SHRI.G.P.M.DEGREE COLLEGE OF SCIENCE AND COMMERCE

SHRI G.P.M. DEGREE COLLEGE OF SCIENCE & COMMERCE

Nava Samaj Mandal, Dixit Road No 1, Vile Parle (E), Mumbai - 400057,

Tel.: 8928387200

CERTIFICATE

This is to certify Mr/Ms_____In the subject of _____, Student of T.Y.BSC.CS of Shri G.P.M. Degree College of Science and Commerce in partial fulfilment of requirement of Bachelor of Science in Computer Science prescribed by the University of Mumbai. Seat No._____
year 2025-26.

Signatories:

Prof.Incharge: (Name&Sign)_____

Principal Sign: Mr. Atul Yadav_____

External Examiner :(Name&Sign)_____

Date: _____

Place:_____

College Stamp

Professor Name: Abhishek Vishwakarma	Class: TYBSC-CS Semester: Sem VI (2025–2026)
Course code:	Subject: Cloud Computing

Sr. No.	Date	Index	Page No.	Sign
1		Define a web service for converting INR to USD and consume it using Java and .NET. Aim: To design and implement a simple currency conversion web service and invoke it from Java and .NET platforms.	……..	…….
2		Create a simple SOAP web service. Aim: To develop and deploy a basic SOAP-based web service.	……..	…….
3		Create a simple REST web service. Aim: To develop and deploy a basic RESTful web service.	……..	…….
4		Develop an application to consume Google Search / Google Maps RESTful Web Service. Aim: To consume Google Search API or Google Maps API using REST architecture.	……..	…….
5		Installation and configuration of virtualization using KVM. Aim: To install and configure KVM for creating and managing virtual machines.	……..	…….

6		Develop application to upload or download image/video using MTOM techniques. Aim: To implement file transfer of image or video between client and server using MTOM.	……..	…….
7		Implement FOSS Cloud functionality – VSI, IaaS and Storage. Aim: To implement Infrastructure as a Service using open-source cloud tools.	……..	…….

8		Implement FOSS Cloud functionality – Platform as a Service (PaaS). Aim: To implement Platform as a Service using open-source cloud environment.	……..	…….
9		Develop a simple workflow using AWS Flow Framework. Aim: To develop and execute a workflow that prints “Hello World” using AWS Flow Framework.	……..	…….
10		Implementation of OpenStack with user and private network creation. Aim: To install OpenStack and configure users and private networks.	……..	…….

Practical 1

Aim: Define a simple service like Converting Rs into Dollar and Call it from different platform like JAVA and .NET. Steps:

Tools & Technologies Used:

- NetBeans
- Java
- Visual Studio (.NET)
- SOAP Web Service

Learning Objectives:

1. To understand the concept of Web Services.
2. To learn how to create SOAP-based web services.
3. To understand cross-platform communication.
4. To deploy and test web services.
5. To consume a Java web service in a .NET application.

Theory:

