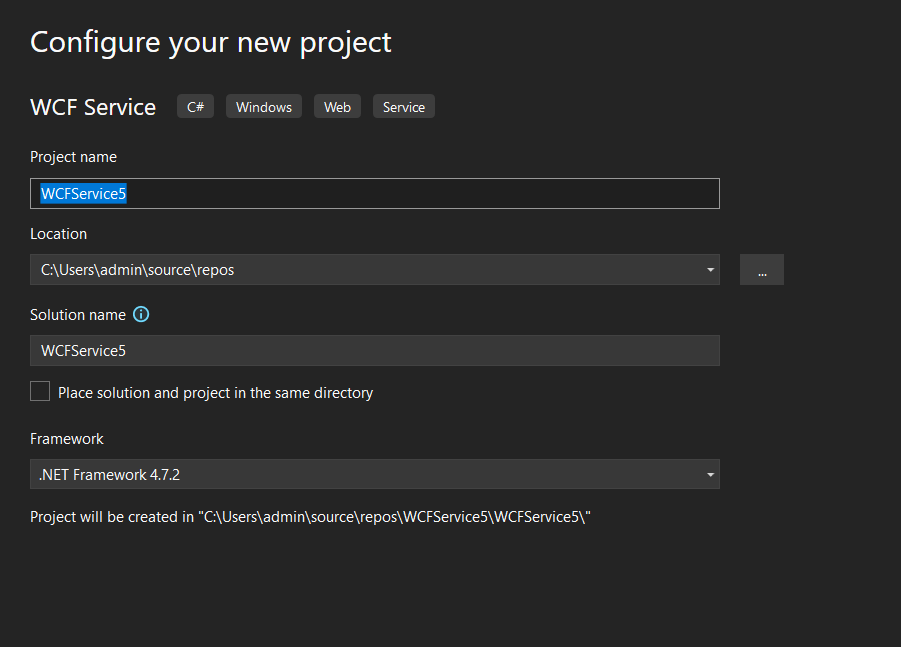
**Practical Number 15: Build a web application to create and use WCF service in ASP.net**



Iservice.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.ServiceModel.Web;

using System.Text;

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

[ServiceContract]

public interface IService

{

[OperationContract]

string GetData(int value);

[OperationContract]

double add(double a, double b);

[OperationContract]

double sub(double a, double b);

[OperationContract]

double mul(double a, double b);

[OperationContract]

double div(double a, double b);

[OperationContract]

CompositeType GetDataUsingDataContract(CompositeType composite);

// TODO: Add your service operations here

}

// Use a data contract as illustrated in the sample below to add composite types to service operations.

[DataContract]

public class CompositeType

{

bool boolValue = true;

string stringValue = "Hello ";

[DataMember]

public bool BoolValue

{

get { return boolValue; }

set { boolValue = value; }

}

[DataMember]

public string StringValue

{

get { return stringValue; }

set { stringValue = value; }

}

}

Servics.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.ServiceModel.Web;

using System.Text;

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

public class Service : IService

{

public string GetData(int value)

{

return string.Format("You entered: {0}", value);

}

public double add(double a, double b)

{

return a + b;

}

public double sub(double a, double b)

{

return a - b;

}

public double mul(double a, double b)

{

return a \* b;

}

public double div(double a, double b)

{

return a / b;

}

public CompositeType GetDataUsingDataContract(CompositeType composite)

{

if (composite == null)

{

throw new ArgumentNullException("composite");

}

if (composite.BoolValue)

