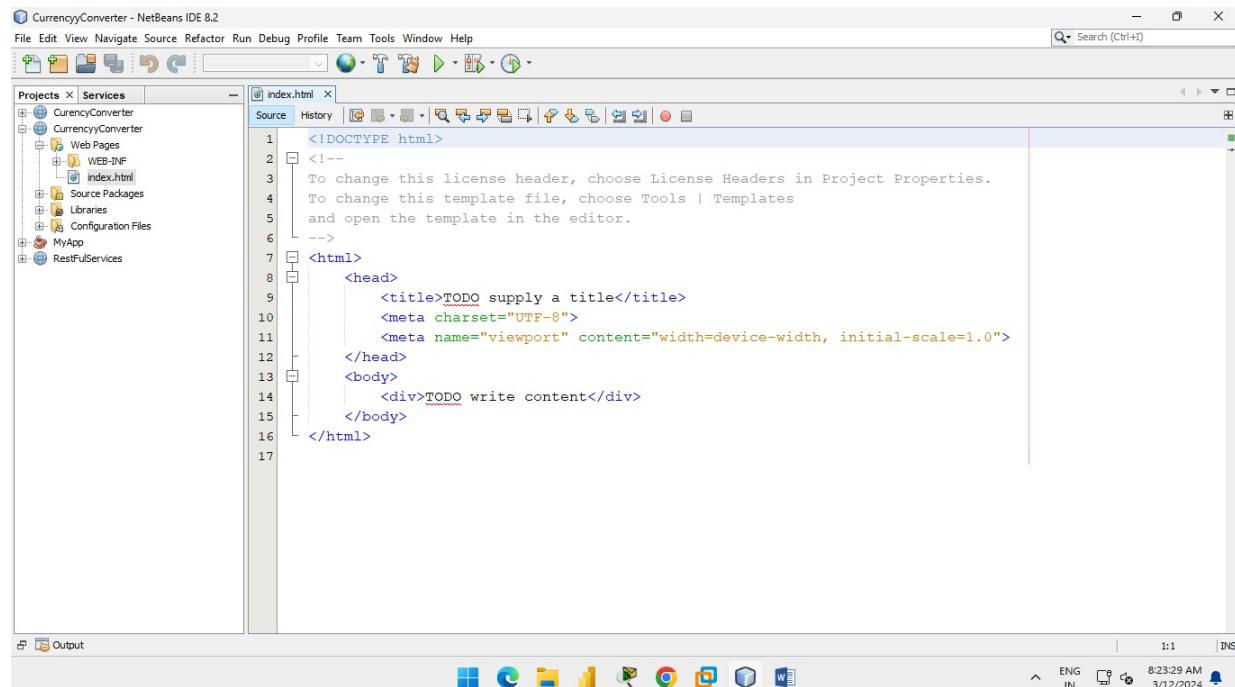


PRACTICAL -1

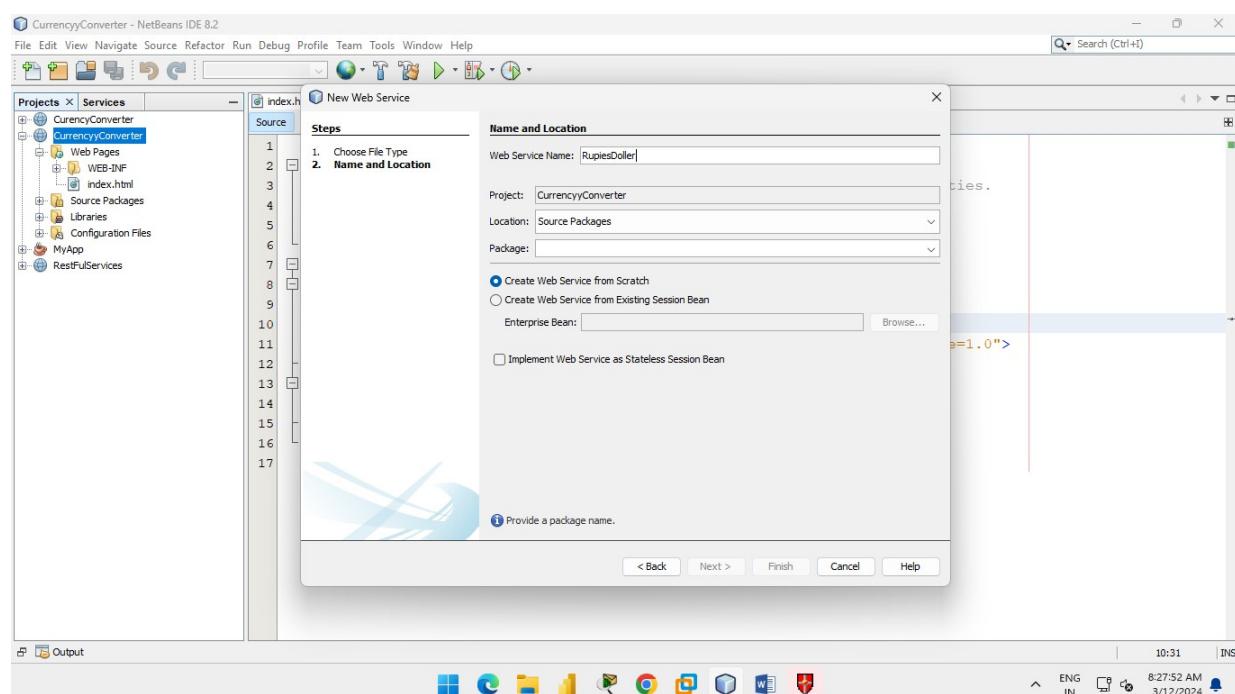
Aim – Define a simple services like Converting Rs into Dollar and Call it from different platform like JAVA and .NET Framework.

A] Convert Rupees into Dollar and Dollar into Rupees using Java.

1. Open Netbeans and create new project.
2. Java web → web application → Project name → Next → Finish



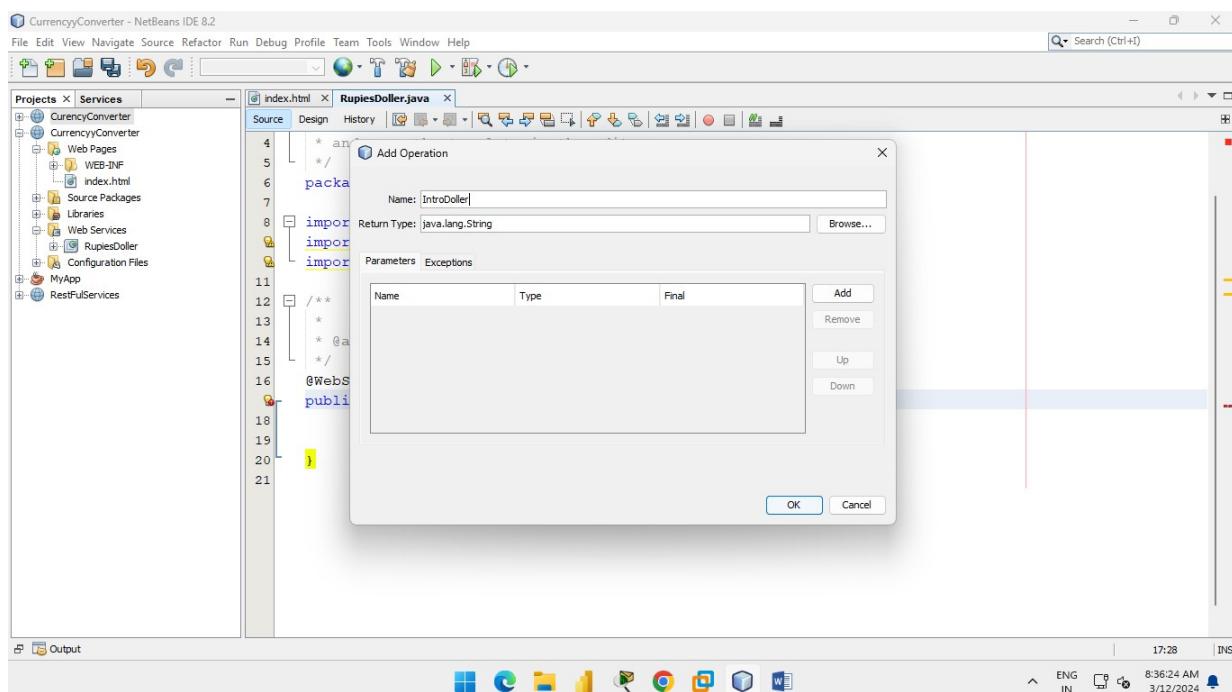
3. Right click on project → new → web service



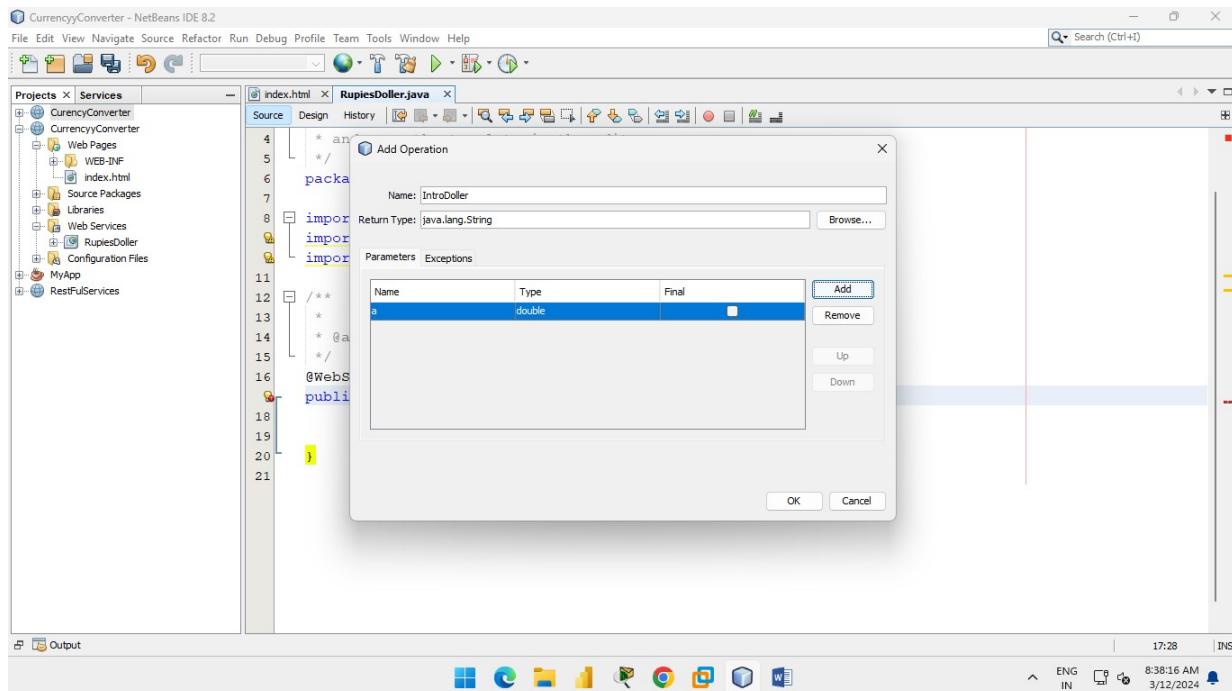
4. Now delete this block of code

```
11
12 /**
13 * @author A
14 */
15 @WebService(serviceName = "RupiesDoller")
16 public class RupiesDoller {
17
18     /**
19      * This is a sample web service operation
20      */
21
22     @WebMethod(operationName = "hello")
23     public String hello(@WebParam(name = "name") String txt) {
24         return "Hello " + txt + " !";
25     }
26 }
27
```

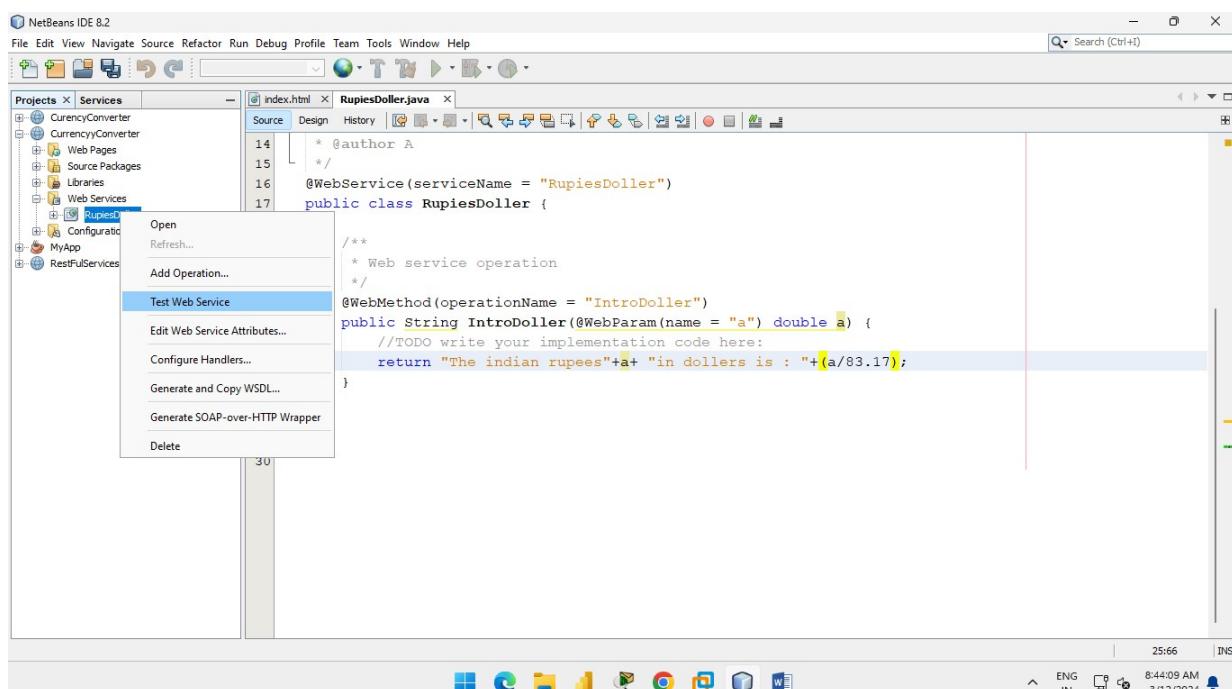
5. Right click→insert code→Add web service operations



6. Give name→Add→



7. Right click on web service file → Test web service



OUTPUT:

RupeesDoller 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 mycurrency.com.RupiesDoller.convertIntoDollar(double)
convertIntoDollar(150)
```

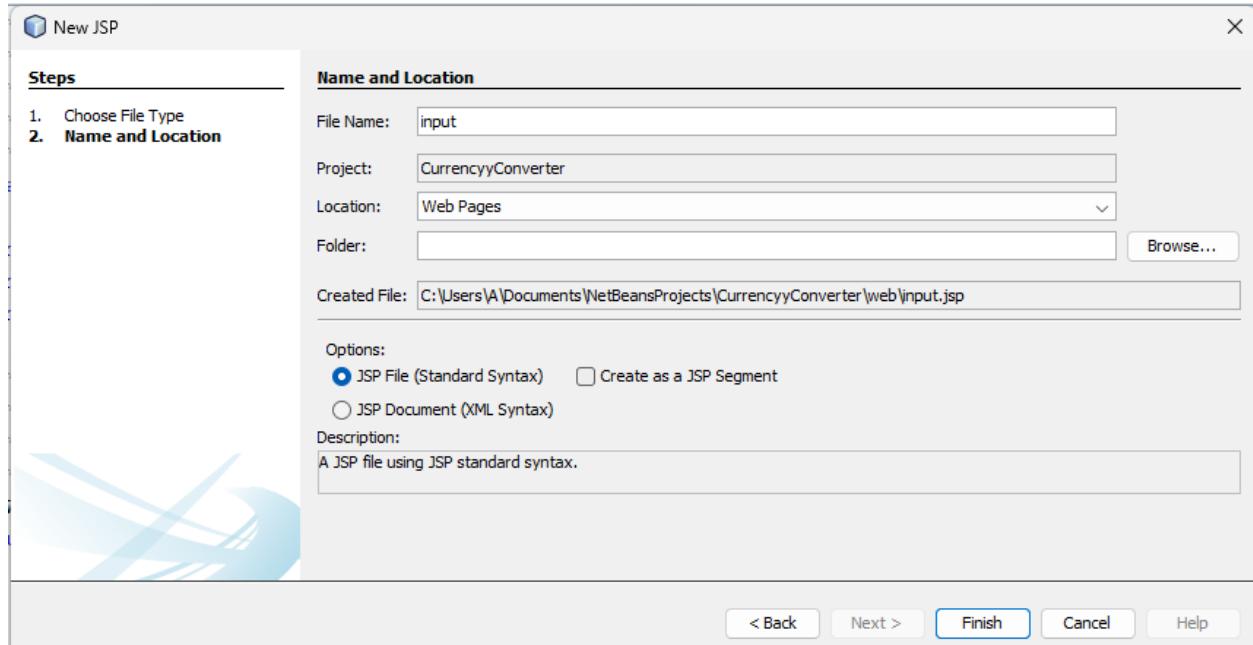
Method parameter(s)

Type	Value
double	150

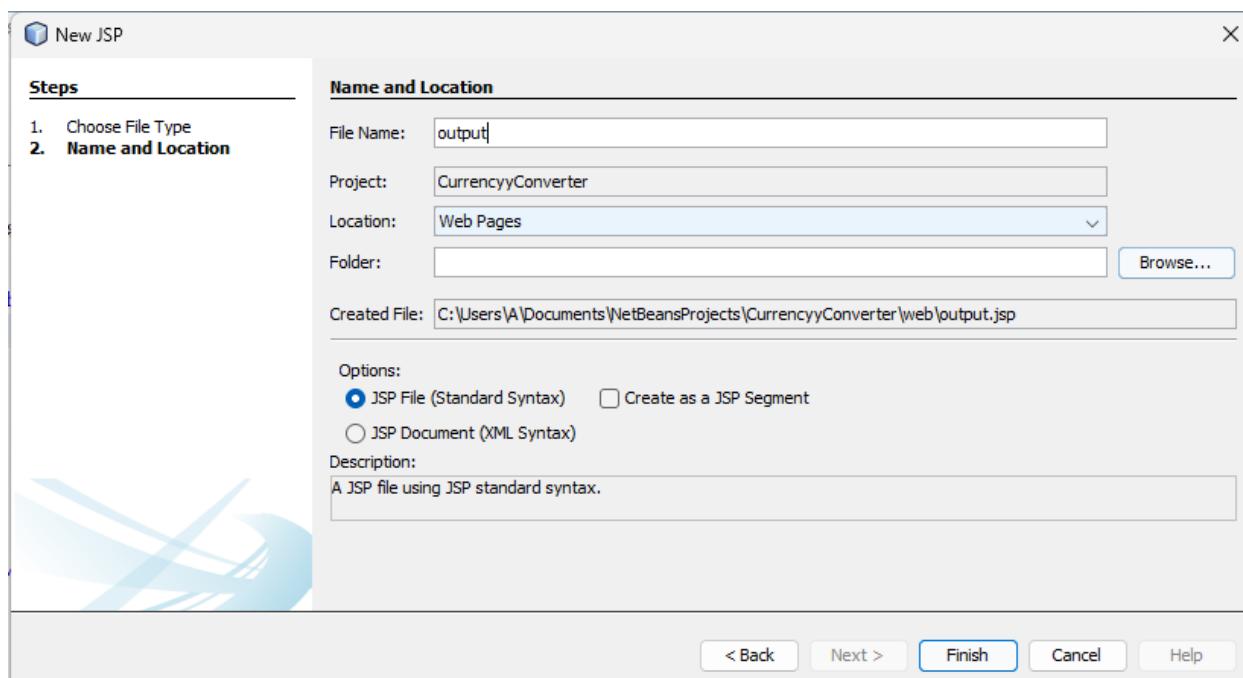
Method returned

