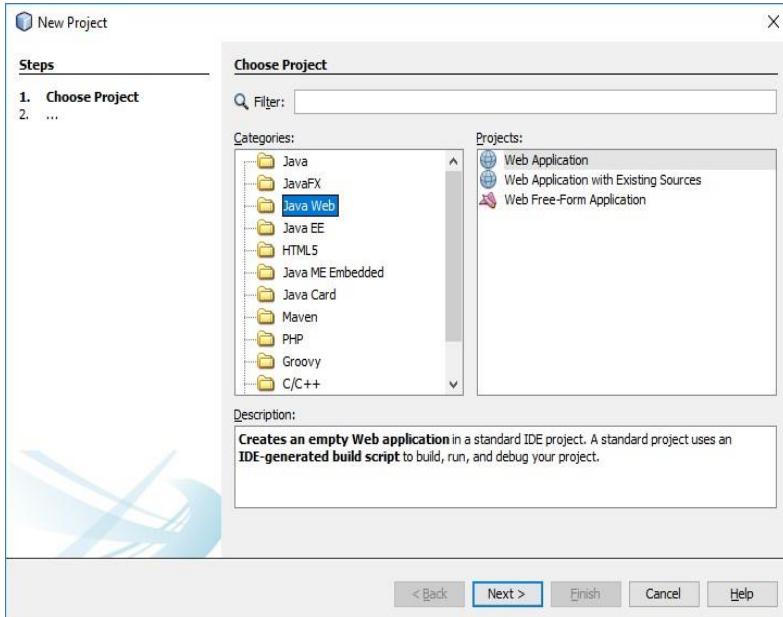
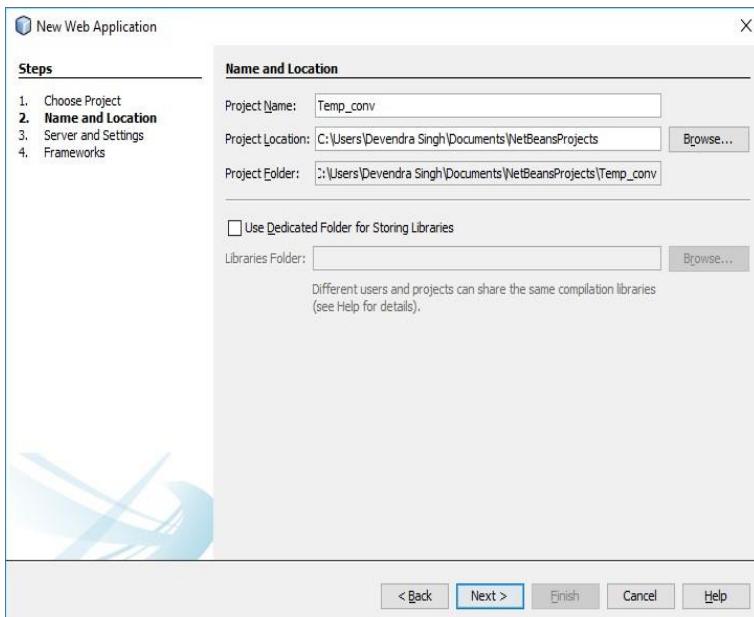


PRACTICAL-01

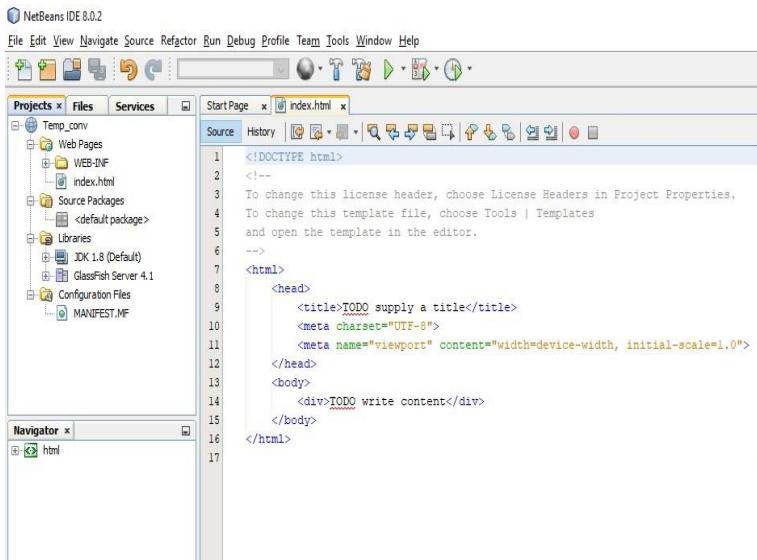
1. Go to File -> New Project. Select Java Web in categories and Web Application in Projects. Click on Next to create web based project.



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



3. After completion of project creation process a window will appear like below.



NetBeans IDE 8.0.2

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

Projects x Files Services Start Page x index.html x

Source History |

Temp_conv

- Web Pages
 - WEB-INF
 - index.html
- Source Packages
 - <default package>
- Libraries
 - JDK 1.8 (Default)
 - GlassFish Server 4.1
- Configuration Files
 - MANIFEST.MF

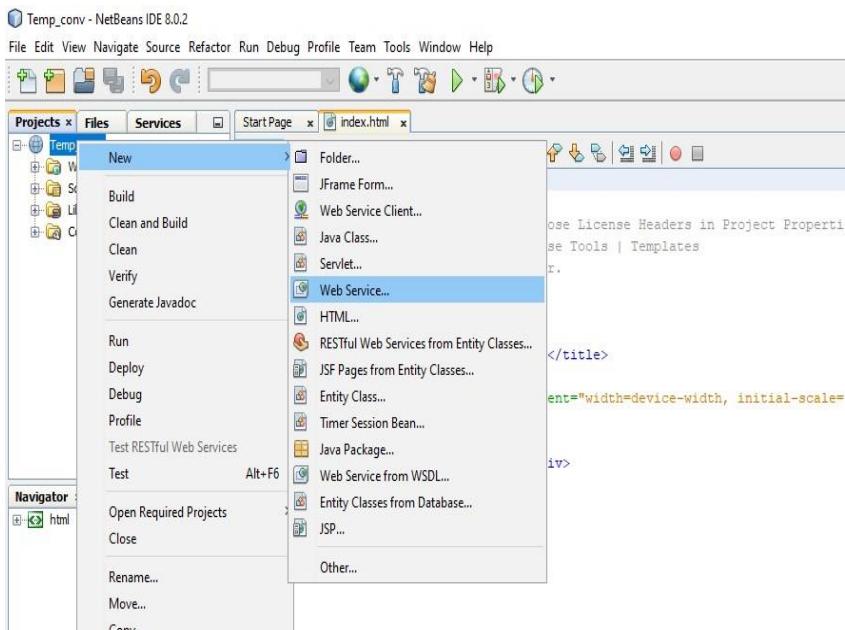
Navigator x

html

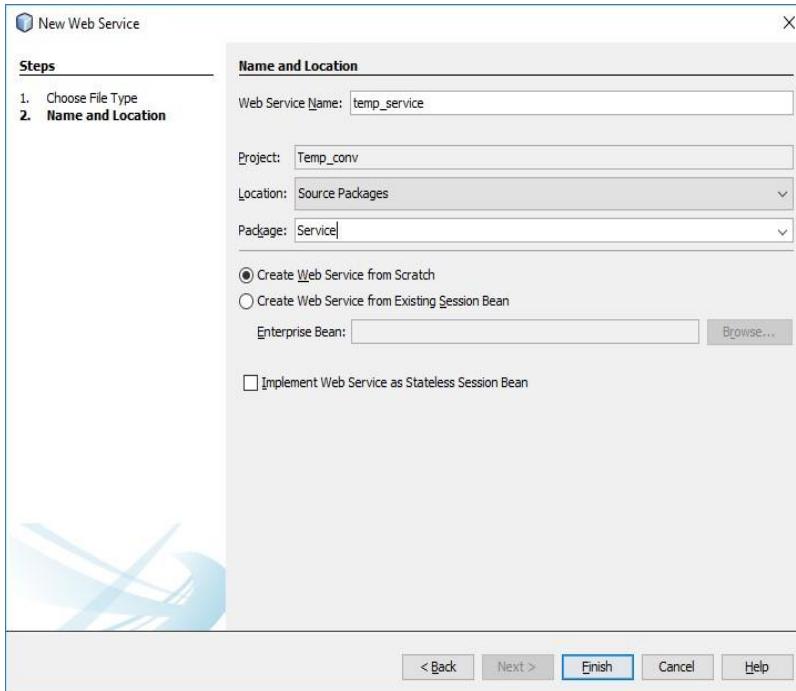
```
<!DOCTYPE html>
<!--
To change this license header, choose License Headers in Project Properties.
To change this template file, choose Tools | Templates
and open the template in the editor.
-->
<html>
    <head>
        <title>TODO supply a title</title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
    </head>
    <body>
        <div>TODO write content</div>
    </body>
</html>
```

4. Create web service.

Right click on Project -> New -> 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 celcious to farhenheit and vice-versa.
Go to design mode by click on **Design** .

Temp_conv - NetBeans IDE 8.0.2

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

```

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package Service;

import javax.jws.WebService;
import javax.jws.WebMethod;
import javax.jws.WebParam;

/**
 *
 * @author Devendra Singh
 */
@WebService(serviceName = "temp_service")
public class temp_service {

    /**
     * This is a sample web service operation
     */
    @WebMethod(operationName = "hello")
    public String hello(@WebParam(name = "name") String txt) {
        return "Hello " + txt + " !";
    }
}

```

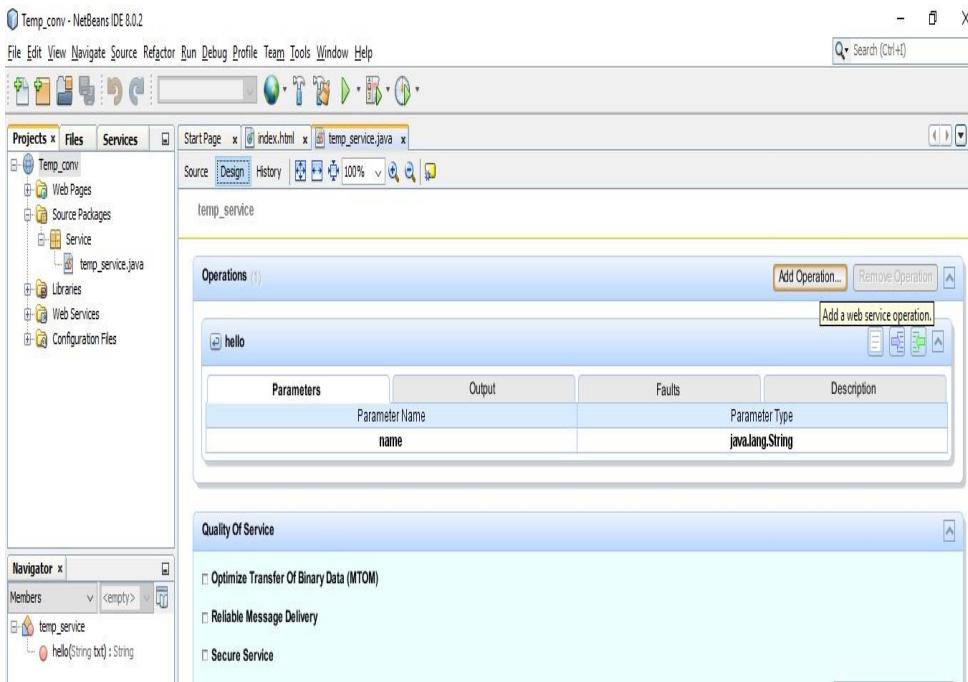
Projects x Files Services Start Page x index.html x temp_service.java x

Source Design History

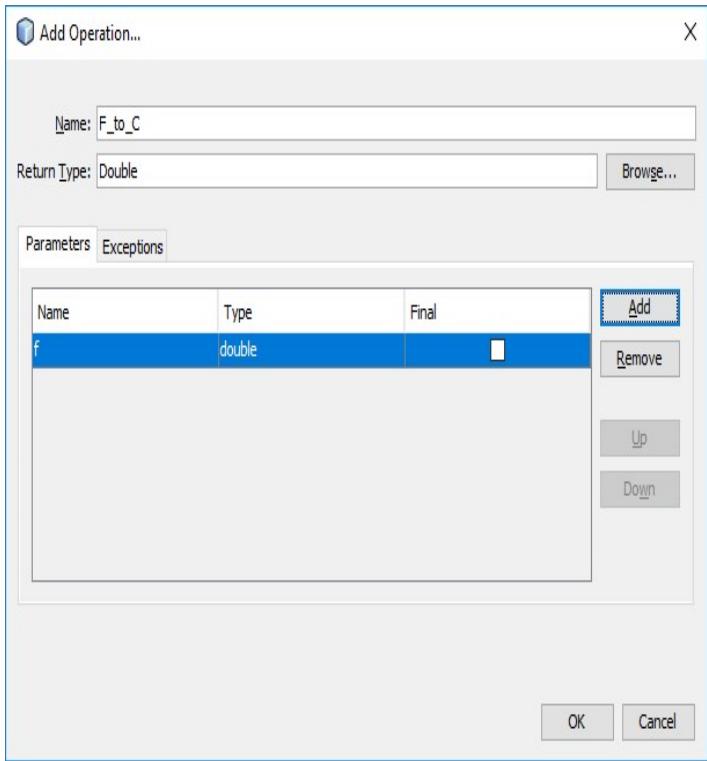
Navigator x Members <empty>

temp_service hello(String txt) : String

7. Click on Add Operation to add operation.



8. 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.



9. **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

10. 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.

The screenshot shows the Eclipse IDE interface with the 'Source' tab selected. The code editor displays Java code for a web service named 'temp_service'. A specific section of the code, which defines a method named 'hello' that takes two parameters and returns a String, is highlighted with a blue selection bar.

```
13  /*
14  * @author Devendra Singh
15  */
16  @WebService(serviceName = "temp_service")
17  public class temp_service {
18
19      /**
20       * This is a sample web service operation
21       */
22      @WebMethod(operationName = "hello")
23      public String hello(@WebParam(name = "name") String txt) {
24          return "Hello " + txt + " !";
25      }
26
27      /**
28       * Web service operation
29       */
30      @WebMethod(operationName = "F_to_C")
31      public Double F_to_C(@WebParam(name = "f") double f) {
32          //TODO write your implementation code here:
33          return null;
34      }
35
36      /**
37       * Web service operation
38       */
39      @WebMethod(operationName = "C_to_F")
40      public Double C_to_F(@WebParam(name = "c") double c) {
41          //TODO write your implementation code here:
42          return null;
43      }
}
```

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

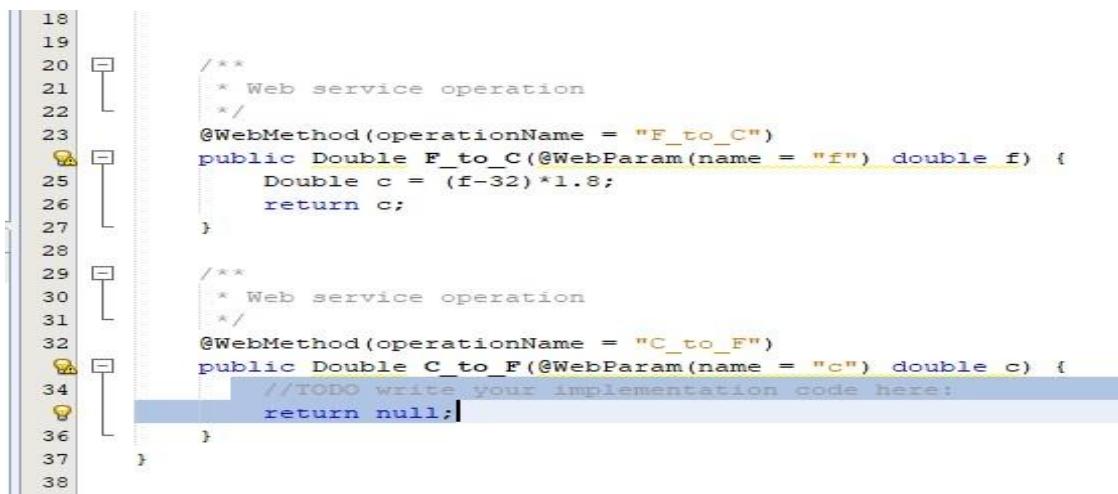
```
Double c = (f-32)*1.8;
return c;
```

The screenshot shows the same Java code for 'temp_service.java' in the Eclipse IDE. The previously selected code in the 'F_to_C' method has been replaced by the provided conversion formula. The replaced code is highlighted with a yellow selection bar.

```
21      /**
22       * Web service operation
23       */
24      @WebMethod(operationName = "F_to_C")
25      public Double F_to_C(@WebParam(name = "f") double f) {
26          Double c = (f-32)*1.8;
27          return c;
28      }
29
30      /**
31       * Web service operation
32       */
33      @WebMethod(operationName = "C_to_F")
34      public Double C_to_F(@WebParam(name = "c") double c) {
35          //TODO write your implementation code here:
36          return null;
37      }
}
```

12. 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;
```

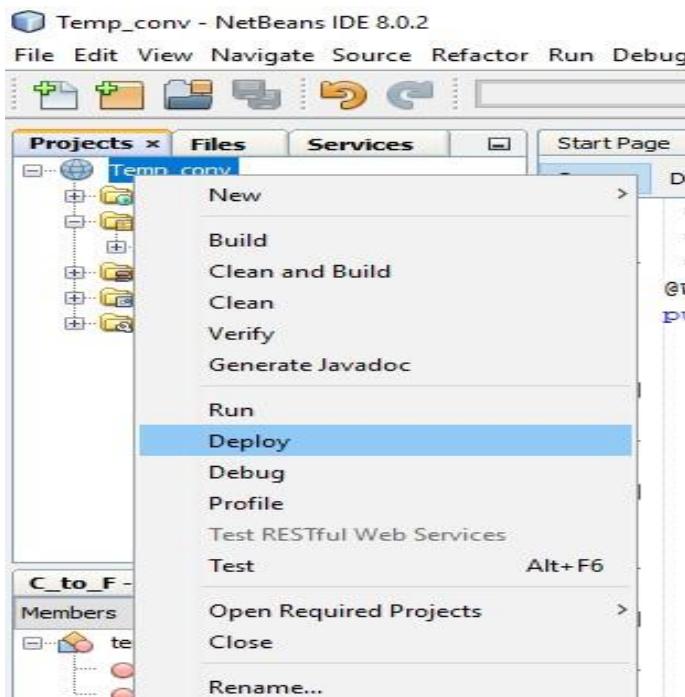


The screenshot shows a Java code editor in NetBeans IDE. The code defines two methods: `F_to_C` and `C_to_F`. The `F_to_C` method converts Celsius to Fahrenheit using the formula $f = (c * 1.8) + 32$. The `C_to_F` method is annotated with `@WebMethod(operationName = "C_to_F")` and contains a comment `//TODO write your implementation code here:`. The code editor highlights the line `return null;` in blue, indicating it is selected for replacement.

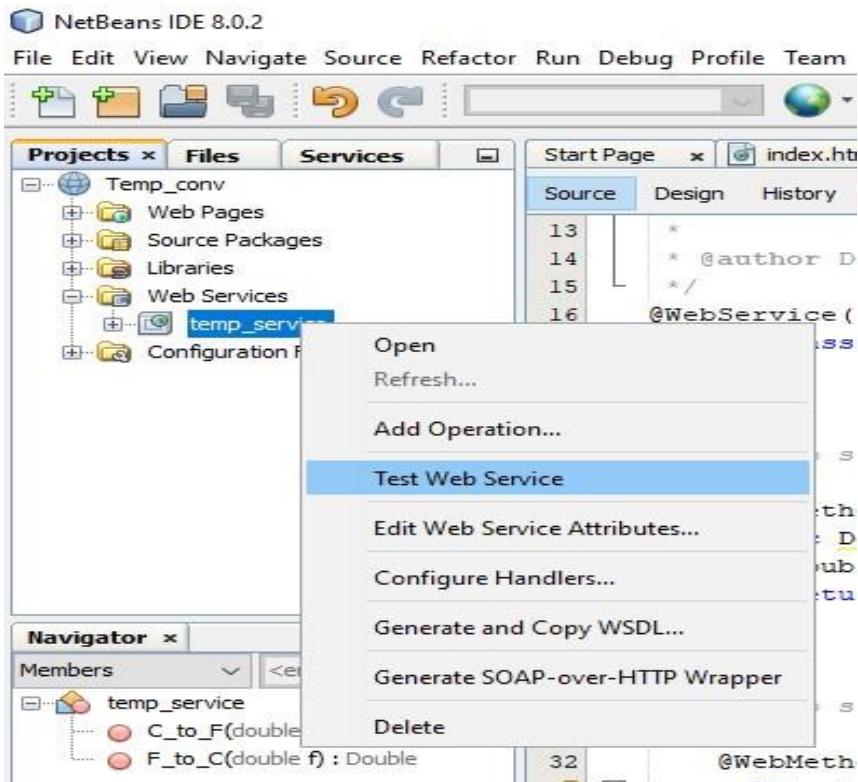
```
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38
```

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

13. Now right click on project name and click on Deploy to deploy your project.



14. To test your web service follow the following process as in picture.



- 15.** 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 convert the entered data into Celsius.

http://localhost:8080/Temp_conv/temp_service?Tester

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)
fToC | ( )
```

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

- 16.** Selected value is in celsius of 56.



fToC Method invocation

Method parameter(s)

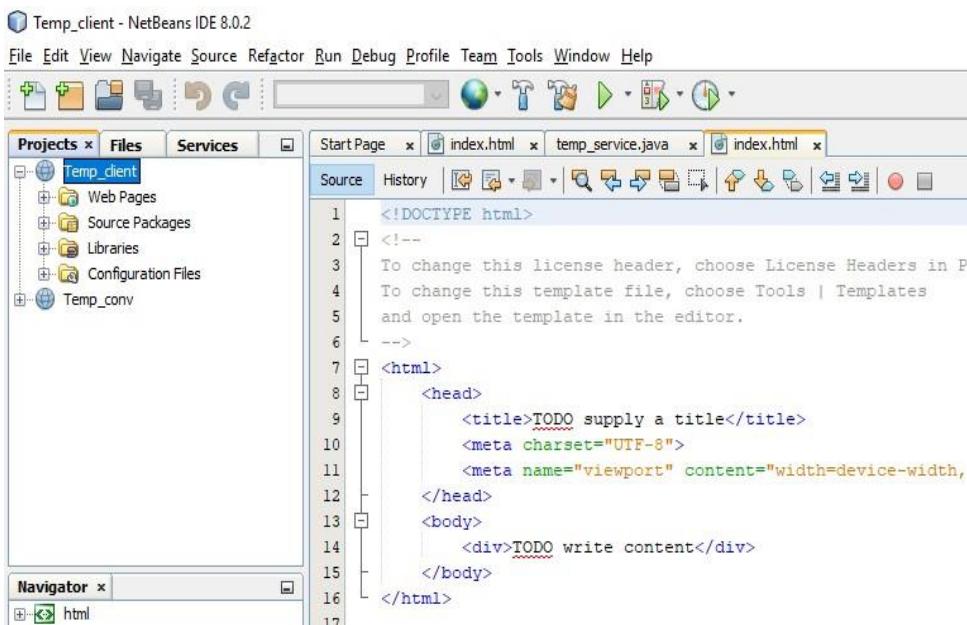
Type	Value
double	56

Method returned

java.lang.Double : "43.2"

SOAP Request

17. Similarly second textbox and button will convert the numeric value into Fahrenheit. Now to consume this web service we are creating a client.
18. Now create a new web application project with **name** as **Temp_client**.

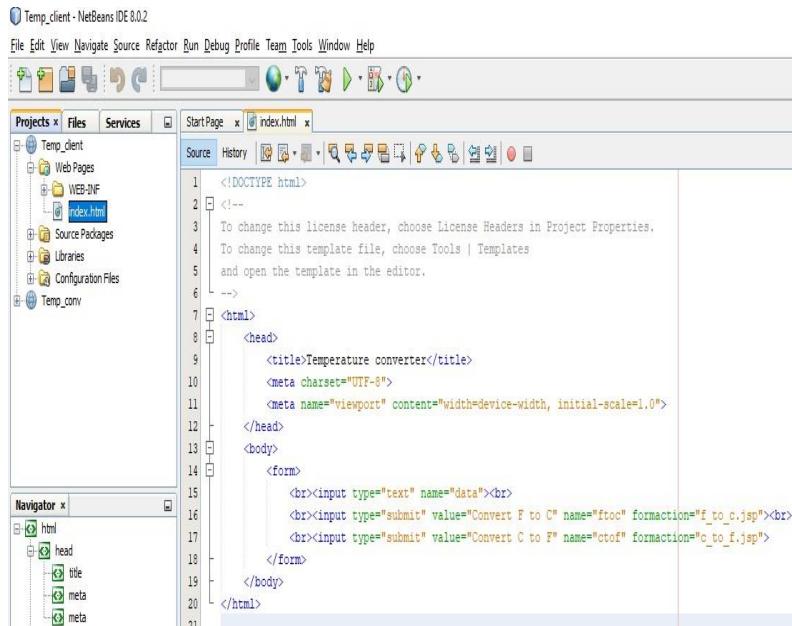


19. Now open the index.html page of Temp_client and write the following code into that.

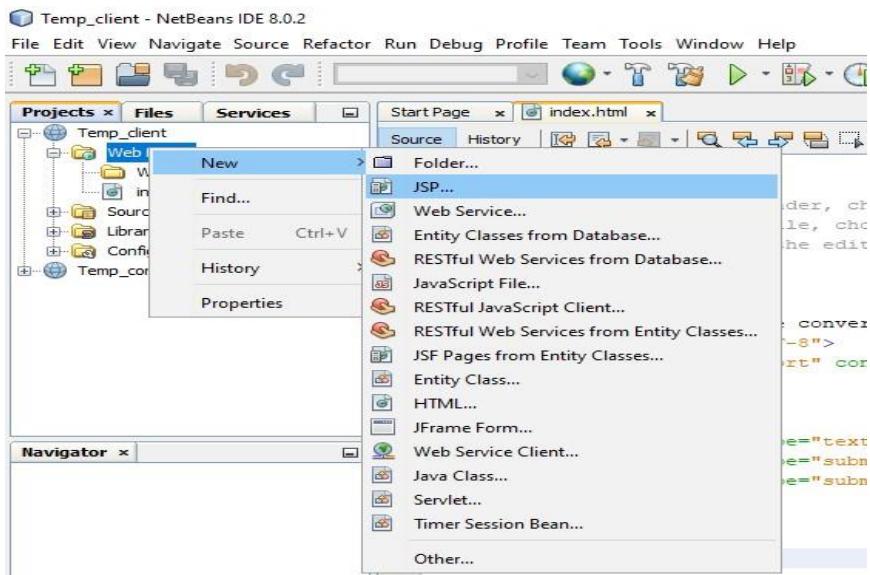
```

<html>
  <head>
    <title>Temperature converter</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
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>

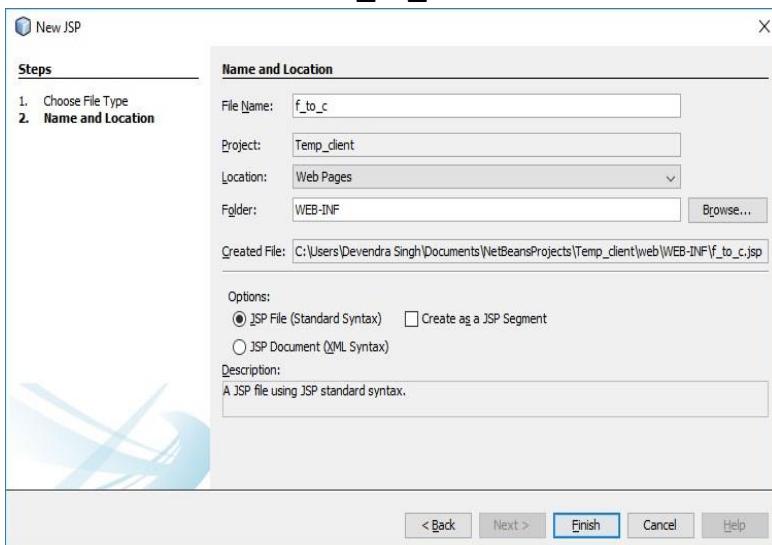
```



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.



22. Now repeat the step number 20 & 21. But give the File Name as c_to_f.

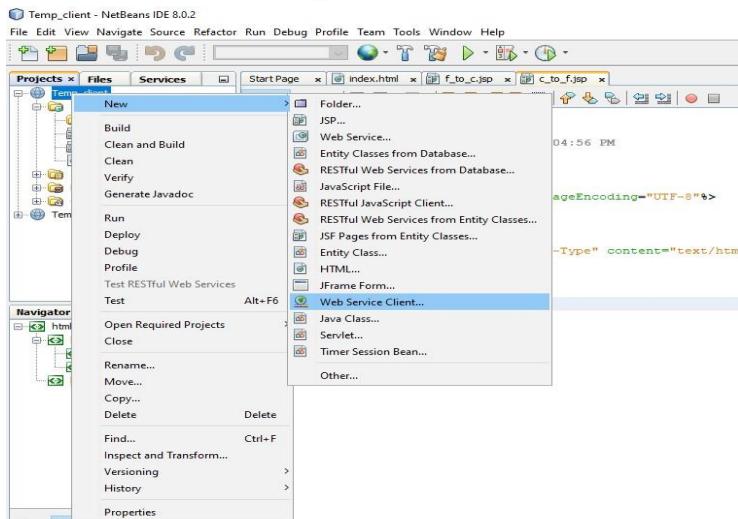
23. Now you have created two jsp files.

```

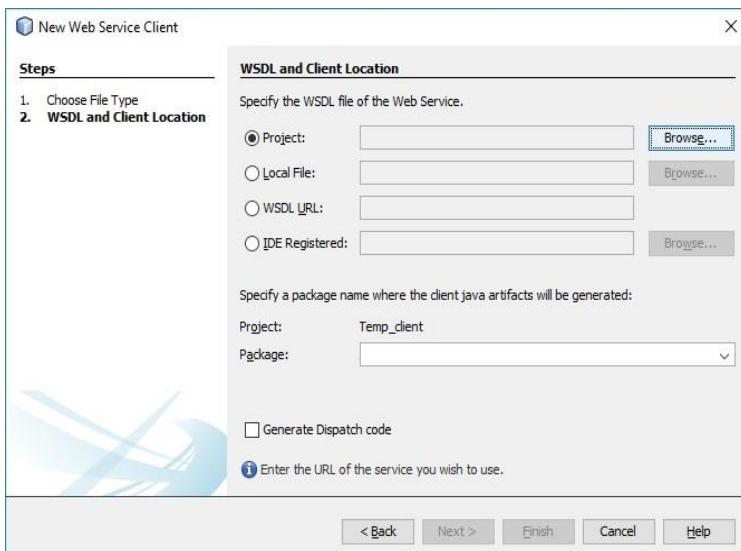
1 Document : c_to_f
2 Created on : Aug 13, 2018, 9:04:56 PM
3 Author : Devendra Singh
4
5
6
7 <%@page contentType="text/html" pageEncoding=
8 <!DOCTYPE html>
9 <html>
10 <head>
11 <meta http-equiv="Content-Type" cont
12 <title>JSP Page</title>

```

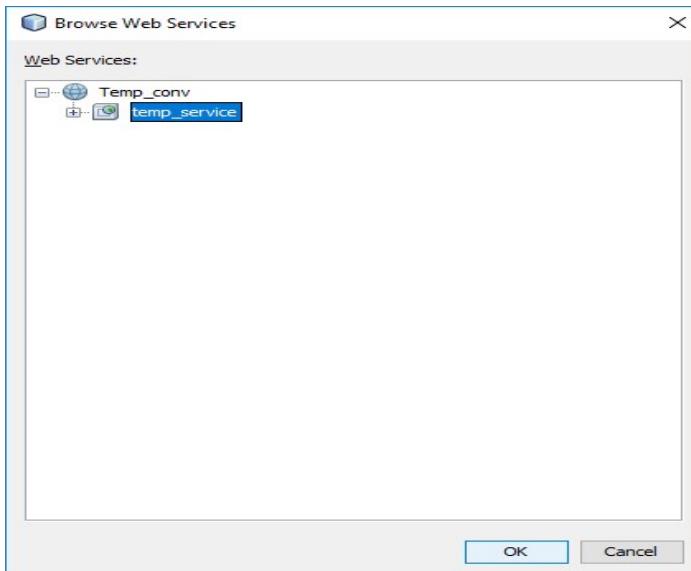
24. Right click on Temp_client and select Web Service Client as below.



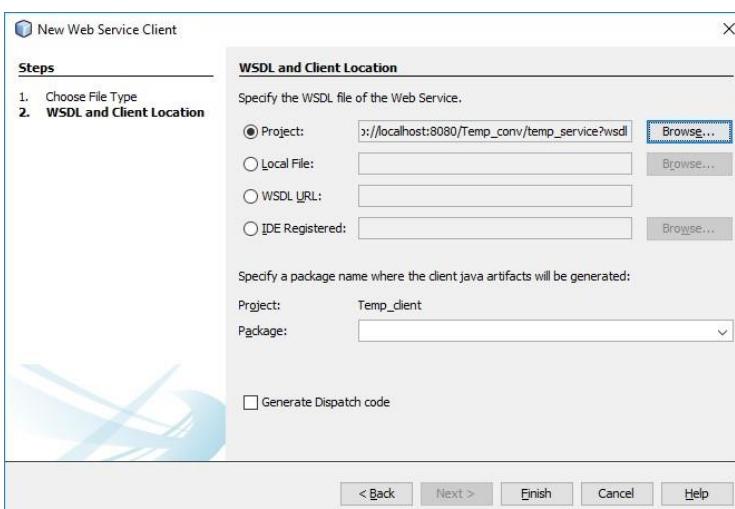
25. Click on Browse.



26. New window will appear and then select temp_service and click on OK.



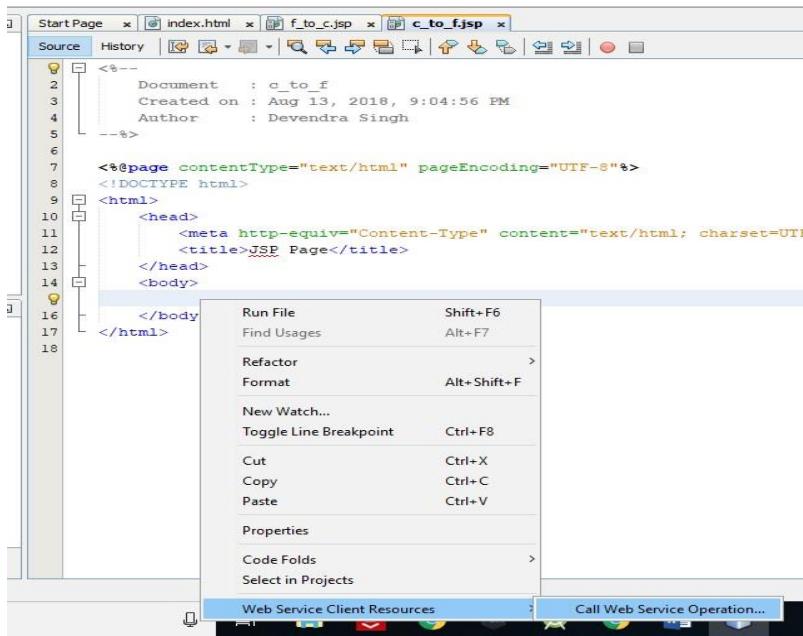
27. Click on Finish.



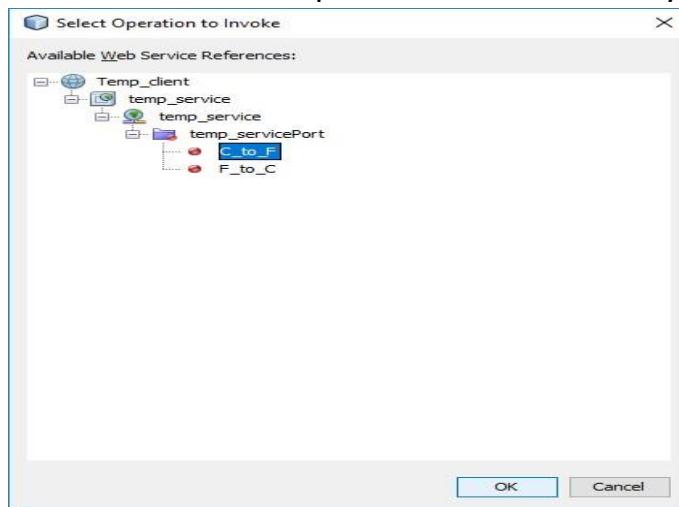
28. Now open the c_to_f.jsp file and delete the selected line.

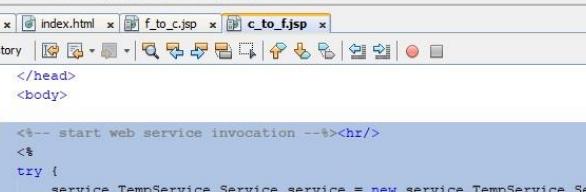
```
<%--  
1 Document : c_to_f  
2 Created on : Aug 13, 2018, 9:04:56 PM  
3 Author : Devendra Singh  
4 --%>  
5  
6  
7 <%@page contentType="text/html" pageEncoding="UTF-8"%>  
8 <!DOCTYPE html>  
9 <html>  
10 <head>  
11 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">  
12 <title>JSP Page</title>  
13 </head>  
14 <body>  
15 <h1>Hello World!</h1>  
16 </body>  
17 </html>  
18
```

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



**30. 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.**





The screenshot shows the Eclipse IDE interface with the Java Development Tools (JDT) perspective selected. The top menu bar includes 'Run', 'Debug', 'Profile', 'Team', 'Tools', 'Window', and 'Help'. Below the menu is a toolbar with icons for file operations like Open, Save, and Print. The 'Start Page' tab is active in the left-hand navigation bar. The main workspace displays a Java code editor with the following content:

```
</head>
<body>

<%-- start web service invocation --%><hr/>
<%
try {
    service.TempService_Service service = new service.TempService_Service();
    service.TempService port = service.getTempServicePort();
    // TODO initialize WS operation arguments here
    double c = 0.0d;
    // TODO process result here
    java.lang.Double result = port.cToF(c);
    out.println("Result = "+result);
} catch (Exception ex) {
    // TODO handle custom exceptions here
}
%>
<%-- end web service invocation --%><hr/>
</body>
</html>
```

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

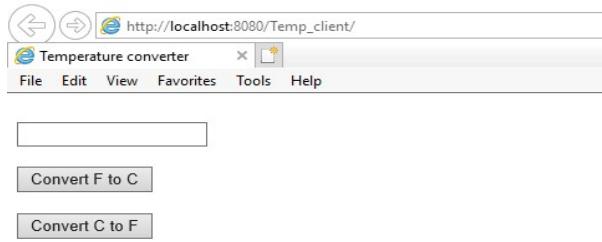
The screenshot shows the NetBeans IDE interface with the title bar "Temp_client - NetBeans IDE 8.0.2". The menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help. The toolbar has icons for file operations like Open, Save, Find, and Run. The Projects tab is selected, showing a hierarchy of files: Temp_client > Web Pages > WEB-INF > wsdl, jax-ws-catalog.xml, C_to_F.jsp, f_to_c.jsp, index.html. The Services tab is also visible. The Navigator tab shows the structure of index.html, including sections for head, body, and hr. The main editor window displays Java code for a web service invocation:

```
</head>
<body>

<%-- start web service invocation --%><hr/>
<%
    String d=request.getParameter("data");
    Integer dd=Integer.parseInt(d);
    try {
        service.TempService_Service service = new service.TempService_Service();
        service.TempService port = service.getTempServicePort();
        // TODO initialize WS operation arguments here
        double c = dd;
        // TODO process result here
        java.lang.Double result = port.cToF(c);
        out.println("Result = "+result);
    } catch (Exception ex) {
        // TODO handle custom exceptions here
    }
%>
<%-- end web service invocation --%><hr/>
</body>
</html>
```

32. 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.

33. Now run the Temp_client project. A window will be open like below.



- 34.** Now you can 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.
- 35.** There are so many methods to consume the web service. But I found it easy.

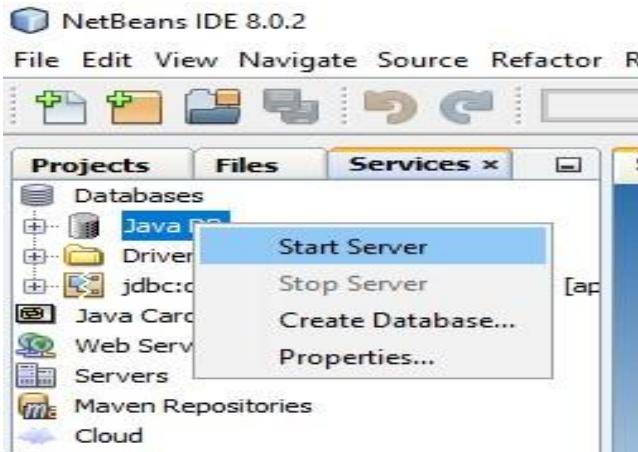
PRACTICAL-2

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.

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 .

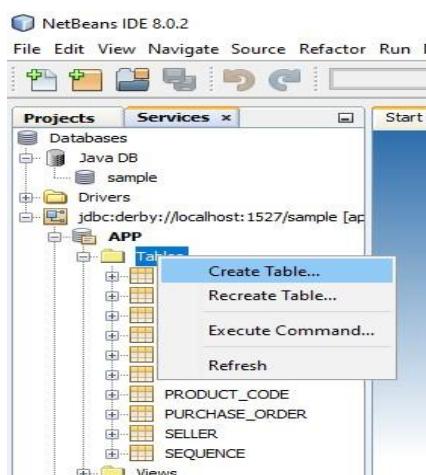


3. Now expand Java DB and right click on sample and then click on connect to connect the sample database with server.

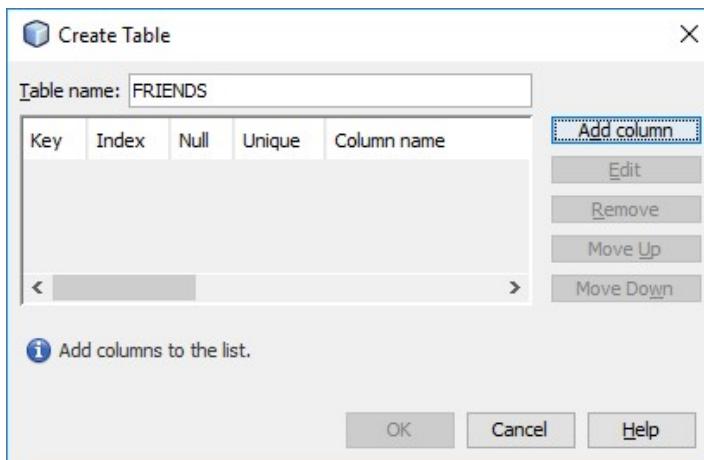


4. Now we are going to create a table in default database sample.

Right click on Table -> Create Table

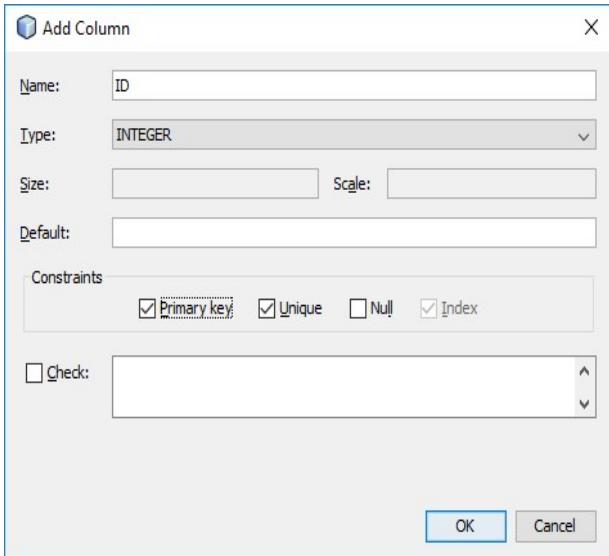


5. Give table name as FRIENDS.

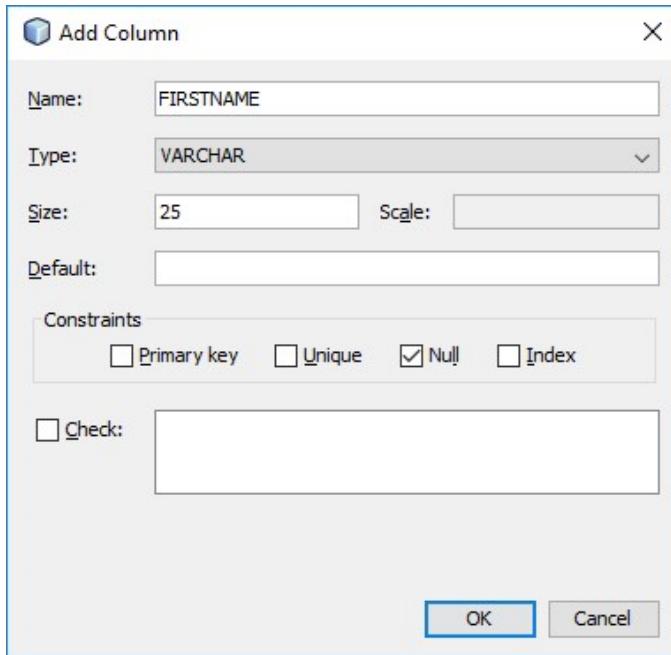


6. click on Add column button to add columns in table.

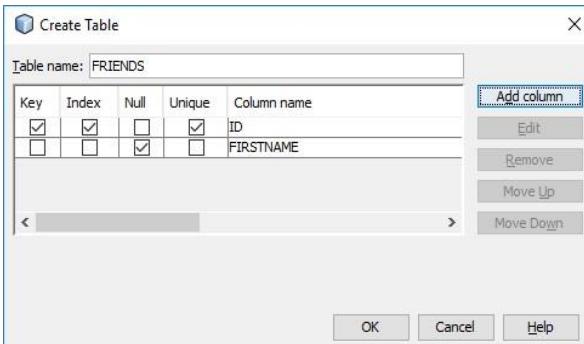
Enter details as in below pic and select Primary key. After that click on OK button.



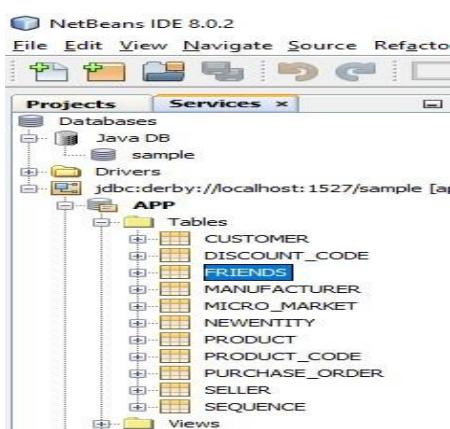
7. Now add second column with following detail. But don't select primary and click on OK button.



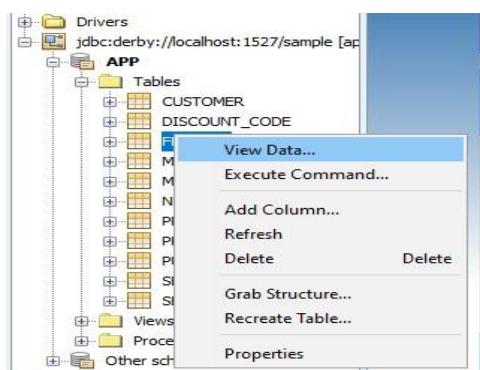
8. click on OK button.



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

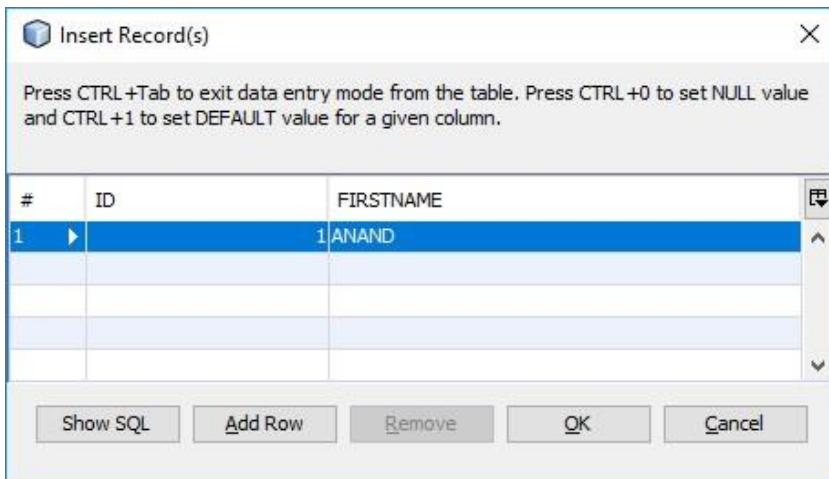


10. Right click on FRIENDS to view and add records into it.



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

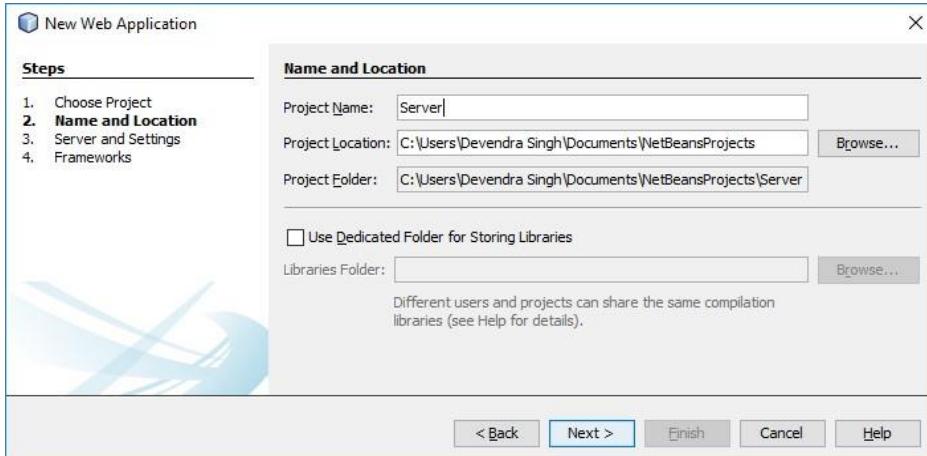
- 12.** Insert a record and then click on Add Row button to insert more record. After that click on OK button to finish.



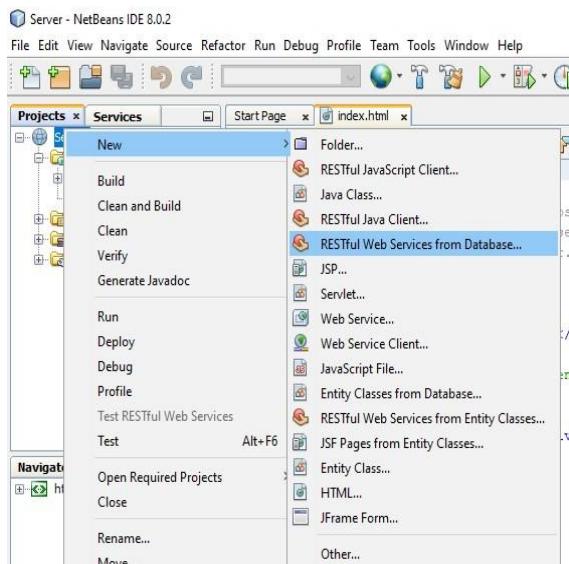
- 13.** As you can see, I have entered 7 records.

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

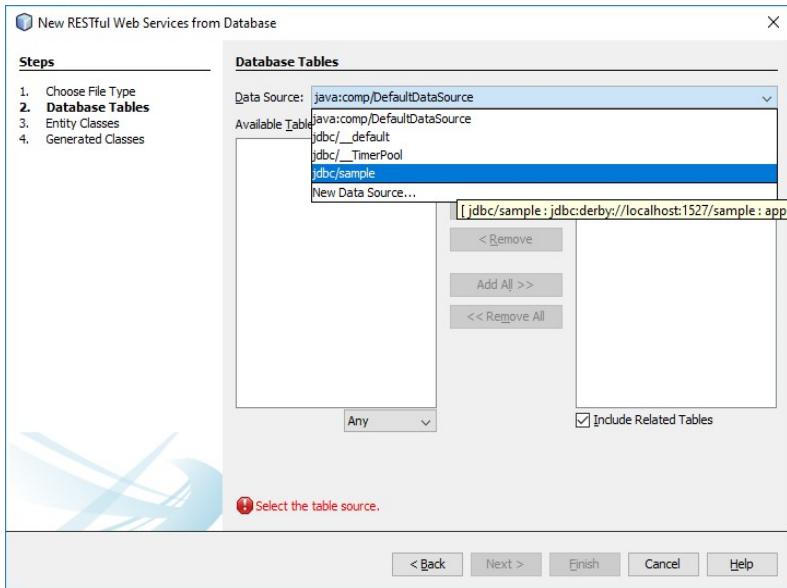
- 14.** create a web application with name Server. After that click on Next and then Finish button.



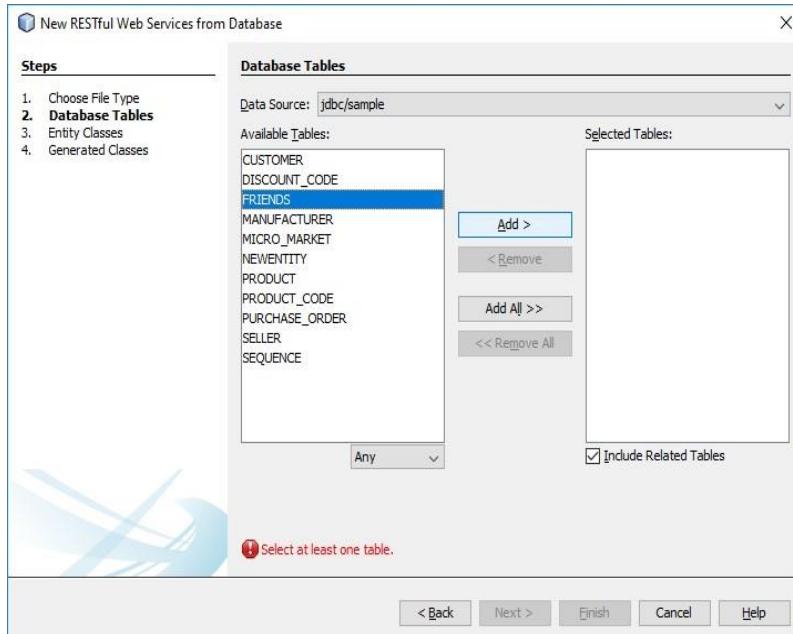
15. Now create a RESTful Web Service from Database by right click on project name.



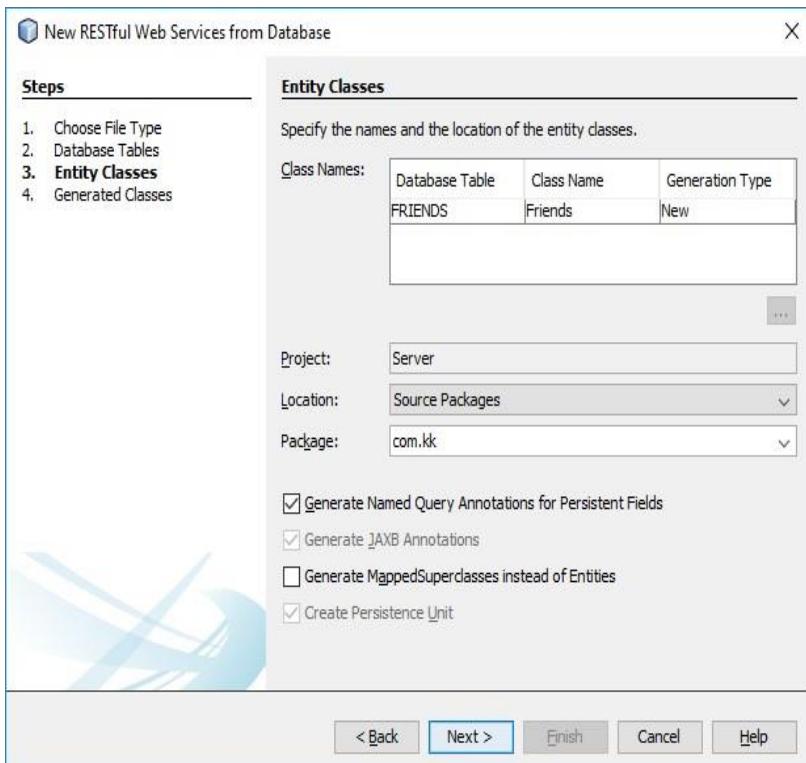
16. Choose Data Source jdbc/sample.



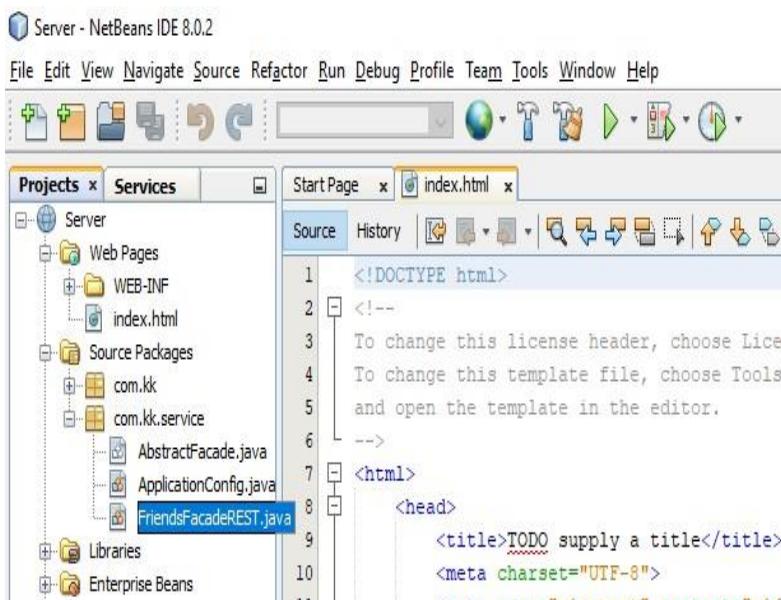
17. Now select FRIENDS and click on Add button. After that click on Next button.



