

**Name: - Rohan Babulnath Kamble**

**Class: - T. Y. B. Sc. CS**

**PRN NO: - 2020420004**

**Subject: - Web Services (USCSP505)**

## **INDEX**

<b>Sr. No.</b>	<b>List of Practical</b>	<b>Date</b>	<b>Page No.</b>	<b>Signature</b>

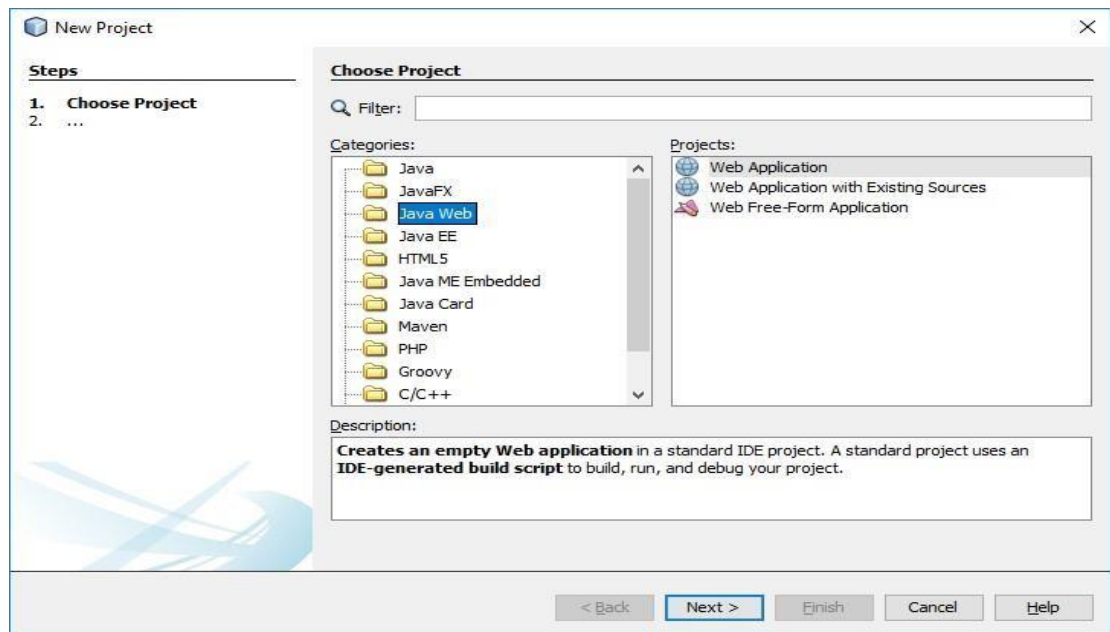
1.	Write a program to implement to create a simple web service that converts the temperature from Fahrenheit to Celsius and vice a versa.	11/10/2022	3 - 23	
2.	Write a program to implement the operation can receive request and will return a response in two ways. a) One -Way operation b)Request- Response	12/10/2022	24 - 67	
3.	Develop client which consumes web services developed in different platform.	13/10/2022	68 - 93	
4.	Define a web service method that returns the contents of a database and performs CRUD operation.	14/10/2022	94 - 121	

## Practical No: - 1

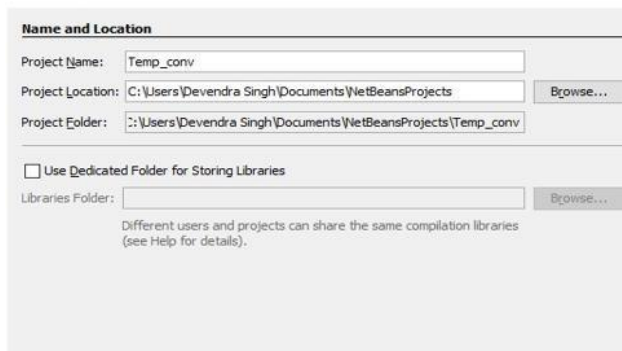
**Aim:** - Write a program to implement to create a simple web service that converts the temperature from Fahrenheit to Celsius and vice a versa. **Steps:**

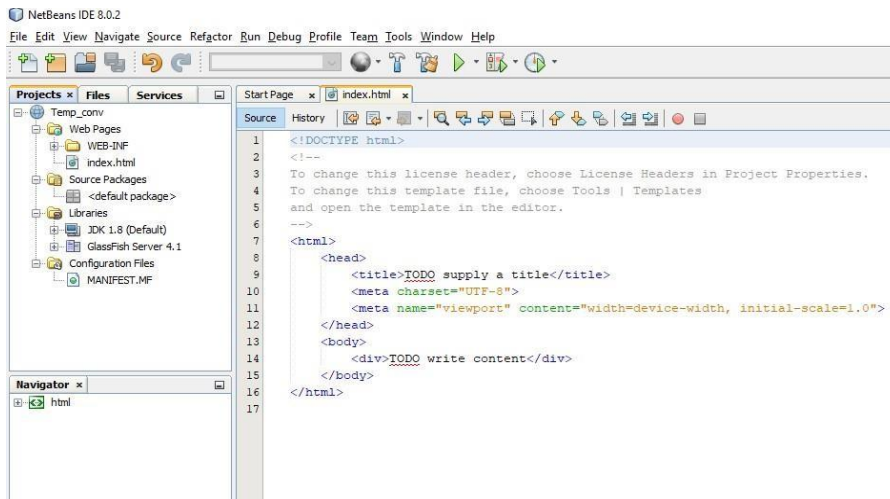
-

1. Go to **File -> New Project**. Select **Java Web in categories** and **Web Application in Projects**. Click on **Next** to create web based project.

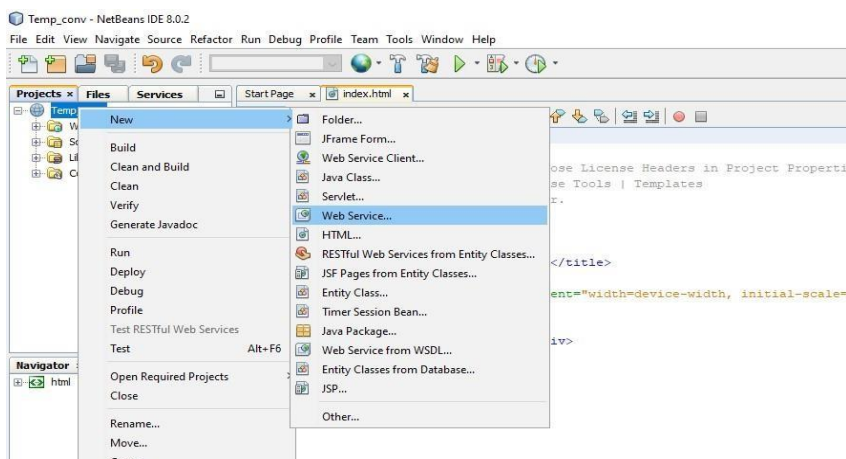


2. Enter a project name whatever you want and then click on **Next**. On next page click **Finish**.





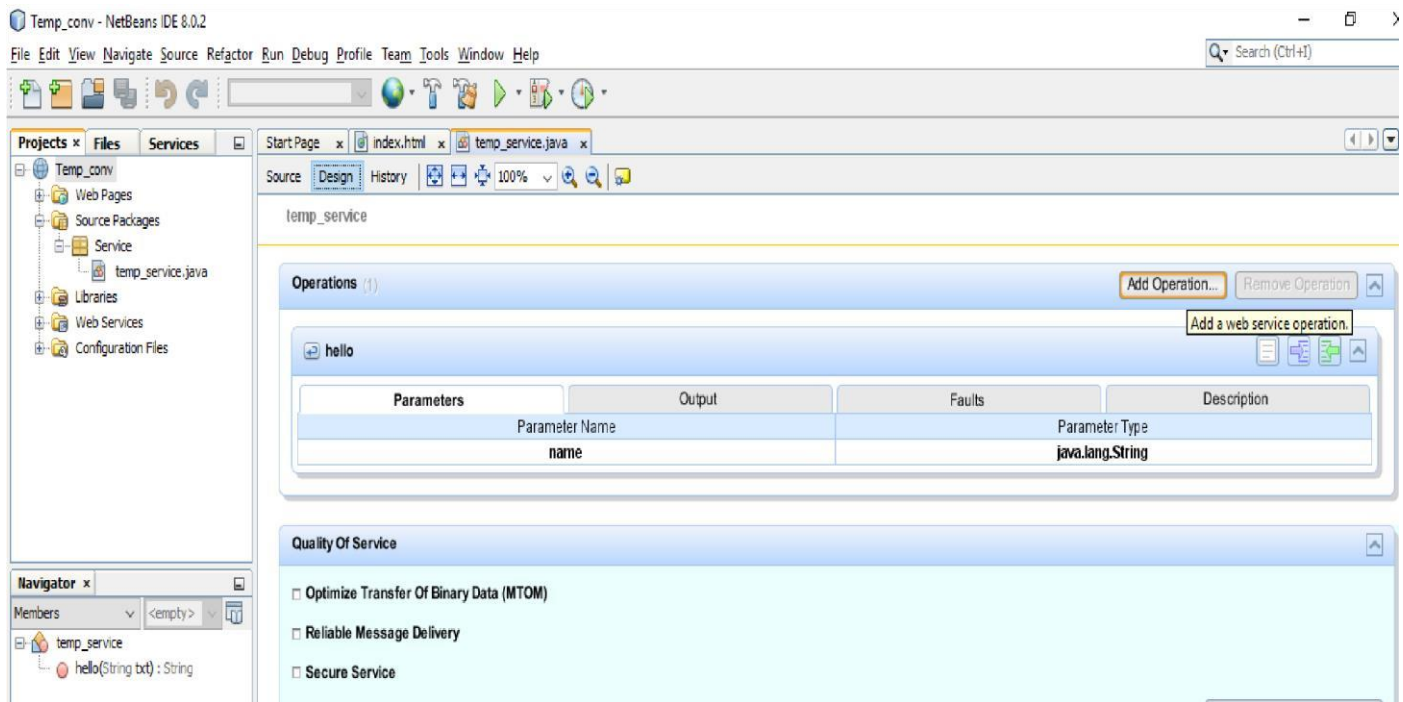
#### 4. Create web service.



#### 5. Enter a Web Service Name and package name and then click on Finish to create a Web Service.

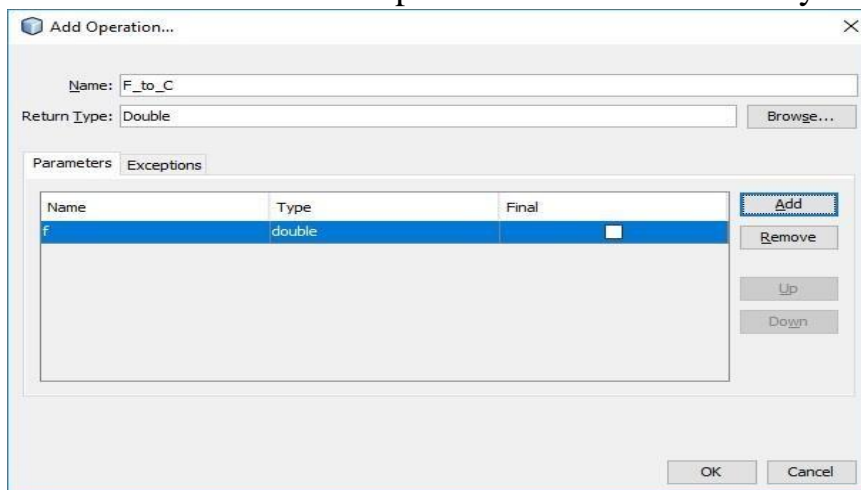
#### 6. As you can see in following pic; In **Source Packages** there is a package **Service** which contains the service file **temp\_service.java**. Open this file by double click on it, So that we can add two operation that will convert temperature from celcius to farhenheit and vice-versa.

Go to design mode by click on **Design** .



**7. Click on Add Operation to add operation.**

Give Operation name **F\_to\_C** and return type as **Double**. So this method will return value in Double data type. After that click on **Add** button to give parameters for method. Give its name as **f** and type as **Double** and then click on **OK** button. Your one operation is now successfully created.



**8. Repeat step 7 & 8 to create second operation. But this time replace some above entered data with following data.**

**F\_to\_C -> C\_to\_F**

**f -> c**

Now go to source mode by click on **Source** and **delete the selected** segment of code. Because it is default operation and we don't need this.

