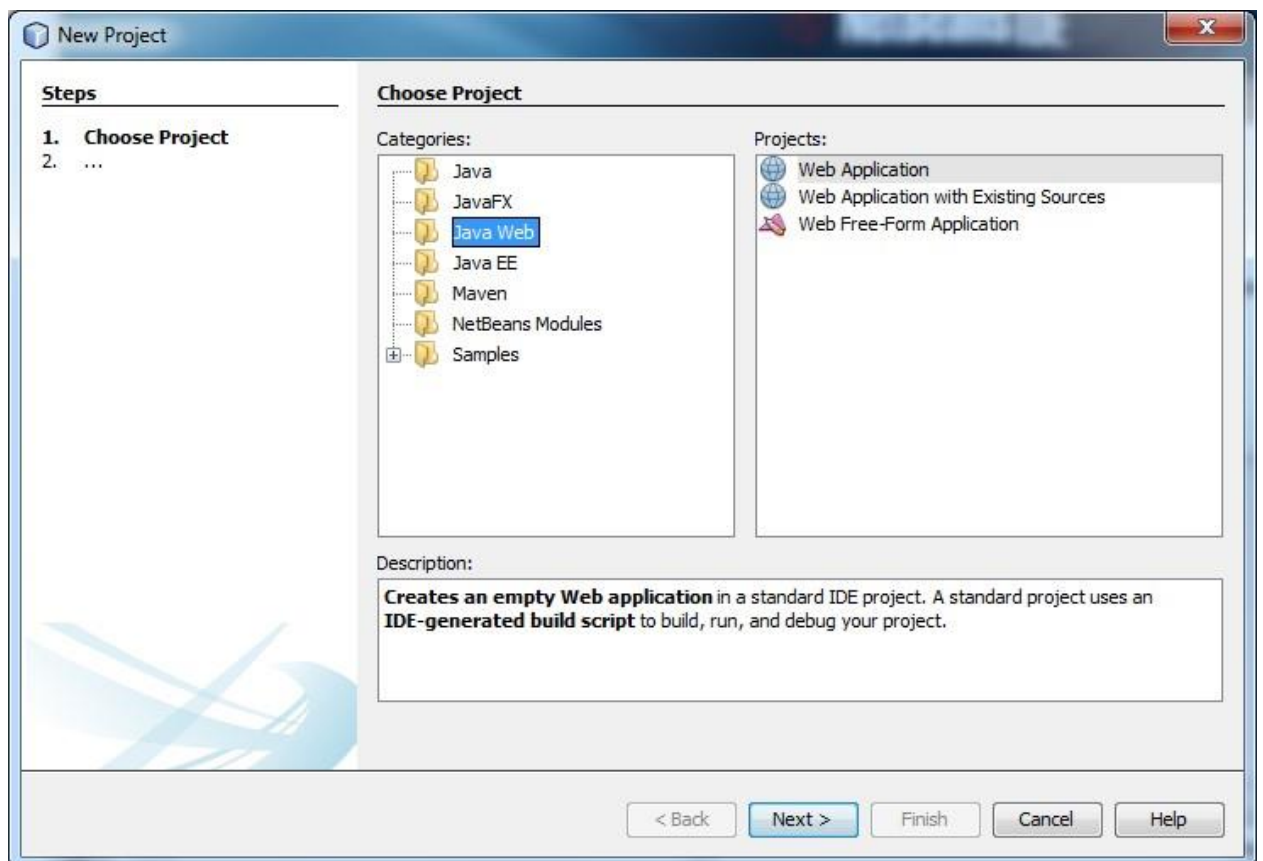
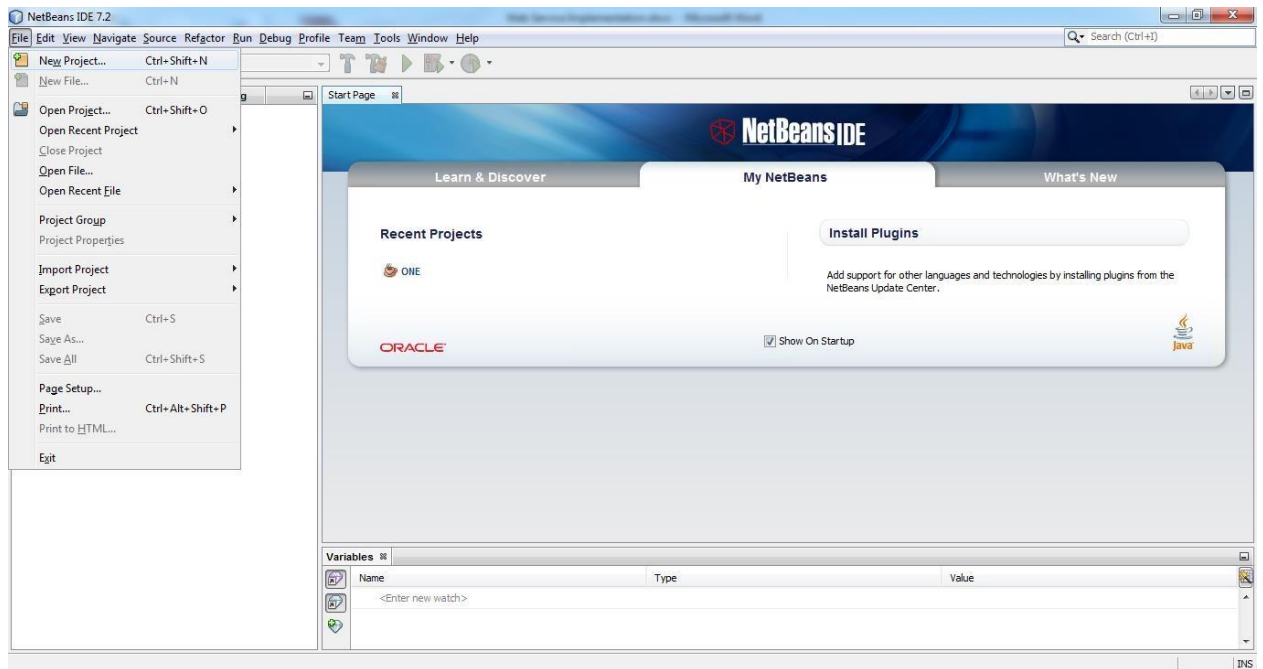