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.

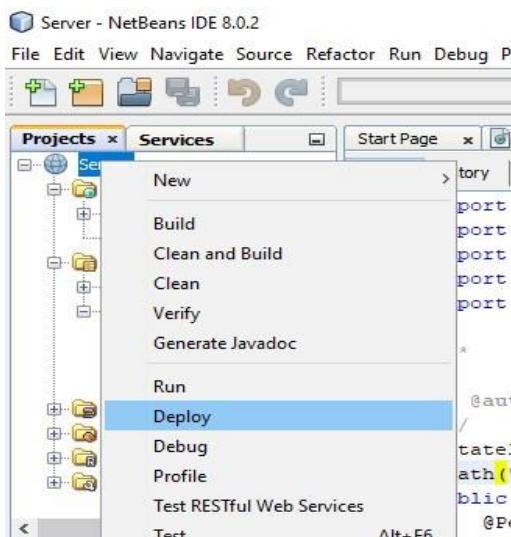
Note : If you get error “javax.ws.rs” doesnot exist, add Java EE 6 API Library in NetBeans IDE by doing myProject->Properties->Libraries->Add Library



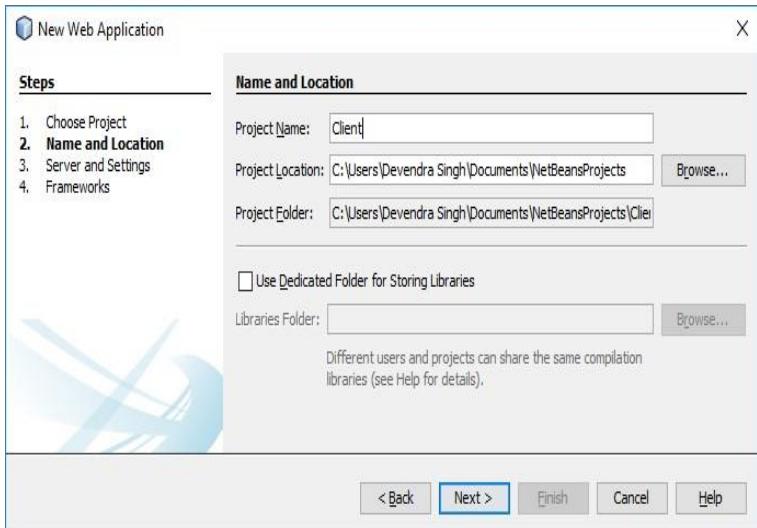
The screenshot shows an IDE interface with multiple tabs at the top: "Start Page", "index.html", and "FriendsFacadeREST.java". The "FriendsFacadeREST.java" tab is active, displaying Java code for a RESTful service. The code includes annotations like @Stateless, @Path, @PersistenceContext, @POST, @Override, @Consumes, @PUT, @Path, and @PathParam. The code is color-coded for syntax highlighting, and several yellow lightbulb icons are visible, indicating potential issues or suggestions.

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

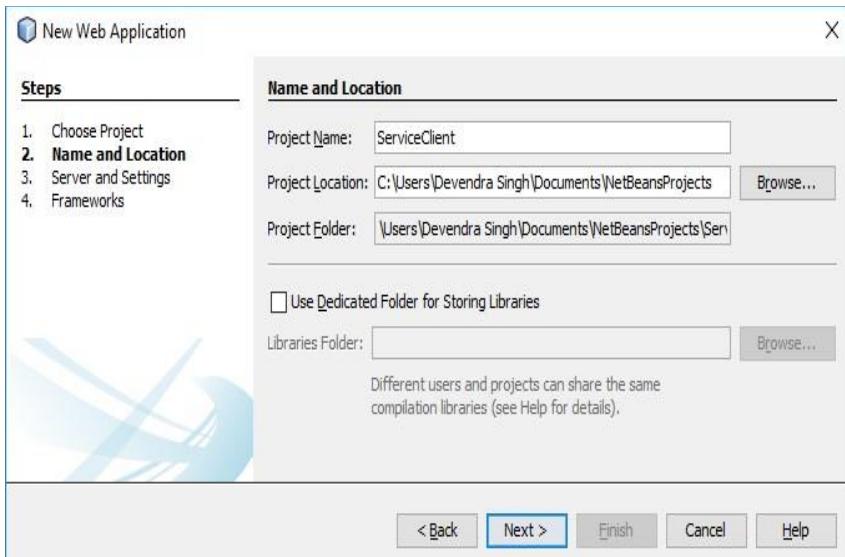
- 21.** After that right click on project name and Deploy it.



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

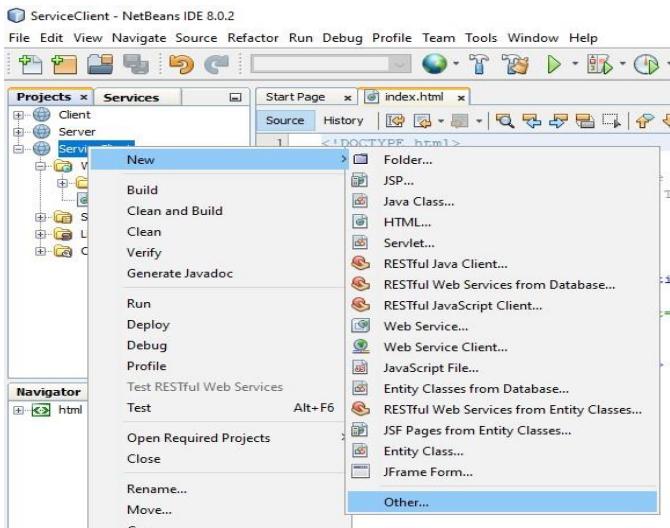


23. Create a Web Application with name ServiceClient.

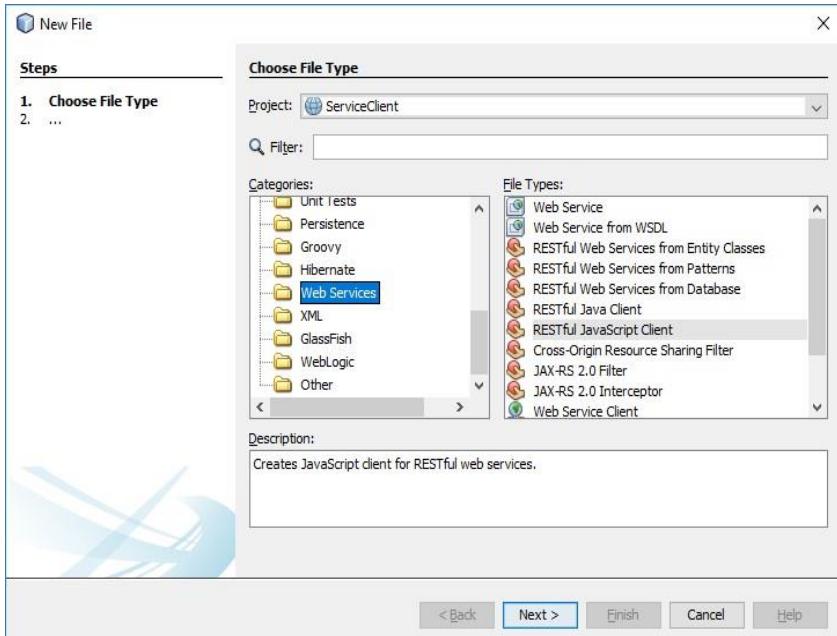


24. Now create a RESTful Java Client.

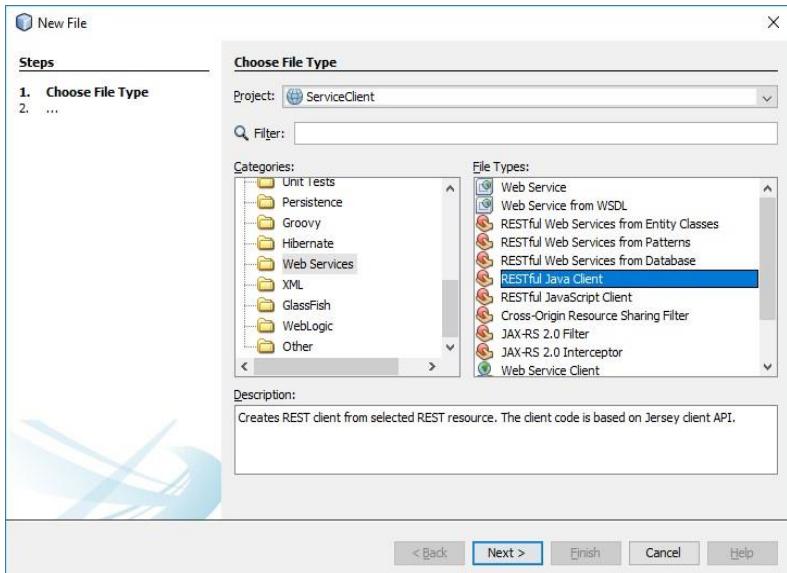
Right click on ServiceClient -> New -> Other.



25. Drag down and select Web Services and in side panel select RESTful Java Client.

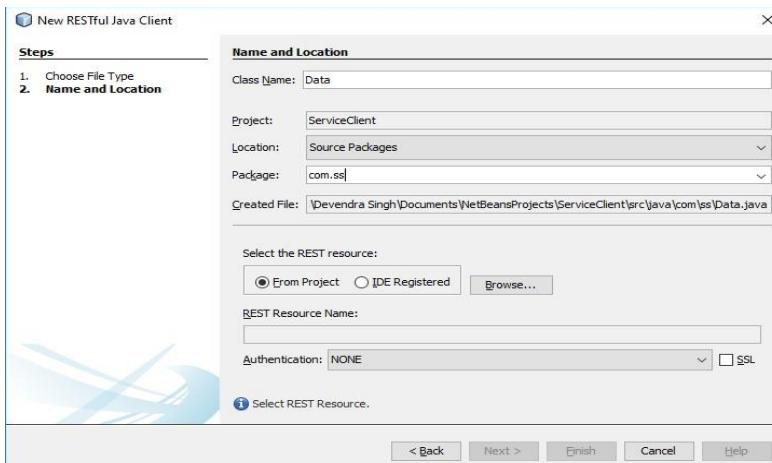


26. After select RESTful Java Client click on Next.

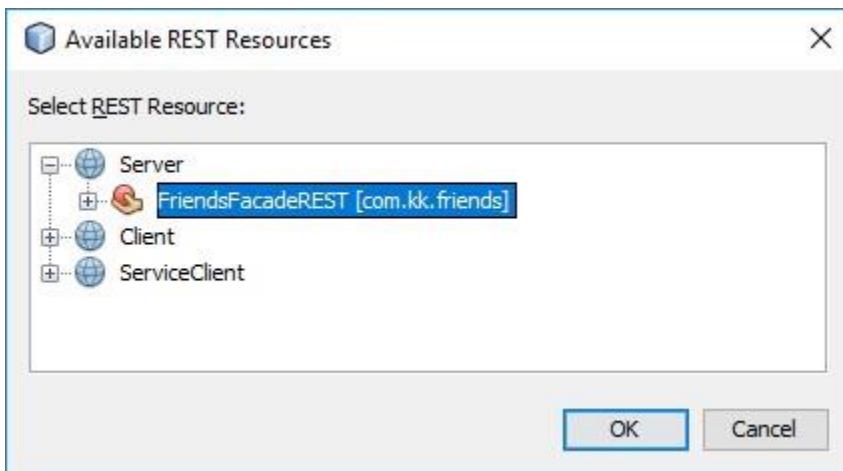


27. Enter following data. Class Name -> Data

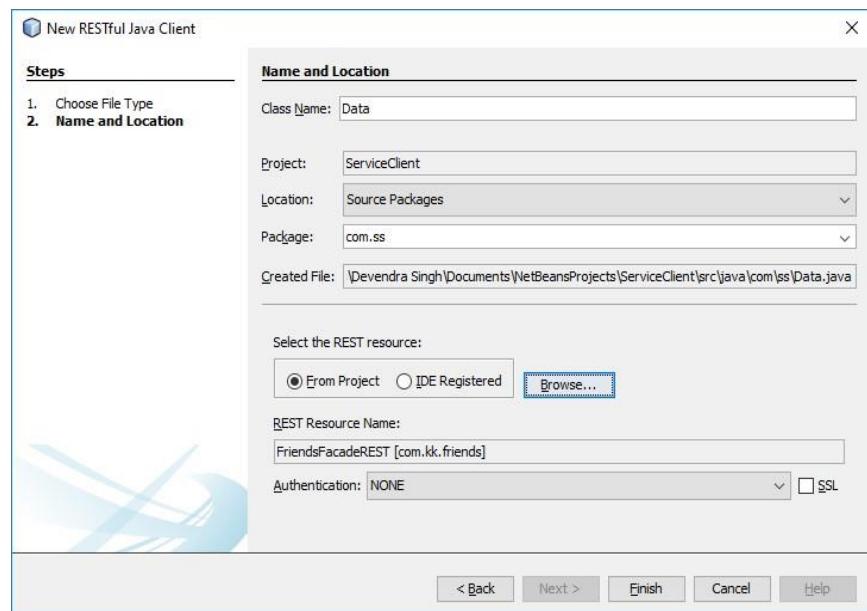
Package -> com.ss



28. Now click on Browse button and select the option into the below pic. After select click on OK button.

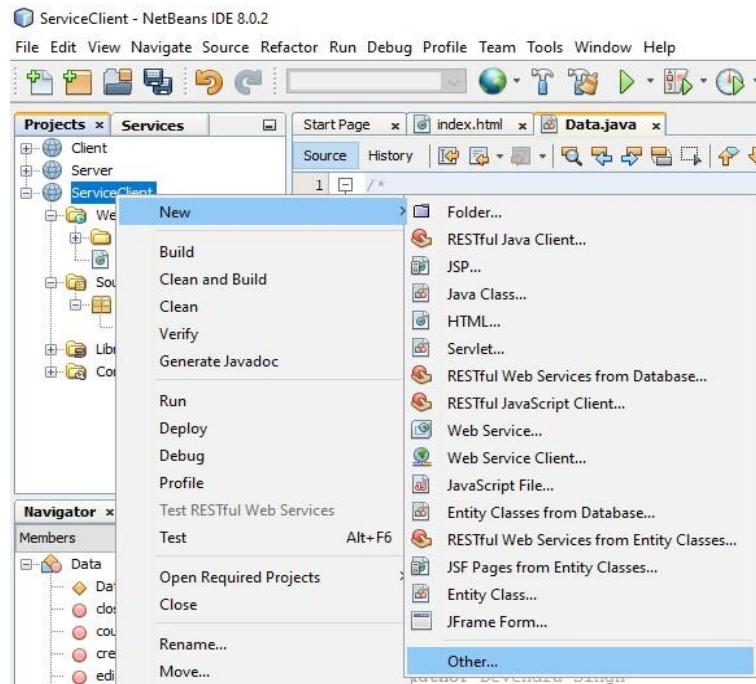


29. Click on Finish.

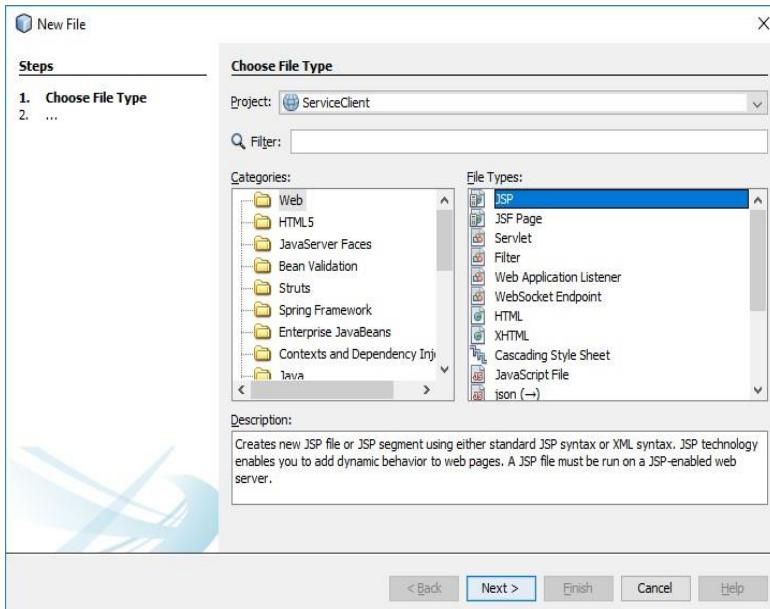


30. Now create a JSP page.

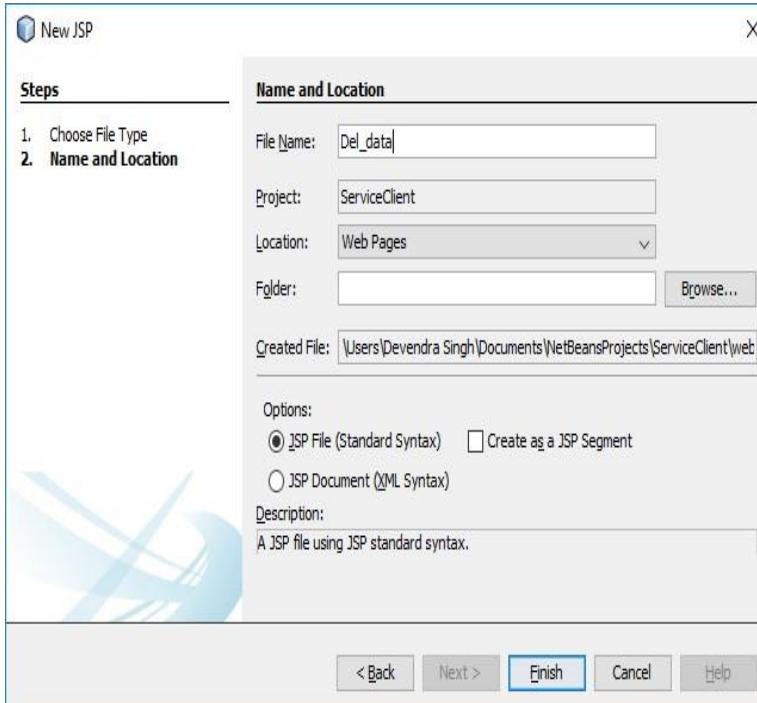
Right click on ServiceClient -> New -> Other



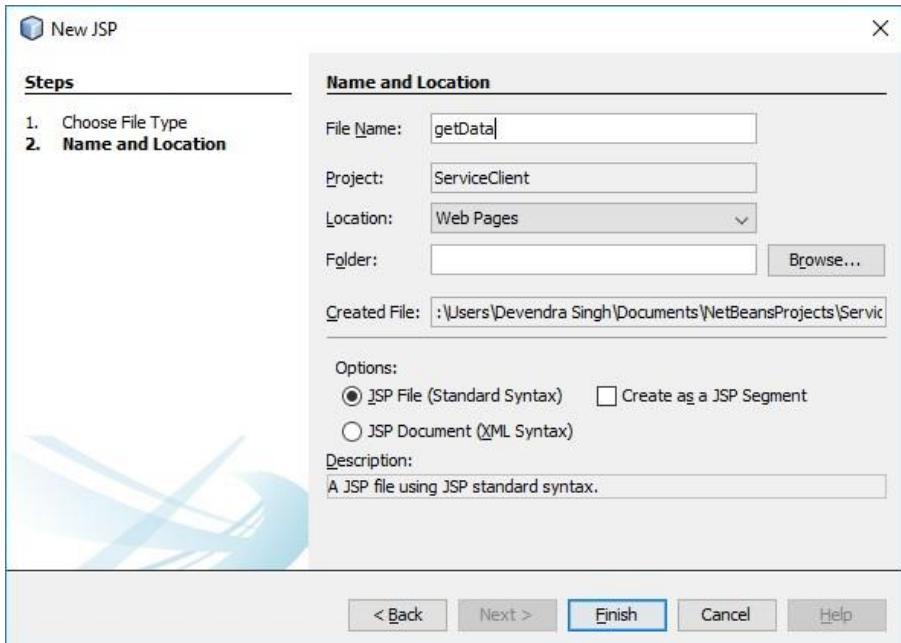
31. Select Web in Categories section -> Select JSP and click on Next button.



32. Enter File Name Del_data and click on Finish button.

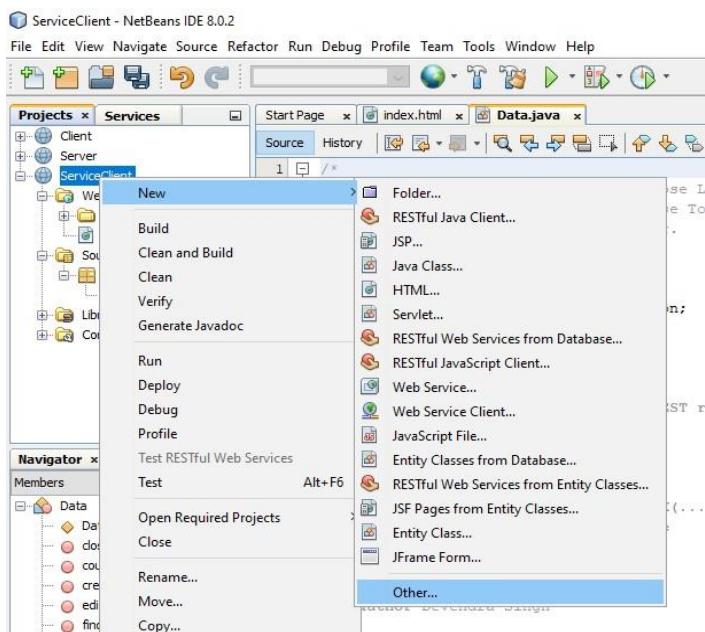


33. Now create one more JSP file by follow the step number 30, 31 & 32. But File Name will be getData.

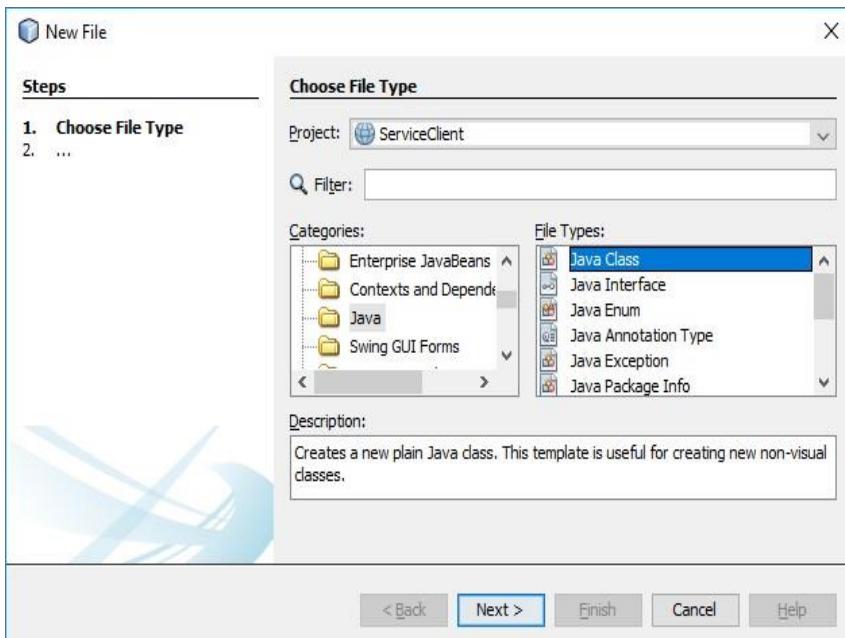


34. Now create a Java class.

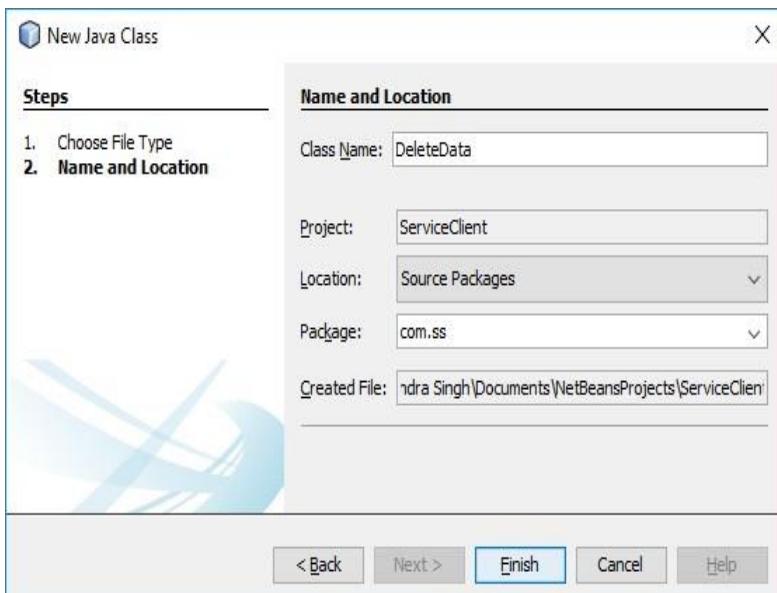
Right click on ServiceClient -> New -> Other



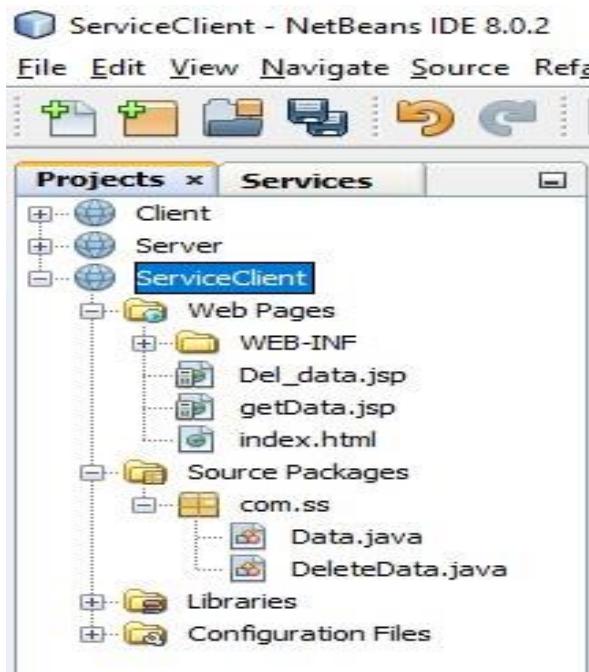
35. In Categories select Java and in File Types select Java Class. Click Next button.



36. Enter Class Name DeleteData and Package com.ss. After that click on Finish.



37. Your project file structure will look like below.



38. Now open the index.html of ServiceClient project by double click on it and add the following code in between body tag.

```
<form>
    <h2>One-way Operation</h2><br>
    <input type="text" name="ID" placeholder="Enter ID"><br><br>
    <input type="submit" formaction="Del_data.jsp" value="Delete
Data"><br>

    <h1>-----</h1>

    <h2>Request-Response operation</h2><br><br>
    <input type="submit" formaction="getData.jsp" value="Get Data"> </form>
```

```
4 To change this template file, choose Tools | Templates  
5 and open the template in the editor.  
-->  
6 <html>  
7   <head>  
8     <title>TODO supply a title</title>  
9     <meta charset="UTF-8">  
10    <meta name="viewport" content="width=device-width, initial-scale=1.0">  
11  </head>  
12  <body>  
13    <form>  
14      <h2>One-way Operation</h2><br>  
15      <input type="text" name="ID" placeholder="Enter ID"><br><br>  
16      <input type="submit" formaction="Del_data.jsp" value="Delete Data"><br>  
17      <h1>-----</h1>  
18      <h2>Request-Response operation</h2><br><br>  
19      <input type="submit" formaction="getData.jsp" value="Get Data">  
20    </form>  
21  </body>  
22 </html>
```

39. Now open DeleteData.java file by double click on it and add the following code in the class and save it by pressing Ctrl+S.

```
public static void deldata(String id){  
    String a = id;  
    Data ob = new Data();    ob.remove(a);  
    System.out.println("Data is deleted.");  
}
```

The screenshot shows a Java code editor window with the file 'DeleteData.java' open. The code defines a static void method 'deldata' that takes a String parameter 'id', creates a new 'Data' object, removes the item with ID 'a', and prints a confirmation message to the console.

```
1  /*
2   * To change this license header, choose License Headers in Project Properties.
3   * To change this template file, choose Tools | Templates
4   * and open the template in the editor.
5   */
6  package com.ss;
7
8  /**
9   * 
10  * @author Devendra Singh
11  */
12 public class DeleteData {
13     public static void deldata(String id){
14         String a = id;
15         Data ob = new Data();
16         ob.remove(a);
17         System.out.println("Data is deleted.");
18     }
19
20 }
21
```

40. Now open the Del_data.jsp file and replace the contents of body with the following code.

```
<%@ page import="com.ss.DeleteData"%>
<%
    String id = request.getParameter("ID");
    DeleteData.deldata(id);
%>
```

The screenshot shows a JSP editor window with the file 'Del_data.jsp' open. The code imports the 'DeleteData' class and calls its 'deldata' method, passing the value of the 'ID' parameter from the request. The code is highlighted in green, indicating it is Java code within the JSP.

```
<%-->
1 Document      : Del_data
2 Created on   : Aug 16, 2018, 8:47:49 PM
3 Author        : Devendra Singh
4 --%>
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/
12         <title>JSP Page</title>
13     </head>
14     <body>
15         <%@ page import="com.ss.DeleteData"%>
16         <%
17             String id = request.getParameter("ID");
18             DeleteData.deldata(id);
19         %>
20     </body>
21 </html>
22
```

41. Now open the getData.jsp file and replace the contents of html tag with the following code and save it.

```
<head>
    <title>TODO supply a title</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <style>        table {            font-
family: arial, sans-serif;
            border-collapse: collapse;
        }

        td, th {
            border: 1px solid #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>

```

```

</script>

</head>
<body>
<table id="myTable">
<tr>
<th> ID</th>
<th>NAME</th>
</tr>
</table>

```

```

</body>

```

The screenshot shows the Eclipse IDE interface with the 'getData.jsp' file open in the editor. The code is as follows:

```

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<title>TODO supply a title</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<style>
table {
    font-family: arial, sans-serif;
    border-collapse: collapse;
}

td, th {
    border: 1px solid #000000;
    text-align: center;
    padding: 8px;
}

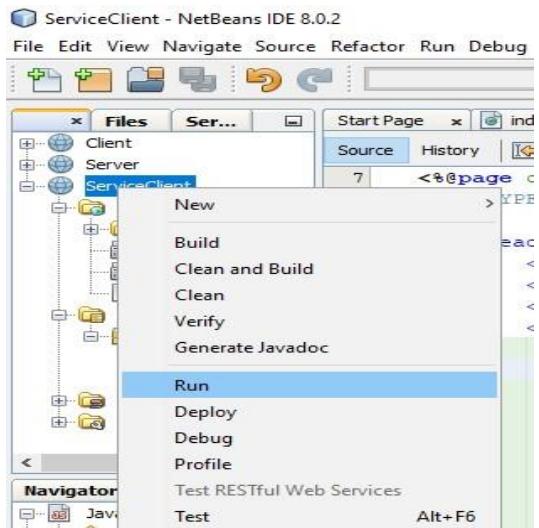
</style>
<script>
var request = new XMLHttpRequest();
request.open('GET', 'http://localhost:8080/Server/webresources/com.kk.friends/');
request.onload = function () {
// begin accessing JSON data here
var data = JSON.parse(this.response);

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].name;
}
</script>

```

The Navigator panel on the left shows the project structure with files like 'Del_data.jsp', 'index.html', 'com.ss', 'Data.java', and 'DeleteData.java'. The CSS panel shows rules for 'table', 'td', and 'th' elements.

42. Now Run the ServiceClient project.



43. On run the project following window will open in browser.

A screenshot of a web browser window. The address bar shows "http://localhost:8080/ServiceClient/". The page content is divided into two sections by a horizontal dashed line. The top section is titled "One-way Operation" and contains a text input field labeled "Enter ID" and a button labeled "Delete Data". The bottom section is titled "Request-Response operation" and contains a button labeled "Get Data".

If you will click on Delete Data button, record of entered ID will be deleted.

If you will click on Get Data button Request for the data will be send and it will return the data as response. Hence it is two way operation.

44. By click on Get Data button.

A screenshot of a web browser window titled "TODO supply a title". The address bar shows "http://localhost:808". The menu bar includes "File", "Edit", "View", "Favorites", and "Tools". Below the menu is a table with two columns: "ID" and "NAME". The data rows are:

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

45. Entering ID 8 and clicking on Delete Data button.

The browser window has tabs for "One-way Operation" and "Request-Response operation".

- One-way Operation:** A text input field contains "8" and a button labeled "Delete Data" is highlighted in blue.
- Request-Response operation:** A button labeled "Get Data" is visible.

46. Now go back and again click on Get Data button. Refresh the page, You will see data with id number 8 is deleted.

A screenshot of a web browser window titled "TODO supply a title". The address bar shows "http://localhost:8080/ServiceClient/". The menu bar includes "File", "Edit", "View", "Favorites", and "Tools". Below the menu is a table with two columns: "ID" and "NAME". The data rows are:

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

PRACTICAL 2 (Another approach)

Create new Web Project give it name pract_2

Press next

Press Finish

Right Click on Project pract_2 and select new then select web service

Give

Name to web service as myws

Give package name as ws

Press Finish

Go to service

Select java DB

Right click on java DB select Start server

Right click on connection string and select connect

```

1 /*
2  * To change this license header, choose License Headers in Project Properties.
3  * To change this template file, choose Tools | Templates.
4  * and open the template in the editor.
5 */
6 package ws;
7
8 import javax.jws.WebService;
9 import javax.jws.WebMethod;
10 import javax.jws.WebParam;
11
12 /**
13  * Author: mahesh
14  */
15
16 @WebService(serviceName = "myws")

```

The Output window shows: Fri Aug 17 12:33:50 IST 2018 : Apache Derby Network Server - 10.9.1.0 - (1344872) started and ready to accept connections on port ^

Expand connection string

It will show APP

Expand APP it will show table. Right click on table and select create table.

Give name to table as tan

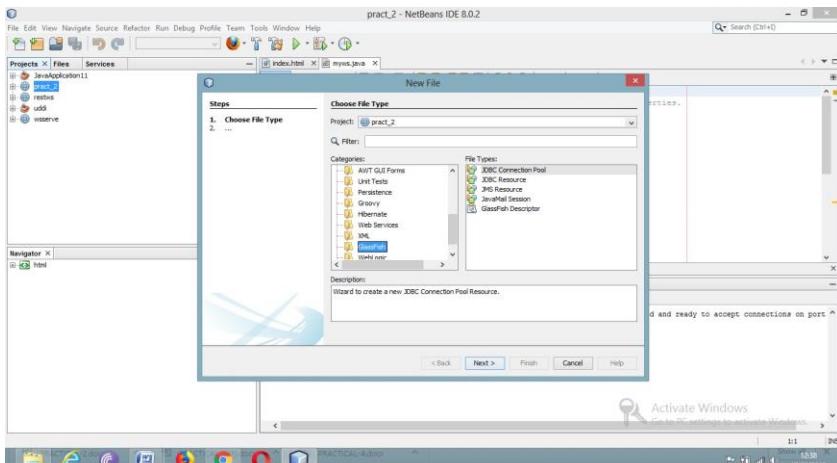
Then select add column

I have given name to table as Tan with column as rn and name

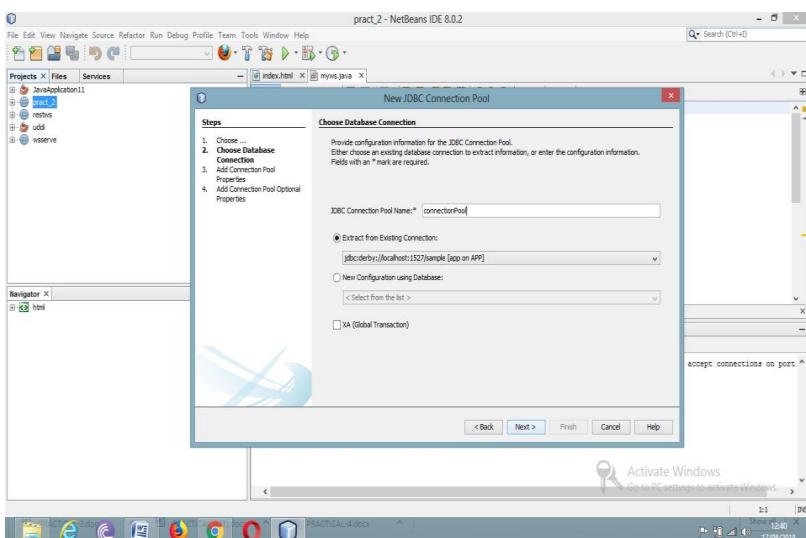
Come back to project

Right click Project name pract_2 select new select others in it select glassfish

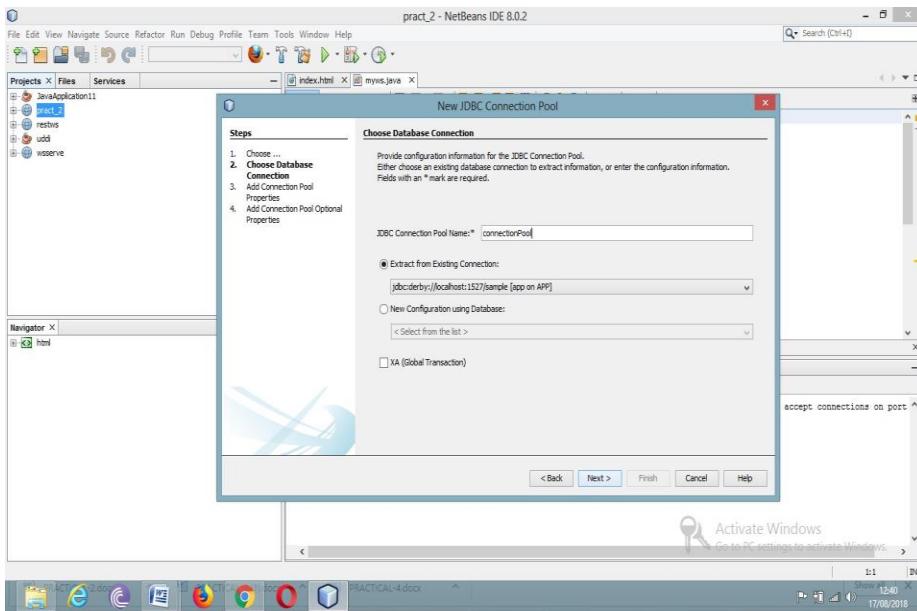
server In that select jdbc connection pool



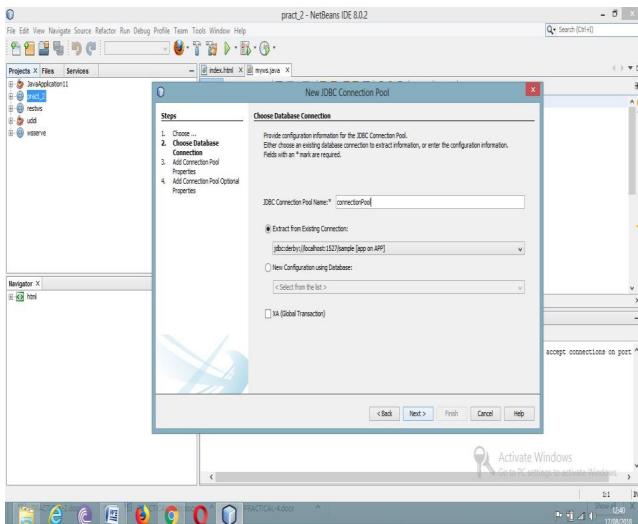
Press next



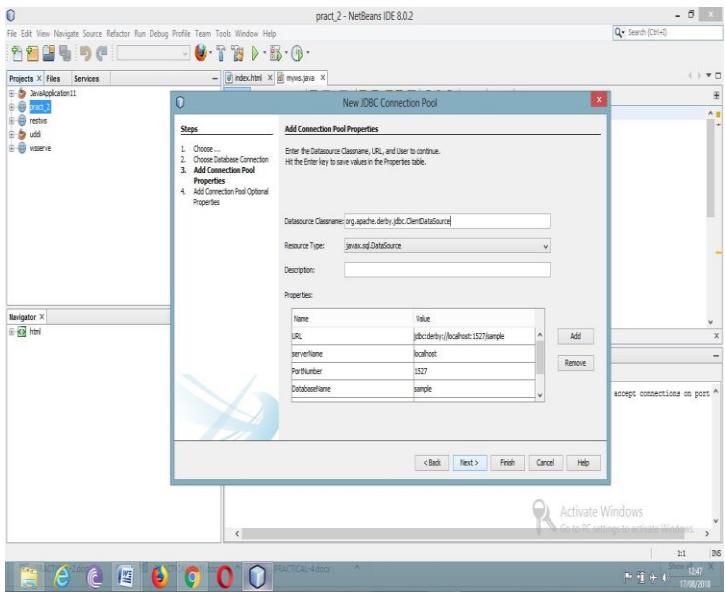
Press next



Press next



Press next

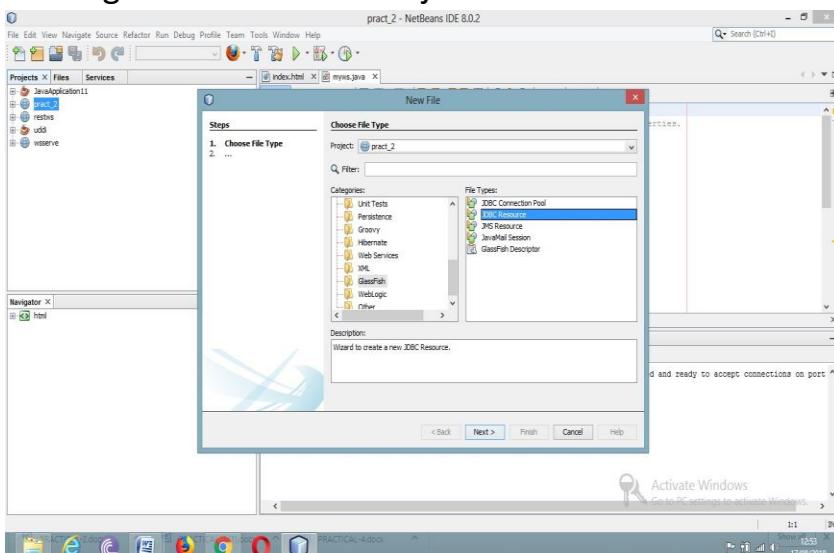


Press next

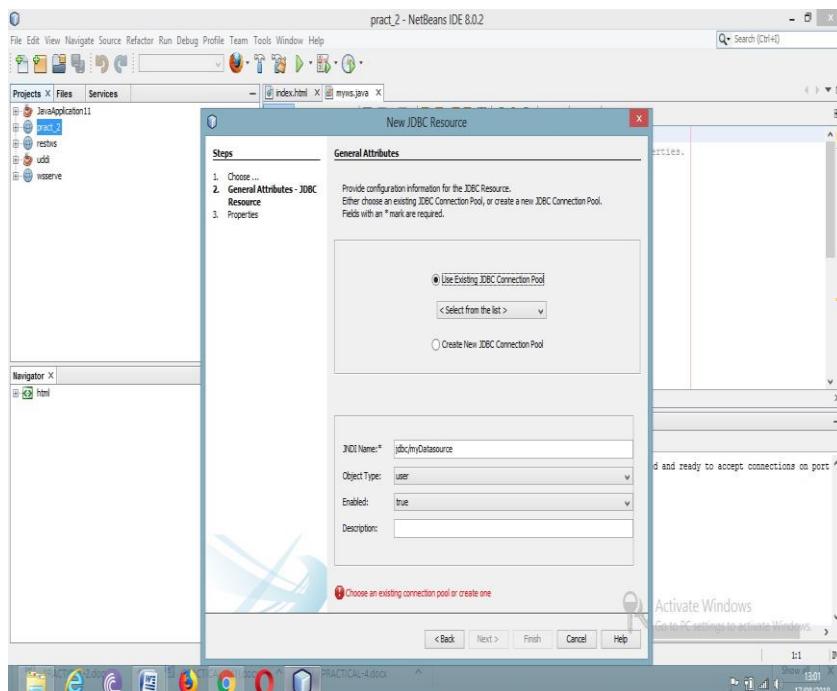
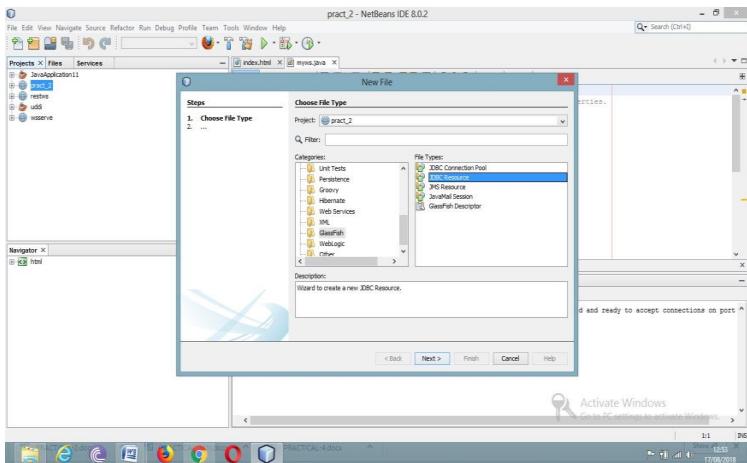
Press finish

Right click project name select new select other

Select glassfish Select jdbc resource



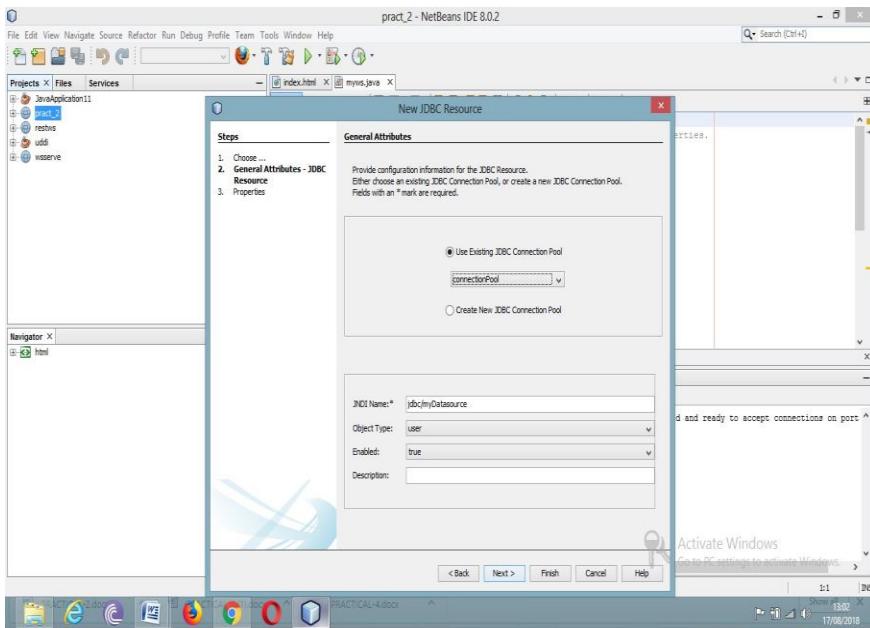
Press next



Press next

Click on existing connection pool

Select from combobox connection pool



Change JNDI name to jdbc/mydb

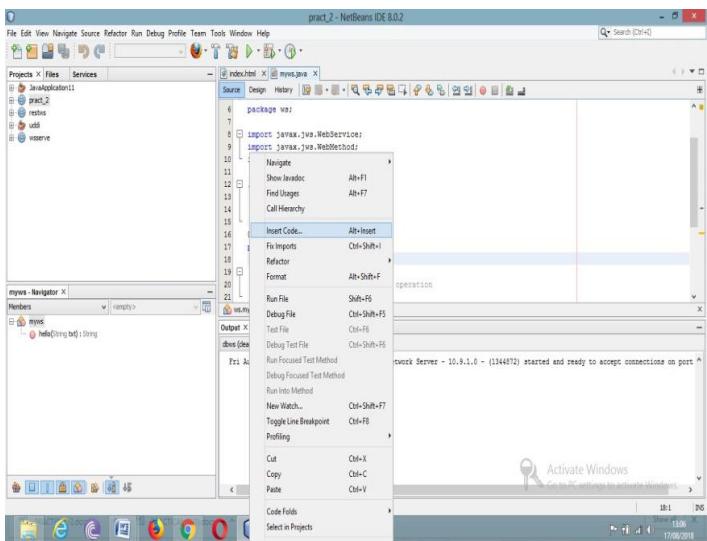
Press next

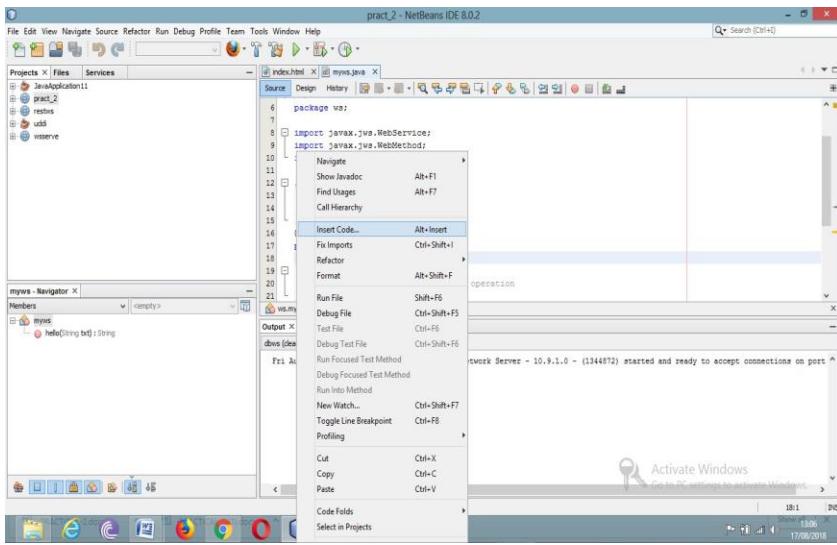
Press finish

It will open file myws.java

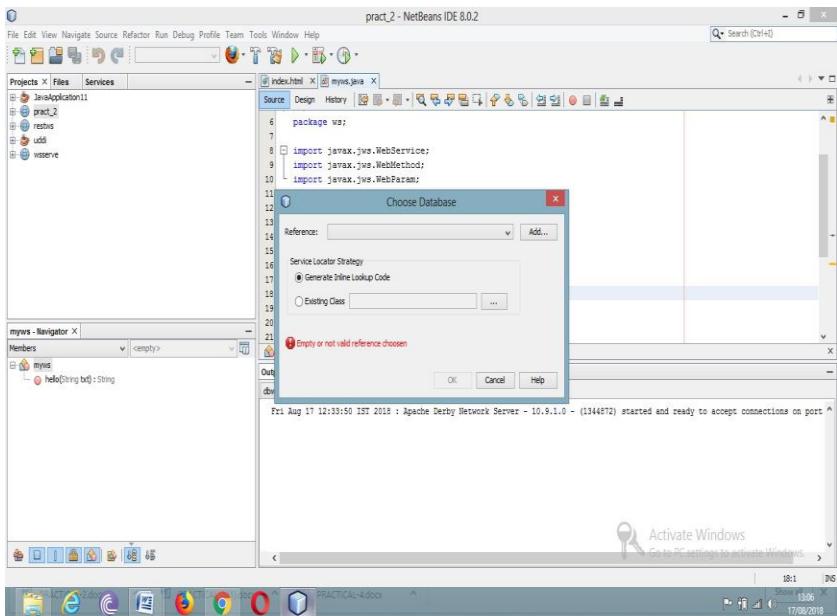
Right click on line number 18 (inside class myws)

Select insert code



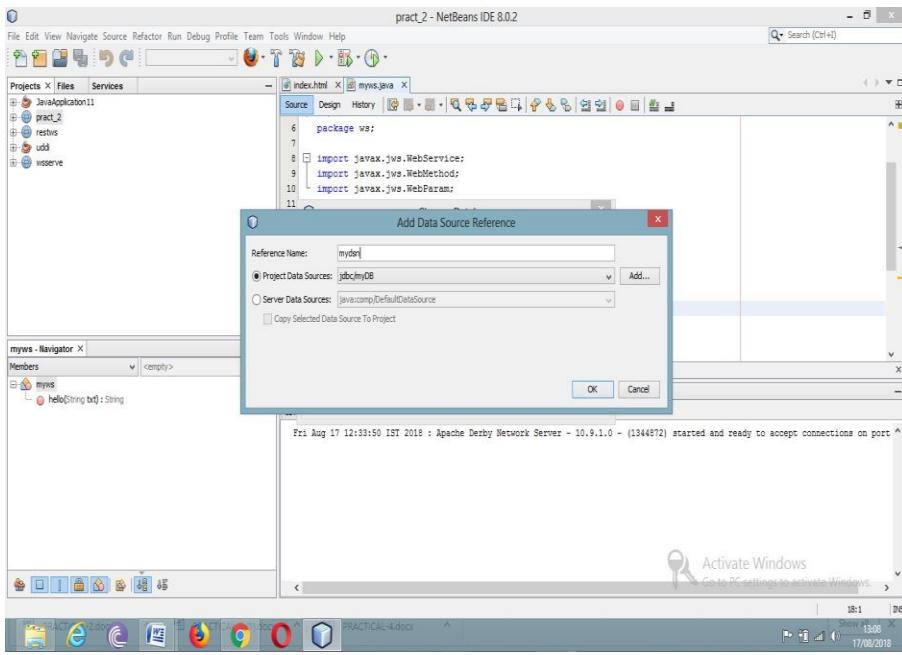


Select use database



Click on add

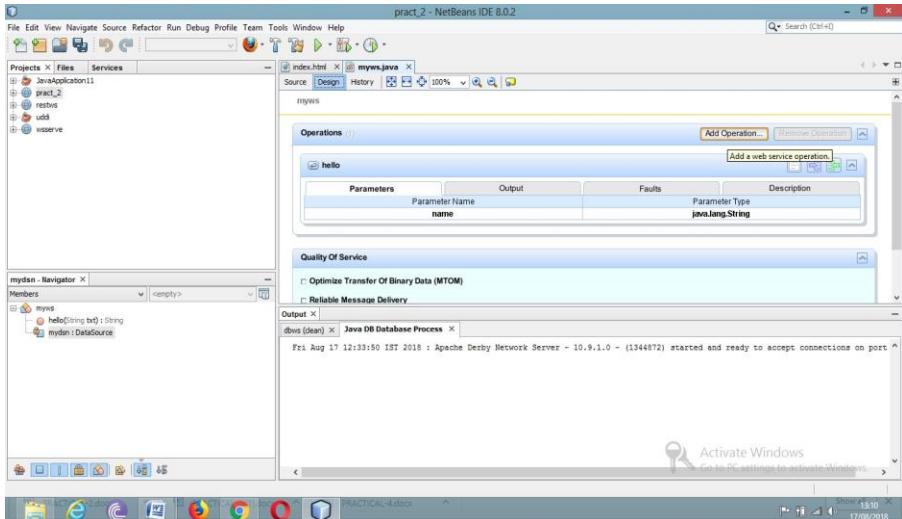
Give any name I have given my dsn



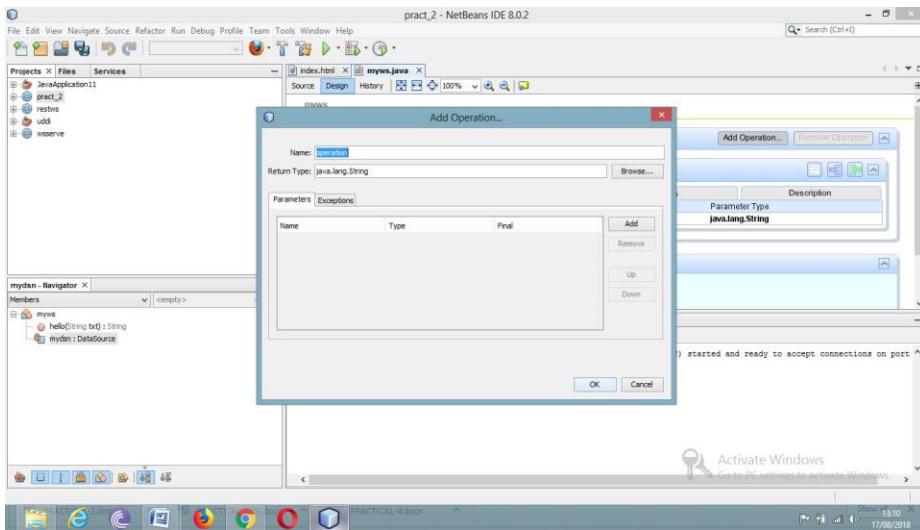
Press ok press ok

Now click on design

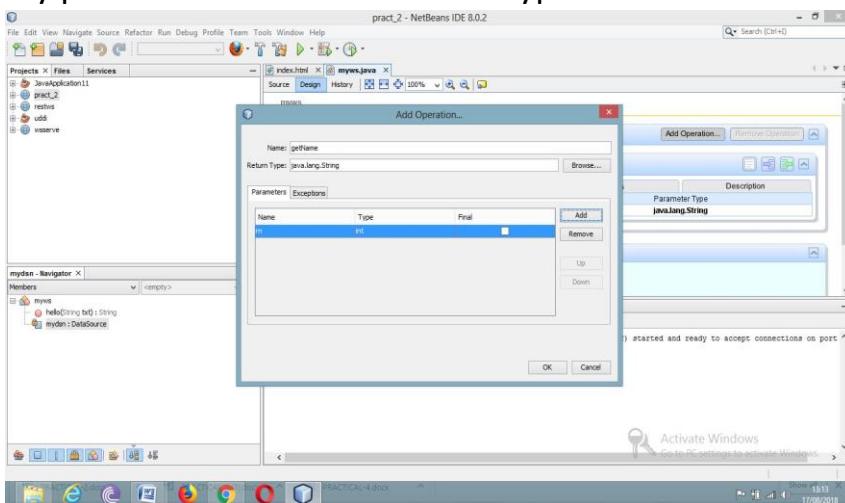
Select add operation



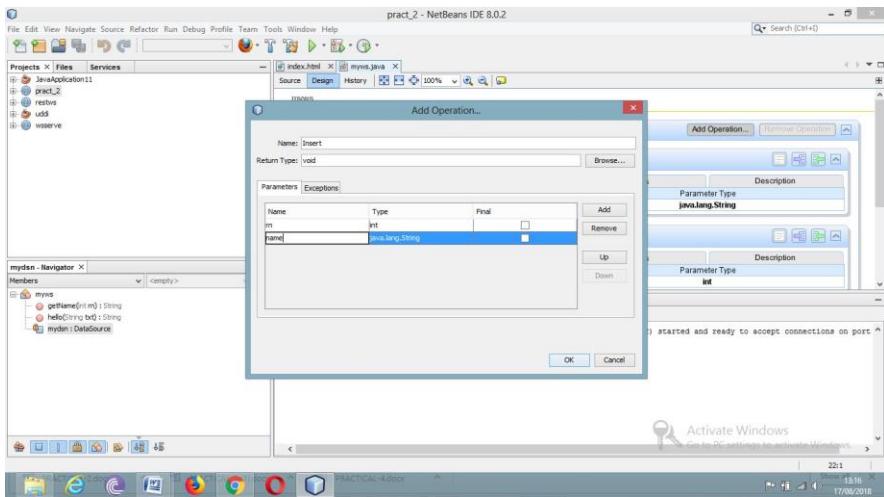
It will open



We want a two way web service which takes rn from user and return name
 So give name as getName and return type is String
 And click on add
 And define parameter
 My parameter name is rn and its type is int

