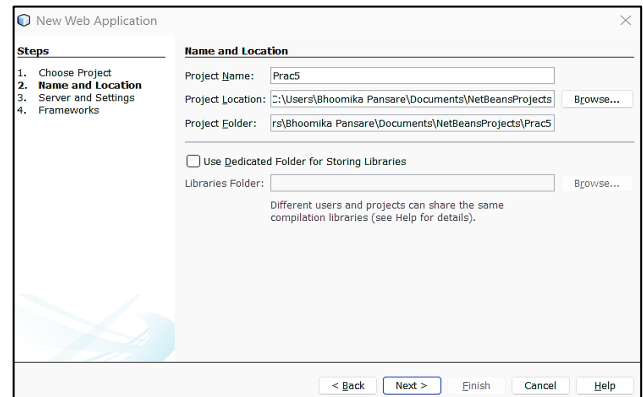
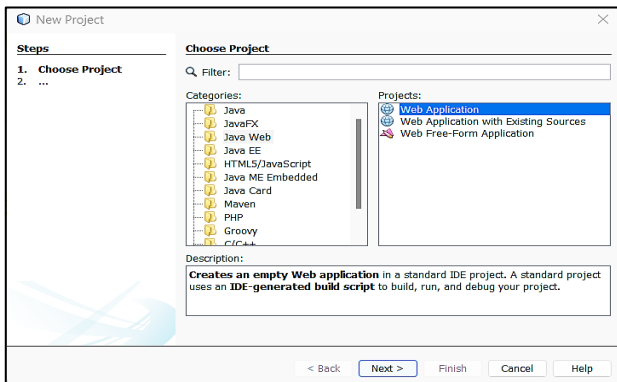


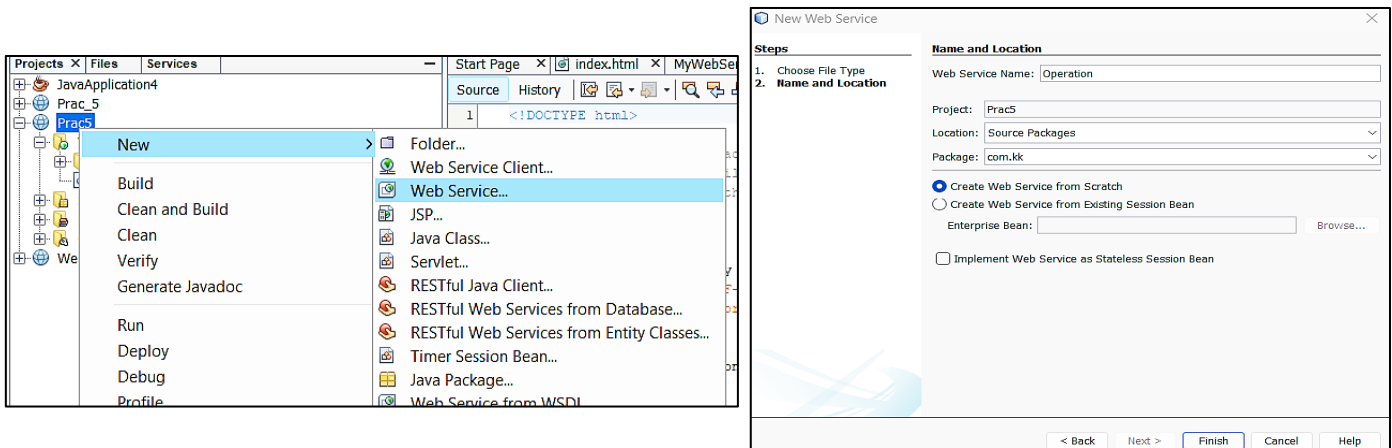
Practical No: 4

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

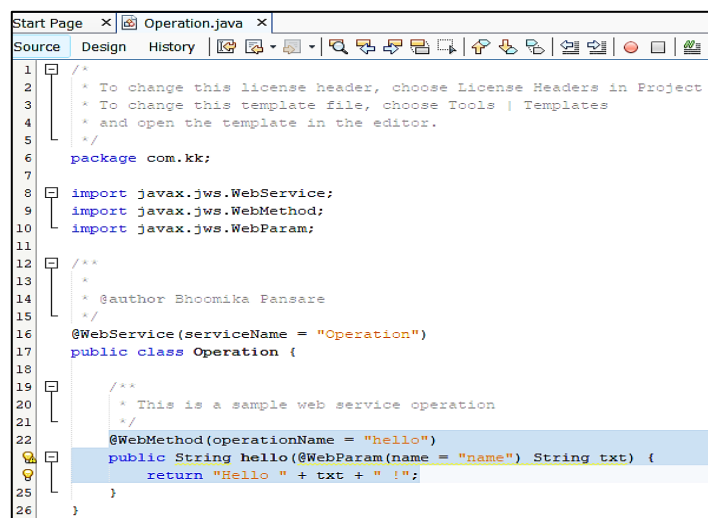
Step 1: Create a new project. Click on web application → Name of project → Next → Finish.



