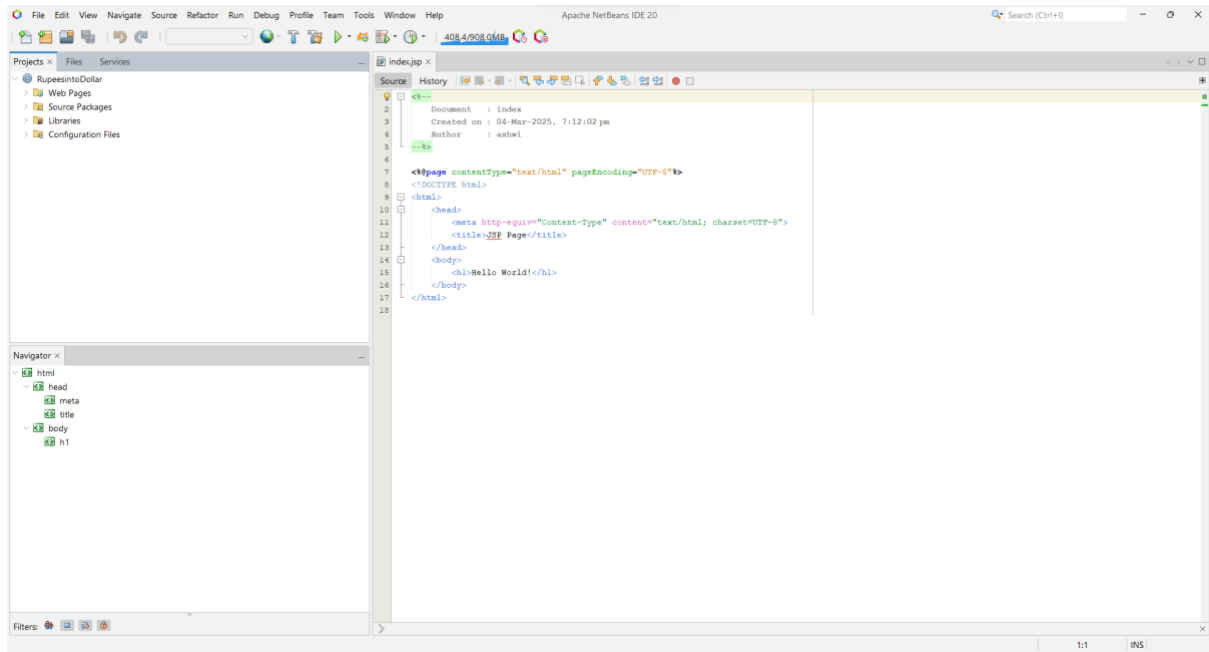


## Practical No. 1

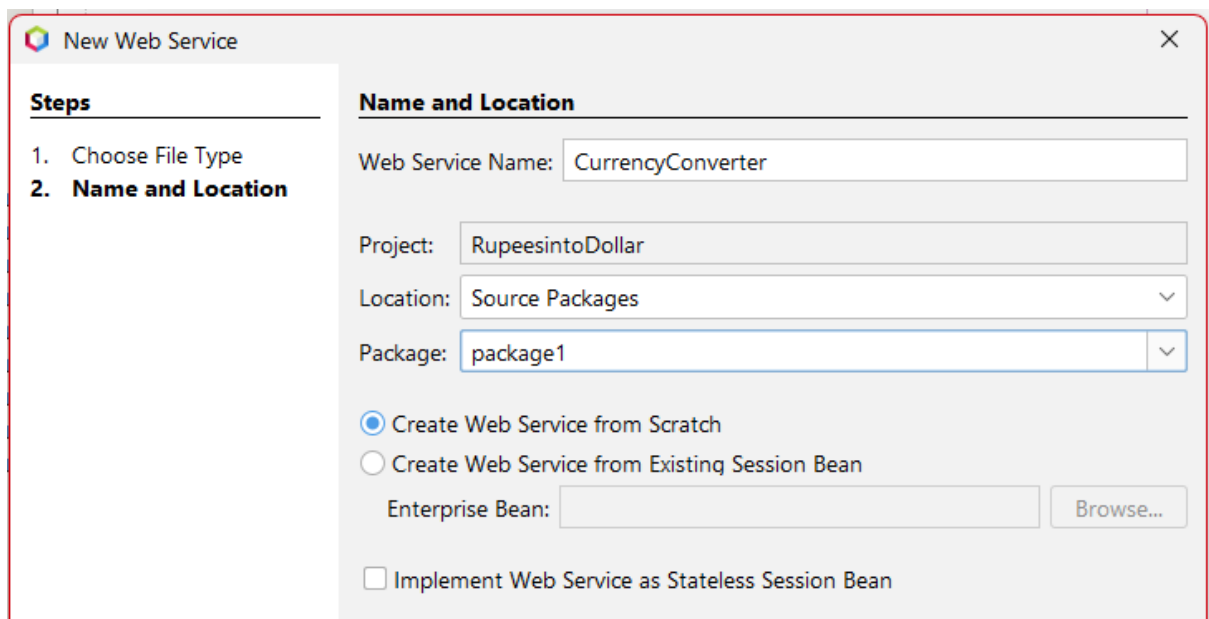
**Aim: Define a simple services like Converting Rupees into Dollar and Call it from different platform like JAVA and .NET**

**Step 1:** Open NetBeans and Create a new Project.

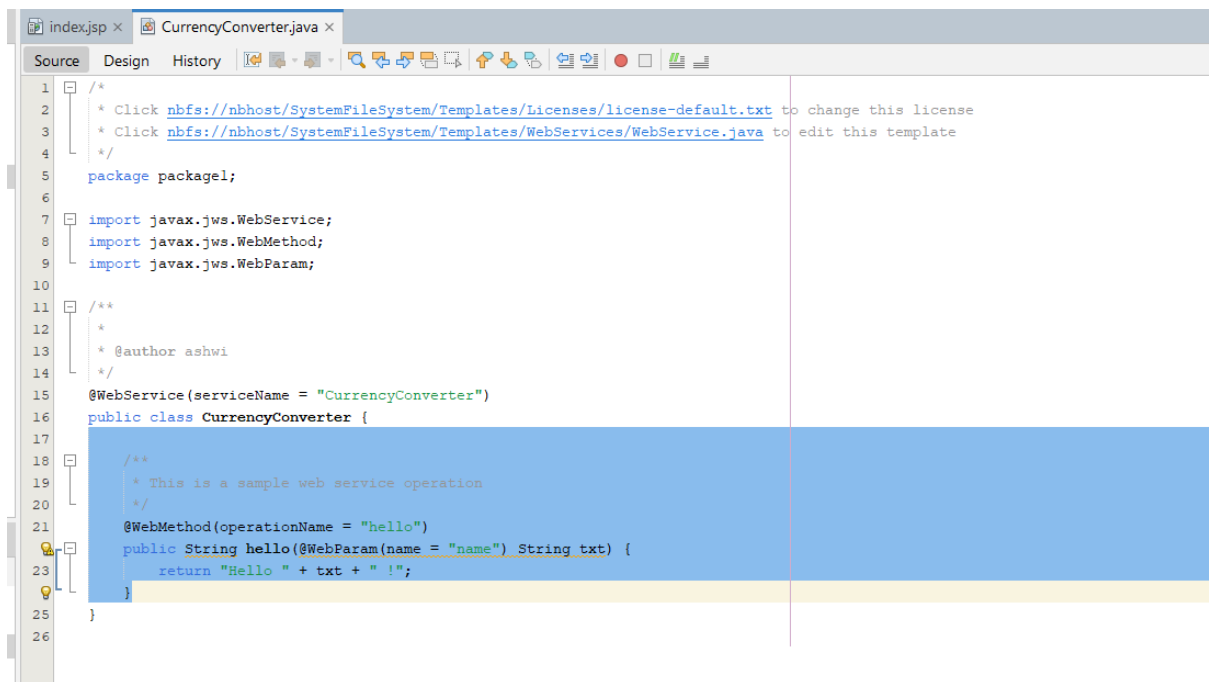
➤ Java Web > Web Application > Name the Project > Finish.



**Step 2:** Right click on the project > Create new web service

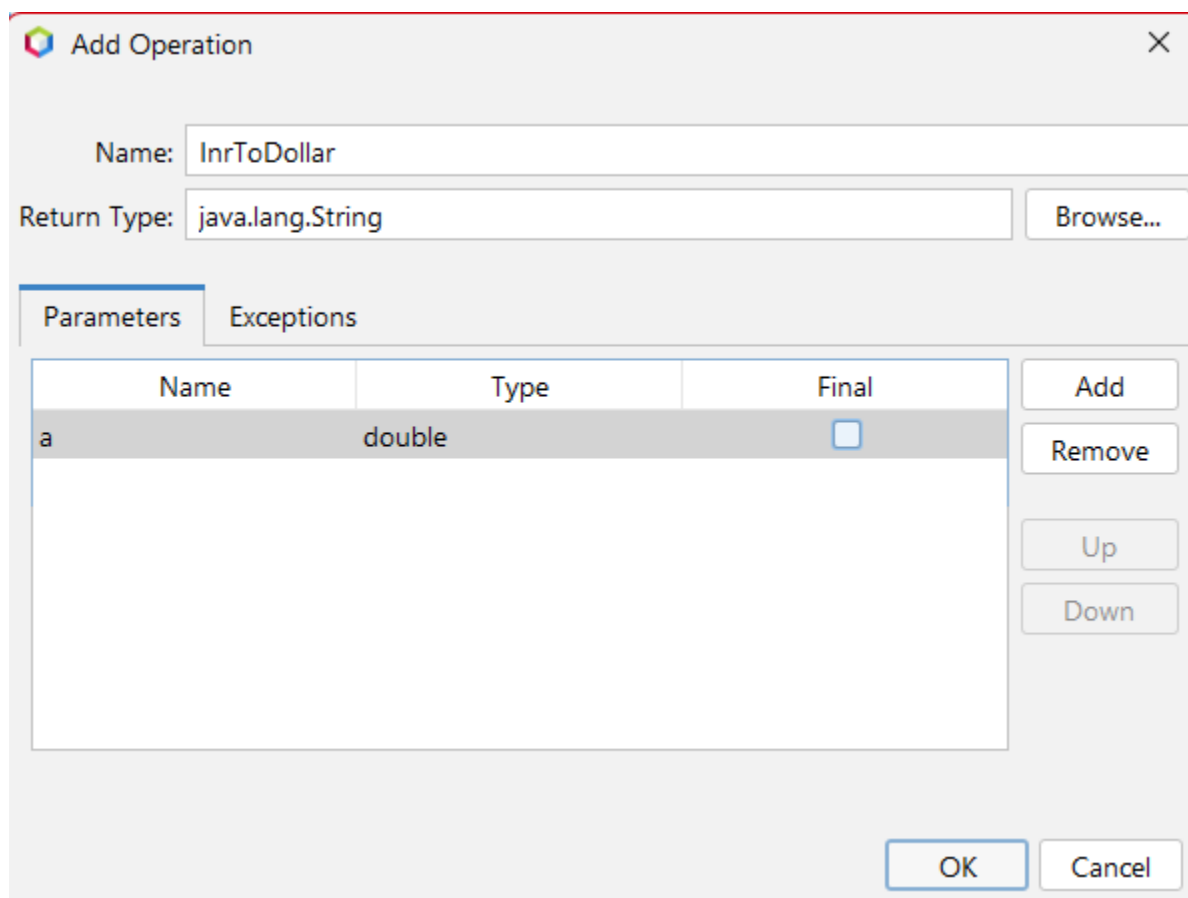


➤ Delete this block of code in the new web service.



```
1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3   * Click nbfs://nbhost/SystemFileSystem/Templates/WebServices/WebService.java to edit this template
4   */
5   package packagel;
6
7   import javax.jws.WebService;
8   import javax.jws.WebMethod;
9   import javax.jws.WebParam;
10
11  /**
12   *
13   * @author ashwi
14   */
15  @WebService(serviceName = "CurrencyConverter")
16  public class CurrencyConverter {
17
18      /**
19       * This is a sample web service operation
20       */
21      @WebMethod(operationName = "hello")
22      public String hello(@WebParam(name = "name") String txt) {
23          return "Hello " + txt + " !";
24      }
25  }
26
```

- Right click on the web service > Add Operation > Add a parameter and set it type as 'Double'.