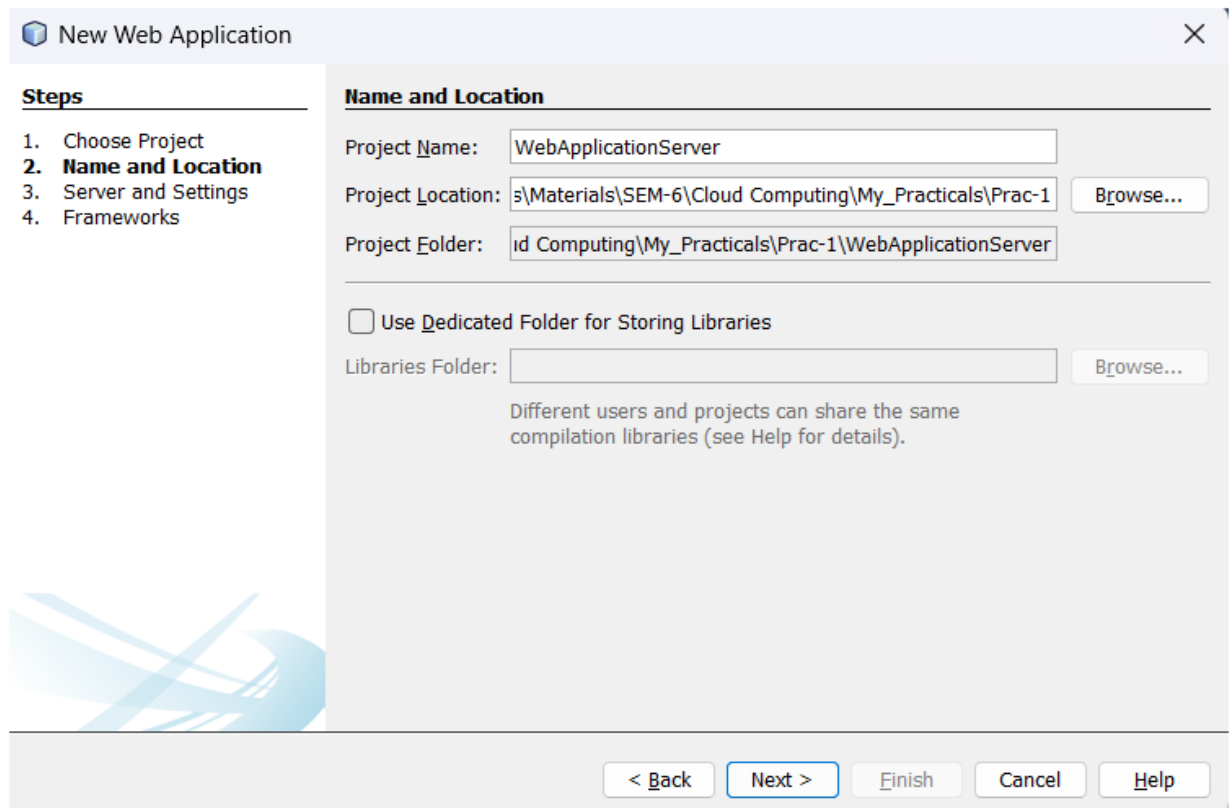
# Practical 1 Web Service Implementation

## 1) Create Web Application in NetBeans

**File > New Project > Java Web > Web Application**



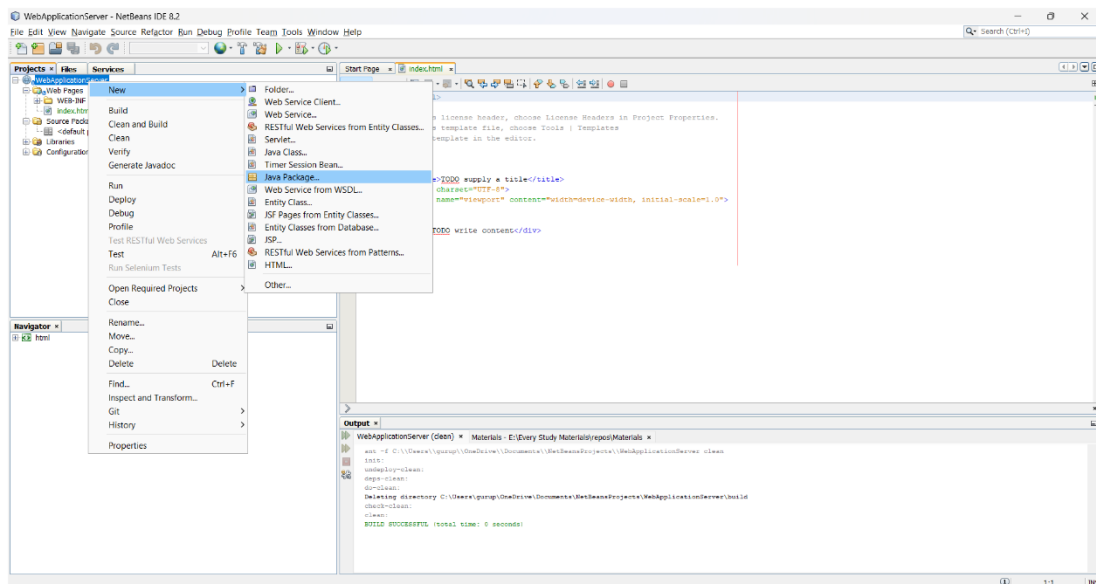
# Practical 1 Web Service Implementation



