

PRACTICAL 3

Date: 02-01-25

Aim: Write a program to show the object communication using RMI.

a) RMI-based application program to display current date and time.

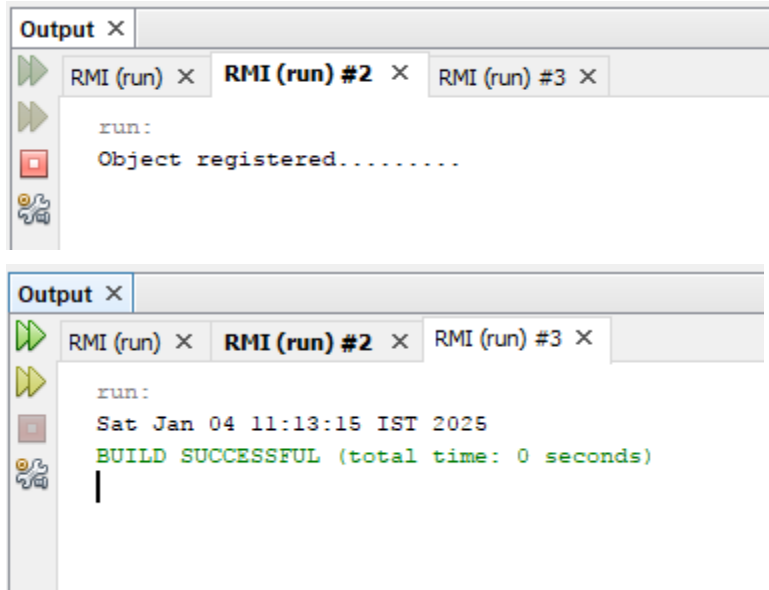
```
import java.rmi.*;
public interface InterDate extends Remote {
    public String display() throws Exception;
}
//Server code
import java.rmi.*;
import java.rmi.server.*;
import java.util.*;
public class ServerDate extends UnicastRemoteObject implements InterDate{
    public ServerDate() throws Exception
    {
    }
    public String display() throws Exception
    {
        String str="";
        Date d=new Date();
        str=d.toString(); return
        str;
    }
    public static void main(String args[]) throws Exception {
        ServerDate s1=new ServerDate();
        Naming.bind("DS", s1);
        System.out.println("Object registered.....");
    }
}
```

Client Code

```
import java.rmi.*;
import java.io.*;
public class ClientDate {
    public static void main(String args[]) throws Exception {
        String
        s1;
        InterDate h1=(InterDate)Naming.lookup("DS");
```

```
s1=h1.display();
System.out.println(s1);
}
}
```

OUTPUT:



- b)** RMI-based application program that converts digits to words. Eg 123 will be converted to ONE TWO THREE.

```
//interface.java
import java.rmi.*;
public interface InterConvert extends Remote{
    public String convertDigit(String no) throws Exception;
}
//server code
import java.rmi.*;
import java.rmi.server.*;
public class ServerConvert extends UnicastRemoteObject implements InterConvert{
    public ServerConvert() throws Exception {
    }
    public String convertDigit(String no) throws Exception {
    }
    String str="";
    for(int i=0;i<no.length();i++)
    {
        int p=no.charAt(i);
        if(p==48)
```

Pratik Patil

```
{
    str+="zero";
}
if(p==49)
{
    str+="one";
}
if(p==50)
{
    str+="two";
}
if(p==51)
{
    str+="three";
}
if(p==52)
{
    str+="four";
}
if(p==53)
{
    str+="five";
}
if(p==54)
{
    str+="six";
}
if(p==55)
{
    str+="seven";
}
if(p==56)
{
    str+="eight";
}
if(p==57)
{
    str+="nine";
}
}
return str;
}
public static void main(String args[]) throws Exception {
```

```

ServerConvert s1=new ServerConvert(); Naming.bind("Wrd",
s1);
System.out.println("Object registered ..... ");
}
}

Client code import
java.rmi.*; import
java.io.*;
public class ClientConvert {
public static void main(String args[]) throws Exception
{
InterConvert h1=(InterConvert)Naming.lookup("Wrd");
BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
System.out.println("Enter a number : \t");
String no=br.readLine();
String ans=h1.convertDigit(no);
System.out.println("The word representation of the entered digit is : "+ans);
}
}

```

OUTPUT:

The screenshot shows an IDE's output window with a tab labeled 'Output'. Below the tab are four tabs for 'RMI (run) #2', 'RMI1 (run)', 'RMI1 (run) #2', and 'RMI1 (run) #3'. The 'RMI1 (run)' tab is active, displaying the output: 'run:' followed by 'Object registered.....' on the next line.

The screenshot shows the same IDE output window. The 'RMI1 (run) #2' tab is now active. The output shows 'run:' followed by 'Enter a number :', then the user input '5' on the next line. This is followed by 'The word representation of the entered digit is : five' and 'BUILD SUCCESSFUL (total time: 6 seconds)' on the final line.

The screenshot shows the same IDE output window. The 'RMI1 (run) #2' tab is active. The output shows 'run:' followed by 'Enter a number :', then the user input '246' on the next line. This is followed by 'The word representation of the entered digit is : twofoursix' and 'BUILD SUCCESSFUL (total time: 13 seconds)' on the final line.