{

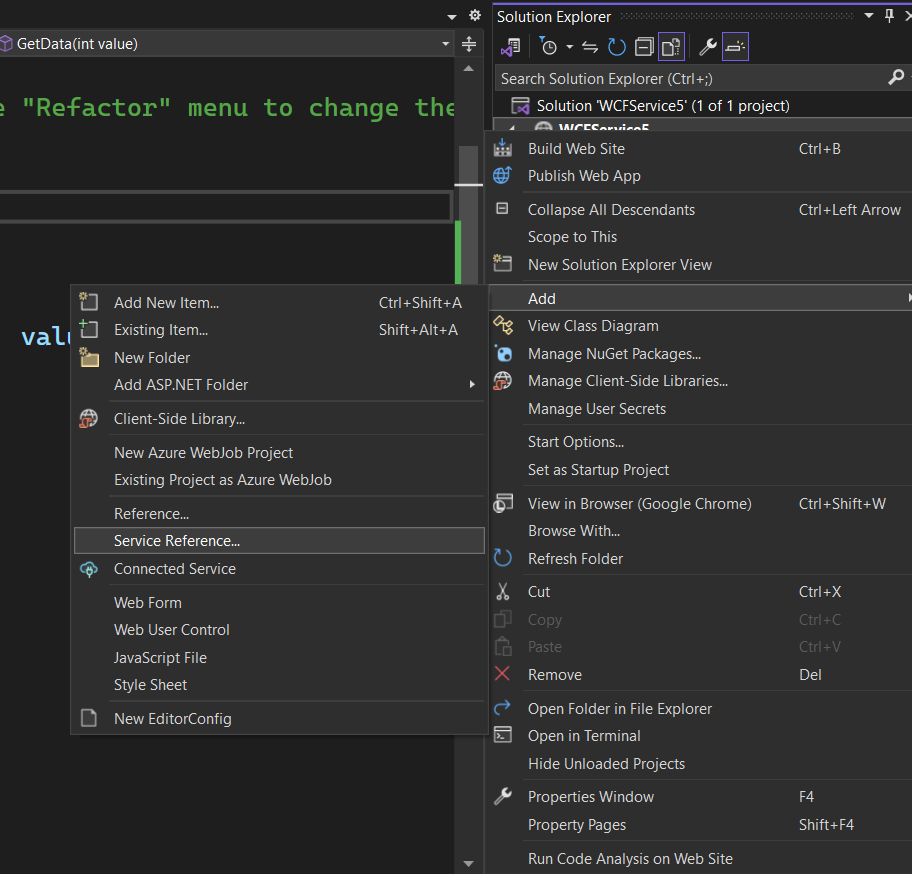
composite.StringValue += "Suffix";

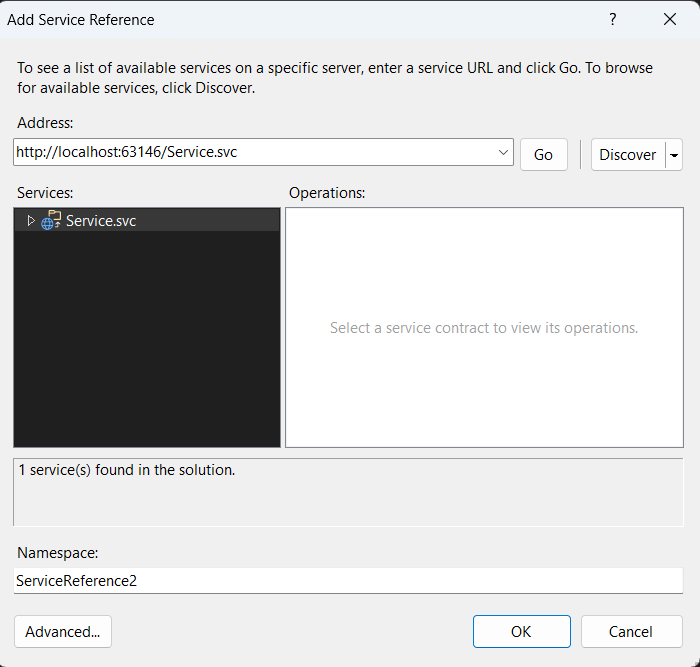
}

return composite;

}

}





WebForm.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

public partial class \_Default : System.Web.UI.Page

{

ServiceReference1.ServiceClient service = new ServiceReference1.ServiceClient();

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

{

}

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

{

double a = Convert.ToDouble(TextBox1.Text);

double b = Convert.ToDouble(TextBox2.Text);

double result= service.add(a, b);

Label1.Text = "Addition" + result.ToString();

}

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

{

double a = Convert.ToDouble(TextBox1.Text);

double b = Convert.ToDouble(TextBox2.Text);

double result = service.sub(a, b);

Label1.Text = "Subtraction" + result.ToString();

}

protected void Button3\_Click(object sender, EventArgs e)

{

double a = Convert.ToDouble(TextBox1.Text);

double b = Convert.ToDouble(TextBox2.Text);

double result = service.mul(a, b);

Label1.Text = "Multiplication" + result.ToString();

}

protected void Button4\_Click(object sender, EventArgs e)

{

double a = Convert.ToDouble(TextBox1.Text);