*Click Next and Finish.*

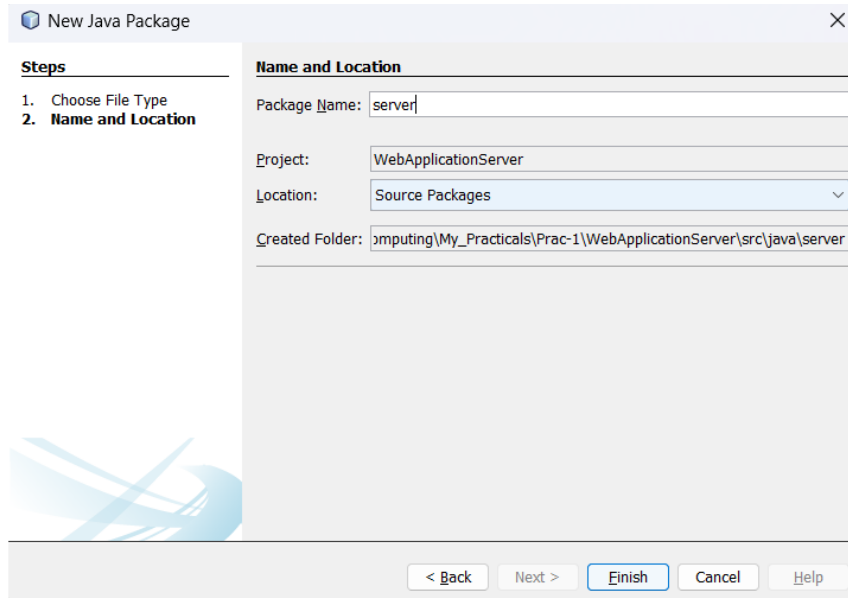
## 2) Create Java Packages from

*Web Application > Source Packages > New > Java Package*



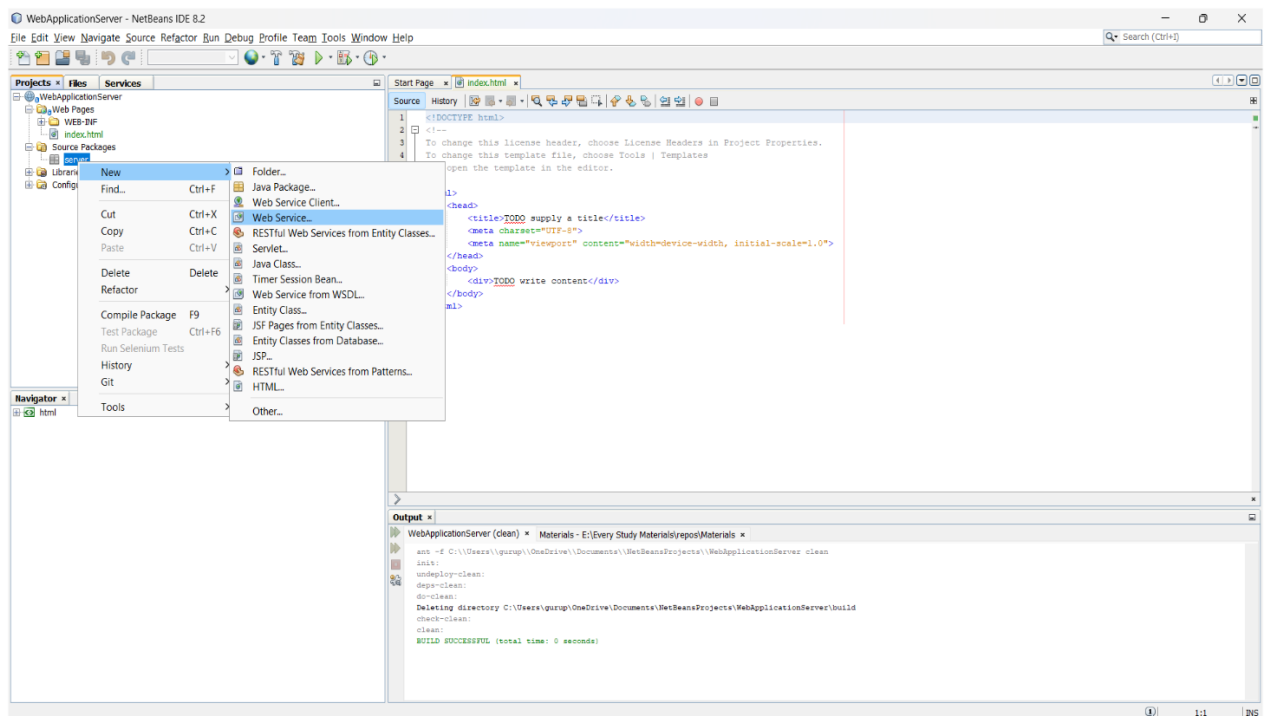
# Practical 1 Web Service Implementation

## 3) Specify Name of Java Package and click on Finish Button



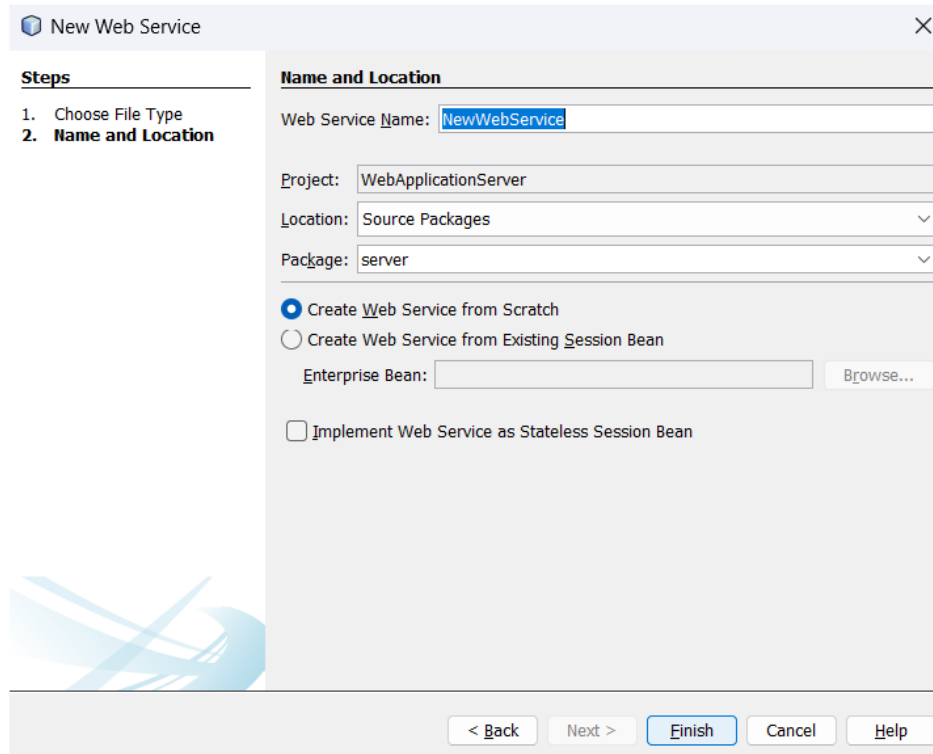
## 4) Right Click on Package and create web service