Step 2: Once your project is successfully created. Expand your project, right click on project. Select New → Web Service. Enter the name of new Web Service as operation & package name as **com.kk**. Click on Finish options.



Step 3: Once your web service Created. Remove the selected part from Operation.java



Step 4: Add Operations. The first operation is Addition.

Operation name is Add. The return type will be integer. Add some parameter.

Now here I am taking two parameters.

Parameter name is a & type will be integer.

Parameter name is b & type will be integer. After that, click on OK button.

Add Operation

Name: Add

Return Type: int Browse...

Parameters Exceptions

Name	Type	Final
a	int	<input type="checkbox"/>
b	int	<input type="checkbox"/>

Add Remove Up Down

Step 5: Now the declaration of variable.

Int c = a+b;

Return c;

```
4  /** @author Bhoomika Fansare
5  */
6  @WebService(serviceName = "Operation")
7  public class Operation {
8
9      /**
10       * Web service operation
11       */
12      @WebMethod(operationName = "Add")
13      public int Add(@WebParam(name = "a") int a, @WebParam(name = "b") int b) {
14          int c = a+b;
15          return c;
16      }
17
18      /**
```

Step 6: Now repeat, the Step 4 & 5 for creation of Subtraction, multiplication & division Operation.

Add Operation

Name: Sub

Return Type: int Browse...

Parameters Exceptions

Name	Type	Final
a	int	<input type="checkbox"/>
b	int	<input type="checkbox"/>

Add Remove Up Down

OK Cancel

Add Operation

Name:

Return Type:

Parameters Exceptions

Name	Type	Final
a	int	<input type="checkbox"/>
b	int	<input type="checkbox"/>

Add Operation

Name:

Return Type:

Parameters Exceptions

Name	Type	Final
a	int	<input type="checkbox"/>
b	int	<input type="checkbox"/>

Step 7: Now, write the following code for subtraction, multiplication & division method.

Subtraction method:

int c = a-b;

return c;

Multiplication method:

int c = a*b;

return c;

Division method:

int c = a/b;

return c;

```

30  |
31  |
32  |
33  |
34  |
35  |
36  |
37  |
38  |
39  |
40  |
41  |
42  |
43  |
44  |
45  |
46  |
47  |
48  |
49  |
50  |
51  |
52  |
53  |
54  |
55  |

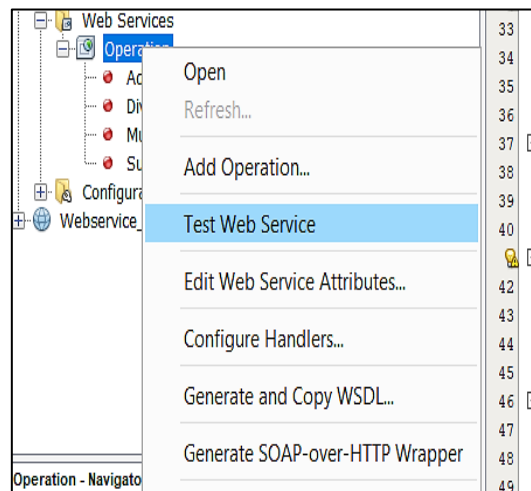
    /**
    * Web service operation
    */
    @WebMethod(operationName = "Sub")
    public int Sub(@WebParam(name = "a") int a, @WebParam(name = "b") int b) {
        int c = a-b;
        return c;
    }

    /**
    * Web service operation
    */
    @WebMethod(operationName = "Multi")
    public int Multi(@WebParam(name = "a") int a, @WebParam(name = "b") int b) {
        int c = a*b;
        return c;
    }

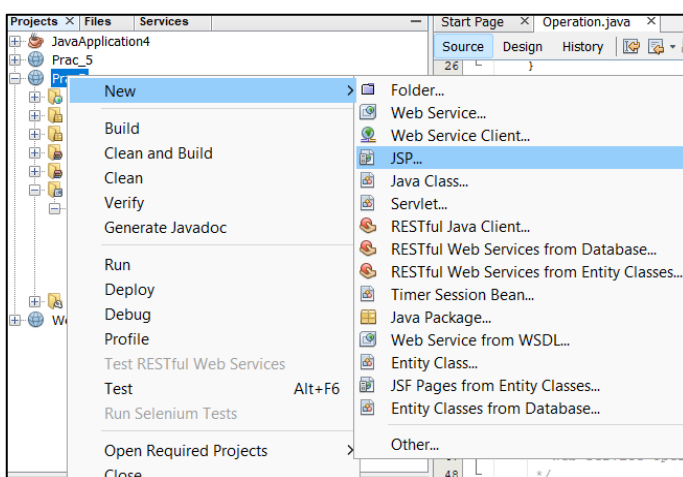
    /**
    * Web service operation
    */
    @WebMethod(operationName = "Div")
    public int Div(@WebParam(name = "a") int a, @WebParam(name = "b") int b) {
        int c = a/b;
        return c;
    }

```

Step 8: Now test your web service.