Now **replace the selected area with following code** to convert Fahrenheit to Celsius.

Double c = (f-32)\*1.8; return c;

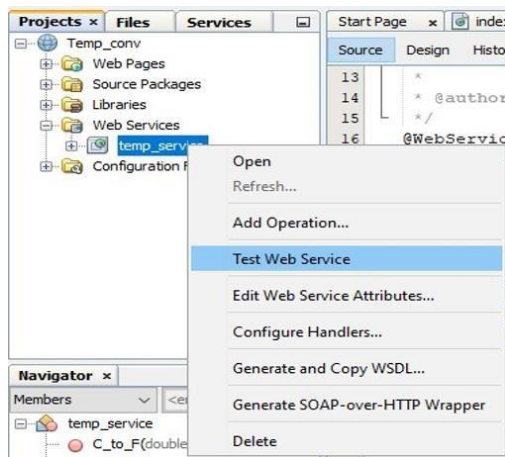
```
/**
 * Web service operation
 */
@WebMethod(operationName = "F_to_C")
public Double F_to_C(@WebParam(name = "f") double f) {
    Double c = (f-32)*1.8;
    return c;
}

/**
 * Web service operation
 */
@WebMethod(operationName = "C_to_F")
public Double C_to_F(@WebParam(name = "c") double c) {
    //TODO write your implementation code here:
    return null;
}
```

9. Now **replace the selected area with following code** to convert Celsius to Fahrenheit and then press Ctrl+S to save.

Double f = (c\*1.8)+32; return  
f;

10. Now **right click on project name** and **click on Deploy** to deploy your project.
11. To test your web service follow the following process as in picture.
12. Following window will open in in browser. Now if you will enter a numeric data into first box and you will click on first button it will



13. convert the entered data into Celsius.  
**temp\_service Web Service Tester**

This form will allow you to test your web service implementation ([WSDL File](#))

To invoke an operation, fill the method parameter(s) input boxes and click on the button labeled with the method name.

**Methods :**

public abstract java.lang.Double service.TempService.fToC(double)

---

public abstract java.lang.Double service.TempService.cToF(double)

Similarly second textbox and button will convert the numeric value into Fahrenheit. Now to consume this web service we are creating a client.

14. Now create a new web application project with **name** as **Temp\_client**.
15. Now **open the index.html** page of Temp\_client and write the following code into that.

```
<html>
  <head>
    <title>Temperature converter</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=devicewidth,
initialscale=1.0">  </head>
  <body>
    <form>
      <br><input type="text" name="data"><br>
<br><input type="submit" value="Convert F to C"
name="ftoc" formaction="f_to_c.jsp"><br>
      <br><input type="submit" value="Convert C to F"
name="ctof" formaction="c_to_f.jsp">
```

```

</form>
</body>
</html>

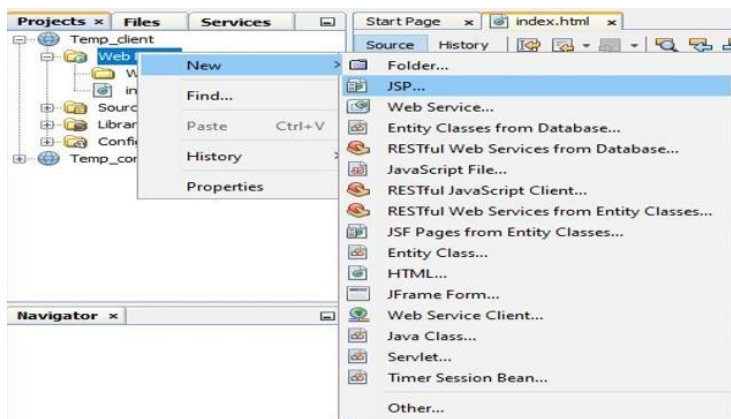
```

```

<html>
<head>
<title>Temperature converter</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
<body>
<form>
<br><input type="text" name="data"><br>
<br><input type="submit" value="Convert F to C" name="ftoc" formaction="f_to_c.jsp"><br>
<br><input type="submit" value="Convert C to F" name="ctof" formaction="c_to_f.jsp">
</form>
</body>
</html>

```

20. Now create two jsp pages for both submit button. **Right click on Web Pages > New -> JSP**



21. Enter file name **f\_to\_c** and then click on **Finish**.

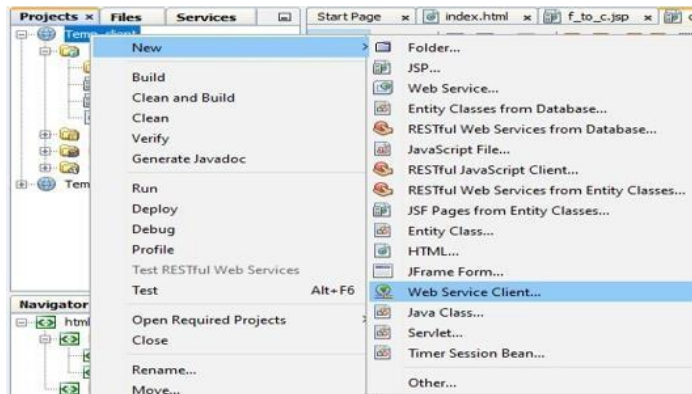
 A screenshot of the 'Name and Location' dialog box in NetBeans. The 'File Name' field contains 'f\_to\_c'. The 'Project' field shows 'Temp\_client'. The 'Location' dropdown is set to 'Web Pages'. The 'Folder' field shows 'WEB-INF' with a 'Browse...' button next to it. The 'Created File' path is displayed as 'C:\Users\Devendra Singh\Documents\NetBeansProjects\Temp\_client\web\WEB-INF\f\_to\_c.jsp'. Under the 'Options' section, the radio button for 'JSP File (Standard Syntax)' is selected, and the checkbox for 'Create as a JSP Segment' is unchecked. The 'Description' field contains the text 'A JSP file using JSP standard syntax.' At the bottom, there are buttons for '< Back', 'Next >', 'Finish' (which is highlighted), 'Cancel', and 'Help'.

22. Now repeat the step number 20 & 21. But give the File Name as **c\_to\_f**.

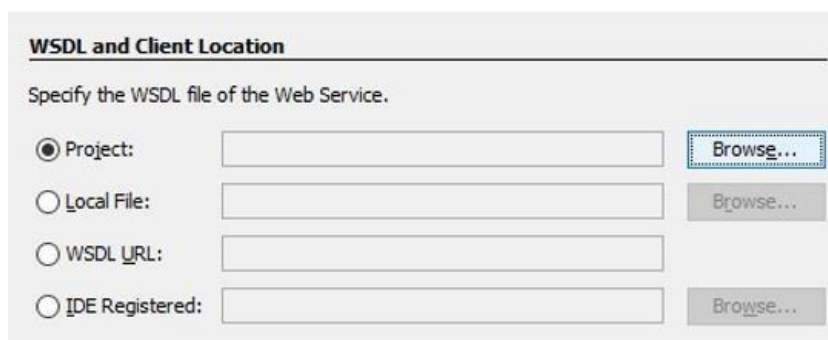


20. Now you have created two jsp files.

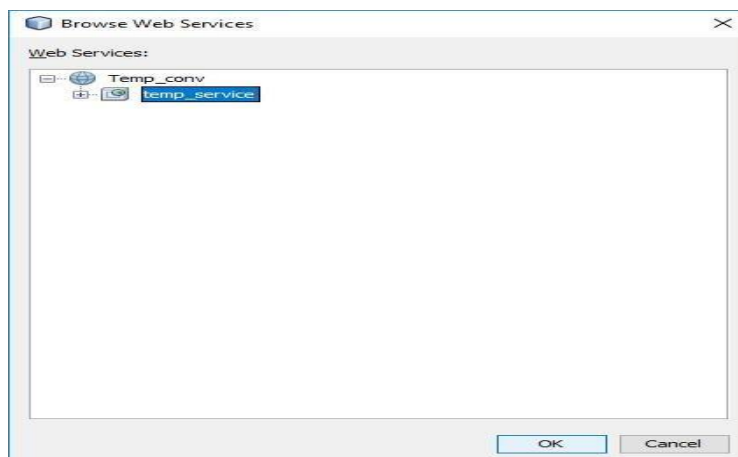
21. Right click on **Temp\_client** and select **Web Service Client** as below.



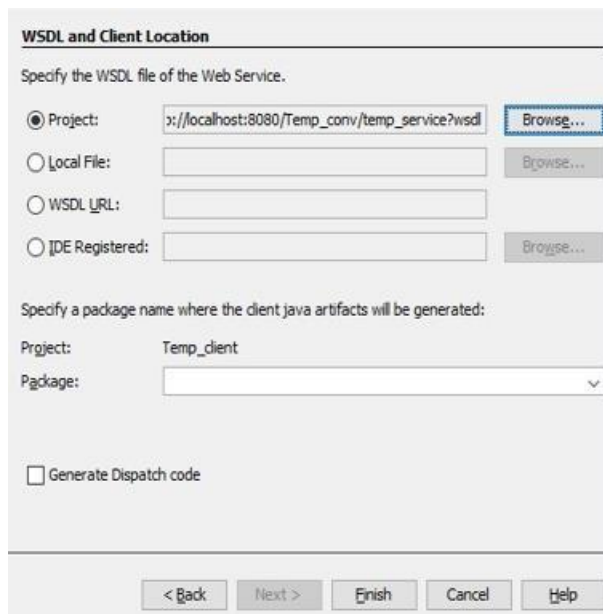
22. Click on **Browse**.



23. New window will appear and then select **temp\_service** and click on **OK**.



Click on **Finish**



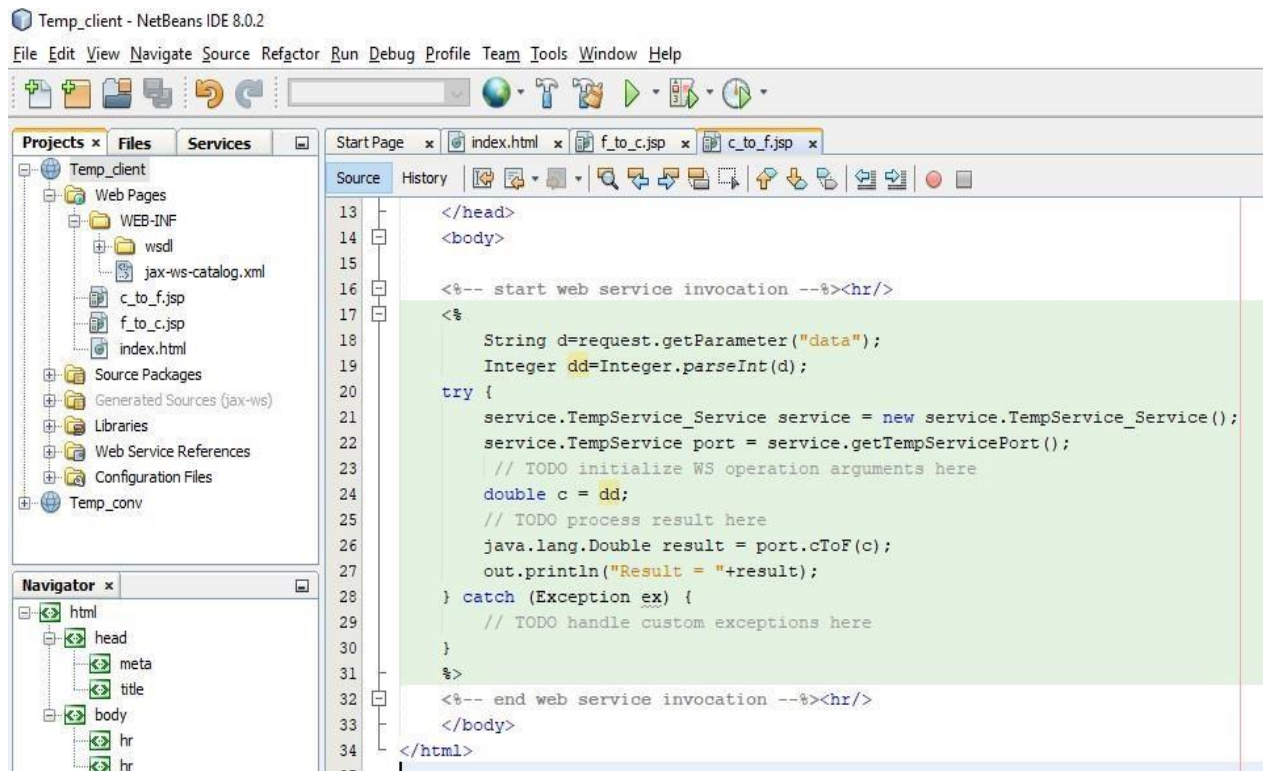
24. Now open the **c\_to\_f.jsp** file and delete the selected line.

