

**Cloud
Computing
And
Web Services
T.Y.B.Sc Computer
Science
(Semester-VI)
For Academic Year
(2023-24)**



JNAN VIKAS MANDAL'S
PADMASHREE DR. R.T.DOSHI DEGREE COLLEGE OF COMPUTER SCIENCE
MOHANLAL RAICHAND MEHTA COLLEGE OF COMMERCE
DIWALIMAA DEGREE COLLEGE OF SCIENCE
AMRATLAL RAICHAND MEHTA COLLEGE OF ARTS
JVM'S DEGREE COLLEGE OF INFORMATION TECHNOLOGY
AIROLI, NAVI MUMBAI – 400708

CERTIFICATE

This is to certify that the Mr./Miss. _____ having Roll No _____ of B.Sc.CS Semester VI has completed the practical work in the subject of **Cloud Computing and Web Services** during the Academic year 2023-24 being the partial requirement for the fulfillment of the curriculum of Degree of Bachelor of Science in Computer Science, University of Mumbai.

Place:

Date:

Sign of Subject In-Charge

External Examiner

CS-IT Coordinator

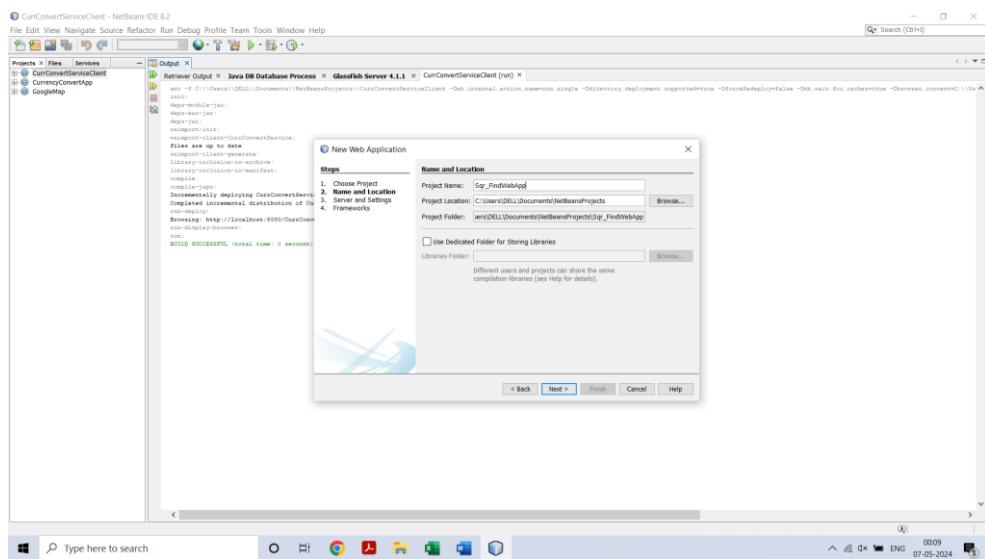
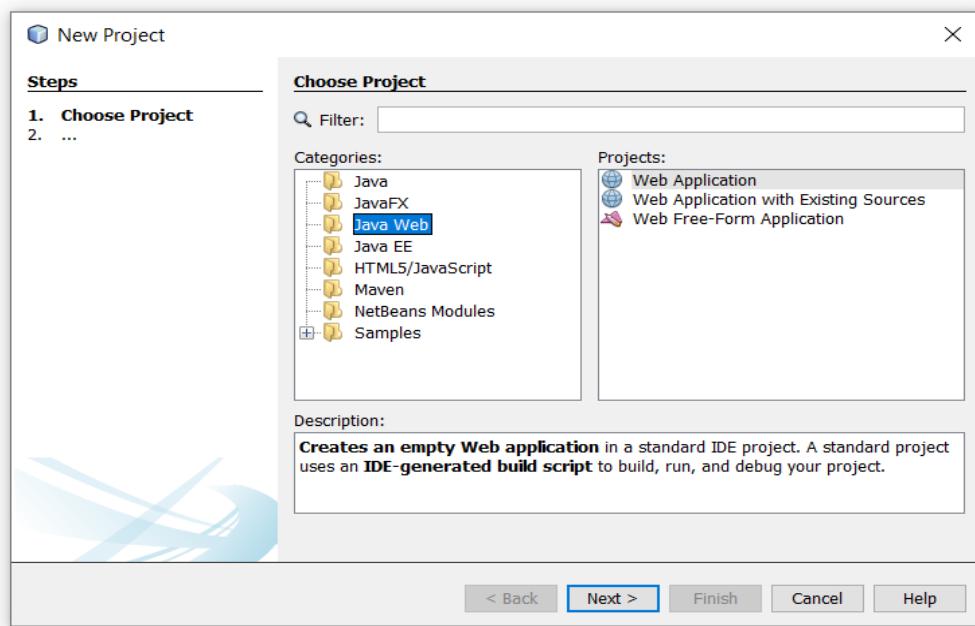
INDEX

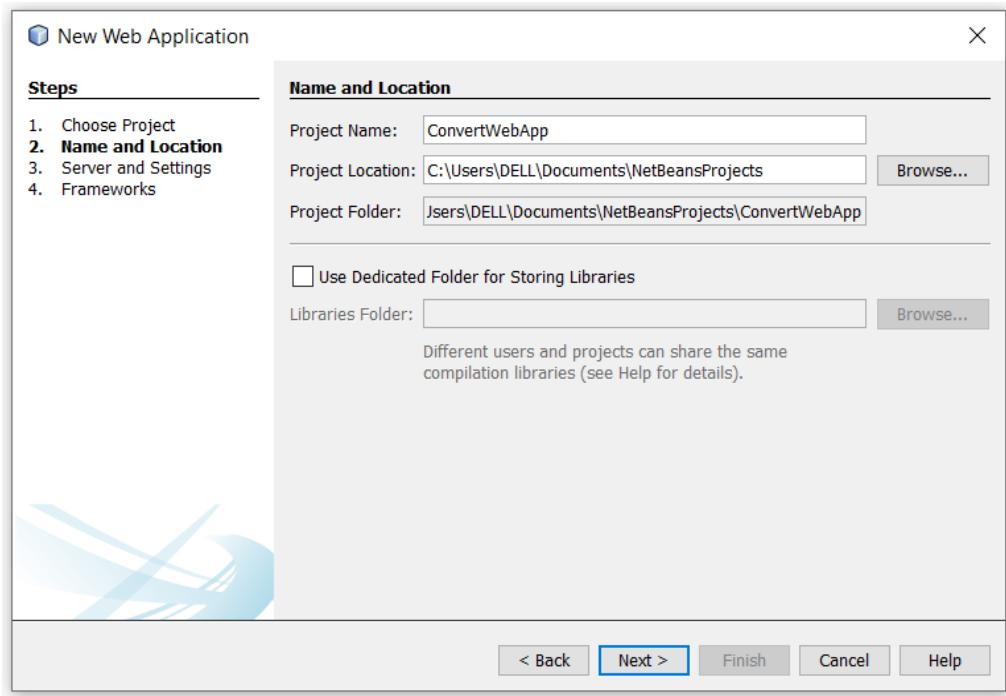
Sr.No.	Practical's	Date	Sign
1	Define a simple services like Converting Rs into Dollar and Call it from different platform like JAVA and .NET		
2	Create a Simple SOAP service to find the Square of a Number		
3	Create a Simple REST Service to demonstrate CRUD operations with “Student” database		
4	Develop application to consume Google’s search / Google’s Map RESTful Web service.		
5	Installation and Configuration of virtualization using KVM.		
6	Develop application to download image/video from server or upload image/video to server using MTOM techniques		
7	Implement FOSS-Cloud Functionality VSI (Virtual Server Infrastructure) Infrastructure as a Service (IaaS), Storage		
8	Implement FOSS-Cloud Functionality - VSI Platform as a Service (PaaS),		
9	Using AWS Flow Framework develop application that includes a simple workflow. Workflow calls an activity to print hello world to the console. It must define the basic usage of AWS Flow Framework, including defining contracts, implementation of activities and workflow coordination logic and worker programs to host them		
10	Implementation of Openstack with user and private network creation.		

Practical-1(A)

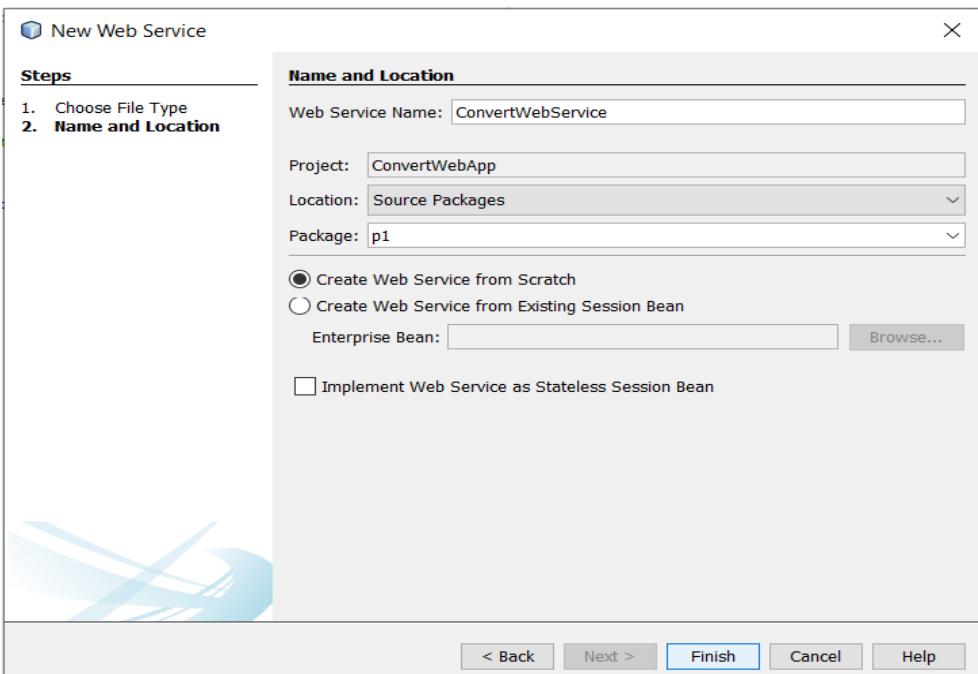
Define a simple services like Converting Rupees into Dollar and Call it from different platform like JAVA.

Step1 : To create Web Service through Net Beans

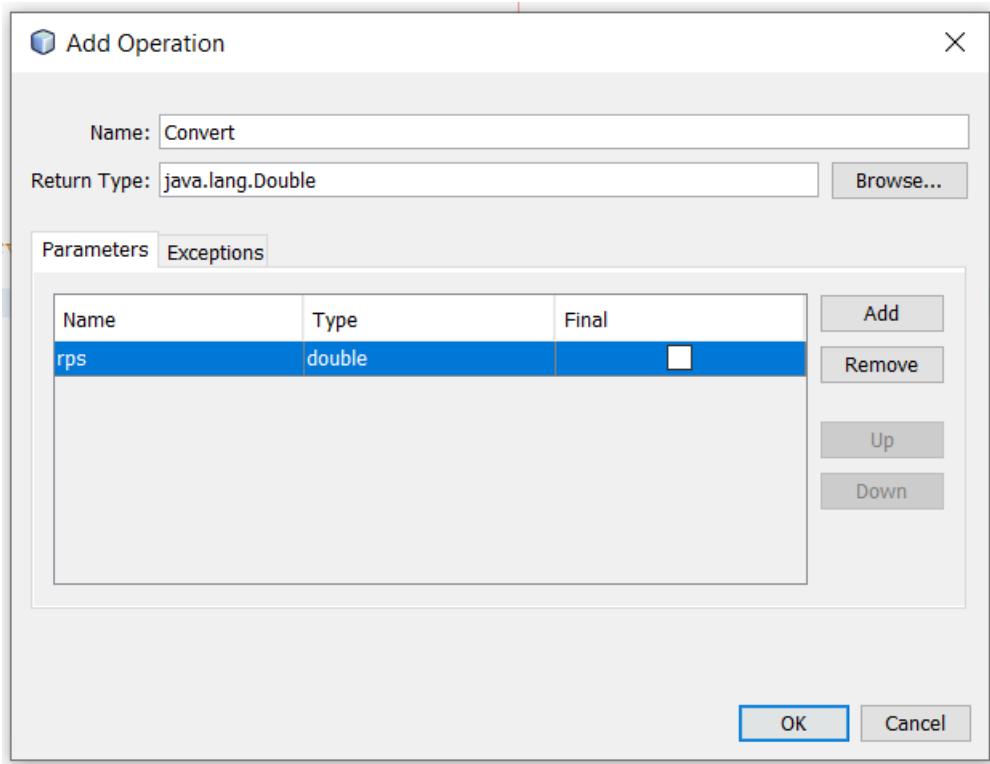




Step 2: To create Web Service “ ConvertWebService”



Step 3 : to Add Operation “Convert” to Web Service



Here Convert is a method having one parameter as rps

Step 4 : After adding code of Operation to web Service

```

ConvertWebApp - NetBeans IDE 8.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Output - ConvertWebApp (clean) index.html ConvertWebService.java
Source Design History
1 package p1;
2
3 import javax.jws.WebService;
4 import javax.jws.WebMethod;
5 import javax.jws.WebParam;
6
7 /**
8 * Author DELL
9 */
10 @WebService(serviceName = "ConvertWebService")
11 public class ConvertWebService {
12
13
14
15
16
17     @WebMethod(operationName = "Convert")
18     public Double Convert(@WebParam(name = "rps") double rps)
19     {
20         double d;
21         d=rps/33.5;
22         return d;
23     }
24
25
26 }
27

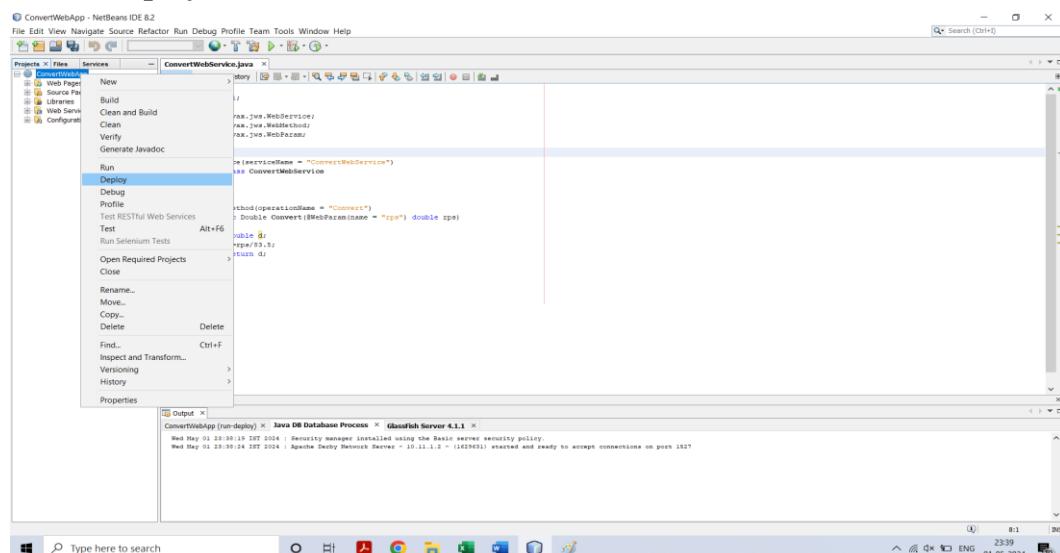
```

Code of Operation :

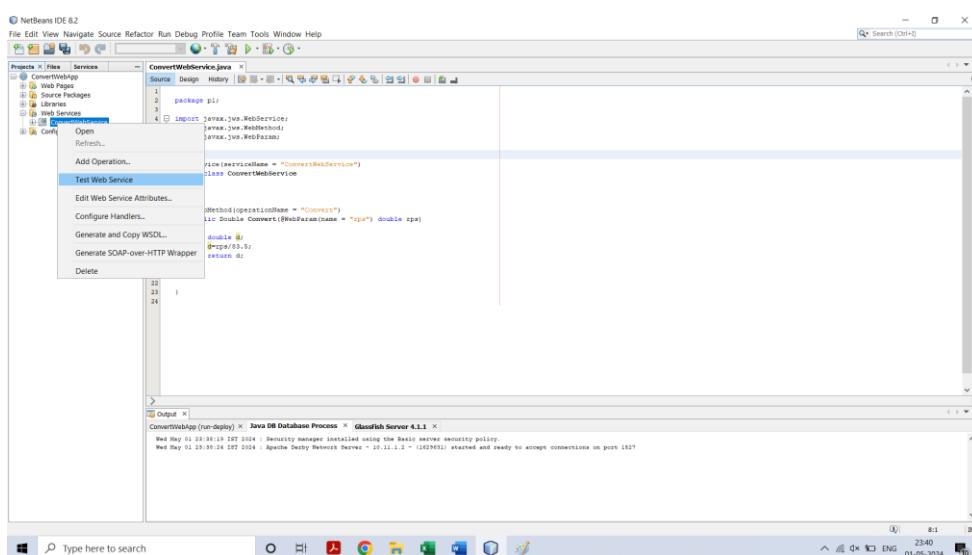
```
@WebMethod(operationName = "Convert")
public Double Convert(@WebParam(name = "rps") double rps)
{
    double d;
    d=rps/83.5;
    return d;
}
```

Step 5 : Now to deploy the web service and Test

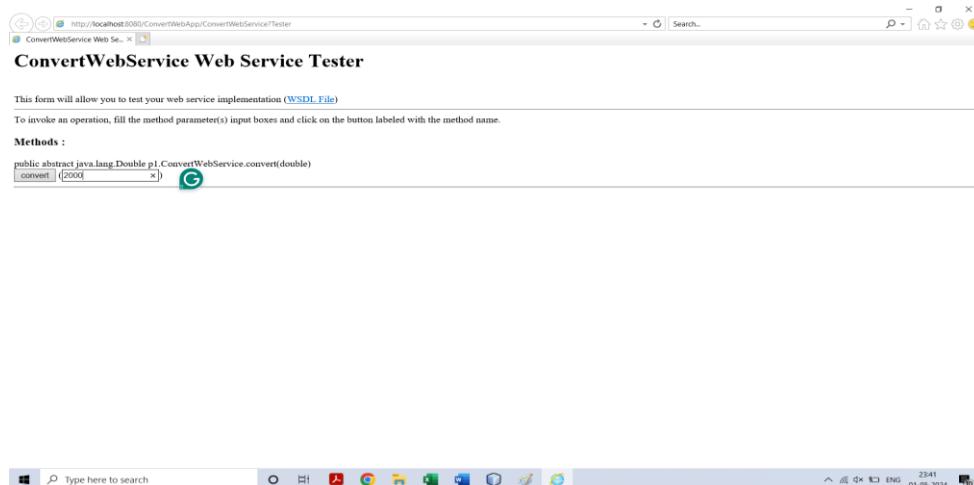
i) Deploy



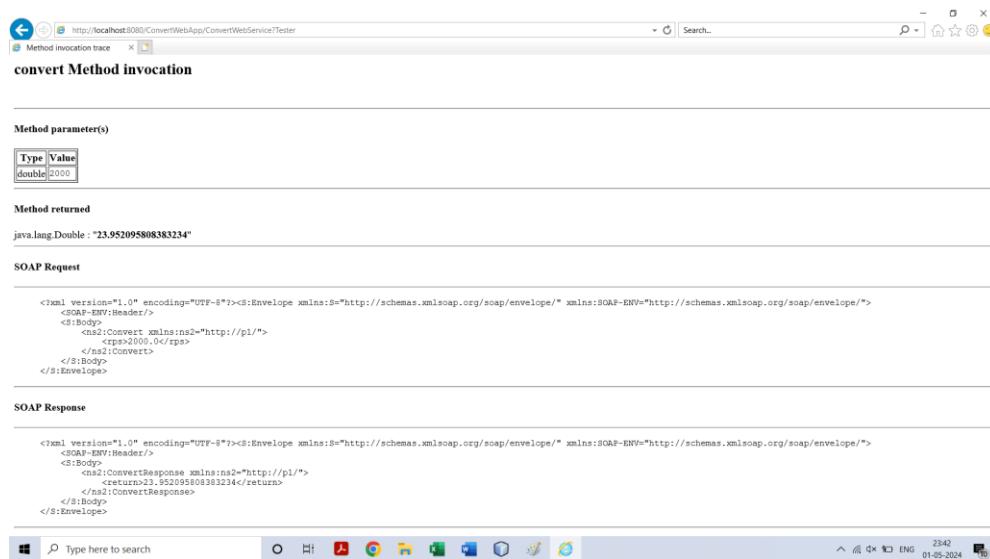
ii) Test Web service



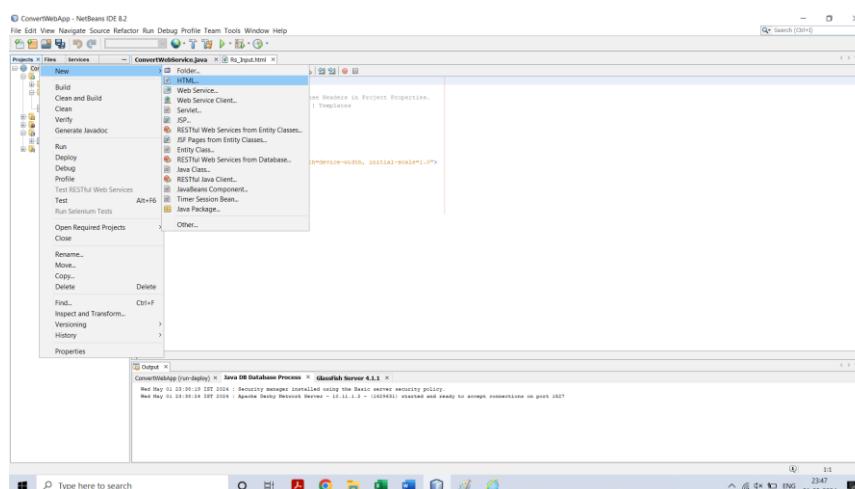
After testing

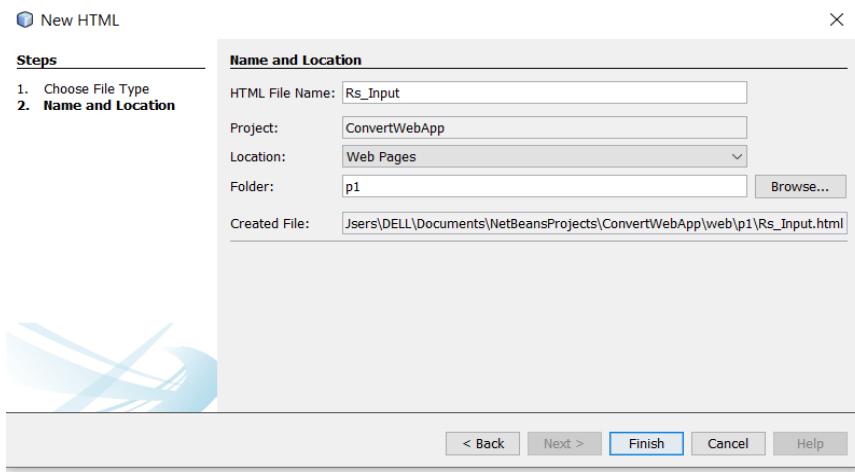


Successfully Deployed and Tested



Step 6: Now to take user input Create input.html page



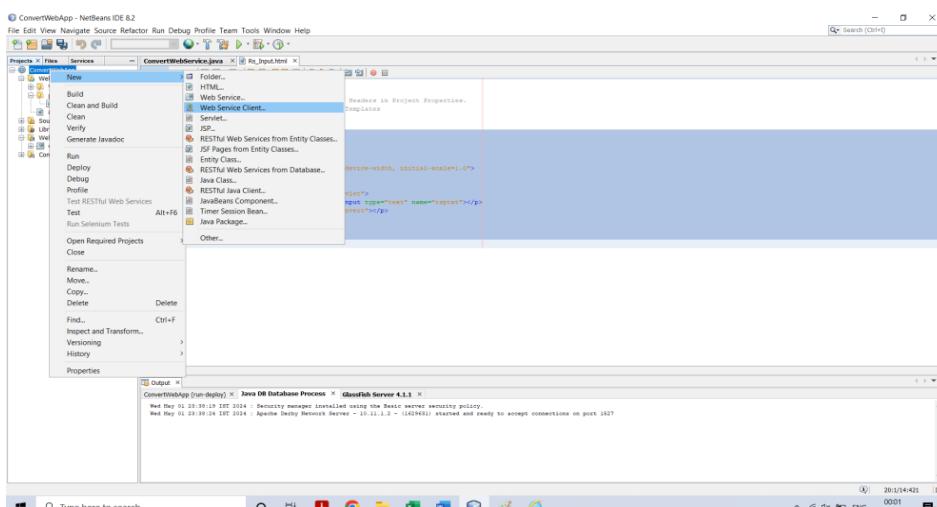


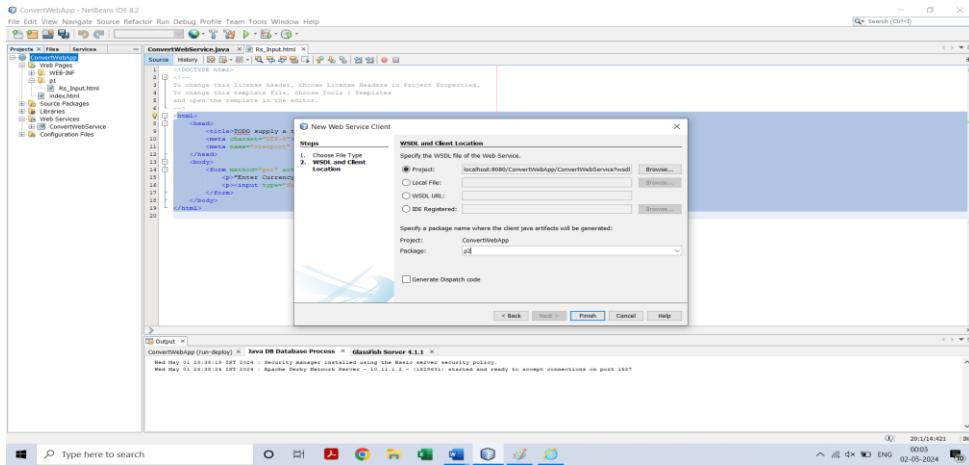
Now insert the code to create html page :

```
<html>
    <head>
        <title>TODO supply a title</title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
    </head>
    <body>
        <form method="get" action="Output.jsp">
            <p>"Enter Currency in Ruppes "<input type="text" name="rsptxt"></p>
            <p><input type="Submit" value="Convert"></p>
        </form>
    </body>
</html>
```

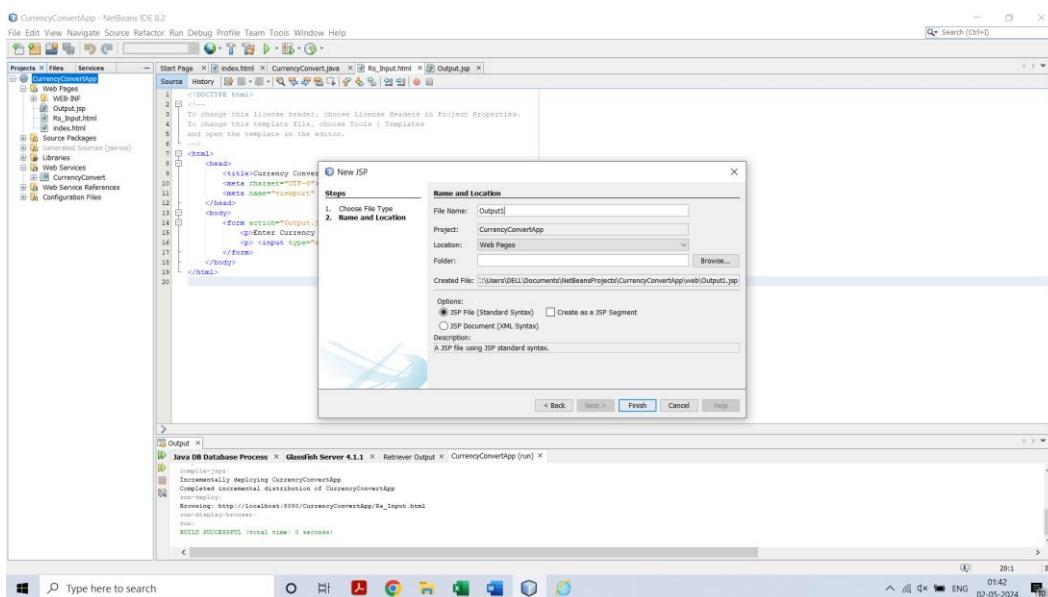
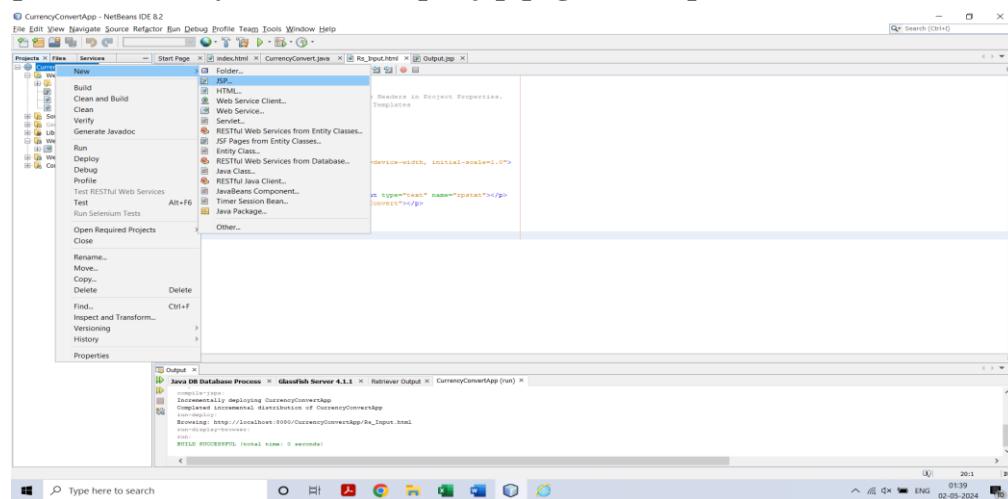
Step 7 :

Now 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:

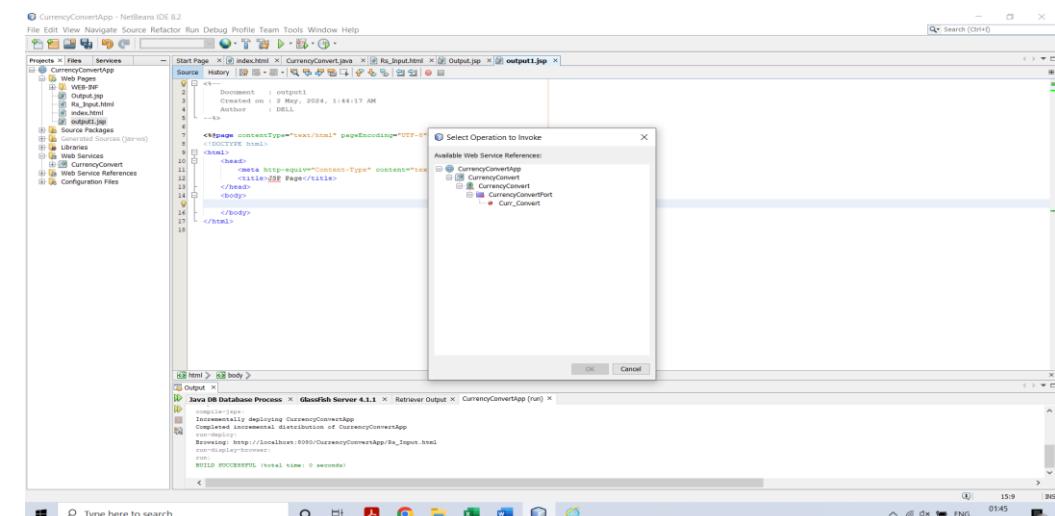
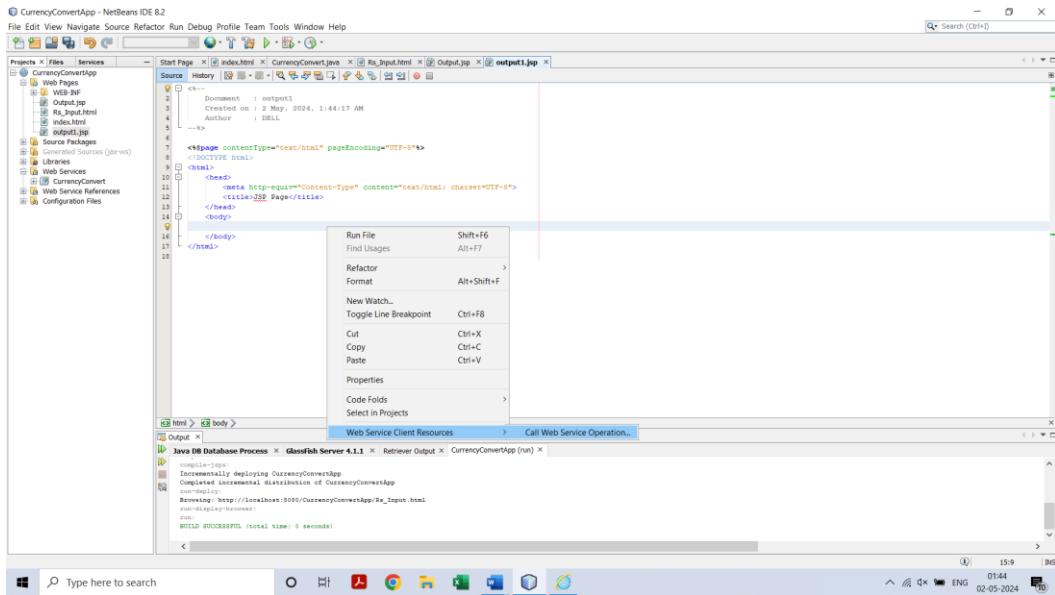




Step 8: Now finally to create Output.jsp page for output



Step 9: After creating the page code is to be added



Code to Output.jsp page

```

<%
try {
p2.CurrencyConvert_Service service = new p2.CurrencyConvert_Service();
p2.CurrencyConvert port = service.getCurrencyConvertPort();

double rps = Double.parseDouble(request.getParameter("rpstxt"));

java.lang.Double result = port.currConvert(rps);
out.println("Currency in Dollar :" +result);
} catch (Exception ex) {
// TODO handle custom exceptions here
}
%>

```

```

<%-- Document : Output
   Created on : 2 May, 2024, 1:31:46 AM
   Author : DEBLI
--%>
<%-- start web service invocation --%><hr/>
<% try {
    p2.CurrencyConvert_Service service = new p2.CurrencyConvert_Service();
    p2.CurrencyConvert port = service.getCurrencyConvertPort();
    java.lang.Double rps = Double.parseDouble(request.getParameter("rps"));
    java.lang.Double result = port.currencyConvert(rps);
    out.println("Currency in dollar :" + result);
} catch (Exception ex) {
    // TODO handle custom exceptions here
}
<%-- end web service invocation --%></hr/>
</body>
</html>
```

Java DB Database Process > GlassFish Server 4.1.1 > Retriever Output | CurrencyConvertApp (run) >

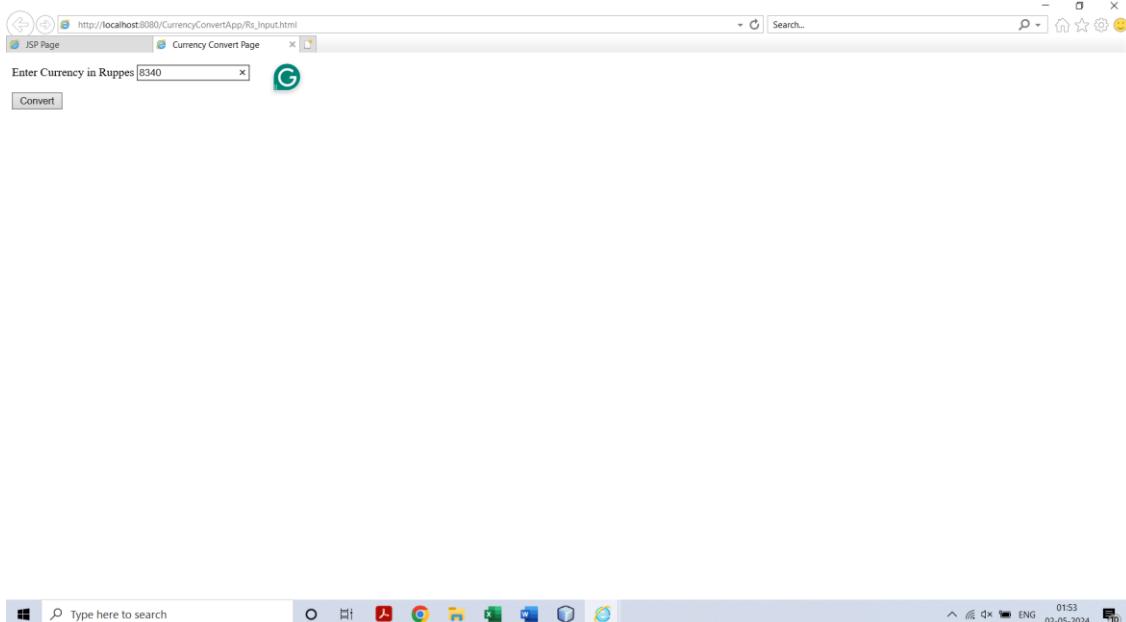
```

[INFO] [Deployer-1-Thread-1] Increasingly deploying CurrencyConvertApp
Completed incremental distribution of CurrencyConvertApp
run-deploy:
[Browsing: http://localhost:8080/CurrencyConvertApp/Rs_Input.html]
run-deployer:
run
[Browsing: http://localhost:8080/CurrencyConvertApp/Rs_Output.html]
run
BUILD SUCCESSFUL (total time: 0 seconds)

```

Step 10 : Run the application. The following output will be generated

Output :

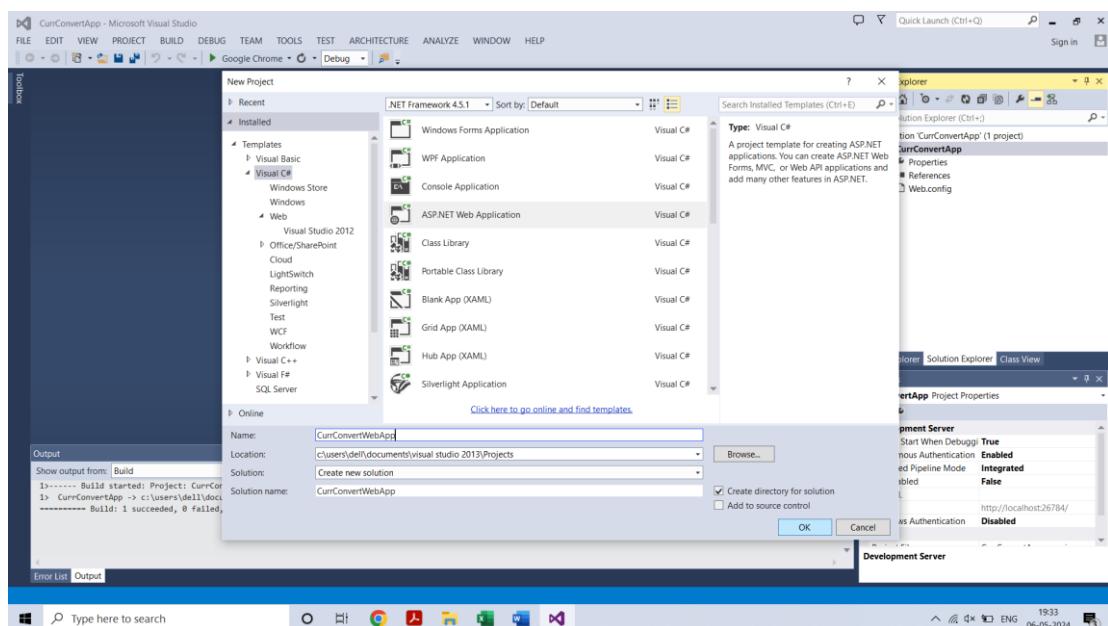
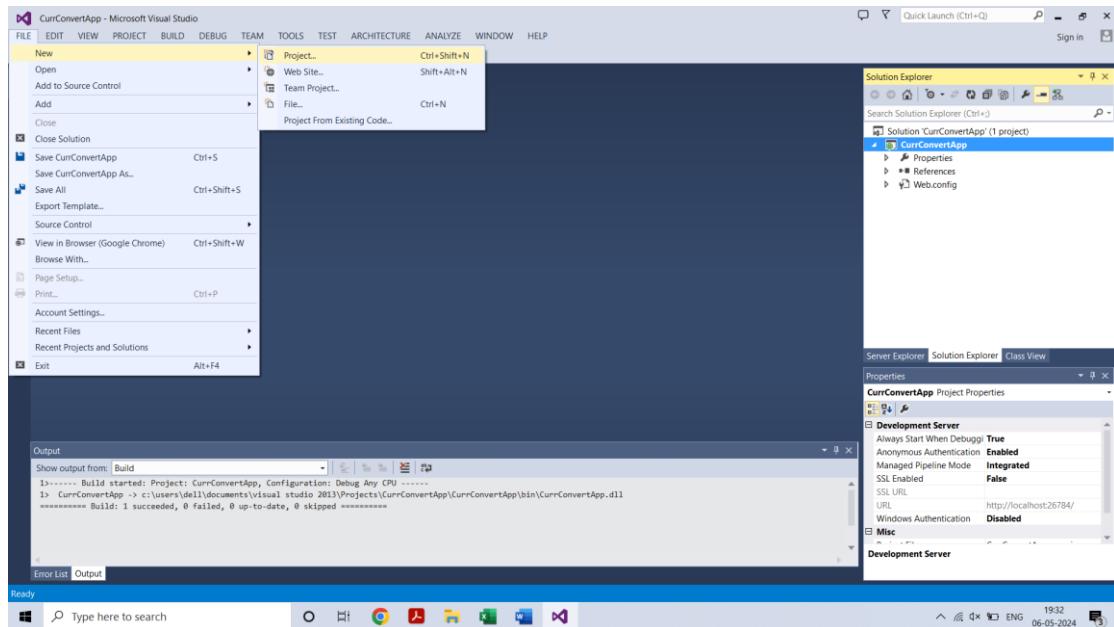


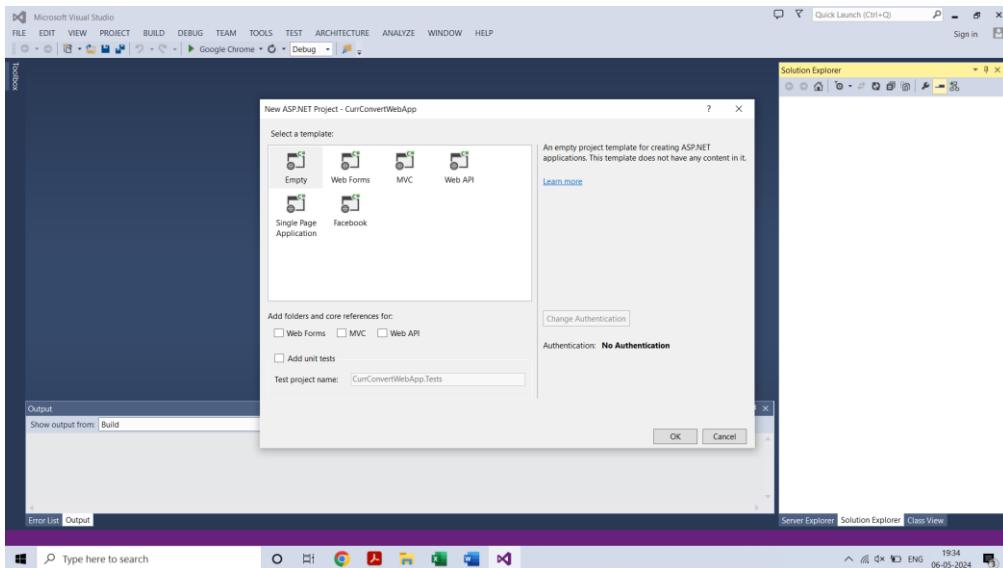


Practical-1(B)

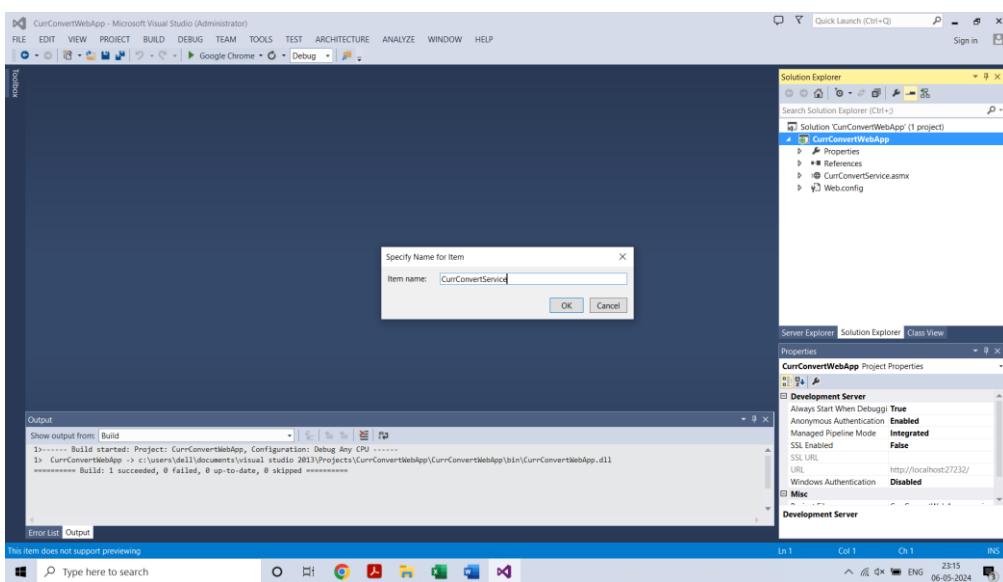
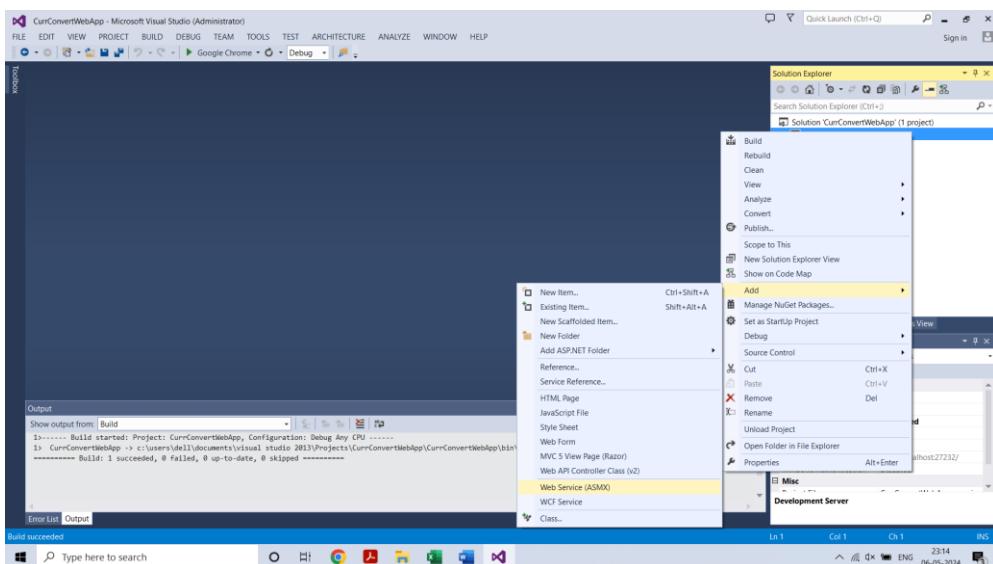
Define a simple services like Converting Rupees into Dollar and Call it from different platform like .NET

Step1 : To create Web Service through Visual Studio



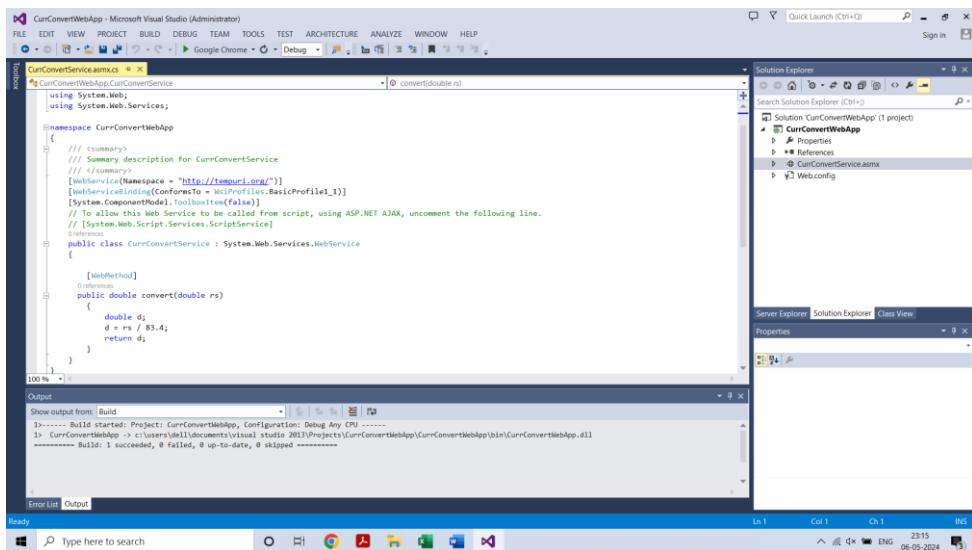


Now to Create a web Service :

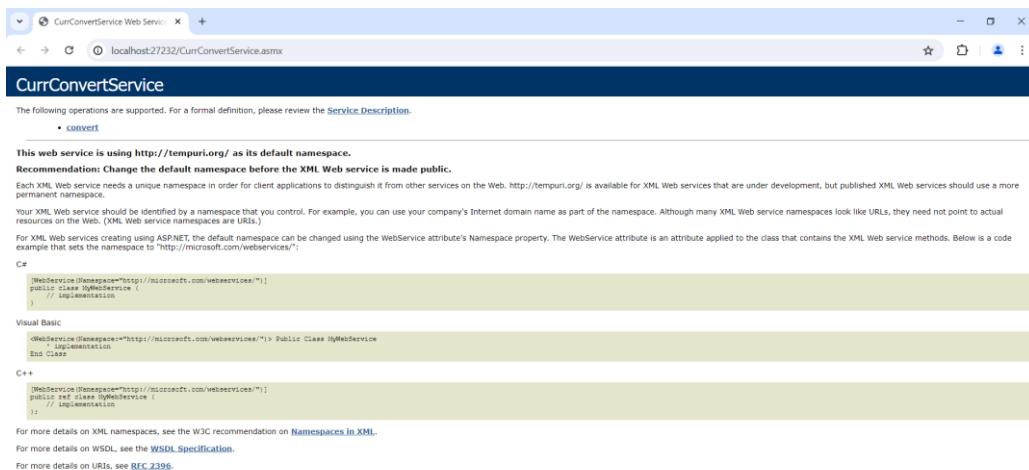


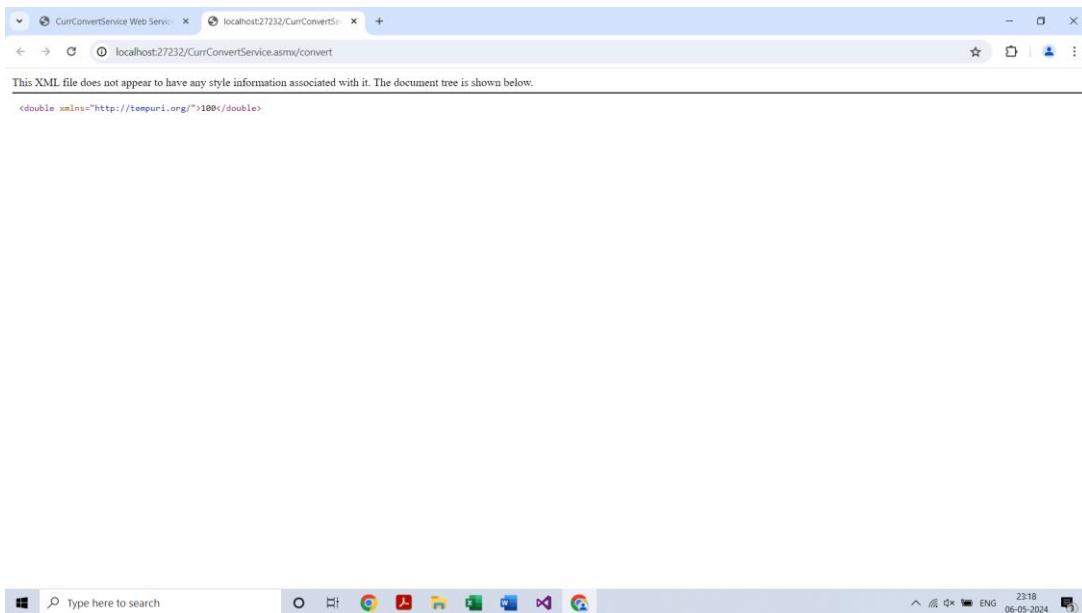
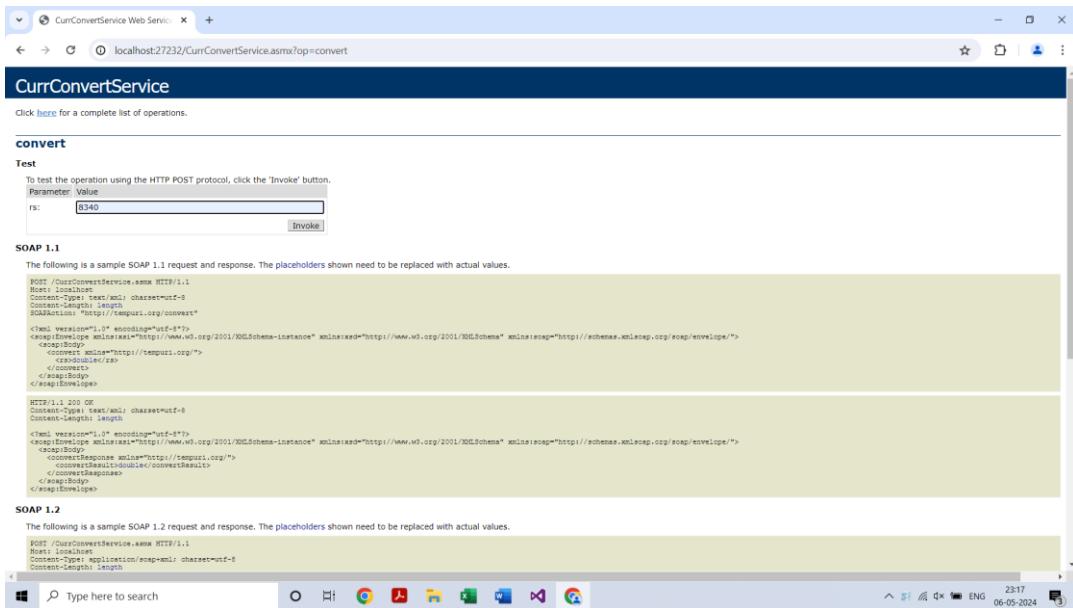
After click on Add button, CurrConvertService.asmx.cs file will be automatically opened.
Add the following code into Class CurrConvertService :

```
public double convert(double rs)
{
    double d;
    d = rs / 83.4;
    return d;
}
```

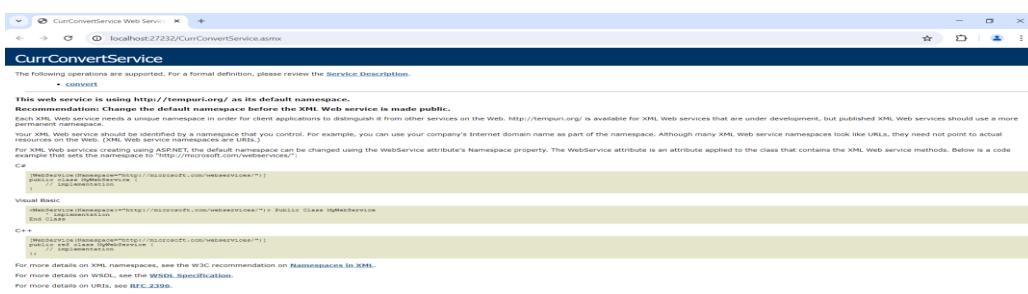


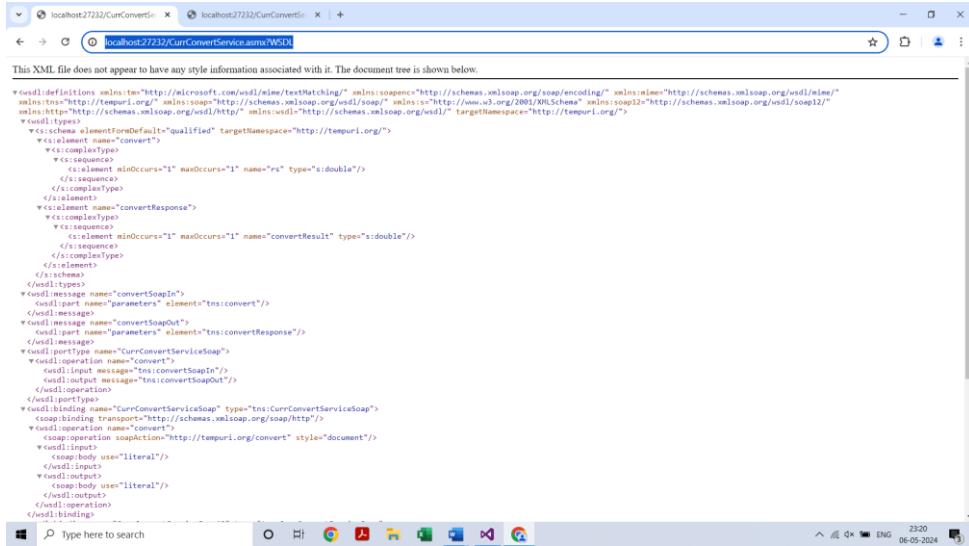
Save the changes. And run the project. A window will open in browser and that is the web service





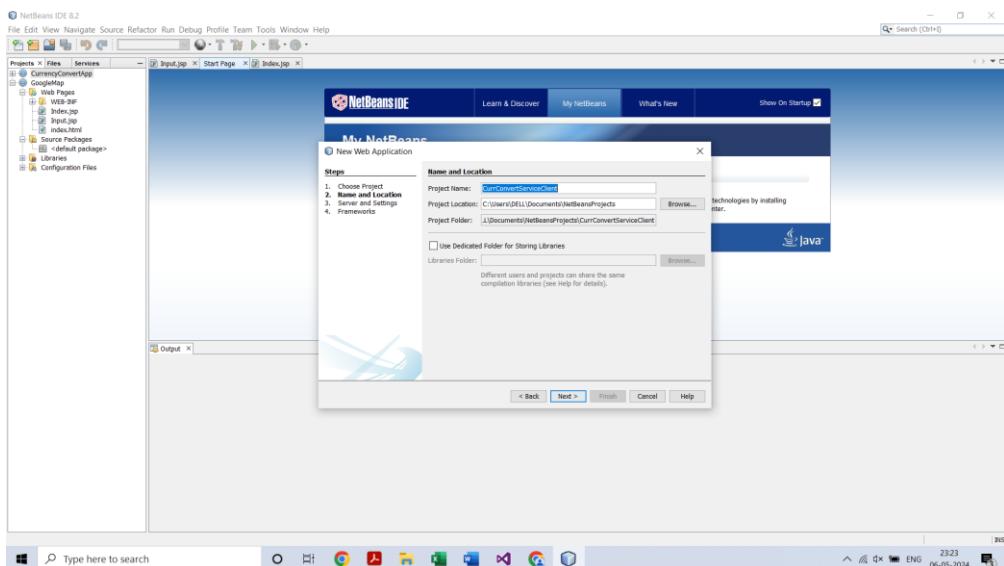
Now click on Service Description option. Now a new window will be opened. The link is selected and copied. We will use this link in NetBeans to consume this service





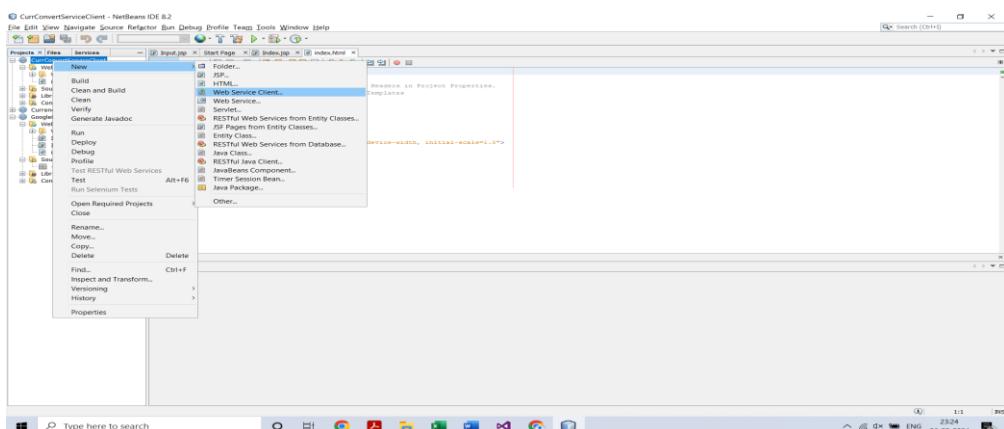
Now NetBeans is opened.