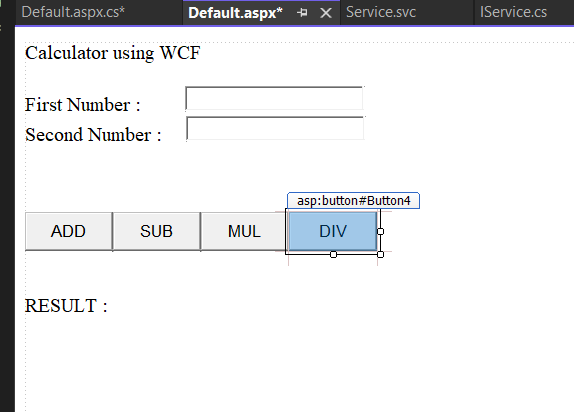
double b = Convert.ToDouble(TextBox2.Text);

double result = service.div(a, b);

Label1.Text = "Division" + result.ToString();

}

}



<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs" Inherits="\_Default" %>

<!DOCTYPE html>

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

<head runat="server">

<title></title>

</head>

<body>

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

<div>

Calculator using WCF<br />

<br />

First Number :<asp:TextBox ID="TextBox1" runat="server" style="margin-left: 44px" Width="171px"></asp:TextBox>

<br />

Second Number :

<asp:TextBox ID="TextBox2" runat="server" style="margin-left: 19px" Width="171px"></asp:TextBox>

<br />

<br />

<br />

<br />

<asp:Button ID="Button1" runat="server" Height="39px" OnClick="Button1\_Click" Text="ADD" Width="88px" />

<asp:Button ID="Button2" runat="server" Height="39px" OnClick="Button2\_Click" Text="SUB" Width="88px" />

<asp:Button ID="Button3" runat="server" Height="39px" OnClick="Button3\_Click" Text="MUL" Width="88px" />

<asp:Button ID="Button4" runat="server" Height="39px" OnClick="Button4\_Click" Text="DIV" Width="88px" />

<br />

<br />

<br />

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

<br />

<br />

<br />

<br />

<br />

<br />

<br />

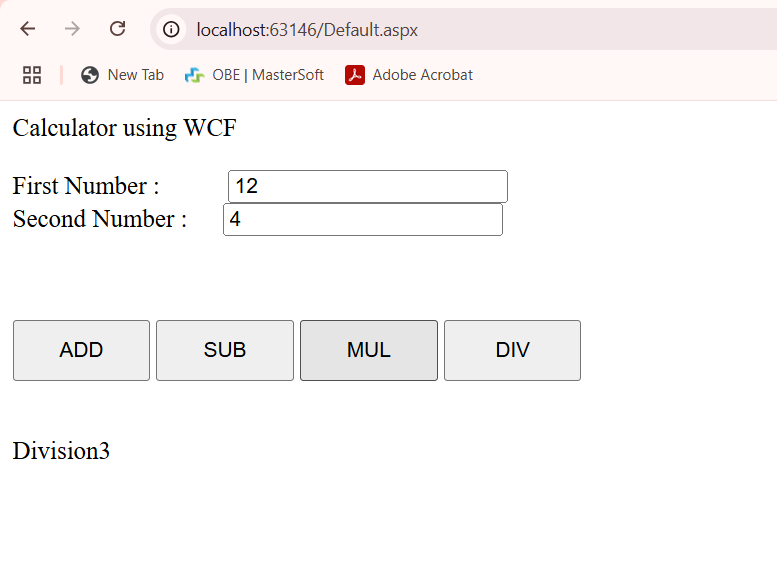
</div>

</form>

</body>

</html>

OUTPUT:



Here’s a complete **step-by-step guide** on how to perform **Practical 15: Build a Web Application to Create and Use a WCF Service in ASP.NET**—with **detailed clicks, where to click, and what to do**.

**🎯 Goal:**

Build a calculator in ASP.NET Web Forms using WCF Service with operations: Add, Subtract, Multiply, Divide.

**✅ STEP-BY-STEP INSTRUCTIONS**

**🌐 Step 1: Open Visual Studio**

1. Click **Start Menu** → Search **Visual Studio** → Open it.

**🔨 Step 2: Create WCF Service Project**

1. In Visual Studio:
   * Click **File** → **New** → **Project**
   * Search for **WCF Service Application**
   * Name it: CalculatorService
   * Click **Create**

**🛠 Step 3: Create Interface (IService.cs)**

1. In **Solution Explorer**, find IService.cs
2. Replace content with the following:

[ServiceContract]

public interface IService

{

[OperationContract] string GetData(int value);

[OperationContract] double add(double a, double b);

[OperationContract] double sub(double a, double b);

