

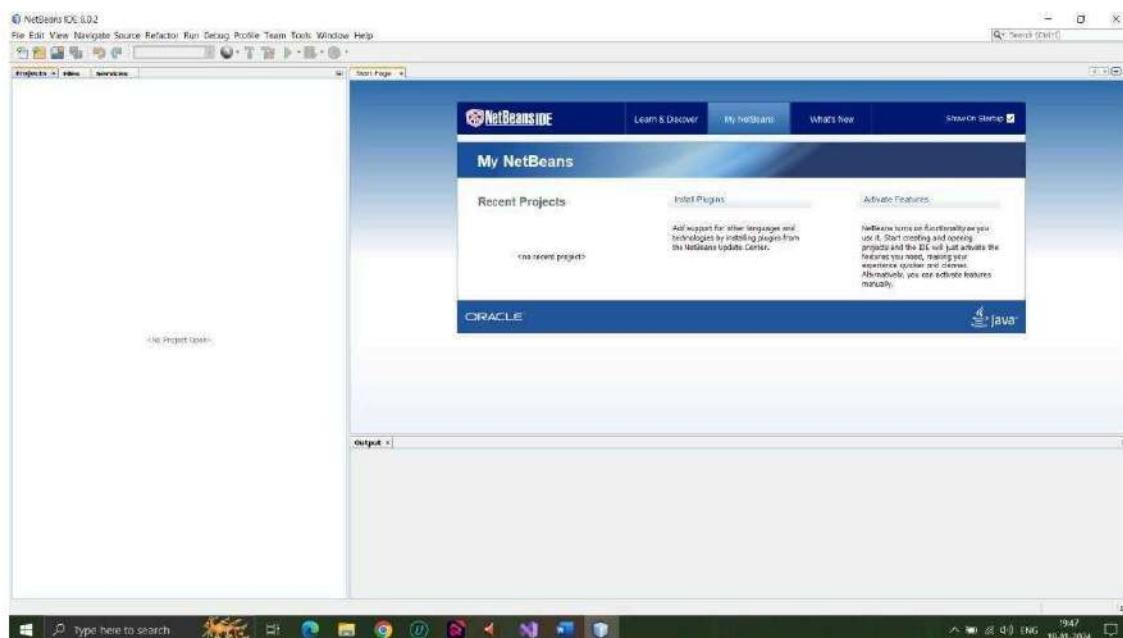
Cloud Computing and Web Services Practical

Practical - 1

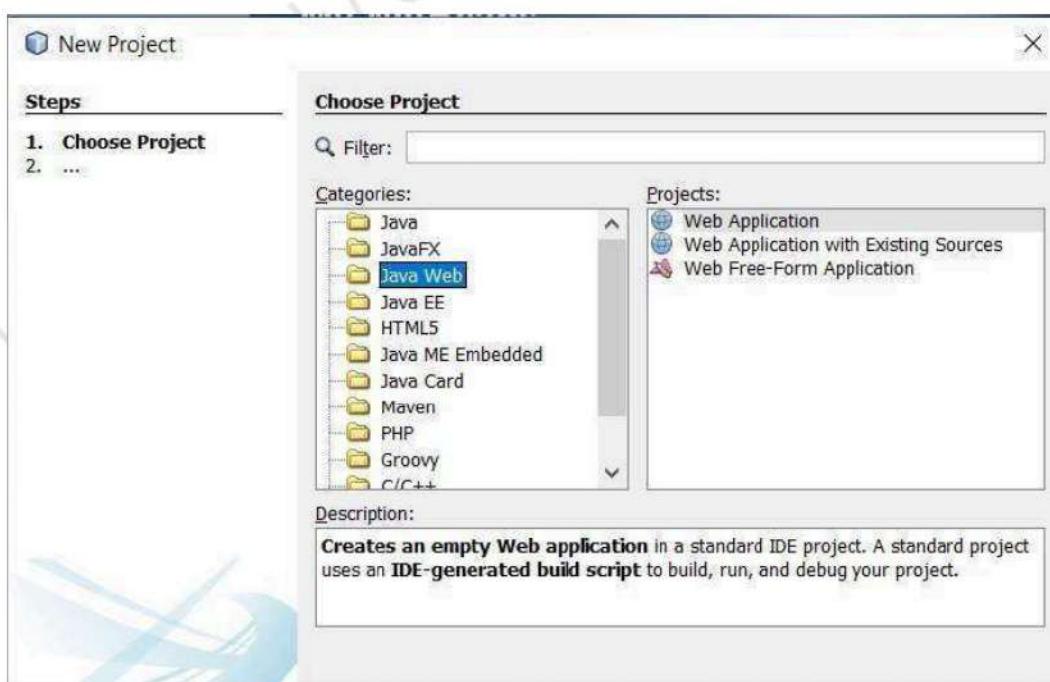
Aim: Define a simple services like Converting Rs into Dollar and Call it from different platform like JAVA

Steps:

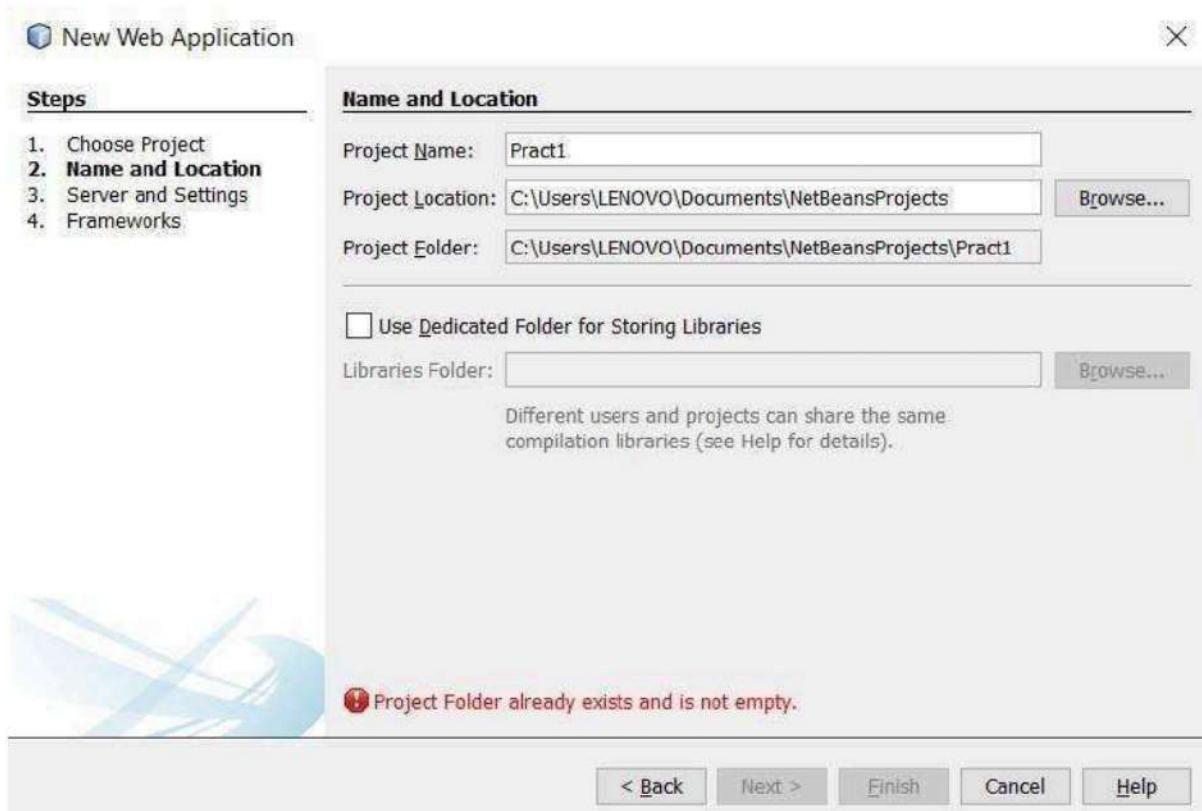
- 1] Open the NetBeans, and you will get the following screen. Close the start page.



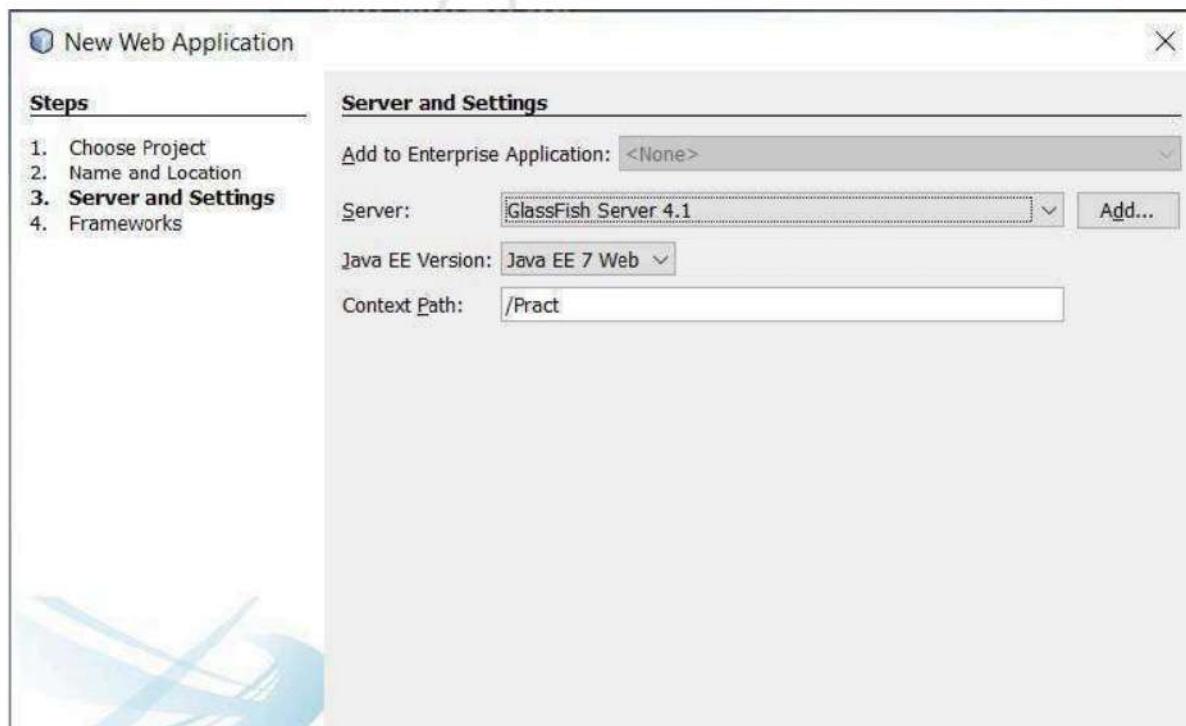
- 2] Now click on the file tab and click on new project you will get the following screen :



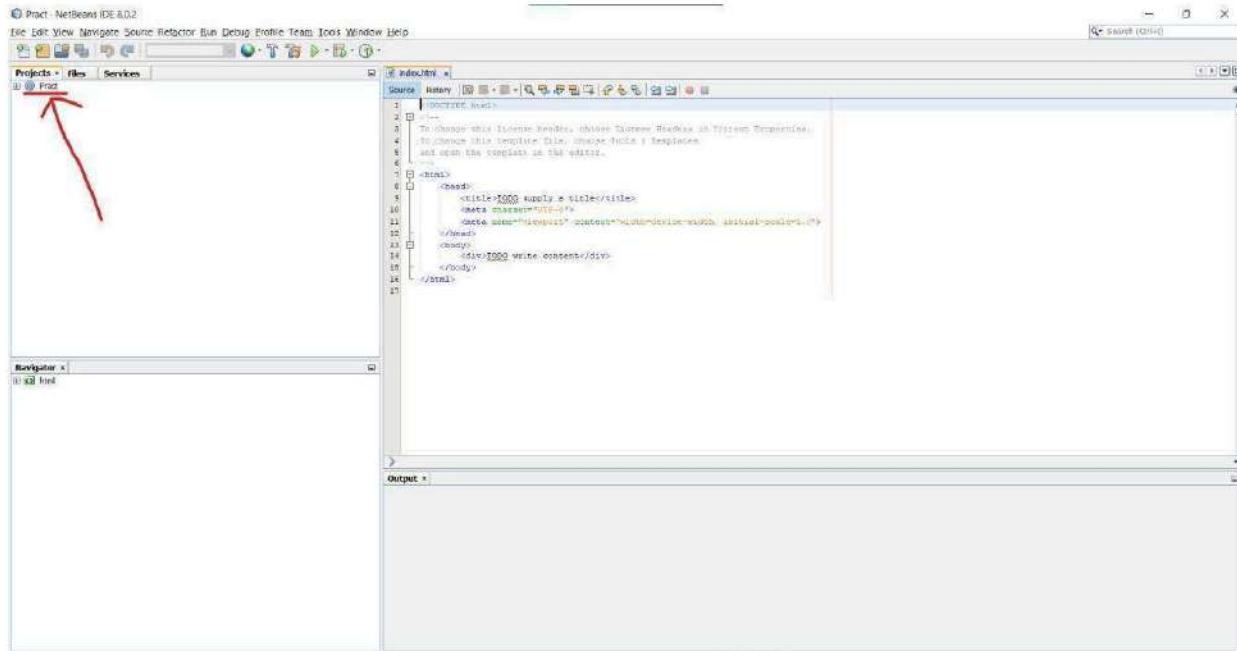
3] In Categories select Java Web and in Projects , Select Web Application. After selecting click on next . You will get the following window:



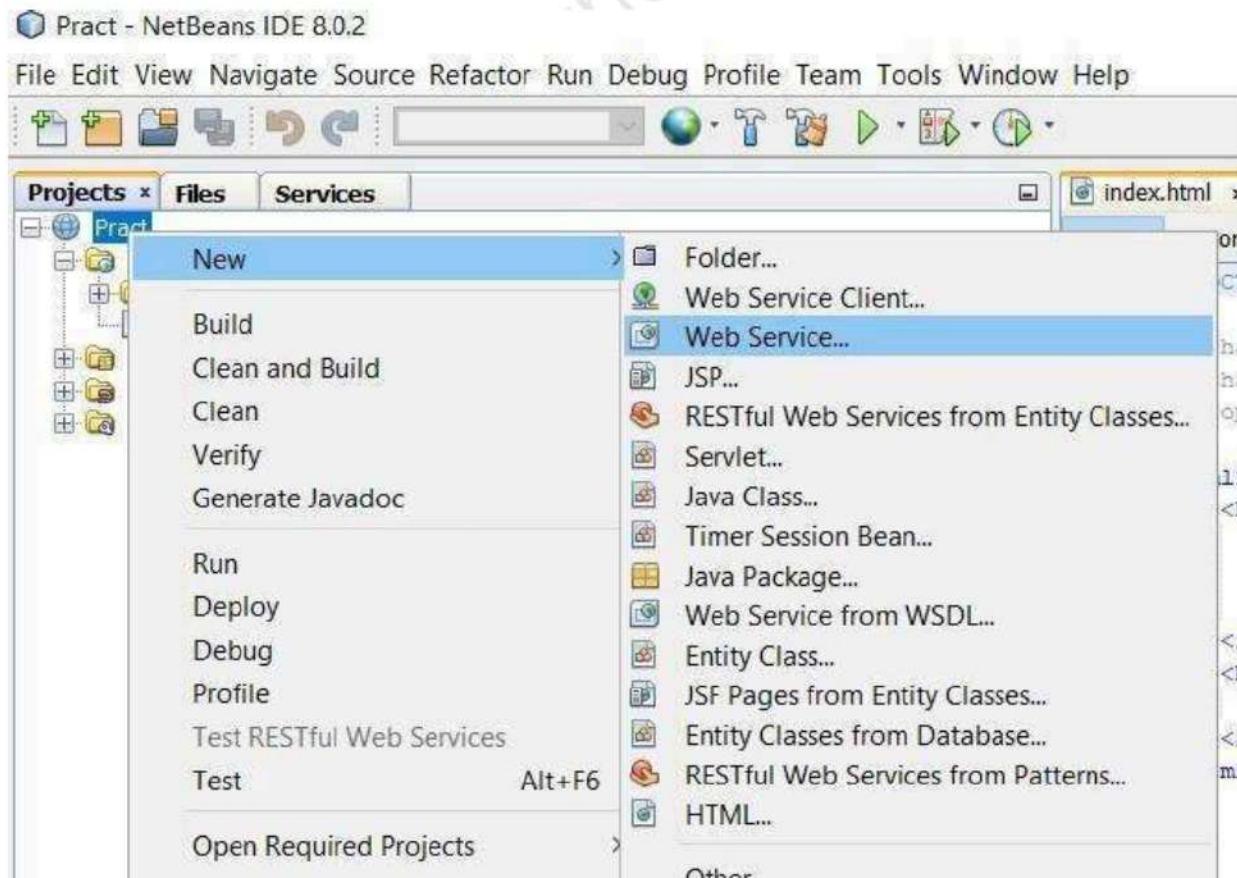
4] Now Give name to the Project Name: , and click on next . you will get the following windowthen again click on finish:



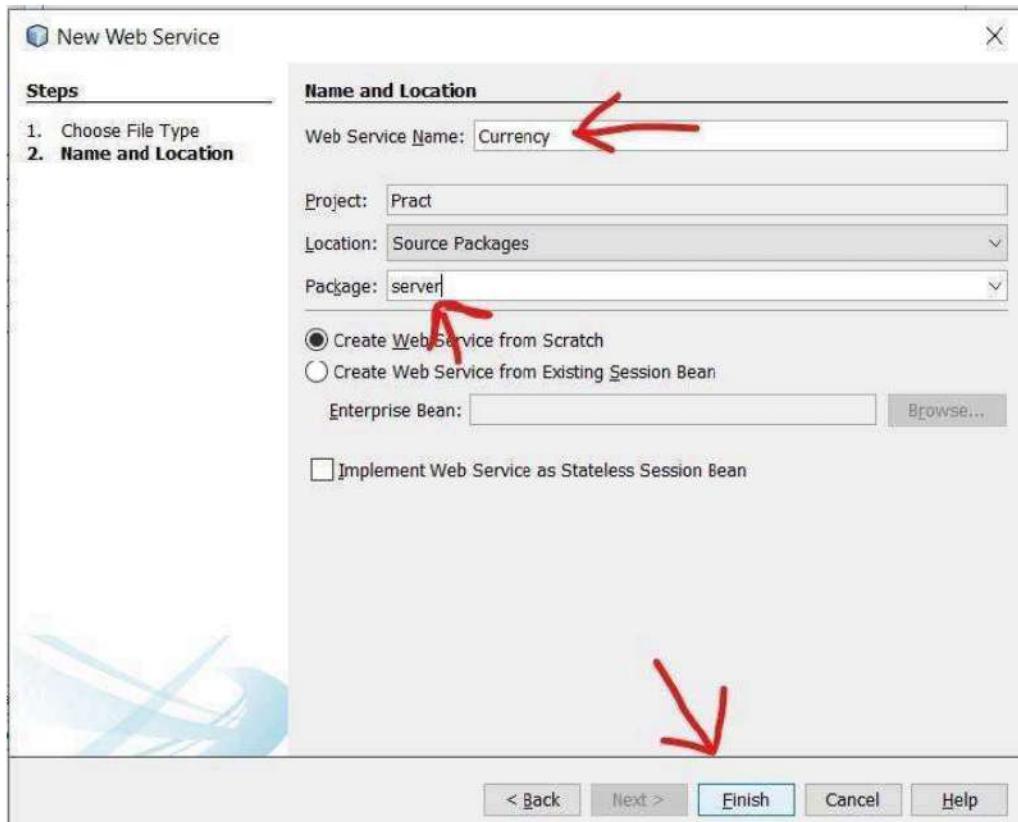
5] You will get the following screen now carefully see in projects section your recently created project appears double click to expand it:



6] Now Right click on the project and select new and then select Web Service ,As shown Below:



7] after clicking Web Service following window should appear , now give name to web serviceand give package as "server". As shown in the image:



After clicking finish you should get the following window erase the mentioned code :

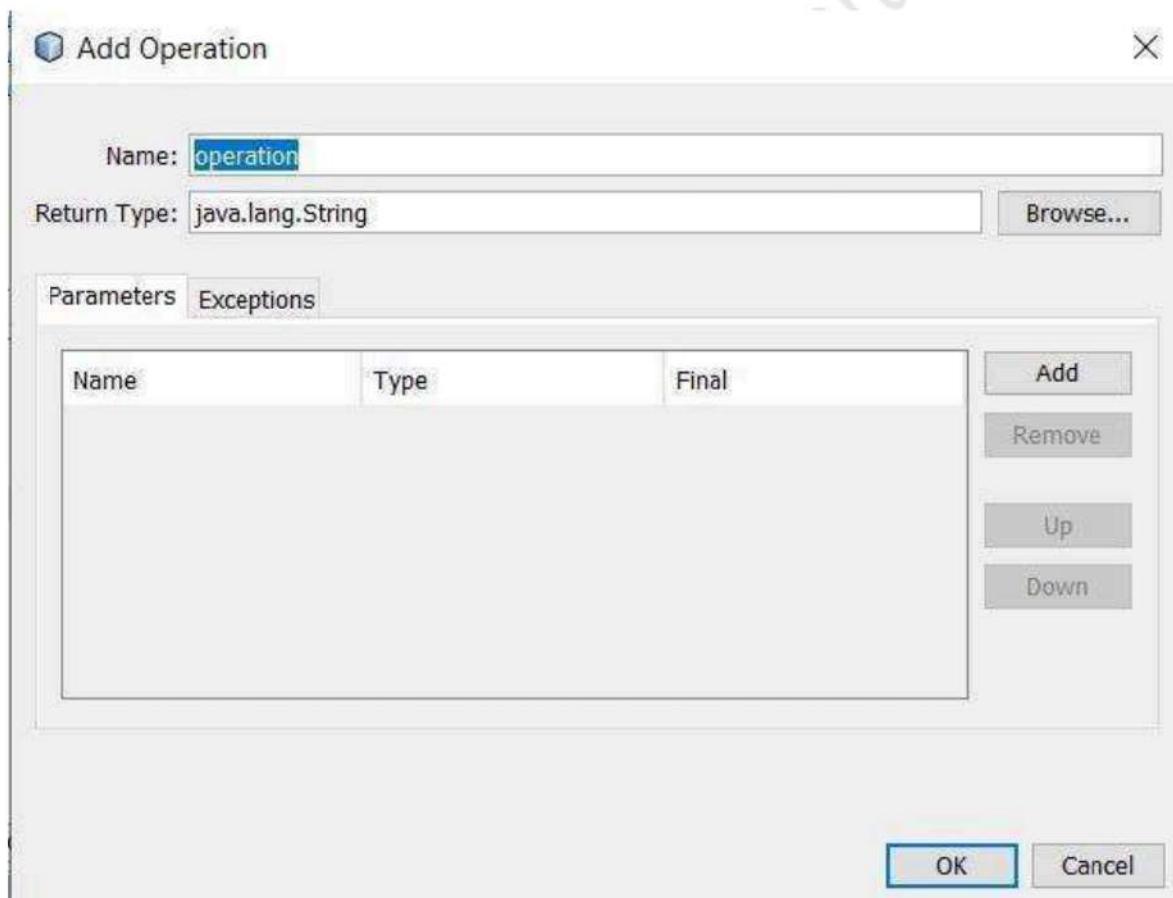
```
1 * To change this license header, choose License Headers in Project Properties.
2 * To change this template file, choose Tools | Templates
3 * and open the template in the editor.
4 */
5 package server;
6
7 import javax.jws.WebService;
8 import javax.jws.WebMethod;
9 import javax.jws.WebParam;
10
11 /**
12 */
13 /**
14 * @author LENOVO
15 */
16 @WebService(serviceName = "Currency")
17 public class Currency {
18
19 /**
20 * This is a sample web service operation
21 */
22
23 /**
24 * WebMethod(operationName = "hello")
25 * public String hello(@WebParam(name = "name") String txt) {
26 *     return "Hello " + txt + " !";
27 }
```

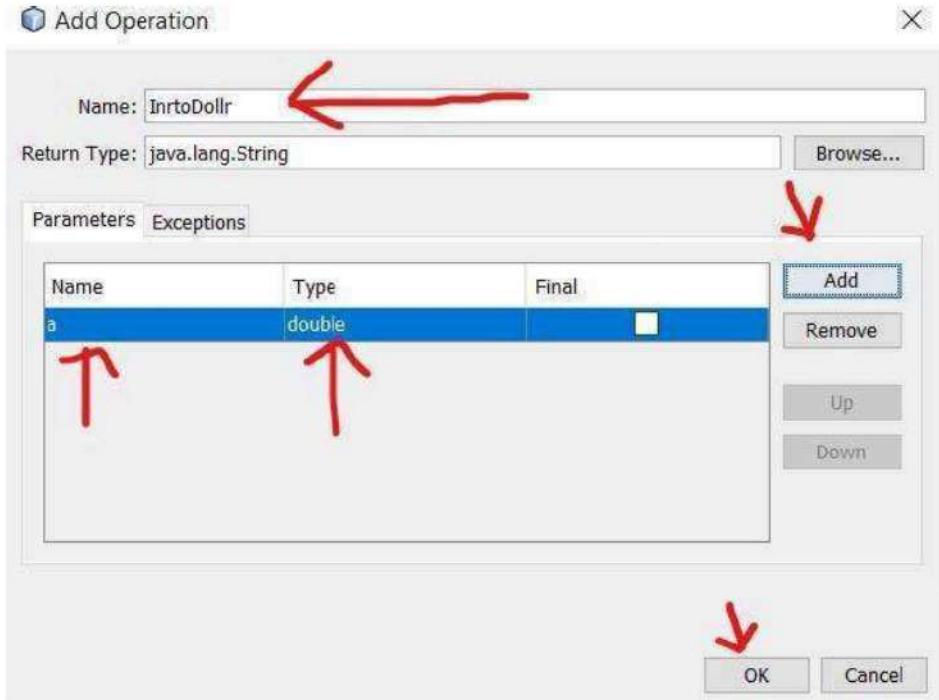
8] Now right click any where and click on insert code and select Add Web Service Operation :



9] the following window would appear :

Just give name to the method or operation and click on add button to add parameters to the method as here we are converting dollar to rupees we should need only one parameter .