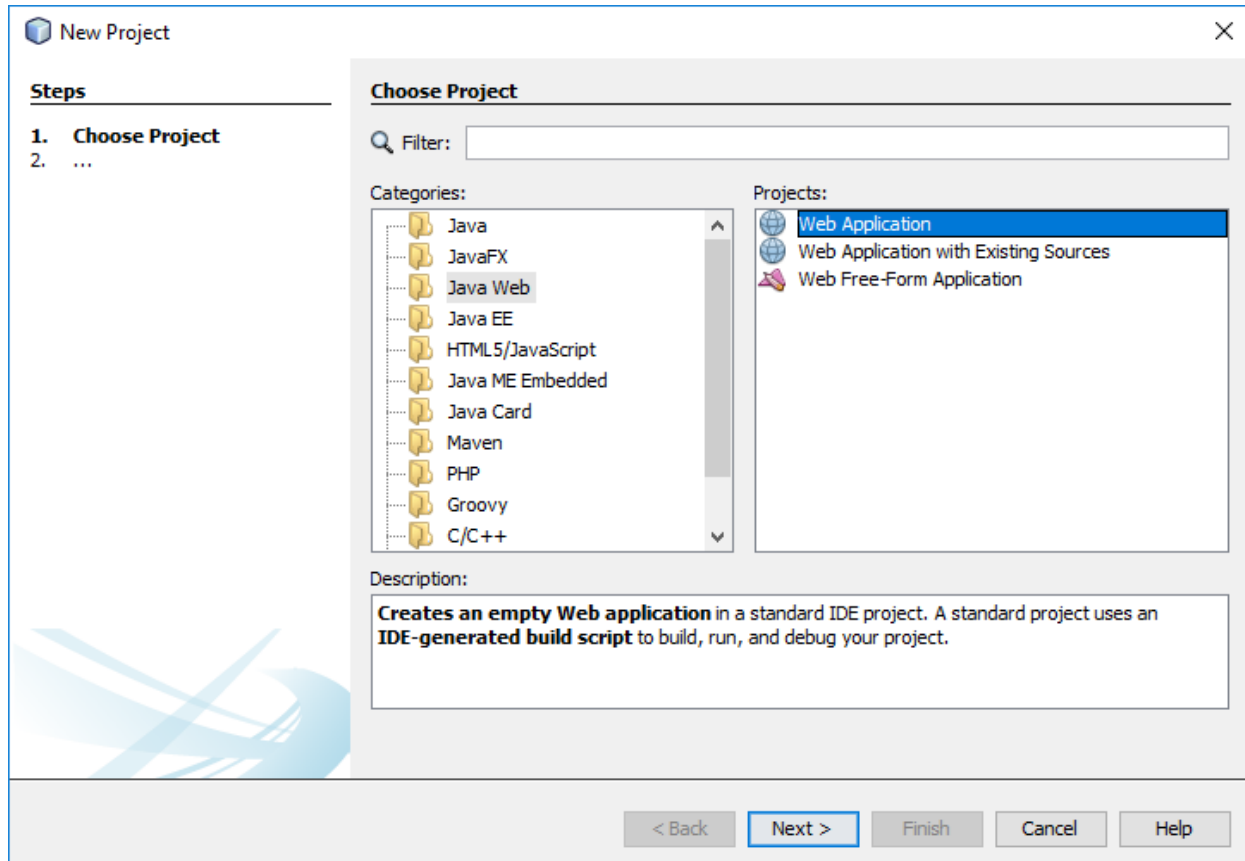
PRACTICAL 4

Date: 16-01-25

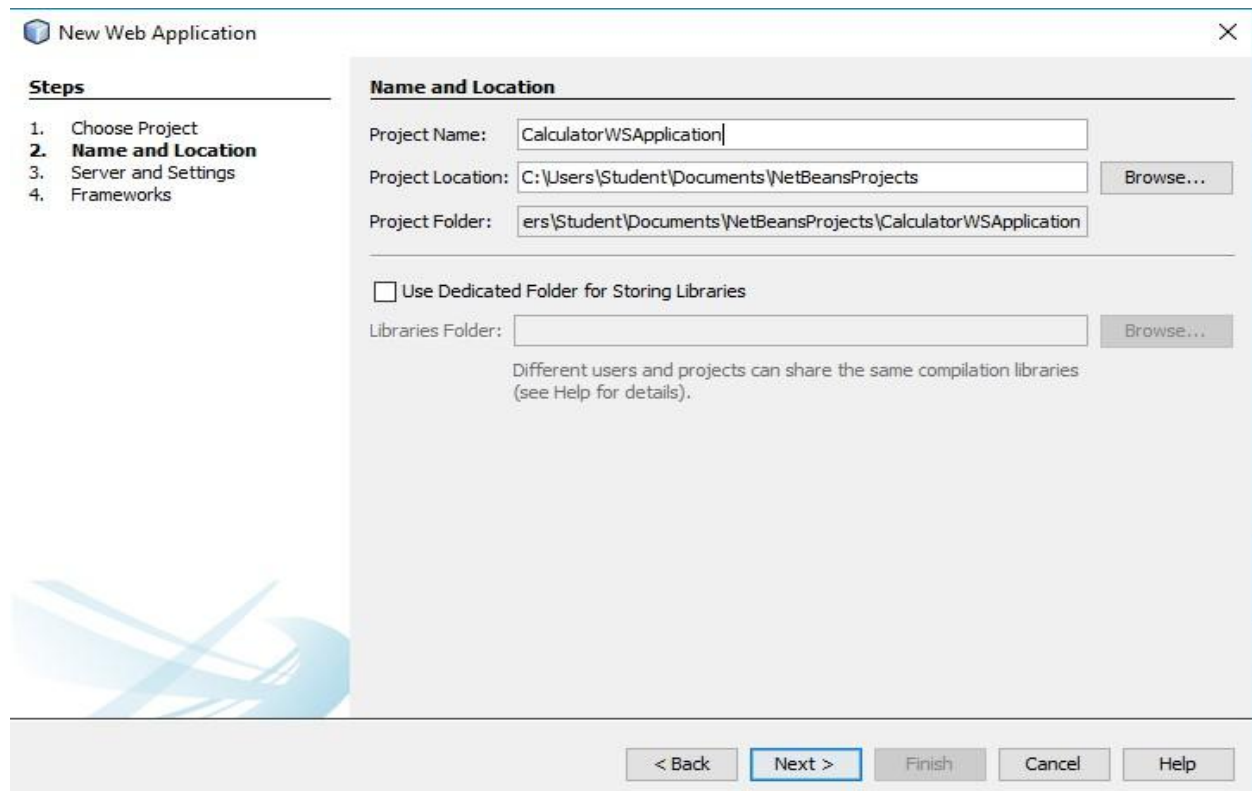
Aim: Show the implementation of webservices.

a) Implement Big web services.

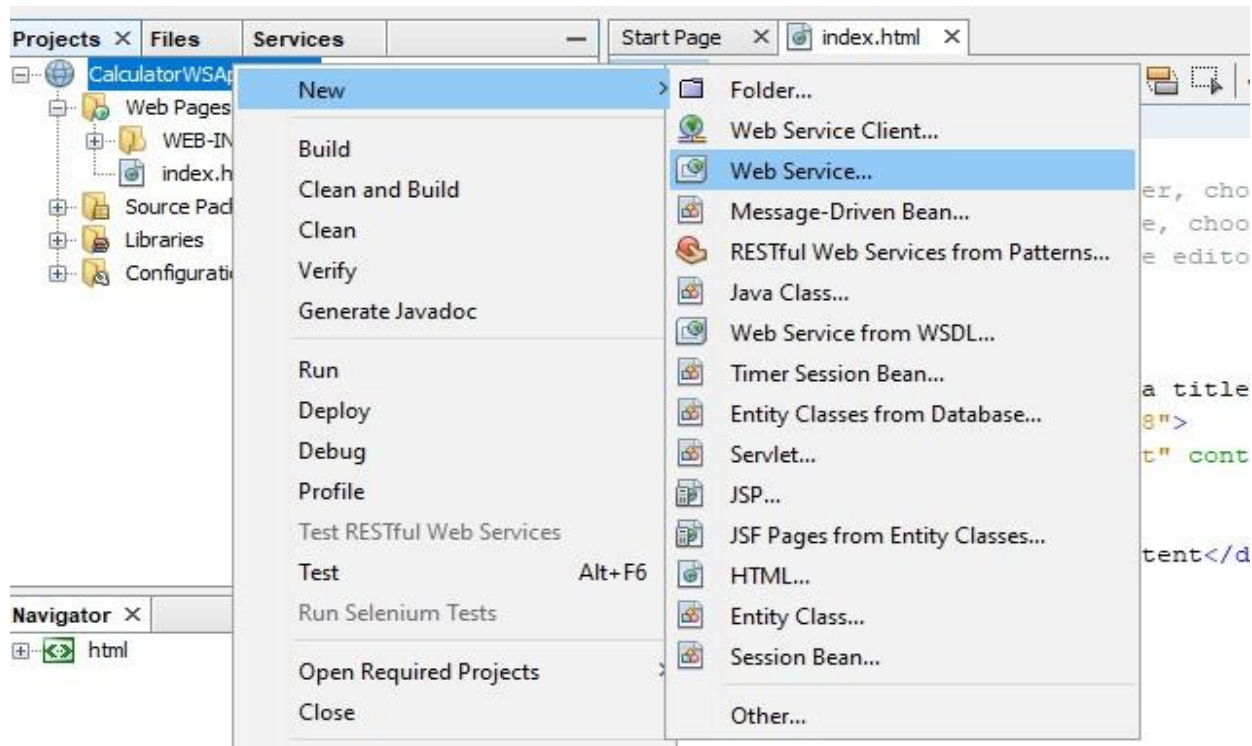
Step 1: Select File → New Project → Java Web → Web Application.