*New > Web Service*



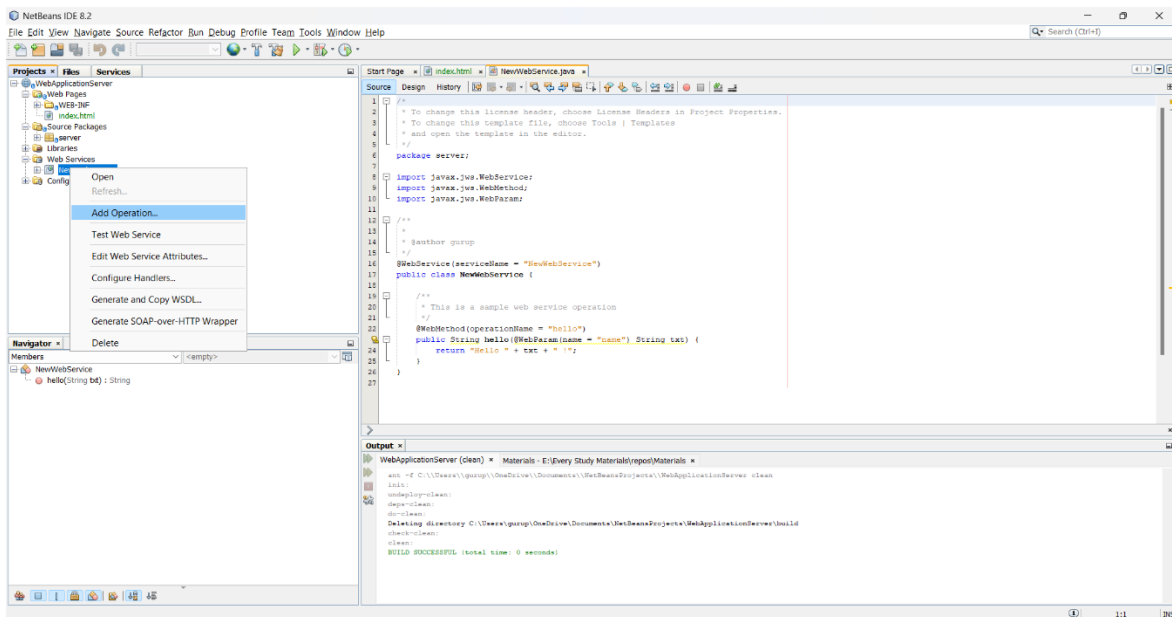
# Practical 1 Web Service Implementation

## 5) Specify Name of Web service and select Java Package



**Web Services Folder is created**

## 6) Under Web Services Folder Select Web service and Right click > Add Operation



---

## Practical 1 Web Service Implementation

### 7) Add Operation

*Specify Name of Operation and click on OK*

**Add Operation**

Name: SayHello

Return Type: java.lang.String Browse...

Parameters Exceptions

Name	Type	Final
------	------	-------

Add Remove Up Down

OK Cancel

### 8) You May also add operation with parameters

**Add Operation**

Name: add

Return Type: java.lang.Integer Browse...

Parameters Exceptions

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

Add Remove Up Down

OK Cancel

**Navigator**

Members

- NewWebService
  - SayHello() : String
  - add(int a, int b) : Integer
  - div(int a, int b) : Integer
  - hello(String txt) : String
  - mul(int a, int b) : Integer
  - sub(int a, int b) : Integer

# Practical 1 Web Service Implementation

- 9) **Modify Method according to requirement.. like I want to return string is “Hello Guru” and perform other operations as below :**

```
/* Web service operation
 */
@WebMethod(operationName = "SayHello")
public String SayHello() {
    //TODO write your implementation code here:
    return "Hello Guru";
}

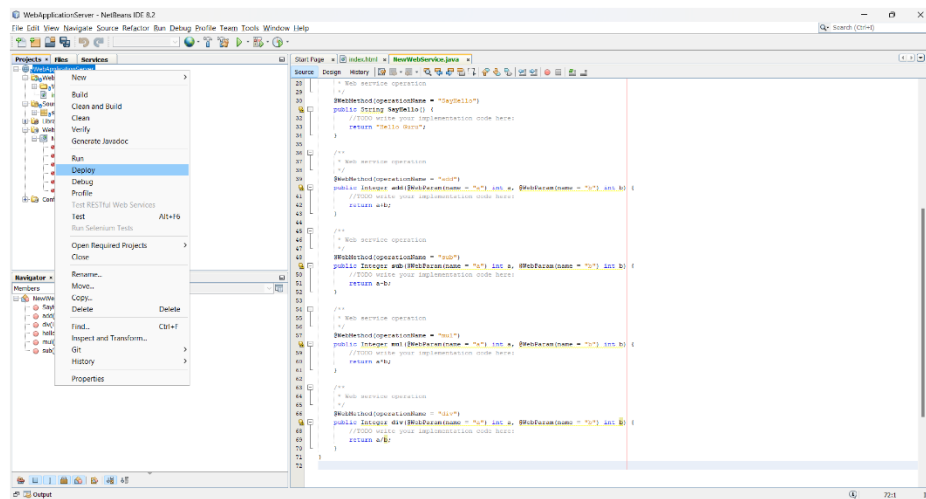
/**
 * Web service operation
 */
@WebMethod(operationName = "add")
public Integer add(@WebParam(name = "a") int a, @WebParam(name = "b") int b) {
    //TODO write your implementation code here:
    return a+b;
}

/**
 * Web service operation
 */
@WebMethod(operationName = "sub")
public Integer sub(@WebParam(name = "a") int a, @WebParam(name = "b") int b) {
    //TODO write your implementation code here:
    return a-b;
}

/**
 * Web service operation
 */
@WebMethod(operationName = "mul")
public Integer mul(@WebParam(name = "a") int a, @WebParam(name = "b") int b) {
    //TODO write your implementation code here:
    return a*b;
}

/**
 * Web service operation
 */
@WebMethod(operationName = "div")
public Integer div(@WebParam(name = "a") int a, @WebParam(name = "b") int b) {
    //TODO write your implementation code here:
    return a/b;
}
}
```

