

Practical 1:

Aim: Write a program to implement a simple web service that converts the temperature from Fahrenheit to Celsius and vice a versa.

Webform1.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm1.aspx.cs" Inherits="Temperature_Convertor.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="Farenheit To Celcius">
            </asp:Label>
            <br />
            Enter Temperature in Farenheit :&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
            <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
            <br />
            <br />
            <asp:Button ID="Button1" runat="server" OnClick="Button1_Click"
                Text="ConvertTOCelcius" />
            <br />
            <br />
            <asp:Label ID="lbl_result1" runat="server" Text="Label"></asp:Label>
            <br />
            <br />
```

Celcius To Farenheit

Enter Temperature in
Celcius:

<asp:TextBox ID="TextBox2" runat="server"></asp:TextBox>

<asp:Button ID="Button2" runat="server" OnClick="Button2_Click"
Text="ConvertToFarenheit" />

<asp:Label ID="lbl_result2" runat="server" Text="Label"></asp:Label>

</div>

</form>

</body>

</html>

WebForm1.aspx.cs file

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

namespace Temperature_Convertor

{

public partial class WebForm1 : System.Web.UI.Page

{

protected void Page_Load(object sender, EventArgs e)

```
{  
}  
  
protected void Button1_Click(object sender, EventArgs e)  
{  
  
    teperature t = new teperature();  
  
    lbl_result1.Text = t.FarenheitToCelcius(Double.Parse(textBox1.Text)).ToString();  
}  
  
protected void Button2_Click(object sender, EventArgs e)  
{  
  
    teperature t1 = new teperature();  
  
    lbl_result2.Text =  
    t1.CelciusToFarenheit(Double.Parse(textBox2.Text)).ToString();  
}  
}  
}  
}
```

Temperature.asmx file

```
using System;  
  
using System.Collections.Generic;  
  
using System.Linq;  
  
using System.Web;  
  
using System.Web.Services;  
  
namespace Temperature_Convertor  
{  
  
    /// <summary>  
    /// Summary description for teperature  
    /// </summary>  
  
    [WebService(Namespace = "http://tempuri.org/")]
```

```
[WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]  
[System.ComponentModel.ToolboxItem(false)]  
  
// To allow this Web Service to be called from script, using ASP.NET AJAX,  
uncomment the following line.  
  
// [System.Web.Script.Services.ScriptService]  
  
public class teperature : System.Web.Services.WebService  
  
{  
  
    [WebMethod]  
  
    public string HelloWorld()  
  
    {  
  
        return "Hello World";  
  
    }  
  
    [WebMethod]  
  
    public double FarenheitToCelcius(double fahr)  
  
    {  
  
        return ((fahr - 32) / 9 * 5);  
  
    }  
  
    [WebMethod]  
  
    public double CelciusToFarenheit(double cel)  
  
    {  
  
        return ((cel*9)/5)+32;  
  
    }  
  
}  
  
}  
  
}  
  
OUTPUT:
```

Celsius To Fahrenheit Conve localhost

← → ⌂ ⌂ https://localhost:44382/WebForm1.aspx

Farenheit To Celcius

Enter Temperature in Farenheit :

ConvertTOCelcius

15.5555555555556

Celcius To Farenheit

Enter Temperature in Celcius:

ConvertToFarenheit

62.6



Type here to search



Practical 2:

Aim: Write a program to implement the operation can receive request and will

return a response in two ways.

1. One-Way operation
2. Two- Way opearation

Service1.svc.cs

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

```
namespace WcfPractical2
```

```
{
```

```
// NOTE: You can use the "Rename" command on the "Refactor" menu to change
// the class name "Service1" in code, svc and config file together.
```

```
// NOTE: In order to launch WCF Test Client for testing this service, please select
// Service1.svc or Service1.svc.cs at the Solution Explorer and start debugging.
```

```
public class Service1 : IService1
```

```
{
```

```
    public string GetData(int value)
```

```
{
```

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

```
}
```

```
    public void subtract(double n1, double n2)
```

```
{
```

```
        double result = n1 - n2;
```

```
        Console.WriteLine("Subtract({0},{1})={2}",n1,n2,result);
```

```
}
```

```
public double add(double n1, double n2)
{
    double result = n1 + n2;
    return result;
}

public void multiply(double n1, double n2)
{
    double result = n1 * n2;
    Console.WriteLine("multiply({0},{1})={2}", n1, n2, result);
}

public void divide(double n1, double n2)
{
    double result = n1 / n2;
    Console.WriteLine("divide({0},{1})={2}", n1, n2, result);
}

public String SayHello(string name)
{
    Console.WriteLine("SayHello({0})", name);
    return "Hello " + name;
}

public CompositeType GetDataUsingDataContract(CompositeType composite)
{
    if (composite == null)
    {
        throw new ArgumentNullException("composite");
    }
}
```

```
if (composite.BoolValue)
{
    composite.StringValue += "Suffix";
}
return composite;
}
}
```

IService1.cs

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

```
using System.ServiceModel;
```

```
using System.ServiceModel.Web;
```

```
using System.Text;
```

```
namespace WcfPractical2
```

```
{
```

```
// NOTE: You can use the "Rename" command on the "Refactor" menu to change
// the interface name "IService1" in both code and config file together.
```

```
[ServiceContract]
```

```
public interface IService1
```

```
{
```

```
[OperationContract]
```

```
string GetData(int value);
```

```
[OperationContract]
```

```
CompositeType GetDataUsingDataContract(CompositeType composite);
```

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

```
[OperationContract(IsOneWay = true)]
```

```
void subtract(double n1, double n2);
```

```
[OperationContract(IsOneWay = true)]
```

```
void multiply(double n1, double n2);
```

```
[OperationContract(IsOneWay = true)]
```

```
void divide(double n1, double n2);
```

```
[OperationContract]
```

```
string SayHello(string name);
```

```
[OperationContract]
```

```
double add(double n1, double n2);
```

```
}
```

```
// 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; }  
}  
}  
}
```

Output:

Process: [580] iisexpress.exe

IService1.cs

WcfPractical2

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17

135 %

N

Autos

Adding service ...

Search (Ctrl+E)



Search Depth:

Name

Name

Value

Type

Autos

Locals

Watch 1

Call Stack Break

Ready



Process: [580] iisexpress.exe

IService1.cs

WcfPractical2

```
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17
```

135 %

N

Autos

Service added successfully.

Search (Ctrl+E)



Search Depth:

Name

Name

Value

Type

Autos

Locals

Watch 1

Call Stack Breakpoints

Ready



IService1.cs

WcfPractica

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17

WCF Test Client

File Tools Help

- My Service Projects
 - http://localhost:54077/Service1.svc
 - IService1 (BasicHttpBinding_IService)
 - GetData()
 - GetDataAsync()
 - GetDataUsingDataContract()
 - GetDataUsingDataContractAsync()
 - subtract()
 - subtractAsync()
 - multiply()
 - multiplyAsync()
 - divide()
 - divideAsync()
 - SayHello()
 - SayHelloAsync()
 - add()
 - addAsync()
 - Config File

multiply

Request

Name	Value
n1	5
n2	2

Security Warning

You are about to send information over the network. Insecure bindings or malicious services could allow private information to be viewed by others. Do not try to invoke services you do not trust, and use a secure binding if you are transmitting information you do not want others to see.

 In the future, do not show this message.

Formatted XML

Autos

Service added successfully.

Search (Ctrl+E)



← → Search Depth:

Name

Name

Value

Type

Autos

Locals

Watch 1

XAML Bin...

Ready



IService1.cs

WcfPractica

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17

WCF Test Client

File Tools Help

My Service Projects	
http://localhost:54077/Service1.svc	
IService1 (BasicHttpBinding_IService)	

- GetData()
- GetDataAsync()
- GetDataUsingDataContract()
- GetDataUsingDataContractAsync()
- subtract()
- subtractAsync()
- multiply()
- multiplyAsync()
- divide()
- divideAsync()
- SayHello()
- SayHelloAsync()
- add()
- addAsync()

Config File

multiply

Request

Name	Value
n1	5
n2	2

Microsoft WCF Test Client



One-way message is successfully delivered.

Formatted XML

Autos

Service invocation completed.

Search (Ctrl+E)



← → Search Depth:

Name

Name

Value

Type

Autos

Locals

Watch 1

XAML Bin...

Ready



IService1.cs

WcfPractices

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17

WCF Test Client

File Tools Help

My Service Projects	
http://localhost:54077/Service1.svc	
IService1 (BasicHttpBinding_IService)	

- GetData()
- GetDataAsync()
- GetDataUsingDataContract()
- GetDataUsingDataContractAsync()
- subtract()
- subtractAsync()
- multiply()
- multiplyAsync()
- divide()
- divideAsync()
- SayHello()
- SayHelloAsync()
- add()
- addAsync()

Config File

multiply

Request

Name	Value
n1	5
n2	2

Response

Name	Value
(return)	(null)

Formatted XML

Autos

Service invocation completed.

Search (Ctrl+E)



← → Search Depth:

Name

Name

Value

Type

Autos

Locals

Watch 1

XAML Bin...

Ready



Screenshot of Microsoft Visual Studio showing the WCF Test Client interface.

The top menu bar includes: File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, and Extensions.

The toolbar includes: Back, Forward, Stop, Refresh, Breakpoints, and other debugging icons.

The status bar shows: Process: [580] iisexpress.exe, Lifecycle Events, Thread: Thread 1.

The main window displays the WCF Test Client interface:

- File**, **Tools**, **Help** menu.
- My Service Projects** node expanded, showing:
 - http://localhost:54077/Service1.svc**
 - IService1 (BasicHttpBinding_IService)** node expanded, showing:
 - GetData()
 - GetDataAsync()
 - GetDataUsingDataContract()
 - GetDataUsingDataContractAsync()
 - subtract()
 - subtractAsync()
 - multiply()
 - multiplyAsync()
 - divide()
 - divideAsync()
 - SayHello()
 - SayHelloAsync()
 - add()
 - addAsync()
 - Config File**

The **Request** pane shows a **multiply** operation with a **Name** value of **name**.

The **Response** pane shows a **Name** value of **(return)** and a **Value** of **"Hello"**.

The bottom status bar shows: Autos, Service invocation completed.

The bottom navigation bar includes: Search (Ctrl+E), Search Depth, Name, Autos, Locals, Watch 1, XAML Bin..., and Ready.

Practical 3: Develop client which consumes web services developed in different platform.

STEP 1: CREATE AN EMPTY WEB APPLICATION IN ASP.NET.

STEP 2: AFTER THE CREATION OF WEB APPLICATION ADD NEW ITEM TO THE WEB APPLICATION BY RIGHT CLICKING ON SOLUTION

AND ADD NEW ITEM.

STEP 3: SELECT WEB SERVICE(ASMX) FILE.

STEP 4: RENAME THE FILE AS Opearation.asmx.

Opearation.asmx

```
using System;
```

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

```
using System.Linq;
```

```
using System.Web;
```

```
using System.Web.Services;
```

```
namespace ClientConsumer
```

```
{
```

```
/// <summary>
```

```
/// Summary description for Operation
```

```
/// </summary>
```

```
[WebService(Namespace = "http://tempuri.org/")]
```

```
[WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
```

```
[System.ComponentModel.ToolboxItem(false)]
```

```
// To allow this Web Service to be called from script, using ASP.NET AJAX,  
uncomment the following line.
```

```
// [System.Web.Script.Services.ScriptService]
```

```
public class Operation : System.Web.Services.WebService
```

```
{
```

```
[WebMethod]
```

```
public double Add(double a,double b)
```

```
{  
    double sum = a + b;  
  
    return sum;  
}  
  
[WebMethod]  
  
public double Multi(double a, double b)  
{  
    double product = a * b;  
  
    return product;  
}  
}  
}
```



Operation

The following operations are supported. For a formal definition, please review the [Service Description](#).

- [Add](#)
- [Multi](#)

This web service is using <http://tempuri.org/> as its default namespace.

Recommendation: Change the default namespace before the XML Web service is made available.

Each XML Web service needs a unique namespace in order for client applications to distinguish it from other services. XML Web services should use a more permanent namespace.

Your XML Web service should be identified by a namespace that you control. For example, you can use <http://microsoft.com/webservices/> as the namespace for your XML Web service. (XML Web service namespaces are URIs.)

For XML Web services created using ASP.NET, the default namespace can be changed using the `WebService` attribute on the service methods. Below is a code example that sets the namespace to "<http://microsoft.com/webservices/>".

C#

```
[WebService(Namespace="http://microsoft.com/webservices/")]
public class MyWebService {
    // implementation
}
```

Visual Basic

```
<WebService(Namespace:="http://microsoft.com/webservices/")> Public Class MyWebService
    ' implementation
End Class
```

C++

```
[WebService(Namespace="http://microsoft.com/webservices/")]
public ref class MyWebService {
    // implementation
};
```

For more details on XML namespaces, see the W3C recommendation on [Namespaces in XML](#).

For more details on WSDL, see the [WSDL Specification](#).

For more details on URIs, see [RFC 2396](#).





https://localhost:44386/Operation.asmx?WSDL

This XML file does not appear to have any style information associated with it. The document may be plain text or it may be a document with no style information stored in the file.