To Create a Web Application with name CurrConvertServiceClient. Next -> Finish

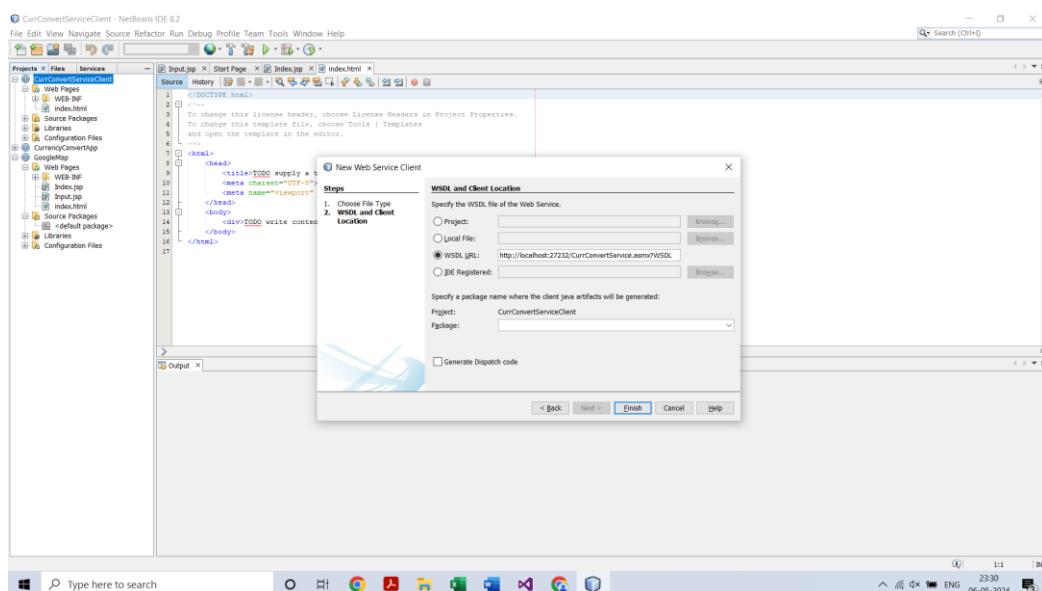


Now a Web Service Client is created.

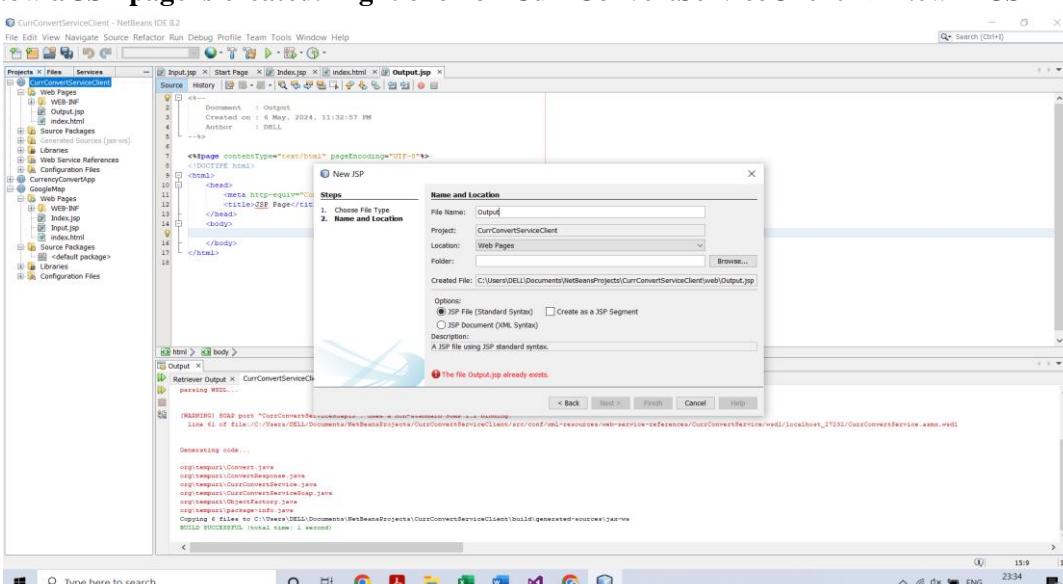
Right click on CurrConvertServiceClient -> New -> Web Service Client.



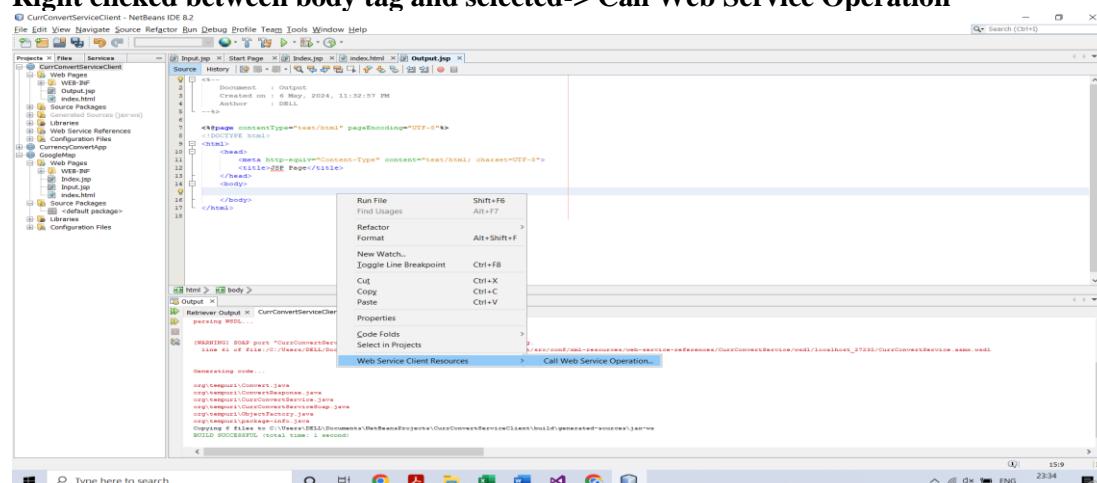
WSDL URL is selected and pasted the link that is copied from browser on run of Visual Studio. After that click on Finish button.



Now a JSP page is created. Right click on CurrConvertServiceClient -> New -> JSP



Right clicked between body tag and selected-> Call Web Service Operation



```

<?xml version="1.0" encoding="UTF-8"?>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>JSF Page</title>
    </head>
    <body>
        <%-- start web service invocation --%><hr/>
        <% try {
            org.tempuri.CurrConvertService service = new org.tempuri.CurrConvertService();
            org.tempuri.CurrConvertServiceSoap port = service.getCurrcanvertServiceSoap();
            port.setPortName("CurrcanvertServiceSoap");
            double rs = Double.parseDouble(request.getParameter("txtrs"));
            Double result = port.convert(rs);
            out.println("Currency in Dollar = "+result);
        } catch (Exception ex) {
            ex.printStackTrace();
        }
        %>
        <%-- end web services invocation --%><hr/>
    </body>
</html>

```

Retriever Output X: CurrConvertServiceClient (import-client-CurrConvertService) X
passing null...

(WARNING) SOAP port "CurrcanvertServiceSoap11", uses a non-standard SOAP 1.1 binding.
line 6 of file:///C:/Users/DELL/Documents/NetBeansProjects/CurrConvertServiceClient/src/conf/xml-resources/web-service-references/CurrConvertService.wsdl

Generating code...

org.tempuri.Convert.java
org.tempuri.ConvertResponse.java
org.tempuri.CurrConvertService.java
org.tempuri.CurrConvertServiceSoap.java
org.tempuri.ObjectFactory.java
org.tempuri.CurrConvertServiceStub.java
Copying 4 files to C:/Users/DELL/Documents/NetBeansProjects/CurrConvertServiceClient/build/generated-sources/jax-ws
BUILD SUCCESSFUL (total time: 1 second)

Now Input.html file of CurrConvertServiceClient project is created and opened and the contents of body tag is replaced with following code. And then saved it.

```

<form action="Output.jsp">
    <p>Enter Currency in Ruppes : <input type="text" name="txtrs"></p>
    <p><input type="submit" value="Convert"></p>
</form>

```

Retriever Output X: CurrConvertServiceClient (import-client-CurrConvertService) X
passing null...

(WARNING) SOAP port "CurrcanvertServiceSoap11", uses a non-standard SOAP 1.1 binding.
line 6 of file:///C:/Users/DELL/Documents/NetBeansProjects/CurrConvertServiceClient/src/conf/xml-resources/web-service-references/CurrConvertService.wsdl

Generating code...

org.tempuri.Convert.java
org.tempuri.ConvertResponse.java
org.tempuri.CurrConvertService.java
org.tempuri.CurrConvertServiceSoap.java
org.tempuri.ObjectFactory.java
org.tempuri.CurrConvertServiceStub.java
org.tempuri.package-info.java
Copying 4 files to C:/Users/DELL/Documents/NetBeansProjects/CurrConvertServiceClient/build/generated-sources/jax-ws
BUILD SUCCESSFUL (total time: 1 second)

The screenshot shows the NetBeans IDE interface with the title bar "CurrConvertServiceClient - NetBeans IDE 8.2". The menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help. The toolbar has icons for New, Open, Save, Cut, Copy, Paste, Find, Replace, and others. The Projects tab shows the "CurrConvertServiceClient" project with files like Web Page, Web Service References, GoogleMap, and CurrencyConverterPage. The Services tab shows "Input.jsp", "Output.jsp", "index.jsp", and "index.html". The Editor tab shows "Input.jsp" with the following code:

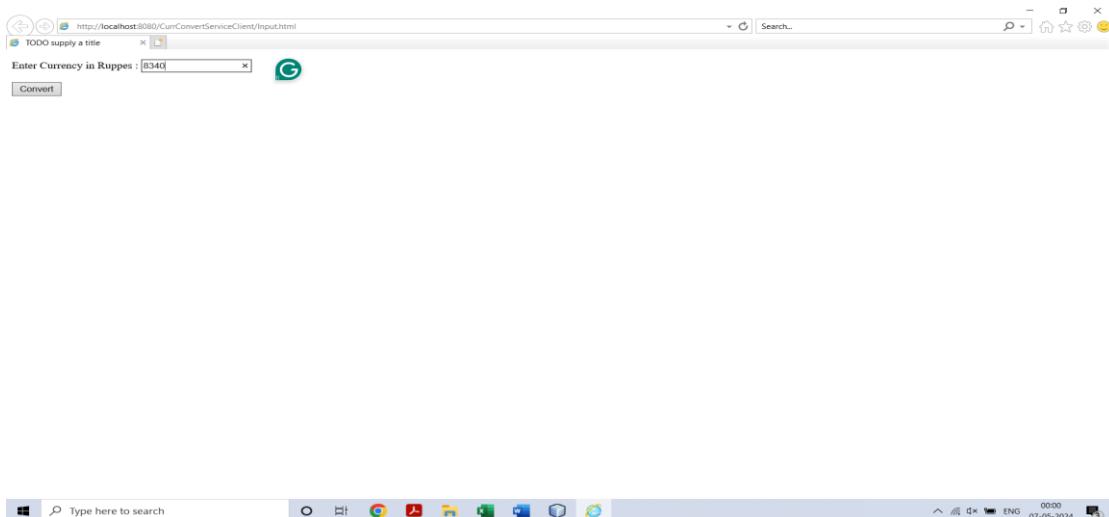
```

<%@PAGE LANGUAGE="JSP" %>
<html>
    <head>
        <title>2000 supply a title</title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
    </head>
    <body>
        <form action="Output.jsp">
            <p>Enter Currency in Ruppes : <input type="text" name="txtrup"></p>
            <p> <input type="button" value="convert" /></p>
        </form>
    </body>
</html>

```

The Output tab shows the generated Java code for the "Input.jsp" file, which includes imports for javax.xml.ws, javax.xml.ws.soap, javax.xml.ws.handler, javax.xml.ws.handler.message, javax.xml.ws.handler.soap, javax.xml.ws.soap.message, javax.xml.ws.soap.message.property, javax.xml.ws.soap.message.property.value, javax.xml.ws.soap.message.property.value.type, javax.xml.ws.soap.message.property.value.type.java, javax.xml.ws.soap.message.property.value.type.java.package, javax.xml.ws.soap.message.property.value.type.java.package.info, and javax.xml.ws.soap.message.property.value.type.java.package.info.type. The code also includes annotations for @WebService, @WebMethod, and @RequestWrapper.

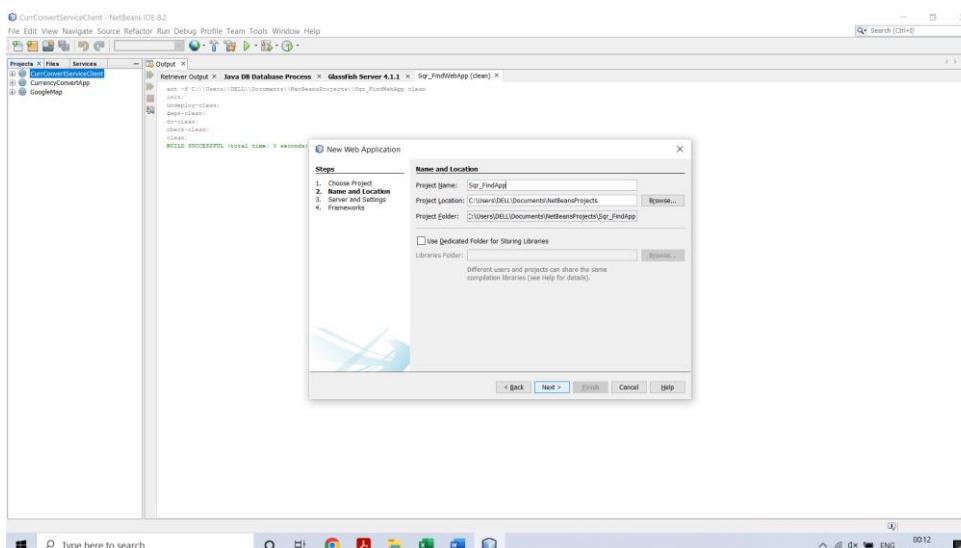
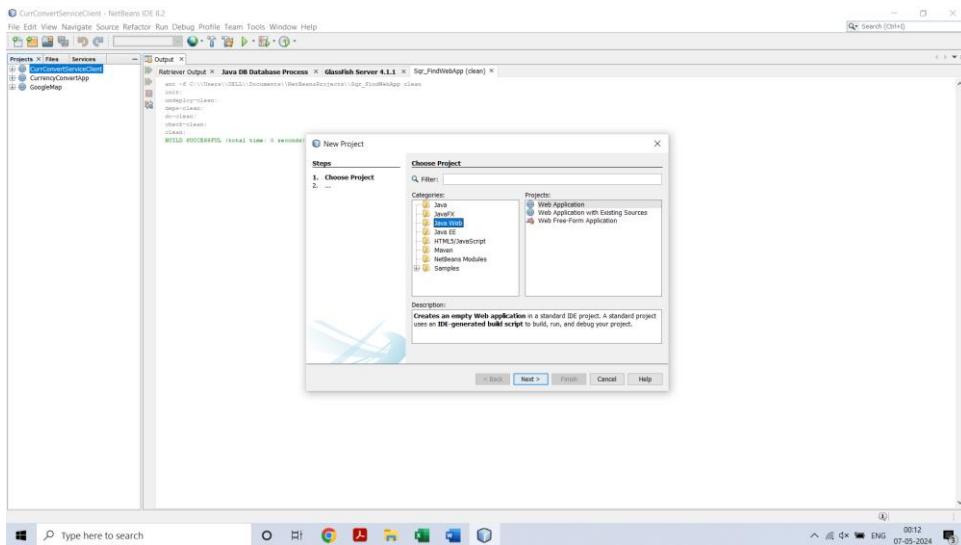
Now the web application is run.



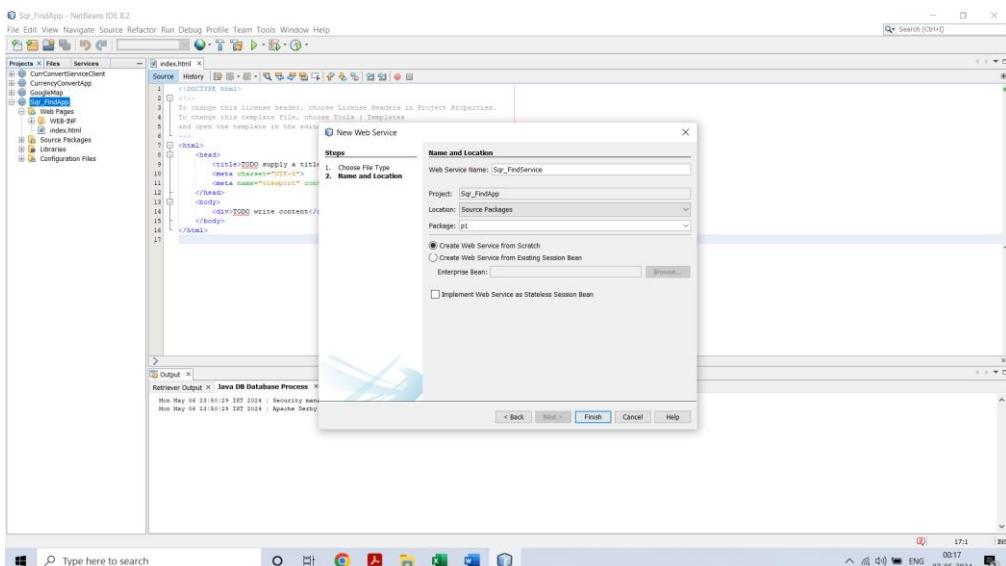
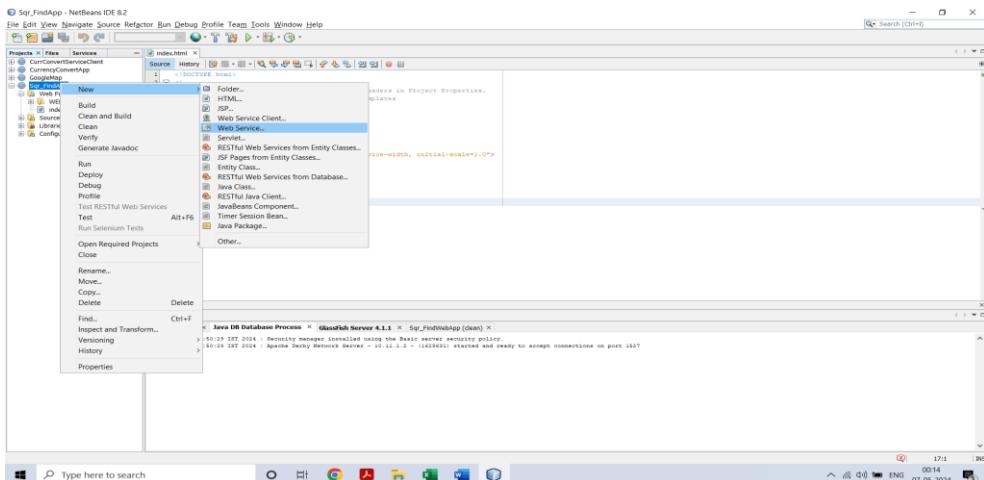
Practical-2

Create a Simple SOAP service to find Square of a Number

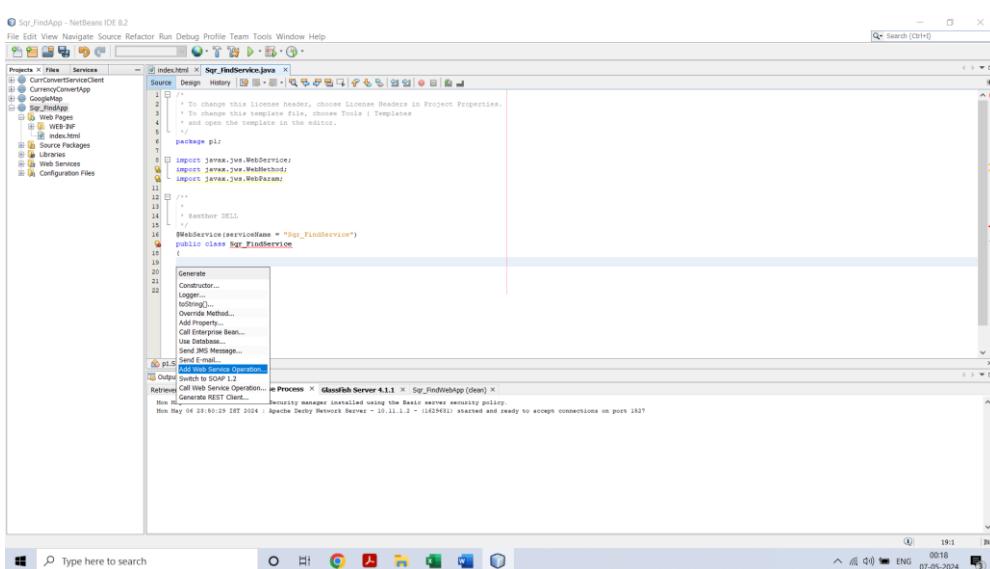
Step1 : To create Web Service through Net Beans

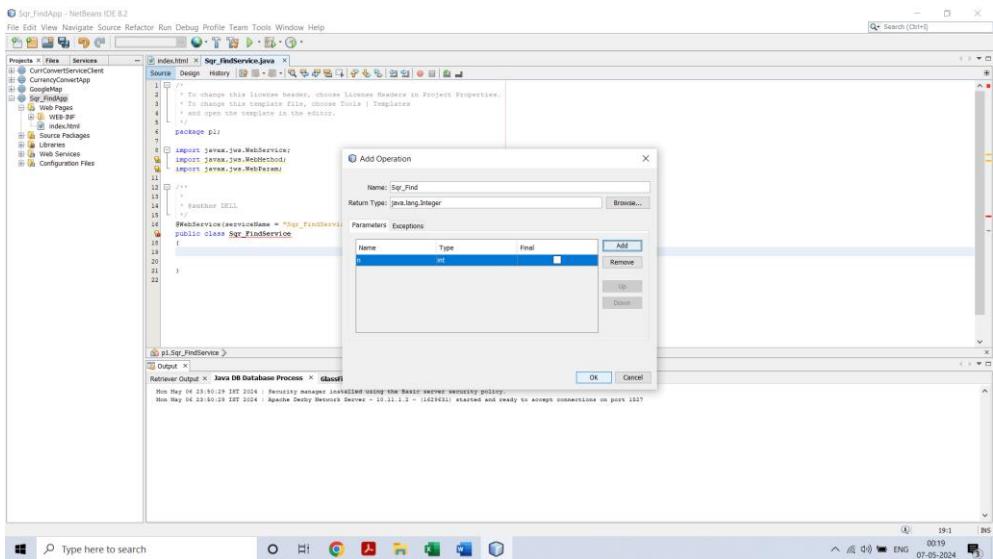


Step 2: To create Web Service “ Sqr_FindService”



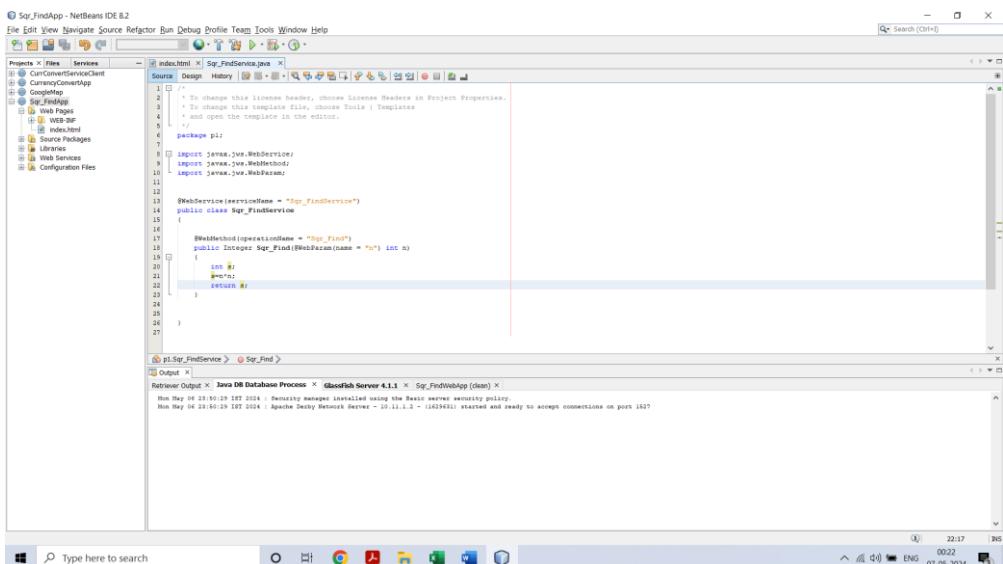
Step 3 : to Add Operation “Sqr_Find” to Web Service





Here Sqr_Find is a method having one parameter as n

Step 4 : After adding code of Operation to web Service

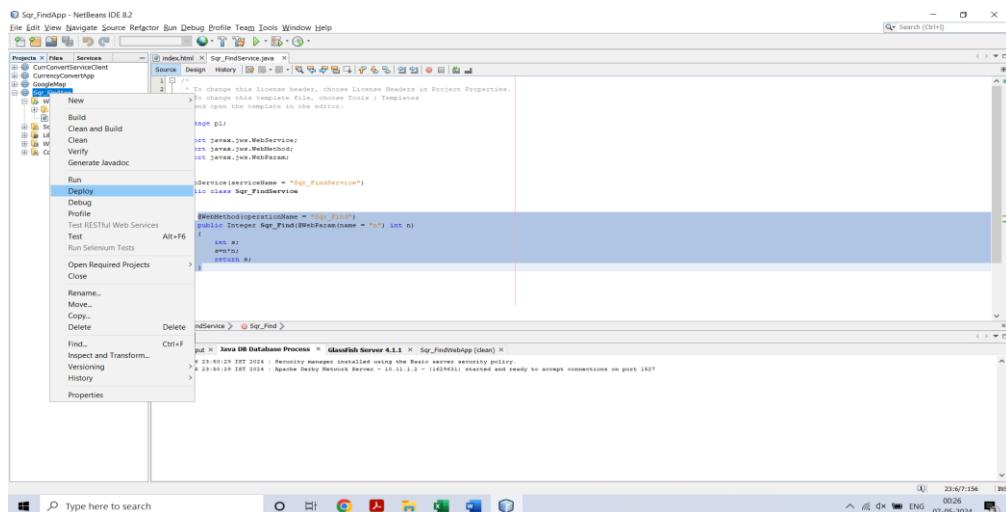


Code of Operation :

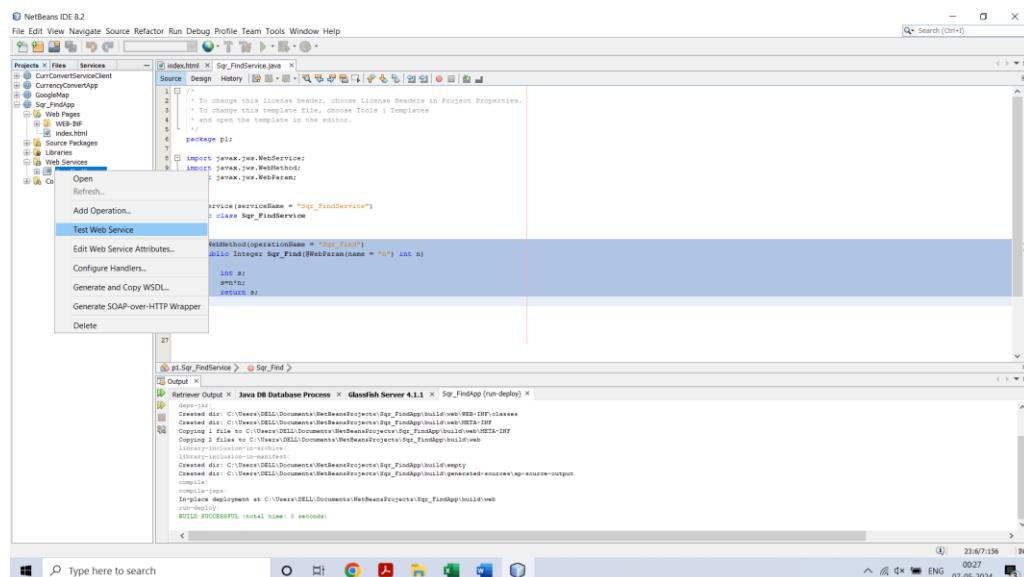
```
@WebMethod(operationName = "Sqr_Find")
public Integer Sqr_Find(@WebParam(name = "n") int n)
{
    int s;
    s=n*n;
    return s;
}
```

Step 5 : Now to deploy the web service and Test

iii) Deploy



iv) Test Web Service



After testing



Successfully Deployed and Tested

Method parameter(s)

Type	Value
int	10

Method returned

```
java.lang.Integer : "100"
```

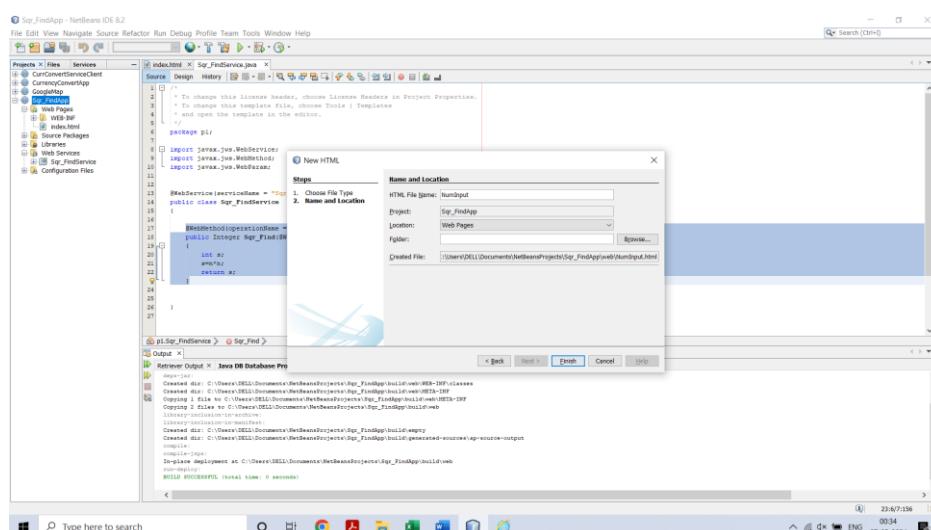
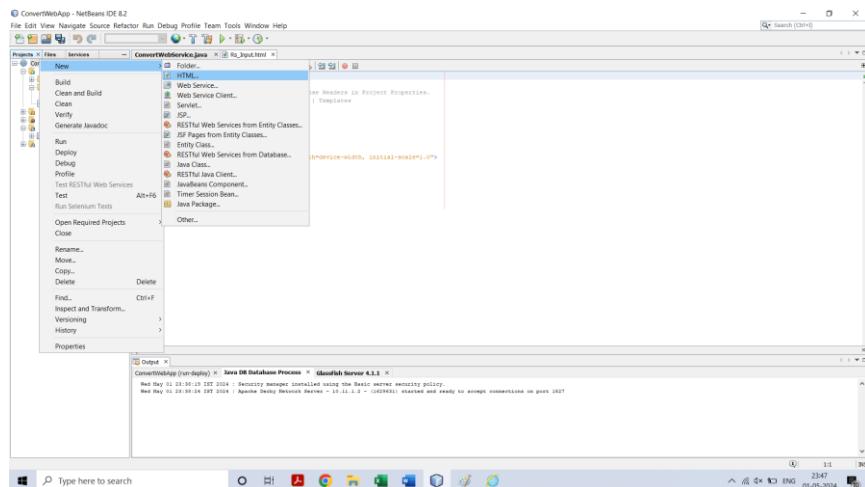
SOAP Request

```
<?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>
<S:Body>
<n2:Sqr_Find xmlns:n2="http://p1/">
</n2:Sqr_Find>
</S:Body>
</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>
<S:Body>
<n2:Sqr_FindResponse xmlns:n2="http://p1/">
<n2:Sqr_FindReturn>
</n2:Sqr_FindResponse>
</S:Body>
</S:Envelope>
```