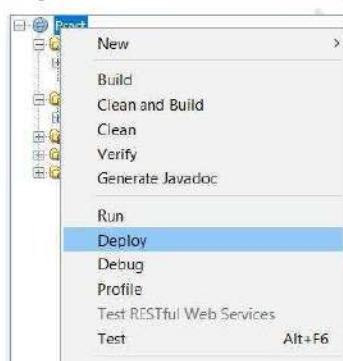




11] After clicking ok code will autogenerated , make changes In that code as mentioned below:

```
@WebMethod(operationName = "InrtoDollar")
public String InrtoDollar(@WebParam(name = "a") double a) {
    //TODO write your implementation code here:
    return "The Indian rupees "+a+" in Dollars is "+(a/83.17);
}
```

12] now our web service is ready now right click on project and click on deploy :





CurrencyConverter 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.String server.CurrencyConvertor.inrtoDollar(double)
inrtoDollar ( )
```

Method invocation trace +

localhost:8080/practical1/CurrencyConverter?Tester

inrtoDollar Method invocation

Method parameter(s)

Type	Value
double	1233

Method returned

java.lang.String : "The Indian rupees 1233.0 in Dollars is 14.825057111939401"

SOAP Request

```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<S:Header>
<ns2:InrtoDollar xmlns:ns2="http://server/">
<a>1233.0</a>
</ns2:InrtoDollar>
</S:Header>
</S:Envelope>
```

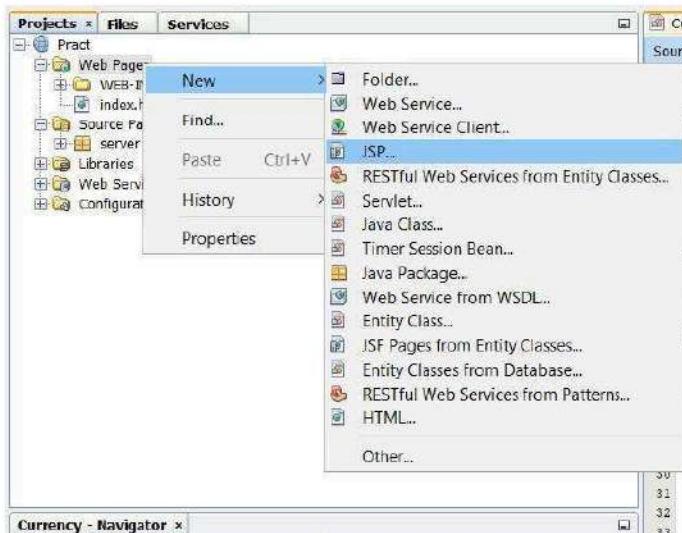
SOAP Response

```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<S:Header>
<ns2:InrtoDollarResponse xmlns:ns2="http://server/">
<return>The Indian rupees 1233.0 in Dollars is 14.825057111939401</return>
</ns2:InrtoDollarResponse>
</S:Header>
</S:Envelope>
```

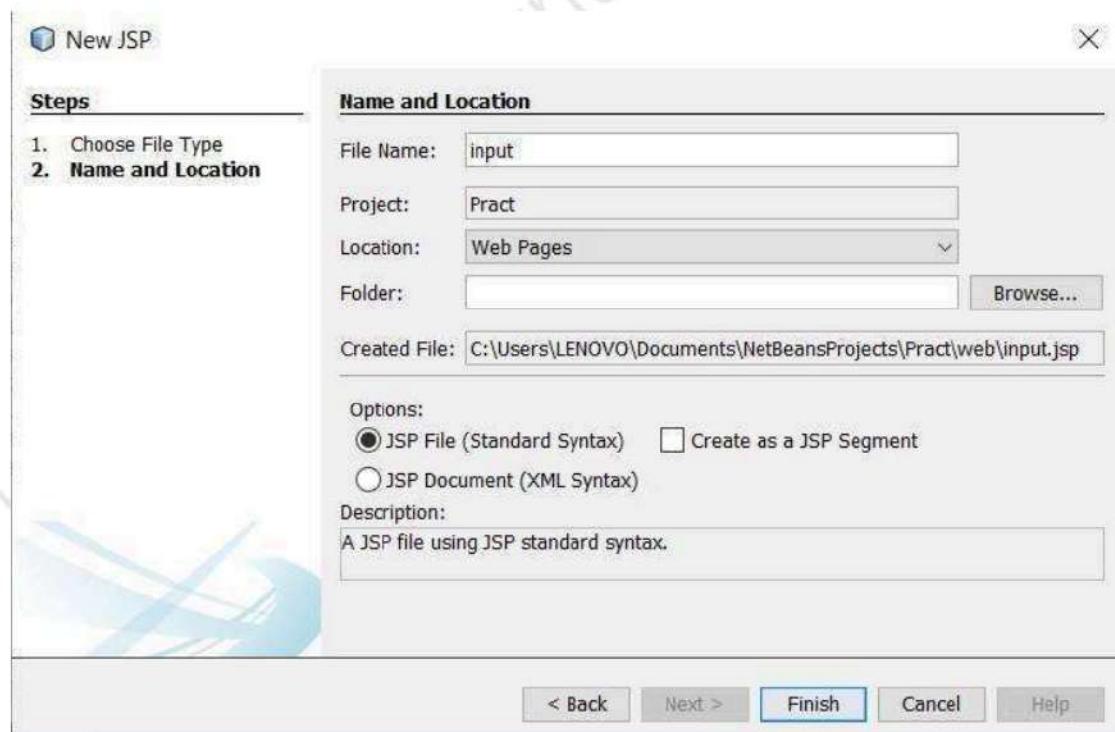
So this is how we created our web service and deployed it.

Creating a java Client using jsp

14] Now our web service is successfully deployed. Right click on web pages and select new and select jsp as shown below:



15] give name and click on finish as shown below do it 2 times on for input and one for output:



16] In input.jsp create a form for taking user input as shown below:

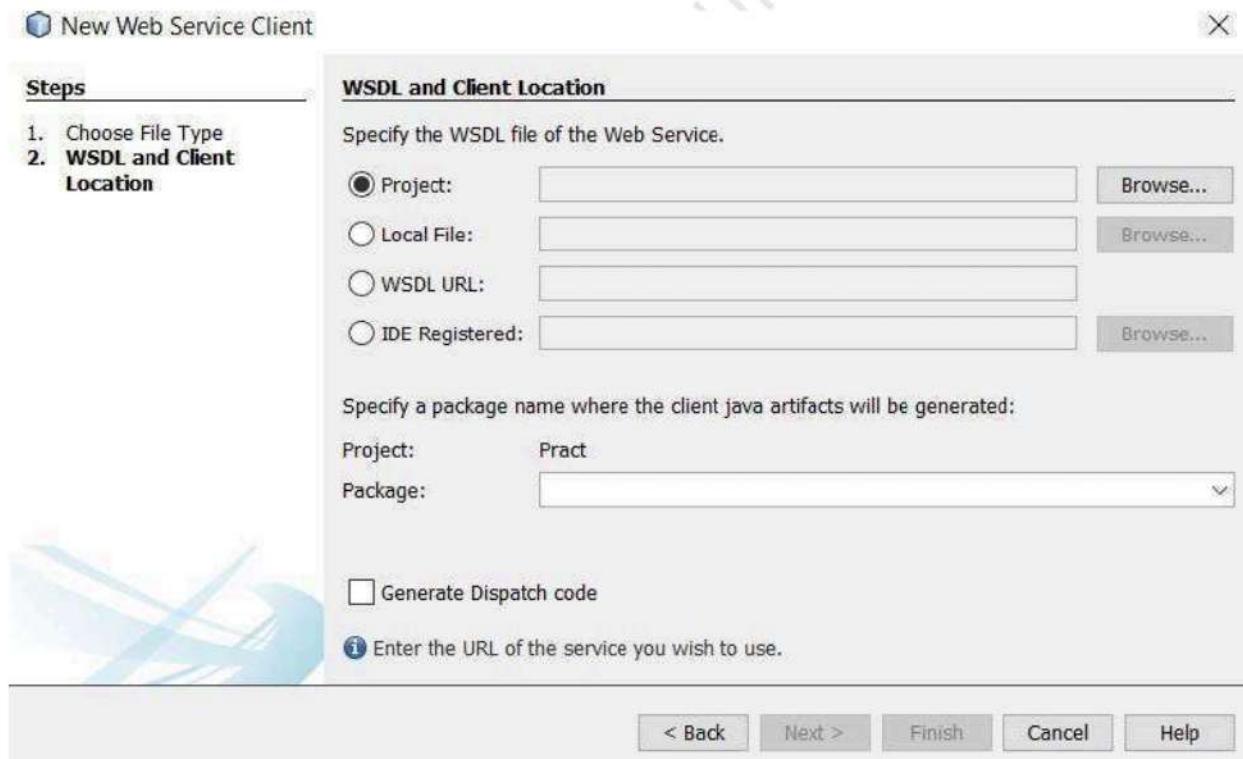
```

<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>JSP Page</title>
    </head>
    <body>
        <form action="output.jsp">
            <pre>
                Enter the currency in rupees : <input type="text" name="t1">
                <input type="submit"> <input type="reset">
            </pre>
        </form>
    </body>
</html>

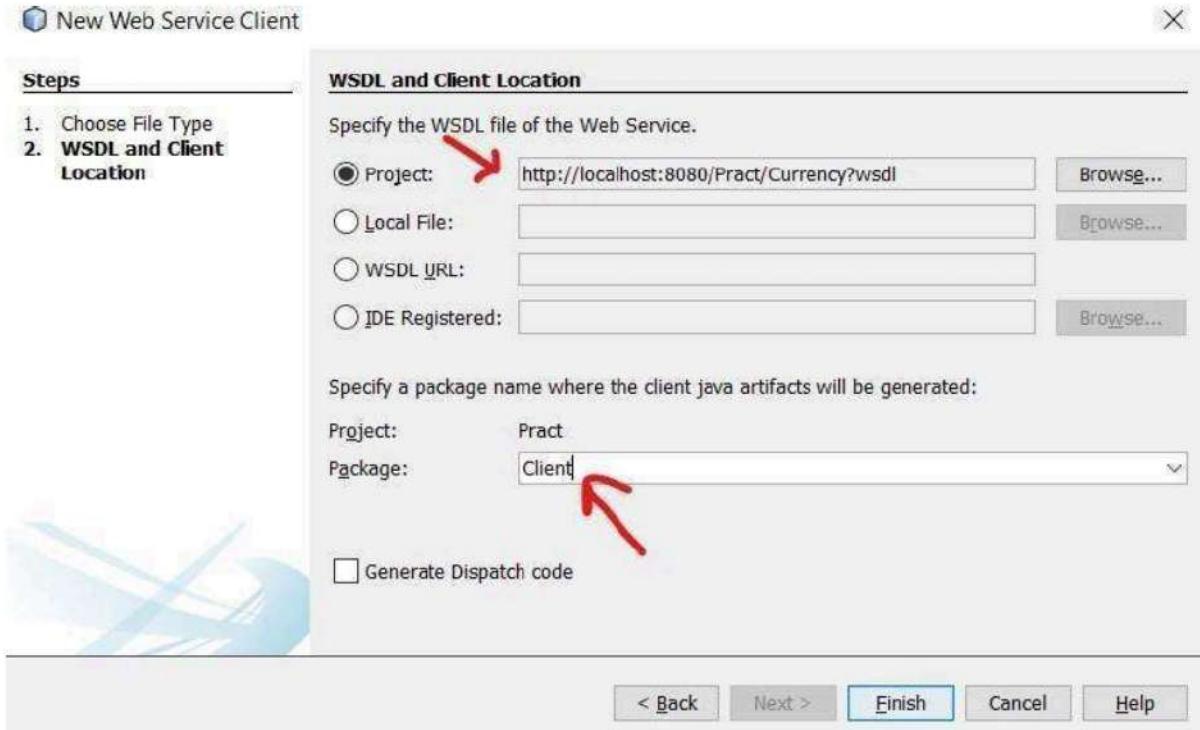
```

In this code set action = the jsp file where u want output and give name to textbox input.

- 17] now we have to create web service client, for same right click on project and select new then select web service client you will get the following screen:



At this step click on browse and select your project and click ok after clicking ok you will get wsdl url as shown below:



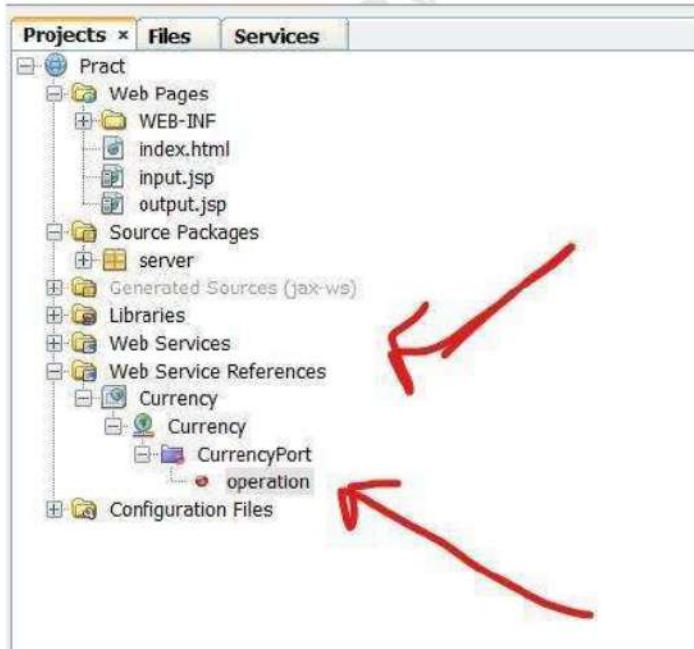
At this stage copy the url and paste it in notepad for future.

Give package name as "Client".

Now click on finish.

18] now in project section you can see a new folder is been created named "web servicerefrence".

double click to expand it you should get the following :



Hold the operation or your operation name and drag it into the output.jsp in body tag as shown : You will get the following auto generated code:

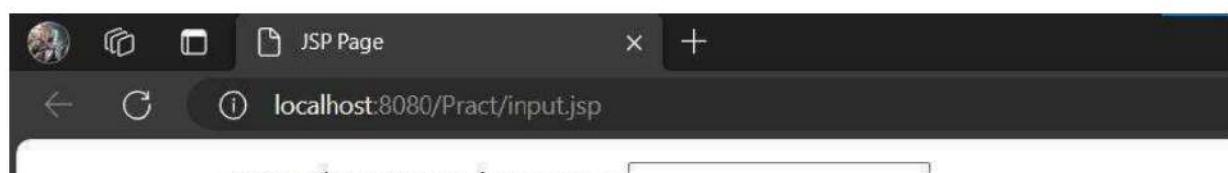
```
<body>
    <%-- start web service invocation --%><hr/>
<%
try {
    Client.Currency_Service service = new Client.Currency_Service();
    Client.Currency port = service.getCurrencyPort();
    // TODO initialize WS operation arguments here
    double a = 0.0d;
    // TODO process result here
    java.lang.String result = port.operation(a);
    out.println("Result = "+result);
} catch (Exception ex) {
    // TODO handle custom exceptions here
}
%>
<%-- end web service invocation --%><hr/>

</body>
```

19] now make changes as shown below in the code:

```
<%-- start web service invocation --%><hr/>
<%
try {
    Client.Currency_Service service = new Client.Currency_Service();
    Client.Currency port = service.getCurrencyPort();
    // TODO initialize WS operation arguments here
    double a = Double.parseDouble(request.getParameter("t1"));
    // TODO process result here
    java.lang.String result = port.operation(a);
    out.println(result); ←
} catch (Exception ex) {
    // TODO handle custom exceptions here
}
%>
<%-- end web service invocation --%><hr/>
```

20] now Deploy our project again and right click on input.jsp and click run file you will redirect to a browser page as shown :



Enter any numerical value in text box and click on submit you will get the following output:



Python client:

1] for consuming java web services in python we should repeat the above steps till step no.19 .

Note: for getting output the project or web service should be deployed .

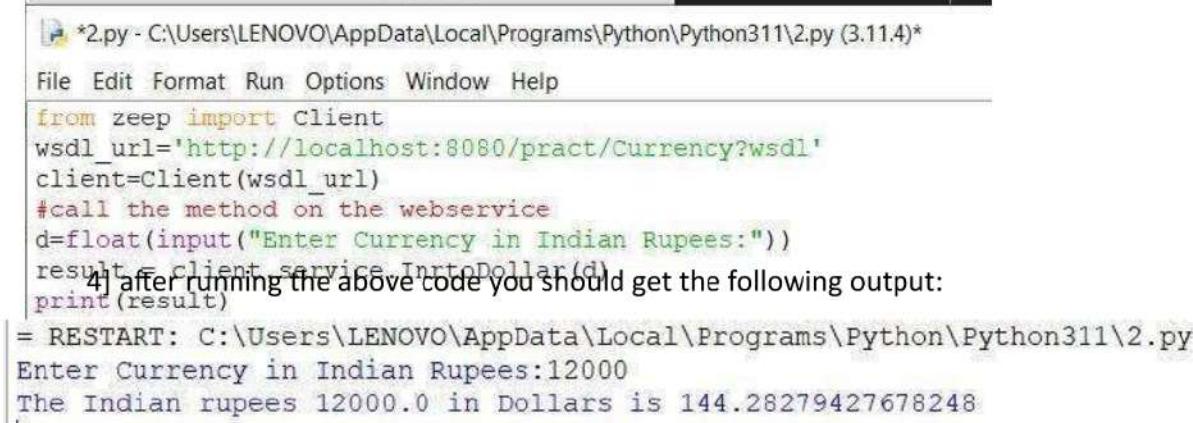
After doing all of the steps write the following code in python idle:

Code :

```
from zeep import Client  
  
wsdl_url='http://localhost:8080/pract/Currency?wsdl' client=Client(wsdl_url)  
  
#call the method on the webservice  
  
d=float(input("Enter Currency in Indian Rupees:"))  
  
result = client.service.InrtoDollar(d)  
  
print(result)2] replace wsdl_url with the url which was copied at the time of web  
service client creation .
```

In step no.17 . and replace the InrtoDollarwith your method or operation name. and pass parameter to it

3] final code should look like following:



```
*2.py - C:\Users\LENOVO\AppData\Local\Programs\Python\Python311\2.py (3.11.4)*  
File Edit Format Run Options Window Help  
from zeep import Client  
wsdl_url='http://localhost:8080/pract/Currency?wsdl'  
client=Client(wsdl_url)  
#call the method on the webservice  
d=float(input("Enter Currency in Indian Rupees:"))  
result = client.service.InrtoDollar(d)  
print(result)  
= RESTART: C:\Users\LENOVO\AppData\Local\Programs\Python\Python311\2.py  
Enter Currency in Indian Rupees:12000  
The Indian rupees 12000.0 in Dollars is 144.28279427678248
```

Practical - 2

Aim: Write a program to implement a simple SOAP WebService

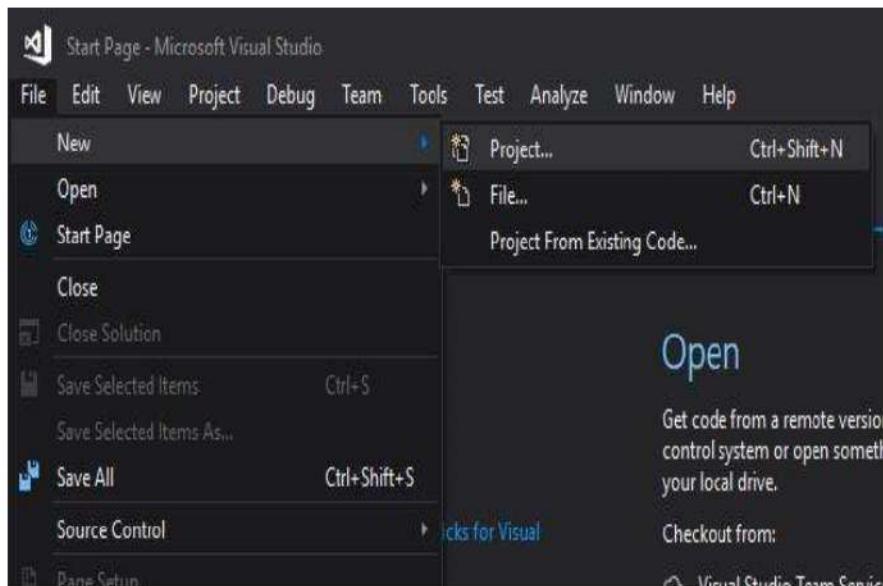
Requirement:

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

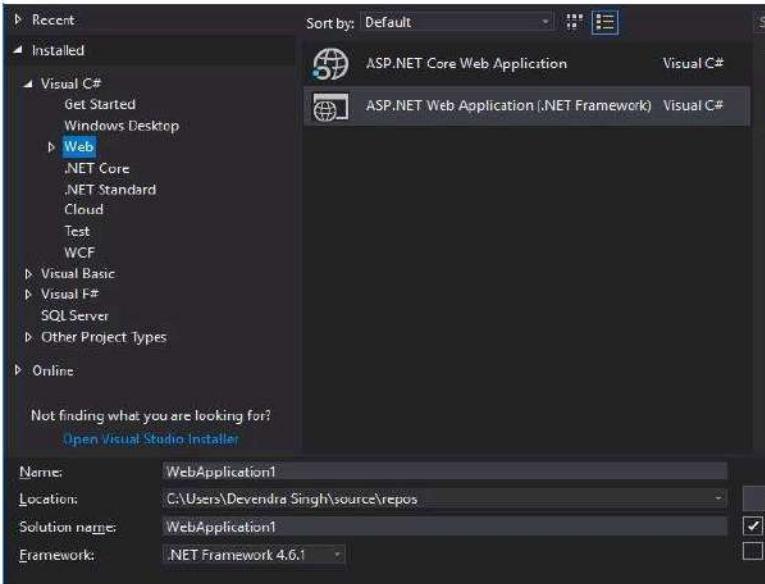
In this practical we are creating a Web Service in Visual Studio and then we will consume it in NetBeans.

1. Open Visual Studio IDE and click on File.

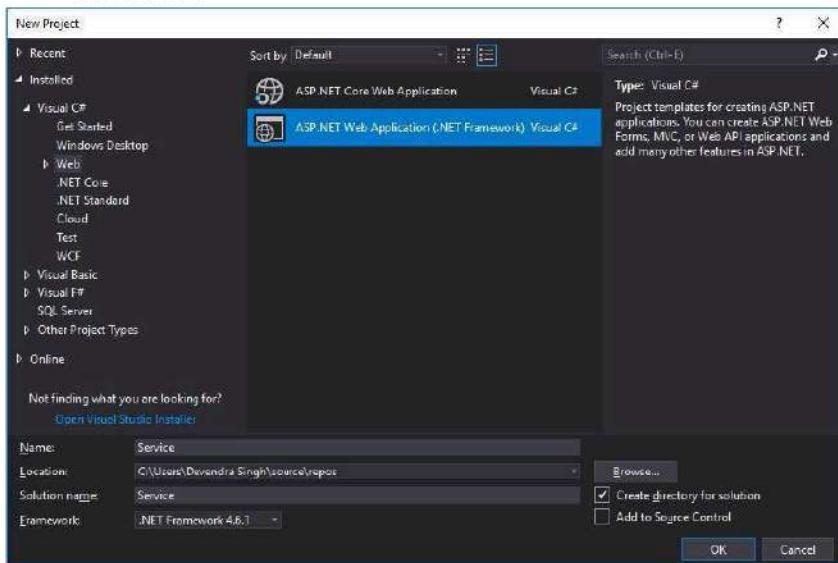
File -> New -> Project



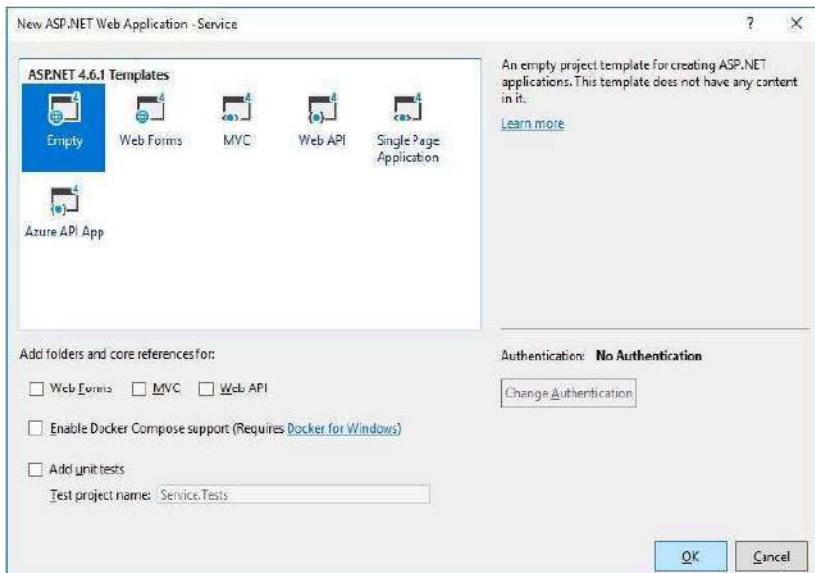
2. Click on Web.



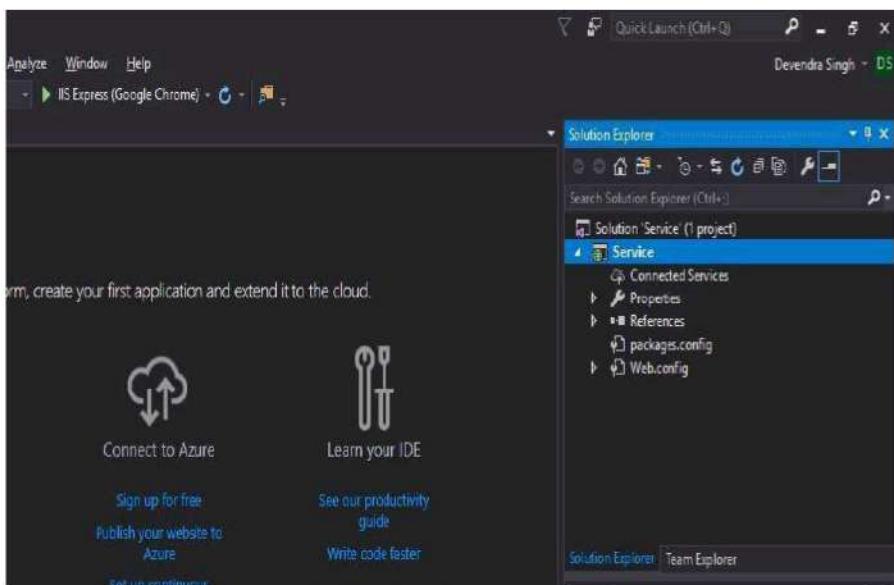
3. Select ASP.NET Web Application and give Name as Service. After that click on OK button.



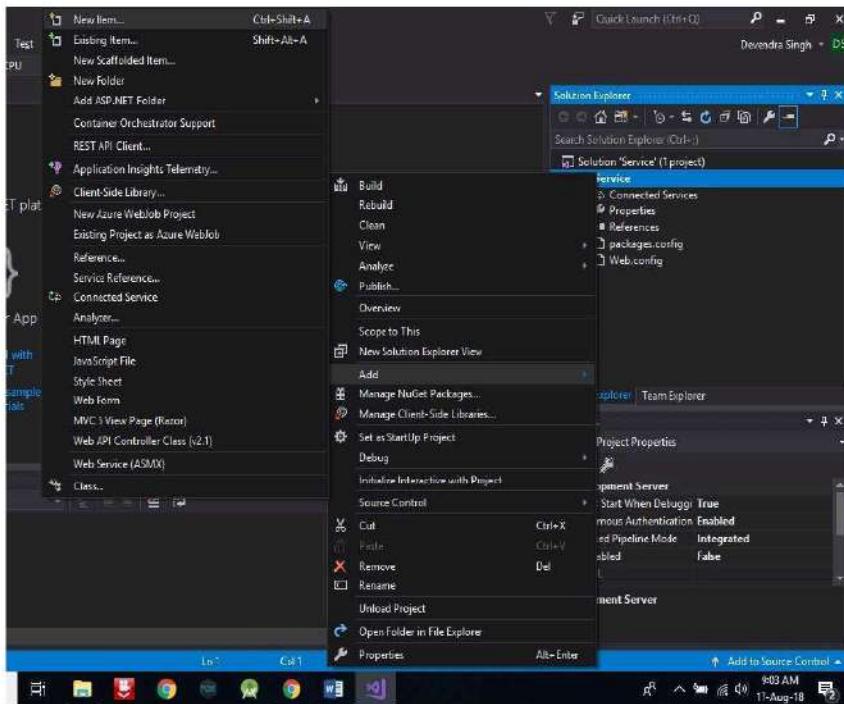
4. Select Empty and click on OK button.



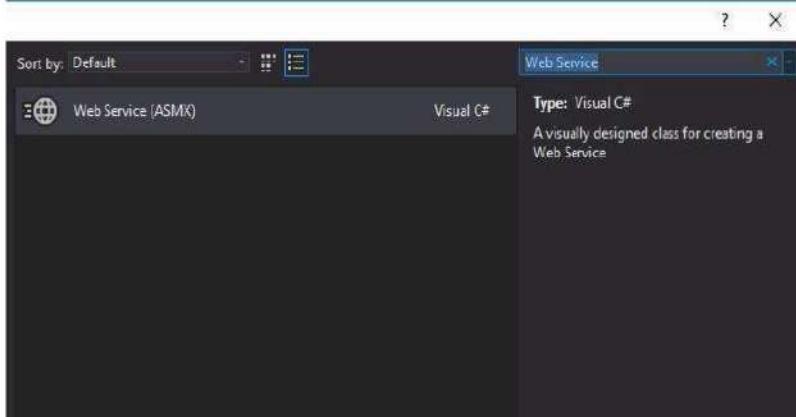
5. Now you can see, on right side in Solution Explorer Service project is created.



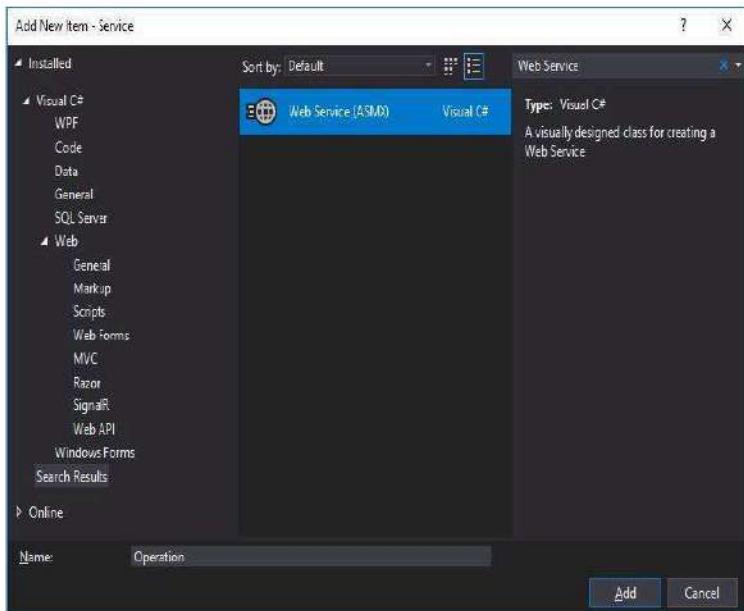
6. Right click on Service -> Add -> New Item.



7. Search for Web Service.



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



- 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 m = a * b;
    return m;
}
```

```
Operation.asmx.cs  Service
Service Operation
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
IIS Express (Google Chrome)
Operation.asmx.cs  Service
Service Operation
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 company's domain name as the root of the namespace. This way, the service can be used across different environments without having to worry about the 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` attribute. 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](#).

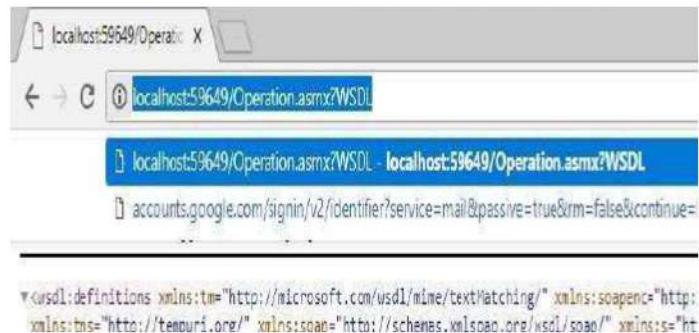
- [Add](#)
- [Multi](#)

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.

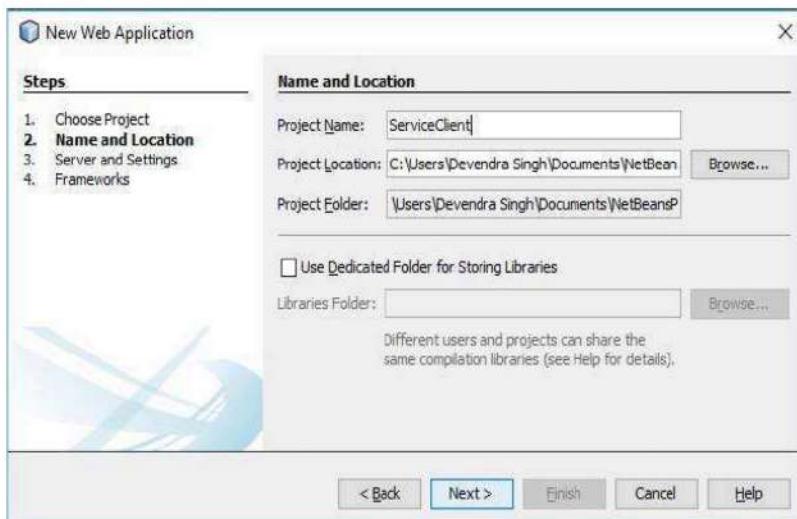
14. Now a new window will open. Select the link and copy it. We will use this link in NetBeans to consume these services.



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.

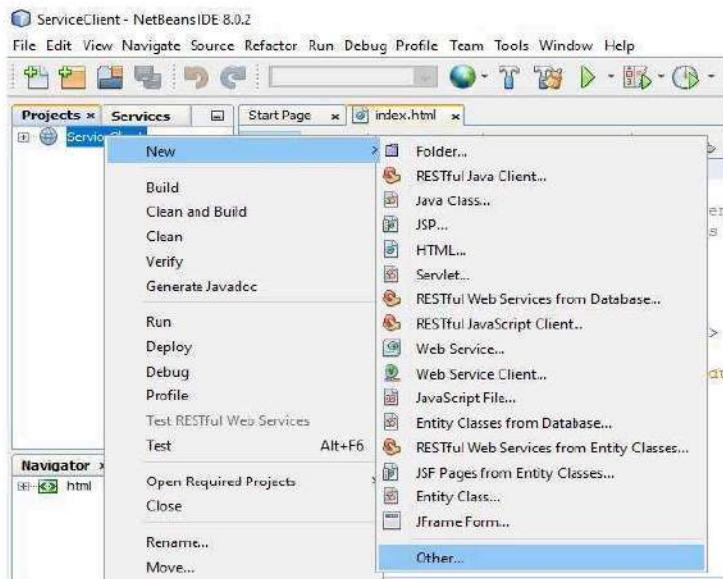
16. Now open NetBeans.

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

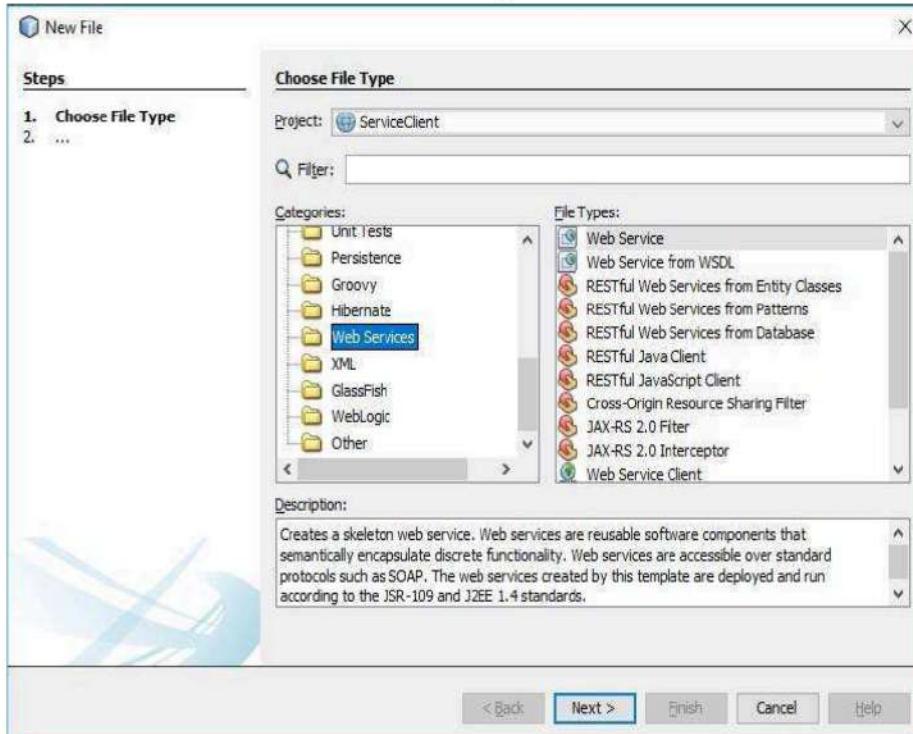


18. Now create a Web Service Client.

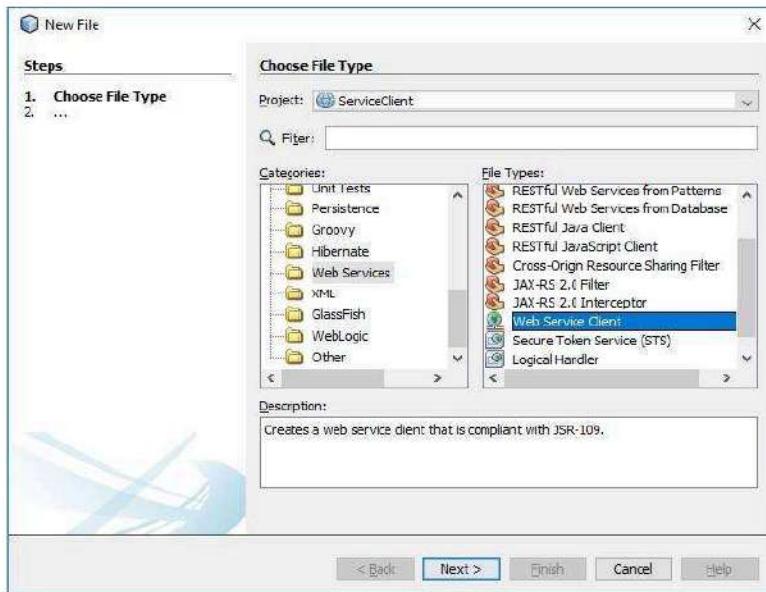
Right click on ServiceClient -> New -> Other.



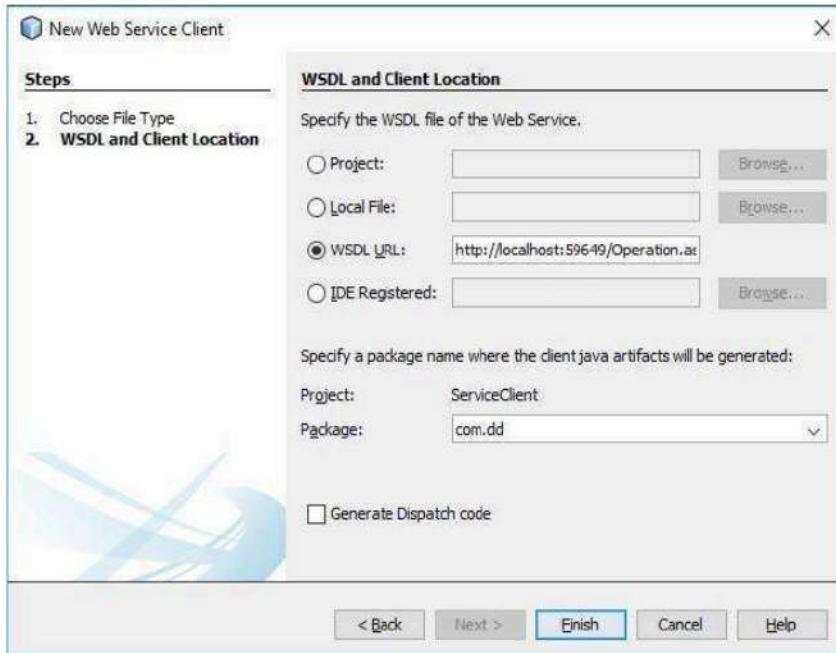
19. Now select Web Services in Categories section.

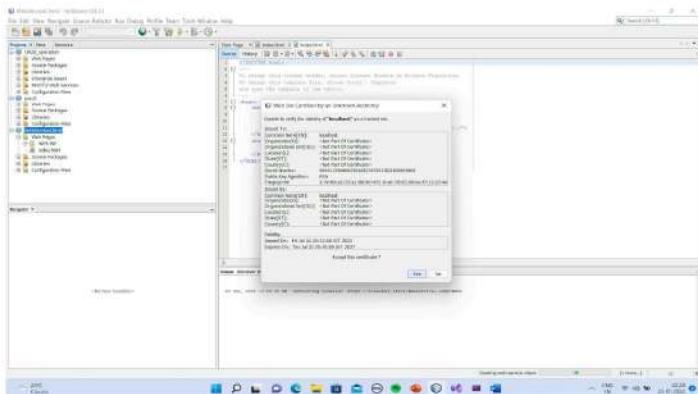


20. Select Web Service Client in File Types and Click on Next button.

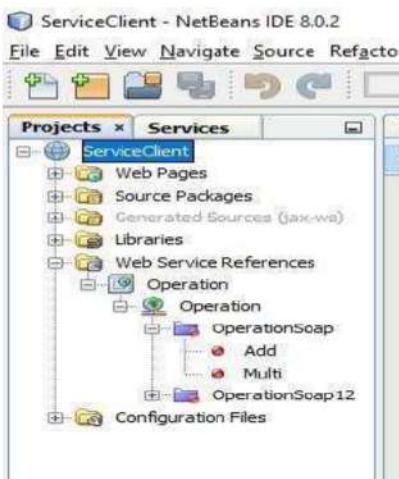


21. Select WSDL URL and paste the link that you have copied from browser on run of Visual Studio enter package name com.dd. After that click on Finish button.





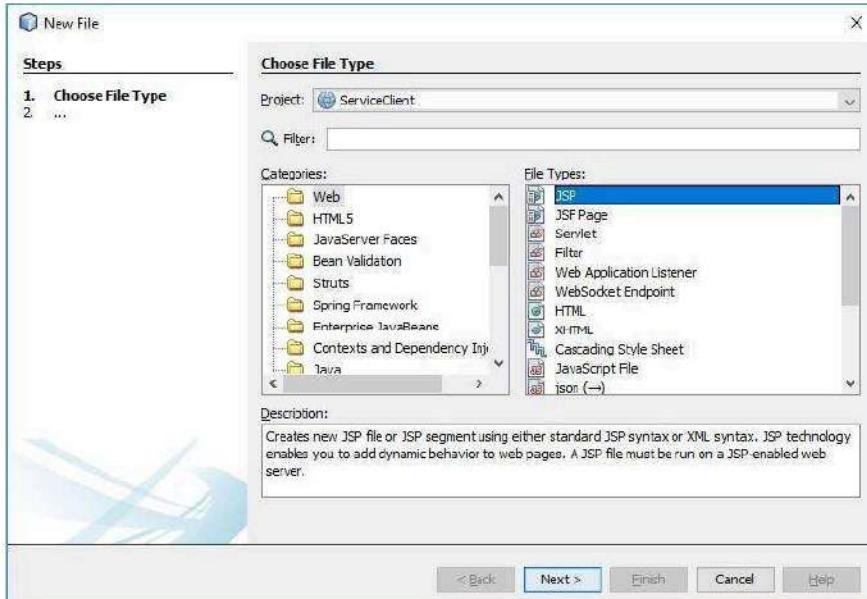
22. As you can see, we got both the service methods i.e. Add & Multi. But in this practical I'm going to use only one service method. You can do for both also.



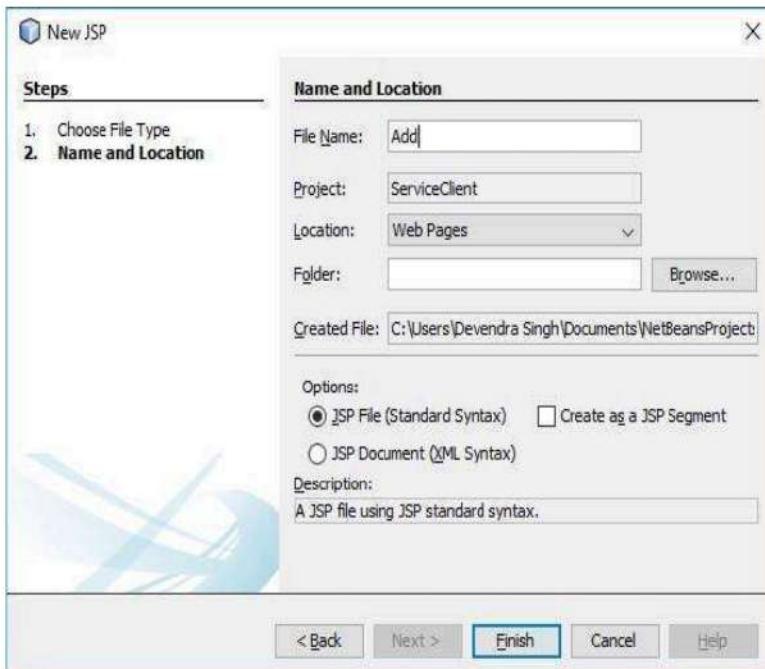
23. Now create a JSP page. Right click on ServiceClient -> New -> Other



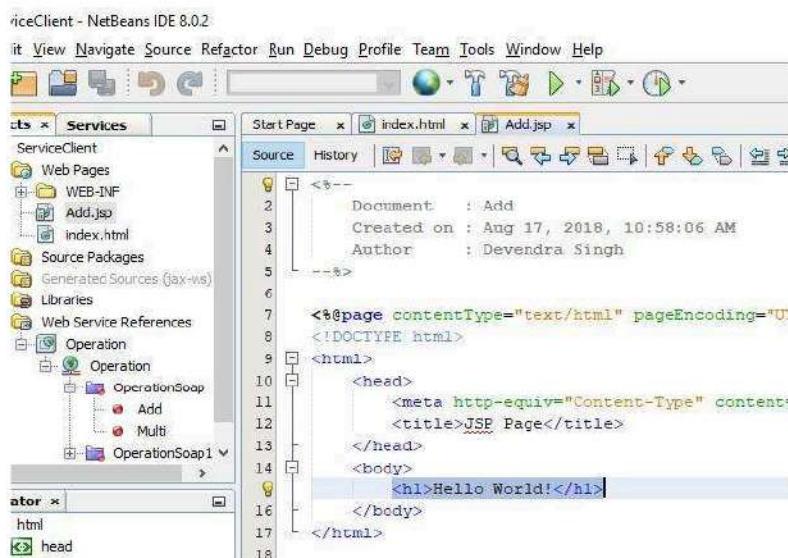
24. Select Web in Categories section -> Select JSP and click on Next button.



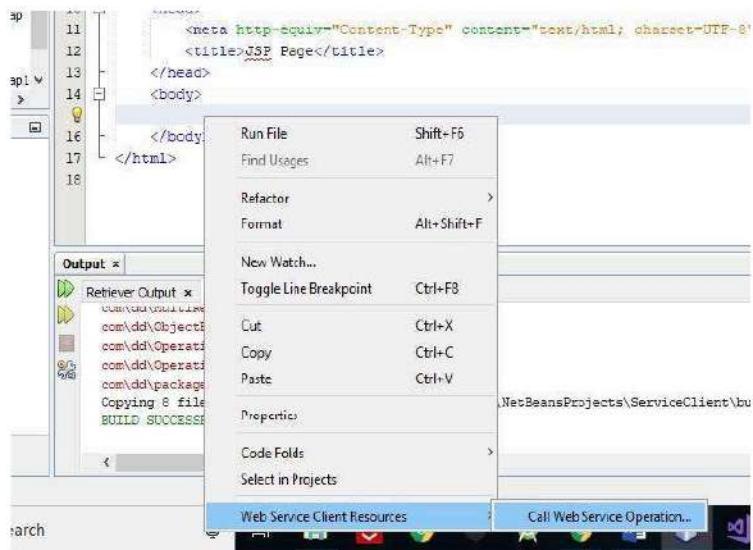
25. Enter File Name Add and click on Finish button.



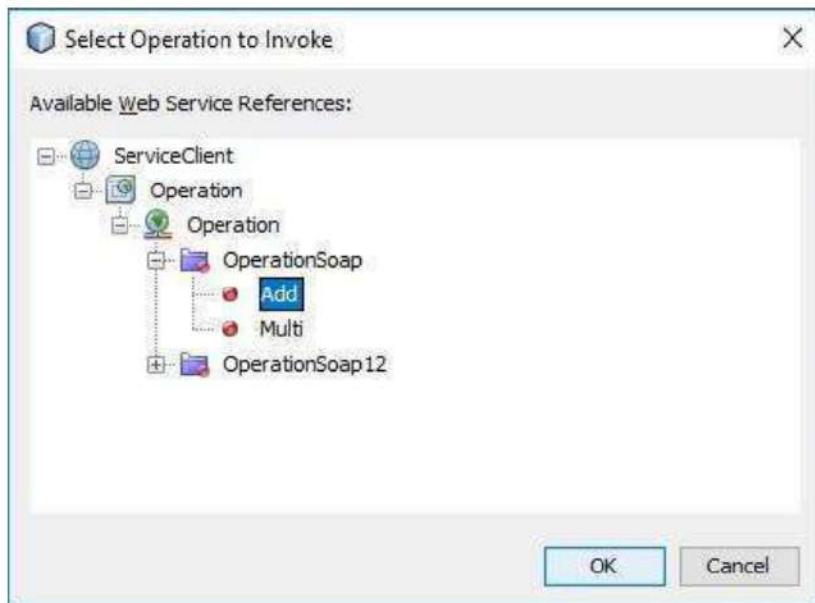
26. In Add.jsp file delete the selected part in body tag, because we don't need this.



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

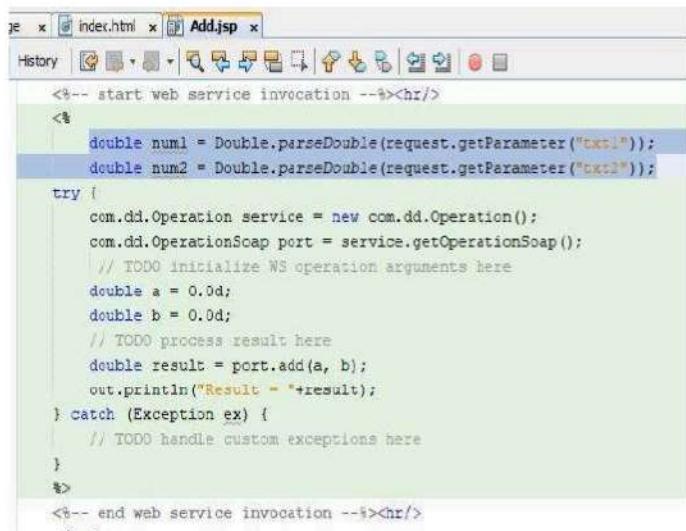


28. Expand and select Add. After select, click on OK button.



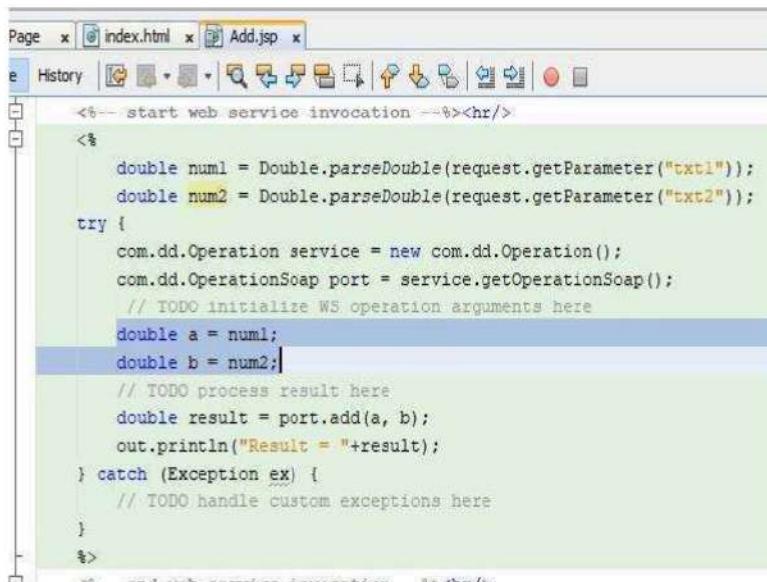
29. Now add the following code outside of try block.

```
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.



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

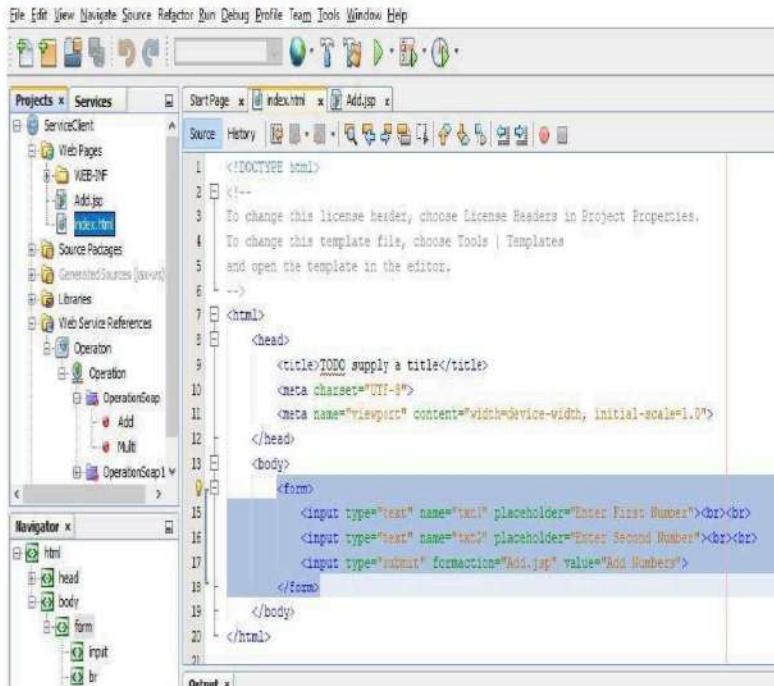
31. Now open index.html file of ServiceClient project and replace the contents of body tag with following code. After that press Ctrl+S to save it.

```
<form>
    <input type="text" name="txt1" placeholder="Enter First
Number"><br><br>
    <input type="text" name="txt2" placeholder="Enter Second
Number"><br><br>
```

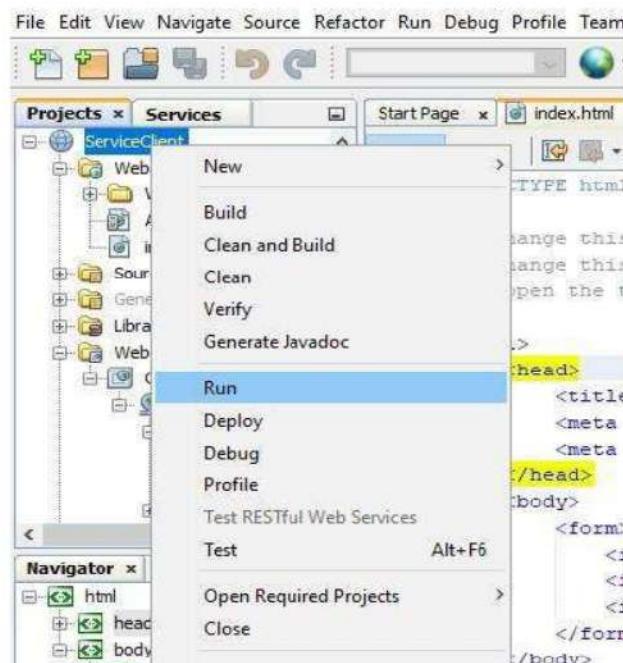
```

<input type="submit" formaction="Add.jsp" value="Add Numbers">
</form>

```



32. Now run the ServerClient web application.



33. A window will open in browser as below.

The screenshot shows a web browser window with the following details:

- Title bar: http://localhost:8080/ServiceClient
- Page title: TODO supply a title
- Toolbar: Back, Forward, Stop, Refresh, Home
- Menu bar: File, Edit, View, Favorites, Tools, Help
- Form fields:
 - Enter First Number:
 - Enter Second Number:
 - 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.

The screenshot shows a web browser window with the following details:

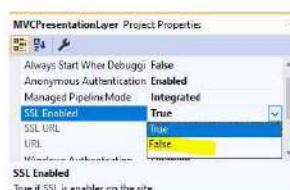
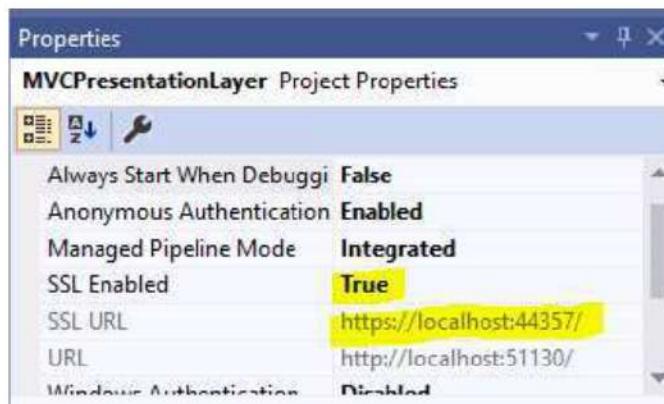
- Title bar: http://localhost:8080/ServiceClient/
- Page title: TODO supply a title
- Toolbar: Back, Forward, Stop, Refresh, Home
- Menu bar: File, Edit, View, Favorites, Tools, Help
- Form fields:
 - 19:
 - 19:
 - Add Numbers:



Note : In visual studio 22 if you don't get the answer follow the following steps

The Solution

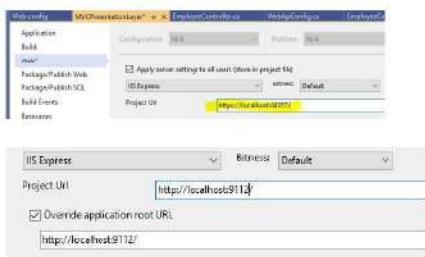
To resolve this issue, just follow below steps. We can remove SSL from web properties. If you try to set property "SSL Enabled" to false it will throw below error.