```
<wsdl:definitions xmlns:tm="http://microsoft.com/wsdl/mime/textMatching/" xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:http="http://schemas.xmlsoap.org/wsdl/http/" xmlns:types="http://tempuri.org/Types.xsd">  
  <wsdl:types>  
    <s:schema elementFormDefault="qualified" targetNamespace="http://tempuri.org/Types.xsd">  
      <s:element name="Add">  
        <s:complexType>  
          <s:sequence>  
            <s:element minOccurs="1" maxOccurs="1" name="a" type="s:double"/>  
            <s:element minOccurs="1" maxOccurs="1" name="b" type="s:double"/>  
          </s:sequence>  
        </s:complexType>  
      </s:element>  
      <s:element name="AddResponse">  
        <s:complexType>  
          <s:sequence>  
            <s:element minOccurs="1" maxOccurs="1" name="AddResult" type="s:double"/>  
          </s:sequence>  
        </s:complexType>  
      </s:element>  
      <s:element name="Multi">  
        <s:complexType>  
          <s:sequence>  
            <s:element minOccurs="1" maxOccurs="1" name="a" type="s:double"/>  
            <s:element minOccurs="1" maxOccurs="1" name="b" type="s:double"/>  
          </s:sequence>  
        </s:complexType>  
      </s:element>  
      <s:element name="MultiResponse">  
        <s:complexType>  
          <s:sequence>  
            <s:element minOccurs="1" maxOccurs="1" name="MultiResult" type="s:double"/>  
          </s:sequence>  
        </s:complexType>  
      </s:element>  
    </s:schema>  
  </wsdl:types>  
  <wsdl:message name="AddSoapIn">  
    <wsdl:part name="parameters" element="tns:Add"/>  
  </wsdl:message>  
  <wsdl:message name="AddSoapOut">
```



Type here to search



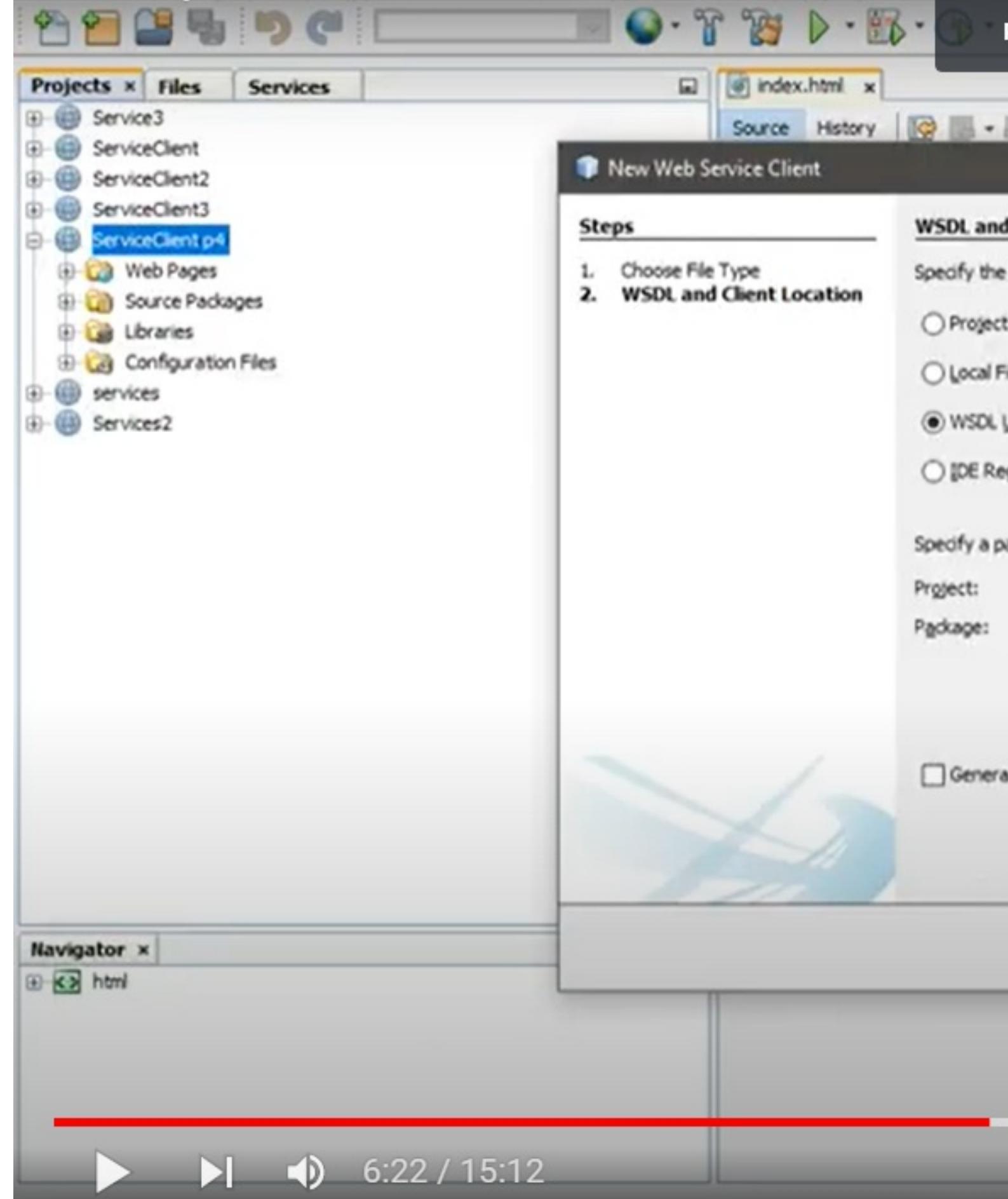
STEP 5: OPEN NETBEANS IDE AND CREATE NEW JAVA WEB APPLICATION.

STEP 6: EXPAND THE PROJECT AND RIGHT CLICK ON PROJECT NAME.

STEP 7: SELECT WEB SERVICE CLIENT AND CLICK ON NEXT.

STEP 8: IN WSDL AND CLIENT LOCATION WINDOW SELECT WSDL URL RADIO BUTTON. COPY AND PASTE THE WSDL FILE URL.

Develop client which consumes web services de



STEP 9: SPECIFY PACKAGE AND CLICK ON FINISH. IN WEB SERVICE REFERENCES OPERATION IS VISIBLE.

CLICK ON OPERATIONSOAP. THE WEB METHODS FROM .NET APPLICATION ARE VISIBLE.

ServiceClient p4 - NetBeans IDE 8.0.1

Develop client which consumes web services de

Projects x Files Services

- Service3
- ServiceClient
- ServiceClient2
- ServiceClient3
- ServiceClient p4
 - Web Pages
 - Source Packages
 - Generated Sources (jax-ws)
 - Libraries
 - Web Service References
 - Operation
 - Operation
 - OperationSoap
 - AND
 - Multi
 - OperationSoap12
 - Configuration Files
 - services
 - Services2

index.html x

Source History

```
<!DOCTYPE html>
<!--
To change this
To change this
and open the t
-->
<html>
<head>
<title>
<meta >
<meta >
</head>
<body>
<div>T
</body>
</html>
```

Output x

Retriever Output x Service

```
com\rc\MultiRespon
com\rc\ObjectFacto
com\rc\Operation.J
com\rc\OperationSo
com\rc\package-inf
Copying 0 files to
BUILD SUCCESSFUL (
```

Navigator x

html

7:08 / 15:12

STEP 10: CREATE 2 JSP FILES BY RIGHT CLICKING ON SOURCE PACKAGES OF YOUR PROJECT. NAME THEM AS ADDITION.JSP AND MULTIPLICATION.JSP.

ADDITION.JSP

Right click in the body tag of the page.

Select Web service client resources and select -> call *web service operation*.

Select operation to invoke window appears.

From *OpearationSoap* select *Add*.

Code of Add method will be added automatically. Do the necessary changes in the file.

Develop client which consumes web services de

File Edit View Favorites Tools Window Debug Style Help Window Help

Projects x Files Services

Service3 ServiceClient ServiceClient2 ServiceClient3 ServiceClient p4 Web Pages WEB-INF Addition.jsp Multiplication.jsp index.html Source Packages Generated Sources (jax-ws) Libraries Web Service References Operation Operation OperationSoap Add Multi OperationSoap12 Configuration Files services Services2

Addition.jsp x Multiplication.jsp

Source History

```

13 </head>
14 <body>
15 <%-- start
16 <%
17 try {
18     com.rc
19     com.rc
20     // TODO
21     double
22     double
23     // TODO
24     double
25     out.pr
26 } catch (E
27     // TODO
28 }
29 %>
30 <%-- end w
31 </body>
32 </html>
33

```

>

Output x

Retriever Output x Services

```

com\rc\MultiRespon
com\rc\ObjectFacto
com\rc\Operation.j
com\rc\OperationSo
com\rc\package-inf
Copying 8 files to
BUILD SUCCESSFUL (1)

```

Navigator x

html head meta title body

Filters: ▶️ 🔍 🔒 ⏪ ⏩ 🔊 11:40 / 15:12

Multiplication.jsp

Right click in the body tag of the page.

Select Web service client resources and select -> call *web service operation*.

Select operation to invoke window appears.

From *OpearationSoap* select *Add*.

Code of Multi method will be added automatically. Do the necessary changes in the file.

Develop client which consumes web services de...

File Edit View Insert Tools Window Help

Projects x Files Services

- Service3
- ServiceClient
- ServiceClient2
- ServiceClient3
- ServiceClient p4
 - Web Pages
 - WEB-INF
 - Addition.jsp
 - Multiplication.jsp
 - index.html
 - Source Packages
 - Generated Sources (jax-ws)
 - Libraries
 - Web Service References
 - Operation
 - Operation
 - OperationSoap
 - Add
 - Multi
 - OperationSoap12
 - Configuration Files
 - services
 - Services2

Navigator x

- html
 - head
 - meta
 - title
 - body

Filters: ▶ | ⌂ | ⌂ | 12:01 / 15:12

Additional.jsp x Multiplication.jsp

Source History

```

13 </head>
14 <body>
15
16 <%-- start
17 <% double
18 double
19 try {
20   com.rc.
21   com.rc.
22 // TODO
23 double
24 double
25 // TODO
26 double
27 double
28 // TODO
29 out.pr
30 } catch (E
31 // TODO
32 }
33 <%-- end w
34 html > body >

```

Output x

Retriever Output x Service

com\rc\MultiResponse
com\rc\ObjectFactory
com\rc\Operation
com\rc\OperationSoap
com\rc\package-info
Copying 0 files to
BUILD SUCCESSFUL (

STEP 11:

Now create index.HTML File.

Develop client which consumes web services de

File Edit View Navigator Tools Window Debug Style Tools Help Home

Projects x Files Services

Service3 ServiceClient ServiceClient2 ServiceClient3 ServiceClient p4 Web Pages WEB-INF Addition.jsp Multiplication.jsp index.html Source Packages Generated Sources (jax-ws) Libraries Web Service References Operation Operation OperationSoap Add Multi OperationSoap12 Configuration Files services Services2

Addition.jsp x Multiplication.jsp

Source History