Step 6: Now to take user input Create NumInput.html page

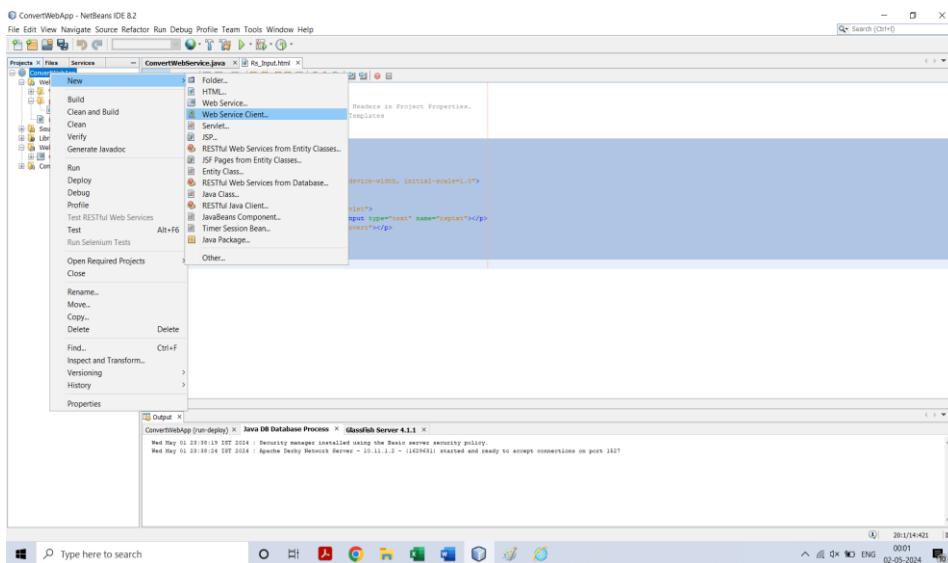


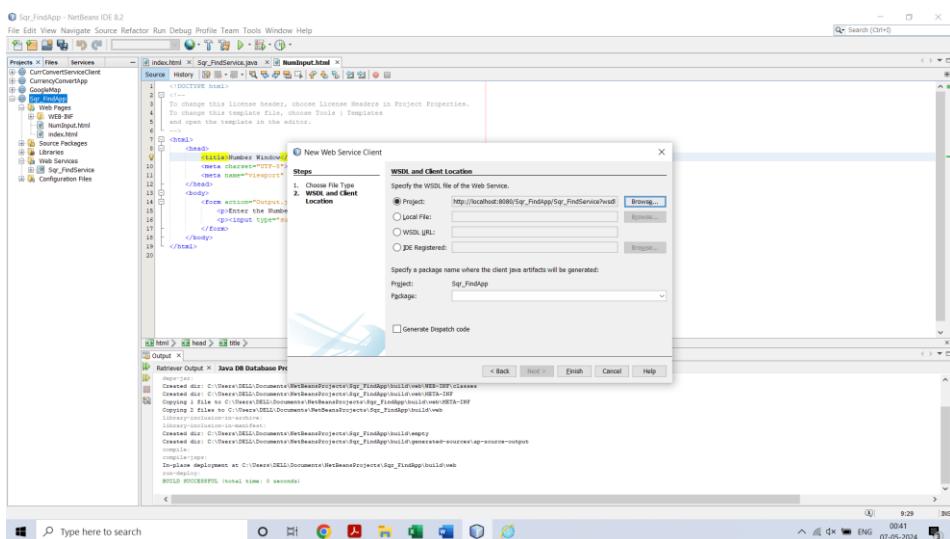
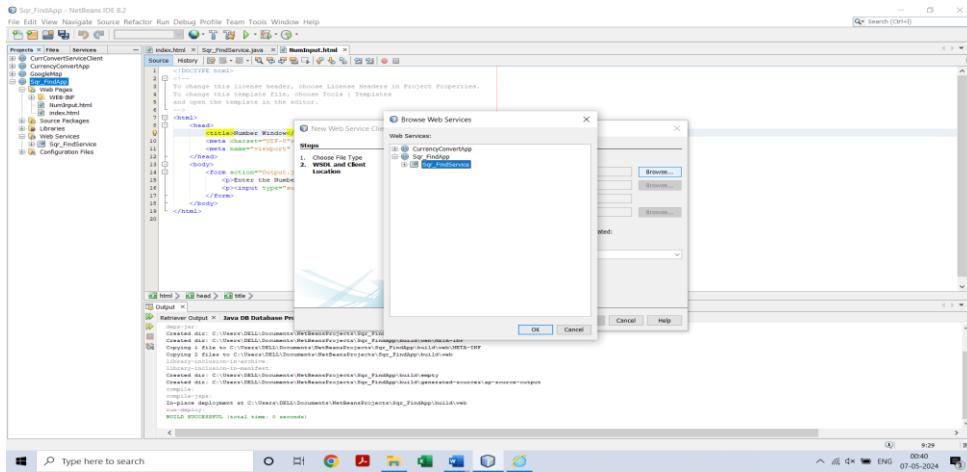
Now insert the code to create html page :

```
<html>
<head>
    <title>Number Window</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
<body>
    <form action="Output.jsp">
        <p>Enter the Number : <input type="text" name="txtnum"></p>
        <p><input type="submit" value="Find_SQR"></p>
    </form>
</body>
</html>
```

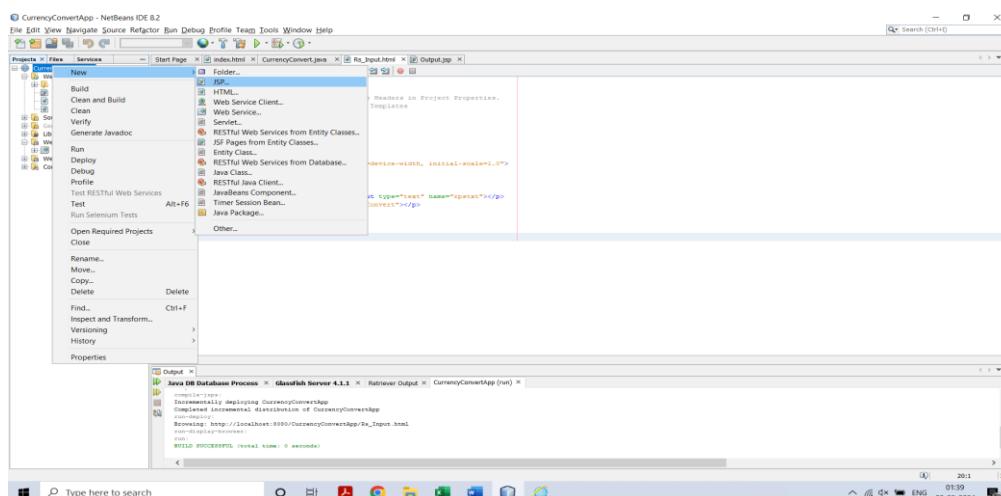
Step 7 :

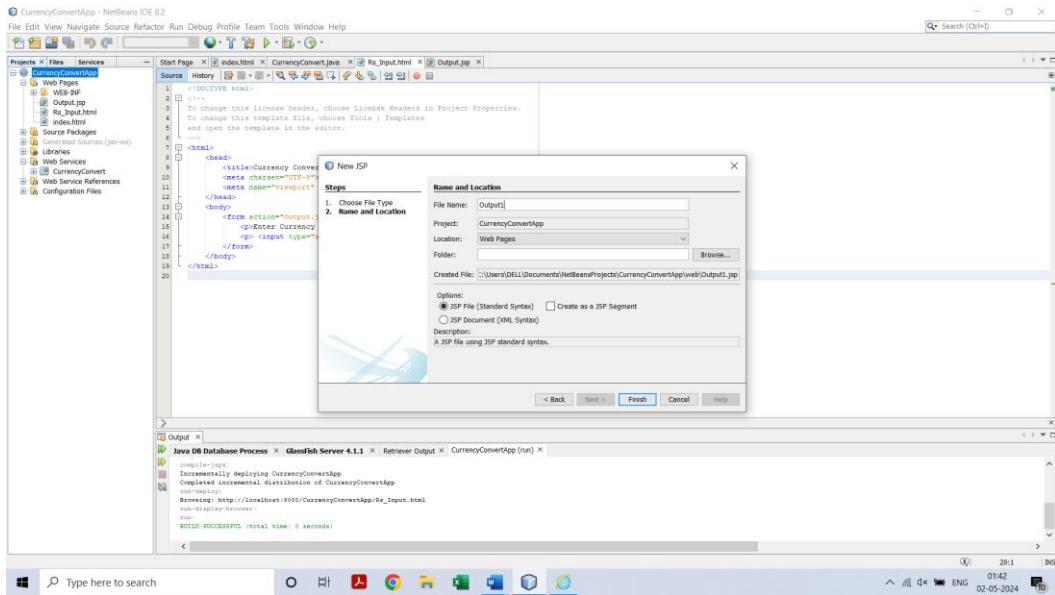
Now to create web service client, for same right click on project and select new then select web service client you will get the following screen:



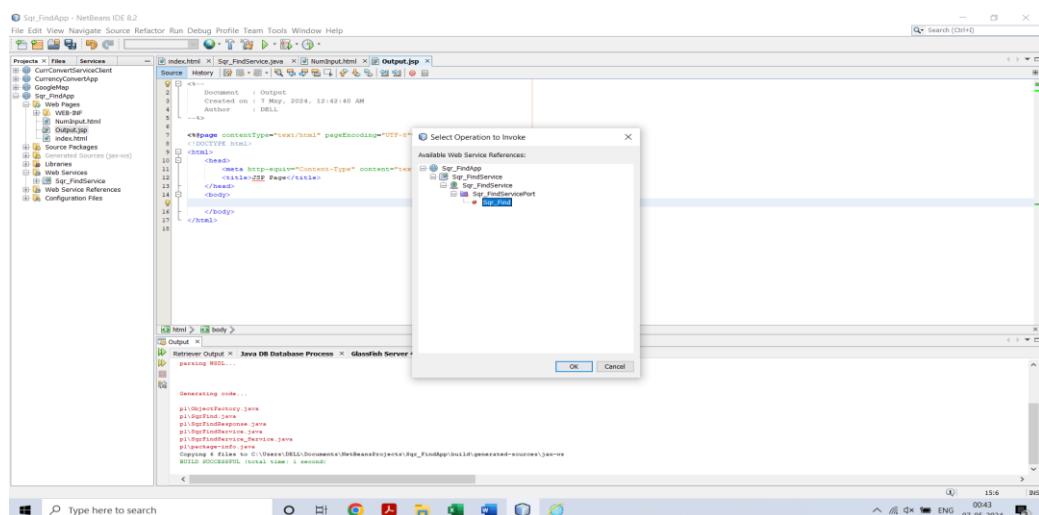
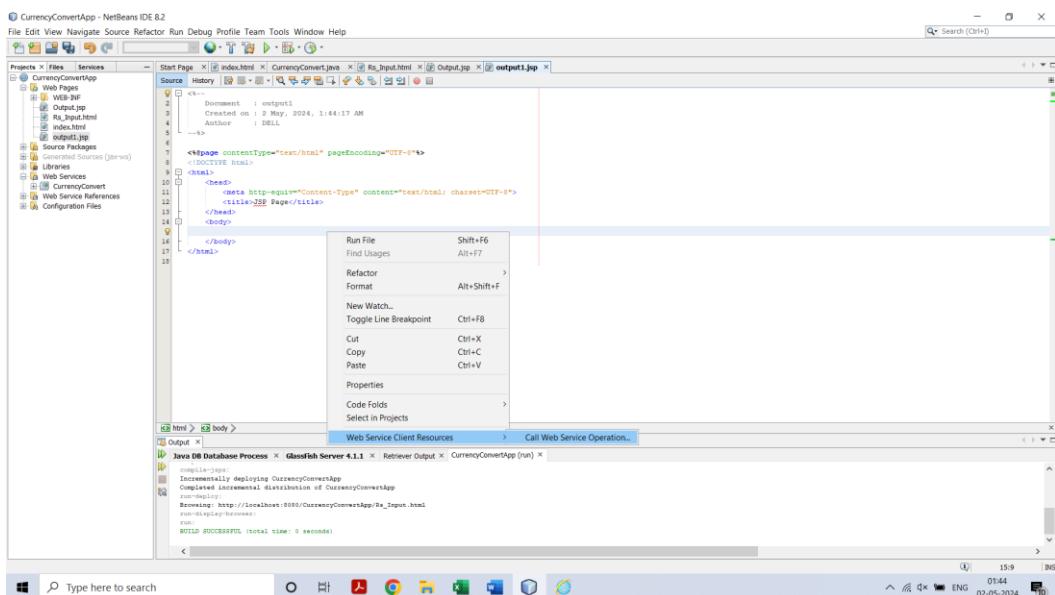


Step 8: Now finally to create Output.jsp page for output





Step 9: After creating the page code is to be added



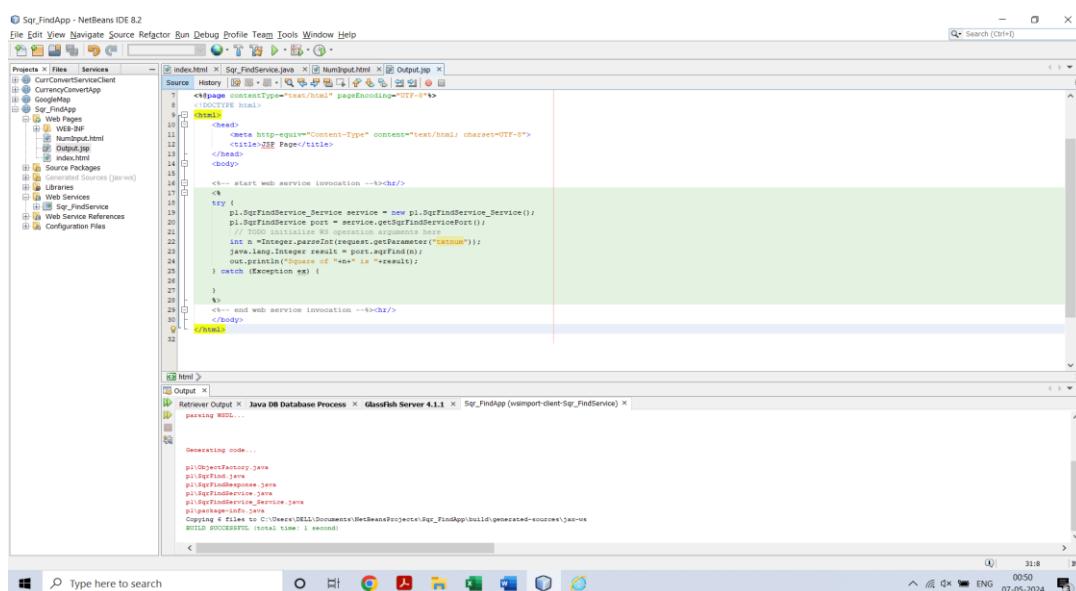
Code to Output.jsp page

```
<%
try {
    p1.SqrFindService_Service service = new p1.SqrFindService_Service();
    p1.SqrFindService port = service.getSqrFindServicePort();

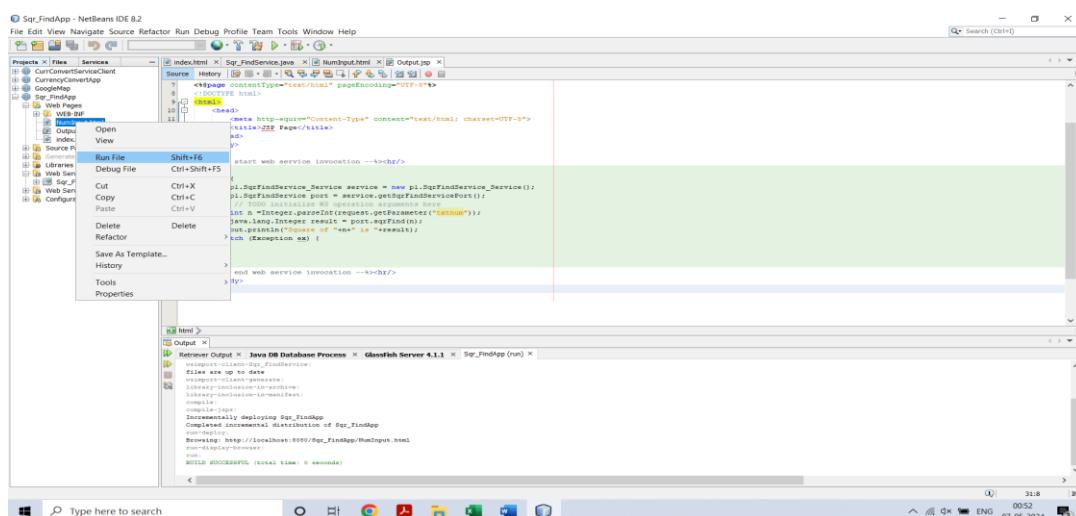
    int n =Integer.parseInt(request.getParameter("txtnum"));
    java.lang.Integer result = port.sqrFind(n);
    out.println("Square of "+n+" is "+result);

}
catch (Exception ex)
{
}

%>
```



Step 10 : Run the application. The following output will be generated



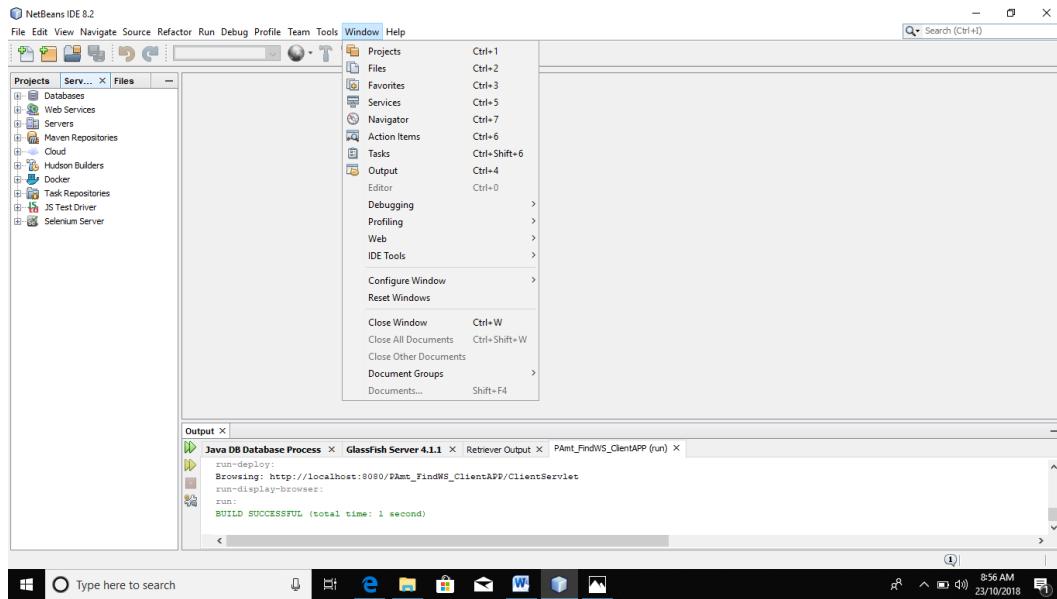
Output :



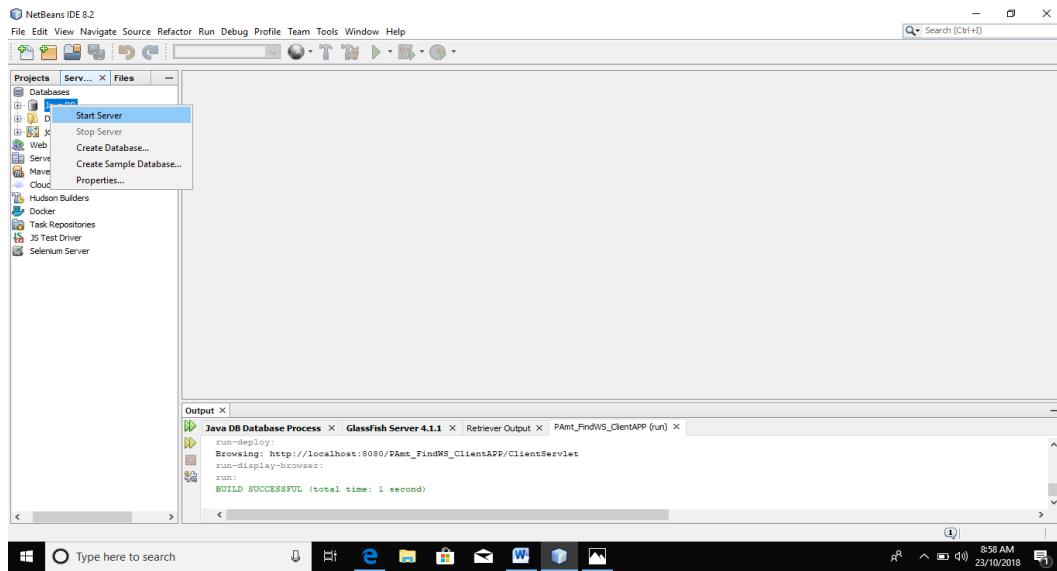
Practical-3

Create a Simple REST Service to demonstrate CRUD operations with “Student” database

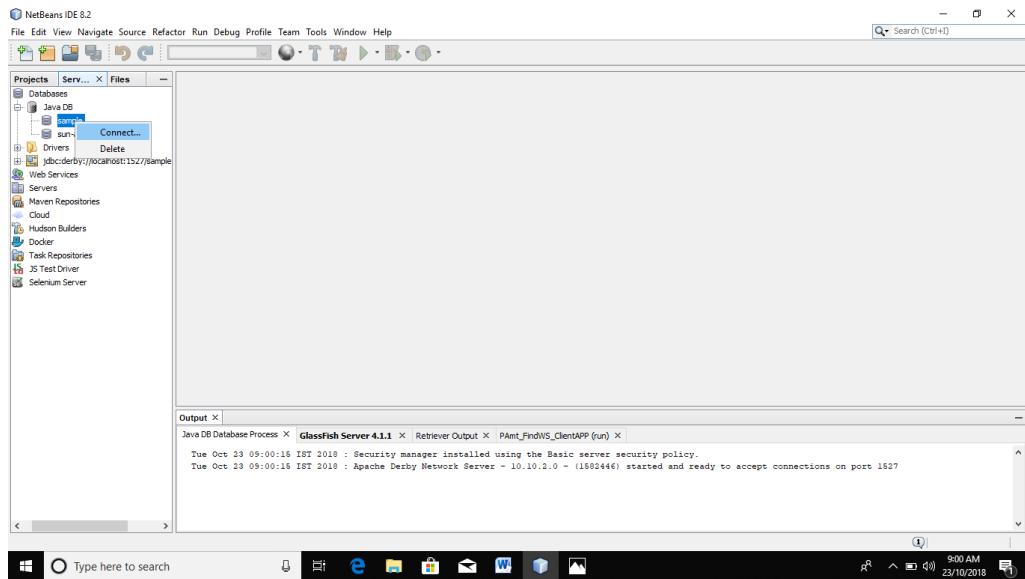
1. Click on Window menu and click on Projects, Files & Services to open it.



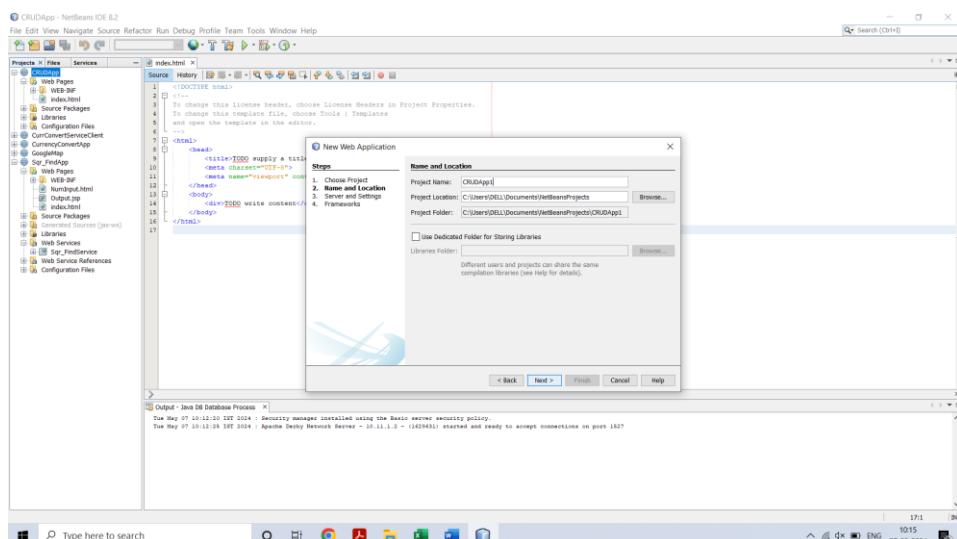
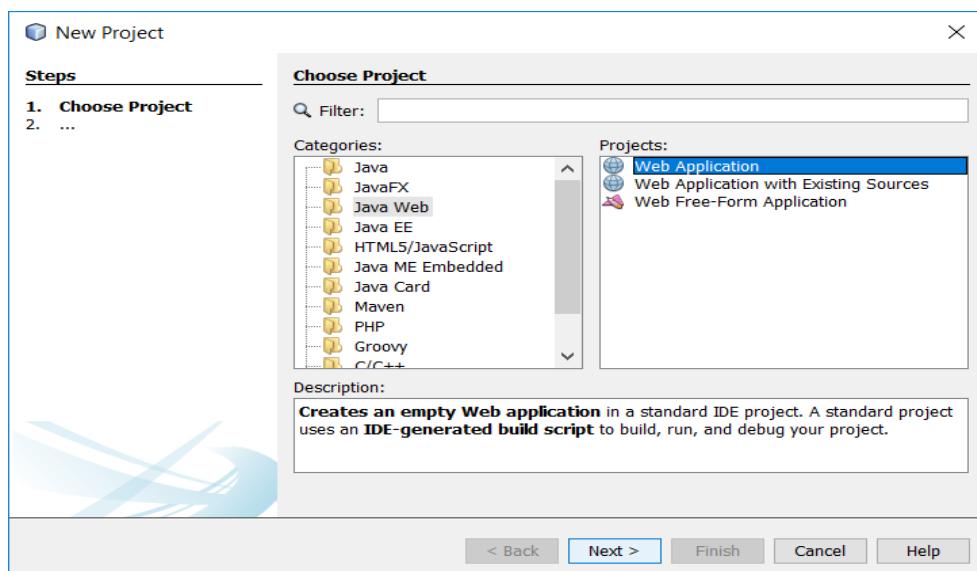
2. Right click on Java DB and then click on Start Server to start the server .



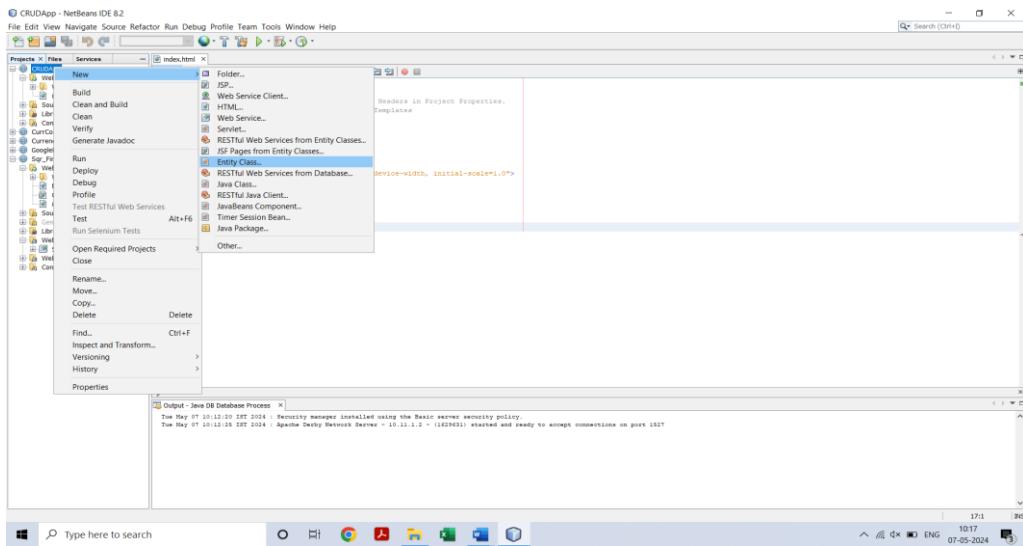
3. Now expand Java DB and right click on sample and then click on connect to connect the sample database with server.



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



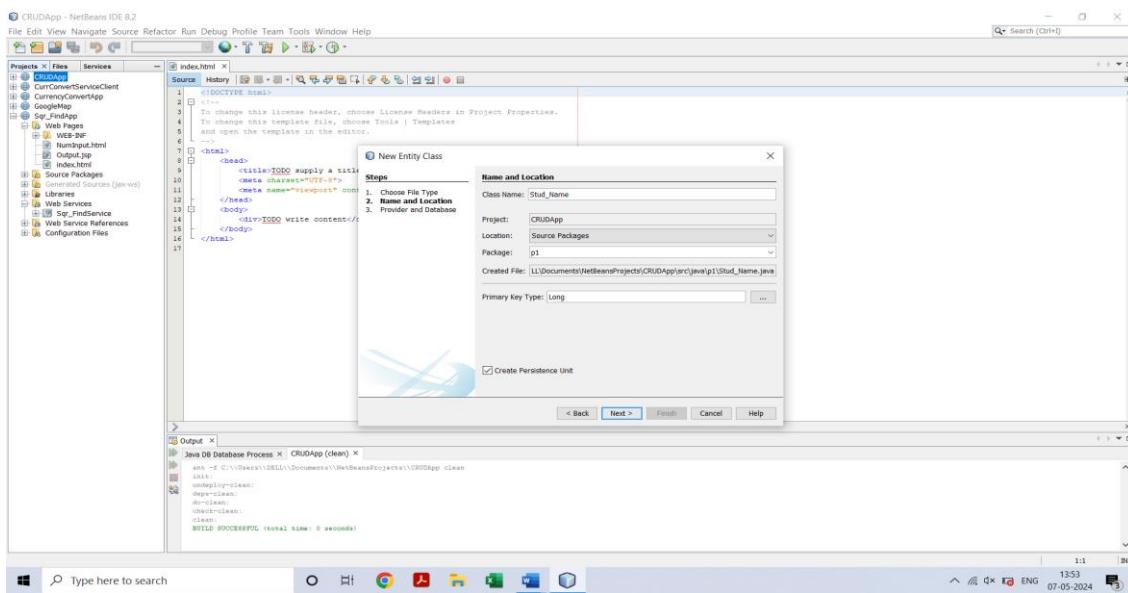
5. Create an entity class. Right click on project name -> New -> Entity Class.