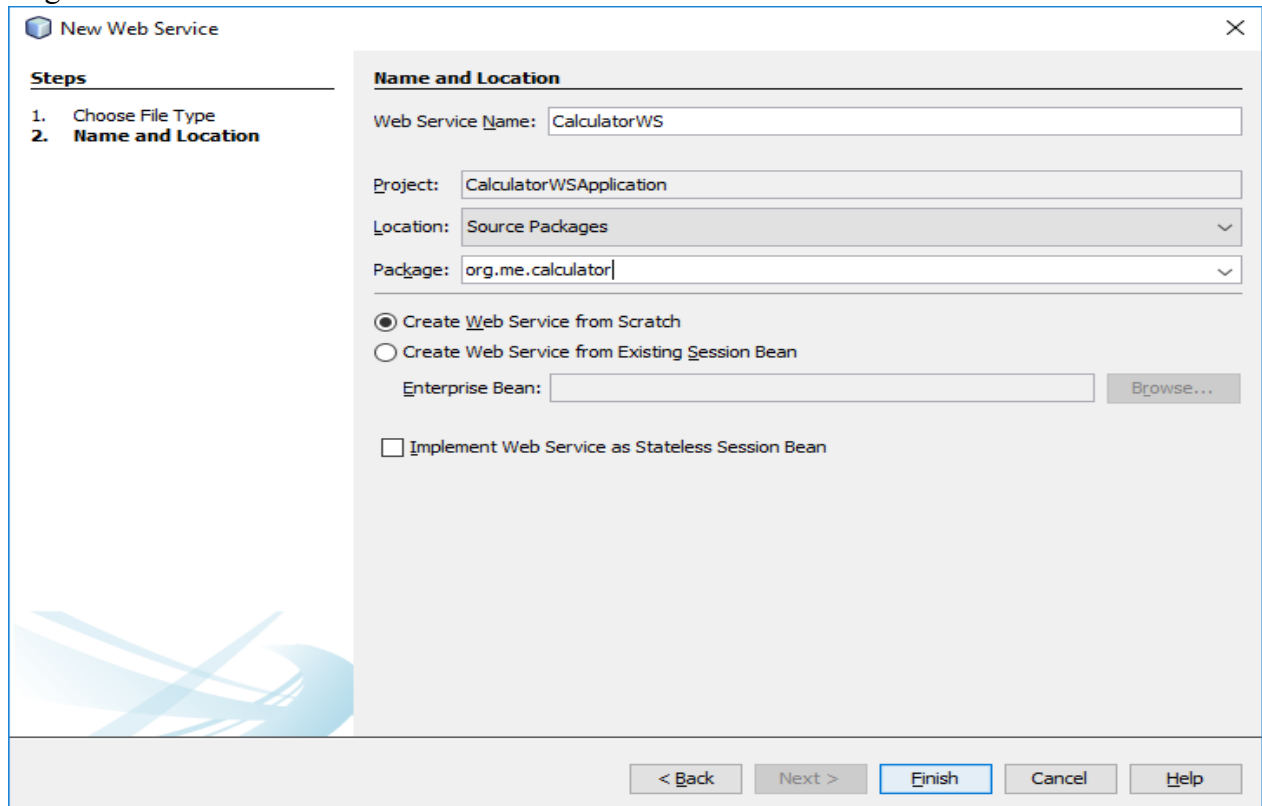
Step 2: Name the project as “CalculatorWSApplication”. Click on Next.



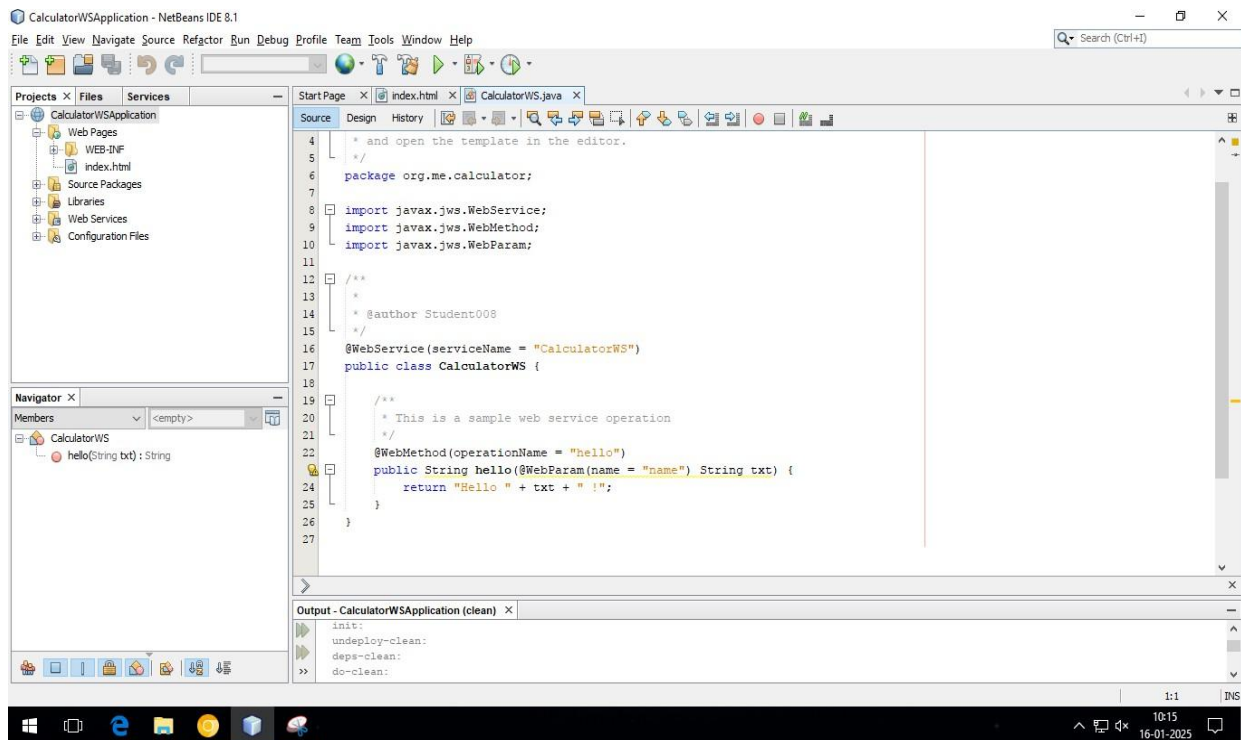
Step 3: A window like this will appear right click on the project that we created New → Web Service.



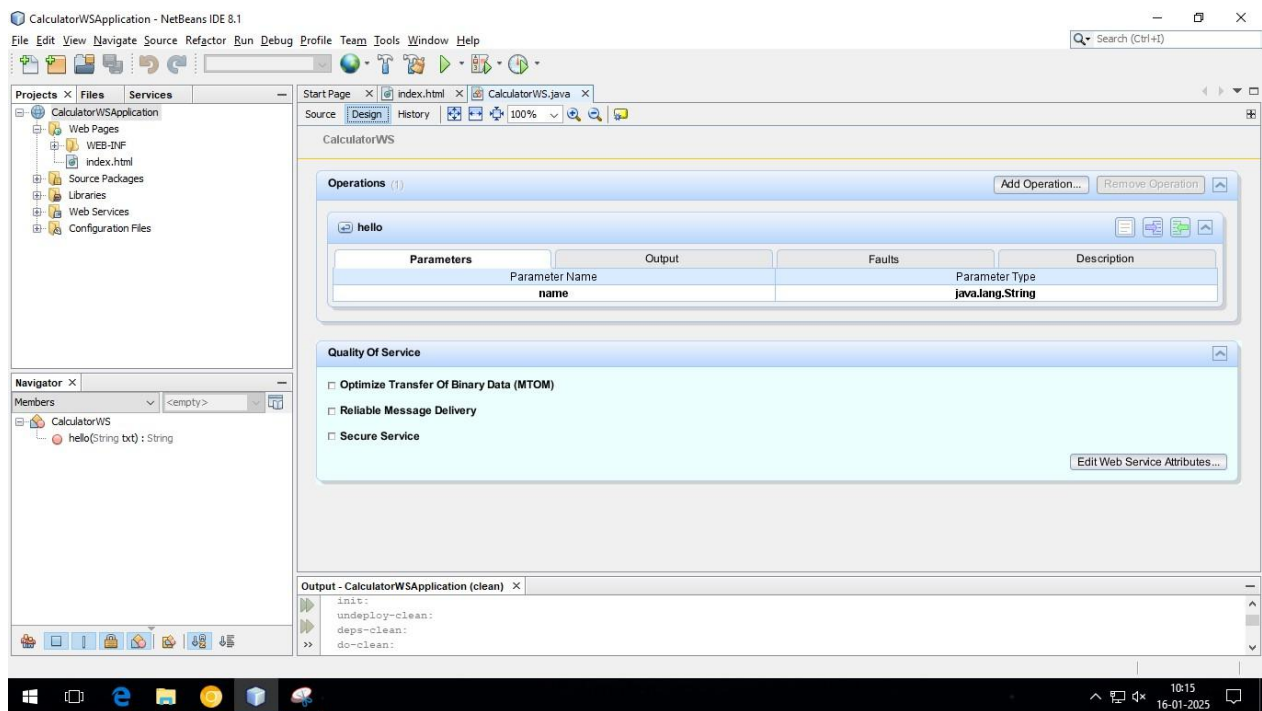
Step 4: Give the name as “CalculatorWS” and also provide a package name as “org.me.calculator”. Click on Finish.



