# Web Services

krunal dhavle
TYCS-713

Name : Krunal Dhavle

Class : TYCS

Roll No : 713

Subject : Web Services Practicals

Teacher : Karishma Jain Maam

# INDEX


| SR NO | DATE | PRACTICAL AIM | PG. NO | REMARK |
|---|---|---|---|---|
| 1 | 19/08/2020 | Write a program to implement to create a simple web service that converts the temperature from Fahrenheit to Celsius and vice versa. | 1-7 | |
| 2 | 26/08/2020 | Write a program to implement the operation can receive request and will return a response in two ways. a) One - Way operation b) Request –Response. | 8-14 | |
| 3 | 30/09/2020 | Demonstrate using the binding attribute of an endpoint element in WCF. | 15-22 | |
| 4 | 07/10/2020 | Use WCF to create a basic ASP.NET Asynchronous JavaScript and XML (AJAX) service | 23-26 | |
| 5 | 13/10/2020 | Implement a typical service and a typical client using WCF | 27-33 | |
| 6 | 14/10/2020 | Develop client which consumes web services developed in different platform. | 33-38 | |
| 7 | 21/10/2020 | 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. | 39-45 | |
| 8 | 28/10/2020 | Define a RESTful web service that accepts the details to be stored in a database and performs CRUD operation. | 46-57 | |
| 9 | 18/11/2020 | Write a JAX-WS web service to perform the following operations. Define a Servlet / JSP that consumes the web service. | 58-64 | |

Date: 19/08/2020

## Practical no 1

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

**Steps:**

1. Create a Web Application of ”ASP.NET” using(.Net Framework 4.7).
2. Give a suitable title to the project and solution.
3. Initialize it as ”Empty” solution.
4. On Solution Explorer, right click and add a ”Web Service(asmx)” to the solution.
5. Moving ahead we already have the files initialized for returning ”Hello World”.
6. Remove/overwrite the ”Hello World” ”WebMethod” and add your own ”WebMethods” to the source file.
7. Save it and try out the web-service using the play button to host the web service on a local ”IIS Express” server.
8. Once the web service successfully runs and gives the desired output in form of XML documents; We will Proceed towards making client side pages.
9. In solution explorer, right click on the connected services tab and click on ”add service reference” option.
10. Next discover the web service we just created and click on the service you created and click ok.
11. Now you have successfully connected the web service to the solution.
12. Add a new WebForm in the same solution and start designing the client side UI.
13. After completing the UI design open the backend C# code.
14. Define methods for Button Clicks either explicitly or by double clicking the respective buttons in the design section of the form .
15. In the button click methods first create a SOAP object for the ”webservice.WebService1SoapClient()” class.
16. Using the SOAP object invoke the web-service methods and pass the appropriate data from the input field casted to the data type used in web-service method.

17. Next try running the ASPX page using the local server, add exception handling for handling exceptions if required.

**Code:**

**WebService1.asmx.cs**

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Services;

namespace Temp
{

    [WebService(Namespace = "http://tempuri.org/")]
    [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
    [System.ComponentModel.ToolboxItem(false)]

    public class WebService1 : System.Web.Services.WebService
    {

        [WebMethod]
        public double celsius_to_farhenheit(double celsius)
        {
            return ((celsius * 9 / 5) + 32);
        }
        [WebMethod]
        public double farhenheit_to_celsius(double farhenheit)
        {
            return ((farhenheit - 32) * 5 / 9);

        }
    }
}
```

**WebForm1.aspx:**

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs"
Inherits="Temp.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="Input Temperature"></asp:Label>
           

         <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>&deg;
         <asp:DropDownList ID="DropDownList1" runat="server">

            <asp:ListItem>Celsius</asp:ListItem>
             <asp:ListItem>Farhenheit</asp:ListItem>

         </asp:DropDownList>

         <br/><br/>

         <asp:Button ID="Button1" runat="server" Text="Convert_to_Farhenhiet"
OnClick="Button1_Click"/>
             
         <asp:Button ID="Button2" runat="server" Text="Convert_to_Celsius"
OnClick="Button2_Click" />
         <br/><br/>

         <asp:Label ID="Label3" runat="server" Text="Result :"></asp:Label>
          
         <asp:Label ID="Label2" runat="server" Text=""></asp:Label>
          <asp:Label ID="Label4" runat="server" Text=""></asp:Label>
</div>
   </form>
</body>
</html>
```

**WebForm1.aspx.cs**

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

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

      }
      protected void Button1_Click(object sender, EventArgs e)
      {

         double result;
         try
         {
            ServiceReference1.WebService1SoapClient client = new
ServiceReference1.WebService1SoapClient();

            result = client.celsius_to_farhenheit(Convert.ToDouble(TextBox1.Text));

            if (DropDownList1.SelectedValue.Equals("Celsius"))
            {
               Label2.Text = result.ToString();
               Label4.Text = "&deg;Farhenheit";
            }
            if (DropDownList1.SelectedValue.Equals("Farhenheit"))
            {
               Label2.Text = "Already in Farhenheit";
               Label4.Text = "";
            }
         }
         catch (System.FormatException)
         {
            Label2.Text = "Invalid Inputs";
            Label4.Text = "";
         }

      }
```

```
    protected void Button2_Click(object sender, EventArgs e)
    {
        double result;
        try
        {
            ServiceReference1.WebService1SoapClient client = new
ServiceReference1.WebService1SoapClient();
            result = client.farhenheit_to_celsius((Convert.ToDouble(TextBox1.Text)));
            if (DropDownList1.SelectedValue.Equals("Celsius"))
            {
                Label2.Text = "Already in Celsius";
                Label4.Text = "";
            }
            if (DropDownList1.SelectedValue.Equals("Farhenheit"))
            {
                Label2.Text = result.ToString();
                Label4.Text = "&deg;celsius";
            }
        }
        catch (System.FormatException)
        {
            Label2.Text = "Invalid Inputs";
            Label4.Text = "";

        }
    }
  }
}
```

**Outputs:**

# WebService1

The following operations are supported. For a formal definition, please review the Service Description.

- celsius_to_farhenheit
- farhenheit_to_celsius

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

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

Each XML Web service needs a unique namespace in order for client applications to distinguish it from other services on the Web. http://tempuri.org/ is available for XML Web services that are under development, but published 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 your company's Internet domain name as part of the namespace. Although many XML Web service namespaces look like URLs, they need not point to actual resources on the Web. (XML Web service namespaces are URIs.)

For XML Web services creating using ASP.NET, the default namespace can be changed using the WebService attribute's Namespace property. The WebService attribute is an attribute applied to the class that contains the XML Web 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.

# WebService1

Click here for a complete list of operations.

## celsius_to_farhenheit

**Test**

To test the operation using the HTTP POST protocol, click the 'Invoke' button.

| Parameter | Value |
|---|---|
| celsius: | |

Invoke

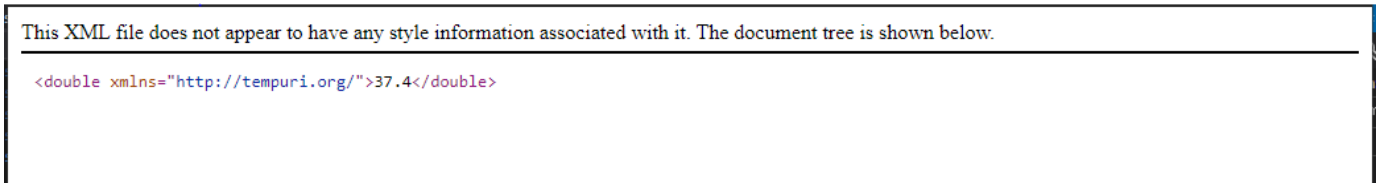
**SOAP 1.1**

The following is a sample SOAP 1.1 request and response. The placeholders shown need to be replaced with actual values.

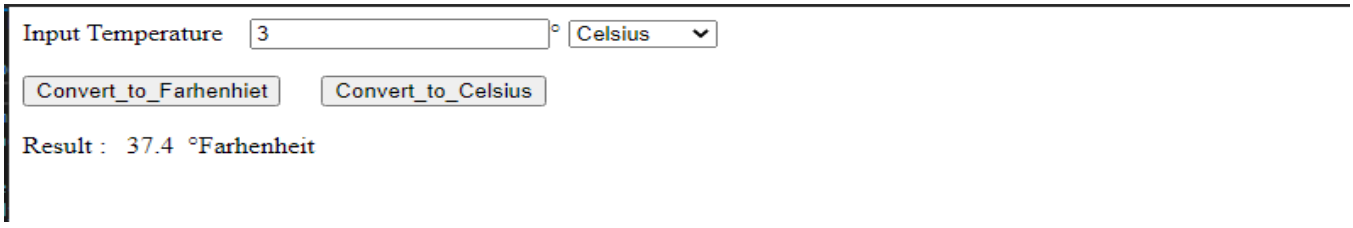
```
POST /WebService1.asmx HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://tempuri.org/celsius_to_farhenheit"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/envel
  <soap:Body>
    <celsius_to_farhenheit xmlns="http://tempuri.org/">
      <celsius>double</celsius>
    </celsius_to_farhenheit>
  </soap:Body>
</soap:Envelope>
```

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/envel
```

Input Temperature
3
°
Celsius
Convert_to_Farhenhiet
Convert_to_Celsius
Result :


This XML file does not appear to have any style information associated with it. The document tree is shown below.
<double xmlns="http://tempuri.org/">37.4</double>

Input Temperature
3
°
Celsius
Convert_to_Farhenhiet
Convert_to_Celsius
Result :


Input Temperature
3
°
Celsius
Convert_to_Farhenhiet
Convert_to_Celsius
Result : 37.4 °Farhenheit

Input Temperature
0
°
Farhenheit
Convert_to_Farhenhiet
Convert_to_Celsius
Result : -17.777777777778 °celsius

Date: 26/08/2020

## Practical no 2

**AIM:** Write a program to implement the operation can receive request and will return a response in two ways. a) One - Way operation b) Request –Response

### A)One-Way

**1. WebForm.aspx file**

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs" Inher-
its="prac2a.WebForm1" %>

<!DOCTYPE html>

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

**2. WebForm.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 prac2a
{
   public partial class WebForm1 : System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {

      }
      protected void Button1_Click(object sender, EventArgs e)
      {
         //Time show before the service is calling
         lblPagedate.Text = "on the load time the time is " + DateTime.Now.ToString();
         ServiceReference1.Service1Client sc = new ServiceReference1.Service1Client();
         sc.OneWayMessage();
         //after service is calling that time show
         lblServicedate.Text = "After Calling the service the time is " + DateTime.Now.ToString();
      }
   }
}
```

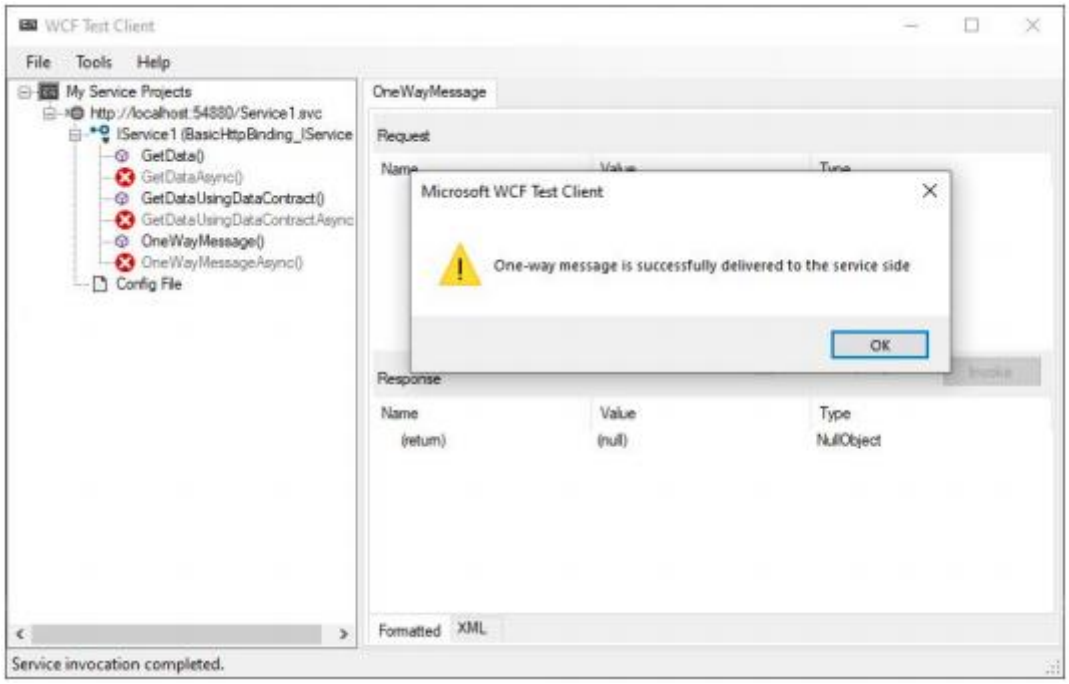
**3. IService.cs:-**

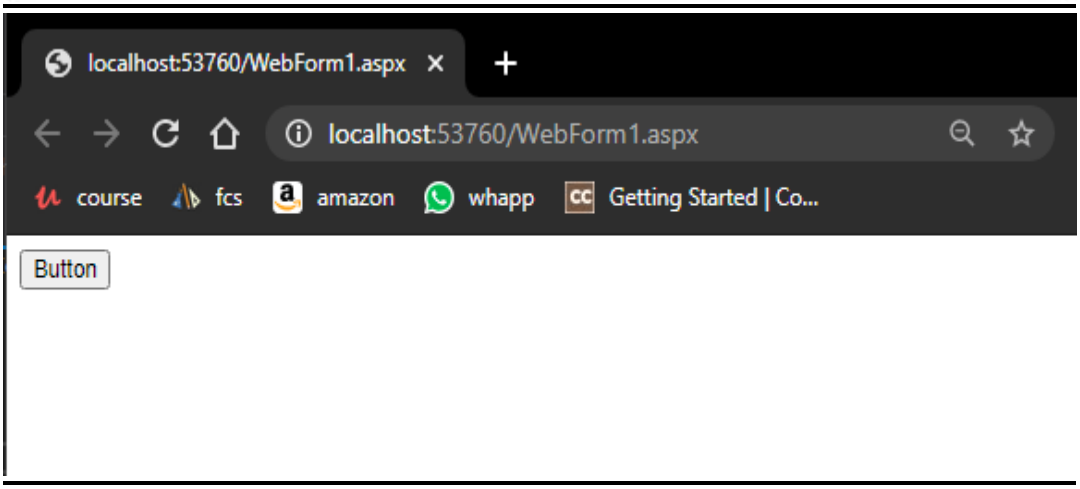
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.ServiceModel.Web;
using System.Text;

namespace prac2a
{
   [ServiceContract]
   public interface IService1
   {
      [OperationContract(IsOneWay = true)]
      void OneWayMessage();
   }
}
```

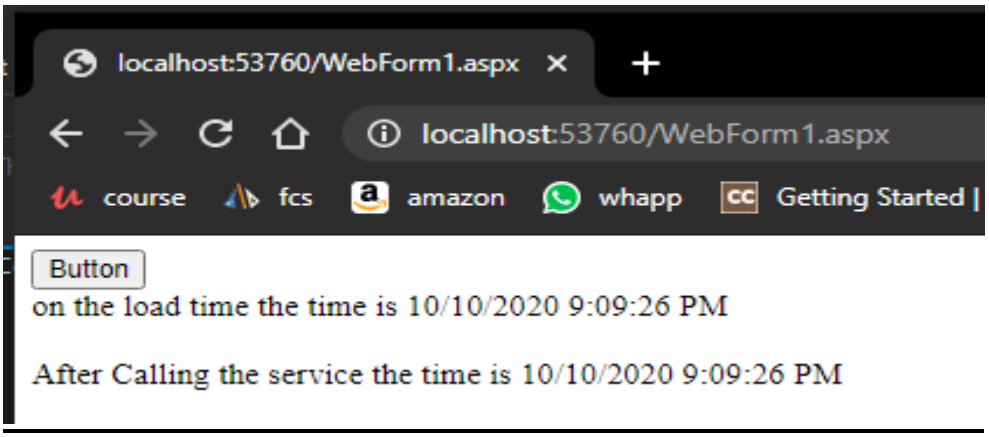
**4. Service1.svc.cs**

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

namespace prac2a
{
    public class Service1 : IService1
    {
        public void OneWayMessage()
        {
            Thread.Sleep(2000);
        }
    }}
```

**Output:-**


WCF Test Client
File Tools Help
My Service Projects
http://localhost:54880/Service1.svc
IService1 (BasicHttpBinding_IService
GetData()
GetDataAsync()
GetDataUsingDataContract()
GetDataUsingDataContractAsync
OneWayMessage()
OneWayMessageAsync()
Config File
OneWayMessage
Request
Microsoft WCF Test Client
One-way message is successfully delivered to the service side
OK
Response
Name
Value
Type
(return)
(null)
NullObject
Formatted
XML
Service invocation completed.


localhost:53760/WebForm1.aspx
localhost:53760/WebForm1.aspx
course
fcs
amazon
whapp
Getting Started | Co...
Button


localhost:53760/WebForm1.aspx
localhost:53760/WebForm1.aspx
course
fcs
amazon
whapp
Getting Started |
Button
on the load time the time is 10/10/2020 9:09:26 PM
After Calling the service the time is 10/10/2020 9:09:26 PM

**B) Request –Response**

**1. WebForm.aspx file**

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs" Inher-
its="WcfService2.WebForm1" %>

<!DOCTYPE html>

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

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

**2. WebForm.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 WcfService2
{
    public partial class WebForm1 : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

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

```
        {
            lblPagedate.Text = "Page date is " + DateTime.Now.ToString();
            ServiceReference1.Service1Client sc = new ServiceReference1.Service1Client();
            lblServicedate.Text = sc.RequestReplyPattern();
        }
    }
}
```

**3. IService.cs:-**

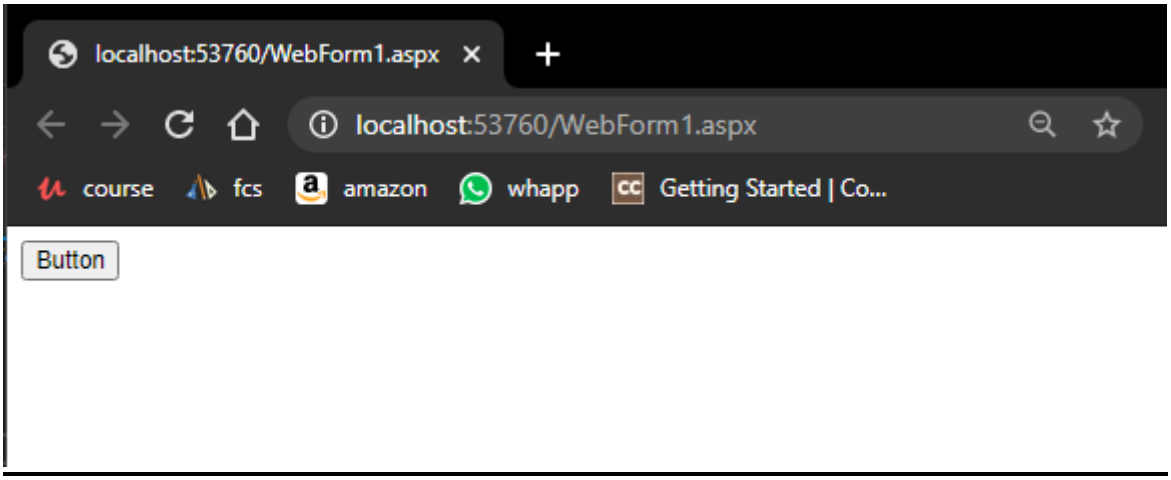
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.ServiceModel.Web;
using System.Text;
namespace WcfService2
{
    [ServiceContract]
    public interface IService1
    {
        [OperationContract(IsOneWay = false)]
        //declare the method which return type is string
        string RequestReplyPattern();

    }
}
```

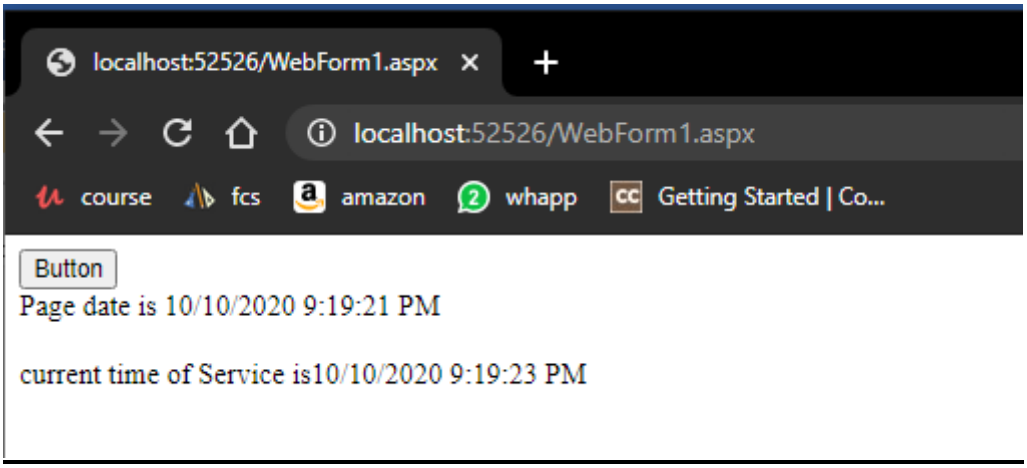
**4. Service1.svc.cs**

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

namespace WcfService2
{
```

```
public class Service1 : IService1
{
    public string RequestReplyPattern()
    {
        Thread.Sleep(2000);
        return "current time of Service is" + DateTime.Now.ToString();
    }
}
```

**Output:-**



Date: 30/09/2020

## Practical no 3

**AIM:** Demonstrates using the binding attribute of an endpoint element in WCF with webform.

**Program Code:-**

**IService1.cs**

```
using System;
using System.Collections.Generic;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.Text;
namespace WcfService7
{
    [ServiceContract]
    public interface IService1
    {
        [OperationContract]
        string Recharge(string Name, string company, string number, int amount);
    }
}
```

**Service1.svc.cs**

```
using System;
using System.Collections.Generic;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.Text;
namespace WcfService7
{
    public class Service1 : IService1
    {
        public string Recharge(string Name, string company, string number, int amount)
        {
            string message = string.Empty;

            if (string.IsNullOrEmpty(Name) || string.IsNullOrEmpty(company))
            {
                message = "Please Enter your name or company name";
```

```
        }
        else
        {
            if (number.Length == 10 && amount > 0)
            {
                message = "Recharge of " + amount + " Rs has been done successfully.";
            }
            else
            {
                message = "Recharge unsuccessfull. Please try again";
            }
        }
        return message;
    }
  }
}
```

**WebForm.aspx**

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs" Inher-
its="WcfService7.WebForm1" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>>Mobile Recharge<</title>
    <style>
 * {
 box-sizing: border-box
 }
 /* Add padding to containers */
 .container {
 padding: 16px;
 }
 /* Full-width input fields */
 input[type=text], input[type=password] {
 width: 100%;
 padding: 15px;
 margin: 5px 0 22px 0;
 display: inline-block;
 border: none;
 background: #f1f1f1;
 }
 input[type=text]:focus, input[type=password]:focus {
```

```
 background-color: #ddd;
 outline: none;
 }
 /* Overwrite default styles of hr */
 hr {
 border: 1px solid #f1f1f1;
 margin-bottom: 25px;
 }
 /* Set a style for the submit/register button */
 .registerbtn {
 background-color: #4CAF50;
 color: white;
 padding: 16px 20px;
 margin: 8px 0;
 border: none;
 cursor: pointer;
 width: 100%;
 opacity: 0.9;
 }
 .registerbtn:hover {
 opacity: 1;
 }

 /* Add a blue text color to links */
 a {
 color: dodgerblue;
 }
 /* Set a grey background color and center the text of the "sign in" section */
 .signin {
 background-color: #f1f1f1;
 text-align: center;
 }
 </style>
</head>
<body>
   <form id="form1" runat="server">
 <div>
 <div class="container">
 <caption class="auto-style1">
 <h1>Recharge</h1>
 <hr>
 <label for="email"><b>Name</b></label>
 <asp:TextBox ID="name" placeholder="Enter full name"
runat="server"></asp:TextBox>
 <label for="company"><b>Company</b></label>
 <asp:TextBox ID="company" placeholder="Enter full name"
runat="server"></asp:TextBox>
```

```
 <label for="no"><b>Mobile number</b></label>
 <asp:TextBox ID="no" placeholder="Enter full name" runat="server"></asp:TextBox>
 <label for="amount"><b>Amount</b></label>
 <asp:TextBox ID="amount" placeholder="Enter amount"
runat="server"></asp:TextBox>
 <hr>
 <asp:Button ID="rechargebtn" runat="server" Text="Recharge" BackColor="#4CAF50"
OnClick="rechargebtn_Click" Height="48px" Width="159px" />
 <hr>
<asp:Label ID="lbl" runat="server" Text=""></asp:Label>
 </div>
 </div>
 </form>

</body>
</html>
```