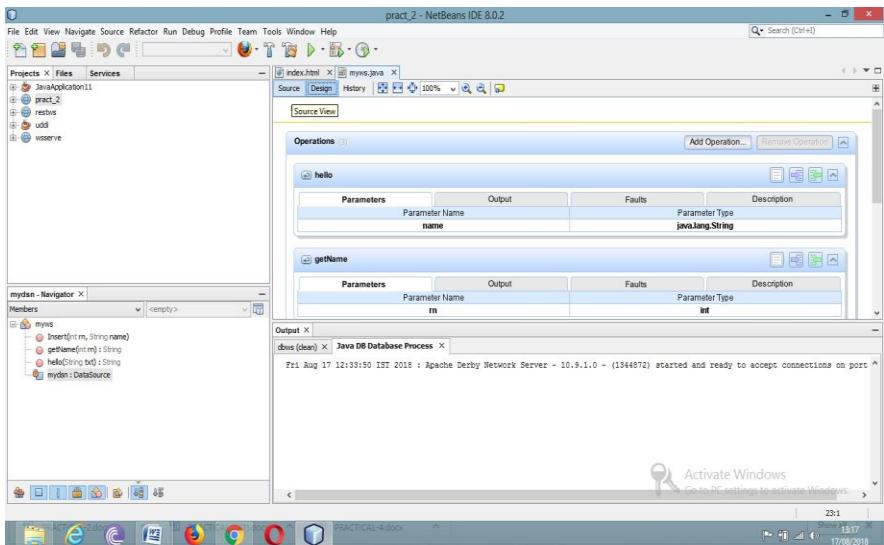


Press ok
 Very similar way define one way function which insert roll no and name in database
 Tan
 So again click on add operation give name as insert and return type as void
 Click on add
 Then give parameter as rn with type as int
 Again press add
 Give parameter as name and type as string

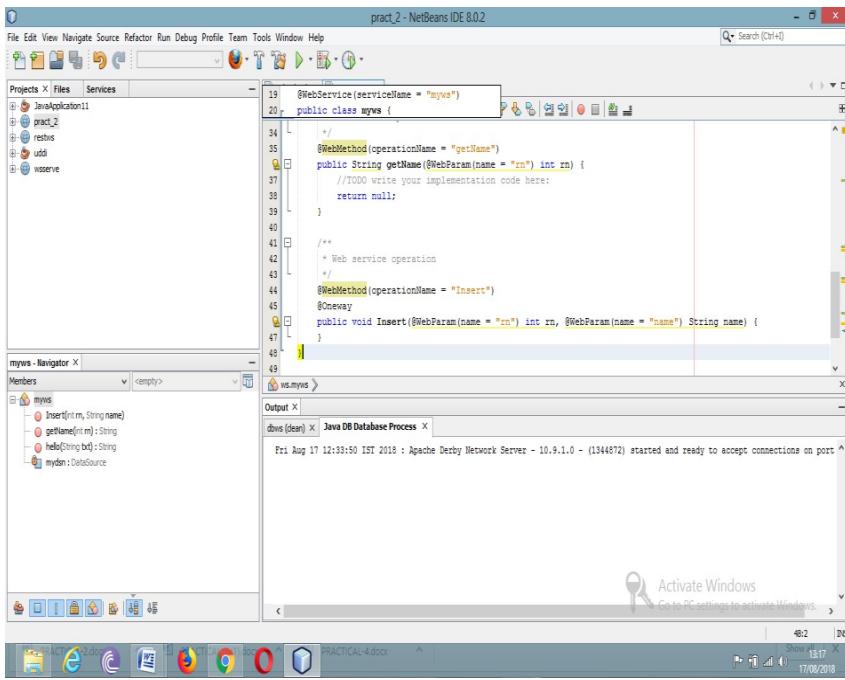


Press ok

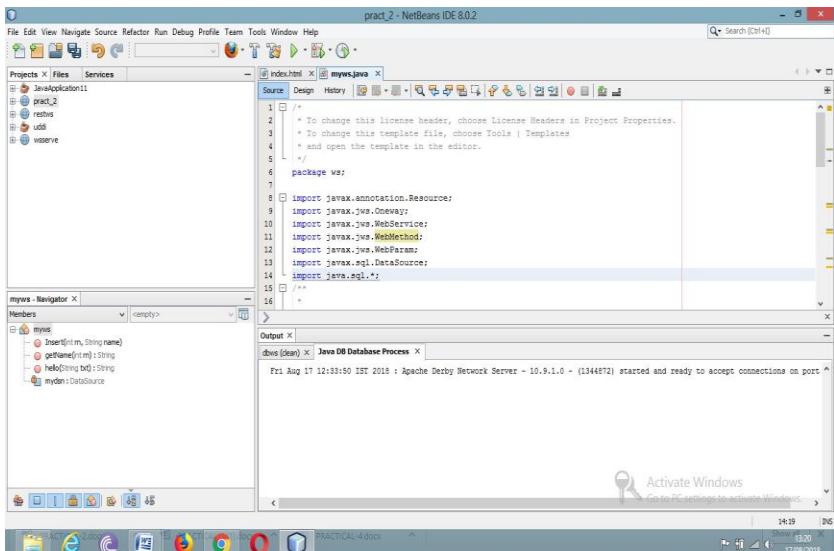
Now click on source



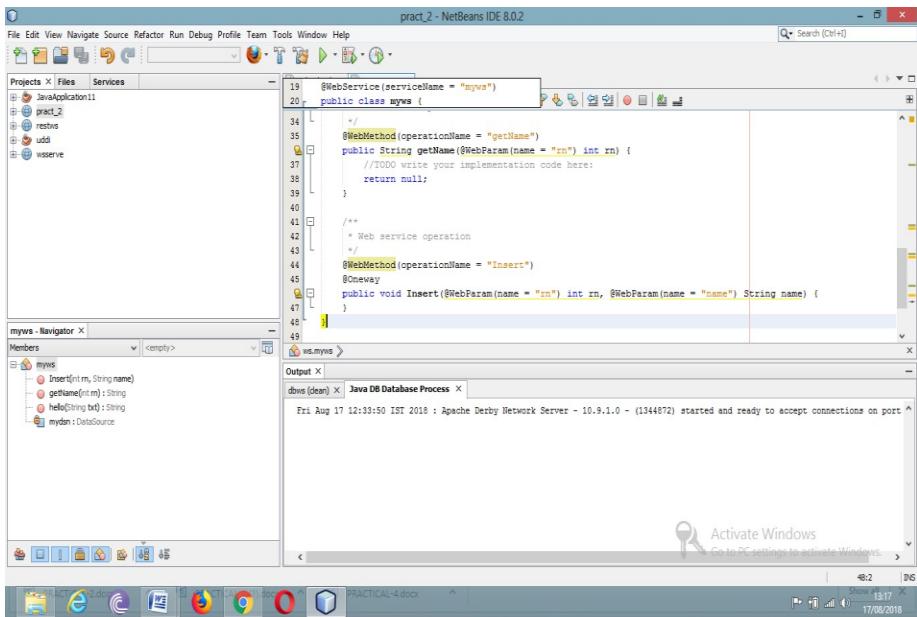
It will show you code (Functions you have created)



As we are working with database so we need to import java.sql.*;
So go on the top of same file and add import java.sql.*; statement



Now come to function getName()
Go inside it and remove the line “todo code here”



And type here

```

try{
    Connection c=mydsn.getConnection();
    PreparedStatement ps=c.prepareStatement("select * from tan where rn=?");
    ps.setInt(1, rn);
    ResultSet r=ps.executeQuery();
    if(r.next())
        return
    r.getString(2);
    else
        return "No name found";

}catch(Exception e)
{return "error";}

```

Now save it

Very similar way write the code for Insert Function (One way)

Right Click on Project name pract_2

Select deploy

Once Deployed our web service is ready

Now We will Create Client (Servlet)

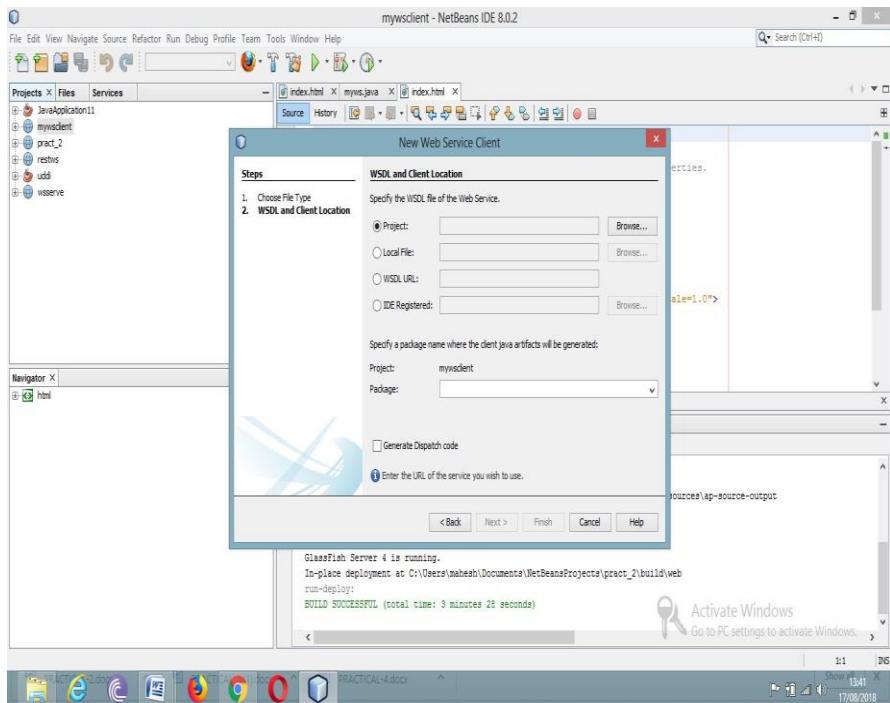
So create new Project (web java)

Select new project select new select web java

Give it name mywsclient Press next Press finish

Now right click on project name mywsclient select

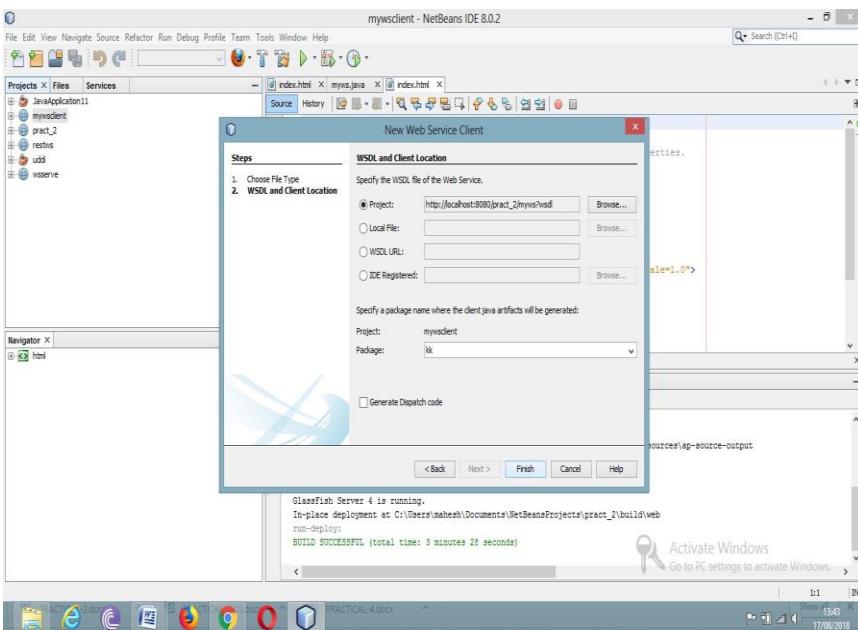
new select web service client



click on Project then select brows

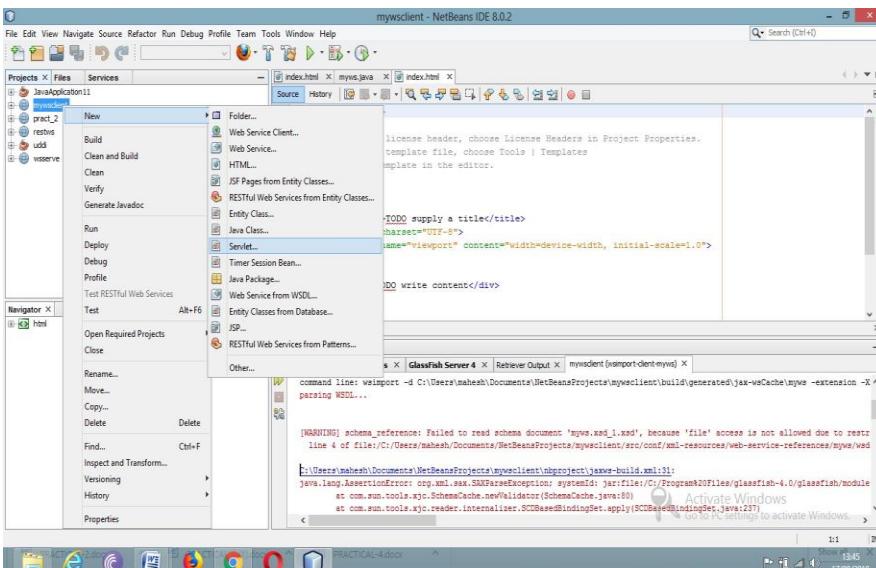
expand pract_2 select myws and press

ok Give package name kk



Press finish

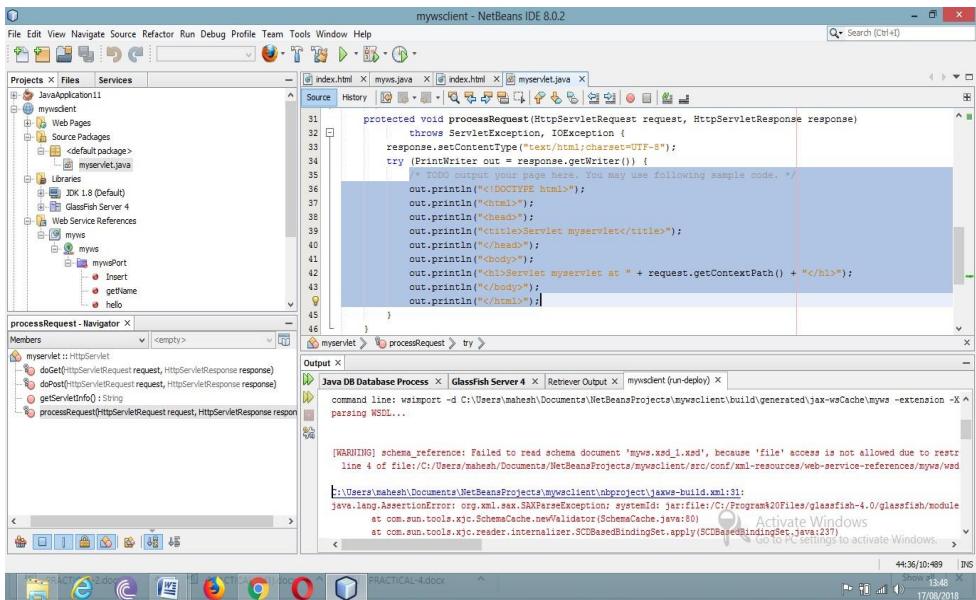
Now right click on mywsclient project select new select servlet



Give servlet name as myservlet

Press finish

Now remove lines from number 34 to 42 of Process request function of myservlet



Now the lines which we have removed i.e. from 34 to 42 ,type there

out.println(getName(1));

Now on the project window

Expand web service reference folder You will

find there getName function

Simply drag it in side myservlet.

It will create code automatically.

Now deploy mywsclient and run mywsclient by right clicking on the code of mywsclient

PRACTICAL-4

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

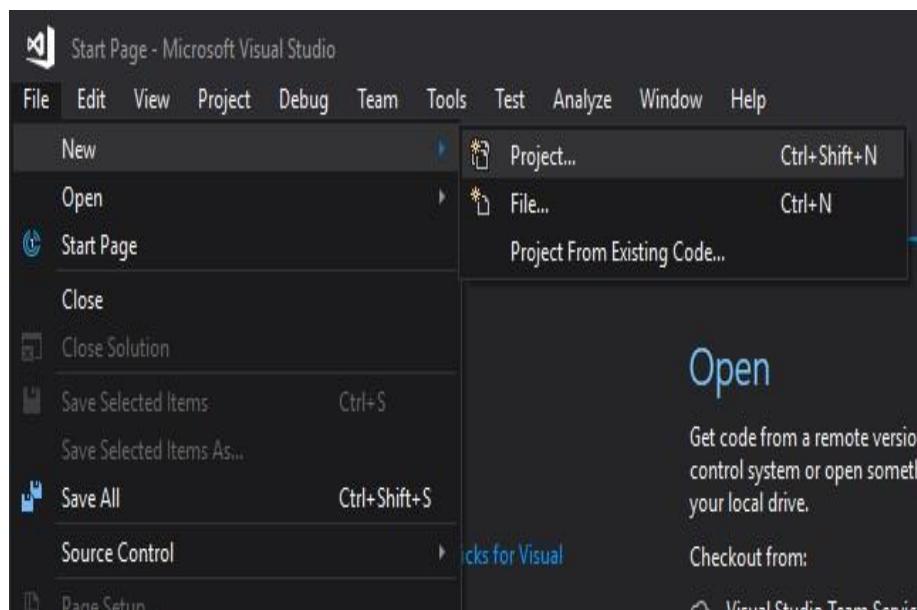
Requirement:

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

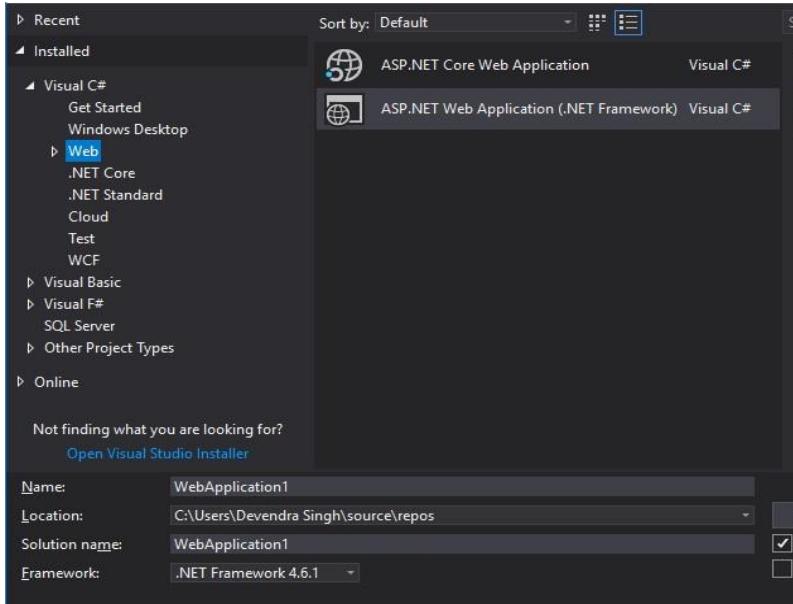
In this practical we are creating Web Service in Visual Studio ant then we will consume it in NetBeans.

1. Open Visual Studio IDE and click on File.

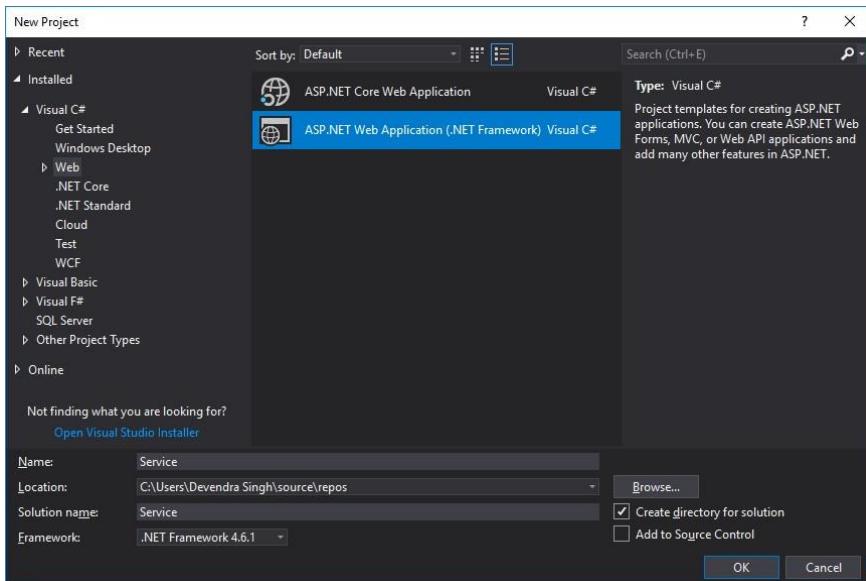
File -> New -> Project



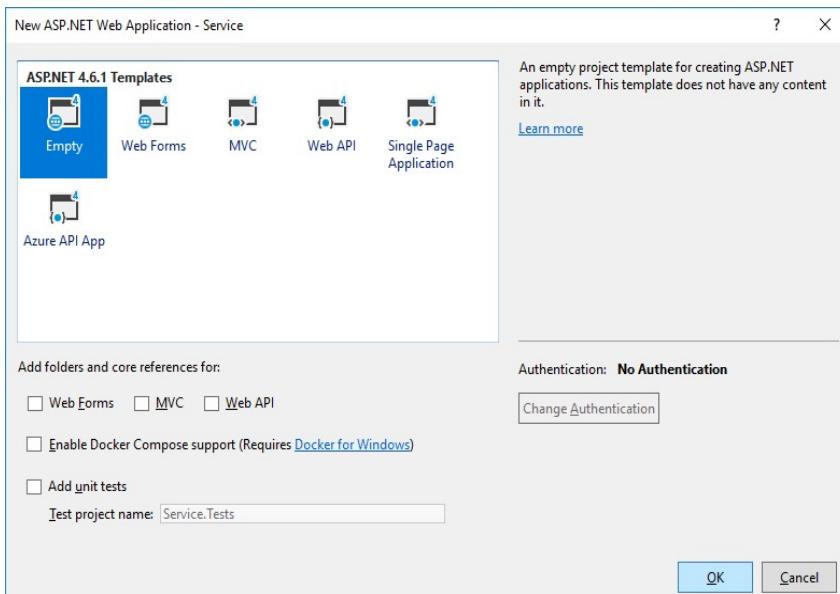
2. Click on Web.



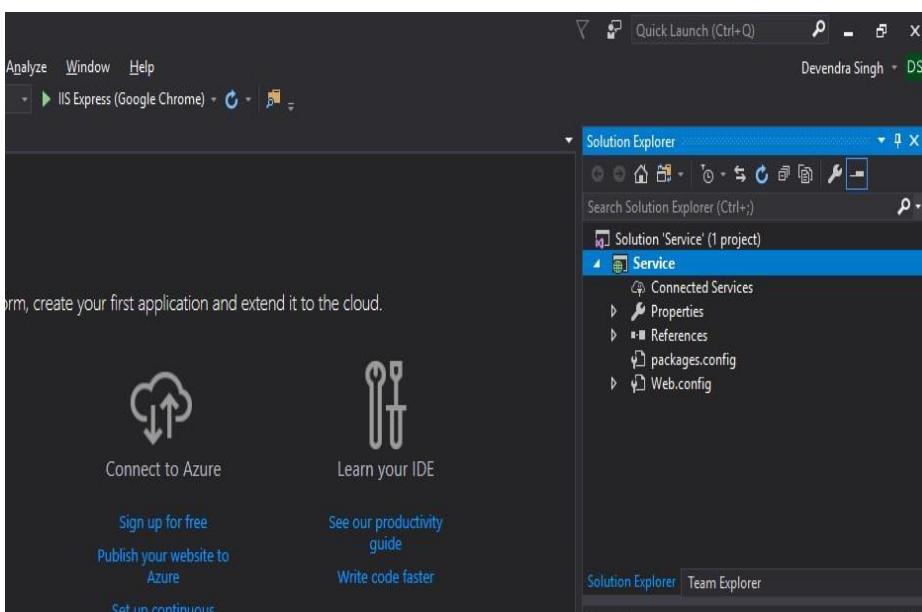
3. Select ASP.NET Web Application and give Name as Service. After that click on OK button.



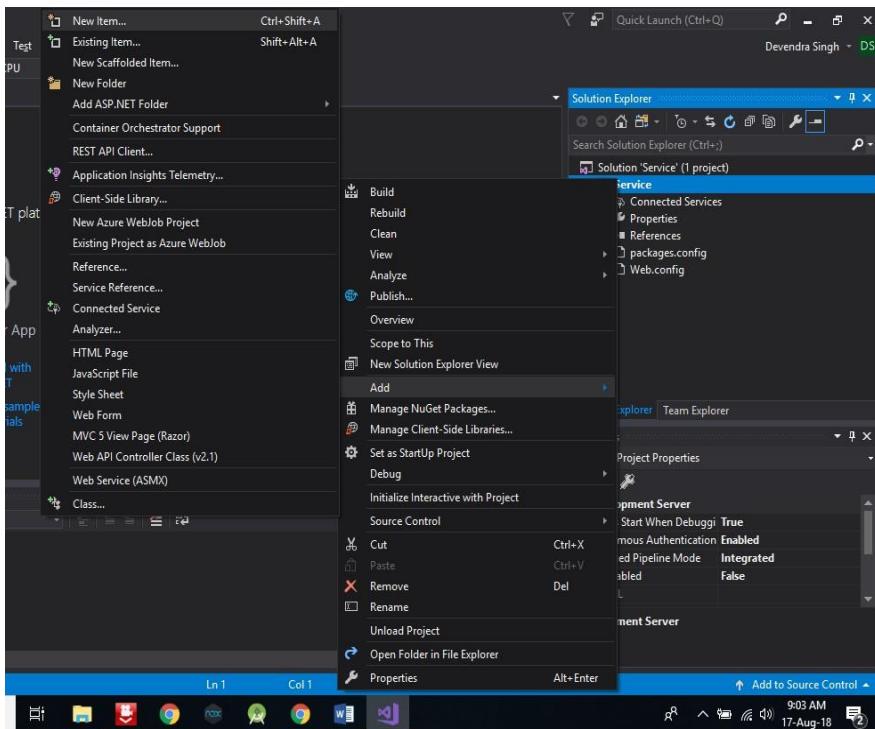
4. Select Empty and click on OK button.



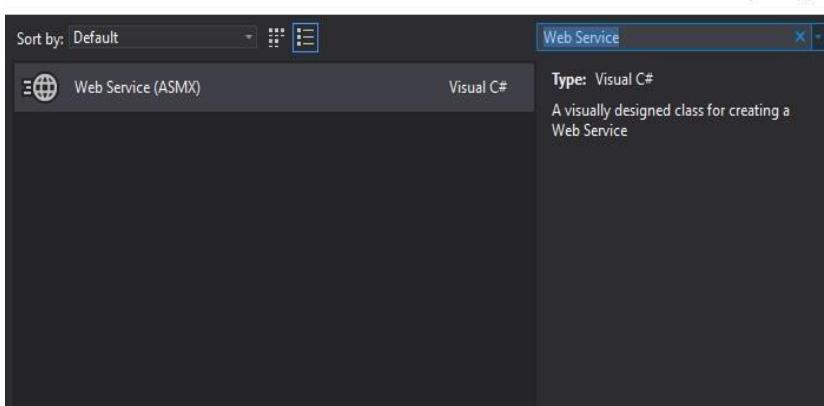
5. Now you can see, on right side in Solution Explorer Service project is created.



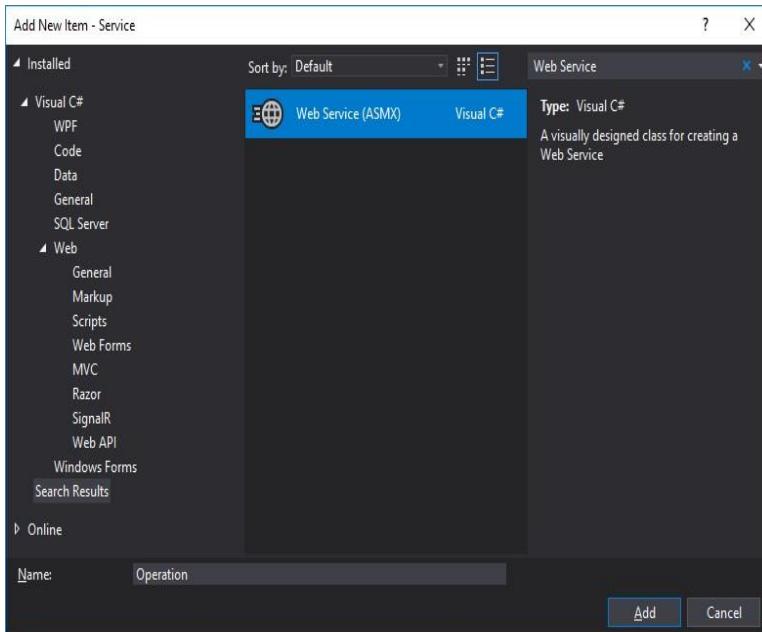
6. Right click on Service -> Add -> New Item.



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  
}
```

```
Operation.asmx.cs*  X  Service
Service
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.

```
Service - Microsoft Visual Studio
Edit View Project Build Debug Team Tools Test Analyze Window Help
Operation.asmx.cs  X  Service
Service
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.NET AJAX, uncomment the following line.
```

11. A window will open in browser and that is our web service.

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

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

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

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

Each XML Web service needs a unique namespace in order for client applications to distinguish it from other Web services. You 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 domain name as the namespace, or you can use a URI that does not point to actual resources on the Web. (XML Web service namespaces are URIs.)

For XML Web services created using ASP.NET, the default namespace can be changed using the `WebService` class's `Namespace` property. Below is a code example that sets the namespace to "http://microsoft.com/webservices/"

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.

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

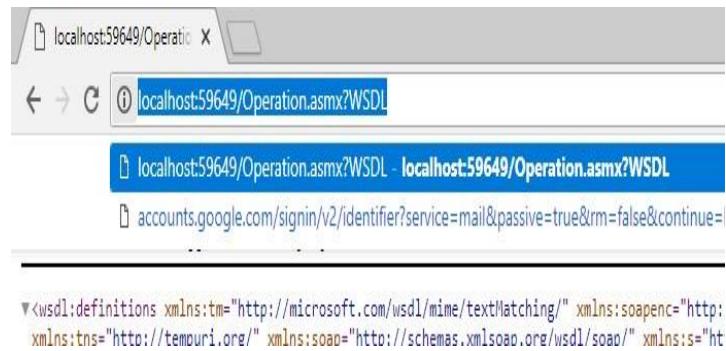
- [Add](#)
- [Multi](#)

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

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

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

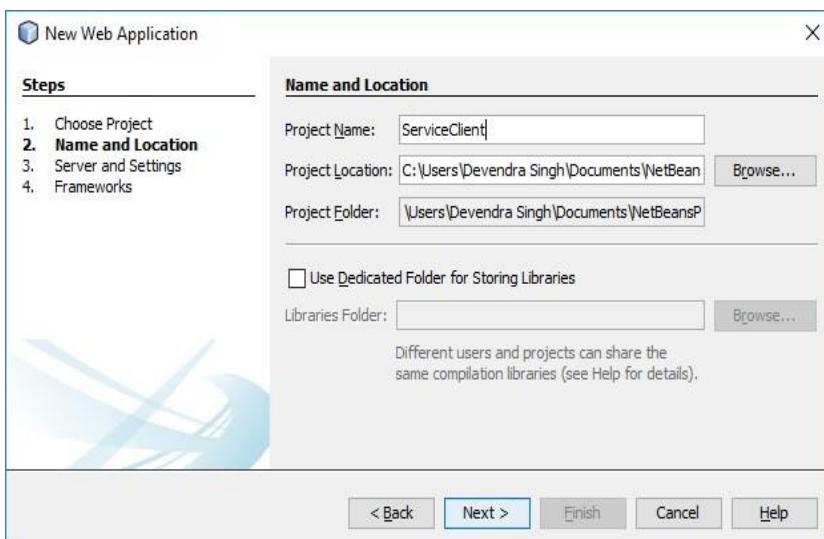
14. Now a new window will open. Select the link and copy it. We will use this link in NetBeans to consume these services.



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.

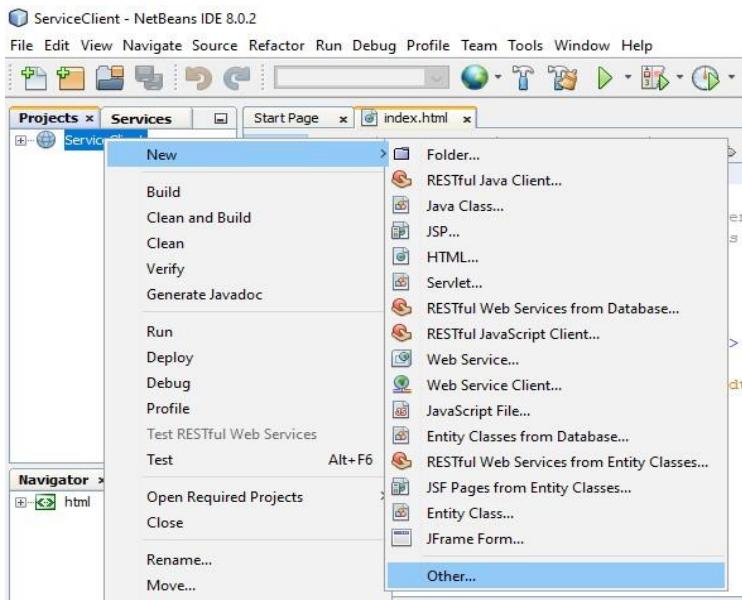
16. Now open NetBeans.

17. Create a Web Application with name ServiceClient. Next -> Finish.

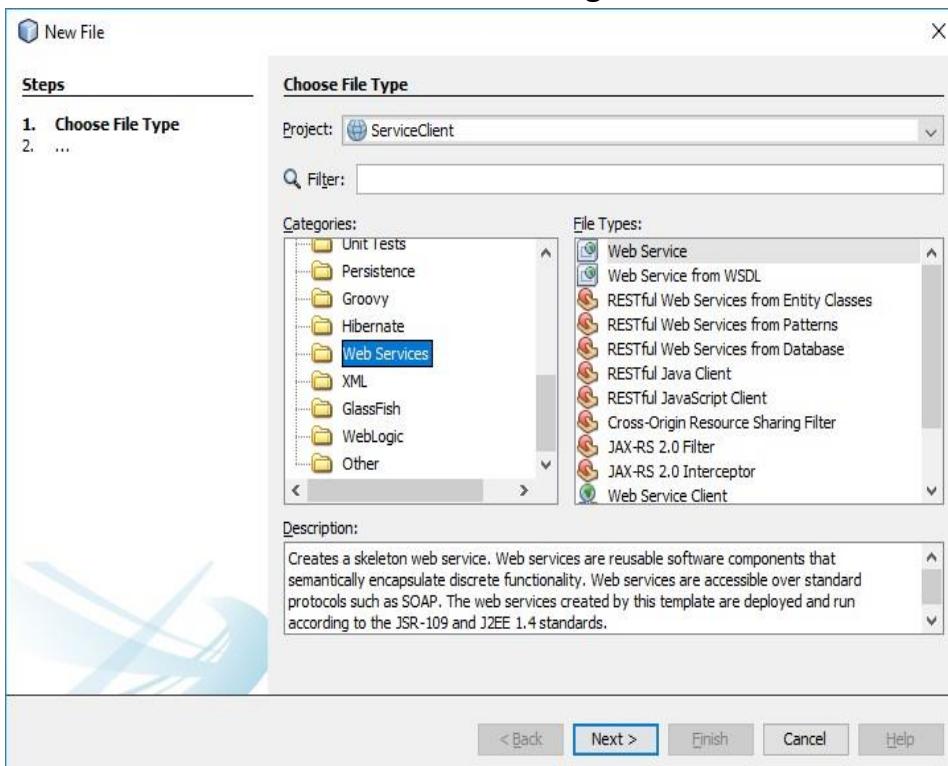


18. Now create a Web Service Client.

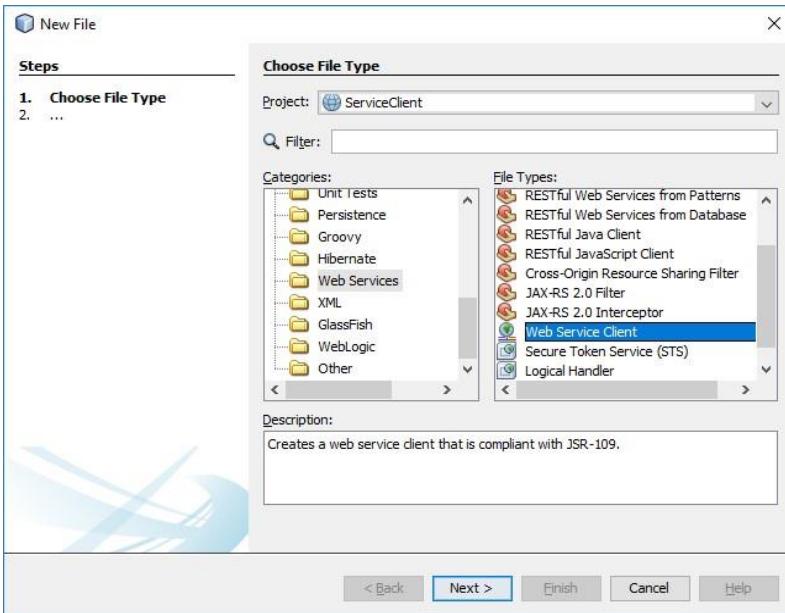
Right click on ServiceClient -> New -> Other.



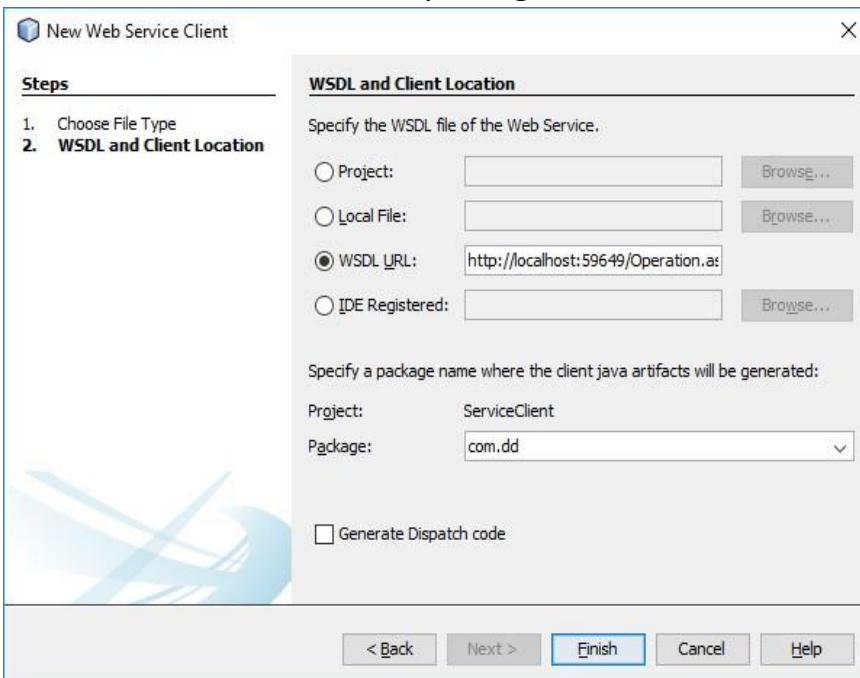
19. Now select Web Services in Categories section.



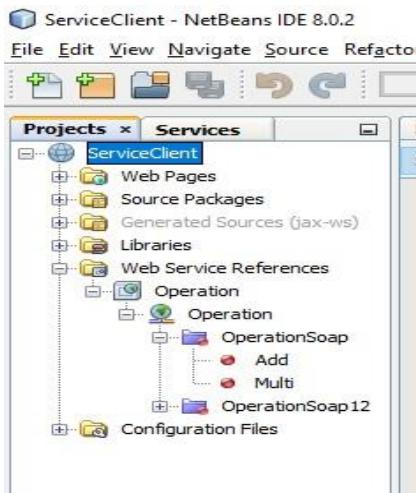
20. Select Web Service Client in File Types and Click on Next button.



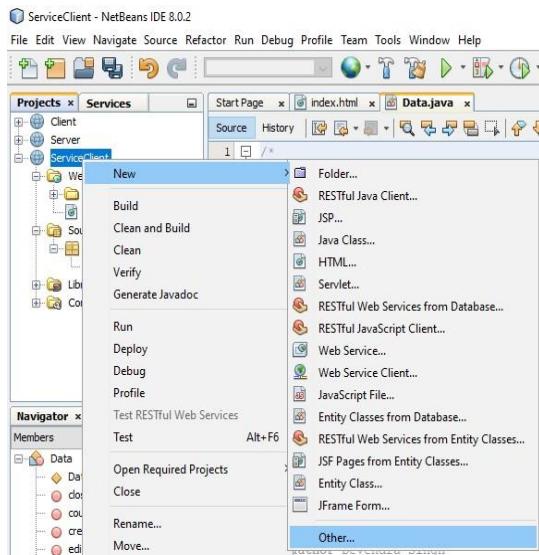
21. Select WSDL URL and paste the link that you have copied from browser on run of Visual Studio enter package name com.dd. After that click on Finish button.



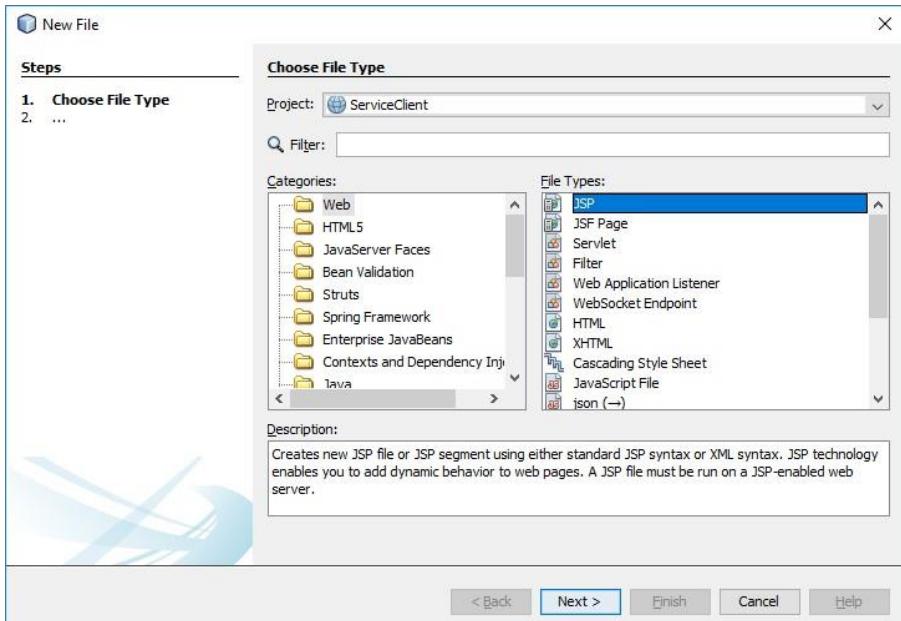
22. As you can see, we got both the service methods i.e. Add & Multi. But in this practical I'm going to use only one service method. You can do for both also.



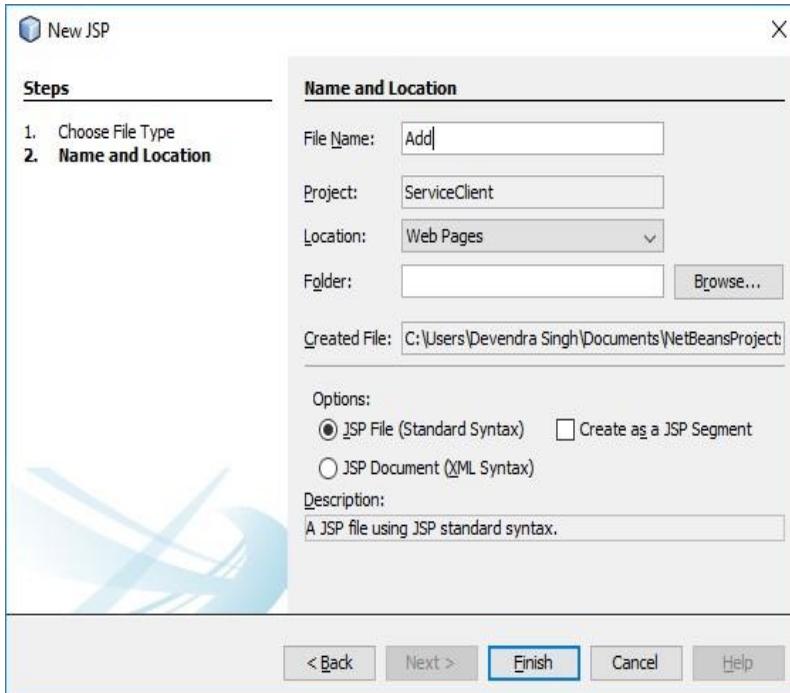
23. Now create a JSP page. Right click on ServiceClient -> New -> Other



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



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



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

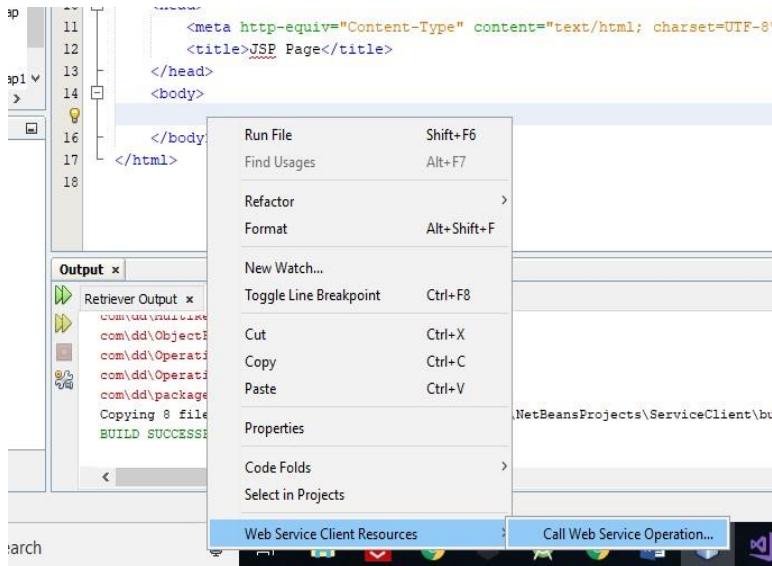
riceClient - NetBeans IDE 8.0.2

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

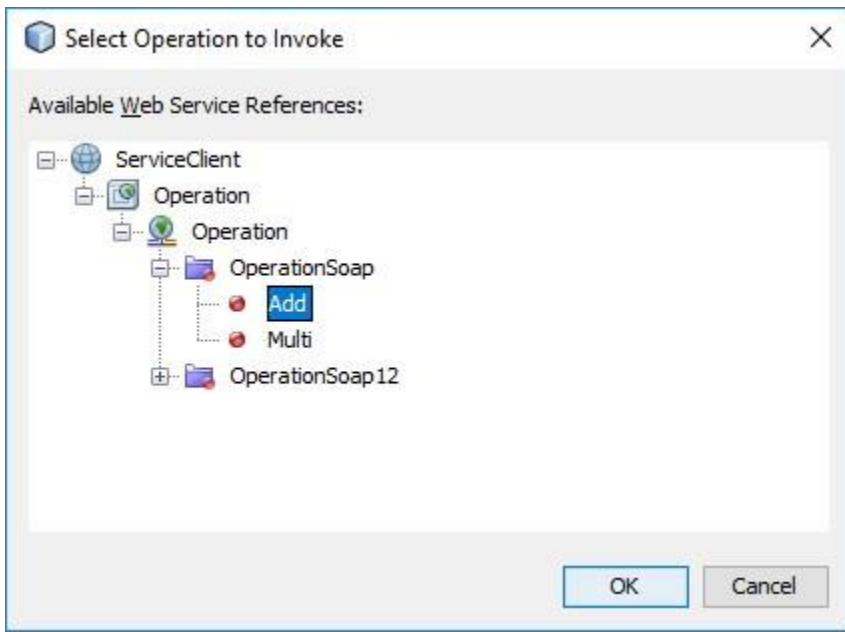
The screenshot shows the NetBeans IDE interface. The Services tab is selected, displaying a tree view of project components: ServiceClient, Web Pages (WEB-INF/Add.jsp, index.html), Source Packages, Generated Sources (jax-ws), Libraries, and Web Service References (Operation, OperationSoap, OperationSoap1). The code editor shows the JSP file Add.jsp with the following content:

```
<%--  
1 Document : Add  
2 Created on : Aug 17, 2018, 10:58:06 AM  
3 Author : Devendra Singh  
4 --%>  
  
<%@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>  
        <h1>Hello World!</h1>  
    </body>  
</html>
```

27. Right click between body tag and select Call Web Service Operation.



28. Expand and select Add. After select, click on OK button.



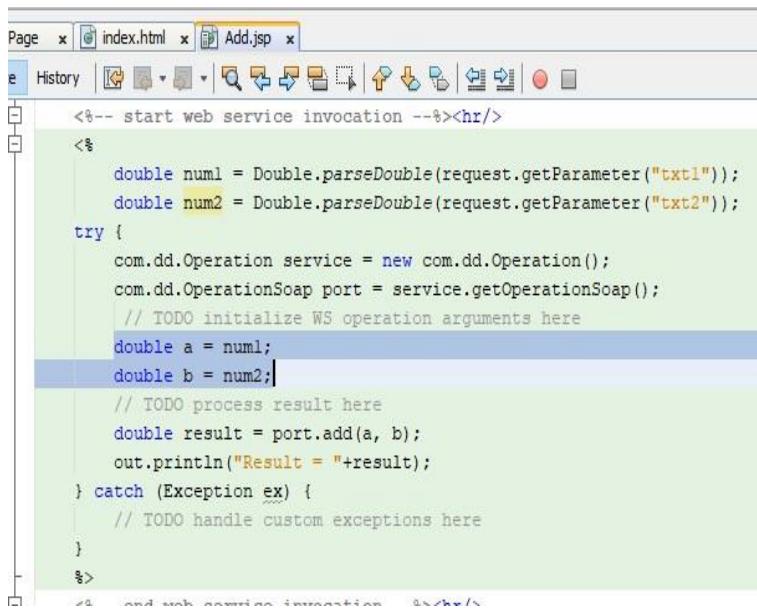
29. Now add the following code outside of try block.

```
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;
    // TODO process result here
    double result = port.add(a, b);
    out.println("Result = "+result);
} catch (Exception ex) {
    // TODO handle custom exceptions here
}
%>
<%-- end web service invocation --%><hr/>
```

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



The screenshot shows a Java IDE interface with the following details:

- Page**: The current tab is "Add.jsp".
- History**: A dropdown menu.
- Toolbar**: Includes icons for file operations like Open, Save, Print, and Find.
- Code Area**: Contains JSP code:

```
<%-- 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
}
%>
```
- Status Bar**: Shows "End web service invocation" and "ctrl+s".

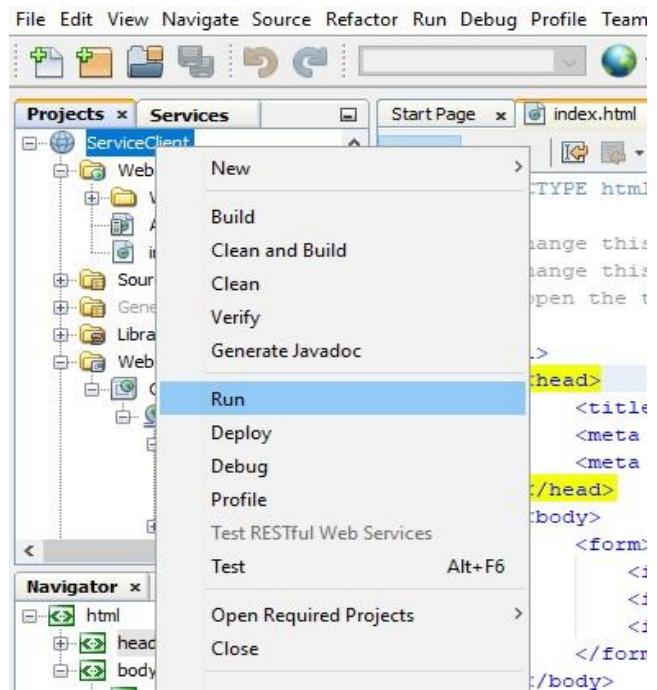
- 31.** Now open index.html file of ServiceClient project and replace the contents of body tag with following code. After that press Ctrl+S to save it.

```
<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>
```

Screenshot of the Eclipse IDE interface showing the code editor for the 'index.html' file. The code defines a simple form for adding two numbers:

```
<!DOCTYPE html>
<!--
To change this license header, choose License Headers in Project Properties.
To change this template file, choose Tools | Templates
and open the template in the editor.
-->
<html>
    <head>
        <title>TODO supply a title</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>
            <input type="text" name="txt2" placeholder="Enter Second Number"><br><br>
            <input type="submit" formaction="Add.jsp" value="Add Numbers">
        </form>
    </body>
</html>
```

32. Now run the ServerClient web application.



33. A window will open in browser as below.

A screenshot of a Microsoft Internet Explorer browser window. The address bar shows the URL <http://localhost:8080/ServiceClient>. The title bar says "TODO supply a title". The menu bar includes File, Edit, View, Favorites, Tools, and Help. Below the menu is a form with three input fields: "Enter First Number", "Enter Second Number", and a button labeled "Add Numbers".

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

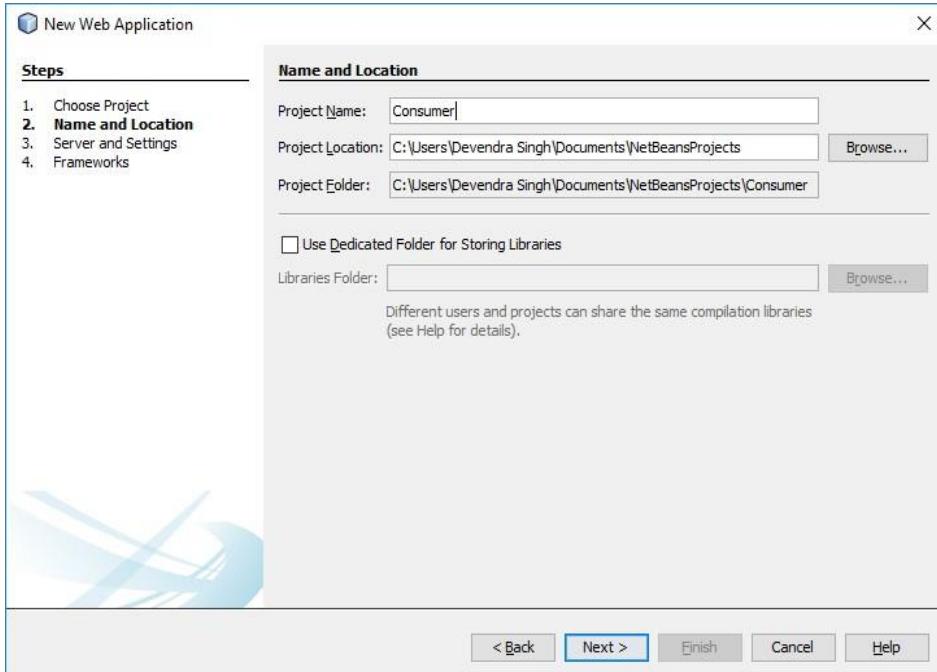
You will get result on a new page.

A screenshot of a Microsoft Internet Explorer browser window. The address bar shows the URL <http://localhost:8080/ServiceClient/>. The title bar says "TODO supply a title". The menu bar includes File, Edit, View, Favorites, Tools, and Help. Below the menu is a form with two input fields containing the number "19" each, and a button labeled "Add Numbers".