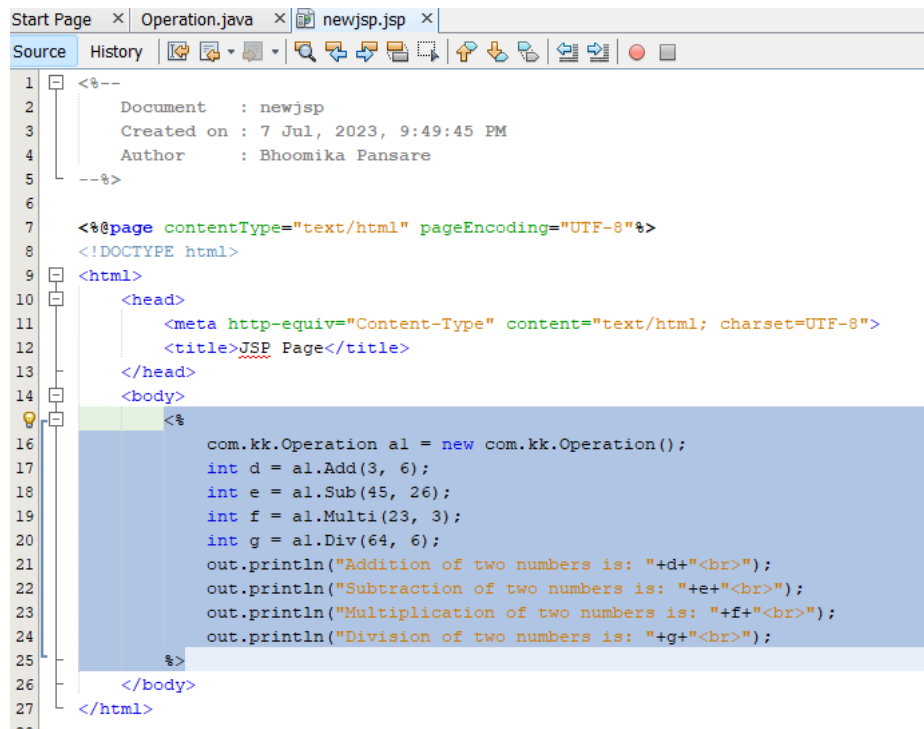
A screenshot of a web browser displaying 'Operation Web Service Tester'. The page has a title bar 'Operation Web Service Tester' and a URL 'localhost:8080/Prac5/Operation?Tester'. The main content area has the title 'Operation Web Service Tester' and a description: 'This form will allow you to test your web service implementation (WSDL File)'. Below this, it says 'To invoke an operation, fill the method parameter(s) input boxes and click on the button label'. Under the heading 'Methods :', there are four sections, each with a method signature and an input field: 1. 'public abstract int com.kk.Operation.add(int,int)' with an 'add' button and an input field. 2. 'public abstract int com.kk.Operation.div(int,int)' with a 'div' button and an input field. 3. 'public abstract int com.kk.Operation.sub(int,int)' with a 'sub' button and an input field. 4. 'public abstract int com.kk.Operation.multi(int,int)' with a 'multi' button and an input field.

Step 9: Now right click on project. Select New → JSP page. Give the file name & click on Finish.

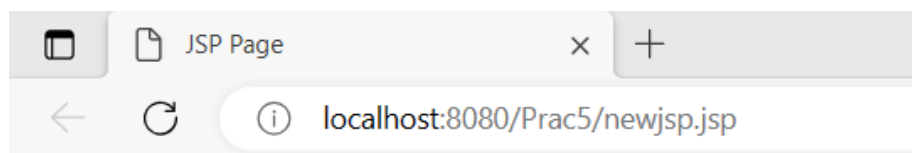
A screenshot of the 'New JSP' dialog box. It has two tabs: 'Steps' and 'Name and Location'. The 'Steps' tab is active, showing a list of steps: 1. Choose File Type, 2. Name and Location. The 'Name and Location' section has fields for 'File Name' (newjsp), 'Project' (Prac5), 'Location' (Web Pages), and 'Folder' (empty). Below these is a 'Browse...' button. The 'Created File' field shows the path 'C:\Users\Shoomika Pansare\Documents\NetBeansProjects\Prac5\web\newjsp.jsp'. There are two options: 'JSP File (Standard Syntax)' (selected) and 'JSP Document (XML Syntax)'. Below the options is a 'Description' field with the text 'A JSP file using JSP standard syntax.' At the bottom are buttons for '< Back', 'Next >', 'Finish', 'Cancel', and 'Help'.