You can't remove SSL from this site because this Web project is currently configured to browse with this URL. You need to use the Web property page to change the project URL to the non-secure URL first.



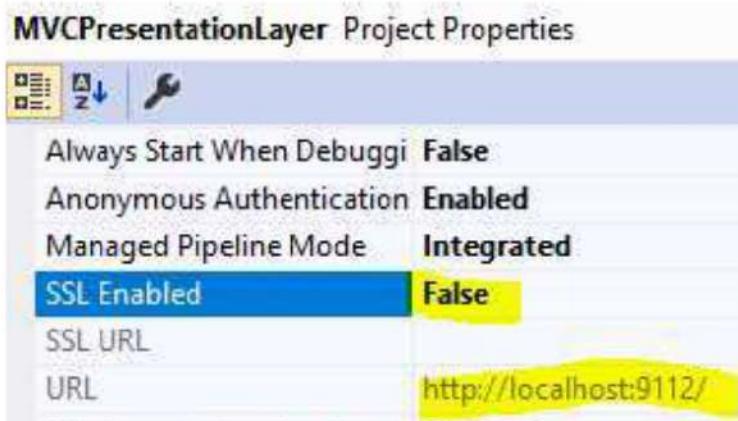
For that first open web property and change "https" to "http"

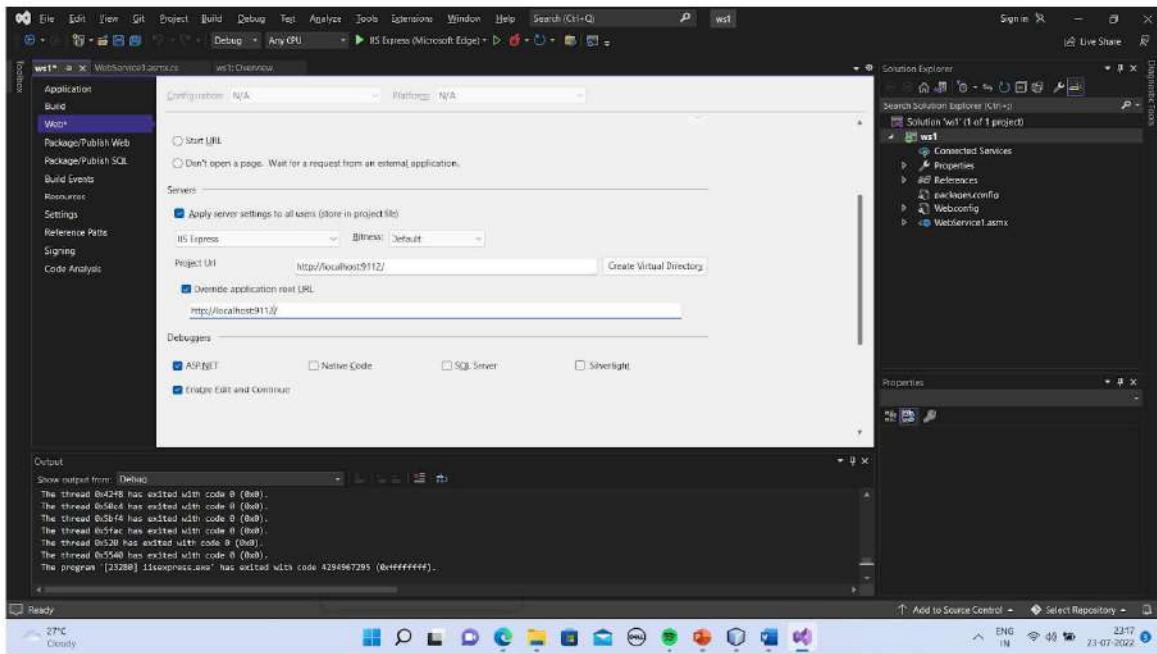


It will popup a message and ask to create a virtual directory to configure the application. Click "Yes" to allow.



Once setting are saved, you can disable the SSL from project property and you are done.





Practical - 3

Aim: Write a program to implement RESTFull WebService.

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



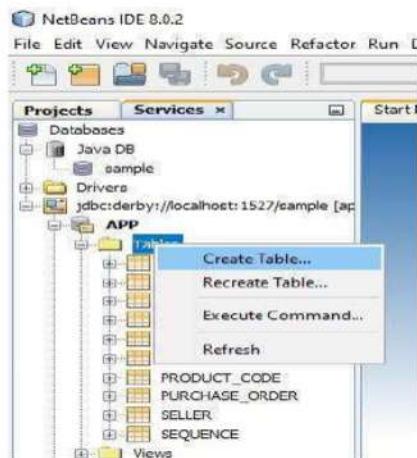
2. Right click on Java DB and then click on Start Server to start the server .



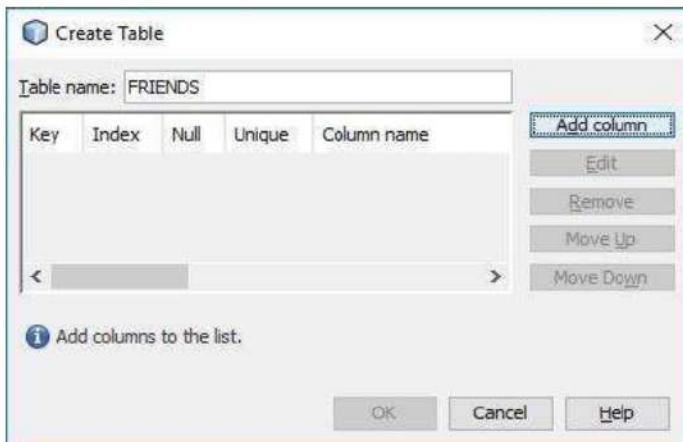
3. Now expand Java DB and right click on sample and then click on connect to connect the sample database with server.



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

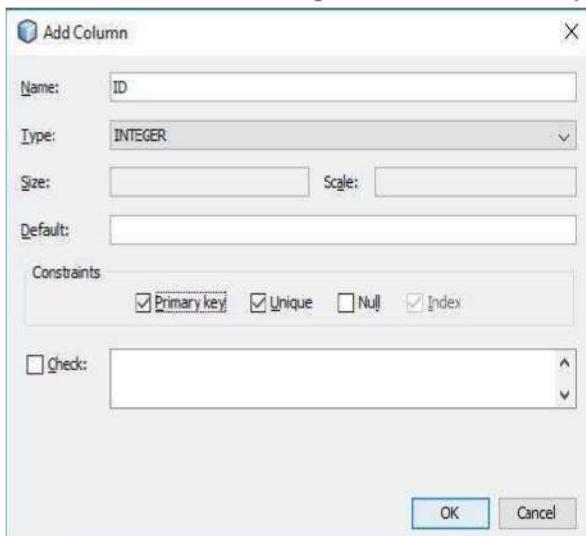


5. Give table name as FRIENDS.

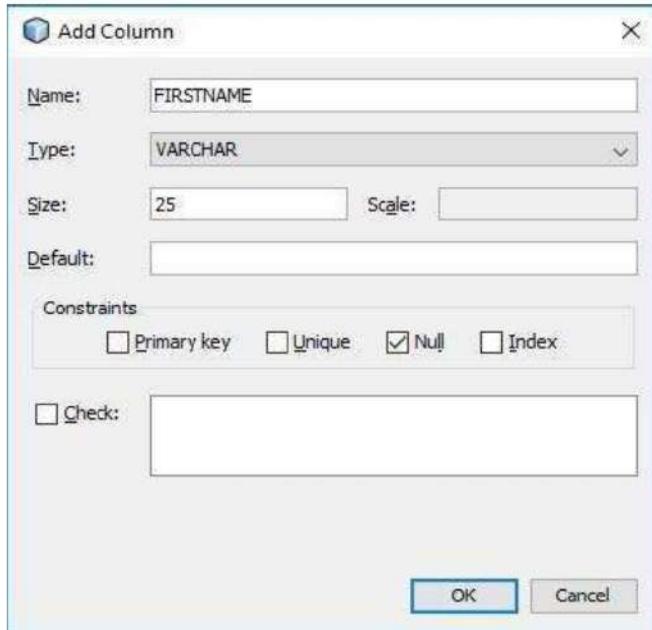


6. click on Add column button to add columns in table.

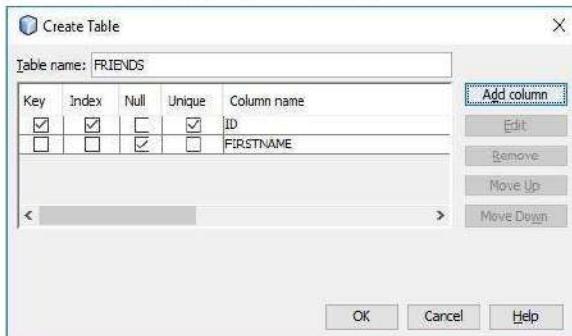
Enter details as in below pic and select Primary key. After that click on OK button.



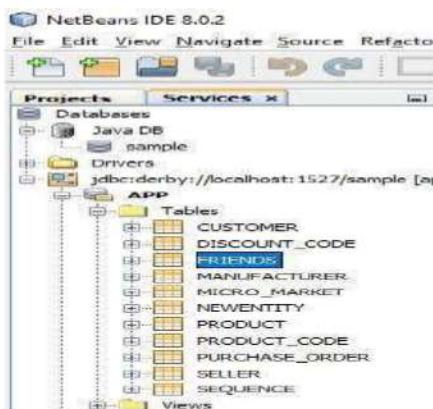
7. Now add second column with following detail. But don't select primary and click on OK button.



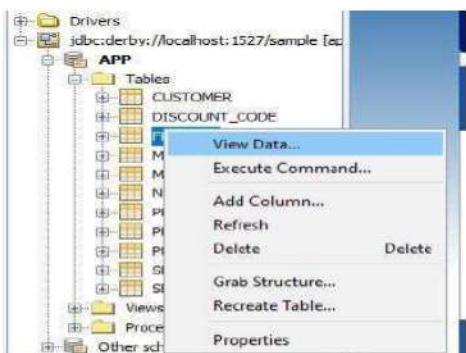
8. click on OK button.



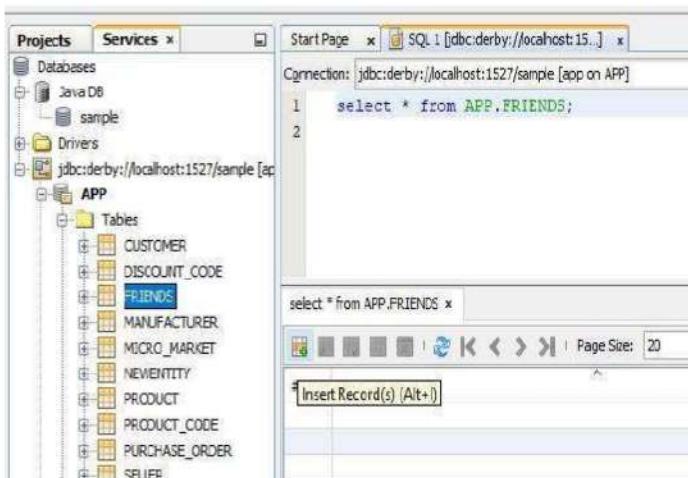
9. Now you can see a table with name FRIENDS in the table.



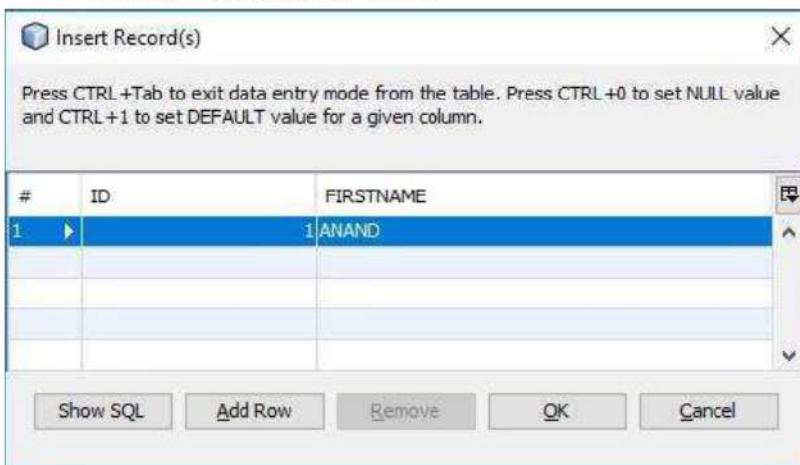
10. Right click on FRIENDS to view and add records into it.



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



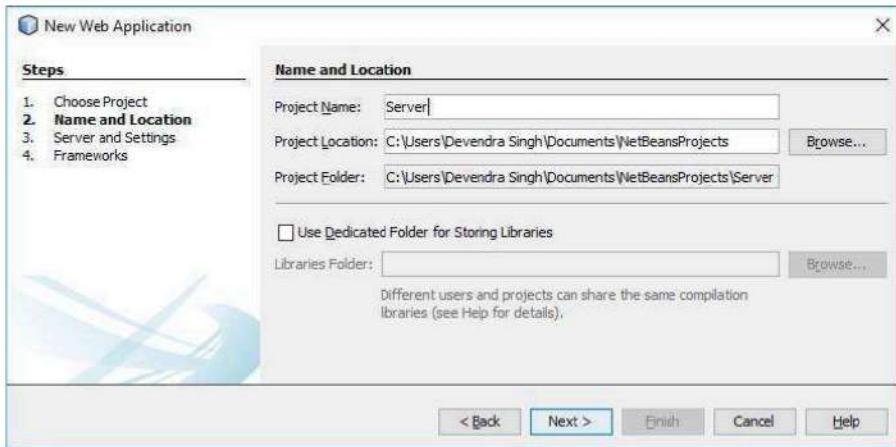
12. Insert a record and then click on Add Row button to insert more record. After that click on OK button to finish.



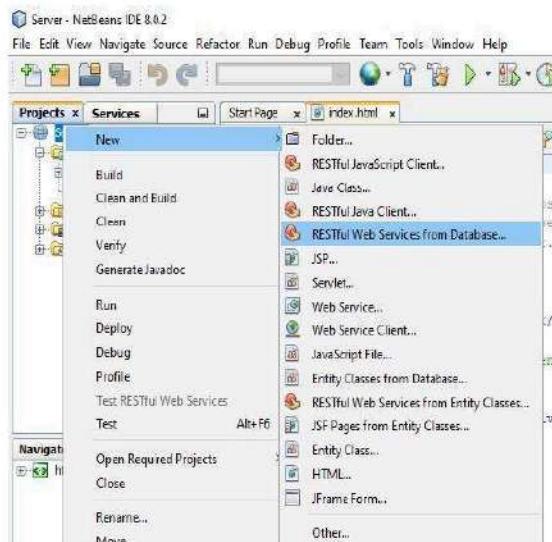
13. As you can see, I have entered 7 records.

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

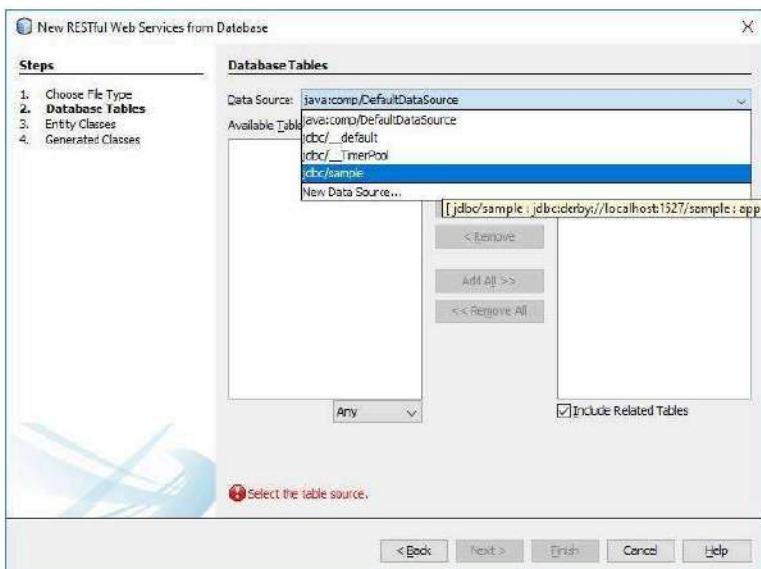
14. create a web application with name Server. After that click on Next and then Finish button.



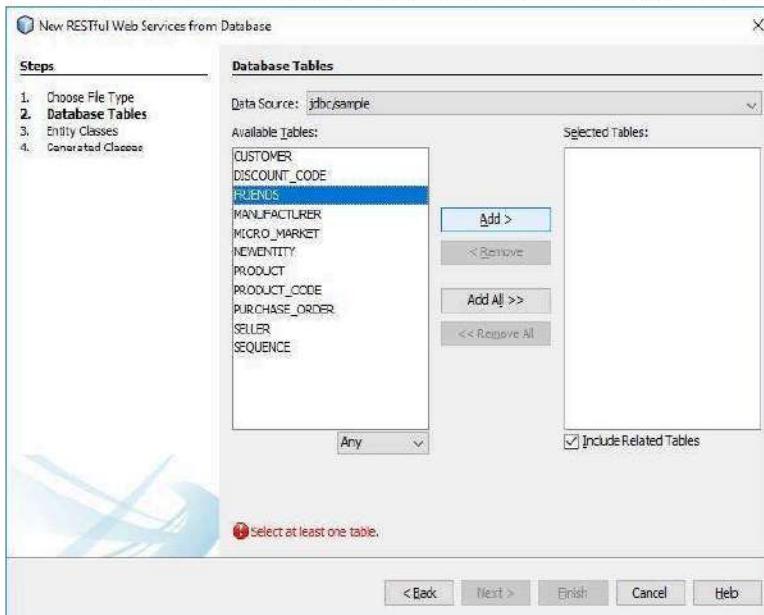
15. Now create a RESTful Web Service from Database by right click on project name.



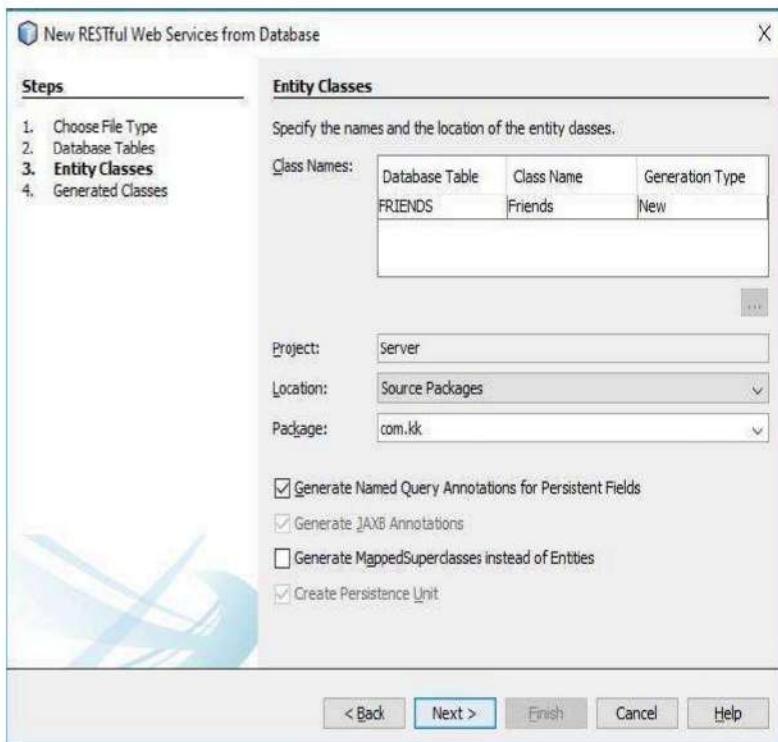
16. Choose Data Source jdbc/sample.



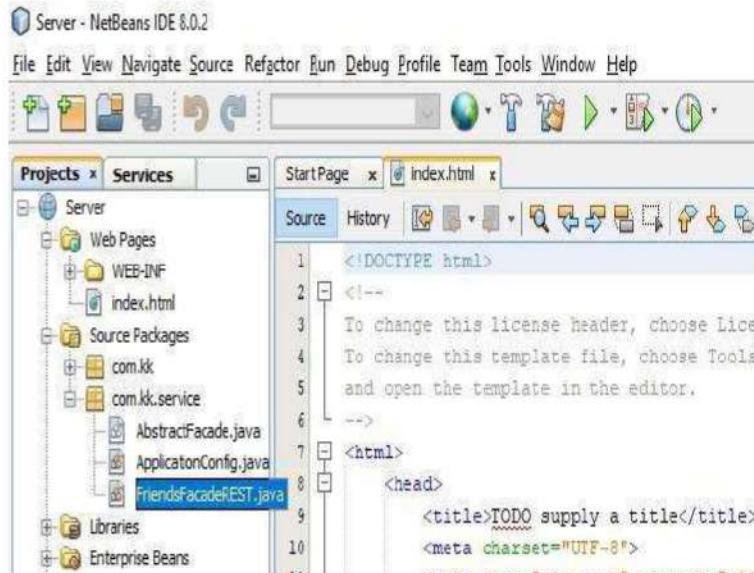
17. Now select FRIENDS and click on Add button. After that click on Next button.



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.

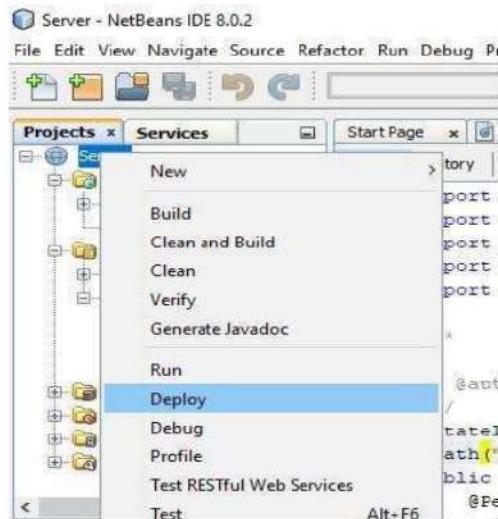
Note : If you get error “javax.ws.rs” doesnot exist, add Java EE 6 API Library in NetBeans IDE by doing myProject->Properties->Libraries->Add Library

```

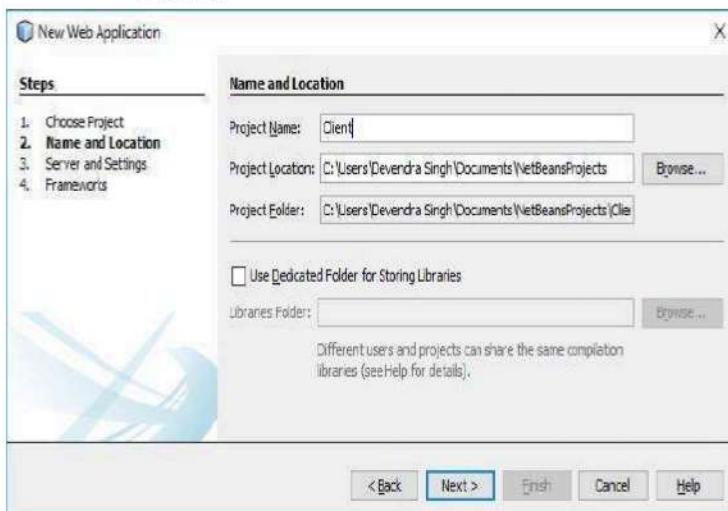
25  /*
26  * Stateless
27  * @Path("com.kk.friends")
28  */
29  public class FriendsFacadeREST extends AbstractFacade<Friends> {
30     @PersistenceContext(unitName = "ServerPU")
31     private EntityManager em;
32
33     public FriendsFacadeREST() {
34         super(Friends.class);
35     }
36
37  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

```

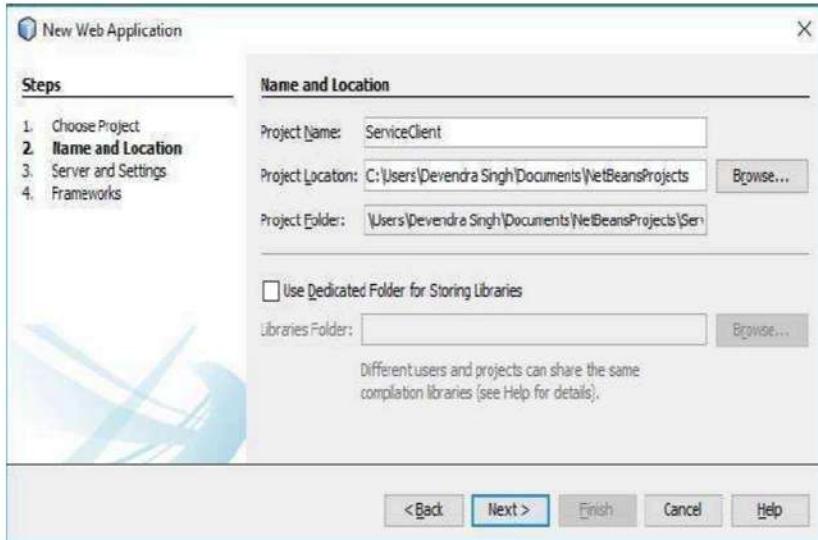
21. After that right click on project name and Deploy it.



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

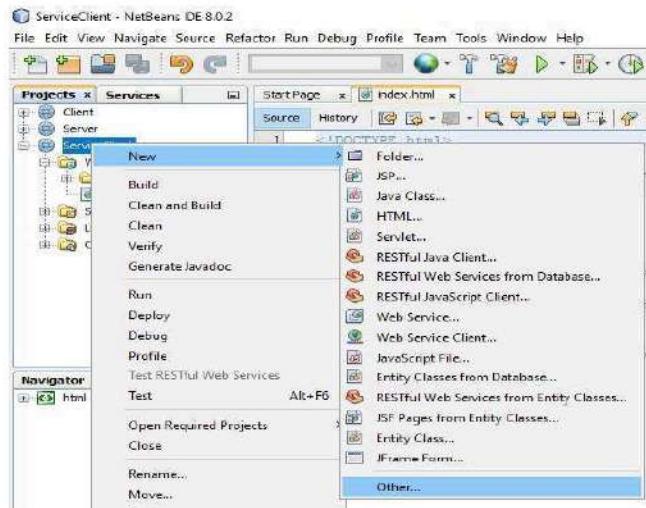


23. Create a Web Application with name ServiceClient.

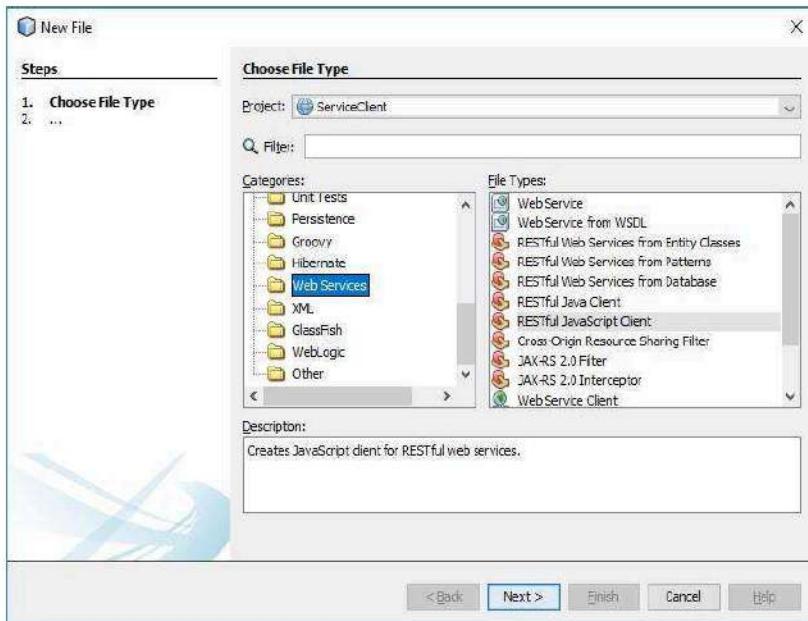


24. Now create a RESTful Java Client.

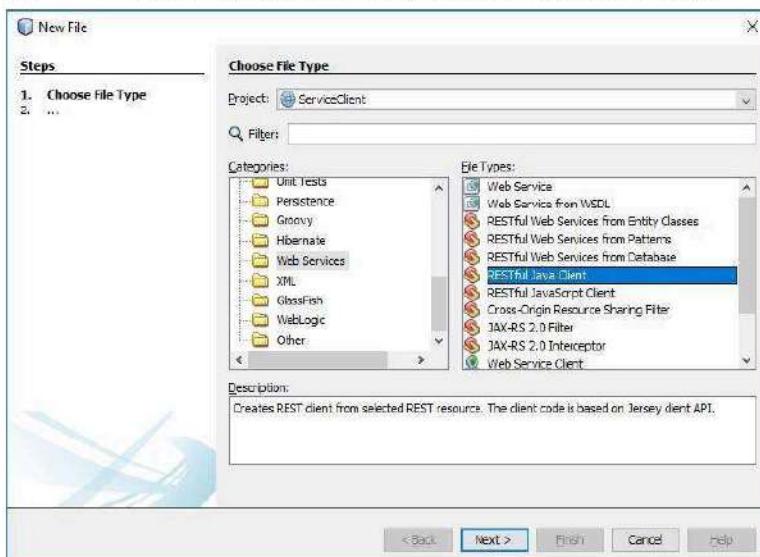
Right click on ServiceClient -> New -> Other.