**WebForm.aspx.cs**

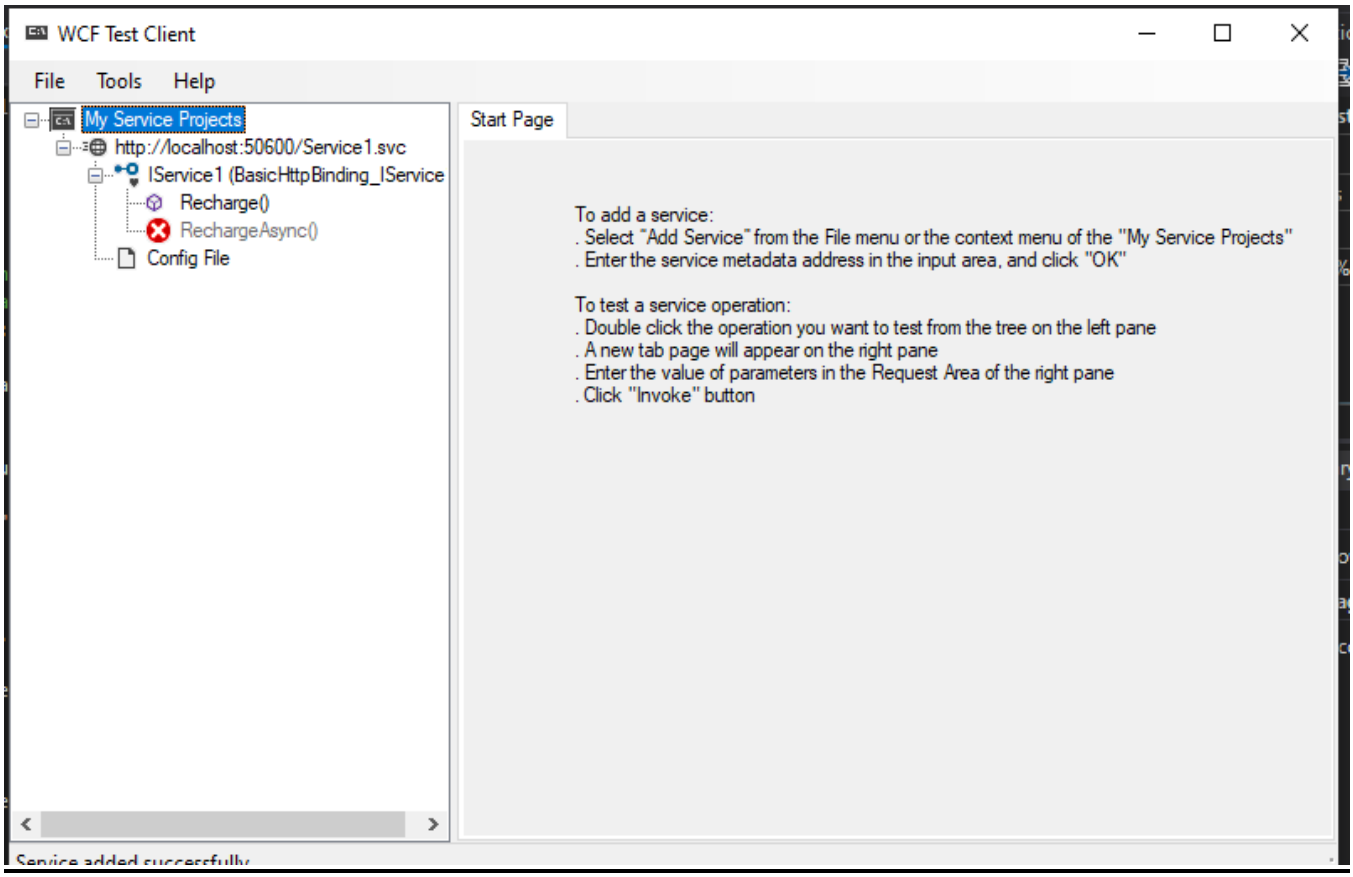
```
using System;
using System.Collections.Generic;

using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
namespace WcfService7
{
   public partial class WebForm1 : System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {

      }
      protected void rechargebtn_Click(object sender, EventArgs e)
      {
         try
         {
            ServiceReference1.Service1Client client = new ServiceReference1.Service1Client();
            lbl.Text = client.Recharge(name.Text, company.Text, no.Text,
           Convert.ToInt32(amount.Text));
         }
         catch (Exception)
         {
         }
      }
```

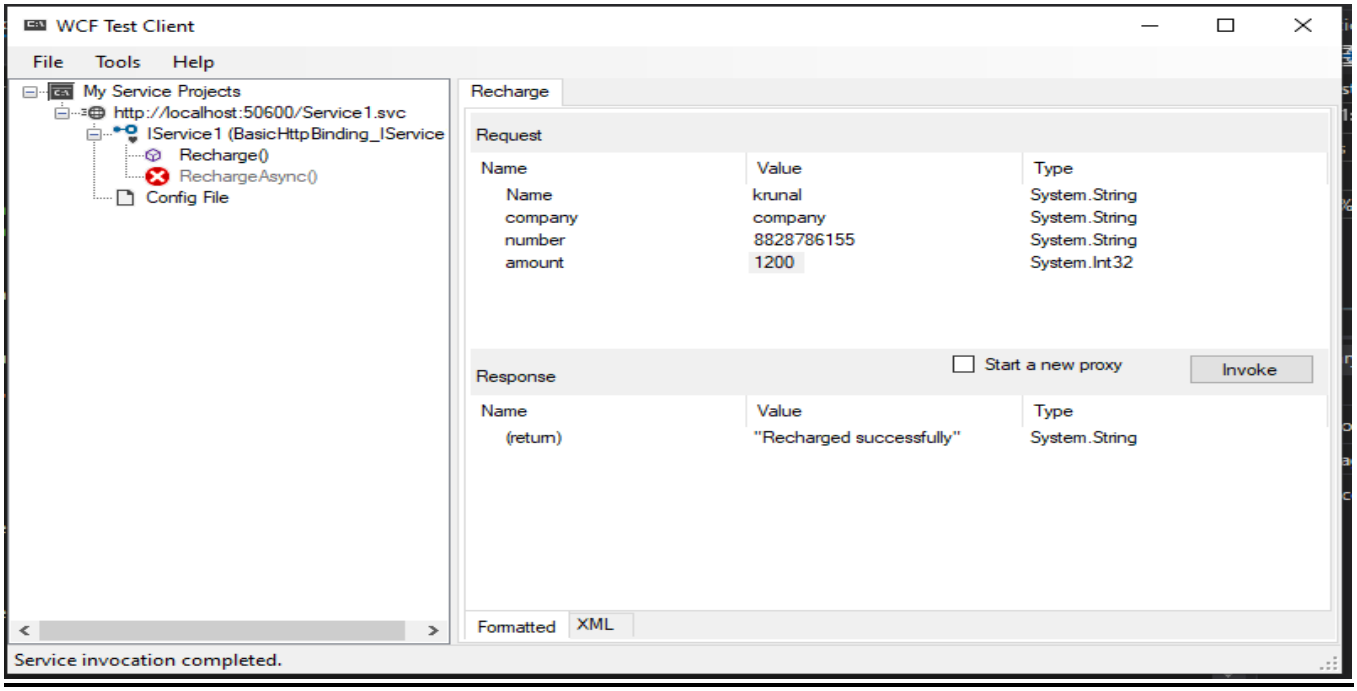
```
    }
}
```

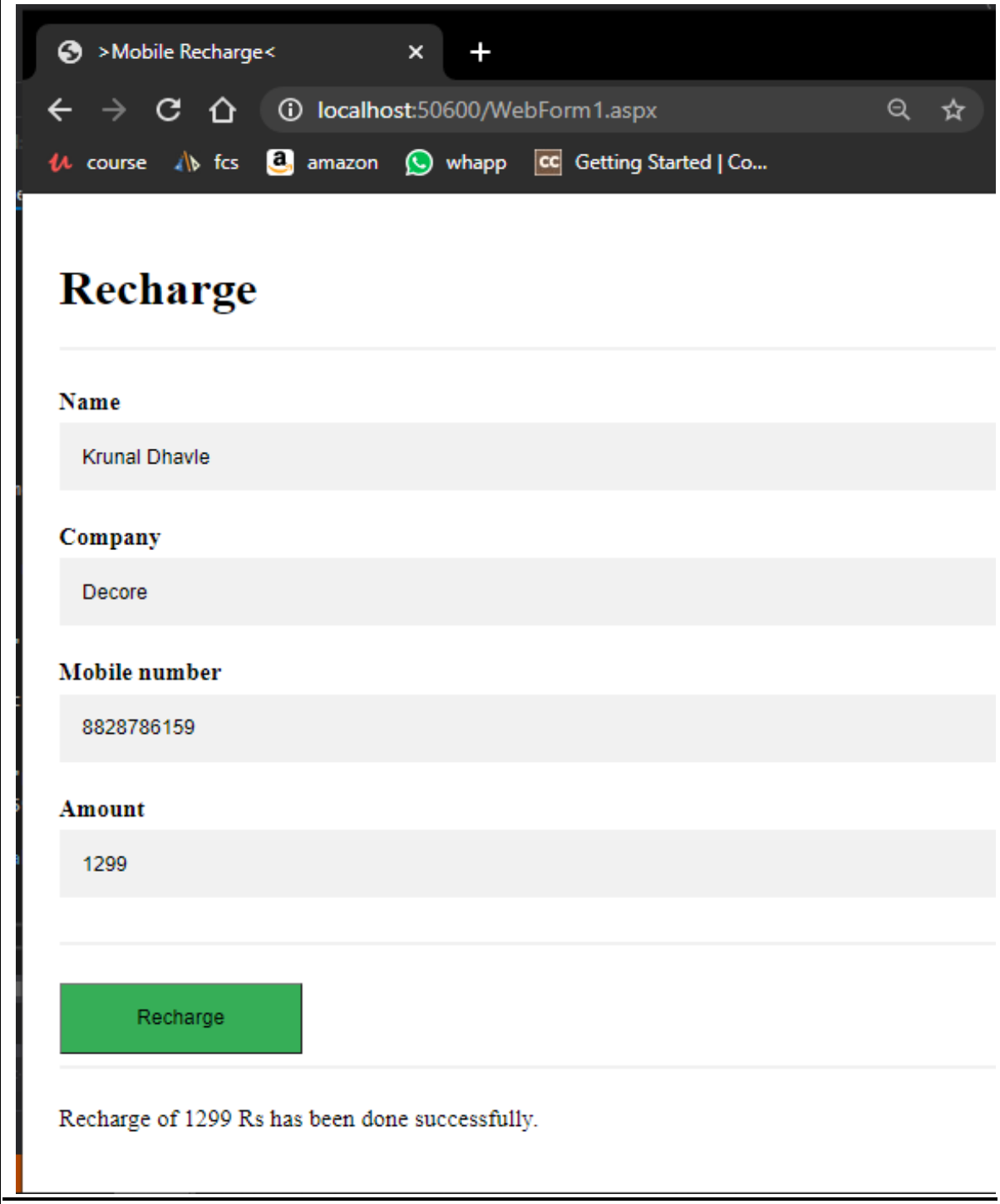
**WebConfig file**

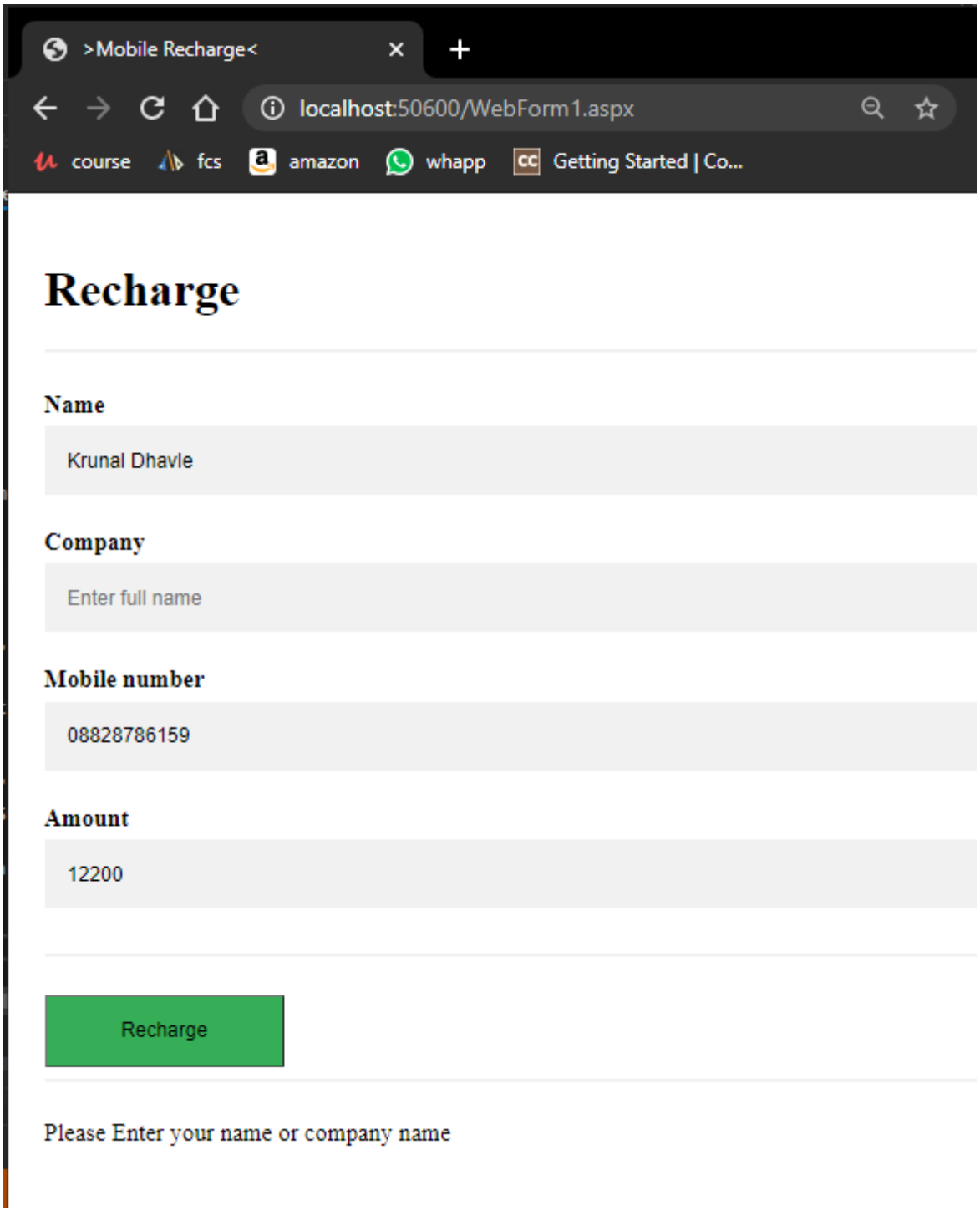
```
<bindings>
    <basicHttpBinding>
      <binding name="BasicHttpBinding_IService1" />
    </basicHttpBinding>
  </bindings>
  <client>
    <endpoint address="http://localhost:50600/Service1.svc" binding="basicHttpBinding"
      bindingConfiguration="BasicHttpBinding_IService1" contract="ServiceReference1.IService1"
      name="BasicHttpBinding_IService1" />
  </client>
```

**Output:**


WCF Test Client
File Tools Help
My Service Projects
http://localhost:50600/Service1.svc
IService1 (BasicHttpBinding_IService
Recharge()
RechargeAsync()
Config File
Start Page
To add a service:
. Select "Add Service" from the File menu or the context menu of the "My Service Projects"
. Enter the service metadata address in the input area, and click "OK"
To test a service operation:
. Double click the operation you want to test from the tree on the left pane
. A new tab page will appear on the right pane
. Enter the value of parameters in the Request Area of the right pane
. Click "Invoke" button


WCF Test Client
File Tools Help
My Service Projects
http://localhost:50600/Service1.svc
IService1 (BasicHttpBinding_IService
Recharge()
RechargeAsync()
Config File
Recharge
Request
Name Value Type
Name krunal System.String
company company System.String
number 8828786155 System.String
amount 1200 System.Int32
Start a new proxy
Invoke
Response
Name Value Type
(return) "Recharged successfully" System.String
Formatted XML
Service invocation completed.


>Mobile Recharge<
localhost:50600/WebForm1.aspx
course
fcs
amazon
whapp
Getting Started | Co...
Recharge
Name
Krunal Dhavle
Company
Decore
Mobile number
8828786159
Amount
1299
Recharge
Recharge of 1299 Rs has been done successfully.


>Mobile Recharge<
localhost:50600/WebForm1.aspx
course
fcs
amazon
whapp
Getting Started | Co...
Recharge
Name
Krunal Dhavle
Company
Enter full name
Mobile number
08828786159
Amount
12200
Recharge
Please Enter your name or company name

Date: 07/10/2020

## Practical no 4

**AIM**: Use WCF to create a basic ASP.NET Asynchronous JavaScript and XML (AJAX) service..

**Program Code:-**

**WebForm1.aspx**

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs" Inher-
its="WebApplication2.WebForm1" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
    <script src="jquery.js"></script>
    <script type="text/javascript">
        $(document).ready(function () {
            $("#btn").click(function () {
                var num1 = $("#txt1").val();
                var num2 = $("#txt2").val();
                $.ajax({
                    url: "Service1.svc/Sum",
                    type: "POST",
                    contentType: "application/json; charset=utf-8",
                    data:JSON.stringify({a: num1, b: num2}),
                    dataType: "json",
                    success : function(data){ $("#txt3").val(data.d); },
                    error : function(err){
                        alert(err);
                    }
                });
            });
        });
        </script>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            <input id="txt1" type="text" />
            <br />
            <br />
            <input id="txt2" type="text" />
            <br />
```

```
            <br />
            <input id="btn" type="button" value="Add Number" />
            <br />
            <br />
            <input id="txt3" type="text" />
            <p> Performed by krunal 713</p>
        </div>
    </form>
</body>
</html>
```

**Service1.svc.cs**

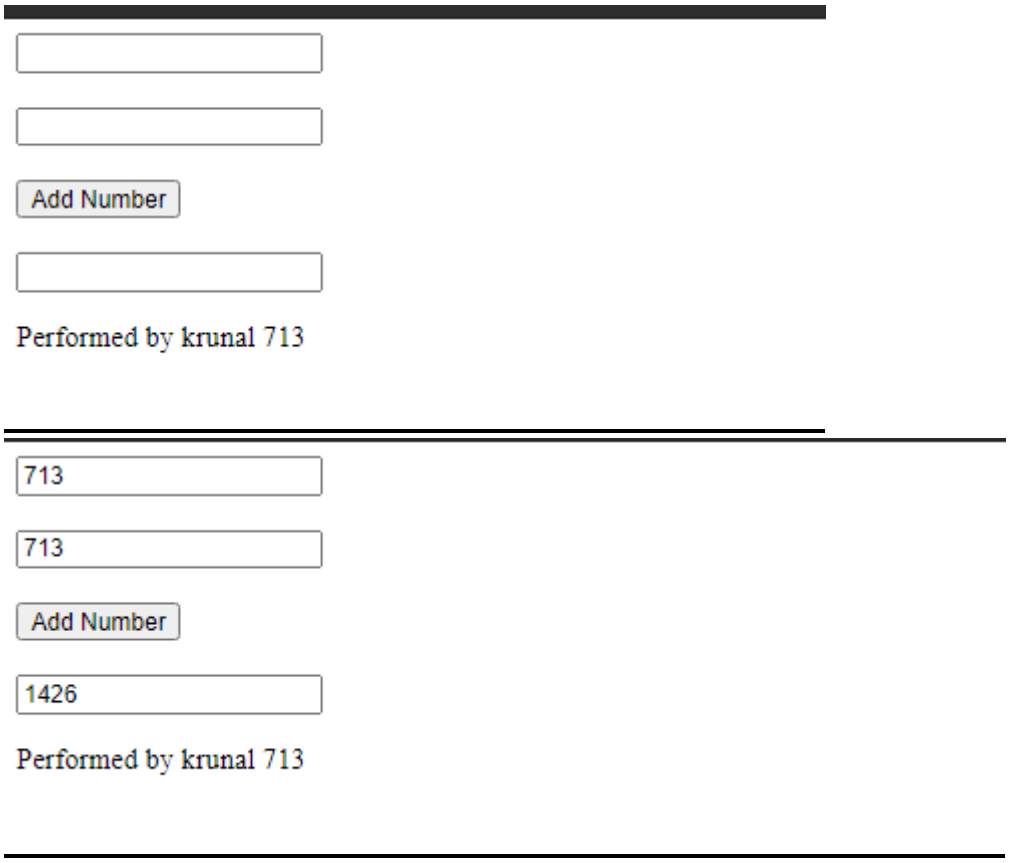
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.ServiceModel.Activation;
using System.ServiceModel.Web;
using System.Text;
namespace WebApplication2
{
    [ServiceContract(Namespace = "Multiplication")]
    [AspNetCompatibilityRequirements(RequirementsMode = AspNetCompatibilityRequire-
mentsMode.Allowed)]
    public class Service1
    {
        [OperationContract]
        public double Sum(double a, double b)
        {
            double result = a + b;
            return result;
        }
    }
}
```

**Web.Config**

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

<!--
  For more information on how to configure your ASP.NET application, please visit
  https://go.microsoft.com/fwlink/?LinkId=169433
  -->
<configuration>
```

```
  <system.web>
    <compilation debug="true" targetFramework="4.7.1"/>
    <httpRuntime targetFramework="4.7.1"/>
  </system.web>
  <system.codedom>
    <compilers>
      <compiler language="c#;cs;csharp" extension=".cs"
        type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.CSharpCodeProvider, Mi-
crosoft.CodeDom.Providers.DotNetCompilerPlatform, Version=2.0.0.0, Culture=neutral, PublicKey-
Token=31bf3856ad364e35"
        warningLevel="4" compilerOptions="/langversion:default /nowarn:1659;1699;1701"/>
      <compiler language="vb;vbs;visualbasic;vbscript" extension=".vb"
        type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.VBCodeProvider, Mi-
crosoft.CodeDom.Providers.DotNetCompilerPlatform, Version=2.0.0.0, Culture=neutral, PublicKey-
Token=31bf3856ad364e35"
        warningLevel="4" compilerOptions="/langversion:default /nowarn:41008 /de-
fine:_MYTYPE=\&amp;quot;Web\&amp;quot; /optionInfer+"/>
    </compilers>
  </system.codedom>