25. Now right click in between the body section and select **Call Web Service Operation** as below.



26. New window will appear select the **C\_to\_F** by expanding it and click on **OK**. So that selected code in second pic will be automatically generated.

27. Now, make the selected area in step 30 as like selected area in below pic by adding some line of code.

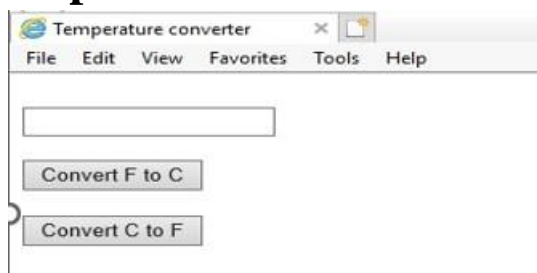


**28. Now Open the f\_to\_c.jsp file and follow the steps from 28 to 31.**

**Only the change is in 30 number step and i.e. instead of C\_to\_F, you have to select F\_to\_C.**

**29. Now run the Temp\_client project. An window will be open like below.**

### **Output:-**



**30. Now you have to enter any numeric data into textbox and if you will click the first button it will convert the numeric value into Celsius and vice-versa for the second button.**

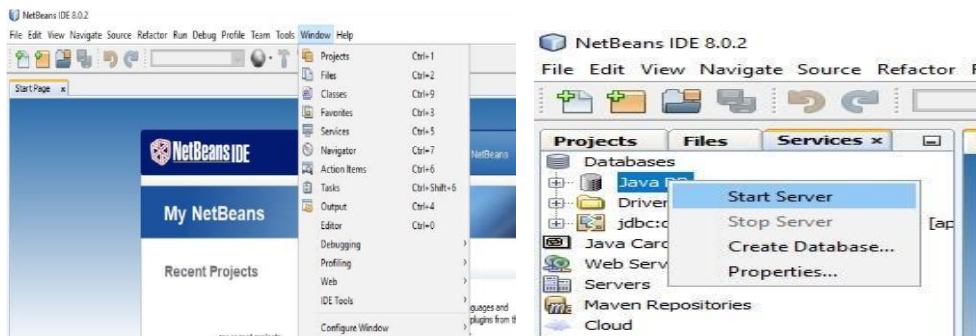
**31. There are so many methods to consume the web service. But I found it easy.**

## **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.

**Steps:** -

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 .

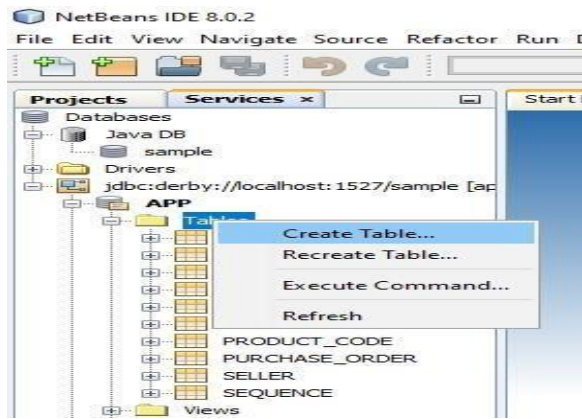


3. Now expand Java DB and **right click on sample** and then click on **connect** to connect the sample database with server.

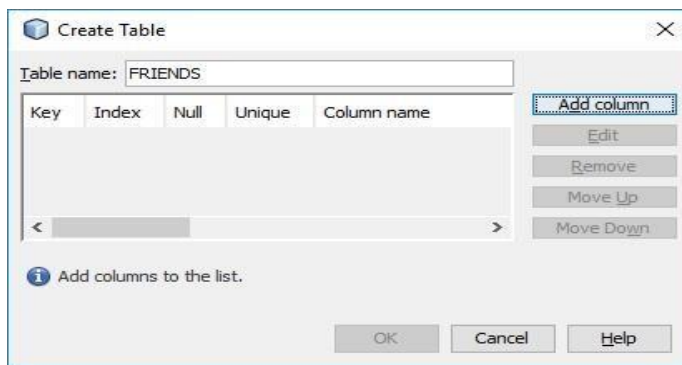


4. Now we are going to create a table in default database **sample**.

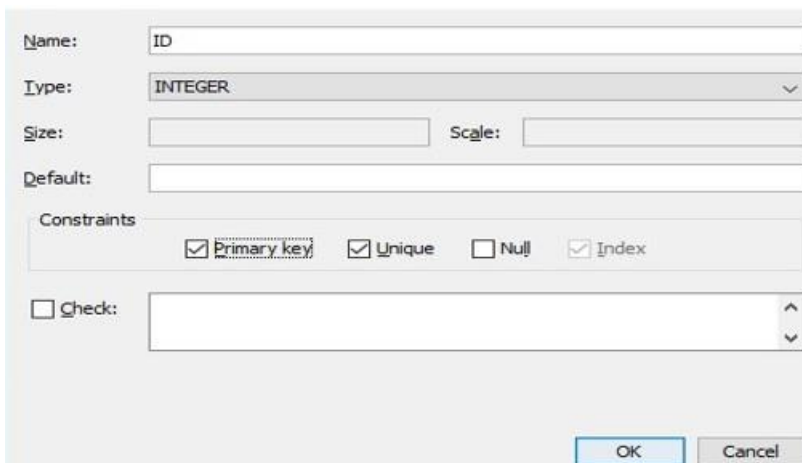
**Right click on Table -> Create Table**



5. Give table name as **FRIENDS**.

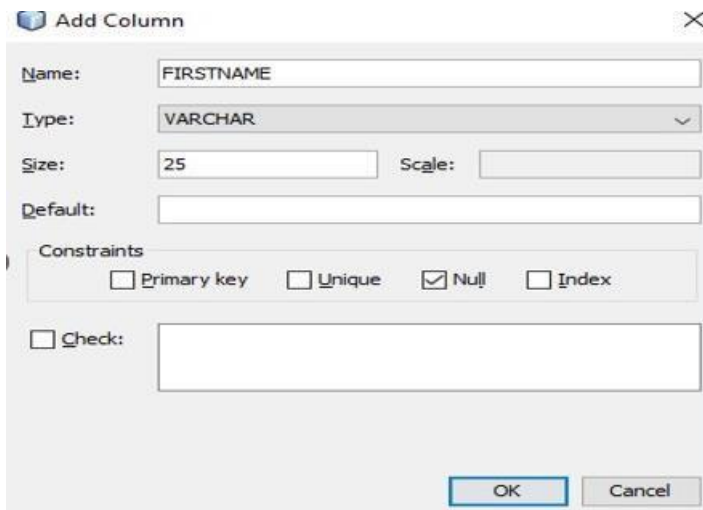


1. Now **click on Add column button to add columns** in table. Enter details as in below pic and **select Primary key**. After that **click on OK** button.

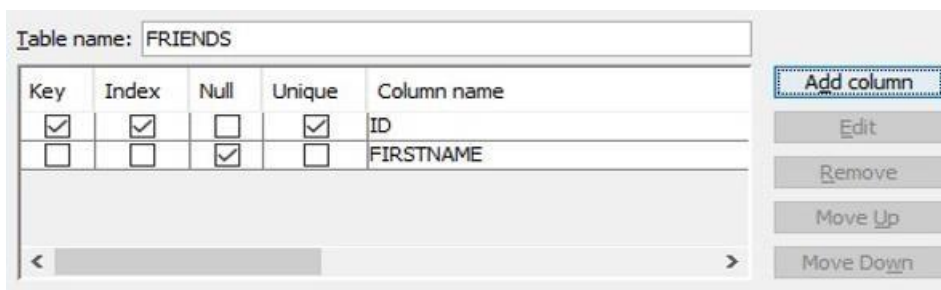


2. Now **add second column** with following detail. But **don't select primary** and click on OK button.
3. Now **click on OK** button.

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



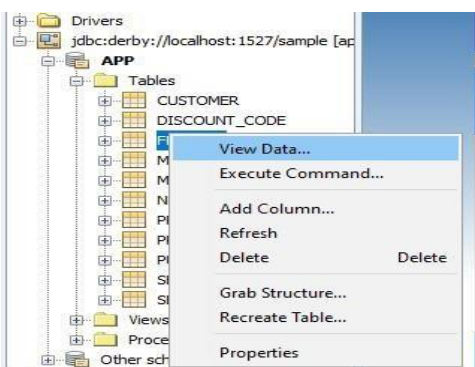
The 'Add Column' dialog box is shown. It has a title bar with a close button. The fields are: Name: FIRSTNAME, Type: VARCHAR (dropdown), Size: 25, Scale: (empty), Default: (empty). Under Constraints, there are checkboxes for Primary key, Unique, Null (checked), and Index. There is also a 'Check' checkbox and a text area for a check constraint. At the bottom are OK and Cancel buttons.



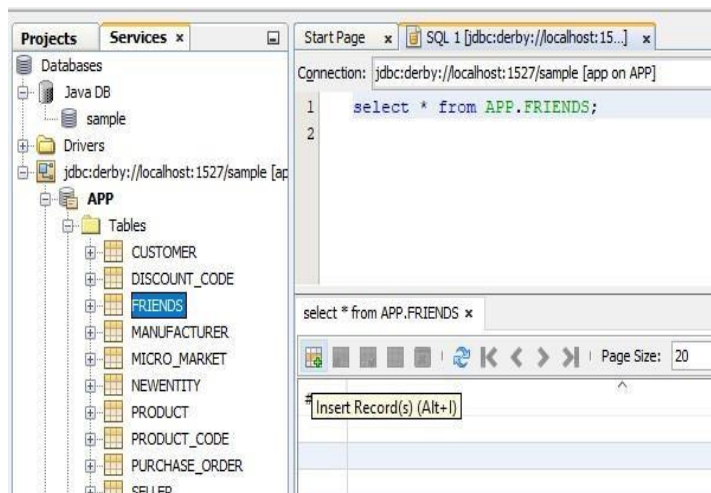
The table structure view for the 'FRIENDS' table is shown. The table name is 'FRIENDS'. It has two columns: 'ID' and 'FIRSTNAME'. The 'ID' column is the primary key, indexed, and unique. The 'FIRSTNAME' column is nullable and not indexed. To the right of the table structure are buttons: 'Add column', 'Edit', 'Remove', 'Move Up', and 'Move Down'.

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

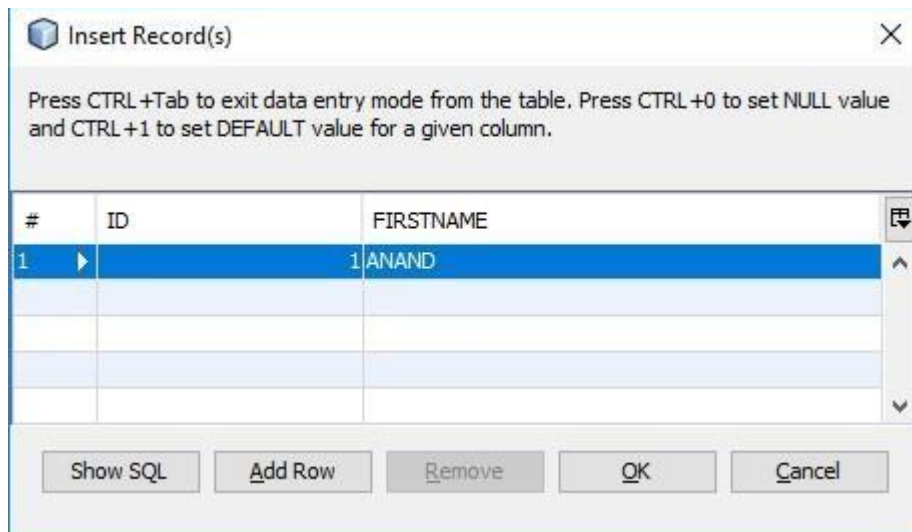
5. Right click on **FRIENDS** to view and add records into it.