```

7 <html>
8 <head>
9 <title>
10 <meta>
11 <meta>
12 </head>
13 <body>
14 <form>
15 <input type="text" name="num1" />
16 <input type="text" name="num2" />
17 <input type="button" value="Add" />
18 <input type="button" value="Multi" />
19 </form>
20 </body>
21 </html>
22
23

```

html > body >

Output x

Retriever Output x Services

Incrementally deployed
Completed incrementally
run-deploy:
Browsing: http://127.0.0.1:8080/ServiceClient/p4/Addition.jsp
run-display-browser:
run:
BUILD SUCCESSFUL (total time: 1 second)

Navigator x

br br input br br
input input

14:20 / 15:12

OUTPUT:

Develop client which consumes web services de

File Edit View Favorites Tools Window Debug Help

Projects x Files Services Addition.jsp x Multiplica

Service3 ServiceClient ServiceClient2 ServiceClient3 ServiceClient p4 Web Pages WEB-INF Addition.jsp Multiplication.jsp index.html Source Packages Generated Sources (jax-ws) Libraries Web Service References Operation Operation OperationSoap Add Multi OperationSoap12 Configuration Files services Services2

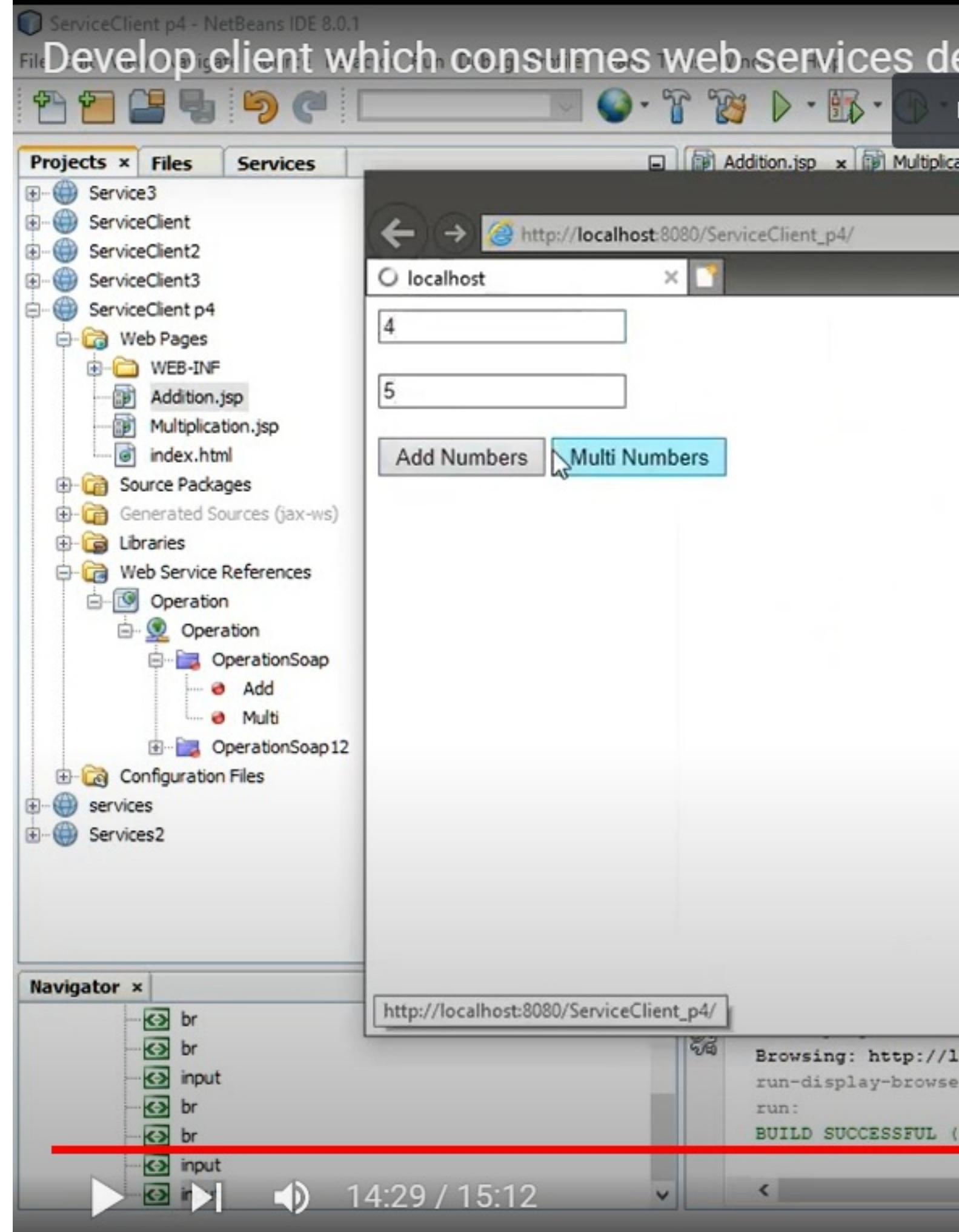
← → http://localhost:8080/ServiceClient_p4/ TODO supply a title Enter FirstNumber Enter SecondNumber Add Numbers Multi Numbers

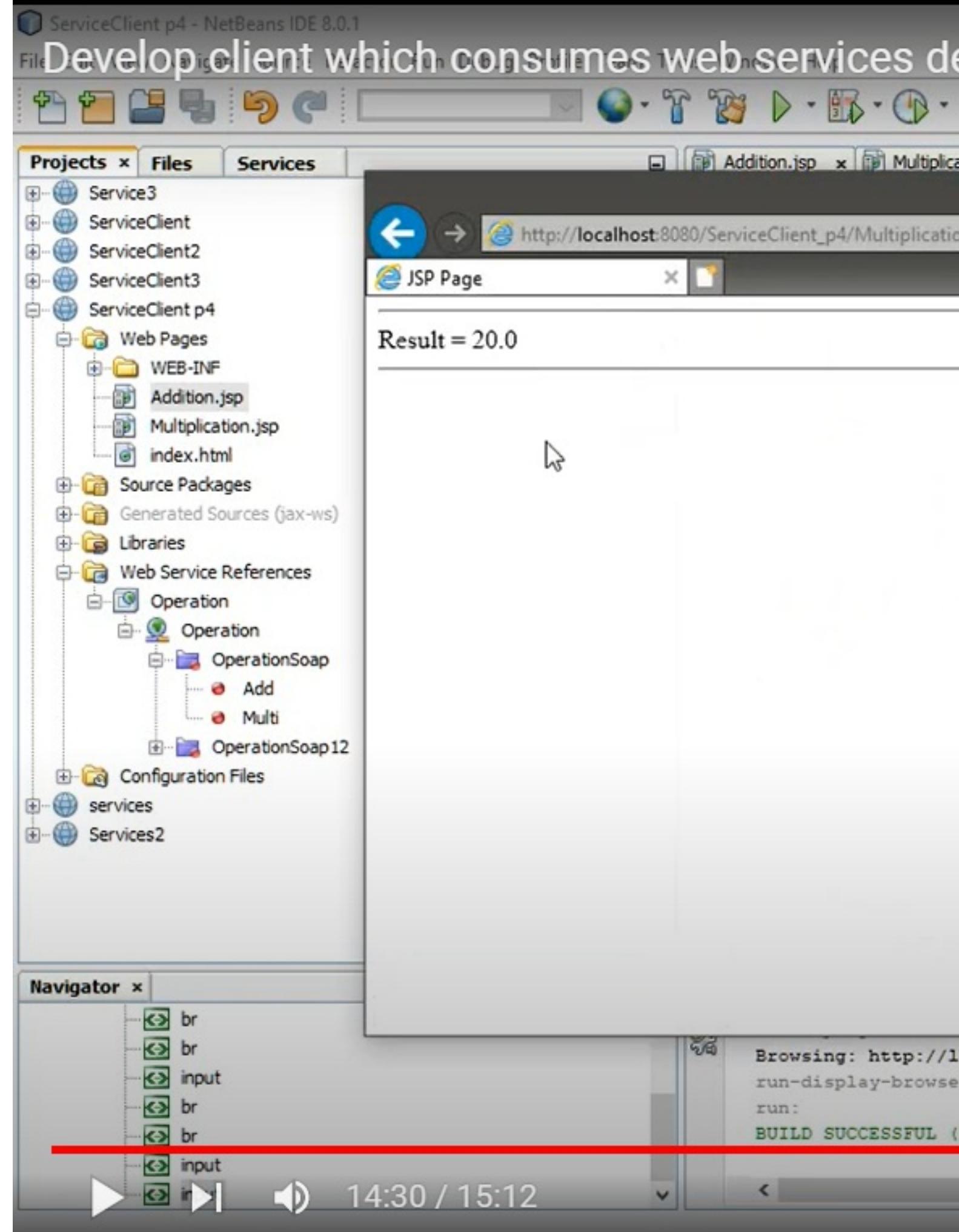
Navigator x br br input br br input input

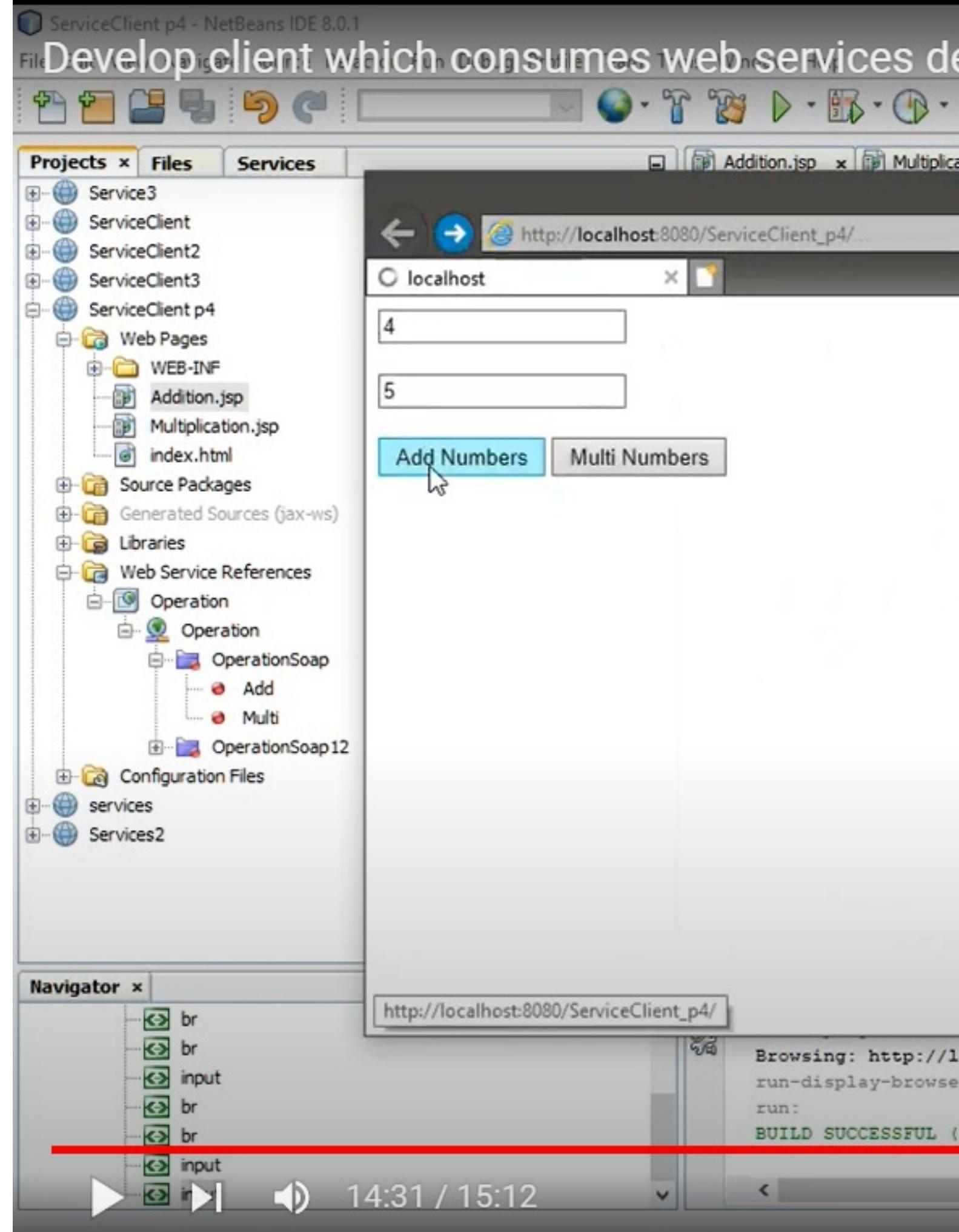
Browsing: http://1 run-display-browser run: BUILD SUCCESSFUL (

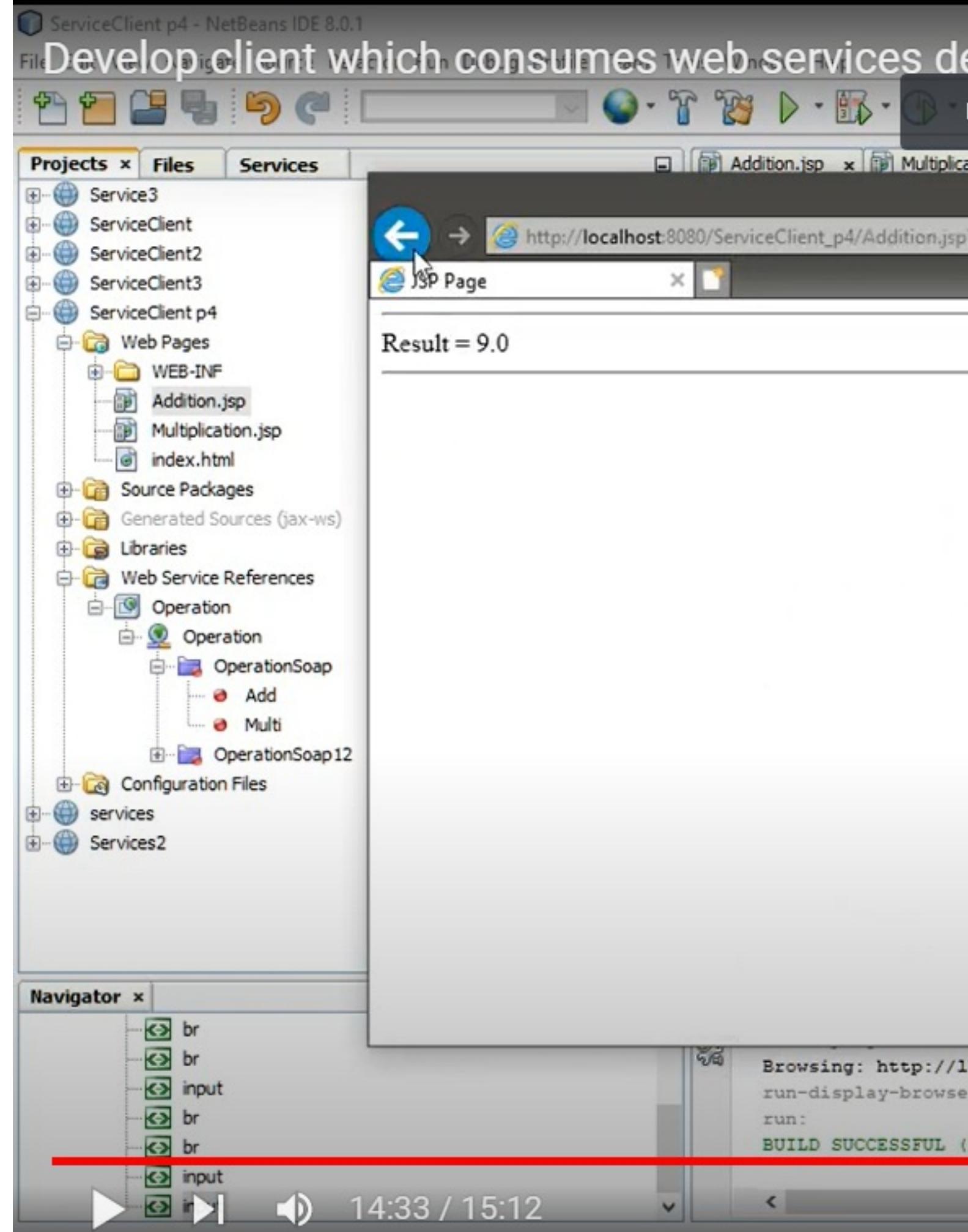
Starting browser

14:25 / 15:12









Practical 4

Aim: Write a program to implement business UDDI Registry entry.

Write a program to implement business UDDI Registry entry

```
<businessEntity businessKey = "uuid:C0E6D5A8-C446-4f01-99DA-  
70E212685A40"  
  
operator = "http://www.ibm.com" authorizedName = "John Doe">  
  
<name>Acme Company</name>  
  
<description>  
We create cool Web services  
</description>  
  
<contacts>  
  
<contact useType = "general info">  
  
<description>General Information</description>  
  
<personName>John Doe</personName>  
  
<phone>(123) 123-1234</phone>  
  
<email>jdoe@acme.com</email>  
  
</contact>  
  
</contacts>  
  
<businessServices>  
  
...  
  
</businessServices>  
  
<identifierBag>  
  
<keyedReference tModelKey = "UUID:8609C81E-EE1F-4D5A-B202-  
3EB13AD01823"  
  
name = "D-U-N-S" value = "123456789" />  
  
</identifierBag>  
  
<categoryBag>  
  
<keyedReference tModelKey = "UUID:C0B9FE13-179F-413D-8A5B-  
5004DB8E5BB2"  
  
name = "NAICS" value = "111336" />
```

</categoryBag>

</businessEntity>

Creating Registry

After obtaining an authentication token from one of the operators Microsoft, for example the XYZ.com developers decide what information to publish to the registry and use one of the UDDI tools provided by Microsoft.

POST /save_business HTTP/1.1

Host: www.XYZ.com

Content-Type: text/xml; charset = "utf-8"

Content-Length: nnnn

SOAPAction: "save_business"

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

<Envelope xmlns = "http://schemas/xmlsoap.org/soap/envelope/">

<Body>

<save_business generic = "2.0" xmlns = "urn:uddi-org:api_v2">

<businessKey = "">

</businessKey>

<name>

XYZ, Pvt Ltd.

</name>

<description>

Company is involved in giving Stat-of-the-art....

</description>

<identifierBag> ... </identifierBag>

...

</save_business>

</Body>

</ Envelope>

Retrieving Information

After XYZ Company has updated its UDDI entry with the relevant information, companies that want to become XYZ distributors can look up contact information in the UDDI registry and obtain the service descriptions and the access points for the two Web services that XYZ.com publishes for online order entry.

POST /get_businessDetail HTTP/1.1

Host: www.XYZ.com

Content-Type: text/xml; charset = "utf-8"

Content-Length: nnnn

SOAPAction: "get_businessDetail"

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

<Envelope xmlns = "http://schemas/xmlsoap.org/soap/envelope/">

<Body>

<get_businessDetail generic = "2.0" xmlns = "urn:uddi-org:api_v2">

<businessKey = "C90D731D-772HSH-4130-9DE3-5303371170C2">

</businessKey>

</get_businessDetail>

</Body>

</Envelope>

Practical 5: Write a JAX-WS web service to perform the following operations. Define a Servlet or JSP that consumes the web service.

1. Create a Web application in Netbeans IDE.
2. Now select web application name and right click and select new -> web service. Give name and select Implement web service by stateless session beans.
3. Select NewWebService and right click and select add operation. Create method add with two parameters.
4. Now test your web service and deploy it.
5. Now consume your web service.

Create new project ->java web ->new web application.

1. Now add new web Service client by right clicking on web application name and add project url for web reference.
2. Now add new servlet.
3. Drag and drop add method to java code.

Code for java file:

```
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.xml.ws.WebServiceRef;
import com.add.NewWebService_Service;
/**
 *
 * @author lab503
 */
@WebServlet(urlPatterns = {" /addition"})
public class addition extends HttpServlet {
    @WebServiceRef(wsdlLocation = "WEB-INF/wsdl/localhost_8080/NewWebService/NewWebService.wsdl")
    private NewWebService_Service service;
    /**
     * Processes requests for both HTTP <code>GET</code> and
     * <code>POST</code>
     * methods.
     *
     * @param request servlet request

```

```
* @param response servlet response
*
* @throws ServletException if a servlet-specific error occurs
*
* @throws IOException if an I/O error occurs
*/
protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
throws ServletException, IOException {
response.setContentType("text/html;charset=UTF-8");
try (PrintWriter out = response.getWriter()) {
/* TODO output your page here. You may use following sample code. */
out.println("<!DOCTYPE html>");
out.println("<html>");
out.println("<head>");
out.println("<title>Servlet addition</title>");
out.println("</head>");
out.println("<body>");
out.println("<form>");
out.println("Enter first number :<input type ='text' name='n1'>");
out.println("Enter second number :<input type ='text' name='n2'>");
out.println("<input type='submit' value='add' /> </form>");
out.println("<h1>Addition of two numbers is "+ add(Integer.parseInt(request.getParameter("n1")),Integer.parseInt(request.getParameter("n2"))));
out.println("</body>");
out.println("</html>");
}
}

// <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on the left to edit the code.">
```

```
/**  
 * Handles the HTTP <code>GET</code> method.  
 *  
 * @param request servlet request  
 * @param response servlet response  
 * @throws ServletException if a servlet-specific error occurs  
 * @throws IOException if an I/O error occurs  
 */  
  
@Override  
  
protected void doGet(HttpServletRequest request, HttpServletResponse  
response)  
throws ServletException, IOException {  
processRequest(request, response);  
}  
  
/**  
 * Handles the HTTP <code>POST</code> method.  
 *  
 * @param request servlet request  
 * @param response servlet response  
 * @throws ServletException if a servlet-specific error occurs  
 * @throws IOException if an I/O error occurs  
 */  
  
@Override  
  
protected void doPost(HttpServletRequest request, HttpServletResponse  
response)  
throws ServletException, IOException {  
processRequest(request, response);
```

```
}

/**

 * Returns a short description of the servlet.
 *
 *
 * @return a String containing servlet description
 */

@Override

public String getServletInfo() {

    return "Short description";
}

// </editor-fold>

private int add(int n1, int n2) {

    // Note that the injected javax.xml.ws.Service reference as well as port objects are
    // not thread safe.