Step 10: Write the following code in JSP page.

```
<%  
  
    com.kk.Operation a1 = new com.kk.Operation();  
    int d = a1.Add(3, 6);  
    int e = a1.Sub(45, 26);  
    int f = a1.Multi(23, 3);  
    int g = a1.Div(64, 6);  
    out.println("Addition of two numbers is: "+d+"<br>");  
    out.println("Subtraction of two numbers is: "+e+"<br>");  
    out.println("Multiplication of two numbers is: "+f+"<br>");  
    out.println("Division of two numbers is: "+g+"<br>");  
  
%>
```

A screenshot of an IDE window titled 'newjsp.jsp'. The code is written in JSP syntax. It starts with a comment block containing document information. Then, it has a page directive: <%@page contentType="text/html" pageEncoding="UTF-8"%>. This is followed by a DOCTYPE declaration and an HTML structure with head and body tags. Inside the body, the Java code from Step 10 is pasted. The code creates an Operation object, performs addition, subtraction, multiplication, and division, and prints the results using out.println. The IDE shows line numbers from 1 to 28 on the left margin.

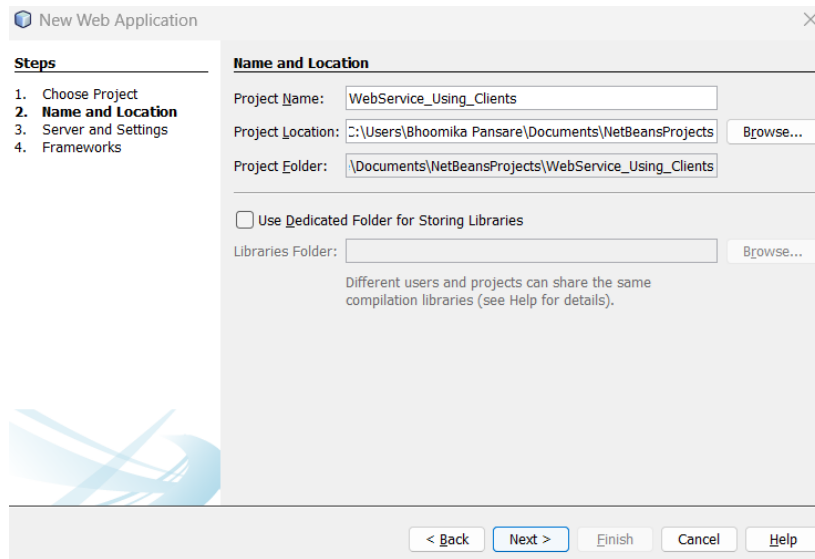
Step 11: Now right click on JSP page page. Click on Run file.



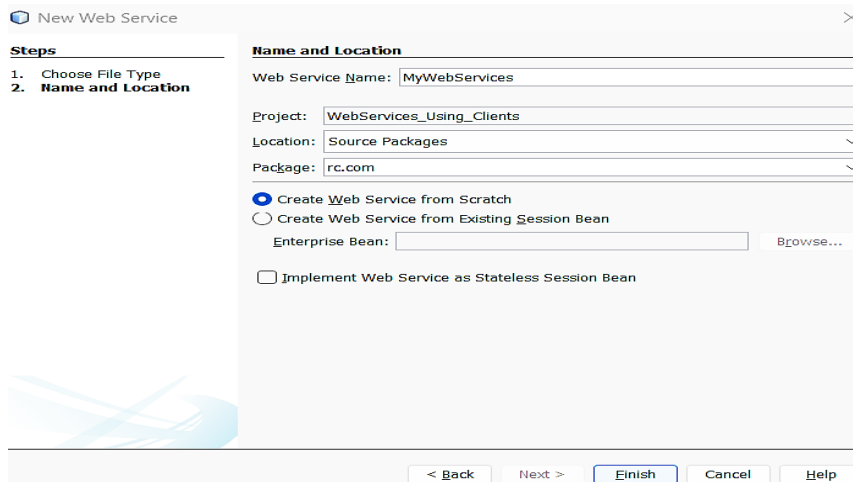
Addition of two numbers is: 9
Subtraction of two numbers is: 19
Multiplication of two numbers is: 69
Division of two numbers is: 10

**Write a JAX-WS web service server & client to perform the following operations.
Define a servlet/ JSP that consumes the web service.**

Step 1: Create a new project. New project → Java Web → Web Application.



Step 2: Right click on newly created project. Create a new web service. The service name is MyWebServices & package name is rc.com.