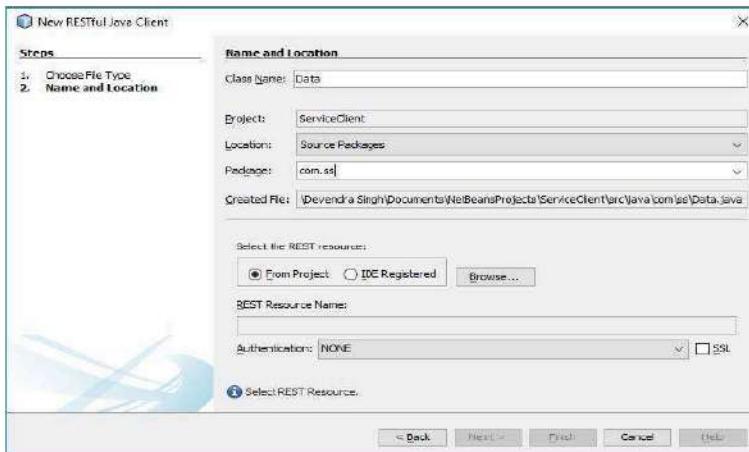
25. Drag down and select Web Services and in side panel select RESTful Java Client.



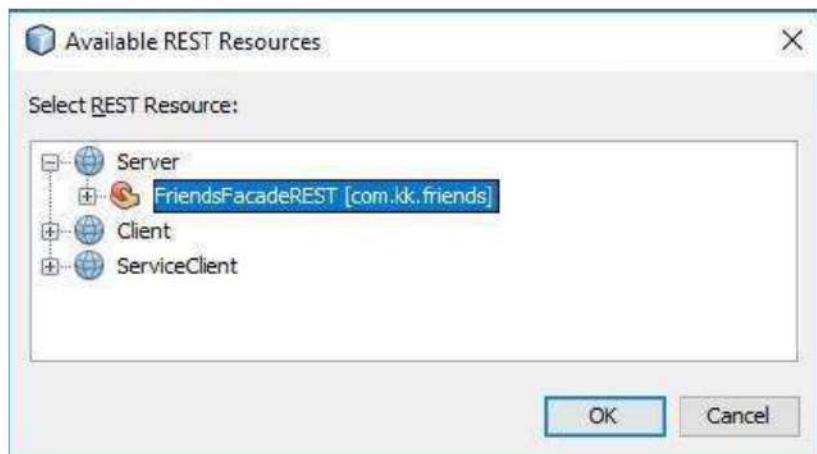
26. After select RESTful Java Client click on Next.



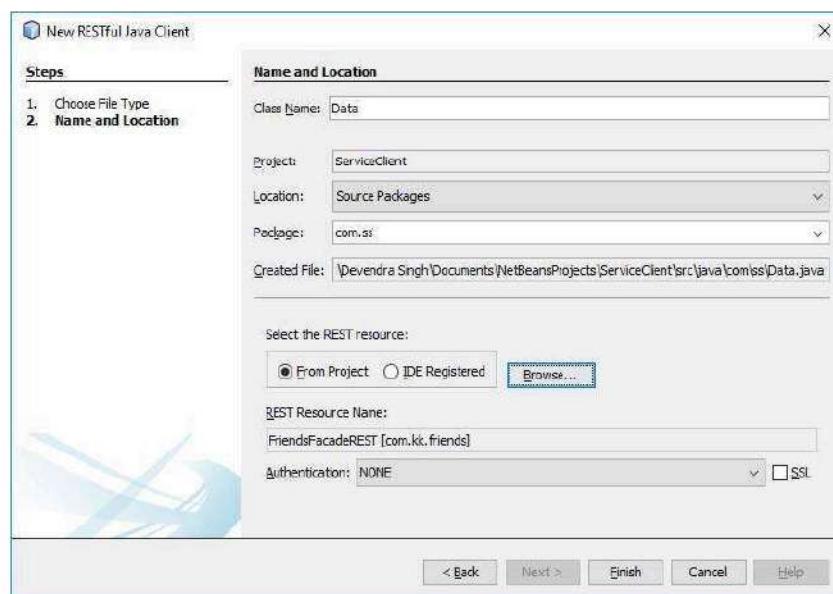
27. Enter following data. Class Name -> Data
Package -> com.ss



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

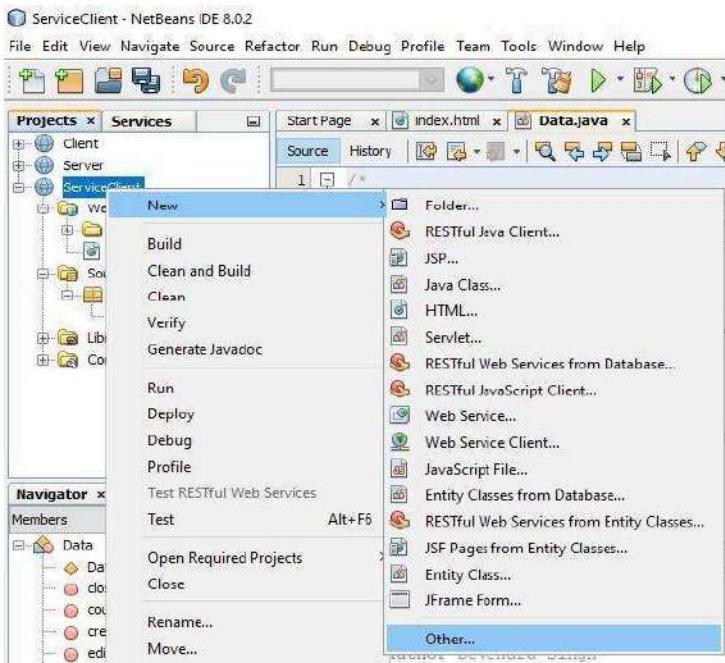


29. Click on Finish.

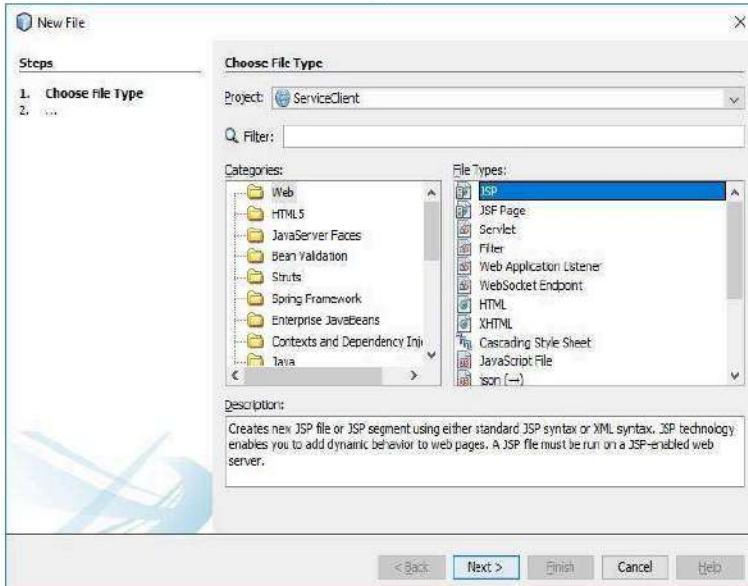


30. Now create a JSP page.

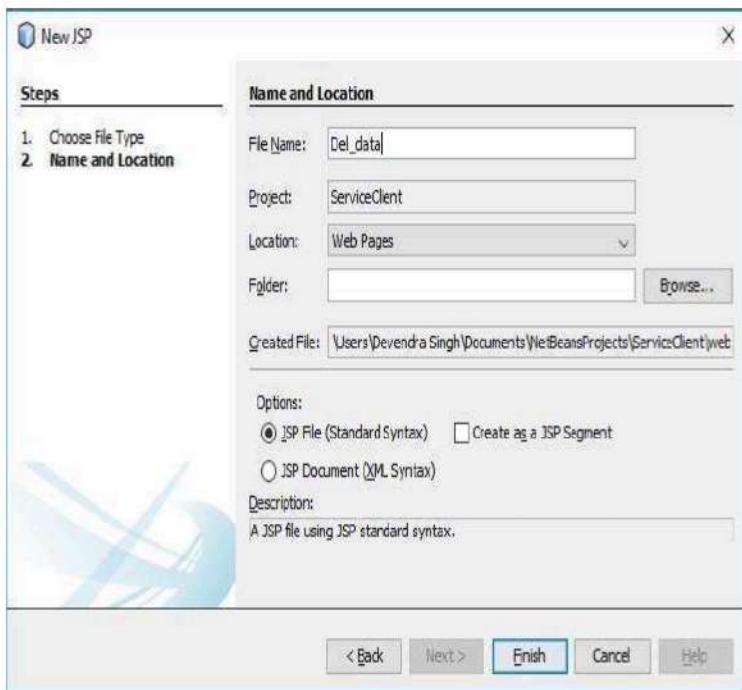
Right click on ServiceClient -> New -> Other



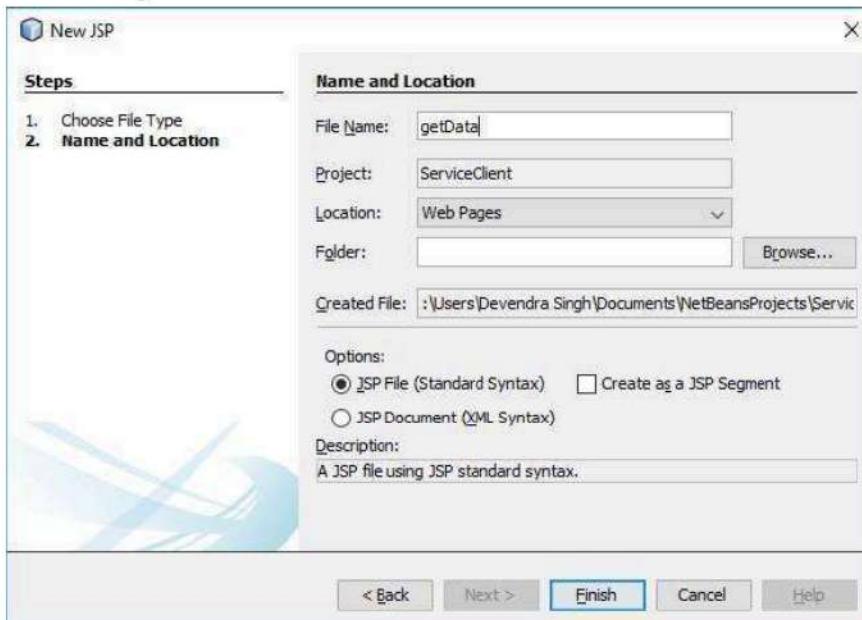
31. Select Web in Categories section -> Select JSP and click on Next button.



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

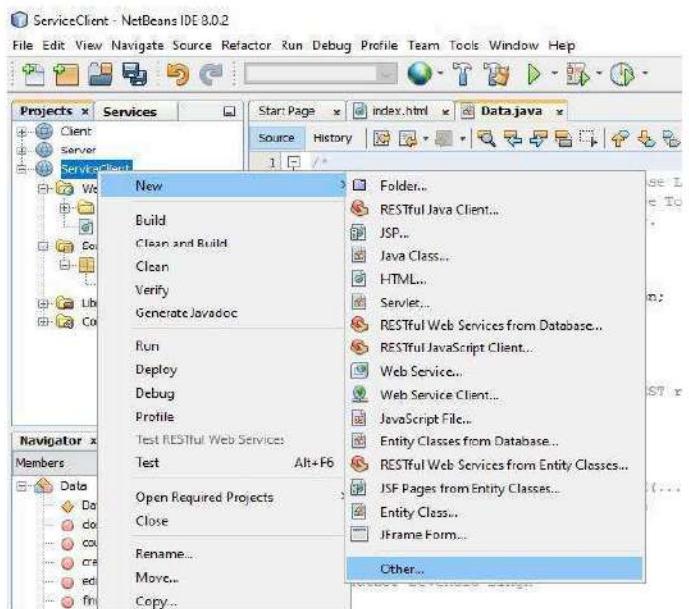


33. Now create one more JSP file by follow the step number 30, 31 & 32. But File Name will be getData.

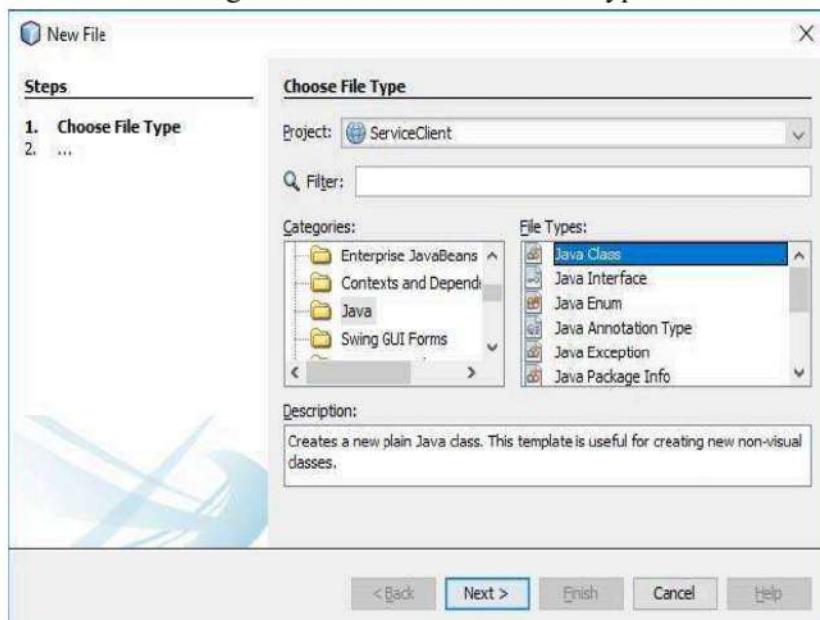


34. Now create a Java class.

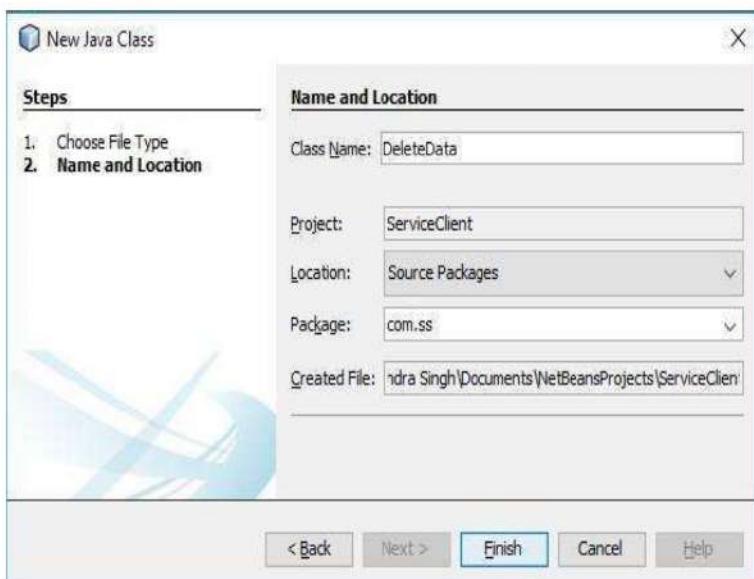
Right click on ServiceClient -> New -> Other



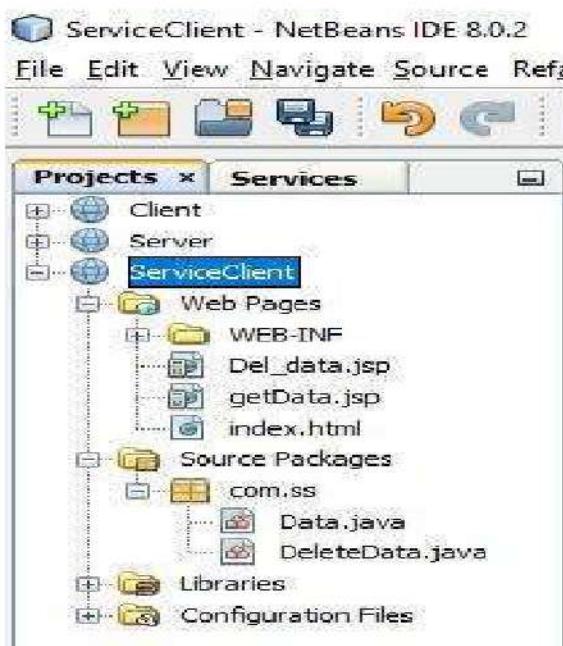
35. In Categories select Java and in File Types select Java Class. Click Next button.



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



37. Your project file structure will look like below.



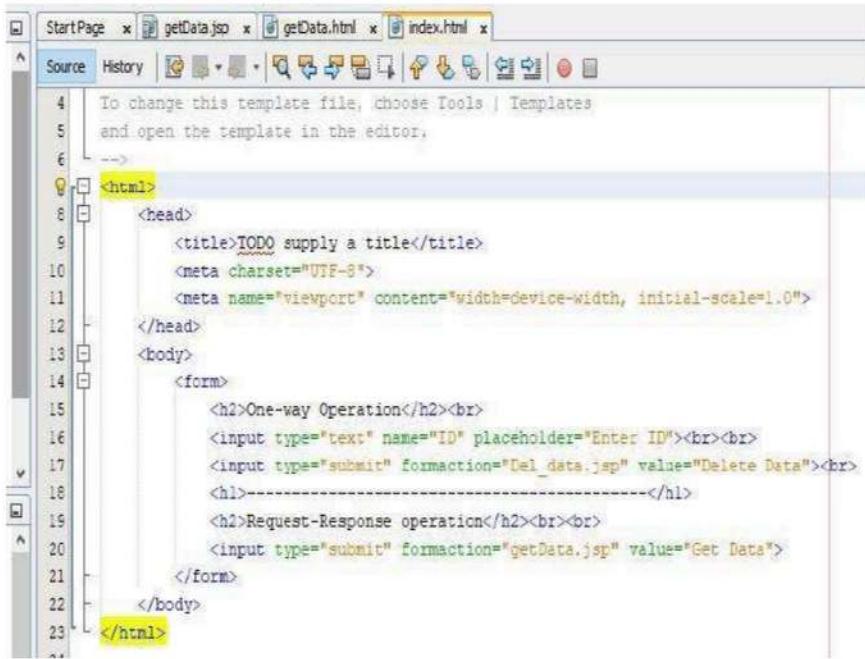
38. Now open the index.html of ServiceClient project by double click on it and add the following code in between body tag.

```
<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>
```

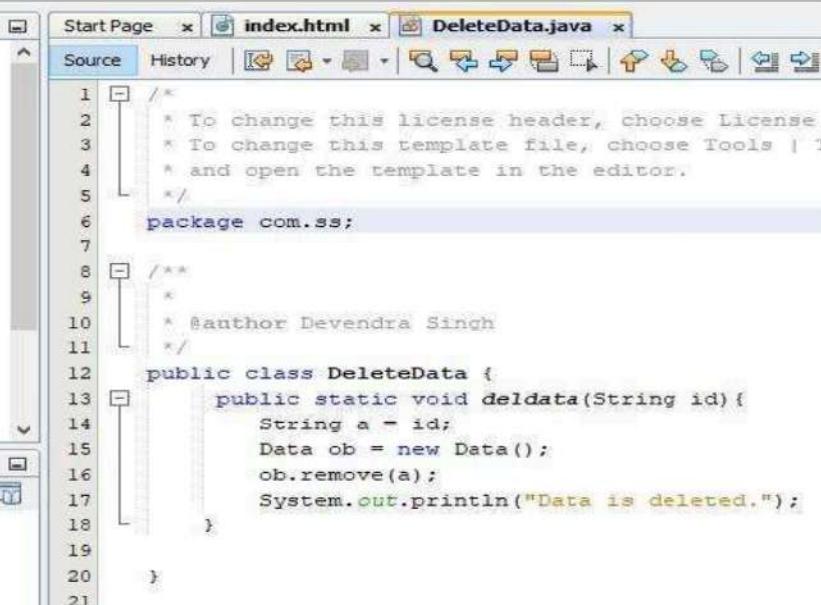
```
<input type="submit" formaction="getData.jsp" value="Get Data">
</form>
```



```
4 To change this template file, choose Tools | Templates
5 and open the template in the editor.
6 -->
7 <html>
8   <head>
9     <title>TODO supply a title</title>
10    <meta charset="UTF-8">
11    <meta name="viewport" content="width=device-width, initial-scale=1.0">
12  </head>
13  <body>
14    <form>
15      <h2>One-way Operation</h2><br>
16      <input type="text" name="ID" placeholder="Enter ID"><br><br>
17      <input type="submit" formaction="Del_data.jsp" value="Delete Data"><br>
18      <h1>-----</h1>
19    <h2>Request-Response operation</h2><br><br>
20    <input type="submit" formaction="getData.jsp" value="Get Data">
21  </form>
22 </body>
23 </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.");
}
```



```

1  /*
2   * To change this license header, choose License |
3   * To change this template file, choose Tools | T
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

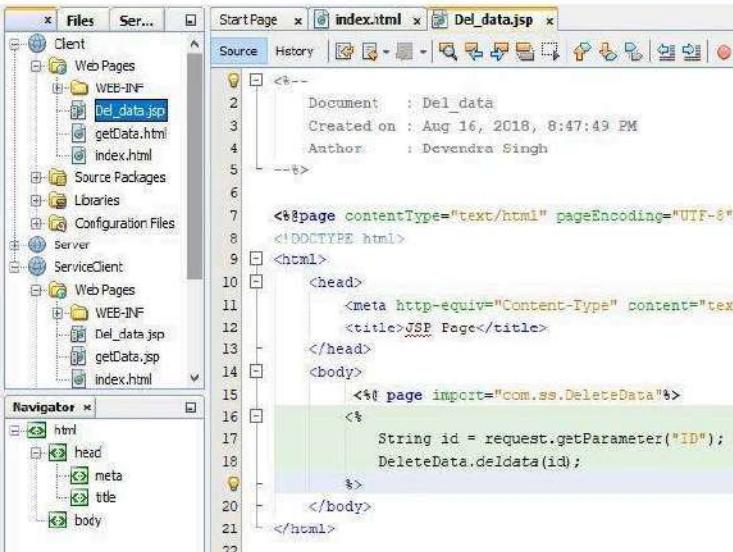
```

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);
%>

```



```

<%-- Document : Del_data
Created on : Aug 16, 2018, 8:47:49 PM
Author : Devendra Singh
--%>

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

```

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, initialscale=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>

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

</body>

```

```

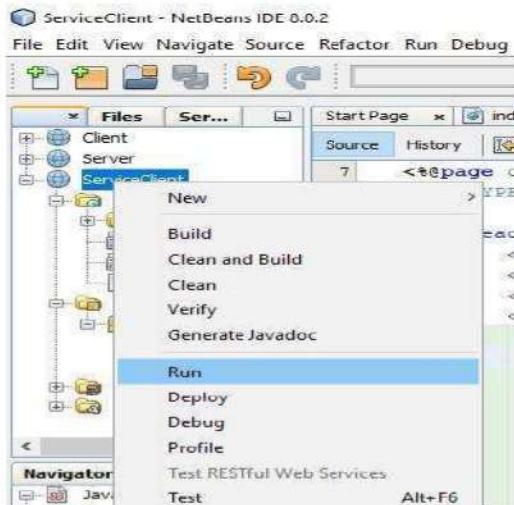
<!DOCTYPE html>
<html>
    <head>
        <title>TODO supply a title</title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <style>
            table {
                font-family: arial, sans-serif;
                border-collapse: 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/webservices/com.mk.Friends/');
            request.onload = function () {
                // begin processing JSON data here
                var data = JSON.parse(this.response);

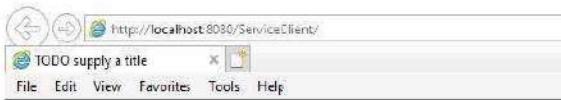
                for (var i = 0; i < data.length; i++) {
                    var table = document.getElementById("myTable");
                    var row = table.insertRow();
                    var cell1 = row.insertCell(0);
                    var cell2 = row.insertCell(1);
                    var cell3 = row.insertCell(2);
                }
            };
        </script>
    </head>
    <body>
        <table border="1">
            <thead>
                <tr>
                    <th>Name</th>
                    <th>Age</th>
                    <th>Address</th>
                </tr>
            </thead>
            <tbody>
                <tr>
                    <td>John Doe</td>
                    <td>25</td>
                    <td>Highway 141</td>
                </tr>
                <tr>
                    <td>Jane Doe</td>
                    <td>25</td>
                    <td>Highway 141</td>
                </tr>
            </tbody>
        </table>
    </body>
</html>

```

42. Now Run the ServiceClient project.



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



One-way Operation

Request-Response operation

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.

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.



Request-Response operation

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. The address bar shows the URL `http://localhost:8080/S`. The title bar says "TODO supply a title". 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

Practical - 4

Aim: Develop application to consume Google's search / Google's Map RESTful Web service.

STEPS:

1. First of all we need to create a Java Web Application with any name, let it be GoogleMap here using Netbeans IDE.
2. The code inside the input.jsp will be similar to this input.jsp

```
<%@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>
    <form action="index.jsp">
        <pre>
            Enter latitude:<input type="text" name="t1" />
            Enter longitude:<input type="text" name="t2" />
            <input type="submit" value="Show" />
        </pre>
    </form>
</body>
</html>
```

3. Before running the application we need the Google API key. The steps are shown here:
 - Visit Google APIs Console (<https://console.developers.google.com>, you have to login with your Google account)

Create a new API Project.

Select a project

NEW PROJECT

Search projects and folders

RECENT ALL

Name	ID
GoogleMap <small>?</small>	eminent-kit-237101
Practical6 <small>?</small>	practical6-235906
My Project <small>?</small>	lateral-shore-180303

CANCEL OPEN

- Enter the name to your project.

https://mahad X | M New announce X | M Sc CS 2018 X | syllabus_msc_ X | Java Web Serv X

← → C https://console.developers.google.com/projectcreate?previousPage=%2Fapis%2Fdashboard

≡ Google APIs

New Project

⚠ You have 7 projects remaining in your quota. Request an increase or delete projects. [Learn more](#)

[MANAGE QUOTAS](#)

Project name * ?

Project ID: zeta-environs-237517. It cannot be changed later. [EDIT](#)

Location * BROWSE

Parent organization or folder

CREATE CANCEL

- Enable the Google Maps API V3.

The screenshot shows a browser window with multiple tabs open at the top. The active tab is 'https://console.developers.google.com/apis/library/maps-backend.googleapis.com?id=fd73ab50-9916-4cde-a0f6-dc8be0a0d425&q=maps'. The main content area is titled 'Maps JavaScript API' and is categorized under 'Google'. It features a circular icon with 'JS' and a map, and a button labeled 'MANAGE' with a green checkmark and the text 'API enabled'. Below this, there's a section for 'Maps for your website'.

Type

APIs & services

Last updated

1/10/19, 2:02 AM

Category

Maps

Service name

maps-backend.googleapis.com

Overview

Add a map to your website, providing imagery and local data from the same source as Google Maps. Style the map to suit your needs. Visualize your own data on the map, bring the world to life with Street View, and use services like geocoding and directions.