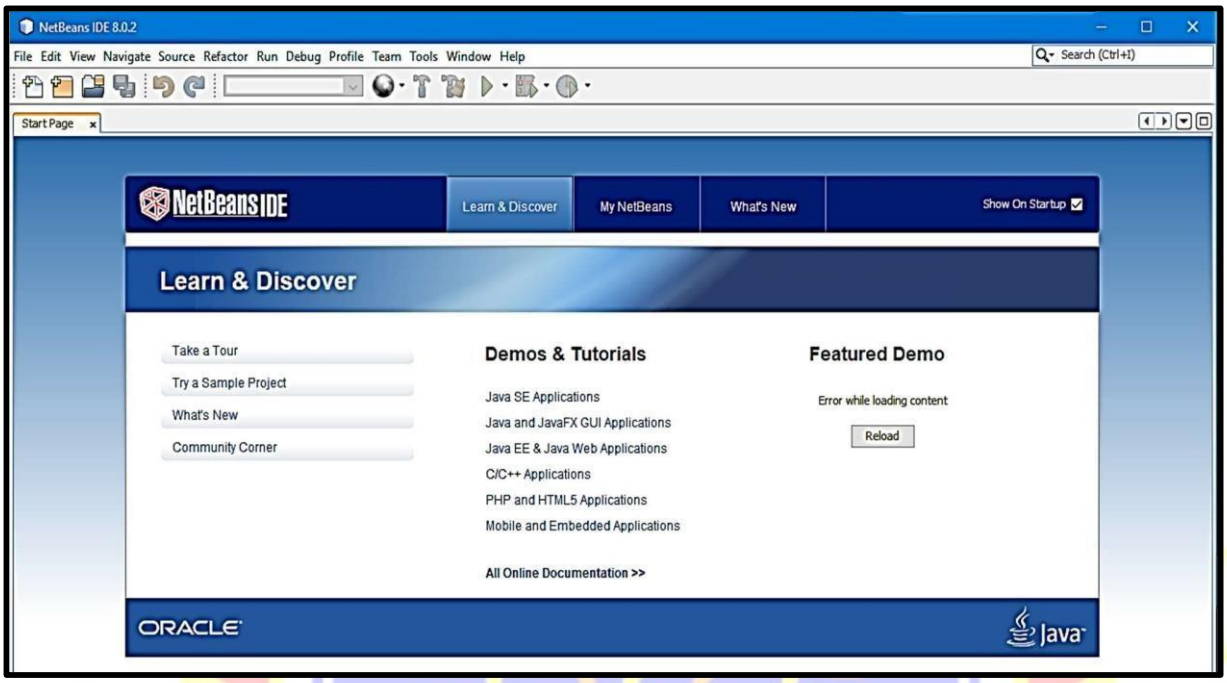
A Web Service is a software component that allows communication between different applications over the internet using standard protocols. It enables interoperability between systems developed in different programming languages and running on different platforms. SOAP (Simple Object Access Protocol) is a protocol used for exchanging structured information in web services. It uses XML format for sending messages over HTTP.
In this practical:

- A dynamic web application is created in NetBeans.
- A SOAP web service is added to the project.
- A method is created to convert INR to Dollar.
- The service is deployed on Apache Tomcat server.
- The service is tested using browser.
- The same service is consumed in Visual Studio (.NET).