```
java.lang.String : "The indian rupees 150.0 in dollar is : 1.803534928459781"
```

8. Right click on Web pages→new→JSP



9. Follow the same steps.



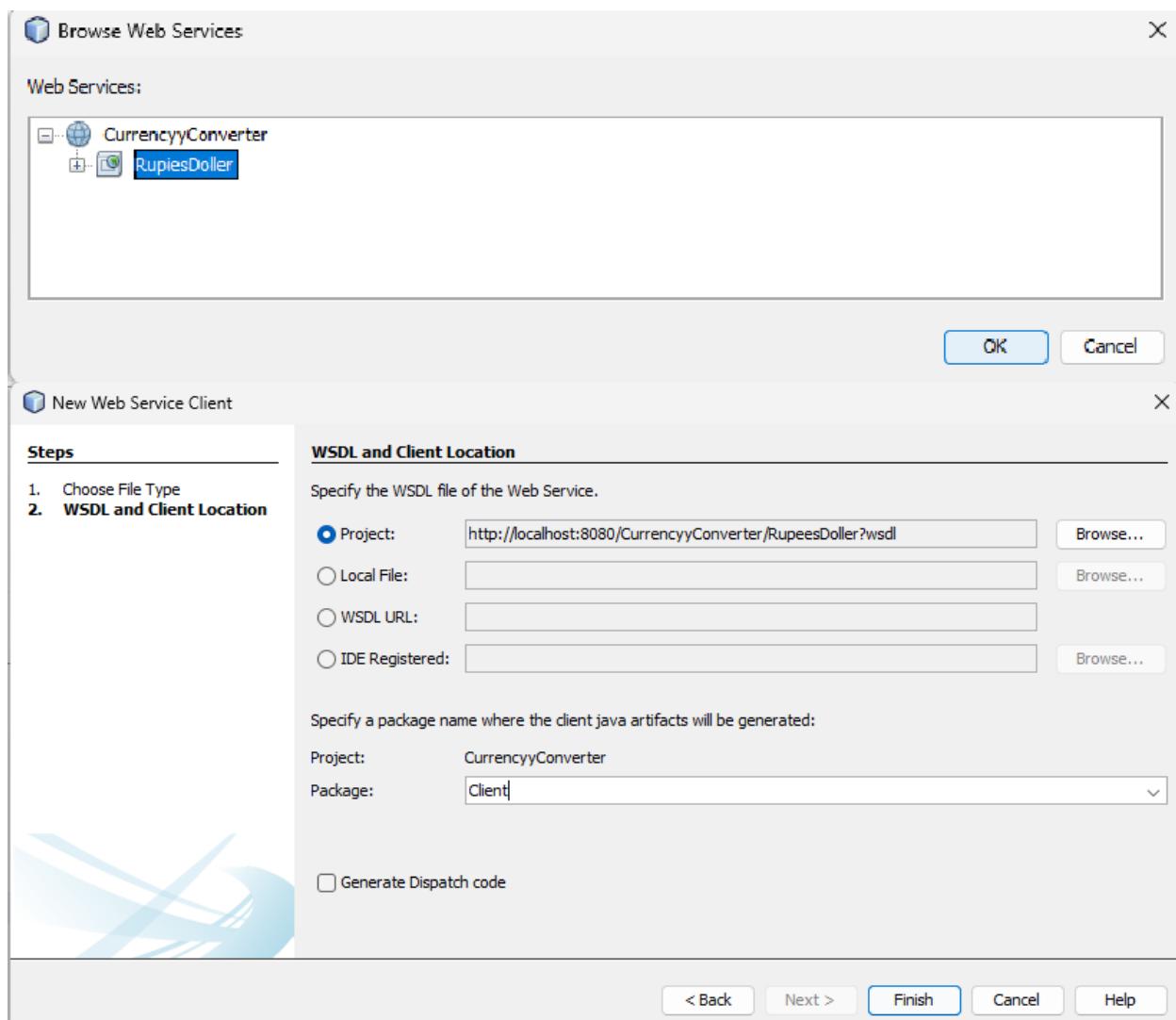
```

<%-->
2   Document      : output
3   Created on   : Mar 12, 2024, 9:06:03 AM
4   Author        : A
5 --%>

6
7   <%@page contentType="text/html" pageEncoding="UTF-8"%>
8   <!DOCTYPE html>
9   <html>
10    <head>
11       <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
12       <title>JSP Page</title>
13    </head>
14    <body>
15       <form action="output.jsp">
16          <pre>
17             Enter Indian Rupees to convert : <input type="text" name="t1">
18             <input type="submit"> <input type="reset">
19          </pre>
20       </form>
21    </body>
22 </html>
23

```

10. Write click on project→New→Web Service client→browse→ok



11. Drag the button in body tang of output.jsp file.

The screenshot shows the NetBeans IDE interface with the following details:

- File Menu:** File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help.
- Toolbar:** Standard Java development toolbar with icons for file operations, search, and project navigation.
- Projects Tab:** Shows the project structure with CurrencyConverter, CurrencyConverter, Web Pages, and Web Services.
- Source Editor:** Displays the `input.jsp` file content. The code is a JSP page that includes a try-catch block to invoke a web service named RupeesDoller. The service is initialized from a Client object, and its port is used to call the `convertIntoDollar` method with a parameter `t1`. The result is then printed to the output.
- Bottom Bar:** Shows the operating system taskbar with various application icons.

12. Right click on project → build → Right click on input.jsp → Run file

This screenshot is identical to the one above, showing the NetBeans IDE interface with the `input.jsp` file open in the source editor. The code remains the same, demonstrating the JSP code for invoking a web service.

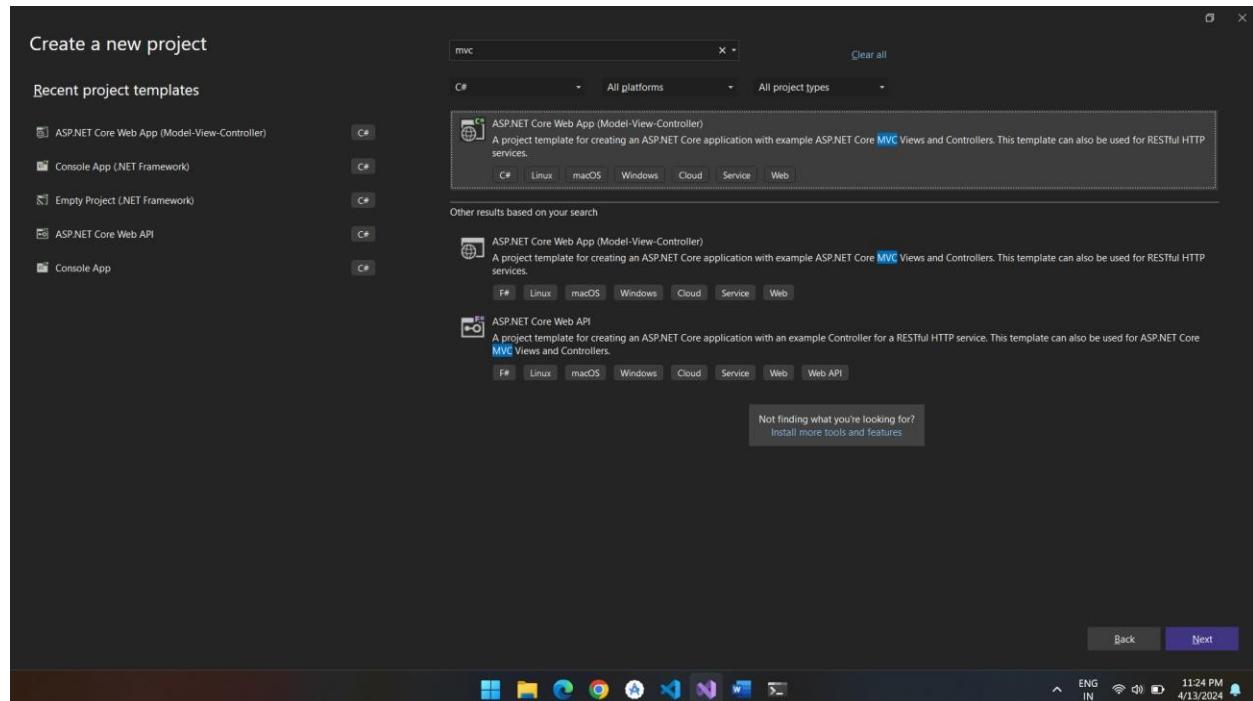
OUTPUT:

The screenshot shows a browser window with the following details:

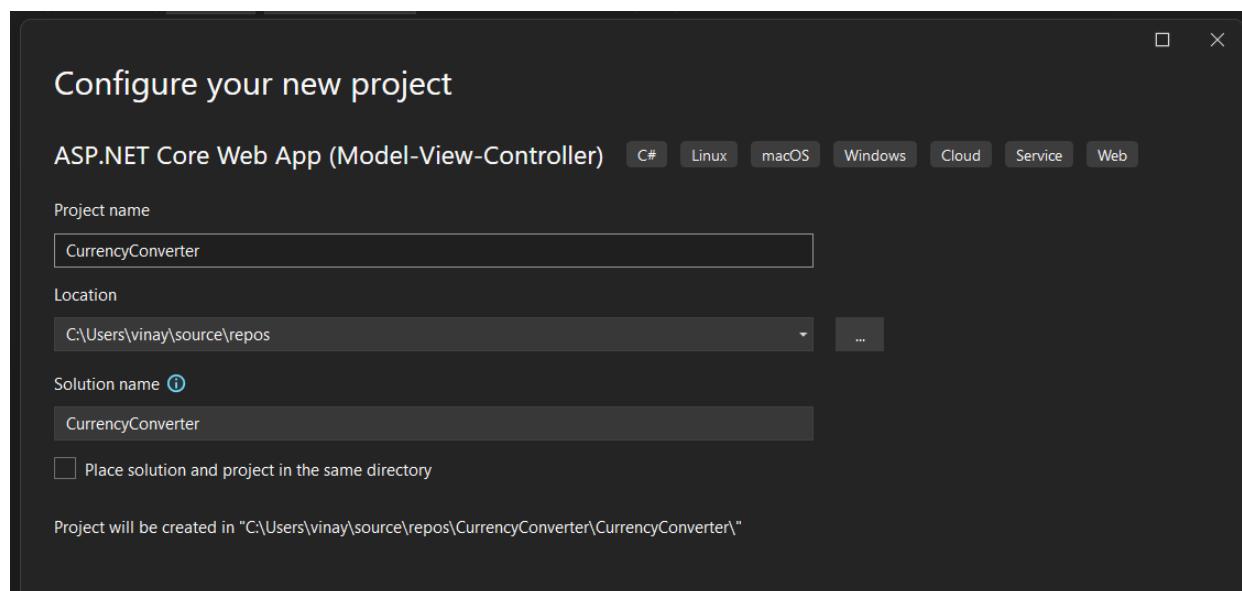
- Address Bar:** localhost:8080/CurrencyConverter/input.jsp
- Form:** An input field labeled "Enter Indian Rupees to convert :" containing the value "250". Below it are "Submit" and "Reset" buttons.
- Output Page:** A separate tab titled "JSP Page" showing the result of the conversion. The page displays the message "Result = The indian rupees 250.0 in dollar is : 3.0058915474329684".

B] Convert Rupees into Dollar and Dollar into Rupees using .NET Framework.

1. create new project→search MVC and select it.



2. Give the project Name→Next→create



3. After successful creation of project→open solution explorer
4. Views→Home→Index.cshtml and write the following code

The screenshot shows the Visual Studio IDE with the 'Index.cshtml' file open in the editor. The code is a Razor view for a currency converter. It includes an H2 header, a form for selecting conversion (from INR to USD or USD to INR), an input field for amount, and a submit button. It also checks for a ViewBag.Result and displays it if present. The Solution Explorer on the right shows the project structure with files like Index.cshtml, Privacy.cshtml, and Program.cs.

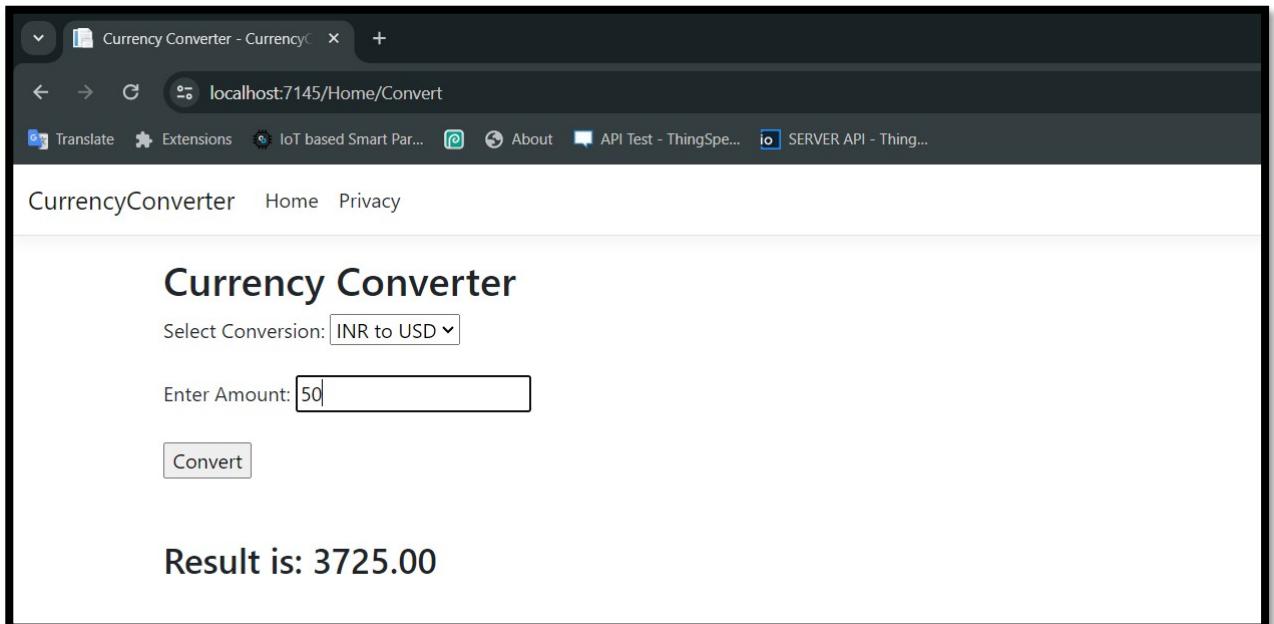
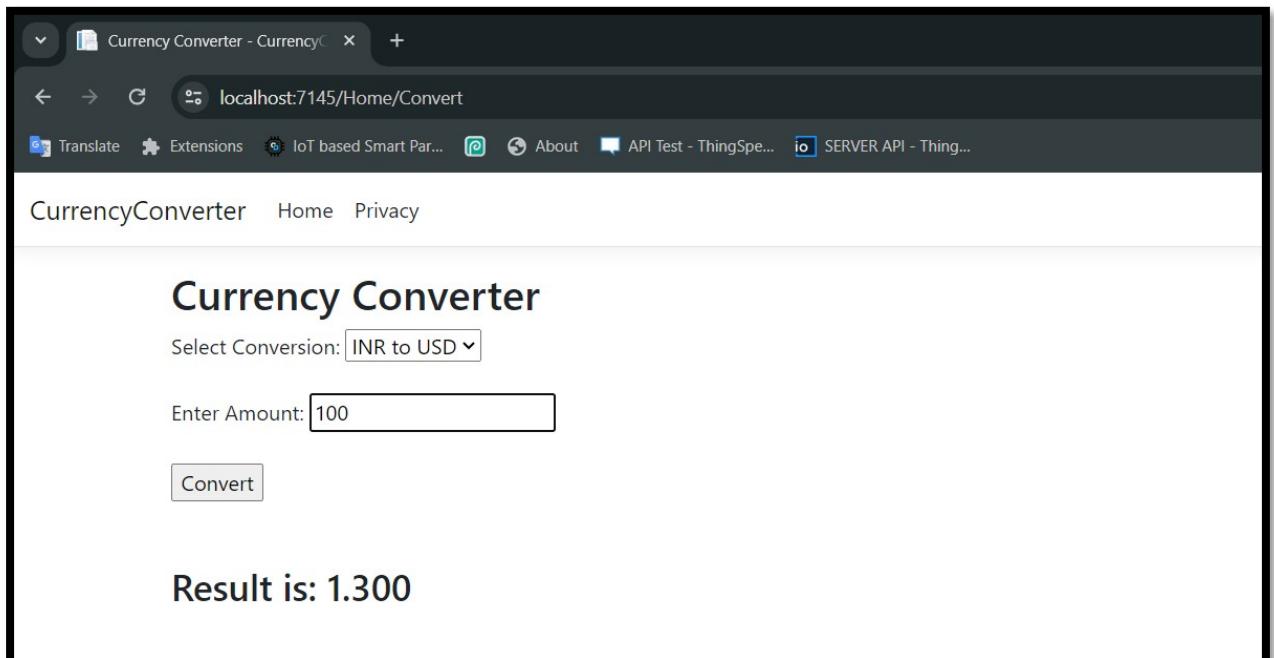
```
1 @{
2     ViewBag.Title = "Currency Converter";
3 }
4 
5 <h2>Currency Converter</h2>
6 
7 <form method="post" action="@Url.Action("Convert", "Home")">
8     <label for="currency">Select Conversion:</label>
9     <select id="currency" name="currency">
10        <option value="INRtoUSD">INR to USD</option>
11        <option value="USDtoINR">USD to INR</option>
12    </select><br><br>
13 
14    <label for="amount">Enter Amount:</label>
15    <input type="text" id="amount" name="amount" /><br><br>
16 
17    <input type="submit" value="Convert" />
18 </form>
19 
20 <@if (ViewBag.Result != null)
21 {
22     <h3>Result: @ViewBag.Result</h3>
23 }
24 
```

5. After that open Controller→HomeController.cs

The screenshot shows the Visual Studio IDE with the 'HomeController.cs' file open in the editor. The code defines a HomeController that inherits from Controller. It has an Index action that returns a View. It also has a Convert action that takes currency and amount parameters, performs conversion logic (either INR to USD or USD to INR based on the selected option), and sets the result in ViewBag.Result before returning the index view. The Solution Explorer on the right shows the project structure with files like HomeController.cs, Index.cshtml, Privacy.cshtml, and Program.cs.