  <system.serviceModel>
    <behaviors>
      <endpointBehaviors>
        <behavior name="WebApplication2.Service1AspNetAjaxBehavior">
          <enableWebScript />
        </behavior>
      </endpointBehaviors>
    </behaviors>
    <serviceHostingEnvironment aspNetCompatibilityEnabled="true"
      multipleSiteBindingsEnabled="true" />
    <services>
      <service name="WebApplication2.Service1">
        <endpoint address="" behaviorConfiguration="WebApplication2.Service1AspNetAjaxBehav-
ior"
          binding="webHttpBinding" contract="WebApplication2.Service1" />
      </service>
    </services>
  </system.serviceModel>
</configuration>
```

**Output:-**


Add Number
Performed by krunal 713
713
713
Add Number
1426
Performed by krunal 713

Date: 13/10/2020

## Practical no 5

**AIM**: Implement a typical service and a typical client using WCF.

**Program Code:-**

**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 WcfService8
{
    [ServiceContract]
    public interface IService1
    {
        [OperationContract]
        int add(int a, int b);
        [OperationContract]
        int Sub(int a, int b);
        [OperationContract]
        int Mul(int a, int b);
        [OperationContract]
        int Div(int a, int b);
```

```
    }
}
```

**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 WcfService8
{
    public class Service1 : IService1
    {
        int IService1.add(int a, int b)
        {
            return (a + b);
        }
        int IService1.Sub(int a, int b)
        {
            if (a >= b)
                return (a - b);
            else
                return (b - a);
        }
        int IService1.Mul(int a, int b)
```

```
        {
            return (a * b);
        }
        int IService1.Div(int a, int b)
        {
            if (a >= b)
                return (a / b);
            else
                return (b / a);
        }
    }
}
```

**Webform1.aspx**

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs"
Inherits="WcfService8.WebForm1" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server" style="background-color:lightgray">
        <div>
            Enter first number
            <asp:TextBox ID="TextBox1" runat="server" ></asp:TextBox>     
            Enter second number
            <asp:TextBox ID="TextBox2" runat="server"></asp:TextBox> <br /><br />
```

```
        <asp:Button ID="Button1" runat="server" Text="Add" OnClick="btnadd_Click" /> 
  
        <asp:Button ID="Button2" runat="server" Text="Sub" OnClick="btnsub_Click"/> 
  
        <asp:Button ID="Button3" runat="server" Text="Mul" OnClick="btnmul_Click"/> 
  
        <asp:Button ID="Button4" runat="server" Text="Div" OnClick="btndiv_Click"/>  <br /><br
/>
        <asp:Label ID="Label2" runat="server" Text="Result:-"></asp:Label>    
        <asp:TextBox ID="TextBox3" runat="server"></asp:TextBox> <br /><br /><br /><br />
        performed by krunal 713
      </div>
   </form>
</body>
</html>
```

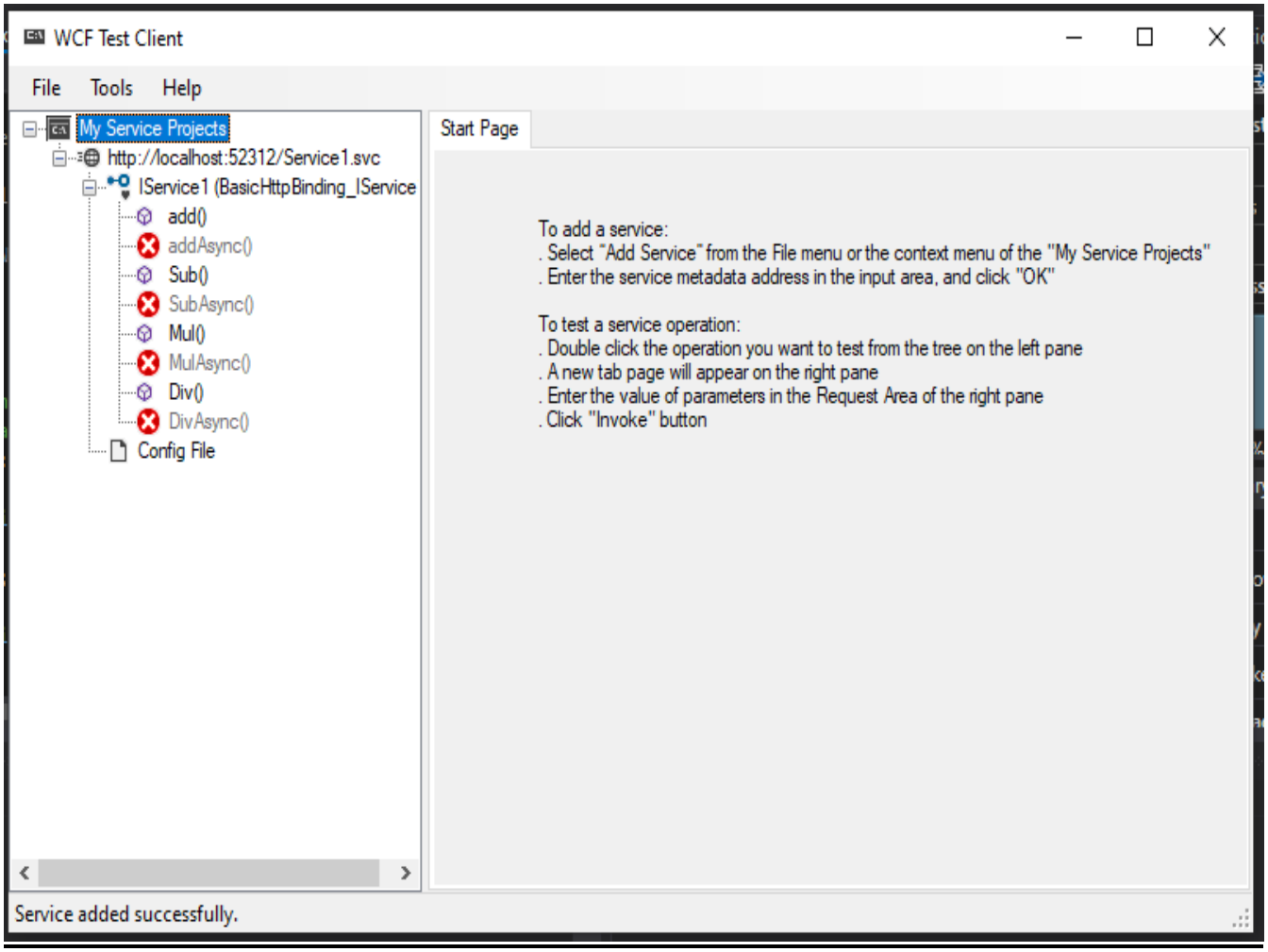
**WebForm.aspx.cs**

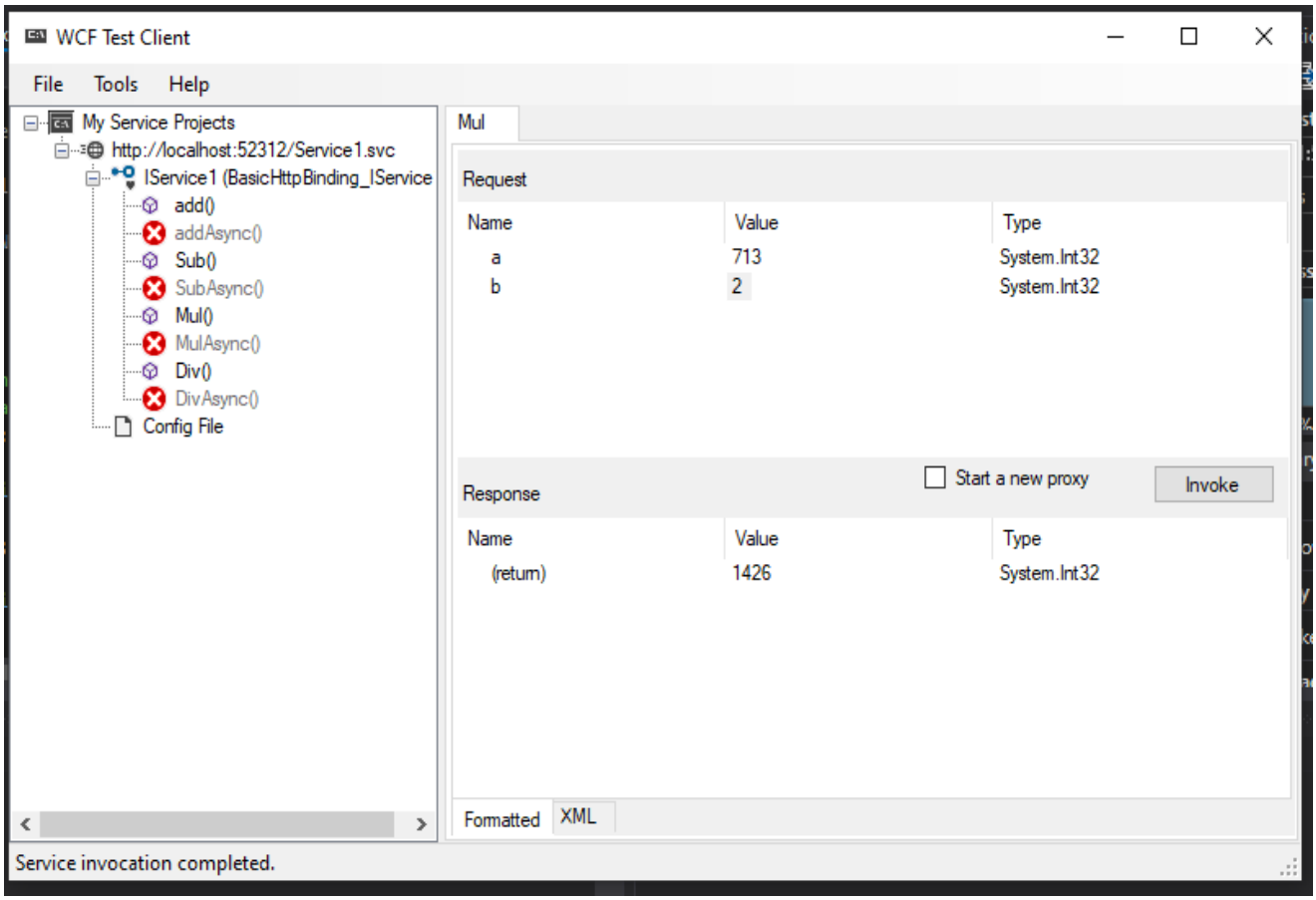
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
namespace WcfService8
{
   public partial class WebForm1 : System.Web.UI.Page
   {
      ServiceReference1.Service1Client Client = new ServiceReference1.Service1Client();
      protected void Page_Load(object sender, EventArgs e)
```

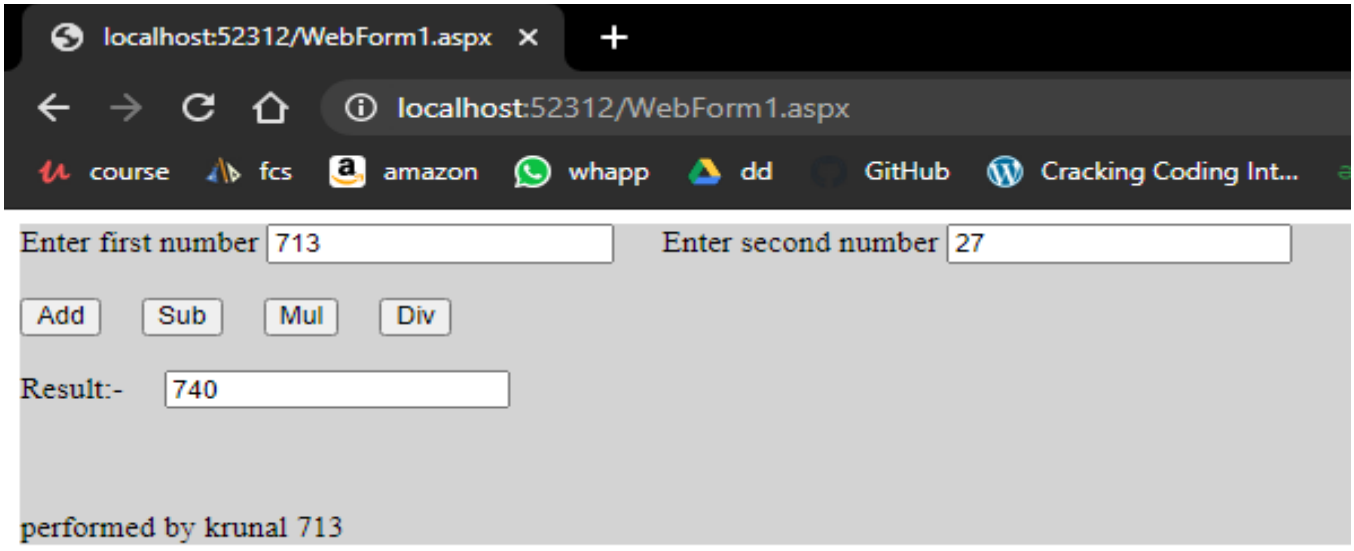
```
{ }
int a, b;
protected void btnadd_Click(object sender, EventArgs e)
{
   a = Convert.ToInt32(TextBox1.Text);
   b = Convert.ToInt32(TextBox2.Text);
   int Addition = Client.add(a, b);
   TextBox3.Text = Addition.ToString();
}
protected void btnsub_Click(object sender, EventArgs e)
{
   a = Convert.ToInt32(TextBox1.Text);
   b = Convert.ToInt32(TextBox2.Text);
   int sub = Client.Sub(a, b);
   TextBox3.Text = sub.ToString();
}
protected void btnmul_Click(object sender, EventArgs e)
{
   a = Convert.ToInt32(TextBox1.Text);
   b = Convert.ToInt32(TextBox2.Text);
   int mul = Client.Mul(a, b);
   TextBox3.Text = mul.ToString();
}
protected void btndiv_Click(object sender, EventArgs e)
{
   a = Convert.ToInt32(TextBox1.Text);
   b = Convert.ToInt32(TextBox2.Text);
   int div = Client.Div(a, b);
```

```
            TextBox3.Text = div.ToString();
        }
    }
}
```

**Output:-**




WCF Test Client
File Tools Help
My Service Projects
http://localhost:52312/Service1.svc
IService1 (BasicHttpBinding_IService
add()
addAsync()
Sub()
SubAsync()
Mul()
MulAsync()
Div()
DivAsync()
Config File
Mul
Request
Name Value Type
a 713 System.Int32
b 2 System.Int32
Start a new proxy
Invoke
Response
Name Value Type
(return) 1426 System.Int32
Formatted XML
Service invocation completed.


localhost:52312/WebForm1.aspx
localhost:52312/WebForm1.aspx
course fcs amazon whapp dd GitHub Cracking Coding Int...
Enter first number 713
Enter second number 27
Add Sub Mul Div
Result:- 740
performed by krunal 713

Date: 14/10/2020

## Practical no 6

**AIM**: Develop client which consumes web services developed in different platform.

**Program Code:-**

**WebService1.asmx.cs**

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Services;
namespace WebApplication4
{
   [WebService(Namespace = "http://tempuri.org/")]
   [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
   [System.ComponentModel.ToolboxItem(false)]
   public class WebService1 : System.Web.Services.WebService
   {
      [WebMethod]
      public double Mod(double number1 , double number2)
      {
         return (number1 % number2);
      }
   }
}
```

**index.html**

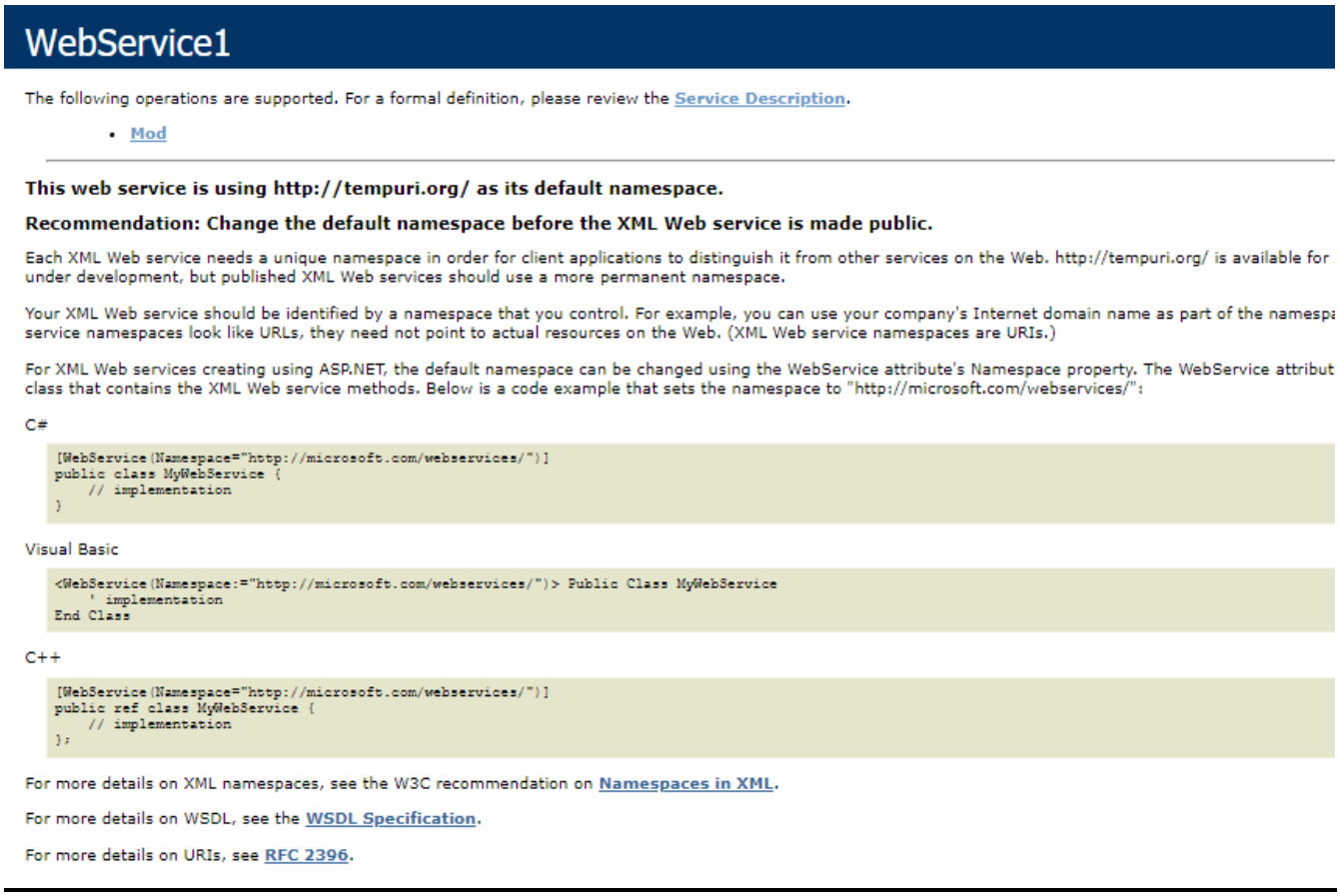
```
<!DOCTYPE html>
<html>
   <head>
      <title>Mod </title>
      <meta charset="UTF-8">
      <meta name="viewport" content="width=device-width, initial-scale=1.0">
   </head>
   <body>
      <form>
         <input type="text" name="txt1" placeholder="Enter First Number"><br><br>
         <input type="text" name="txt2" placeholder="Enter Second Number"><br><br>
         <input type="submit" formaction="Mod.jsp" value="Mod Number"><br><br>
          performed by krunal dhavle 713;
      </form>
   </body>
</html>
```

**Mod.jsp**

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title>JSP Page</title>
   </head>
   <body>
```

```
    <%-- start web service invocation --%><hr/>
    <%
        double num1 = Double.parseDouble(request.getParameter("txt1"));
        double num2 = Double.parseDouble(request.getParameter("txt2"));
    try {
            com.dd.WebService1 service = new com.dd.WebService1();
            com.dd.WebService1Soap port = service.getWebService1Soap();
             // TODO initialize WS operation arguments here
            double number1 = num1;
            double number2 = num2;
            // TODO process result here
            double result = port.mod(number1, number2);
            out.println("Result = "+result);
    } catch (Exception ex) {
    }
    %>
    </body>
</html>
```

**Output:-**



WebService1

The following operations are supported. For a formal definition, please review the Service Description.

- Mod

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

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

Each XML Web service needs a unique namespace in order for client applications to distinguish it from other services on the Web. http://tempuri.org/ is available for under development, but published 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 your company's Internet domain name as part of the namespa service namespaces look like URLs, they need not point to actual resources on the Web. (XML Web service namespaces are URIs.)

For XML Web services creating using ASP.NET, the default namespace can be changed using the WebService attribute's Namespace property. The WebService attribut class that contains the XML Web 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
```