    // If the calling of port operations may lead to race condition some synchronization
    // is required.

    com.add.NewWebService port = service.getNewWebServicePort();

    return port.add(n1, n2);
}

}
```

Output:

Pract5_2 - NetBeans IDE 8.2

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Projects

Start Page index.html NewWebService.java index.html addition.java

```
* @param response servlet response
* @throws ServletException if a servlet-specific error occurs
* @throws IOException if an I/O error occurs
*/
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    response.setContentType("text/html");
    try (PrintWriter out = response.getWriter()) {
        /* I
        out.
        out.
        out.
        out.
        out.
        out.println("</head>");
        out.println("<body>");
        out.println("<form>");
        out.println("Enter first number :<input type ='text' name='num1' value='0' />");
        out.println("Enter second number :<input type ='text' name='num2' value='0' />");
        out.println("<input type='submit' value='add' /> <br/>");
        out.println("<h1>Addition of two numbers is " + add(Integer.parseInt(num1), Integer.parseInt(num2)) + "</h1>");
        out.println("</body>");
    }
}
```

Set Servlet Execution URI

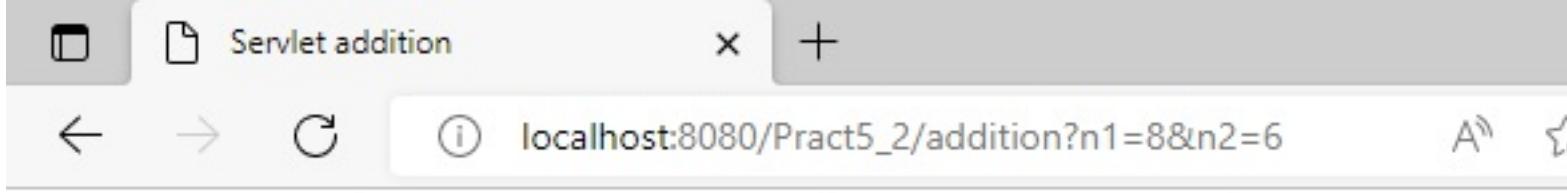
Select servlet execution URI, optionally add some request parameters :
e.g. /flowerServlet?flower=rose&color=red

/addition

OK Cancel

Output HTTP Server Monitor

File Explorer Find Replace Recent Files Task List Project Tools Help



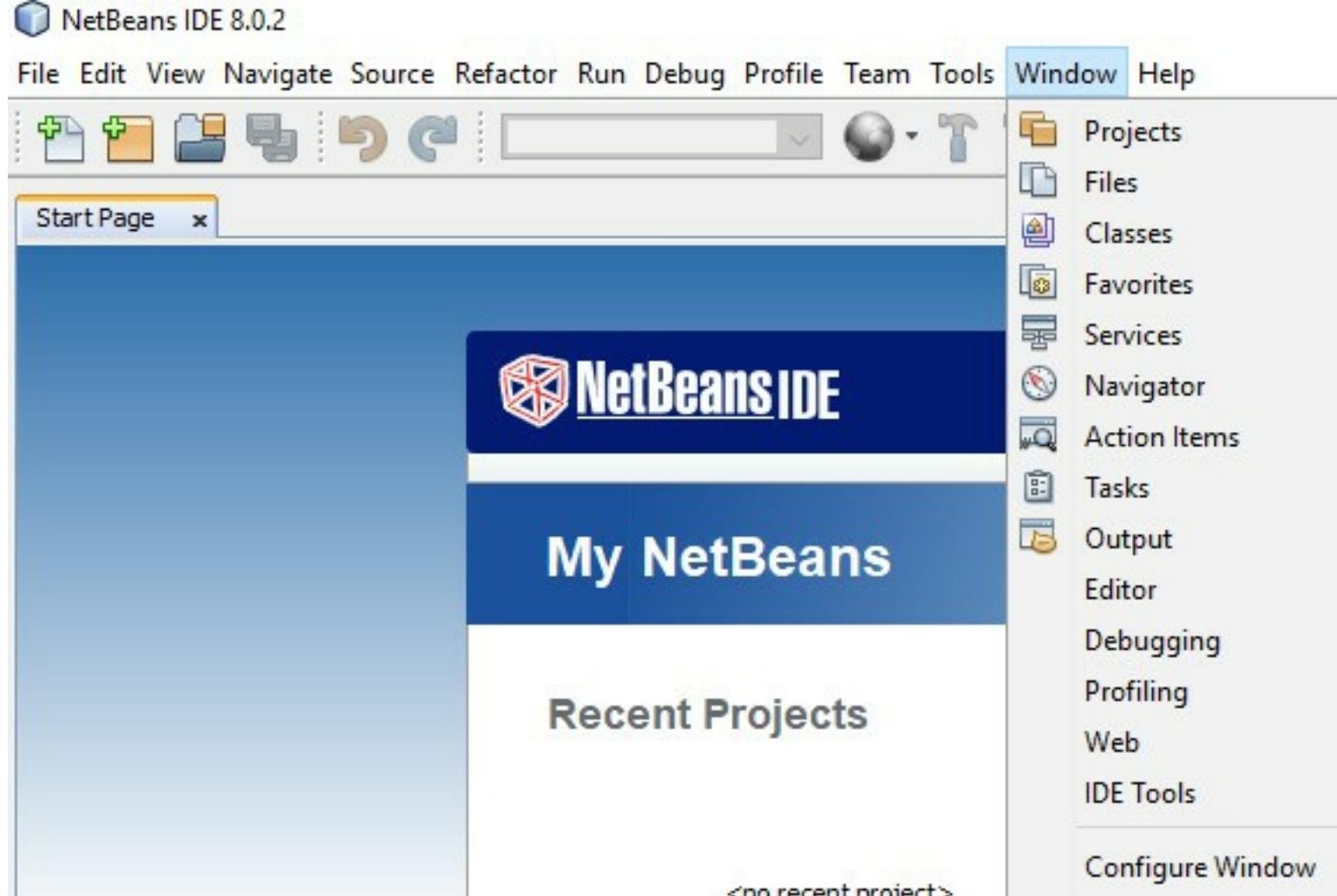
Addition of two numbers is 14



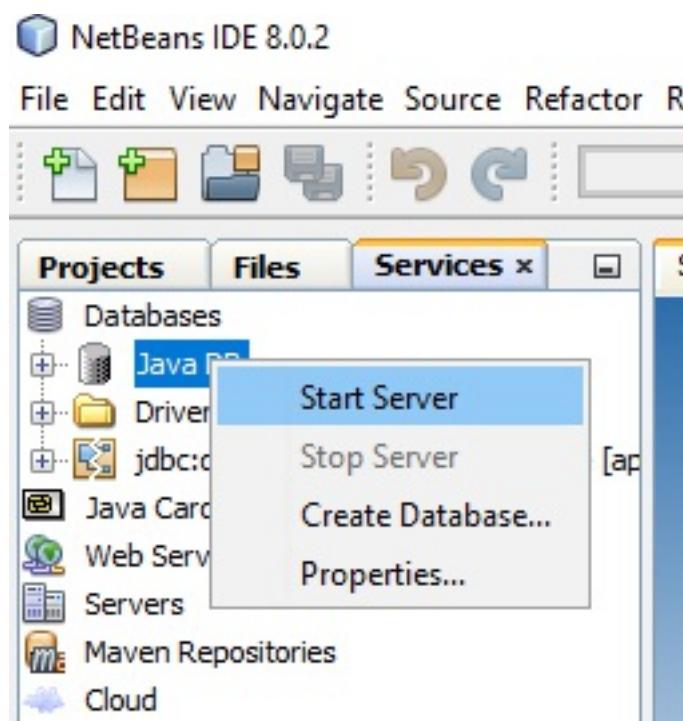
Practical 6:

Aim: Define a web service method that returns the contents of a database in a JSON string. The contents should be displayed in a tabular format.

1. Click on Window menu and click on **Projects, Files & Services** to open it.

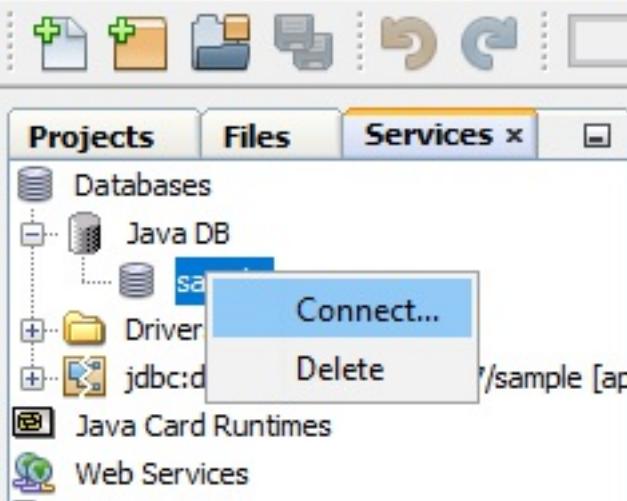


1. Right click on Java DB and then click on Start Server to start the server .



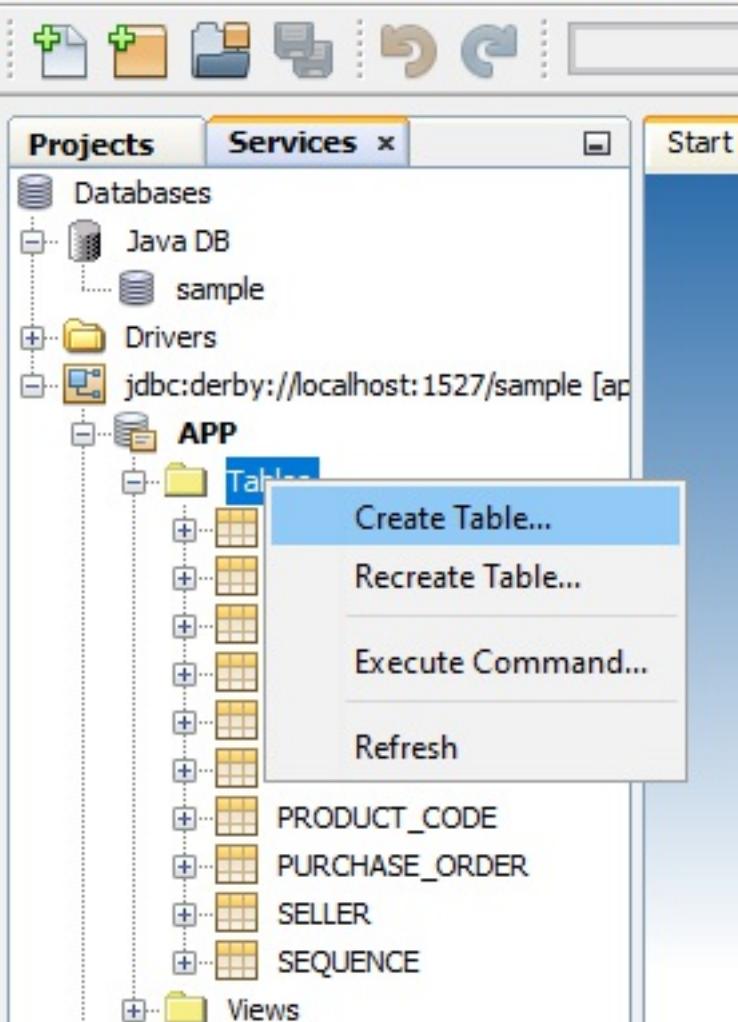
1. Now expand Java DB and right click on sample and then click on connect to connect the sample database with server.

File Edit View Navigate Source Refactor

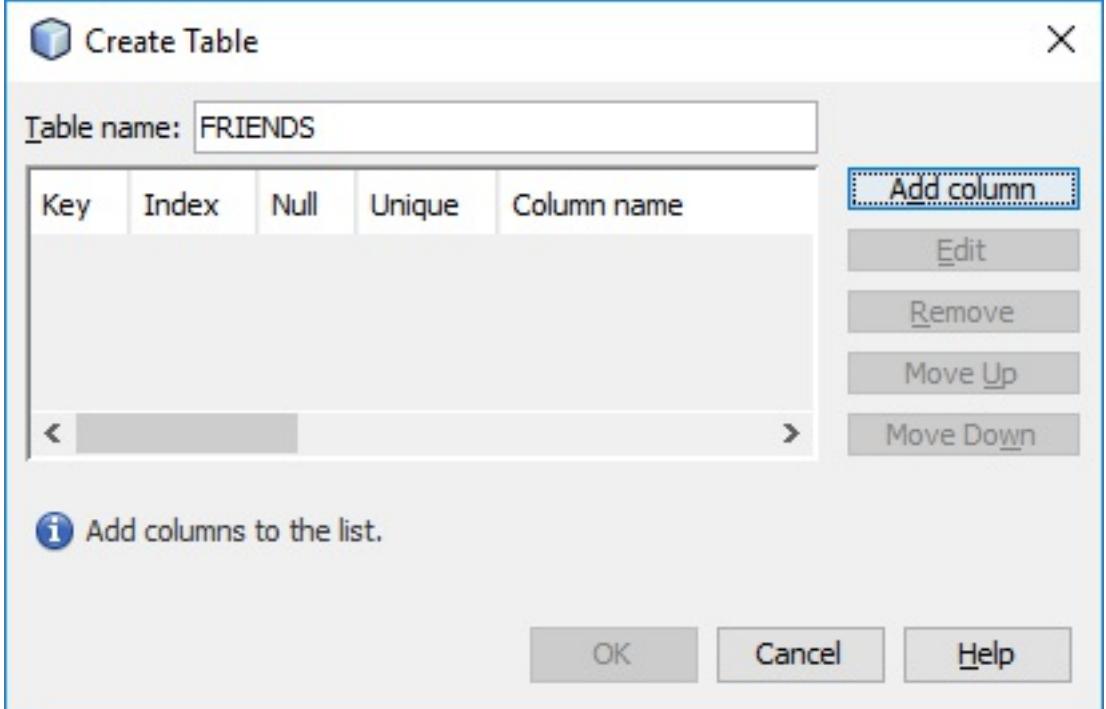


1. Now we are going to create a table in default database **sample**. Right click on Table -> **Create Table**

File Edit View Navigate Source Refactor Run

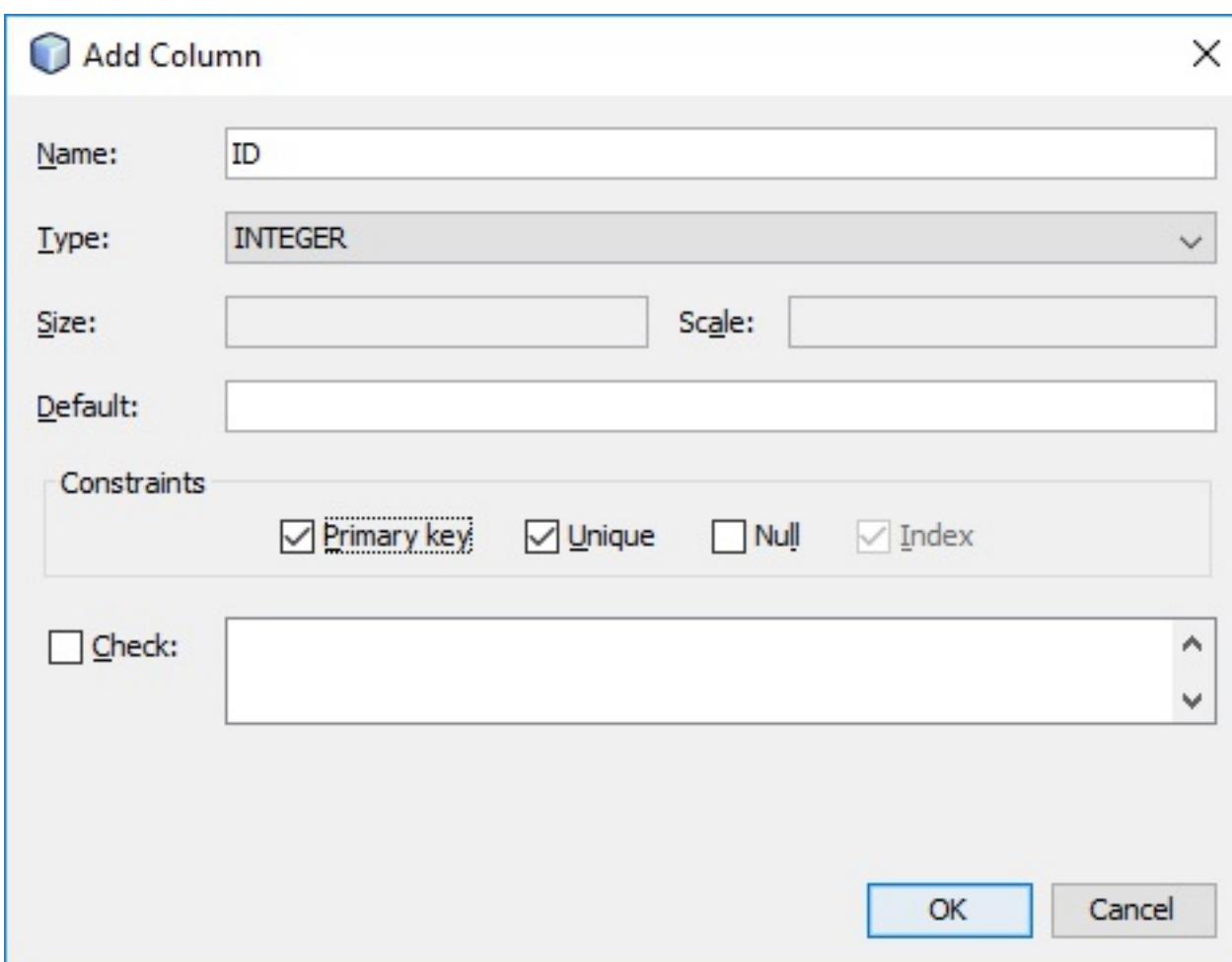


1. Give **table name as FRIENDS**.



1. Now **click on Add column button to add columns** in table.

Enter details as in below pic and **select Primary key**. After that **click on OK** button.



1. Now **add second column** with following detail. But **don't select primary** and click on OK button.

 Add Column X

Name: FIRSTNAME

Type: VARCHAR

Size: 25 Scale:

Default:

Constraints

Primary key Unique Null Index

Check:

OK Cancel

1. Now click on OK button.

 Create Table X

Table name: FRIENDS

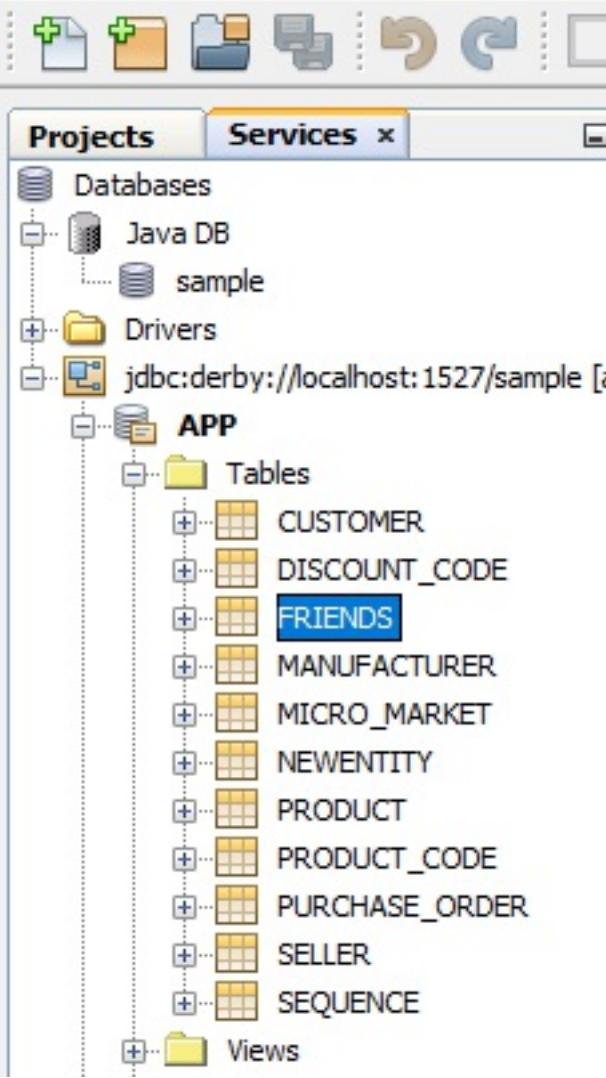
Key	Index	Null	Unique	Column name
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	ID
		<input checked="" type="checkbox"/>	<input type="checkbox"/>	FIRSTNAME

Add column Edit Remove Move Up Move Down

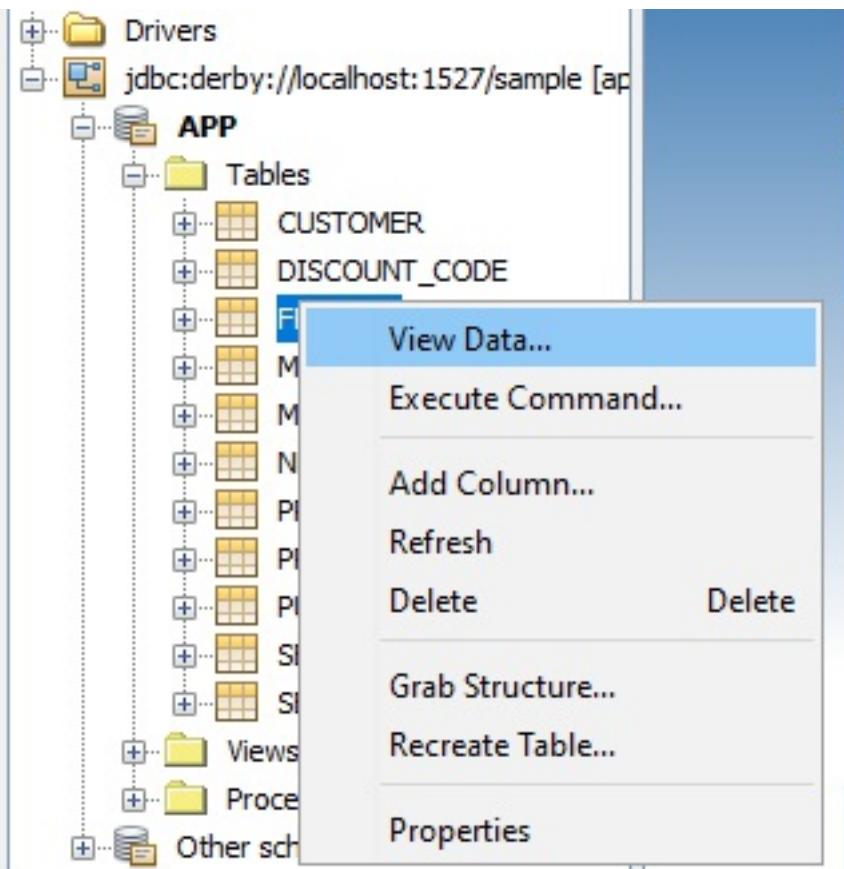
OK Cancel Help

1. Now you can see a table with name **FRIENDS** in the table.

File Edit View Navigate Source Refactor



1. Right click on FRIENDS to view and add records into it.



1. Now click on the leftmost icon in second panel to insert some record.

Projects Services

Databases Java DB sample Drivers jdbc:derby://localhost:1527/sample [app on APP] APP Tables CUSTOMER DISCOUNT_CODE FRIENDS MANUFACTURER MICRO_MARKET NEWENTITY PRODUCT PRODUCT_CODE PURCHASE_ORDER SELLER

Start Page SQL 1 [jdbc:derby://localhost:1527/sample [app on APP]]