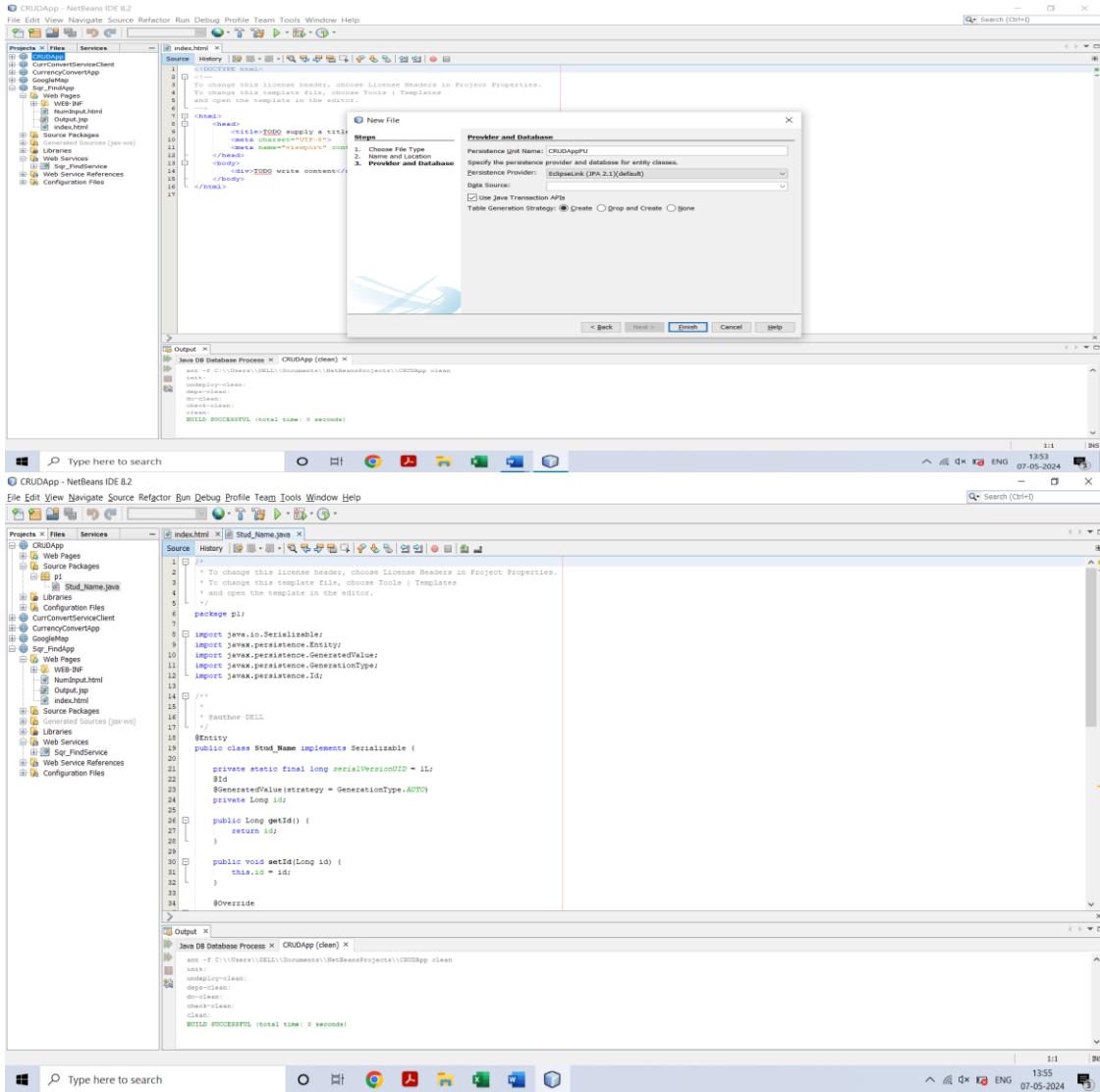


6. A window will appear like bellow pic. Enter following data and click on Next

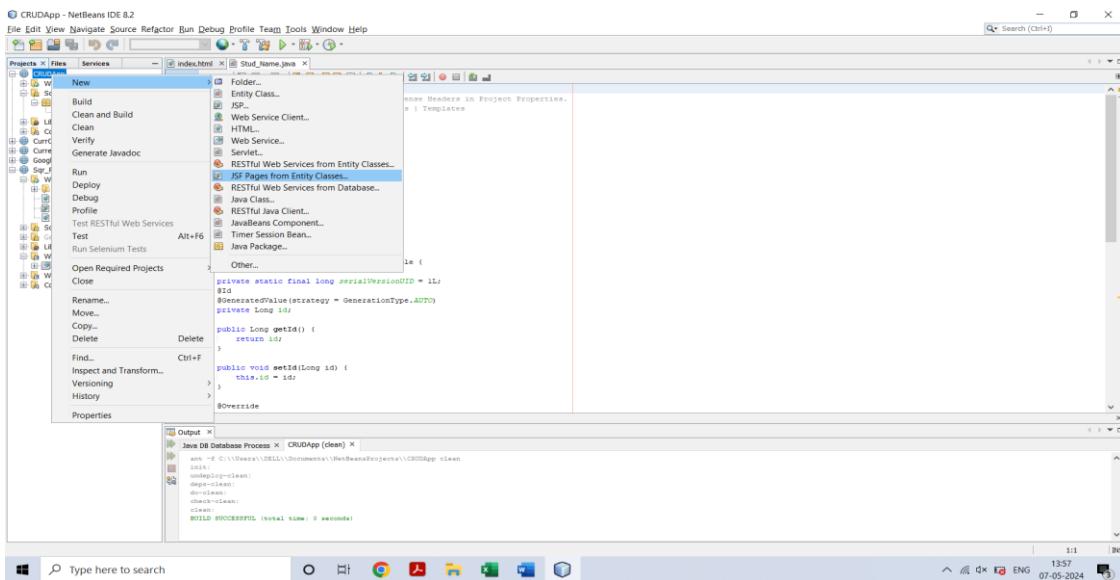
Class Name ->Stud_Name Package Name -> p1

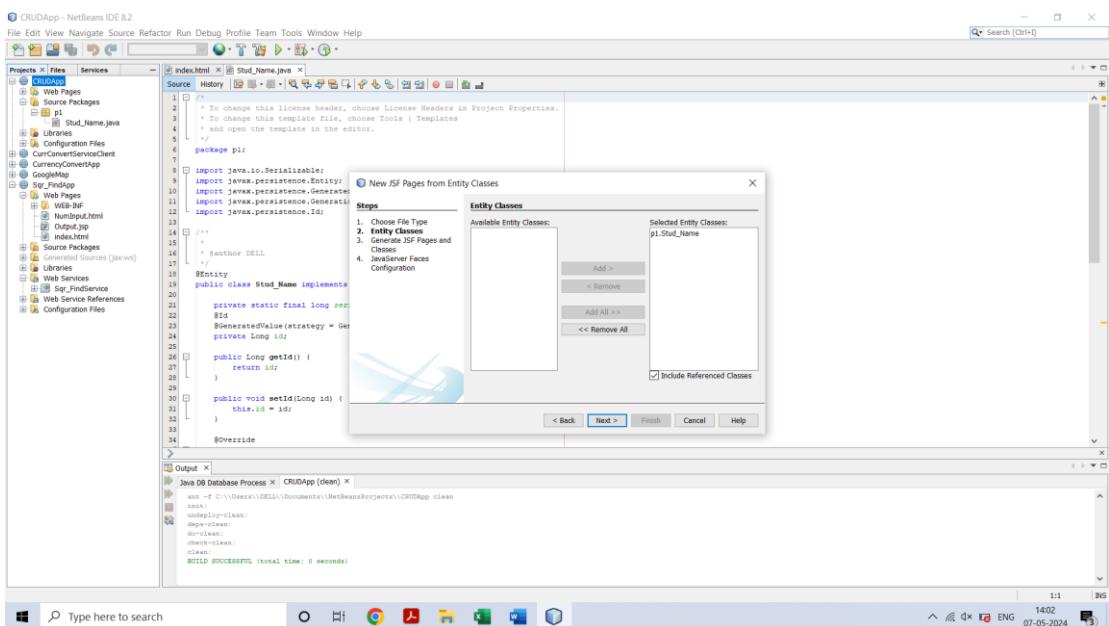
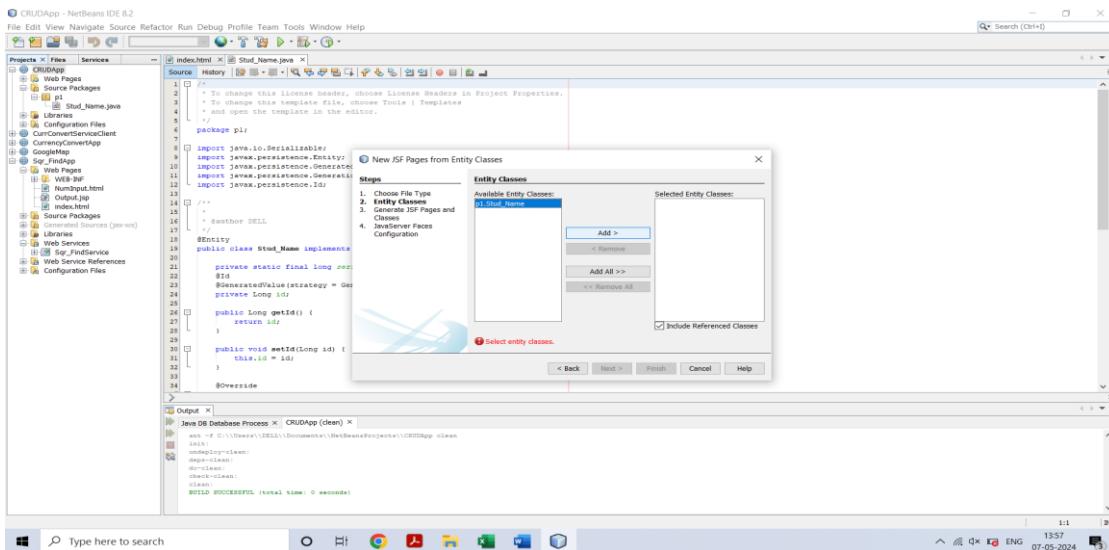
7. Click on finish.



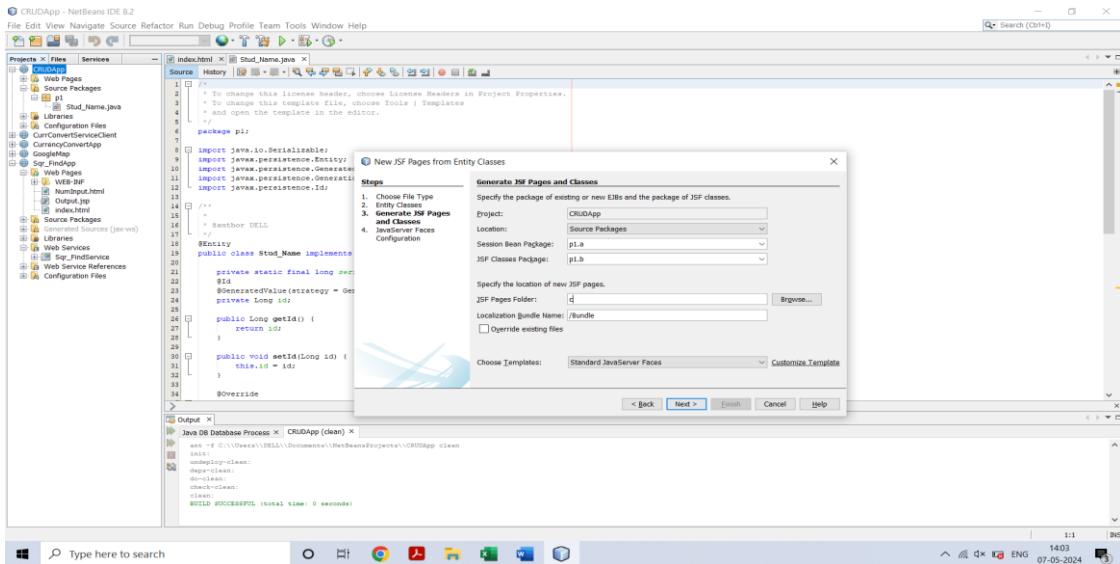


8. Right click on project name and create JSF Pages from Entity Classes.
9. Select **Add** and click on Add button and then Next button on below

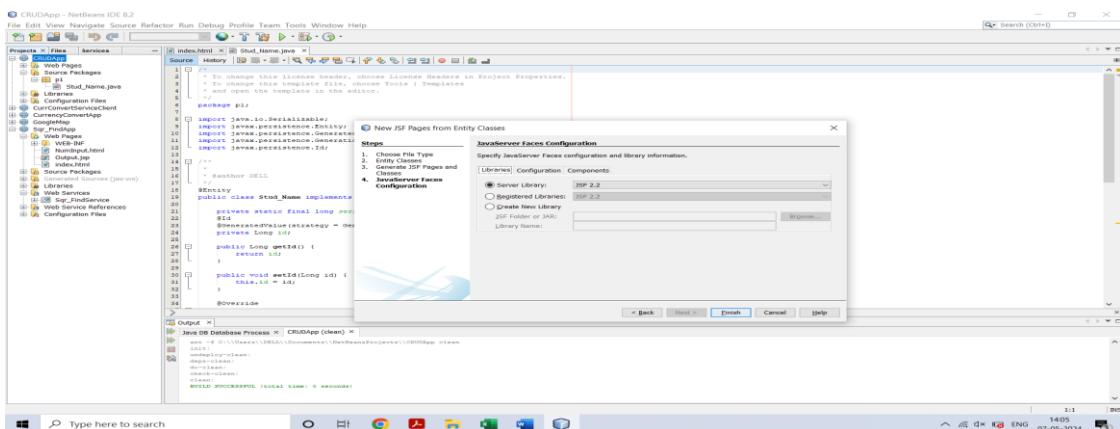




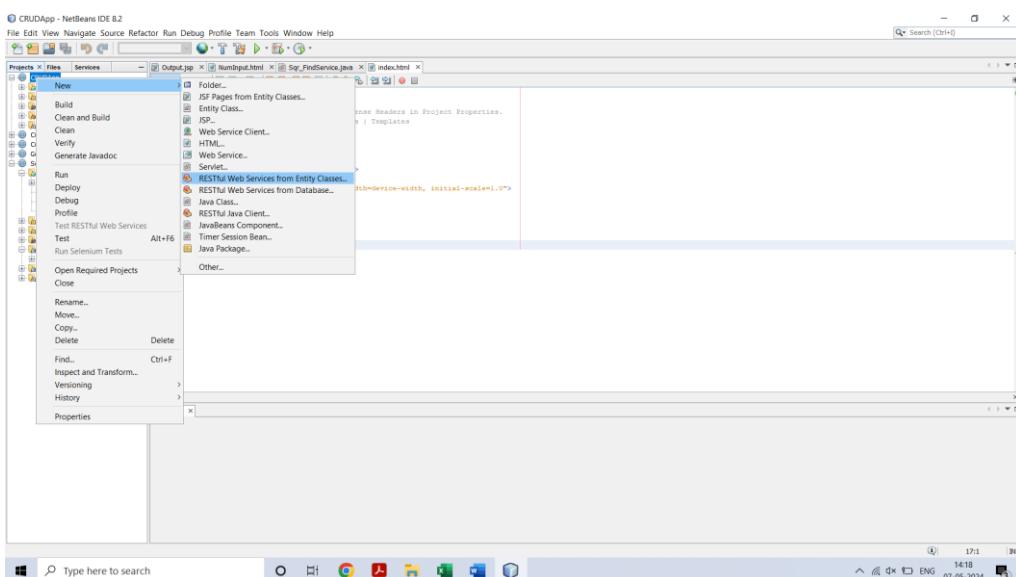
10. A window like below will appear on the screen. Enter the data into that window as entered in below pic and click on Next button.

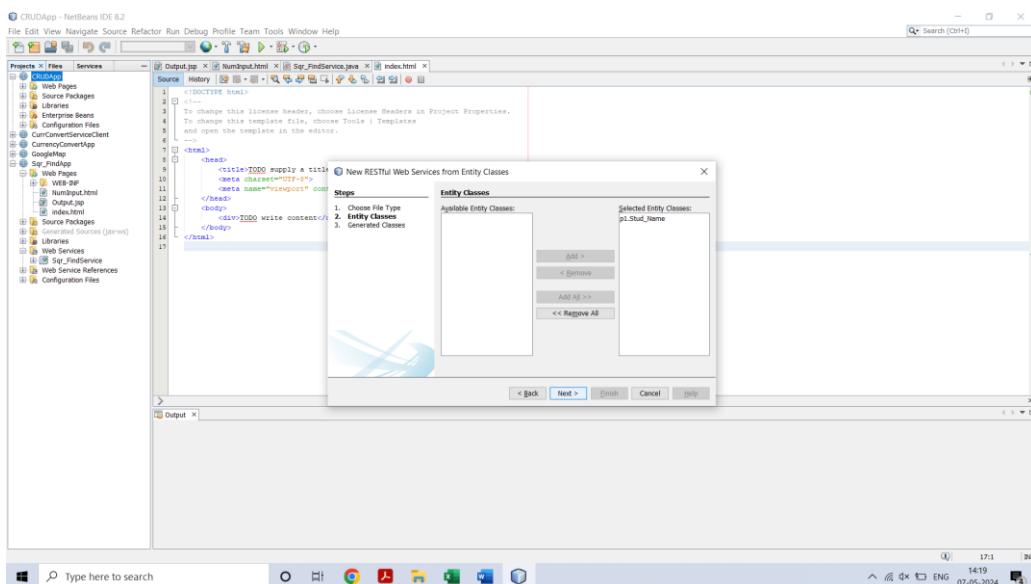
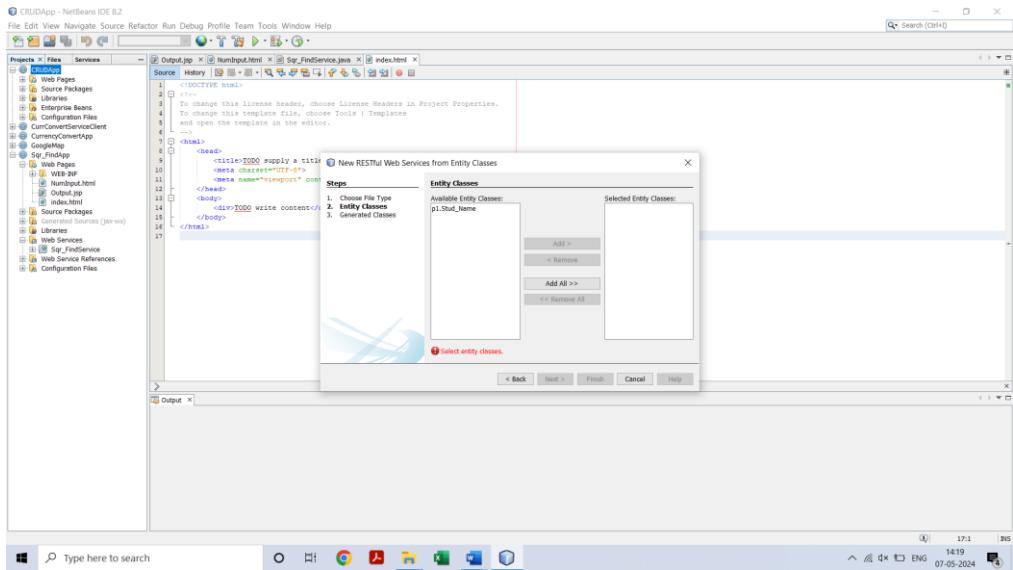


11. Now click on Finish.

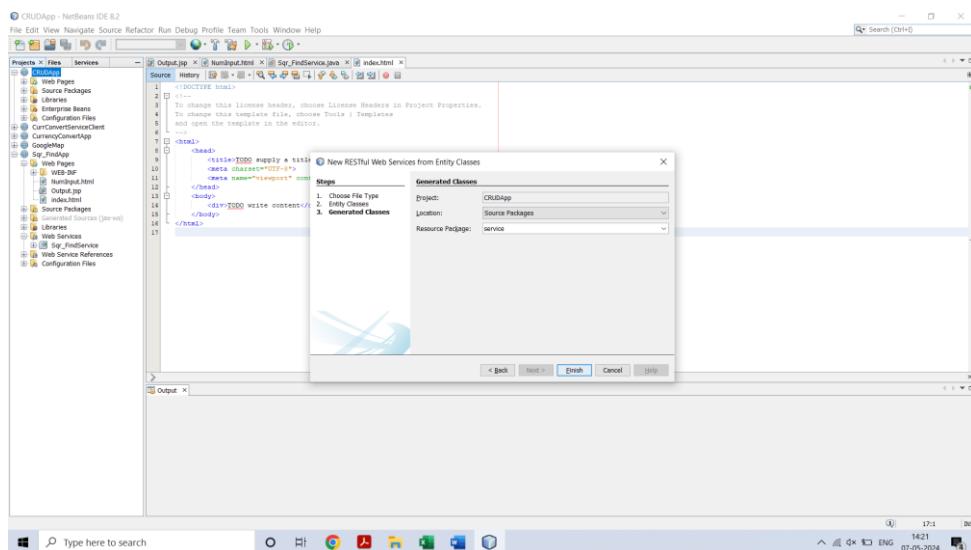


12. Right click on project name and create RESTful Web Services from Entity Classes.





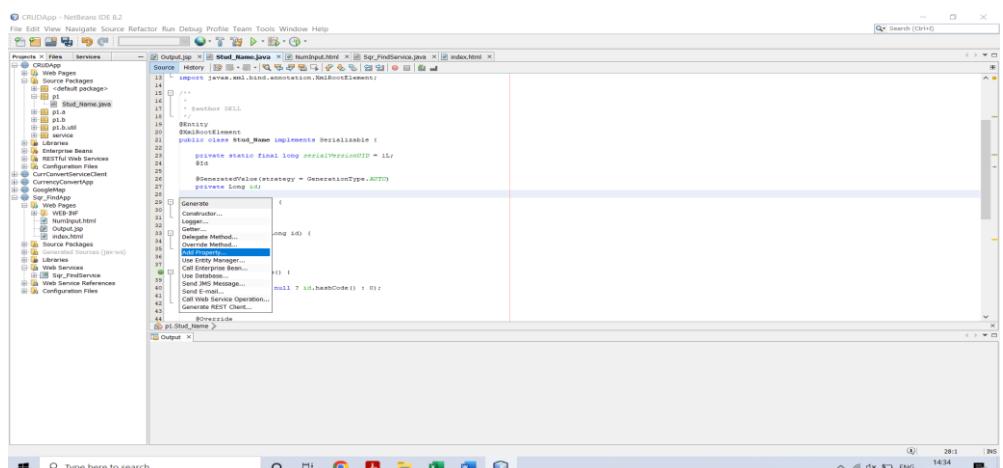
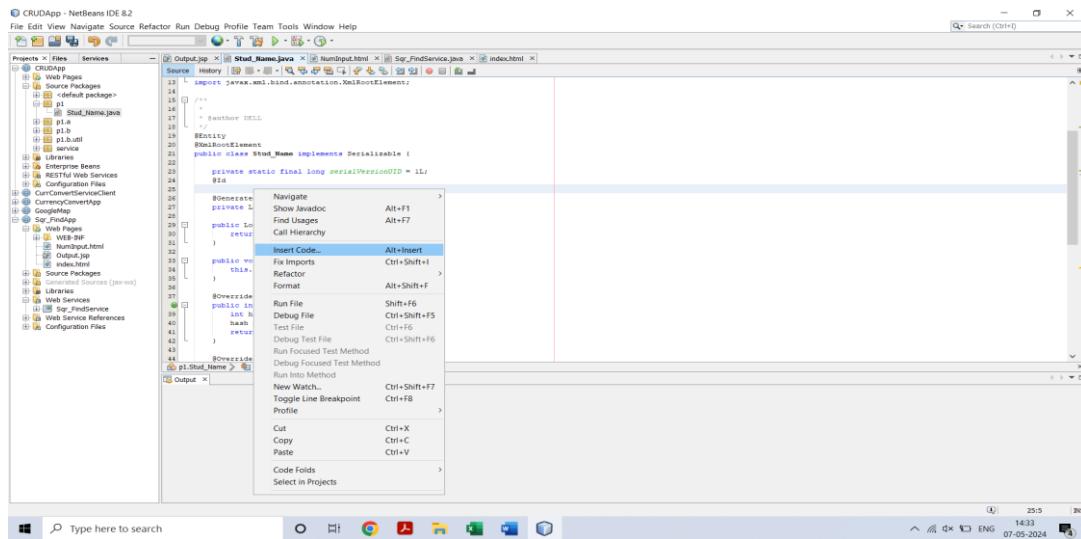
- 13.** Repeat step 9 and then it will go on next page. Then enter the **p1.service** in Resource Package and then click on Finish button.



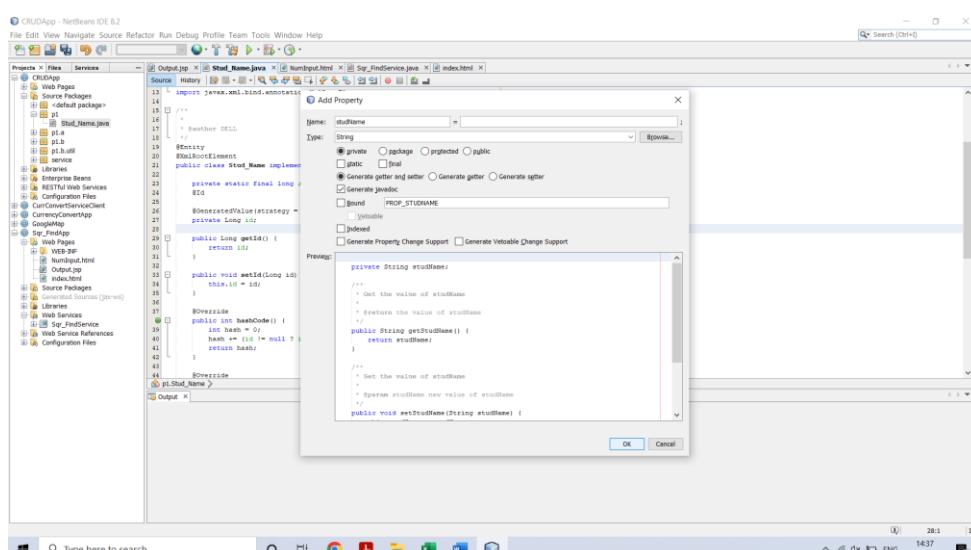
14. Now open Stud_Name.java file under p1 package.

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

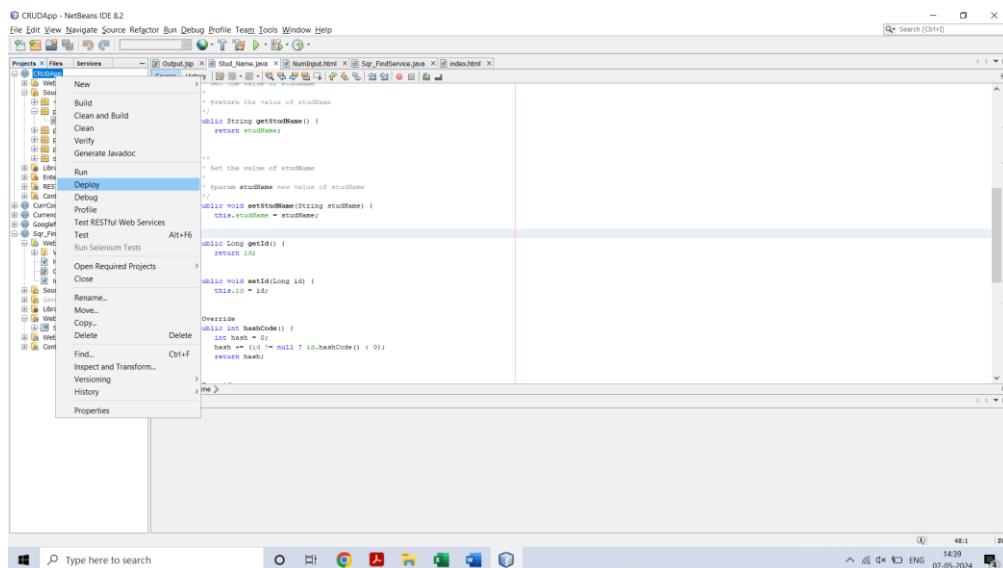
16. A new list will appear. Click on Add Property



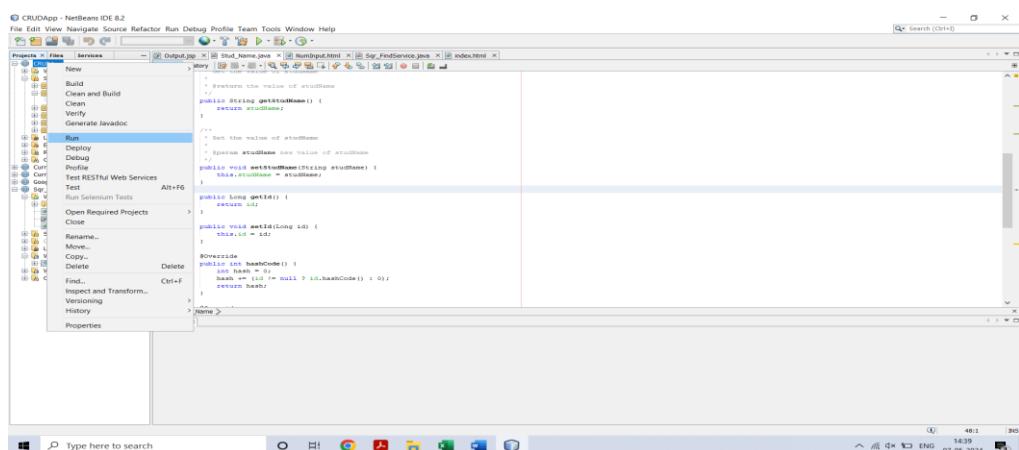
17. A new window will open. Enter name as studName. Make sure name should be exact same as of mine and then click on OK button. Actually we are setting getter and setter method for studName.