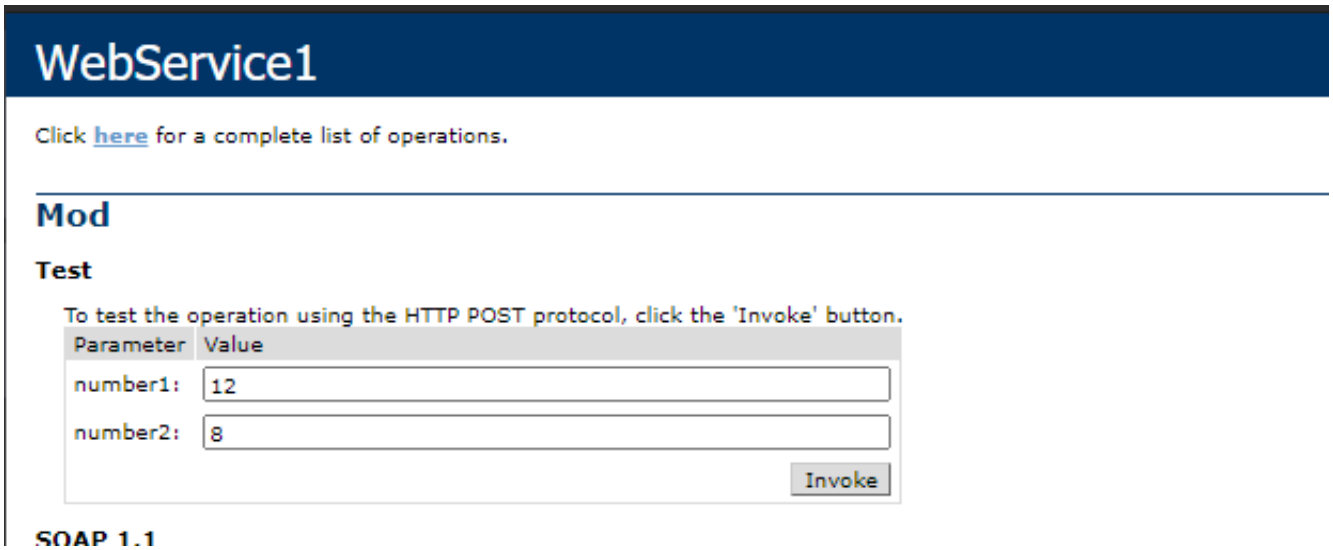
C++

```
[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.



WebService1

Click here for a complete list of operations.

**Mod**

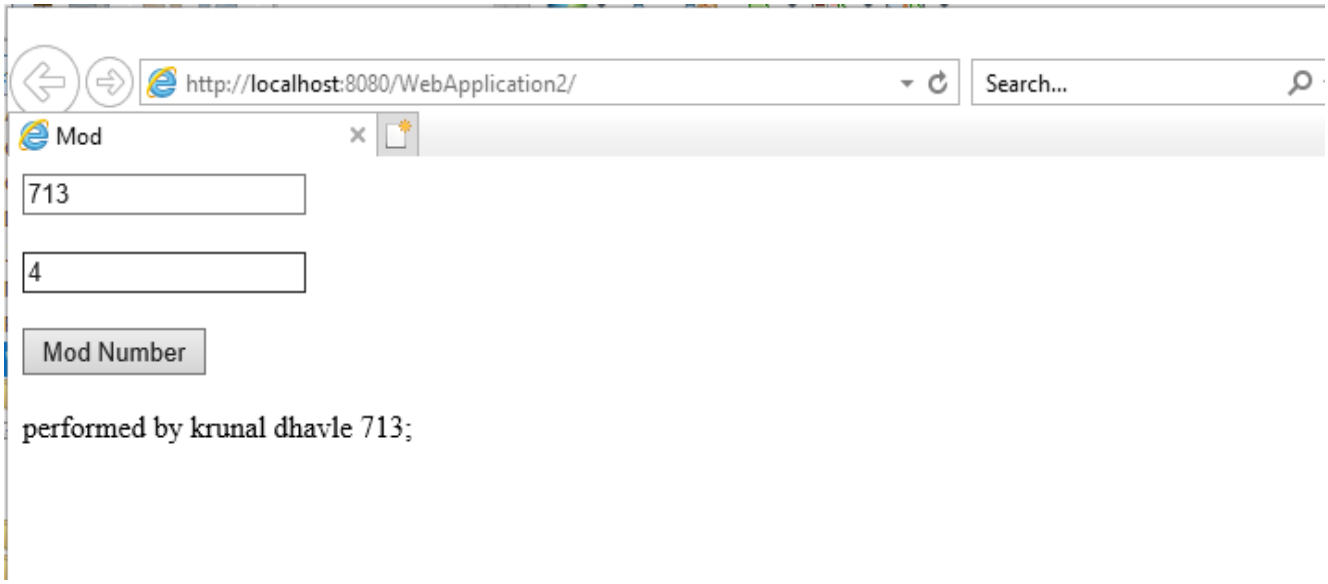
**Test**

To test the operation using the HTTP POST protocol, click the 'Invoke' button.

| Parameter | Value |
|---|---|
| number1: | 12 |
| number2: | 8 |

Invoke

**SOAP 1.1**


http://localhost:8080/WebApplication2/
Search...
Mod
713
4
Mod Number
performed by krunal dhavle 713;

JSP Page
Result = 1.0

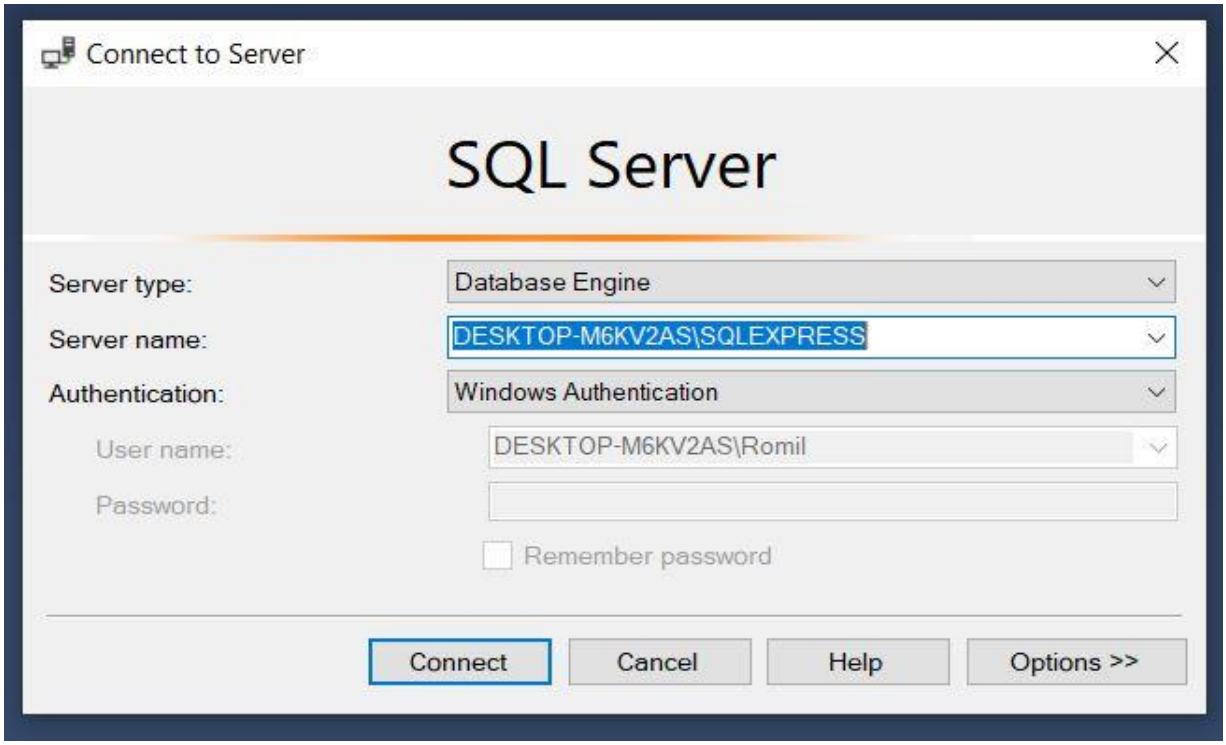
**Date: 21/10/2020**

## Practical no 7

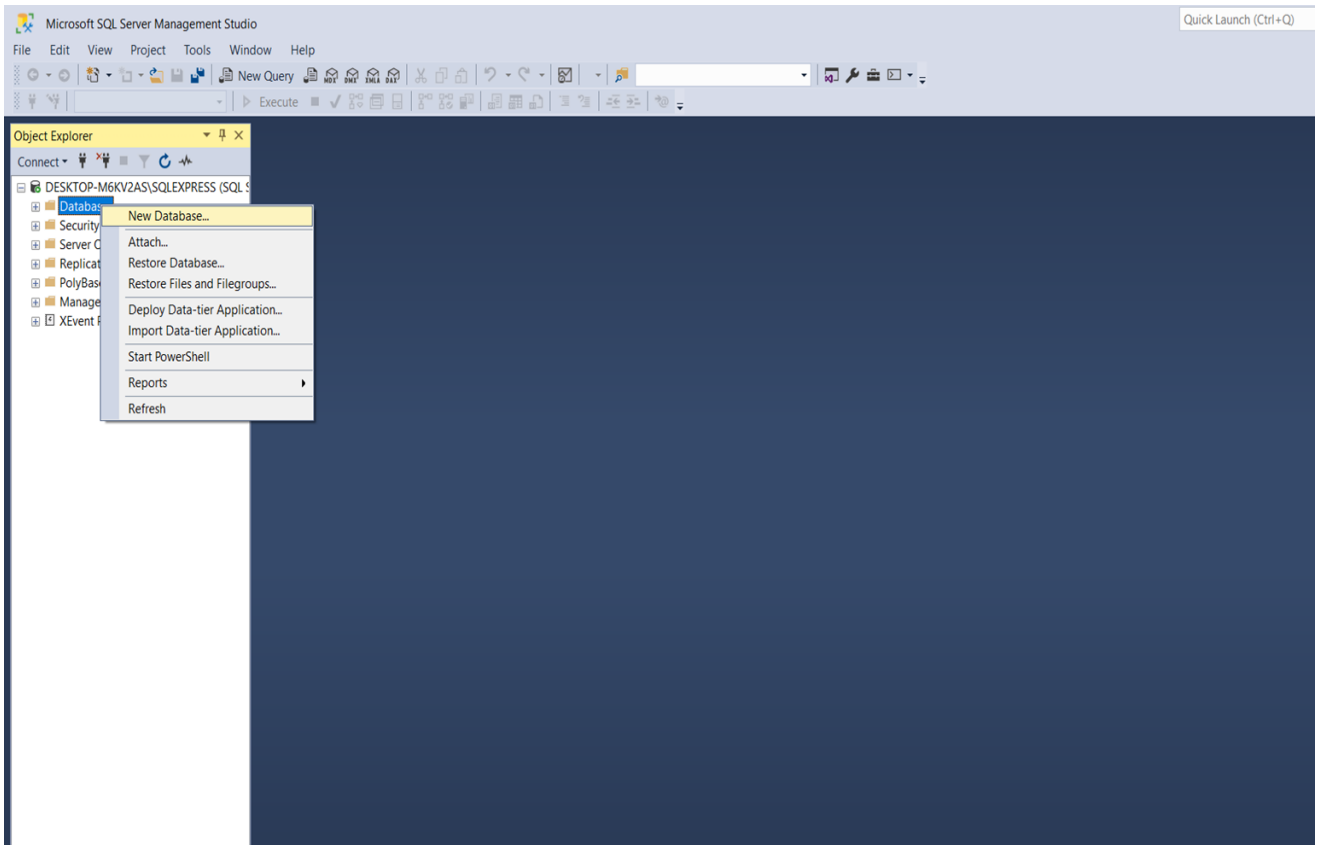
**AIM: :** Define a web service method that returns the contents of a database in a JSON string. The contents should be displayed in a tabular format.

**Steps for creating a Database:**

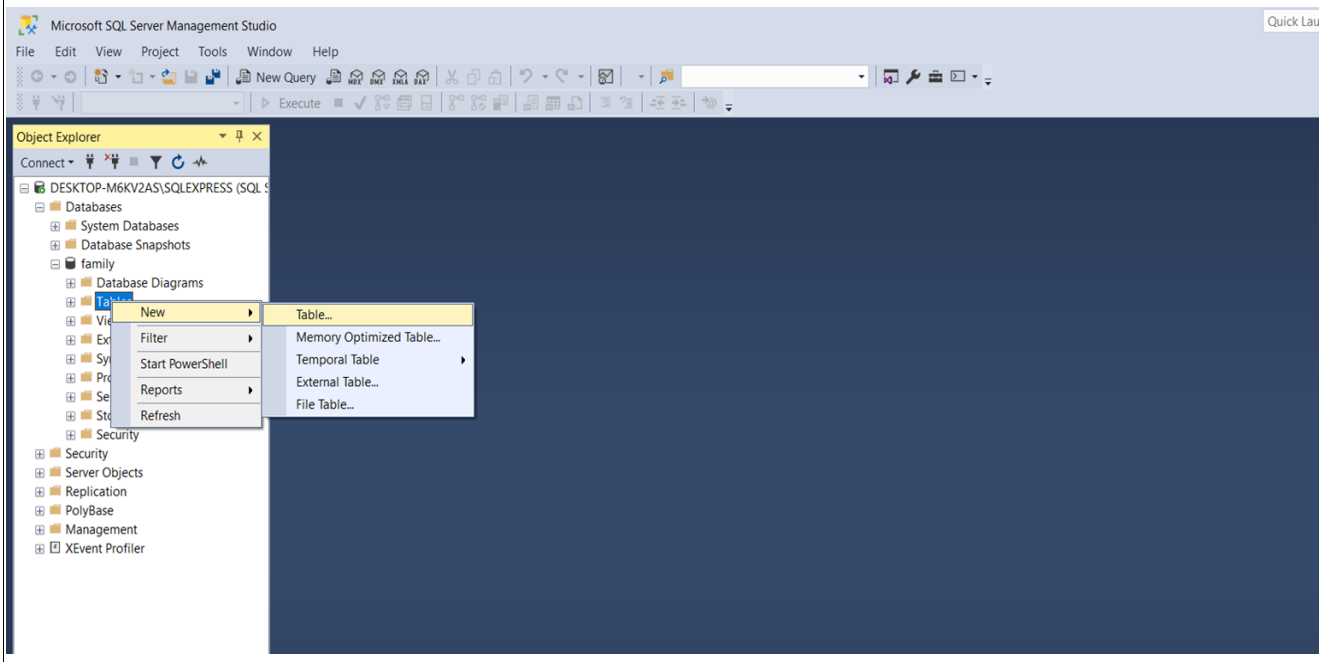
**1]** Open Microsoft SQL Server Management Studio. After opening the Management Studio it will ask to connect to a server. Select a proper server and click connect.



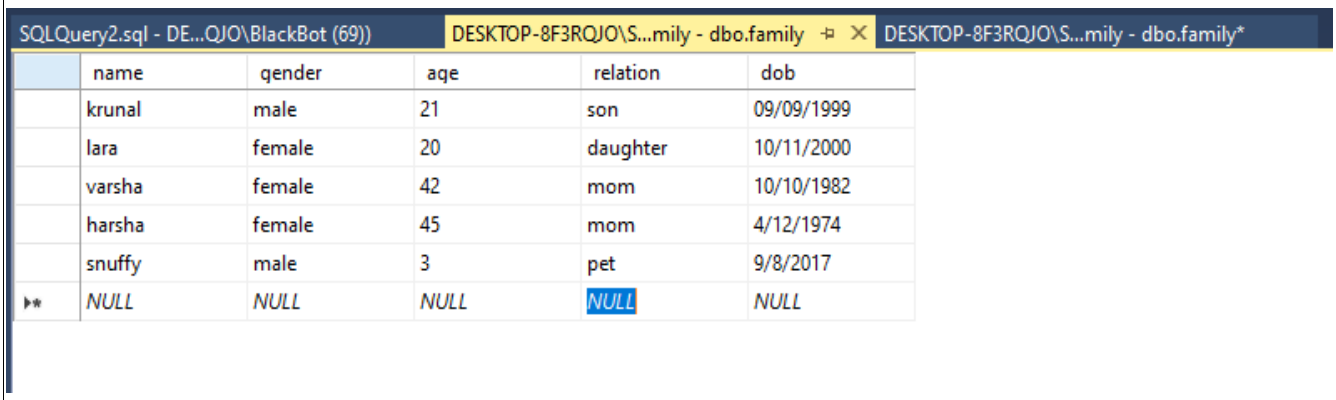

**2]** From object explorer right click on **Databases** and click **New Database.** Give name to Database.




3] Right Click on **Tables** and select **New ->Table**.



**4]** Edit column name and datatypes.
**5]** Save the table and give a name.
**6]** From **Tables** select the table you created. Right click on it and select **Edit top 200 rows.**
**7]** Enter the data into the table and save.




| name | qender | aqe | relation | dob |
|---|---|---|---|---|
| krunal | male | 21 | son | 09/09/1999 |
| lara | female | 20 | daughter | 10/11/2000 |
| varsha | female | 42 | mom | 10/10/1982 |
| harsha | female | 45 | mom | 4/12/1974 |
| snuffy | male | 3 | pet | 9/8/2017 |
| *NULL* | *NULL* | *NULL* | *NULL* | *NULL* |

**CODE:-**

**Family.cs:**

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
namespace prac7
{
   public class Family
   {
      public string name { get; set; }
      public string gender { get; set; }
      public string age { get; set; }
      public string relation { get; set; }
      public string dob { get; set; }
   }
}
```

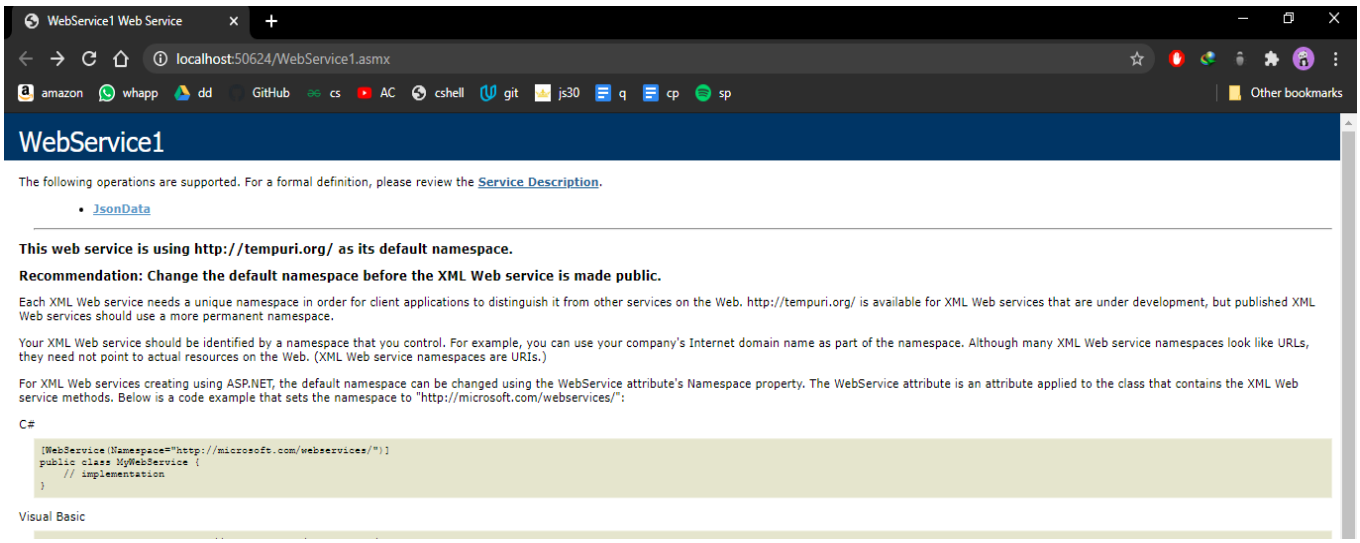
**Webservice.asmx.cs:**

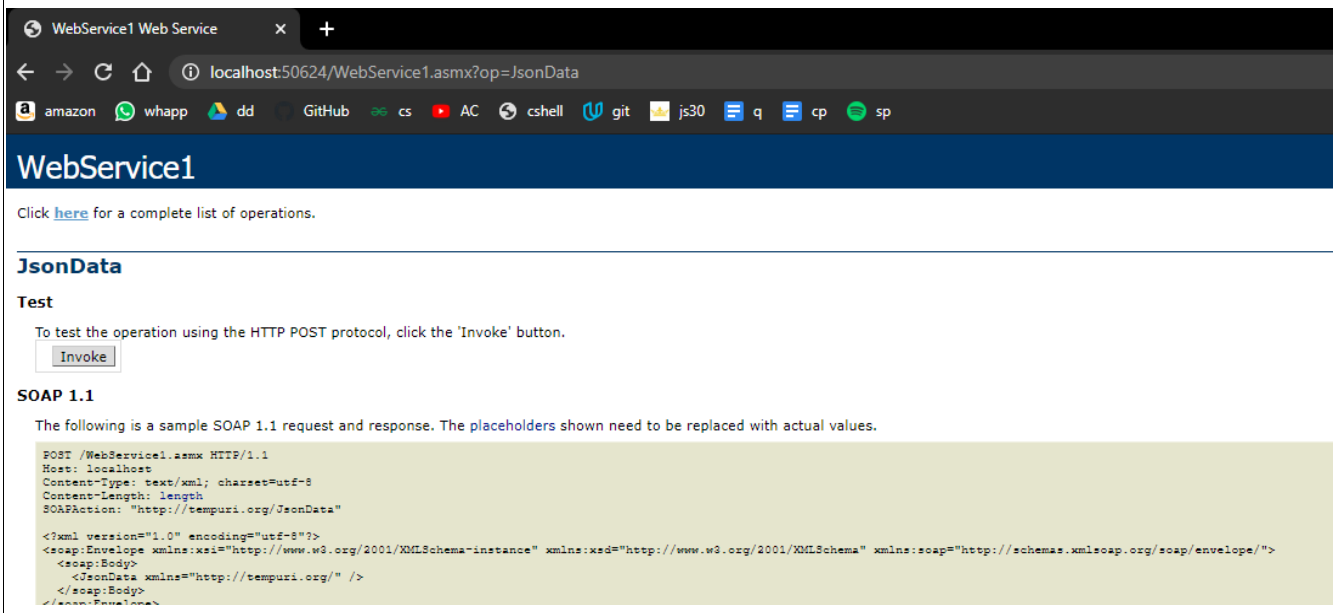
```
using System;
using System.Collections.Generic;
using System.Configuration;
using System.Data.SqlClient;
using System.Linq;
using System.Web;
```

```
using System.Web.Script.Serialization;
using System.Web.Services;

namespace prac7
{
    /// <summary>
    /// Summary description for WebService1
    /// </summary>
    [WebService(Namespace = "http://tempuri.org/")]
    [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
    [System.ComponentModel.ToolboxItem(false)]
    public class WebService1 : System.Web.Services.WebService
    {

        [WebMethod]
        public string JsonData()
        {
            List<Family> objFamilylist = new List<Family>();
            string str = ConfigurationManager.ConnectionStrings["DBContext"].ConnectionString;
            using (SqlConnection con = new SqlConnection(str))
            {
                SqlCommand cmd = new SqlCommand("select * from family", con);
                cmd.Connection = con;
                con.Open();
                SqlDataReader rdr = cmd.ExecuteReader();
                while (rdr.Read())
                {
                    Family family = new Family();
                    family.name = rdr["Name"].ToString();
                    family.gender = rdr["Gender"].ToString();
                    family.age = rdr["Age"].ToString();
                    family.relation = rdr["Relation"].ToString();
                    family.dob = rdr["Dob"].ToString();
                    objFamilylist.Add(family);
                }
                JavaScriptSerializer js = new JavaScriptSerializer(); return js.Serialize(objFamilylist);
            }
        }
    }
}
```

**Web.config**:-

```
<?xml version="1.0" encoding="utf-8"?>
<!--
  For more information on how to configure your ASP.NET application, please visit
  https://go.microsoft.com/fwlink/?LinkId=169433
  -->
<configuration>
  <system.web>
    <compilation debug="true" targetFramework="4.7.1"/>
    <httpRuntime targetFramework="4.7.1"/>
  </system.web>
  <connectionStrings>
    <add name="DBContext" providerName="System.Data.SqlClient" connectionString="Data
Source=DESKTOP-8F3RQJO\SQLEXPRESS01;Initial Catalog=family;Integrated Security=True "/>
  </connectionStrings>
  <system.serviceModel>
    <bindings>
      <basicHttpBinding>
        <binding name="WebService1Soap" />
      </basicHttpBinding>
    </bindings>
    <client>
      <endpoint address="http://localhost:58555/WebService1.asmx"
      binding="basicHttpBinding"
      bindingConfiguration="WebService1Soap"
      contract="ServiceReference1.WebService1Soap"
      name="WebService1Soap" />
    </client>
  </system.serviceModel>
</configuration>
```






**Now adding Webform in the same solution explorer .**

**(adding newtonsoft.json)**

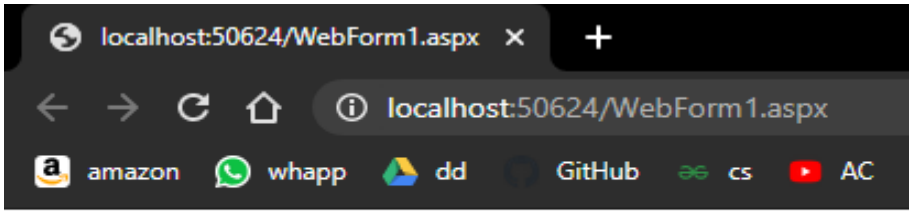
**Webform1.aspx.cs:**

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs" Inher-
its="prac7.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:GridView runat="server" ID="GridView"
OnSelectedIndexChanged="GridView_SelectedIndexChanged"></asp:GridView>
</div>
<h3>Performed By krunal, 713</h3>
</form>
</body>
</html>
```

Webform.aspx:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs" Inher-
its="prac7.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:GridView runat="server" ID="GridView"
OnSelectedIndexChanged="GridView_SelectedIndexChanged"></asp:GridView>
</div>
<h3>Performed By krunal, 713</h3>
</form>
</body>
</html>
```

**Output:-**




| name | gender | age | relation | dob |
|---|---|---|---|---|
| krunal | male | 21 | son | 09/09/1999 |
| lara | female | 20 | daughter | 10/11/2000 |
| varsha | female | 42 | mom | 10/10/1982 |
| harsha | female | 45 | mom | 4/12/1974 |
| snuffy | male | 3 | pet | 9/8/2017 |