```
1 using Microsoft.AspNetCore.Mvc;
2 
3 namespace CurrencyConverter.Controllers
4 {
5     public class HomeController : Controller
6     {
7         public ActionResult Index()
8         {
9             return View();
10        }
11 
12        [HttpPost]
13        public ActionResult Convert(string currency, decimal amount)
14        {
15            decimal result = 0;
16            decimal conversionRate = 0;
17 
18            if (currency == "INRtoUSD")
19            {
20                // Conversion rate from INR to USD
21                conversionRate = 0.013m;
22            }
23            else if (currency == "USDtoINR")
24            {
25                // Conversion rate from USD to INR
26                conversionRate = 74.56m;
27            }
28 
29            result = amount * conversionRate;
30 
31            ViewBag.Result = result;
32            return View("Index");
33        }
34    }
35 
```

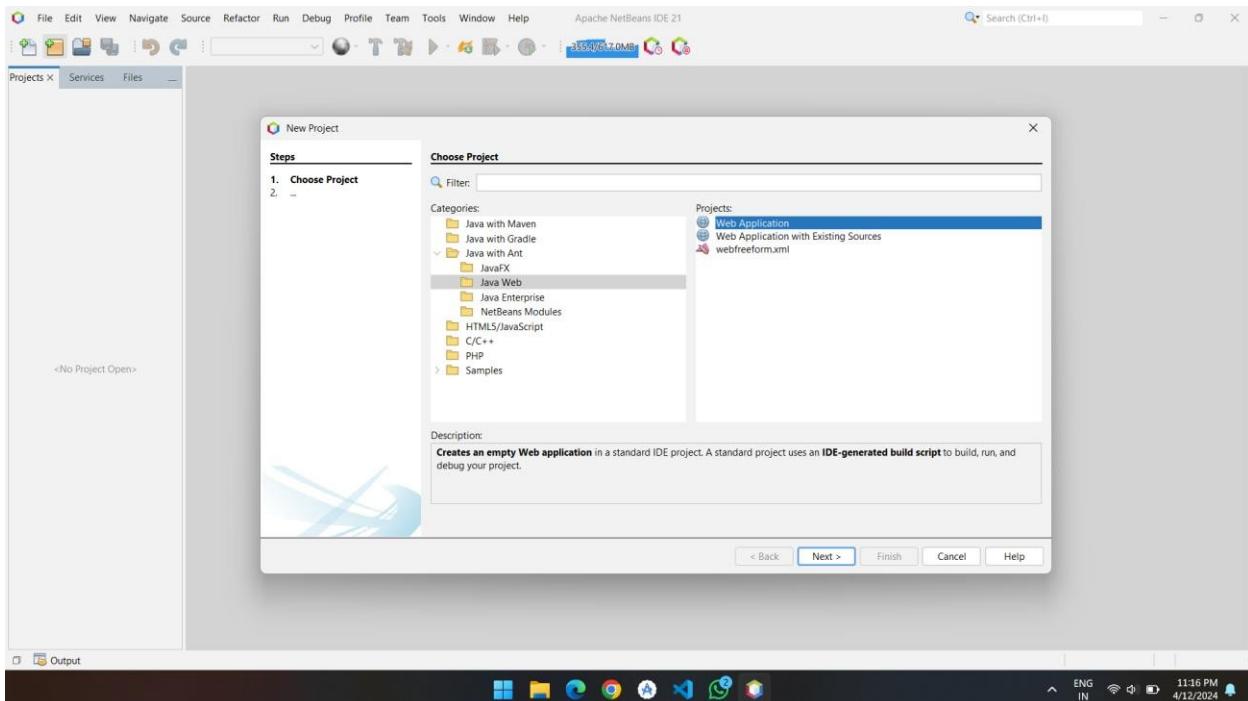
OUTPUT:



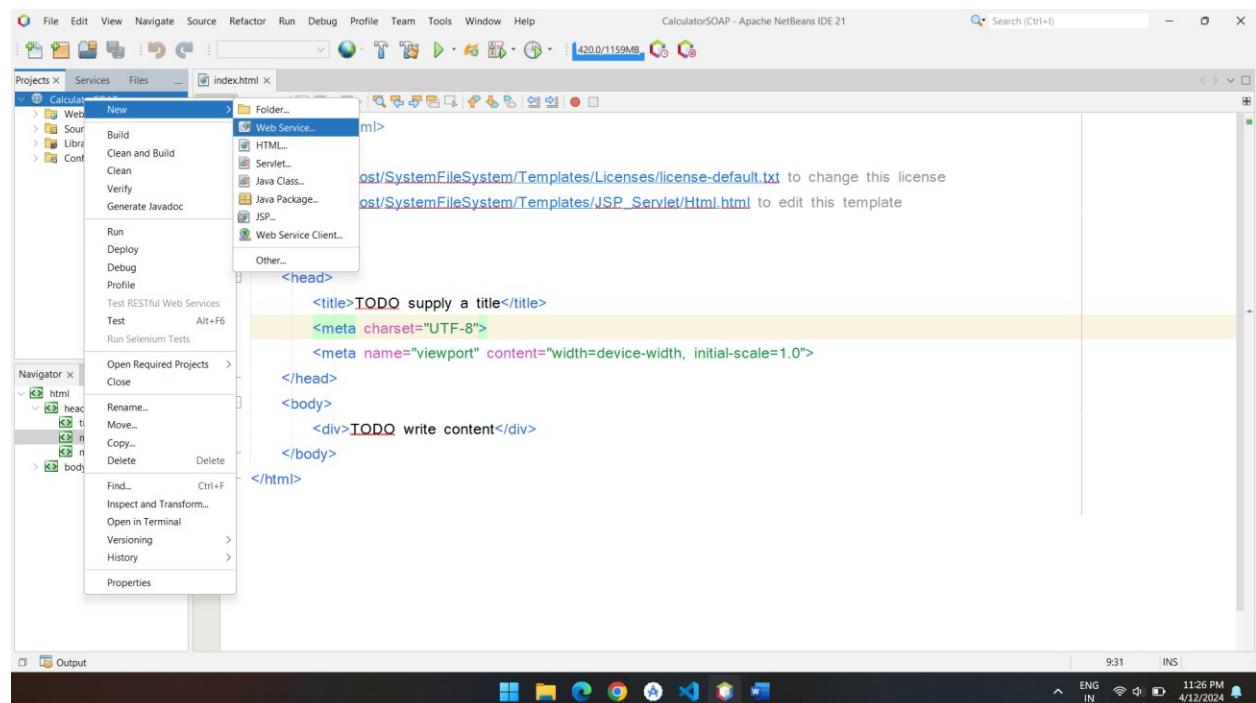
PRACTICAL NO – 2

Aim – Create a simple SOAP Service.

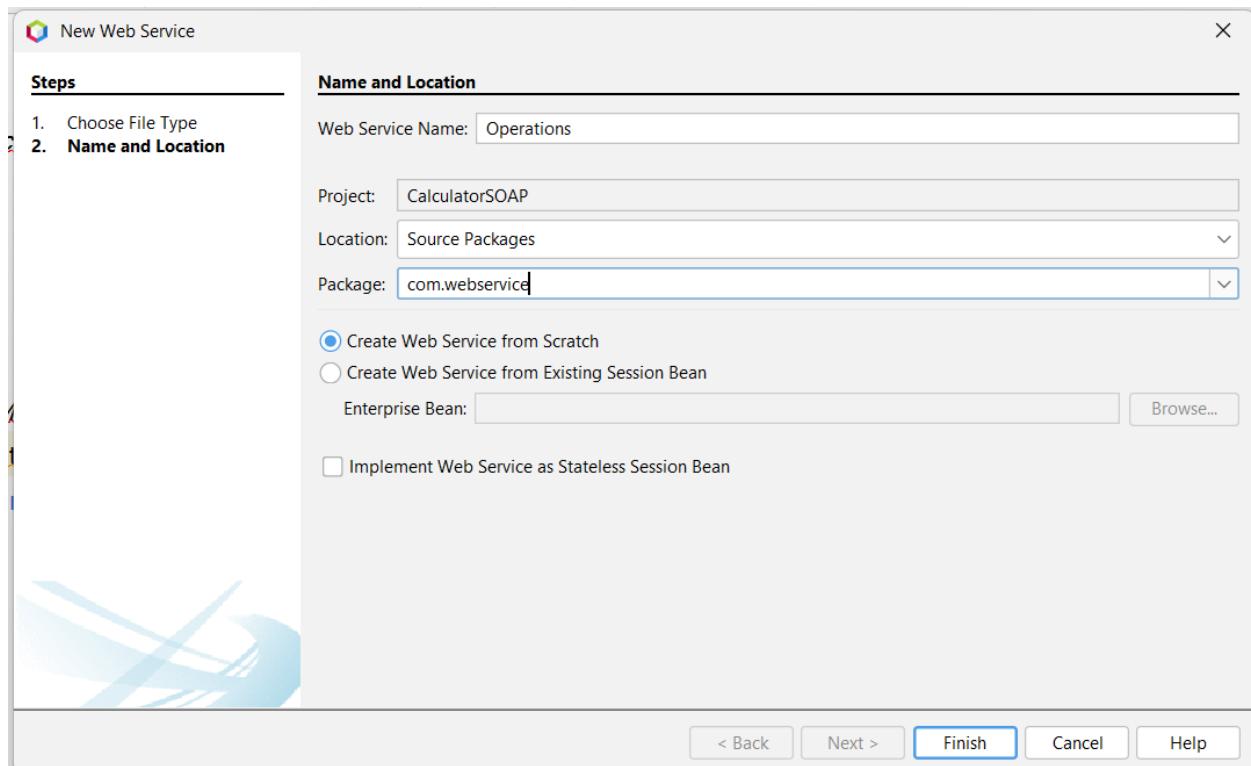
1. Create the new project in netbence software
2. File→New Project→Java web→web Application→Next



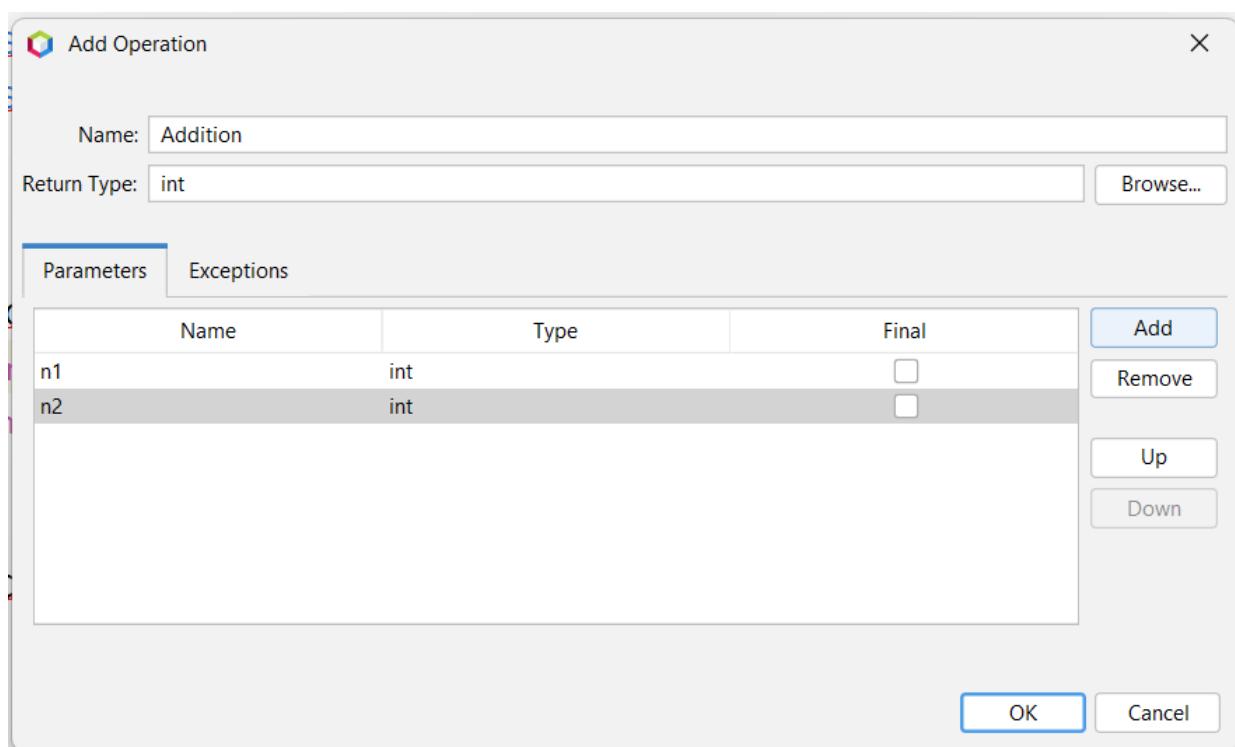
3. Now right click on your project and Add→New→Web Service.



4. Give the web service name and package Name.



5. Now open Web services folder → Right click on your web webservice → Add operation
6. After that give operation name → give return type → click on add button → give the name of twonumbers → ok.



7. Now you can see the method of Addition will generate automatically. You have to edit it according to your function.

CalculatorSOAP - NetBeans IDE 8.2

```

8  /**
9  * @author vinay
10 */
11
12 @WebService(serviceName = "ArthOperations")
13 public class ArthOperations {
14
15
16     @WebMethod(operationName = "Addition")
17     public int Addition(@WebParam(name = "n1") int n1, @WebParam(name = "n2") int n2) {
18
19         int sum = n1+n2;
20         return sum;
21     }
22 }
23

```

Output x CalculatorSOAP (run-deploy) x Java DB Database Process x Glassfish Server 4.1.1 x

Sat Apr 13 09:25:05 IST 2024 : Security manager installed using the Basic server security policy.
Sat Apr 13 09:25:05 IST 2024 : Apache Derby Network Server - 10.10.2.0 - (1582446) started and ready to accept connections on port 1527

8. Now follow the same steps to create the methods for Subtraction, Multiplication and Division.

CalculatorSOAP - NetBeans IDE 8.2

```

21 }
22
23 /**
24 * Web service operation
25 */
26 @WebMethod(operationName = "Subtraction")
27 public int Subtraction(@WebParam(name = "n1") int n1, @WebParam(name = "n2") int n2) {
28     int sub = n1-n2;
29     return sub;
30 }
31
32 /**
33 * Web service operation
34 */
35 @WebMethod(operationName = "Multiplication")
36 public int Multiplication(@WebParam(name = "n1") int n1, @WebParam(name = "n2") int n2) {
37     int mul = n1*n2;
38     return mul;
39 }
40
41 /**
42 * Web service operation
43 */
44 @WebMethod(operationName = "Division")
45 public int Division(@WebParam(name = "n1") int n1, @WebParam(name = "n2") int n2) {
46     int div = n1/n2;
47     return div;
48 }
49

```

9. Right click on project → Deploy

10. Write click on webservice → Test Web Service

OUTPUT:

ArthOperations Web Service Tester

This form will allow you to test your web service implementation ([WSDL File](#))

To invoke an operation, fill the method parameter(s) input boxes and click on the button labeled with the method name.

Methods :

```
public abstract int com.webservice.ArthOperations.multiplication(int,int)
multiplication |(23| |24|)
```

```
public abstract int com.webservice.ArthOperations.division(int,int)
division |(20| |2|)
```

```
public abstract int com.webservice.ArthOperations.subtraction(int,int)
subtraction |(122| |22|)
```

```
public abstract int com.webservice.ArthOperations.addition(int,int)
addition |(500| |500|)
```

11. Now we can perform any Addition, Subtraction, Multiplication and Division.

ArthOperations Web Service Tester

This form will allow you to test your web service implementation ([WSDL File](#))

To invoke an operation, fill the method parameter(s) input boxes and click on the button labeled with the method name.

Methods :

```
public abstract int com.webservice.ArthOperations.multiplication(int,int)
multiplication |(23| |24|)
```

```
public abstract int com.webservice.ArthOperations.division(int,int)
division |(20| |2|)
```

```
public abstract int com.webservice.ArthOperations.subtraction(int,int)
subtraction |(122| |22|)
```

```
public abstract int com.webservice.ArthOperations.addition(int,int)
addition |(500| |500|)
```

MU_TYBSC_CS | ArthOperations | Method invocation | + - ×

localhost:8080/CalculatorSOAP/ArthOperations?Tester

Dell Gmail YouTube Maps Translate News McAfee Security Extensions IoT based Smart Pa... API Test - ThingSpe... SERVER API - Thing... Placement Cell Other favorites

division Method invocation

Method parameter(s)

Type	Value
int	20
int	2

Method returned

int : "10"

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/">
<SOAP-ENV:Header/>
<S:Body>
<ns2:Division xmlns:ns2="http://webservice.com/">
<n1>20</n1>
<n2>2</n2>
</ns2:Division>
</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/">
<SOAP-ENV:Header/>
<S:Body>
<ns2:DivisionResponse xmlns:ns2="http://webservice.com/">
<return>10</return>
</ns2:DivisionResponse>
</S:Body>
</S:Envelope>
```

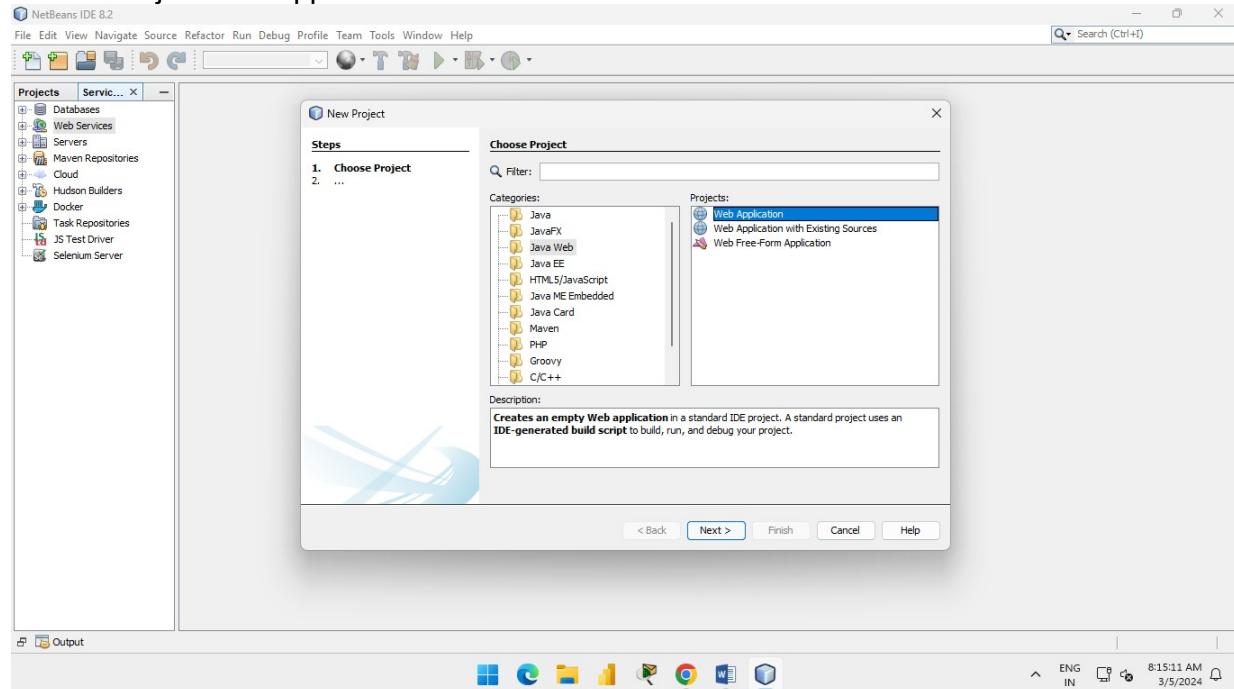
Windows taskbar icons: File Explorer, Task View, Edge, Google Chrome, File, Print, Task Manager, Power, Start.