- 10) **Deploy Project**

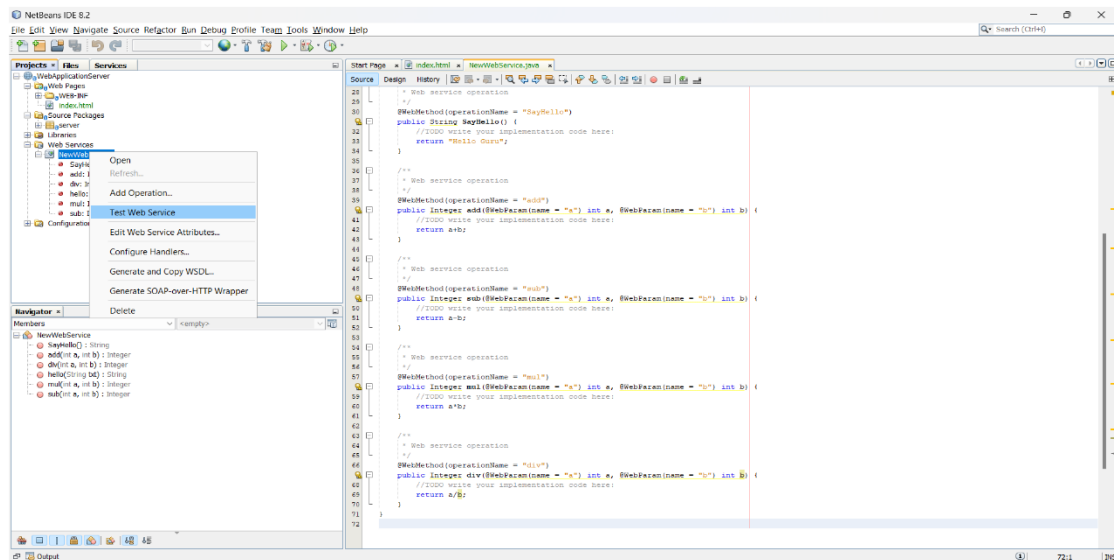


**For Successful deployment**

# Practical 1 Web Service Implementation

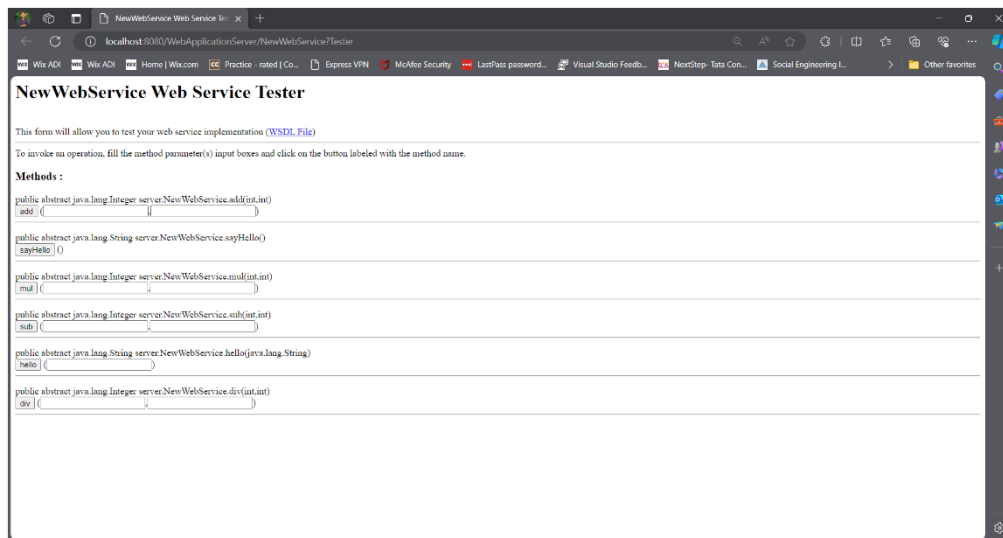
```
Output
Materials - E:\Every Study Materials\repos\Materials x WebApplicationServer (run-deploy) x Java DB Database Process x GlassFish Server 4.1.1 x
deps-ear-jar:
deps-jar:
library-inclusion-in-archive:
library-inclusion-in-manifest:
compile:
compile-jsp:
Undeploying ...
In-place deployment at E:\Every Study Materials\repos\Materials\SEM-6\Cloud Computing\My_Practicals\Prac-1\WebApplicationServer\build\web
run-deploy:
BUILD SUCCESSFUL (total time: 1 second)
```

## 11) Test Web Service



The screenshot shows the NetBeans IDE interface. On the left, the 'Services' tab is active, displaying a context menu for 'NewWebService' with options like 'Open', 'Refresh', 'Add Operation...', 'Test Web Service', 'Edit Web Service Attributes...', 'Configure Handlers...', 'Generate and Copy WSDL...', and 'Delete'. The 'Test Web Service' option is highlighted. The main editor area shows the source code of 'NewWebService.java', which implements a web service with methods: sayHello(), add(), mul(), sub(), and div(). Each method is annotated with @WebMethod and includes a comment indicating where to write the implementation code.

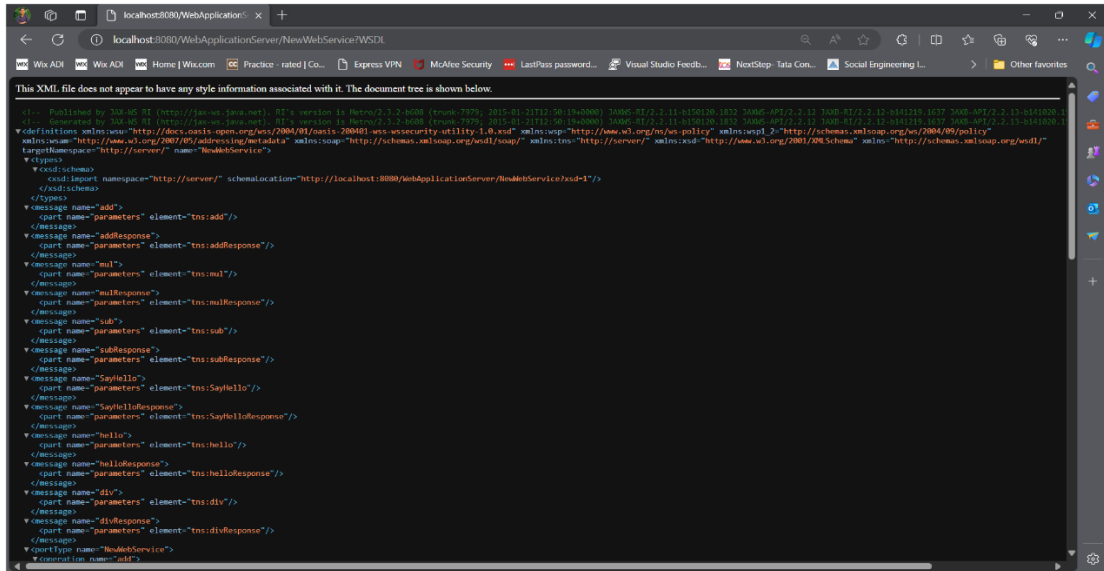
## Output of Test Web service in web Browser