**Performed By krunal, 713**

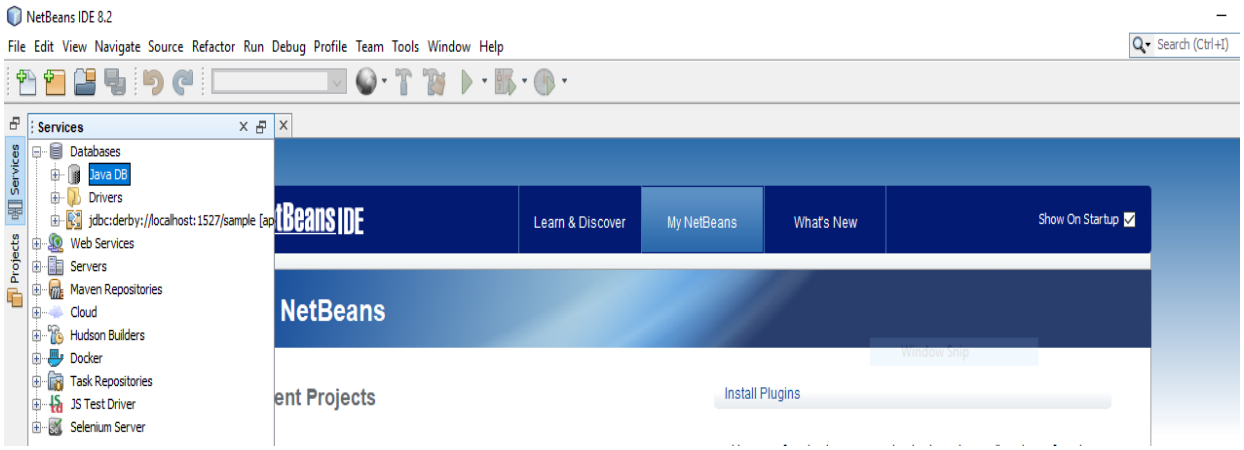
Date: 28/10/2020

## Practical no 8

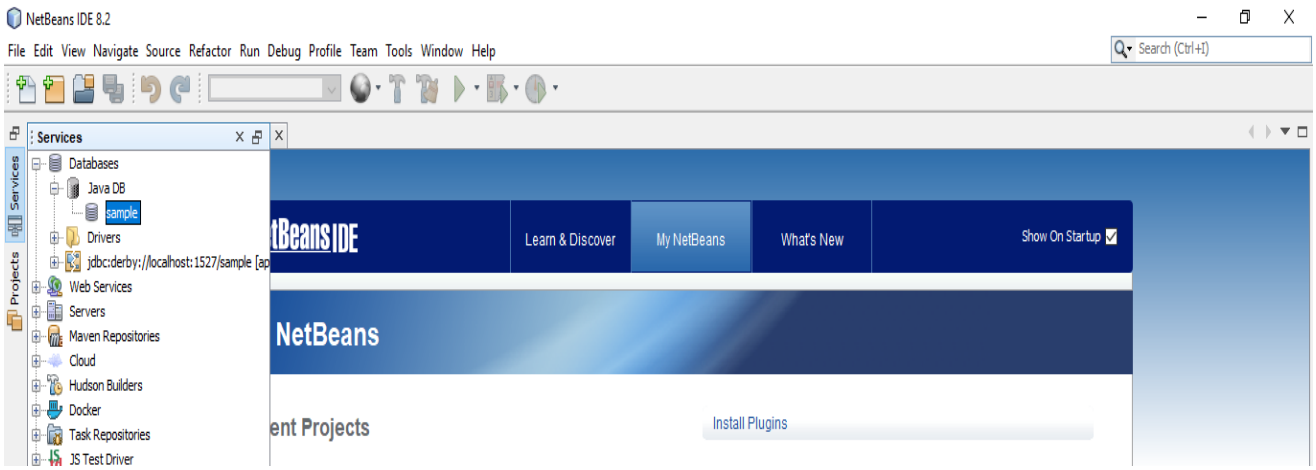
**AIM**: Define a RESTful web service that accepts the details to be stored in a database and performs CRUD operation**.**

**Steps :-**

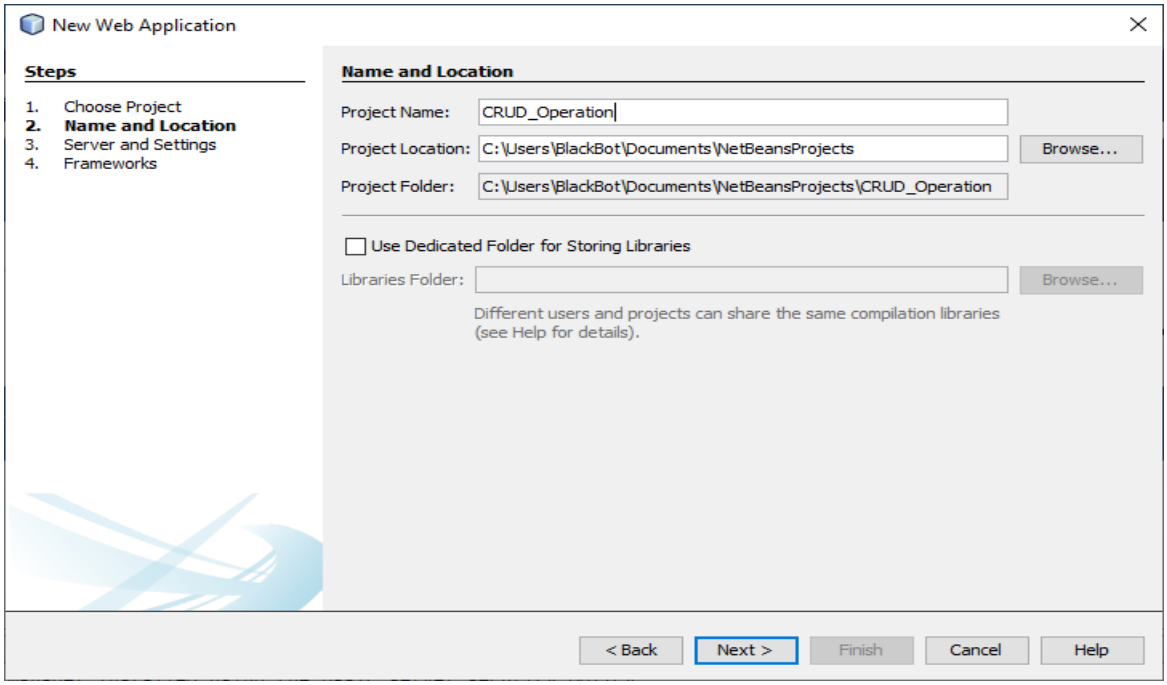
**1) Right click on Java DB** and then **click on Start Server** to start the server .

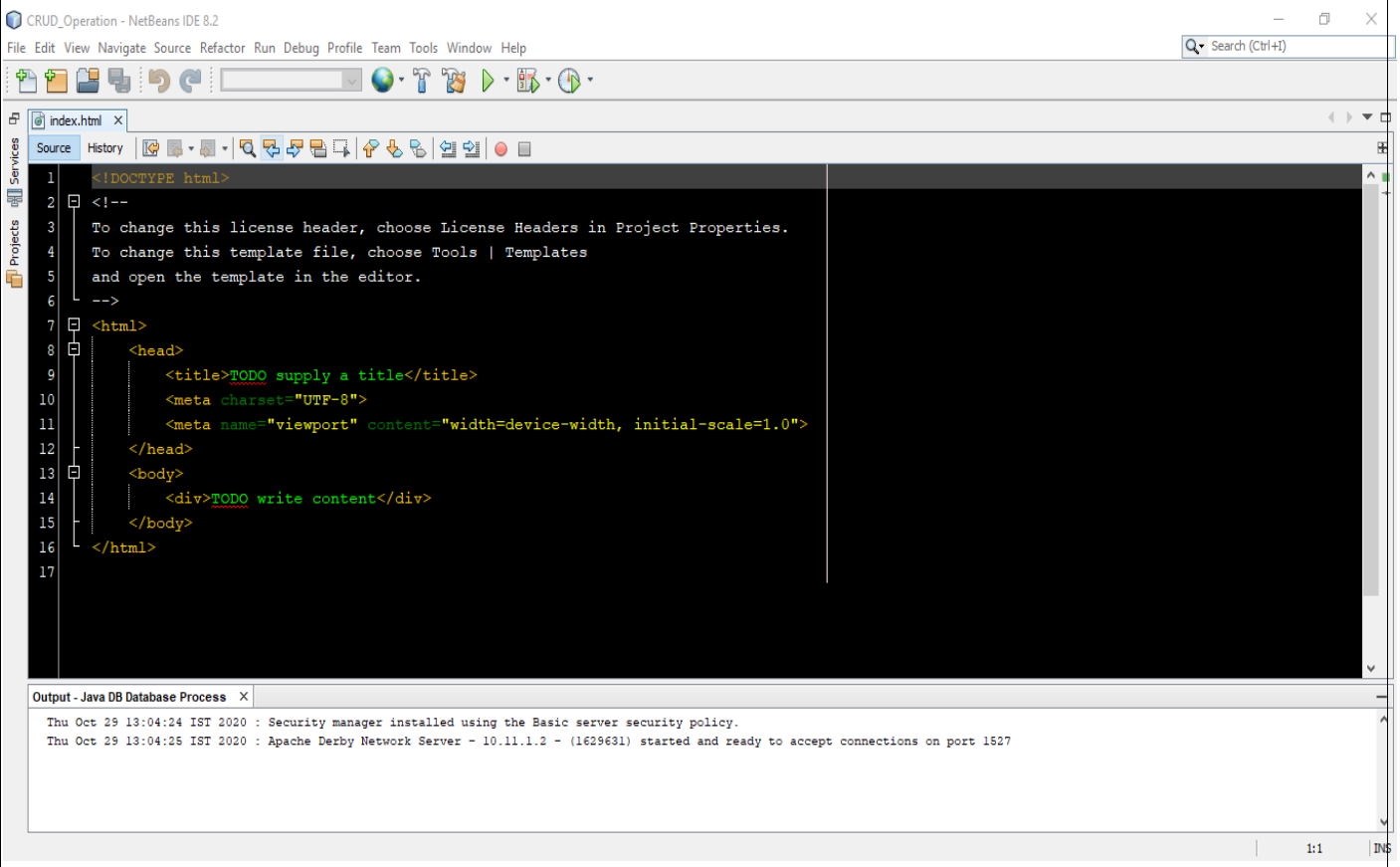



**2)** Now expand Java DB and **right click on sample** and then **click on connect** to connect the sample database with server.

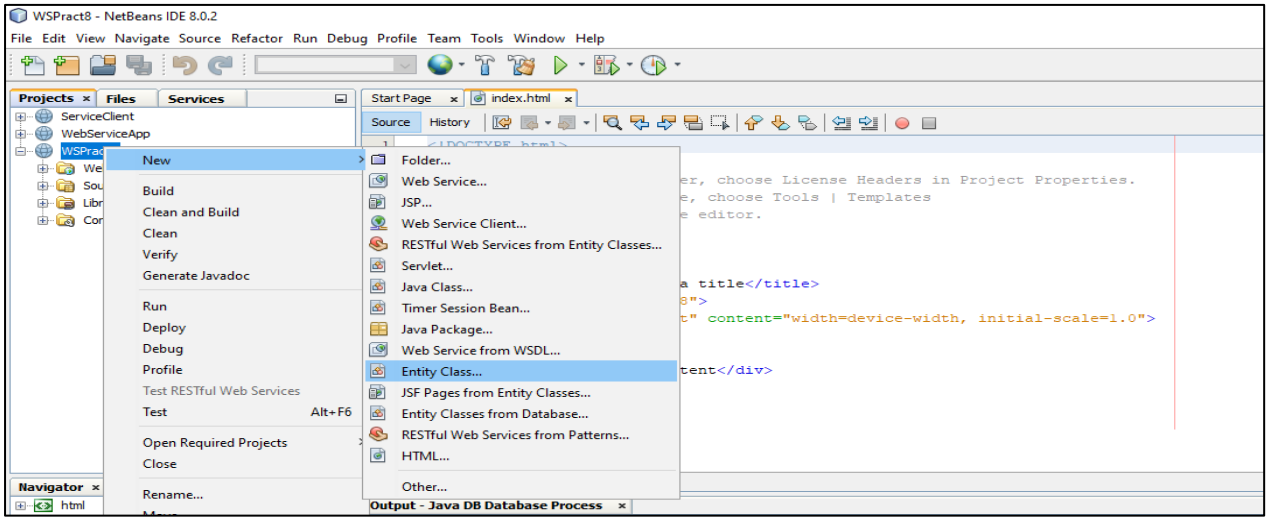


**3)** Now create a web application with the name **CRUD_Operation**. A window will open like following pic.



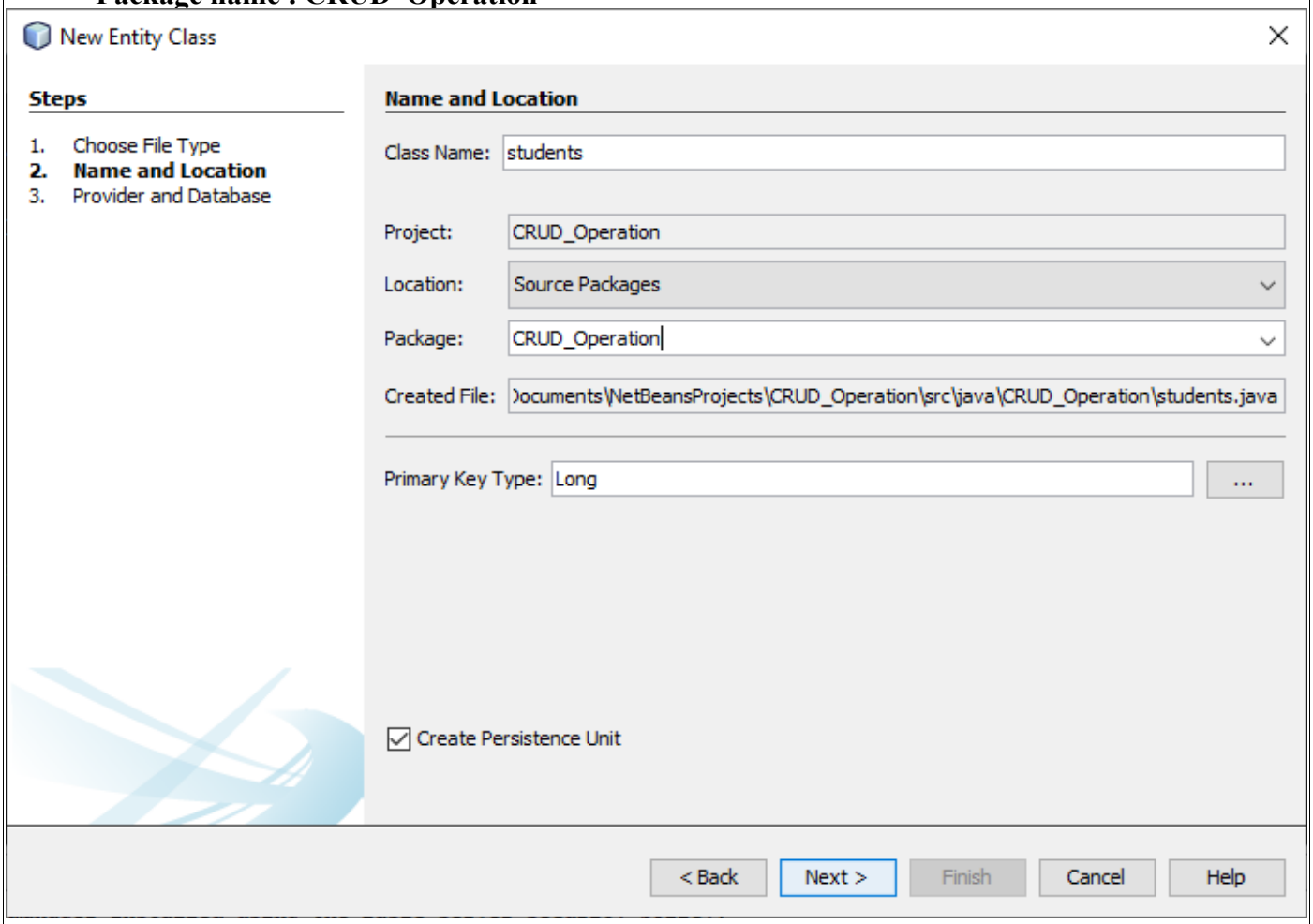
**4)** Create an entity class. **Right click on project name -> New -> Entity Class**.