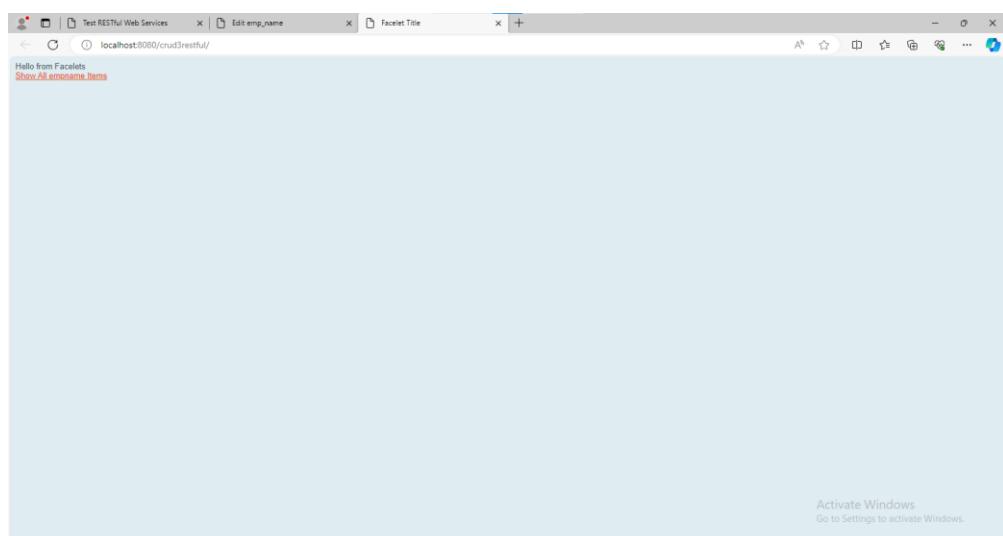
18. Now right click on web application name and Deploy it.

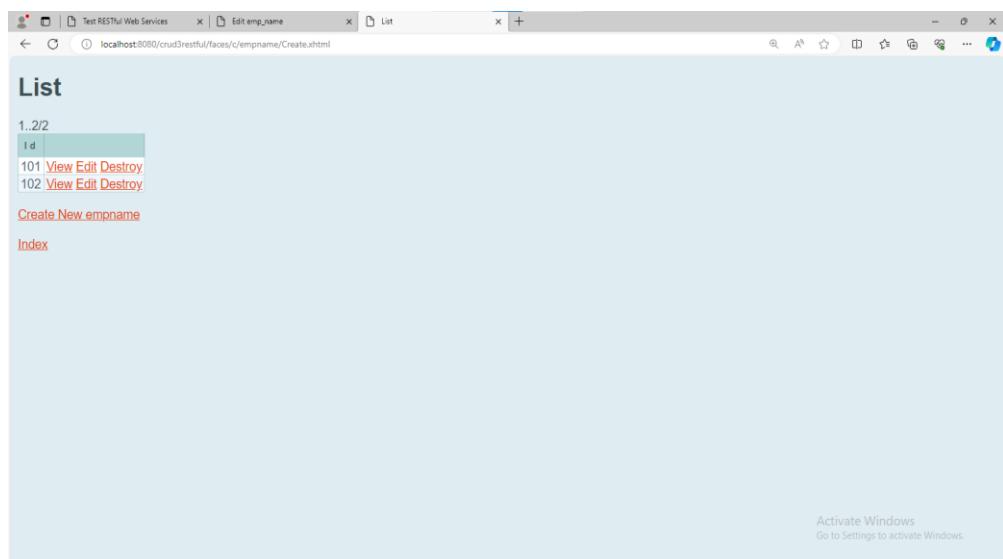


19. Now right click on project name and run it



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

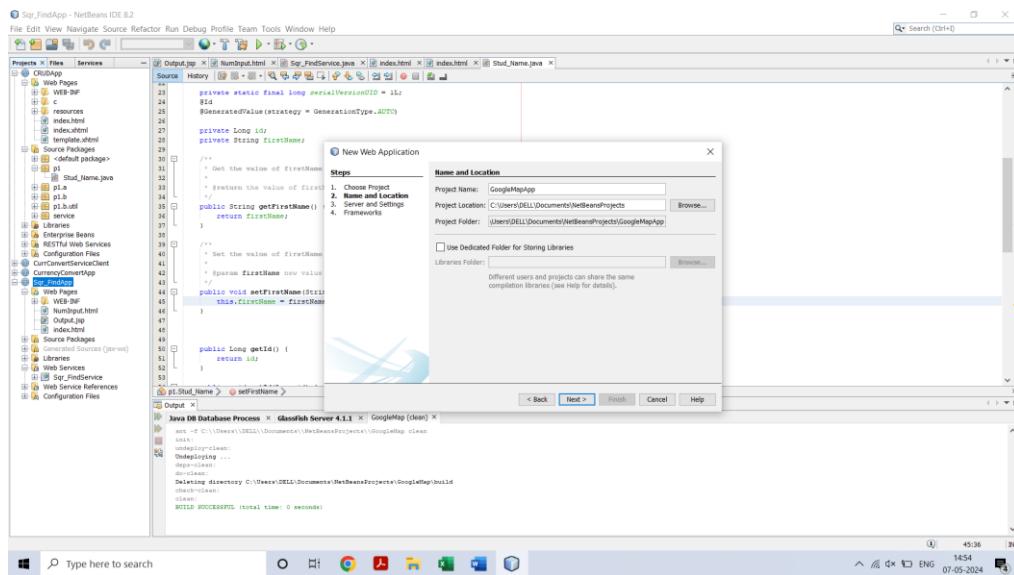




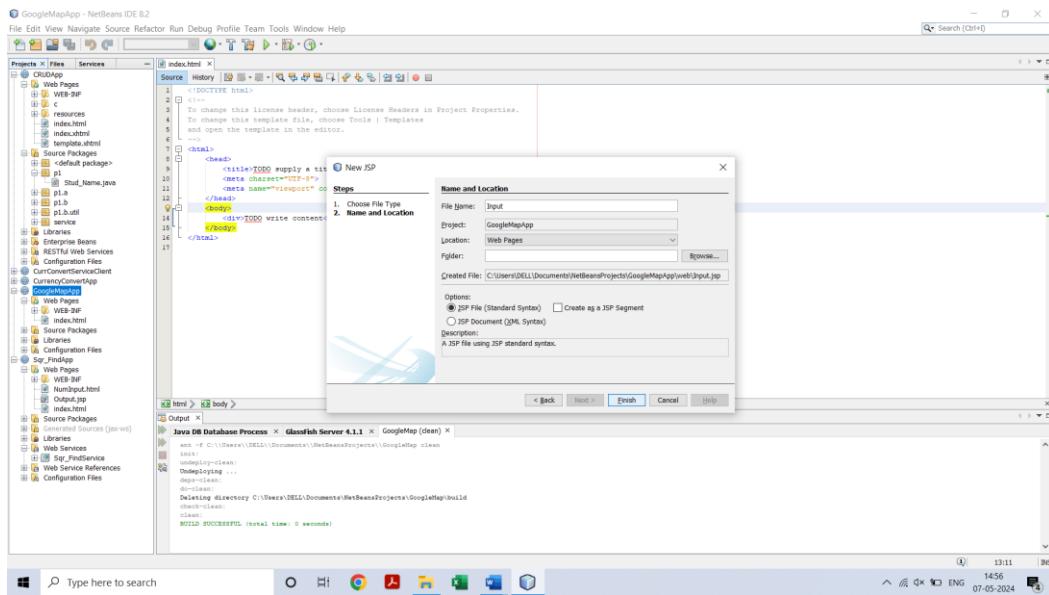
Practical-4

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

- First of all we need to create an Java Web Application with any name, let it be GoogleMap here using Netbeans IDE.



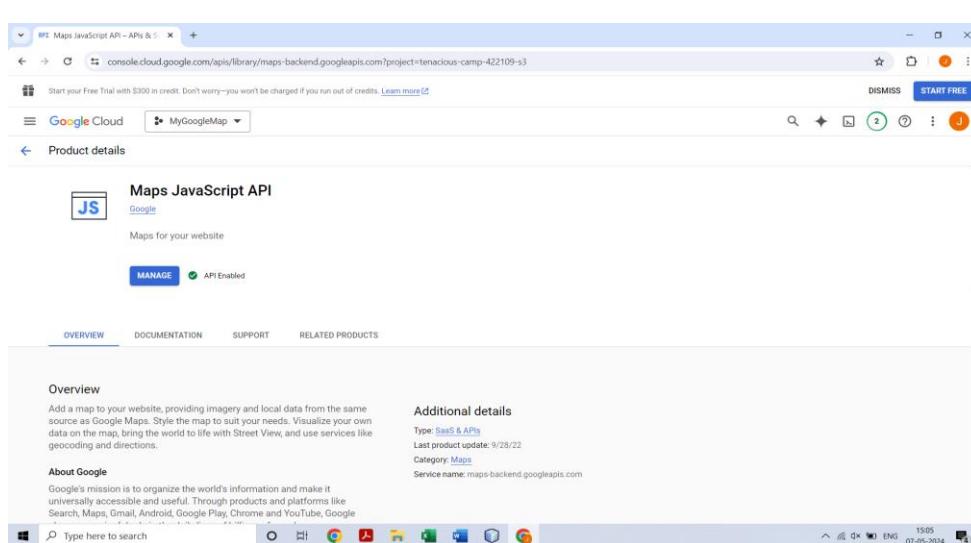
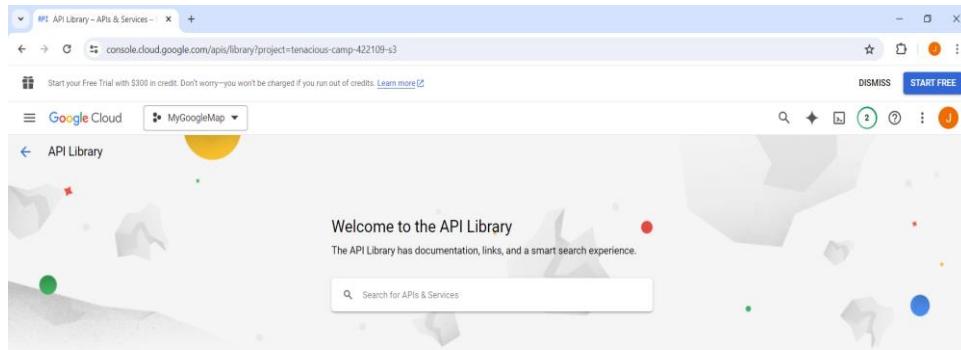
- Create a Input.jsp page to get Latitude and Longitude

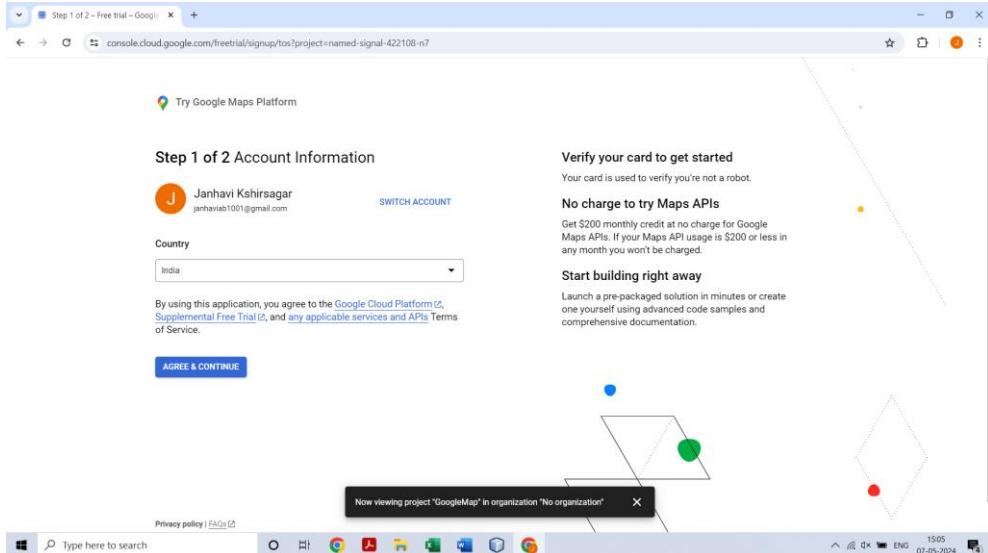


3. The following code is inserted in that Input.jsp page

```
<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>
```

4. 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).





5. Enable the Google Maps API V3

6. To Create Index.jsp page

```

<%@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>

```

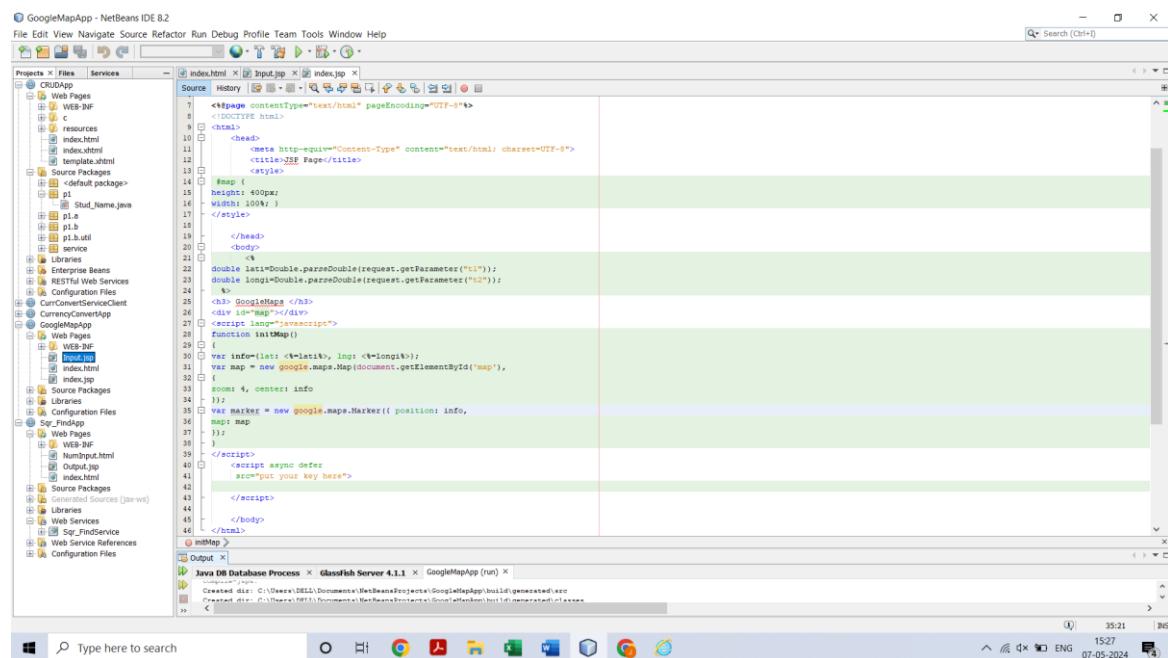
```

<script async defer
      src="put your key here">

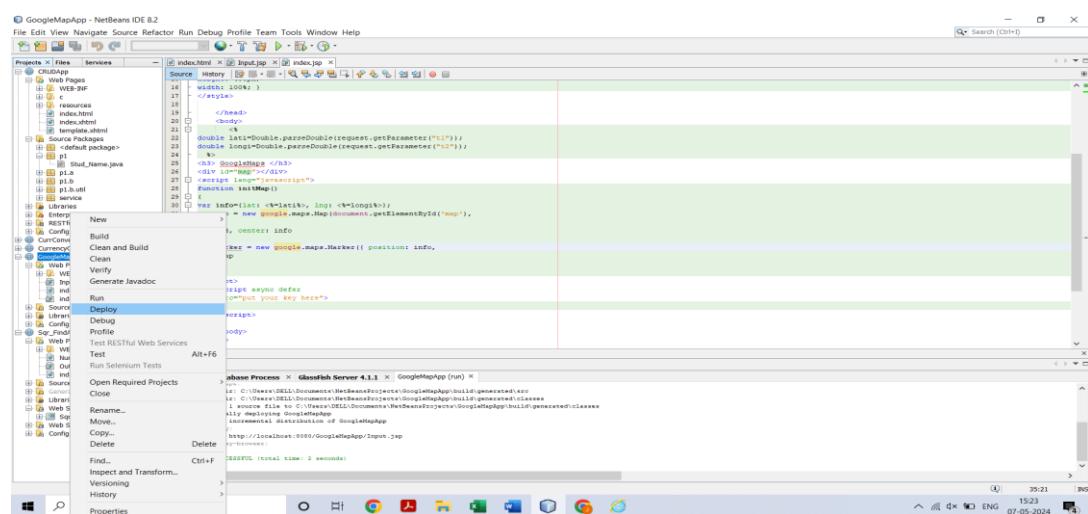
</script>
</body>
</html>

```

Index. Jsp



To run application :



GoogleMapApp - NetBeans IDE 8.2

File Edit View Navigate Source Refactor Profile Team Tools Window Help

Projects | Files | Services index.html | Input.jsp | index.jsp

```

<html>
<head>
    <style>
        width: 100%;
    </style>
</head>
<body>
    <div id="map"></div>
    <script language="javascript">
        function initMap() {
            var info = {lat: 19.0269, lng: 72.8553};
            var map = new google.maps.Map(document.getElementById('map'), {
                zoom: 4, center: info
            });
            var marker = new google.maps.Marker({ position: info,
                map: map
            });
        }
    </script>
    <pt async defer
        *put your key here*
    >
    <ips>
        <?>
    </ips>
</html>

```

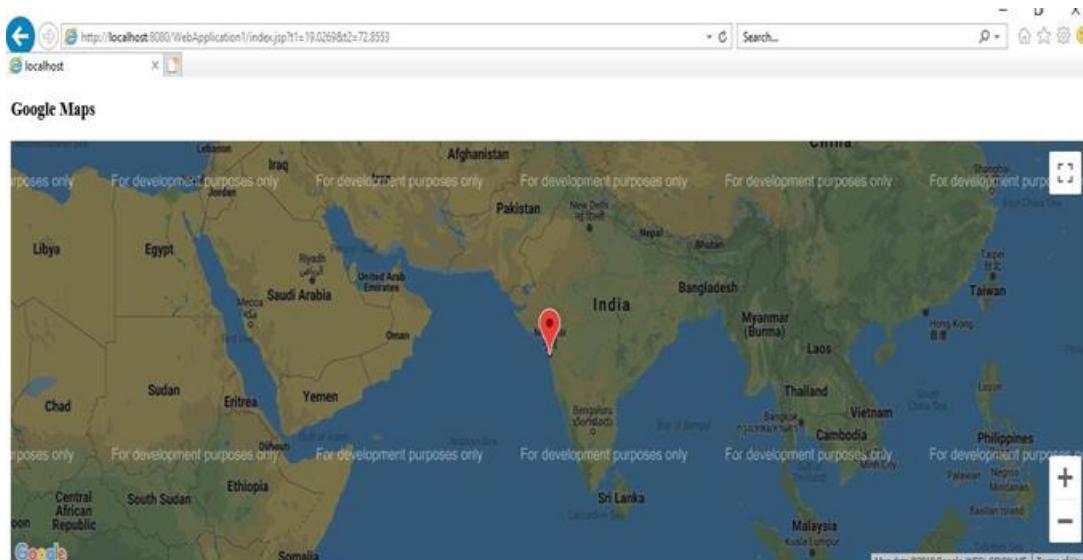
Open View Servlet Compile File F9 Run File Shift+F6 Debug File Ctrl+Shift+F5 Profile File History Cut Ctrl+X Copy Ctrl+C Paste Ctrl+V Add Delete Rename... Save As Template... Tools Properties Type here to search

Process: GlassFish Server 4.1.1 X GoogleMapApp (run) X

C:\Users\DELL\Documents\NetBeansProjects\GoogleMapApp\build\generated\ear C:\Users\DELL\Documents\NetBeansProjects\GoogleMapApp\build\generated\classes source file to C:\Users\DELL\Documents\NetBeansProjects\GoogleMapApp\build\generated\classes deploying GoogleMapApp install application to GoogleMapApp http://localhost:8080/GoogleMapApp/Input.jsp success! [total time: 2 seconds]

35:23 15:23 ENG 07-05-2024

Output :

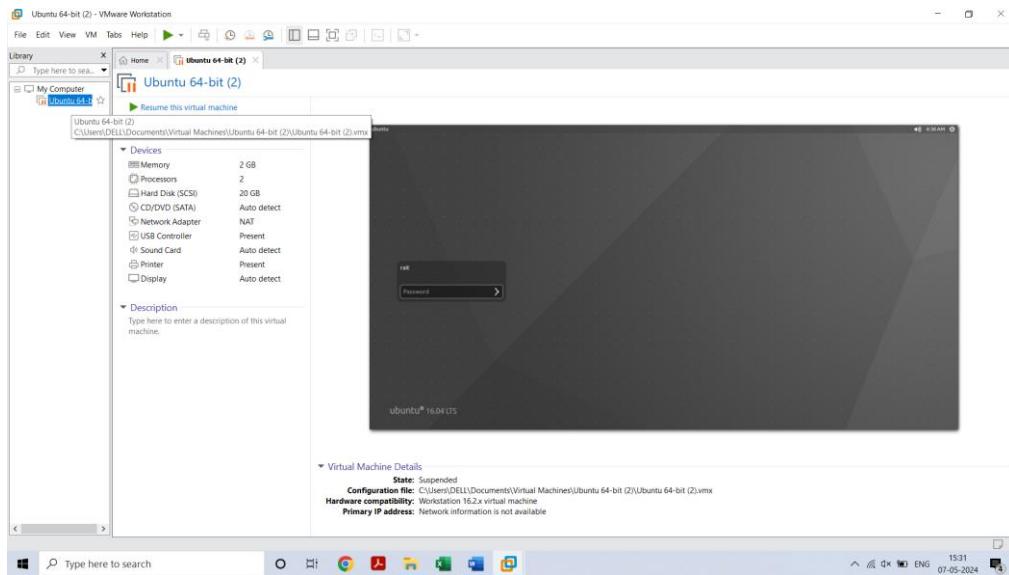


Practical-5

Installation and Configuration of virtualization using KVM

Installation Steps :

First we need to open VMware WorkStation



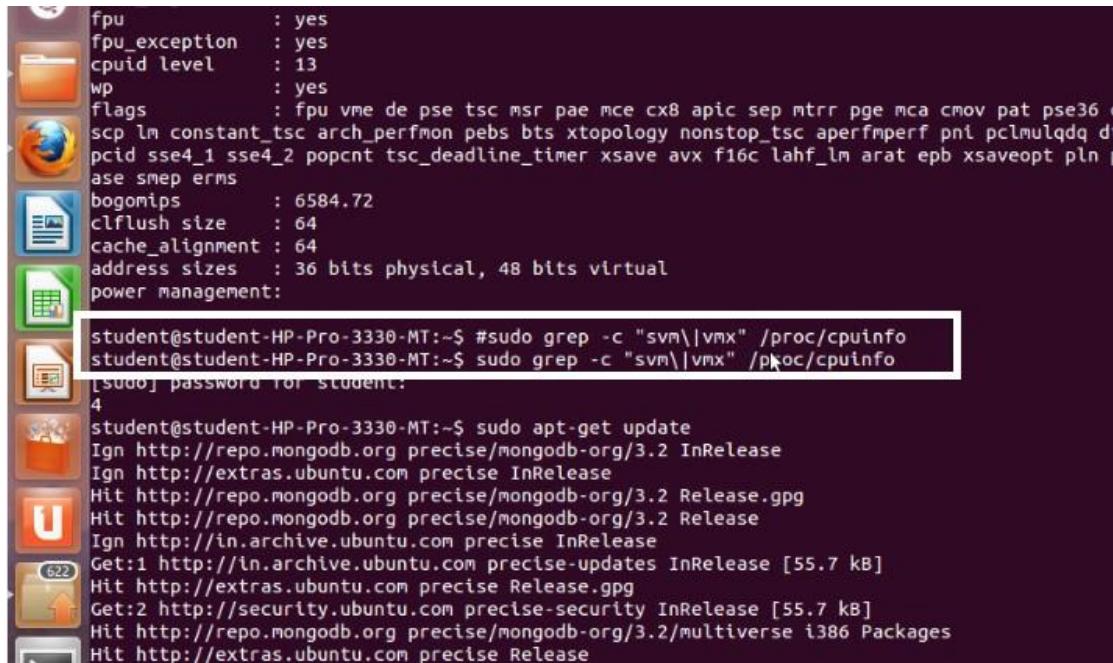
For the installation following commands are to be executed:

Step 1 : #sudo grep -c "svm\|vmx" /proc/cpuinfo

```
student@student-HP-Pro-3330-MT:~$ sudo grep -c "svm\|vmx" /proc/cpuinfo
student@student-HP-Pro-3330-MT:~$ clear

student@student-HP-Pro-3330-MT:~$ cat /proc/cpuinfo
processor       : 0
vendor_id      : GenuineIntel
cpu family     : 6
model          : 30
model name     : Intel(R) Core(TM) i3-3220 CPU @ 3.30GHz
stepping        : 9
microcode      : 0x19
cpu MHz        : 1600.000
cache size     : 3072 KB
physical id    : 0
siblings        : 4
core id         : 0
cpu cores      : 2
```

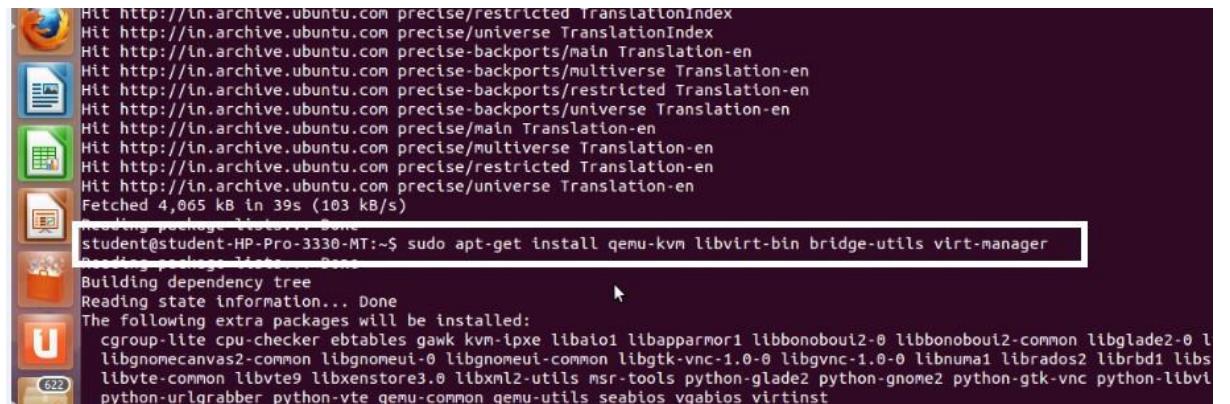
Step 2 : #sudo apt-get install qemu-kvm libvirt-bin bridge-utils virt-manager



```
fpu      : yes
fpu_exception : yes
cpuid level   : 13
wp       : yes
flags      : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36
scp lm constant_tsc arch_perfmon pebs bts xtopology nonstop_tsc aperfmpf perf pni pclmuldq d
pcid sse4_1 sse4_2 popcnt tsc_deadline_timer xsave avx f16c lahf_lm arat epb xsaveopt pln
ase smep erms
bogomips   : 6584.72
clflush size  : 64
cache_alignment : 64
address sizes : 36 bits physical, 48 bits virtual
power management:

student@student-HP-Pro-3330-MT:~$ sudo grep -c "svm\|vmx" /proc/cpuinfo
student@student-HP-Pro-3330-MT:~$ sudo grep -c "svm\|vmx" /proc/cpuinfo
[sudo] password for student:
4
student@student-HP-Pro-3330-MT:~$ sudo apt-get update
Ign http://repo.mongodb.org precise/mongodb-org/3.2 InRelease
Ign http://extras.ubuntu.com precise InRelease
Hit http://repo.mongodb.org precise/mongodb-org/3.2 Release.gpg
Hit http://repo.mongodb.org precise/mongodb-org/3.2 Release
Ign http://in.archive.ubuntu.com precise InRelease
Get:1 http://in.archive.ubuntu.com precise-updates InRelease [55.7 kB]
Hit http://extras.ubuntu.com precise Release.gpg
Get:2 http://security.ubuntu.com precise-security InRelease [55.7 kB]
Hit http://repo.mongodb.org precise/mongodb-org/3.2/multiverse i386 Packages
Hit http://extras.ubuntu.com precise Release
```

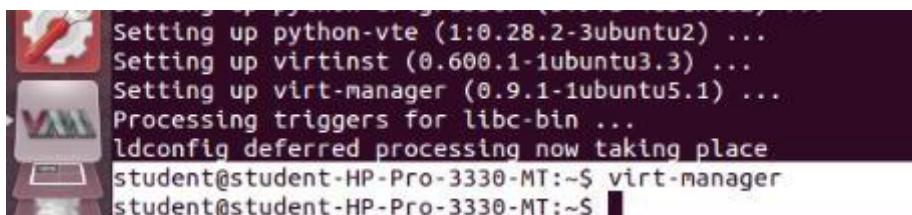
Step 3 : #sudo adduser rait



```
Hit http://in.archive.ubuntu.com precise/restricted TranslationIndex
Hit http://in.archive.ubuntu.com precise/universe TranslationIndex
Hit http://in.archive.ubuntu.com precise-backports/main Translation-en
Hit http://in.archive.ubuntu.com precise-backports/multiverse Translation-en
Hit http://in.archive.ubuntu.com precise-backports/restricted Translation-en
Hit http://in.archive.ubuntu.com precise-backports/universe Translation-en
Hit http://in.archive.ubuntu.com precise/main Translation-en
Hit http://in.archive.ubuntu.com precise/multiverse Translation-en
Hit http://in.archive.ubuntu.com precise/restricted Translation-en
Hit http://in.archive.ubuntu.com precise/universe Translation-en
Fetched 4,065 kB in 39s (103 kB/s)
Reading package lists...
student@student-HP-Pro-3330-MT:~$ sudo apt-get install qemu-kvm libvirt-bin bridge-utils virt-manager
[sudo] password for student:
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  cgroup-lite cpu-checker ebtables gawk kvm-ipxe libaio1 libapparmor1 libbonoboui2-0 libbonoboui2-common libglade2-0 libgnomecanvas2-common libgnomeui-0 libgnomeui-common libgtk-vnc-1.0-0 libgvnc-1.0-0 libnuma1 librados2 librbd1 libs
libvte-common libvte9 libxenstore3.0 libxml2-utils msr-tools python-glade2 python-gnome2 python-gtk-vnc python-libvi
python-urlgrabber python-vte qemu-common qemu-utils seabios vgabios virtinst
```