Step 3: Remove the selected content from MyWebService.java

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

Step 4: Write the following code in deleted content.

```
@WebMethod(operationName = "Square")
```

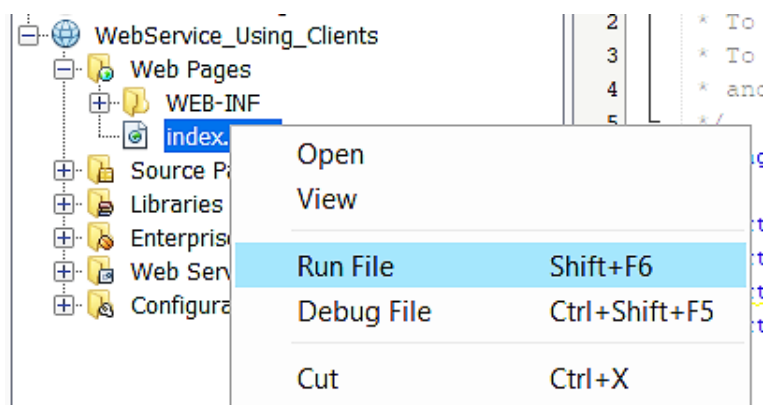
```
public double square(double val) {
```

```
    return val*val;
```

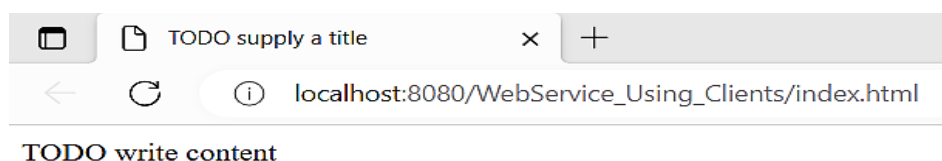
```
}
```

```
21  /**
22   * This is a sample web service operation
23   */
24  @WebMethod(operationName = "Square")
25  public double square(double val) {
26      return val*val;
27  }
28 }
```

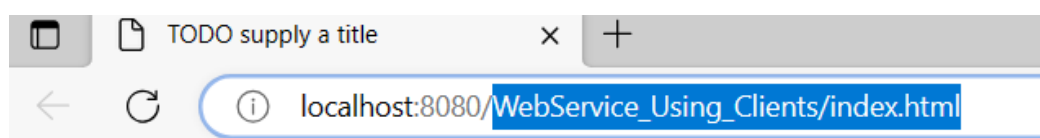
Step 5: Now go to the palette. Right click on index.html in project. Click on run file.



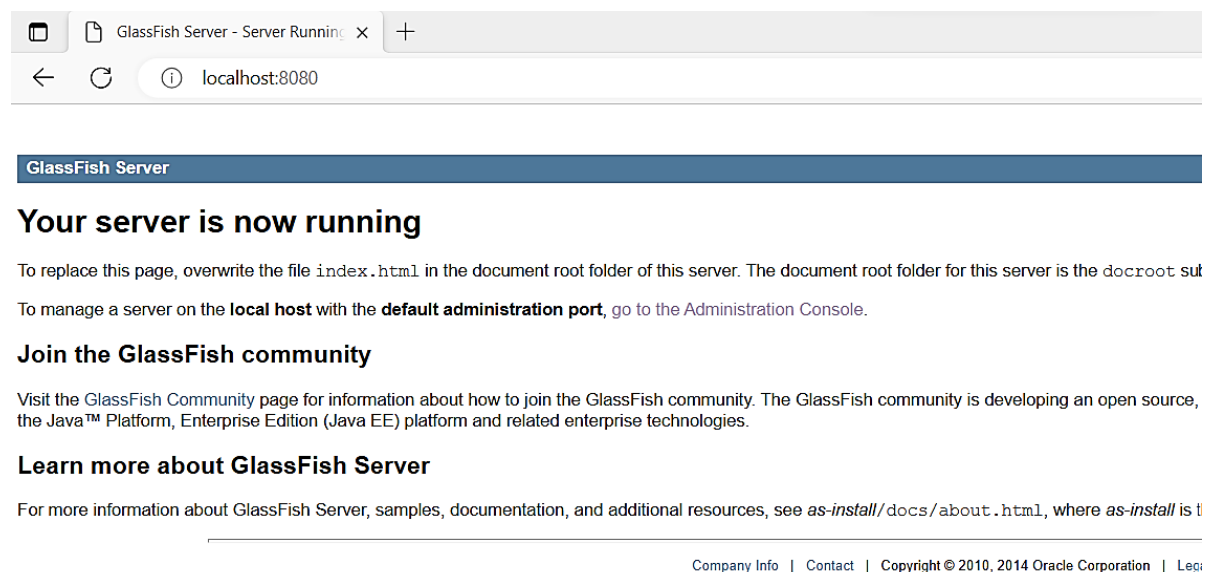
The following window will open after run your index.html file.



Step 6: Remove the selected part from link.