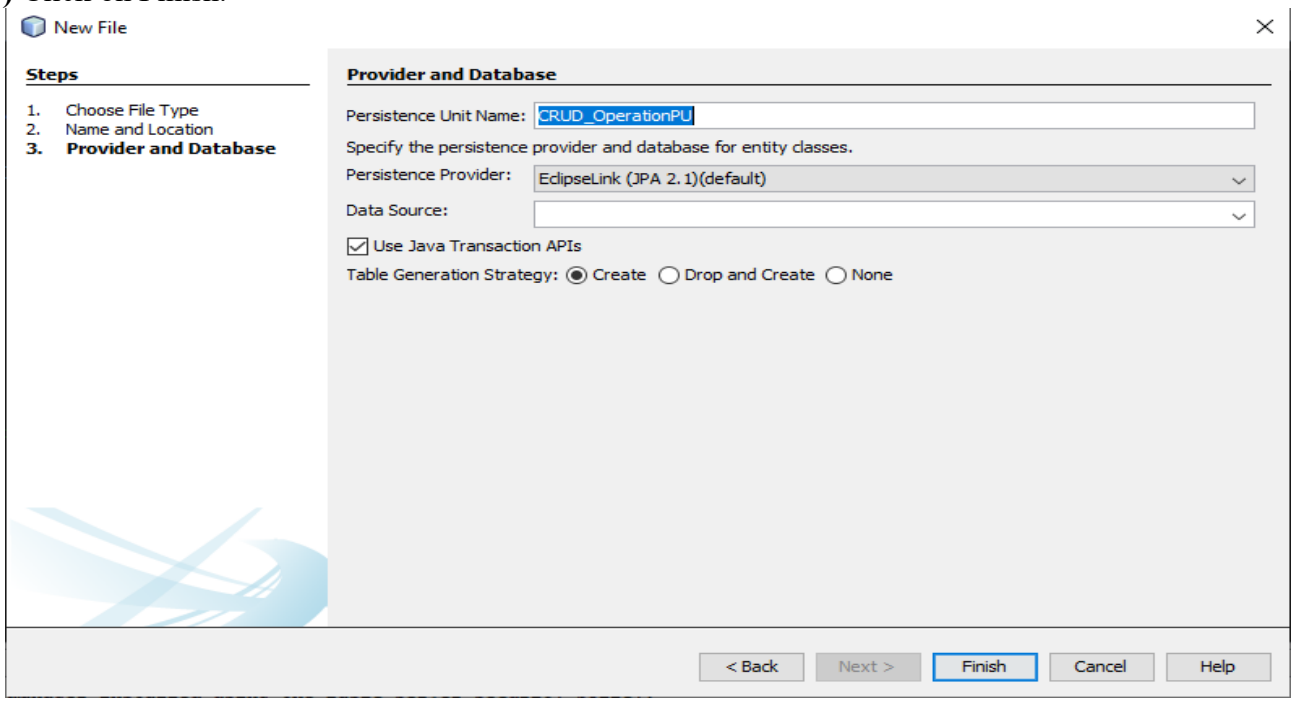

**5)** A window will appear like bellow pic. Enter following data and click on Next

**Class Name : students**

**Package name : CRUD_Operation**

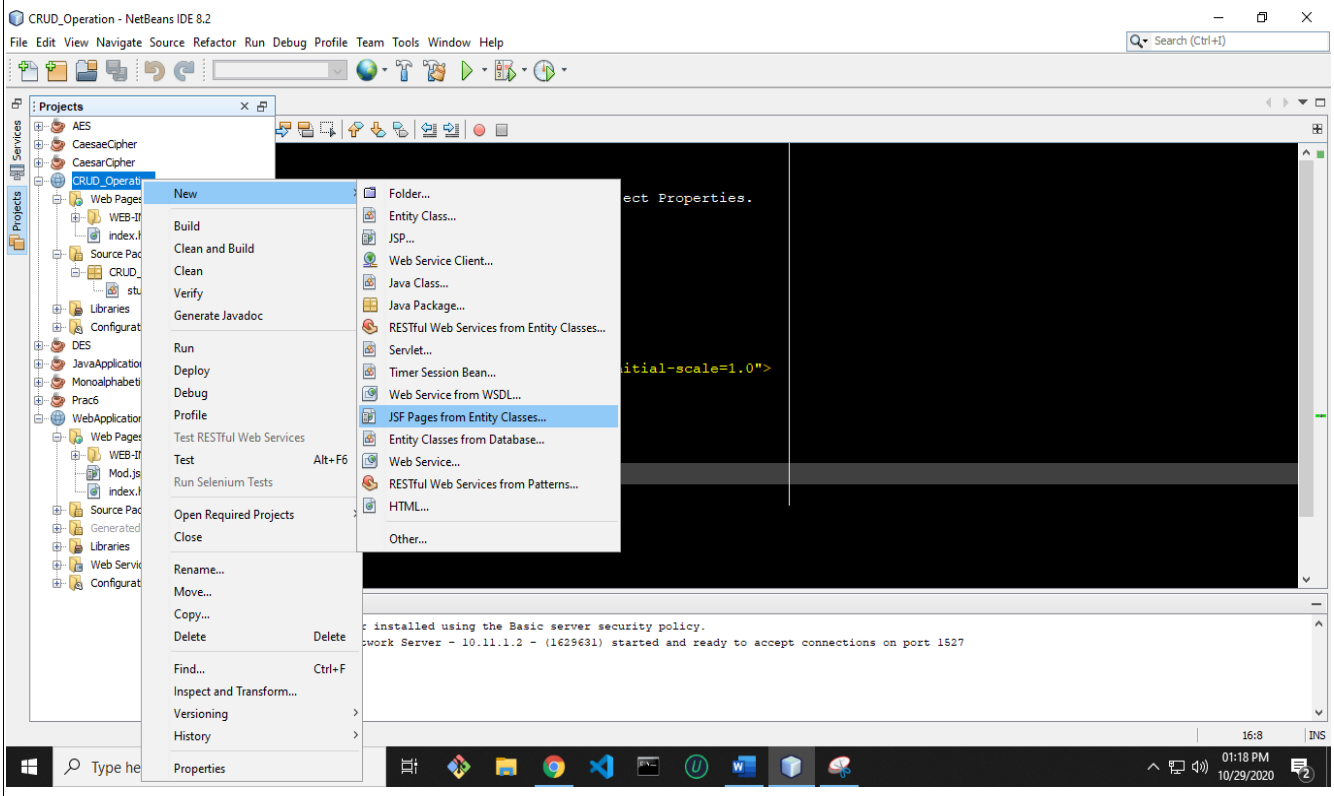


**6)** Click on Finish.

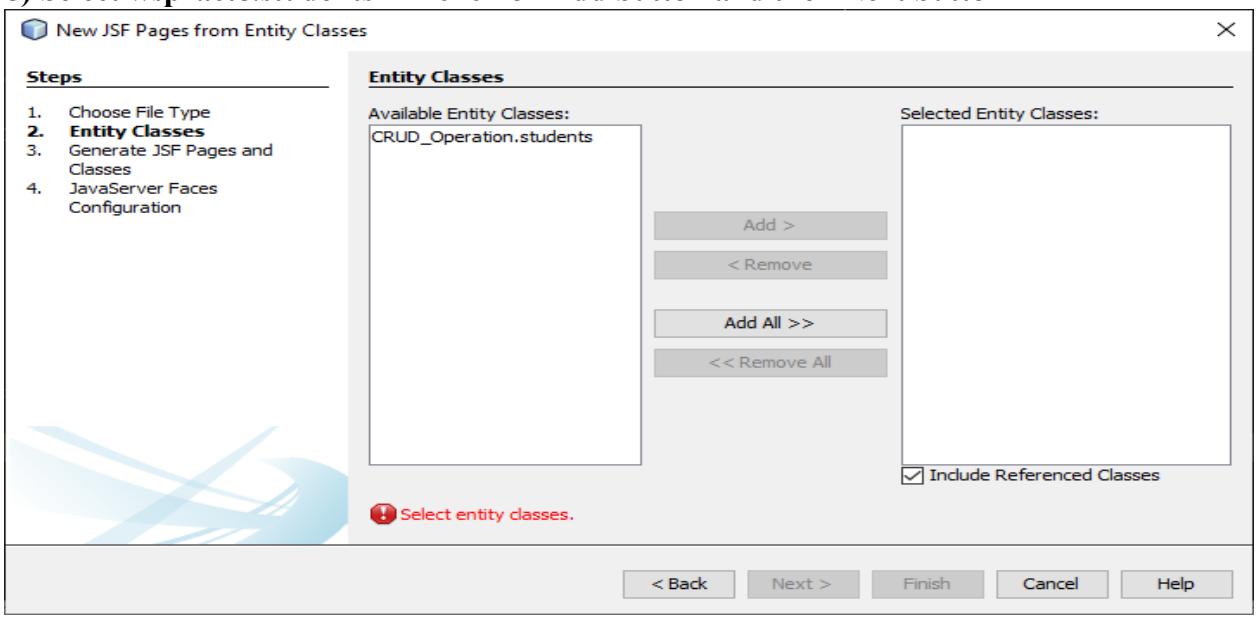


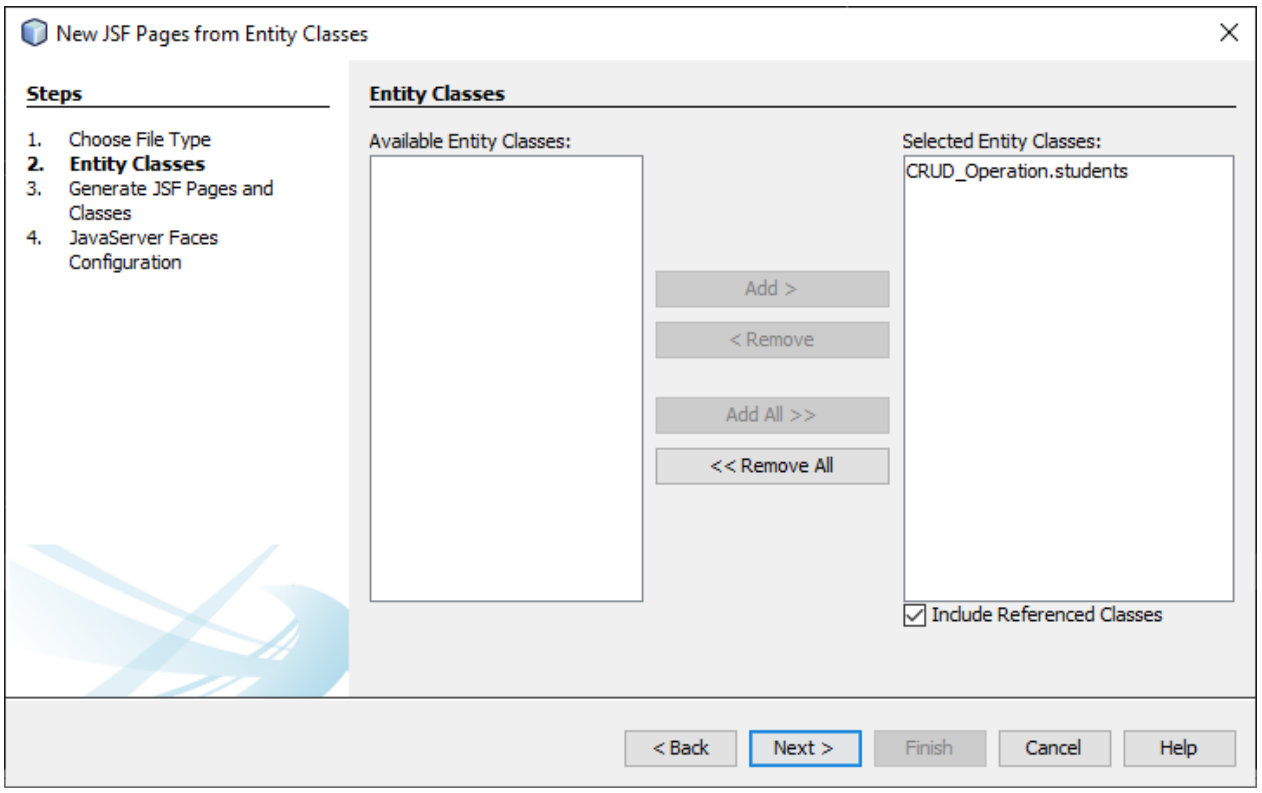

**7)** Right click on project name and create JSF Pages from Entity Classes.

**Right click on project name -> New -> JSF Pages from Entity Classes**

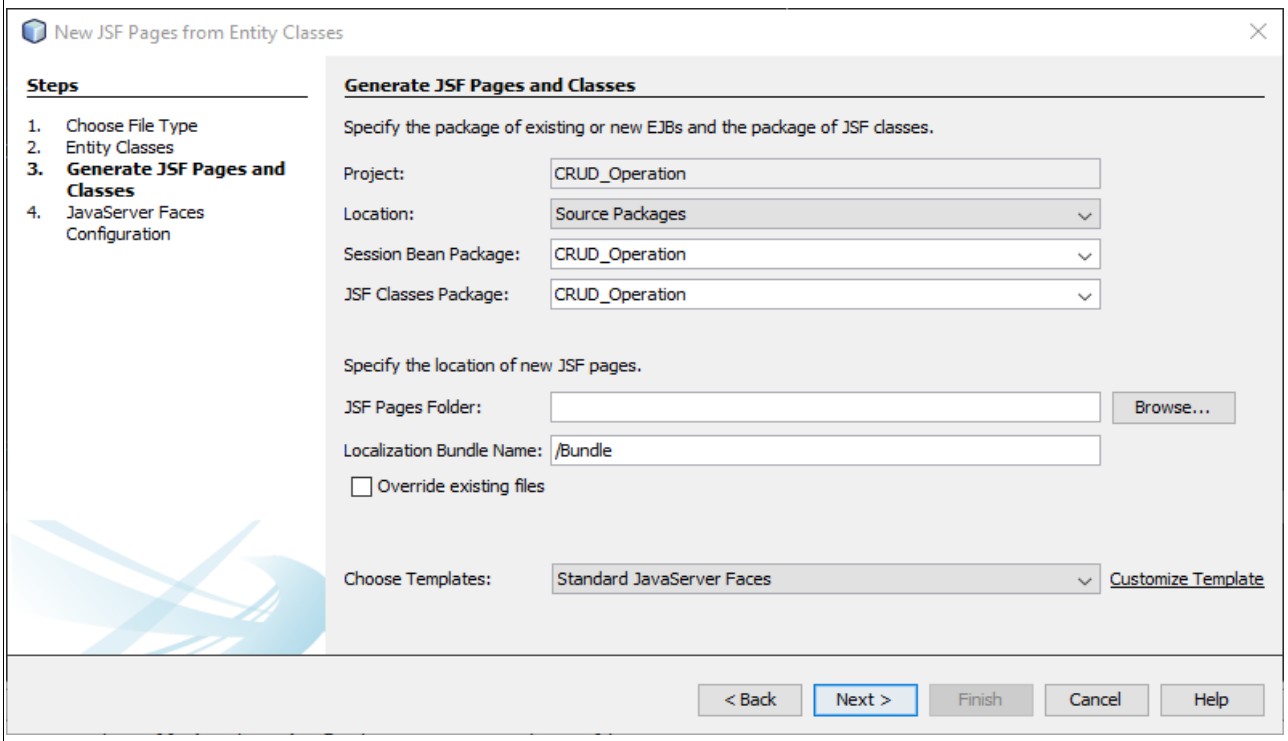


**8) Select wspract8.students** and **click on Add button and then Next button** on below.

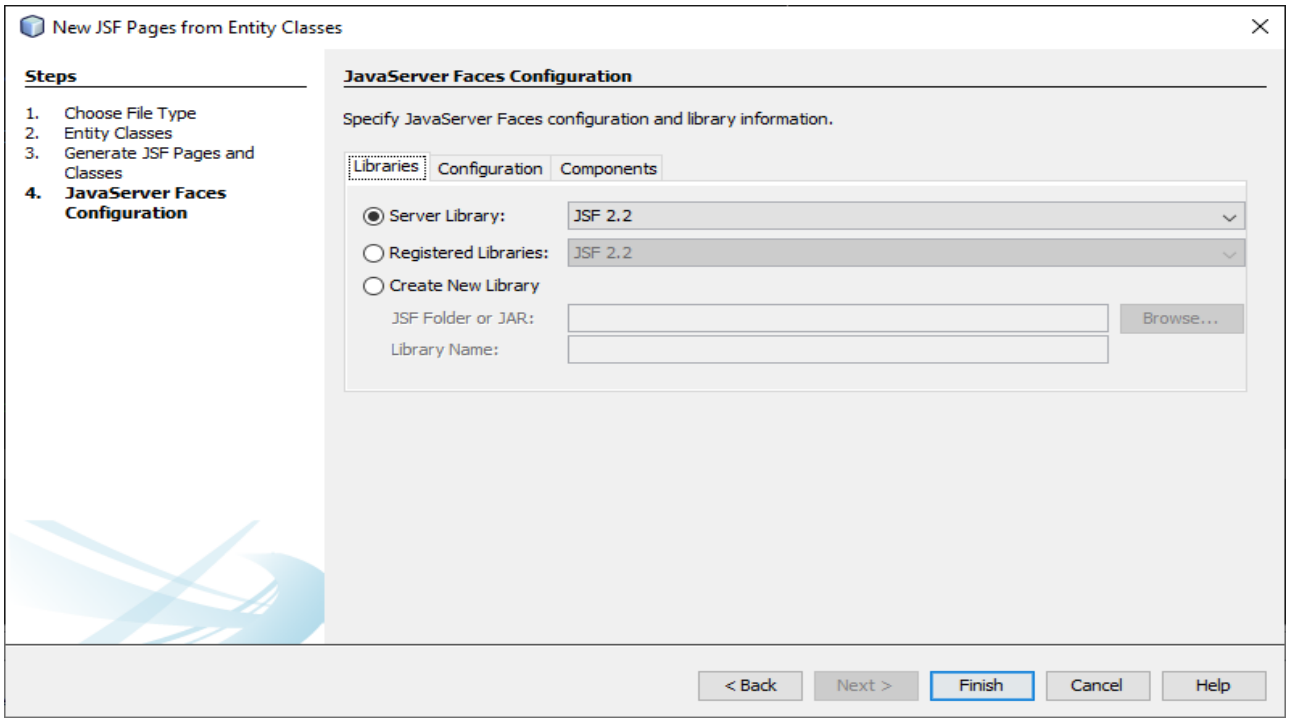


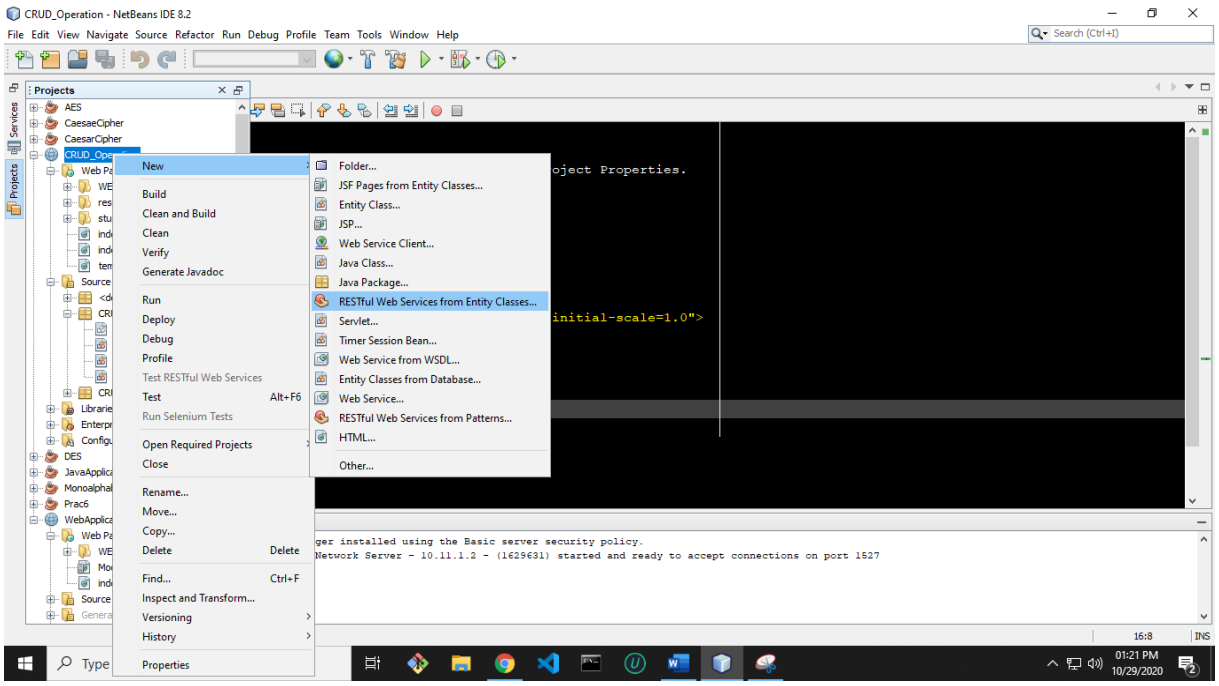
**9)**A window like below will appear on the screen. **Enter the data into that window as entered in below pic** and **click on Next button**



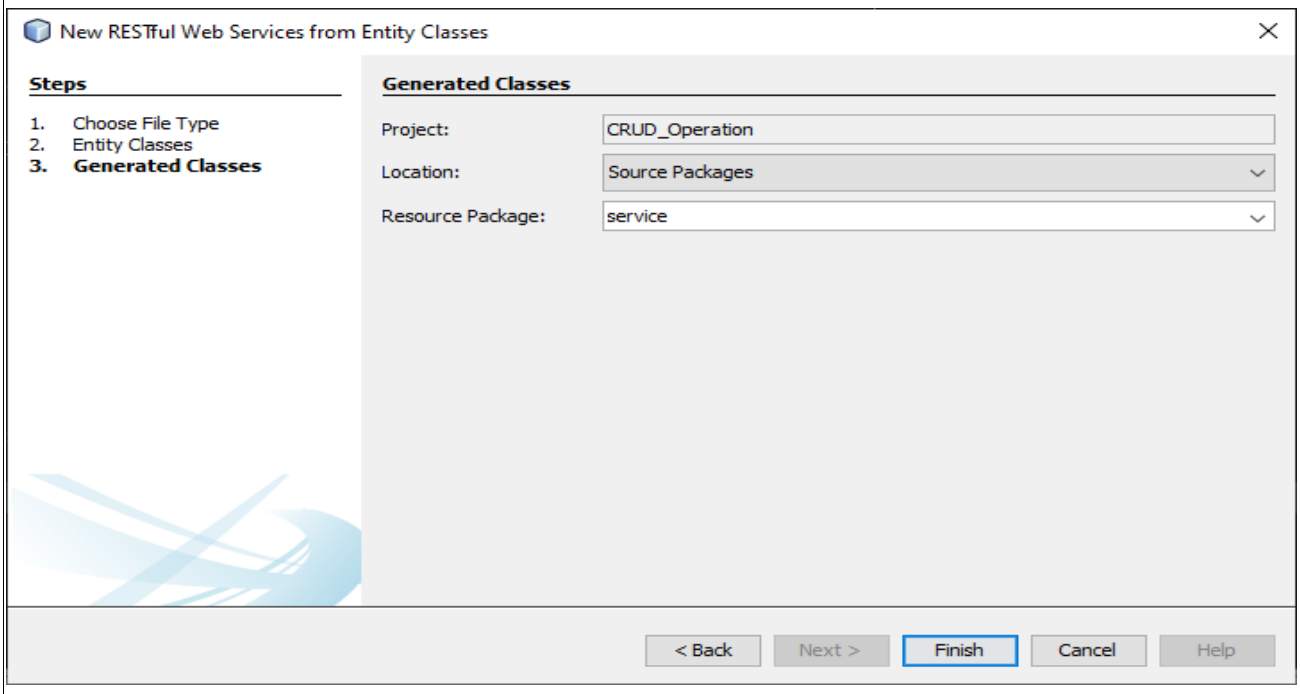

**10**)Now **click on Finish**



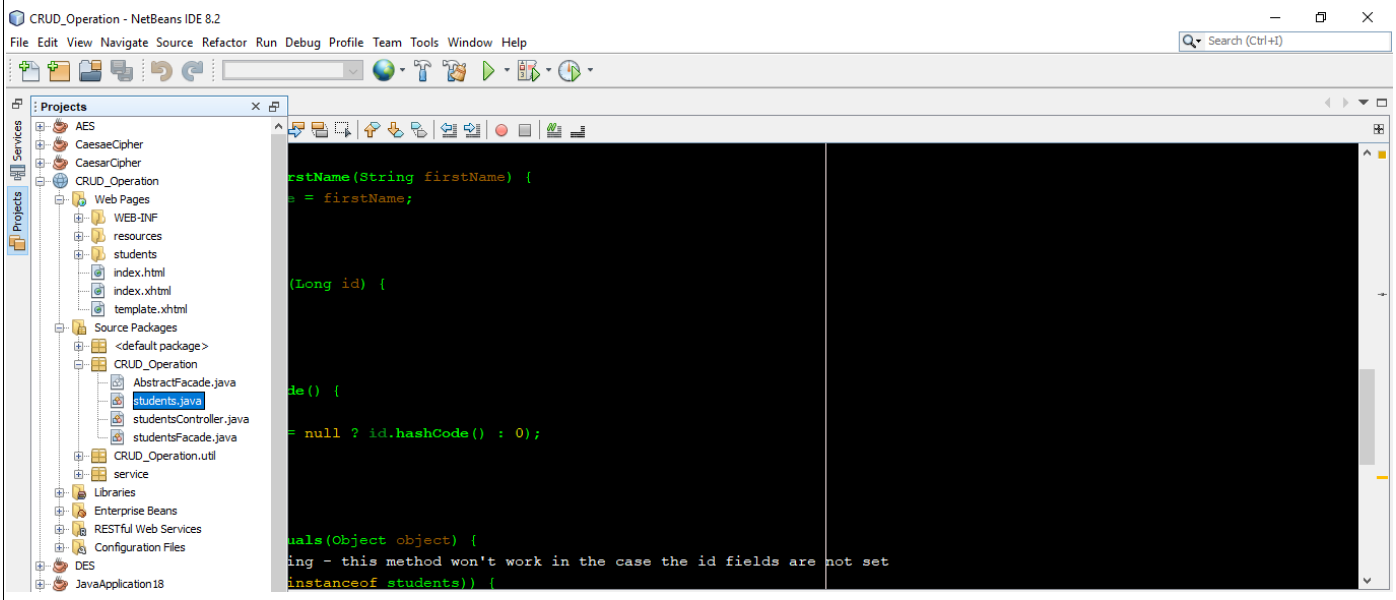
**11)** Right click on project name and create RESTful Web Services from Entity Classes.
**Right click on project name -> New -> RESTful Web Services from Entity Classes**



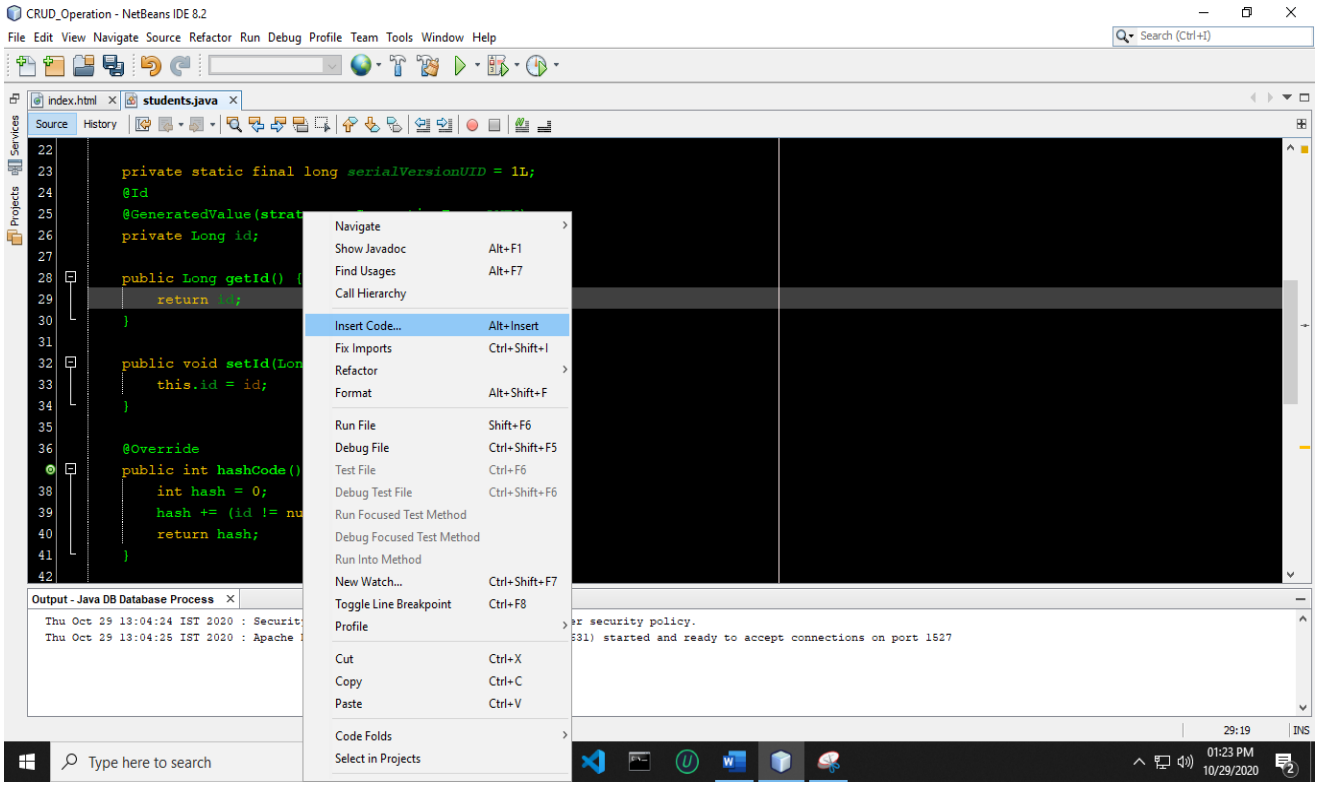

**12) Repeat step 9** and then it will go on next page. Then **enter the service** in Resource Package and then **click on Finish button.**



