

{

if (!IsPostBack)

{

string str = " Hi, This the View State Demo";

if (ViewState["name"] == null)

{

ViewState["name"] = str;

}

}

}

protected void Button1\_Click(object sender, EventArgs e)

{

Label1.Text = ViewState["name"].ToString();

}

}

}