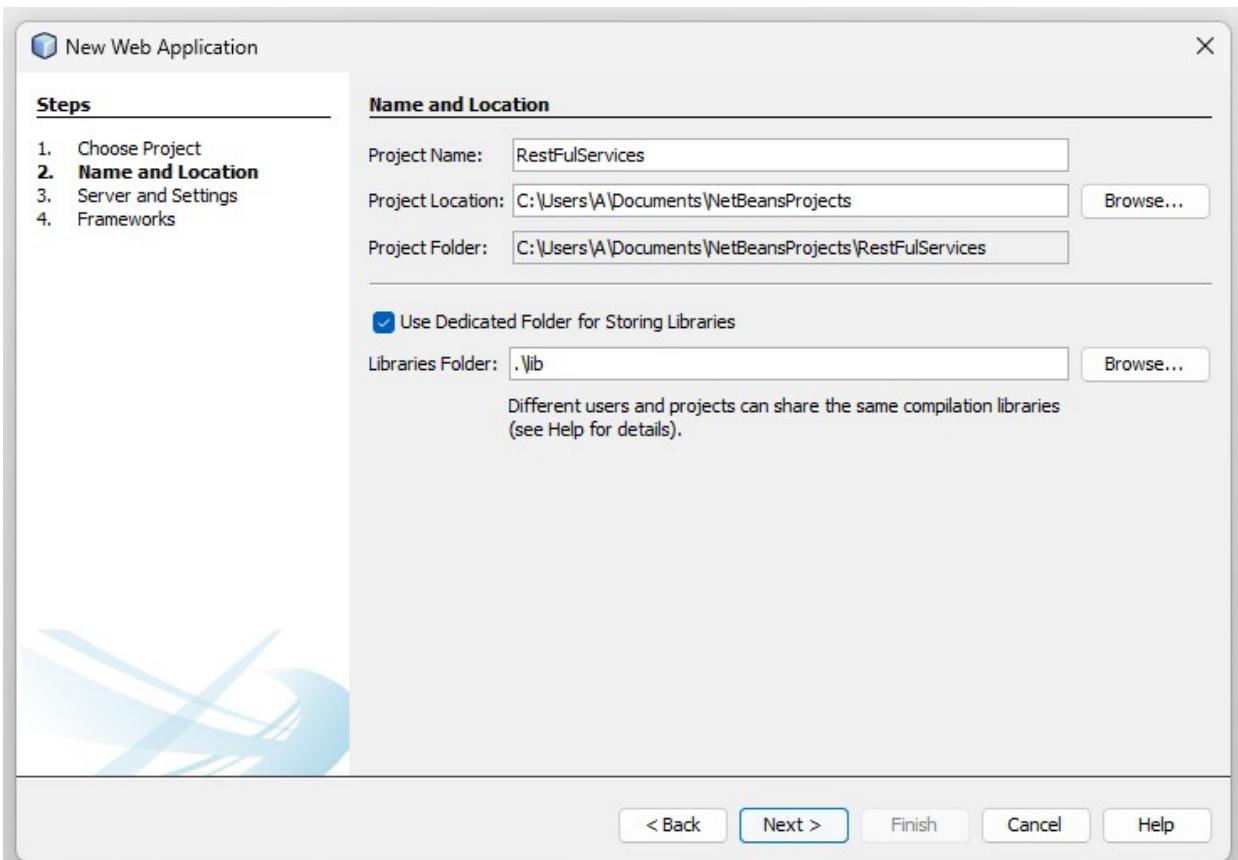
System tray icons: ENG IN, WiFi signal, Battery level, 10:05 AM, 4/13/2024, Bell icon.

PRACTICAL NO – 3

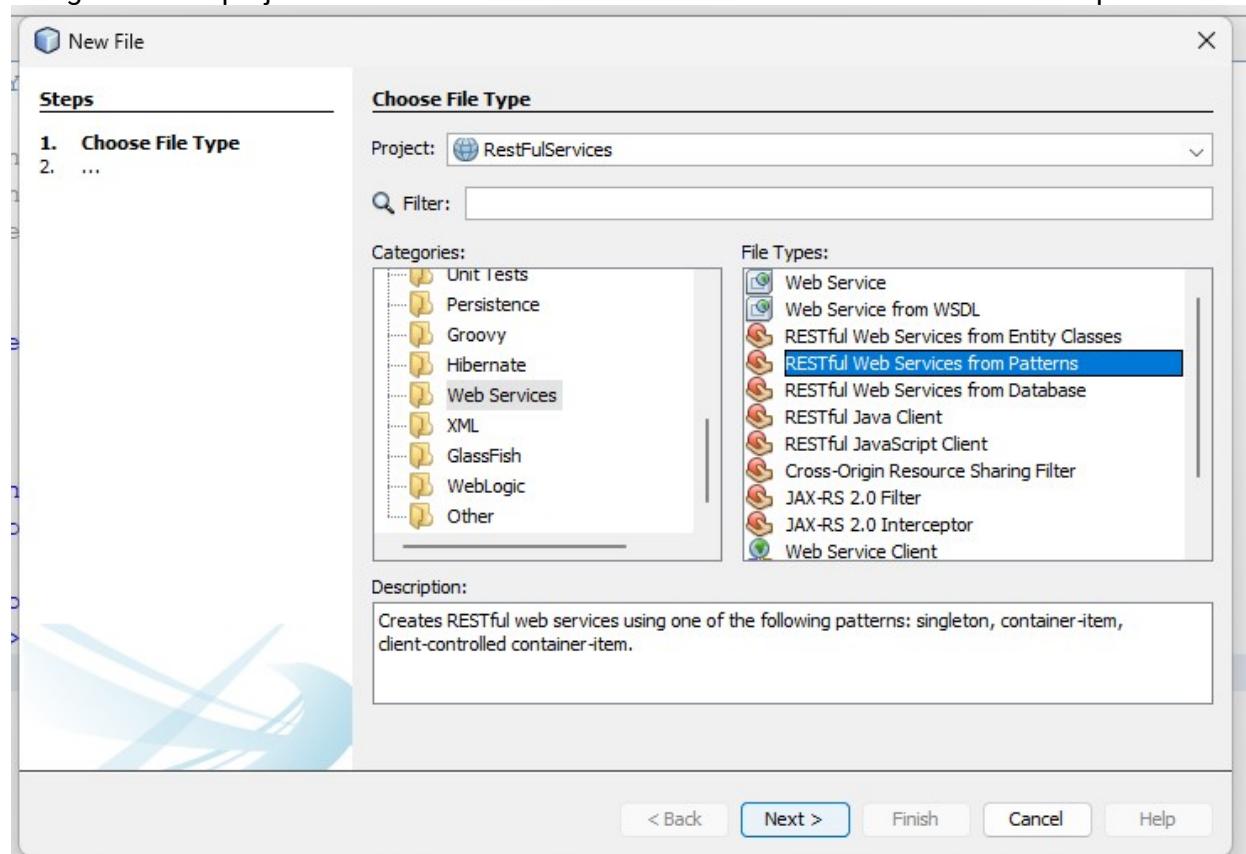
Aim – Create simple REST service.

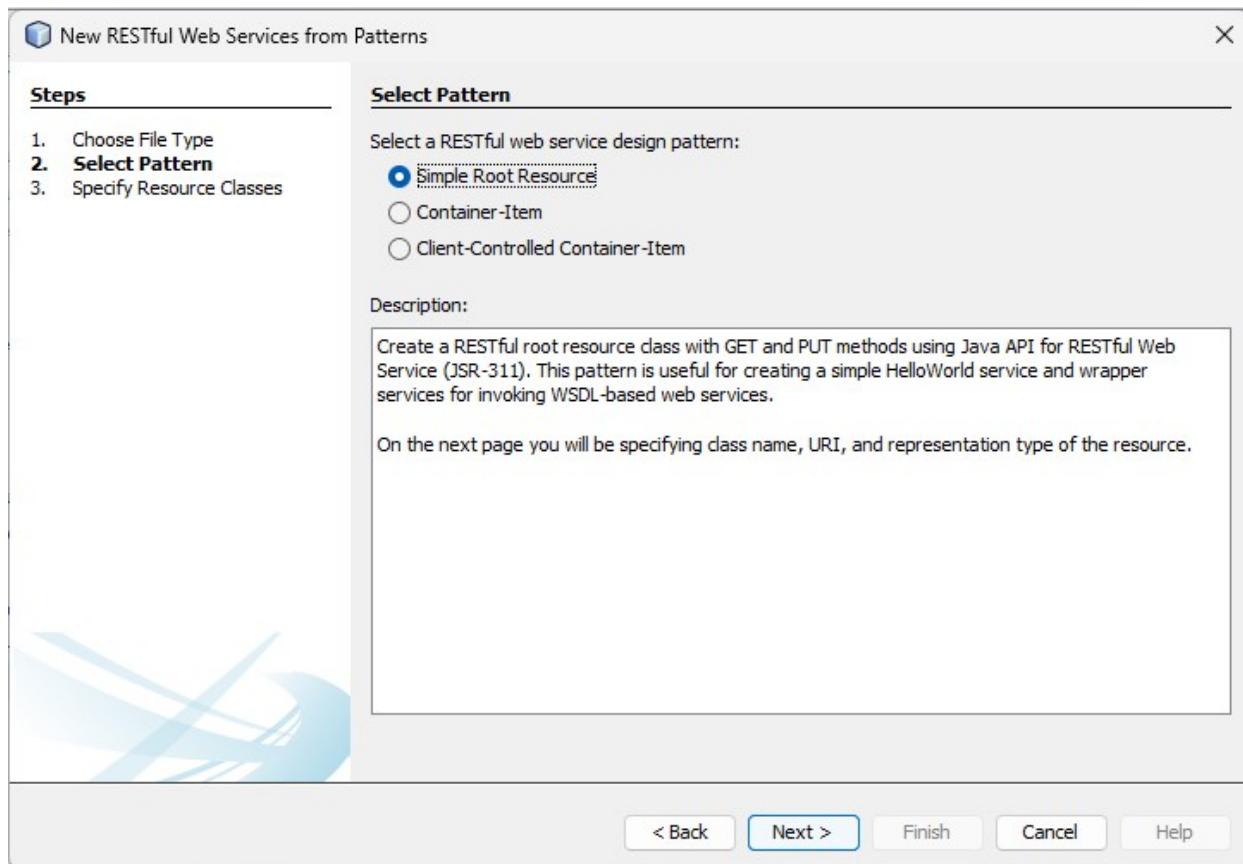
1. Create a java web application.





2. Right click on project → new → others → web services → RESTful web services from patterns.





3. Change the code and create methods as following...

```

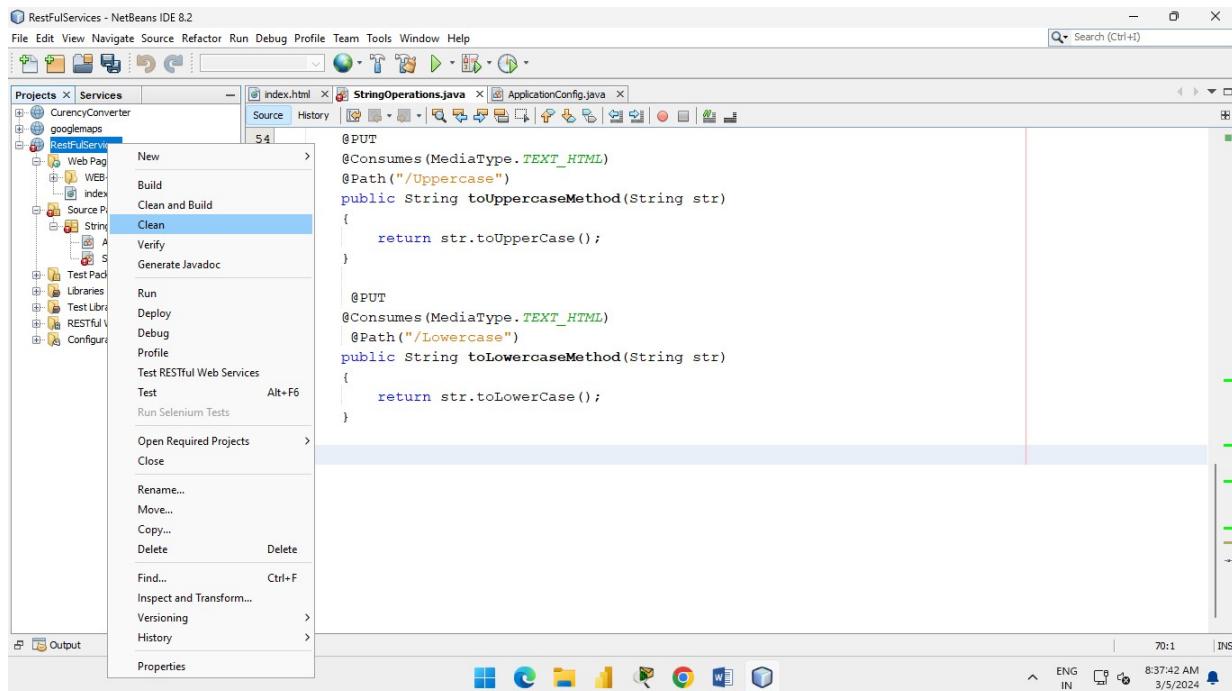
    /**
     * PUT method for updating or creating an instance of StringOperations
     * @param content representation for the resource
     */
    @PUT
    @Consumes(MediaType.TEXT_HTML)
    public void putHtml(String content) {
    }

    @PUT
    @Consumes(MediaType.TEXT_HTML)
    @Path("/Uppercase")
    public String toUppercaseMethod(String str)
    {
        return str.toUpperCase();
    }

    @PUT
    @Consumes(MediaType.TEXT_HTML)
    @Path("/Lowercase")
    public String toLowercaseMethod(String str)
    {
        return str.toLowerCase();
    }
}

```

4. Right click on project and clean the project and after deploy it.



RestfulServices - NetBeans IDE 8.2

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Projects Services index.html StringOperations.java ApplicationConfig.java

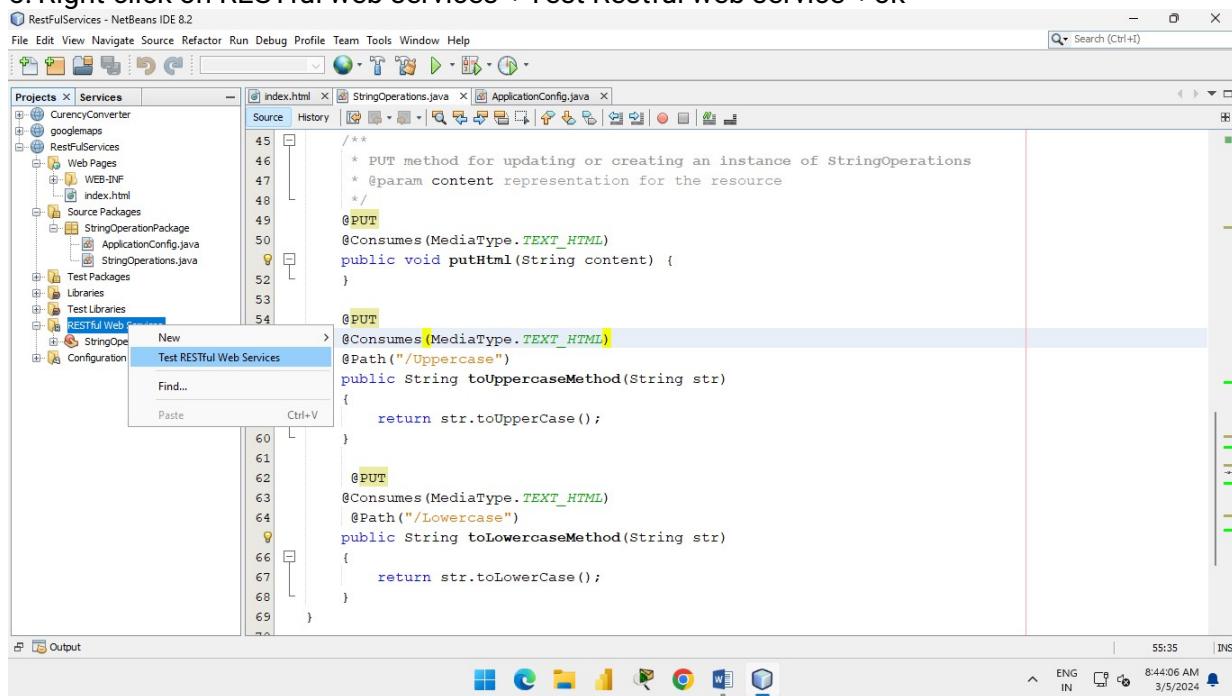
New Build Clean and Build Clean Verify Generate Javadoc Run Deploy Debug Profile Test RESTful Web Services Test Run Selenium Tests Open Required Projects Close Rename... Move... Copy... Delete Find... Inspect and Transform... Versioning History Properties