**Add Operation**

Name:

Return Type:

**Parameters** **Exceptions**

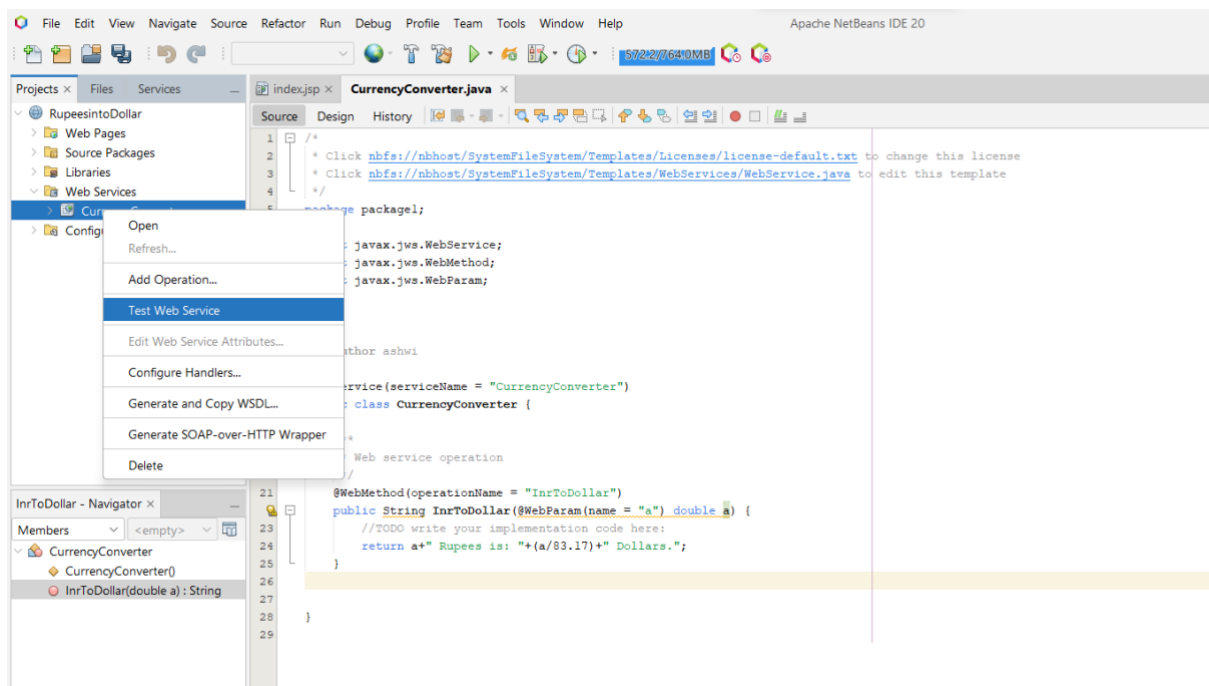
| Name | Type   | Final                    |
|------|--------|--------------------------|
| a    | double | <input type="checkbox"/> |

- Modify the Code of the function as given below:

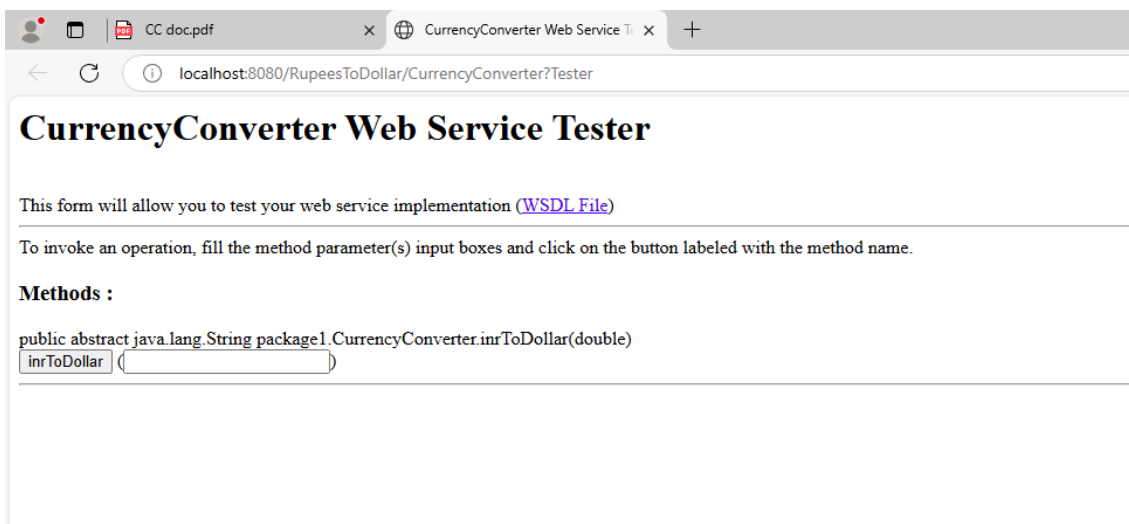
```
@WebMethod(operationName = "InrToDollar")
public String InrToDollar(@WebParam(name = "a") double a) {
    //TODO write your implementation code here:
    return a+" Rupees is: "+(a/83.17)+" Dollars.";
}
```

### Step 3: Test the Web Service.

- Right click on web service file > Test web service.



- Enter values to check the web service.



Method invocation trace

localhost:8080/RupeesToDollar/CurrencyConverter?Tester

### inrToDollar Method invocation

**Method parameter(s)**

| Type   | Value |
|--------|-------|
| double | 10    |

**Method returned**

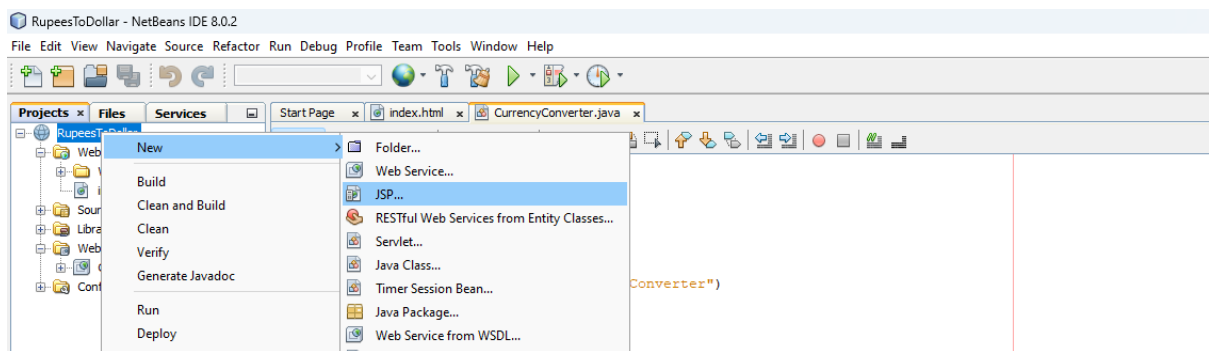
java.lang.String : "10.0 Rupees is 0.12023566189731874 Dollars."

**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:InrToDollar xmlns:ns2="http://Package1/">
      <a>10.0</a>
    </ns2:InrToDollar>
  </S:Body>
</S:Envelope>
```

#### Step 4: Create two JSP files names 'Input' and 'Outout'

- Right click on Web pages > new > JSP



New JSP

**Steps**

1. Choose File Type
2. **Name and Location**

**Name and Location**

File Name:

Project:

Location:

Folder:

Created File:

Options:

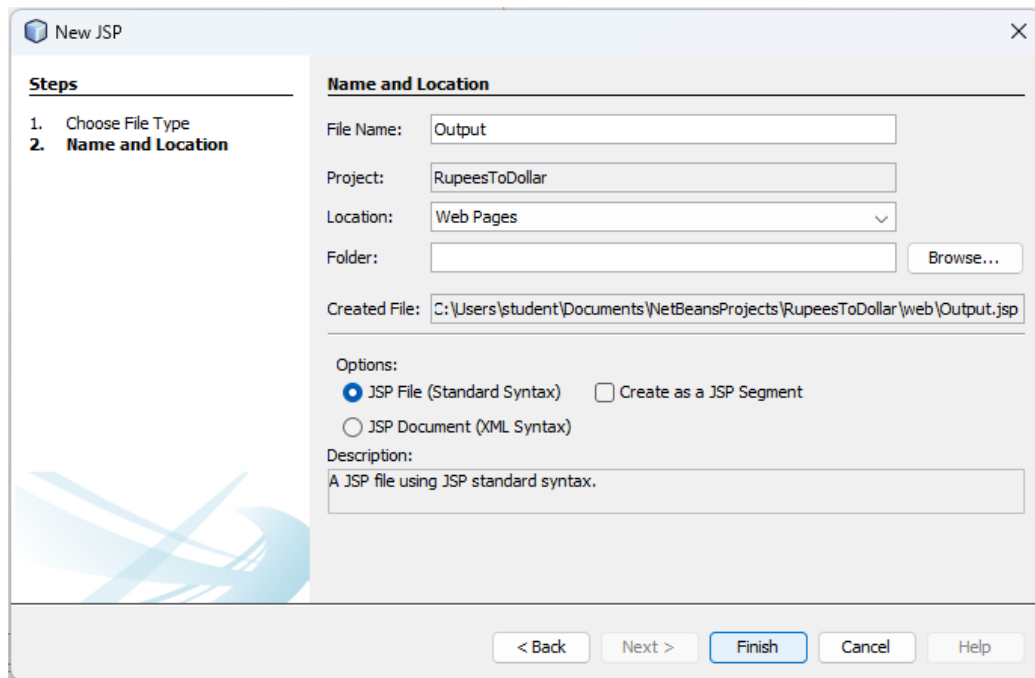
☒ JSP File (Standard Syntax) ☐ Create as a JSP Segment

☐ JSP Document (XML Syntax)

Description:

A JSP file using JSP standard syntax.

< Back Next > Finish Cancel Help



**Step 5:** Enter the following code in 'Input.jsp':

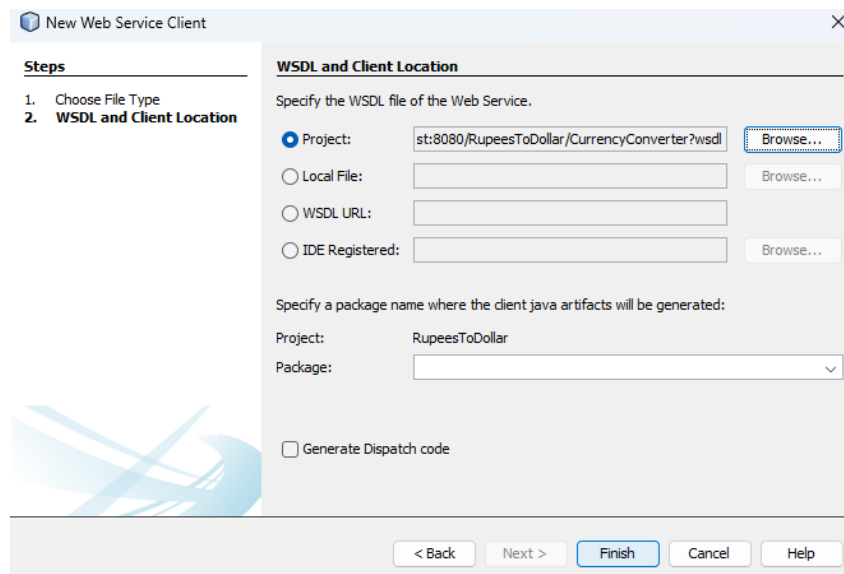
```

Start Page x index.html x CurrencyConverter.java x Inout.jsp x Output.jsp x
Source History
1 <%--
2     Document    : Inout
3     Created on  : 5 Mar, 2025, 8:57:35 AM
4     Author     : student
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         Enter the Rupees to Convert: <input type="text" name="t1">
17         <input type="submit"> <input type="reset">
18     </form>
19 </body>
20 </html>
21