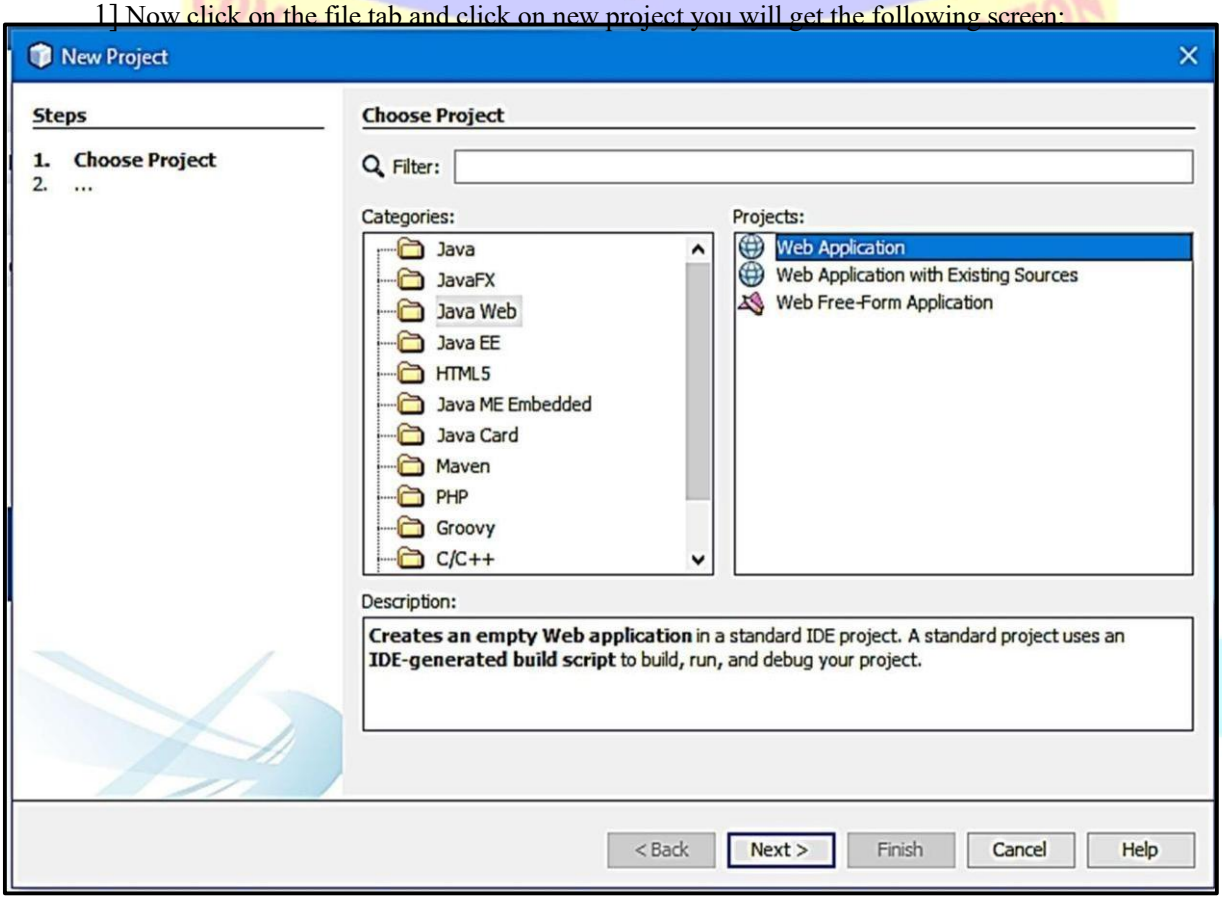
This demonstrates that web services are platform independent and support interoperability between Java and .NET environments.

Steps:

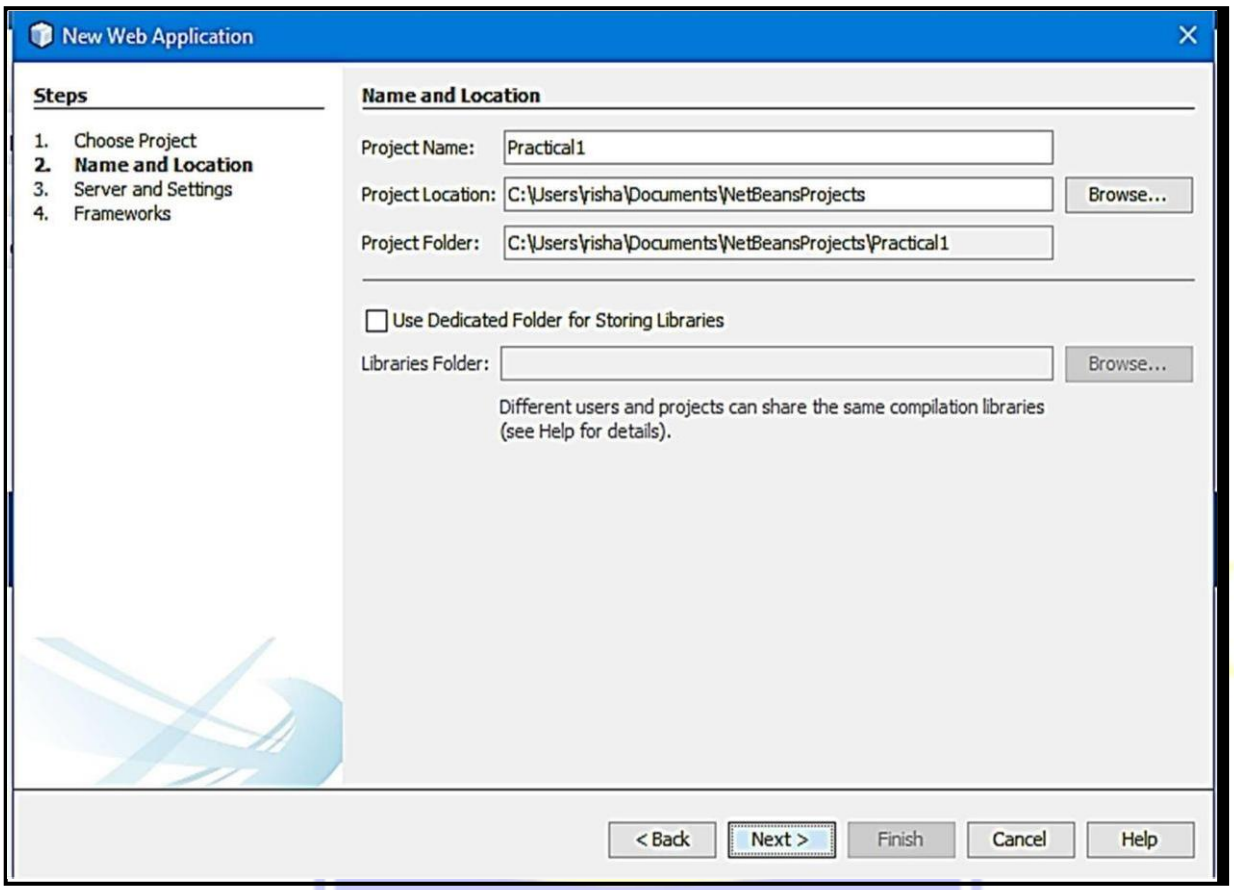
1.) Open the NetBeans, and you will get the following screen. Close the start page.



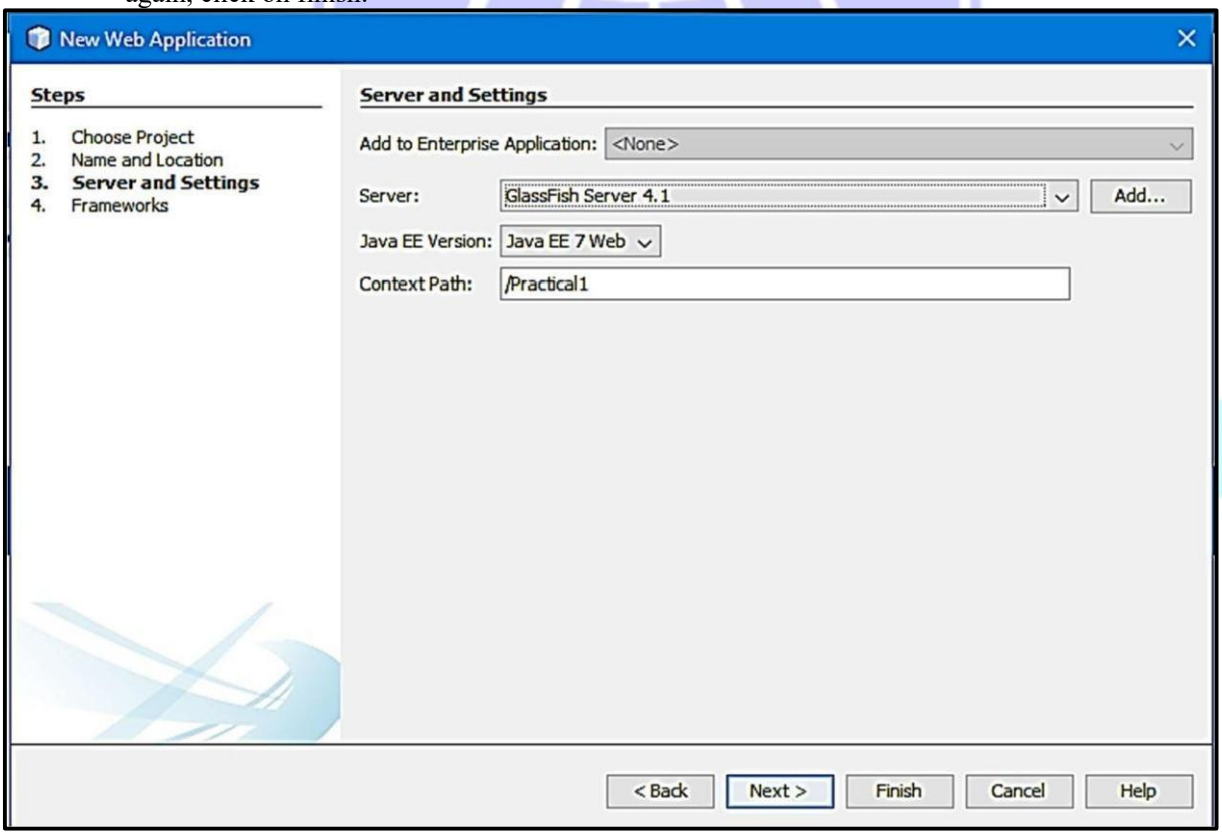
1] Now click on the file tab and click on new project you will get the following screen:



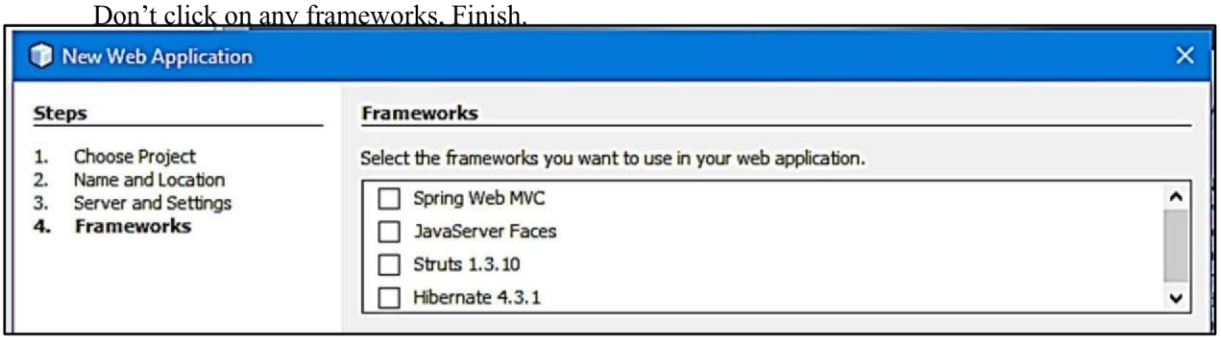
2] In Categories select Java Web and in Projects, Select Web Application. After selecting click on next. You will get the following window:



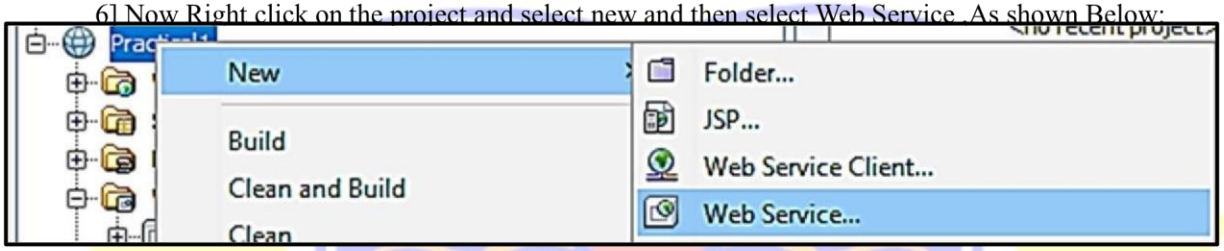
3] Now Give name to the Project Name, and click on next. You will get the following window then again, click on finish:



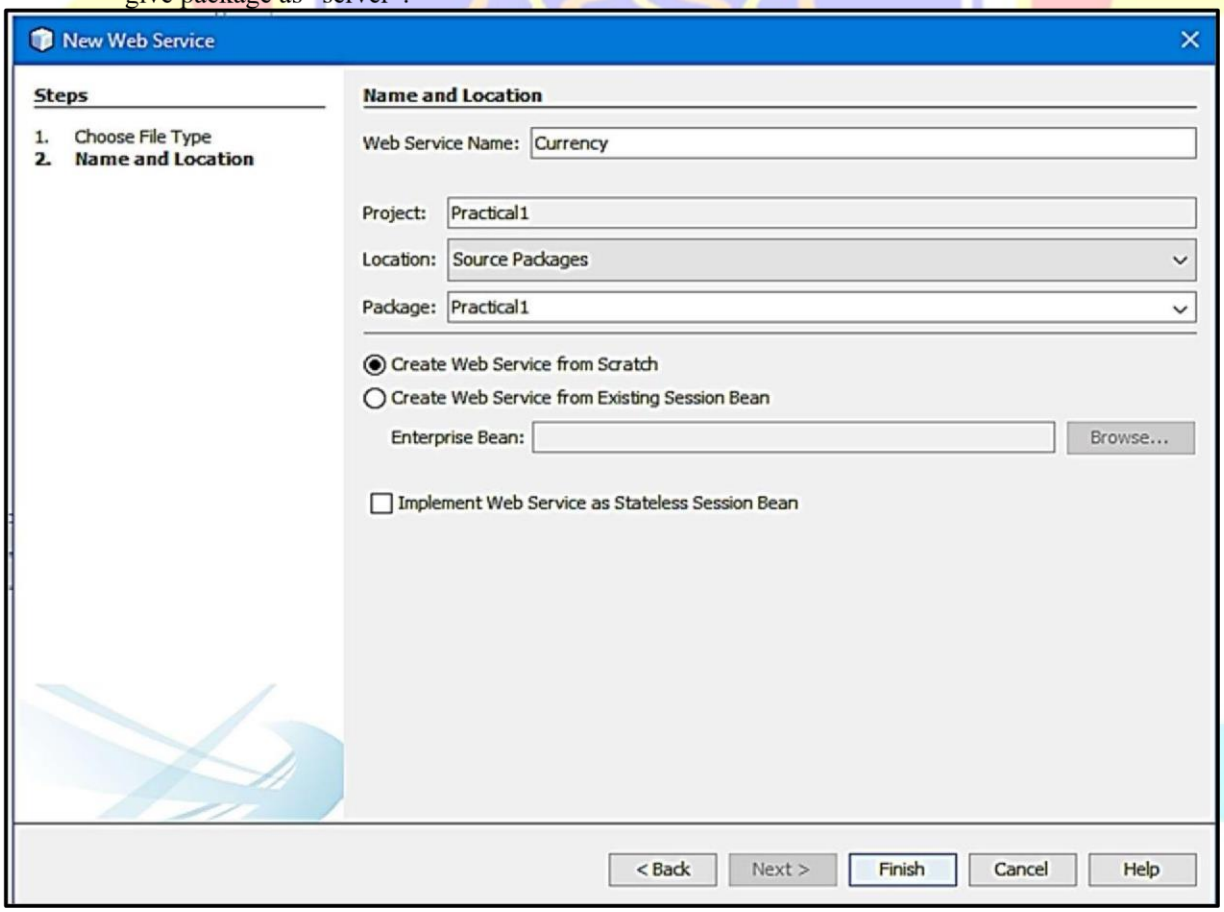
Don't click on any frameworks. Finish.