Source History

```
@PUT  
@Consumes(MediaType.TEXT_HTML)  
@Path("/Uppercase")  
public String toUppercaseMethod(String str)  
{  
    return str.toUpperCase();  
}  
  
@PUT  
@Consumes(MediaType.TEXT_HTML)  
@Path("/Lowercase")  
public String toLowercaseMethod(String str)  
{  
    return str.toLowerCase();  
}
```

Output

5. Right click on RESTful web services → Test Restful web service → ok



RestfulServices - NetBeans IDE 8.2

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

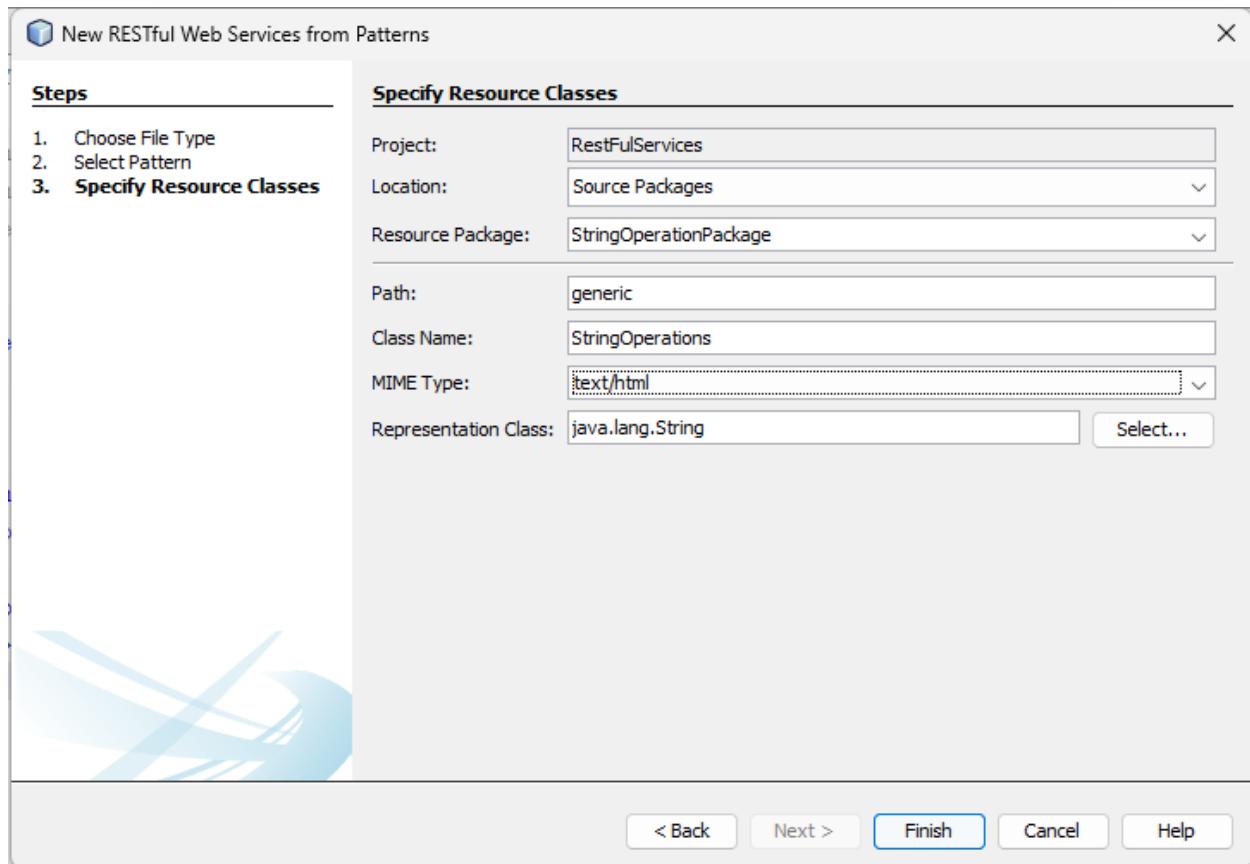
Projects Services index.html StringOperations.java ApplicationConfig.java

New Build Clean and Build Clean Verify Generate Javadoc Run Deploy Debug Profile Test RESTful Web Services Test RESTful Web Services Find... Paste Ctrl+V

Source History

```
/**  
 * PUT method for updating or creating an instance of StringOperations  
 * @param content representation for the resource  
 */  
@PUT  
@Consumes(MediaType.TEXT_HTML)  
public void putHtml(String content) {  
  
    @PUT  
    @Consumes(MediaType.TEXT_HTML)  
    @Path("/Uppercase")  
    public String toUppercaseMethod(String str)  
{  
        return str.toUpperCase();  
    }  
  
    @PUT  
    @Consumes(MediaType.TEXT_HTML)  
    @Path("/Lowercase")  
    public String toLowercaseMethod(String str)  
{  
        return str.toLowerCase();  
    }  
}
```

Output



OUTPUT:

Test RESTful Web Services what is getX in flutter - Google

localhost:8080/RestFulServices/test-resbeans.html

WADL : http://localhost:8080/RestFulServices/webresources/application.wadl

Test RESTful Web Services

RestFulServices > generic > Lowercase

Resource: generic/Lowercase
(http://localhost:8080/RestFulServices/webresources/generic/Lowercase)

Choose method to test: PUT(text/html) Test

Content: MY NAME IS VINAY.

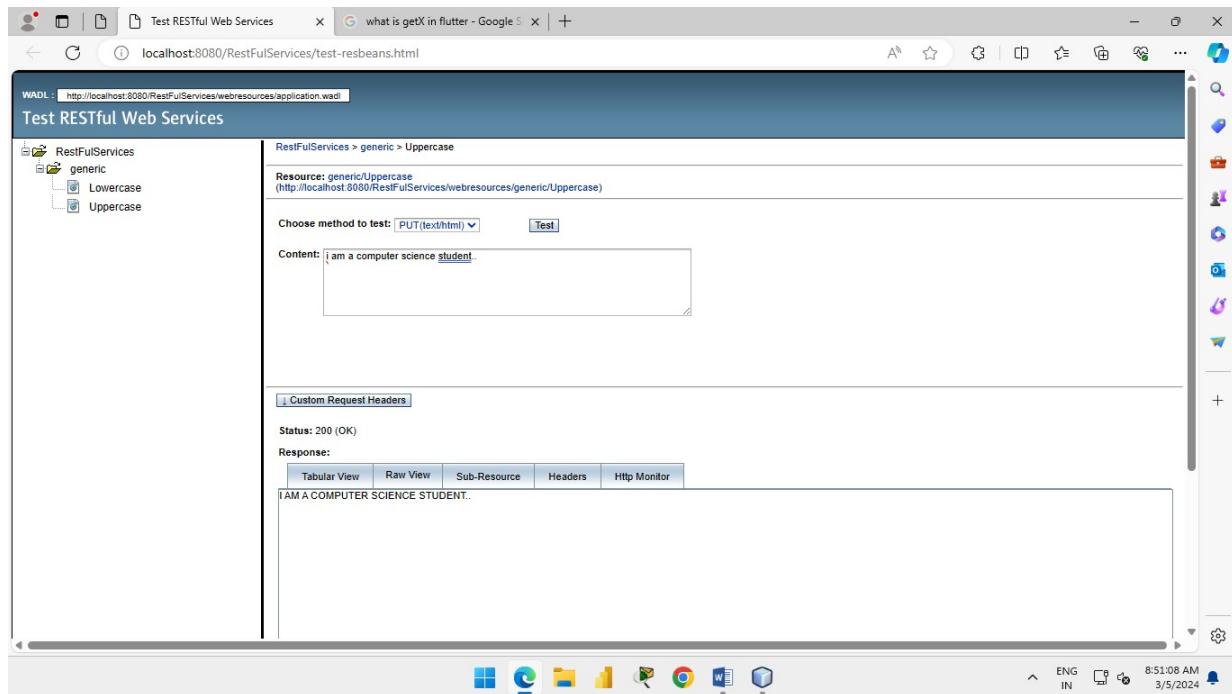
Custom Request Headers

Status: 200 (OK)

Response:

Tabular View Raw View Sub-Resource Headers Http Monitor

my name is vinay.



PRACTICAL NO – 4

Aim – Develop application to consume Google's search / Google map RESTful Web service.

```
import requests

def get_geolocation(api_key, search_string):
    base_url = "https://us1.locationiq.com/v1/search"
    params = {
        'key': api_key,
        'q': search_string,
        'format': 'json',
    }

    response = requests.get(base_url, params=params)
    data = response.json()

    if response.status_code == 200 and data:
        result = {
            'place_id': data[0].get('place_id', ''),
            'lat': data[0].get('lat', ''),
            'lon': data[0].get('lon', ''),
            'display_name': data[0].get('display_name', ''),
        }
        return result
    else:
        print(f"Error: {response.status_code} - {data.get('error', 'No error message')}")
        return None

api_key = 'pk.71c93f03731ac10faf75d7e071df51c1'
search_string = input("Enter the location : ")

result = get_geolocation(api_key, search_string)

if result:
    print("Output:")
    for key, value in result.items():
        print(f"{key}: {value}")
```

BROWSE LOCATION IQ>SIGNUP WITH EMAIL> CLICK ON THE LINK PROVIDED BY LOCATION IQ (on your email)>YOU WILL GET YOUR ACCESS TOKEN COPY THE KEY AND PASTE IN THE PYTHON CODE:

The screenshot shows a web browser window with multiple tabs open at the top. The active tab is 'my.locationiq.com/dashboard#accesstoken'. The main content area is titled 'Manage your API Access Tokens'. A sidebar on the left lists various API categories: Playground, API Access Tokens (which is selected and highlighted in blue), Reports, Geocoding APIs, Routing APIs, Maps, Account, and Account details. The central content area contains a message: 'You need Access Tokens to use our APIs. We recommend creating one token per application or website. If you use them on public websites where anyone can see your code (websites, apps etc), rotate them often and use HTTP Referrer restrictions to limit abuse.' Below this is a yellow box stating, 'Your plan allows 1 Access Token and you've reached this limit. To create more tokens, please [upgrade your account](#)'. A table displays the single access token:

Label	Access Token	Created On	
Access Token 1	pk.71c93f03731ac10faf75d7e071df51c1	04 March 2024 07:55 AM UTC	VIEW LOGS

RUN THE CODE

Output:

```
= RESTART: C:\Users\Lab201\Desktop\geolocation.py
Enter the location : mumbai
Output:
place_id: 337978786
lat: 19.08157715
lon: 72.88662753964906
display_name: Mumbai, Maharashtra, India
```

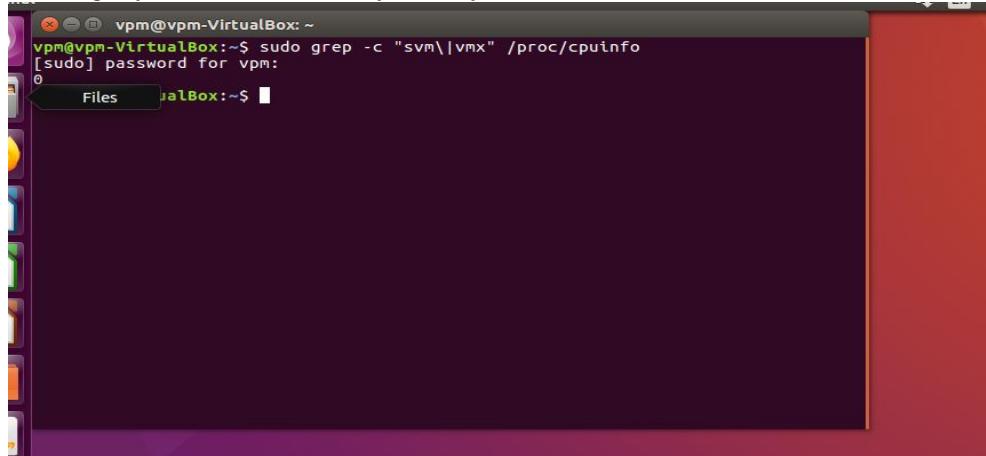
PRACTICAL NO – 5

Aim – Installation and configuration of virtualization using a KVM.

Open Ubuntu in virtual machine.

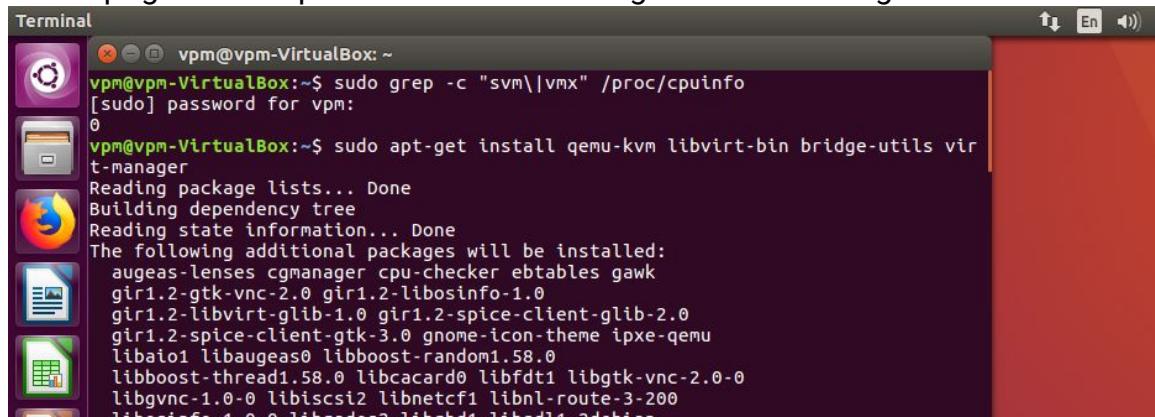
Now open the terminal and perform the following command

Sudo grep -c "svm\|vmx" /proc/cpuinfo



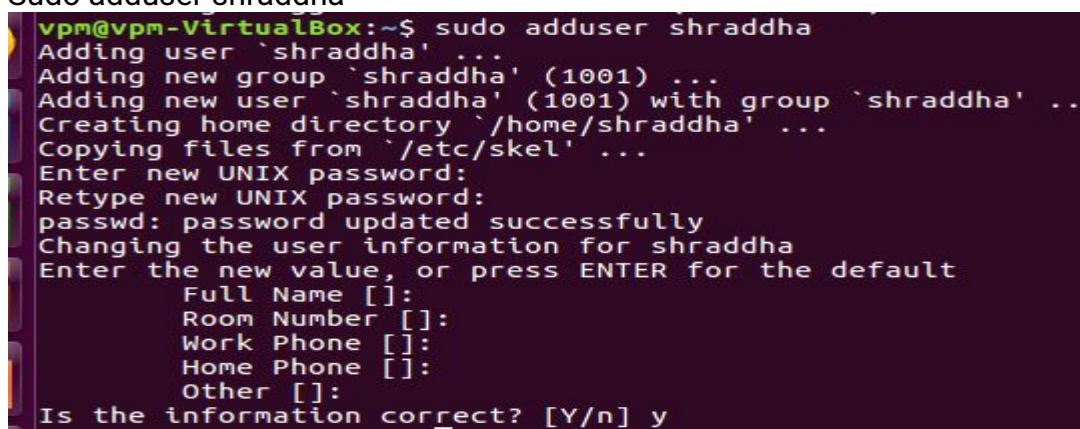
```
vpm@vpm-VirtualBox:~$ sudo grep -c "svm\|vmx" /proc/cpuinfo
[sudo] password for vpm:
0
```

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



```
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
  libaio1 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
  libosinfo1.0-0 librados2 librbd1 libSDL1.2debian
```

Sudo adduser shraddha



```
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
```

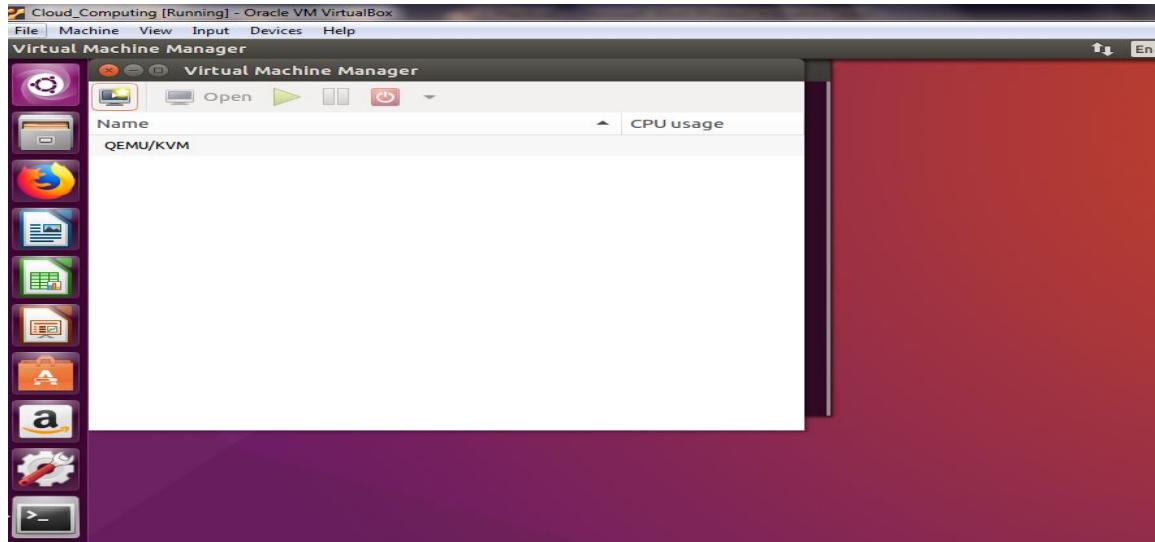
Sudo adduser shraddha libvirtd

```
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:~$
```