About Google

Google's mission is to organize the world's information and make it universally accessible and useful. Through products and platforms like Search, Maps, Gmail, Android, Google Play, Chrome and YouTube, Google plays a meaningful role in the daily lives of billions of people.

Tutorials and documentation

4. Create another file index.jsp

Index.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>

<html>
<head>
<style>
#map {
height: 400px;
width: 100%;
}
</style>
</head>
<body>
<%

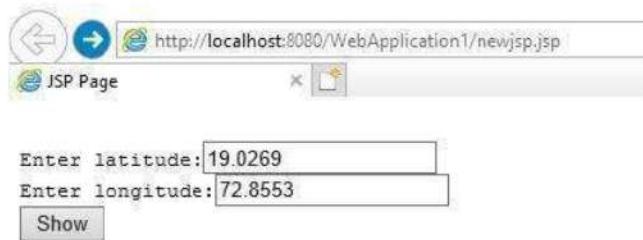
```

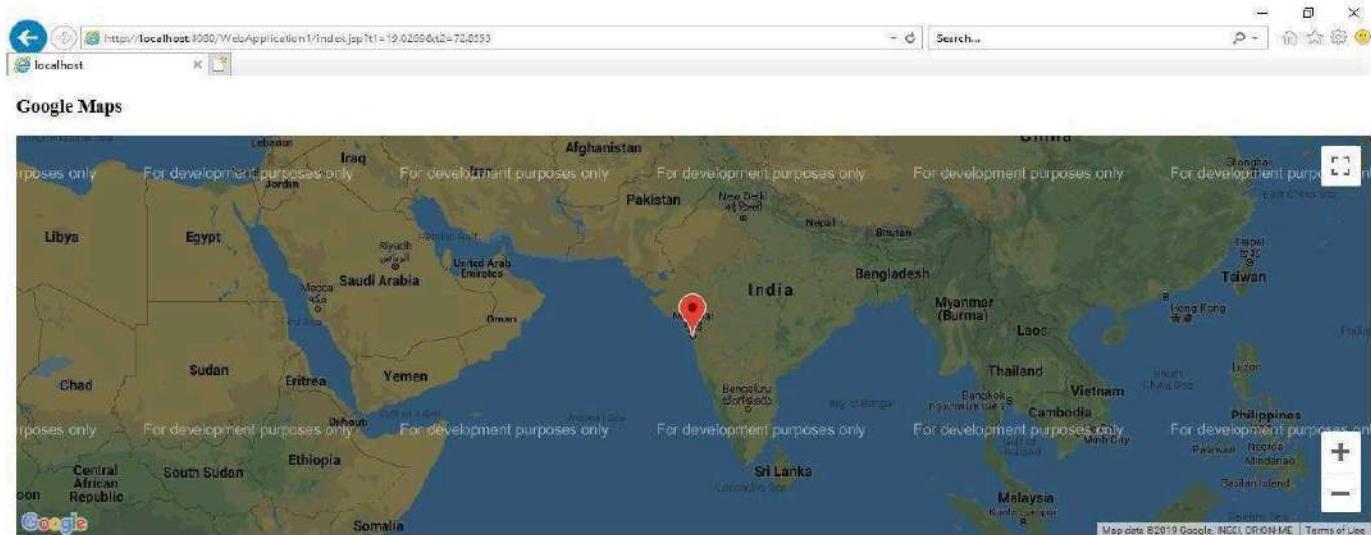
```

double lati=Double.parseDouble(request.getParameter("t1")); double
longi=Double.parseDouble(request.getParameter("t2"));
%>
<h3> Google Maps </h3>
<div id="map"></div>
<script lang="javascript">
function initMap() {
var info={lat: <%=lati%>, lng: <%=longi%>};
var map = new google.maps.Map(document.getElementById('map'),
{
zoom: 4, center: info
});
var marker = new google.maps.Marker({
position: info,
map: map
});
}
</script>
<script async defer src="put
your key here">
</script>
</body>
</html>

```

Output :-





Practical 5

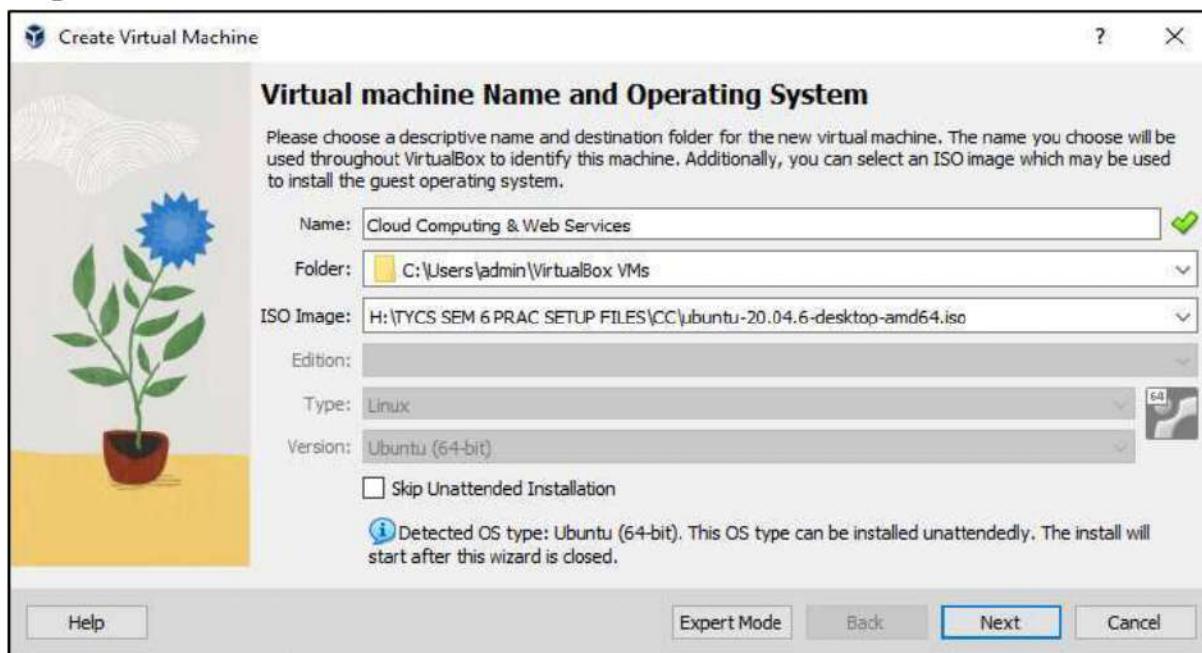
Aim: Installation and Configuration of virtualization using KVM.

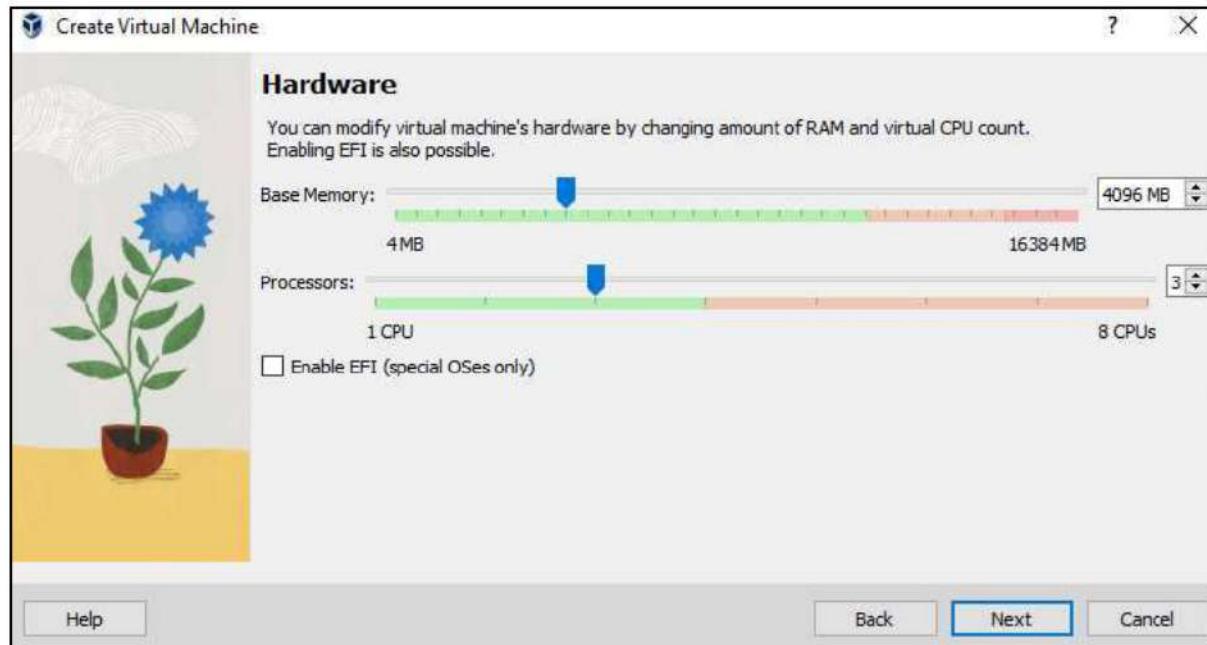
Theory :

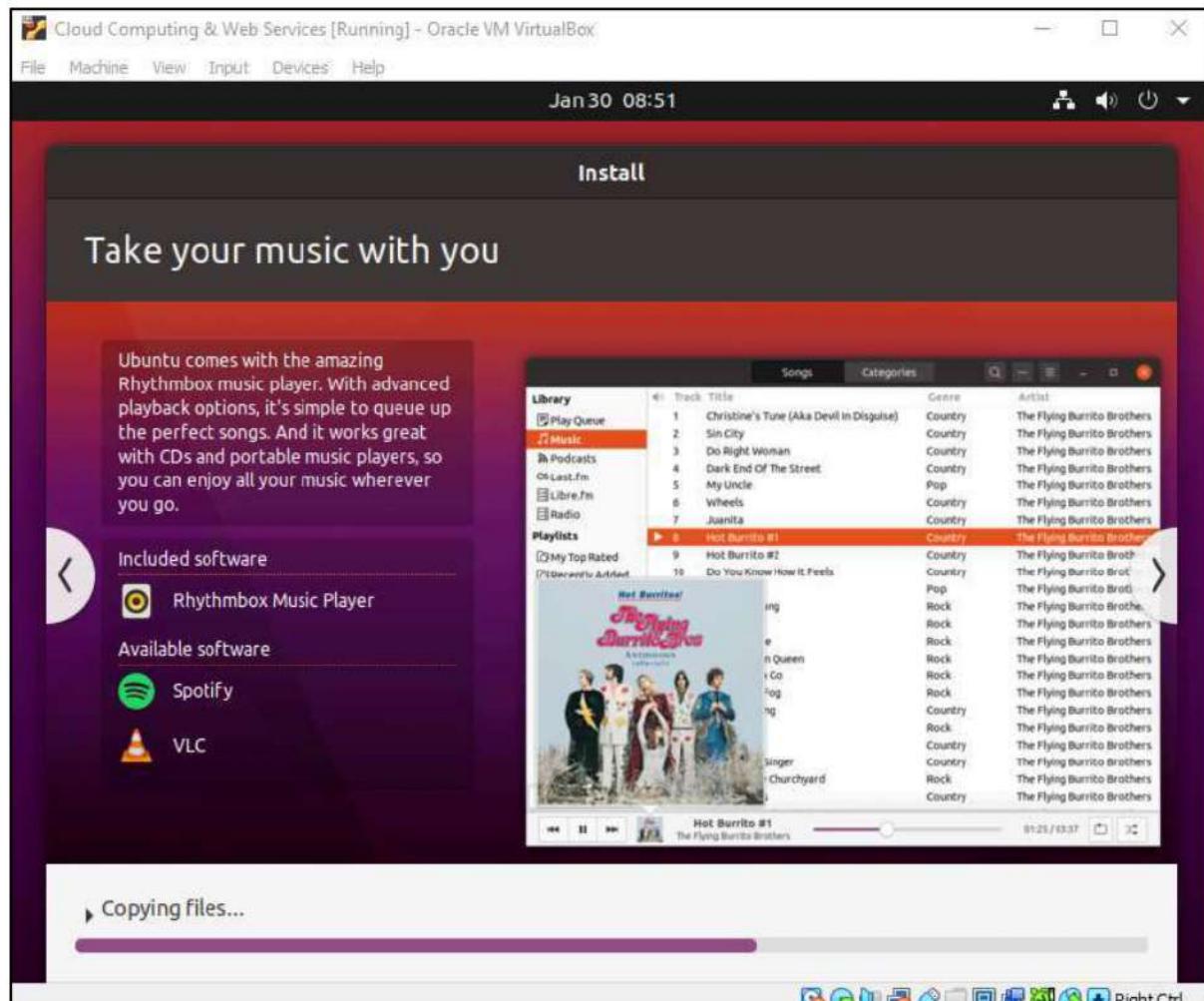
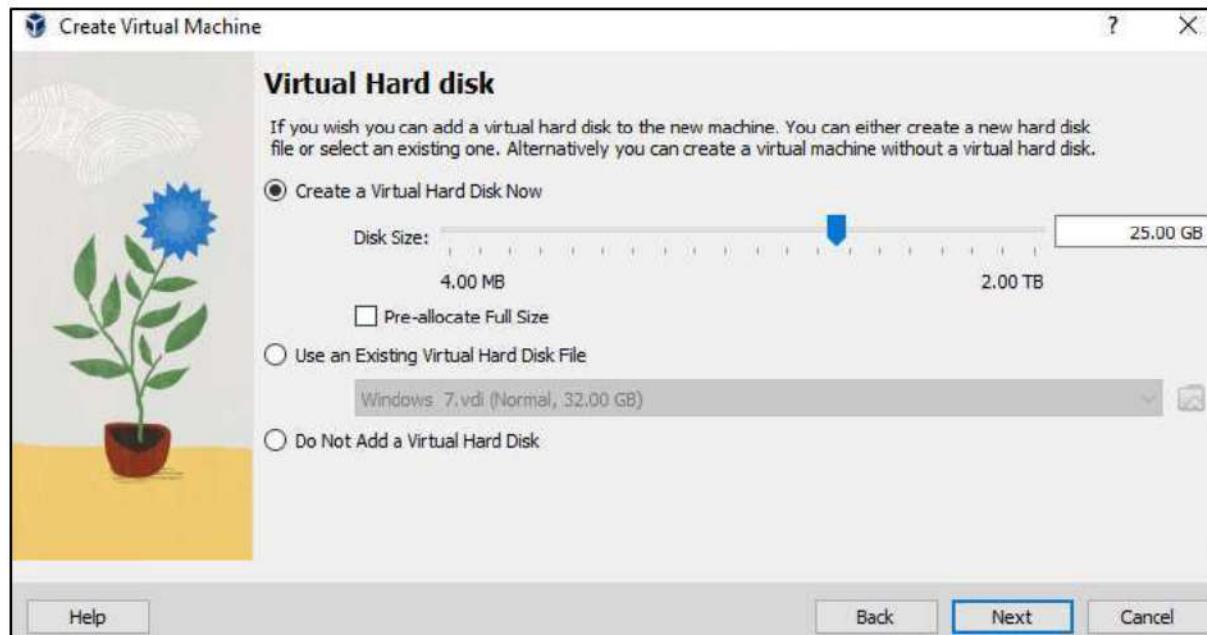
- KVM (for Kernel-based Virtual Machine) is a full virtualization solution for Linux on x86 hardware containing virtualization extensions (Intel VT or AMD-V). It consists of a loadable kernel module, kvm that provides the core virtualization infrastructure and a processor specific module, kvm-intel or kvm-amd.
- Using KVM, one can run multiple virtual machines running unmodified Linux or Windows images. Each virtual machine has private virtualized hardware: a network card, disk, graphics adapter, etc.
- KVM is open source software. The kernel component of KVM is included in mainline Linux, as of 2.6.20. The user space component of KVM is included in mainline QEMU, as of 1.3.

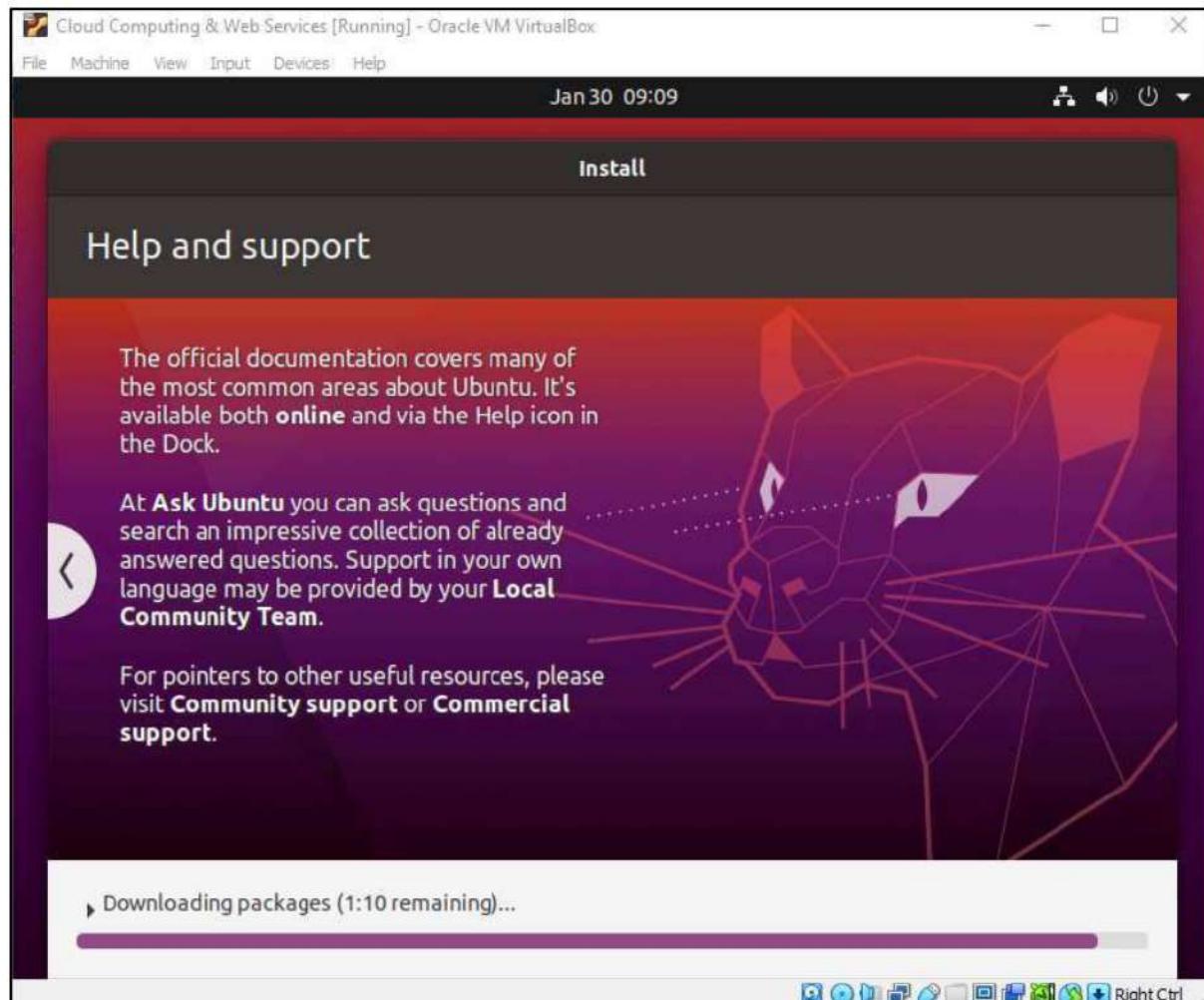
Requirements: Virtual Box, Ubuntu ISO file (latest version)

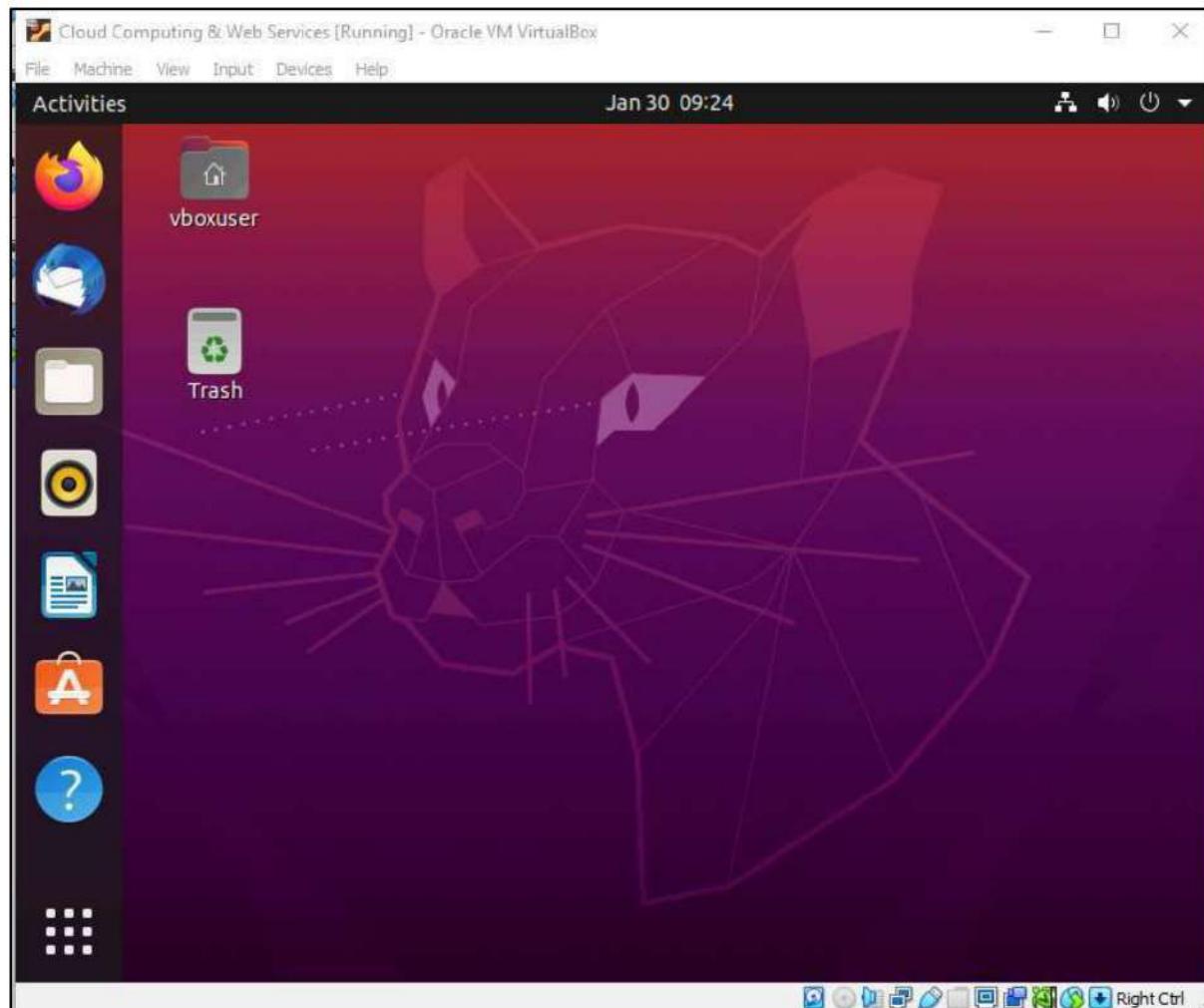
Steps:









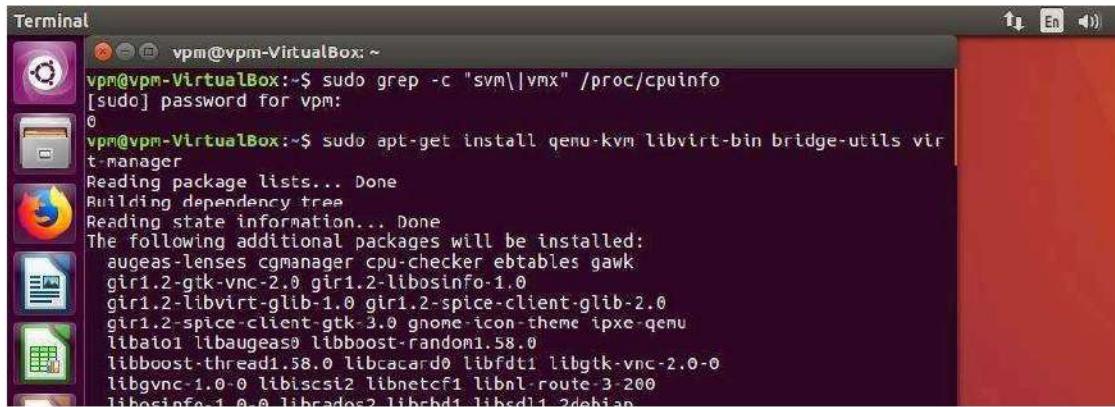


```
vpm@vpm-VirtualBox:~$ sudo grep -c "svm\|vmx" /proc/cpuinfo
[sudo] password for vpm:
0
```

```
sudo apt-get install qemu-kvm libvirt-bin bridge-utils virt-manager
```

OR

```
sudo apt-get install qemu-kvm libvirt-daemon-system libvirt-clients bridge-utils
```



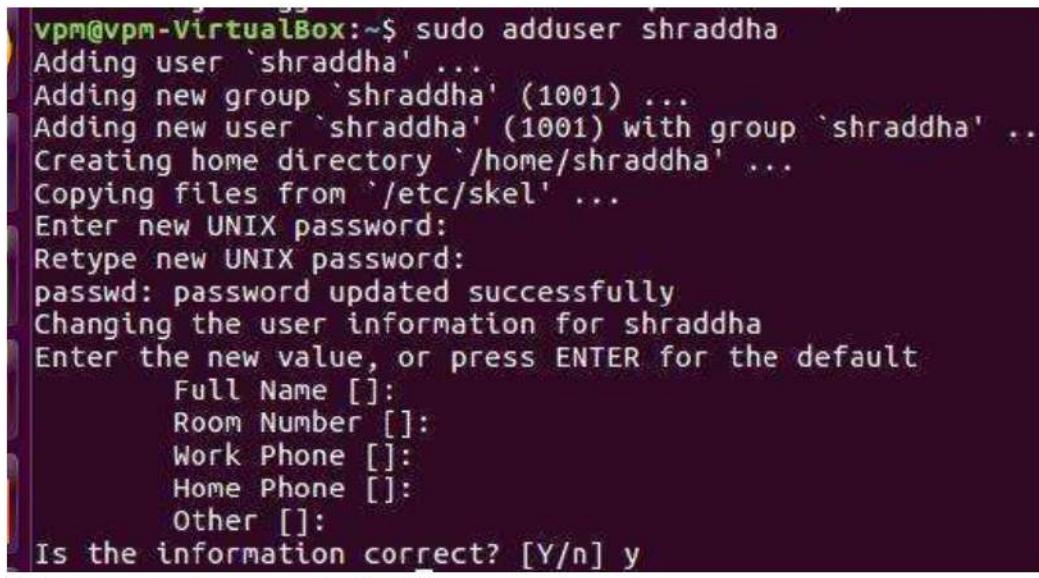
A screenshot of a Linux desktop environment showing a terminal window titled "Terminal". The terminal window has a dark background and contains the following text:

```
vpm@vpm-VirtualBox:~$ sudo grep -c "svm\|vmx" /proc/cpuinfo
[sudo] password for vpm:
0
vpm@vpm-VirtualBox:~$ sudo apt-get install qemu-kvm libvirt-bin bridge-utils virt-manager
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  augeas-lenses cgmanager cpu-checker ebttables gawk
  gir1.2-gtk-vnc-2.0 gir1.2-libosinfo-1.0
  gir1.2-libvirt-glib-1.0 gir1.2-spice-client-glib-2.0
  gir1.2-spice-client-gtk-3.0 gnome-icon-theme ipxe-qemu
  libatoui libaugeas0 libboost-random1.58.0
  libboost-thread1.58.0 libcacard0 libfdt1 libgtk-vnc-2.0-0
  libgvnc-1.0-0 libiscsi2 libnetcf1 libnl-route-3-200
  libosinfo-1.0-0 librbd2 libsd1 libSDL1.2debian
```

If the above commands give error then run