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



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



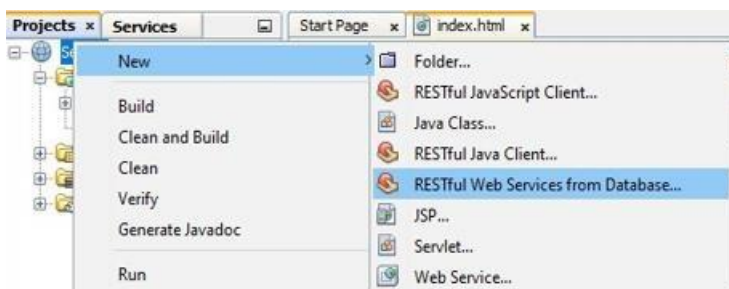
8. As you can see, I have entered 7 records.
9. **create a web application** with name **Server**. After that **click on Next** and then **Finish** button.



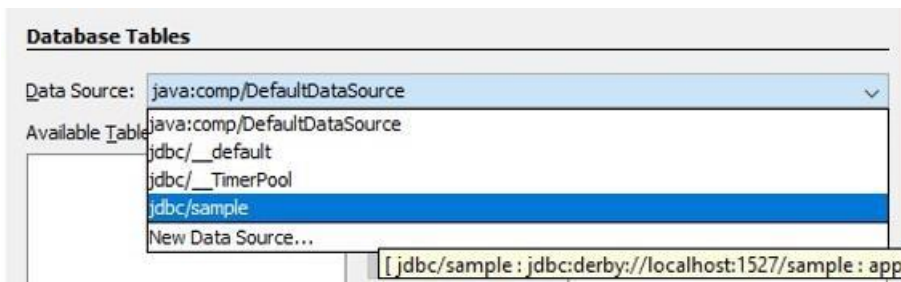
select \* from APP.FRIENDS x

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

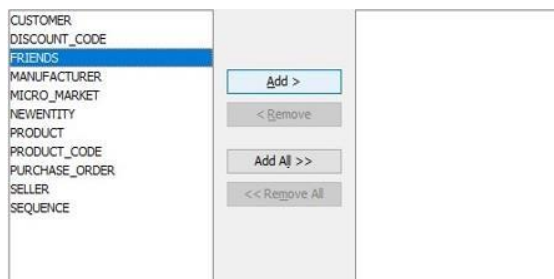
10. Now create a **RESTful Web Service** from Database by right click on **project** name.



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

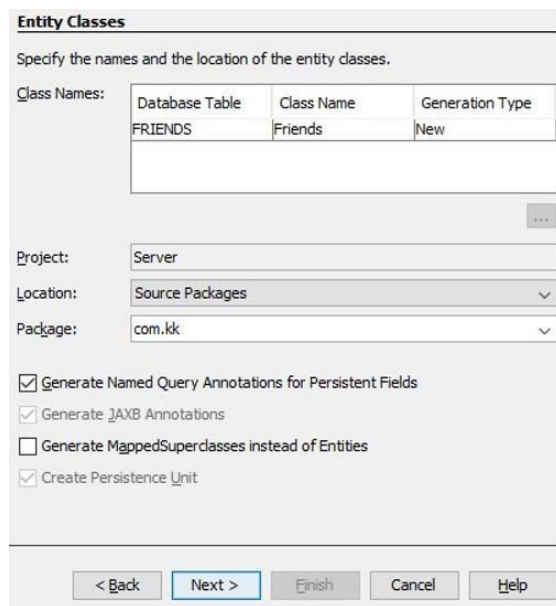


12. Now select **FRIENDS** and click on **Add button**. After that click on **Next** button.





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

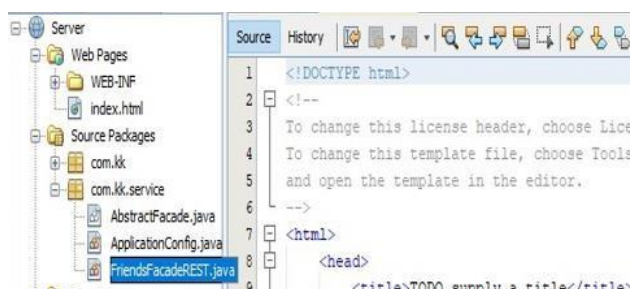


The 'Entity Classes' dialog box is shown. It has a title bar 'Entity Classes' and a subtitle 'Specify the names and the location of the entity classes.' Below this, there is a 'Class Names:' section with a table:

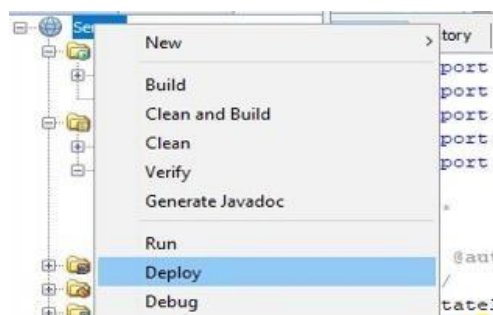
Database Table	Class Name	Generation Type
FRIENDS	Friends	New
<input type="text"/>		

Below the table is a 'Project:' field with 'Server' entered. Below that is a 'Location:' dropdown menu with 'Source Packages' selected. Below that is a 'Package:' dropdown menu with 'com.kk' selected. At the bottom, there are four checkboxes: 'Generate Named Query Annotations for Persistent Fields' (checked), 'Generate JAXB Annotations' (checked), 'Generate MappedSuperclasses instead of Entities' (unchecked), and 'Create Persistence Unit' (checked). At the very bottom are five buttons: '< Back', 'Next >', 'Finish', 'Cancel', and 'Help'.

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



15. 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.
16. After that **right click on project name and Deploy** it.



## Practical No :- 3

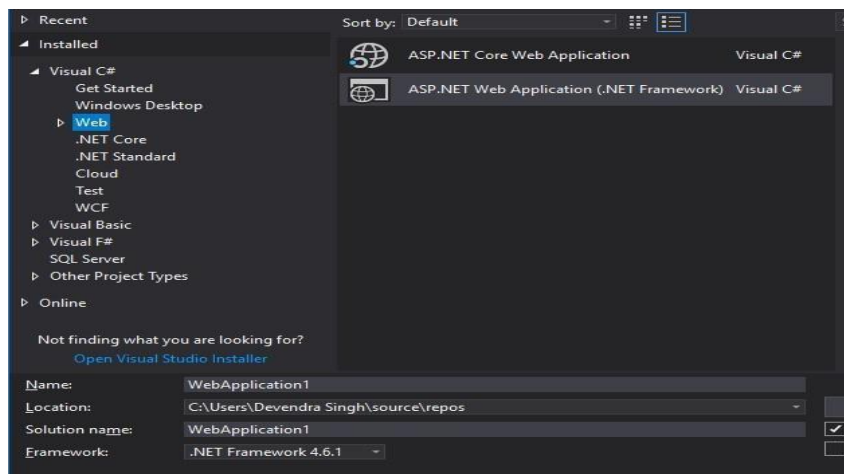
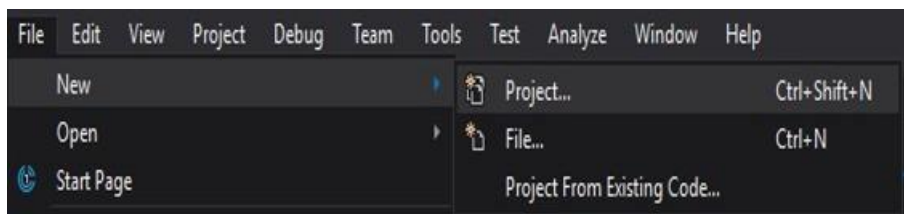
**Aim:-** Develop client which consumes web services developed in different platform.     **Requirement: -**

1. Visual Studio Community 2017
2. Version : 15.8 or latest

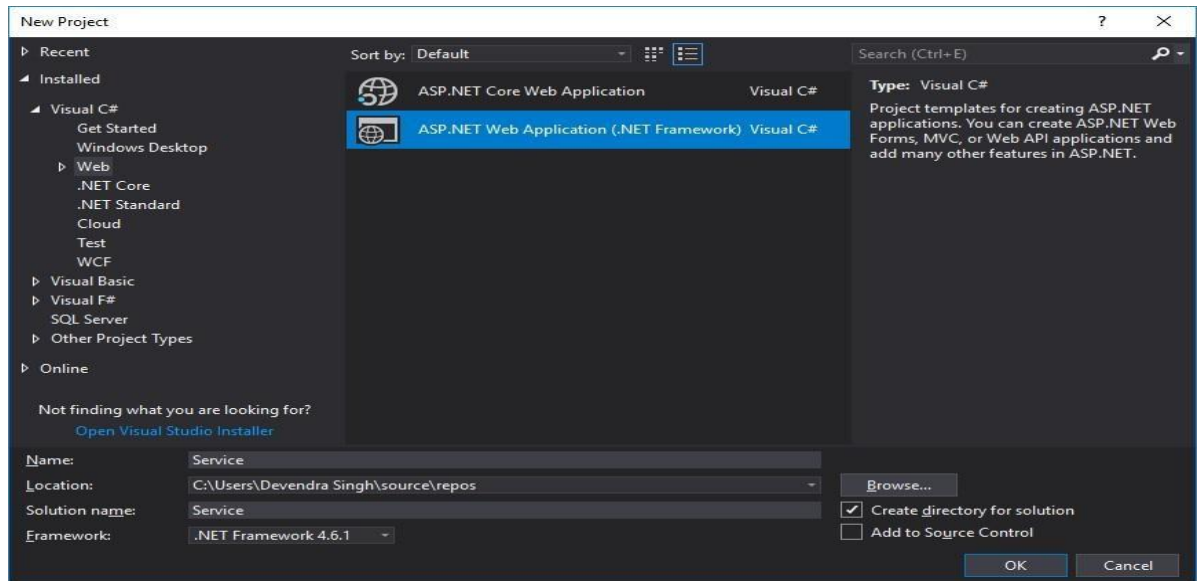
**In this practical we are creating Web Service in Visual Studio ant then we will consume it in NetBeans.**

### Steps:-

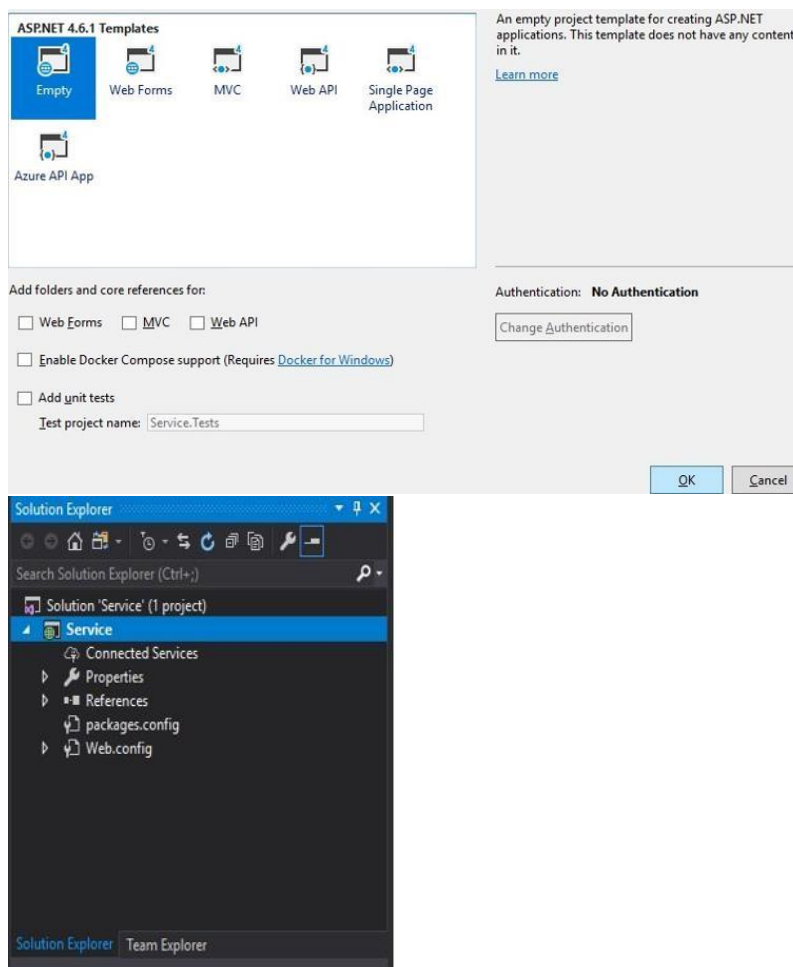
1. **Open Visual Studio IDE** and click on **File**. **File -> New -> Project**
2. **Click on Web.**