```
Connection: jdbc:derby://localhost:1527/sample [app on APP]
1   select * from APP.FRIENDS;
2
```

select * from APP.FRIENDS

Insert Record(s) (Alt+I)

1. Insert a record and then click on Add Row button to insert more record. After that click on OK button to finish.

Insert Record(s)

Press CTRL+Tab to exit data entry mode from the table. Press CTRL+0 to set NULL value and CTRL+1 to set DEFAULT value for a given column.

#	ID	FIRSTNAME
1	1	ANAND

Show SQL Add Row Remove OK Cancel

1. As you can see, I have entered 7 records.

select * from APP.FRIENDS *

#	ID	FIRSTNAME
1		1 ANAND
2		2 JULHAS
3		3 NIKHIL
4		4 GAGAN
5		5 RAVI
6		6 DHARMENDRA
7		7 ADARSH

1. Now **create a web application** with name **Server**. After that **click on Next and then Finish** button.

New Web Application

Steps

1. Choose Project
2. **Name and Location**
3. Server and Settings
4. Frameworks

Name and Location

Project Name:

Project Location:

Project Folder:

Use Dedicated Folder for Storing Libraries

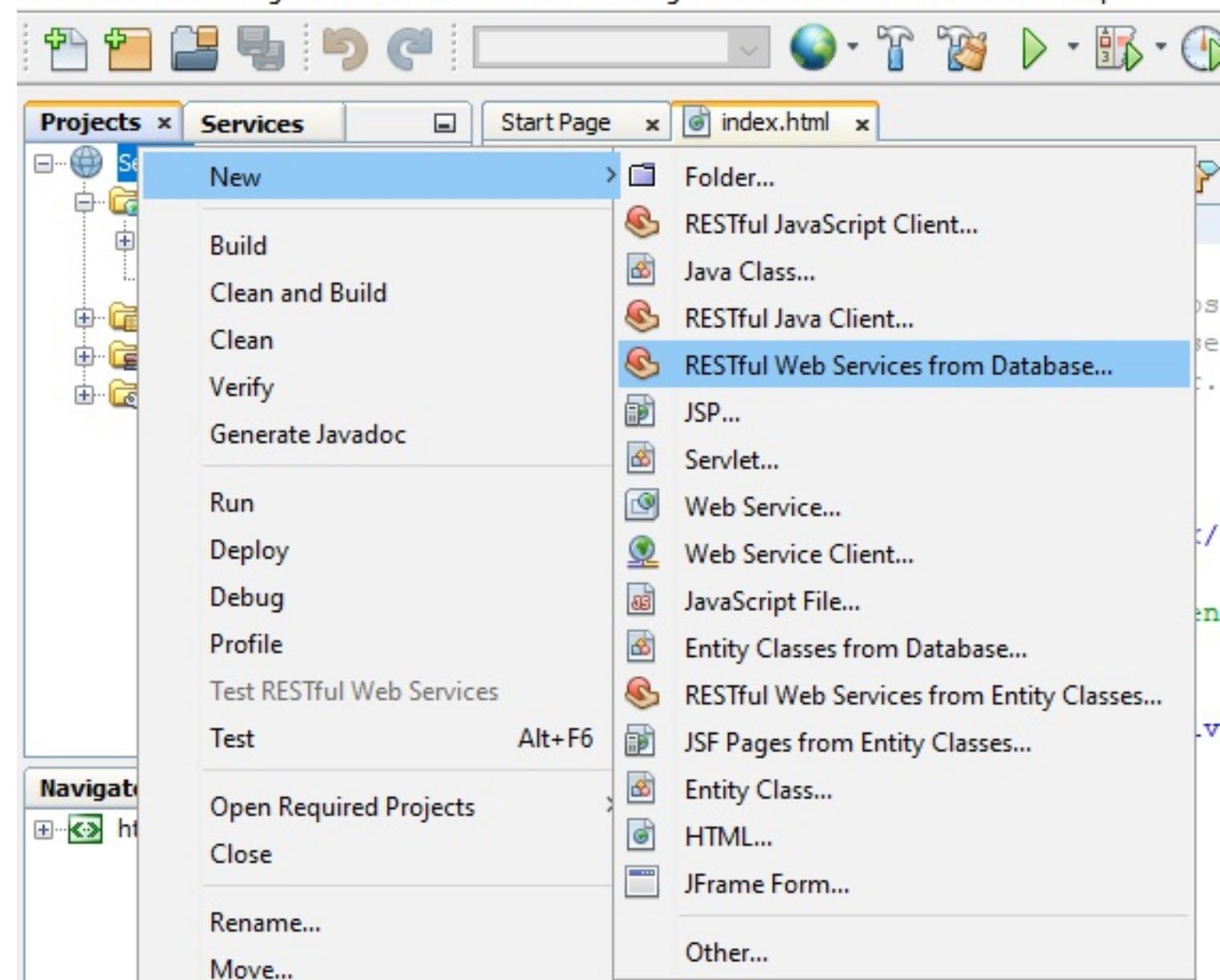
Libraries Folder:

Different users and projects can share the same compilation libraries (see Help for details).

< Back

1. Now **create a RESTful Web Service from Database by right click on project**

name.



1. Choose **Data Source jdbc/sample**.

Steps

1. Choose File Type
2. **Database Tables**
3. Entity Classes
4. Generated Classes

Database TablesData Source: `java:comp/DefaultDataSource`Available Tables: `java:comp/DefaultDataSource``jdbc/__default``jdbc/__TimerPool``jdbc/sample`

New Data Source...

[`jdbc/sample : jdbc:derby://localhost:1527/sample`][Remove](#)[Add All >>](#)[<< Remove All](#)

Any

 [Include Related Tables](#) Select the table source.[Back](#)[Next >](#)[Finish](#)

1. Now **select FRIENDS** and **click on Add button**. After that **click on Next** button.

Steps

1. Choose File Type
2. **Database Tables**
3. Entity Classes
4. Generated Classes

Database Tables

Data Source: jdbc/sample

Available Tables:

CUSTOMER
DISCOUNT_CODE
FRIENDS
MANUFACTURER
MICRO_MARKET
NEWENTITY
PRODUCT
PRODUCT_CODE
PURCHASE_ORDER
SELLER
SEQUENCE

Add >

< Remove

Add All >>

<< Remove All

Any

 Include Re Select at least one table.**< Back****Next >****Finish**

1. Enter Package name as **com.kk** and click on Next button and then Finish.

Steps

1. Choose File Type
2. Database Tables
3. **Entity Classes**
4. Generated Classes

Entity Classes

Specify the names and the location of the entity classes.

Class Names:

Database Table	Class Name	Generation Type
FRIENDS	Friends	New

Project:

Server

Location:

Source Packages

Package:

com.kk

 Generate Named Query Annotations for Persistent Fields Generate JAXB Annotations Generate MappedSuperclasses instead of Entities Create Persistence Unit

< Back

Next >

Finish

Cancel

Help

1. Now open selected file by double click on it.



Projects x Services

- Server
 - Web Pages
 - + WEB-INF
 - index.html
 - Source Packages
 - + com.kk
 - + com.kk.service
 - AbstractFacade.java
 - ApplicationConfig.java
 - FriendsFacadeREST.java
 - + Libraries
 - + Enterprise Beans

Start Page x index.html x

Source History

1 <!DOCTYPE html>
2 <!--
3 To change this license header, choose L
4 To change this template file, choose To
5 and open the template in the editor.
6 -->
7 <html>
8 <head>
9 <title>TODO supply a title</tit
10 <meta charset="UTF-8">

1. Now **remove the selected part from every method in this file**. So that it will communicate only in JSON format. You can also use methods to convert it. But this is easiest method.

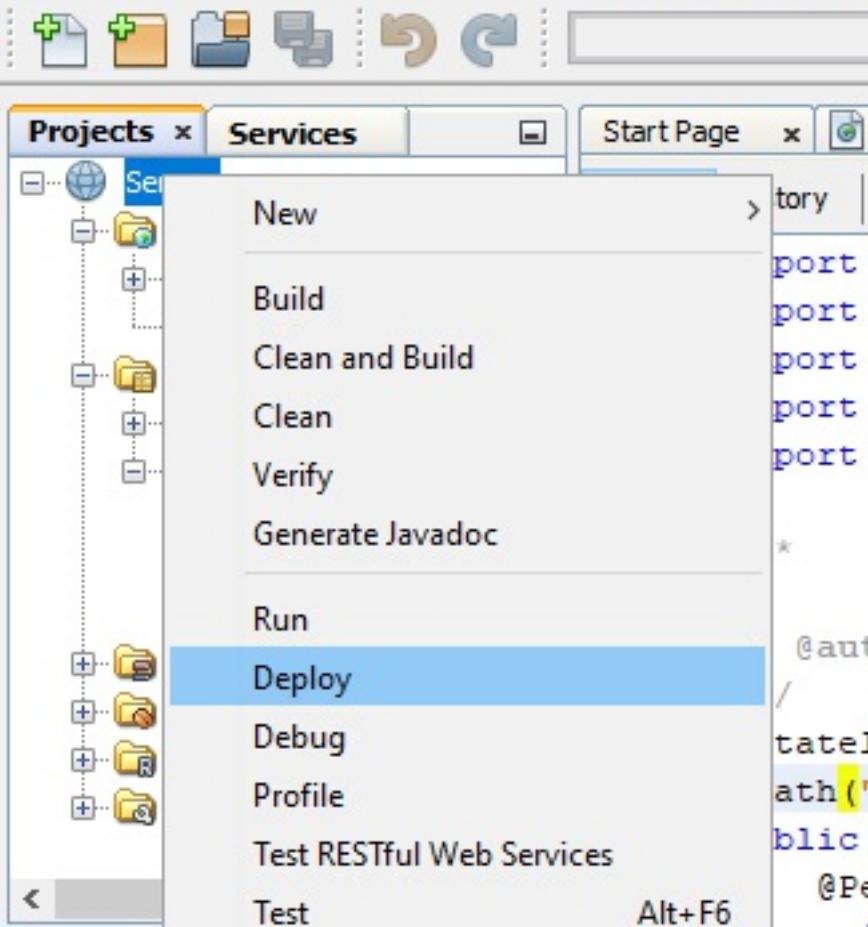
Source History



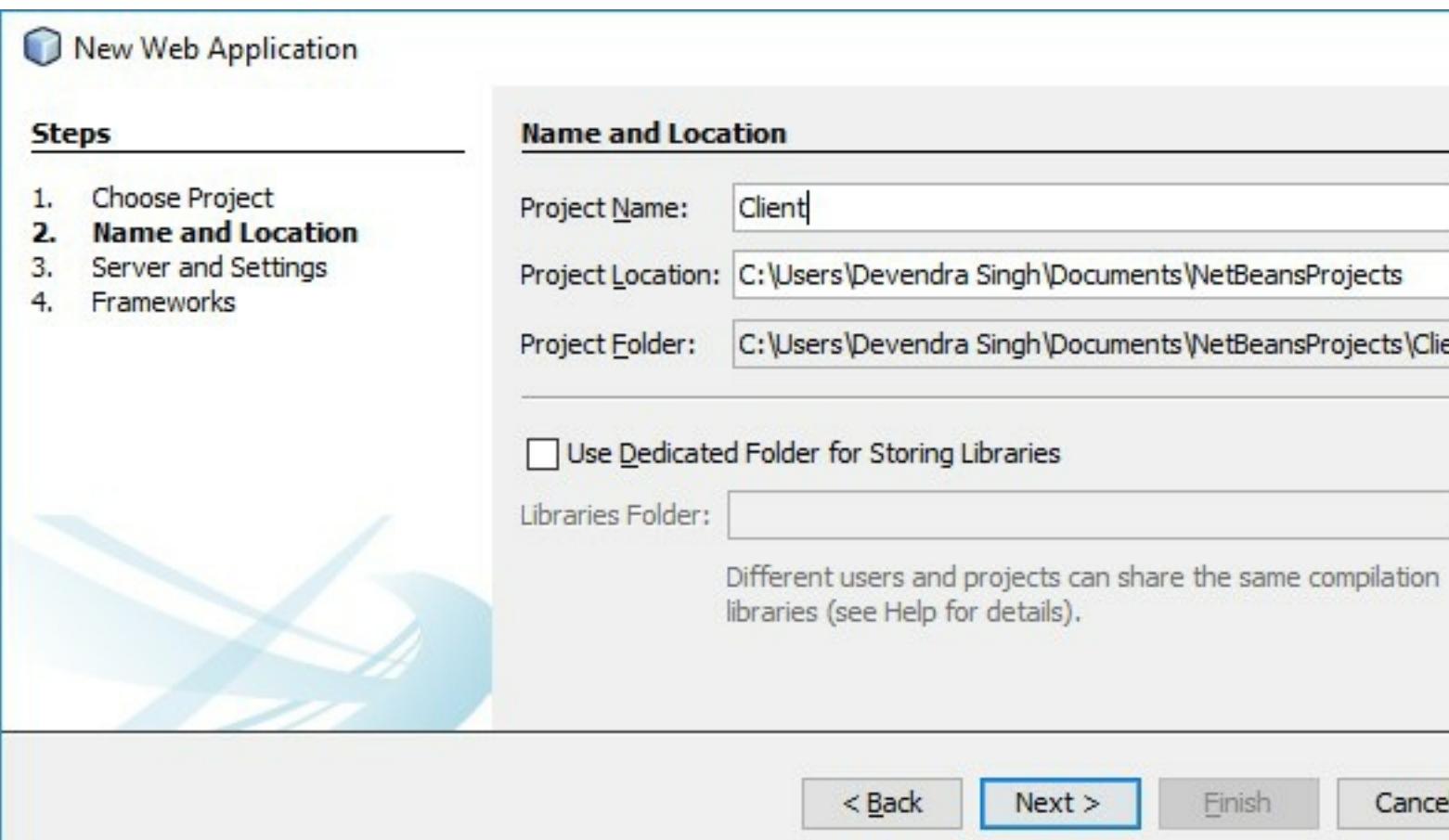
```
25  */
26  @Stateless
27  @Path("com.kk.friends")
28  public class FriendsFacadeREST extends AbstractFacade<Friends> {
29      @PersistenceContext(unitName = "ServerPU")
30      private EntityManager em;
31
32      public FriendsFacadeREST() {
33          super(Friends.class);
34      }
35
36      @POST
37      @Override
38      @Consumes({"application/xml", "application/json"})
39      public void create(Friends entity) {
40          super.create(entity);
41      }
42
43      @PUT
44      @Path("{id}")
45      @Consumes({"application/xml", "application/json"})
46      public void edit(@PathParam("id") Integer id, Friends entity) {
47          super.edit(entity);
48      }
49  }
```

1. After that right click on project name and Deploy it.

File Edit View Navigate Source Refactor Run Debug Project



1. Now **create one more Web Application** as Client. After that **click on Next** and then **Finish** button.



1. Now **open the index.html file of Client project** and add the following

code in between HEAD tag.