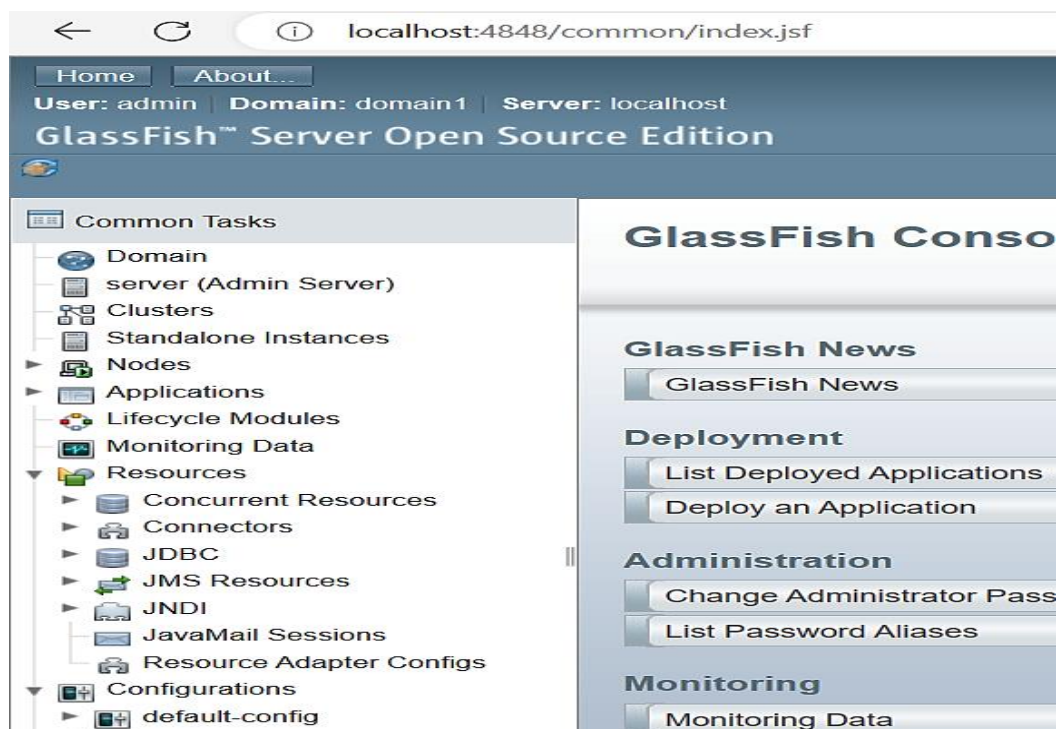
Now, the below window will open.



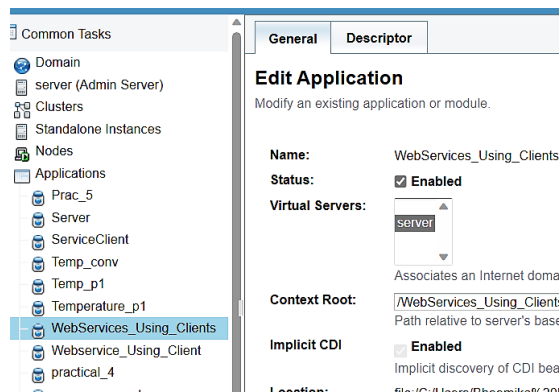
Step 7: Click on go to the administration console.

port, go to the [Administration Console](#).

The following window will open.



Step 8: In left side panel, expand the Application task. Click on your project.



Step 9: Go to the general. Scroll down the page, there is one table which name is Modules & Components. In action column, click on View Endpoint.

Action
Launch
View Endpoint

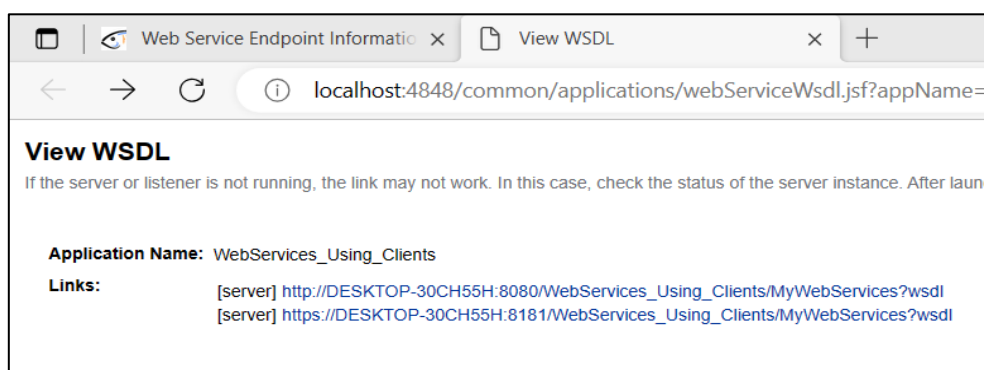
Step 10: Click on [/WebServices_Using_Clients/MyWebServices?wsdl](#) link.

Web Service Endpoint Information

View details about a web service endpoint.