A screenshot of a Microsoft Internet Explorer browser window. The address bar shows the URL <http://localhost:8080/ServiceClient/Add>. The title bar says "JSP Page". The menu bar includes File, Edit, View, Favorites, Tools, and Help. Below the menu is a single line of text: "Result = 38.0".

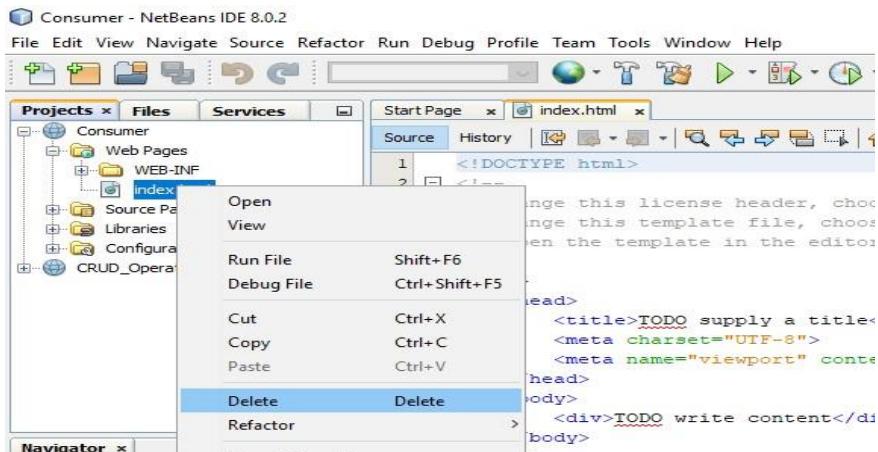
Practical -05

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

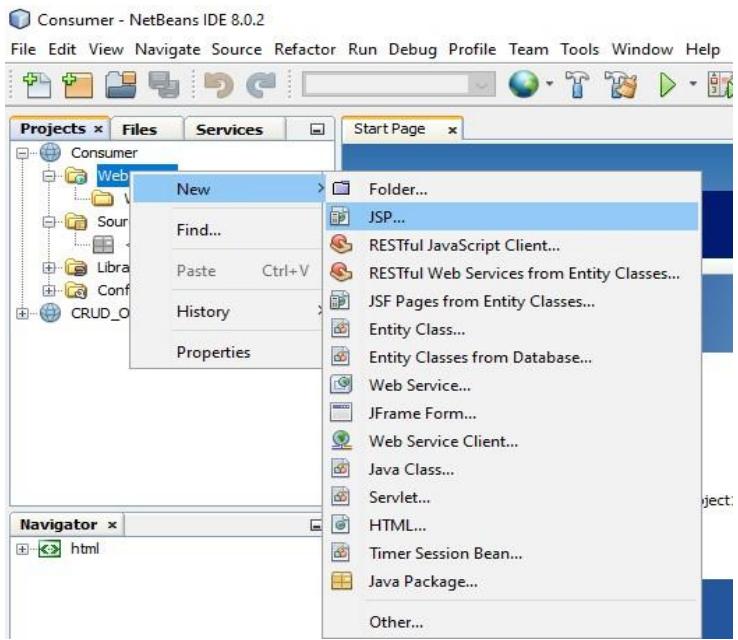


2. And create it. Next -> Finish.

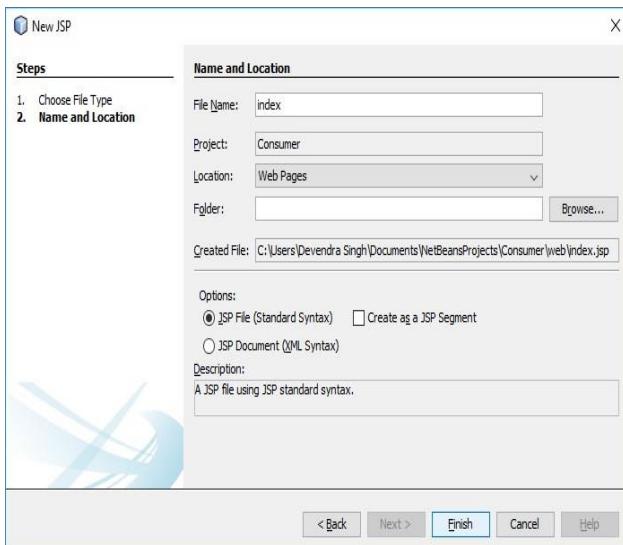
3. Now right click on index.html and delete it.



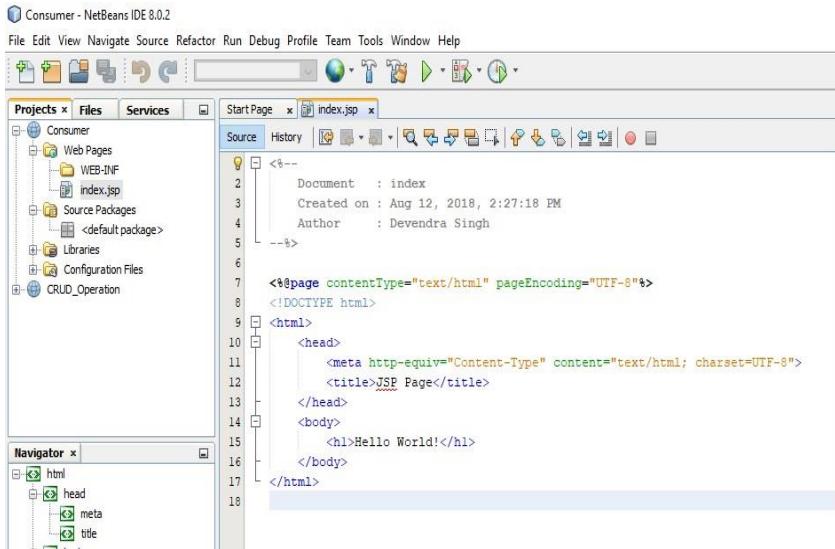
4. Now right click on Web Pages and select JSP to add a JSP page.



5. Give index name to it and then click on Finish.



6. Now index.jsp page will open like below.



- 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.

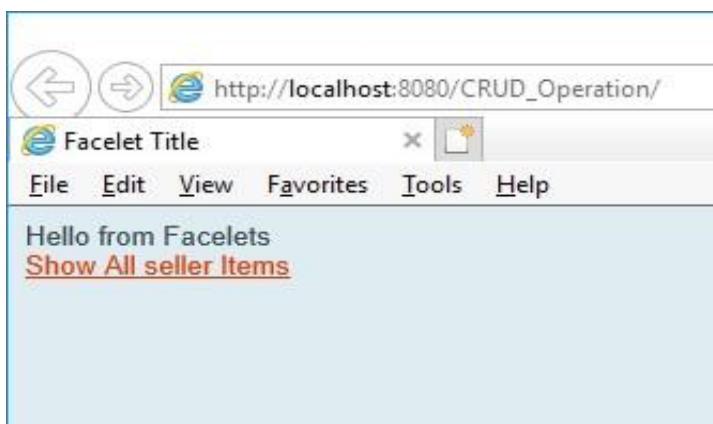
```

StartPage x index.html x seller.java x index.jsp x
Source History | 
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/

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.

```

 34     }
 35
 36     @POST
 37     @Override
 38     @Consumes({"application/xml", "application/json"})
 39     public void create(seller entity) {
 40         super.create(entity);
 41     }
 42
 43     @PUT
 44     @Path("{id}")
 45     @Consumes({"application/xml", "application/json"})
 46     public void edit(@PathParam("id") Long id, seller entity) {
 47         super.edit(entity);
 48     }

```

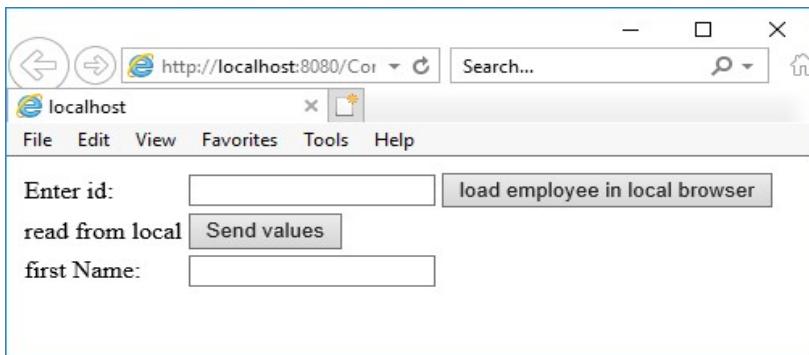
11. Now delete the selected part. Because it will return data in XML format to the consumer. But we have written javascript in index.jsp for JSON data format only. After that deploy the CRUD_Operation web application; so that it will update the changes.

```

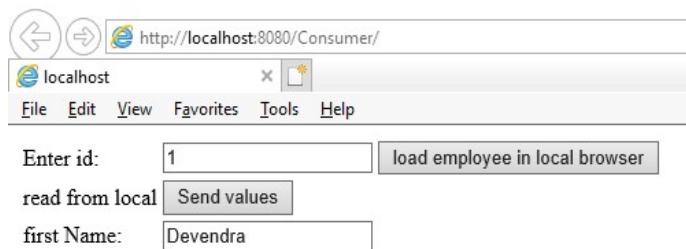
 34     }
 35
 36     @POST
 37     @Override
 38     @Consumes({"application/xml", "application/json"})
 39     public void create(seller entity) {
 40         super.create(entity);
 41     }
 42
 43     @PUT
 44     @Path("{id}")
 45     @Consumes("application/json")
 46     public void edit(@PathParam("id") Long id, seller entity) {
 47         super.edit(entity);
 48     }
 49
 50     @DELETE
 51     @Path("{id}")
 52     public void remove(@PathParam("id") Long id) {
 53         super.remove(super.find(id));
 54     }
 55
 56     @GET
 57     @Path("{id}")
 58     @Produces("application/xml", "application/json")
 59     public seller find(@PathParam("id") Long id) {
 60         return super.find(id);
 61     }
 62
 63     @GET
 64     @Override

```

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



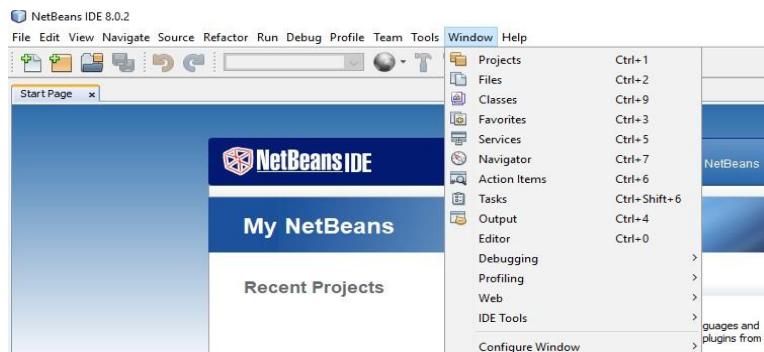
13. Now if you will enter an id into Enter id textbox which you have created in database (id from data in last step of practical 7) and then click on load employee in local browser button. It will get the detail of particular entered id and will store it into local storage of browser. Now click on Send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



PRACTICAL-6

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

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



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

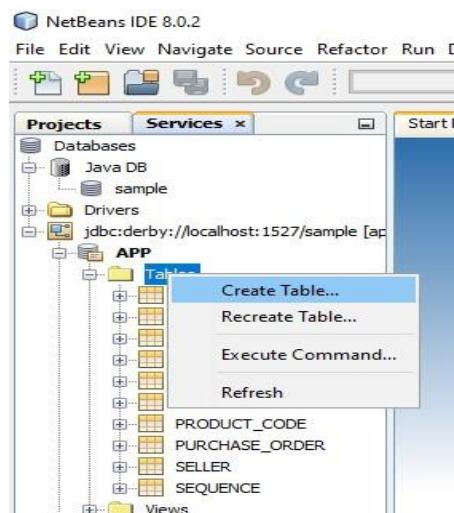


3. Now expand Java DB and right click on sample and then click on connect to connect the sample database with server.

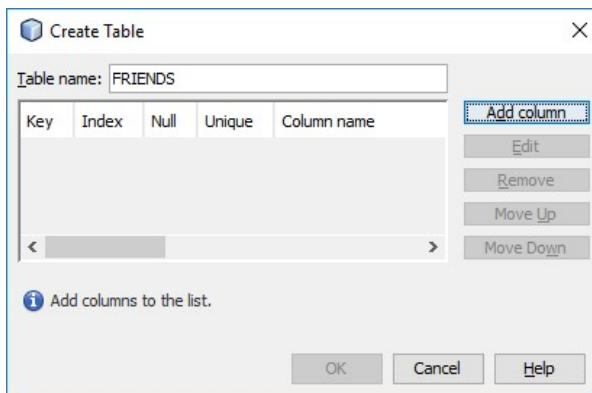


4. Now we are going to create a table in default database sample.

Right click on Table -> Create Table

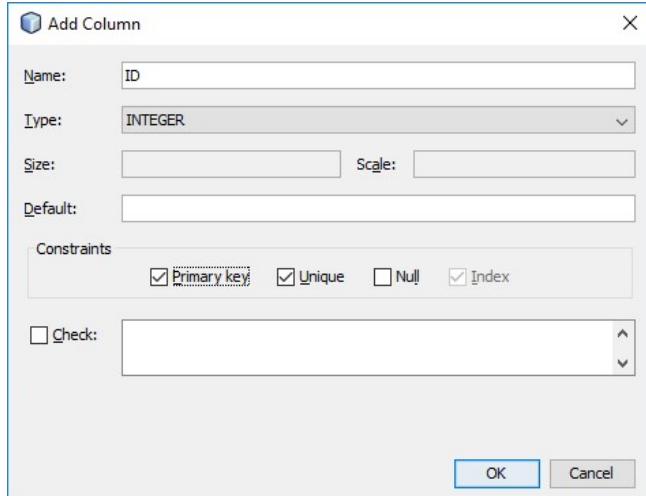


5. Give table name as FRIENDS.



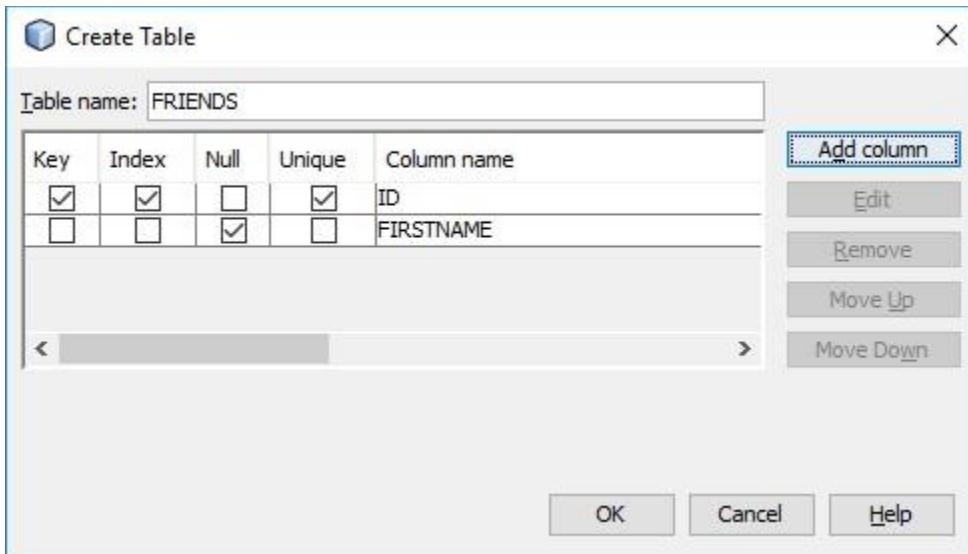
6. click on Add column button to add columns in table.

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

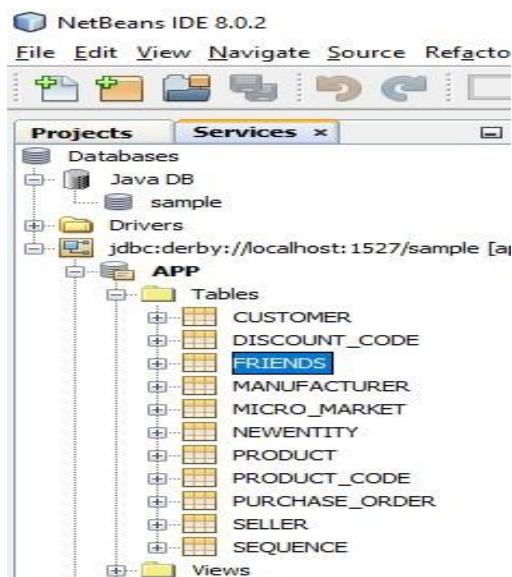


7. Now add second column with following detail. But don't select primary and click on OK button.

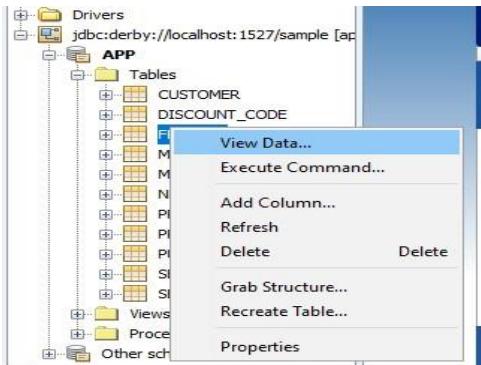
click on OK button.



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



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



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

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

#	ID	FIRSTNAME
1		ANAND

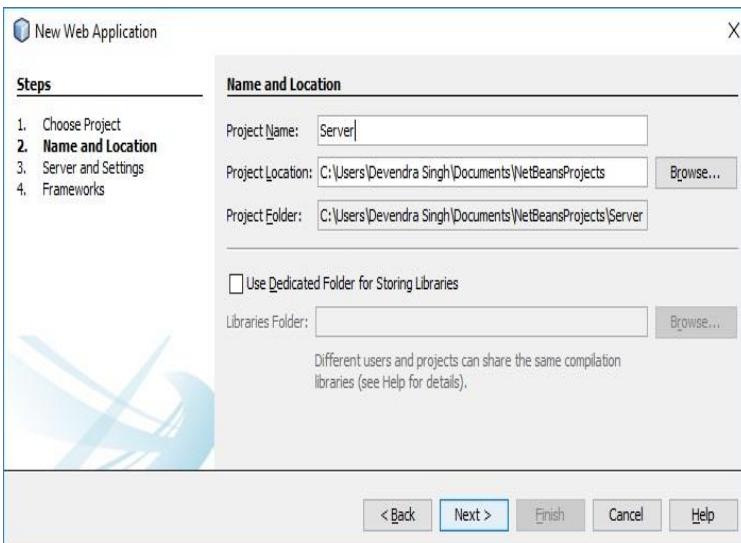
Buttons at the bottom: Show SQL, Add Row, Remove, OK, Cancel.

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

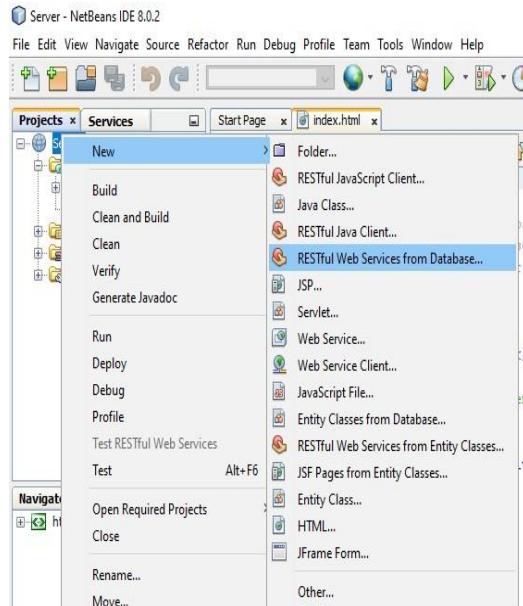
select * from APP.FRIENDS x

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

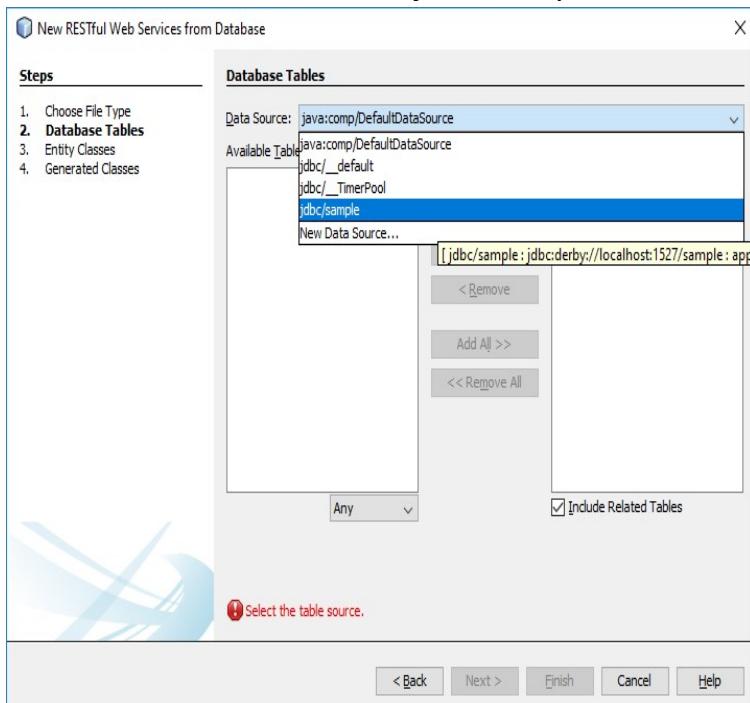
- 13.** create a web application with name Server. After that click on Next and then Finish button.



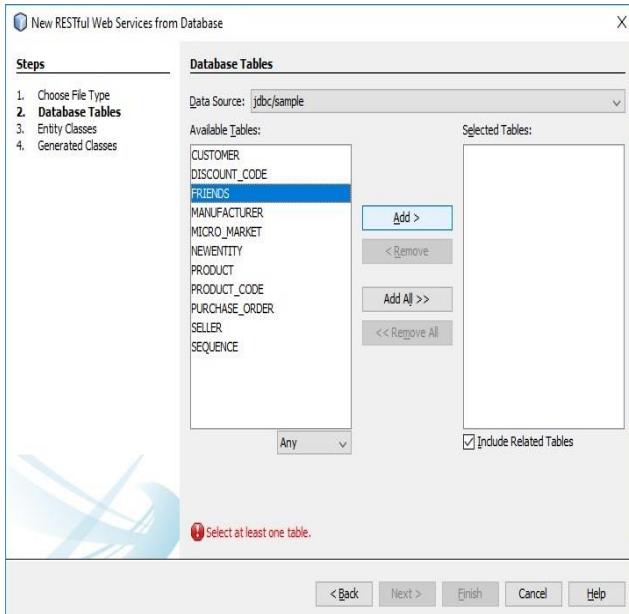
- 14.** Now create a RESTful Web Service from Database by right click on project name.



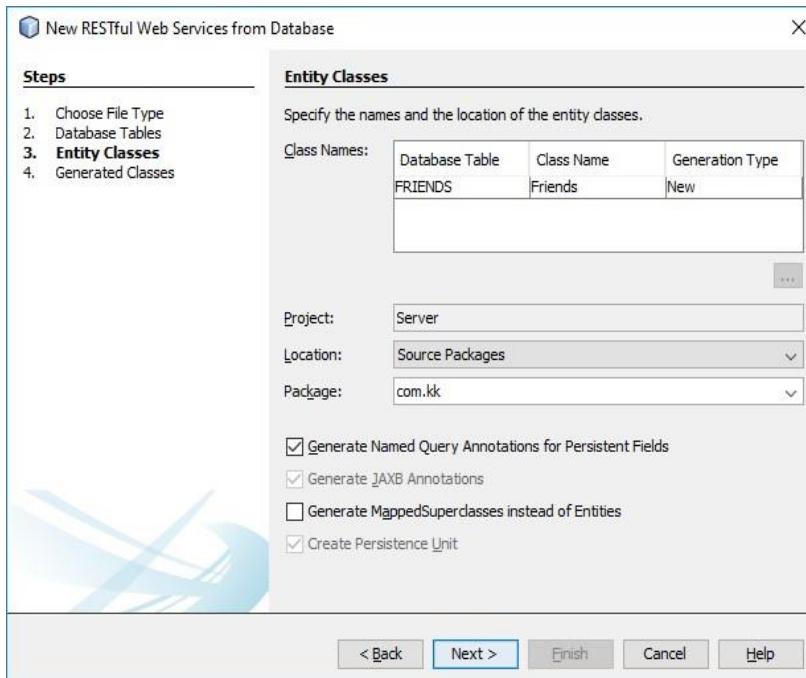
15. Choose Data Source jdbc/sample.



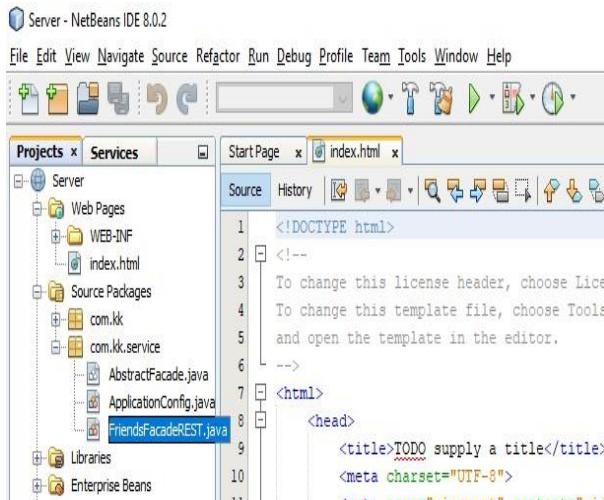
16. Now select FRIENDS and click on Add button. After that click on Next button.



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



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



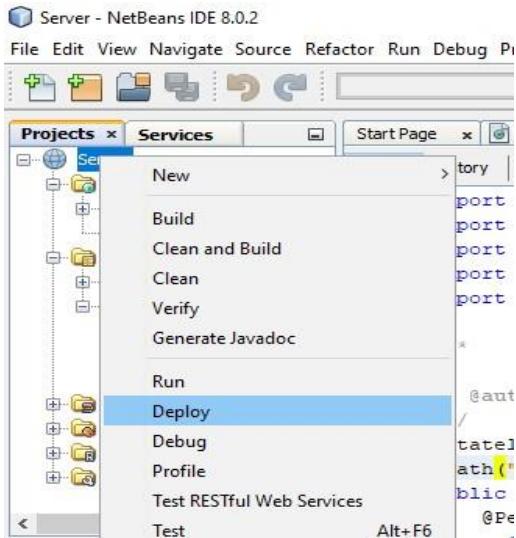
- 19.** Now remove the selected part from every method in this file. So that it will communicate only in JSON format. You can also use methods to convert it. But this is easiest method.

```

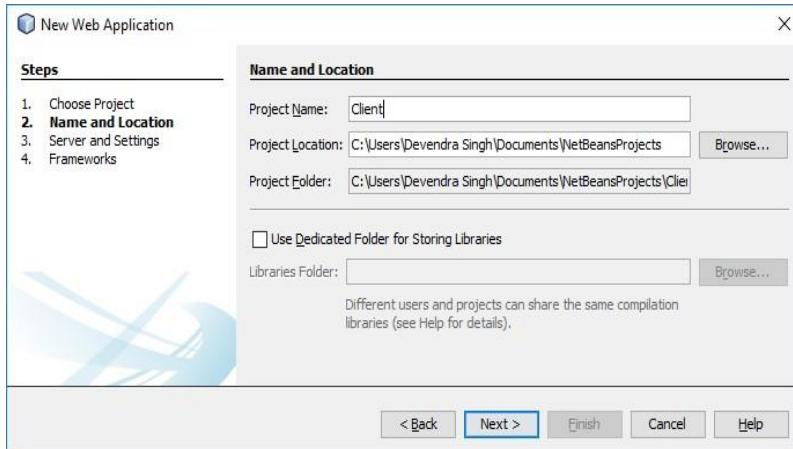
25 */
26 @Stateless
27 @Path("com.kk.friends")
28 public class FriendsFacadeREST extends AbstractFacade<Friends> {
29   @PersistenceContext(unitName = "ServerPU")
30   private EntityManager em;
31
32   public FriendsFacadeREST() {
33     super(Friends.class);
34   }
35
36   @POST
37   @Override
38  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

```

- 20.** After that right click on project name and Deploy it.



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



- 22.** 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>
```

```

<html>
  <head>
    <title>TODO supply a title</title>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <style>
      table {
        font-family: arial, sans-serif;
        border-collapse: collapse;
      }
      td, th {
        border: 1px solid #000000;
        text-align: center;
        padding: 8px;
      }
    </style>
    <script>
      var request = new XMLHttpRequest();
      request.open('GET', 'http://localhost:8080/Server/webservices/com.nn.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(i);
          var cell1 = row.insertCell(0);
          var cell2 = row.insertCell(1);
          cell1.innerHTML = data[i].id;
          cell2.innerHTML = data[i].firstname;
        }
      }
    </script>
  </head>
  <body>
    <table id="myTable">
      <tr>
        <th>ID</th>
        <th>NAME</th>
      </tr>
    </table>
  </body>
</html>

```

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>

```

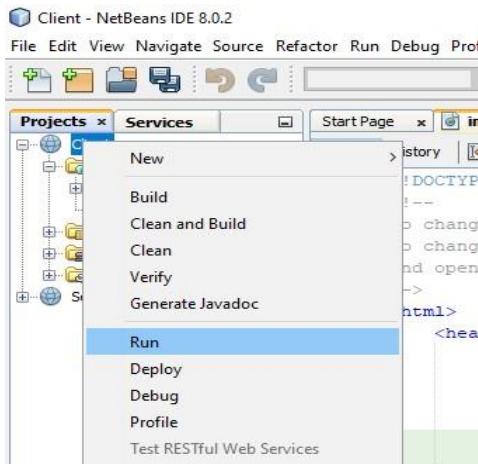
```