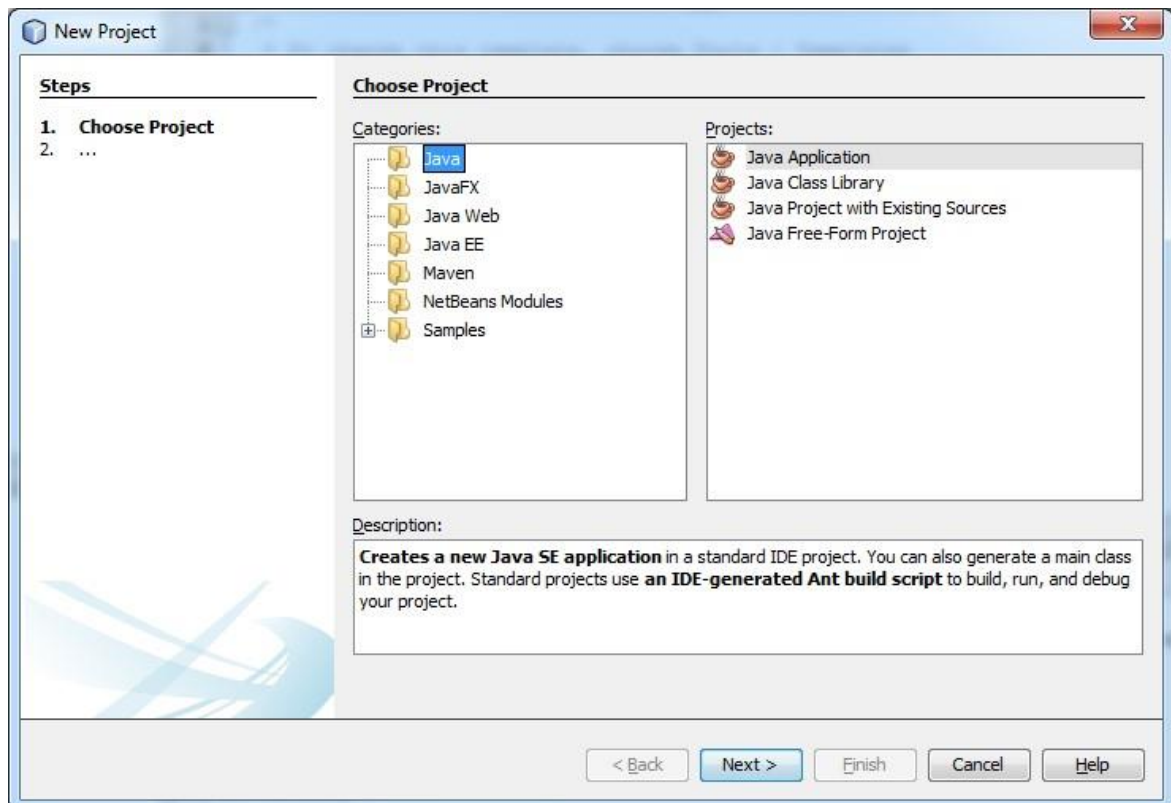
The screenshot shows a web browser window displaying the 'NewWebService Web Service Tester' page. The page title is 'NewWebService Web Service Tester'. Below the title, there is a message: 'This form will allow you to test your web service implementation (WSDL File)'. A sub-message states: 'To invoke an operation, fill the method parameter(s) input boxes and click on the button labeled with the method name.' Under the heading 'Methods:', there are six rows, each representing a method from the web service interface. Each row contains the method signature and a button labeled with the method name. The methods are: add(int a, int b), sayHello(), mul(int a, int b), sub(int a, int b), hello(java.lang.String), and div(int a, int b). Each method signature is followed by a button labeled with the method name.

# Practical 1 Web Service Implementation

## 12) Click on WSDL File Link and copy newly open window URL



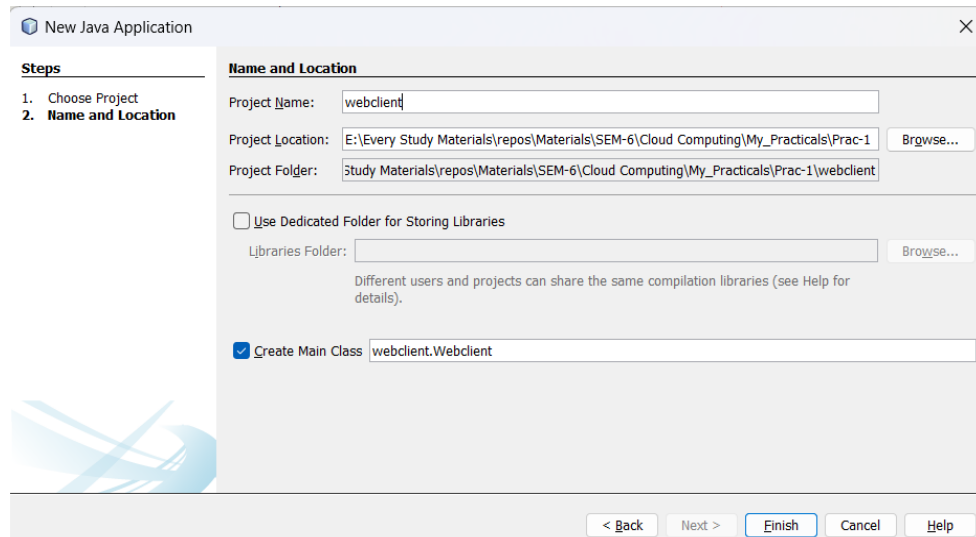
## 13) Create Simple Java Application from File > New > Java > java Application