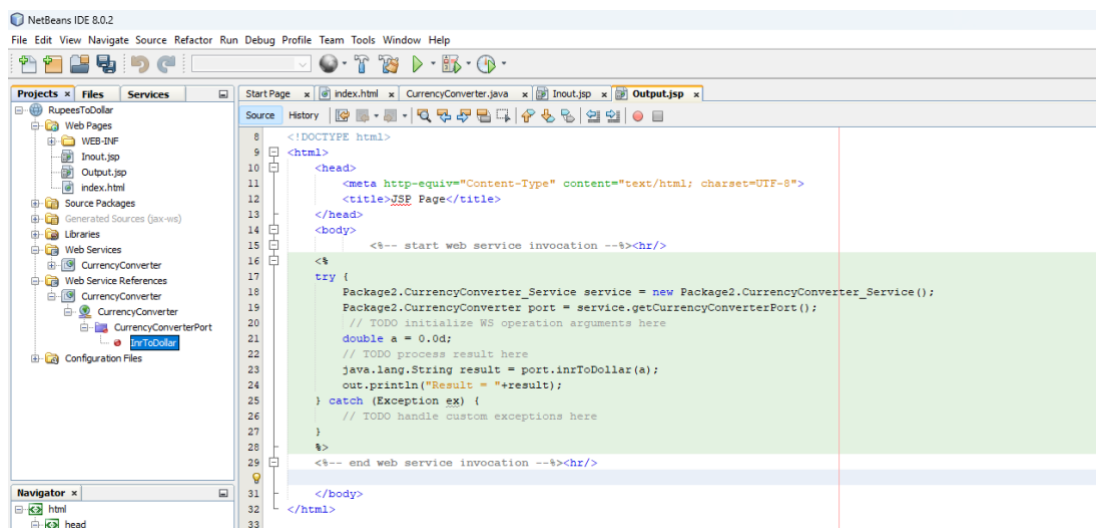
```

**Step 6:** Create a new Web Service Client.

- Right click on the project > new > Web Service Client > browse > Select The Web Service > Finish.



**Step7:** Expand the ‘Web Service References’ tree > Drag the Button into the Body tag of the ‘Output.jsp’ file.

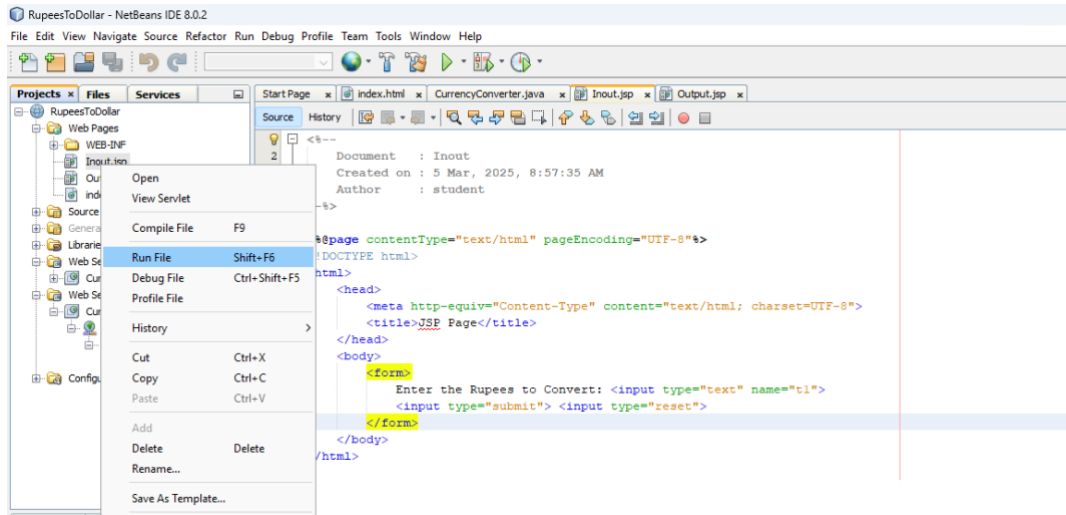


➤ Modify the code as given below:

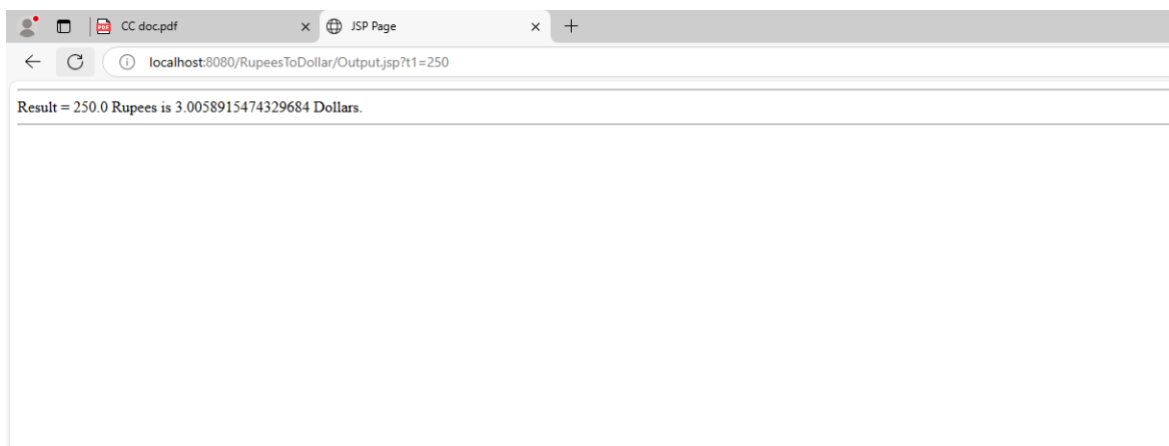
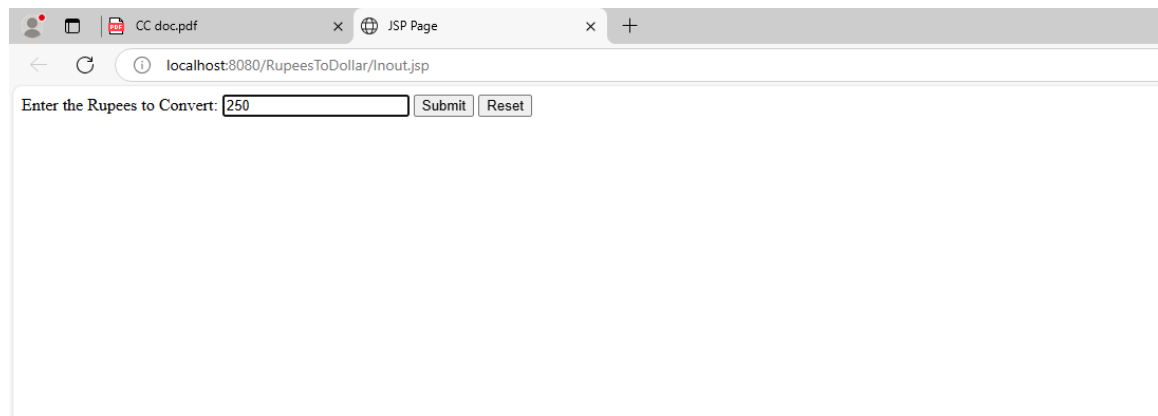
```
try {
    Package2.CurrencyConverter_Service service = new Package2.CurrencyConverter_Service();
    Package2.CurrencyConverter port = service.getCurrencyConverterPort();
    // TODO initialize WS operation arguments here
    double a = Double.parseDouble(request.getParameter("t1"));
    // TODO process result here
    java.lang.String result = port.inrToDollar(a);
```

**Step 8:** Build and run the project.

➤ Build the Project > Right Click on the ‘Input.jsp’ file > Run.



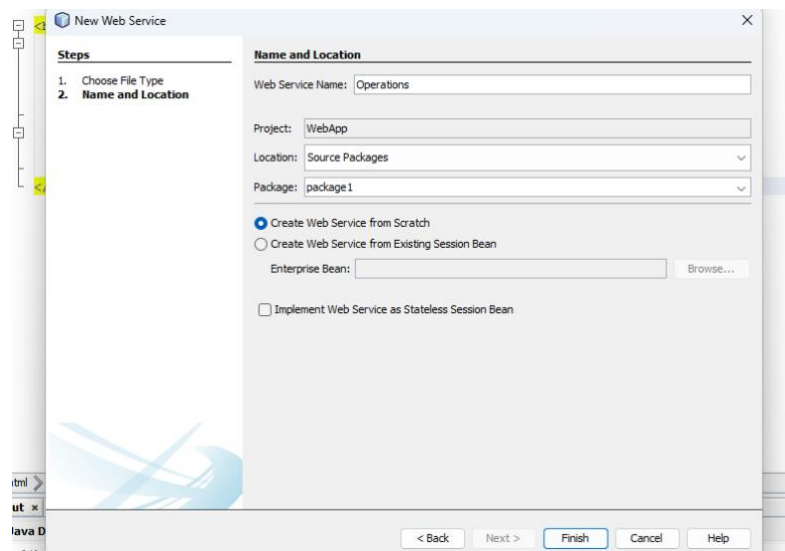
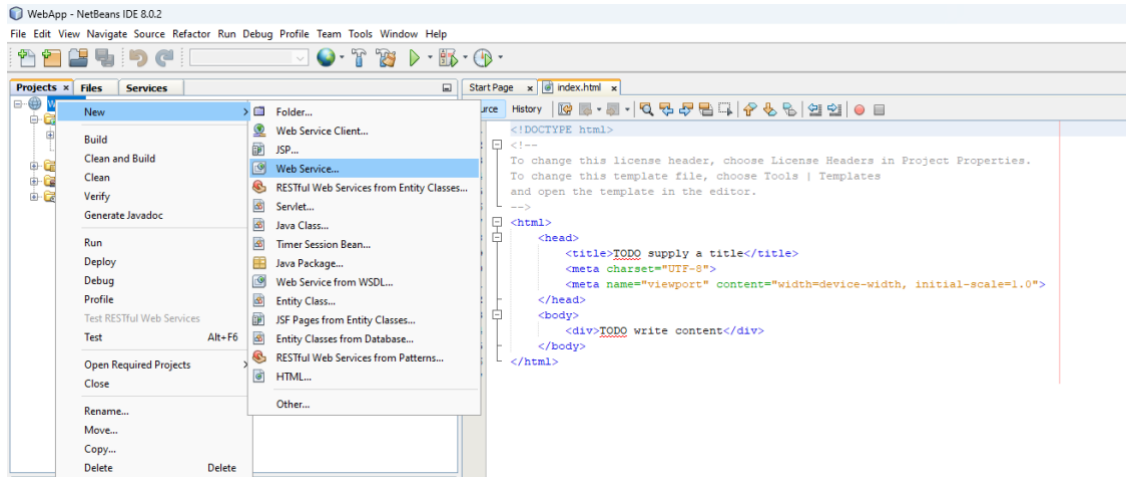
## Output:



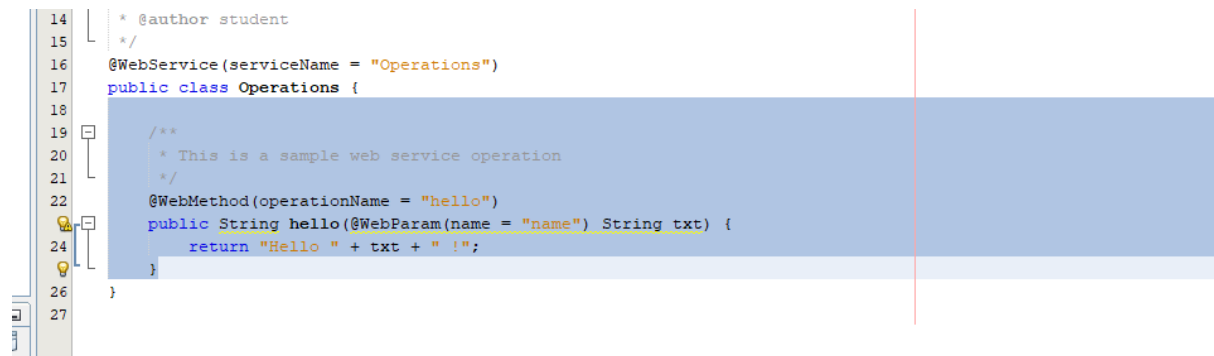
## Practical No. 2

**Aim: Create a Simple SOAP service.**

**Step 1:** Create a new Web Application in NetBeans > Create New web service and name it 'Operations'.



**Step 2:** Delete the Block of code given below:





- Right click on the Web Service > Add Operation > Give Operation name as 'Addition' > Choose Return Type 'int' > Add two Parameters with Type 'int'.

**Add Operation**

Name: Addition

Return Type: int Browse...

Parameters Exceptions

| Name | Type | Final                    |
|------|------|--------------------------|
| n1   | int  | <input type="checkbox"/> |
| n2   | int  | <input type="checkbox"/> |

Add Remove Up Down

OK Cancel

This automatically creates an Addition Operation.

```

1  /*
2   * To change this license header, choose License Headers in Project Properties.
3   * To change this template file, choose Tools | Templates
4   * and open the template in the editor.
5   */
6  package packagel;
7
8  import javax.ws.WebService;
9  import javax.ws.WebMethod;
10 import javax.ws.WebParam;
11
12 /**
13  *
14  * @author student
15  */
16 @WebService(serviceName = "Operations")
17 public class Operations {
18
19     /**
20      * Web service operation
21      */
22     @WebMethod(operationName = "Addition")
23     public int Addition(@WebParam(name = "n1") int n1, @WebParam(name = "n2") int n2) {
24         //TODO write your implementation code here:
25         return 0;
26     }
27
28 }

```

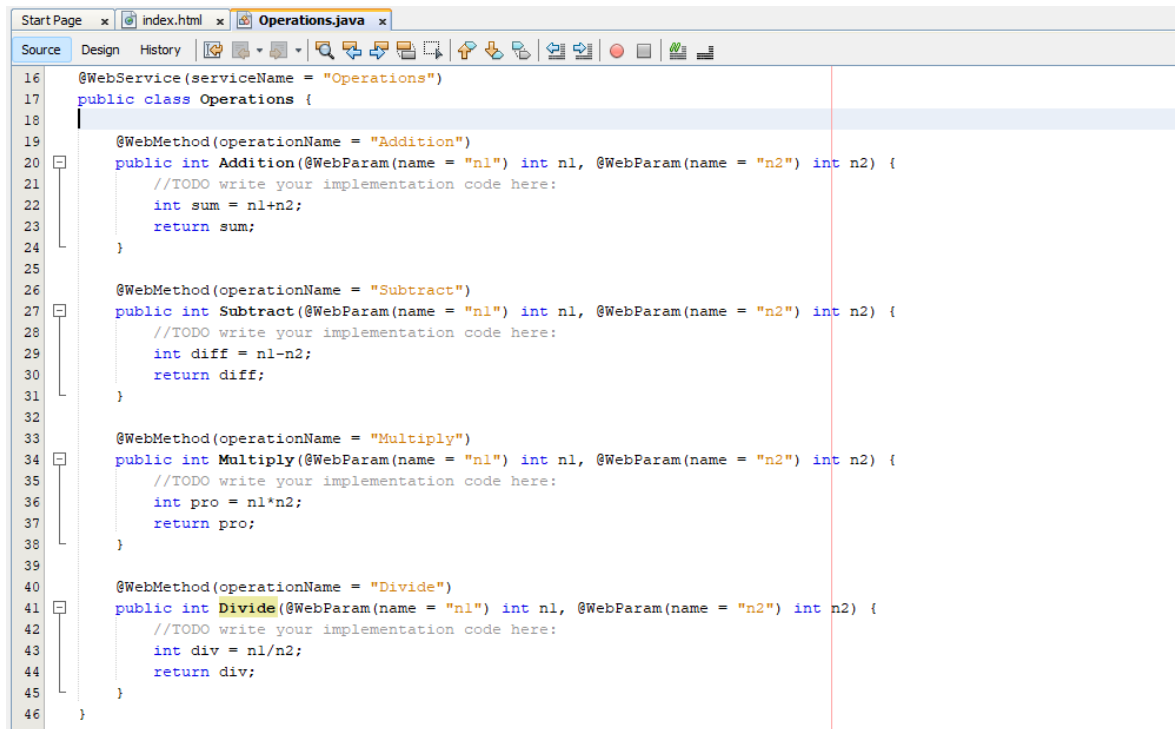
- Modify the code of the Operation as given below:

```

1  @WebMethod(operationName = "Addition")
2  public int Addition(@WebParam(name = "n1") int n1, @WebParam(name = "n2") int n2) {
3      //TODO write your implementation code here:
4      int sum = n1+n2;
5      return sum;
6  }

```

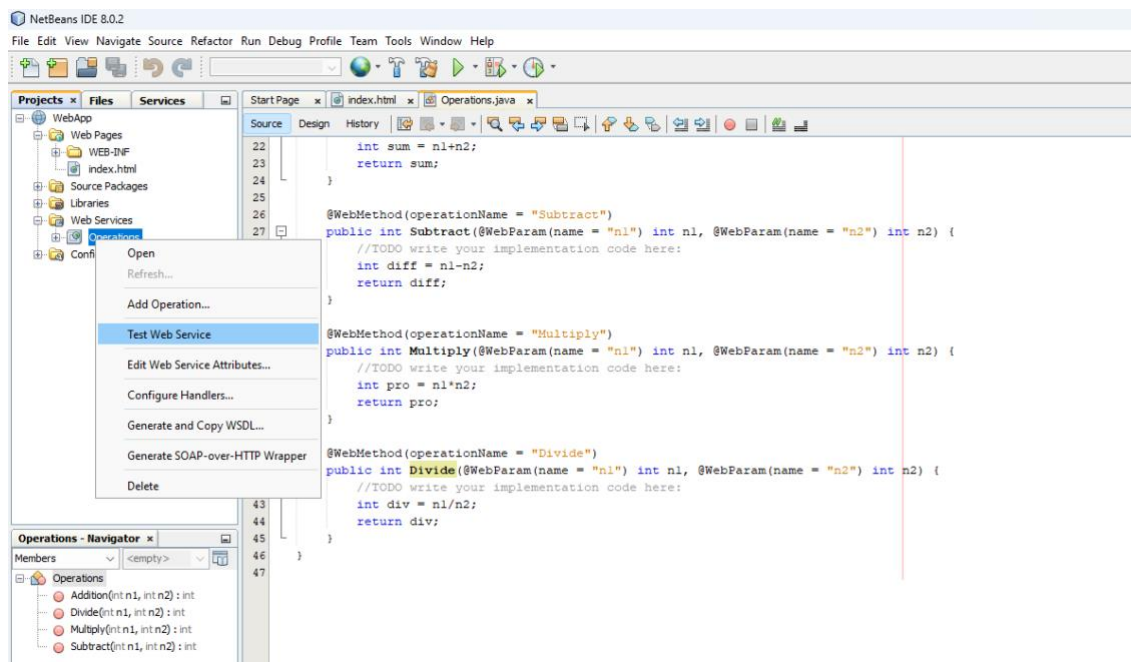
**Step 3:** Similarly create Operations for Subtraction, Multiplication and Division and then modify the Logic according to the operation.