<html>
  <head>
    <title>TODO supply a title</title>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <style>
      table {
        font-family: arial, sans-serif;
        border-collapse: collapse;
      }
      td, th {
        border: 1px solid #000000;
        text-align: center;
        padding: 8px;
      }
    </style>
    <script>
      var request = new XMLHttpRequest();
      request.open('GET', 'http://localhost:8080/Server/webservices/com.nn.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(i);
          var cell1 = row.insertCell(0);
          var cell2 = row.insertCell(1);
          cell1.innerHTML = data[i].id;
          cell2.innerHTML = data[i].firstname;
        }
      }
    </script>
  </head>
  <body>
    <table id="myTable">
      <tr>
        <th>ID</th>
        <th>NAME</th>
      </tr>
    </table>
  </body>
</html>

```

25. Now run the Client Web Application.



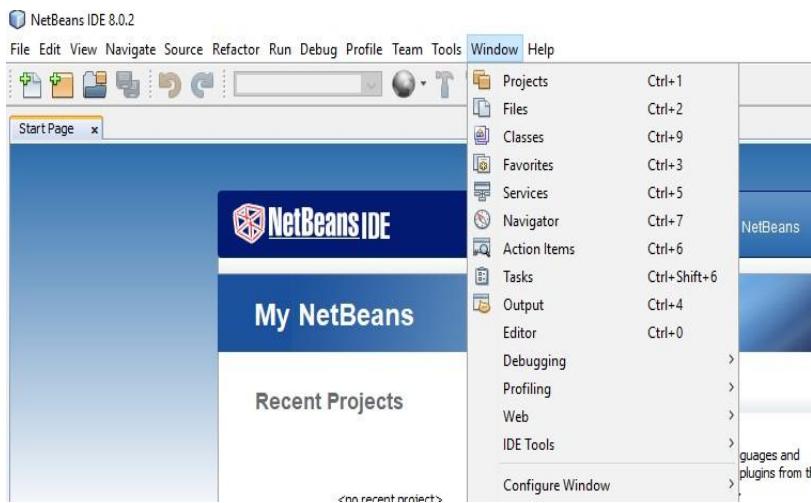
26. A window will open in browser which represents a data in tabular format. These data are the records entered in FRIEND table.

A screenshot of a web browser window. The address bar shows "http://localhost:8080/Client/". The page title is "TODO supply a title". The browser menu bar includes File, Edit, View, Favorites, Tools, and Help. Below the menu is a table with the following data:

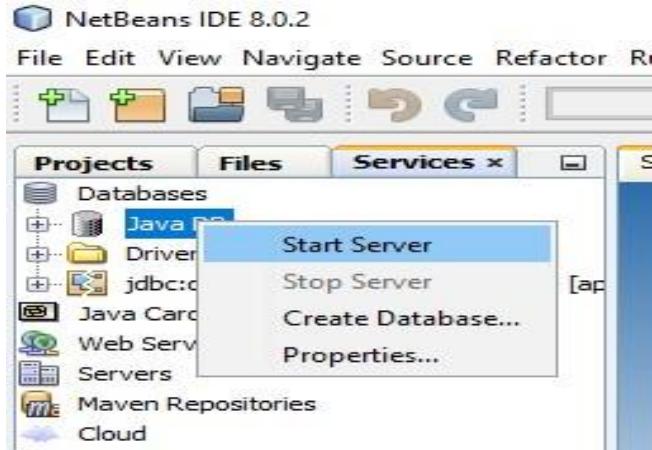
ID	NAME
1	ANAND
2	JULHAS
3	NIKHIL
4	GAGAN
5	RAVI
6	DHARMENDRA
7	ADARSH

Practical-07

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



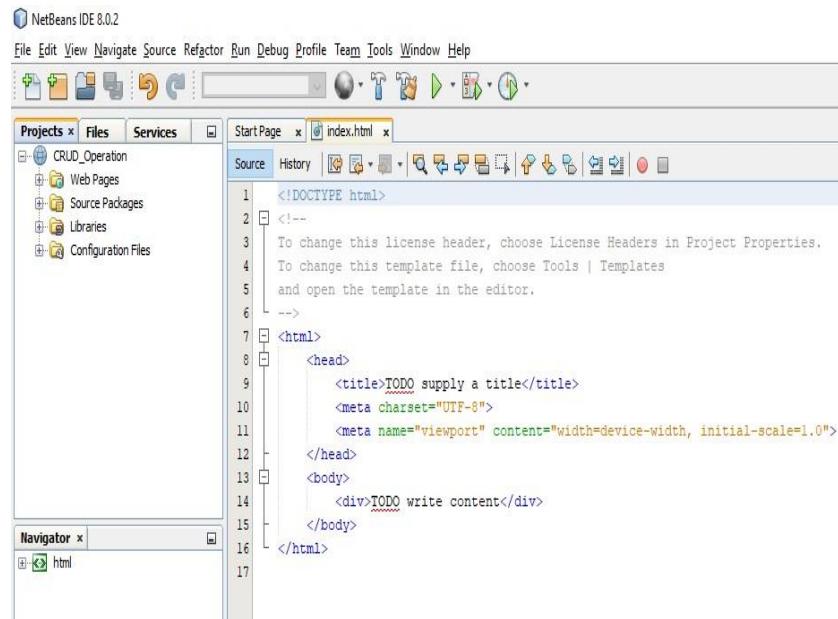
2. Right click on Java DB and then click on Start Server to start the server .



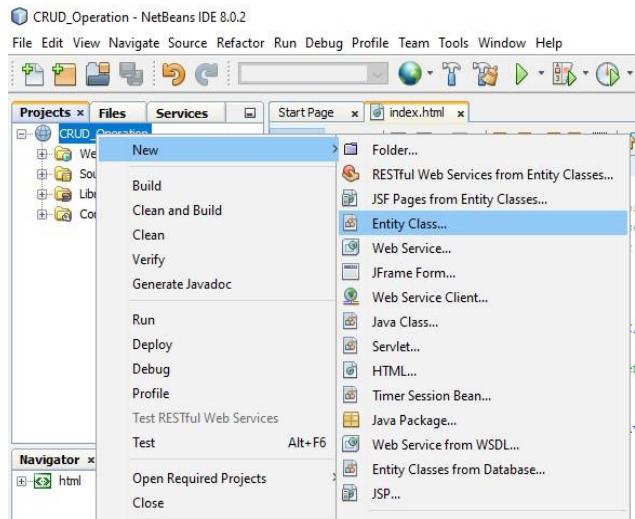
3. Now expand Java DB and right click on sample and then click on connect to connect the sample database with server.



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



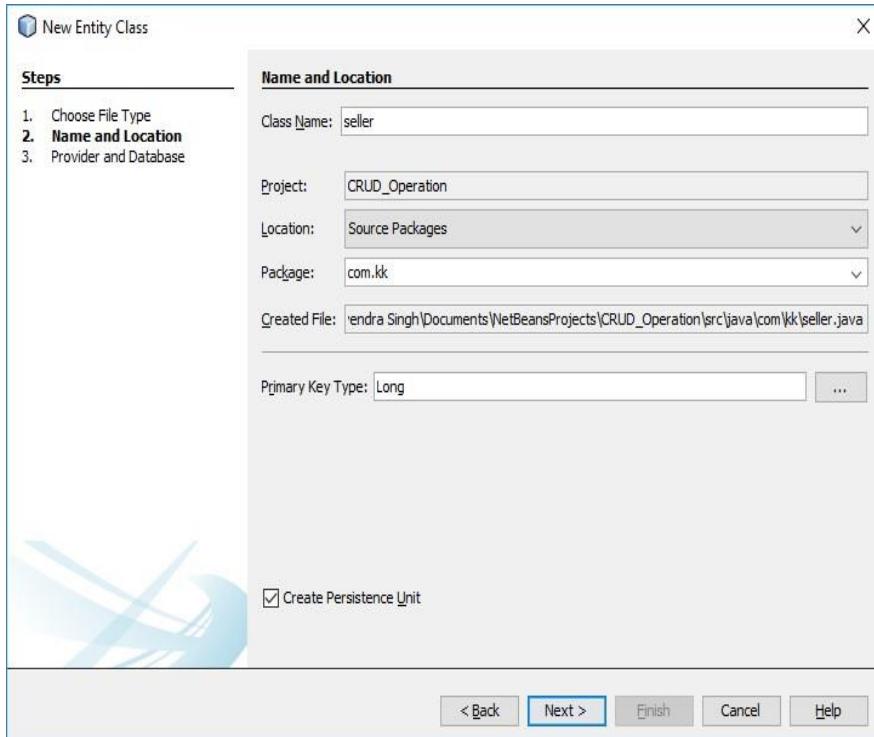
5. Create an entity class. Right click on project name -> New -> Entity Class.



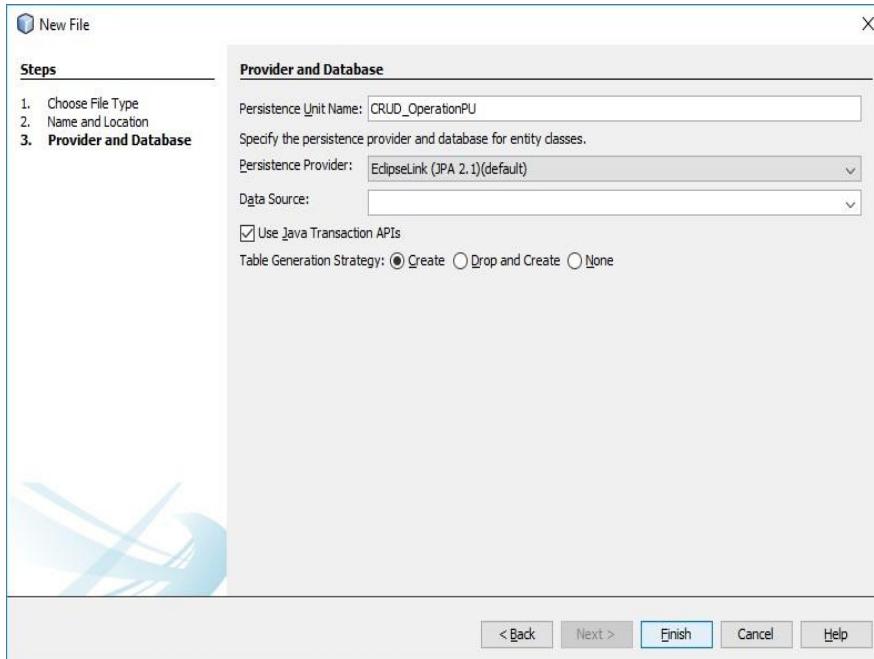
6. A window will appear like bellow pic. Enter following data and click on Next

Class Name -> seller

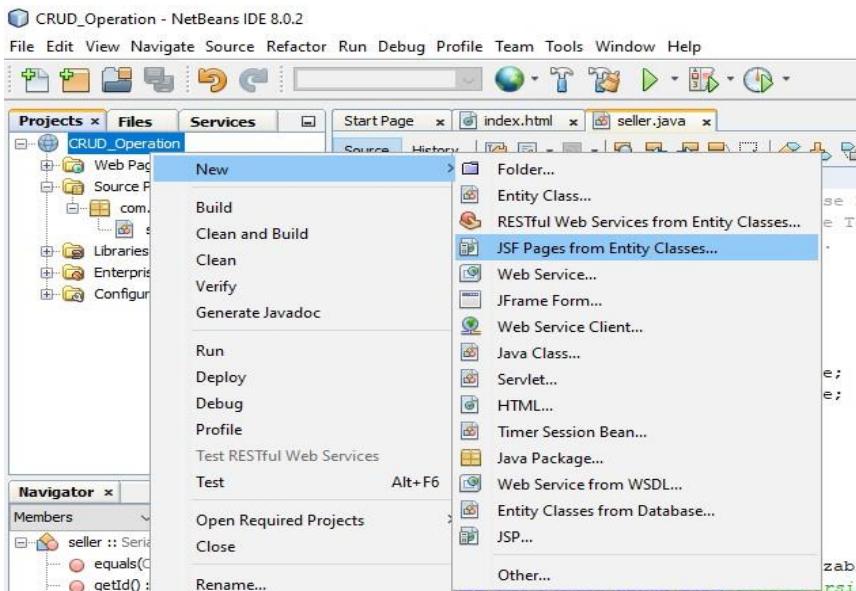
Package Name -> com.kk



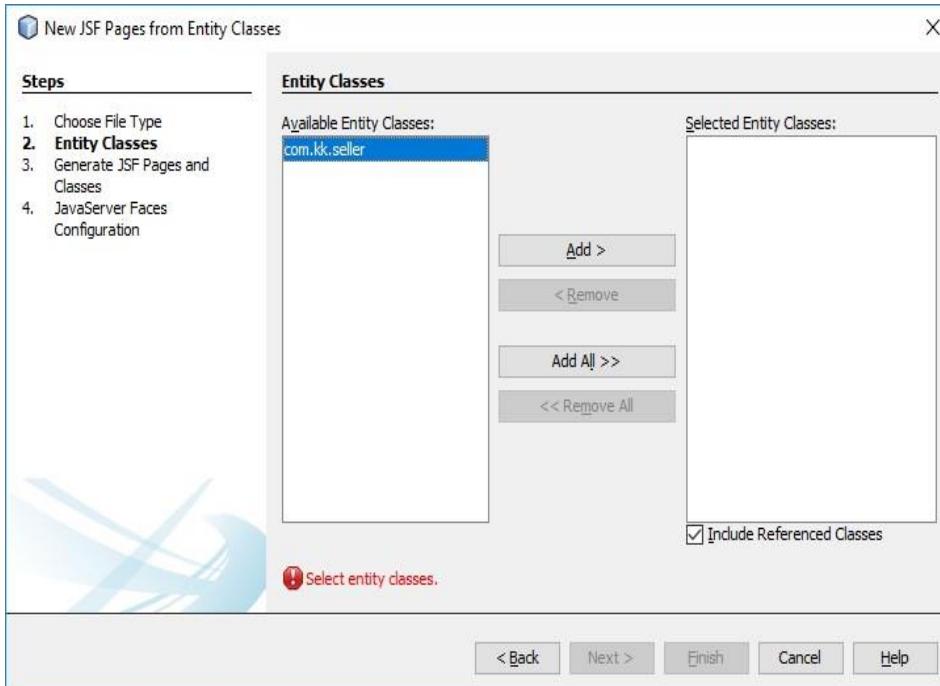
7. Click on Finish.



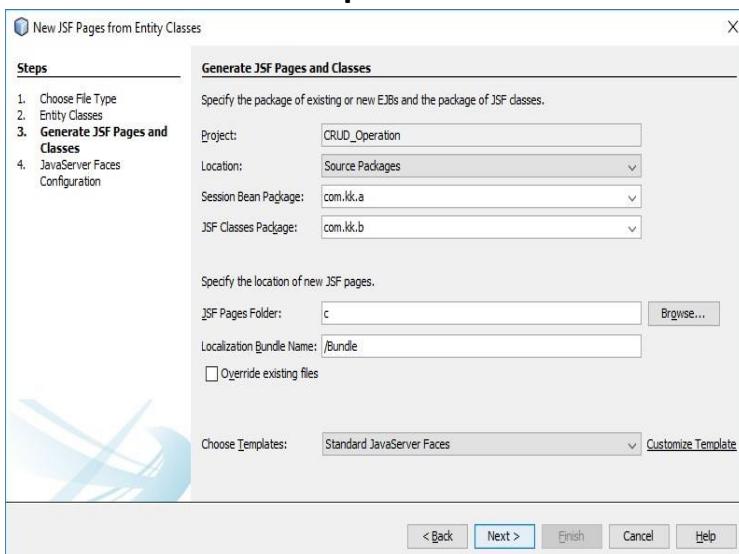
8. Right click on project name and create JSF Pages from Entity Classes. Right click on project name -> New -> JSF Pages from Entity Classes



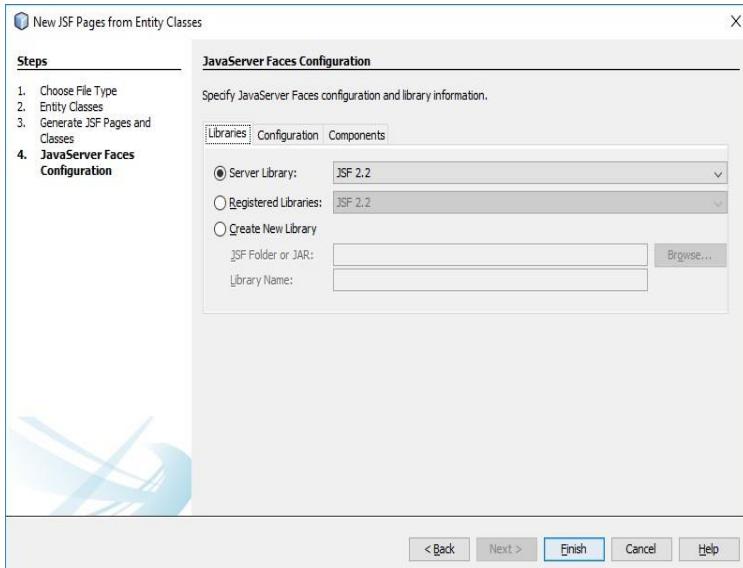
9. Select com.kk.seller and click on Add button and then Next button on below.



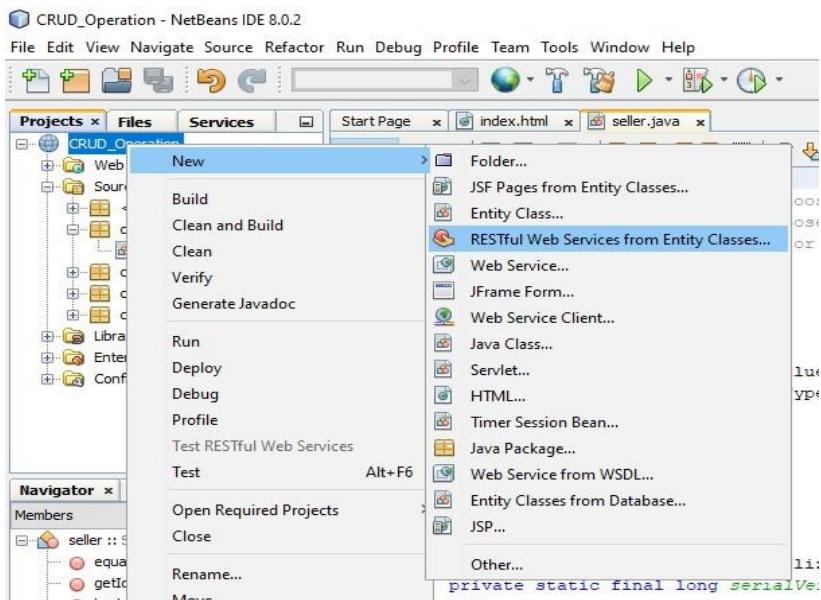
10. A window like below will appear on the screen. Enter the data into that window as entered in below pic and click on Next button.



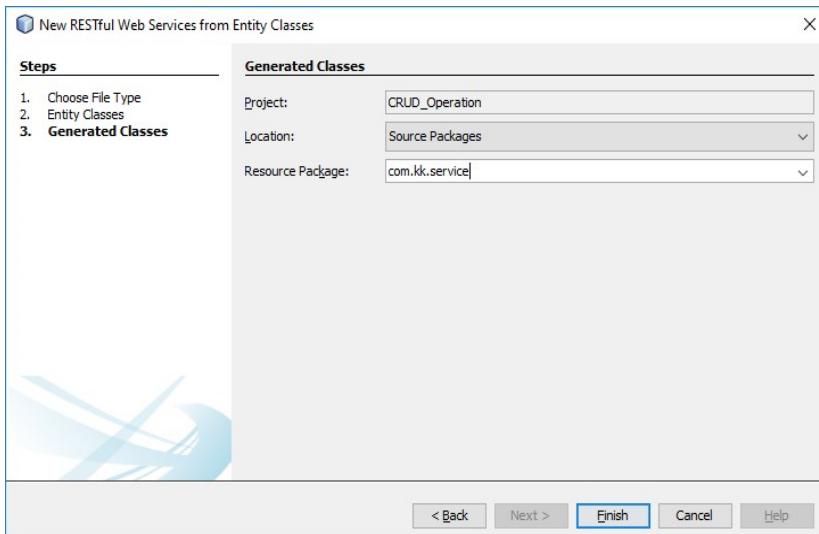
11. Now click on Finish.



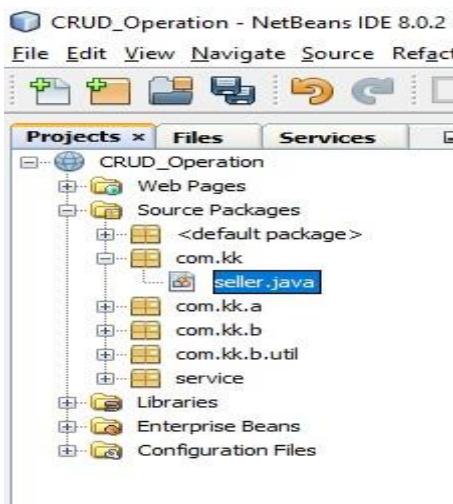
- 12.** Right click on project name and create RESTful Web Services from Entity Classes.
Right click on project name -> New -> RESTful Web Services from Entity Classes



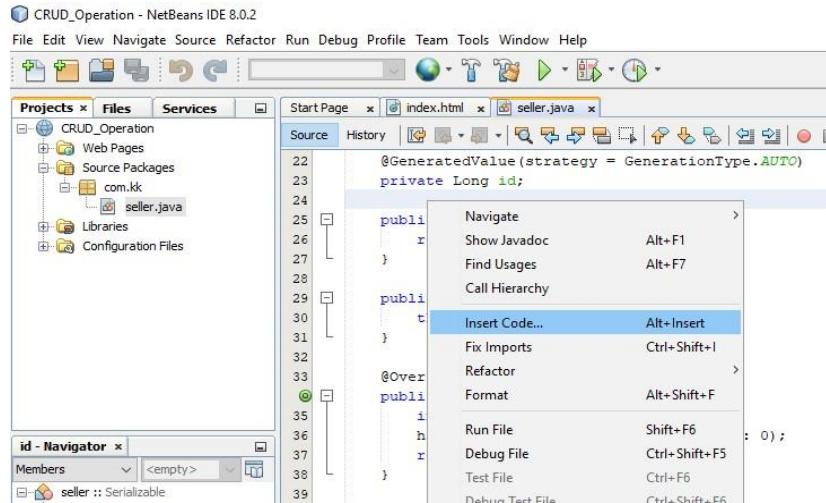
- 13.** Repeat step 9 and then it will go on next page. Then enter the com.kk.service in Resource Package and then click on Finish button.



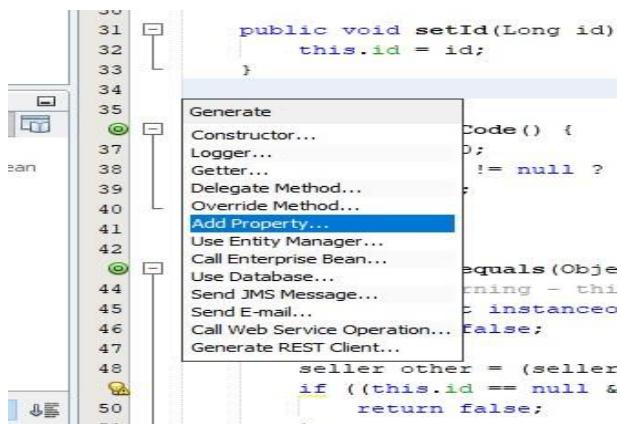
14. Now open seller.java file under com.kk package.



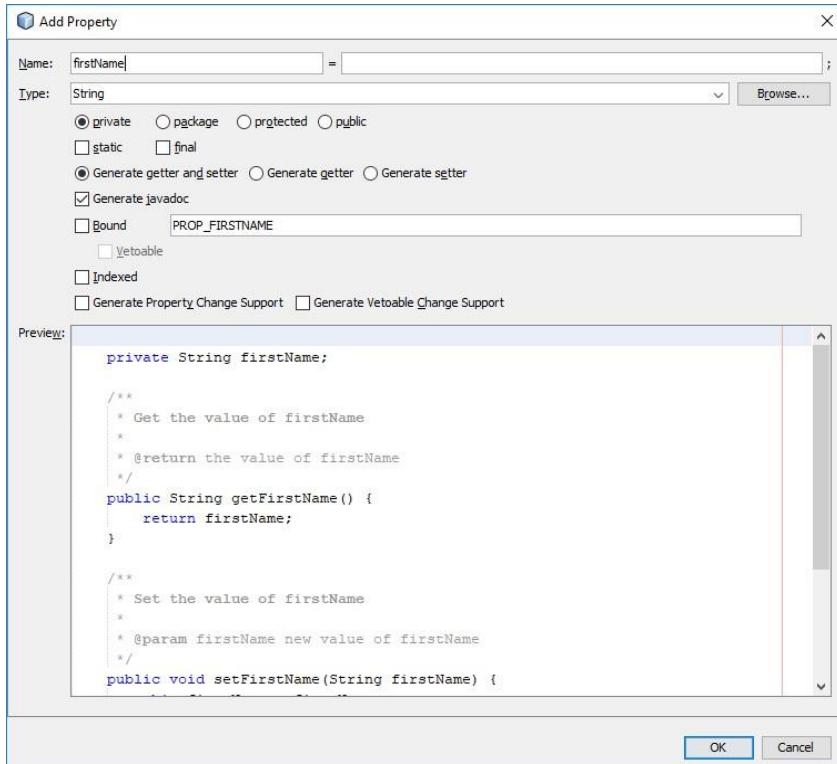
15. In this file at line number 24, do the right click and select Insert Code.



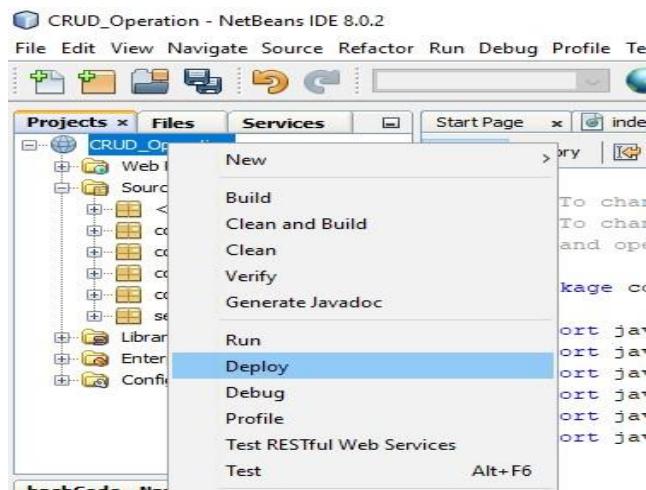
16. A new list will appear. Click on Add Property.



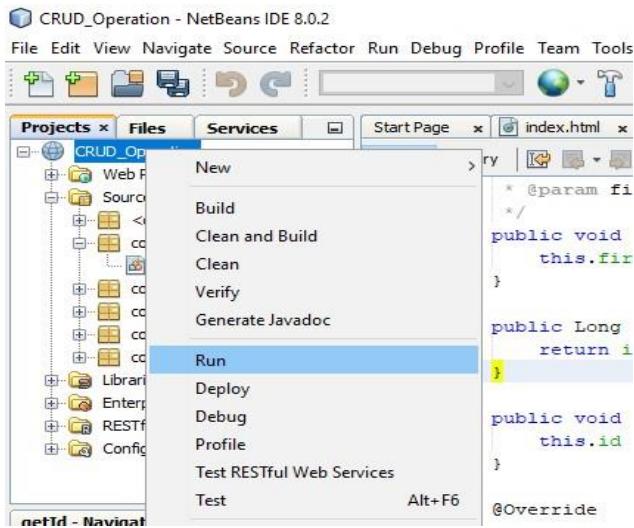
17. A new window will open. Enter name as firstName. Make sure name should be exact same as of mine and then click on OK button. Actually we are setting getter and setter method for firstName.



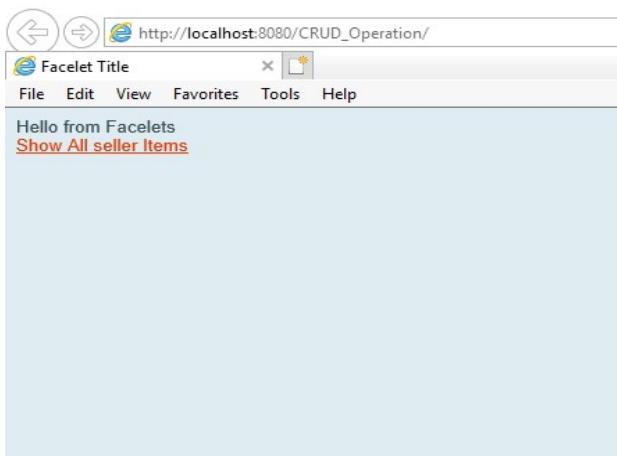
18. Now right click on web application name and Deploy it.



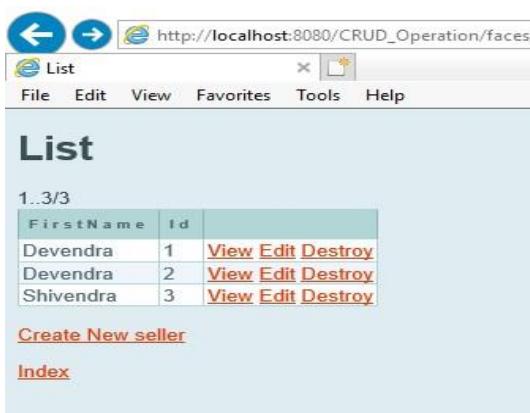
19. Now right click on project name and run it.



20. A window will open in browser like below....



21. Now click on Show All sellers Items for CRUD operation.



- 22.** 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.

FirstName	Id	
Devendra	1	View Edit Destroy
Devendra	2	View Edit Destroy
Shivendra	3	View Edit Destroy

[Create New seller](#)
[Index](#)

- 23.** 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](#)

- 24.** Now click on Show All seller Items to view all records whether our data is entered or not. We can see that one more Shivendra name is appearing.

FirstName	Id	
Devendra	1	View Edit Destroy
Devendra	2	View Edit Destroy
Shivendra	3	View Edit Destroy
Shivendra	4	View Edit Destroy

[Create New seller](#)

Practical-8 (Database)

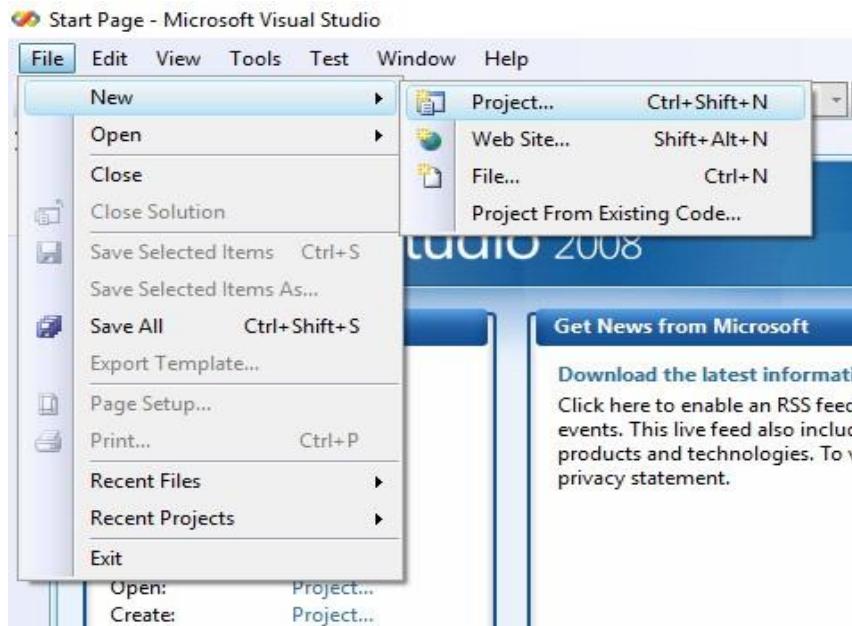
Aim: Implement a typical service and a typical client using WCF.

Requirements:

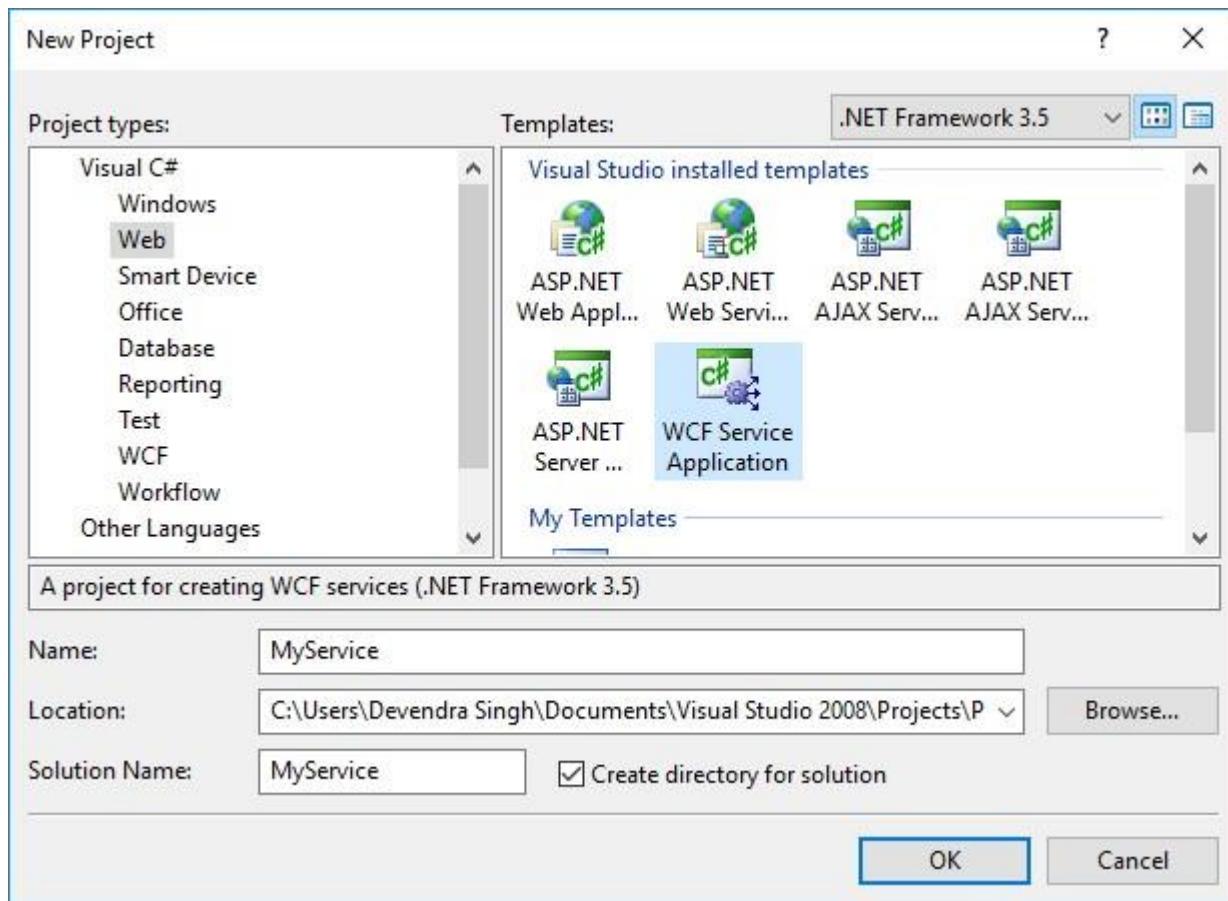
Visual Studio 2008 Professional.

Note: For practical 8, 9 & 10 Visual Studio is mandatory. Because we can implement WCF Service only on Visual Studio.

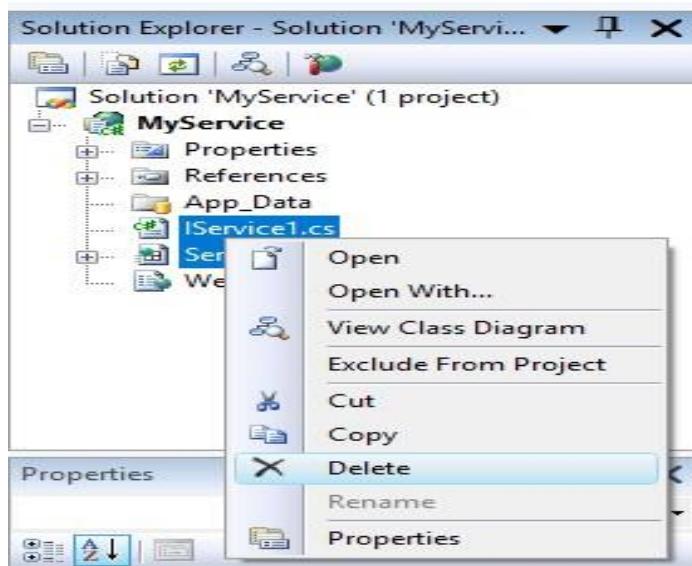
1. Open Visual Studio and open Project.



2. In Project Types, select Web and in Templates select WCF Service Application. Give it name as MyService and click on OK button.

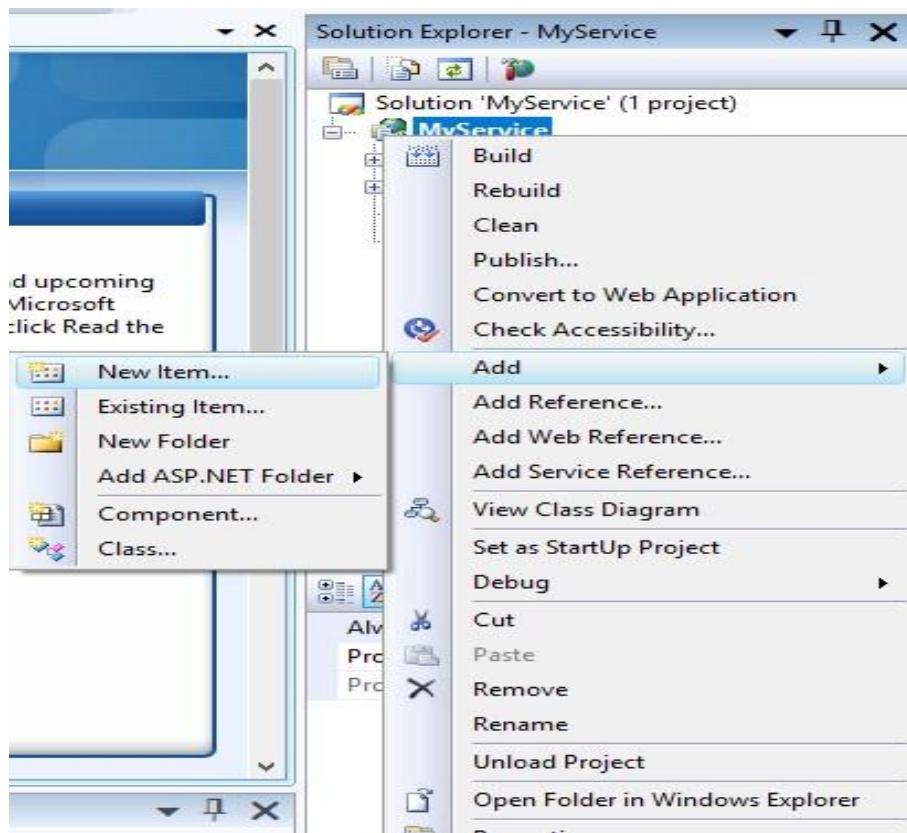


3. **Delete the selected two files** from Solution Explorer placed on right hand side. Actually we are going to delete these files, because this is default WCF Service file and we want to create our own WCF Web Service.



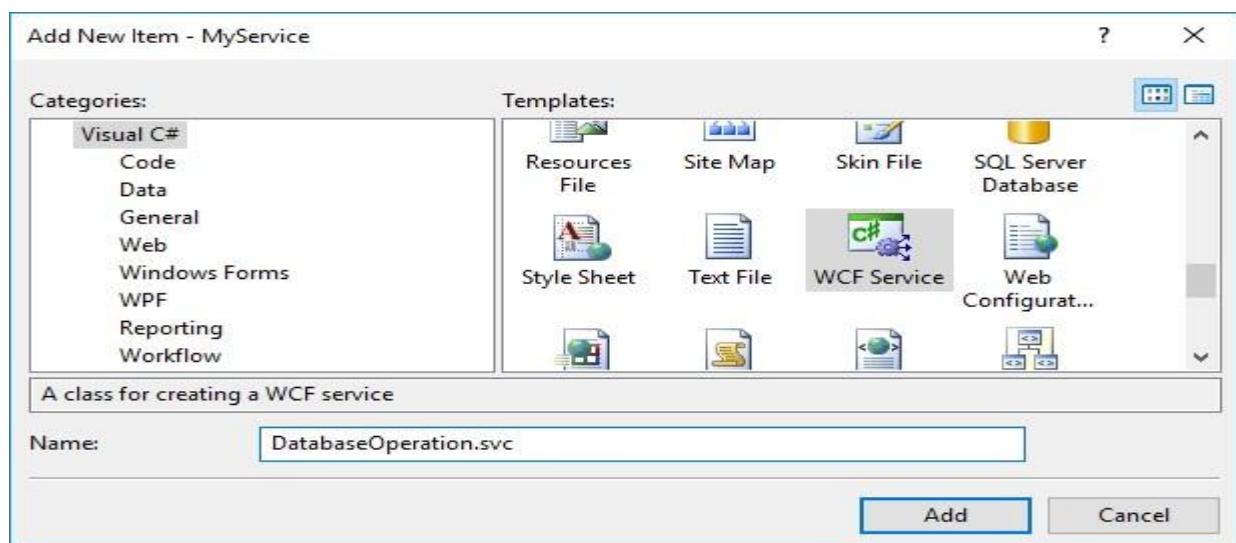
4. Now to create our own WCF web service.

Right click on MyService -> Add -> New Item

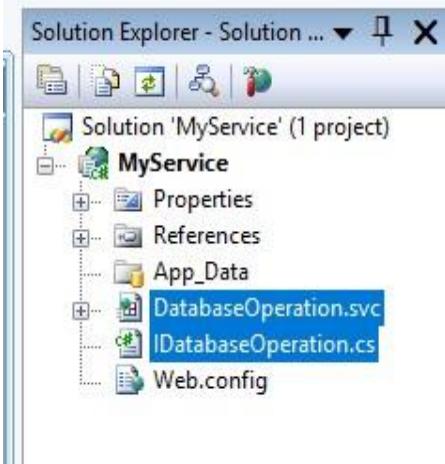


5. Select WCF Service in Templates and give its name as DatabaseOperation.

Press Add button.



6. You can see DatabaseOperation service is added in Solution Explorer.

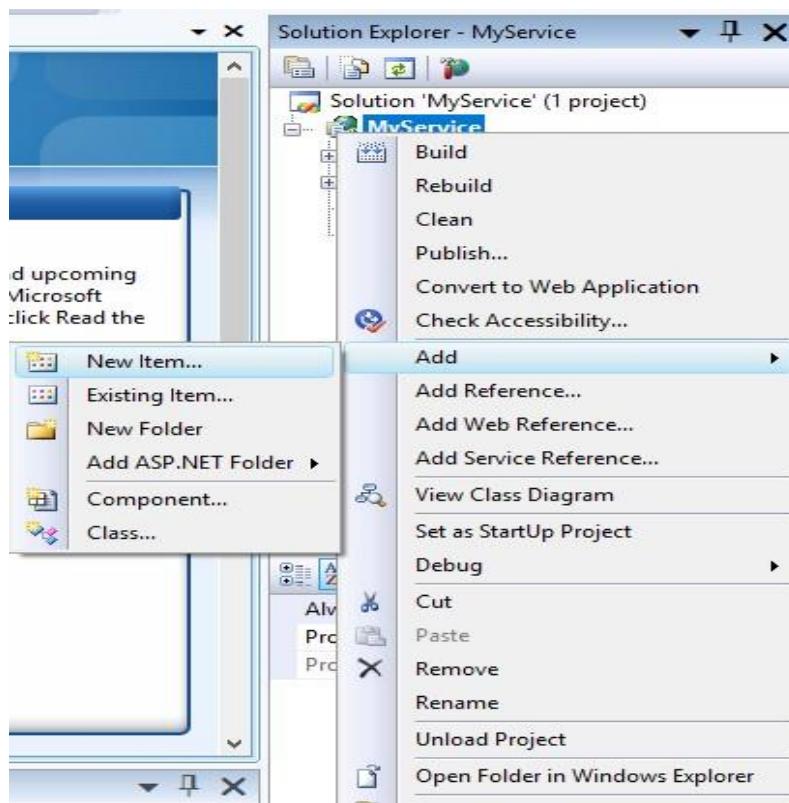


7. Now we are going to create and add a database. On which we will do insert and retrieve operation.

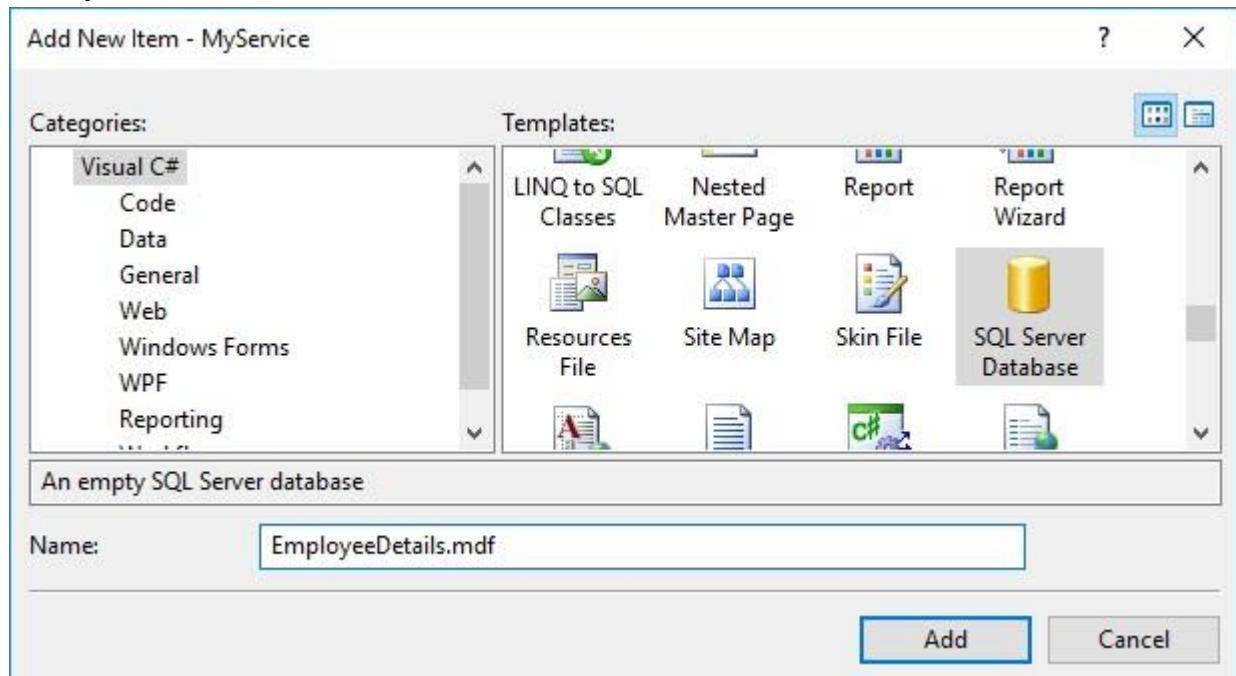
Creating Database with table

8. Now to create database and table.

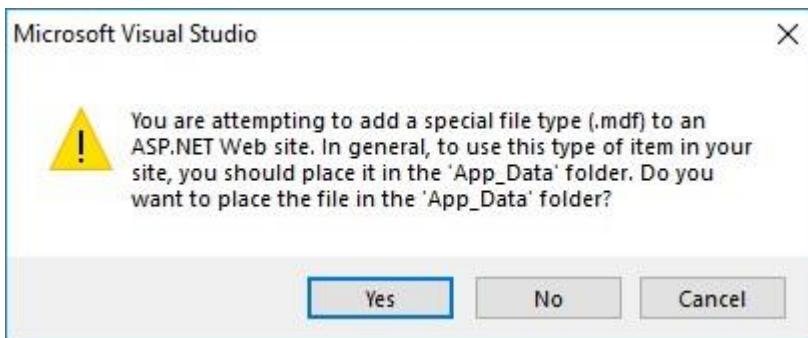
Right click on MyService -> Add -> New Item



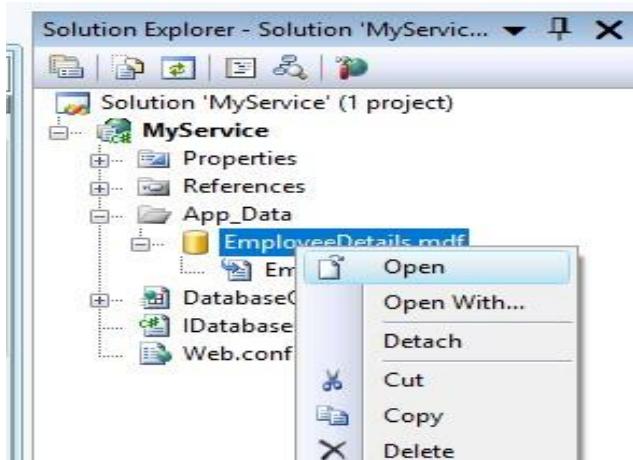
- 9. Select SQL Server Database in Templates. Give it name as EmployeeDetail and press Add button.**



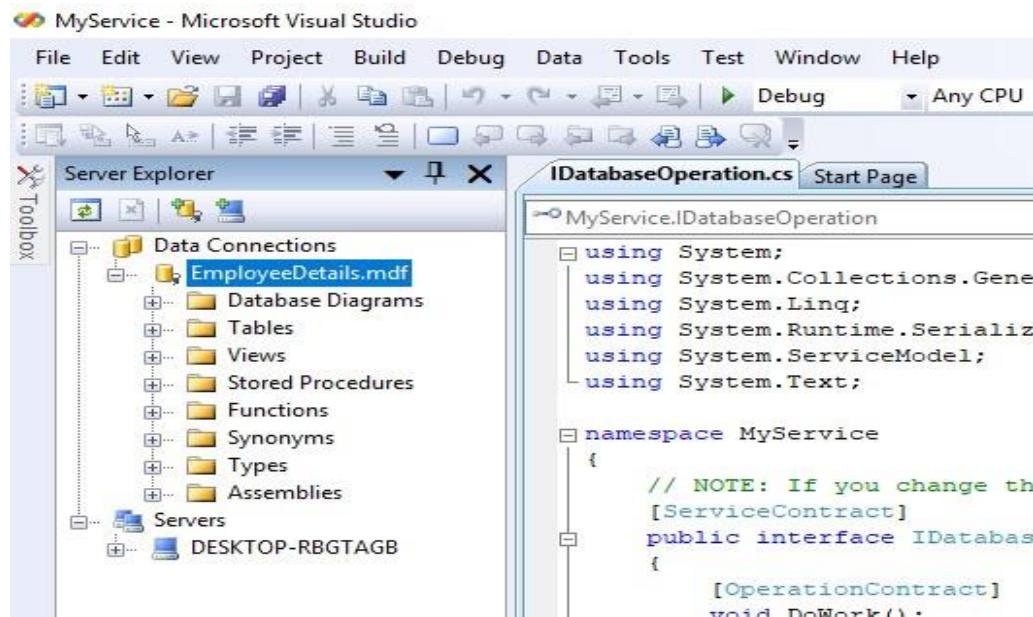
- 10. Press Yes.**



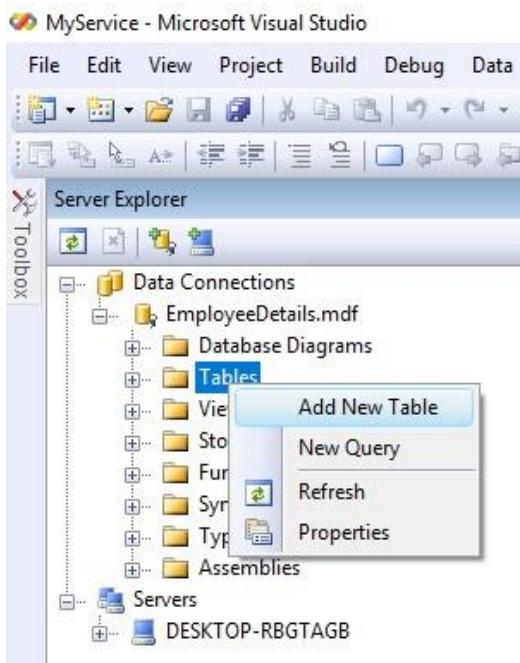
- 11. Right click on EmployeeDetail.mdf -> Open**



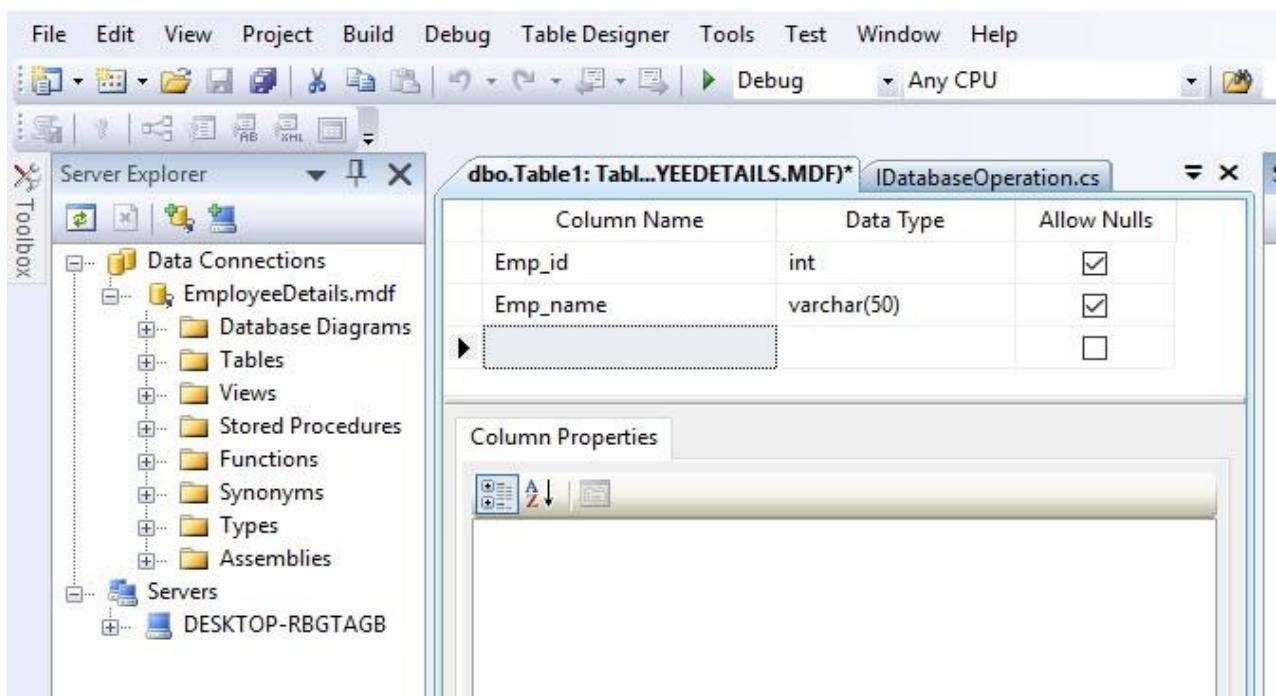
12. As you can see Server Explorer is opened in the right side of my screen. Now I will add a table with name Employee.



13. Right click on Tables and select Add New Table.



14. Create two columns. One Emp_id and second Emp_name.



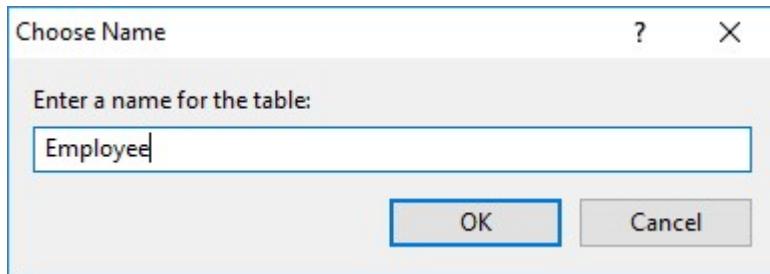
15. Select Emp_id column and make it as primary key by click on Key icon.

The screenshot shows the SSMS interface. In the Object Explorer on the left, under 'EmployeeDetails.mdf', 'Tables' is selected. A context menu is open over the 'Tables' node, with 'Set Primary Key' highlighted. The main pane displays the table structure for 'dbo.Table3'. The 'Column Name' column contains 'Emp_id' and 'Emp_name'. The 'Data Type' column shows 'int' for 'Emp_id' and 'varchar(50)' for 'Emp_name'. The 'Allow Nulls' column has checked boxes for both. Below the table structure, the 'Column Properties' section shows '(Name)' as 'Emp_id', 'Allow Nulls' as 'Yes', and 'Data Type' as 'int'.

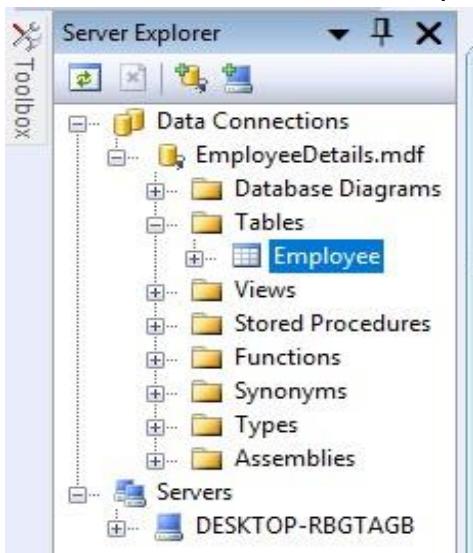
16. Save All.

The screenshot shows the Microsoft Visual Studio interface with the title bar 'MyService - Microsoft Visual Studio'. The 'File' menu is open, and the 'Save All' option is highlighted with a blue selection bar. Other options in the menu include 'New', 'Open', 'Add', 'Close', 'Close Solution', 'Save Selected Items', 'Save Selected Items As...', 'Export Template...', 'Page Setup...', 'Print...', and 'Recent Files'.

17. It will ask for table name. Give Employee name to it and press OK.

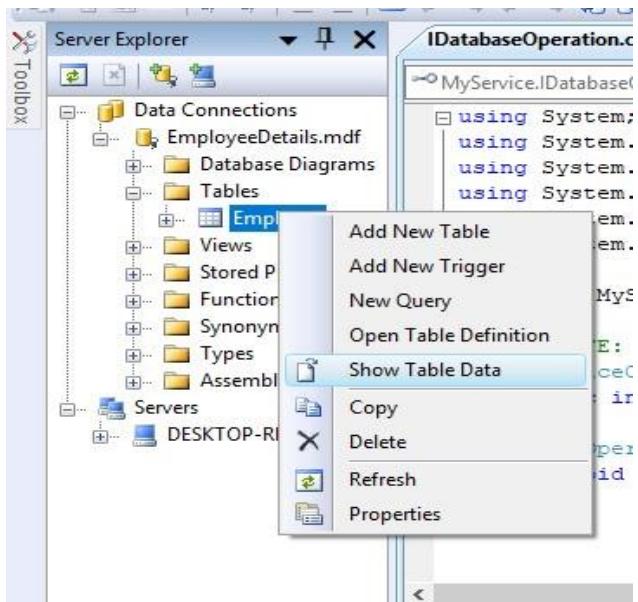


18. You can see a table with Employee name is created.

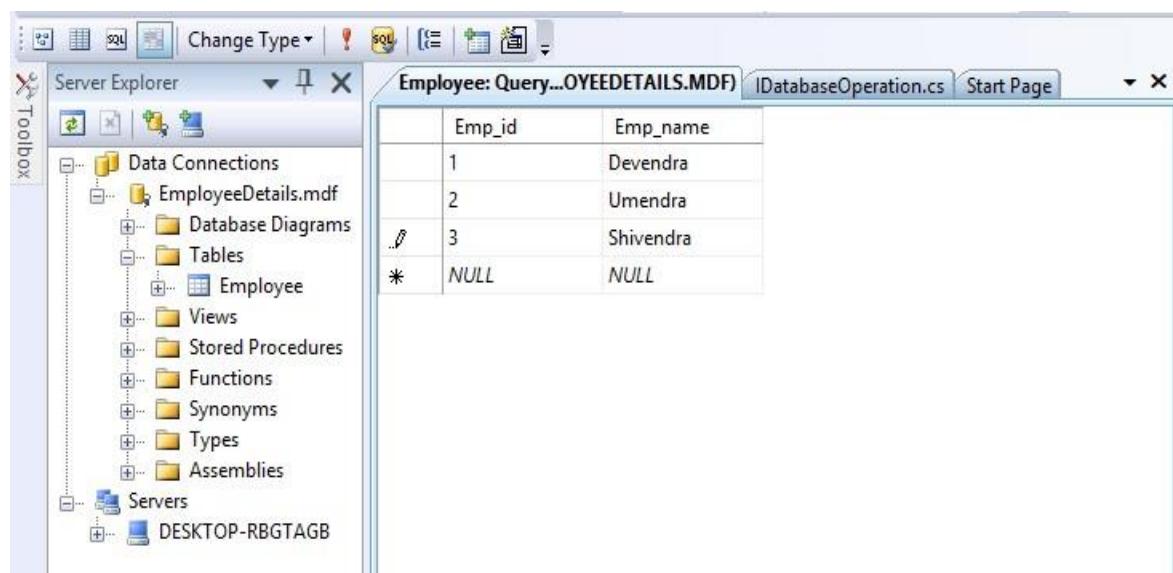


19. Now enter some data into this table.

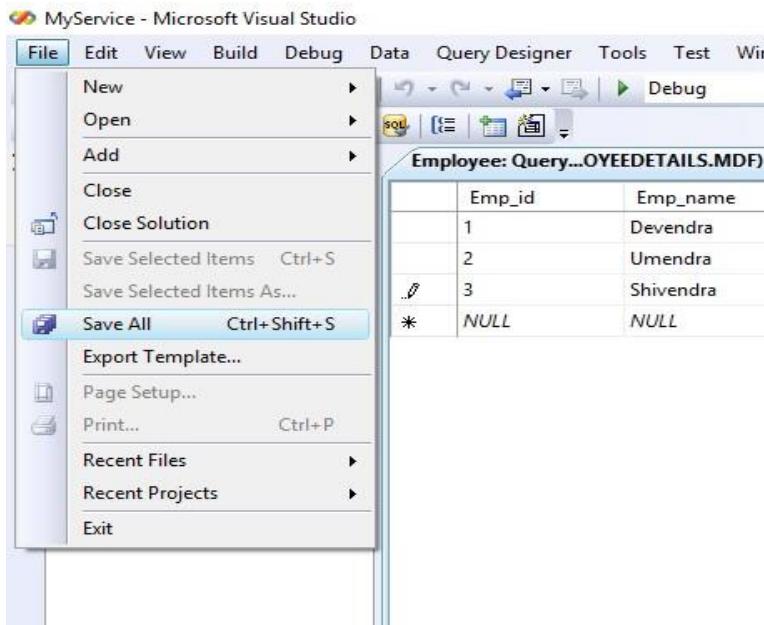
Right click on Employee -> Show Table Data



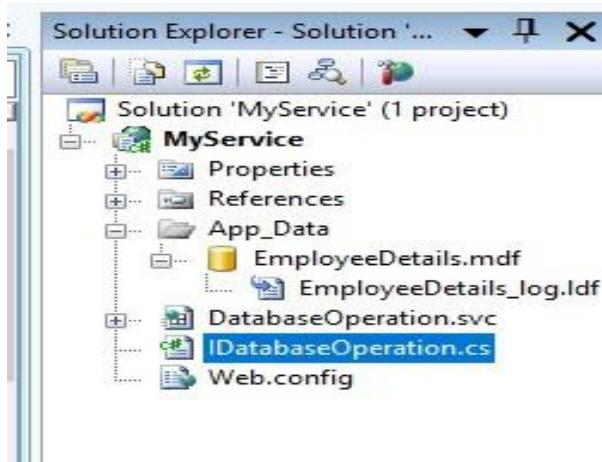
20. As Emp_id is primary key, so do not repeat it.



21. Now again Save All.



22. So in this way we have created required database with table. Now we are going to connect this database to WCF service. So we can do operations on it.
23. Double click on **IDatabaseOperation.cs** file to open it and inserting some code.



It will look like below.

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

namespace MyService
{
    // NOTE: If you change the interface name "IDatabaseOperation"
    [ServiceContract]
    public interface IDatabaseOperation
    {
        [OperationContract]
        void DoWork();
    }
}
```

24. Now add the following code into above selected part and press **Ctrl+S** to save the changes.

```
[OperationContract]
string InsertData(Employee tbl);

[OperationContract]
Employee GetData(int Emp_id);
}

[DataContract]
public class Employee
{

    [DataMember]
    public int Emp_id { get; set; }

    [DataMember]
    public string Emp_name { get; set; }
```

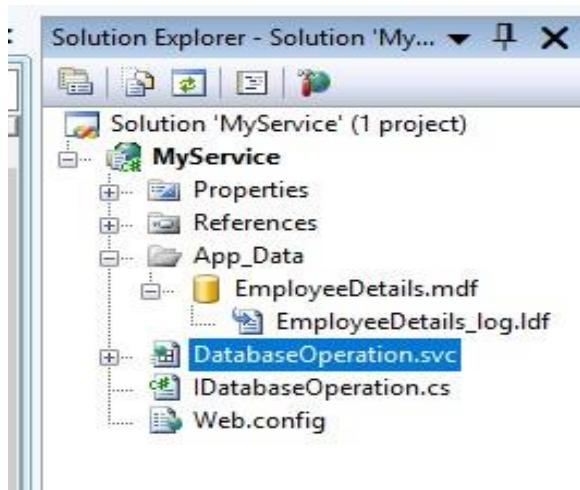
```
DatabaseOperation.svc.cs* | IDatabaseOperation.cs | Start Page
MyService.IDatabaseOperation
{
    // NOTE: If you change the interface name "IDatabaseOperation", please update the reference to "MyService.IDatabaseOperation" in DatabaseOperation.svc.cs
    [ServiceContract]
    public interface IDatabaseOperation
    {
        [OperationContract]
        string InsertData(Employee tbl);

        [OperationContract]
        Employee GetData();
    }

    [DataContract]
    public class Employee
    {
        [DataMember]
        public int Emp_id { get; set; }

        [DataMember]
        public string Emp_name { get; set; }
    }
}
```

25. Now open DatabaseOperation.svc.cs file by double click on selected file.



26. Now import following package.

```
using System.Data.SqlClient;
```

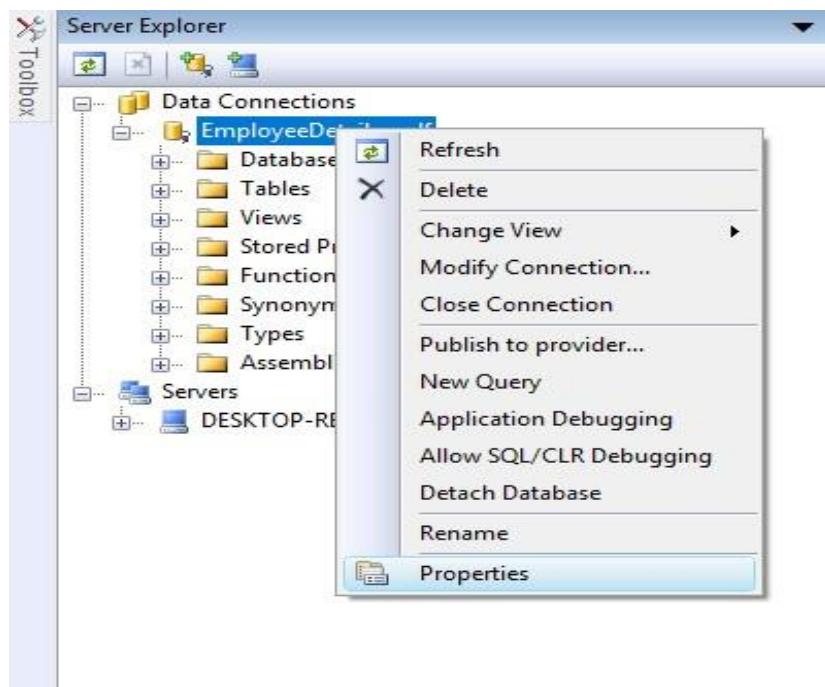
```
DatabaseOperation.svc.cs* IDatabaseOperation.cs Start Page
MyService.DatabaseOperation
DoWork

using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.Text;
using System.Data.SqlClient;

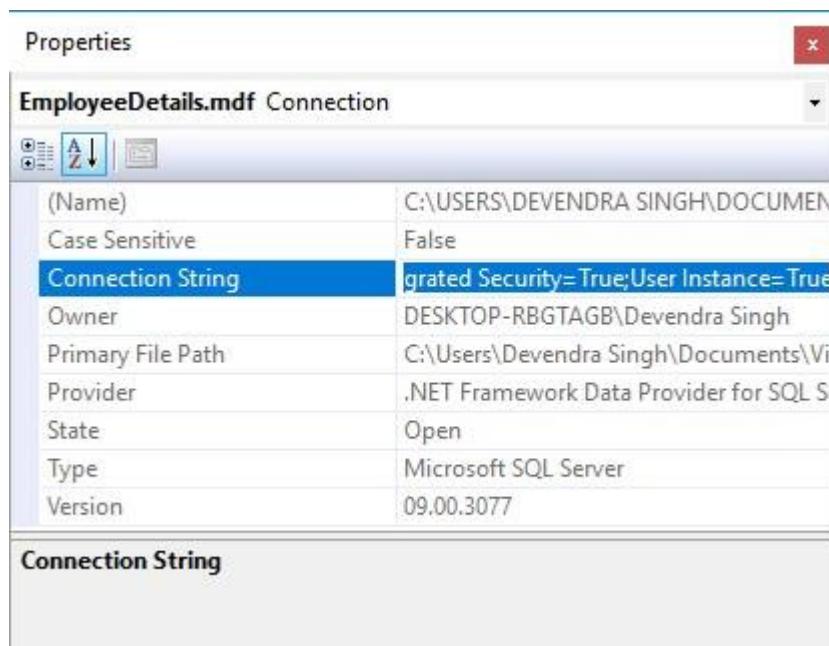
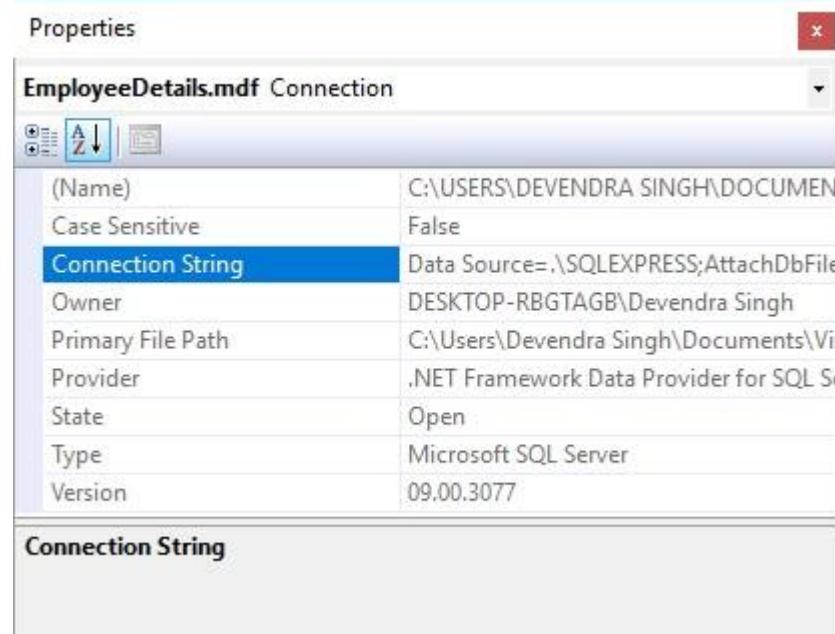
namespace MyService
{
    // NOTE: If you change the class name "DatabaseOperation"
    public class DatabaseOperation : IDatabaseOperation
    {
```

Before giving code, I'm going to tell you that how we get the database link which is used to connect with database.

If database is not open, then open it by follow step 11. Right click on EmployeeDetails.mdf -> Properties.



Copy the full link in Connection String.



Below is copied link.

```
Data Source=.\SQLEXPRESS;AttachDbFilename="C:\Users\Devendra  
Singh\Documents\Visual Studio  
2008\Projects\Project1\MyService\MyService\App_Data\EmployeeDetails.md  
f";Integrated Security=True;User Instance=True
```

But **above link is not appropriate to use. So to make it appropriate link by replacing \ to \\ and replace " to '.**

Appropriate link will be as bellow....

```
Data Source=.\\\SQLEXPRESS;AttachDbFilename='C:\\\\Users\\\\Devendra  
Singh\\\\Documents\\\\Visual Studio  
2008\\\\Projects\\\\Project1\\\\MyService\\\\MyService\\\\App_Data\\\\EmployeeDetai  
ls.mdf';Integrated Security=True;User Instance=True
```

In above link red marked parts are the changes to make it appropriate and I will use this link for connection establishment.

Note: I have changed from " to '. Because I will paste this link between " ". If you will paste it in ' ', then don't make any changes in link for ". Leave as it is. Just change the \ only.

27. Resuming from step 26.

```

DatabaseOperation.svc.cs IDatabaseOperation.cs Start Page
MyService.DatabaseOperation DoWork()
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.Text;
using System.Data.SqlClient

namespace MyService
{
    // NOTE: If you change the class name "DatabaseOperation" here, you must also
    public class DatabaseOperation : IDatabaseOperation
    {
        public void DoWork()
        {
        }
    }
}

```

28. Now add the following code into selected part in above pic. And press Ctrl+S to save the changes.

```

public string InsertData(Employee tbl) {

    string resp;

    SqlConnection con = new SqlConnection("Data
Source=.\SQLExpress;AttachDbFilename='C:\\Users\\Devendra
Singh\\Documents\\Visual Studio
2008\\Projects\\Project1\\MyService\\MyService\\App_Data\\EmployeeDetails.mdf';Integrated Security=True;User Instance=True");
    SqlCommand cm = new SqlCommand("insert into
Employee(Emp_id,Emp_name) values (@Emp_id, @Emp_name)", con);
    cm.Parameters.AddWithValue("@Emp_id", tbl.Emp_id);
    cm.Parameters.AddWithValue("@Emp_name", tbl.Emp_name);
    con.Open();
    int res = cm.ExecuteNonQuery();
if (res == 1)
{
    resp = "Data inserted";
}
else
{
    resp = "Data not inserted";
}

```

```

        }
con.Close();
return resp;

    }
public Employee GetData(int Emp_id) {

    Employee tbl = new Employee();

    SqlConnection con = new SqlConnection("Data
Source=.\SQLEXPRESS;AttachDbFilename='C:\\Users\\Devendra
Singh\\Documents\\Visual Studio
2008\\Projects\\Project1\\MyService\\MyService\\App_Data\\Employ
eeDetails.mdf';Integrated Security=True;User Instance=True");
    SqlCommand cm = new SqlCommand("select * from Employee
where Emp_id = @Emp_id", con);
    cm.Parameters.AddWithValue("@Emp_id", Emp_id);

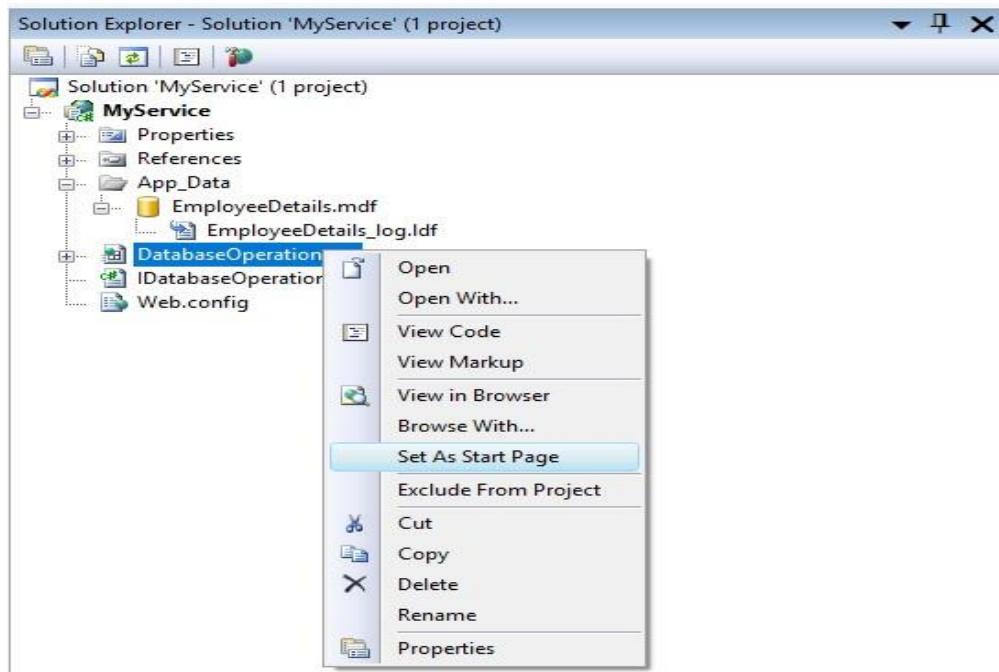
    con.Open();
SqlDataReader data;
cm.ExecuteReader();

    if (data != null)
    {
        if (data.Read())
        {
            tbl.Emp_id = (int)data["Emp_id"];
tbl.Emp_name = data["Emp_name"].ToString();
        }
    else
        {
            tbl = null;
        }
    }
con.Close();
return tbl;
}

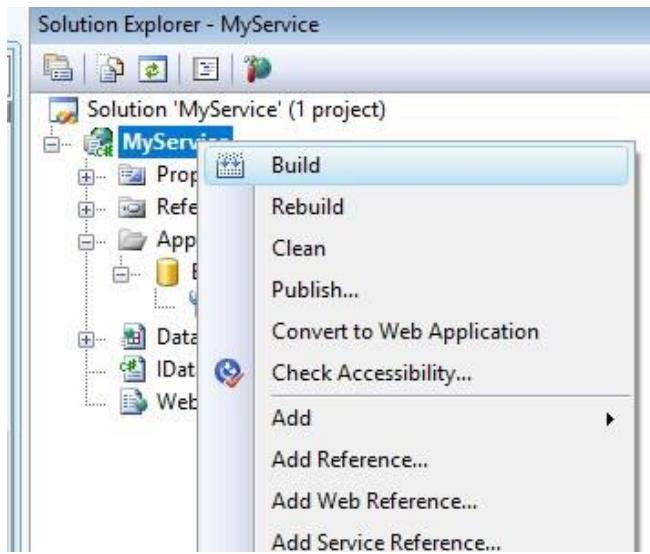
```

NOTE: Use your own database link in `sqlConnection()` method in above code.

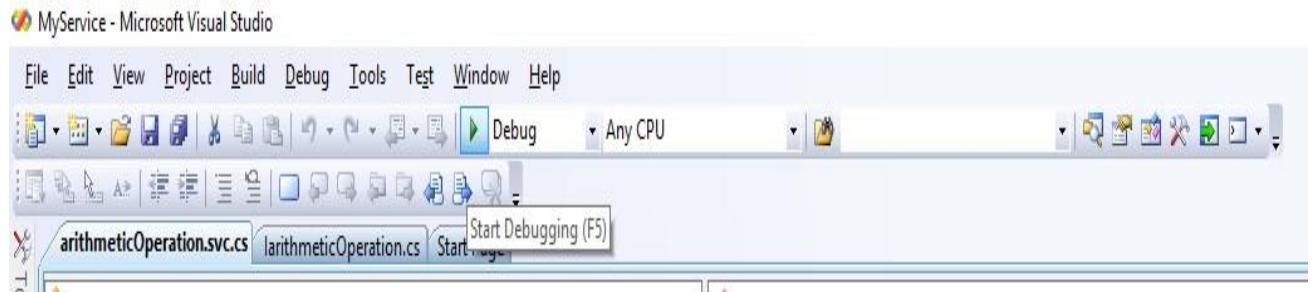
29. Now make DatabaseOperation.svc file as start page.



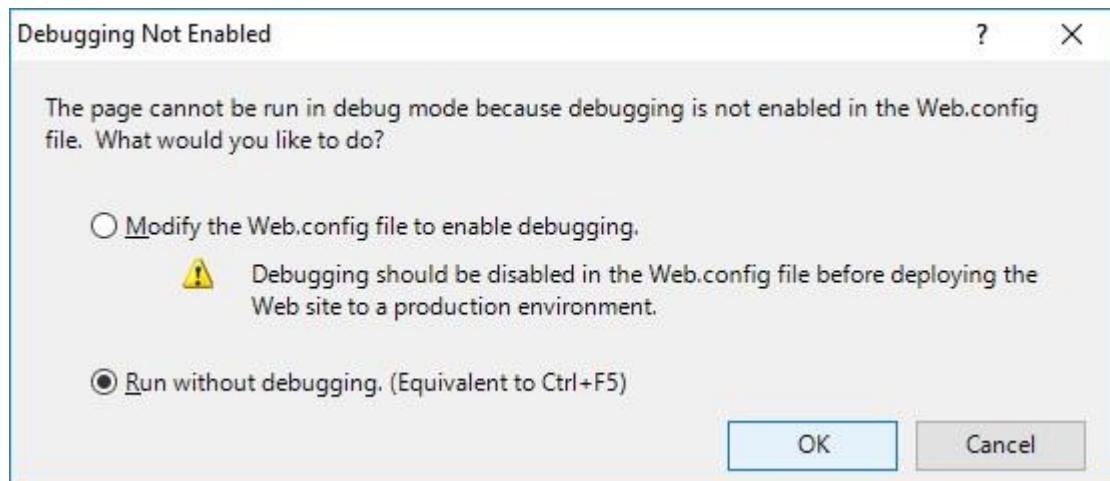
30. Now right click on **MyService** in Solution Explorer and **select Build** to build the project.



31. Now **click on run button** to run the project.



32. If any window appears like below then **select Run without debugging** and **click on OK button**.



33. A new window will open in browser like below.

The screenshot shows a web browser window with two tabs open. The active tab is titled "DatabaseOperation Service". The URL bar shows "localhost:64430/DatabaseOperation.svc". Below the URL bar, there is a message: "For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...". The main content area has a title "DatabaseOperation Service". It includes a note: "You have created a service. To test this service, you will need to create a client and use it to call the service. You can do this using the svcutil.exe tool from the command line with the following command: svcutil.exe http://localhost:64430/DatabaseOperation.svc?wsdl". Below this, there are two code snippets: one for C# and one for Visual Basic.

C#

```
class Test
{
    static void Main()
    {
        DatabaseOperationClient client = new DatabaseOperationClient();
        // Use the 'client' variable to call operations on the service.
        // Always close the client.
        client.Close();
    }
}
```

Visual Basic

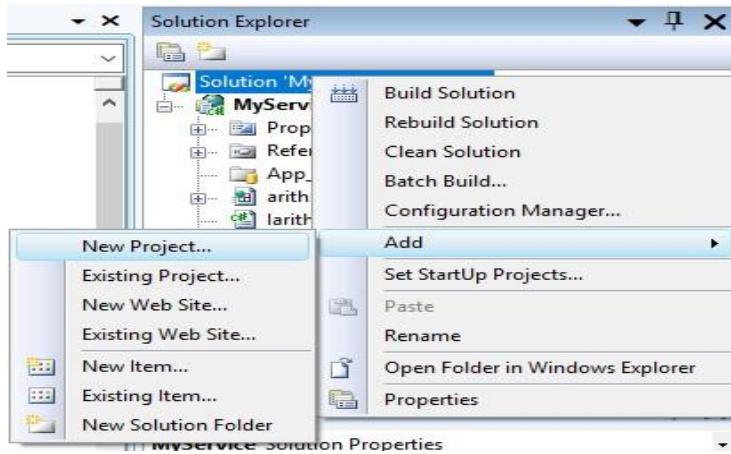
```
Class Test
    Shared Sub Main()
        Dim client As DatabaseOperationClient = New DatabaseOperationClient()
        ' Use the 'client' variable to call operations on the service.
        ' Always close the client.
        client.Close()
    End Sub
End Class
```

34. Now copy the URL. As in my case it is....

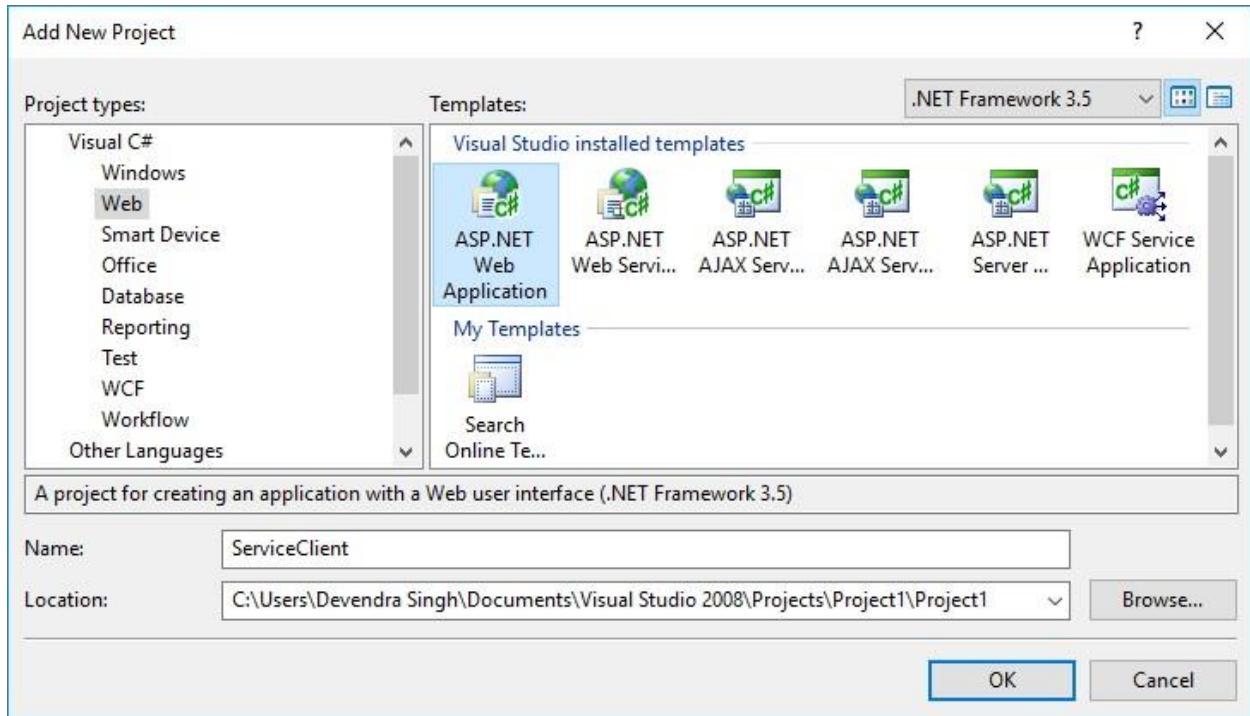
<http://localhost:64430/DatabaseOperation.svc>

35. Now take care, you can close the browser. But don't close the Solution Explorer or Visual Studio, otherwise inbuilt server will stop and you will get error during consuming this service. Even, you have to create the client within the same Solution Explorer. To solve this problem we can host this service on IIS. But it will take more steps and for our practical we don't need this.