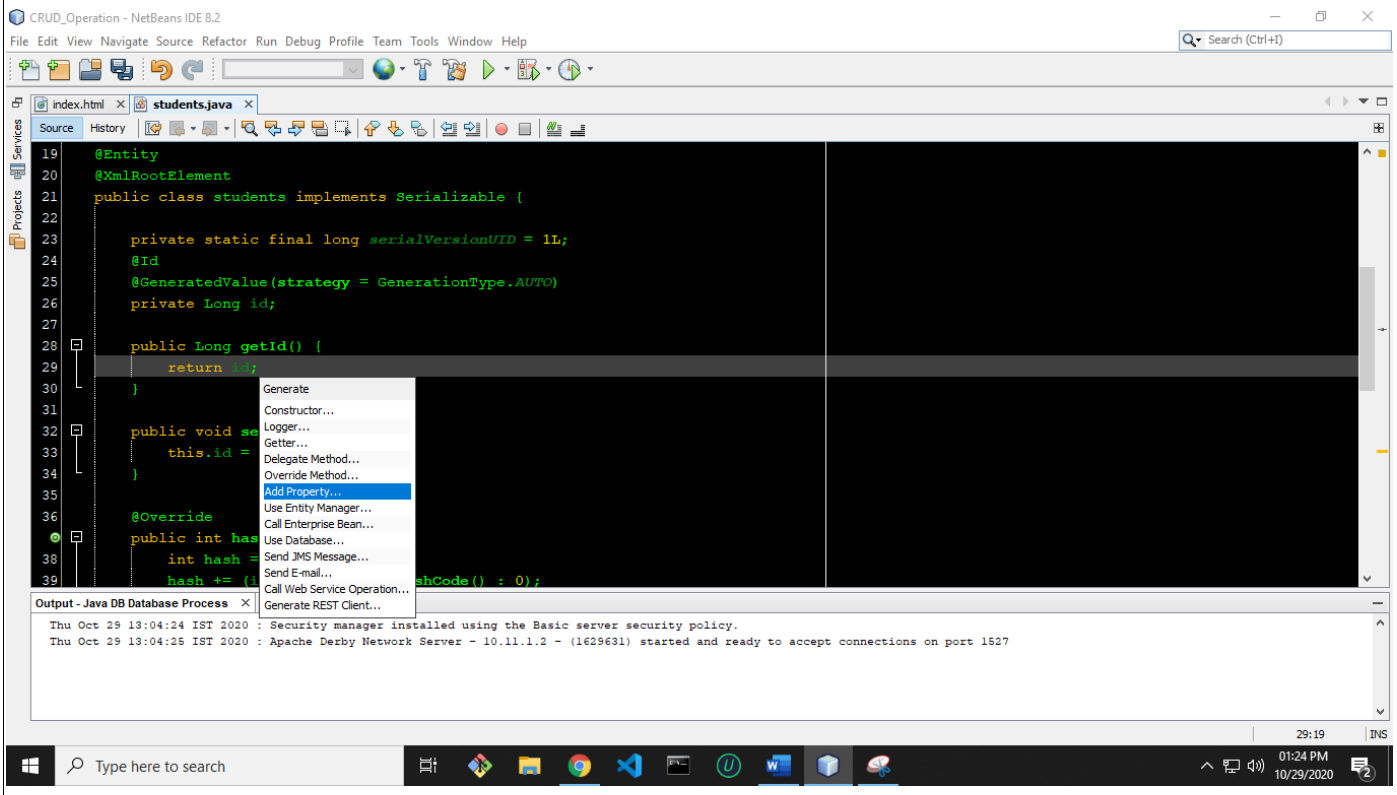
**13)** Now **open students.java file under wspract8 package.**



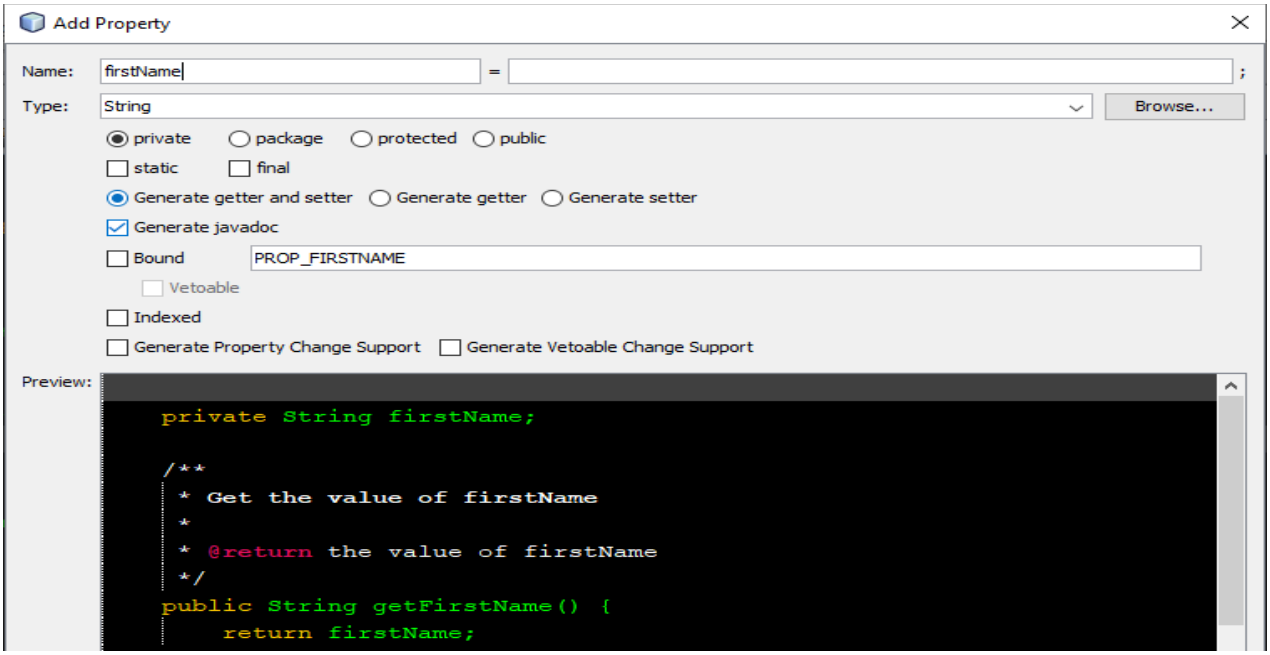

**14)** In this file **below line private Long id;** do the **right click and select Insert Code**.



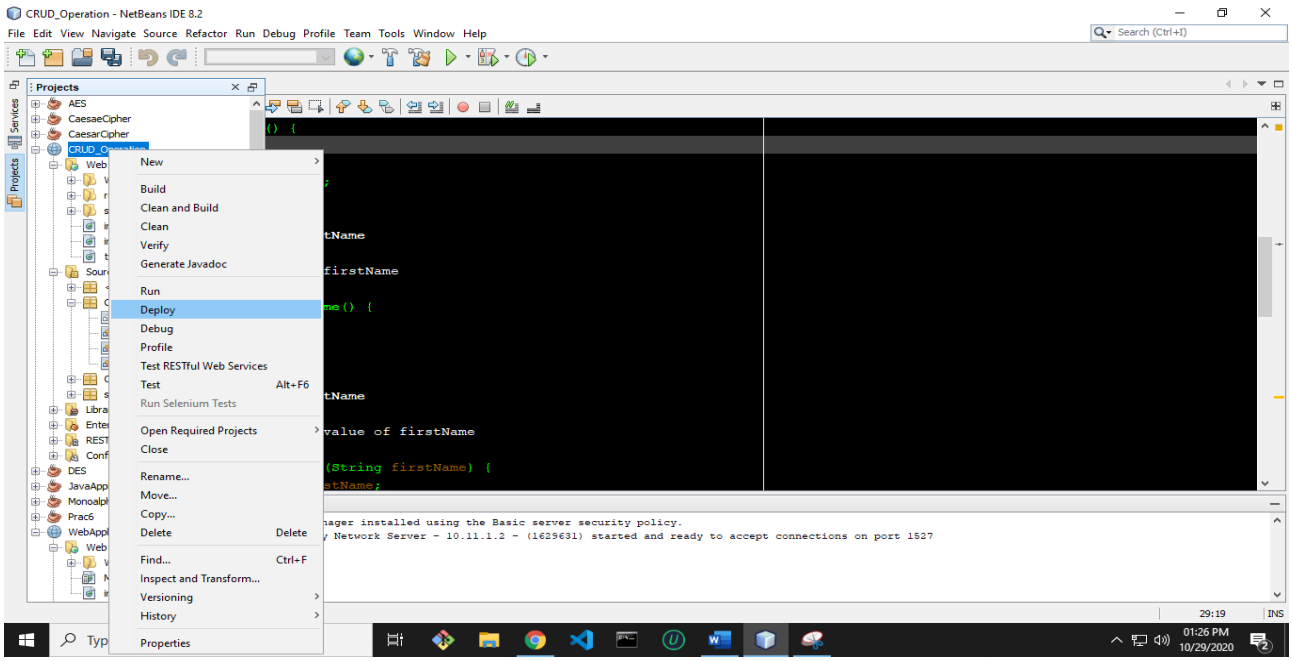
**15)** A new list will appear. **Click on Add Property**.



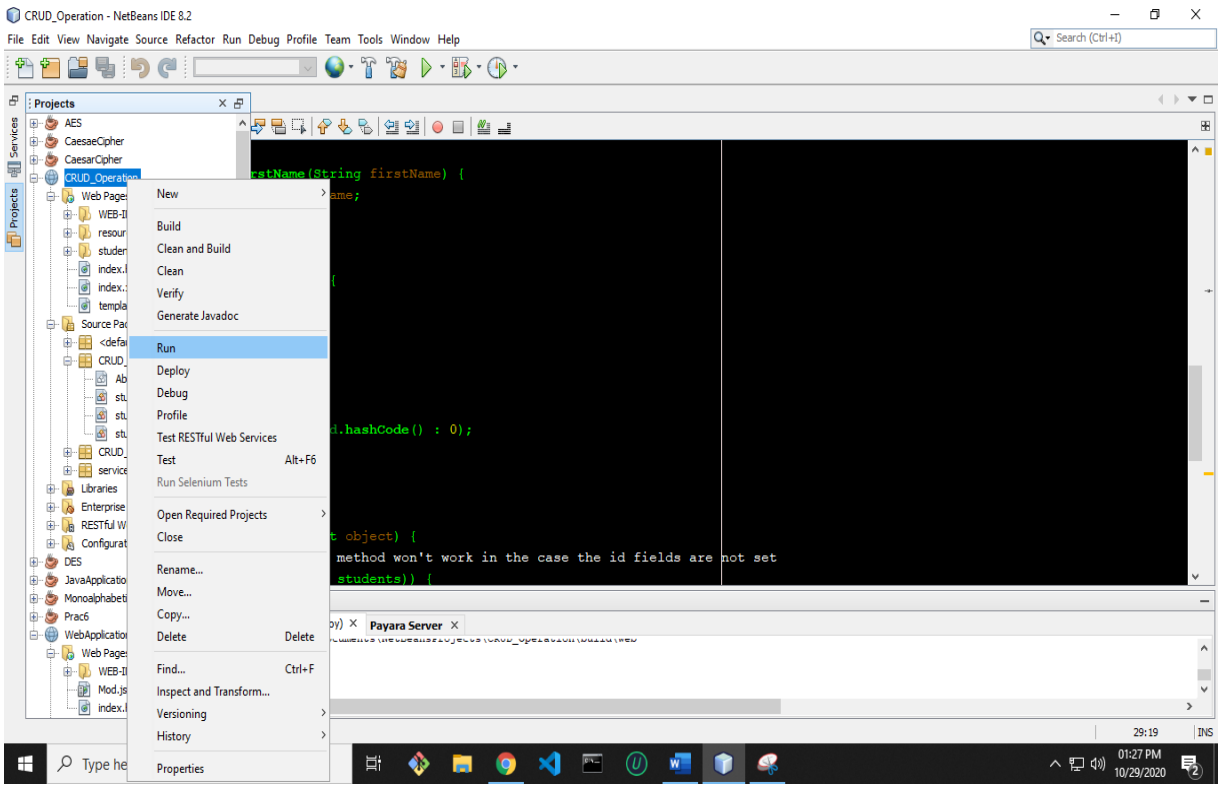

**16)** A new window will open. Enter name as **firstName**. Make sure name should be exact same as of mine and then **click on OK button**. Actually we are setting getter and setter method for firstName.



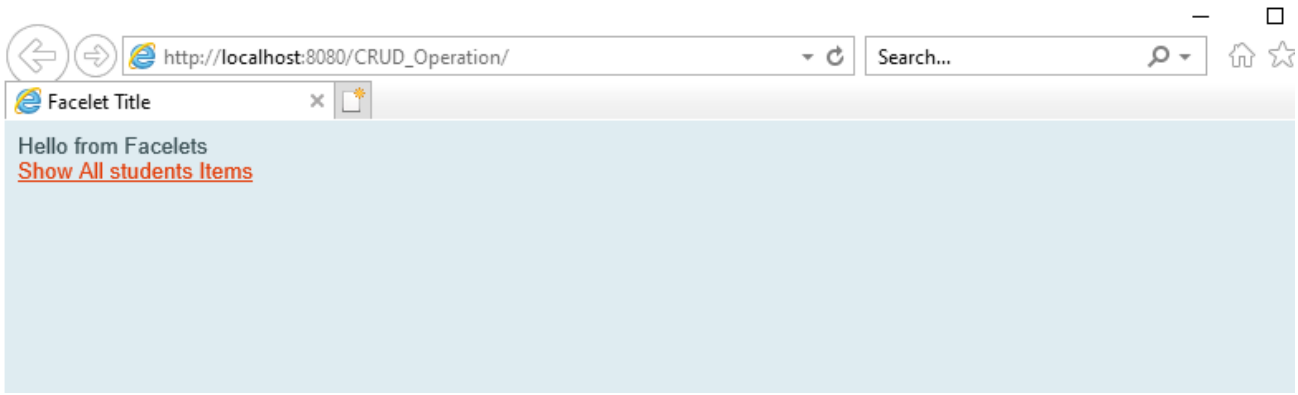
**17)** Now **right click on web application name and Deploy** it.



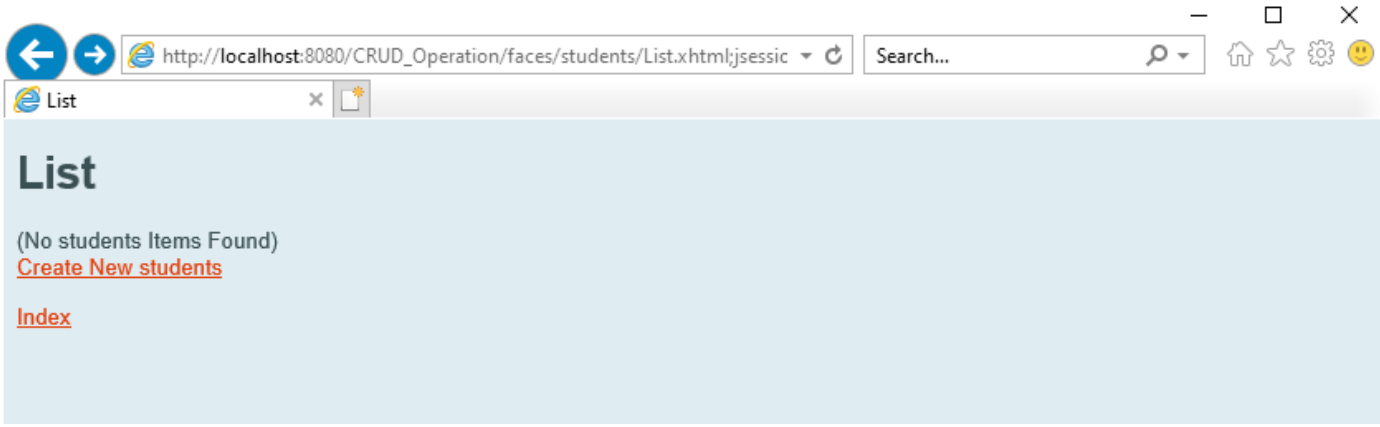

**18)** Now **right click on project name** and **run** it.

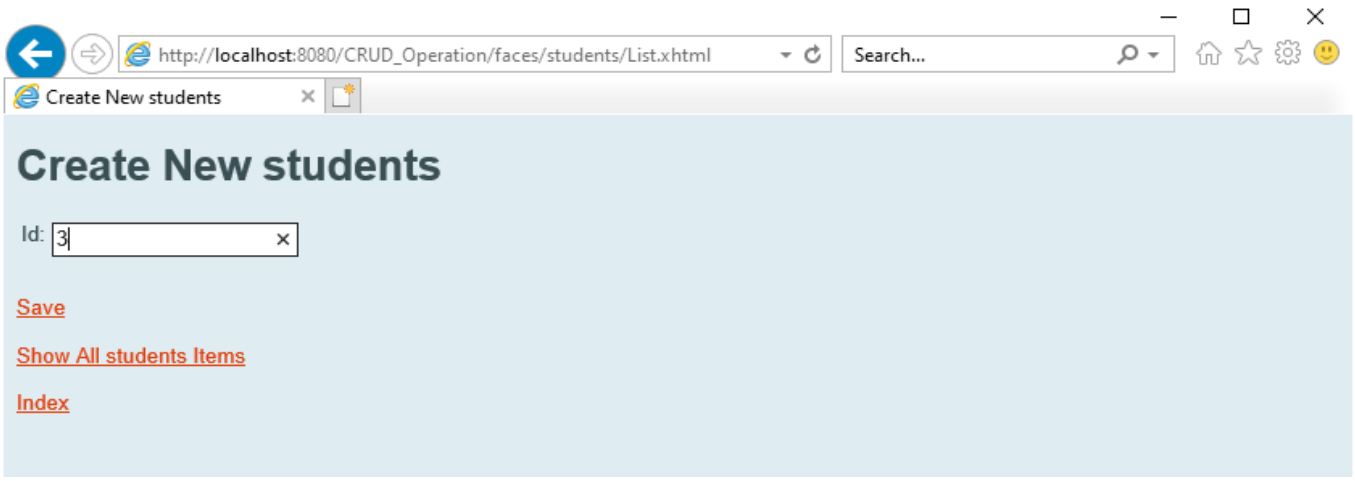


**19)** A window will open in browser like below.

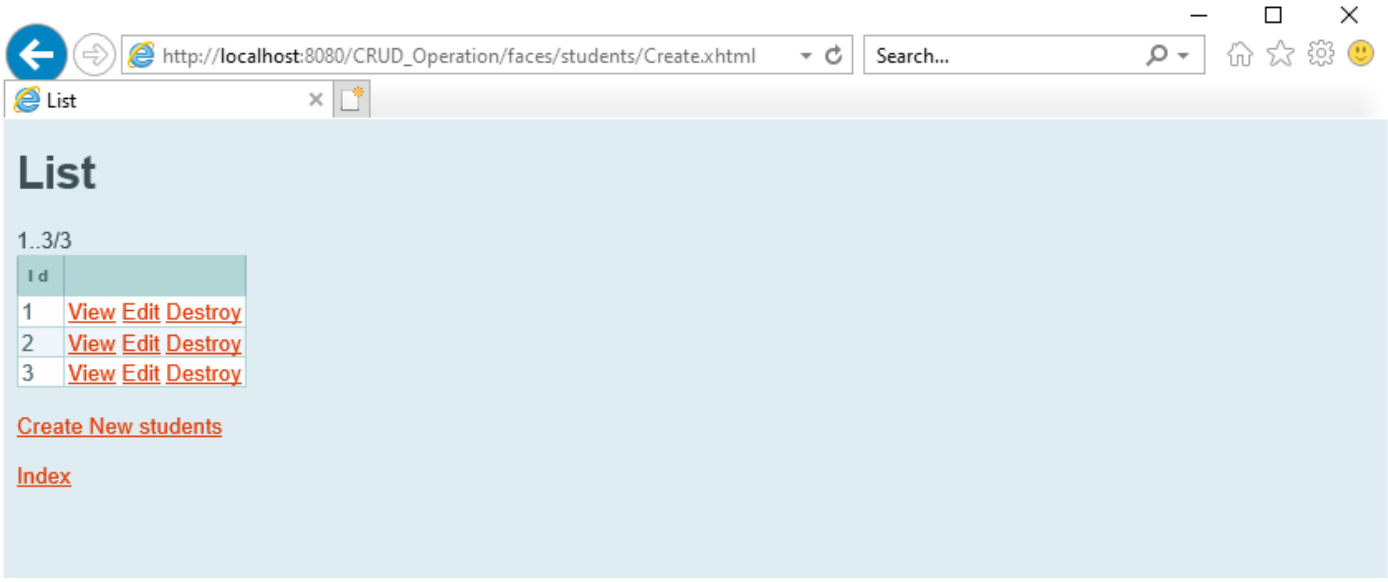



**20)** Now **click on Show All students** Items for CRUD operation.
Just click on Create New students. **Enter a name into FirstName and id into Id**. Now **click on Save** option to save the data.

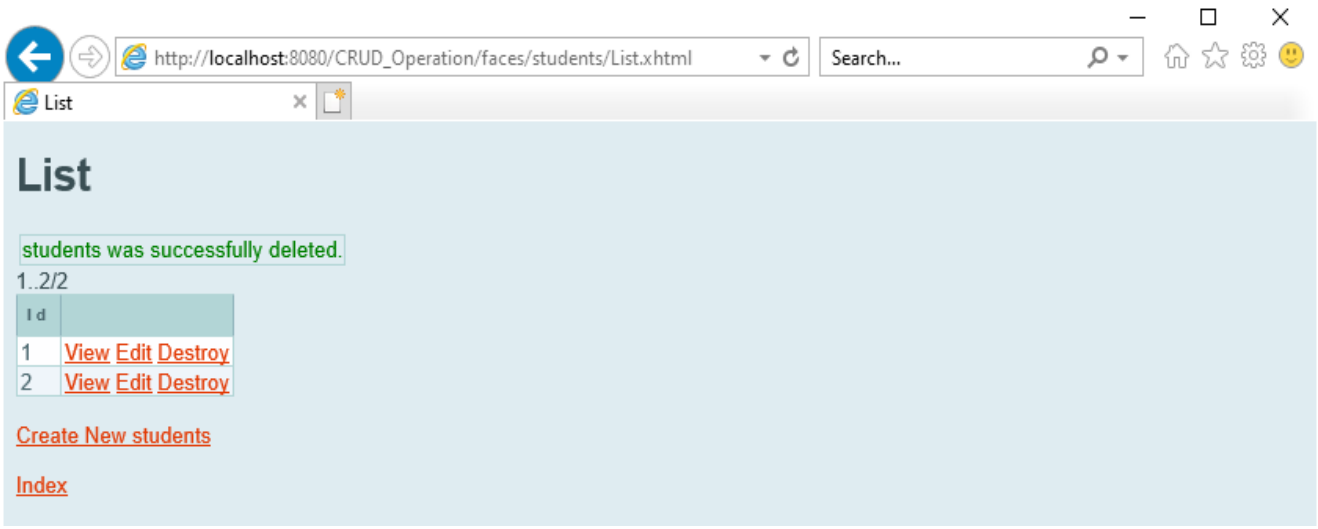


**21)** Now **click on Show All students Items to view all records** whether our data is entered or not.




**22) Delete the Record**



Date: 18/11/2020

## Practical no 9

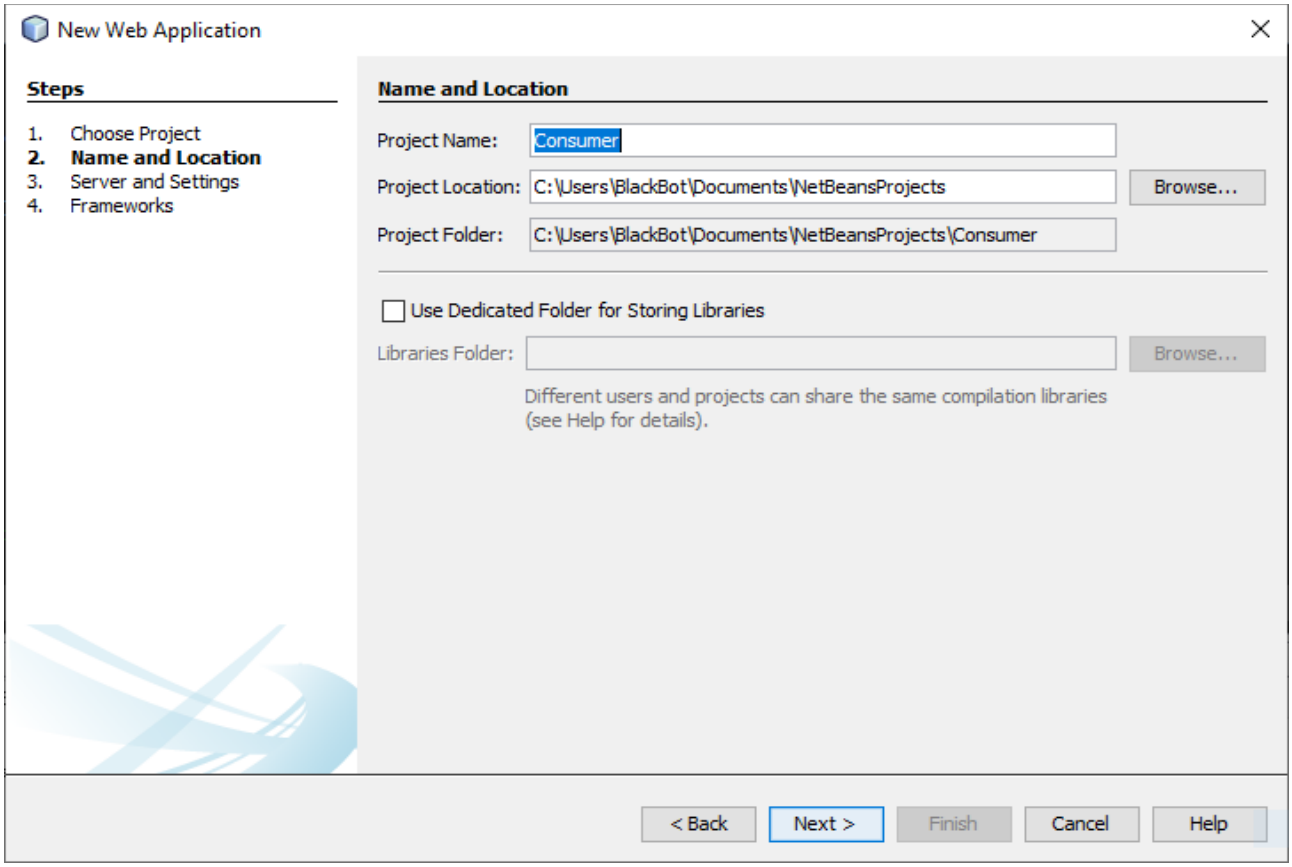
**AIM**: Write a JAX-WS web service to perform the following operations. Define a Servlet / JSP that consumes the web service.