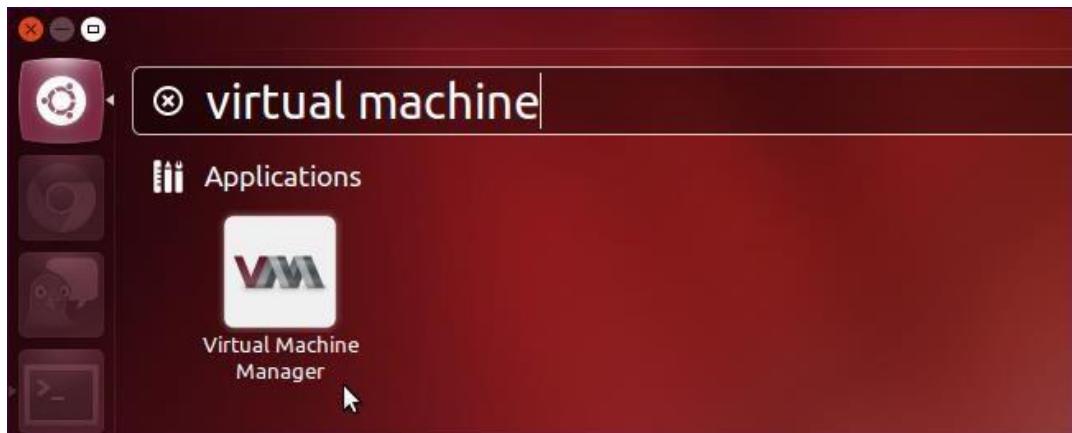
Step 4 : #sudo adduser rait libvirt

After running this command, log out and log back in as rait

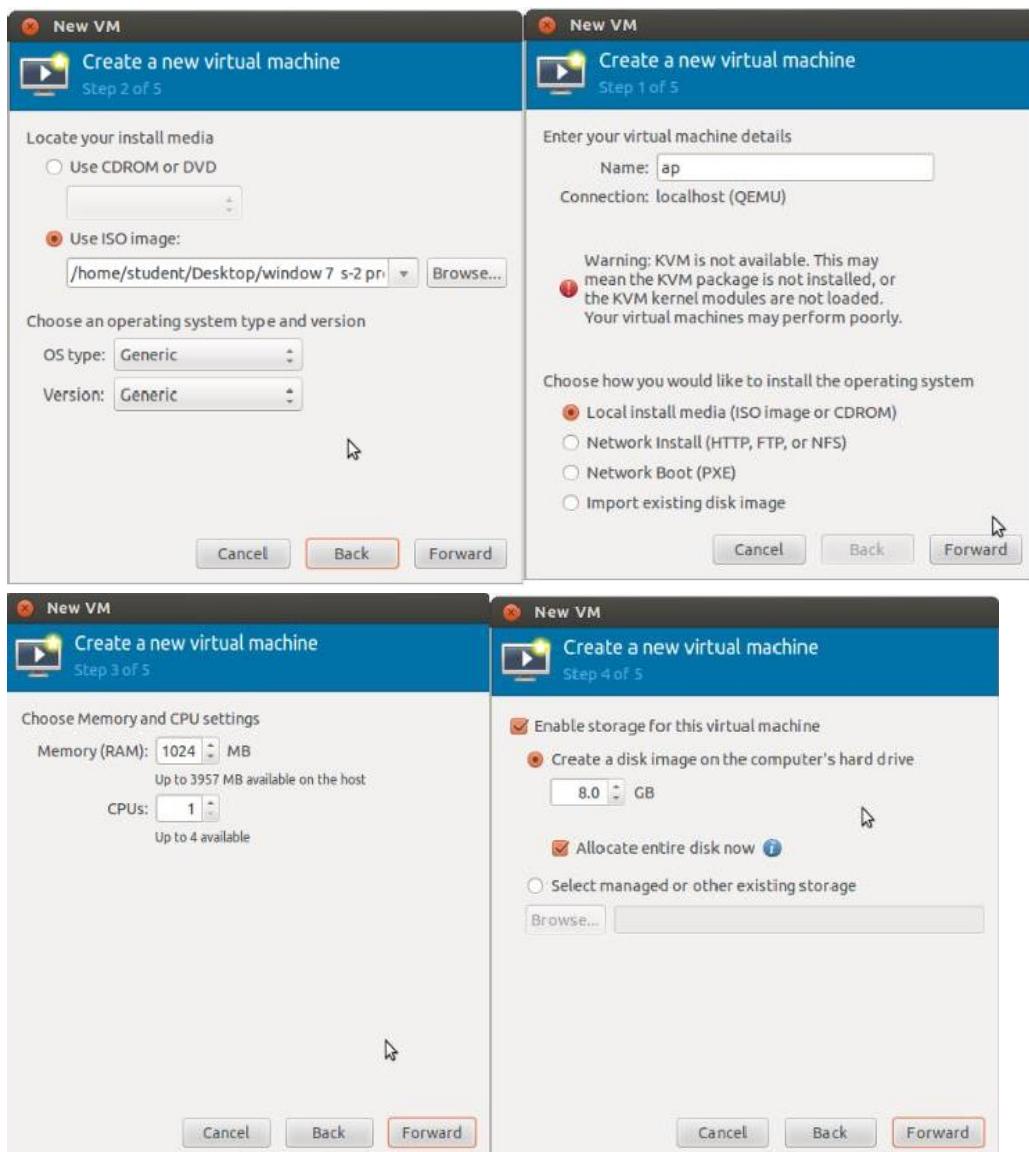


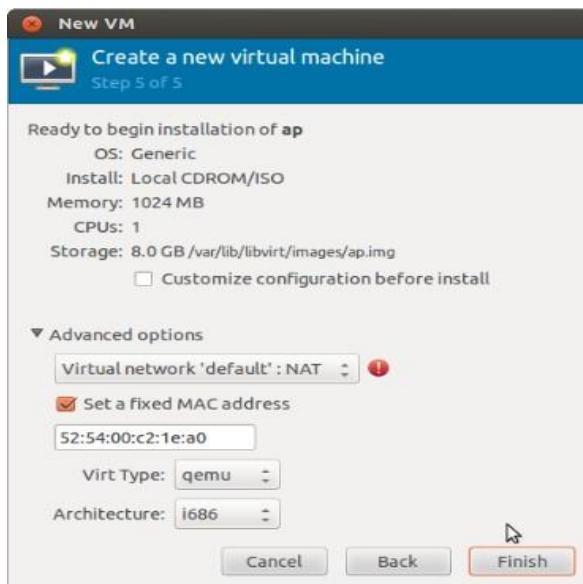
```
Setting up python-vte (1:0.28.2-3ubuntu2) ...
Setting up virtinst (0.600.1-1ubuntu3.3) ...
Setting up virt-manager (0.9.1-1ubuntu5.1) ...
Processing triggers for libc-bin ...
ldconfig deferred processing now taking place
student@student-HP-Pro-3330-MT:~$ virt-manager
student@student-HP-Pro-3330-MT:~$
```

Step 5 : Open Virtual Machine Manager application and Create Virtual Machine #virt-manager as shown below

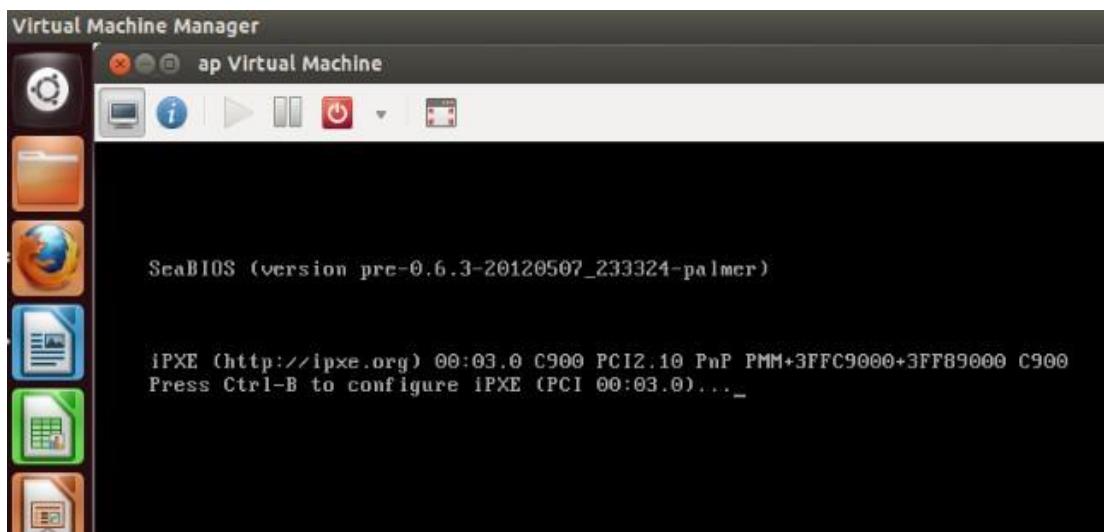


Step 6 : Create a new virtual machine as shown below

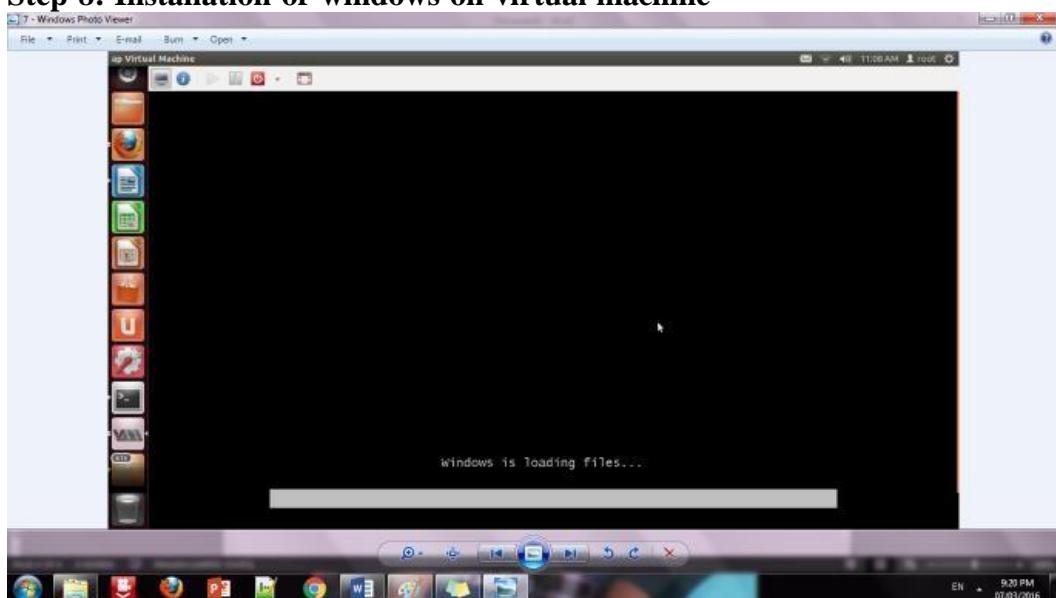




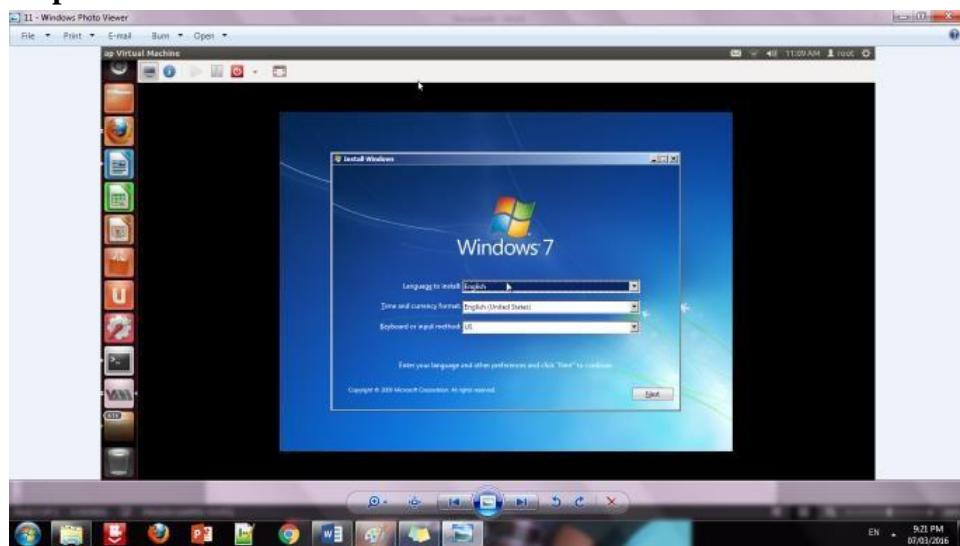
Step 7 : Install windows operating system on virtual machine



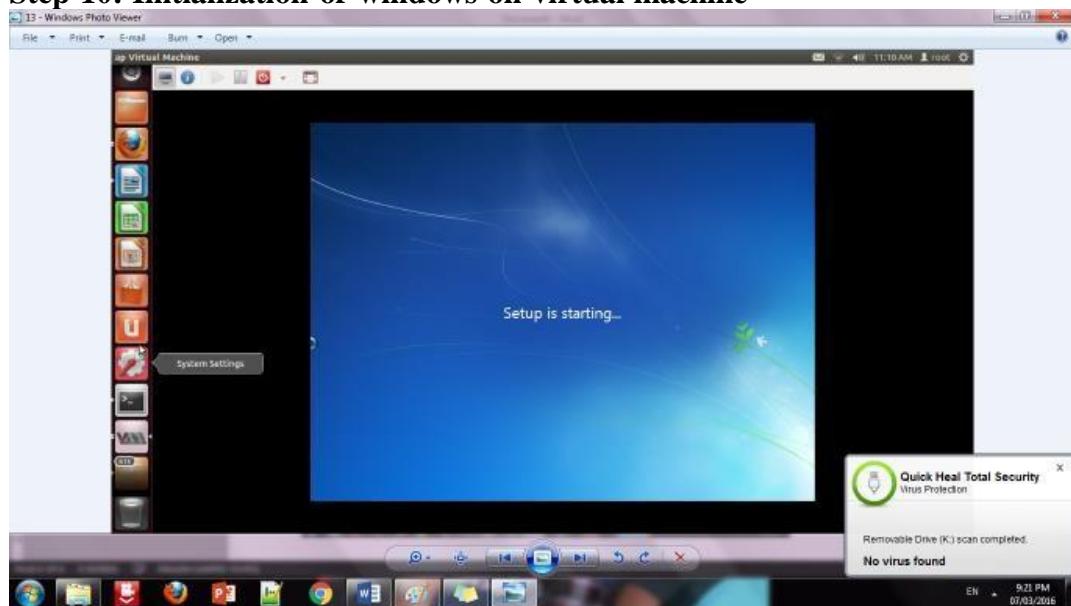
Step 8: Installation of windows on virtual machine



Step 9: Installation of windows 7 on virtual machine



Step 10: Initialization of windows on virtual machine

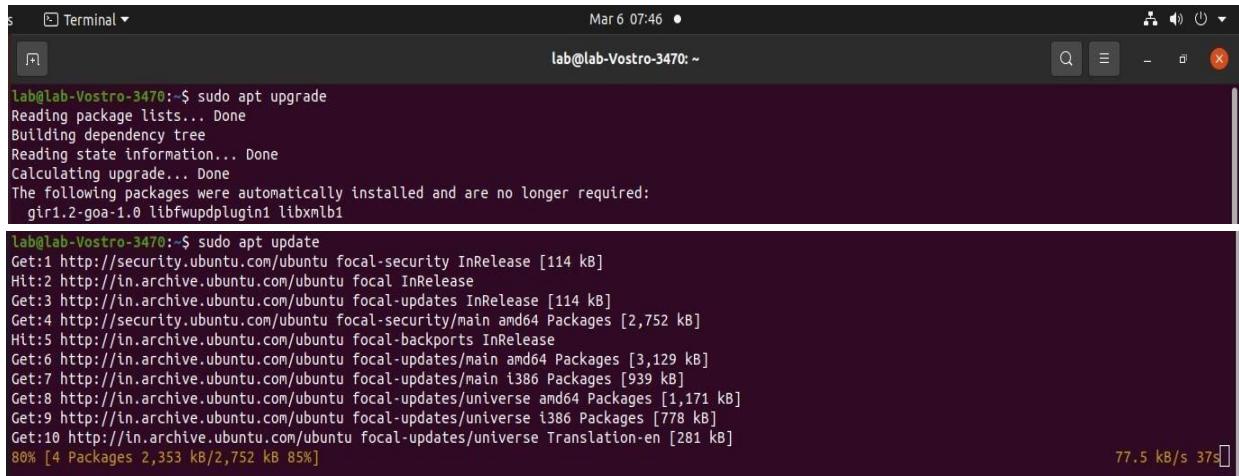


Practical-7

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

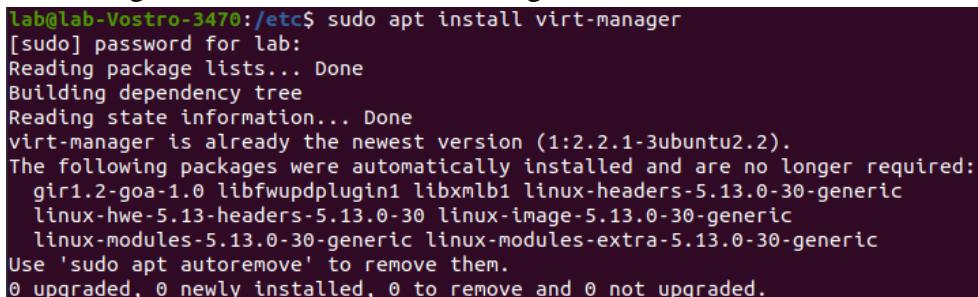
Steps:

1. Open terminal and enter the following Command “sudo apt upgrade” and “sudo apt update”



```
lab@lab-Vostro-3470:~$ sudo apt upgrade
Reading package lists... Done
Building dependency tree
Reading state information... Done
Calculating upgrade... Done
The following packages were automatically installed and are no longer required:
  gir1.2-goa-1.0 libfwupdplugin1 libxml2
lab@lab-Vostro-3470:~$ sudo apt update
Get:1 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Hit:2 http://in.archive.ubuntu.com/ubuntu focal InRelease
Get:3 http://in.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:4 http://in.archive.ubuntu.com/ubuntu focal-security/main amd64 Packages [2,752 kB]
Hit:5 http://in.archive.ubuntu.com/ubuntu focal-backports InRelease
Get:6 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [3,129 kB]
Get:7 http://in.archive.ubuntu.com/ubuntu focal-updates/main i386 Packages [939 kB]
Get:8 http://in.archive.ubuntu.com/ubuntu focal-updates/universe amd64 Packages [1,171 kB]
Get:9 http://in.archive.ubuntu.com/ubuntu focal-updates/universe i386 Packages [778 kB]
Get:10 http://in.archive.ubuntu.com/ubuntu focal-updates/universe Translation-en [281 kB]
80% [4 Packages 2,353 kB/2,752 kB 85%]
77.5 kB/s 37s
```

2. Enter the following command to install virt manager

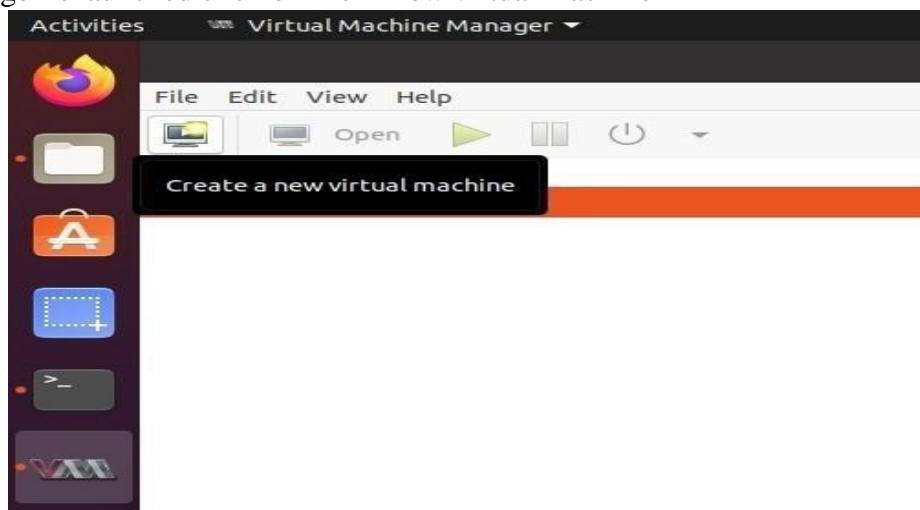


```
lab@lab-Vostro-3470:/etc$ sudo apt install virt-manager
[sudo] password for lab:
Reading package lists... Done
Building dependency tree
Reading state information... Done
virt-manager is already the newest version (1:2.2.1-3ubuntu2.2).
The following packages were automatically installed and are no longer required:
  gir1.2-goa-1.0 libfwupdplugin1 libxml2 linux-headers-5.13.0-30-generic
  linux-hwe-5.13-headers-5.13.0-30 linux-image-5.13.0-30-generic
  linux-modules-5.13.0-30-generic linux-modules-extra-5.13.0-30-generic
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
```

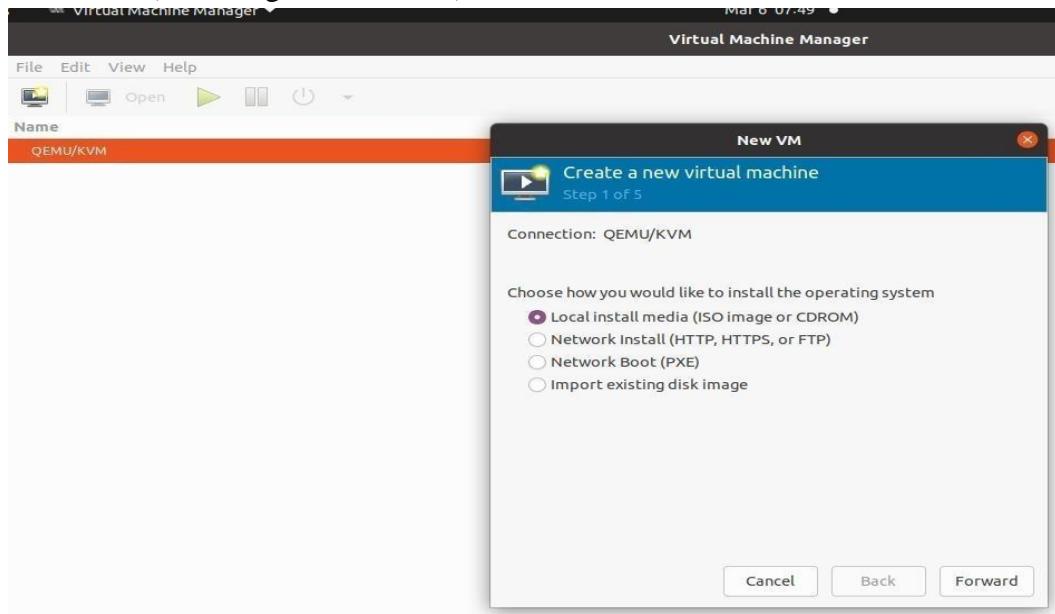
3. Launch Virt Manager using Following Command:

\$ Sudo virt-manager

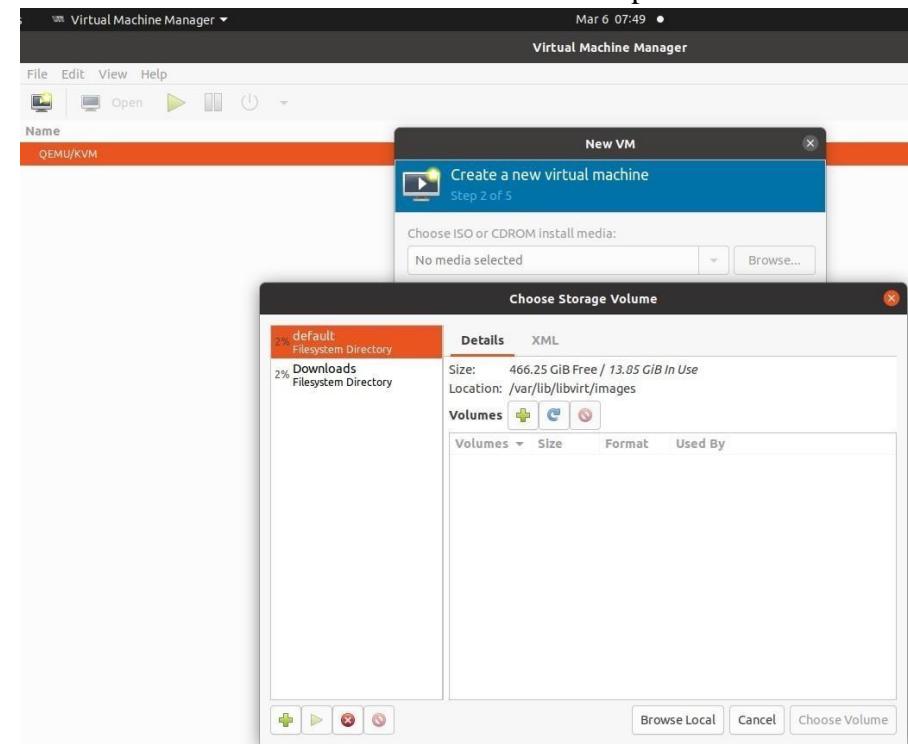
4. Once the virt manager is launched click on file → new virtual machine



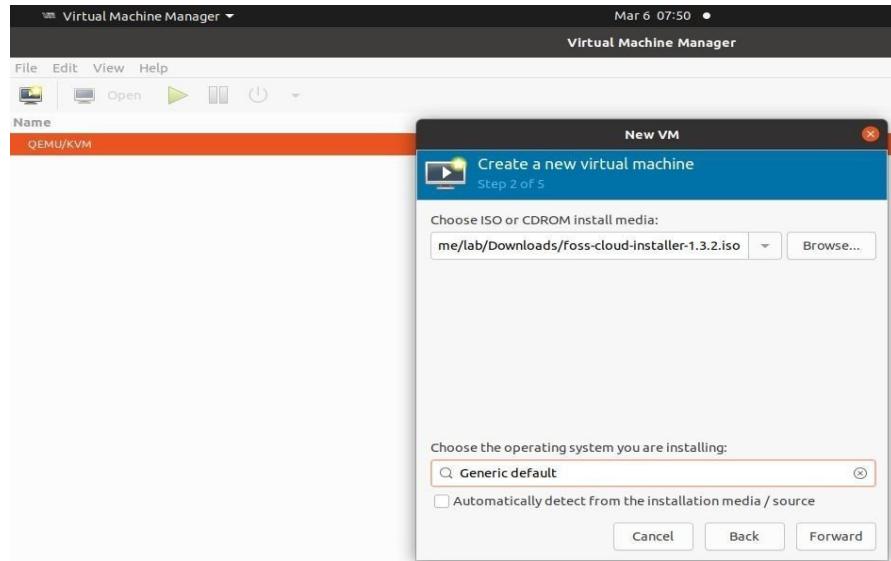
Select local install media (ISO image or CDROM) and click Forward



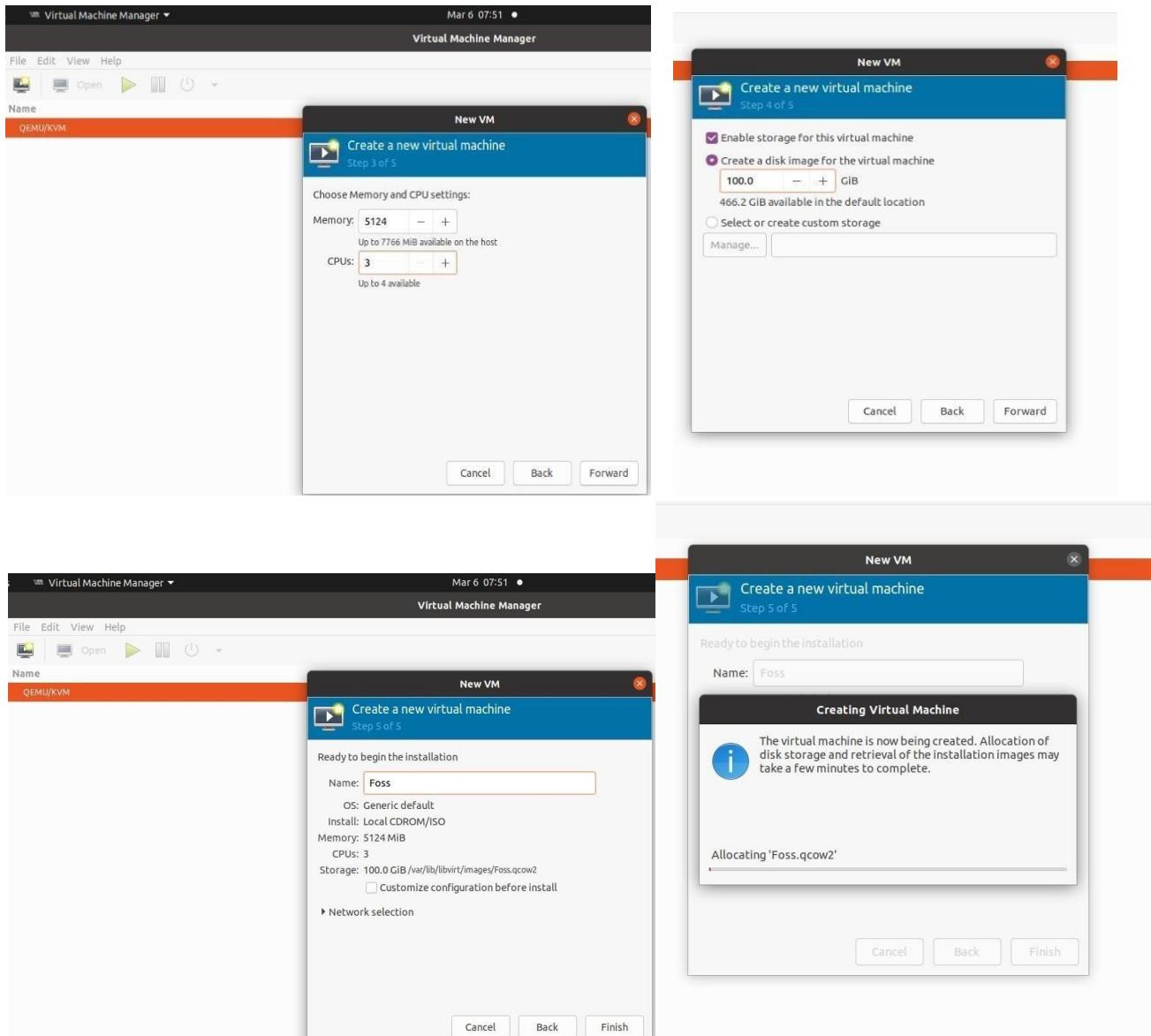
Click on Browse → Browse local → select downloaded iso file → Open



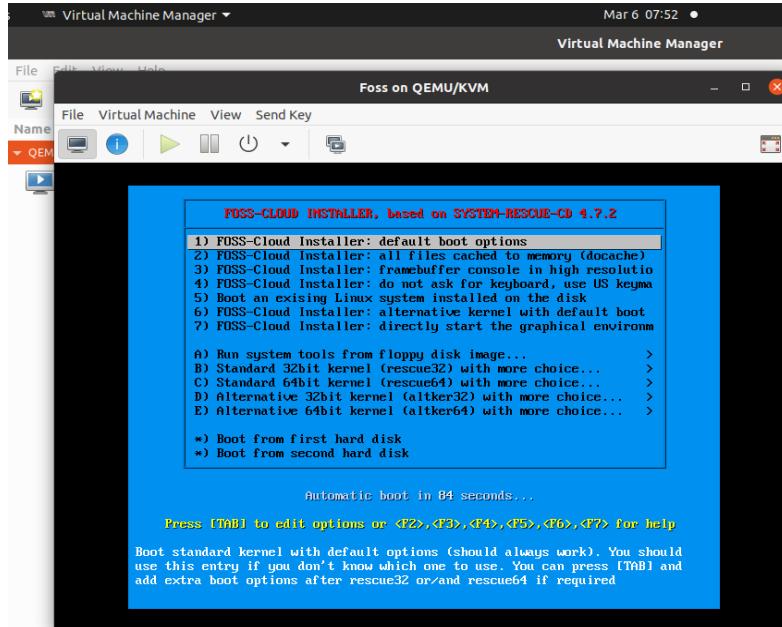
Uncheck The checkbox and Choose the OS you are installing as “Generic default” and click “Forward”