```
16 @WebService(serviceName = "Operations")
17 public class Operations {
18
19     @WebMethod(operationName = "Addition")
20     public int Addition(@WebParam(name = "n1") int n1, @WebParam(name = "n2") int n2) {
21         //TODO write your implementation code here:
22         int sum = n1+n2;
23         return sum;
24     }
25
26     @WebMethod(operationName = "Subtract")
27     public int Subtract(@WebParam(name = "n1") int n1, @WebParam(name = "n2") int n2) {
28         //TODO write your implementation code here:
29         int diff = n1-n2;
30         return diff;
31     }
32
33     @WebMethod(operationName = "Multiply")
34     public int Multiply(@WebParam(name = "n1") int n1, @WebParam(name = "n2") int n2) {
35         //TODO write your implementation code here:
36         int pro = n1*n2;
37         return pro;
38     }
39
40     @WebMethod(operationName = "Divide")
41     public int Divide(@WebParam(name = "n1") int n1, @WebParam(name = "n2") int n2) {
42         //TODO write your implementation code here:
43         int div = n1/n2;
44         return div;
45     }
46 }
47
```

**Step 4:** Deploy the Project and Test the Web Service.

- Right Click on the Project > Deploy > Right Click on the Web Service (Operations) > Test Web Service.



**Output:**

← → ↻ ⓘ localhost:8080/WebApp/Operations?Tester

# Operations 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 package1.Operations.divide(int,int)

divide ( 10 ) 2

public abstract int package1.Operations.multiply(int,int)

multiply ( )

public abstract int package1.Operations.subtract(int,int)

subtract ( )

public abstract int package1.Operations.addition(int,int)

addition ( )

← ↻ ⓘ localhost:8080/WebApp/Operations?Tester

## divide Method invocation

**Method parameter(s)**

| Type | Value |
|------|-------|
| int  | 10    |
| int  | 2     |

**Method returned**

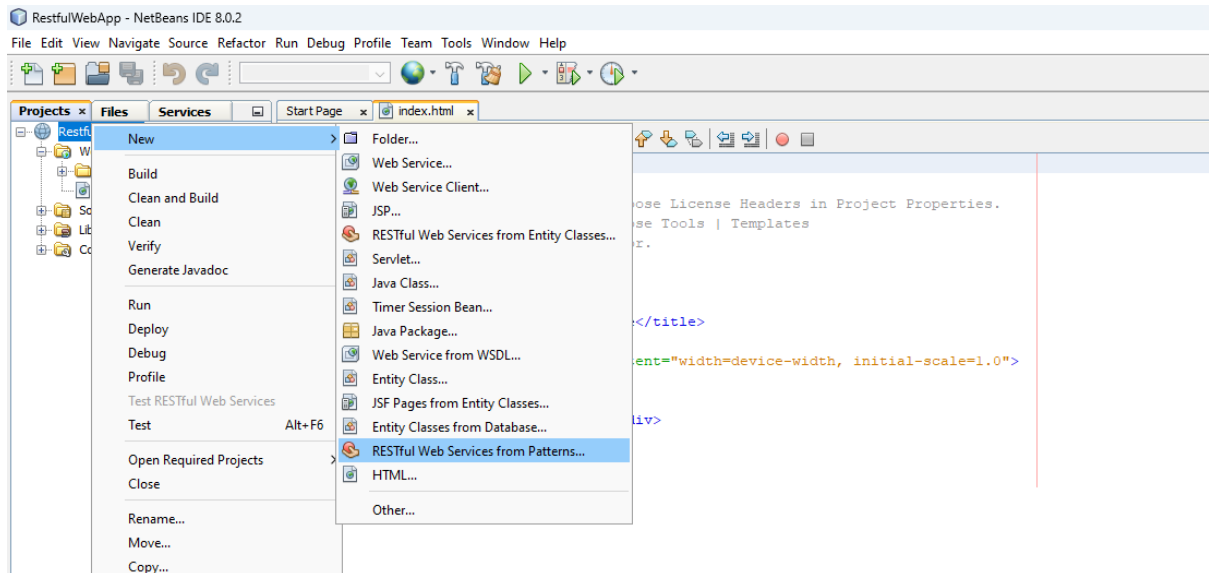
int : "5"

## Practical No. 3

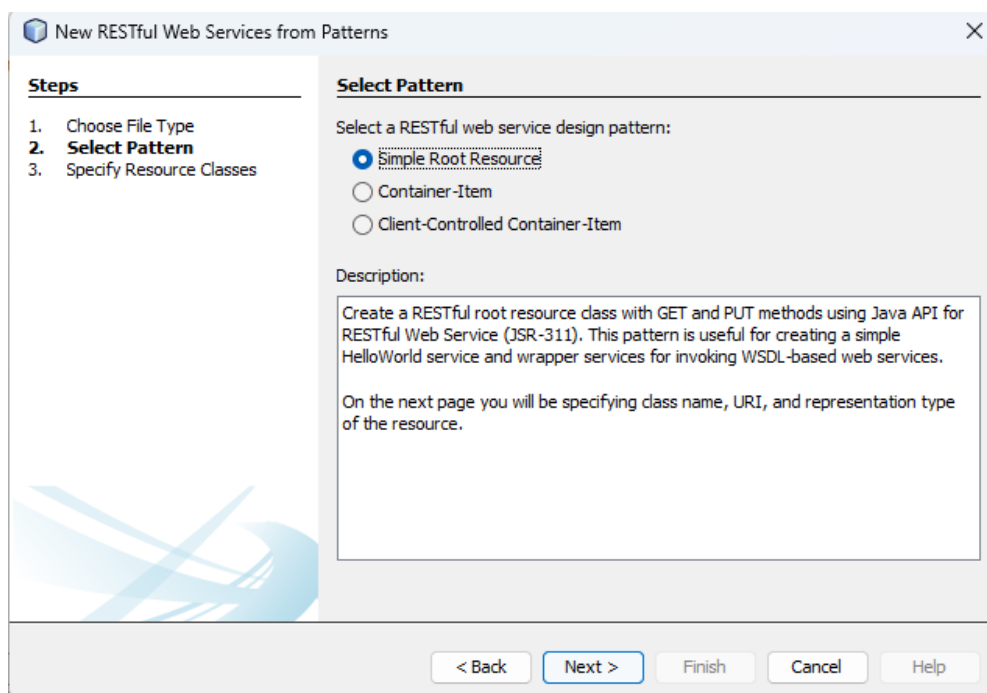
**Aim: Create a Simple REST Service.**

**Step 1:** Create a new Web Application in NetBeans > Create a RESTful Web Service.

- Right Click on the Project > New > RESTful Web Service from Patterns.



- Click Next > Give Package Name > Finish.



**Step 2:** Modify the Code in the RESTful Web Service as Given Below:

```
package package1;

import javax.ws.rs.core.Context;
import javax.ws.rs.core.UriInfo;
import javax.ws.rs.PathParam;
import javax.ws.rs.Produces;
import javax.ws.rs.Consumes;
import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.PUT;
import javax.ws.rs.core.MediaType;

@Path("generic")
public class StringOperations {

    @Context
    private UriInfo context;

    public StringOperations() {
    }

    @GET
    @Produces("application/xml")
    public String getXml() {
        //TODO return proper representation object
        throw new UnsupportedOperationException();
    }

    @PUT
    @Consumes("application/xml")
    public void putXml(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)

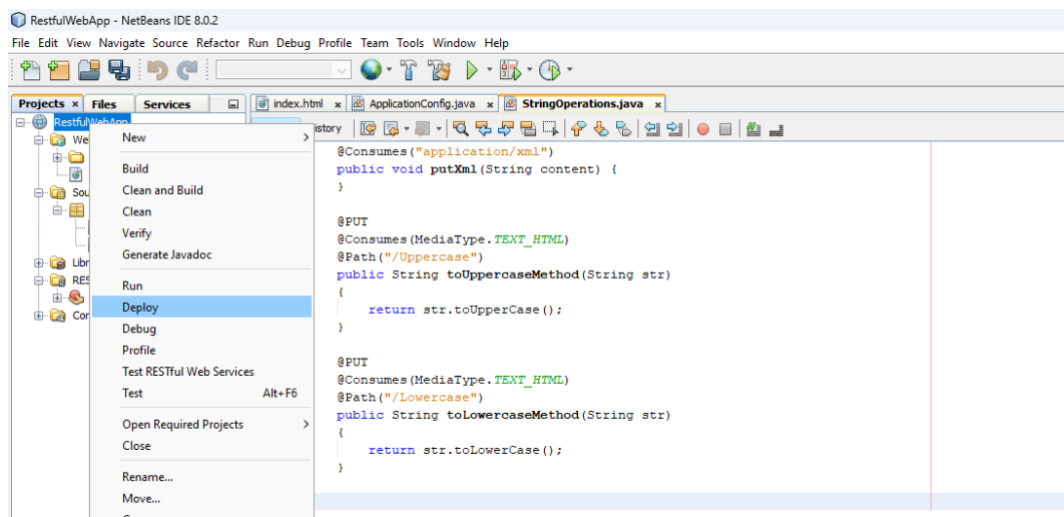
{

    return str.toLowerCase();

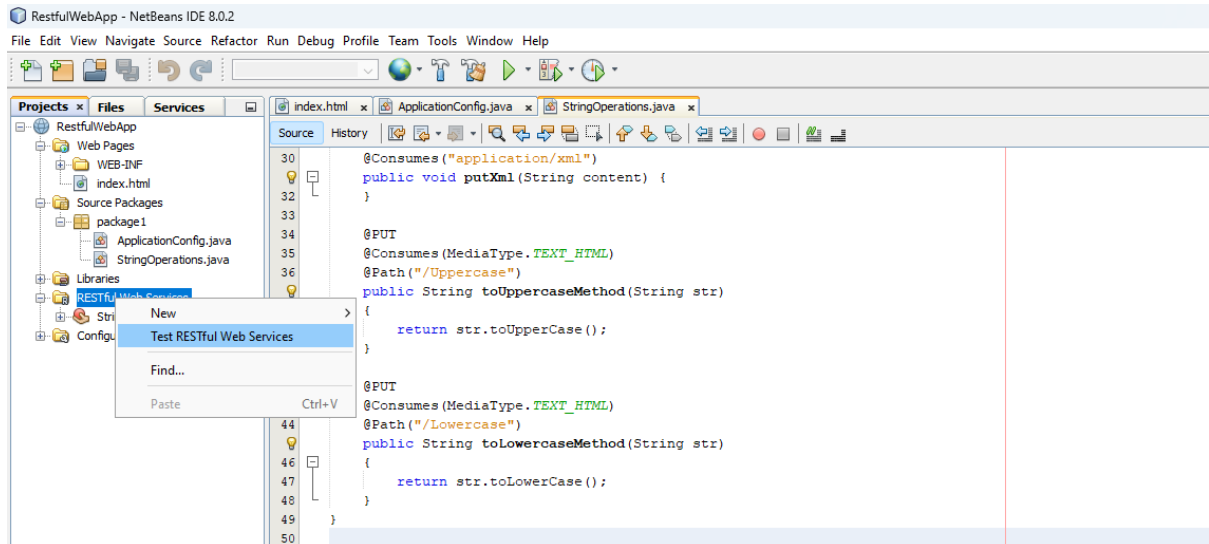
}

}

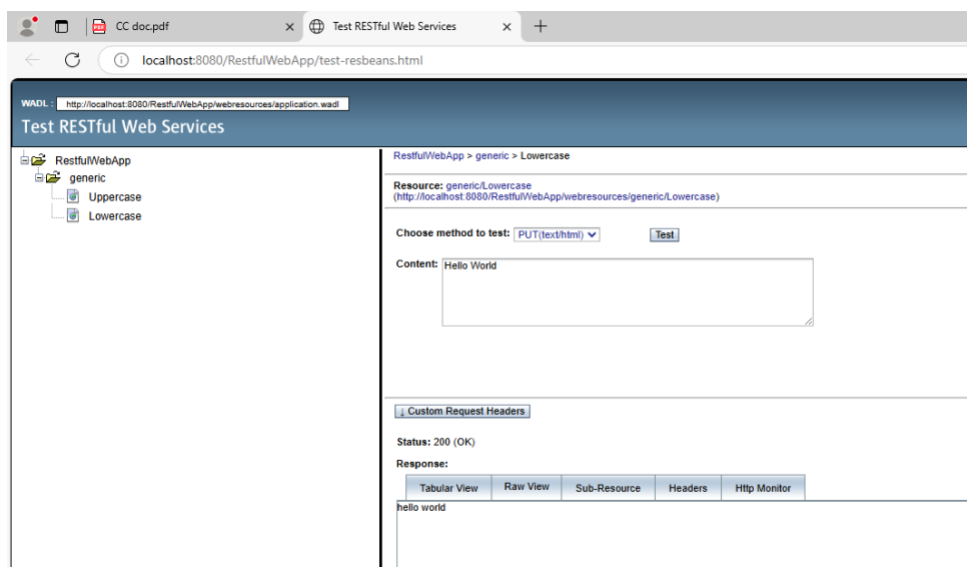
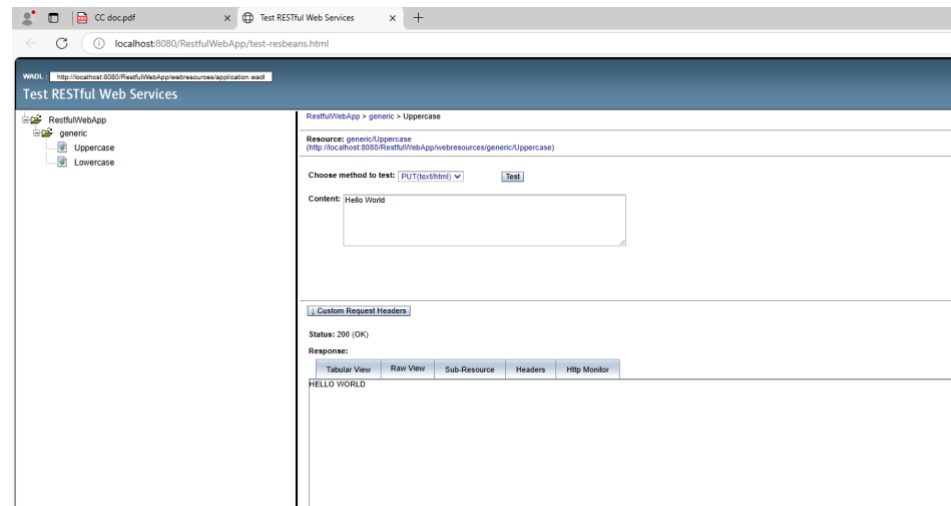
#### Step 4: Deploy the Project.