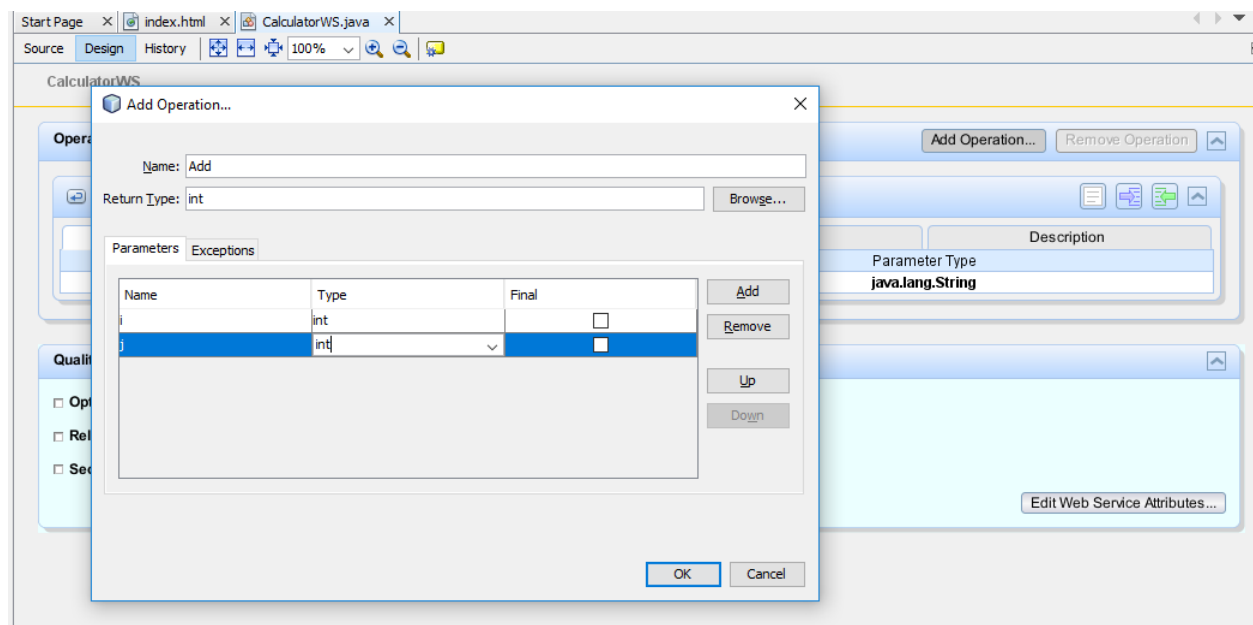
Step 5: Now a window like this will appear along with the source code as below.



Step 6: Go to the Design tab as shown below.



Step 7: In the first dialog box give the Name as “Add” and Return Type as “int”. Then click on the add button to add “i” and “j” parameters and provide the type as “int”. Click on Ok.



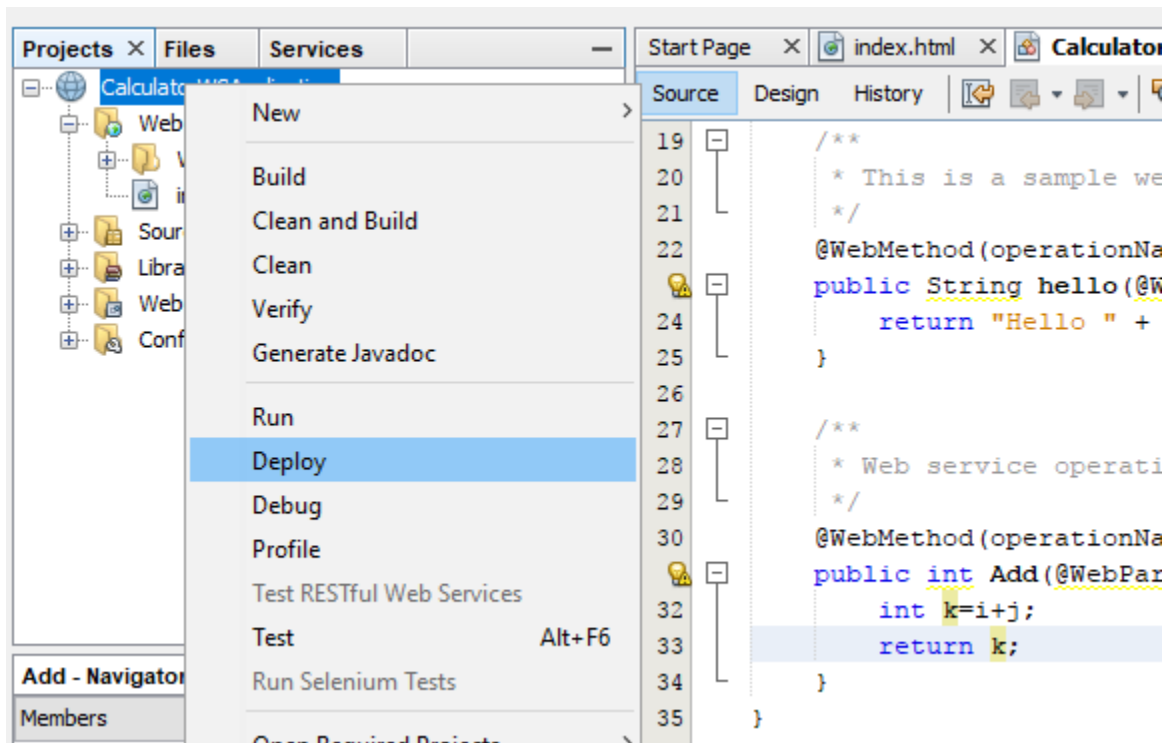
Step 8: The following code will be visible in the source code .

```

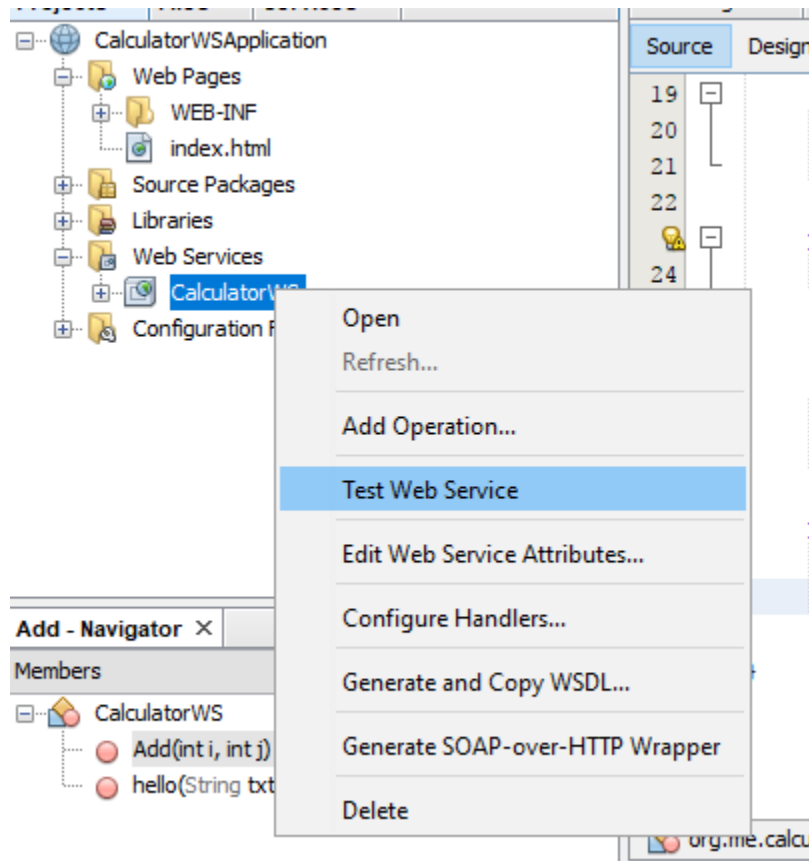
    * Web service operation
    */
    @WebMethod(operationName = "Add")
    public int Add(@WebParam(name = "i") int i, @WebParam(name = "j") int j) {
        int k=i+j;
        return k;
    }
}

```

Step 9: Now Right Click on the project and select “Deploy”.