```
sudo apt-get update
```

```
Sudo adduser <yourname>
```



A screenshot of a Linux terminal window showing the process of adding a new user named "shraddha". The terminal window has a dark background and contains the following text:

```
vpm@vpm-VirtualBox:~$ sudo adduser shraddha
Adding user `shraddha' ...
Adding new group `shraddha' (1001) ...
Adding new user `shraddha' (1001) with group `shraddha' ...
Creating home directory `/home/shraddha' ...
Copying files from `/etc/skel' ...
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for shraddha
Enter the new value, or press ENTER for the default
    Full Name []:
    Room Number []:
    Work Phone []:
    Home Phone []:
    Other []:
Is the information correct? [Y/n] y
```

```
Sudo adduser <yourname> libvirtd
```

```
Is the information correct? [y/n] y  
vpm@vpm-VirtualBox:~$ sudo adduser shraddha libvirtd  
Adding user 'shraddha' to group 'libvirtd' ...  
Adding user shraddha to group libvirtd  
Done.  
vpm@vpm-VirtualBox:~$
```

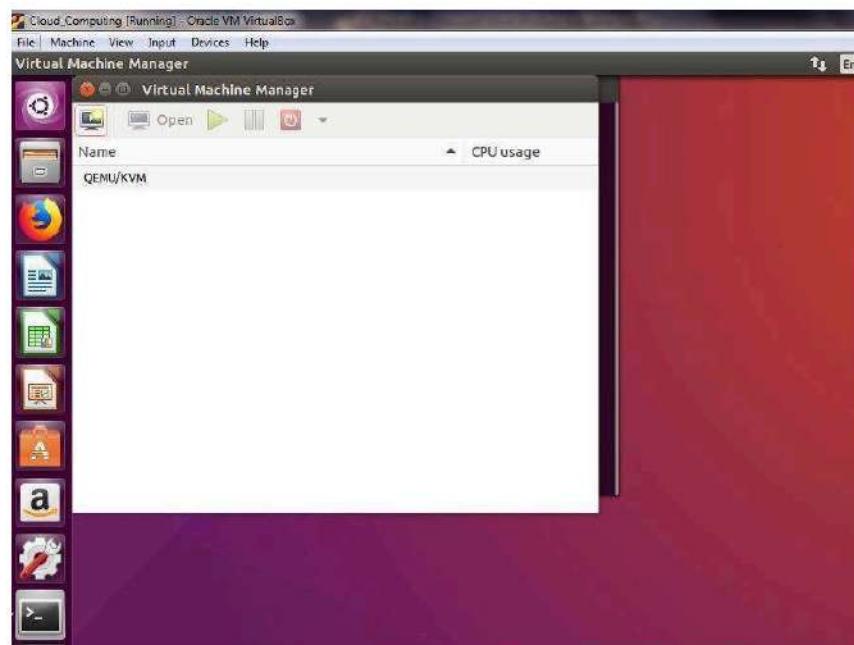
Sudo su <username>

```
virsh -c qemu:///system list
```

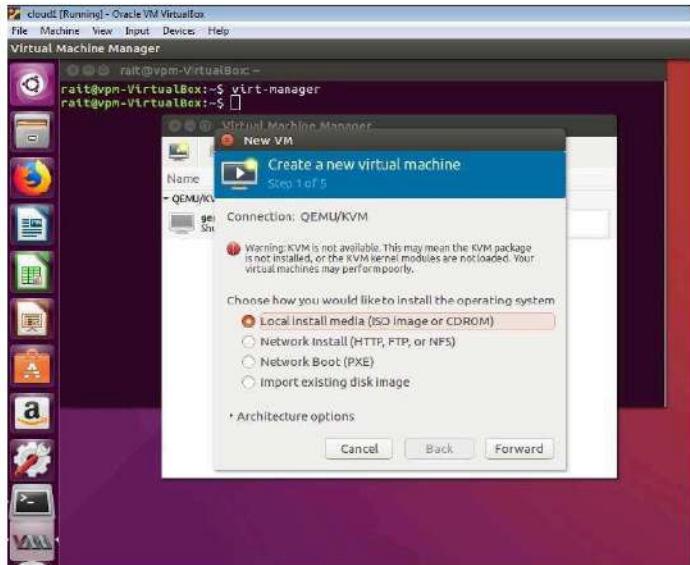
```
shraddha@vpm-VirtualBox:~$ virsh -c qemu:///system list  
  Id   Name           State  
----  
shraddha@vpm-VirtualBox:~$
```

Virt-manager

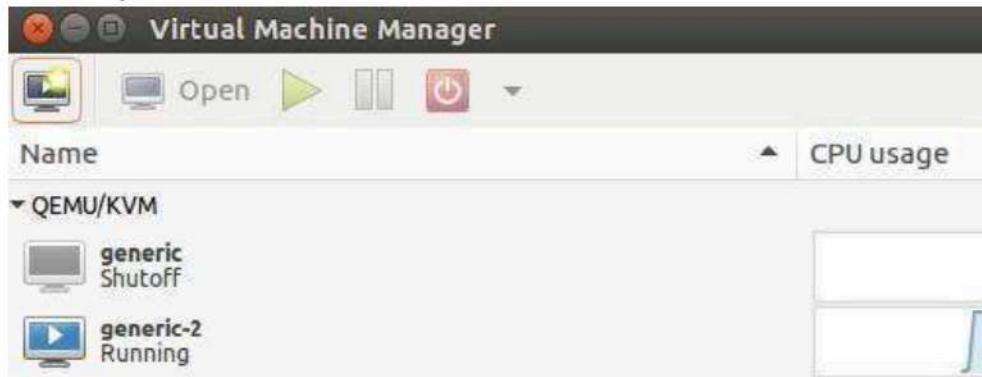
```
shraddha@vpm-VirtualBox:~$ virt-manager  
shraddha@vpm-VirtualBox:~$
```



Click on new virtual machine in qemu and perform the following step



Final output



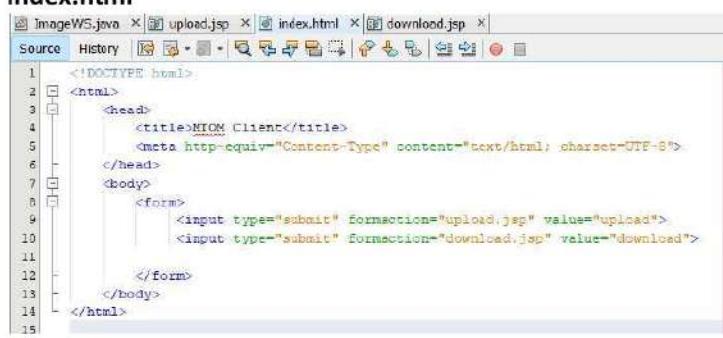
Practical - 6

Aim : Develop application to download image/video from server or upload image/video to server using MTOM techniques

Step 1:

MTOMClient

Index.html



```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>MTOM Client</title>
5     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
6   </head>
7   <body>
8     <form>
9       <input type="submit" formaction="upload.jsp" value="upload">
10      <input type="submit" formaction="download.jsp" value="download">
11    </form>
12  </body>
13</html>
14
15
```

upload.jsp

```
1 <%@page import="java.io.BufferedOutputStream"%>
2 <%@page import="java.io.FileOutputStream"%>
3 <%@page import="java.io.FileInputStream"%>
4 <%@page import="java.io.BufferedInputStream"%>
5 <%@page import="javax.imageio.stream.FileImageInputStream"%>
6 <%@page import="java.io.File"%>
7 <%@page import="javax.xml.ws.soap.MTOMFeature"%>
8 <%@page contentType="text/html" pageEncoding="UTF-8"%>
9 <%@page %>
10 <!DOCTYPE html>
11 <html>
12     <head>
13         <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
14         <title>JSP Page</title>
15     </head>
16     <body>
17
18     <hr/>
19
20
21     <%-- start web service invocation --%><hr/>
22     <%try {
23
24         pkg.ImageWS_Service service = new pkg.ImageWS_Service();
25         pkg.ImageWS port = service.getImageWSService(new MTOMFeature(60000));
26         // TODO initialize WS operation arguments here
27
28         String filePath="c:/mac/ABC.jpg";
29         File file=new File(filePath);
30         FileInputStream fis=new FileInputStream(file);
31         BufferedInputStream bis=new BufferedInputStream(fis);
32         java.lang.String filename = file.getName();
33         byte[]imageBytes=new byte[(int)file.length()];
34         bis.read(imageBytes);
35         port.upload(filename, imageBytes);
36         bis.close();
37         out.println("File uploaded :"+filePath);
38     } catch (Exception ex) {
39         // TODO handle custom exceptions here
40     }
41     <%>
42     <%-- end web service invocation --%><hr/>
43
44     </body>
45 </html>
```

```
<%@page import="java.io.BufferedOutputStream">
<%@page import="java.io.FileOutputStream">
<%@page import="java.io.FileInputStream">
<%@page import="java.io.BufferedInputStream">
<%@page import="javax.imageio.stream.FileImageInputStream">
<%@page import="java.io.File">
<%@page import="javax.xml.ws.soap.MTOMFeature">
<%@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>
<%-- start web service invocation --%><hr/>
<%
try {
    pkg.ImageWS_Service service = new pkg.ImageWS_Service();
    pkg.ImageWS port = service.getImageWSService(new MTOMFeature(60000));
```

```
// TODO initialize WS operation arguments here
String filePath="c:/msc/ABC.jpg";
File file=new File(filePath);
FileInputStream fis=new FileInputStream(file);
BufferedInputStream bis=new BufferedInputStream(fis);
java.lang.String filename = file.getName();
byte[]imageBytes=new byte[(int)file.length()];
bis.read(imageBytes);
port.upload(filename, imageBytes);
bis.close();
out.println("File uploaded :" + filePath);
} catch (Exception ex) {
    // TODO handle custom exceptions here
}
> <%-- end web service invocation --><hr/>
```

download.jsp

```
2  <%@page import="java.io.BufferedOutputStream"%>
3  <%@page import="java.io.FileOutputStream"%>
4  <%@page import="java.io.FileInputStream"%>
5  <%@page import="java.io.BufferedInputStream"%>
6  <%@page import="javax.imageio.stream.FileImageInputStream"%>
7  <%@page import="java.io.File"%>
8  <%@page import="javax.xml.ws.soap.MTOMFeature"%>
9  <%@page contentType="text/html" pageEncoding="UTF-8"%>
10 <!DOCTYPE html>
11 <html>
12   <head>
13     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
14     <title>JSP Page</title>
15   </head>
16   <body>
17
18   <hr/>
19
20
21
22 <%-- start web service invocation --%><hr/>
23 <%
24 try {
25   pkg.ImageWS_Service service = new pkg.ImageWS_Service();
26   pkg.ImageWS port = service.getImageWSPort();
27   // TODO initialize WS operation arguments here
28   java.lang.String filename = "ABC.jpg";
29   String filePath="c:/msc/download/" +filename;
30   // TODO process result here
31   byte[] fileBytes = port.download(filename);
32   FileOutputStream fos=new FileOutputStream(filePath);
33   BufferedOutputStream bos=new BufferedOutputStream(fos);
34   bos.write(fileBytes);
35   bos.close();
36   out.println("File downloaded"+filePath);
37 } catch (Exception ex) {
38   // TODO handle custom exceptions here
39 }
40 <%-- end web service invocation --%><hr/>
41
42   </body>
43 </html>
44
```

```
<%@page import="java.io.BufferedOutputStream">
<%@page import="java.io.FileOutputStream">
<%@page import="java.io.FileInputStream">
<%@page import="java.io.BufferedInputStream">
<%@page import="javax.imageio.stream.FileImageInputStream">
<%@page import="java.io.File">
<%@page import="javax.xml.ws.soap.MTOMFeature">
<%@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>

<%-- start web service invocation --><hr/>
<%
try {
    pkg.ImageWS_Service service = new pkg.ImageWS_Service();
    pkg.ImageWS port = service.getImageWSService();
    // TODO initialize WS operation arguments here
    java.lang.String filename = "ABC.jpg";
    String filePath="c:/msc/download/"+filename;
    // TODO process result here
    byte[] fileBytes = port.download(filename); FileOutputStream
    fos=new FileOutputStream(filePath); BufferedOutputStream
    bos=new BufferedOutputStream(fos); bos.write(fileBytes);
    bos.close();
    out.println("File downloaded"+filePath);
} catch (Exception ex) {
    // TODO handle custom exceptions here
}
> <%-- end web service invocation --><hr/>
</body>
</html>

```

Step 2 :

MTOMServer

ImageWS.java

```

package mypkg;
import java.io.*;
import javax.jws.Oneway;
import javax.jws.WebService;
import javax.jws.WebMethod;
import javax.jws.WebParam;
import javax.xml.ws.soap.MTOM;

@MTOM(enabled=true,threshold=60000)
@WebService(serviceName = "ImageWS")
public class ImageWS {
    @WebMethod(operationName = "upload")
    @Oneway
    public void upload(@WebParam(name = "Filename") String Filename, @WebParam(name =

```

```

"ImageBytes") byte[] ImageBytes)
{
    String filePath="C:/MSC/upload/"+Filename;
try
{
    FileOutputStream
    fos=new
    FileOutputStream(filePath);
    BufferedOutputStream
    bos=new
    BufferedOutputStream(fos);
    bos.write(ImageBytes);
    bos.close();
    System.out.println("Recieved file :" + filePath);
}
catch(Exception ex)
{
    System.out.println(ex);
}