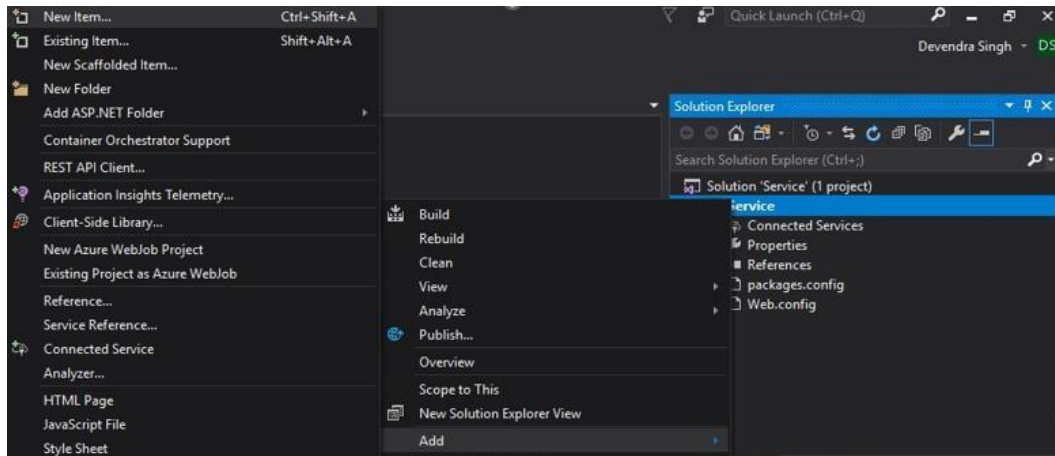
3. **Select ASP.NET Web Application** and give **Name** as **Service**. After that click on **OK** button.



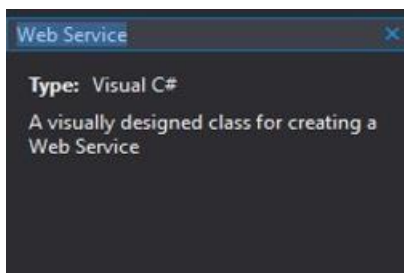
4. **Select Empty** and **click on OK** button.
5. Now you can see, on right side in Solution Explorer Service project is created.



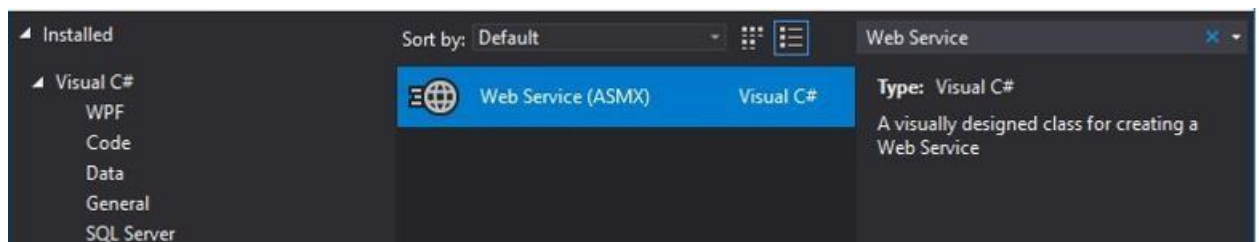
6. **Right click on Service -> Add -> New Item.**



## 7. Search for Web Service.



## 8. Select Web Service and give Name as Operation. After then click on Add button.

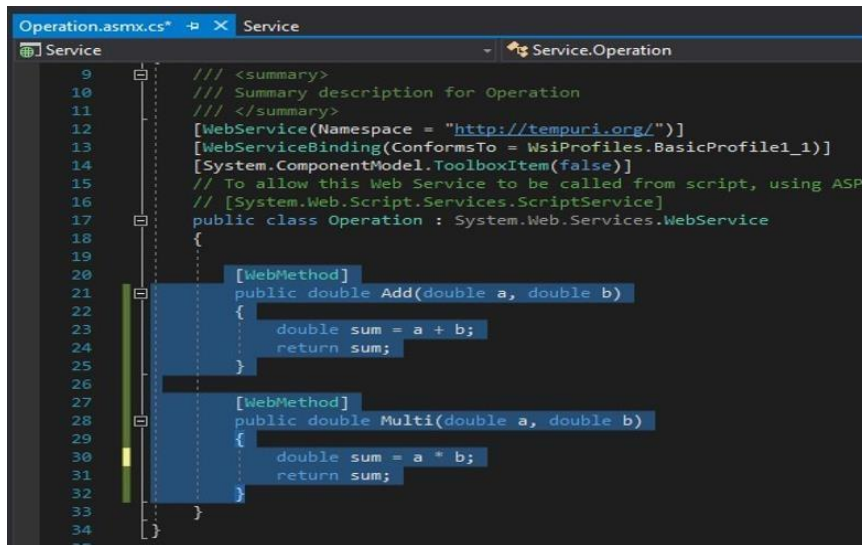


## 9. After click on Add button, Operation.asmx.cs file will be automatically open otherwise open it from Solution Explorer and Add the following code into Class Operation.

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

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

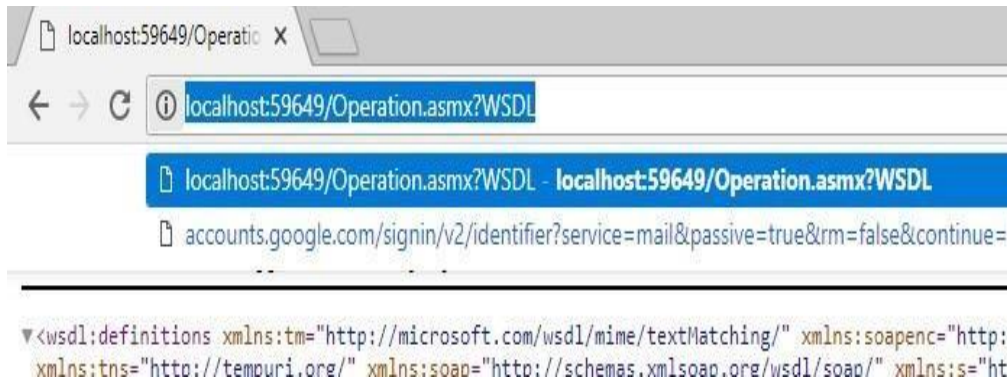
```
    return s  
}
```



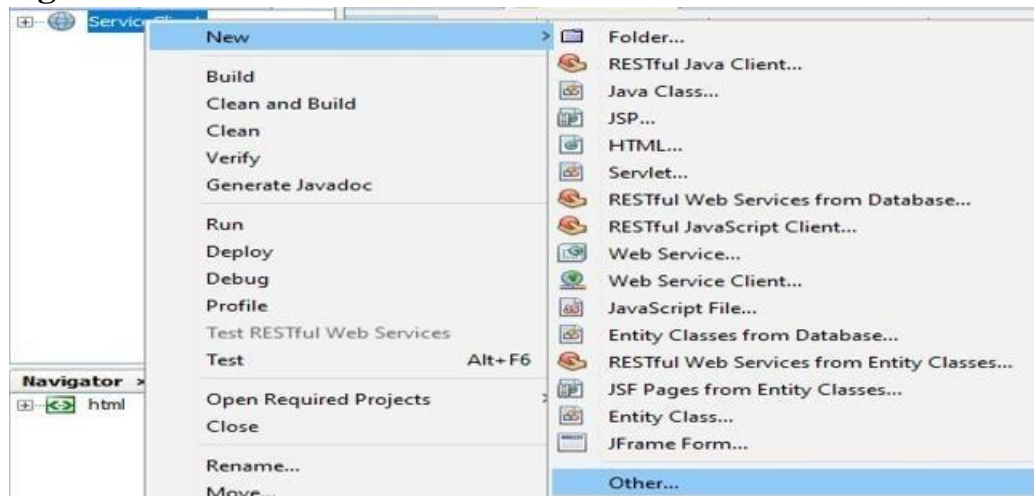
```
9      /// <summary>  
10     /// Summary description for Operation  
11     /// </summary>  
12     [WebService(Namespace = "http://tempuri.org/")]  
13     [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]  
14     [System.ComponentModel.ToolboxItem(false)]  
15     // To allow this Web Service to be called from script, using ASP  
16     // [System.Web.Script.Services.ScriptService]  
17     public class Operation : System.Web.Services.WebService  
18     {  
19  
20         [WebMethod]  
21         public double Add(double a, double b)  
22         {  
23             double sum = a + b;  
24             return sum;  
25         }  
26  
27         [WebMethod]  
28         public double Multi(double a, double b)  
29         {  
30             double sum = a * b;  
31             return sum;  
32         }  
33     }  
34  
35 }
```

After that press **Ctrl+S** to save the methods. Actually we are creating two methods for Web Service. One is for addition of two numbers and second one is for multiplication of two numbers.

10. Now **run the project** by click on **Green arrow button** below the **Window menu**.
11. A window will open in browser and that is our web service.
12. You can check services by click on **Add** or **Multi** option. But we don't need this.
13. Now **click on Service Description** option.
14. Now a new window will open. **Select the link and copy it. We will use this link in NetBeans to consume these services.**

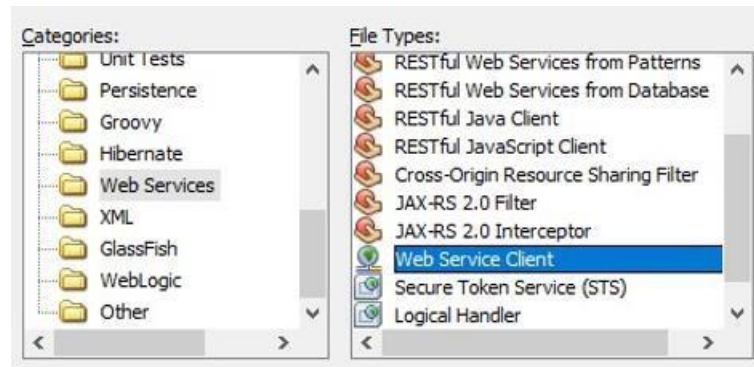


15. **Don't close Visual Studio and browser, just minimize it otherwise server will stop. But save the link anywhere, so that we can use it later.**
16. **Now open NetBeans.**
17. **Create a Web Application with name ServiceClient. Next -> Finish.**
18. **Now create a Web Service Client.**  
**Right click on ServiceClient -> New -> Other.**



19. **Now select Web Services in Categories section.**
20. **Select Web Service Client in File Types and Click on Next button.**

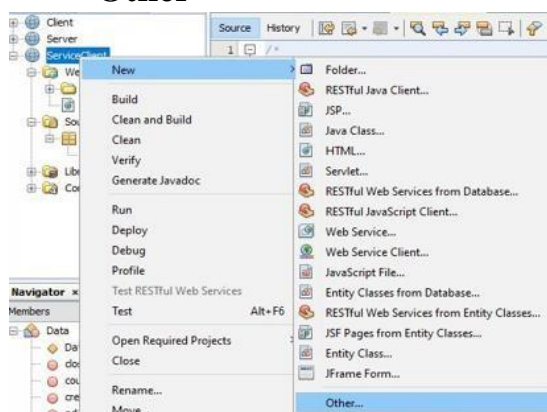




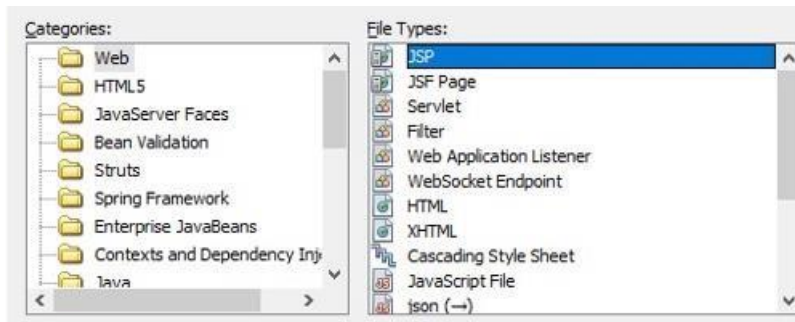
21. **Select WSDL URL and paste the link** that you have copied from browser on run of Visual Studio **enter package name com.dd**. After that **click on Finish** button.
22. As you can see, we got both the service methods i.e. Add & Multi. But in this practical I'm going to use only one service method. You can do for both also.