[OperationContract] double mul(double a, double b);

[OperationContract] double div(double a, double b);

[OperationContract]

CompositeType GetDataUsingDataContract(CompositeType composite);

}

[DataContract]

public class CompositeType

{

bool boolValue = true;

string stringValue = "Hello ";

[DataMember]

public bool BoolValue

{

get { return boolValue; }

set { boolValue = value; }

}

[DataMember]

public string StringValue

{

get { return stringValue; }

set { stringValue = value; }

}

}

**🧮 Step 4: Implement the Service (Service.cs)**

1. Open **Service.cs**
2. Replace content with:

public class Service : IService

{

public string GetData(int value) => $"You entered: {value}";

public double add(double a, double b) => a + b;

public double sub(double a, double b) => a - b;

public double mul(double a, double b) => a \* b;

public double div(double a, double b) => a / b;

public CompositeType GetDataUsingDataContract(CompositeType composite)

{

if (composite == null) throw new ArgumentNullException("composite");

if (composite.BoolValue) composite.StringValue += "Suffix";

return composite;

}

}

**▶️ Step 5: Run WCF Service**

1. Right-click project → **Set as Startup Project**
2. Press **F5** to run and host the service.
3. Copy the service URL (like http://localhost:12345/Service.svc)

**🌐 Step 6: Create ASP.NET Web Application to Consume WCF**

1. **File** → **New** → **Project**
2. Choose: **ASP.NET Web Application (.NET Framework)**
3. Name: WCFClientApp
4. Template: Choose **Empty**, then check **Web Forms** → Click **Create**

**🔌 Step 7: Add Service Reference**

1. In **Solution Explorer**:
   * Right-click project WCFClientApp → **Add** → **Service Reference**
2. In **Address**: paste the WCF Service URL → Click **Go**
3. Name the namespace: ServiceReference1
4. Click **OK**

**🖼 Step 8: Add Web Form and UI**

1. Right-click the project → **Add** → **Web Form**
2. Name it: Default.aspx
3. Replace the default UI with:

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

<div>

Calculator using WCF<br />

<br />

First Number :<asp:TextBox ID="TextBox1" runat="server" Width="171px"></asp:TextBox><br />

Second Number :<asp:TextBox ID="TextBox2" runat="server" Width="171px"></asp:TextBox><br /><br />

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

<asp:Button ID="Button2" runat="server" Text="SUB" OnClick="Button2\_Click" />

<asp:Button ID="Button3" runat="server" Text="MUL" OnClick="Button3\_Click" />

<asp:Button ID="Button4" runat="server" Text="DIV" OnClick="Button4\_Click" />

<br /><br />

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

</div>

</form>

**💻 Step 9: Code Behind (Default.aspx.cs)**

1. Open **Default.aspx.cs**
2. Replace the code with:

public partial class \_Default : System.Web.UI.Page

{

ServiceReference1.ServiceClient service = new ServiceReference1.ServiceClient();

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

{

double a = Convert.ToDouble(TextBox1.Text);

double b = Convert.ToDouble(TextBox2.Text);

double result = service.add(a, b);

Label1.Text = "Addition: " + result;

}

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

{

double a = Convert.ToDouble(TextBox1.Text);

double b = Convert.ToDouble(TextBox2.Text);

double result = service.sub(a, b);

Label1.Text = "Subtraction: " + result;

}

protected void Button3\_Click(object sender, EventArgs e)

{

double a = Convert.ToDouble(TextBox1.Text);

double b = Convert.ToDouble(TextBox2.Text);

double result = service.mul(a, b);

Label1.Text = "Multiplication: " + result;

}

protected void Button4\_Click(object sender, EventArgs e)

{

double a = Convert.ToDouble(TextBox1.Text);

double b = Convert.ToDouble(TextBox2.Text);

double result = service.div(a, b);

Label1.Text = "Division: " + result;

}

}

**🧪 Step 10: Run the Client Application**

1. Right-click **WCFClientApp** → Set as **Startup Project**
2. Press **F5**
3. Enter values, click buttons → see results

**✅ OUTPUT SAMPLE:**

Calculator using WCF

First Number: 10

Second Number: 5

Click ADD ➜ RESULT: Addition: 15

Click SUB ➜ RESULT: Subtraction: 5

Click MUL ➜ RESULT: Multiplication: 50

Click DIV ➜ RESULT: Division: 2

Would you like a **PDF version** of this or **viva questions** based on Practical 15?