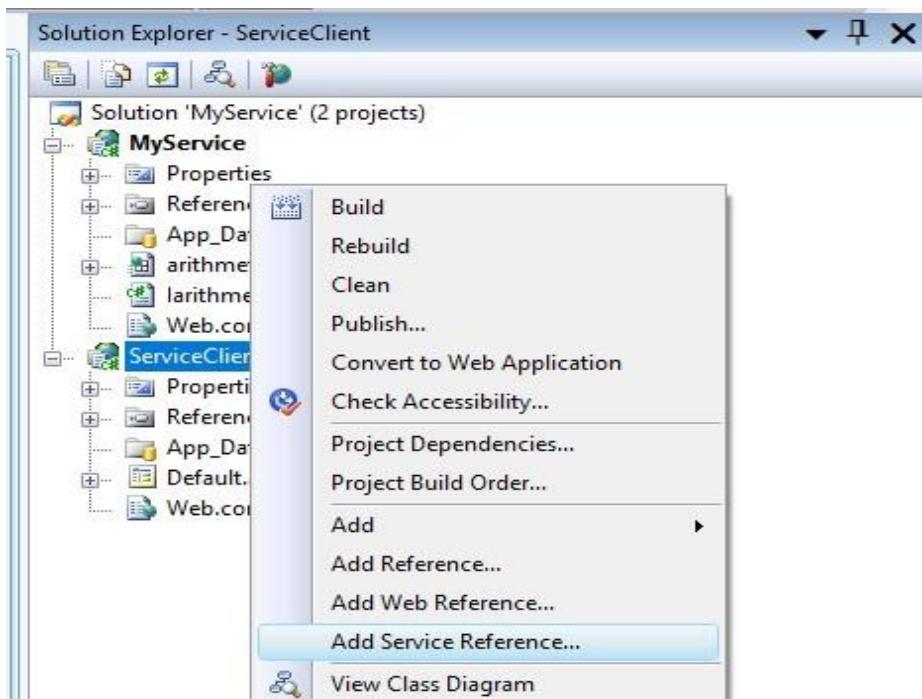
36. Now right click on solution explorer and add a new project to create client.



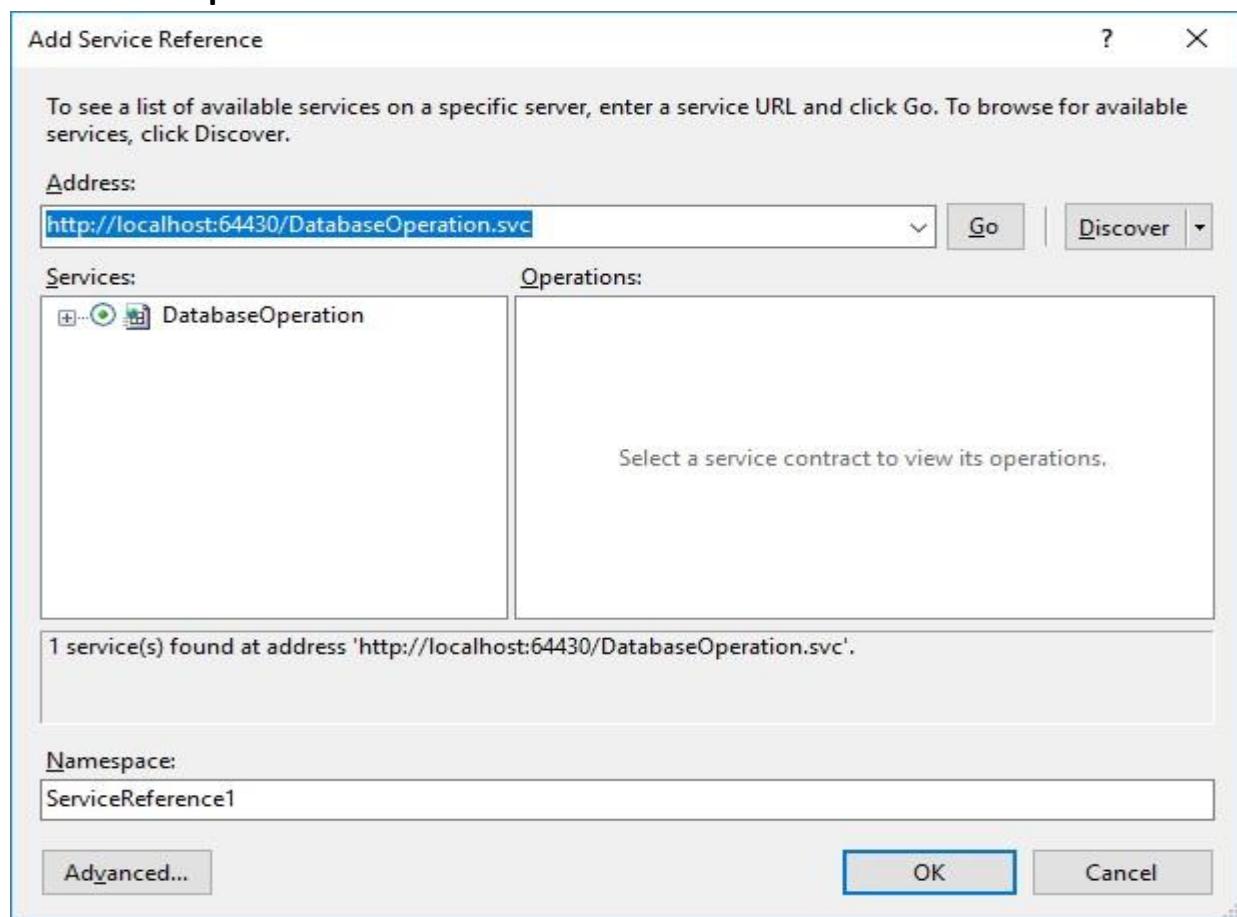
37. In Project Type select Web and in Templates select ASP.Net Web Application. Give it name as ServiceClient and then click on OK button.



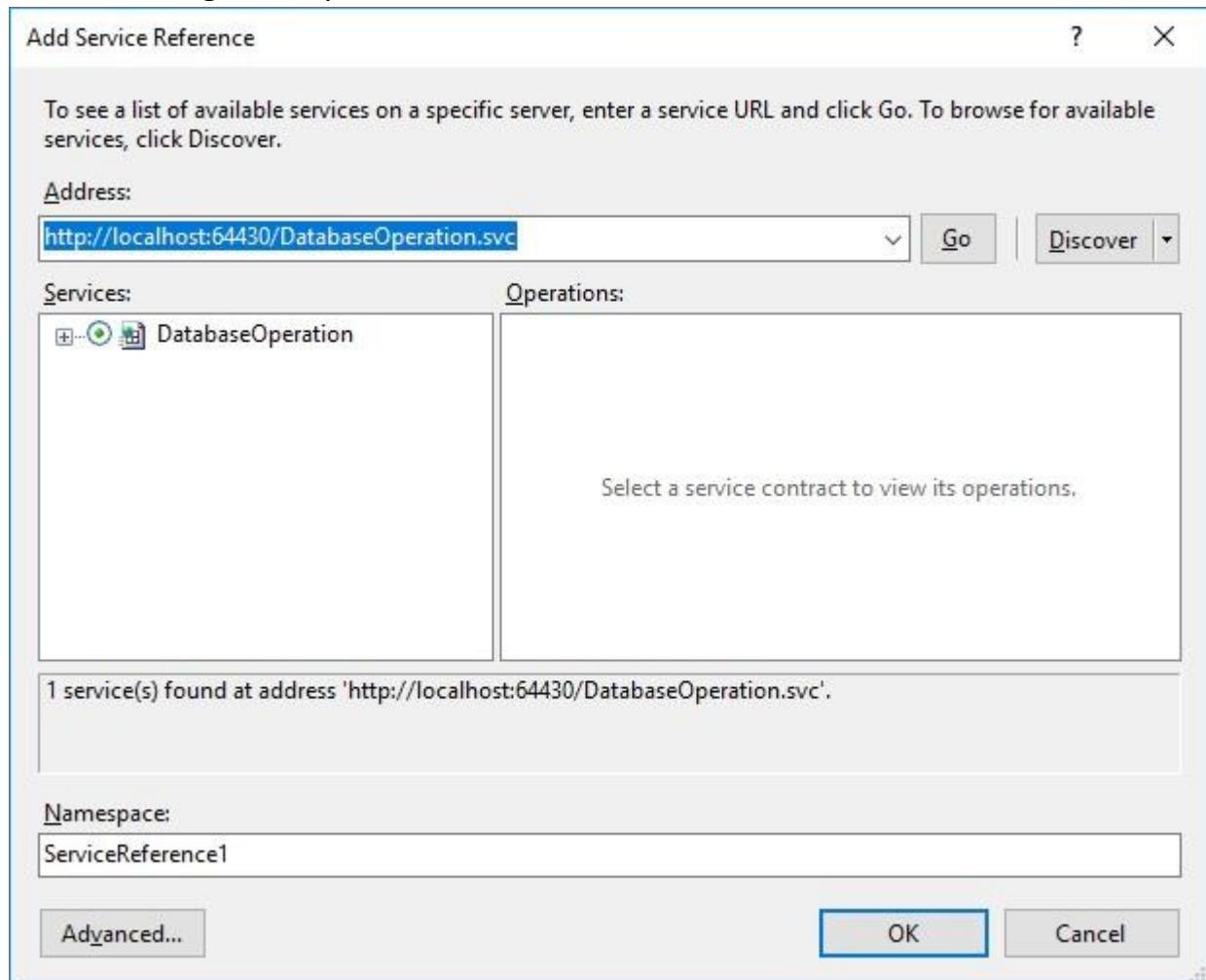
38. Right click on Service ServiceClient in Solution Explorer and select Add Service Reference.



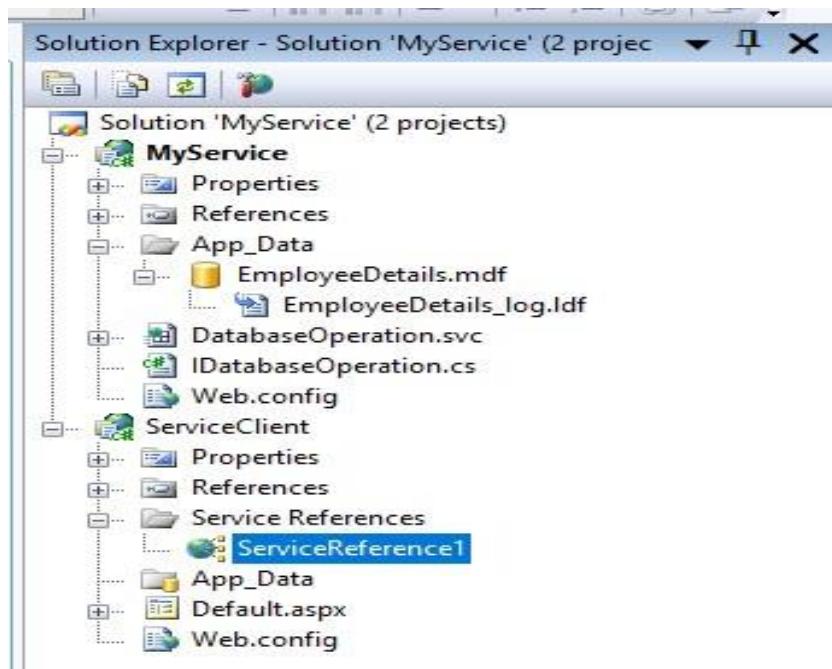
39. Paste the copied link into Address and click on Go button.



40. As you can see DatabaseOperation Service is fetched. Now I'm leaving **Namespace as it is i.e. ServiceReference1**. Actually this namespace will be used to call the Service methods. If you will change it then you have to use the according namespace to call the service methods. **Click on OK button.**



41. As you can see Service Reference is added with Namespace in Solution Explorer.



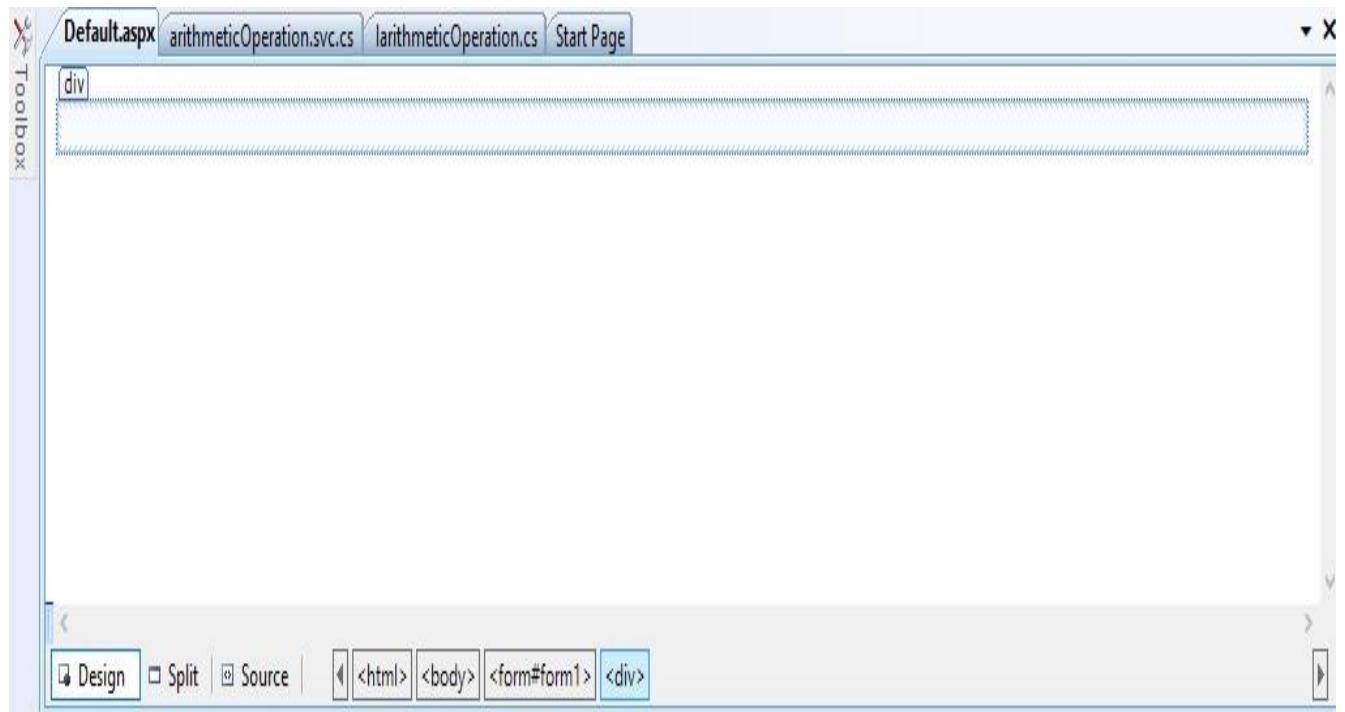
42. Default.aspx page of ServiceClient will appear by default in Source mode.
Otherwise you can open it from solution explorer.

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs" Inherits="ServiceClient._Default" %>

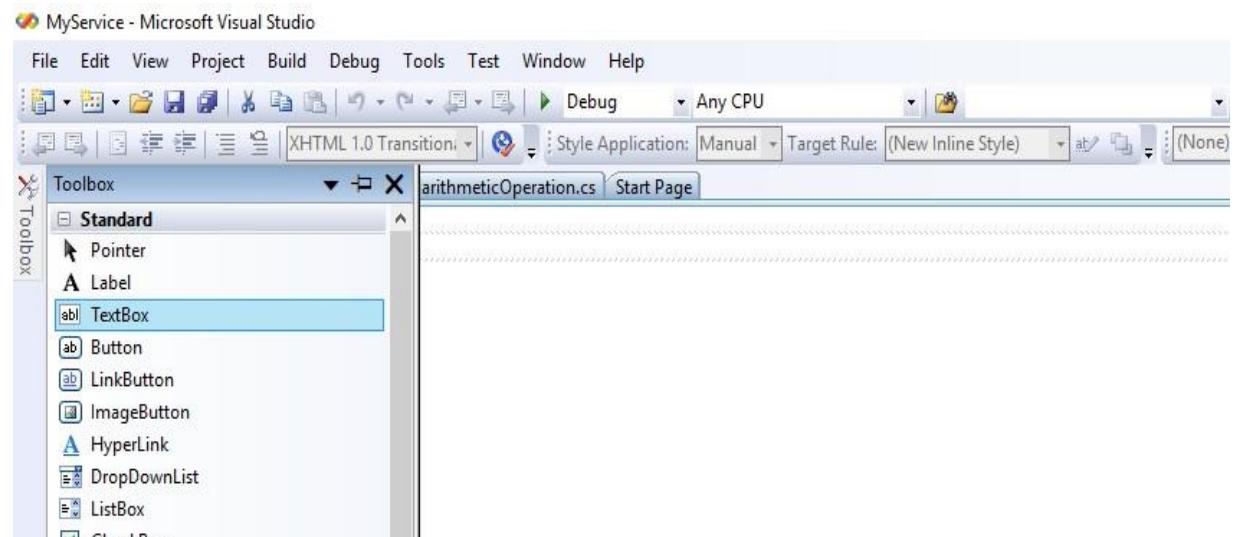
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
  <head runat="server">
    <title>Untitled Page</title>
  </head>
  <body>
    <form id="form1" runat="server">
      <div>
        </div>
      </form>
    </body>
  </html>
```

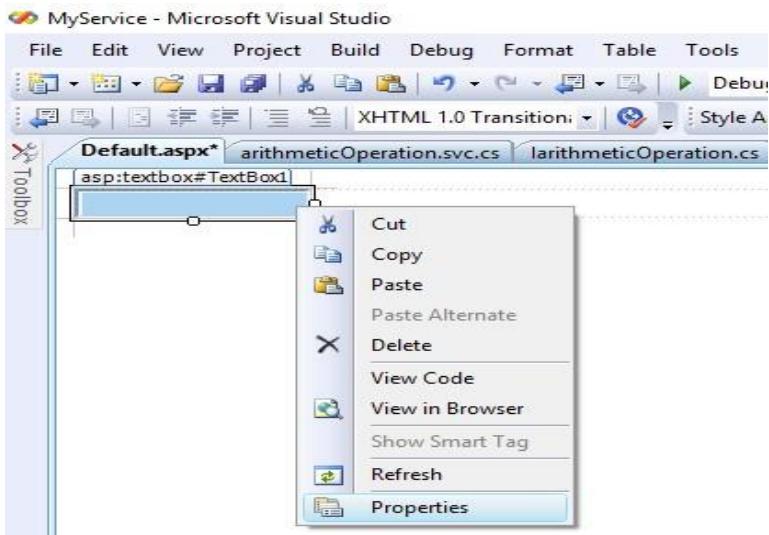
43. Now go to design mode by click on Design.



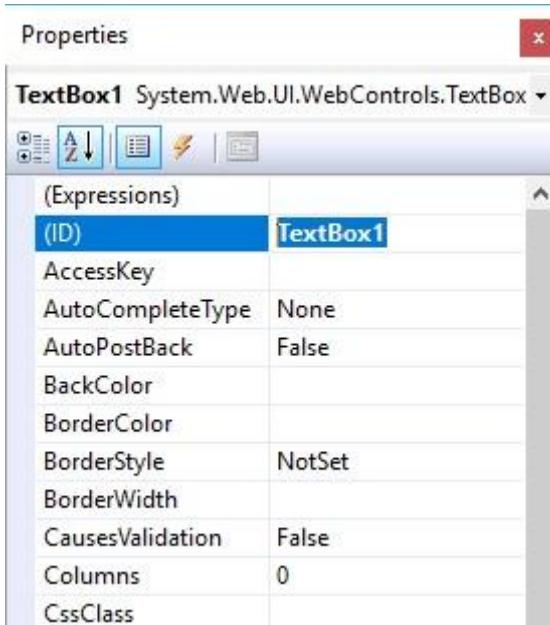
44. Now click on Toolbox on the top-left in above pic to add components. To add a textbox select TextBox from Toolbox and drag it into the screen that is appearing as blank.



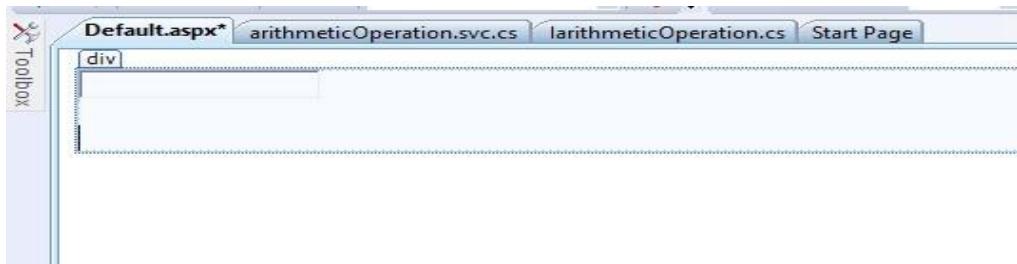
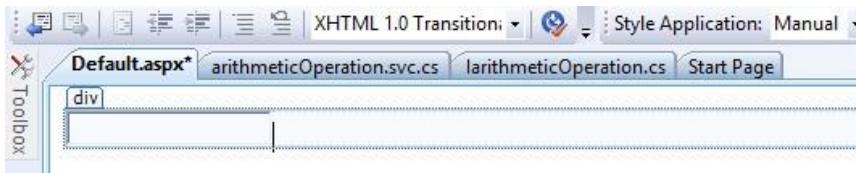
45. As you can see a TextBox is added to page. Right click on it and select Properties to set it's ID.



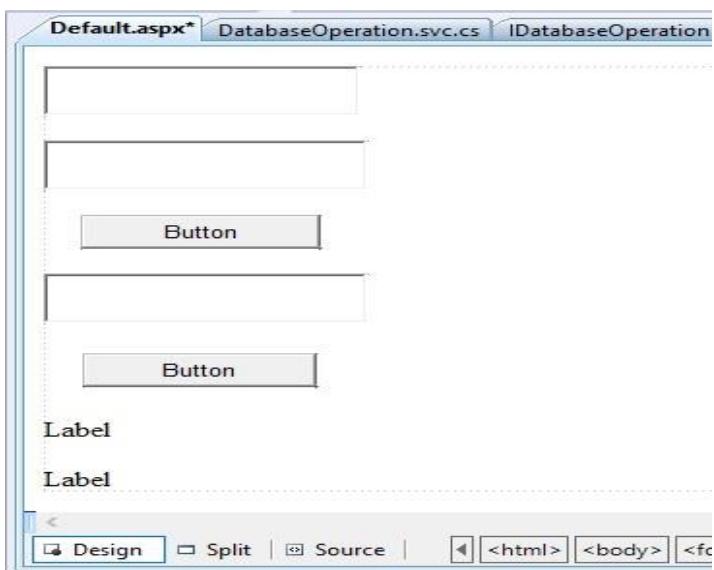
46. Property list of selected TextBox will open and you can see its ID is TextBox1. I'm going to leave it as it is. So its ID is TextBox1. You can change its ID, But accordingly it will change in C# program, where we will fetch the values of textboxes using its ID.



47. Now click after textbox and press two times enter key to create vertical space between two textboxes.



48. Using step 44 and 47 adding two more textboxes, Buttons, Labels. After that it will look like below.



49. Now changing the text of 1st and 2nd buttons.

**Right click on 1st button -> Properties -> Text = Insert Data
Right click on 2nd button -> Properties -> Text = Get Data**

Properties

Button1 System.Web.UI.WebControls.Button	
	A Z
SkinID	
TabIndex	0
Text	Insert Data
ToolTip	

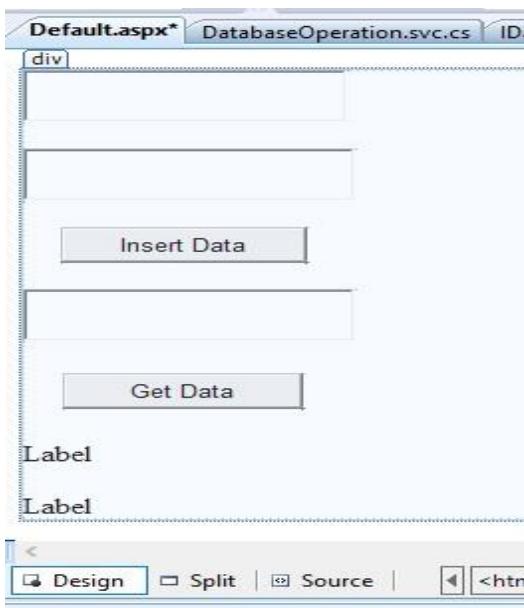
Text
The text to be shown on the button.

Properties

Button2 System.Web.UI.WebControls.Button	
	A Z
SkinID	
TabIndex	0
Text	Get Data
ToolTip	

Text
The text to be shown on the button.

Screen will look like below.



Default ID for each components will be following....

Textbox -> **TextBox1**

Textbox -> **TextBox2**

Button -> **Button1**

Textbox -> **TextBox3**

Button -> **Button2**

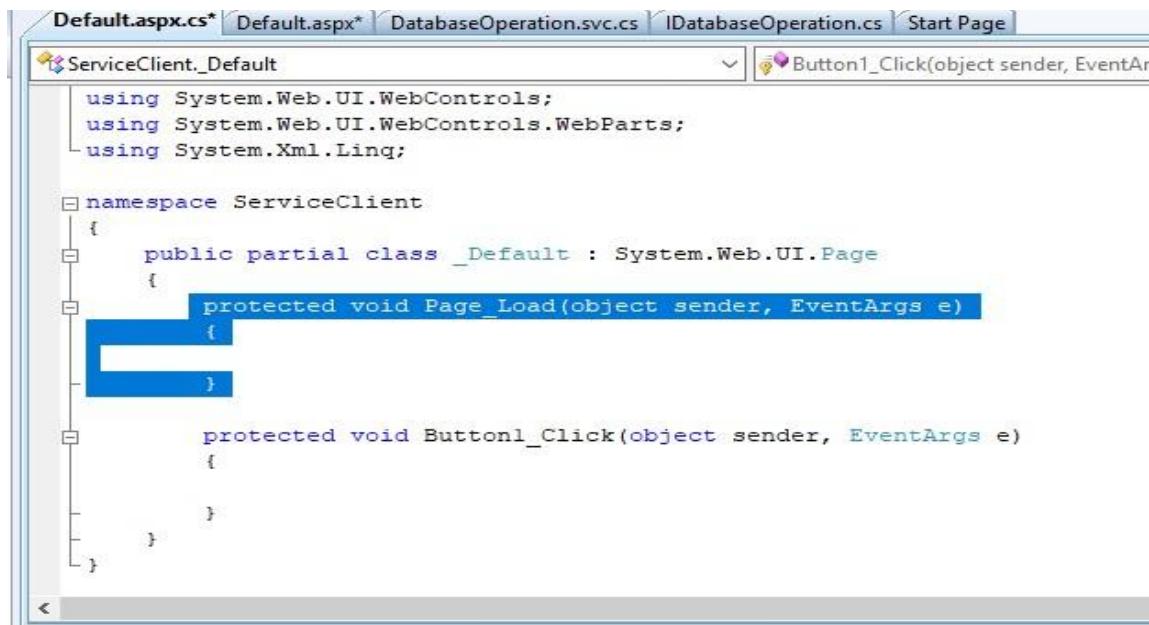
Label -> **Label1**

Label -> **Label2**

I'm not changing ID's of any component. I will use as it is.

50. **double click on First button** to add event. So that, we can insert data into database by click on this.

Double click on Insert Data button.



```
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Xml.Linq;

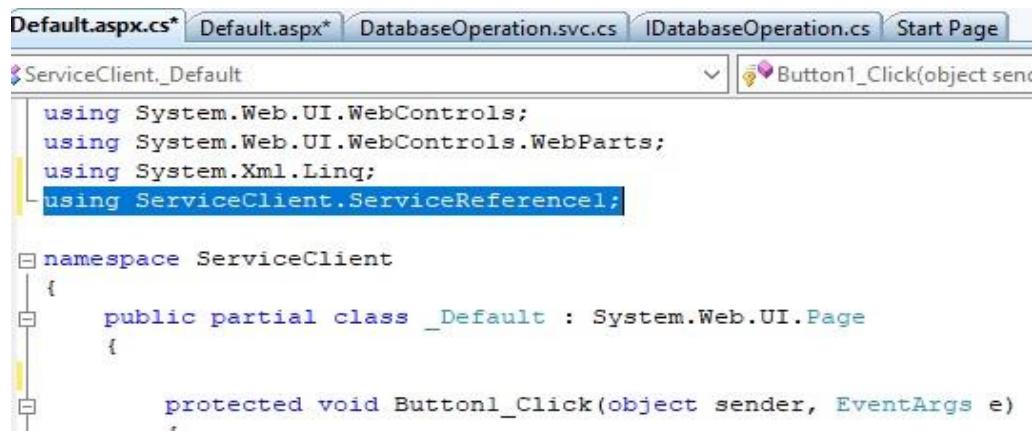
namespace ServiceClient
{
    public partial class _Default : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
        }

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

Delete the selected part. We do not need this.

51. **Import the following package.**

```
using ServiceClient.ServiceReference1;
```

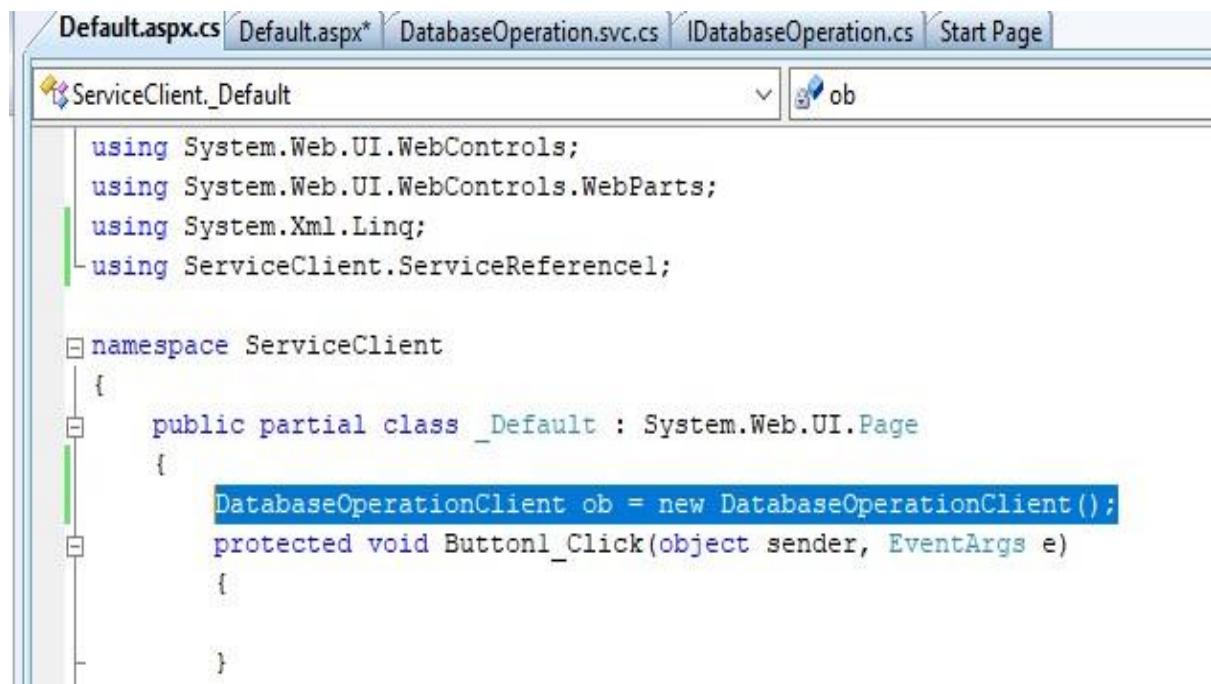


```
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Xml.Linq;
using ServiceClient.ServiceReference1;

namespace ServiceClient
{
    public partial class _Default : System.Web.UI.Page
    {
        protected void Button1_Click(object sender, EventArgs e)
        {
        }
    }
}
```

52. **Making the object of web service by adding the following line inside the class, but outside of method.**

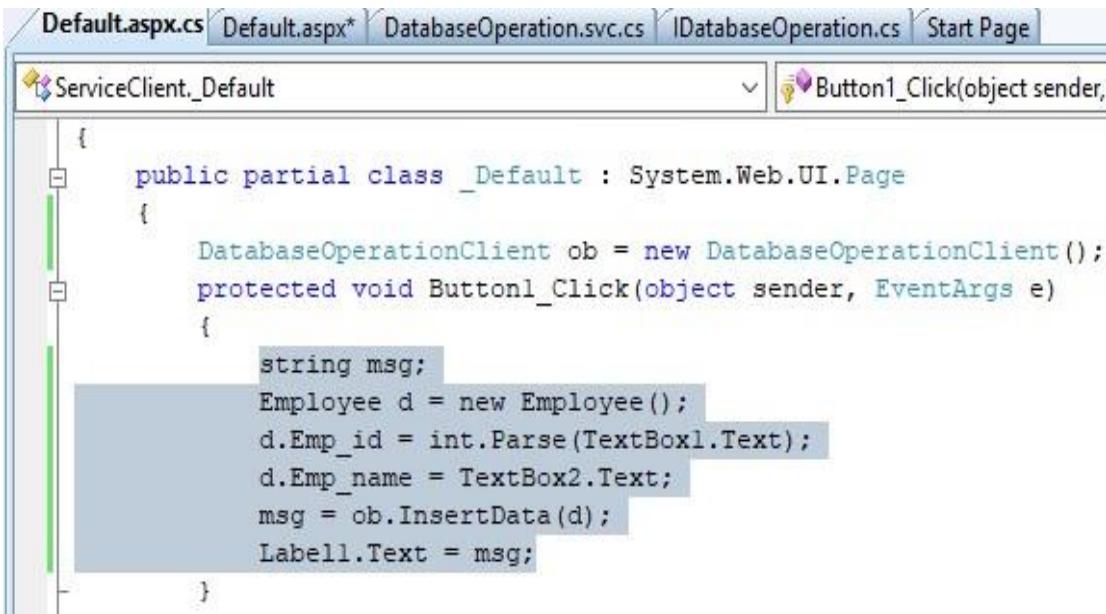
```
DatabaseOperationClient ob = new  
DatabaseOperationClient();
```



```
Default.aspx.cs Default.aspx* DatabaseOperation.svc.cs IDatabaseOperation.cs Start Page  
ServiceClient_Default ob  
using System.Web.UI.WebControls;  
using System.Web.UI.WebControls.WebParts;  
using System.Xml.Linq;  
using ServiceClient.ServiceReference1;  
  
namespace ServiceClient  
{  
    public partial class _Default : System.Web.UI.Page  
    {  
        DatabaseOperationClient ob = new DatabaseOperationClient();  
        protected void Button1_Click(object sender, EventArgs e)  
        {  
        }  
    }  
}
```

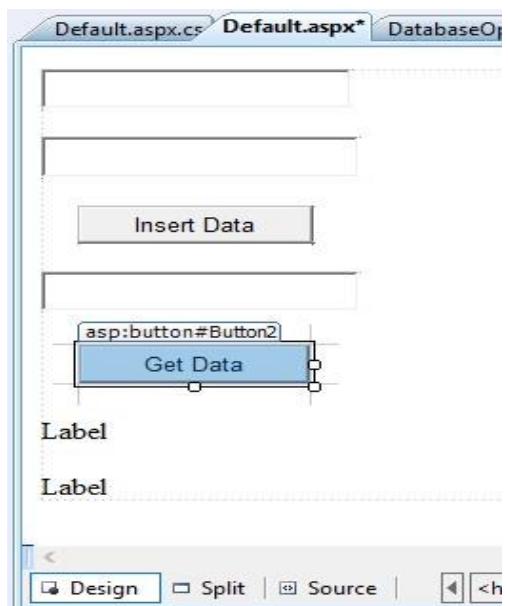
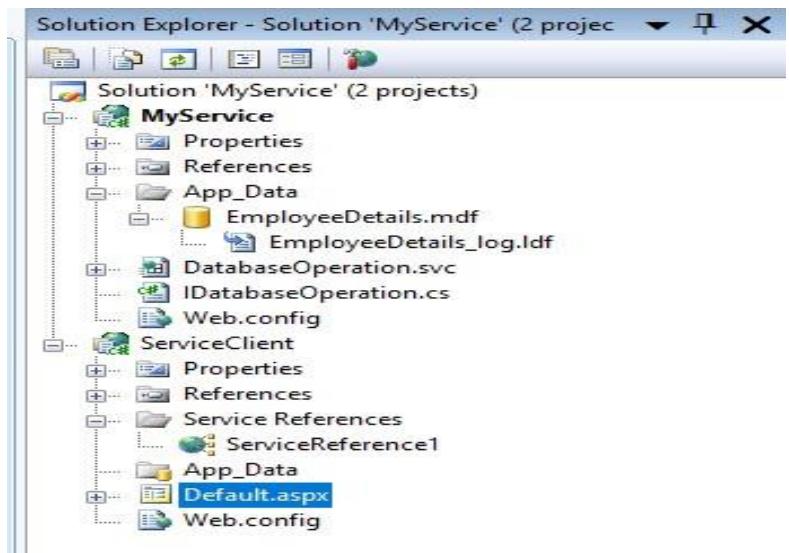
53. Now add the following code in first button event method and press **Ctrl+S**.

```
string msg;  
Employee d = new Employee();  
d.Emp_id = int.Parse(textBox1.Text);  
d.Emp_name = textBox2.Text;  
msg = ob.InsertData(d);  
label1.Text = msg;
```



```
Default.aspx.cs | Default.aspx* | DatabaseOperation.svc.cs | IDatabaseOperation.cs | Start Page
ServiceClient._Default
Button1_Click(object sender, EventArgs e)
{
    public partial class _Default : System.Web.UI.Page
    {
        DatabaseOperationClient ob = new DatabaseOperationClient();
        protected void Button1_Click(object sender, EventArgs e)
        {
            string msg;
            Employee d = new Employee();
            d.Emp_id = int.Parse(TextBox1.Text);
            d.Emp_name = TextBox2.Text;
            msg = ob.InsertData(d);
            Label1.Text = msg;
        }
    }
}
```

54. again Open Default.aspx file by double click on it. Go into Design mode and double click on Get Data button so that we can add event for that also.



```
Default.aspx.cs* Default.aspx* DatabaseOperation.svc.cs IDatabaseOperation.cs Start Page
ServiceClient._Default
using ServiceClient.ServiceReference1;

namespace ServiceClient
{
    public partial class _Default : System.Web.UI.Page
    {
        DatabaseOperationClient ob = new DatabaseOperationClient();
        protected void Button1_Click(object sender, EventArgs e)
        {
            string msg;
            Employee d = new Employee();
            d.Emp_id = int.Parse(textBox1.Text);
            d.Emp_name = textBox2.Text;
            msg = ob.InsertData(d);
            Label1.Text = msg;
        }

        protected void Button2_Click(object sender, EventArgs e)
        {
        }
    }
}
```

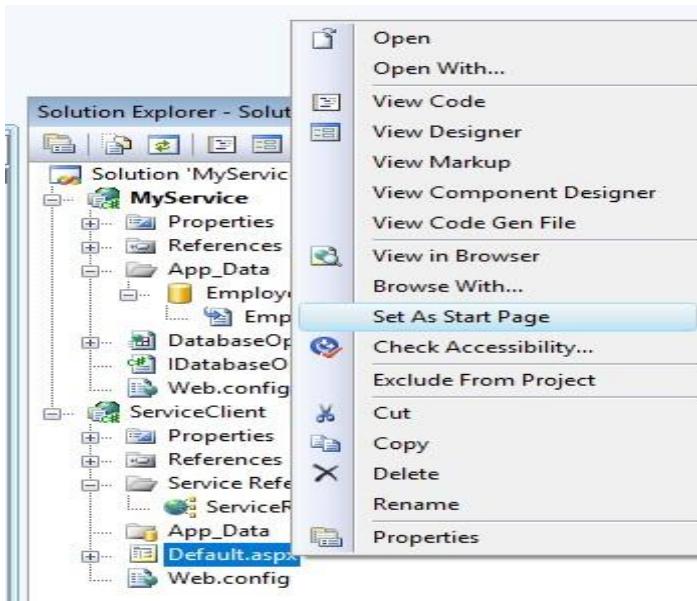
55. Now add the following code, where the curser is indicating in above pic to add event for second button. After that press **Ctrl+S**.

```
Employee data = new Employee();
int id = int.Parse(textBox3.Text);
data = ob.GetData(id);
    Label1.Text = (data.Emp_id).ToString();
    Label2.Text = data.Emp_name;
```

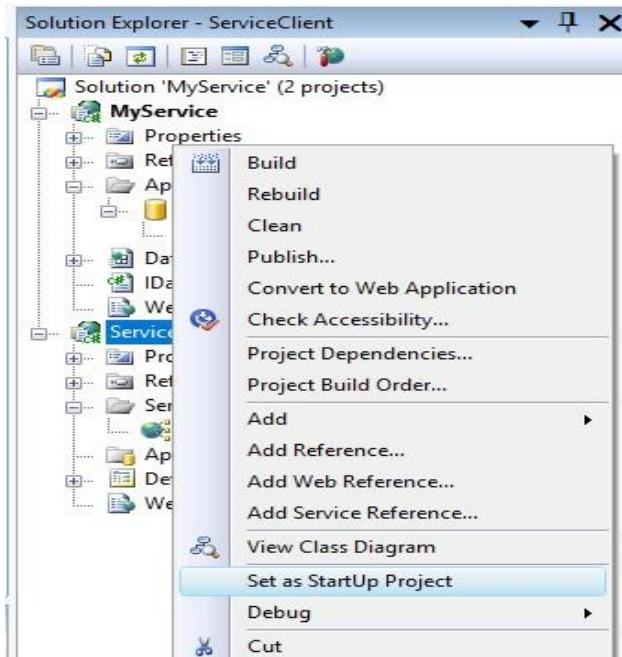
```
DatabaseOperationClient ob = new DatabaseOperationClient();
protected void Button1_Click(object sender, EventArgs e)
{
    string msg;
    Employee d = new Employee();
    d.Emp_id = int.Parse(textBox1.Text);
    d.Emp_name = textBox2.Text;
    msg = ob.InsertData(d);
    Label1.Text = msg;
}

protected void Button2_Click(object sender, EventArgs e)
{
    Employee data = new Employee();
    int id = int.Parse(textBox3.Text);
    data = ob.GetData(id);
    Label1.Text = (data.Emp_id).ToString();
    Label2.Text = data.Emp_name;
}
```

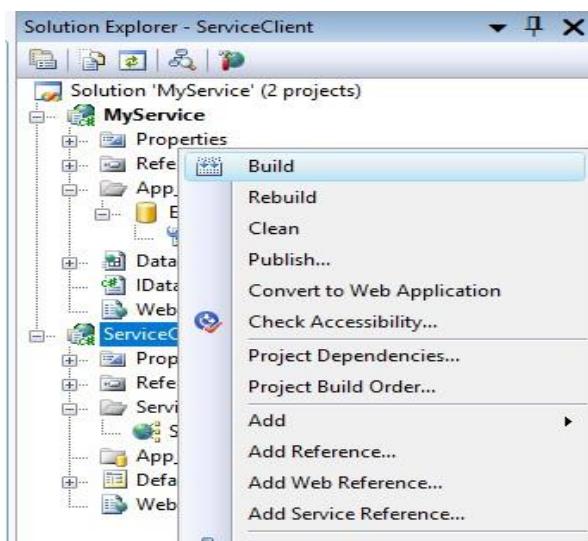
56. right click on Default.aspx file of ServiceClient, and set it as start page.



57. Now right click on ServiceClient and set it as startup project.



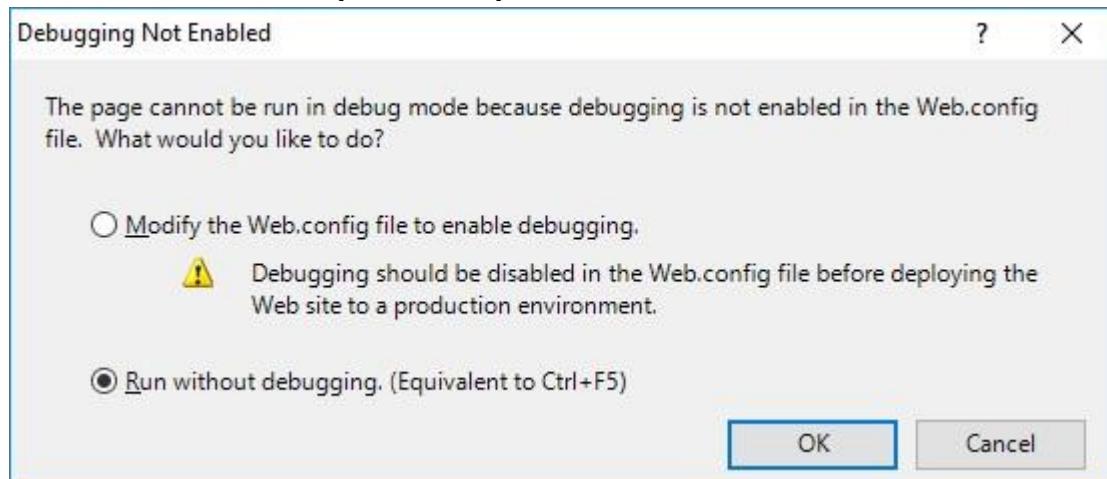
58. Now Build the ServiceClient project.



59. Now run the ServiceClient project by pressing Run button in toolbar.



60. Select second option and press OK.



61. Following window will open in browser.

A screenshot of a web browser window titled "Untitled Page". The address bar shows "localhost:64865/Default.aspx". Below the address bar is a toolbar with icons for back, forward, and search. A message says "For quick access, place your bookmarks here on the bar". There are three empty text input fields stacked vertically. Below them is an "Insert Data" button. Underneath the text input fields is a "Get Data" button. To the left of the "Get Data" button are two "Label" elements, one above the other.

62. First we will do Get data Operation.

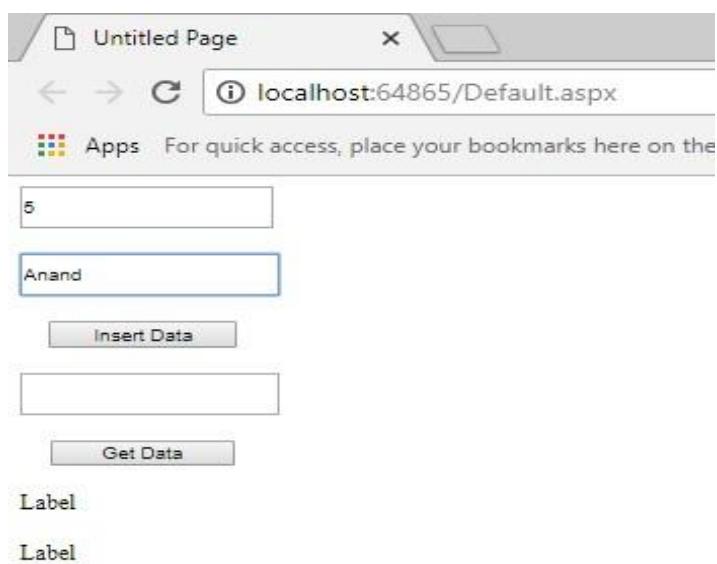
So I will use 3rd textbox to enter ID.

After giving ID = 1 and pressing Get Data button.

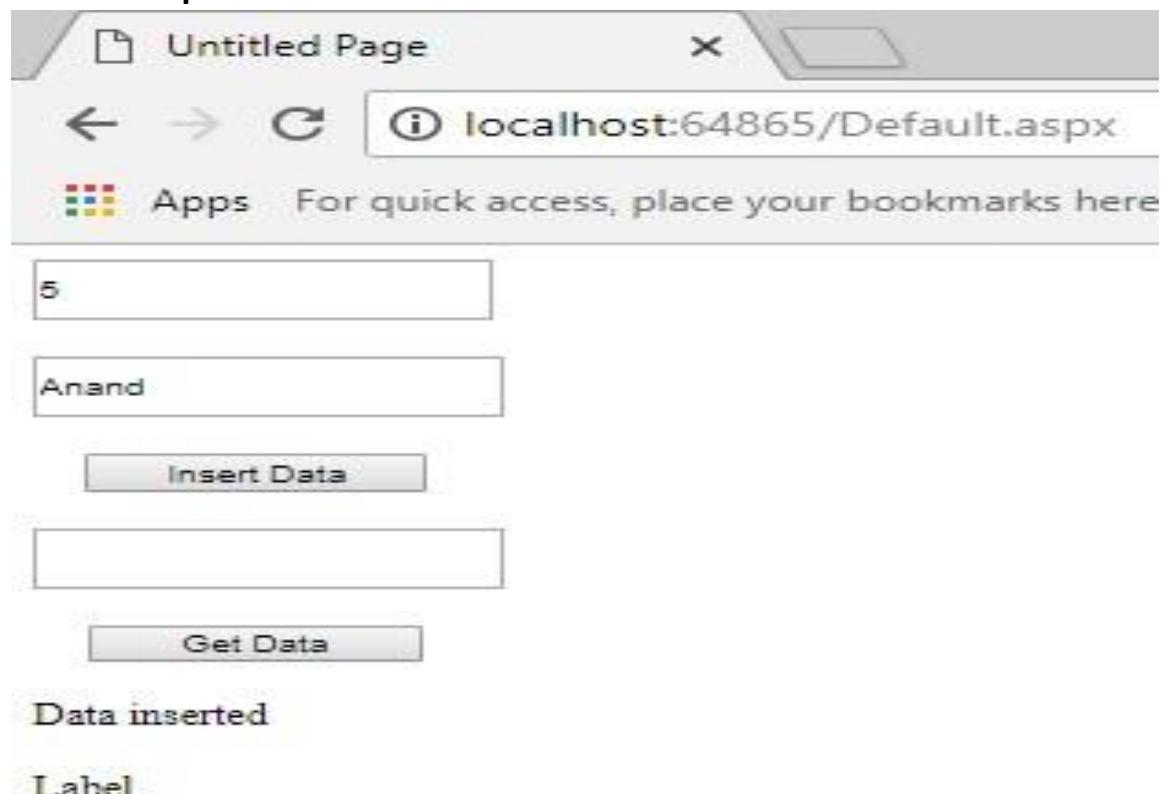
A screenshot of a web browser window titled "Untitled Page". The address bar shows "localhost:64865/Default.aspx". Below the address bar is a toolbar with icons for back, forward, and search. A message says "For quick access, place your bookmarks here on the bar". There are three text input fields. The first field contains "1". Below the text input fields is an "Insert Data" button. Underneath the text input fields is a "Get Data" button. To the left of the "Get Data" button is a "Label" element. The output below the "Label" element shows the value "1" followed by "Devendra".

I got the detail of ID=1 that I have inserted record in step 20.

63. Now again run the ServiceClient project for InsertData operation.



After Insert Data button press.



As it is showing that data is inserted. So let's check it in database table.

MyService (Running) - Microsoft Visual Studio

File Edit View Project Build Debug Data Query Designer Tools Test Window Help

Change Type | SQL | Default.aspx | DatabaseOperation.svc.cs

Server Explorer

Data Connections

- EmployeeDetails.mdf
 - Database Diagrams
 - Tables
 - Employee
 - Views
 - Stored Procedures
 - Functions
 - Synonyms
 - Types
 - Assemblies

Servers

- DESKTOP-RBGTAGB

Employee: Query...OYEEDETAILS.MDF

	Emp_id	Emp_name
1	Devendra	
2	Umendra	
3	Shivendra	
5	Anand	
*	NULL	NULL

If we will compare Employee table data from step 20 data, we can see that a new data Anand with ID=5 is added in Employee table.

Practical-8 (Without Database)

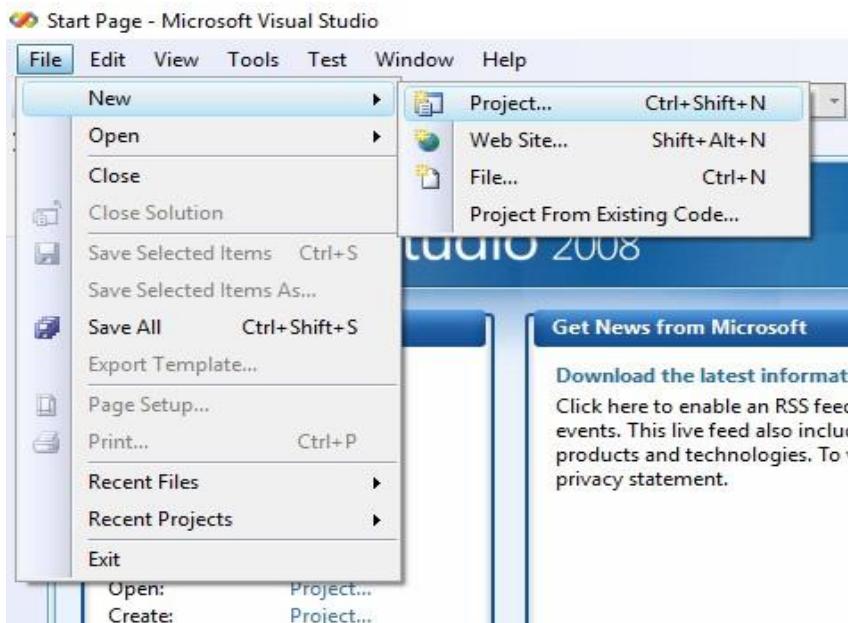
Aim: Implement a typical service and a typical client using WCF.

Requirements:

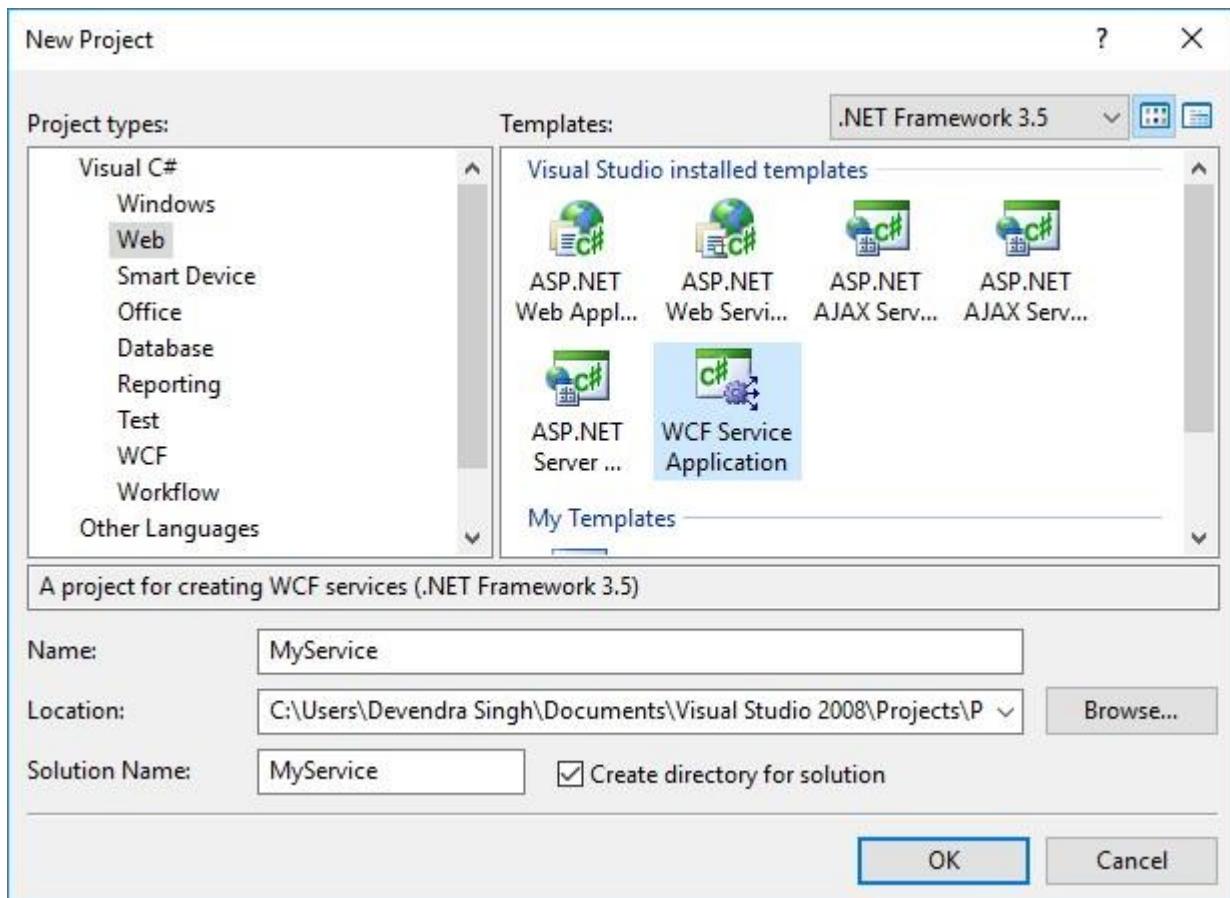
Visual Studio 2008 Professional.