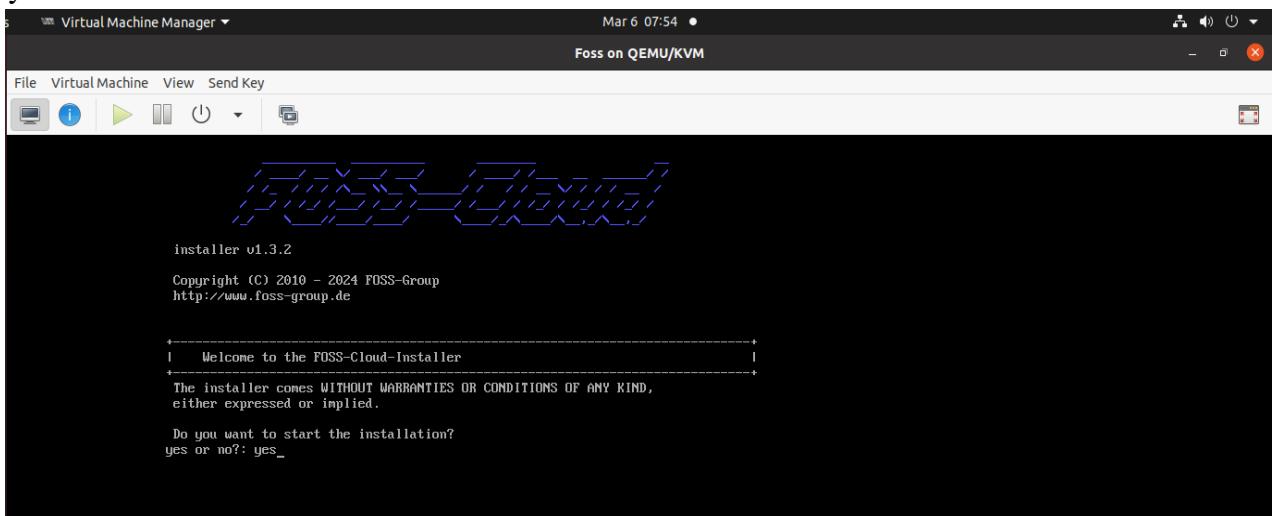
5. Provide the Memory, CPU, Storage, Name For Virtual Machine and click “Forward”



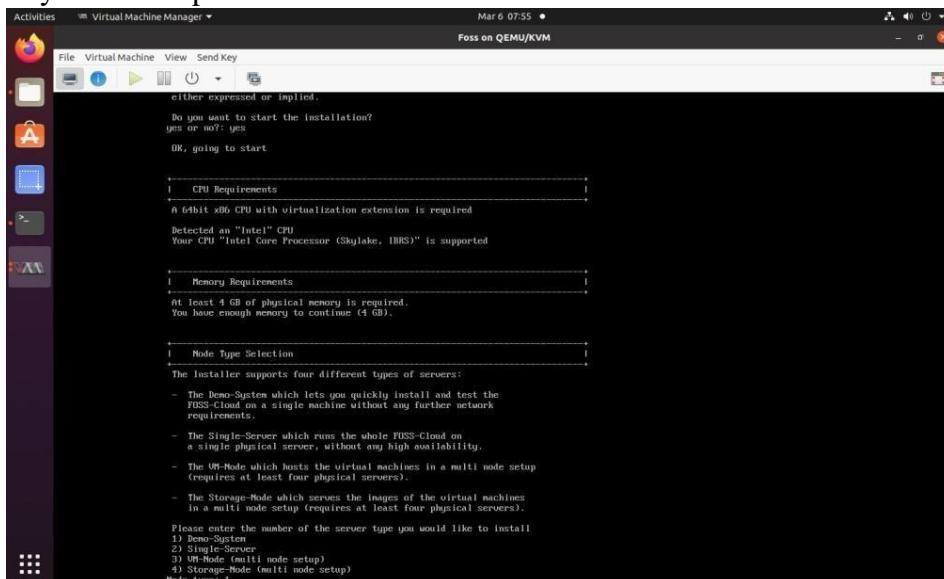
6. Once the foss Cloud is launched select “Foss-Cloud installer :default boot options” and press Enter



Types yes to start the installation

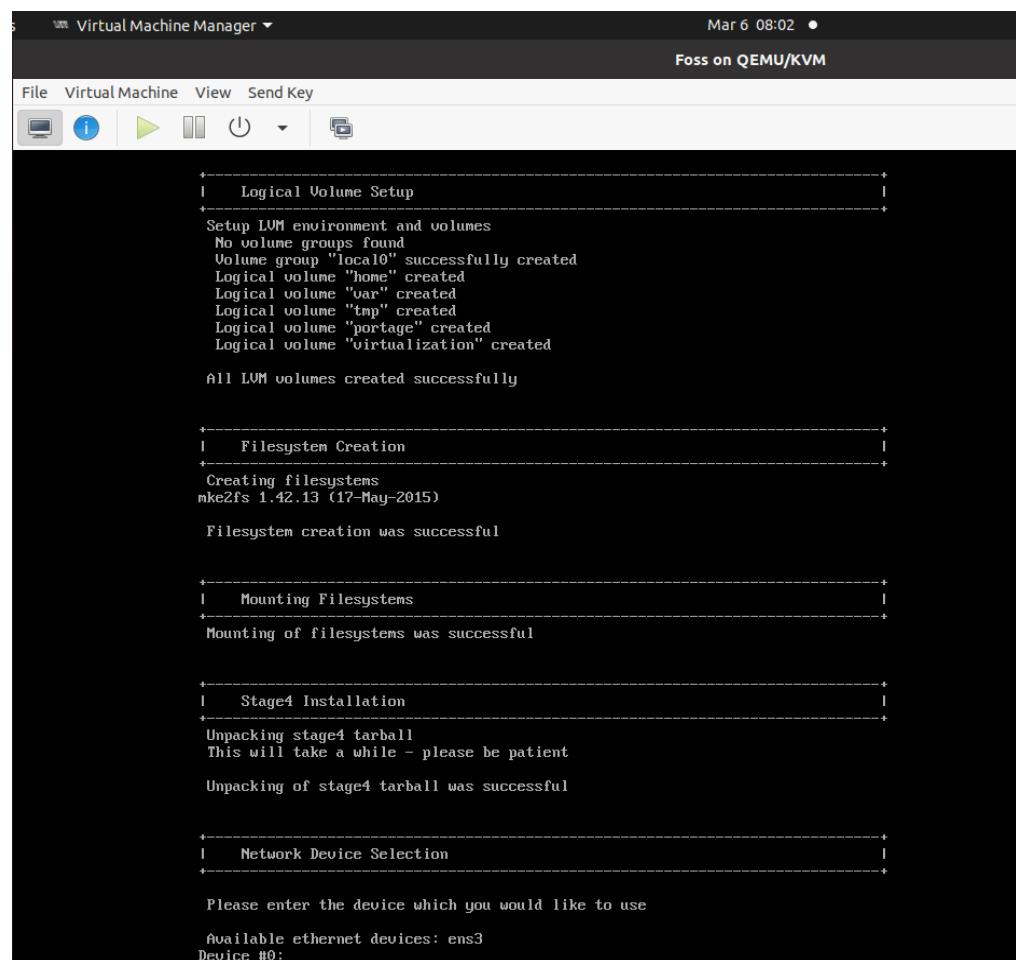


7. Select “Demo System” and press Enter



Enter the Device name as “sda” and press enter

```
+-----+  
| Installation Device Selection |  
+-----+  
A dedicated SCSI, SATA or PATA disk is required for the installation  
The disk has to be at least 130 GB in size  
  
Found sda (150 GB). Size is OK  
  
Below you will find a list of all detected and supported disks  
sda (150 GB)  
  
Please enter the device name on which you would like to install  
Device: sda  
'sda' will be used as the installation device.  
  
+-----+  
| Logical Volume Cleanup and Preparation |  
+-----+  
Checking for existing volume groups and physical volumes  
No volume groups found  
No volume groups found  
  
+-----+  
| Installation Device Partitioning |  
+-----+  
Below is the existing partition layout of your selected device
```



The screenshot shows a terminal window within the Virtual Machine Manager application. The title bar indicates "Virtual Machine Manager" and the date "Mar 6 08:02". The main area displays a series of terminal commands and their outputs:

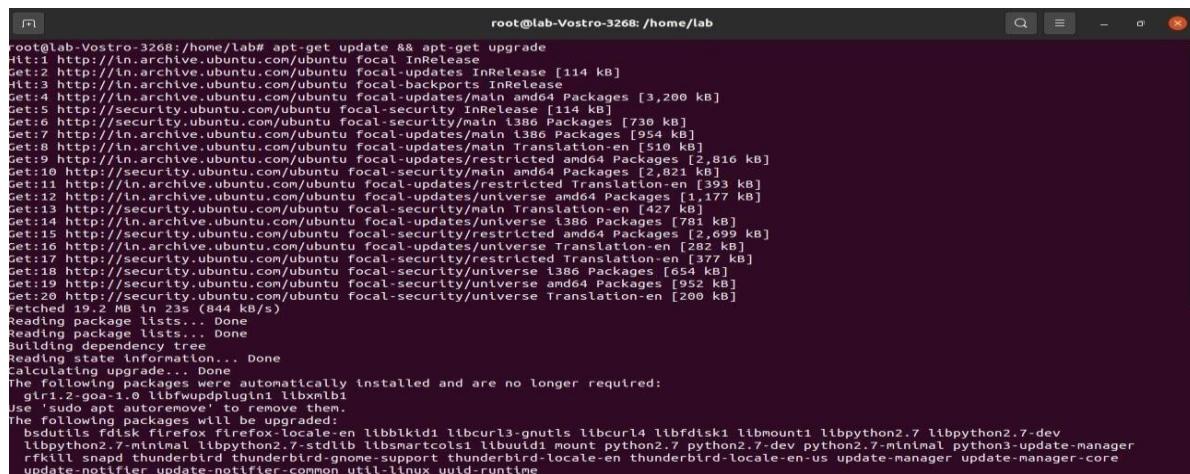
```
s  Virtual Machine Manager ▾ Mar 6 08:02 •  
Foss on QEMU/KVM  
File Virtual Machine View Send Key  
[Icons] i ▶ II ⏪ ⏹  
+-----+  
| Logical Volume Setup |  
+-----+  
Setup LVM environment and volumes  
No volume groups found  
Volume group "local0" successfully created  
Logical volume "home" created  
Logical volume "var" created  
Logical volume "tmp" created  
Logical volume "portage" created  
Logical volume "virtualization" created  
  
All LVM volumes created successfully  
  
+-----+  
| Filesystem Creation |  
+-----+  
Creating filesystems  
mke2fs 1.42.13 (17-May-2015)  
  
Filesystem creation was successful  
  
+-----+  
| Mounting Filesystems |  
+-----+  
Mounting of filesystems was successful  
  
+-----+  
| Stage4 Installation |  
+-----+  
Unpacking stage4 tarball  
This will take a while - please be patient  
  
Unpacking of stage4 tarball was successful  
  
+-----+  
| Network Device Selection |  
+-----+  
Please enter the device which you would like to use  
Available ethernet devices: ens3  
Device #0:
```

Practical-9

Using AWS Flow Framework develop application that includes a simple workflow. Workflow calls an activity to print hello world to the console. It must define the basic usage of AWS Flow Framework, including defining contracts, implementation of activities and workflow coordination logic and worker programs to host them

Step 1: Open Terminal and Update and Upgrade your system by command

```
sudo apt-get update && sudo apt-get upgrade
```



```
root@lab-Vostro-3268:/home/lab# apt-get update && apt-get upgrade
Get:1 http://in.archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://in.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:3 http://in.archive.ubuntu.com/ubuntu focal-backports InRelease
Get:4 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Get:5 http://security.ubuntu.com/ubuntu focal-security/main i386 Packages [3,200 kB]
Get:6 http://security.ubuntu.com/ubuntu focal-security/main amd64 Packages [114 kB]
Get:7 http://in.archive.ubuntu.com/ubuntu focal-updates/main i386 Packages [730 kB]
Get:8 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [954 kB]
Get:9 http://in.archive.ubuntu.com/ubuntu focal-updates/main Translation-en [510 kB]
Get:10 http://in.archive.ubuntu.com/ubuntu focal-updates/restricted amd64 Packages [2,816 kB]
Get:11 http://in.archive.ubuntu.com/ubuntu focal-security/main amd64 Packages [2,821 kB]
Get:12 http://in.archive.ubuntu.com/ubuntu focal-updates/restricted Translation-en [393 kB]
Get:13 http://in.archive.ubuntu.com/ubuntu focal-updates/universe amd64 Packages [1,177 kB]
Get:14 http://in.archive.ubuntu.com/ubuntu focal-updates/universe Translation-en [981 kB]
Get:15 http://security.ubuntu.com/ubuntu focal-security/restricted i386 Packages [781 kB]
Get:16 http://in.archive.ubuntu.com/ubuntu focal-security/universe amd64 Packages [2,699 kB]
Get:17 http://in.archive.ubuntu.com/ubuntu focal-security/restricted Translation-en [282 kB]
Get:18 http://security.ubuntu.com/ubuntu focal-security/universe i386 Packages [654 kB]
Get:19 http://security.ubuntu.com/ubuntu focal-security/universe amd64 Packages [592 kB]
Get:20 http://in.archive.ubuntu.com/ubuntu focal-security/universe Translation-en [200 kB]
Fetched 19,4 MB in 35s (844 kB/s)
Reading package lists... Done
Reading package lists... Done
Building dependency tree
Building state information... Done
Calculating upgrade... Done
The following packages were automatically installed and are no longer required:
  gir1.2-goa-1.0 libfwupdplugin1 libxkbfile1
Use 'sudo apt autoremove' to remove them.
The following packages will be upgraded:
  libcurl4-openssl-dev libcurl4-openssl-dev:i386 libcurl4-gnutls libcurl4 libfdisk1 libmount1 libpython2.7 libpython2.7-dev
  libpython2.7-minimal libpython2.7-stdlib libsmartcols1 libubuid1 mount python2.7 python2.7-dev python2.7-minimal python3-update-manager
  rfkill snapd thunderbird thunderbird-gnome-support thunderbird-locale-en thunderbird-locale-en-us update-manager update-manager-core
  update-notifier update-notifier-common util-linux uuid-runtime
0 upgraded, 19 newly installed, 0 to remove and 0 not upgraded.
Need to get 19,4 MB/19,4 MB of archives.
After this operation, 1,1 MB disk space will be freed.
```

Step 2: Download awscliv2.zip with command

```
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
```



```
root@lab-Vostro-3268:/home/lab# curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
% Total    % Received % Xferd  Average Speed   Time   Time     Current
          Dload  Upload Total   Spent   Left  Speed
100 57.5M  100 57.5M    0      0  2866k      0  0:00:20  0:00:20  --:--:-- 4225k
root@lab-Vostro-3268:/home/lab#
```

Step 3: Download awscliv2.sig file with command

```
curl -o awscliv2.sig https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip.sig
```



```
root@lab-Vostro-3268:/home/lab# curl -o awscliv2.sig https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip.sig
% Total    % Received % Xferd  Average Speed   Time   Time     Current
          Dload  Upload Total   Spent   Left  Speed
100  566  100  566    0      0   765      0  --:--:--  --:--:--  --:--:--  765
root@lab-Vostro-3268:/home/lab#
```

Step 4: unzip awscliv2.zip with command

unzip awscliv2.zip

```
root@lab-Vostro-3268:/home/lab# unzip awscliv2.zip
Archive: awscliv2.zip
  creating: aws/
  creating: aws/dist/
inflating: aws/THIRD_PARTY LICENSES
inflating: aws/install
inflating: aws/README.md
  creating: aws/dist/awscli/
  creating: aws/dist/cryptography/
  creating: aws/dist/docutils/
  creating: aws/dist/lib-dynload/
inflating: aws/dist/aws
inflating: aws/dist/aws_completer
inflating: aws/dist/libpython3.11.so.1.0
inflating: aws/dist/_awscrt.abi3.so
inflating: aws/dist/_cffi_backend.cpython-311-x86_64-linux-gnu.so
inflating: aws/dist/_ruamel_yaml.cpython-311-x86_64-linux-gnu.so
inflating: aws/dist/libbz.so.1
inflating: aws/dist/liblzma.so.0
inflating: aws/dist/libbz2.so.1
inflating: aws/dist/libffi.so.5
inflating: aws/dist/libsqlite3.so.0
inflating: aws/dist/base_library.zip
inflating: aws/dist/lib-dynload/_pickle.cpython-311-x86_64-linux-gnu.so
inflating: aws/dist/lib-dynload/_hashlib.cpython-311-x86_64-linux-gnu.so
inflating: aws/dist/lib-dynload/_sha3.cpython-311-x86_64-linux-gnu.so
inflating: aws/dist/lib-dynload/_blake2.cpython-311-x86_64-linux-gnu.so
inflating: aws/dist/lib-dynload/_sha256.cpython-311-x86_64-linux-gnu.so
inflating: aws/dist/lib-dynload/_md5.cpython-311-x86_64-linux-gnu.so
inflating: aws/dist/lib-dynload/_sha1.cpython-311-x86_64-linux-gnu.so
inflating: aws/dist/lib-dynload/_sha512.cpython-311-x86_64-linux-gnu.so
inflating: aws/dist/lib-dynload/_random.cpython-311-x86_64-linux-gnu.so
```

Step 5: Run command

```
sudo ./aws/install
```

```
root@lab-Vostro-3268:/home/lab# sudo ./aws/install  
You can now run: /usr/local/bin/aws --version
```

Step 6: Type command

pip3 install aws-sam-cli

```
root@lab-Vostro-3268:/home/lab# pip3 install aws-sam-cli
Collecting aws-sam-cli
  Downloading aws_sam_cli-1.113.0-py3-none-any.whl (5.9 MB)
    |██████████| 5.9 MB 21 kB/s
Collecting aws-lambda-builders==1.47.0
  Downloading aws_lambda_builders-1.47.0-py3-none-any.whl (130 kB)
    |██████████| 130 kB 547 kB/s
Collecting tzlocal==5.2
  Downloading tzlocal-5.2-py3-none-any.whl (17 kB)
Collecting dateparser~=1.2
  Downloading dateparser-1.2.0-py2.py3-none-any.whl (294 kB)
    |██████████| 294 kB 2.4 MB/s
Collecting Flask<3.1
  Downloading flask-3.0.2-py3-none-any.whl (101 kB)
    |██████████| 101 kB 447 kB/s
Collecting boto3<2,>=1.29.2
  Downloading boto3-1.34.76-py3-none-any.whl (139 kB)
    |██████████| 139 kB 558 kB/s
Collecting pyopenssl~=24.1.0
  Downloading pyOpenSSL-24.1.0-py3-none-any.whl (56 kB)
    |██████████| 56 kB 580 kB/s
Collecting requests~=2.31.0
  Using cached requests-2.31.0-py3-none-any.whl (62 kB)
Collecting click~=8.1
  Downloading click-8.1.7-py3-none-any.whl (97 kB)
    |██████████| 97 kB 377 kB/s
Collecting watchdog==4.0.0
  Downloading watchdog-4.0.0-py3-none-manylinux2014_x86_64.whl (82 kB)
    |██████████| 82 kB 113 kB/s
Collecting chevron~=0.12
  Downloading chevron-0.14.0-py3-none-any.whl (11 kB)
Collecting ruamel.yaml~=0.18.6
  Downloading ruamel.yaml-0.18.6-py3-none-any.whl (117 kB)
    |██████████| 117 kB 363 kB/s
Collecting aws-sam-translator==1.86.0
```

Step 7: Type command sam init in terminal to launch Sam

CLISelect 1st option to use AWS Quick Start Templates

```
root@lab-Vostro-3268:/home/lab# sam init
SAM CLI now collects telemetry to better understand customer needs.

You can OPT OUT and disable telemetry collection by setting the
environment variable SAM_CLI_TELEMETRY=0 in your shell.
Thanks for your help!

Learn More: https://docs.aws.amazon.com/serverless-application-model/latest/developerguide/serverless-sam-telemetry.html

/usr/lib/python3/dist-packages/paramiko/transport.py:220: CryptographyDeprecationWarning: Blowfish has been deprecated and will be removed in
a future release
  "class": algorithms.Blowfish,
You can preselect a particular runtime or package type when using the `sam init` experience.
Call `sam init --help` to learn more.

Which template source would you like to use?
  1 - AWS Quick Start Templates
  2 - Custom Template Location
Choice: 1
```

Step 8: Select Template no.1 Hello World Example

```
Choose an AWS Quick Start application template
  1 - Hello World Example
  2 - Data processing
  3 - Hello World Example with Powertools for AWS Lambda
  4 - Multi-step workflow
  5 - Scheduled task
  6 - Standalone function
  7 - Serverless API
  8 - Infrastructure event management
  9 - Lambda Response Streaming
 10 - Serverless Connector Hello World Example
 11 - Multi-step workflow with Connectors
 12 - GraphQL API Hello World Example
 13 - Full Stack
 14 - Lambda EFS example
 15 - Hello World Example With Powertools for AWS Lambda
 16 - DynamoDB Example
 17 - Machine Learning
Template: 1
```

Step 9: Type “N” if it ask to use most popular runtime and package type

Open new terminal by pressing **ctrl+shift+T** and check for python version by command **python –version**

Select the option according to your python version in my case its option 19- python 3.11

```
Use the most popular runtime and package type? (Python and zip) [y/N]: n

Which runtime would you like to use?
  1 - aot.dotnet7 (provided.al2)
  2 - dotnet8
  3 - dotnet6
  4 - go1.x
  5 - go (provided.al2)
  6 - go (provided.al2023)
  7 - graalvm.java11 (provided.al2)
  8 - graalvm.java17 (provided.al2)
  9 - java21
 10 - java17
 11 - java11
 12 - java8.al2
 13 - nodejs20.x
 14 - nodejs18.x
 15 - nodejs16.x
 16 - python3.9
 17 - python3.8
 18 - python3.12
 19 - python3.11
 20 - python3.10
 21 - ruby3.2
 22 - rust (provided.al2)
 23 - rust (provided.al2023)

Runtime: 17
```

Step 10: Select package type as Zip

```
What package type would you like to use?
  1 - Zip
  2 - Image
Package type: 1
```

Step 11: Now choose Yes option everytime it ask .

Give project name as per your preference in my case its sam-app-test

```
Based on your selections, the only dependency manager available is pip.
We will proceed copying the template using pip.

Would you like to enable X-Ray tracing on the function(s) in your application? [y/N]: y
X-Ray will incur an additional cost. View https://aws.amazon.com/xray/pricing/ for more details

Would you like to enable monitoring using CloudWatch Application Insights?
For more info, please view https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/cloudwatch-application-insights.html [y/N]: y
AppInsights monitoring may incur additional cost. View https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/appinsights-what-is.html
#appinsights-pricing for more details

Would you like to set Structured Logging in JSON format on your Lambda functions? [y/N]: y
Structured Logging in JSON format might incur an additional cost. View https://docs.aws.amazon.com/lambda/latest/dg/monitoring-cloudwatchlogs.html#monitoring-cloudwatchlogs-pricing for more details

Project name [sam-app]: sam-app-test

Cloning from https://github.com/aws/aws-sam-cli-app-templates (process may take a moment)

-----
Generating application:
-----
Name: sam-app-test
Runtime: python3.8
Architectures: x86_64
Dependency Manager: pip
Application Template: hello-world
Output Directory: .
Configuration file: sam-app-test/samconfig.toml

Next steps can be found in the README file at sam-app-test/README.md
```

Step 12: Now one folder will be created by your provided project name go into that folder by command cd (folder name)

After entering the project folder we will invoke the HelloWorldFunction by using command sam local invoke „HelloWorldFunction“

```
root@lab-Vostro-3268:/home/lab/sam-app-test# sam local invoke 'HelloWorldFunction'
/usr/lib/python3/dist-packages/paramiko/transport.py:220: CryptographyDeprecationWarning: Blowfish has been deprecated and will be removed in a future release
  "class": algorithms.Blowfish,
Invoking app.lambda_handler (python3.8)
Local image was not found.
Removing rapid images for repo public.ecr.aws/sam/emulation-python3.8
Building image.....  
.....  
Using local image: public.ecr.aws/lambda/python:3.8-rapid-x86_64.  
Mounting /home/lab/sam-app-test/hello_world as /var/task:ro,delegated, inside runtime container
START RequestId: 065fefd3-5d41-4b87-bb31-8d820fb635e6 Version: $LATEST
END RequestId: 4b9baa5c-2523-443d-b340-f86e2b139f6a
REPORT RequestId: 4b9baa5c-2523-443d-b340-f86e2b139f6a Init Duration: 0.06 ms Duration: 99.11 ms Billed Duration: 100 ms Memory Size: 1
28 MB Max Memory Used: 128 MB
{"statusCode": 200, "body": "{\"message\": \"hello world\"}"}
root@lab-Vostro-3268:/home/lab/sam-app-test#
```

It should give you StatusCode:200

Use command sudo snap install docker if docker error occurs.

Step 13: Type command sam local start-api this will give you URL open it in any browser.

```
root@lab-Vostro-3268:/home/lab/sam-app-test# sam local start-api
/usr/lib/python3/dist-packages/paramiko/transport.py:220: CryptographyDeprecationWarning: Blowfish has been deprecated and will be removed in a future release
  "class": algorithms.Blowfish,
Initializing the lambda functions containers.
Local image was not found.
Removing rapid images for repo public.ecr.aws/sam/emulation-python3.11
Building Image.....  
.....  
Using local image: public.ecr.aws/lambda/python:3.11-rapid-x86_64.  
Mounting /home/lab/sam-app-test/hello_world as /var/task:ro,delegated, inside runtime container
Containers Initialization is done.
Mounting HelloWorldFunction at http://127.0.0.1:3000/hello [GET]
You can now browse to the above endpoints to invoke your functions. You do not need to restart/reload SAM CLI while working on your functions, changes will be reflected instantly/automatically. If you used sam build before running local commands, you will need to re-run sam build for the changes to be picked up. You only need to restart SAM CLI if you update your AWS SAM template
2024-04-03 09:26:18 WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:3000
2024-04-03 09:26:18 Press CTRL+C to quit
Invoking app.lambda_handler (python3.11)
Reuse the created warm container for Lambda function 'HelloWorldFunction'
Lambda function 'HelloWorldFunction' is already running
START RequestId: 0652f01c-89a7-43d9-85fb-3844c8f1a32b Version: $LATEST
END RequestId: b229bc70-c5b6-4bf2-b9a3-97b2cf9340a1
REPORT RequestId: b229bc70-c5b6-4bf2-b9a3-97b2cf9340a1 Init Duration: 0.04 ms Duration: 1153.99 ms Billed Duration: 1154 ms Memory Size: 128 MB Max Memory Used: 128 MB
No Content-Type given. Defaulting to 'application/json'.
2024-04-03 09:26:38 127.0.0.1 - - [03/Apr/2024 09:26:38] "GET /hello HTTP/1.1" 200 -
2024-04-03 09:26:39 127.0.0.1 - - [03/Apr/2024 09:26:39] "GET /favicon.ico HTTP/1.1" 403 -
```

Output:



Practical-10

Implementation of Openstack with user and private network creation.

Implementation of Openstack with user and private network creation. 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

With the minimum requirements satisfied, we can now proceed.

Steps:

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
```

Step 2: Create Stack user and assign sudo privilege

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

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

```
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 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. Sample

```
git clone https://git.openstack.org/openstack-dev/devstack
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. Then create

```
cd devstack
```

```
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 cont 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 internet connection. In our case, it took roughly 12 minutes. At the very end, you shoul 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: 16d11d27f375b8c027bbc3aldb1885e90ce6c604 Merge "Option "lock_path" from group "DEFAULT"
OS Version: Ubuntu 18.04 bionic

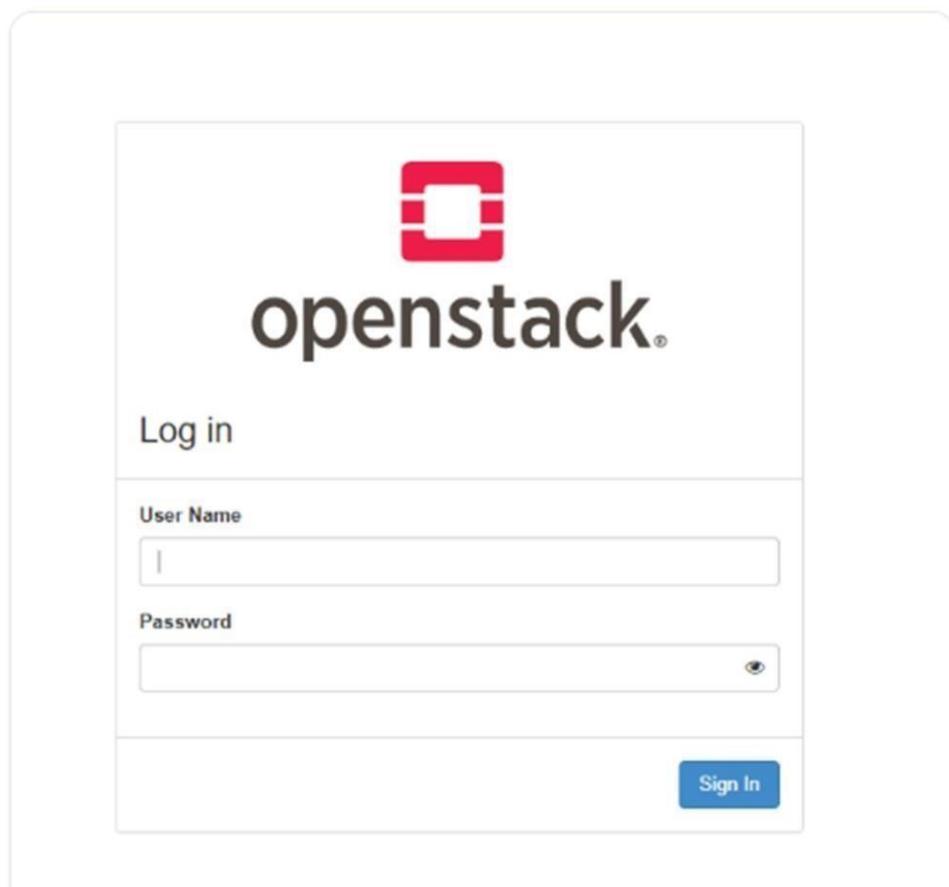
2019-06-04 12:19:19.207 | stack.sh completed in 884 seconds.
```

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.



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