Application Name:	WebServices_Using_Clients
Tester:	/WebServices_Using_Clients/MyWebServices?Tester
WSDL:	/WebServices_Using_Clients/MyWebServices?wsdl
Endpoint Name:	MyWebServices

Step 11: When you click on link. The View WSDL window will open

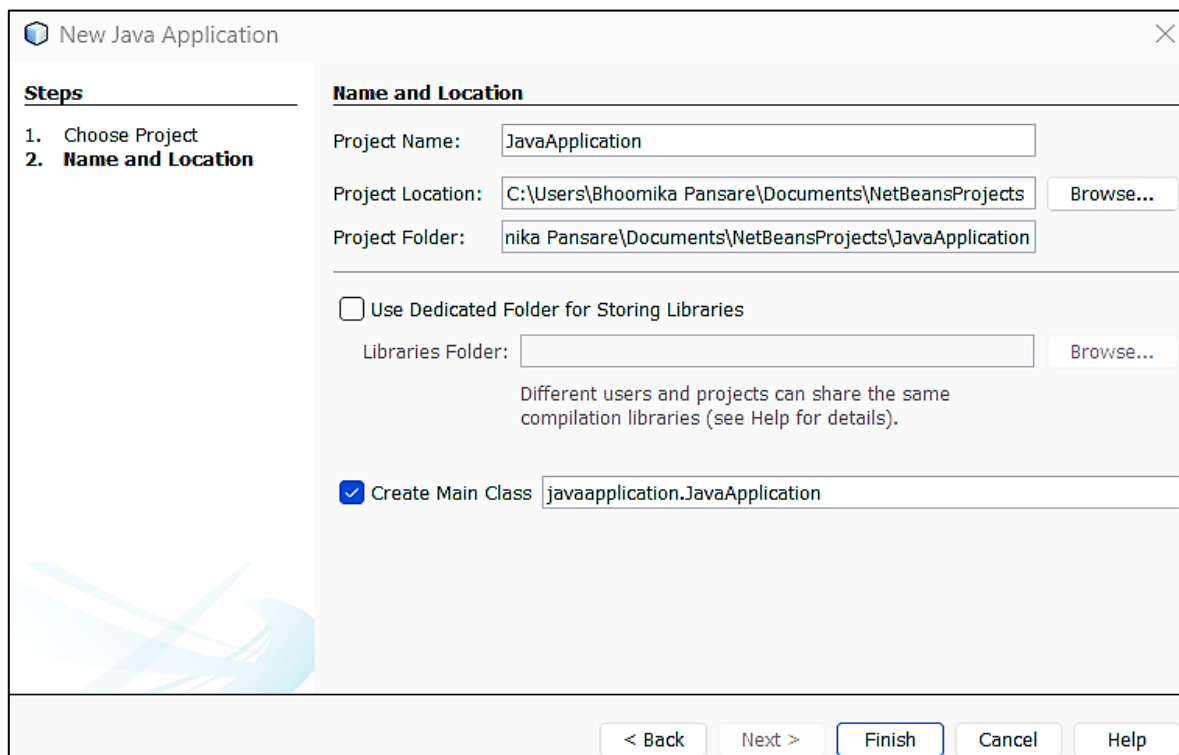
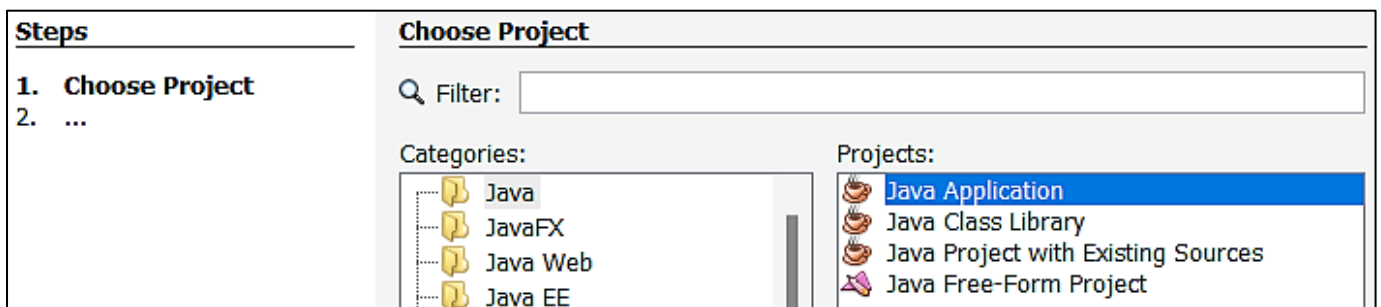


Step 12: Click on first link which will provide the access of server.