23. Now create a JSP page. **Right click on ServiceClient -> New -> Other**



24. **Select Web in Categories section -> Select JSP and click on Next button.**



25. **Enter File Name Add and click on Finish button.**

**Name and Location**

File Name:

Project:

Location:

Folder:

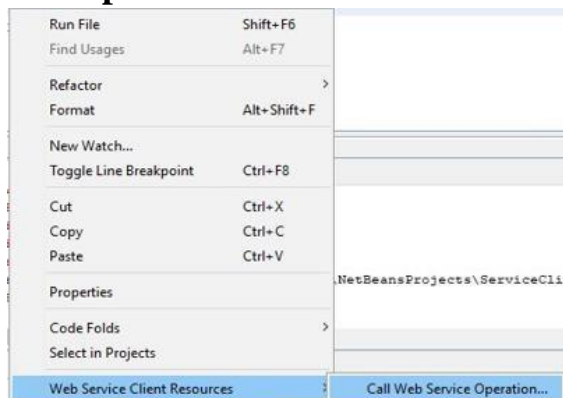
26. **In Add.jsp file delete the selected part in body tag, because we don't need this.**

```

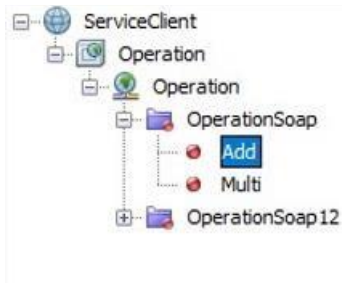
Source History
1  <%--
2      Document    : Add
3      Created on  : Aug 17, 2018, 10:58:06 AM
4      Author     : Devendra Singh
5  --%>
6
7  <%@page contentType="text/html" pageEncoding="UTF-8"%>
8  <!DOCTYPE html>
9  <html>
10     <head>
11         <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"%>
12         <title>JSP Page</title>
13     </head>
14     <body>
15         <h1>Hello World!</h1>
16     </body>
17 </html>

```

27. **Right click between body tag and select Call Web Service Operation.**
28. **Expand and select Add. After select, click on OK button.**







29. Now add the following code outside of try block.

```
double num1 = Double.parseDouble(request.getParameter("txt1"));
double num2 = Double.parseDouble(request.getParameter("txt2"));
```

```

<!-- start web service invocation --><hr/>
<%
    double num1 = Double.parseDouble(request.getParameter("txt1"));
    double num2 = Double.parseDouble(request.getParameter("txt2"));
    try {
        com.dd.Operation service = new com.dd.Operation();
        com.dd.OperationSoap port = service.getOperationSoap();
        // TODO initialize WS operation arguments here
        double a = 0.0d;
        double b = 0.0d;
    }
  %>
  
```

30. Now pass num1 & num2 to a & b variable respectively. After that press Ctrl+S to save this code.

```

Page x index.html x Add.jsp x
History
<!-- start web service invocation --><hr/>
<%
    double num1 = Double.parseDouble(request.getParameter("txt1"));
    double num2 = Double.parseDouble(request.getParameter("txt2"));
    try {
        com.dd.Operation service = new com.dd.Operation();
        com.dd.OperationSoap port = service.getOperationSoap();
        // TODO initialize WS operation arguments here
        double a = num1;
        double b = num2;
        // TODO process result here
        double result = port.add(a, b);
        out.println("Result = "+result);
    } catch (Exception ex) {
        // TODO handle custom exceptions here
    }
  %>
  
```

31. Now open index.html file of ServiceClient project and replace the contents of body tag with following code. After that press Ctrl+S to save it.

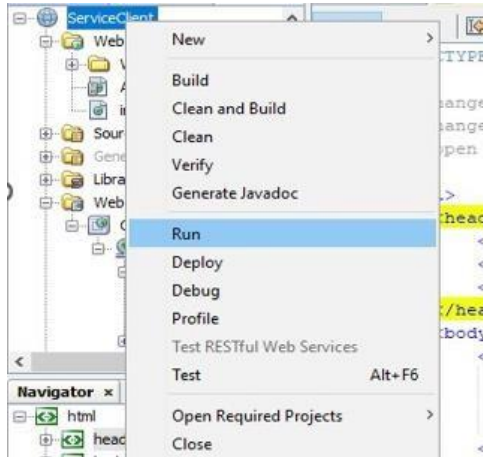
```

<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="Add.jsp" value="Add  
Numbers"> </form>
```

```
<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="Add.jsp" value="Add Numbers">  
</form>
```

**32.** Now run the **ServerClient** web application.

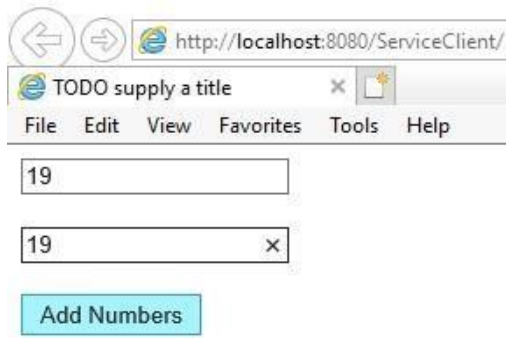


**33.** A window will open in browser as below.

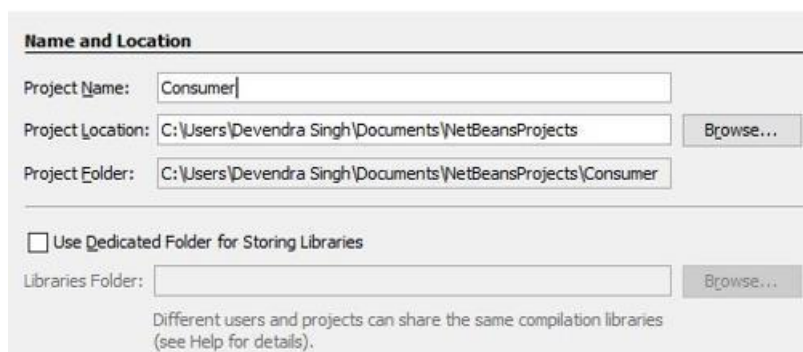


**34.** Now enter two numbers and click on **Add Numbers** button. Wait to get result.....

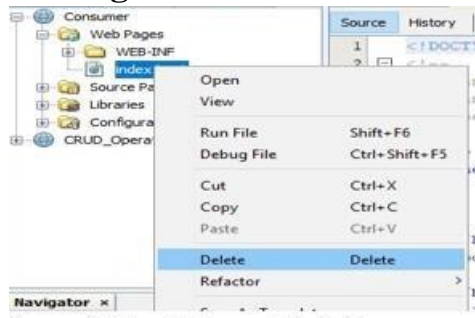
You will get result on a new page.



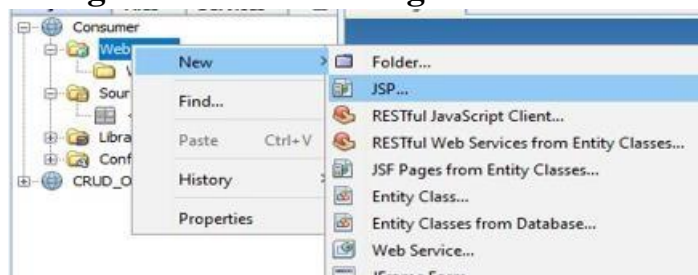
1. Create an another Web Application project and Give name as **Consumer**.



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.

```
<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);
```

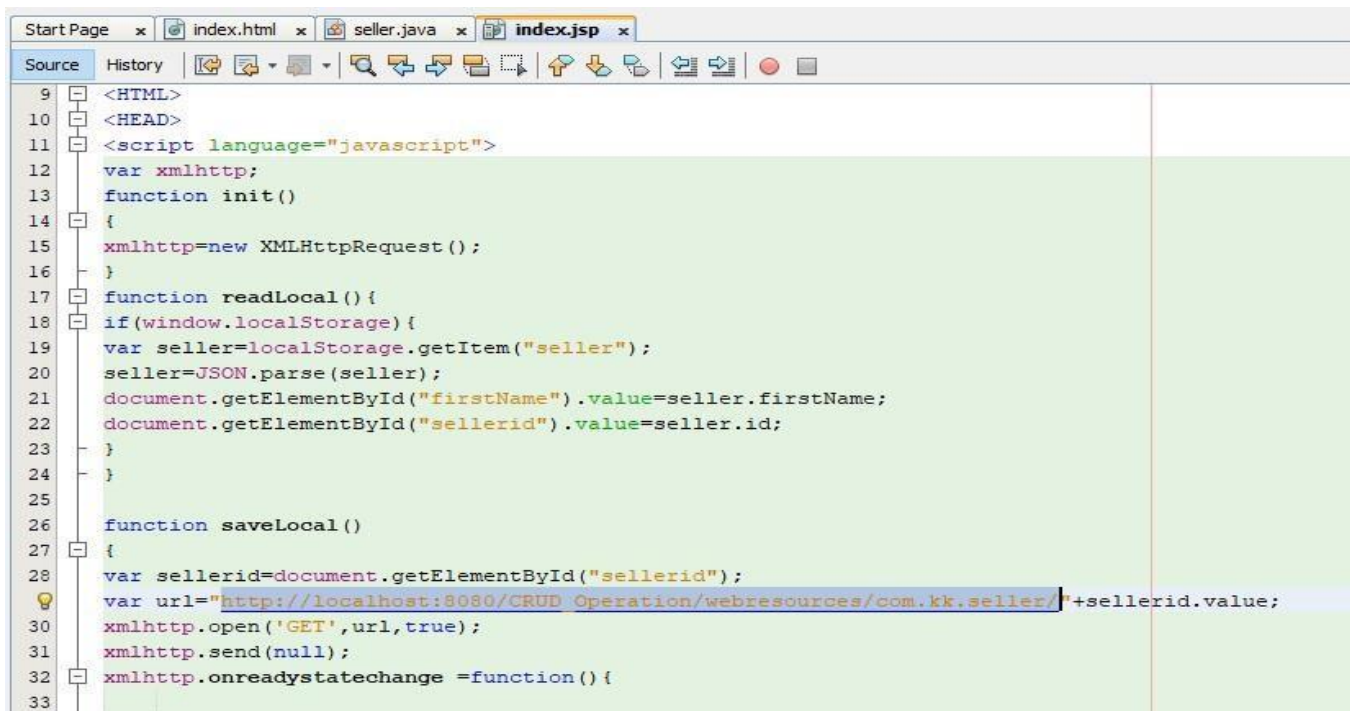
```

        }
        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>
</table>
</body>
</html>

```

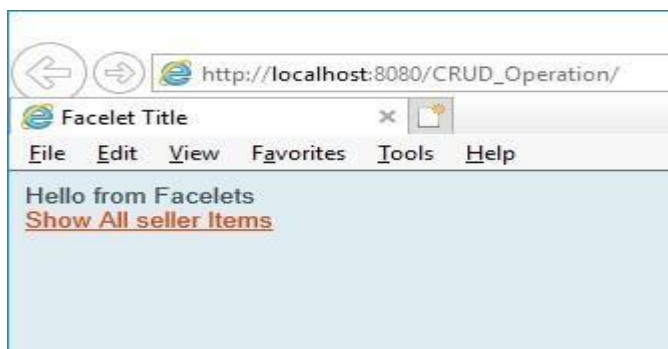
8. Now in the following pic, **highlighted URL is most important**. I'm explaining it next step.



```
9 <HTML>
10 <HEAD>
11 <script language="javascript">
12   var xmlhttp;
13   function init()
14   {
15     xmlhttp=new XMLHttpRequest();
16   }
17   function readLocal() {
18     if(window.localStorage) {
19       var seller=localStorage.getItem("seller");
20       seller=JSON.parse(seller);
21       document.getElementById("firstName").value=seller.firstName;
22       document.getElementById("sellerid").value=seller.id;
23     }
24   }
25
26   function saveLocal()
27   {
28     var sellerid=document.getElementById("sellerid");
29     var url="http://localhost:8080/CRUD_Operation/webresources/com.kk.seller/" + sellerid.value;
30     xmlhttp.open('GET',url,true);
31     xmlhttp.send(null);
32     xmlhttp.onreadystatechange =function() {
33
```

9. [http://localhost:8080/CRUD\\_Operation/webresources/com.kk.seller/](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.