```
<style>
```

```
table {
```

```
font-family: arial, sans-serif; border-collapse: collapse;
```

```
}
```

```
td, th {
```

```
border: 1px solid #000000; text-align: center; padding: 8px;
```

```
}
```

```
</style>
```

```
<script>
```

```
var request = new XMLHttpRequest(); request.open('GET',
```

```
'http://localhost:8080/Server/webresources/com.kk.friends/', true);  
request.onload = function () {
```

```
// begin accessing JSON data here var data = JSON.parse(this.response);
```

```
for (var i = 0; i < data.length; i++) {
```

```
var table = document.getElementById("myTable"); var row = table.insertRow();
```

```
var cell1 = row.insertCell(0); var cell2 = row.insertCell(1); cell1.innerHTML =  
data[i].id;
```

```
cell2.innerHTML = data[i].firstname;
```

```
}
```

```
};
```

```
request.send();
```

```
</script>
```

The screenshot shows the NetBeans IDE interface with a Java web project named "Client". The "index.html" file is open in the editor. The Navigator panel displays the DOM structure of the page, including elements like "table", "td", "th", and "script". The code editor shows the following HTML and JavaScript code:

```
<html>
    <head>
        <title>TODO supply a title</title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <style>
            table {
                font-family: arial, sans-serif;
                border-collapse: collapse;
            }
            td, th {
                border: 1px solid #000;
                text-align: center;
                padding: 8px;
            }
        </style>
        <script>
            var request = new XMLHttpRequest();
            request.open('GET', 'http://localhost:8080/JSONTest/index.html');
            request.onload = function () {
                // begin accessing JSON data here
                var data = JSON.parse(this.responseText);

                for (var i = 0; i < data.length; i++) {
                    var table = document.getElementById("myTable");
                    var row = table.insertRow(i);
                    var cell1 = row.insertCell(0);
                    var cell2 = row.insertCell(1);
                    cell1.innerHTML = data[i].id;
                    cell2.innerHTML = data[i].firstname;
                }
            };
        </script>
    </head>
    <body>
        <table id="myTable">
            <tr>

```

1. Replace the content of body tag with following code.

```
<table id="myTable">
```

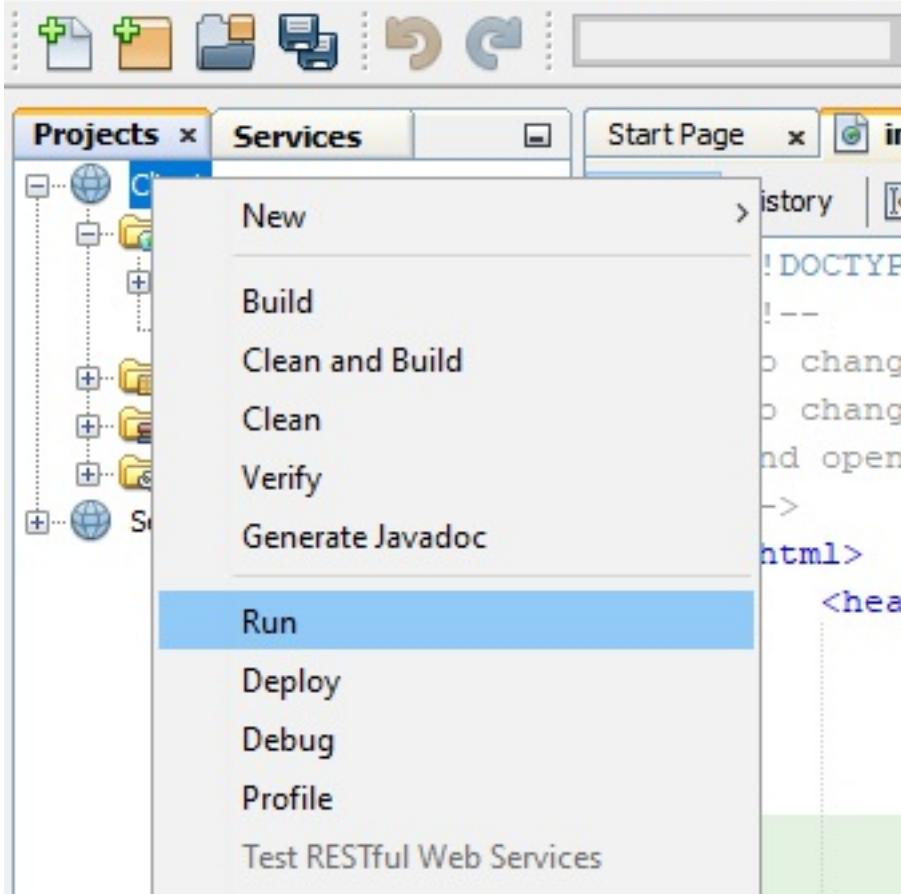
```
<tr>
```