Step 10: Now go to the project tab→web services→CalculatorWS. Right click on it and select “Test Web Services”.



Step 11: A window like this will appear provide the numbers as “3” and “4” click on the add button.

CalculatorWS 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 int org.me.calculator.CalculatorWS.add(int,int)
```

add (3 4 x)

```
public abstract java.lang.String org.me.calculator.CalculatorWS.hello(java.lang.String)
```

hello ()

Step 12: Here we can see the method returned as “int 7”.

add Method invocation

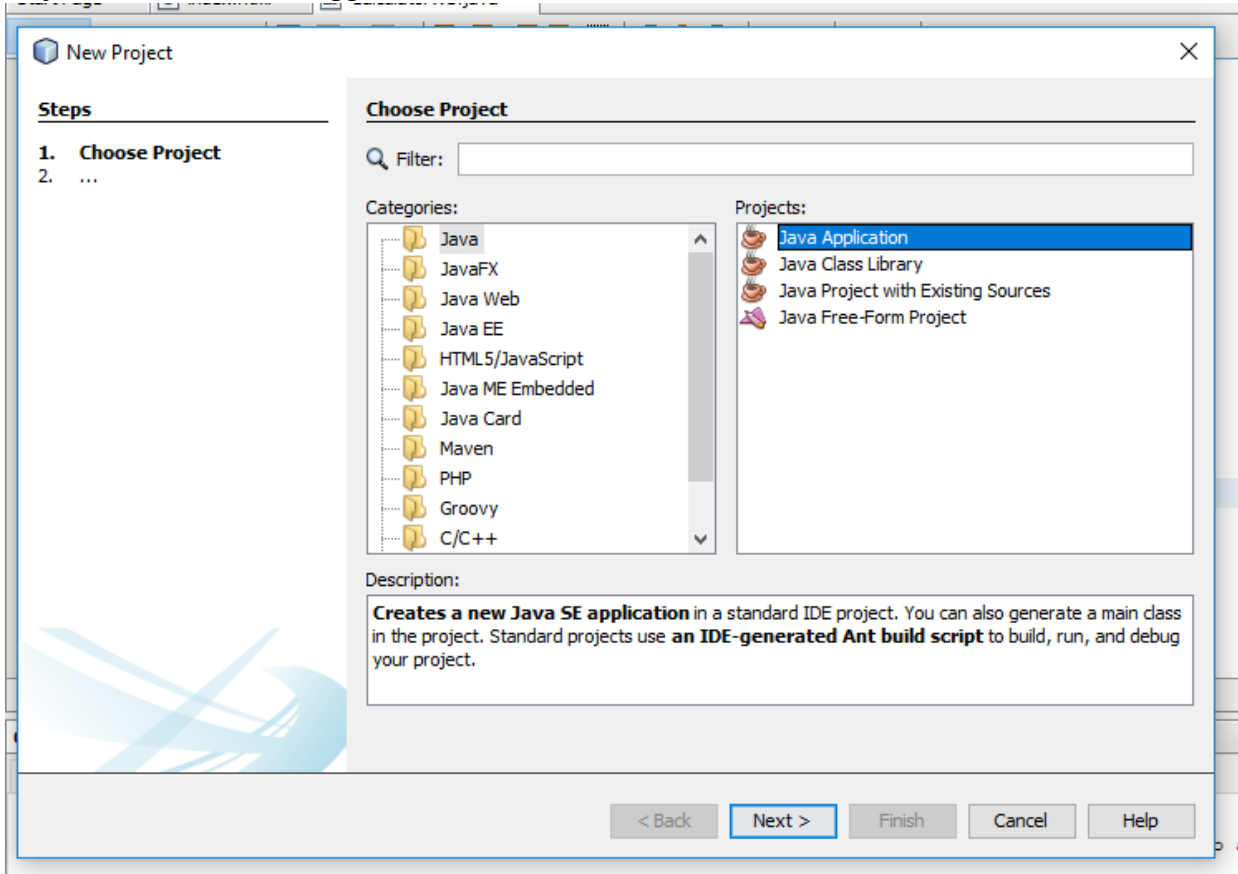
Method parameter(s)