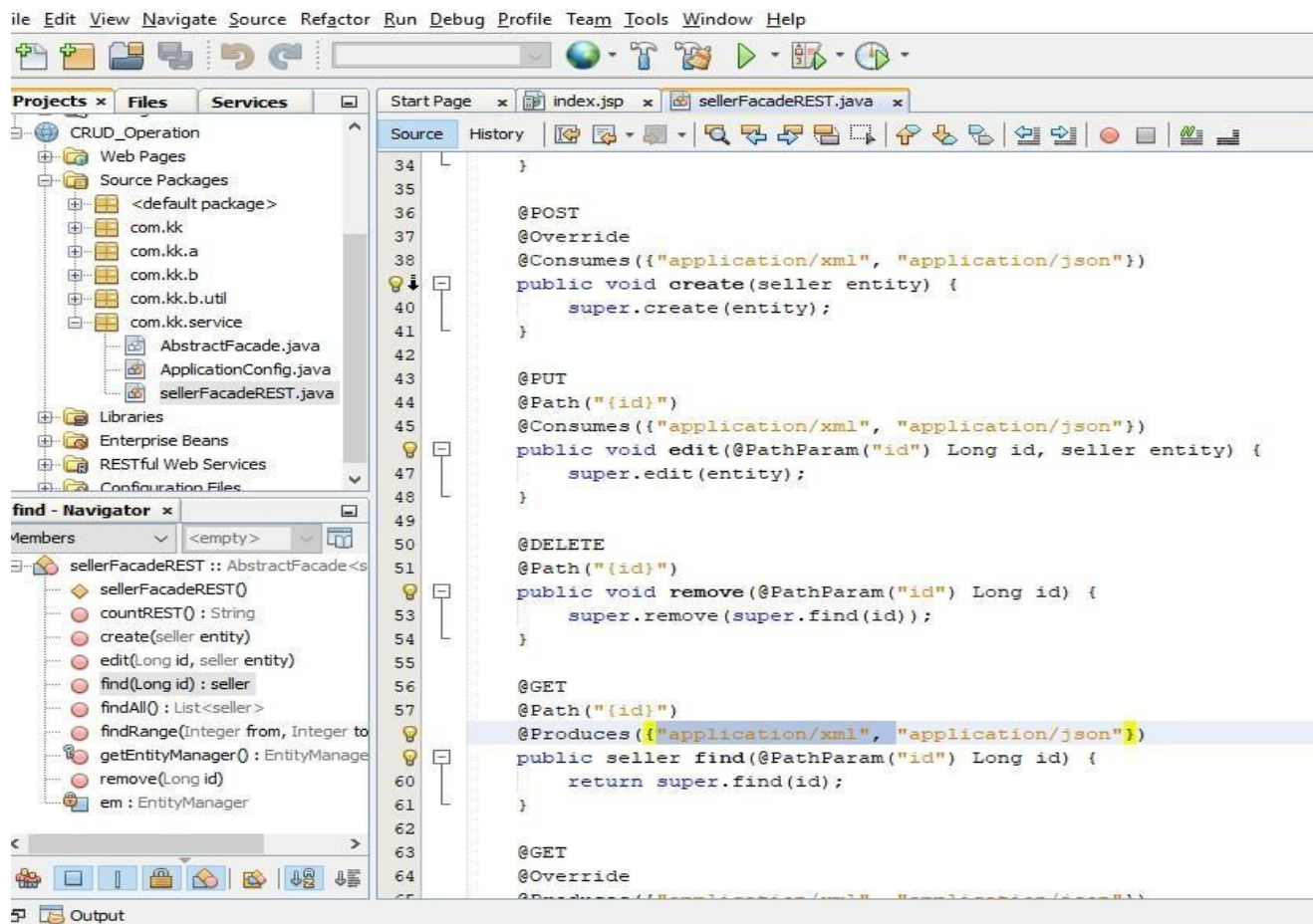


**Blue part** in above link is **static**. But **red part** is **dynamic** which is based on practical 7. So if you have changed name anywhere then the above URL will change accordingly.

10. Now open sellerFacadeREST.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.

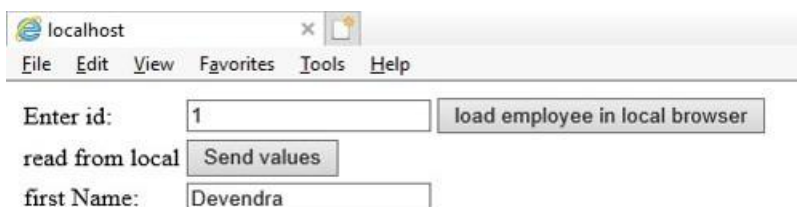




12. Now run the Consumer application. A window will open in browser like below.



**Output:-**

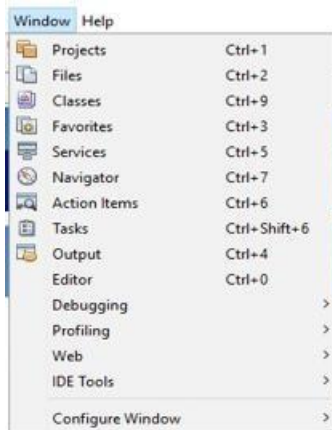


## Practical No :- 4

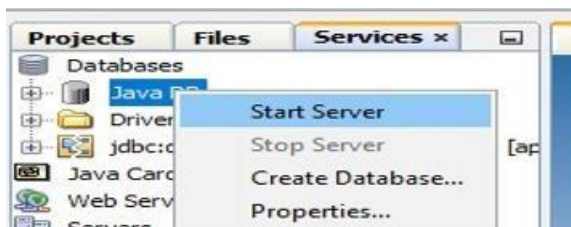
**Aim:-** Define a web service method that returns the contents of a database in a JSON string. The contents should be displayed in a tabular format.

### Steps:-

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.

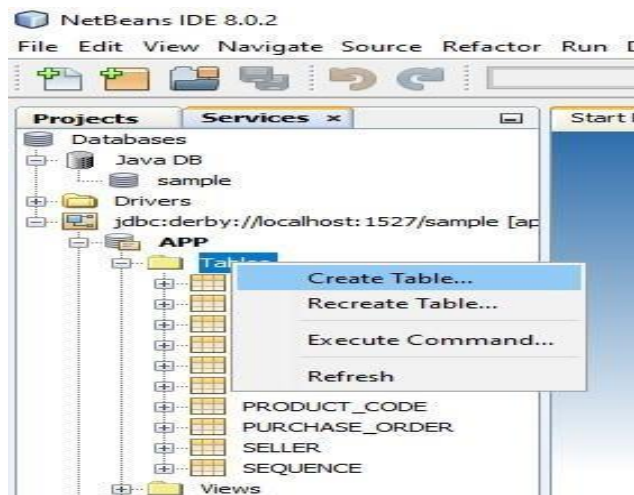


3. Now expand Java DB and **right click on sample** and then **click on connect** to connect the sample database with server.

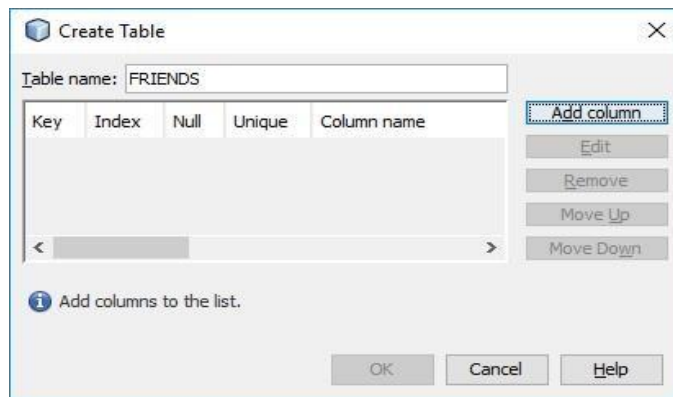


4. Now we are going to create a table in default database **sample**.  
**Right click on Table -> Create Table**





5. Give table name as **FRIENDS**.



6. **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.

Name: ID

Type: INTEGER

Size: Scale:

Default:

Constraints

☒ Primary key ☒ Unique ☐ Null ☒ Index

☐ Check:

OK Cancel

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

Add Column

Name: FIRSTNAME

Type: VARCHAR

Size: 25 Scale:

Default:

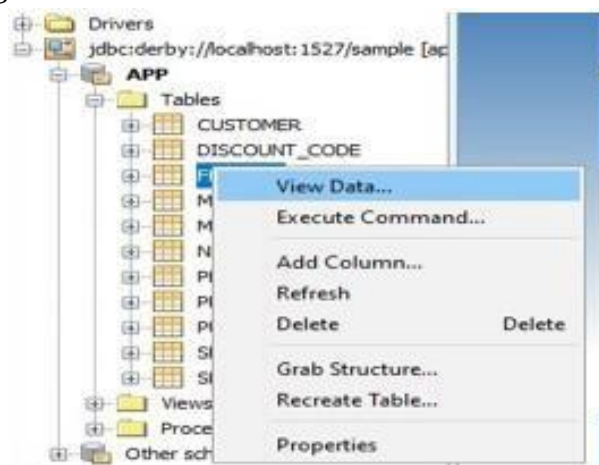
Constraints

☐ Primary key ☐ Unique ☒ Null ☐ Index

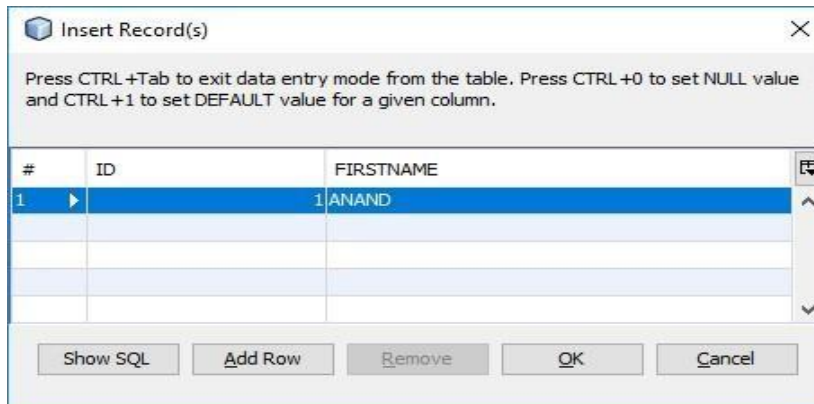
☐ Check:

OK Cancel

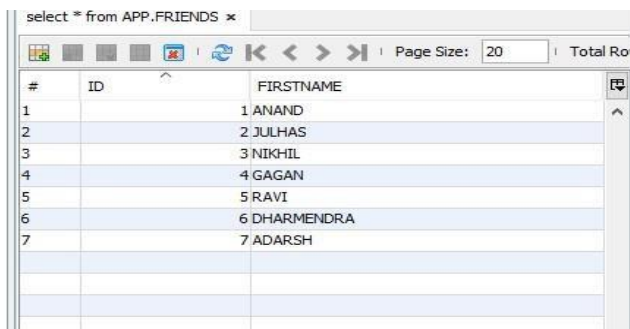
8. Now **click on OK** button.
9. Now you can see a table with name **FRIENDS** in the table.
10. **Right click on FRIENDS** to view and add records into it



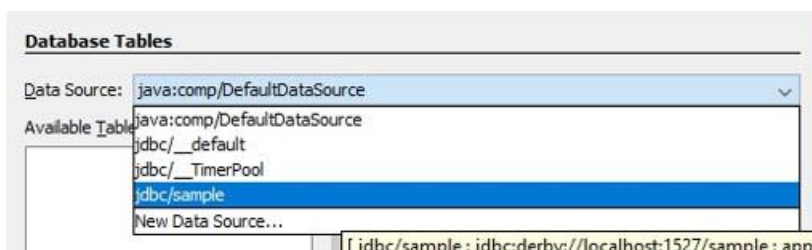
11. click on the leftmost icon in second panel to insert some record.
12. Insert a record and then click on Add Row button to insert more record. After that click on OK button to finish.