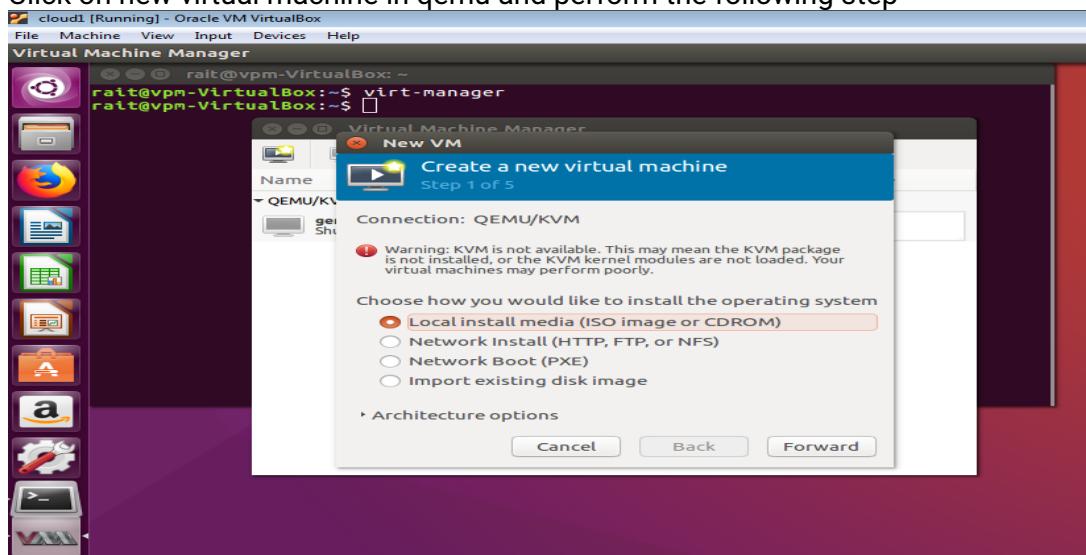
```
virsh -c qemu:///system list  
shraddha@vpm-VirtualBox:~$ virsh -c qemu:///system list  
  Id   Name           State  
-----  
shraddha@vpm-VirtualBox:~$
```

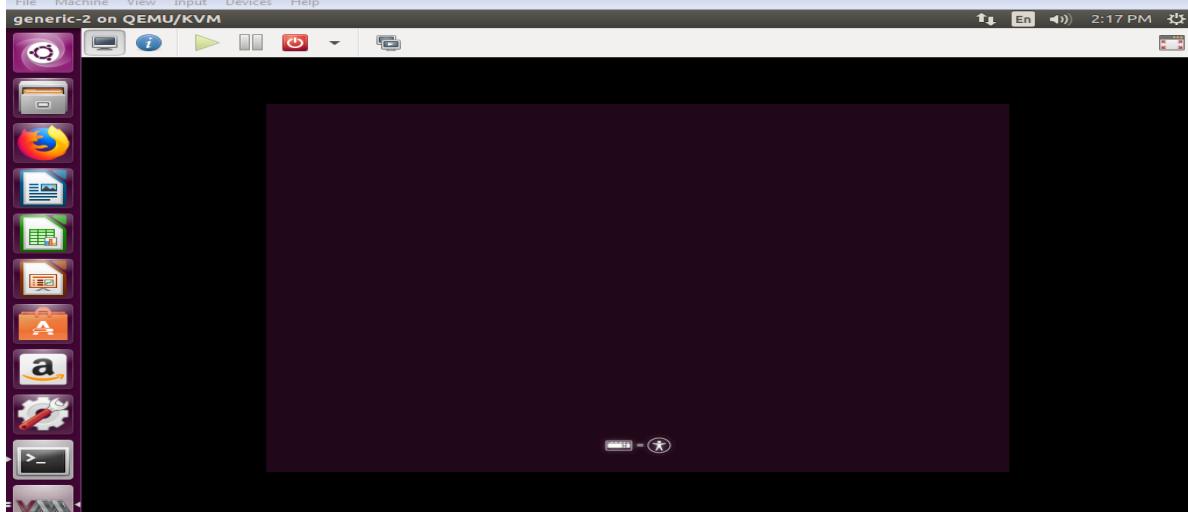
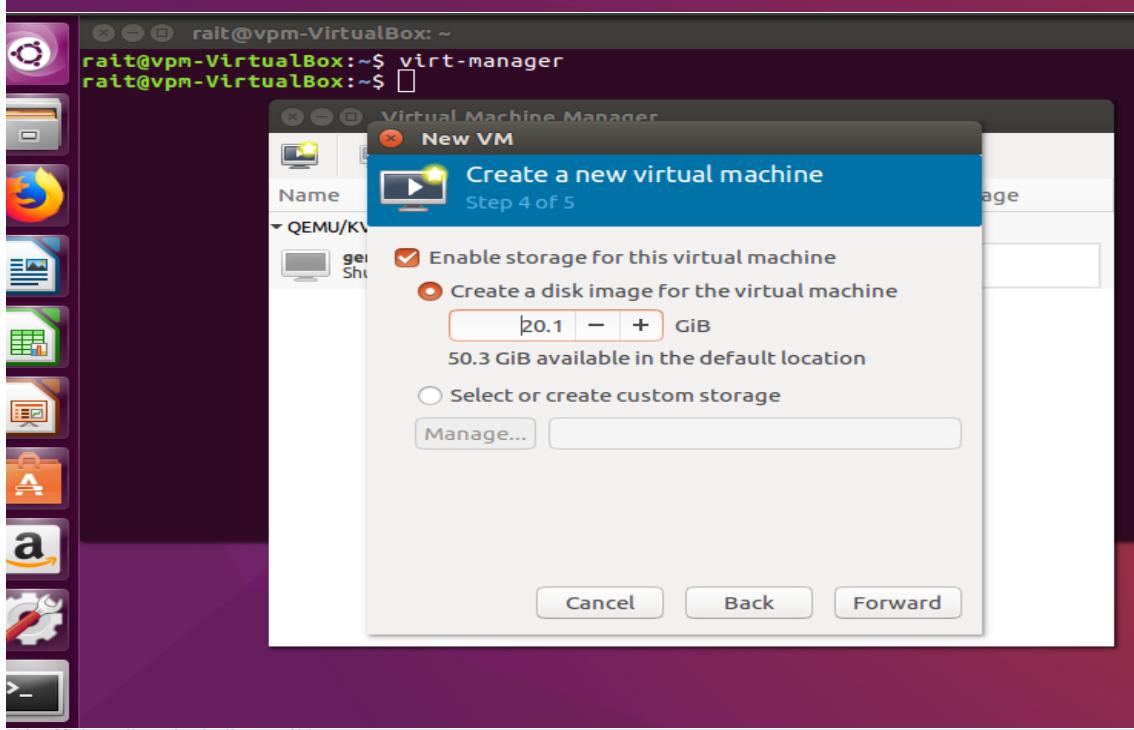
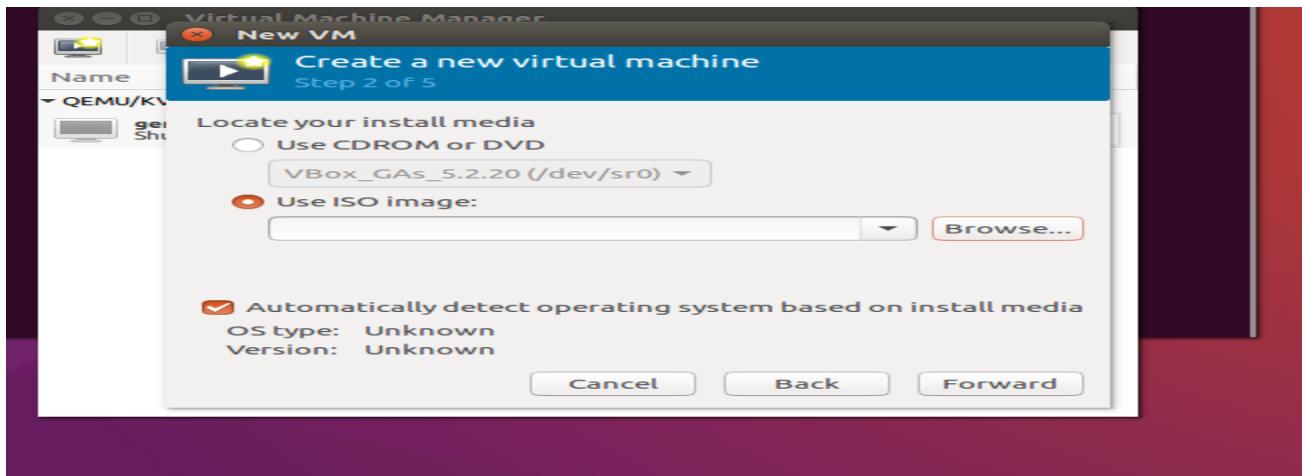
virt-manager

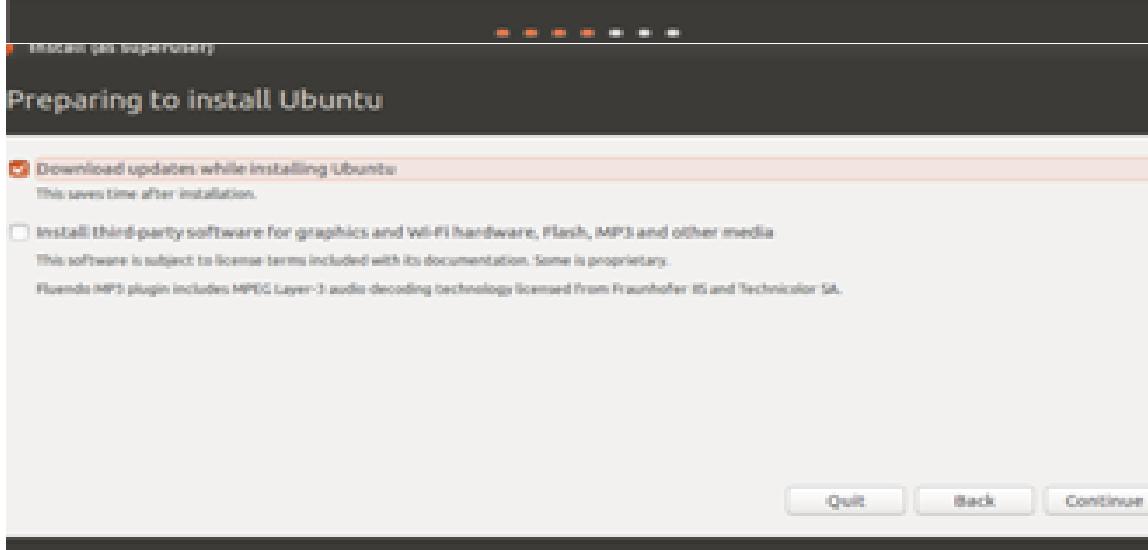
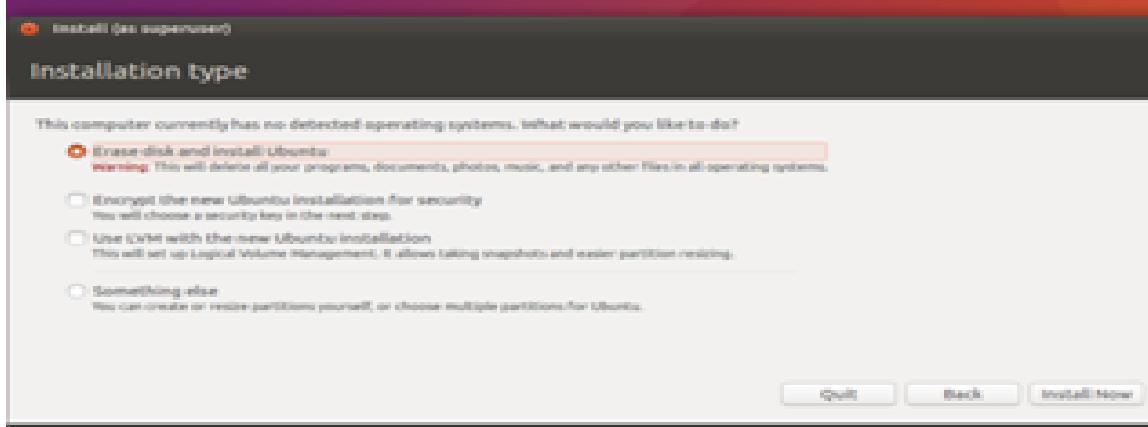
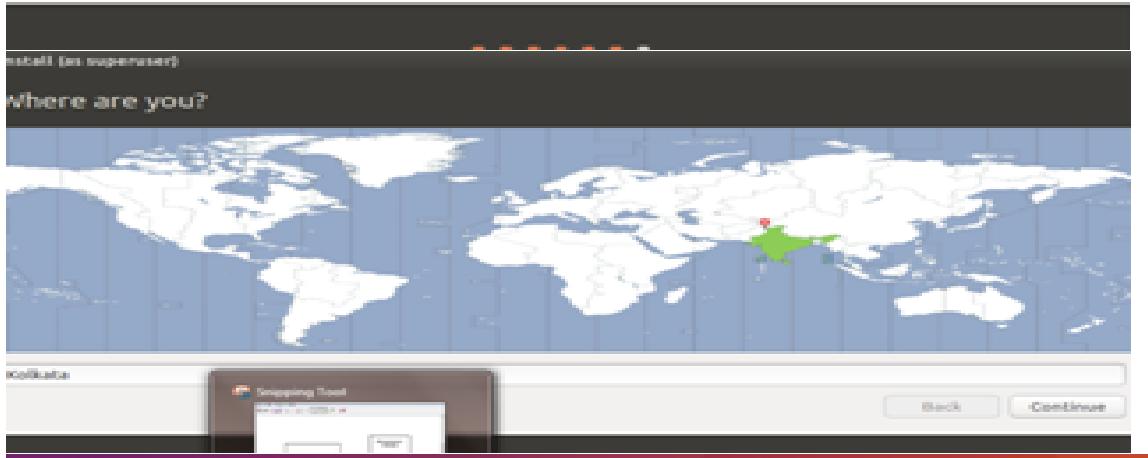
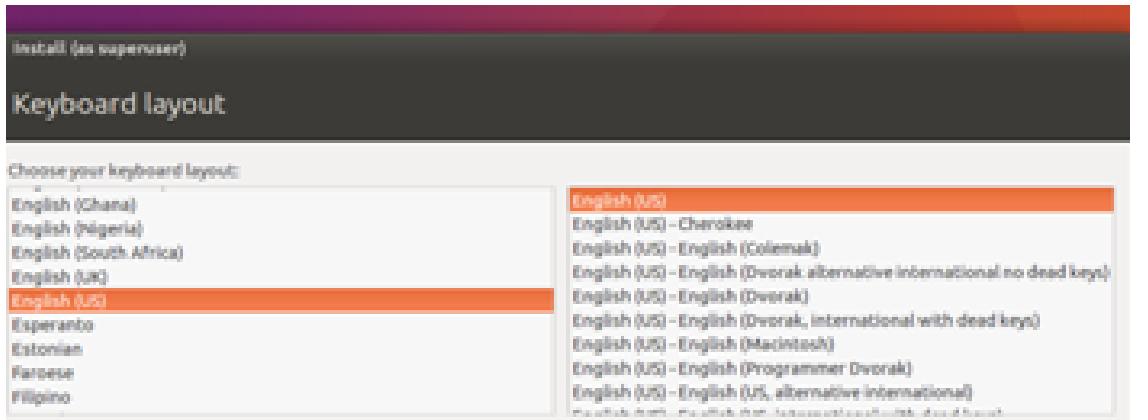
```
shraddha@vpm-VirtualBox:~$ virt-manager  
shraddha@vpm-VirtualBox:~$
```

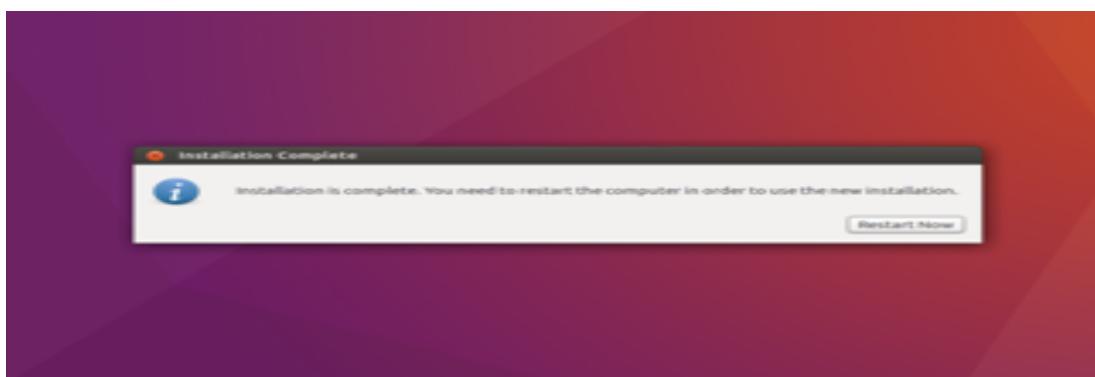
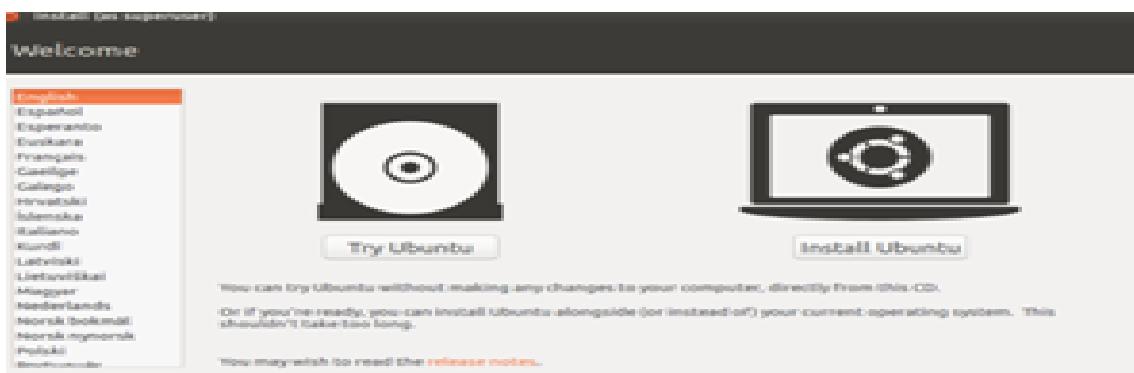