Type	Value
int	3
int	4

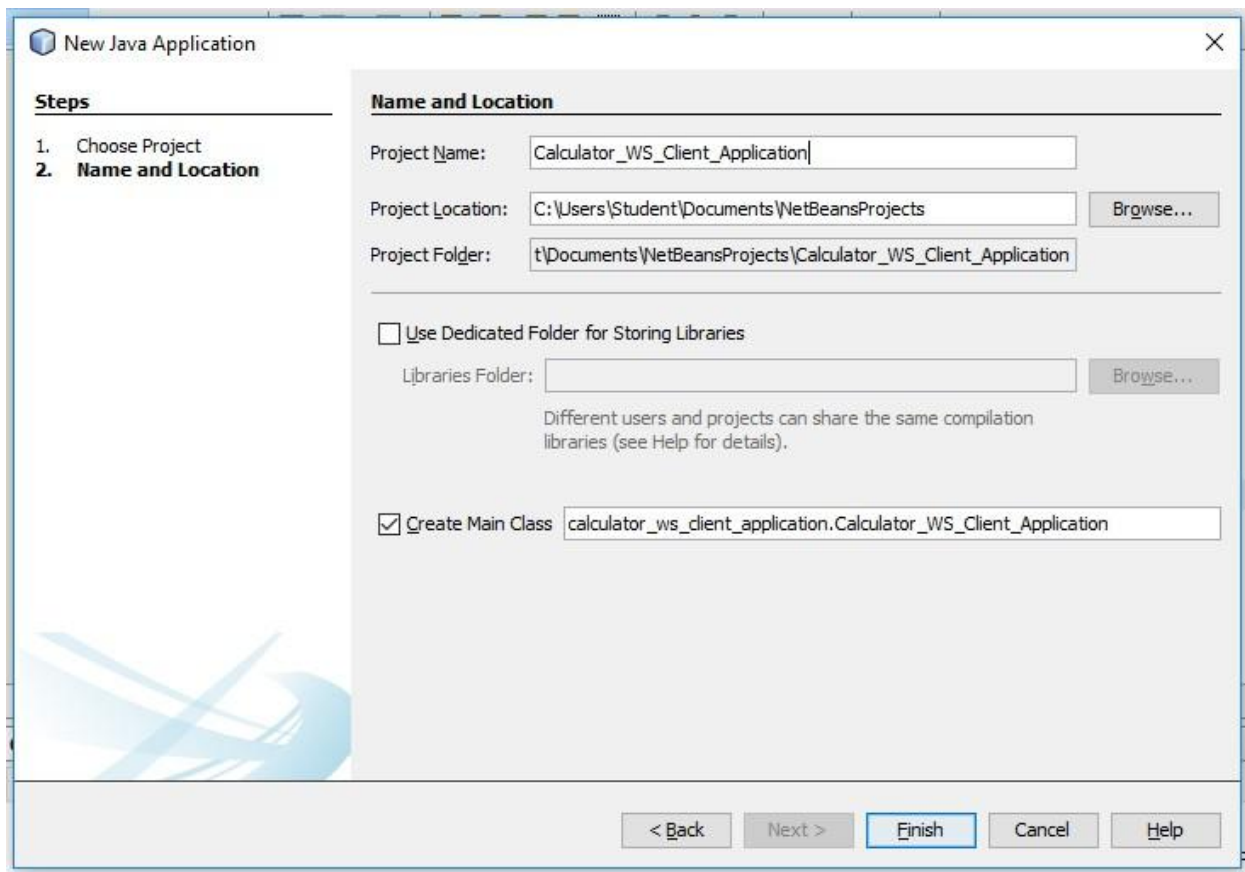
Method returned

int : "7"

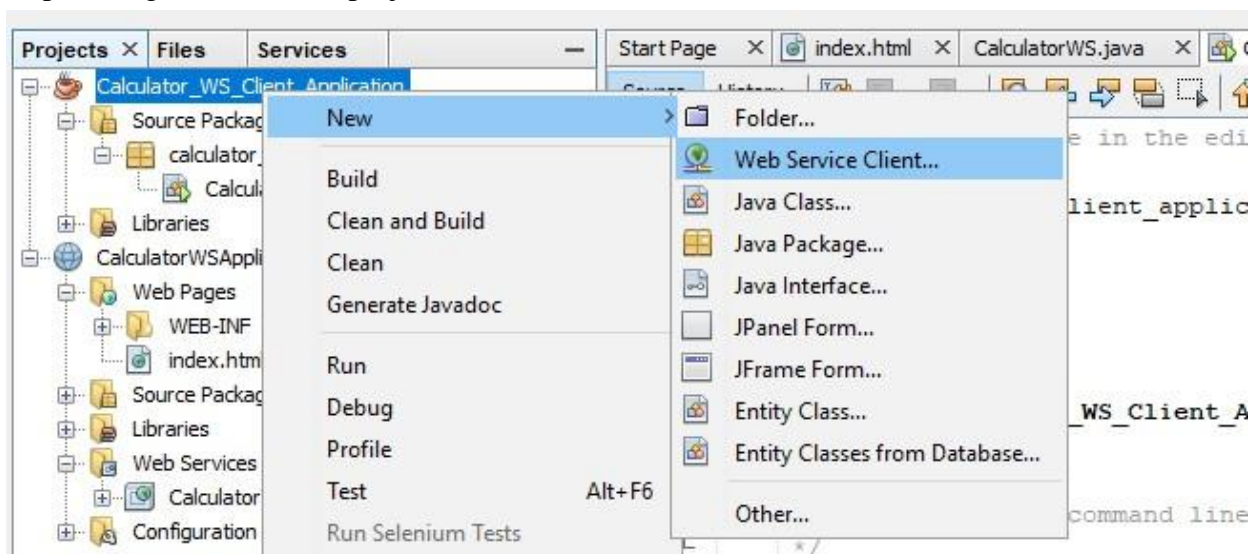
Step 13: Now we have to create a client for consuming the web services . Click on File → New Project → Java → Java Application.



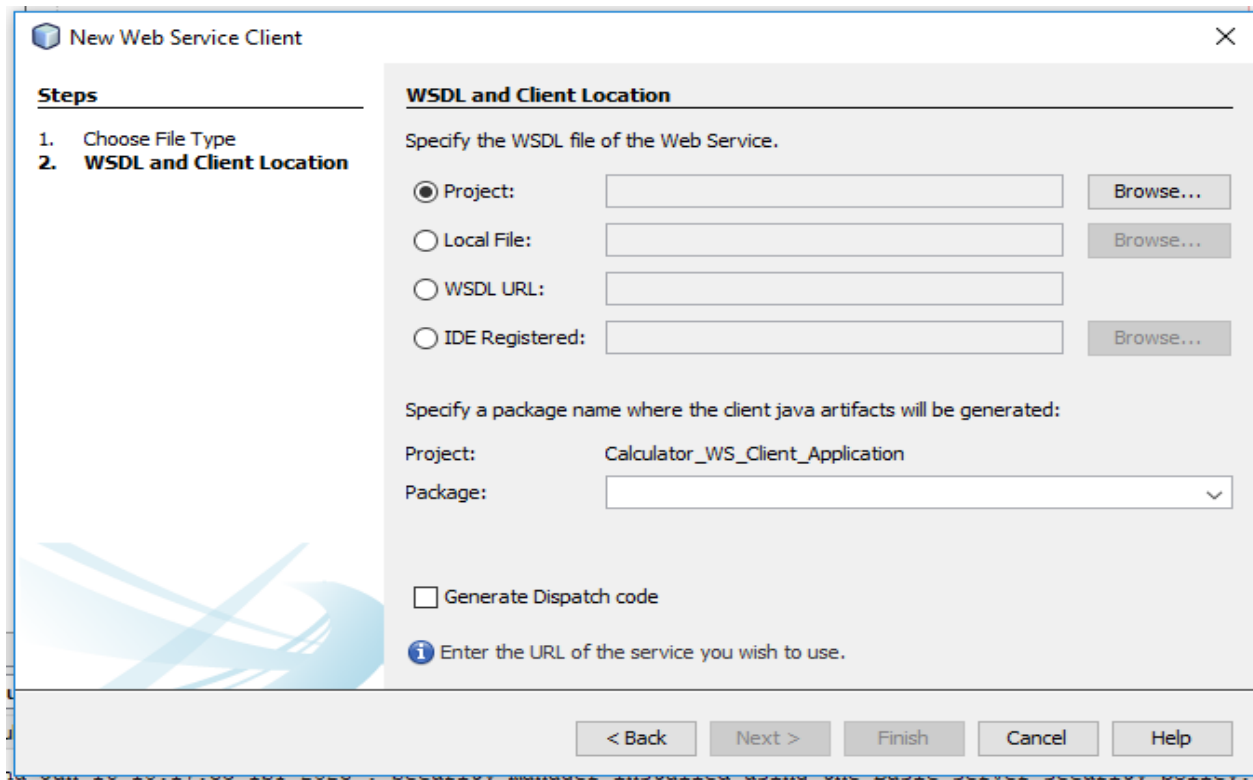
Step 14: Give the Name as “Calculator_WS_Client_Application. Click on Finish.



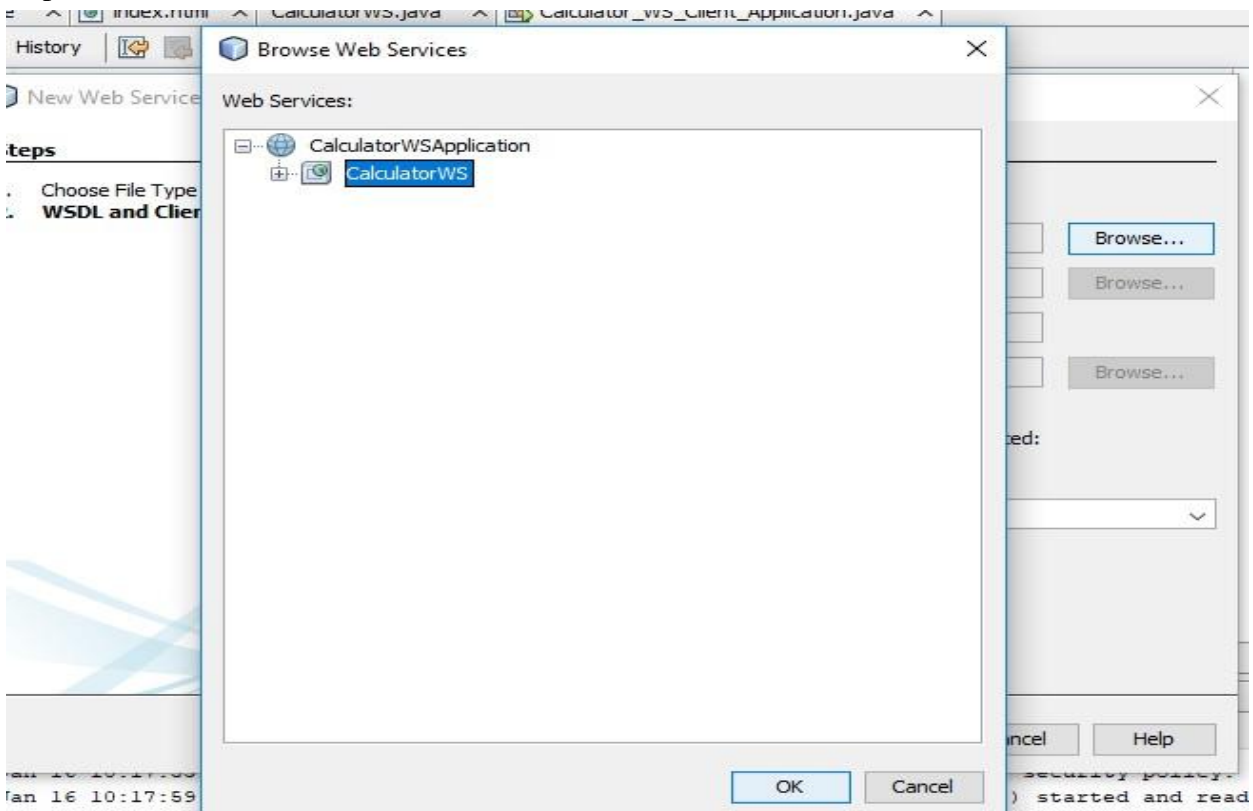
Step 15: Right click on the project New → Web Service → Client.



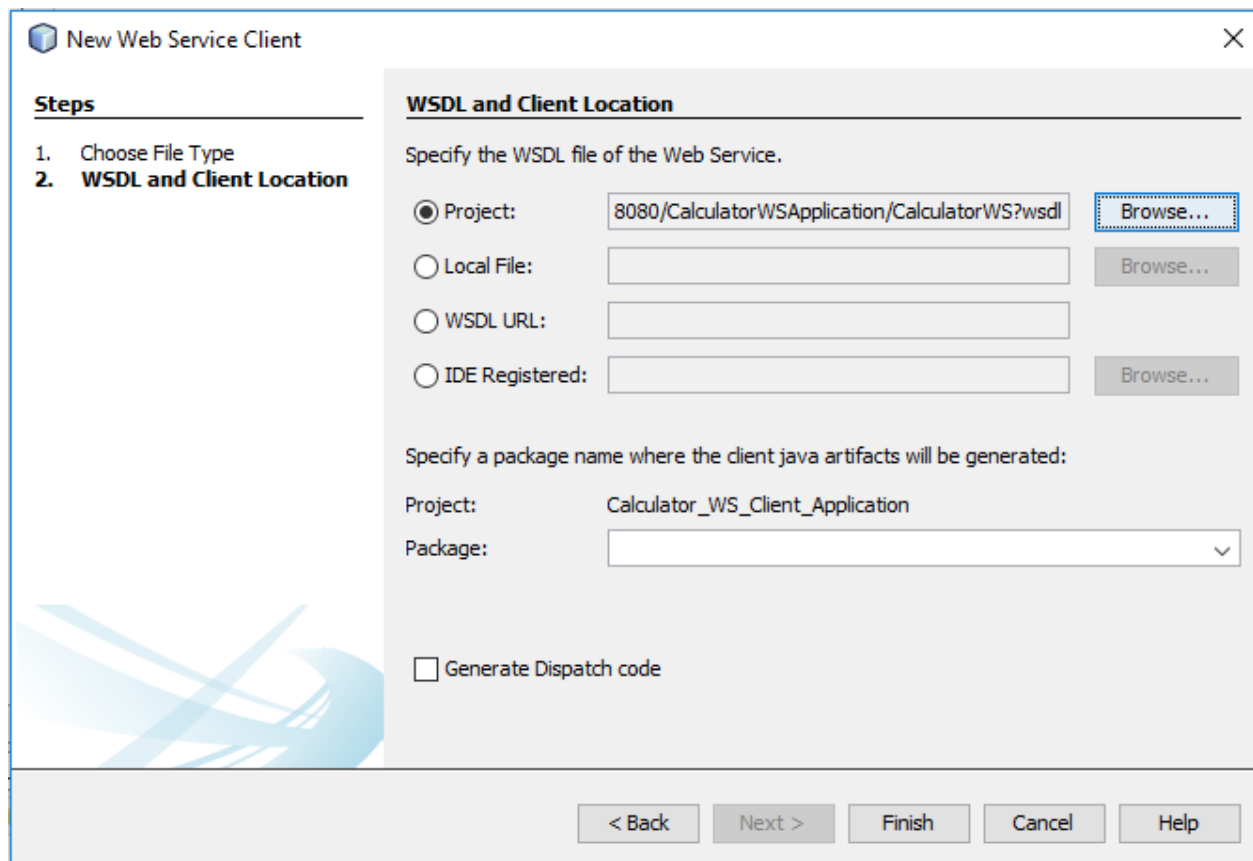
Step 16: Select project as the WSDL File of the Web Service and click on Browse.