Note: For practical 8, 9 & 10 Visual Studio is mandatory. Because we can implement WCF Service only on Visual Studio.

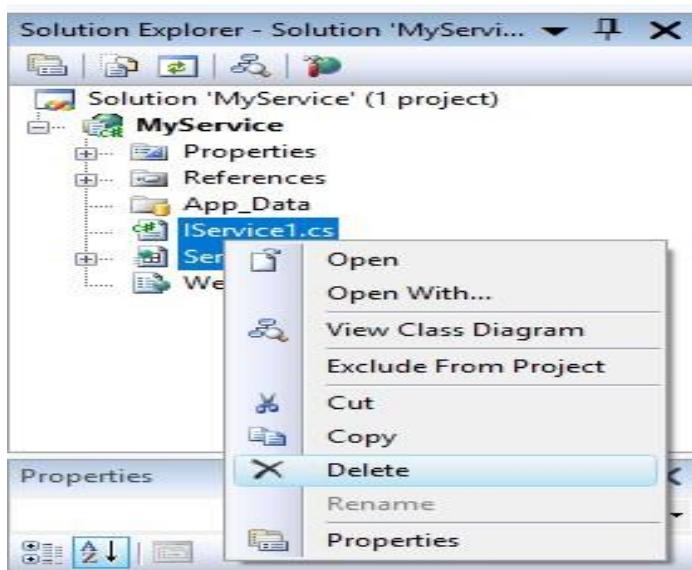
1. Open Visual Studio and open Project.



2. In Project Types, select Web and in Templates select WCF Service Application. Give it name as MyService and click on OK button.

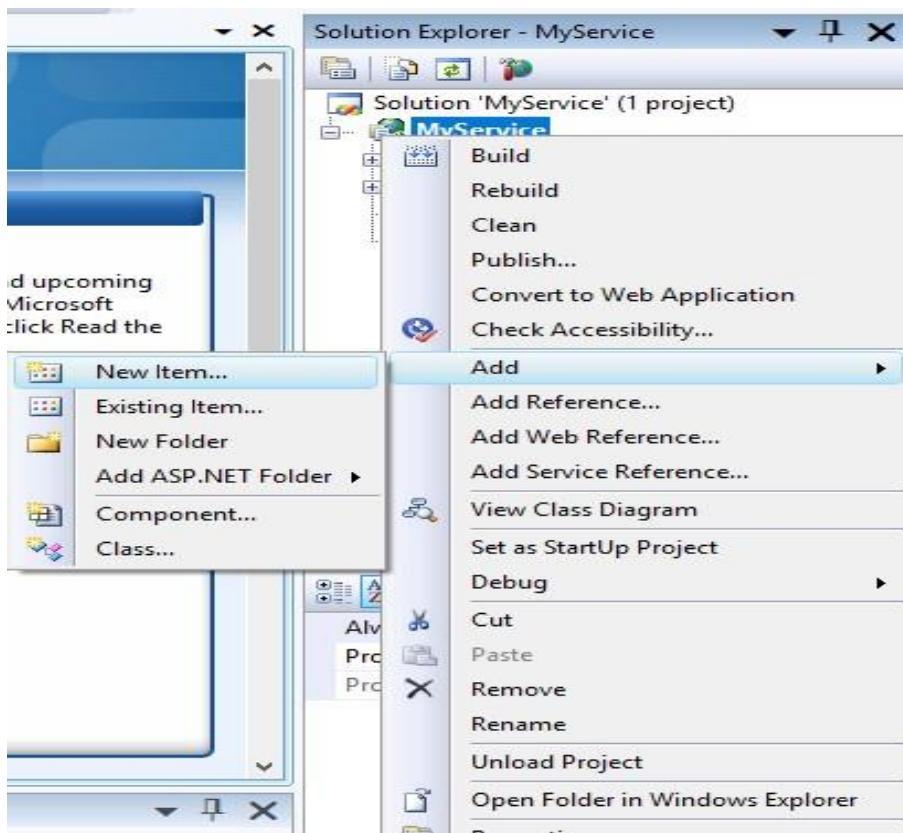


3. **Delete the selected two files** from Solution Explorer placed on right hand side. Actually we are going to delete these files, because this is default WCF Service file and we want to create our own WCF Web Service.

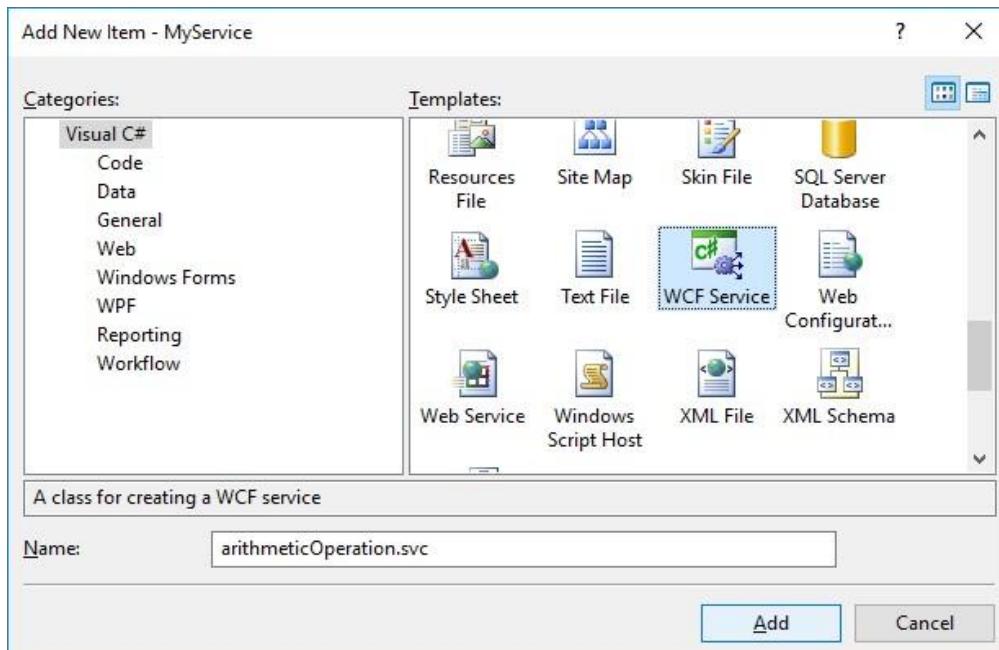


4. Now to create our own WCF web service.

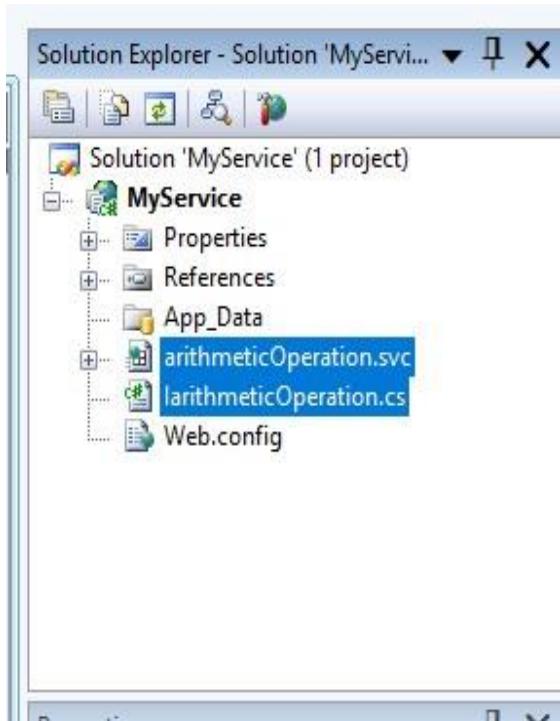
Right click on MyService -> Add -> New Item



5. Select WCF Service in Templates. Give it name as arithmeticOperation and click on Add button.



6. Now open the both selected files by clicking on both one by one alternatively.



7. Now open **arithmetricOperation.cs** file. It will look like below.

The screenshot shows the code editor with the file 'arithmetricOperation.cs' open. The title bar indicates the file name. On the left, there is a 'Toolbox' panel. The code itself starts with 'using' statements for System, System.Collections.Generic, System.Linq, System.Runtime.Serialization, System.ServiceModel, and System.Text. It then defines a namespace 'MyService' containing a service contract 'IarithmeticOperation'. The contract has a single method 'DoWork()' decorated with the [OperationContract] attribute.

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

namespace MyService
{
    // NOTE: If you change the interface name "Iarithmet
    [ServiceContract]
    public interface IarithmeticOperation
    {
        [OperationContract]
        void DoWork();
    }
}
```

8. Now replace the void **DoWork()** interface by following two new interfaces.

After that press **Ctrl+S** to save the changes.

```

        double Sum(double a, double b);
double Multi(double a, double b);

```

```

arithmeticOperation.svc.cs IarithmeticOperation.cs Start Page
MyService.IarithmeticOperation
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.Text;

namespace MyService
{
    // NOTE: If you change the interface name "Iarit
    [ServiceContract]
    public interface IarithmeticOperation
    {
        [OperationContract]
        double Sum(double a, double b);
        double Multi(double a, double b);
    }
}

```

You can add as many more interfaces as many WCF service methods you want. After that you will use these interfaces in arithmeticOperation.svc.cs file to define the Service methods. In this WCF service I'm creating only two methods that can be used by client.

9. Now open arithmeticOperation.svc.cs page. It will look like below.

```

arithmeticOperation.svc.cs IarithmeticOperation.cs Start Page
MyService.arithmeticOperation
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.Text;

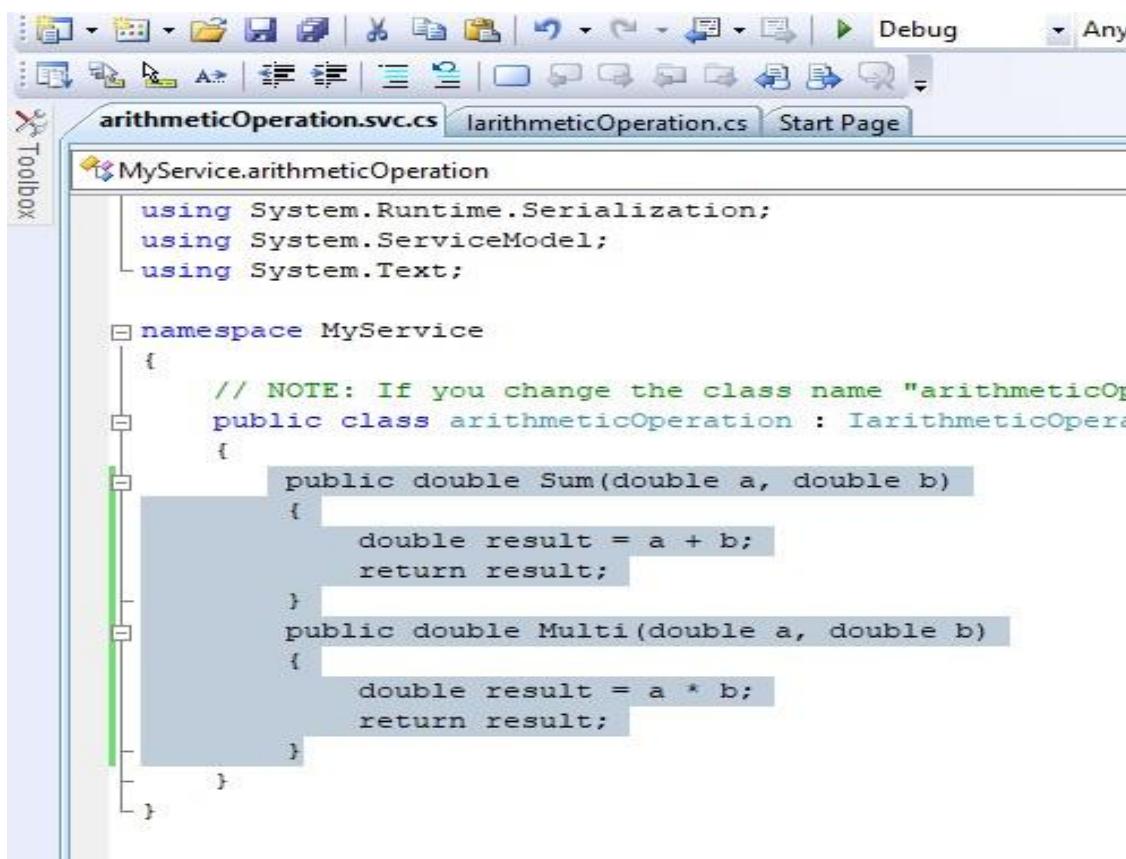
namespace MyService
{
    // NOTE: If you change the class name "arithmeticOperation" h
    public class arithmeticOperation : IarithmeticOperation
    {
        public void DoWork()
        {
        }
    }
}

```

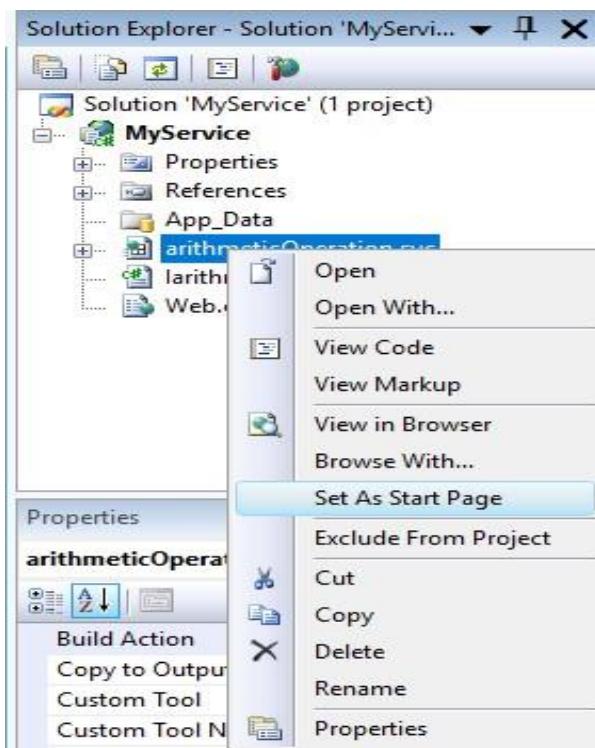
10. As we can see it contains the DoWork() method already, that interface we have deleted in previous file and we replaced this by two new interfaces. So we will

use those two interfaces as method in this file. After add following code press **Ctrl+S** to save the changes.

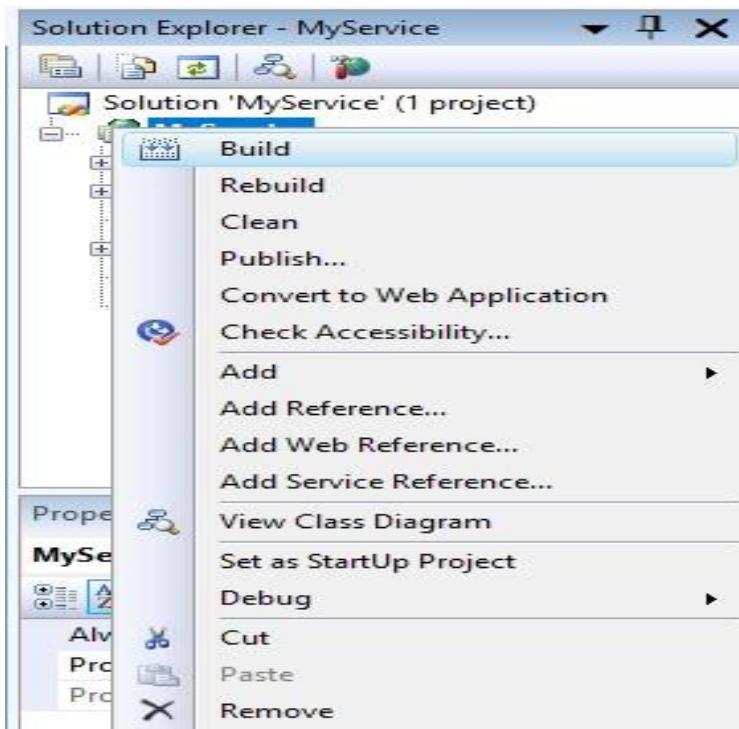
```
public double Sum(double a, double b)
{
    double result = a + b;
return result;
}
public double Multi(double a, double b)
{
    double result = a * b;
return result;
}
```



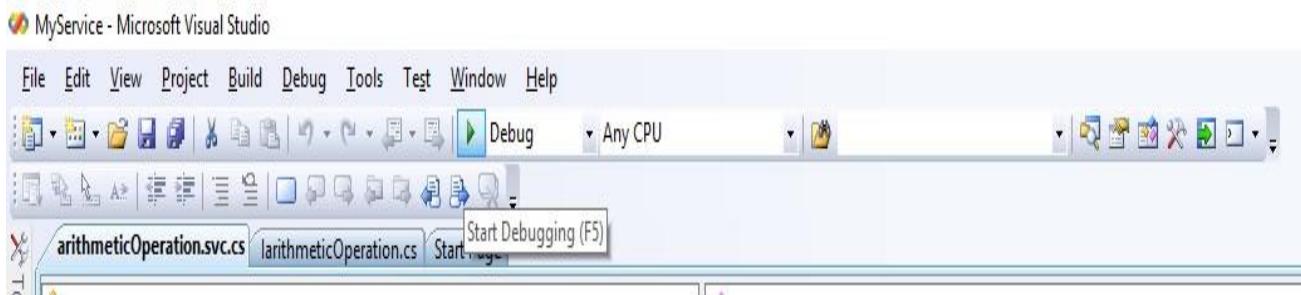
11. Now right click on **arithmeticOperation.svc** file name in **Solution Explorer** and click on **Set As Start Page**.



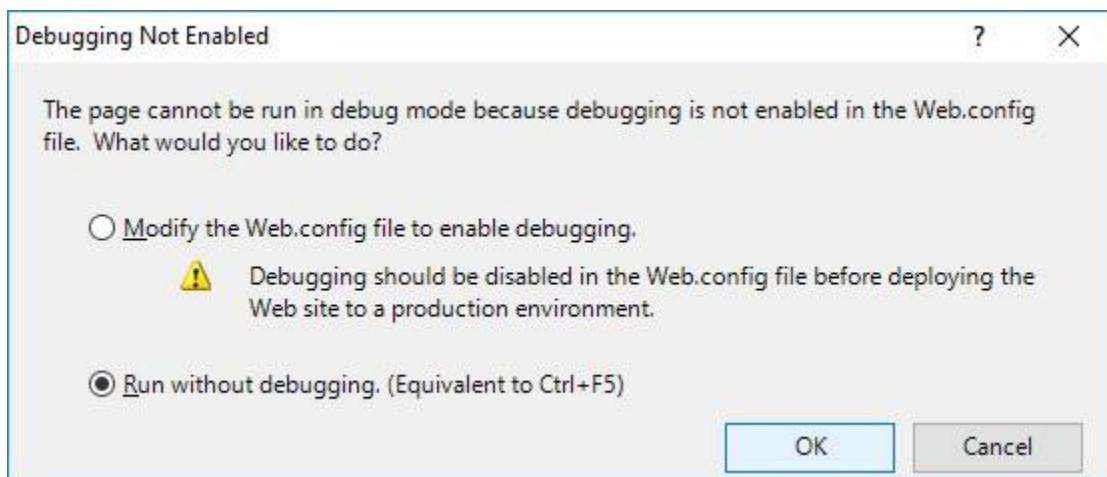
12. Now right click on **MyService** in Solution Explorer and **select Build** to build the project.



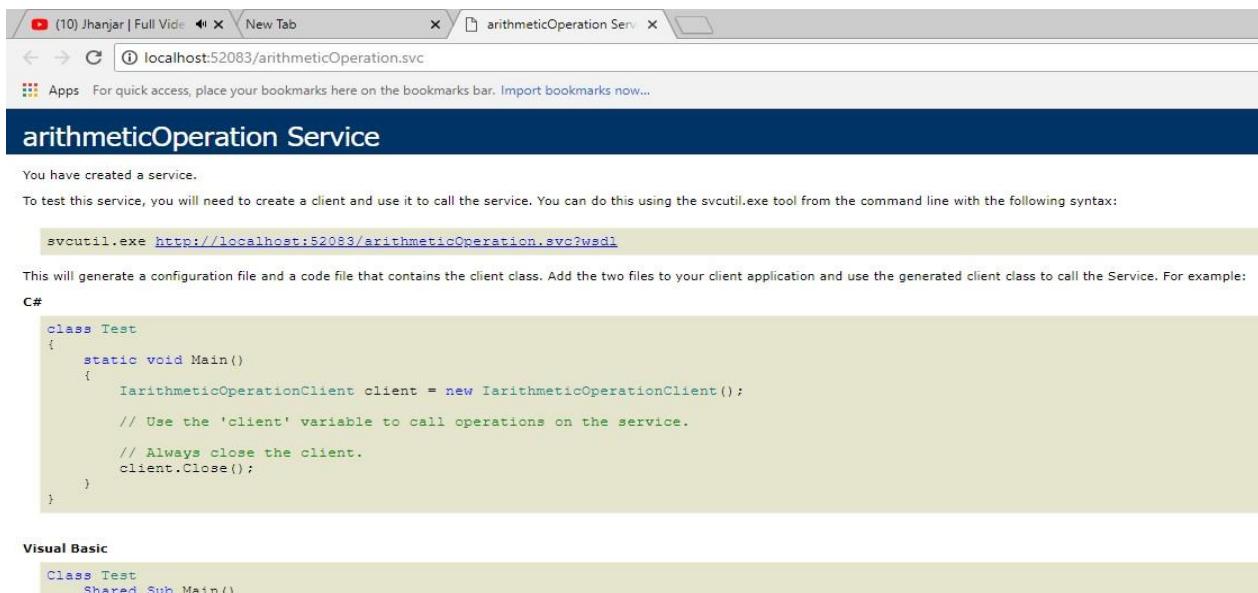
13. Now **click on run button** to run the project.



14. If any window appears like below then **select Run without debugging** and **click on OK button**.



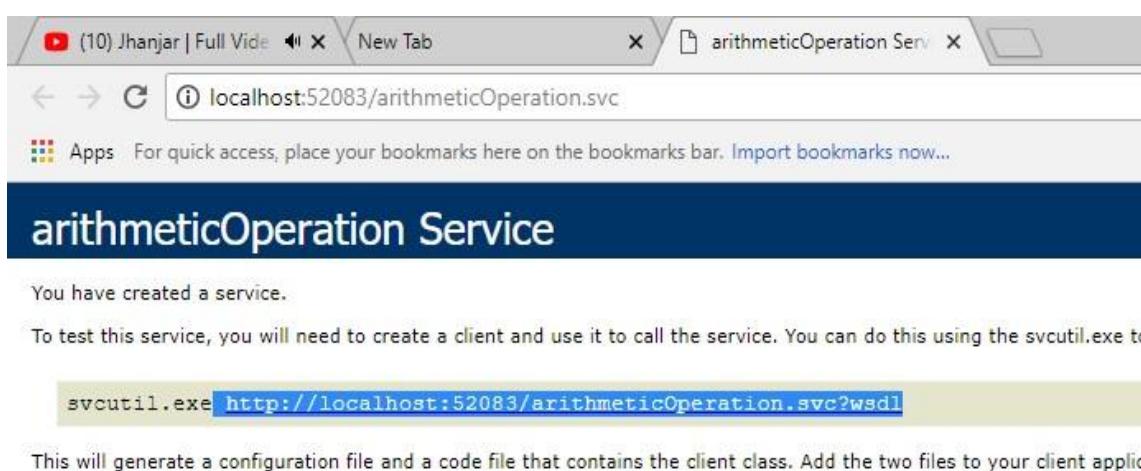
15. A new window will open in browser like below.



- 16. So we have successfully created a WCF Service and it is running on inbuilt server of visual studio. You can see it in taskbar.**

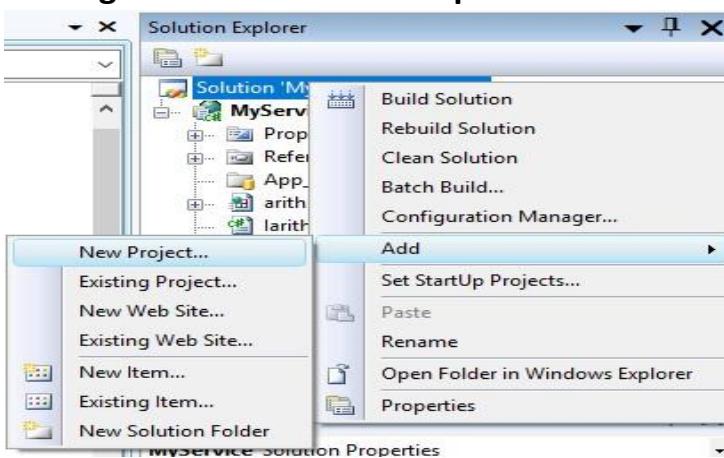


- 17. Now copy the selected link to consume this service on client side.**



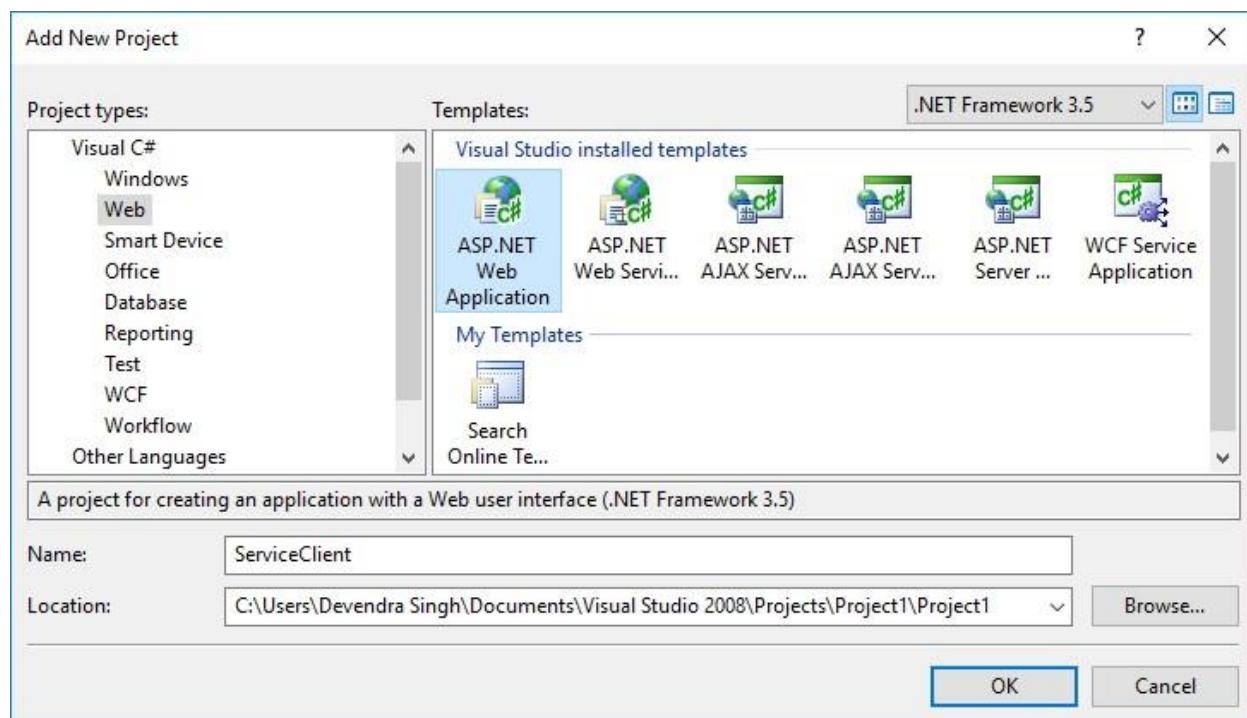
- 18. Now take care, you can close the browser. But don't close the Solution Explorer or Visual Studio, otherwise inbuilt server will stop and you will get error during consuming this service. Even, you have to create the client within the same Solution Explorer. To solve this problem we can host this service on IIS. But it will take more steps and for our practical we don't need this.**

- 19. Now right click on solution explorer and add a new project to create client.**

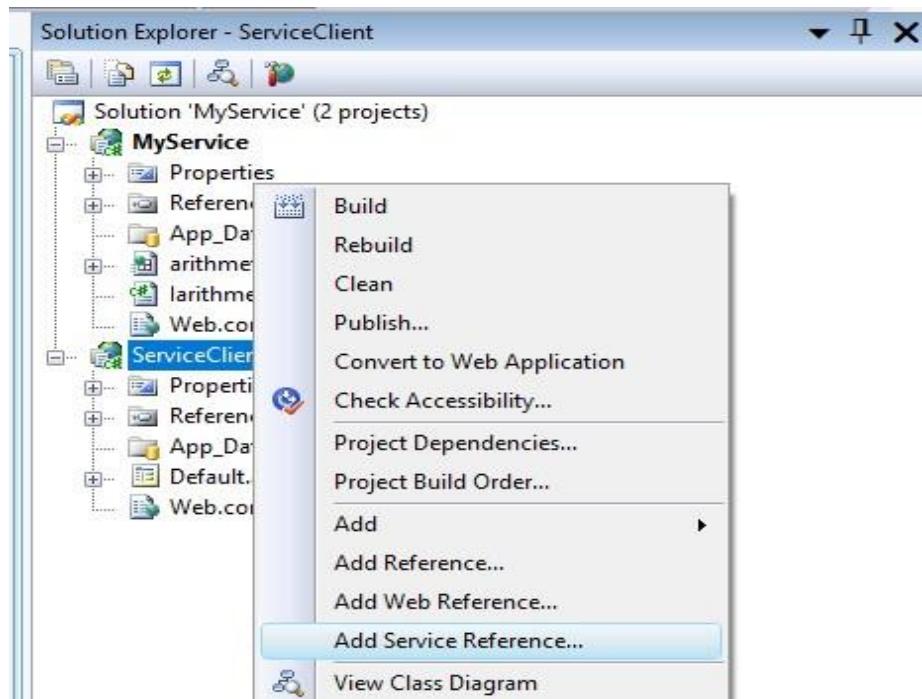


20. In Project Type select Web and in Templates select ASP.Net Web Application.

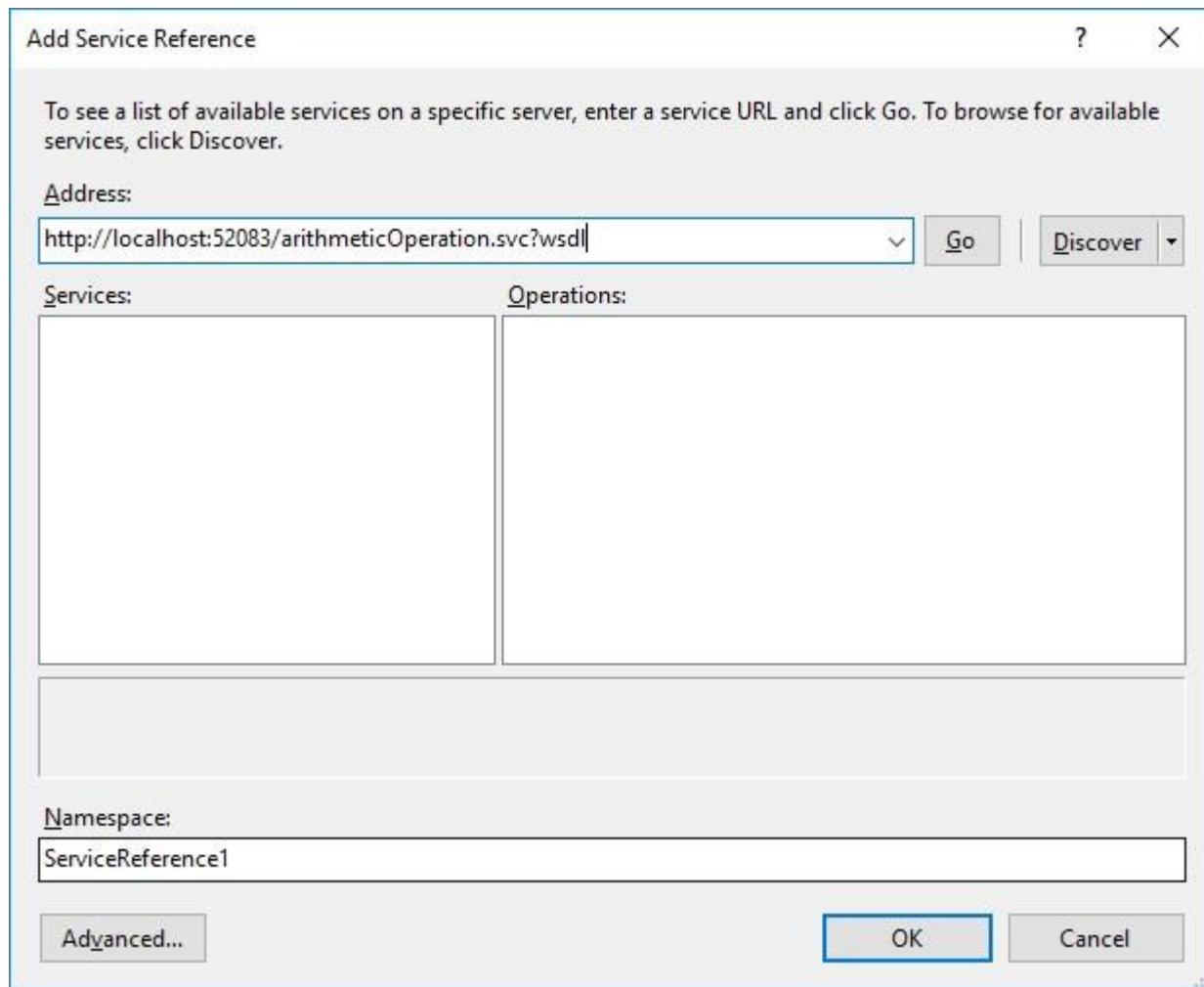
Give it name as ServiceClient and then click on OK button.



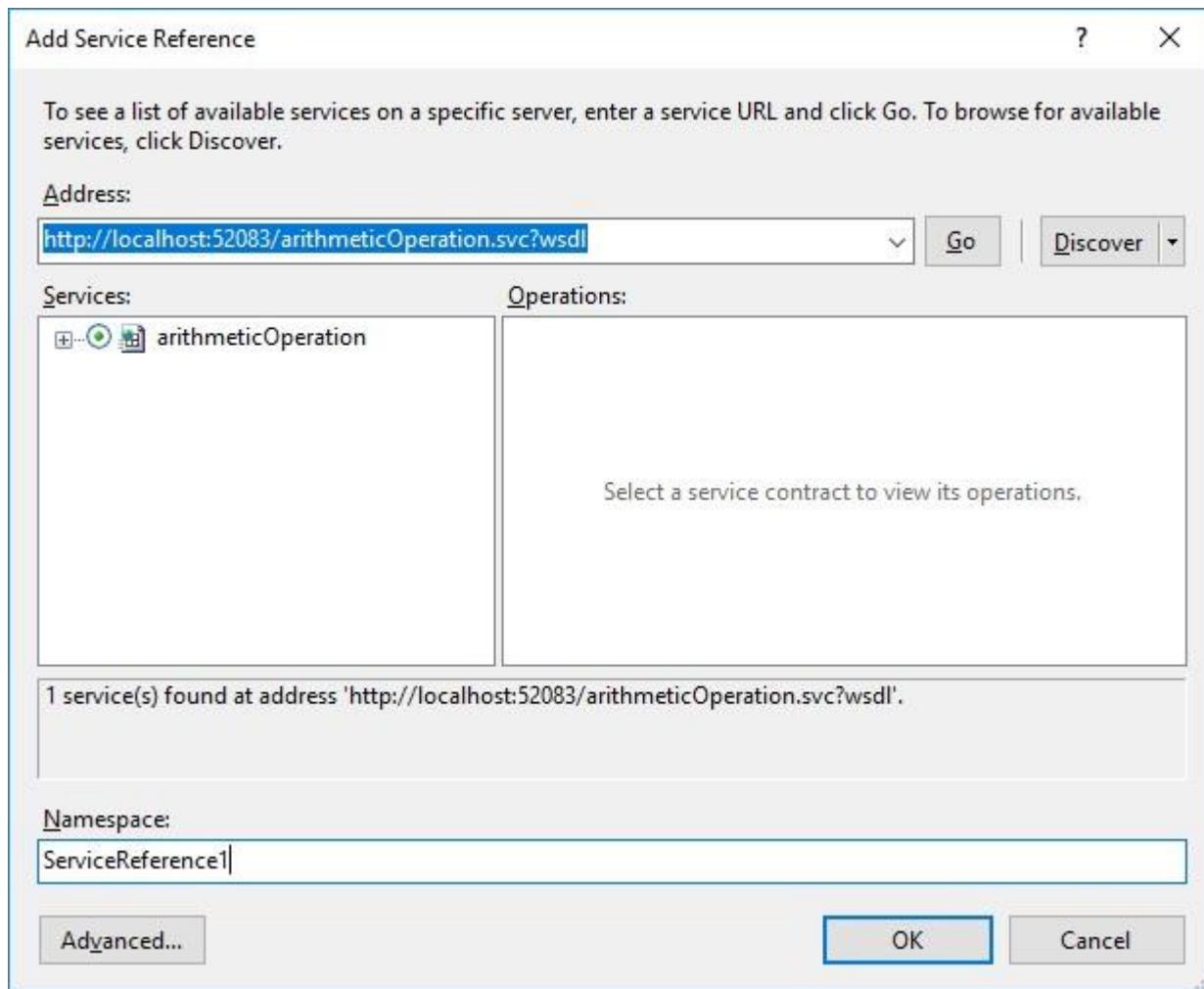
21. Right click on Service ServiceClient in Solution Explorer and select Add Service Reference.



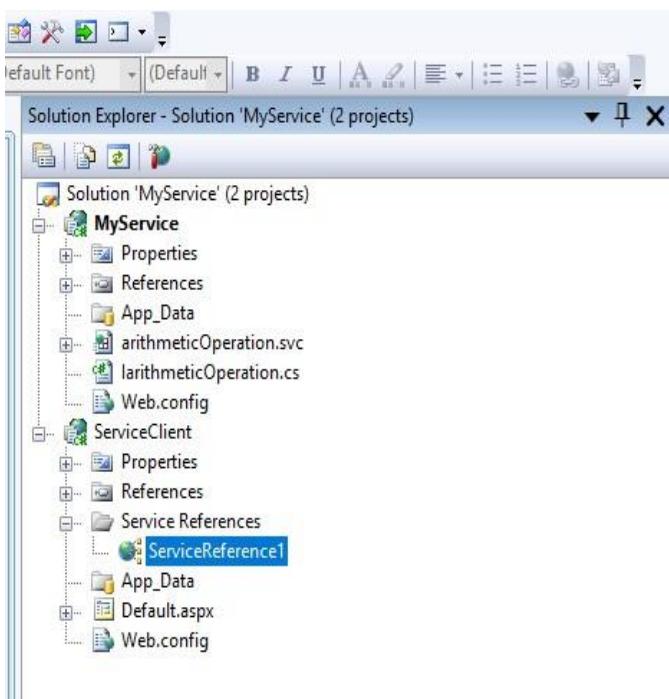
22. Paste the copied link into Address and click on Go button.



23. As you can see arithmeticOperation Service is fetched. Now I'm leaving **Namespace as it is i.e. ServiceReference1**. Actually this namespace will be used to call the Service methods. If you will change it then you have to use the according namespace to call the service methods. **Click on OK button.**



24. As you can see Service Reference is added with Namespace in Solution Explorer.



**25. Default.aspx page of ServiceClient will appear by default in Source mode.
Otherwise you can open it from solution explorer.**

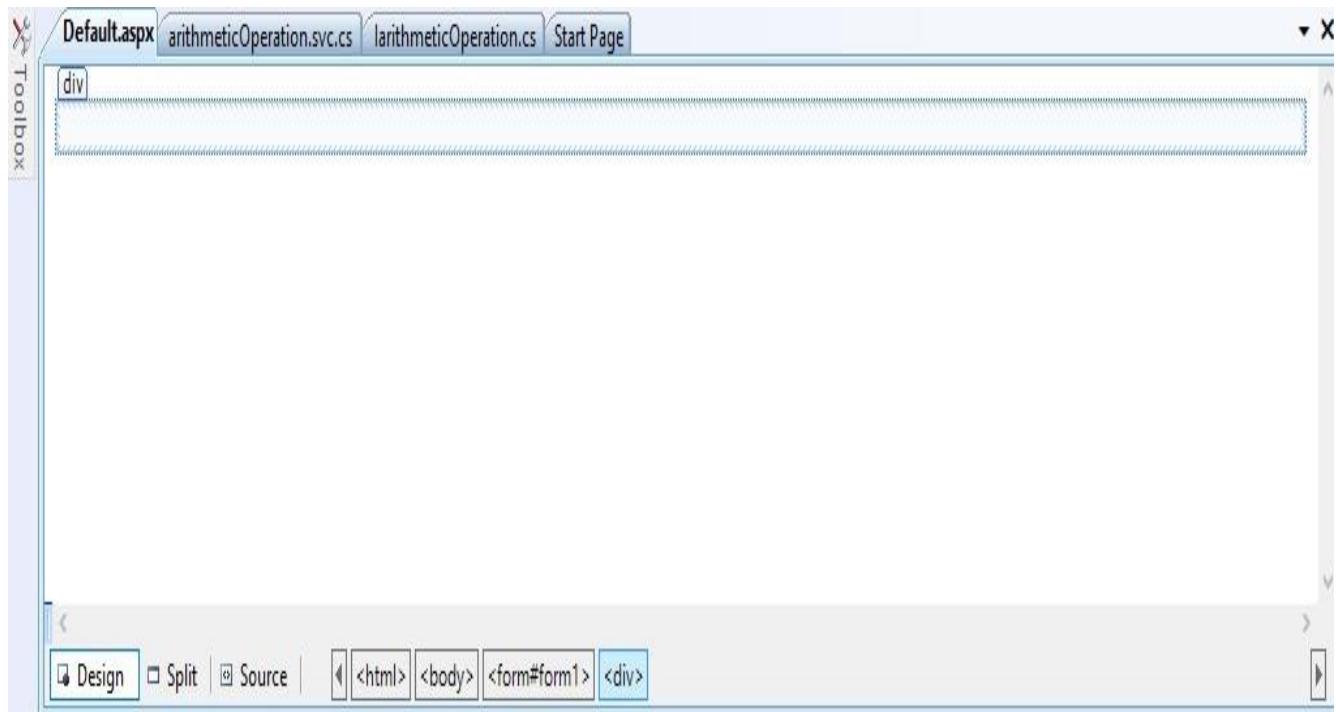
The screenshot shows the Microsoft Visual Studio code editor with the "Default.aspx" tab active. The code editor displays the ASPX markup and the corresponding C# code behind. The code includes the declaration of the Default.aspx page as a ServiceClient page, the DOCTYPE declaration, and the HTML structure with a form and a div.

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs" Inherits="ServiceClient._Default" %>

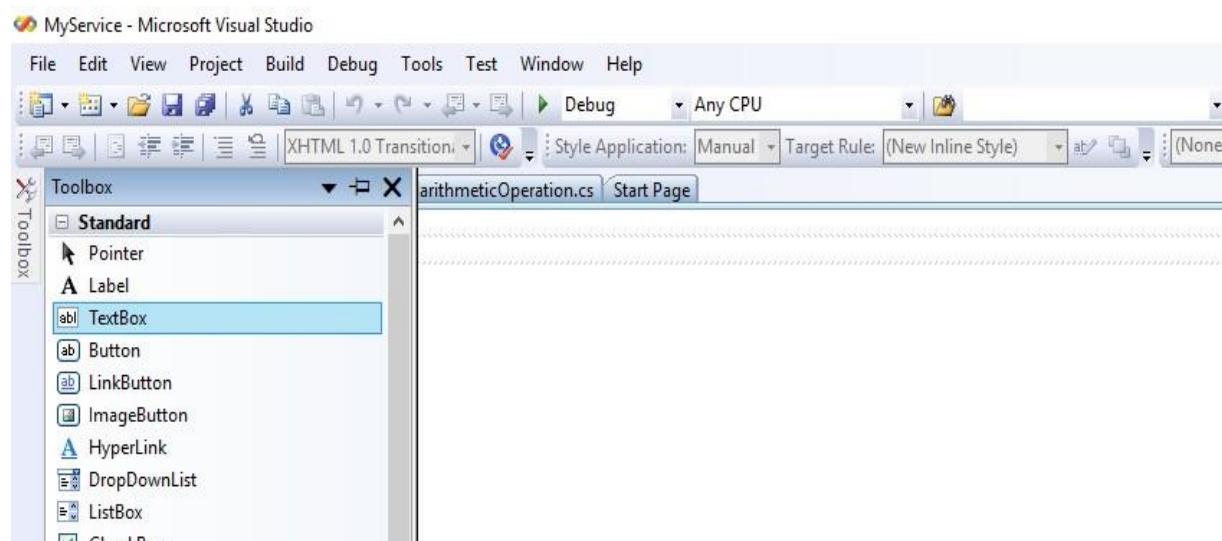
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Untitled Page</title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
        </div>
    </form>
</body>
</html>
```

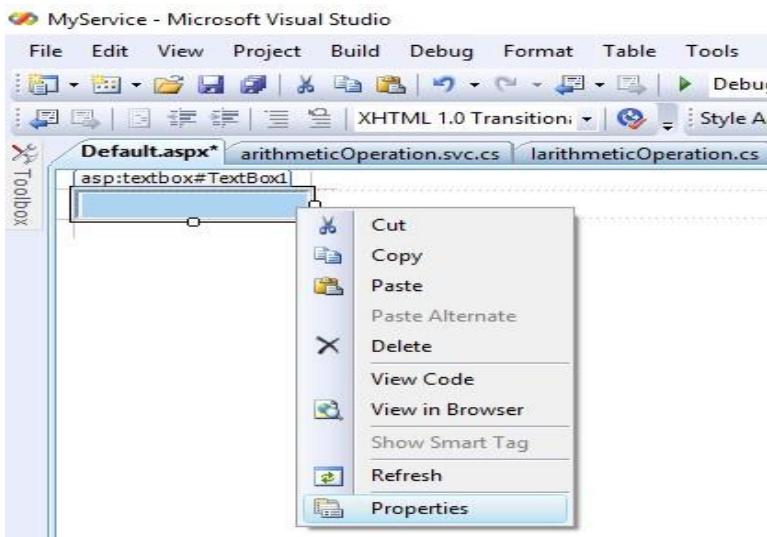
26. Now go to design mode by click on Design.



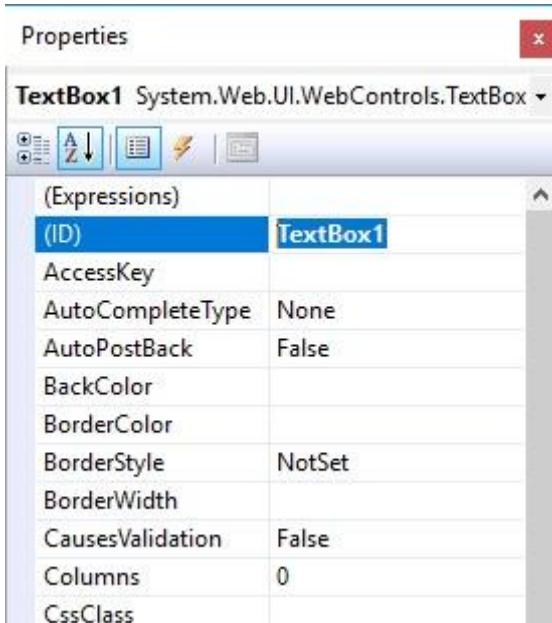
27. Now **click on Toolbox** on the top-left in above pic to add components. To add a textbox **select TextBox** from Toolbox and **drag it into the screen** that is appearing as blank.



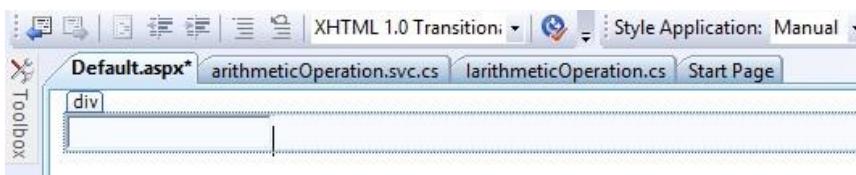
28. As you can see a TextBox is added to page. **Right click on it and select Properties to set it's ID.**

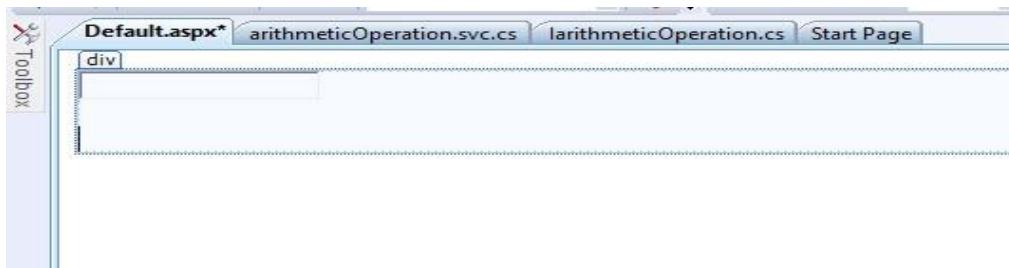


- 29.** Property list of selected TextBox will open and you can see its ID is TextBox1.
I'm going to leave it as it is. So its ID is TextBox1. You can change its ID, But accordingly it will change in C# program, where we will fetch the values of textboxes using its ID.

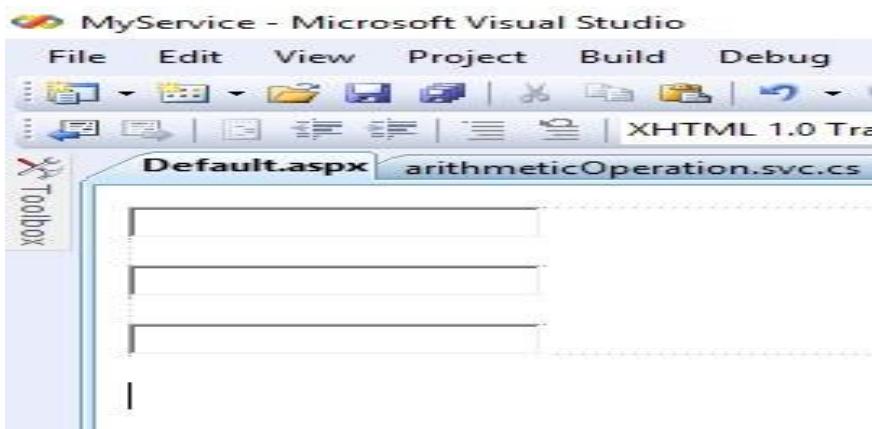


- 30.** Now click after textbox and press two times enter button to create vertical space between two textboxes.



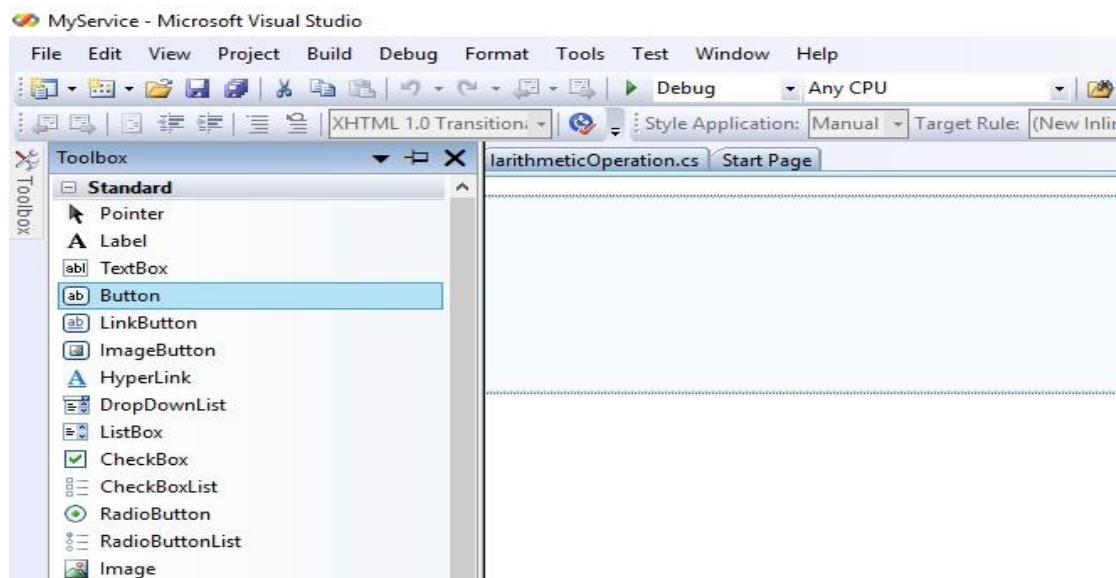


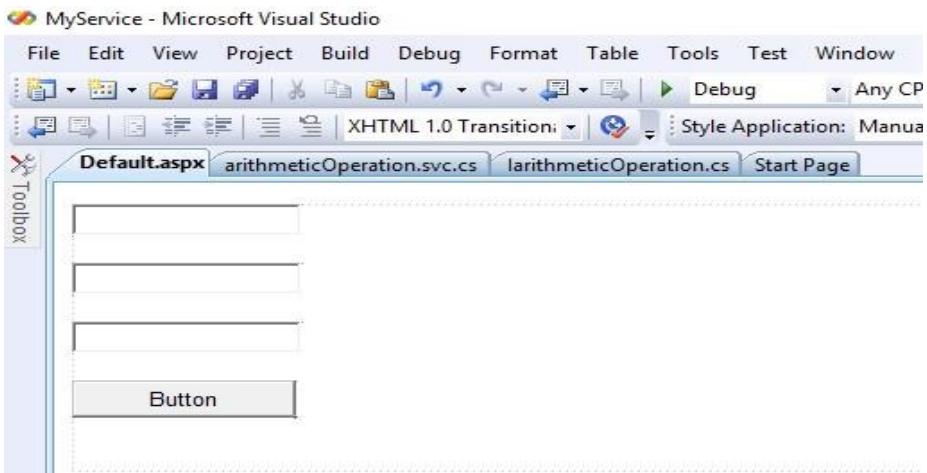
31. Repeat step number 27 & 30 to create two more textboxes. Default ID for these two textboxes will be **TextBox2** and **TextBox3** respectively. It will look like below.



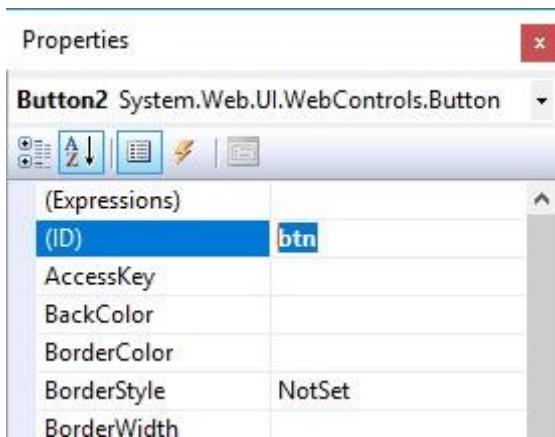
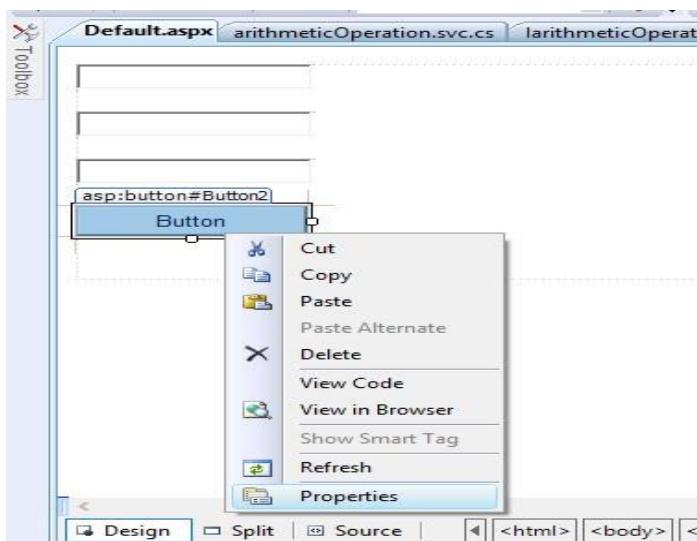
32. Now adding one button by follow below steps.

Click on Toolbox, select button and drag it into the blank screen.





33. Right click on button and select Properties to give its ID and text.



Properties

btn System.Web.UI.WebControls.Button

(Expressions)

(ID)	btn
AccessKey	
BackColor	
BorderColor	
BorderStyle	NotSet
BorderWidth	
CausesValidation	True
CommandArgument	
CommandName	
CssClass	
Enabled	True
EnableTheming	True
EnableViewState	True

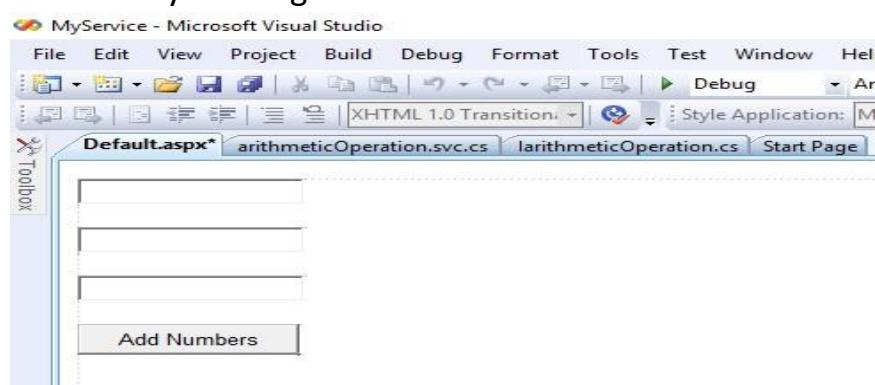
Font

ForeColor	
Height	
OnClientClick	
PostBackUrl	
SkinID	
TabIndex	0
Text	Add Numbers
ToolTip	
UseSubmitBehavior	True
ValidationGroup	
Visible	True

Text

The text to be shown on the button.