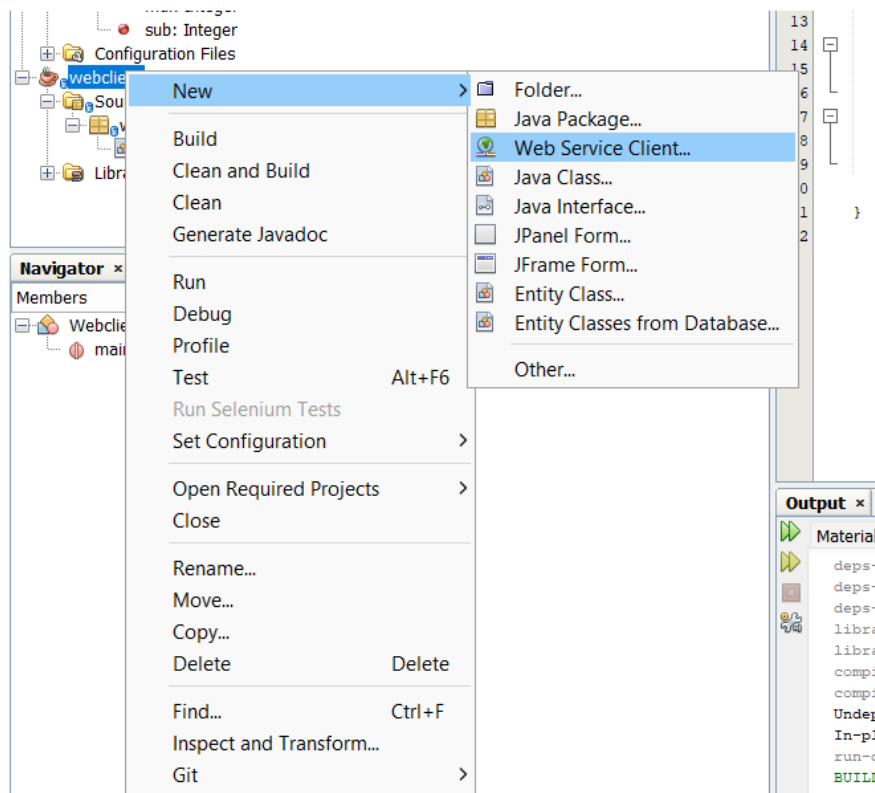
# Practical 1 Web Service Implementation