➤ Test the Web Service.



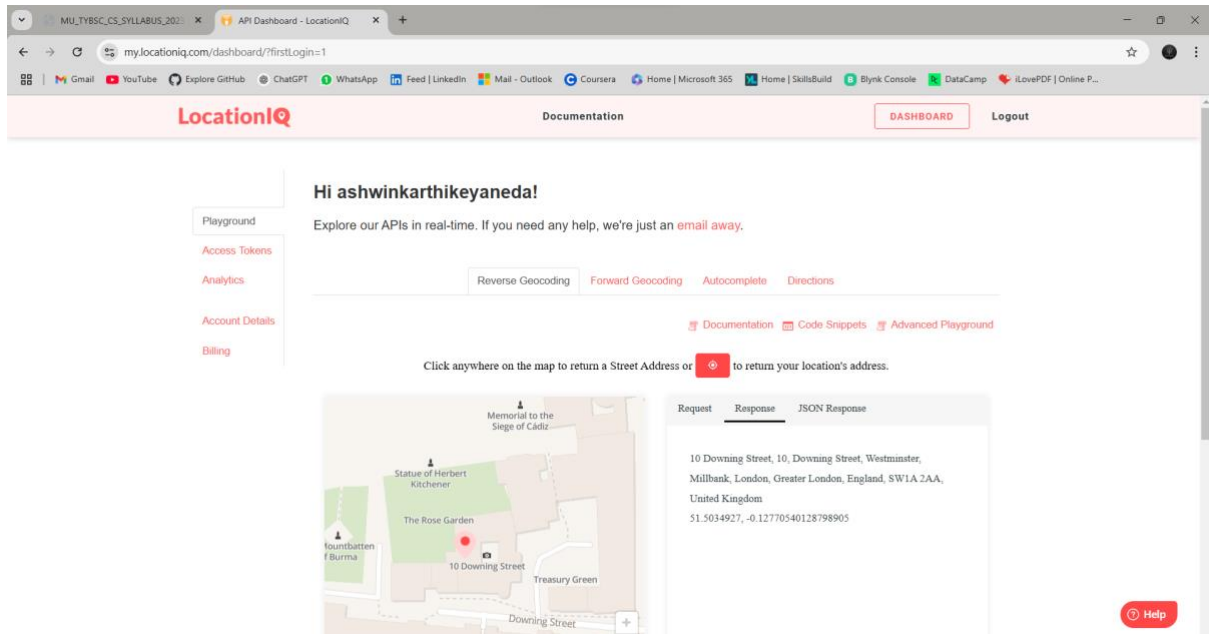
## Output:



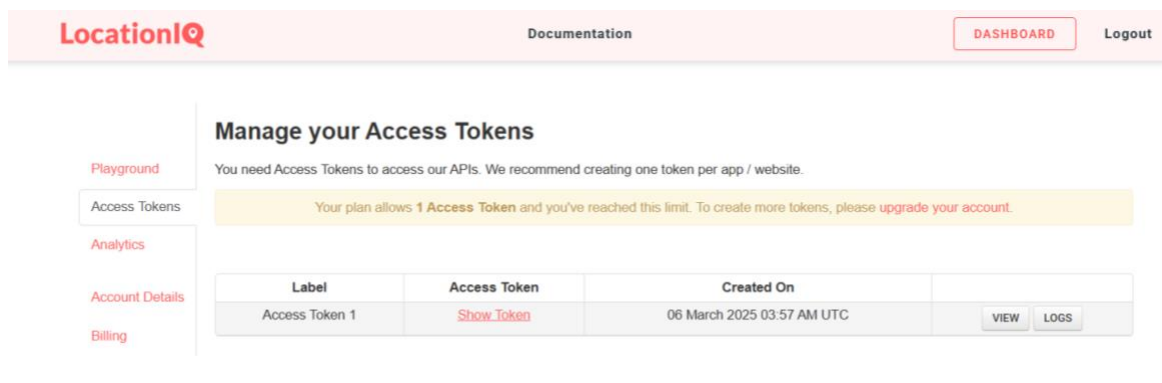
## Practical No. 4

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

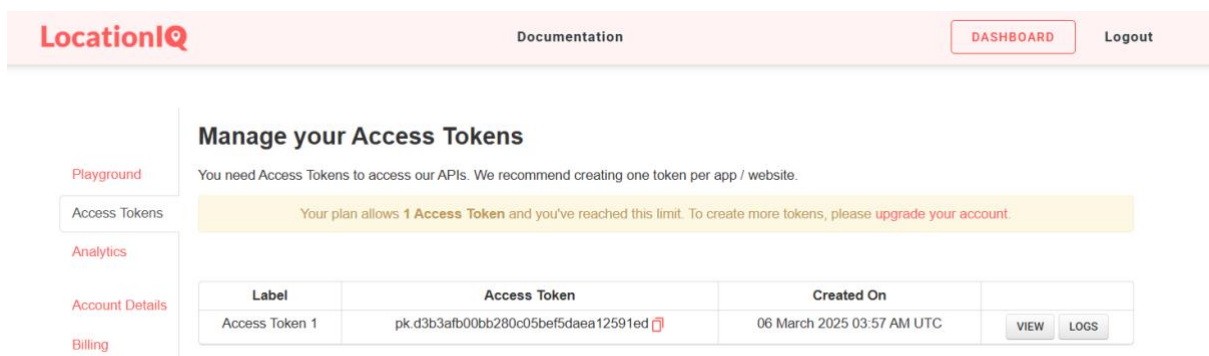
**Step 1: Browse to 'LocationIQ' Website and login to the website.**



➤ Navigate to the 'Access Tokens' in the Menu.



➤ Click on Show Token to reveal the Access Token > Copy the Access Token.





**Step 2:** Enter the Following Code in Python IDLE:

- Paste your Access Token in the 'api\_key' variable in the code.

```
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.d3b3afb00bb280c05bef5daea12591ed'
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}")
```

➤ Run the Program.

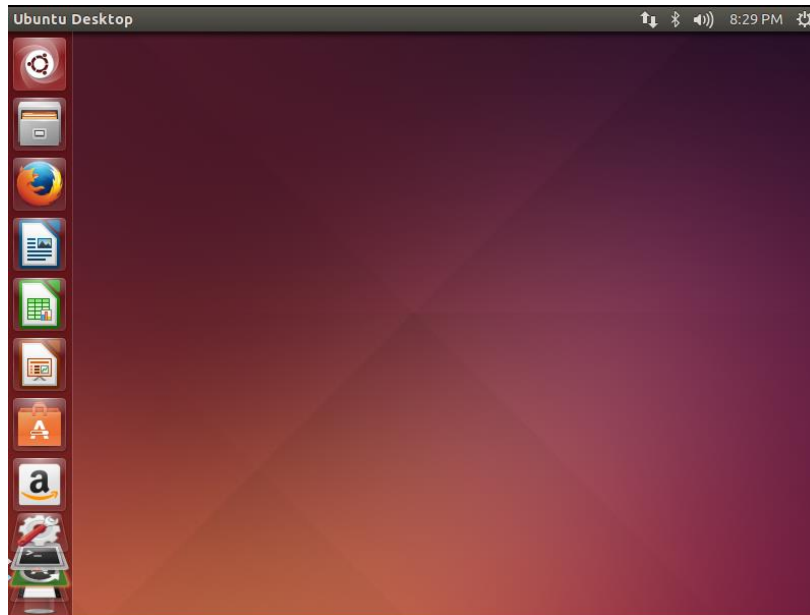
**Output:**

```
>>> |===== RESTART: F:/CC/CC_prac4.py
      |Enter the location : mumbai
      |Output:
      |place_id: 227883413
      |lat: 19.08157715
      |lon: 72.88662753964906
      |display_name: Mumbai, Maharashtra, India
>>> |
```

## Practical No. 5

**Aim: Installation and Configuration of virtualization using KVM.**

**Step 1:** Open VMWare > Install & Launch Ubuntu.



**Step 2:** Open Terminal and Enter the following Command:

- `sudo apt-get update`

```
sds@ubuntu: ~  
sds@ubuntu:~$ sudo apt-get update  
[sudo] password for sds:  
Hit http://security.ubuntu.com trusty-security InRelease  
Hit http://security.ubuntu.com trusty-security/main Sources  
Ign http://extras.ubuntu.com trusty InRelease  
Hit http://security.ubuntu.com trusty-security/restricted Sources  
Hit http://security.ubuntu.com trusty-security/universe Sources  
Hit http://extras.ubuntu.com trusty Release.gpg  
Ign http://us.archive.ubuntu.com trusty InRelease  
Hit http://security.ubuntu.com trusty-security/multiverse Sources  
Hit http://extras.ubuntu.com trusty Release  
Hit http://us.archive.ubuntu.com trusty-updates InRelease  
Hit http://security.ubuntu.com trusty-security/main i386 Packages  
Hit http://extras.ubuntu.com trusty/main Sources  
Hit http://security.ubuntu.com trusty-security/restricted i386 Packages  
Hit http://us.archive.ubuntu.com trusty-backports InRelease  
Hit http://extras.ubuntu.com trusty/main i386 Packages  
Hit http://security.ubuntu.com trusty-security/universe i386 Packages  
Hit http://us.archive.ubuntu.com trusty Release.gpg  
Hit http://security.ubuntu.com trusty-security/multiverse i386 Packages
```

- `sudo grep -c "svm\|vmx" /proc/cpuinfo`

```
sds@ubuntu:~$ sudo grep -c "svm\|vmx" /proc/cpuinfo  
0  
sds@ubuntu:~$
```

- cat /proc/cpuinfo

```
sds@ubuntu:~$ cat /proc/cpuinfo
processor       : 0
model          : 6
model name     : GenuineIntel
stepping       : 5
microcode      : 0x2e
cpu MHz        : 3302.396
cache size     : 12288 KB
physical id    : 0
siblings       : 1
core id        : 0
cpu cores      : 1
apicid         : 0
initial apicid : 0
fdiv_bug       : no
```

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

```
sds@ubuntu:~$ sudo apt install qemu-kvm libvirt-daemon-system libvirt-clients b
ridge-utils virt-manager
Reading package lists... Done
Building dependency tree
Reading state information... Done
E: Unable to locate package libvirt-daemon-system
E: Unable to locate package libvirt-clients
sds@ubuntu:~$ sudo apt-get install qemu-kvm libvirt-bin bridge-utils virt-manage
r
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  augeas-lenses cgroup-lite cpu-checker ebttables gawk ipxe-qemu libaio1
  libappindicator1 libaugeas0 libbonobo2-0 libbonobo2-common libbonoboui2-0
  libbonoboui2-common libboost-thread1.54.0 libfdt1 libglade2-0 libgnome2-0
  libgnome2-bin libgnome2-common libgnomecanvas2-0 libgnomecanvas2-common
  libgnomeui-0 libgnomeui-common libgnomevfs2-0 libgnomevfs2-common
  libgtk-vnc-1.0-0 libgvnc-1.0-0 libidl-common libidl0 libindicator7 libnetcf1
  liborbit-2-0 liborbit2 librados2 librbdl1 libSDL1.2debian libseccomp2
  libsigsegv2 libspice-server1 libusbredirparser1 libvirt0 libvte-common
  libvte9 libxen-4.4 libxenstore3.0 libxml2-utils msr-tools
  python-appindicator python-glade2 python-gnome2 python-gtk-vnc
  python-libvirt python-pycurl python-pyorbit python-urlgrabber python-vte
  qemu-keymaps qemu-system-common qemu-system-x86 qemu-utils seabios sharutils
  virtinst
Suggested packages:
```

- sudo adduser ashwin1

```

sdsm@ubuntu:~$ sudo adduser ashwin1
Adding user `ashwin1' ...
Adding new group `ashwin1' (1002) ...
Adding new user `ashwin1' (1002) with group `ashwin1' ...
Creating home directory `/home/ashwin1' ...
Copying files from `/etc/skel' ...
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for ashwin1
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
sdsm@ubuntu:~$ █

```

- `sudo adduser ashwin1 libvirtd`

```

sdsm@ubuntu:~$ sudo adduser ashwin1 libvirtd
Adding user `ashwin1' to group `libvirtd' ...
Adding user ashwin1 to group libvirtd
Done.
sdsm@ubuntu:~$ █

```

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

```

sdsm@ubuntu:~$ sudo virsh -c qemu:///system list
 Id      Name                                     State
-----
sdsm@ubuntu:~$ █