Click on new virtual machine in qemu and perform the following step

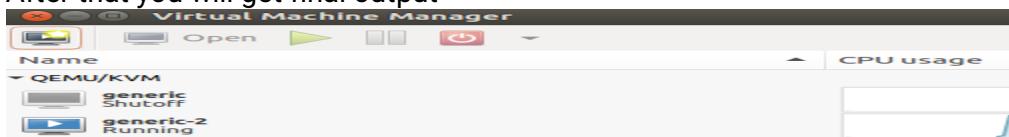




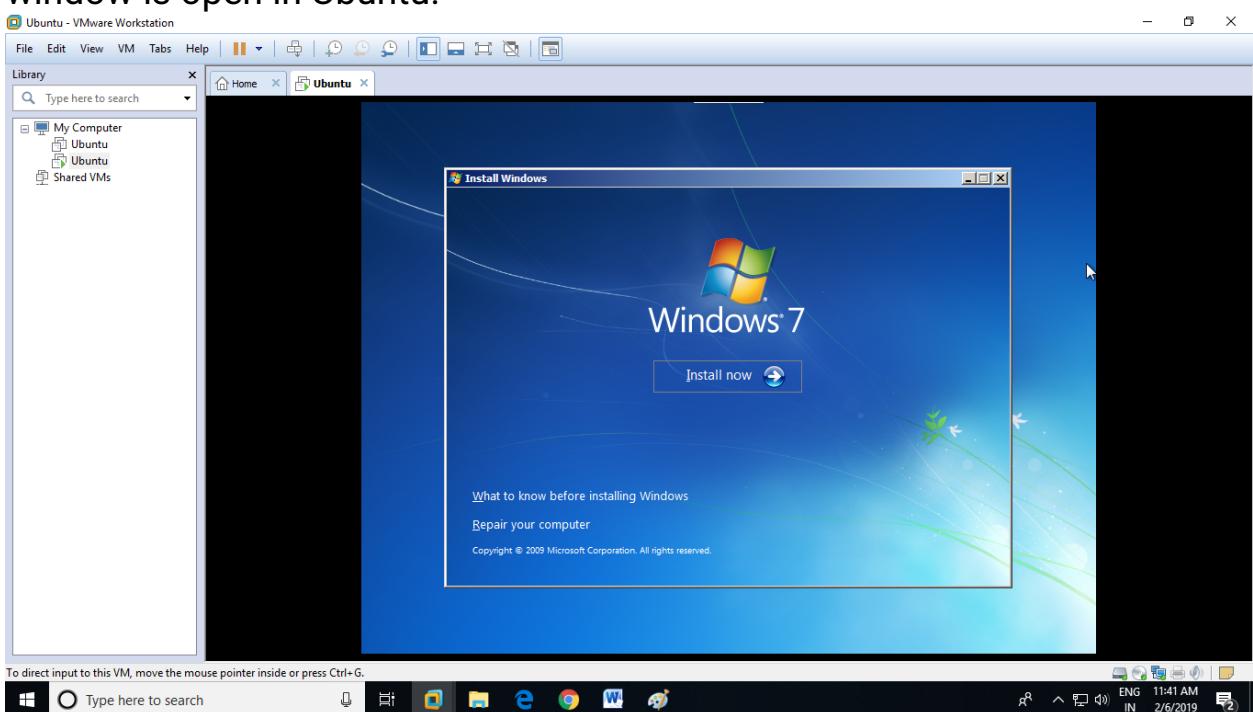




After that you will get final output



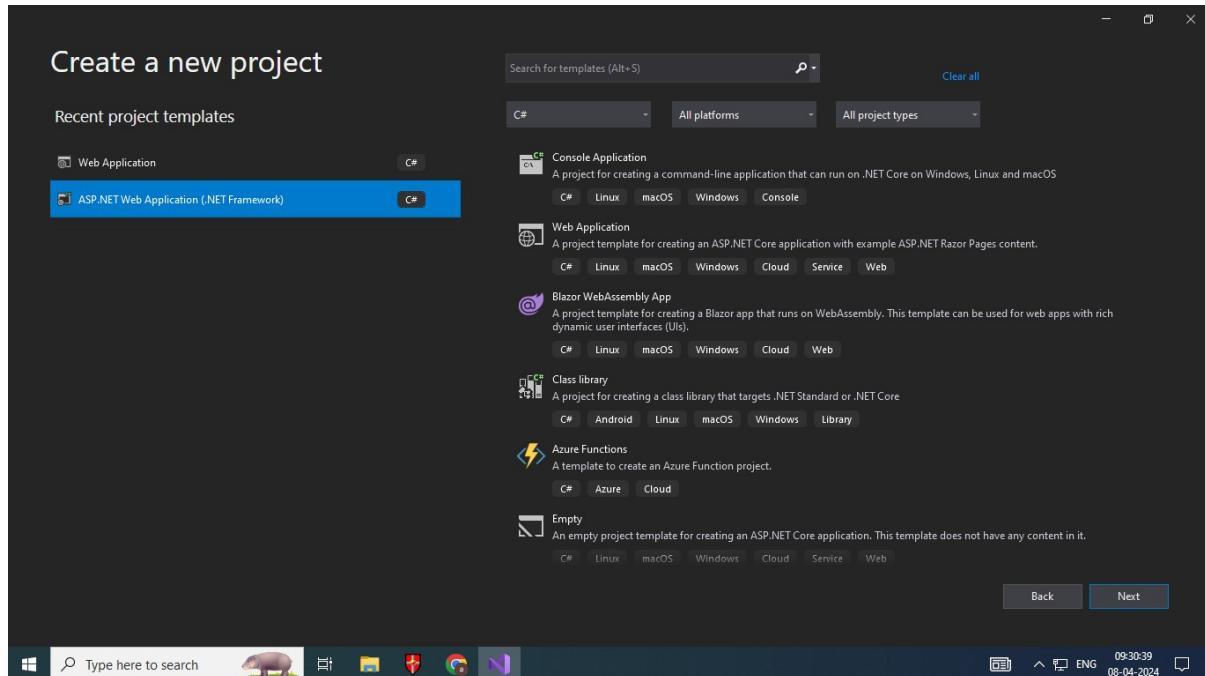
window is open in Ubuntu.



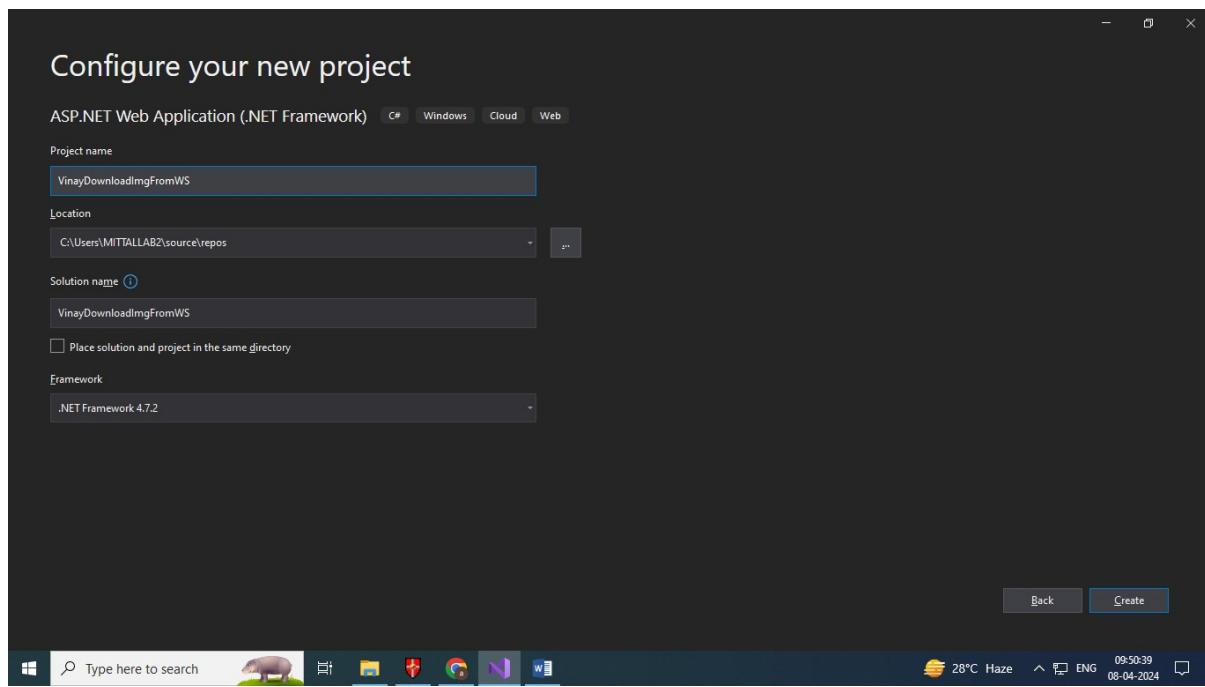
PRACTICAL NO – 6

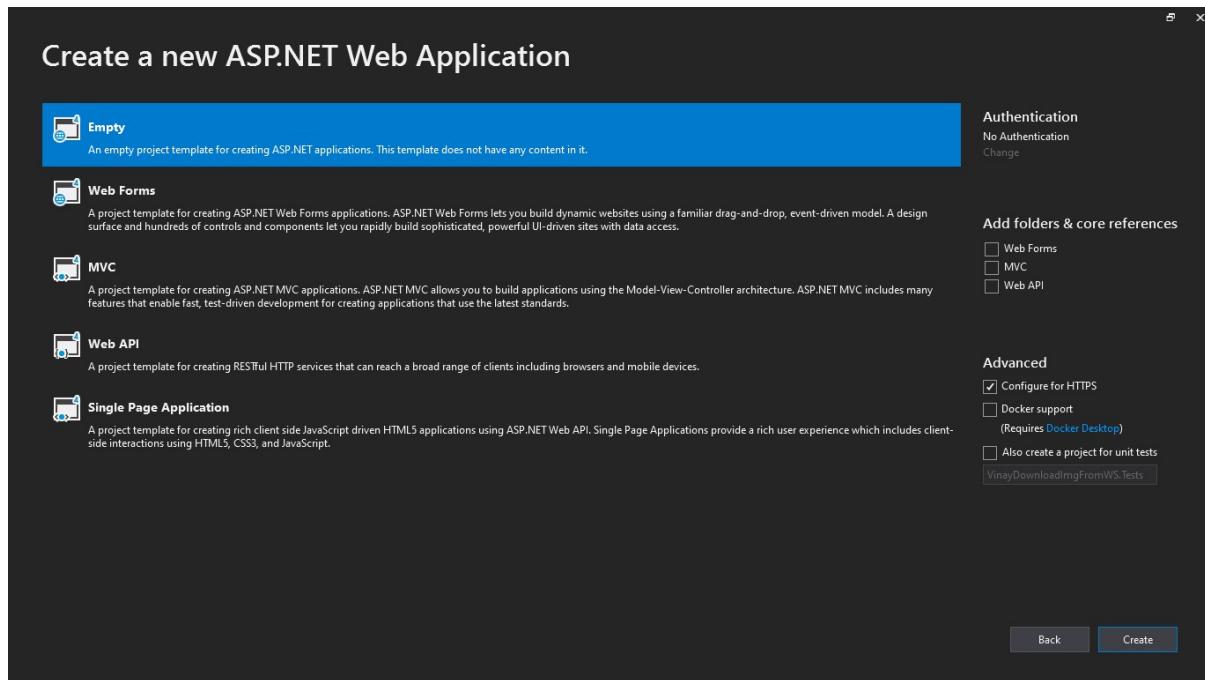
Aim – Download image from server.

1. Open Visual studio and create new project
2. Select ASP.NET Application (.NET Framework)

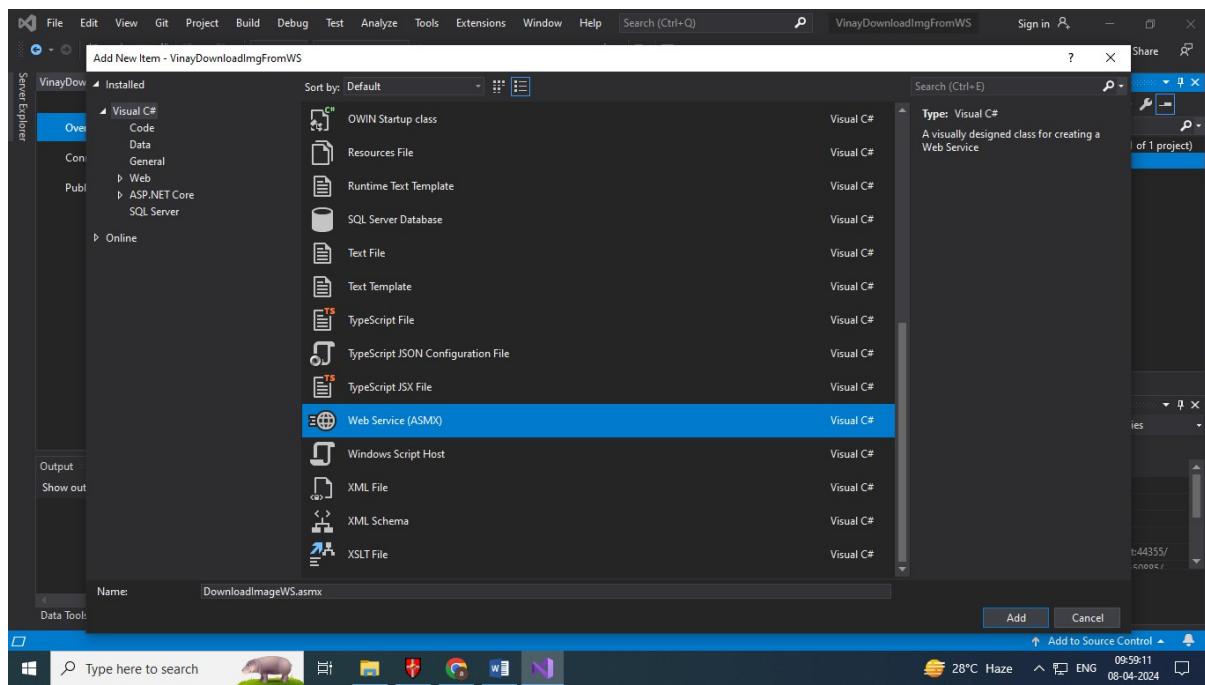


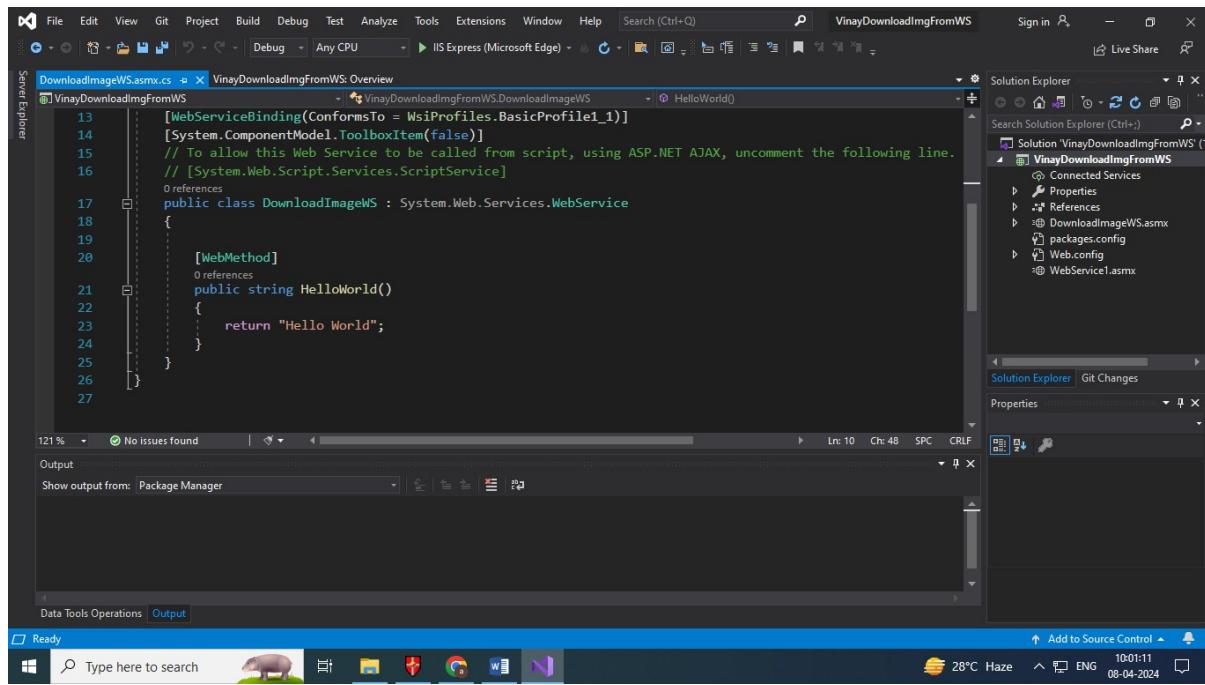
3. Give the project name.





4. After that open solution explorer → Right click on project → Add → New Item → webservice(ASMX) and give the name → Add





DownloadImageWS.asmx.cs file

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Linq;
using System.Web;
using System.Web.Services;

namespace VinayDownloadImgFromWS
{
    /// <summary>
    /// Summary description for DownloadImageWS
    /// </summary>
    [WebService(Namespace = "http://tempuri.org/")]
    [WebServiceBinding(ConformsTo =
        WsProfiles.BasicProfile1_1)]
    [System.ComponentModel.ToolboxItem(false)]
    // To allow this Web Service to be called from script, using ASP.NET AJAX,
    uncomment the following line.
    // [System.Web.Script.Services.ScriptService]
    public class DownloadImageWS : System.Web.Services.WebService
    {

        [WebMethod]
        public string HelloWorld()
        {
            return "Hello World";
        }

        [WebMethod, Description("Get Image
Content")]
        public byte[] GetImageFile(String
filename)
    }
```

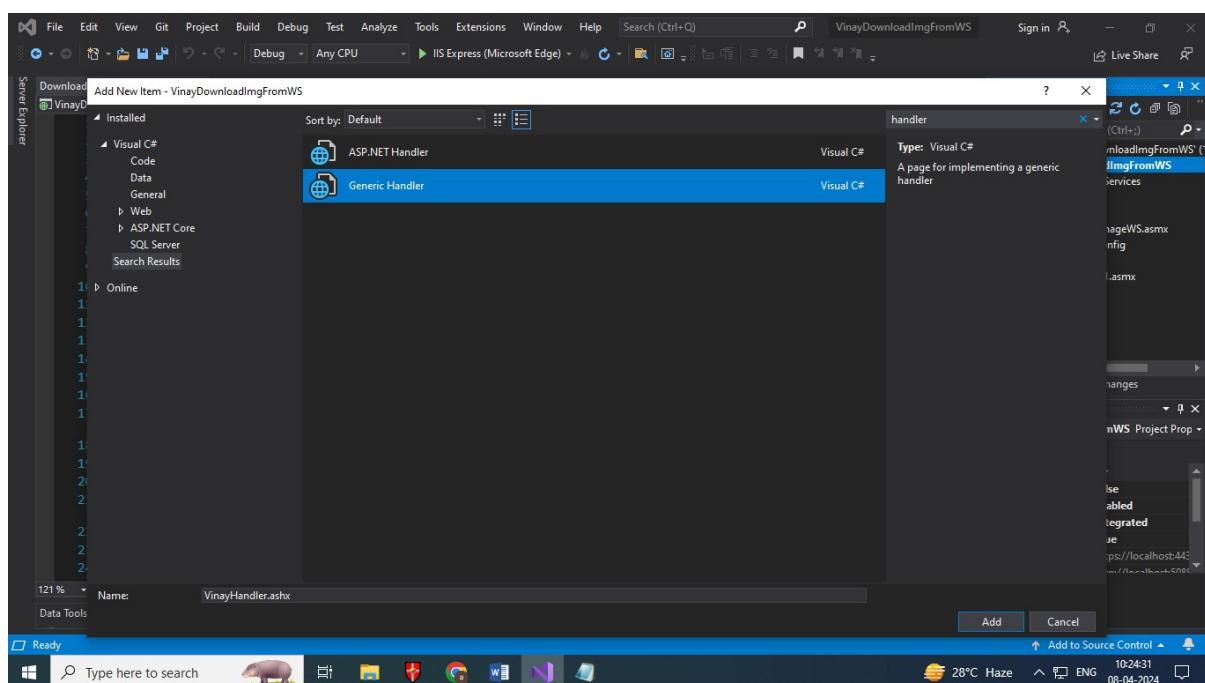
```
if (System.IO.File.Exists(Server.MapPath("~/Images/") + filename))
```