## 14) Specify Name and Finish



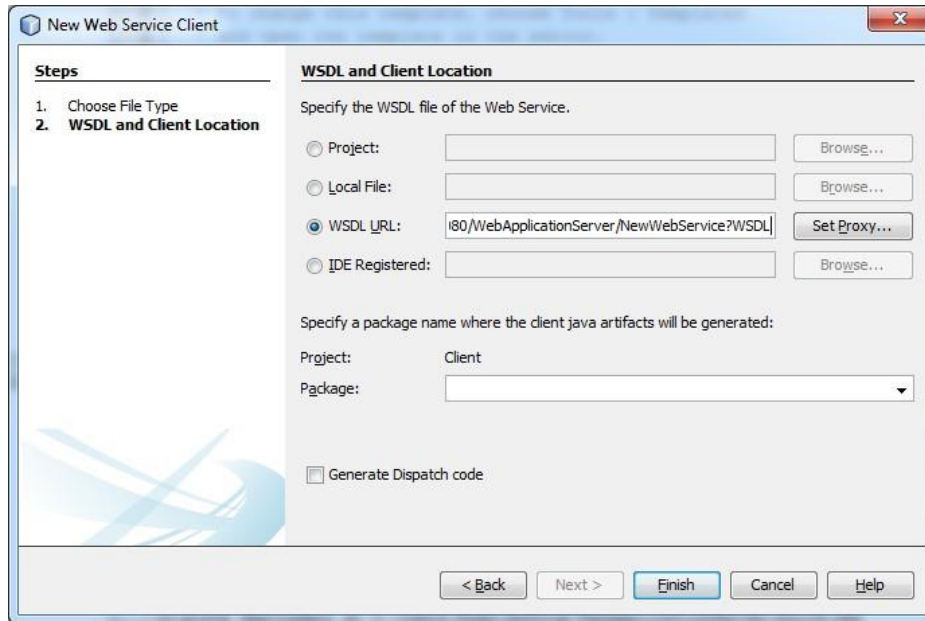
## 15) Add Web Service Client from

*Client Project > Source Package > New > Web Service Client*

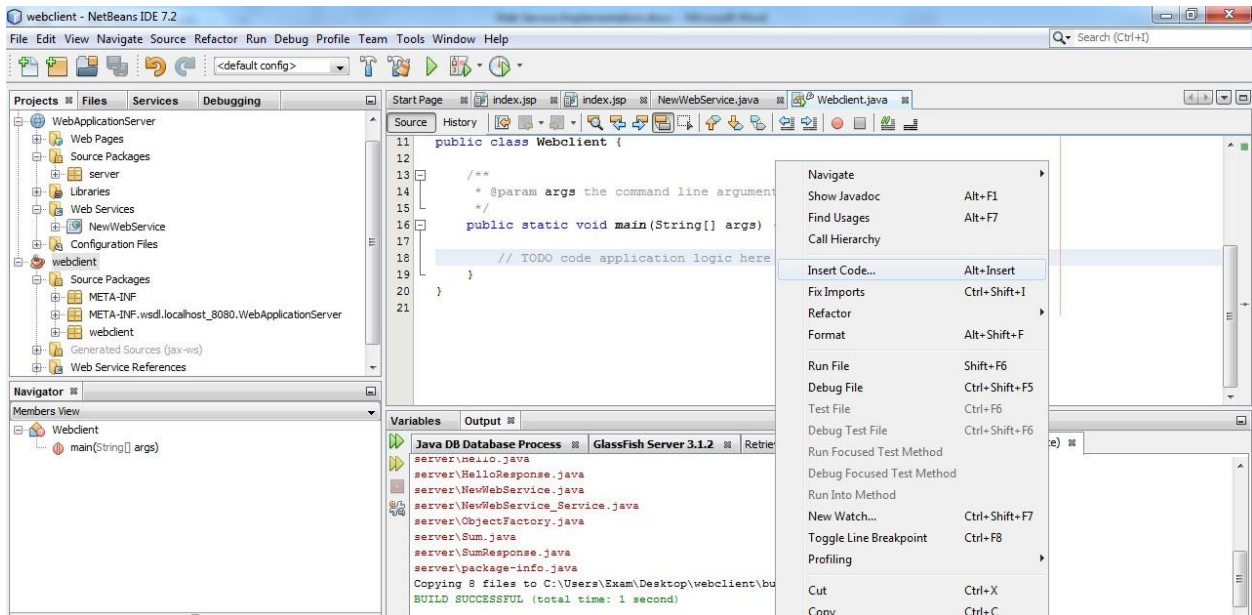


# Practical 1 Web Service Implementation

## 16) Add WSDL and click on Finish

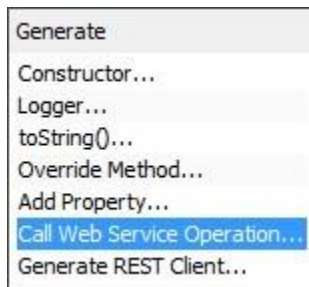


## 17) Open code of java file and Right click > Insert Code

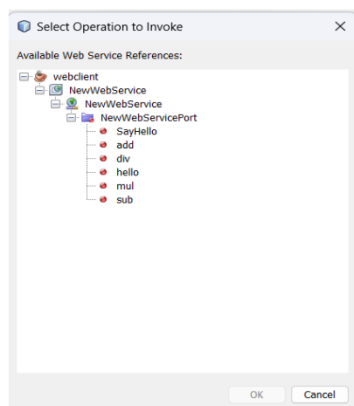


## Practical 1 Web Service Implementation

*Then Select Call Web Service Operation*



### 18) Select Operation from Web Service



### 19) You may Invoke multiple operation one by one

```
//  
public class Webclient {  
  
    /**  
     * @param args the command line arguments  
     */  
    public static void main(String[] args) {  
        // TODO code application logic here  
    }  
  
    private static String sayHello() {  
        server.NewWebService_Service service = new server.NewWebService_Service();  
        server.NewWebService port = service.getNewWebServicePort();  
        return port.sayHello();  
    }  
  
    private static Integer add(int a, int b) {  
        server.NewWebService_Service service = new server.NewWebService_Service();  
        server.NewWebService port = service.getNewWebServicePort();  
        return port.add(a, b);  
    }  
  
    private static Integer sub(int a, int b) {  
        server.NewWebService_Service service = new server.NewWebService_Service();  
        server.NewWebService port = service.getNewWebServicePort();  
        return port.sub(a, b);  
    }  
  
    private static Integer mul(int a, int b) {  
        server.NewWebService_Service service = new server.NewWebService_Service();  
        server.NewWebService port = service.getNewWebServicePort();  
        return port.mul(a, b);  
    }  
  
    private static Integer div(int a, int b) {  
        server.NewWebService_Service service = new server.NewWebService_Service();  
        server.NewWebService port = service.getNewWebServicePort();  
        return port.div(a, b);  
    }  
}
```

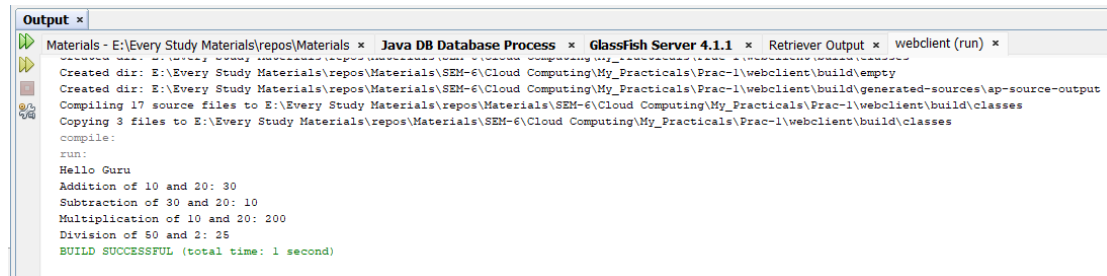
---

## Practical 1 Web Service Implementation

### 20) Modify code of Main Method in webClient.java

```
*/  
public static void main(String[] args) {  
    // TODO code application logic here  
    try{  
        server.NewWebService_Service service = new server.NewWebService_Service();  
        server.NewWebService port = service.getNewWebServicePort();  
        System.out.println(port.sayHello());  
  
        int add = port.add(10, 20);  
        System.out.println("Addition of 10 and 20: "+add);  
  
        int sub = port.sub(30, 20);  
        System.out.println("Subtraction of 30 and 20: "+sub);  
  
        int mul = port.mul(10, 20);  
        System.out.println("Multiplication of 10 and 20: "+mul);  
  
        int div = port.div(50, 2);  
        System.out.println("Division of 50 and 2: "+div);  
    } catch (Exception ex) {  
        System.out.println(ex);  
    }  
}
```

### 21) Run Client Java File



Output x

Materials - E:\Every Study Materials\repos\Materials x Java DB Database Process x GlassFish Server 4.1.1 x Retriever Output x webclient (run) x

Created dir: E:\Every Study Materials\repos\Materials\SEM-6\Cloud Computing\My\_Practicals\Prac-1\webclient\build\empty  
Created dir: E:\Every Study Materials\repos\Materials\SEM-6\Cloud Computing\My\_Practicals\Prac-1\webclient\build\generated-sources\ap-source-output  
Compiling 17 source files to E:\Every Study Materials\repos\Materials\SEM-6\Cloud Computing\My\_Practicals\Prac-1\webclient\build\classes  
Copying 3 files to E:\Every Study Materials\repos\Materials\SEM-6\Cloud Computing\My\_Practicals\Prac-1\webclient\build\classes  
compile:  
run:  
Hello Guru  
Addition of 10 and 20: 30  
Subtraction of 30 and 20: 10  
Multiplication of 10 and 20: 200  
Division of 50 and 2: 25  
BUILD SUCCESSFUL (total time: 1 second)