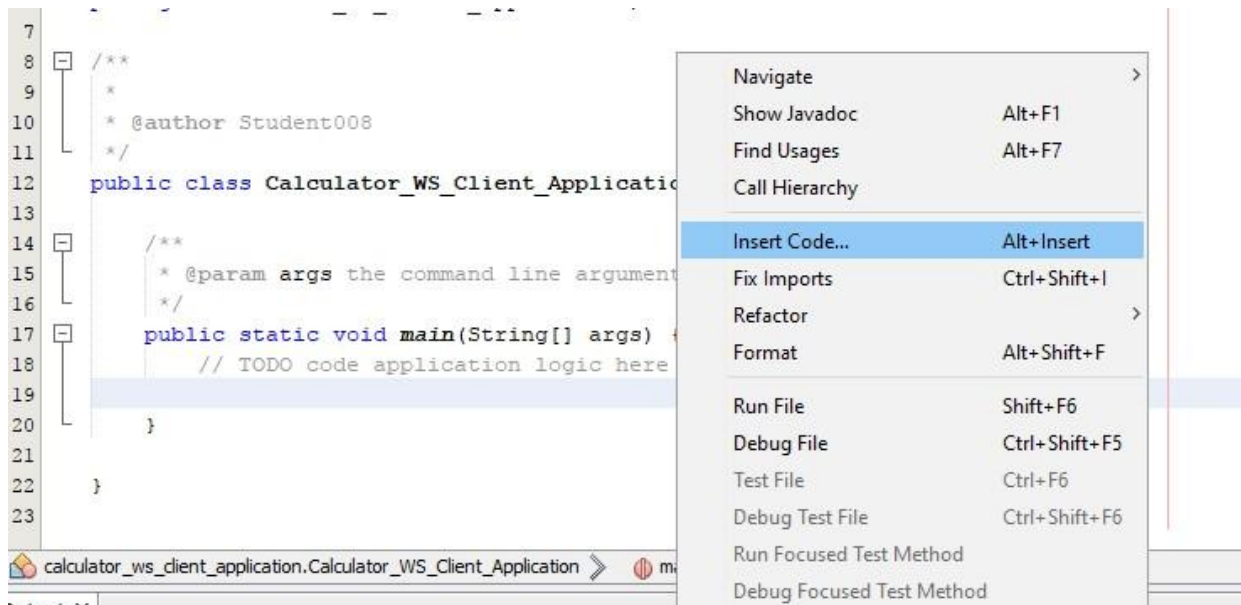
Step 17: Select CalculatorWS. Click on Ok.



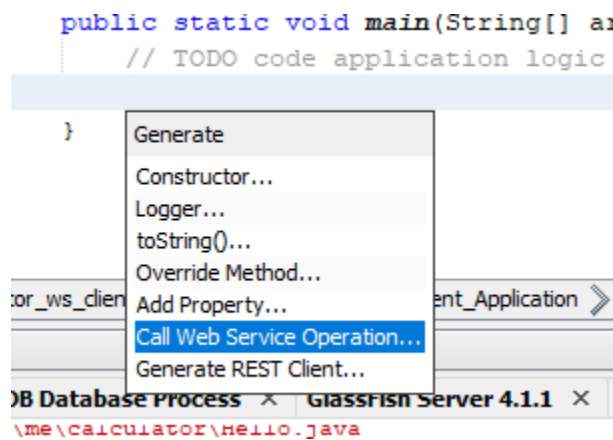
Step 18: A window like this will appear .



Step 17: Right click on the main method area and select “Insert Code”.

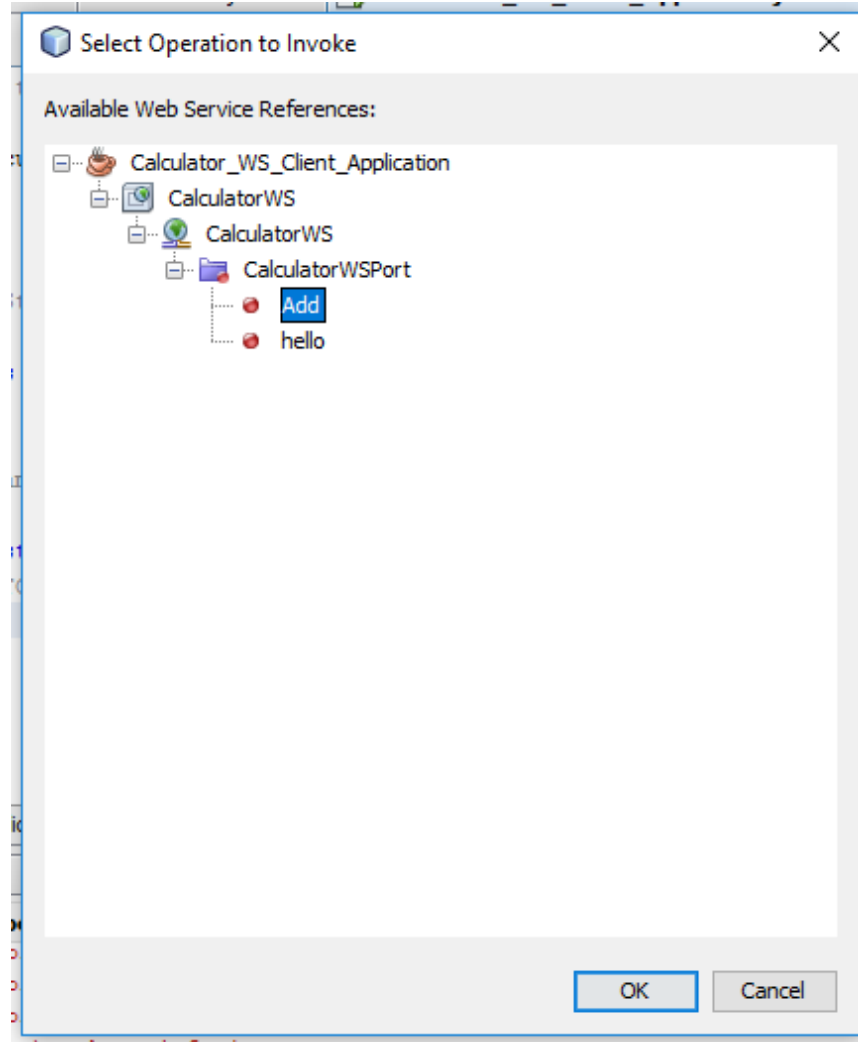


Step 18: Select on the “Call Web Service Operation”.



Step 19: Select

Calculator_WS_Client_Application→CalculatorWS→CalculatorWSPort→Add.Click on Ok.



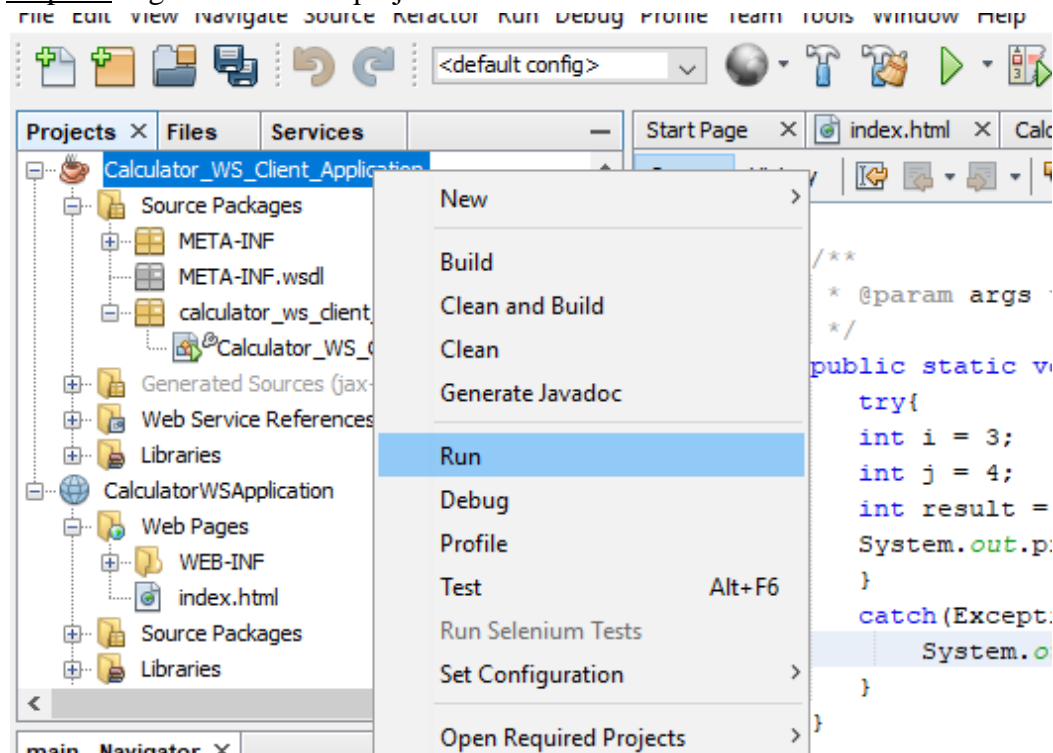
Step 20: Type the following code as shown below.

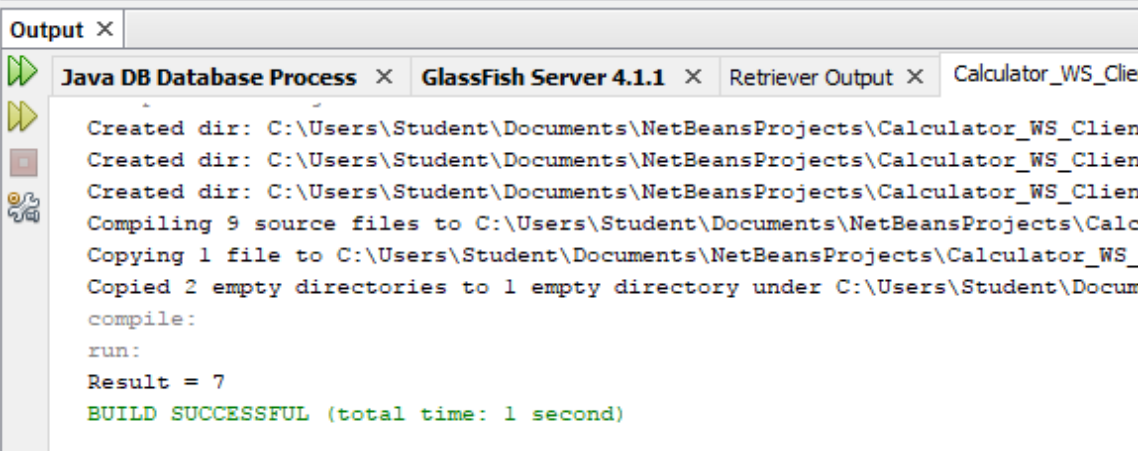
```