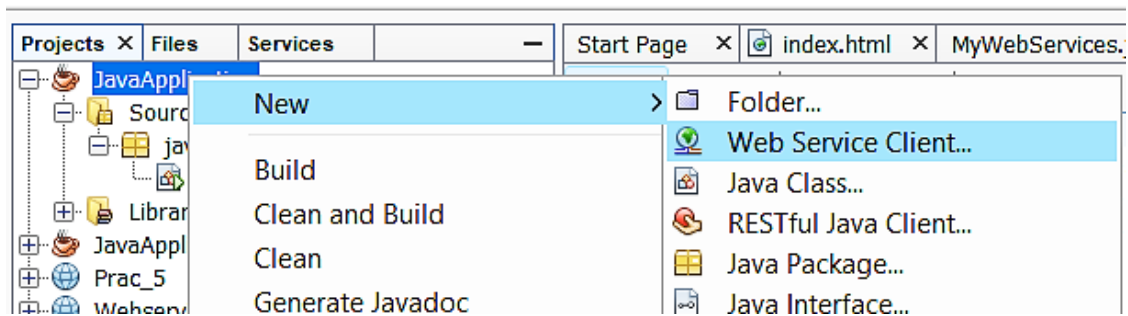
Now minimize the Microsoft edge window.

Step 13: Create a new Java project.

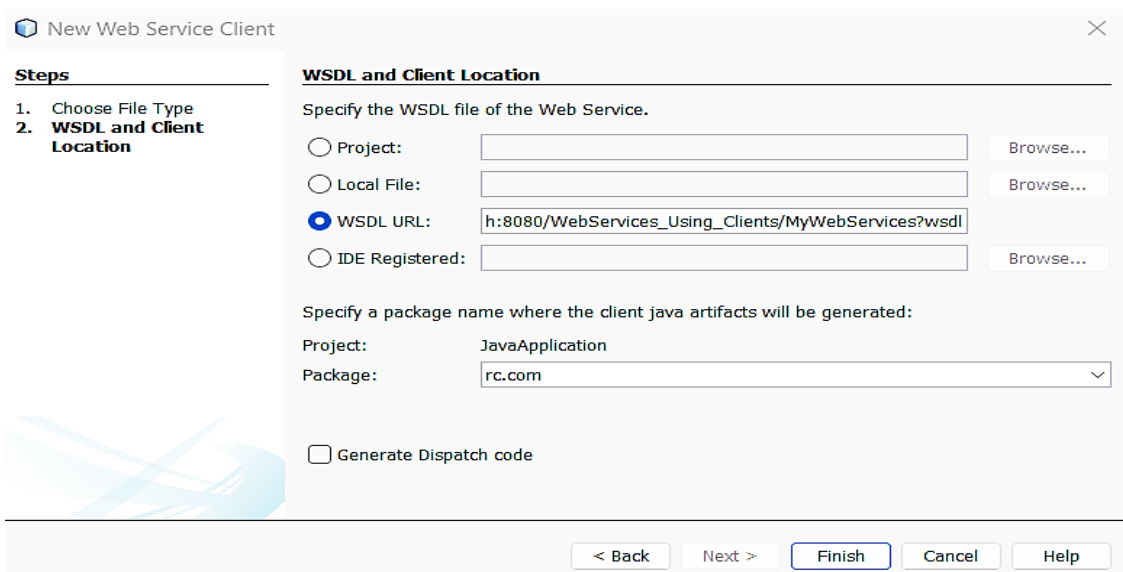


Step 14: In Java Application project, create a new web service client.

Right click on project → New → Web Service Client



Step 15: In new web service client. Select wsdl URL. Paste the copied link.



Step 16: Open JavaApplication.java. Type the following code.

MyWebServices_Service service = new MyWebServices_Service();

final MyWebService proxy = service.getMyWebServicePort();

System.out.println(proxy.square(7));

```
11  L  /*
12  public class JavaApplication {
13
14      /**
15       * @param args the command line arguments
16       */
17      public static void main(String[] args) {
18          MyWebServices_Service service = new MyWebServices_Service();
19          final MyWebService proxy = service.getMyWebServicePort();
20          System.out.println(proxy.square(7));
21      }
22  }
23
24
```

Step 17: Now import the service packages.

```
3      * To change this template file, choose Tools | Templ
4      * and open the template in the editor.
5      */
6      package javaapplication;
7
8      import rc.com.MyWebServices;
9      import rc.com.MyWebServices_Service;
10
11     /**
12     *
13     * @author Bhoomika Pansare
14     */
15     public class JavaApplication {
16
17     /**
```

Step 18: Run the Java application project.

Notifications	Output X
Java DB Database Process X	GlassFish Server 4.1.1 X
Retriever Output X	JavaApplication (run) X
<pre>wsimport-client-MyWebServices: files are up to date wsimport-client-generate: Compiling 1 source file to C:\Users\Bhoomika Pansare\Documents\NetBeansProjects\JavaApplication\build\classes compile: run: 49.0 BUILD SUCCESSFUL (total time: 0 seconds)</pre>	