```

    {
        return System.IO.File.ReadAllBytes(Server.MapPath("~/Images") +
            filename);
    }
    else
    {
        return new byte[] { 0 };
    }
}
}

```

5. Now again right click on project → Add → new item → search handler → select generic handler and give name



VinayHandler.ashx.cs file

```

using System;
using
System.Collections.Generic;
using System.Linq;
using System.Web;

namespace VinayDownloadImgFromWS
{
    /// <summary>
    /// Summary description for VinayHandler
    /// </summary>
    public class VinayHandler : IHttpHandler
    {

        public void ProcessRequest(HttpContext context)
        {
            DownloadImageWS ws = new DownloadImageWS();
            byte[] binImage =
                ws.GetImageFile(context.Request["filename"]);if

```

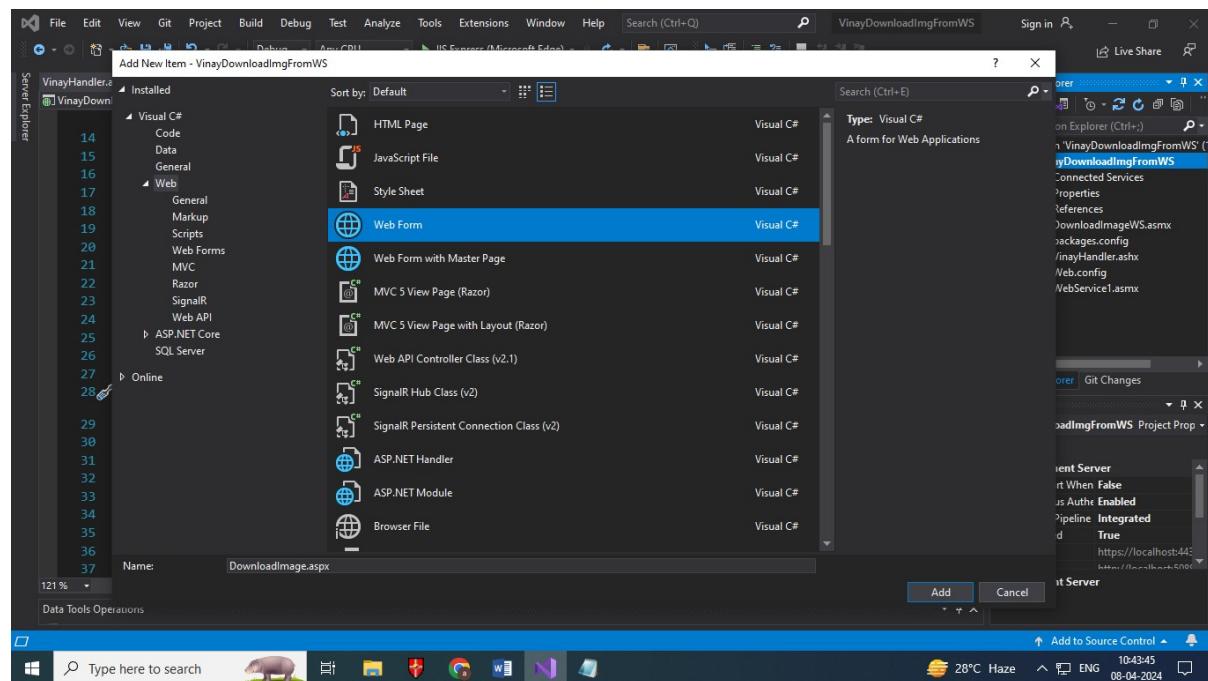
```

        }
    else
    {
        context.Response.ContentType =
            "image/jpeg";
        context.Response.BinaryWrite(binImage);
    }
}

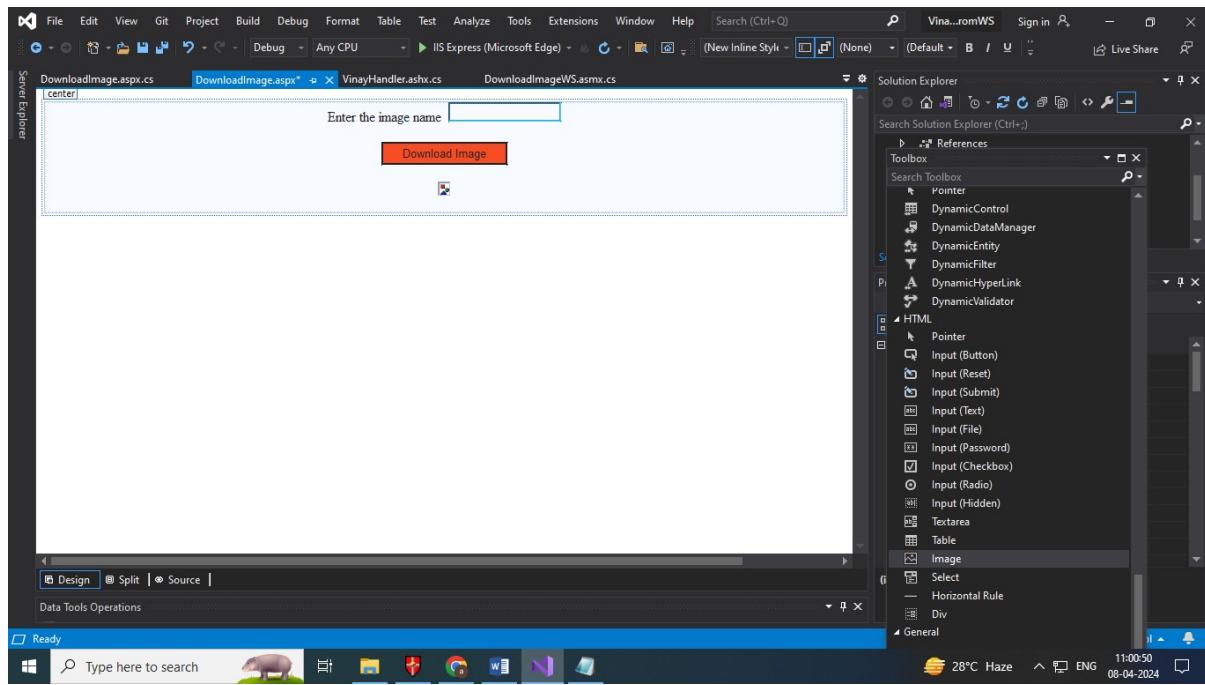
public bool IsReusable
{
    get
    {
        return false;
    }
}
}
}

```

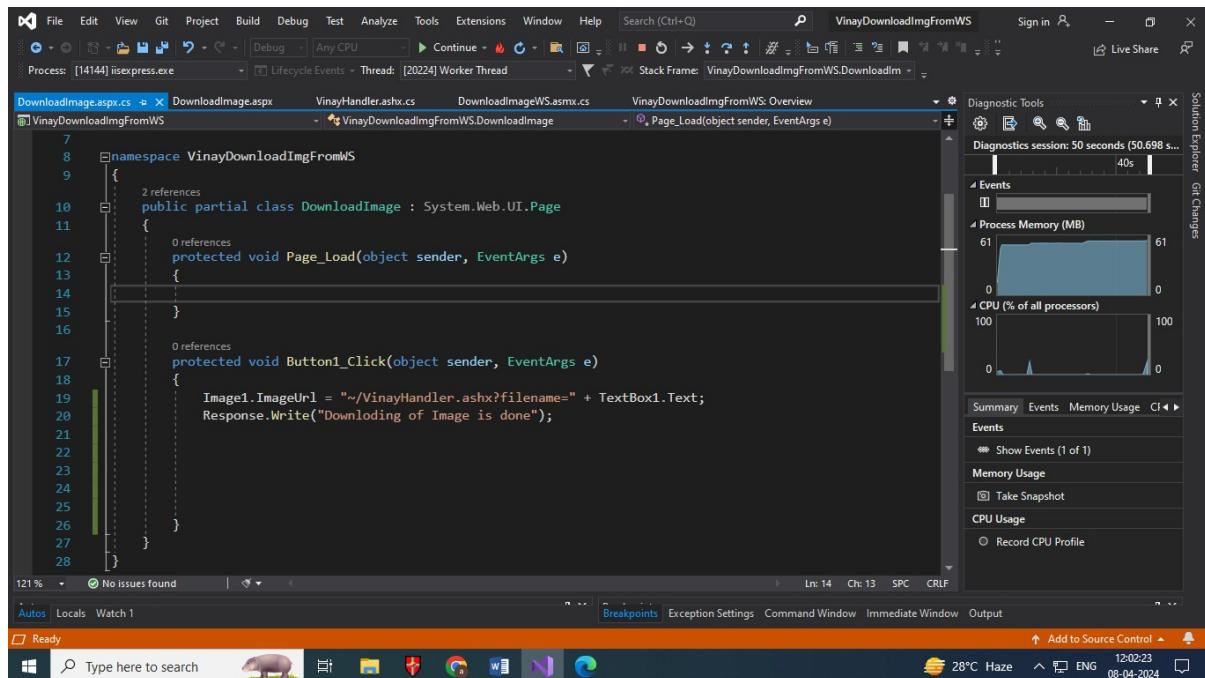
- Again right click on project → Add → new item → select web form → Give the name



- Click on View → toolbox → design the following form



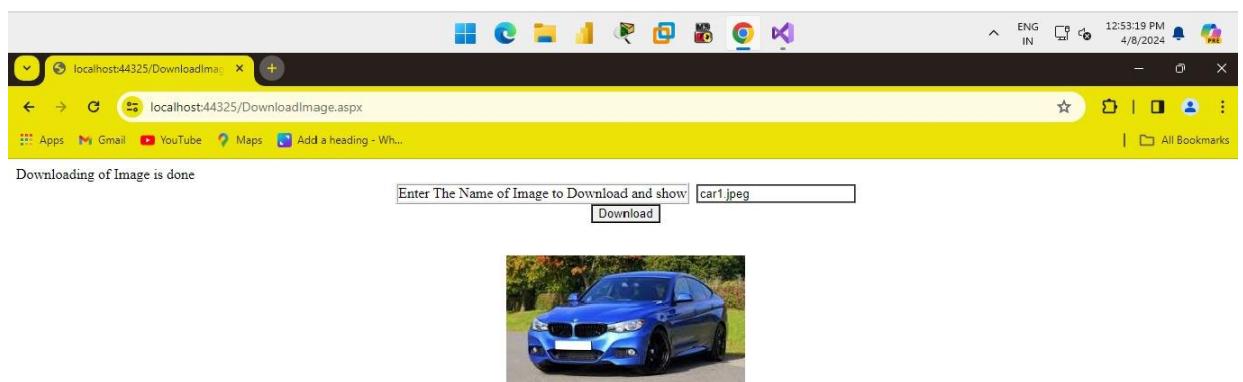
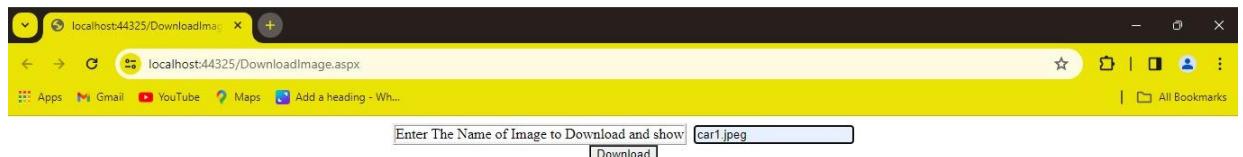
8. Now double click on button and change code



9. Write click on Project and create folder images and paste the images.

10. After that execute the code and Enter the Image name in Textfield.

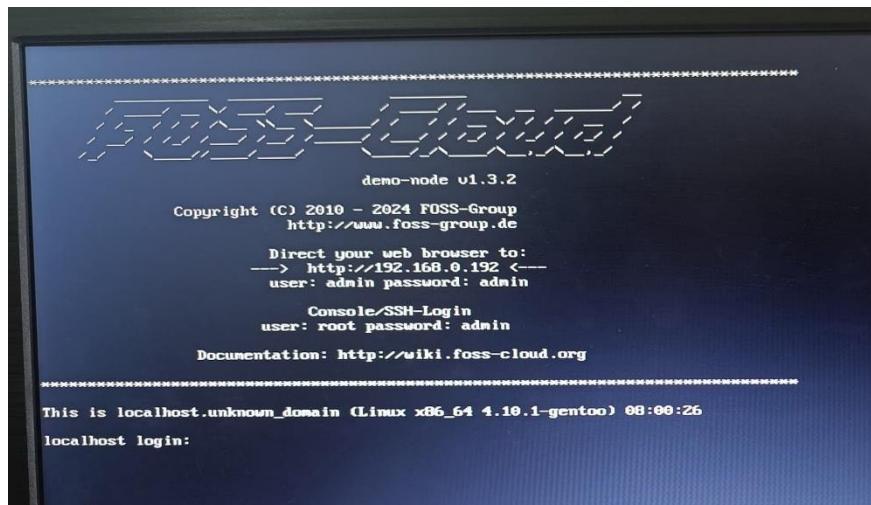
OUTPUT:



PRACTICAL NO – 7

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

1. Open the System where you have downloaded the FOSS- Cloud.
2. Enter user-root & Password-admin.



Now Right following command fc-node- configuration -n demo- system –password admin

A screenshot of a terminal window showing network configuration output and a fc-node configuration menu. The terminal shows statistics for interfaces like "em0" and "em1", and details for the "umbr0" interface. Below this, a series of "l" commands are listed, starting with "Demo-System Installation" and "Retrieving local network configuration ...", followed by "Local network configuration retrieval ok!". At the bottom, there is a note about unpacking a tarball and a Dell logo in the background.

3. Now enter ifconfig command.

```

localhost ~ #
localhost ~ #
localhost ~ #
localhost ~ # ifconfig
enp3s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 192.168.0.192 netmask 255.255.255.0 broadcast 192.168.0.255
                ether d8:5e:d3:4a:d9:c7 txqueuelen 1000 (Ethernet)
                RX packets 1019 bytes 95249 (93.0 KiB)
                RX errors 0 dropped 0 overruns 0 frame 0
                TX packets 480 bytes 43354 (42.3 KiB)
                TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

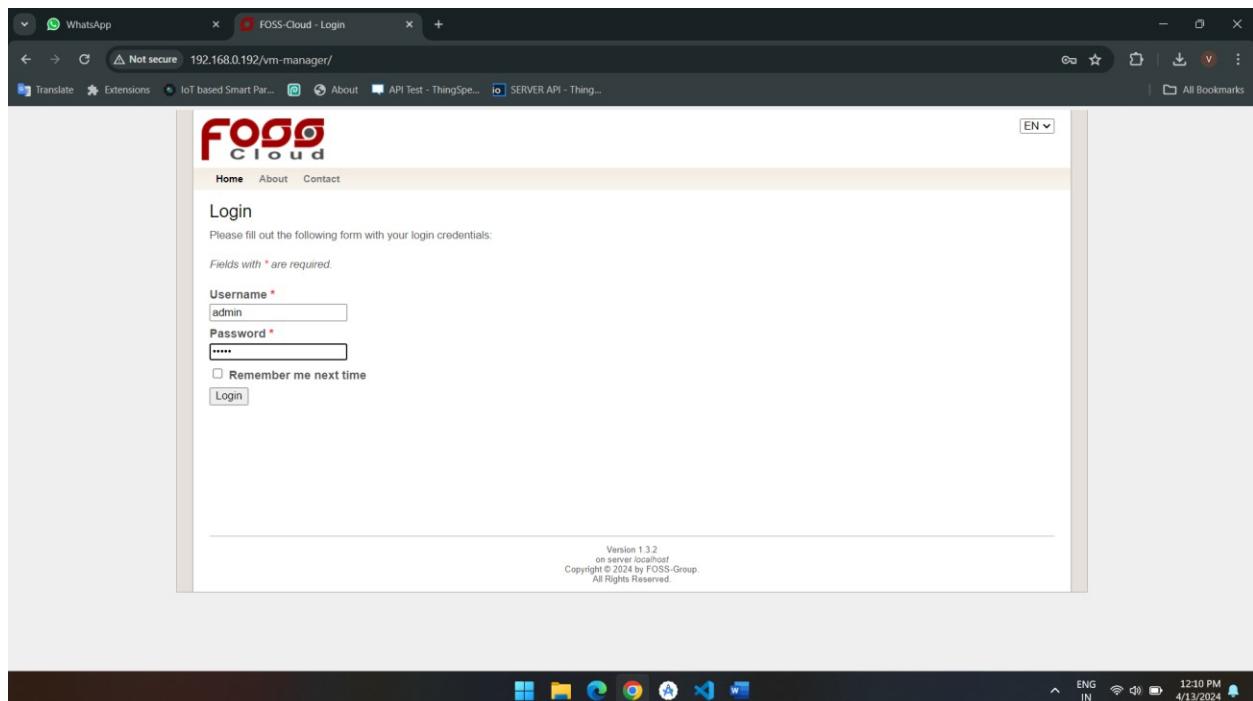
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
        inet 127.0.0.1 netmask 255.0.0.0
                loop txqueuelen 1000 (Local Loopback)
                RX packets 9111 bytes 2403555 (2.2 MiB)
                RX errors 0 dropped 0 overruns 0 frame 0
                TX packets 9111 bytes 2403555 (2.2 MiB)
                TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

umbr0: flags=4419<UP,BROADCAST,RUNNING,PROMISC,MULTICAST> mtu 1500
        inet 172.31.255.1 netmask 255.255.255.0 broadcast 172.31.255.255
                ether 22:3c:91:21:2b:ed txqueuelen 1000 (Ethernet)
                RX packets 0 bytes 0 (0.0 B)
                RX errors 0 dropped 0 overruns 0 frame 0
                TX packets 0 bytes 0 (0.0 B)
                TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

localhost ~ #

```

4. Now copy or save the IP Address (192.168.0.192)
5. Now open the browser of another device and type that IP address in new tab.



6. Username-admin & Password-admin