```

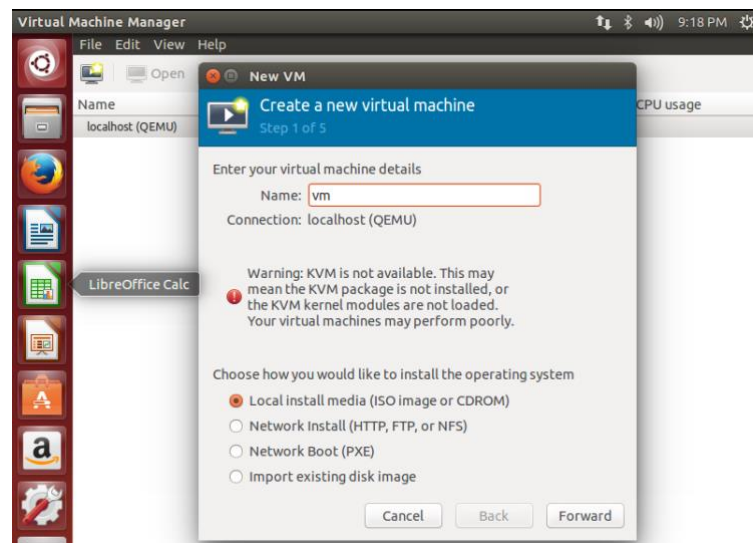
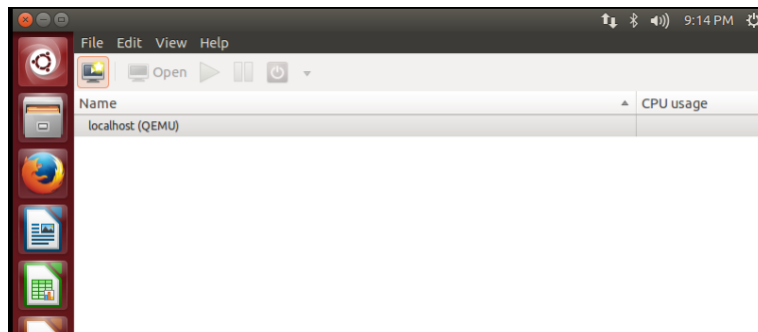
- `sudo virt-manager`

```

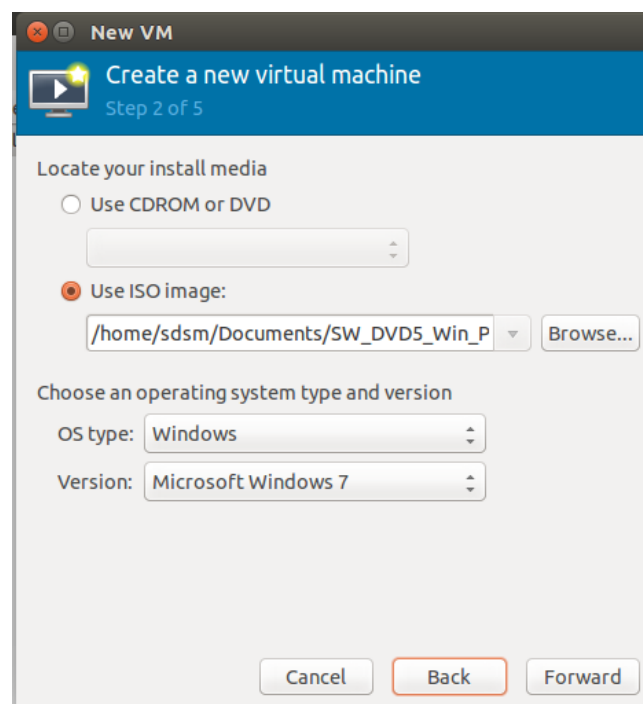
sdsm@ubuntu:~$ sudo virt-manager
sdsm@ubuntu:~$ █

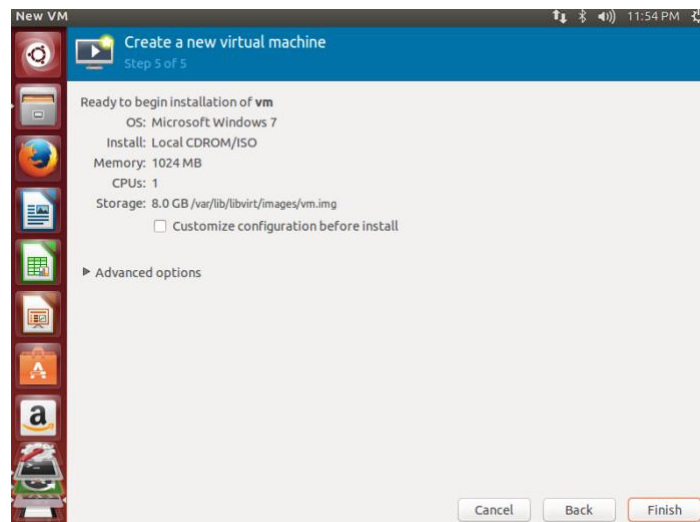
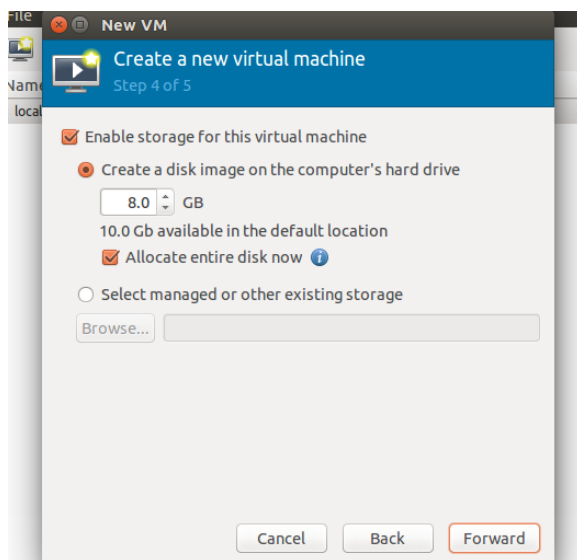
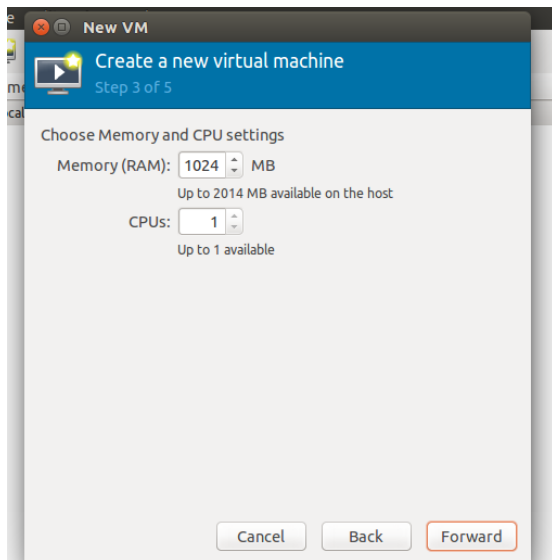
```

This will open the Virtual Machine Manager window.



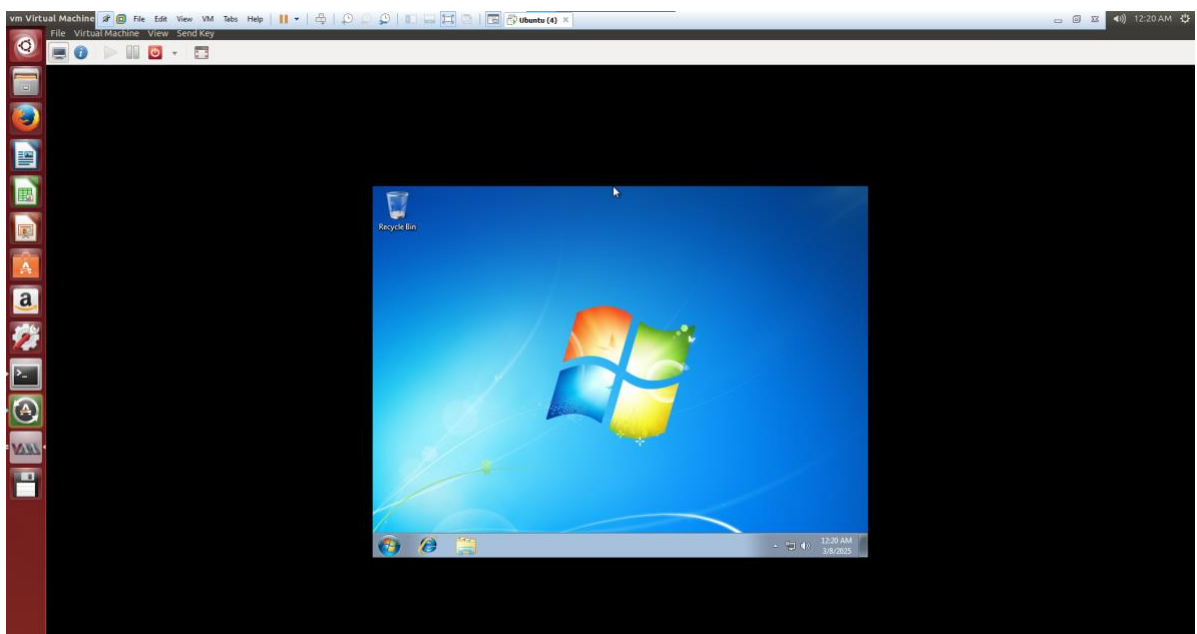
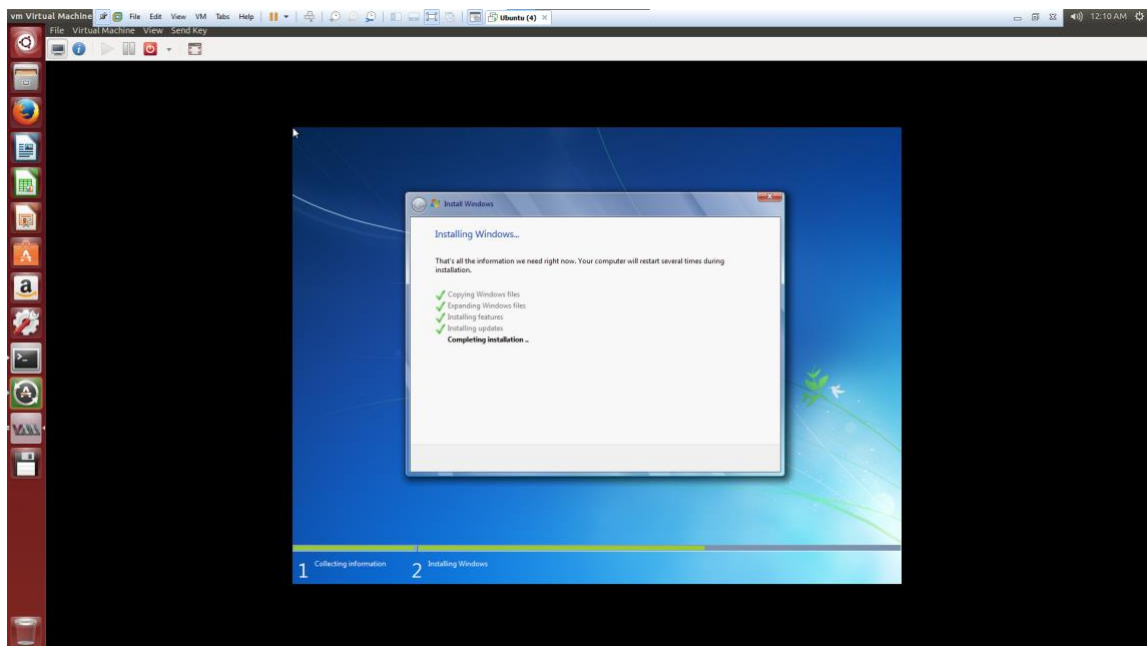
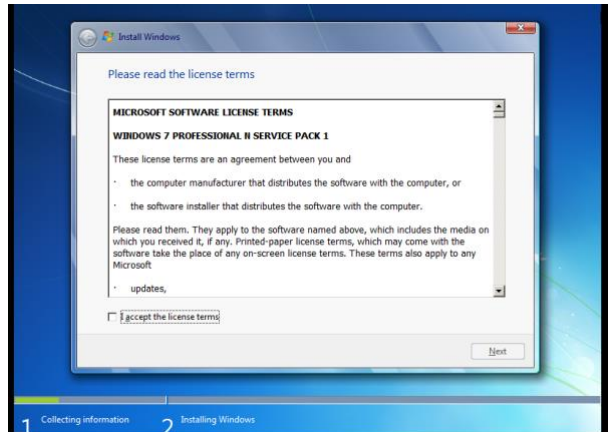
- Copy & Paste the Windows '.iso' file anywhere in Ubuntu. Browse & Select the file.





➤ Now Setup the Windows OS.







## Practical No. 6

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

**Step 1:** Create a Folder named 'upload' in any drive and copy its path.

- Open Notepad or any other Editor > Enter the Below Program > Save the file in JavaScript Format.

### Code:

```
const express = require('express');
const multer = require('multer');
const fs = require('fs');
const path = require('path');

const app = express();
const PORT = 3000;
const UPLOAD_FOLDER = 'D:/upload';

if (!fs.existsSync(UPLOAD_FOLDER)) {
  fs.mkdirSync(UPLOAD_FOLDER, { recursive: true });
}

// Multer setup
const upload = multer({ storage: multer.diskStorage({
  destination: UPLOAD_FOLDER,
  filename: (req, file, cb) => cb(null, file.originalname)
})});

// Upload Route
app.post('/upload', upload.single('file'), (req, res) =>
  req.file ? res.send(`Uploaded: ${req.file.originalname}`) : res.status(400).send('No file
uploaded.'))
```

```
);
```

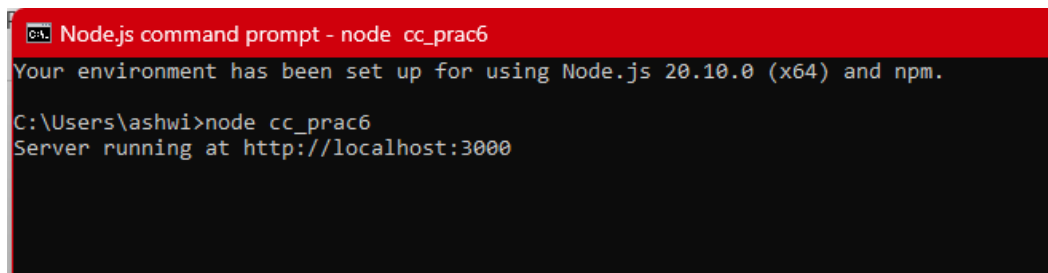
```
// Download Route
```

```
app.get('/download', (req, res) => {  
    const filePath = path.join(UPLOAD_FOLDER, req.query.filename || '');  
    return fs.existsSync(filePath) ? res.download(filePath) : res.status(404).send('File not  
found.');
```

```
// Start Server
```