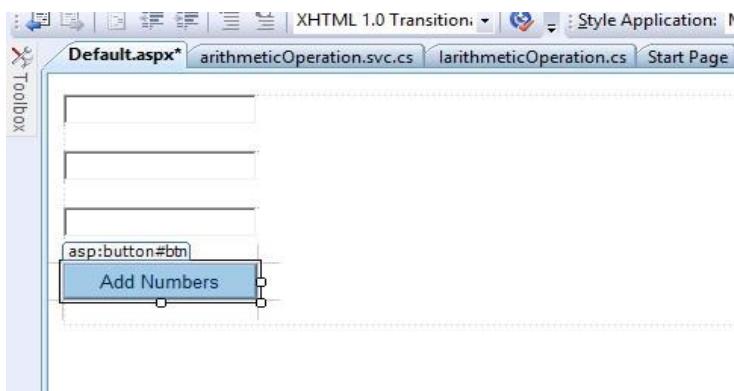
So in this way I have given ID as btn and Text as Add Numbers.



Now I'm going to create an even that will trigger by click on this button. It will get the data from first two TextBoxes and Using WCF service it will add those two numbers. Result of the service will displayed in third textbox.

As we have created two methods in WCF service. You can add one more button to call the second service method. But I'm going to use only one service method.

34. Now to add event on button click, **double click on button**. It will automatically navigate you to the block where you can add your codes for event.

A screenshot of the Visual Studio IDE in code mode. The window title is 'Default.aspx.cs'. The toolbar tabs include 'Default.aspx*', 'arithmeticOperation.svc.cs', 'arithmetricOperation.cs', and 'Start Page'. The code editor shows the following C# code:using System.Linq;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Xml.Linq;

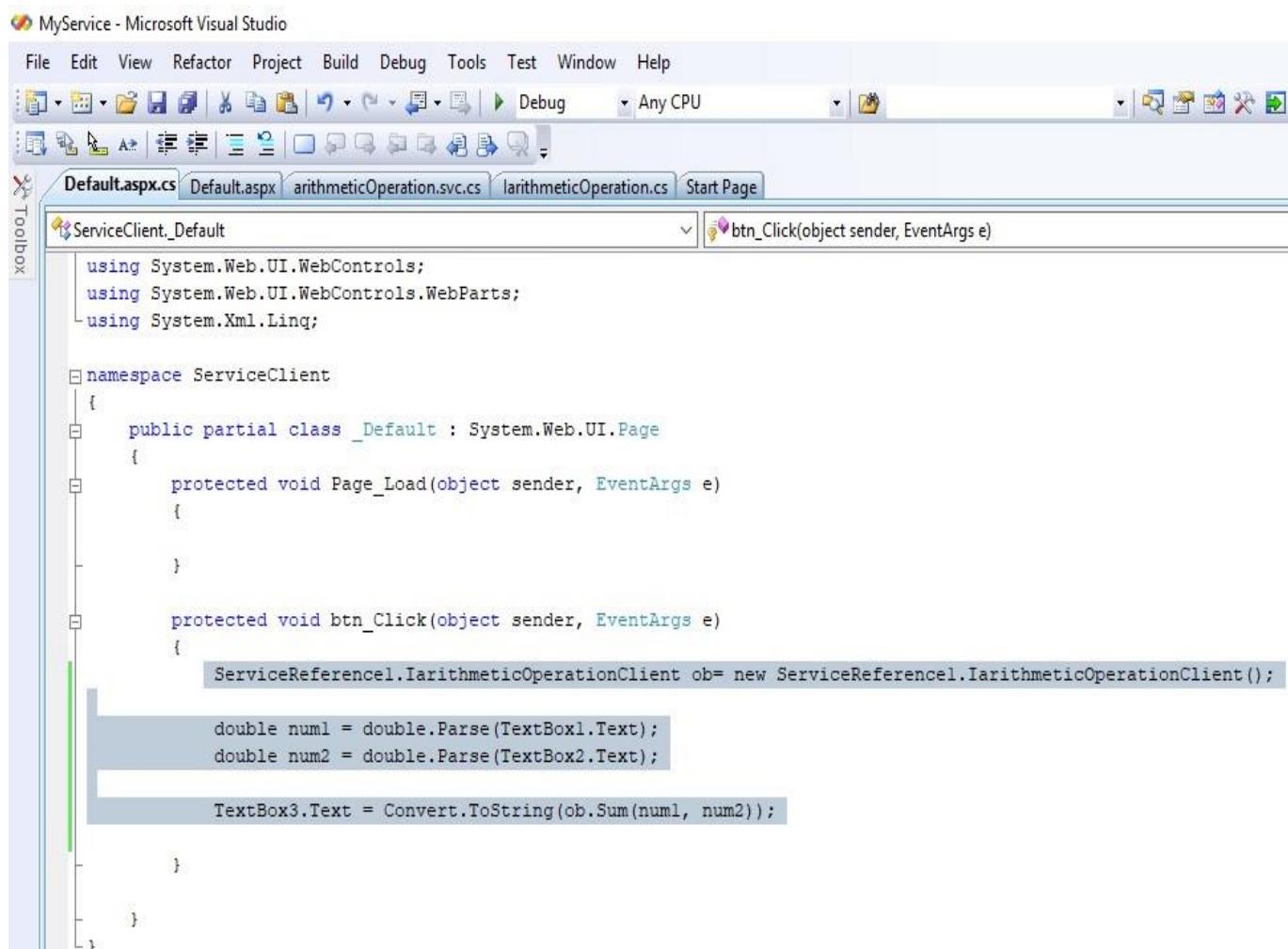
namespace ServiceClient
{
 public partial class _Default : System.Web.UI.Page
 {
 protected void Page_Load(object sender, EventArgs e)
 {

 }

 protected void btn_Click(object sender, EventArgs e)
 {
 }
 }
}The 'btn_Click' event handler is currently selected in the code editor.

- 35. Now add the following code into this block and press Ctrl+S to save the changes.**

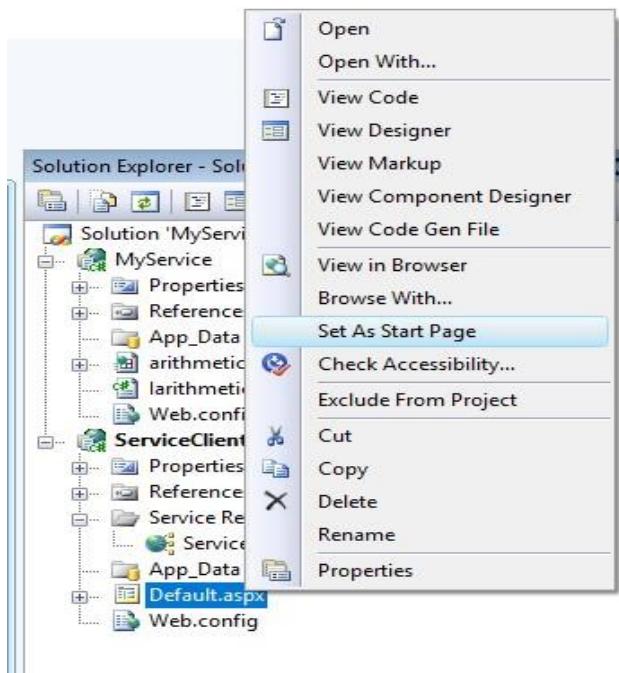
```
ServiceReference1.IarithmeticOperationClient ob= new  
ServiceReference1.IarithmeticOperationClient();  
double num1 = double.Parse(TextBox1.Text);  
double num2 = double.Parse(TextBox2.Text);  
  
TextBox3.Text = Convert.ToString(ob.Sum(num1, num2));
```



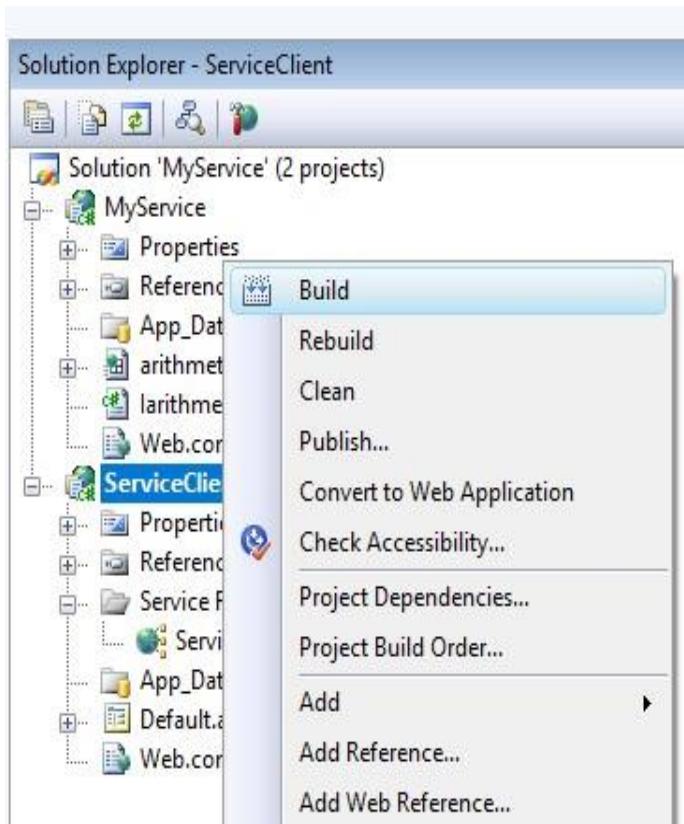
The screenshot shows the Microsoft Visual Studio interface with the title bar "MyService - Microsoft Visual Studio". The menu bar includes File, Edit, View, Refactor, Project, Build, Debug, Tools, Test, Window, Help. The toolbar has various icons for file operations. The solution explorer on the left shows files like Default.aspx.cs, Default.aspx, arithmeticOperation.svc.cs, IarithmeticOperation.cs, and Start Page. The code editor window displays the C# code for the Default.aspx.cs page. A specific line of code, "double num1 = double.Parse(TextBox1.Text);", is highlighted with a gray selection bar, indicating it is the current focus for modification.

```
using System.Web.UI.WebControls;  
using System.Web.UI.WebControls.WebParts;  
using System.Xml.Linq;  
  
namespace ServiceClient  
{  
    public partial class _Default : System.Web.UI.Page  
    {  
        protected void Page_Load(object sender, EventArgs e)  
        {  
        }  
  
        protected void btn_Click(object sender, EventArgs e)  
        {  
            ServiceReference1.IarithmeticOperationClient ob= new ServiceReference1.IarithmeticOperationClient();  
  
            double num1 = double.Parse(TextBox1.Text);  
            double num2 = double.Parse(TextBox2.Text);  
  
            TextBox3.Text = Convert.ToString(ob.Sum(num1, num2));  
        }  
    }  
}
```

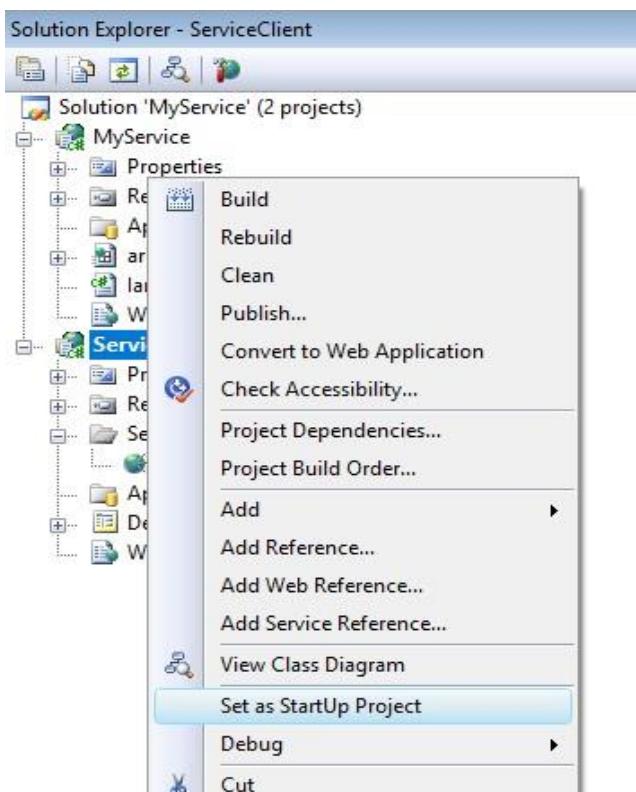
36. Now right click on Default.aspx in ServiceClient set it as start page.



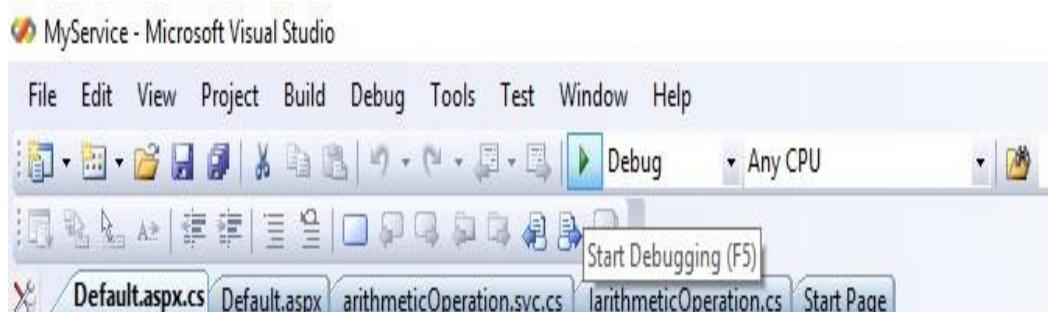
37. Right click on ServiceClient and build the project.



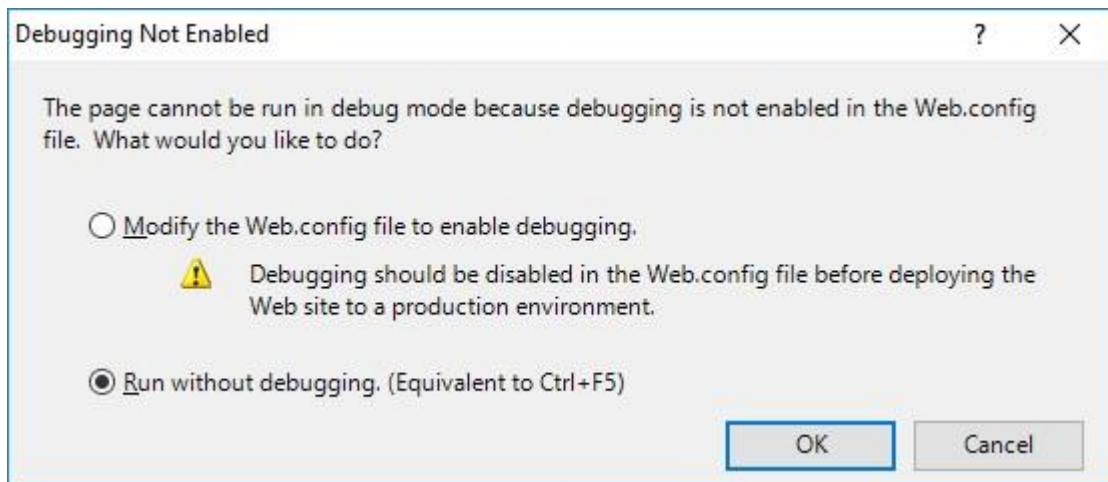
38. Again right click on ServiceClient and set it as StartUp project.



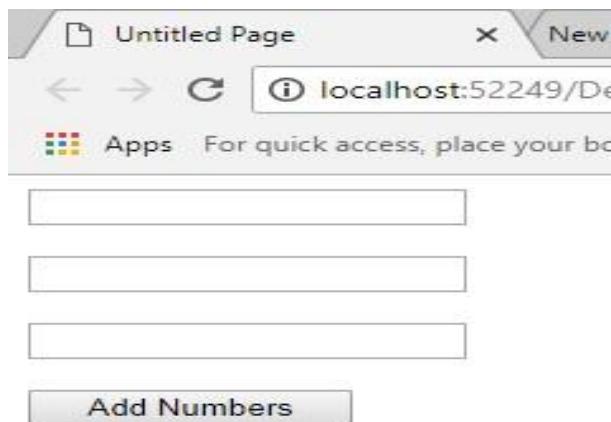
39. Now click on Run button or press F5 to run this project.



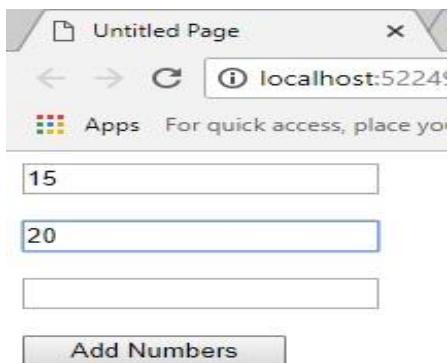
40. Select run without debugging and click OK button.



41. A window will open in browser like below.



42. Now enter two numbers in first two boxes and click on Add Numbers button.
Then addition of above two numbers will displayed into third textbox.



A screenshot of a web browser window titled "Untitled Page". The address bar shows "localhost:52249/". Below the address bar, there is a "Apps" section with the placeholder "For quick access, place your". Three input fields are present, each containing a number: "15", "20", and "35". Below these input fields is a button labeled "Add Numbers".

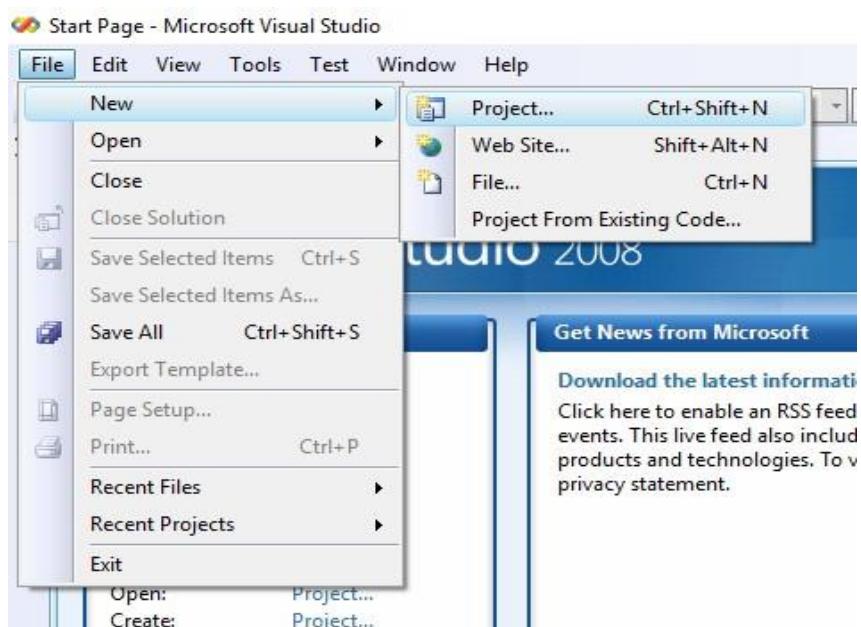
You can create one more button for multiplication event that we have created in WCF service.

Practical-9

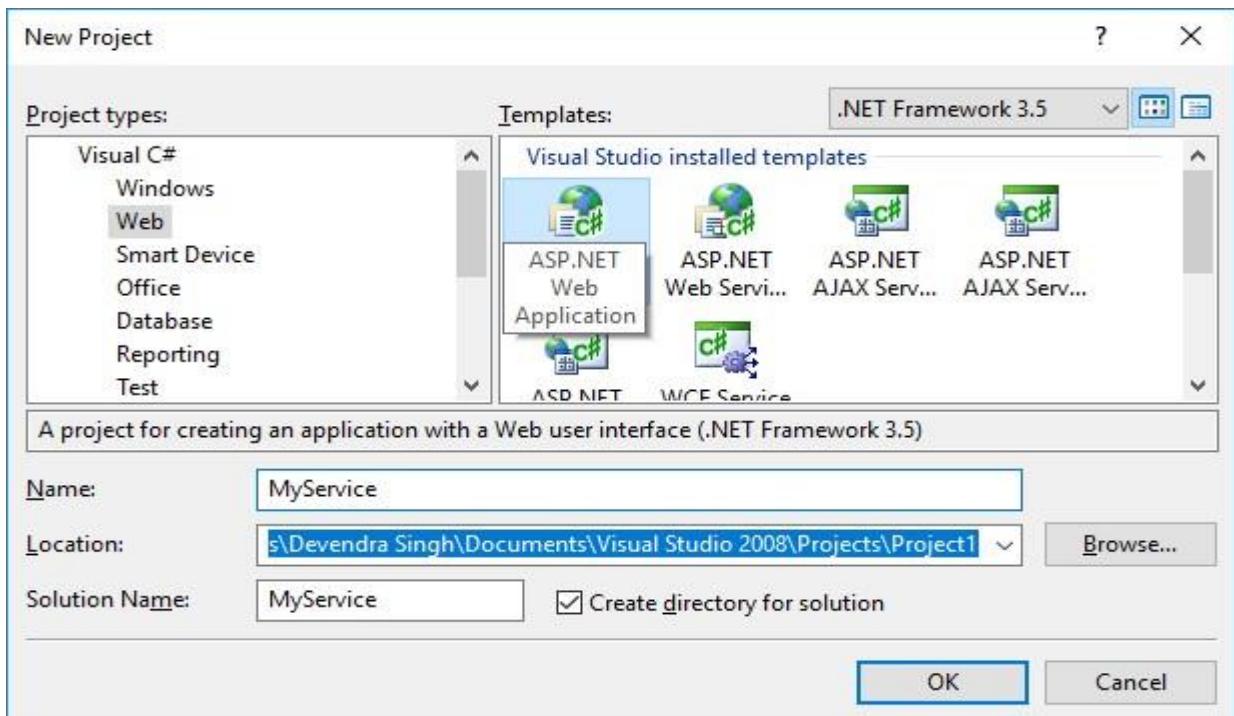
Aim: Use WCF to create a basic ASP.NET Asynchronous JavaScript and XML (AJAX) service.

1. Open Visual Studio to create a Web Application.

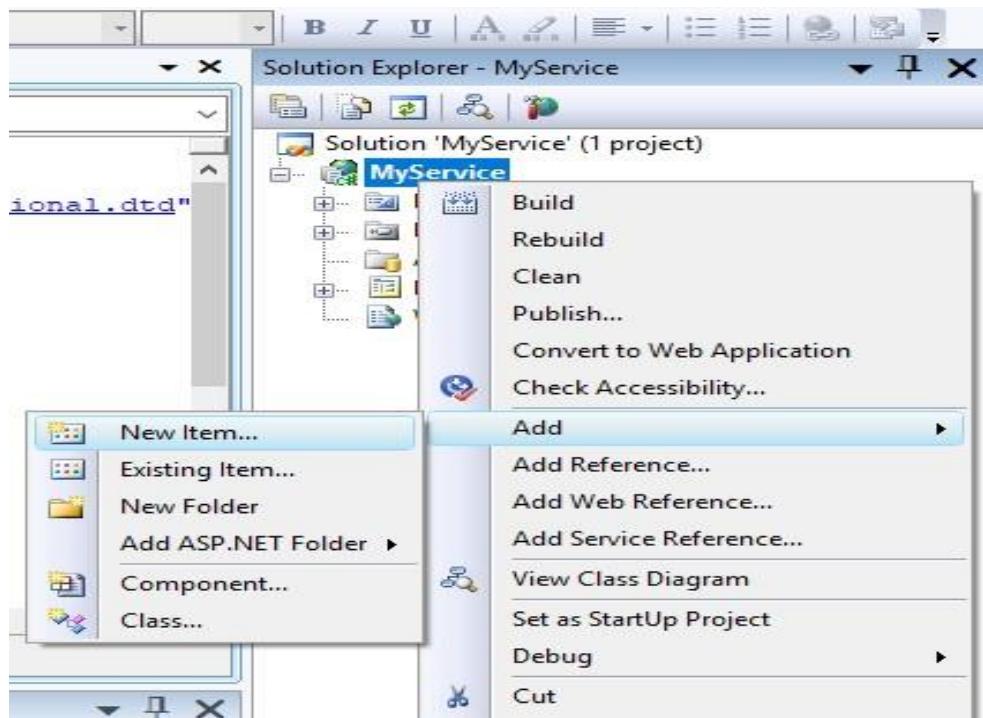
File -> New -> Project



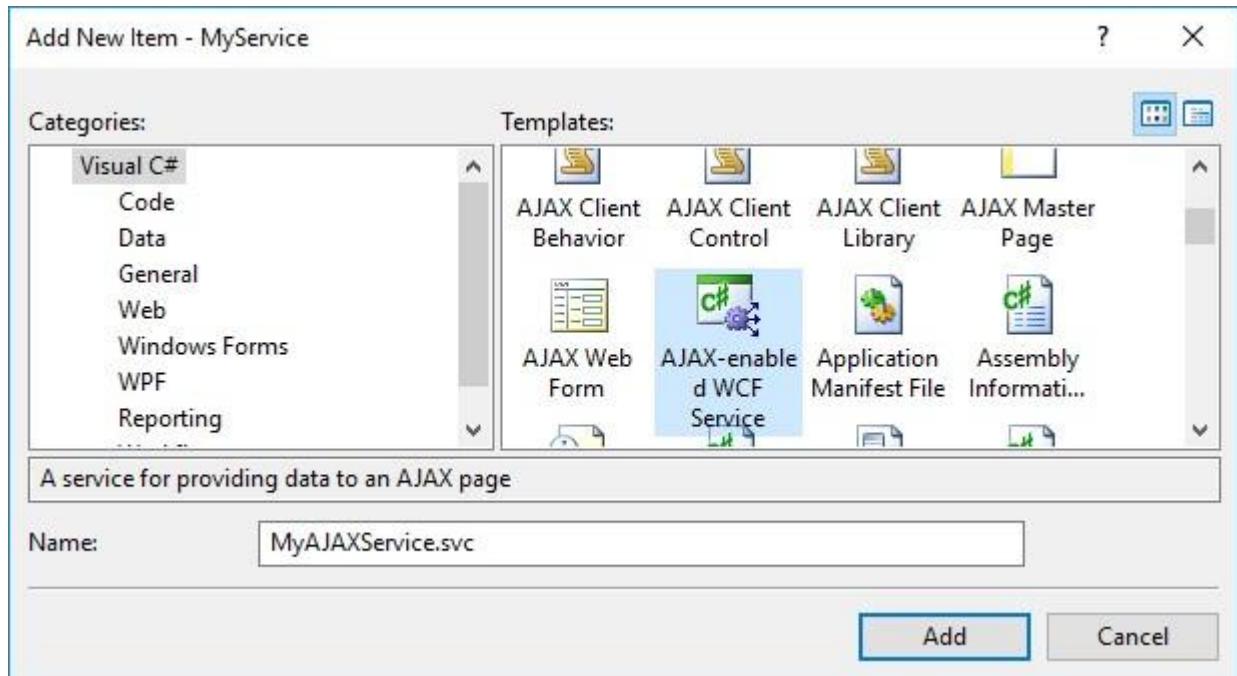
2. In Project Types, select Web and in Templates, select ASP.NET Web Application. Give MyService name to this web application and click on OK button.



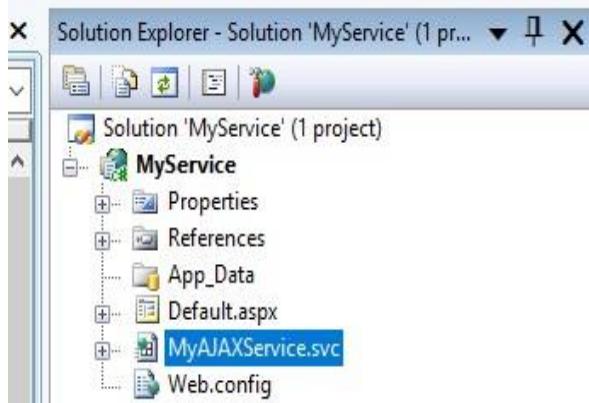
3. Now we are going to add AJAX -enabled WCF Service. Go to solution Explorer. Right click on **MyService** -> Add -> New Item.



4. In Templates select **AJAX-enabled WCF Service** and give **MyAJAXService** name to it. After that press **OK** button.



5. As you can see in solution explorer, service is added to the web application.



6. Open it by double click on it. It will look like below.

```
MyAJAXService.svc.cs Default.aspx Start Page
MyService.MyAJAXService DoWork()

namespace MyService
{
    [ServiceContract(Namespace = "")]
    [AspNetCompatibilityRequirements(RequirementsMode = AspNetCompatibilityRequirementsMode.Allowed)]
    public class MyAJAXService
    {
        // Add [WebGet] attribute to use HTTP GET
        [OperationContract]
        public void DoWork()
        {
            // Add your operation implementation here
            return;
        }

        // Add more operations here and mark them with [OperationContract]
    }
}
```

7. Replace the **DoWork** service method with another method. As I'm going to add a **service method Sum** for addition of two numbers. You can add any method. But accordingly you have to change the calling of this service. **After adding the below code, press Ctrl+S to save the changes.**

```
public double Sum(double a, double b)
{
    double result = a + b;
return result;
}
```

```
MyAJAXService.svc.cs Default.aspx Start Page
MyService.MyAJAXService Sum(double a, double b)

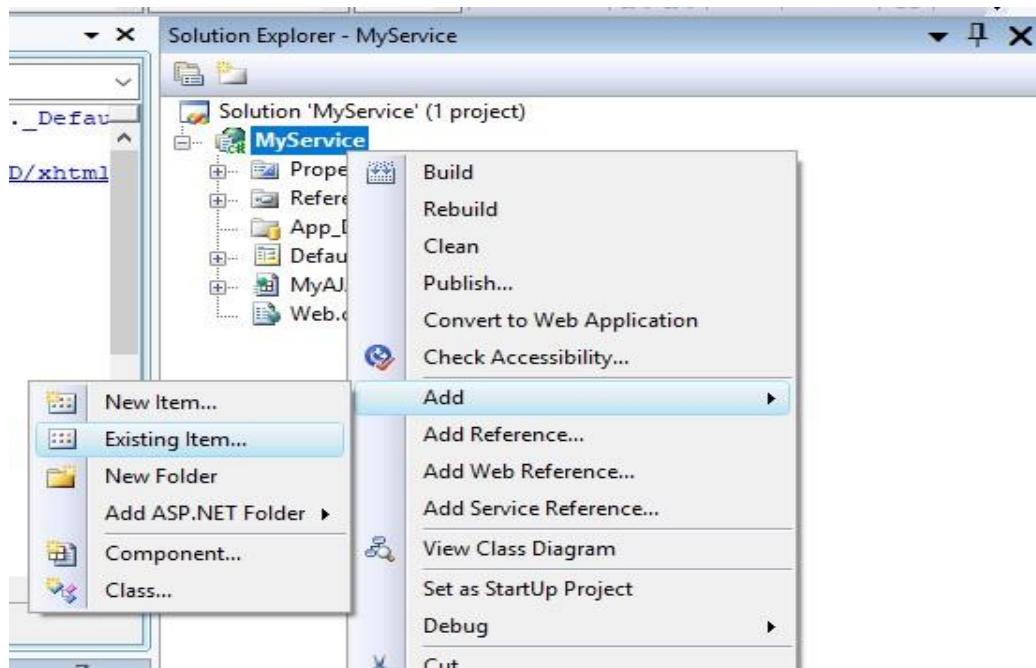
namespace MyService
{
    [ServiceContract(Namespace = "")]
    [AspNetCompatibilityRequirements(RequirementsMode = AspNetCompatibilityRequirementsMode.Allowed)]
    public class MyAJAXService
    {

        [OperationContract]
        public double Sum(double a, double b)
        {
            double result = a + b;
            return result;
        }

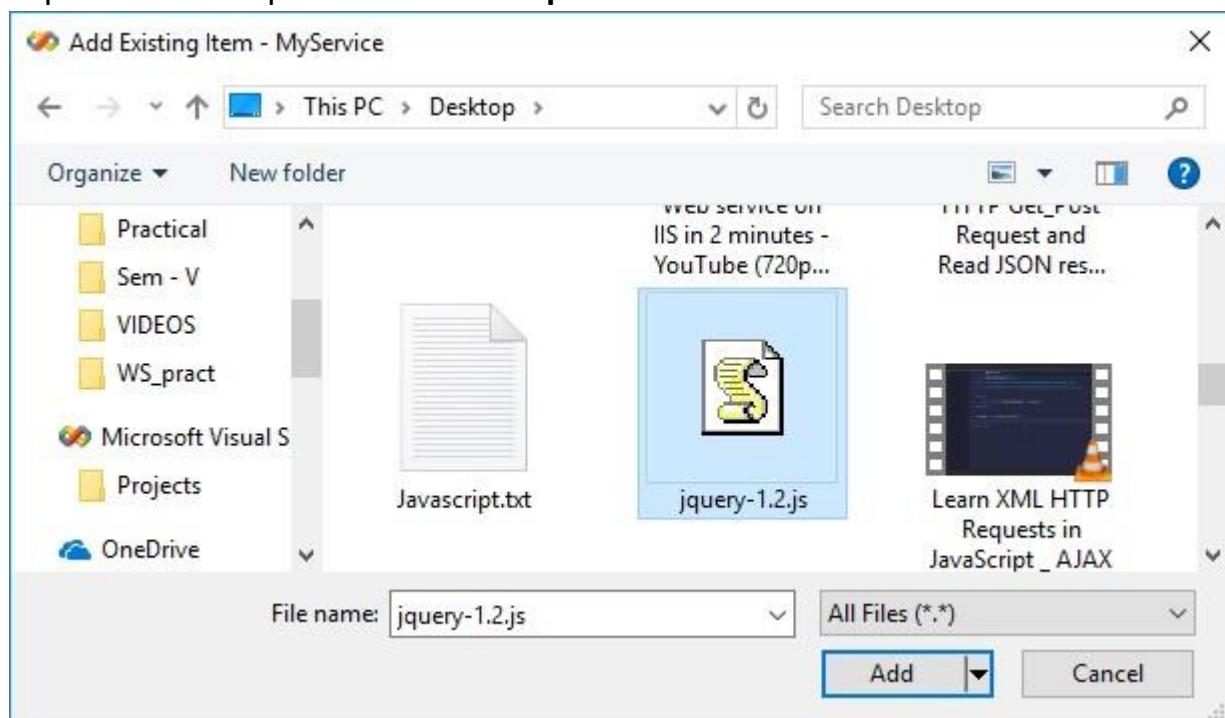
        // Add more operations here and mark them with [OperationContract]
    }
}
```

8. As I will to use jquery to call above WCF Service. So I need to add library for jquery. I have downloaded it and I'm going to add it to my Web Application.

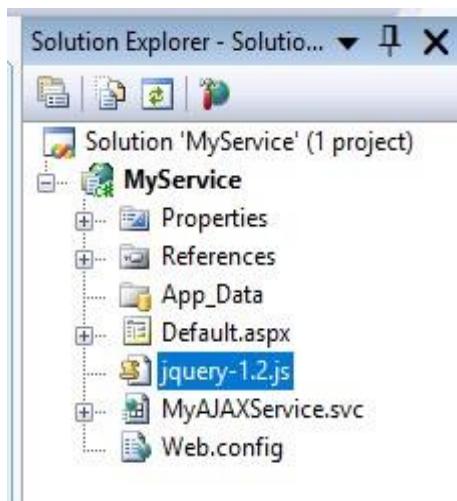
Right click on MyService -> Add -> Existing Item



9. I have kept it on Desktop. So select it and press Add button.

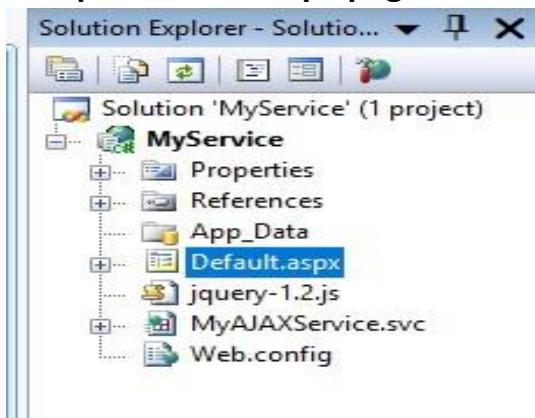


10. As you can see library is added to application.



11. Now I'm going to consume that service.

Open Default.aspx page from Solution Explorer by double click on it.



12. It will open in Source mode like below.

The screenshot shows the Microsoft Visual Studio IDE with the "Default.aspx" page open in Source mode. The window title is "Default.aspx". The code editor displays the following ASPX code:

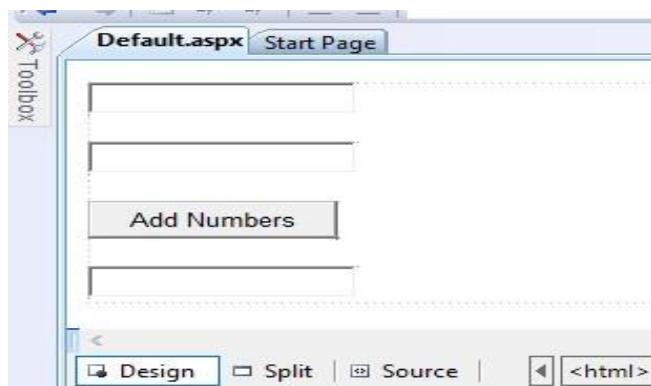
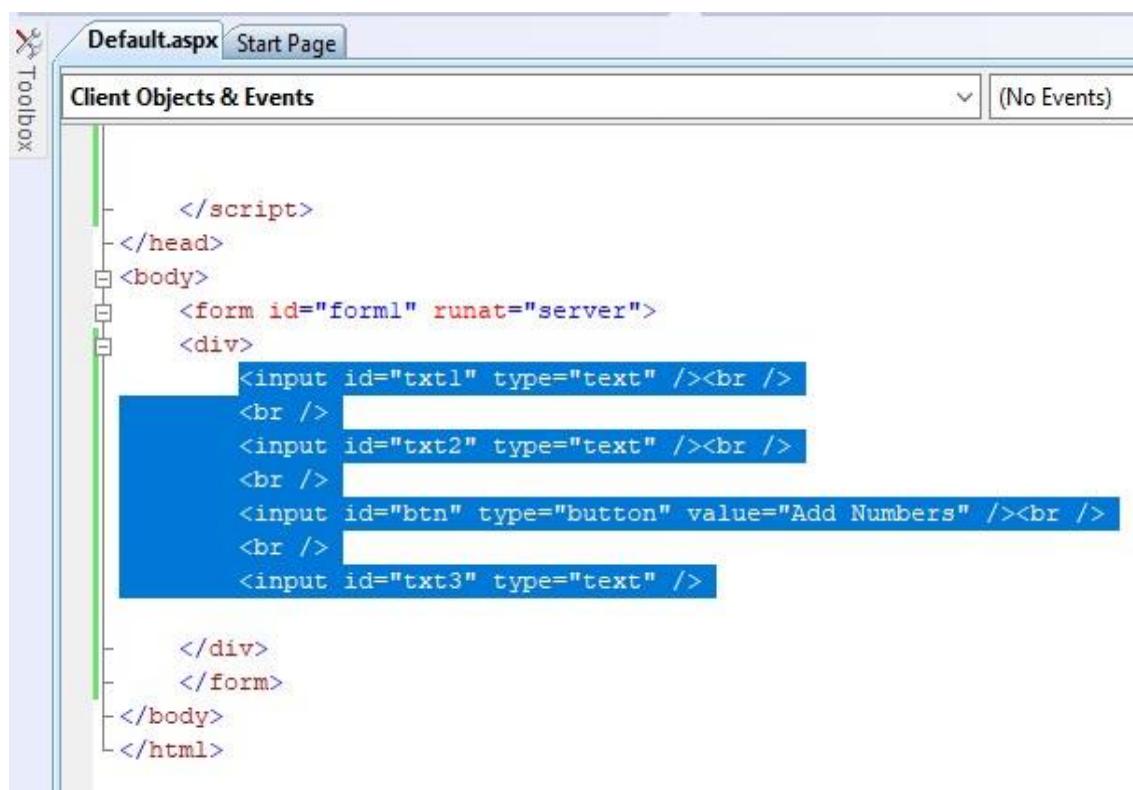
```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/1999/xhtml">
<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Untitled Page</title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
        </div>
    </form>
</body>
</html>
```

The "Toolbox" is visible on the left side of the IDE. At the bottom of the screen, there are tabs for "Design", "Split", and "Source", with "Source" currently selected. Below the tabs, there are navigation icons for navigating through the code.

13. Add the following code in between DIV tag. It will create three textboxes and one button. You can see it in Design mode.

```
<input id="txt1" type="text" /><br />
<br />
<input id="txt2" type="text" /><br />
<br />
<input id="btn" type="button" value="Add
Numbers" /><br />
<br />
<input id="txt3" type="text" />
```



14. We have given following ID's to above components. We will use these ID's in jquery

1st Textbox = **txt1**

2nd Textbox = **txt2**

Button = **btn**

3rd Textbox = **txt3**

1st and 2nd textboxes are for enter two numbers. When you will click on button then jquery will get triggered and by calling the service we will get output in 3rd textbox.

Now again go into Source mode.

15. Add the following jquery code in HEAD tag and then press Ctrl+S to save the changes.

```
<script type="text/javascript" src="jquery-1.2.js"></script>

<script type="text/javascript" >

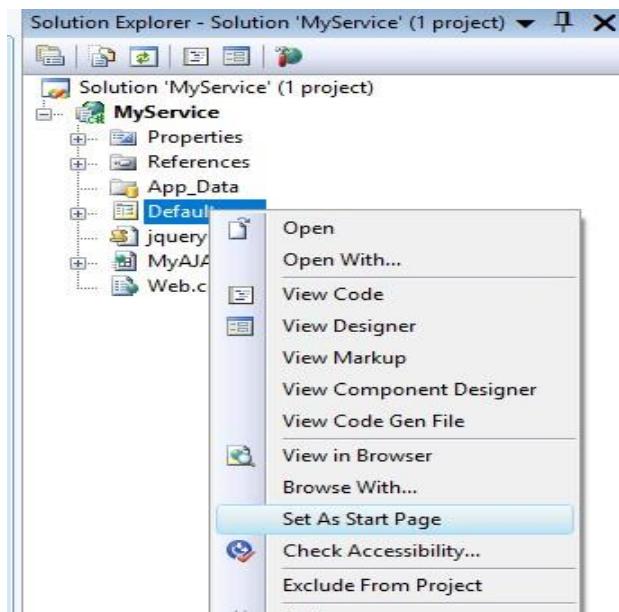
    $(document).ready(function () {
        $("#btn").click(function () {
            var num1 = $("#txt1").val();                                var
            num2 = $("#txt2").val();

            $.ajax({
                url: "MyAJAXService.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>
```

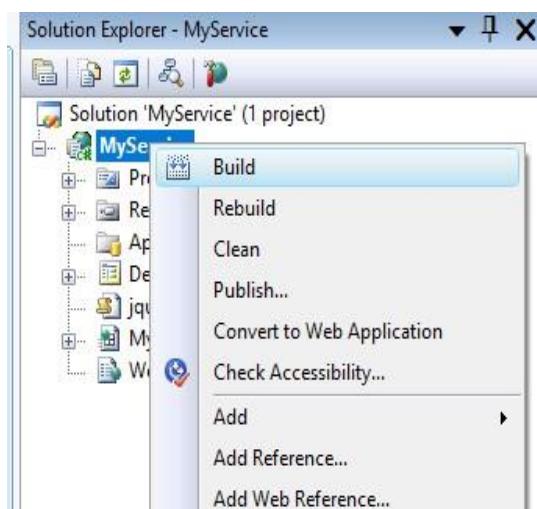
As you can see in first line of code, in src I have given the path of library that I have added using step 8 and 9. Without this library our code will not work. So it is mandatory.

16. Now set Default.aspx page as start page.

Right click on Default.aspx -> Set As Start Page



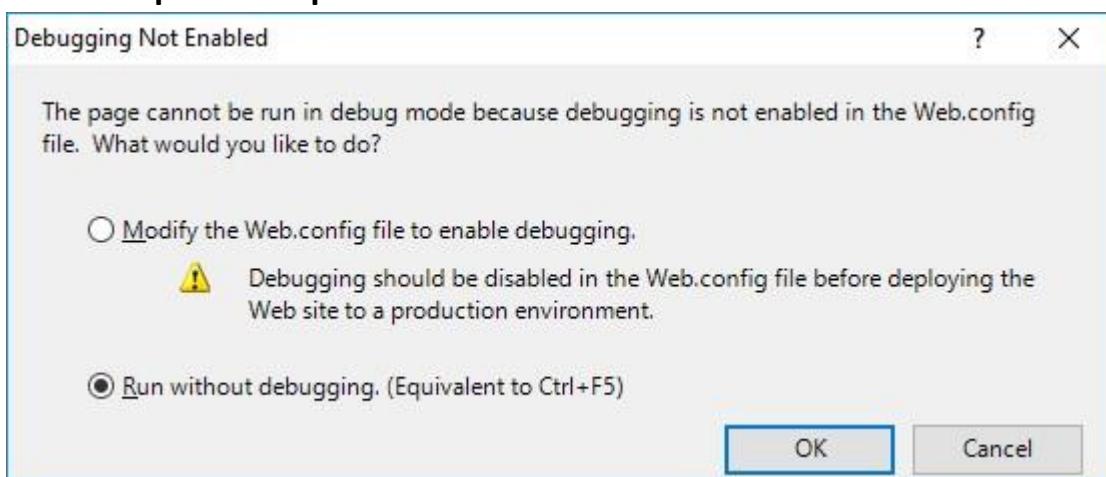
17. Build application.



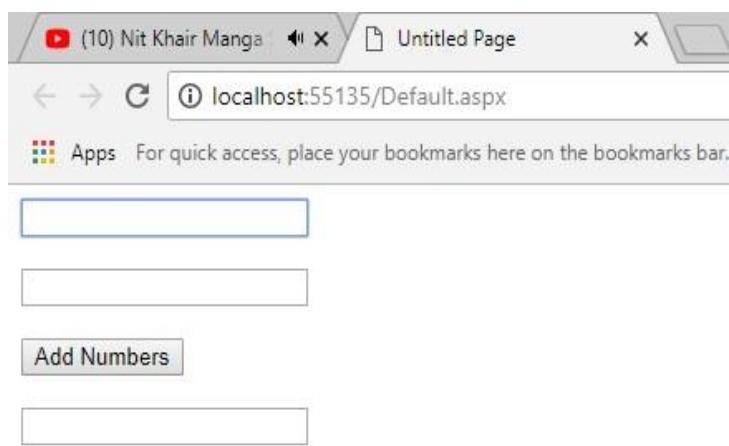
18. Press Run button in Toolbar.



19. Select second option and press OK.



20. A window will open in browser like below.



21. Giving two numbers.

A screenshot of a web browser window titled "Untitled Page". The address bar shows "localhost:55135/Default.aspx". Below the address bar is a toolbar with icons for back, forward, and refresh. A message bar says "For quick access, place your bookmarks here on the bookmarks". There are two input fields: the first contains "20.5" and the second contains "5". Below the inputs is a button labeled "Add Numbers". Underneath the button is a large empty rectangular box.

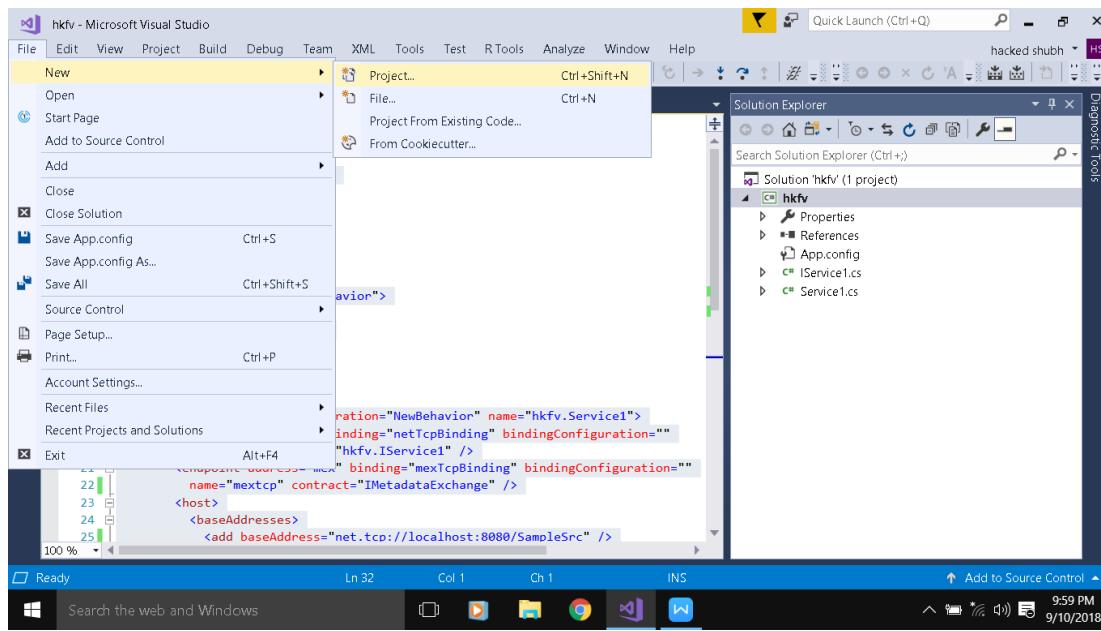
After button press.

A screenshot of the same web browser window after the "Add Numbers" button was pressed. The inputs remain the same ("20.5" and "5"). The large empty box now contains the result "25.5".

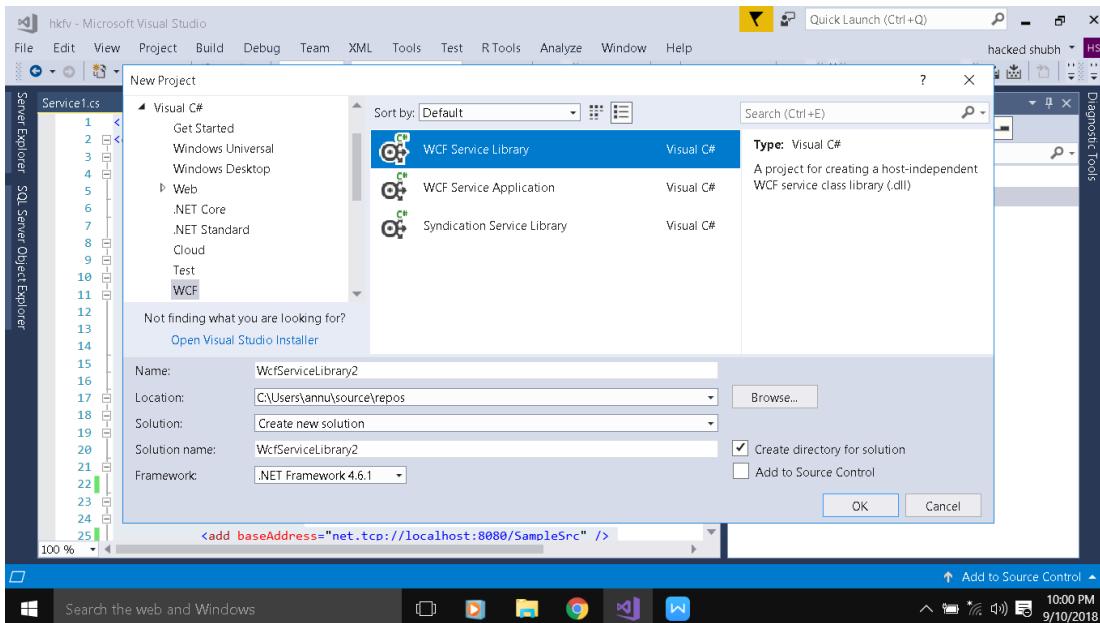
PRACTICAL-10

Aim: Demmonstrating endpoints in wcf

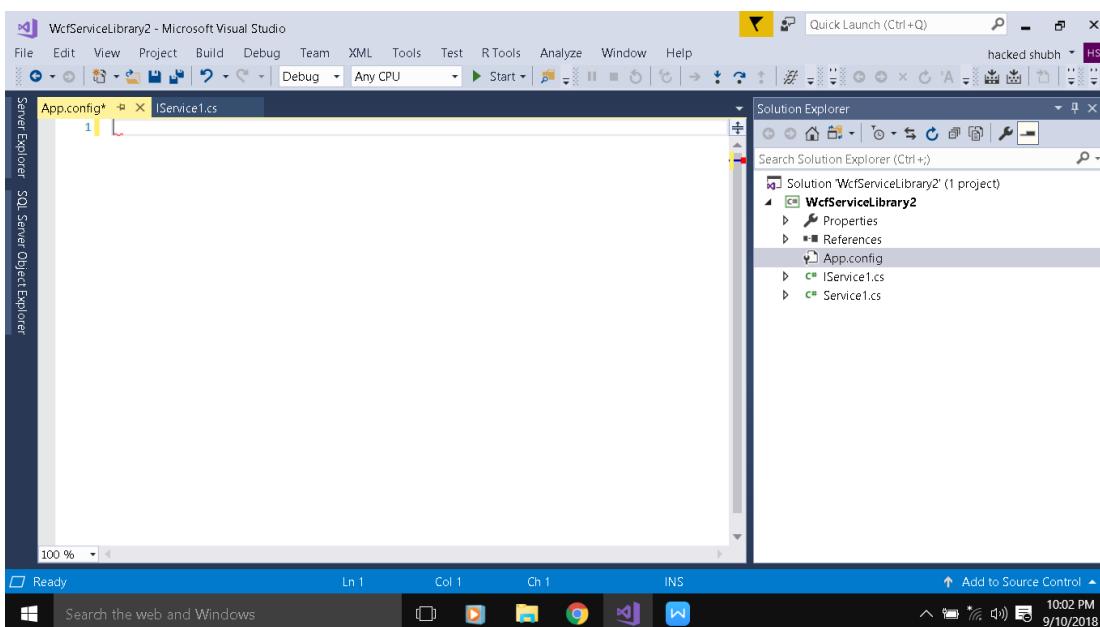
1.Theory:create a new project



2. Select wcf service library and rename it accordingly



3. Keep everything as it is and open Appconfig file and erase every data and copy paste the following code



```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <system.web>
    <compilation debug="true">
      </compilation>
  </system.web>

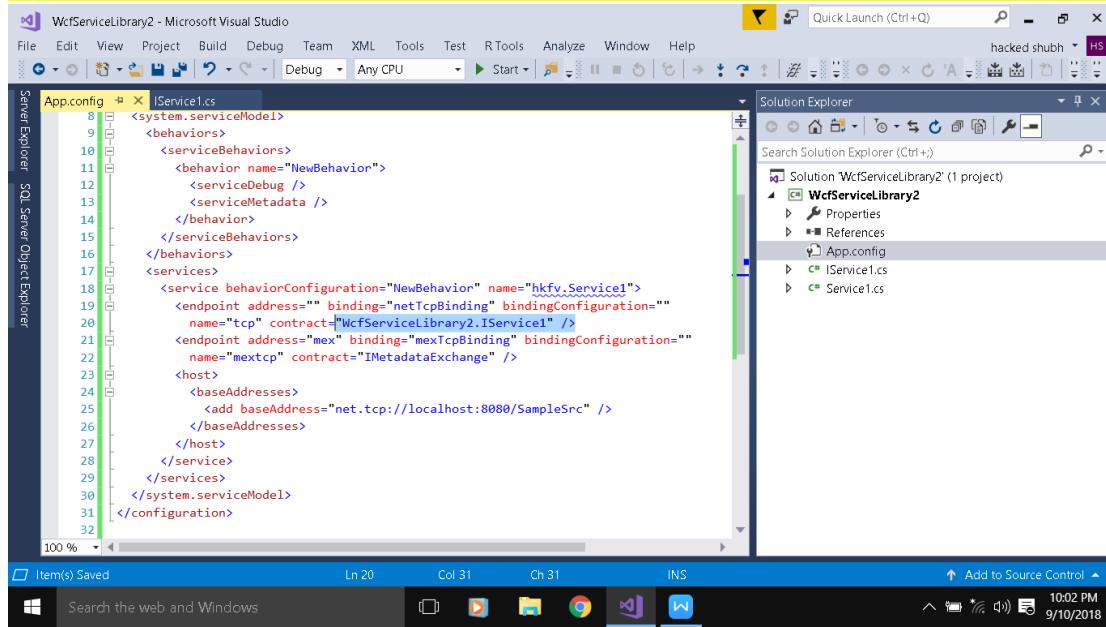
  <system.serviceModel>
    <behaviors>
```

```

<serviceBehaviors>
    <behavior name="NewBehavior">
        <serviceDebug />
        <serviceMetadata />
    </behavior>
</serviceBehaviors>
</behaviors>
<services>
    <service behaviorConfiguration="NewBehavior" name="hkfv.Service1">
        <endpoint address="" binding="netTcpBinding" bindingConfiguration=""
            name="tcp" contract="hkfv.IService1" />
        <endpoint address="mex" binding="mexTcpBinding" bindingConfiguration=""
            name="mextcp" contract="IMetadataExchange" />
        <host>
            <baseAddresses>
                <add baseAddress="net.tcp://localhost:8080/SampleSrc" />
            </baseAddresses>
        </host>
    </service>
</services>
</system.serviceModel>
</configuration>

```

Note: in Name and Contract attribute write the name of your wcf application



4. Build and run the program