13
14 /**
15  * @param args the command line arguments
16  */
17 public static void main(String[] args) {
18     try{
19         int i = 3;
20         int j = 4;
21         int result = add(i,j);
22         System.out.println("Result = "+ result);
23     }
24     catch(Exception ex){
25         System.out.println("Exception: "+ex);
26     }
27 }
28
29 private static int add(int i, int j) {
30     org.me.calculator.CalculatorWS_Service service = new org.me.calculator.CalculatorWS_Service();
31     org.me.calculator.CalculatorWS port = service.getCalculatorWSPort();
32     return port.add(i, j);
33 }
34
35 }
36

```

Step 21: Right click on the project→Run.



OUTPUT:

The screenshot shows the NetBeans IDE's Output window. The title bar reads 'Output X'. Below the title bar, there are four tabs: 'Java DB Database Process X', 'GlassFish Server 4.1.1 X', 'Retriever Output X', and 'Calculator_WS_Client'. The 'Calculator_WS_Client' tab is active, displaying the following output text:

```
Created dir: C:\Users\Student\Documents\NetBeansProjects\Calculator_WS_Client
Created dir: C:\Users\Student\Documents\NetBeansProjects\Calculator_WS_Client
Created dir: C:\Users\Student\Documents\NetBeansProjects\Calculator_WS_Client
Compiling 9 source files to C:\Users\Student\Documents\NetBeansProjects\Calculator_WS_Client
Copying 1 file to C:\Users\Student\Documents\NetBeansProjects\Calculator_WS_Client
Copied 2 empty directories to 1 empty directory under C:\Users\Student\Documents\NetBeansProjects\Calculator_WS_Client
compile:
run:
Result = 7
BUILD SUCCESSFUL (total time: 1 second)
```