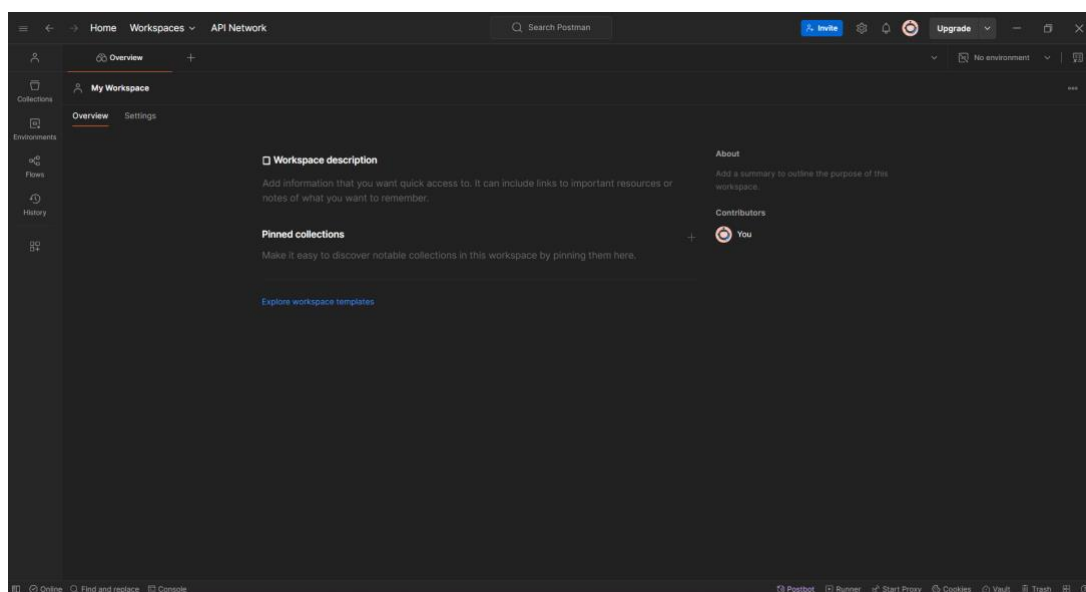
```
app.listen(PORT, () => console.log(`Server running at http://localhost:\${PORT}`));
```

- Paste the Path of the 'upload' folder in the 'UPLOAD\_FOLDER' variable > Save the Program > Run the Program from NodeJS Command Prompt

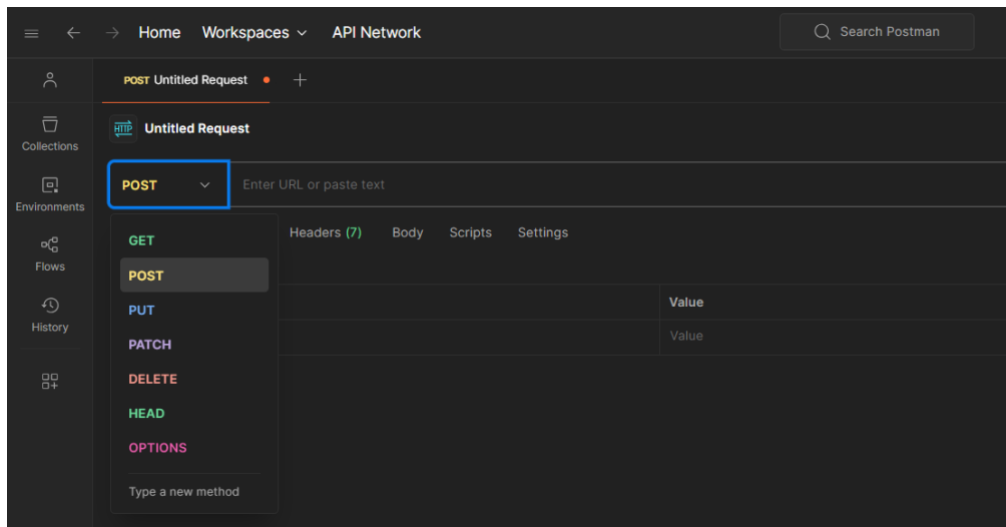


```
Node.js command prompt - node cc_prac6  
Your environment has been set up for using Node.js 20.10.0 (x64) and npm.  
  
C:\Users\ashwi>node cc_prac6  
Server running at http://localhost:3000
```

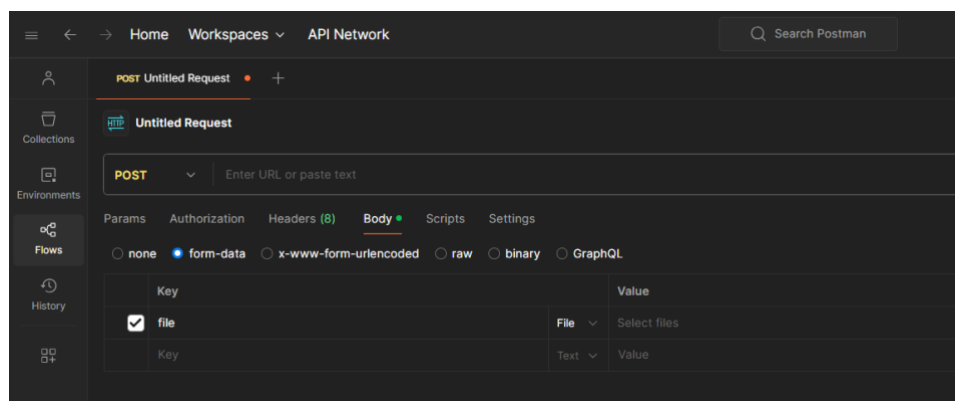
**Step 2: Open Postman and Login with Valid Credentials.**



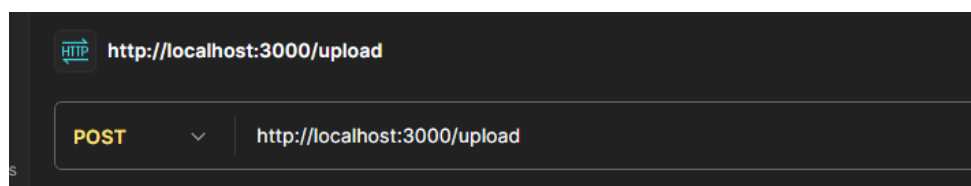
- Create a new Request > Select the POST method.



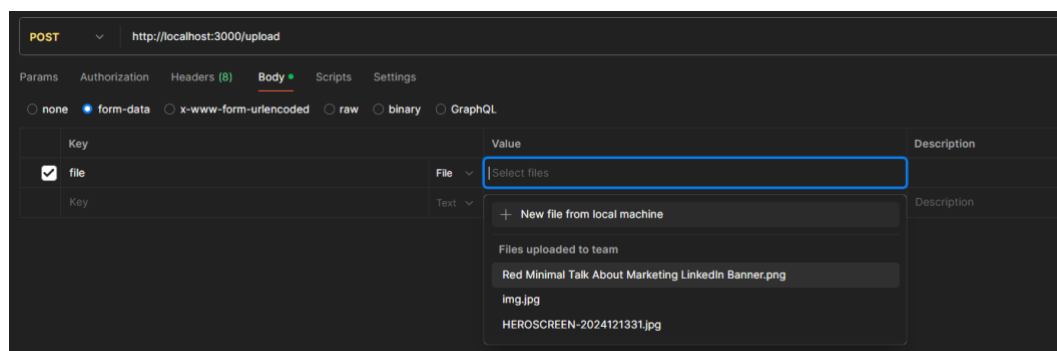
- Navigate to Body > Form – Data > Create a key named ‘file’ > Select the Type as ‘File’.



- Enter the host link in the URL field > Enter ‘/upload’ to upload a file to the server along with the link.



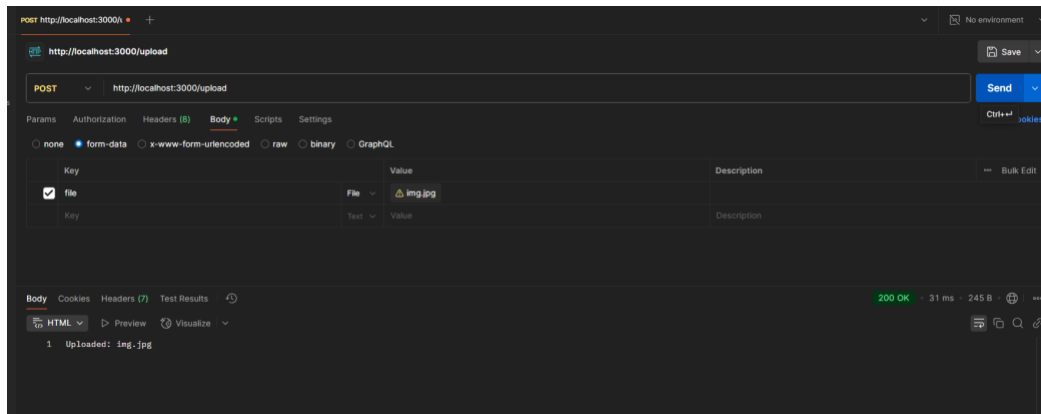
- Navigate to the Value Field and Upload an Image file from the device.



☐ none
 ☒ form-data
 ☐ x-www-form-urlencoded
 ☐ raw
 ☐ binary
 ☐ GraphQL

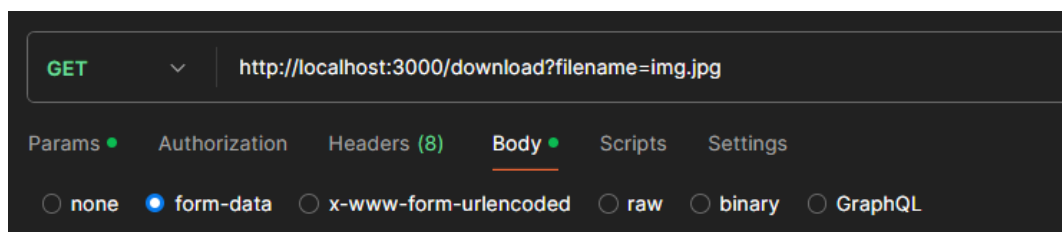
| Key                                      | Value                           |
|--|---------------------------------|
| <input checked="" type="checkbox"/> file | File  img.jpg                   |
| Key                                      | Text <input type="text"/> Value |

➤ Click on Send.

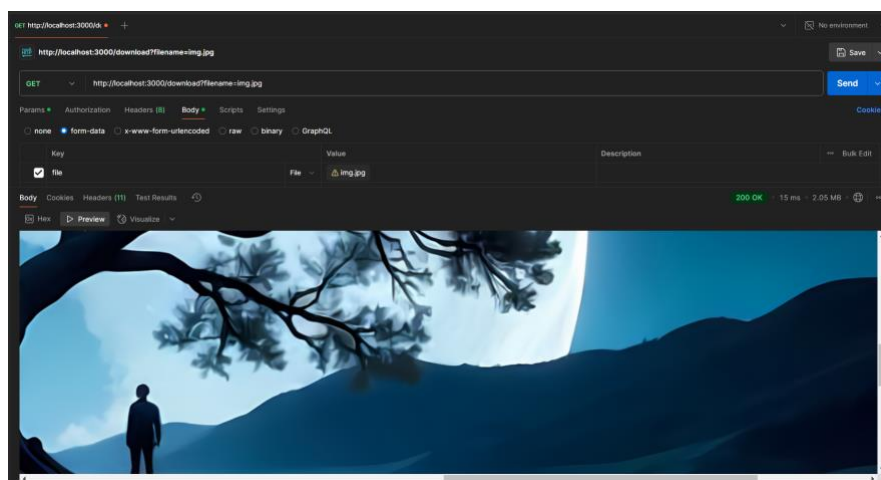


## To Download the File:

➤ Select the method as GET > Enter ‘/download?filename=img.jpg’ with the URL. (img.jpg > the name of the image file).



➤ Click on Send.



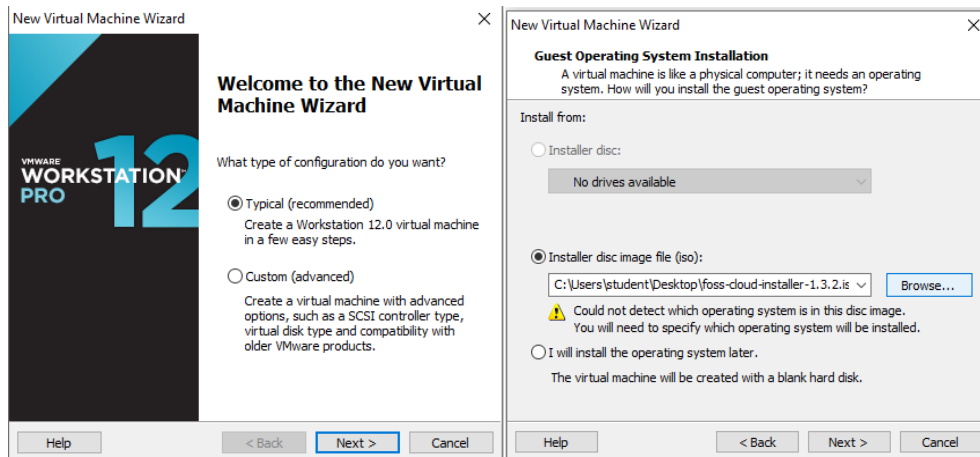
The Image will be downloaded from the server.

## Practical No. 7

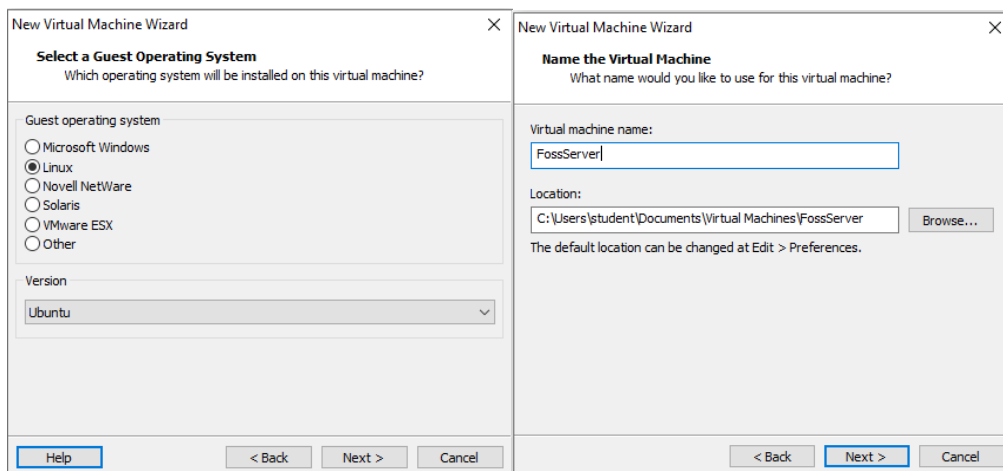
### Aim: Implement FOSS-Cloud Functionality VSI (Virtual Server Infrastructure) Infrastructure as a Service (IaaS), Storage.