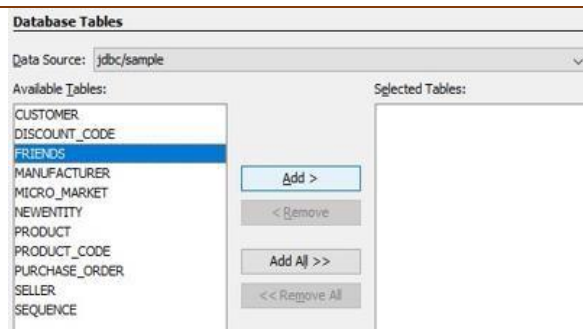
13. As you can see, I have entered 7 records.



14. create a web application with name Server. After that click on Next and then Finish button.
15. Now create a RESTful Web Service from Database by right click on project name.
16. Now Choose Data Source jdbc/sample.



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



18. Enter Package name as **com.kk** and click on **Next** button and then **Finish**.
19. Now open selected file by double click on it.
20. 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.

```

@POST
@Override
@Consumes({ "application/xml", "application/json" })
public void create(Friends entity) {
    super.create(entity);
}

```

21. After that **right click on project name and Deploy** it.
22. Now **create one more Web Application as Client**. After that **click on Next** and then **Finish** button.



23. 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>

```

---



```

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>

```

**URL in the red font is sensitive.** It will change accordingly if you have not given the project and file names as of mine.

**24. Replace the content of body tag with following code.**

```

<table id="myTable">
    <tr>
        <th> ID</th>

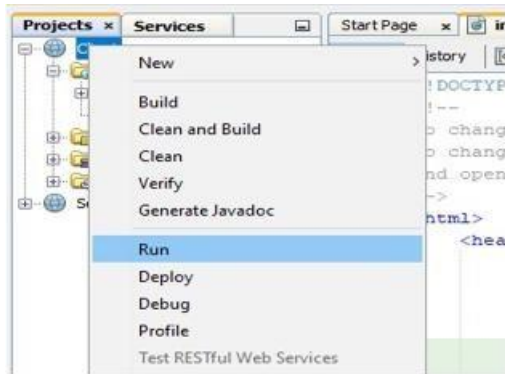
```

```

        <th>NAME</th>
    </tr>
</table>

```

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



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

 A screenshot of a web browser window. The address bar shows 'http://localhost:8080/Client/'. The browser title is 'TODO supply a title'. The main content area displays a table with 7 rows and 2 columns. The columns are labeled 'ID' and 'NAME'. The rows contain the following data: (1, ANAND), (2, JULHAS), (3, NIKHIL), (4, GAGAN), (5, RAVI), (6, DHARMENDRA), and (7, ADARSH).
 

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

### Steps for creating server:-

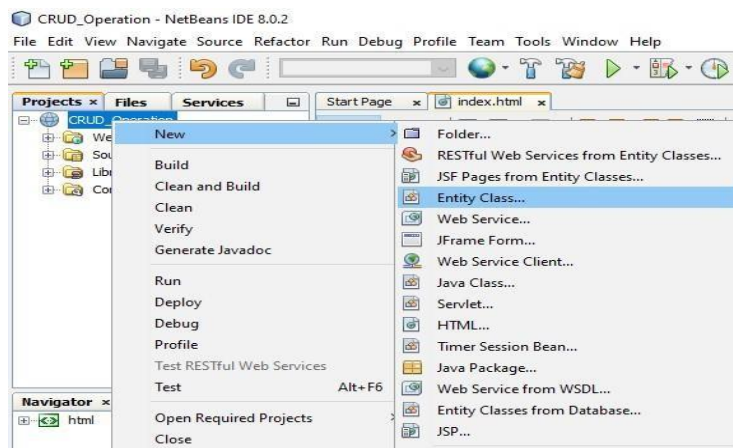
1. Click on Window menu and click on **Projects, Files & Services** to open it.
2. **Right click on Java DB** and then **click on Start Server** to start the server



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**. A window will open like following pic.
5. Create an entity class. **Right click on project name -> New -> Entity Class**.

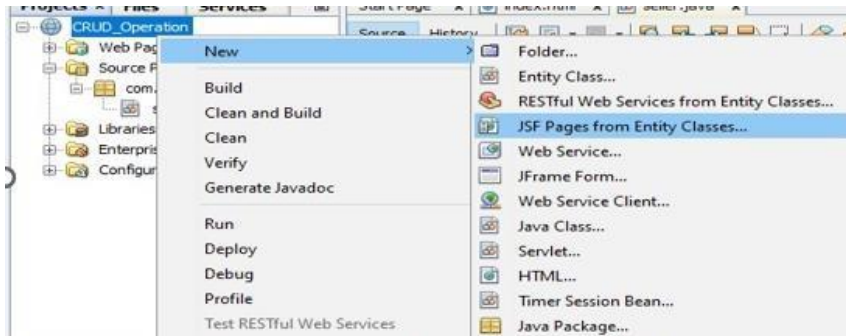


6. A window will appear like bellow pic. Enter following data and click on Next ....

**Class Name -> seller**

**Package Name -> com.kk**

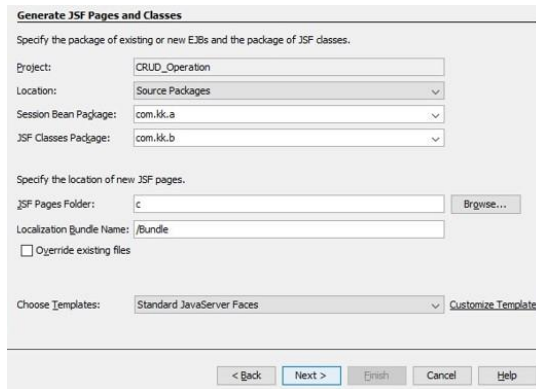
7. **Click on Finish.**
8. Right click on project name and create JSF Pages from Entity Classes.  
**Right click on project name -> New -> JSF Pages from Entity Classes**
9. **Select com.kk.seller and click on Add button and then Next button on below.**



10. 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.**

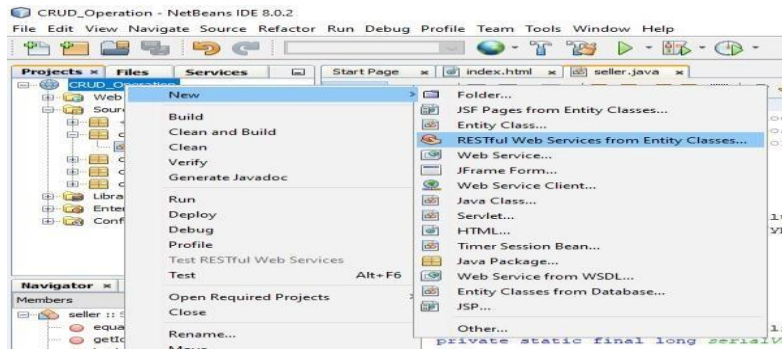






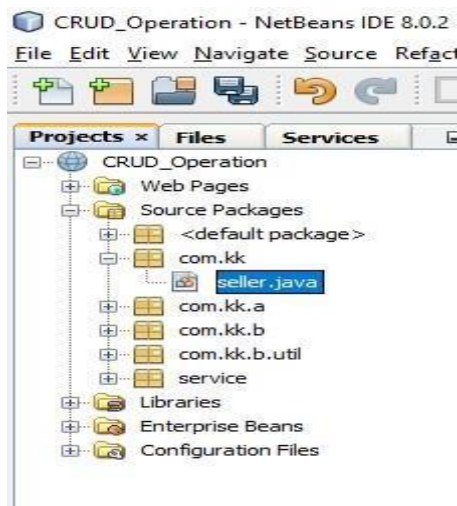
**11. Right click on project name and create RESTful Web Services from Entity Classes.**

**Right click on project name -> New -> RESTful Web Services from Entity Classes**



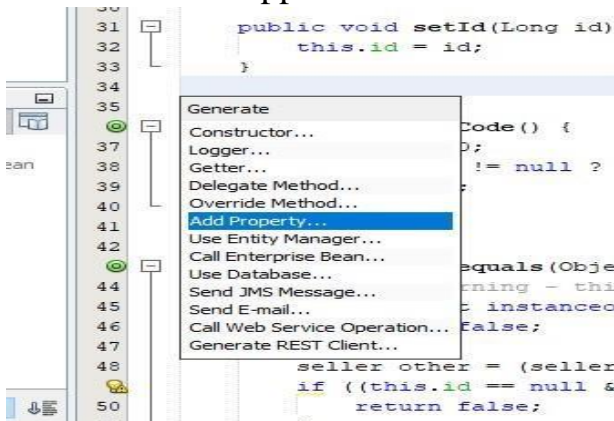
**12. Repeat step 9 and then it will go on next page. Then enter the com.kk.service in Resource Package and then click on Finish button.**

**13. Now open seller.java file under com.kk package.**

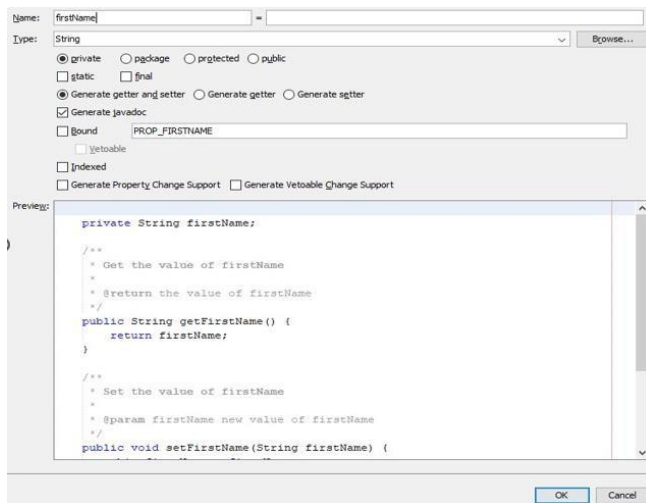


14. In this file at line number 24, do the right click and select **Insert Code**.

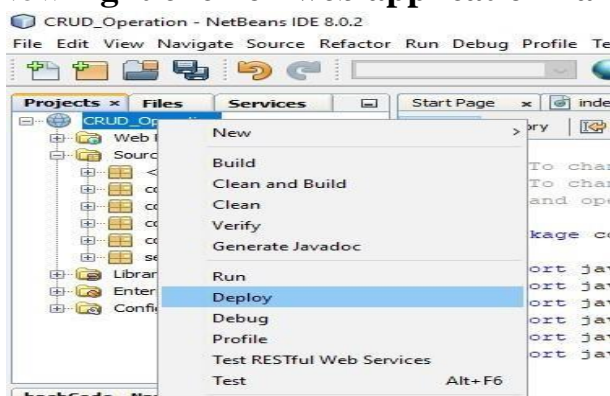
15. A new list will appear. Click on **Add Property**.



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**.

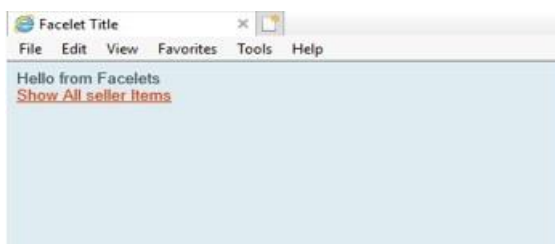


17. Now **right click** on web application name and **Deploy** it.



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 sellers** Items for CRUD operation.



21. As I have added three data into database. **You can add more data by click on Create New seller and can view, edit and delete by click on View, Edit and Destroy option.**

**List**

1..3/3

FirstName	Id	
Devendra	1	<a href="#">View</a> <a href="#">Edit</a> <a href="#">Destroy</a>
Devendra	2	<a href="#">View</a> <a href="#">Edit</a> <a href="#">Destroy</a>
Shivendra	3	<a href="#">View</a> <a href="#">Edit</a> <a href="#">Destroy</a>

[Create New seller](#)

[Index](#)

22. Adding one more data into database for demo. Just click on Create New seller. **Enter a name into FirstName and id into Id.** Now **click on Save** option to save the data.

**Create New seller**

FirstName:

Id:

[Save](#)

[Show All seller Items](#)

[Index](#)