}

@WebMethod(operationName = "download")
public byte[] download(@WebParam(name = "Filename") String Filename) {
String filePath="C:/MSC/upload/"+Filename;
System.out.println("Sending file :" + filePath);
try
{
    File file=new File(filePath);
    FileInputStream fis=new FileInputStream(file);
    BufferedInputStream bis=new BufferedInputStream(fis);
    byte[] fileBytes=new byte[(int)file.length()];
    bis.read(fileBytes);
    bis.close();
    return fileBytes;
}
catch(Exception ex)
{
    System.out.println(ex);
}
return null;
}
}

```

Output:

localhost:8080/MTOMClient/

upload download

Practical 7

Aim: Implement FOSS-Cloud Functionality VSI (Virtual Server Infrastructure) Infrastructure as a Service (IaaS), Creating Virtual Machine or Storage

FOSS Cloud Installation:

Steps:

1. Choose your keymap.
2. Confirm that you want to start: yes
3. Choose Demo-System: 1
4. Choose a Block-Device: sda
5. Confirm that you want to continue: yes
6. Confirm that you want to continue: yes
7. Choose the network interface: eth0
8. Choose if you want to use automatic network configuration: Yes
9. Reboot your system: yes
10. Login as root and run "fc-node-configuration -n demo-system --password admin"

Login as admin

Password : admin

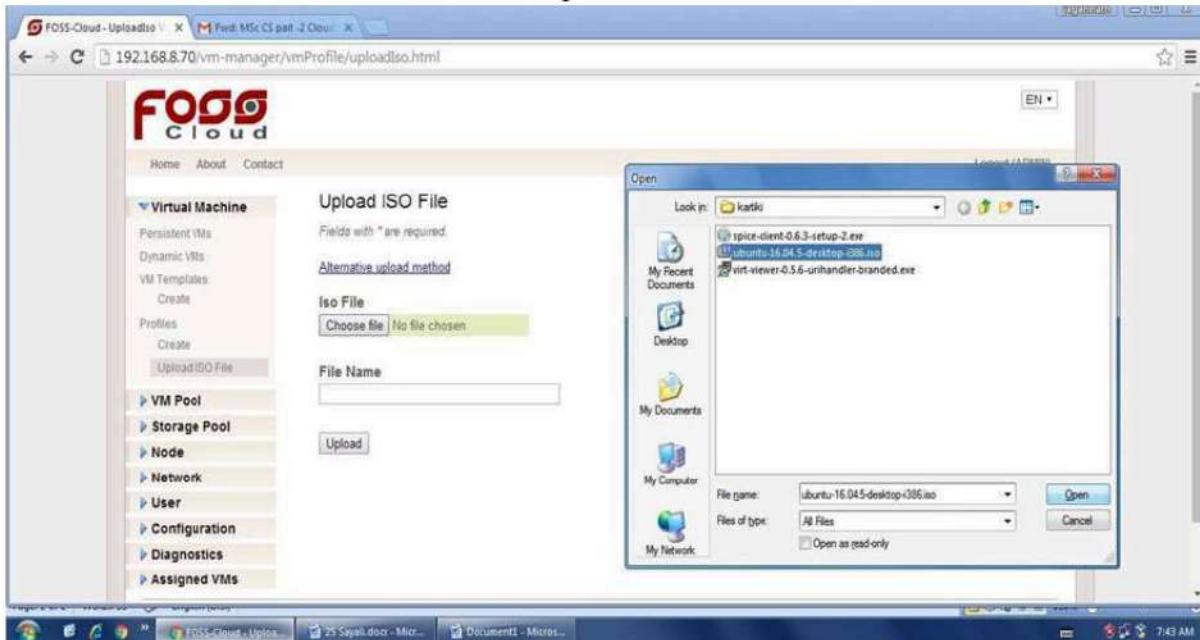


Home page of FOSS Cloud at Client



Uploading ISO files to foss cloud

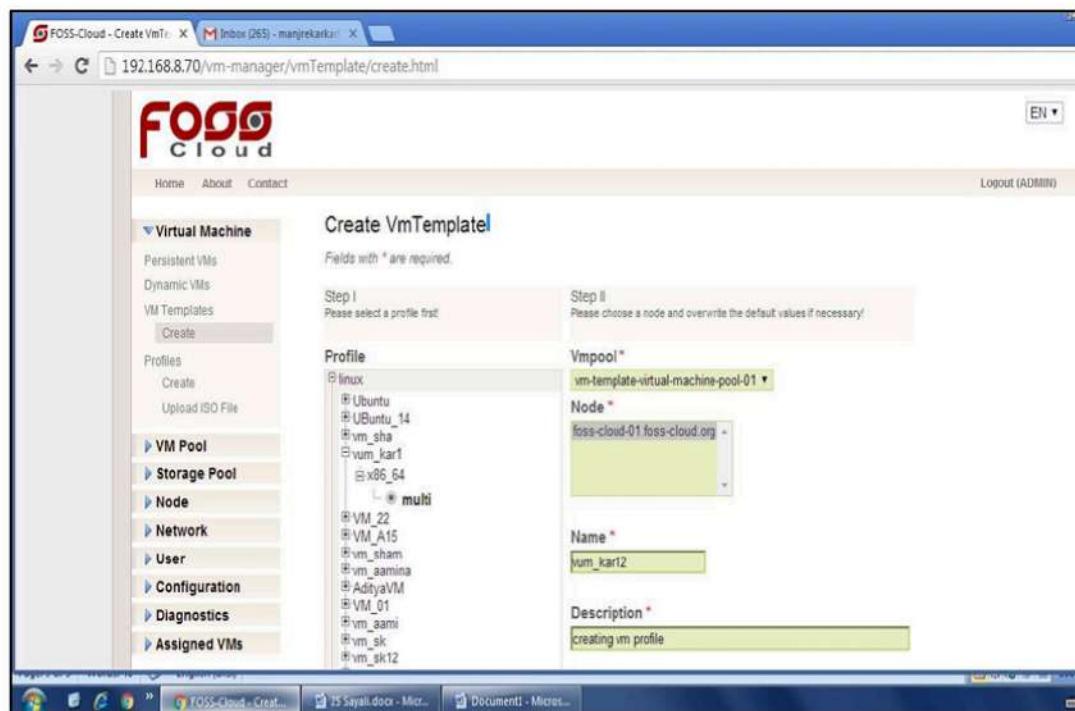
Go to virtual machine>Profile>Upload ISO file >file name.



Creating a Profile

The profile creates the relationship between the ISO file and the FOSS-Cloud.

1. Open Virtual Machines
2. Choose vm-Profiles
3. Choose the right Base Profile
4. Choose the right architecture (Windows only x86_64)
5. Choose the language (it is a information - not keyboard relevant)
6. Choose the ISO file
7. Fill out name and description
8. Choose the amount of memory and volume capacity
9. Choose amount of CPU
10. Choose clock offset (normally Windows is "localtime" and Linux is "utc")



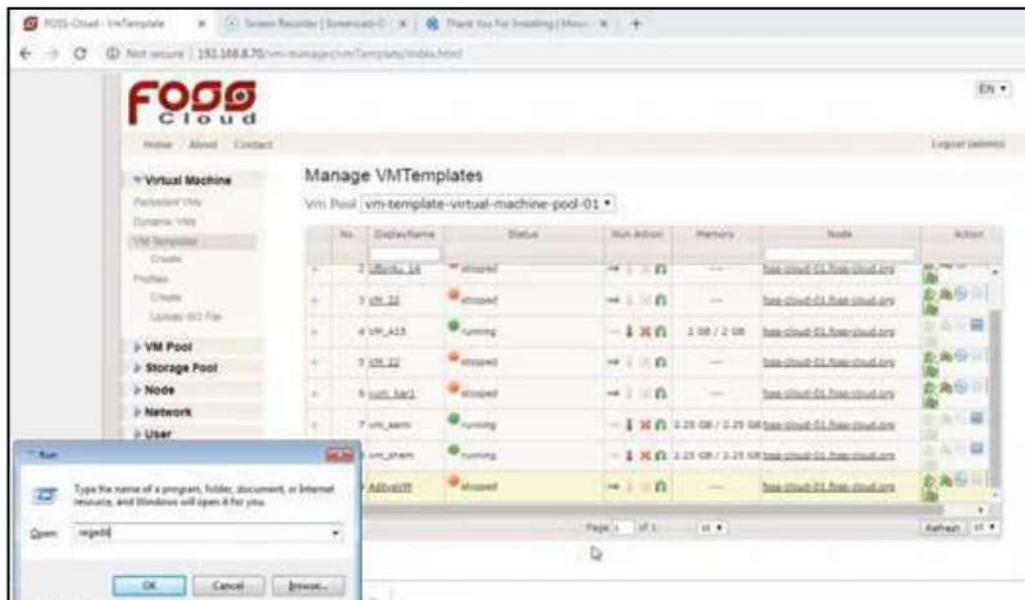
Creating Template

1. Choose the profile you have prepared before
 2. Add the vm-pool and one or more nodes, where you will run this vm (when the chosen vm-pool has only one node assigned, you don't have a choice)
 3. You can change all the other informations you have entered before
- Click on "create" and the template is ready for installing the guest operating system.**

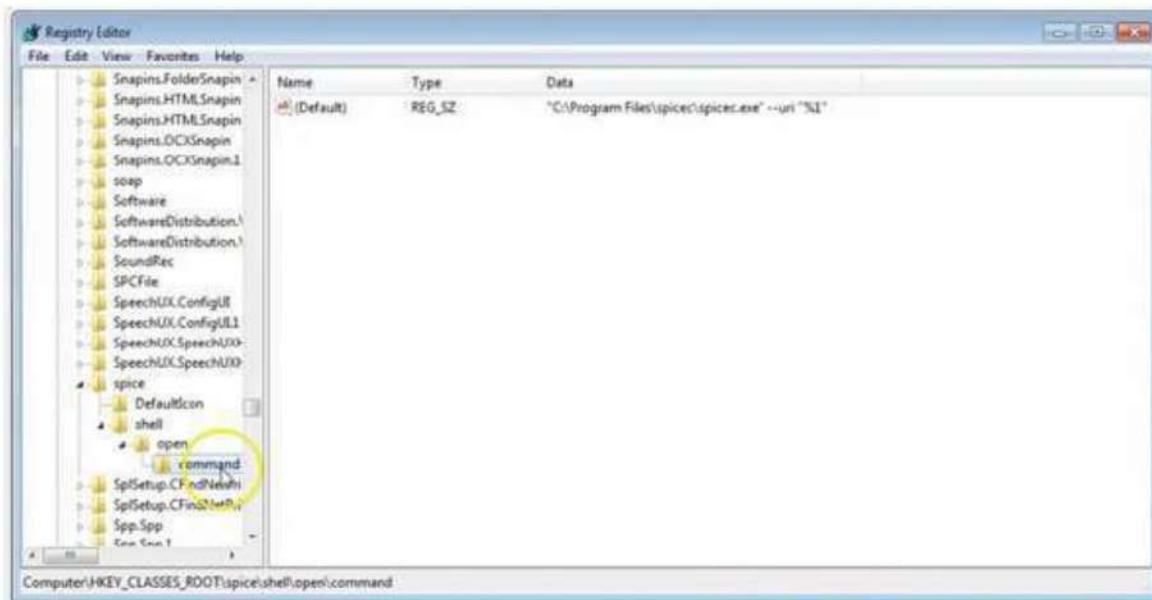
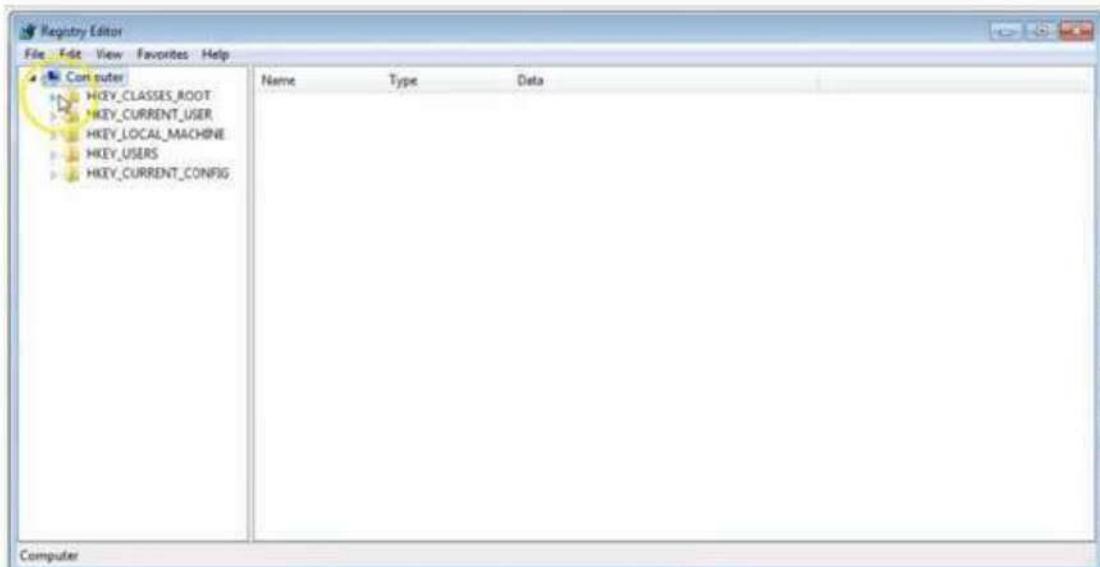
In stall virt Viewer and spice



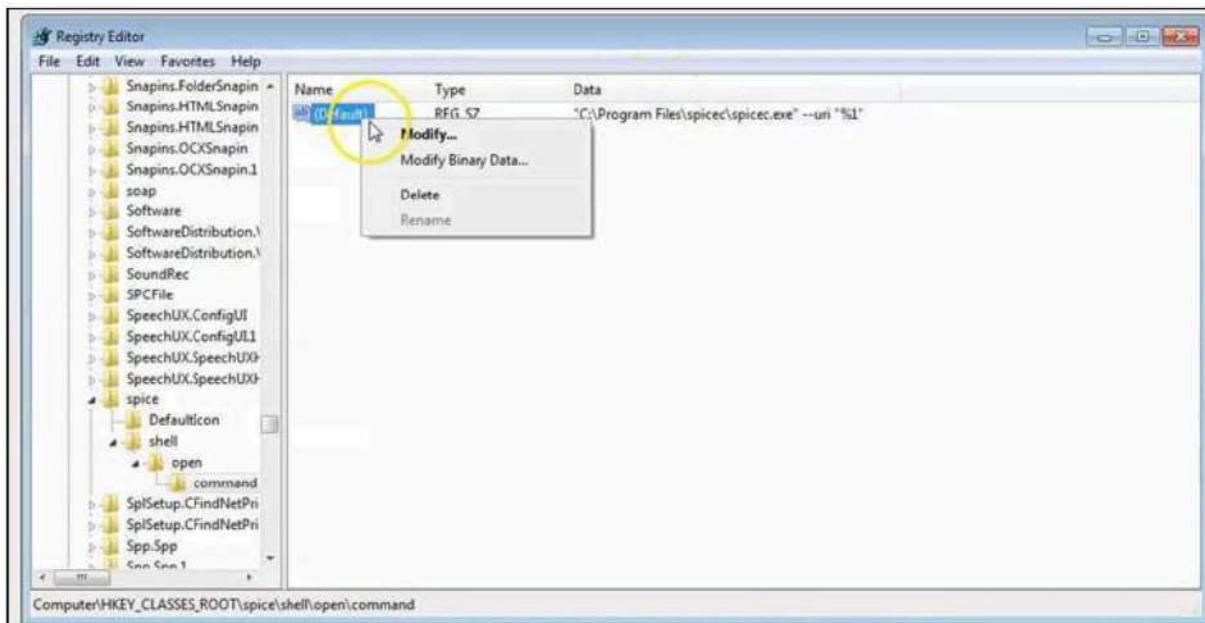
Go to regedit



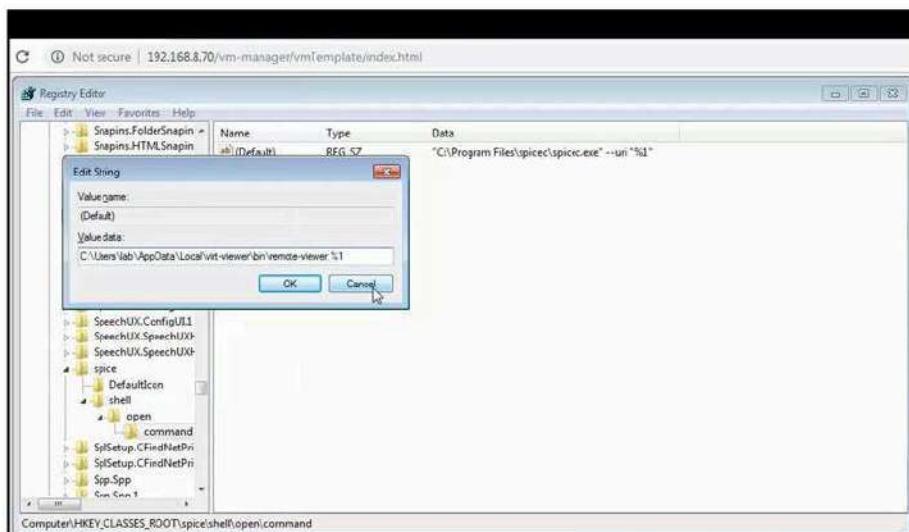
Go to Computer > HKEY_CLASSES_ROOT > SPICE SHELL > COMMAND



Go to Default and click Modify



Paste path of virt viewer à `virt_viewer_path %1` (Actual path of Virt Viewer)



Run Your VM and View using VM Template

Foss Cloud

Home About Contact Logout (admin)

Virtual Machine

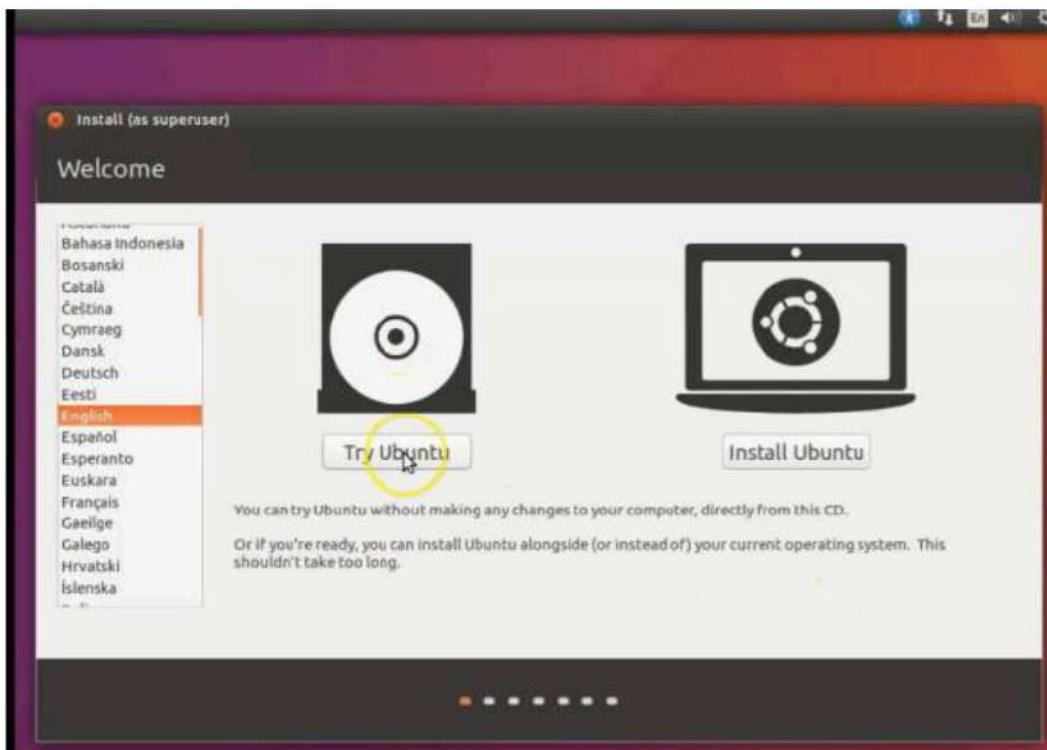
- Persistent VMs
- Dynamic VMs
- VM Templates**
 - Create
 - Profiles
 - Create
 - Upload ISO File
- VM Pool**
- Storage Pool**
- Node**
- Network**
- User**
- Configuration**
- Diagnostics**
- Assigned VMs**

Manage VMTemplates

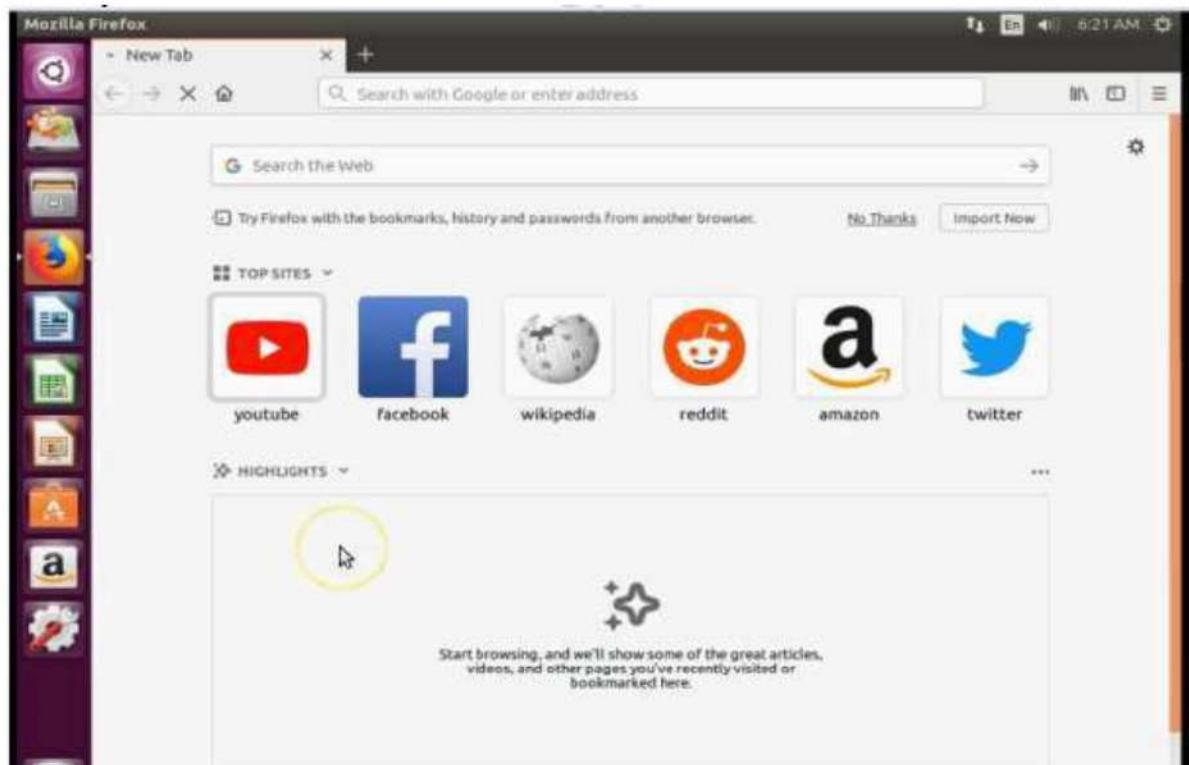
Vm Pool **vm-template-virtual-machine-pool-01**

No.	DisplayName	Status	Run Action	Memory	Node	Action
+	2 vUbuntu_14	stopped	→ ↳ ✎	---	foss-cloud-01.foss-cloud.org	
+	3 VM_22	stopped	→ ↳ ✎	---	foss-cloud-01.foss-cloud.org	
+	4 VM_A15	running	→ ↳ ✎	2 GB / 2 GB	foss-cloud-01.foss-cloud.org	
+	5 VM_22	stopped	→ ↳ ✎	---	foss-cloud-01.foss-cloud.org	
+	6 yum_kar1	running	→ ↳ ✎	2 GB / 2 GB	foss-cloud-01.foss-cloud.org	
+	7 vm_aam1	running	→ ↳ ✎	2.25 GB / 2.25 GB	foss-cloud-01.foss-cloud.org	
+	8 vm_sham	running	→ ↳ ✎	2.25 GB / 2.25 GB	foss-cloud-01.foss-cloud.org	
+	9 AdityaVM	running	→ ↳ ✎	2.5 GB / 2.5 GB	foss-cloud-01.foss-cloud.org	

Page 1 of 1 Refresh 10 ▾



Open Browser in Ubuntu.

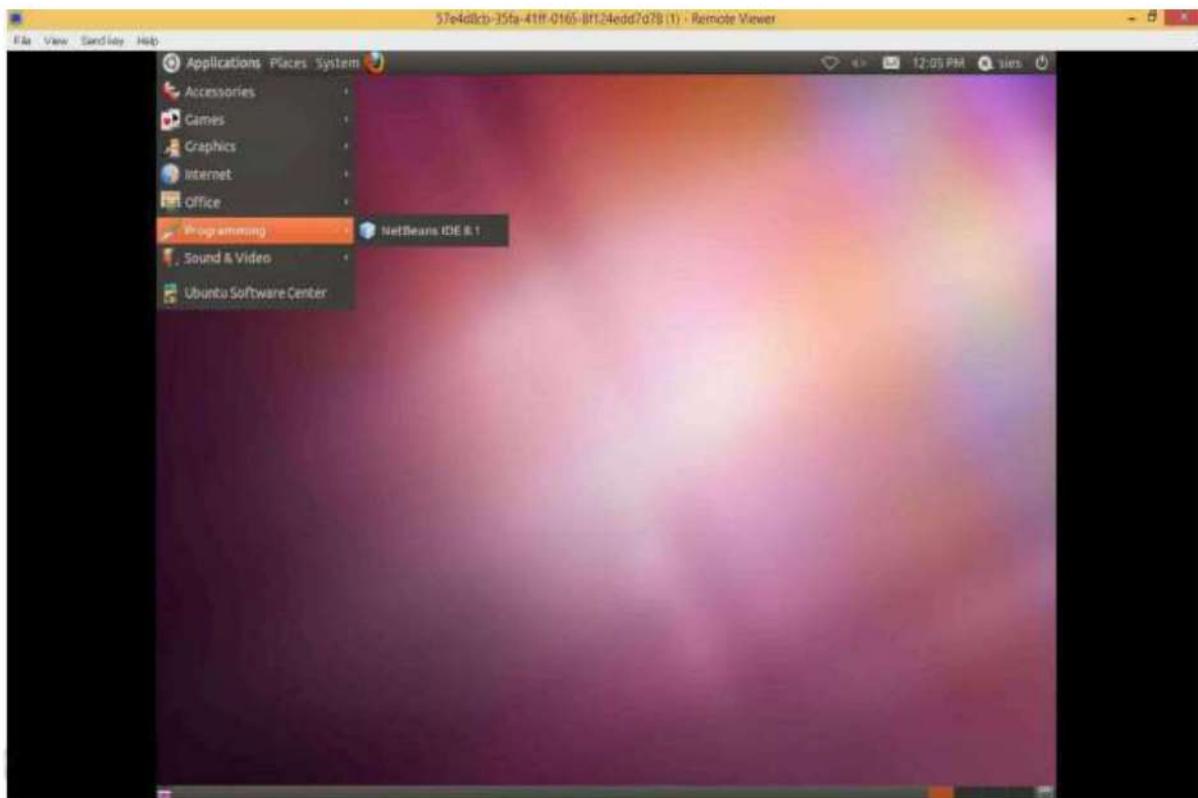


Practical 8

Aim: Implement FOSS-Cloud Functionality - VSI Platform as a Service (PaaS)

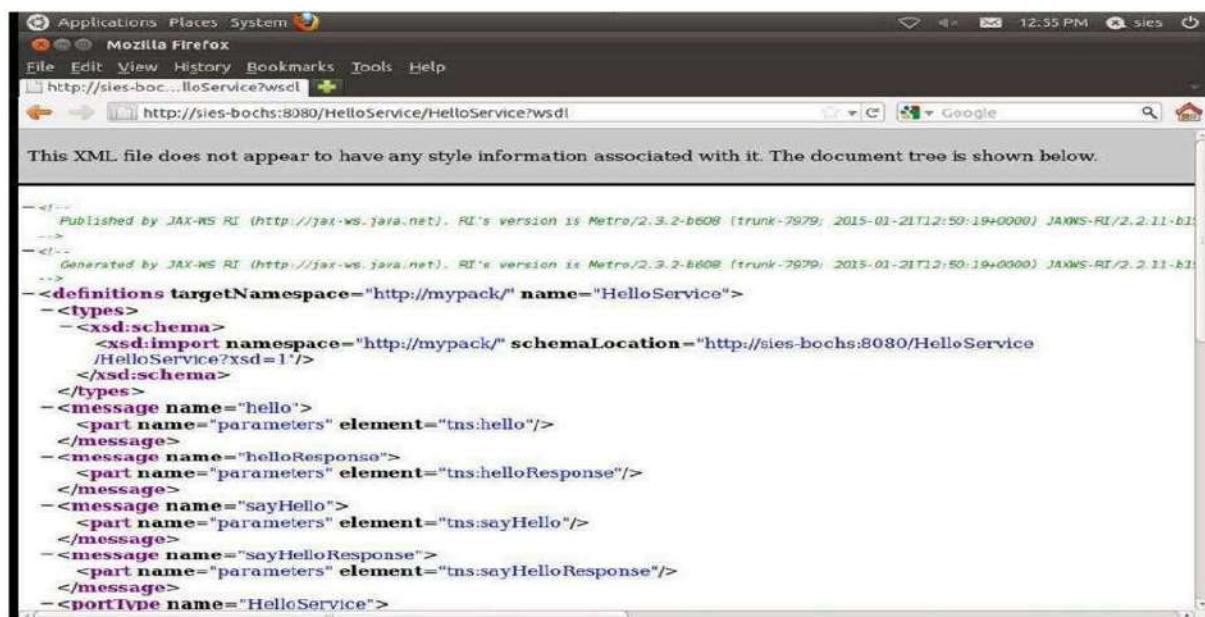
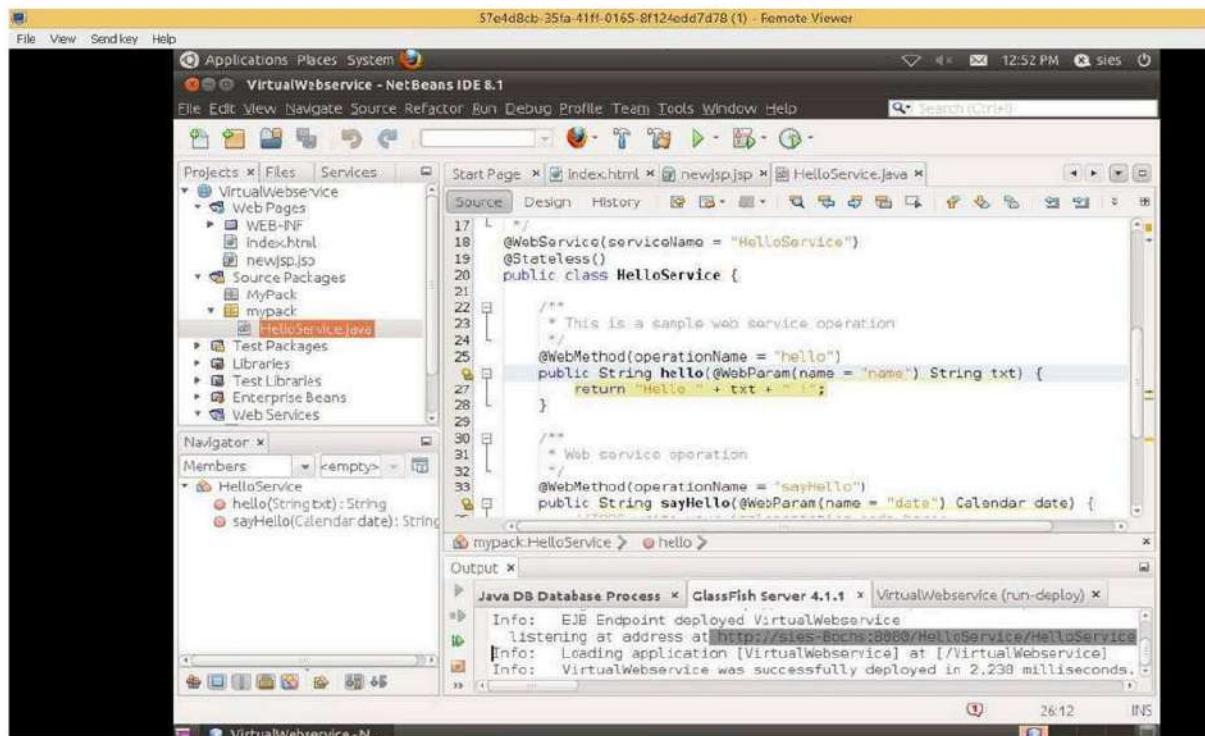
Software development Kits can be made available in the Virtual Machines That can be implemented as Platform as a Service.

Installation of Netbeans, Eclipse, Visual Studio and DBMS can be done in the appropriate Virtual Machines.



Aim: Implement FOSS-Cloud Functionality - VSI Software as a Service (SaaS)

Applications created and deployed in the virtual machines can be accessed by the outside world. This can be implemented as Software as a Service



Practical - 10

Aim: How to Install OpenStack on Ubuntu with DevStack

Minimum Requirements

Before we begin, ensure you have the following minimum prerequisites

1. A fresh Ubuntu 18.04 installation
2. User with sudo privileges
3. 4 GB RAM
4. 2 vCPUs
5. Hard disk capacity of 10 GB
6. Internet connection

Step 1: Update and Upgrade the System

To start off, log into your Ubuntu 18.04 system using SSH protocol and update & upgrade system repositories using the following command.

apt update -y && apt upgrade -y

Sample Output

```
root@ubuntu:/# apt update -y && apt upgrade -y
Hit:1 http://us-central1.gce.archive.ubuntu.com/ubuntu bionic InRelease
Get:2 http://us-central1.gce.archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]
Get:3 http://us-central1.gce.archive.ubuntu.com/ubuntu bionic-backports InRelease [74.6 kB]
Get:4 http://us-central1.gce.archive.ubuntu.com/ubuntu bionic/universe amd64 Packages [8570 kB]
Get:5 http://archive.canonical.com/ubuntu bionic InRelease [10.2 kB]
Get:6 http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 kB]
Get:7 http://us-central1.gce.archive.ubuntu.com/ubuntu bionic/universe Translation-en [4941 kB]
Get:8 http://us-central1.gce.archive.ubuntu.com/ubuntu bionic/multiverse amd64 Packages [151 kB]
Get:9 http://us-central1.gce.archive.ubuntu.com/ubuntu bionic/multiverse Translation-en [108 kB]
Get:10 http://us-central1.gce.archive.ubuntu.com/ubuntu bionic-updates/main amd64 Packages [627 kB]
```

Next reboot the system using the command.

sudo reboot

OR

init 6

Step 2: Create Stack user and assign sudo privilege

Best practice demands that devstack should be run as a regular user with sudo privileges. With that in mind, we are going to add a new user called “stack” and assign sudo privileges. To create stack user execute

sudo adduser -s /bin/bash -d /opt/stack -m stack

Next, run the command below to assign sudo privileges to the user

echo "stack ALL=(ALL) NOPASSWD: ALL" | sudo tee /etc/sudoers.d/stack

Sample Output

```
root@ubuntu:/# sudo useradd -s /bin/bash -d /opt/stack -m stack
root@ubuntu:/#
root@ubuntu:/# echo "stack ALL=(ALL) NOPASSWD: ALL" | sudo tee /etc/sudoers.d/stack
stack ALL=(ALL) NOPASSWD: ALL
root@ubuntu:/#
```

Step 3: Install git and download DevStack

Once you have successfully created the user ‘stack’ and assigned sudo privileges, switch to the user using the command.

su - stack

In most Ubuntu 18.04 systems, git comes already installed. If by any chance git is missing, install it by running the following command.

sudo apt install git -y

Sample output

```
root@ubuntu:~# su - stack
stack@ubuntu:~$ 
stack@ubuntu:~$ sudo apt install git -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
git is already the newest version (1:2.17.1-1ubuntu0.4).
The following packages were automatically installed and are no longer required:
  grub-pc-bin libnuma1
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
```

Using git, clone devstack’s git repository as shown.

git clone https://git.openstack.org/openstack-dev/devstack

Sample output

```
stack@ubuntu:~$ git clone https://git.openstack.org/openstack-dev/devstack
Cloning into 'devstack'...
warning: redirecting to https://opendev.org/openstack/devstack/
remote: Enumerating objects: 43615, done.
remote: Counting objects: 100% (43615/43615), done.
remote: Compressing objects: 100% (12575/12575), done.
remote: Total 43615 (delta 31152), reused 42370 (delta 30360)
Receiving objects: 100% (43615/43615), 8.27 MiB | 24.61 MiB/s, done.
Resolving deltas: 100% (31152/31152), done.
stack@ubuntu:~$ 
stack@ubuntu:~$ ls
devstack
stack@ubuntu:~$
```

Step 4: Create devstack configuration file

In this step, navigate to the devstack directory.

cd devstack

Then create a local.conf configuration file.

vim local.conf

Paste the following content

[[local|localrc]]

```
# Password for KeyStone, Database, RabbitMQ and Service
ADMIN_PASSWORD=StrongAdminSecret
DATABASE_PASSWORD=$ADMIN_PASSWORD
RABBIT_PASSWORD=$ADMIN_PASSWORD
SERVICE_PASSWORD=$ADMIN_PASSWORD

# Host IP - get your Server/VM IP address from ip addr command
HOST_IP=10.208.0.10
```

Save and exit the text editor. NOTE:

1. The **ADMIN_PASSWORD** is the password that you will use to log in to the OpenStack login page. The default username is **admin**.
2. The **HOST_IP** is your system's IP address that is obtained by running **ifconfig** or **ip addr** commands.

Step 5: Install OpenStack with Devstack

To commence the installation of OpenStack on Ubuntu 18.04, run the script below contained in devstack directory.

./stack.sh

The following features will be installed:

- Horizon – OpenStack Dashboard
- Nova – Compute Service
- Glance – Image Service
- Neutron – Network Service
- Keystone – Identity Service
- Cinder – Block Storage Service
- Placement – Placement API

The deployment takes about 10 to 15 minutes depending on the speed of your system and internet connection. In our case, it took roughly 12 minutes. At the very end, you should see output similar to what we have below.

```

        print a[2]
    }
' /opt/stack/devstack/local.conf
+./stack.sh:main:1489                         set +o xtrace

=====
DevStack Component Timing
(times are in seconds)
=====
run_process          53
test_with_retry      2
apt-get-update       1
osc                  177
wait_for_service     21
dbsync               56
pip_install          149
apt-get              7

Unaccounted time    418
=====
Total runtime        884

This is your host IP address: 10.128.0.8
This is your host IPv6 address: ::1
Horizon is now available at http://10.128.0.8/dashboard
Keystone is serving at http://10.128.0.8/identity/
The default users are: admin and demo
The password: StrongAdminSecret

WARNING:
Using lib/neutron-legacy is deprecated, and it will be removed in the future

Services are running under systemd unit files.
For more information see:
https://docs.openstack.org/devstack/latest/systemd.html

DevStack Version: train
Change: 16d11d27f375b8c027bbc3a1db1885e90ce6c604 Merge "Option "lock_path" from group "DEFAULT"
OS Version: Ubuntu 18.04 bionic

2019-06-04 12:19:19.207 | stack.sh completed in 884 seconds.

```

This confirms that all went well and that we can proceed to access OpenStack via a web browser.

Step 6: Accessing OpenStack on a web browser

To access OpenStack via a web browser browse your Ubuntu's IP address as shown.

https://server-ip/dashboard This directs you to a login page as shown.



openstack®

Log in

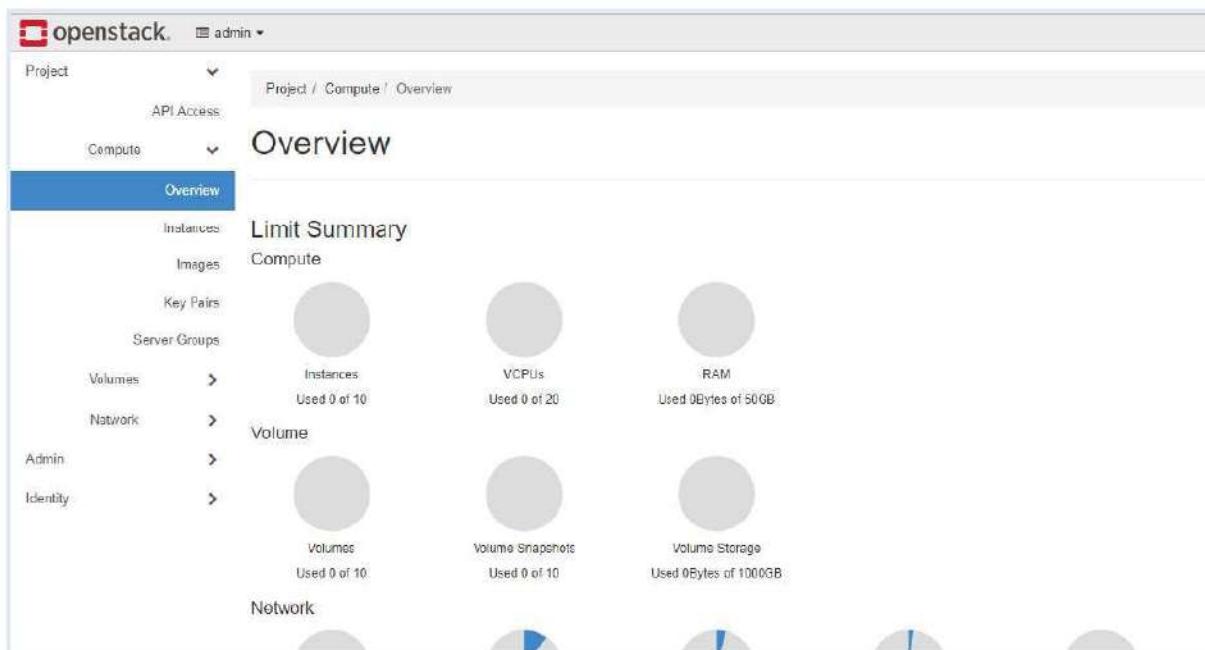
User Name

Password

 eye icon

Sign In

Enter the credentials and hit “Sign In” You should be able to see the Management console dashboard as shown below.



The screenshot shows the OpenStack Management console's Overview dashboard. At the top, there is a navigation bar with the OpenStack logo and the word "admin". Below the navigation bar, there is a sidebar with dropdown menus for "Project", "API Access", "Compute", and "Network". The "Compute" menu is currently selected and expanded, showing sub-options like "Instances", "Images", "Key Pairs", "Server Groups", "Volumes", "Network", "Admin", and "Identity". The main content area is titled "Overview" and contains a "Limit Summary" section for "Compute". It shows four resource metrics: Instances (Used 0 of 10), VCPUs (Used 0 of 20), RAM (Used 0Bytes of 50GB), and Volume Storage (Used 0Bytes of 1000GB). Each metric is represented by a circular progress bar. Below the limit summary, there is a "Network" section with three progress bars for "Volumes", "Volume Snapshots", and "Volume Storage".