```
<th> ID</th>  
<th> NAME</th>  
</tr>  
</table>
```

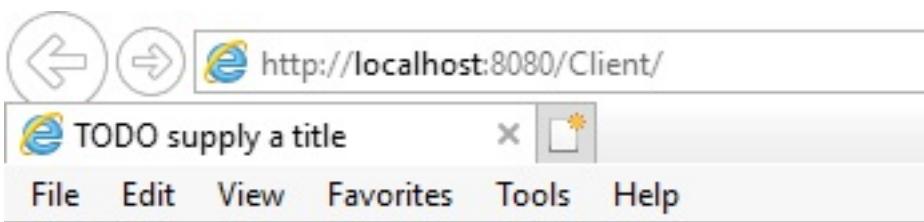
```
55         var cell1 = row.insertCell(1);  
56         cell1.innerHTML = data[i].id;  
57         cell2.innerHTML = data[i].firstname;  
58     }  
59 };  
60  
61 request.send();  
62     </script>  
63 </head>  
64 <body>  
65     <table id="myTable">  
66         <tr>  
67             <th> ID</th>  
68             <th> NAME</th>  
69         </tr>  
70     </table>  
71 </body>  
72 </html>  
73
```

1. Now **run** the **Client** Web Application.

File Edit View Navigate Source Refactor Run Debug Prof



1. A window will open in browser which represents a data in tabular format. These **data are the records entered in FRIEND table**.



Practical 7:

Aim: Define a RESTFUL web service that accepts the details to be stored in a database and performs CRUD operation.

1. Click on Window menu and click on Projects, Files & Services to open it.
2. Right click on Java DB and then click on Start Server to start the server.
3. Now expand Java DB and right click on sample and then click on connect to connect the sample database with server.
4. Now create a web application with the name CRUD_Operation.
5. Create an entity class. Right click on project name -> New -> Entity Class.
6. A window will appear.

Class Name -> Employee

Package Name -> com.emp

1. Click on Finish.



106

of 114



New File

Steps

1. Choose File Type
2. Name and Location
3. **Provider and Database**

Provider and Database

Persistence Unit Name: CRUD_OperationPU

Specify the persistence provider and database for e

Persistence Provider: EclipseLink (JPA 2.1)(defau

Data Source:

Use Java Transaction APIs

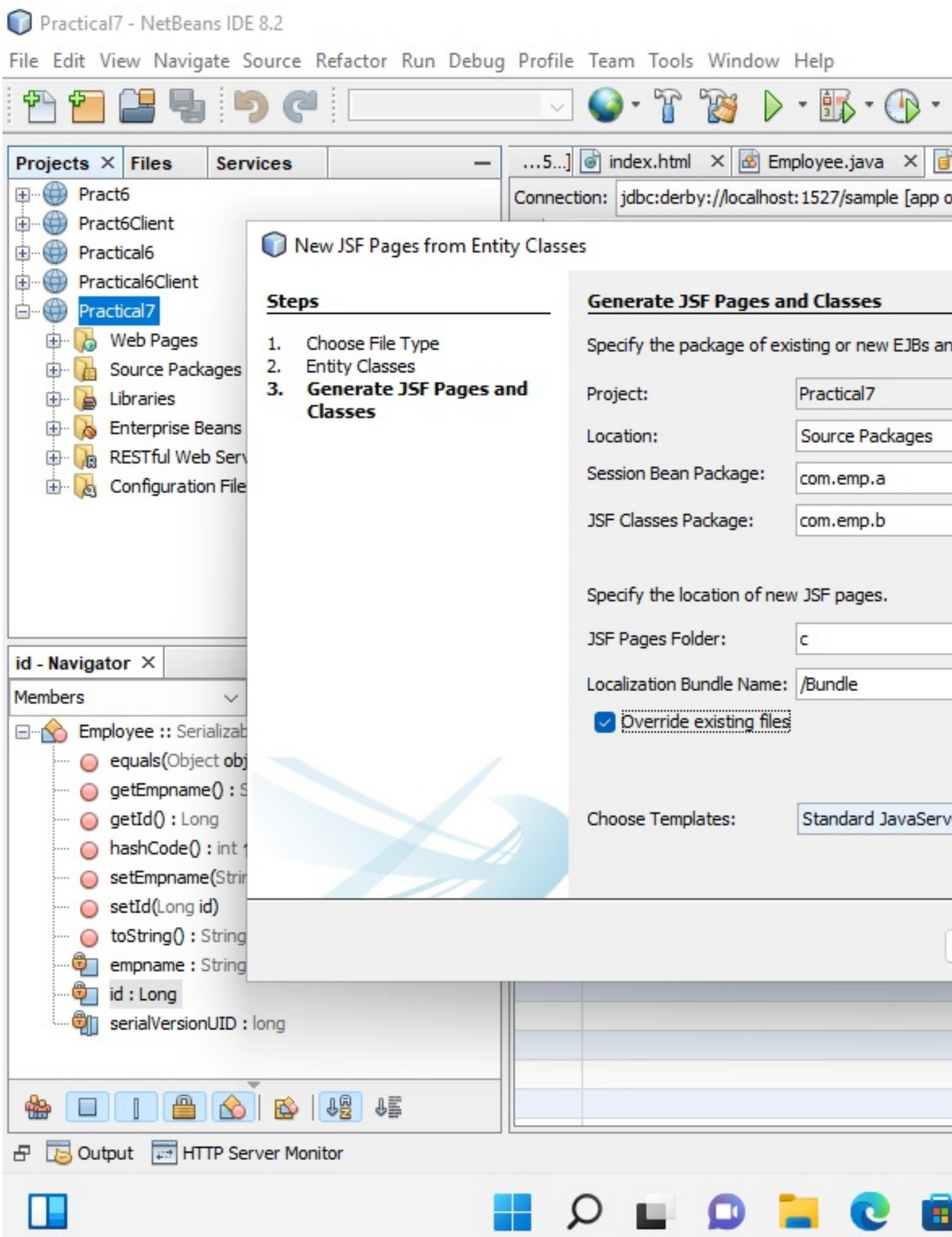
Table Generation Strategy: Create Drop an



1. Right click on project name and create JSF Pages from Entity Classes.
Right click on project name -> New -> JSF Pages from Entity Classes
2. Select com.emp.Employee and click on Add button and then Next button on

below.

3. A window like below will appear on the screen. Enter the data into that window as entered in below pic and click on Next button.



1. Now click on Finish.
2. Right click on project name and create RESTful Web Services from Entity Classes. Right click on project name -> New -> RESTful Web Services from Entity Classes.
3. Repeat step 9 and then it will go on next page. Then enter the com.emp.service in Resource Package and then click on Finish button.
4. Now open Employee.java file under com.emp package.
5. In this file at line number 24, do the right click and select Insert Code.
6. A new list will appear. Click on Add Property.
7. A new window will open. Enter name as EmpName. Click on OK button. Actually we are setting getter and setter method for EmpName.
8. Now right click on web application name and Deploy it.
9. Now right click on project name and run it.

Output:



Facelet Title



localhost:8080/Practical7/

Hello from Facelets

Show All Employee Items



List

1..3/3

Emp Name	Id	
AAA	1	View Edit Destroy
BBB	2	View Edit Destroy
CCC	3	View Edit Destroy

[Create New Employee](#)

[Index](#)



Practical 8:

Aim: Implement a typical service and typical client using WCF.

Steps:

1. Open Visual studio and click on File>New>Project
2. Select the option Visual C#>Web>WCF Service Application
3. Give name to the application as 'Practical8'.
4. Inside Solution Explorer double click on IService1.cs file.
5. Modify the IService1.cs file as follows:

```
namespace Practical8
```

```
{
```

```
[ServiceContract]
```

```
Public interface IService1
```

```
{
```

```
[OperationContract]
```

```
int Addition(int no1, int no2);
```

```
}
```

```
}
```

1. Modify the Service1.cs file as follows:

```
namespace Practical8
```

```
{
```

```
public class Service1: IService1
```

```
{
```

```
public int Addition(int no1,int no2)
```

```
{
```

```
Return no1 + no2;
```

```
}
```

```
}
```

```
}
```

1. Then right click on project name and build it.
2. After that click on Run button.
3. In browser copy the link i.e. [htto://localhost:7812/Service1.svc](http://localhost:7812/Service1.svc) and minimize the browser.

4. Create a new ASP.NET web application and name it as Practical8_Client.
5. Right click on this project and click on Add Service Reference.
6. Add the copied link into the Address and click on Go button. Then click on OK.
7. Open Default.aspx page from Practical8_Client project and go to Design view of this page.
8. Add three text boxes and one Button on this page.
9. Double click on the button to generate click event of that button and add following code.

```
void AddBtn_Click(Object sender, EventArgs e)
```

```
{
```

```
ServiceReference1.IServiceClient obj = ServiceReference1.IServiceClient();
```

```
int num1 = Convert.ToInt32(TextBox1.Text);
```

```
int num2 = Convert.ToInt32(TextBox2.Text);
```

```
TextBox3.Text = Convert.ToString(obj.Addition(num1,num2));
```

```
}
```

1. Build Practical8_Client project and Run it.

Output:

Service1 Service

You have created a service.

To test this service, you will need to create a client and run the command line with the following syntax:

```
svchost.exe http://localhost:62943/Service1.svc?wsdl
```

You can also access the service description as a single URL:

```
http://localhost:62943/Service1.svc?wsdl
```

This will generate a configuration file and a code file that you can use the generated client class to call the Service. For example:

C#

```
class Test
{
    static void Main()
    {
```



Application name

[Home](#)[About](#)

Enter first number :

Enter Second number :

Output :

Add

Learn more »

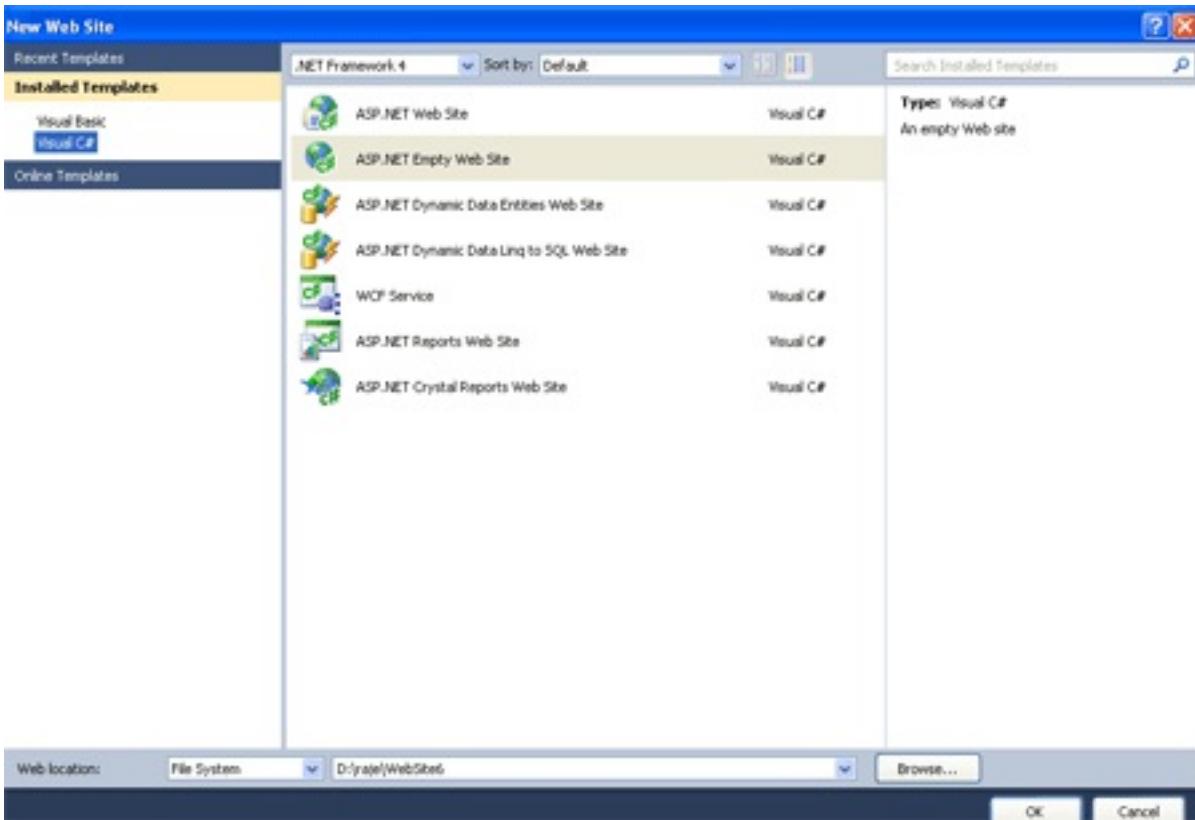


Practical 9:

Aim: Use WCF to create a basic ASP.NET AJAX service.

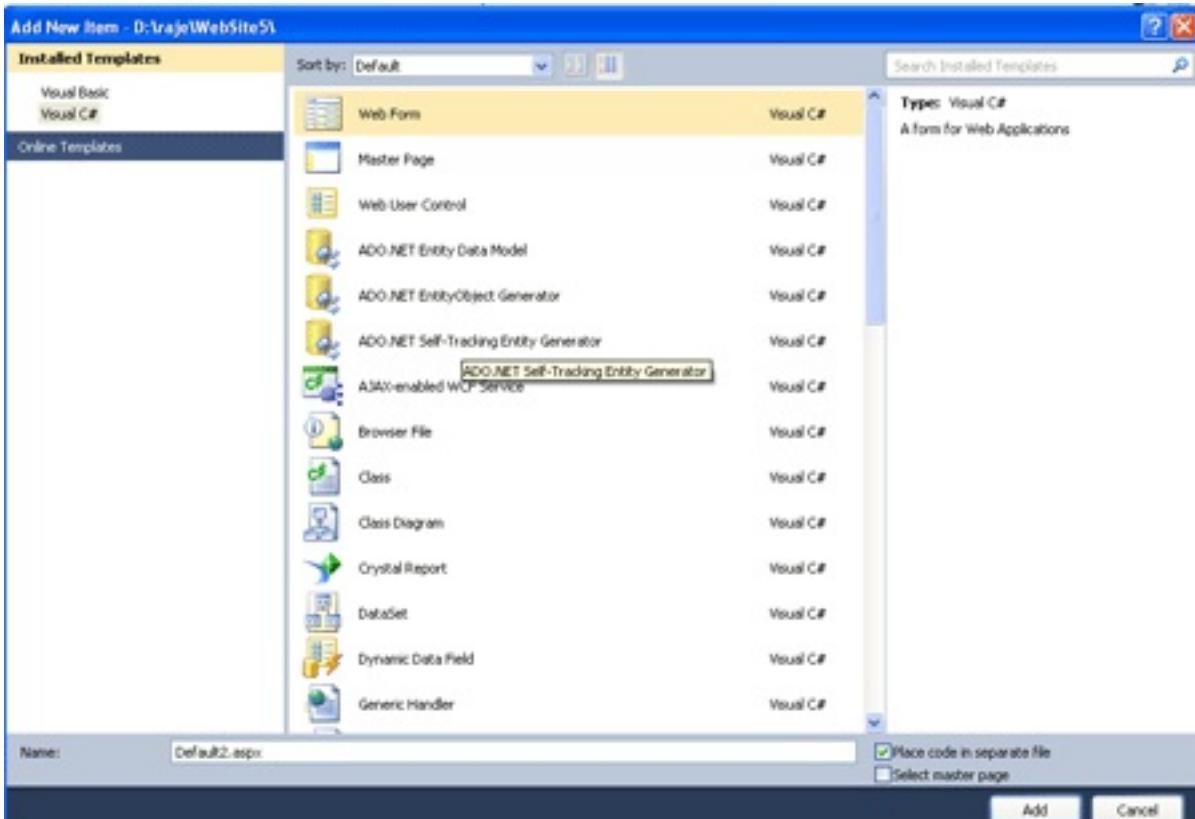
Step 1 : Open Visual Studio 2010.

- Go to File->New->WebSite
- Select ASP.NET Empty WebSite



Step 2 : Go to Solution Explorer and right-click.

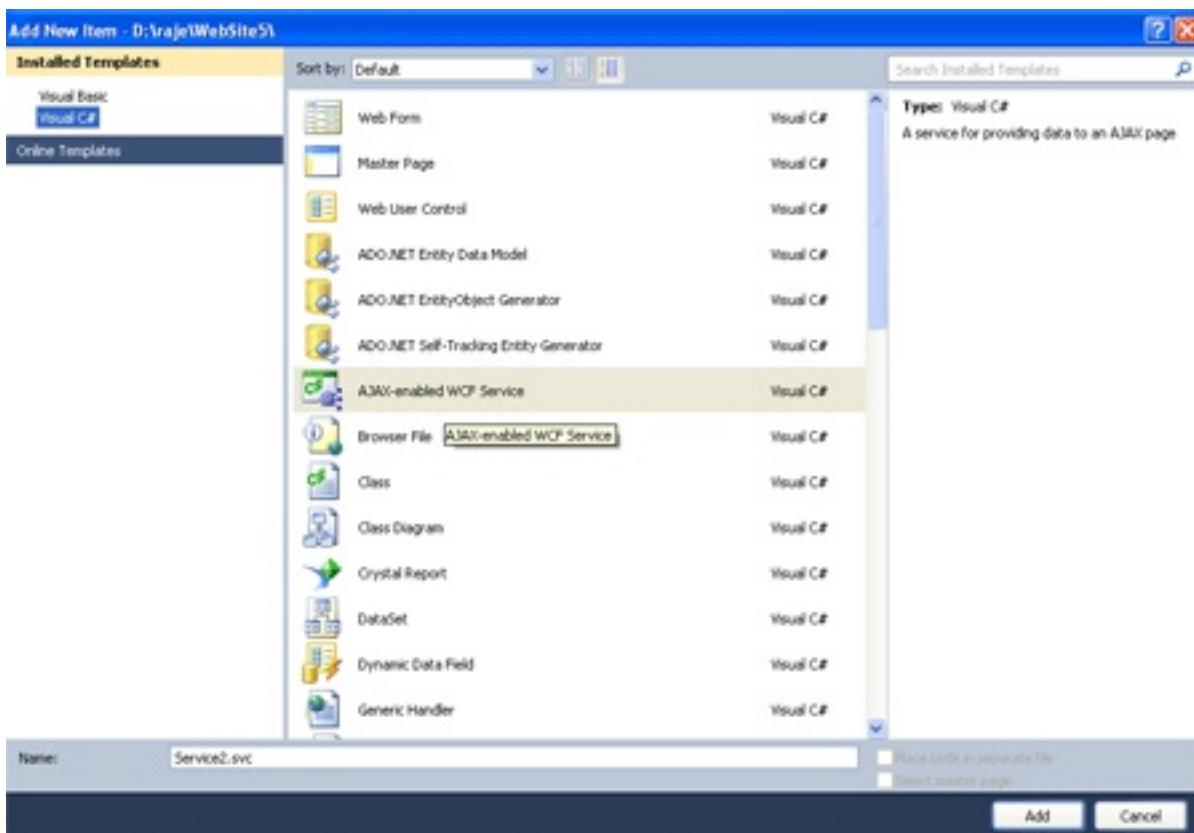
- Select Add->New Item
- Select WebForm
- Default.aspx page open



Define WCF Service :

Step 3 : Go to Solution Explorer and right-click.

- Select Add->New Item
- Select AJAX Enabled WCF Service
- Define the Namespace



Namespace :