**Step 1:** Download the Foss Server Disk Image file ‘.iso’ file.

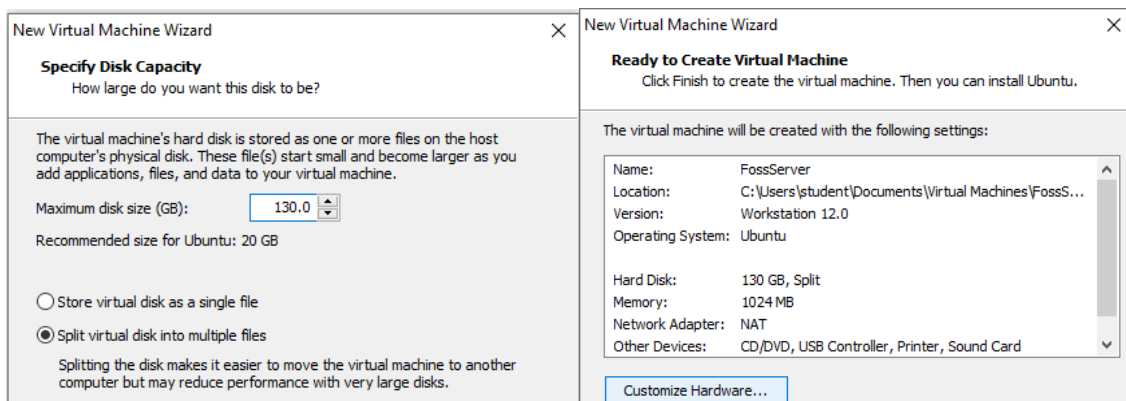
- Open VMware > Create new virtual machine > Browse and add the path of the Foss server Disk Image file in the provided Field.



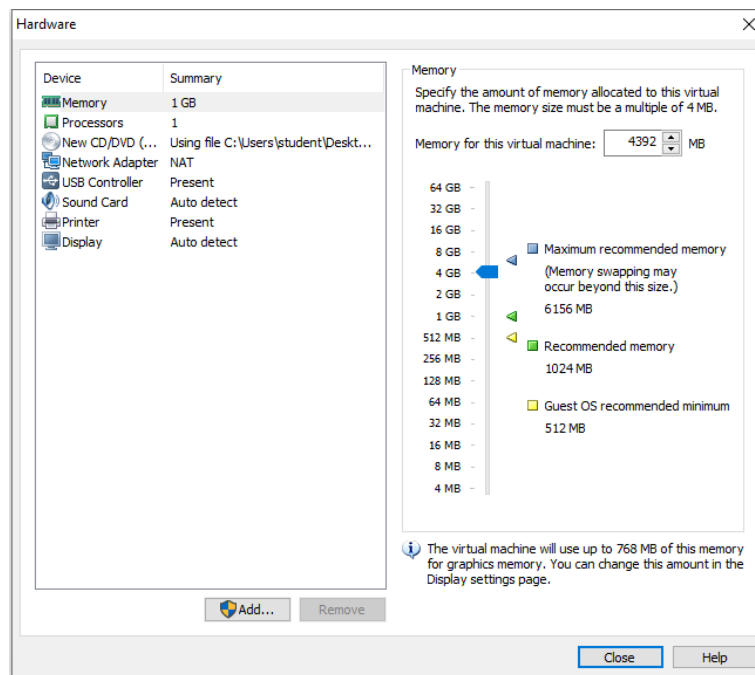
- Select Linux > Next > Provide a name to the Virtual machine > Next.



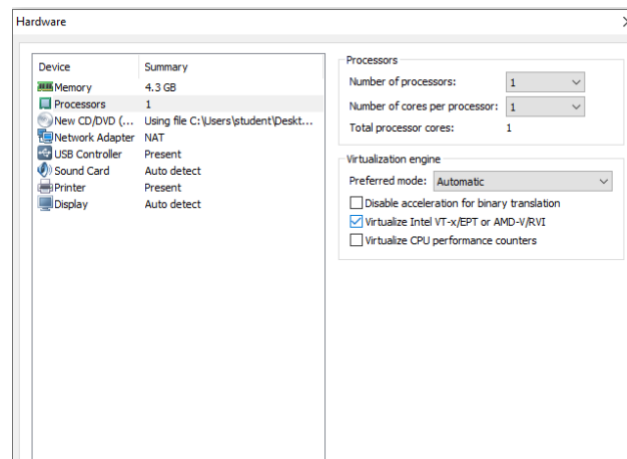
- Set the maximum Disk size to 130 > Next > Click on Customize Hardware.



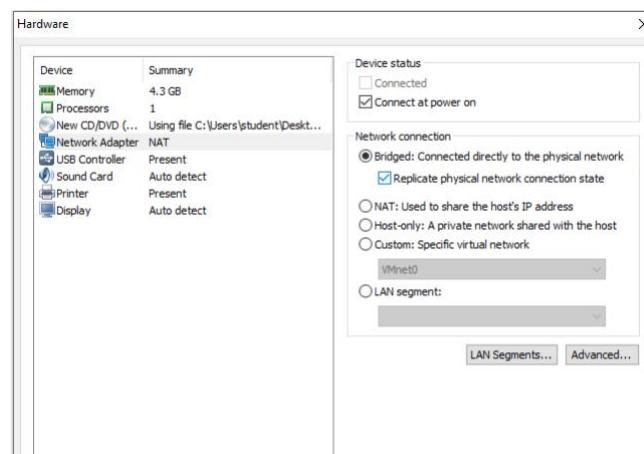
- In the Hardware Configuration, set Memory (Ram) more than 4GB.



- Navigate to 'Processors' > Check the Second Option.



- Go to Network Adapter > Set the Network Connection to 'Bridged' & Check the Option under it > Close the Hardware Configuration > Click on Finish.



## Step 2: Power on the Virtual Machine.

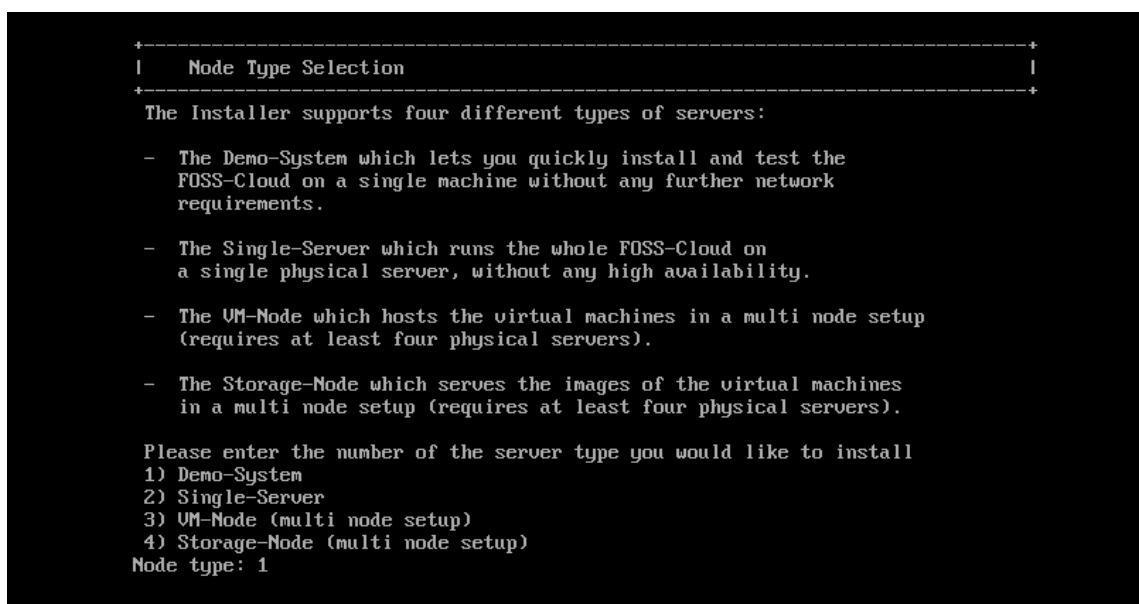
- Click on the First Option & Press Enter.



- To start the Installation, Type yes and hit Enter.



- Enter the Node Type as 1.



- Enter the Device as 'sda' (Same as it shows above).

```
Please enter the number of the server type you would like to install
1) Demo-System
2) Single-Server
3) VM-Node (multi node setup)
4) Storage-Node (multi node setup)
Node type: 1

+-----+
| 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 (130 GB). Size is OK

Below you will find a list of all detected and supported disks
sda (130 GB)

Please enter the device name on which you would like to install
Device: sda_
```

- Type 'yes' > Enter.

```
+-----+
| Installation Device Partitioning |
+-----+

Below is the existing partition layout of your selected device

Error: /dev/sda: unrecognised disk label
Model: VMware, VMware Virtual S (scsi)
Disk /dev/sda: 140GB
Sector size (logical/physical): 512B/512B
Partition Table: unknown
Disk Flags:

All existing partitions have to be deleted in order to continue
THIS MEANS THAT ALL DATA ON THIS DISK WILL BE LOST
Do you want to continue?
yes or no?: yes
```

- For Network Device Selection, Enter the same device that is given above > Type yes to Reboot the System.

```
+-----+
| Network Device Selection |
+-----+

Please enter the device which you would like to use

Available ethernet devices: eno16777736
Device #0: eno16777736

+-----+
| Network Configuration |
+-----+

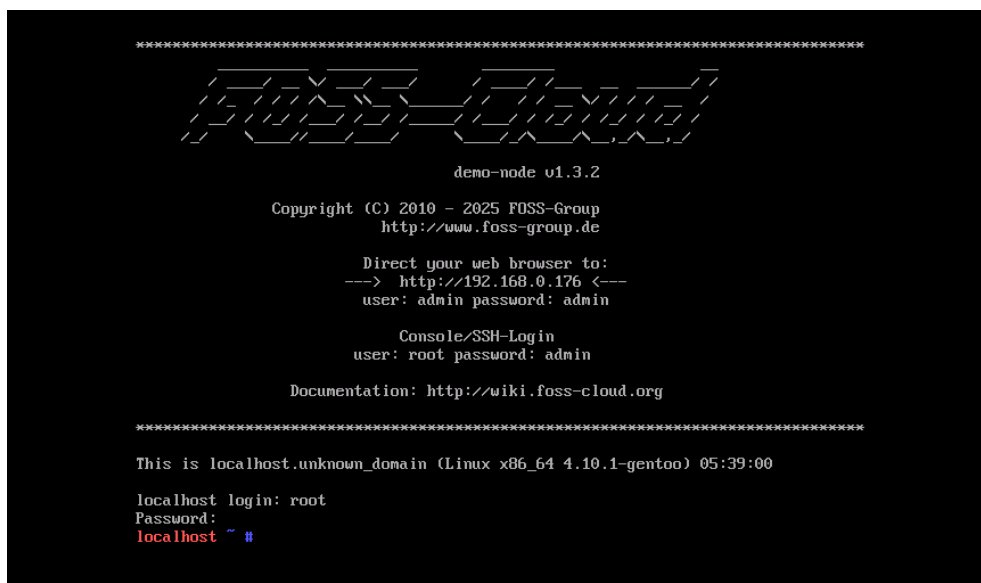
Do you want to use automatic network configuration (via DHCP)?
yes or no?: yes
```



**Step 3:** Click on the First option while the System reboots.

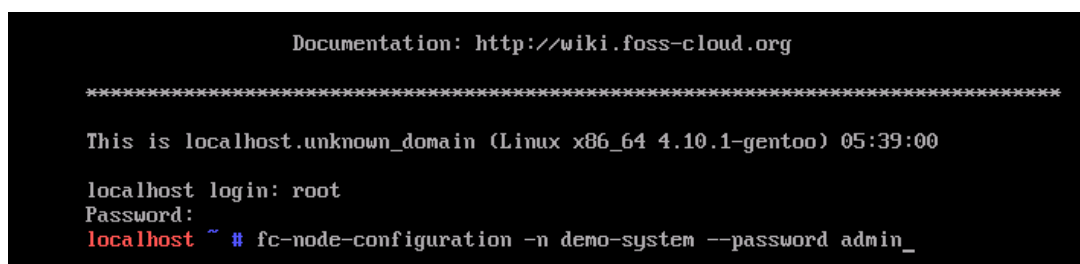


- Once the system Boots, Enter the Localhost as 'root' and Password 'admin' to access the Server CLI.



- In the Foss CLI, Enter the Following Command:

**fc-node-configuration -n demo-system --password admin**



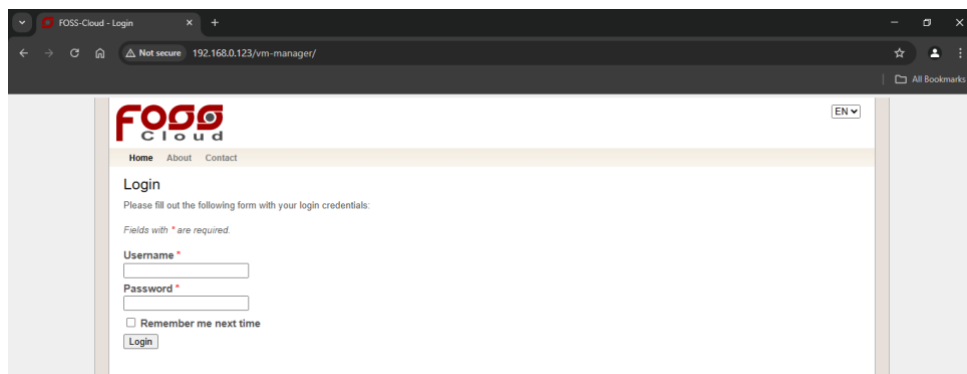
**Ifconfig**

```
localhost ~ # ifconfig
eno16777736: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.0.123 netmask 255.255.255.0 broadcast 192.168.0.255
    ether 00:0c:29:31:6a:d7 txqueuelen 1000 (Ethernet)
    RX packets 1462 bytes 105780 (103.3 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 105 bytes 9405 (9.1 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 19 base 0x2000

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 1168 bytes 292883 (286.0 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1168 bytes 292883 (286.0 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

vmb0: 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 42:6d:d4:b3:98:ef 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
```

- Copy the IP of the device ‘eno1677736’ (192.168.0.123) and open it in a browser.



- Enter Username & Password as ‘admin’ to Login.