**Note**: To do this practical, follow the steps from 1 to 18 present in practical – 7.

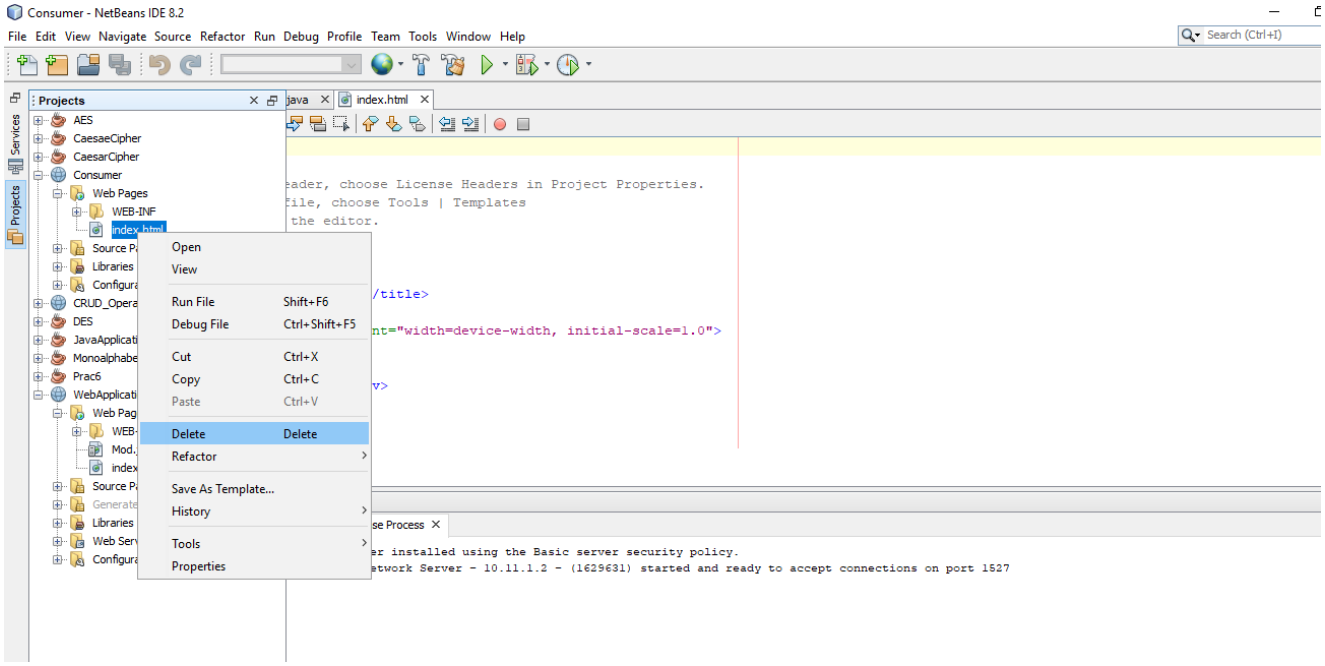
After that do not close or restart the NetBeans.

**Steps :-**

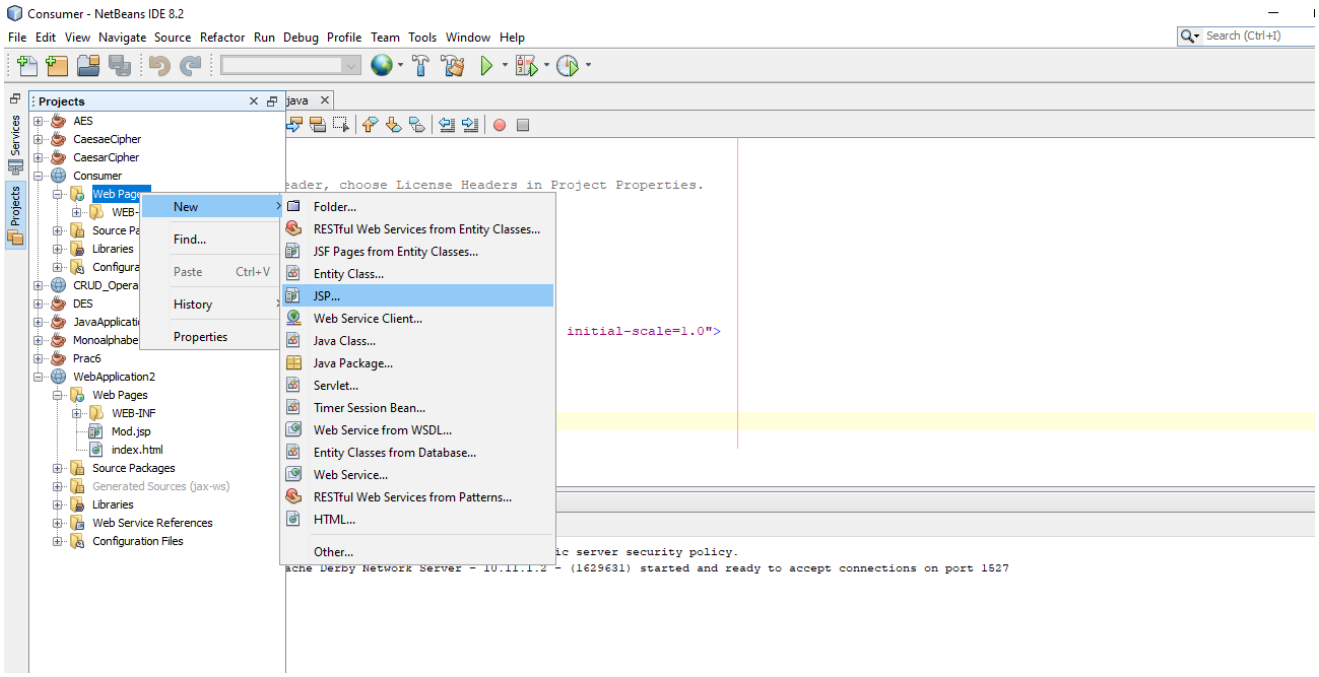
1) Create an another Web Application project and Give name as Consumer**.**



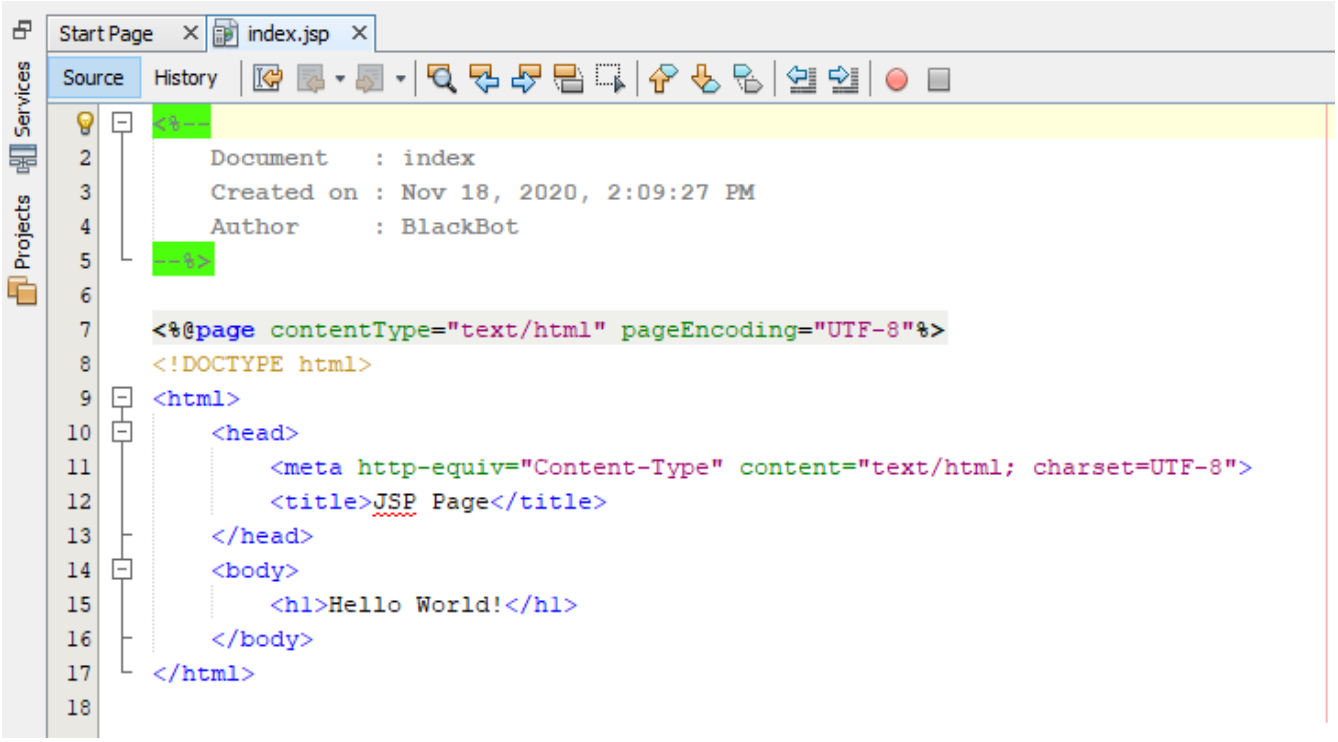
2) And create it. Next Finish.
3) Now right click on index.html and delete it.




4) Now right click on Web Pages and select JSP to add a JSP page.



5) Give index name to it and then click on Finish.
6) Now index.jsp page will open like below.




7) Now select HTML content of index.jsp file and replace it with following bold letter codes.

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<<HTML>
<HEAD>
<script language="javascript"> var xmlhttp;
function init()
{
xmlhttp=new XMLHttpRequest();
}
function readLocal(){ if(window.localStorage){
var seller=localStorage.getItem("seller");
seller=JSON.parse(seller);
document.getElementById("firstName").value=seller.firstName;
document.getElementById("sellerid").value=seller.id;
}
}
function saveLocal()
{
var sellerid=document.getElementById("sellerid");
```

```
var url="http://localhost:8080/CRUD_Operation/webresources/com.kk.seller/"+ sellerid.value;
xmlhttp.open('GET',url,true); xmlhttp.send(null);
xmlhttp.onreadystatechange =function(){
if(xmlhttp.readyState===4){alert("6"+sellerid);
if(xmlhttp.status===200){alert("7"+sellerid);
var seller =eval("("+xmlhttp.responseText+")");
if(window.localStorage){
localStorage.setItem("seller",JSON.stringify(seller));
alert("information stored successfully"+seller.firstName);
}
else{ alert("notstored");
}}
else
alert("error");
}
}
}
</script>
</head>
<body onLoad ="init()">
<table>
<tr>
<td>Enter id:</td>
<td><input type="text" id="sellerid"/>
<input type="button" value="load employee in local browser" onClick="saveLocal()"/>
</td>
</tr>
<tr>
<td>read from local</td>
<td><input type="button" value="Send values" onClick="readLocal()"/></td>
</tr>
<tr>
<td>first Name:</td>
<td> <input type="text" id="firstName"/></td>
</tr>
<tr>
   performed by krunal 713
</tr>
</table>
</body>
</html>
```

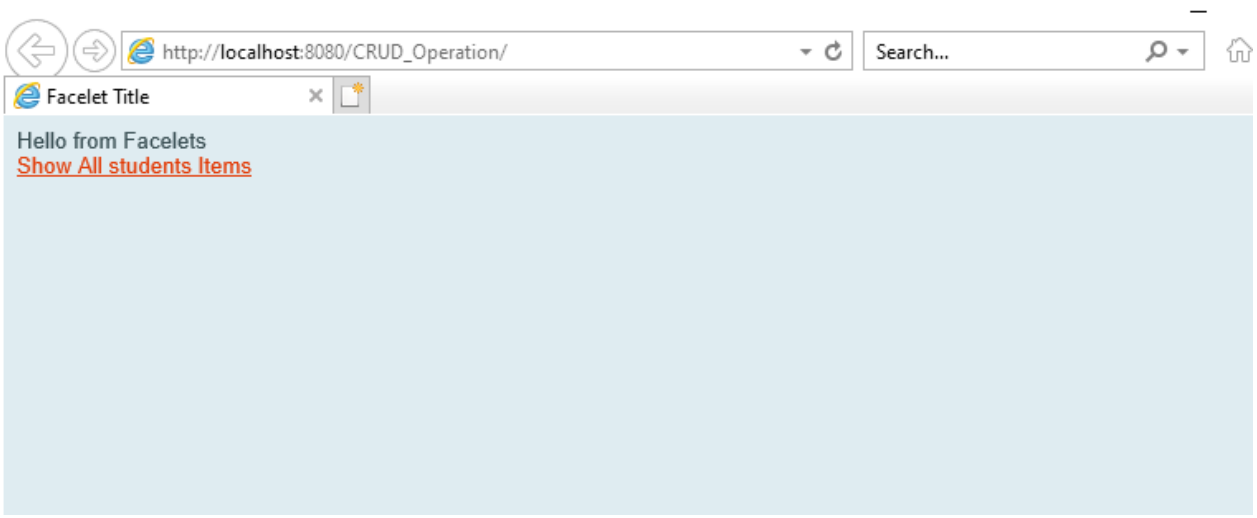
Start Page | index.jsp

Source | History

```
<%--
    Document   : index
    Created on : Nov 18, 2020, 2:09:27 PM
    Author     : BlackBot
--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<<HTML>
<HEAD>
<script language="javascript"> var xmlhttp;
function init()
{
xmlhttp=new XMLHttpRequest();
}
function readLocal(){ if(window.localStorage){
var seller=localStorage.getItem("seller"); seller=JSON.parse(seller); document.getElementById("firstName").val
}
```

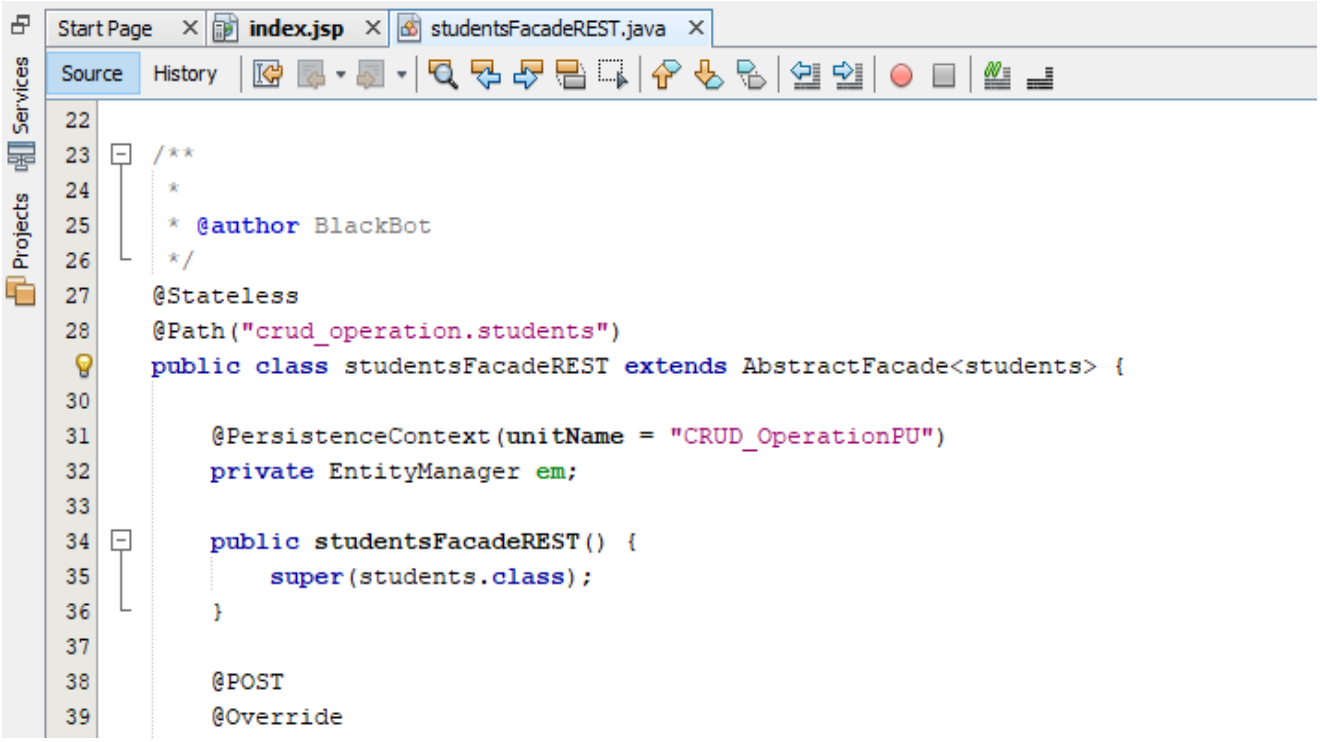
9) http://localhost:8080/CRUD_Operation/webresources/com.kk.seller/

Red part is the URL of which is obtained in browser by running of CRUD_Operation web application.

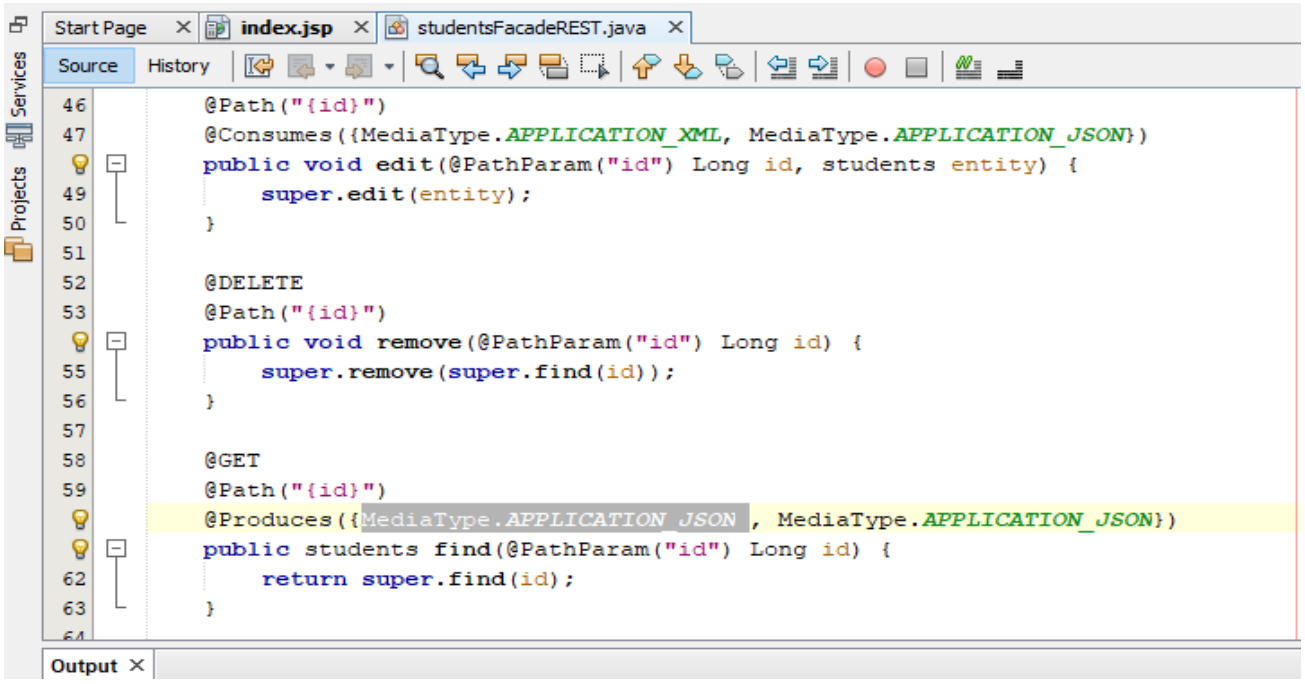


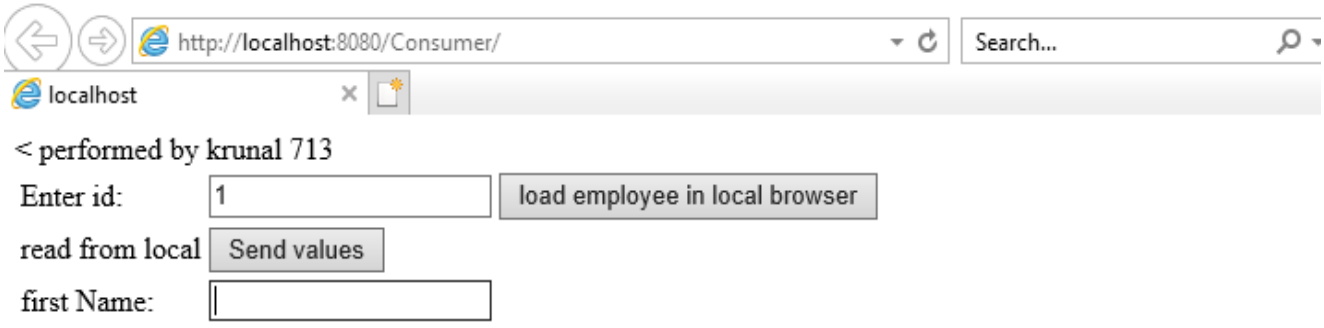

**Blue part** in above link is static. But red part is dynamic which is based on practical 7. So if you have changed name anywhere then the above URL will change accordingly.

10) Now open studentsFacadeREST.java file by follow below pic available in CRUD_Operation web application.




11) Now delete the selected part. Because it will return data in XML format to the consumer.
But we have written javascript in index.jsp for JSON data format only.
After that deploy the **CRUD_Operation** web application; so that it will update the changes.




12) Now run the Consumer application. A window will open in browser like below.

13) Now if you will enter an id into Enter id textbox which you have created in database (id from data in last step of practical 7) and then click on load employee in local browser button. It will get the detail of particular entered id and will store it into local storage of browser. Now click on Send Values to get the first name of entered id.