```
[ServiceContract(Namespace = "MCNSOLUTION")]
[AspNetCompatibilityRequirements(RequirementsMode
= AspNetCompatibilityRequirementsMode.Allowed)]
public class Service
{
    // To use HTTP GET, add [WebGet] attribute. (Default ResponseFormat is
    WebMessageFormat.Json)
    // To create an operation that returns XML,
    // add [WebGet(ResponseFormat=WebMessageFormat.Xml)],
    // and include the following line in the operation body:
    //     WebOperationContext.Current.OutgoingResponse.ContentType =
    "text/xml";
    [OperationContract]
    public void DoWork()
    {
        // Add your operation implementation here
        return;
    }
}
```

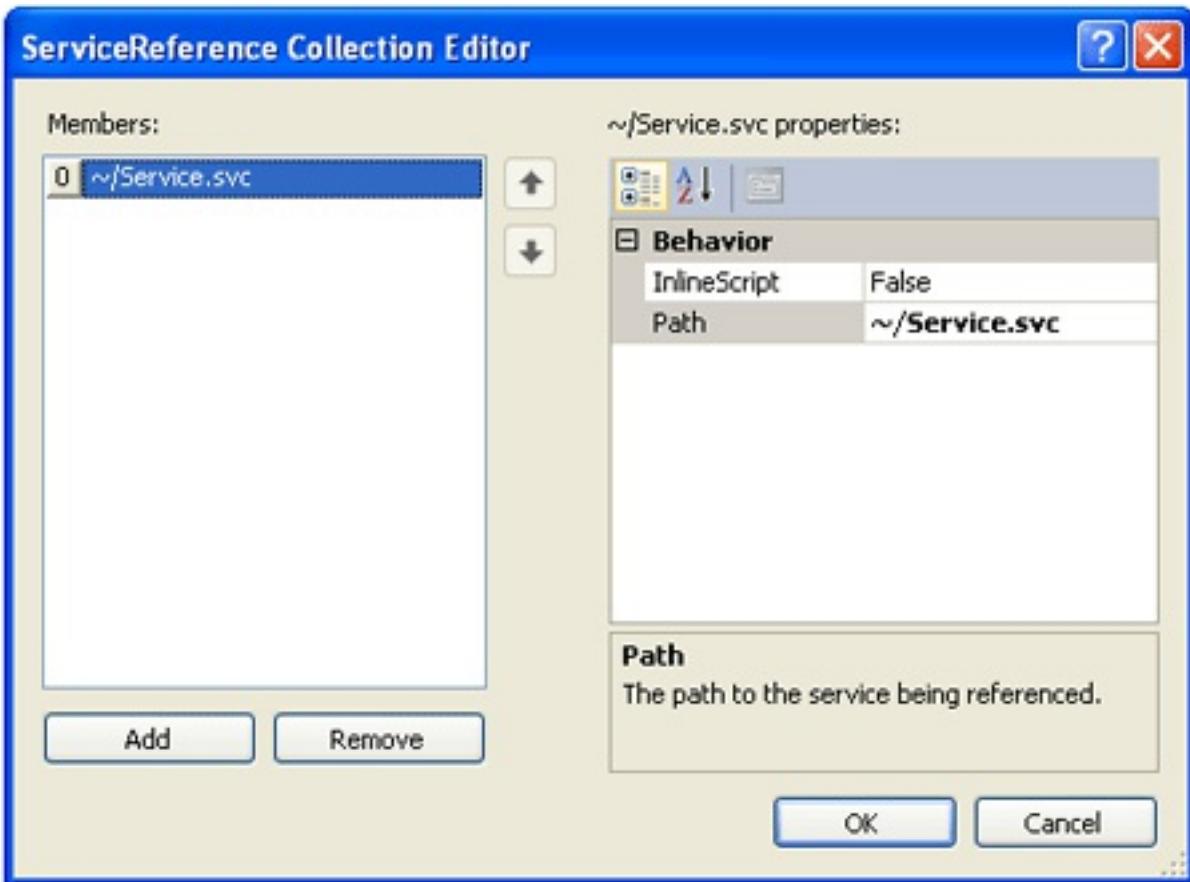
Step 4 : Open Default.aspx page and click in [Design option].

- Drag and drop Scriptmanager Control

Define Service Path :

Step 5 : In [Design] page right-click in Scriptmanager Control

- Select Properties option
- Define Services
- Click the Add option and select Service Path



Step 6 : Now add a Html Button, an HTML Input box and <Div> to the Default.aspx page. After renaming the controls and adding the onClick event of the button, the markup will appear similar to the following:

```
<div id ="dispService">
    <input id="btnCallWCF" type="button" value="MCN User" onclick="return
    btnCallWCF_onclick()" />
    <input id="txtUNm" type="text" />
```

Java Script code :

Step 7 : Now go to Default.aspx page and select Design option.

- Click in the input Button
- Define Java Script code for the on click event

Code :

```
<script language="javascript" type="text/javascript">
    function btnCallWCF_onclick() {
    }
</script>
</head>
```

Step 8 : Now call the WebService then add the namespace to the service called "MCN Solution".

Code :

```
script language="javascript" type="text/javascript">
    function btnCallWCF_onclick()
    {
        var MCN = new MCNSOLUTION.Service();
        MCN.MCNUSER($get("txtUNm").Value, OnServiceComplete, onerror);
    }
    function OnServiceComplete(result) {
        $get("dispService").innerHTML = result;
    }
    function onerror(result) {
        alert(result.get_message());
    }
</script>
```

Add Method :

Step 9 : Now we add a method called mcnuser which take a user name and return string value.

Method :

```
// Add more operations here and mark them with [OperationContract]
[OperationContract]
public string MCNUSER(string UNAME)
{
    return "HELLOMCN" + UNAME;
}
```

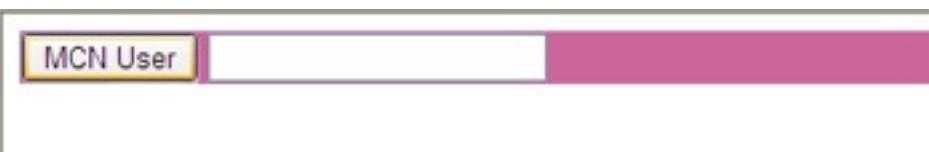
Step 10 : Now the complete code for calling the WCF Service in an AJAX Enabled WebService.

Code :

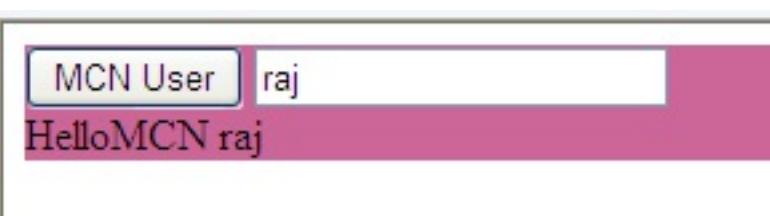
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.ServiceModel.Activation;
using System.ServiceModel.Web;
using System.Text;
[ServiceContract(Namespace = "MCNSOLUTION")]
[AspNetCompatibilityRequirements(RequirementsMode
= AspNetCompatibilityRequirementsMode.Allowed)]
public class Service
```

```
{
    // To use HTTP GET, add [WebGet] attribute. (Default ResponseFormat is
    WebMessageFormat.Json)
    // To create an operation that returns XML,
    // add [WebGet(ResponseFormat=WebMessageFormat.Xml)],
    // and include the following line in the operation body:
    //     WebOperationContext.Current.OutgoingResponse.ContentType =
    "text/xml";
    [OperationContract]}
public void DoWork()
{
    // Add your operation implementation here
    return;
}
// Add more operations here and mark them with [OperationContract]
[OperationContract]
public string MCNUSER(string UNAME)
{
    return "HELLOMCN" + UNAME;
}
}
```

Step 11: Now run the application by Pressing F5.



Now click on the MCN USER Button and see the following output.



Practical 10:

Aim: Demonstrate using the Binding Attribute of an Endpoint Element in WCF.

Steps:

1. Open Visual studio.
2. File>New>Project.
3. Select WCF inside C# and WCF class Library.
4. Give name as 'myWCFService'.
5. Now open App.Config file and write following code.

App.config file

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

```
<configuration>

<system.web>
<compilation debug="true" />
</system.web>

<system.serviceModel>

<behaviors>
<serviceBehaviors>
<behavior name="NewBehaviour">
<serviceMetadata />
<serviceDebug includeExceptionDetailInFaults="False" />
</behavior>
</serviceBehaviors>
</behaviors>

<services>
<service name="myWcfService.Service1"
behaviorConfiguration="NewBehaviour">
<endpoint address="" binding="netTcpBinding" bindingConfiguration=""
name="tcp" contract="myWcfService.IService1"/>
<endpoint address ="mex" binding="mexTcpBinding" name="mextcp"
contract="IMetadataExchange"/>
<host>
<baseAddresses>
<add baseAddress="net.tcp://localhost:8081/SampleSrc"/>
</baseAddresses>
</host>
</service>
</services>
</system.serviceModel>
```

</configuration>

1. Write click on project name and select Clean then Build.
2. Run the project.
3. WCF Test client.
4. Click on getData().
5. Enter any value in Request section and click on invoke.
6. Then you will receive value in Response section.

Output:

Process: [16616] WcfSvcHost.exe

Service1.cs

WCF Test Client

File Tools Help

... My Service Projects

Start Page

To add a service:
1. Select "Add Service" from t
2. Enter the service metadata

To test a service operation:
1. Double click the operation y
2. A new tab page will appear

Adding...



112 %

N

Autos

Adding service ...

Search (Ctrl+E)



< > Search Depth:

Name

Name

Value

Type

Autos Locals Watch 1

Call St... Break... Except... Com... Imm...

Ready



Process: [16616] WcfSvcHost.exe

Service1.cs

WCF Test Client

File Tools Help

My Service Projects

- net.tcp://localhost:8081/SampleSrc/mex
 - IService1 (tcp)
 - GetData()
 - GetDataAsync()
 - GetDataUsingDataContract()
 - GetDataUsingDataContractAsync
 - Config File

Start Page

To add a service:
1. Select "Add Service" from t
2. Enter the service metadata

To test a service operation:
1. Double click the operation y
2. A new tab page will appear
3. Enter the value of parameter
4. Click "Invoke" button

112 %

Autos

Service added successfully.

Search (Ctrl+E)



Search Depth:

Name

Name

Value

Type

Autos Locals Watch 1

Call St... Break... Except... Com... Imm...

Ready



Process: [16616] WcfSvcHost.exe

Service1.cs

WCF Test Client

File Tools Help

My Service Projects

net.tcp://localhost:8081/SampleSrc/mex

IService1 (tcp)

GetData()

GetDataAsync()

GetDataUsingDataContract()

GetDataUsingDataContractAsyn...

Config File

GetData

Request

Name

value

5

Security Warning

You are about to send information over the network. Insecure bindings or malicious services could allow private information to be viewed by others. Do not try to invoke services you do not trust, and use a secure binding if you are transmitting information you do not want others to see.

 In the future, do not show this message.

Formatted XML

Autos

Service added successfully.

Search (Ctrl+E)



Search Depth:

Name

Name

Value

Type

Autos Locals Watch 1

Call St... Break... Except... Com... Imm...

Ready



Process: [16616] WcfSvcHost.exe

Service1.cs

WCF Test Client

File Tools Help

My Service Projects	
net.tcp://localhost:8081/SampleSrc/mex	
IService1 (tcp)	
GetData()	
GetDataAsync()	
GetDataUsingDataContract()	
GetDataUsingDataContractAsync	
Config File	

GetData

Request

Name

value

Value

5

Response

Name

(return)

Value

"You

Formatted XML

Autos

Service invocation completed.

Search (Ctrl+E)



Search Depth:

Name

Name

Value

Type

Autos Locals Watch 1

Call St... Break... Except... Com... Imm...

Ready