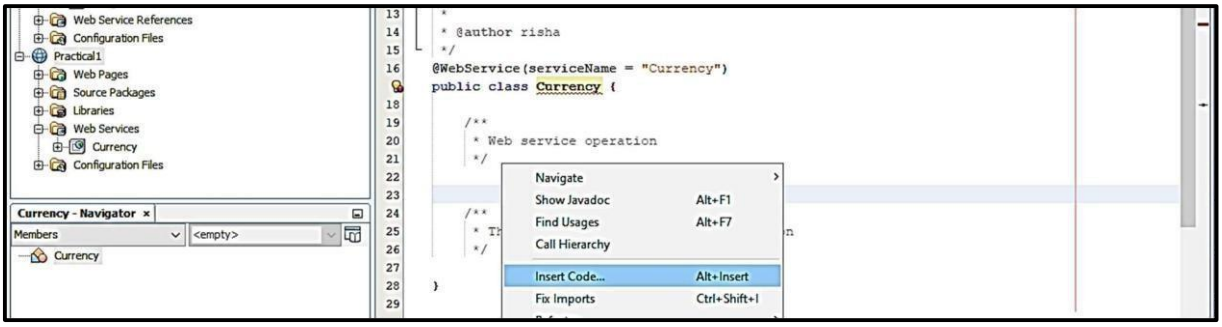
6] Now Right click on the project and select new and then select Web Service. As shown Below:



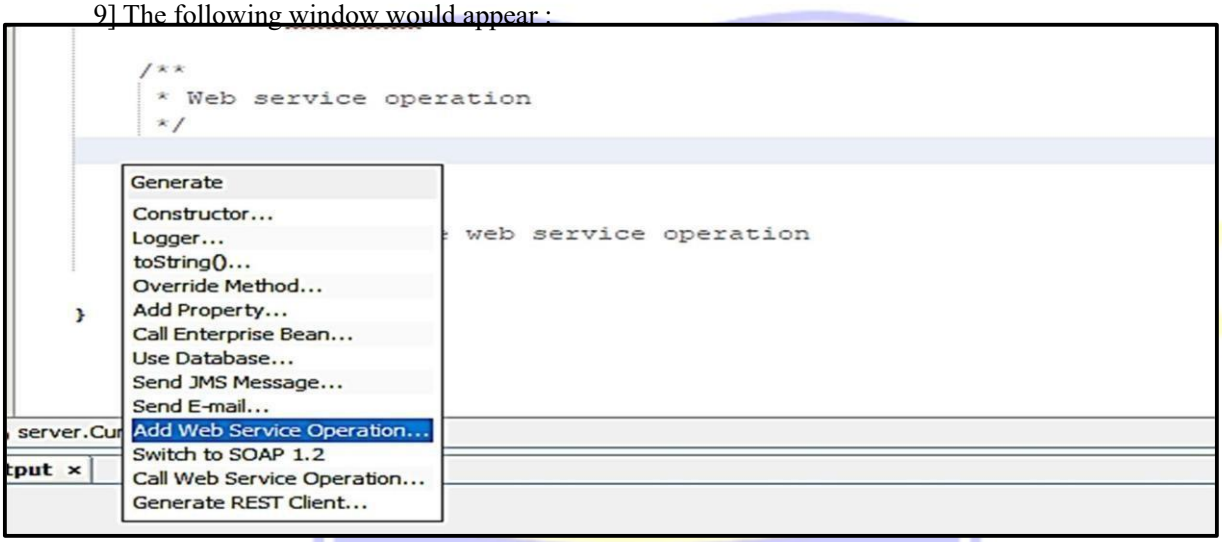
7] After clicking Web Service following window should appear, now give name to web service and give package as "server".



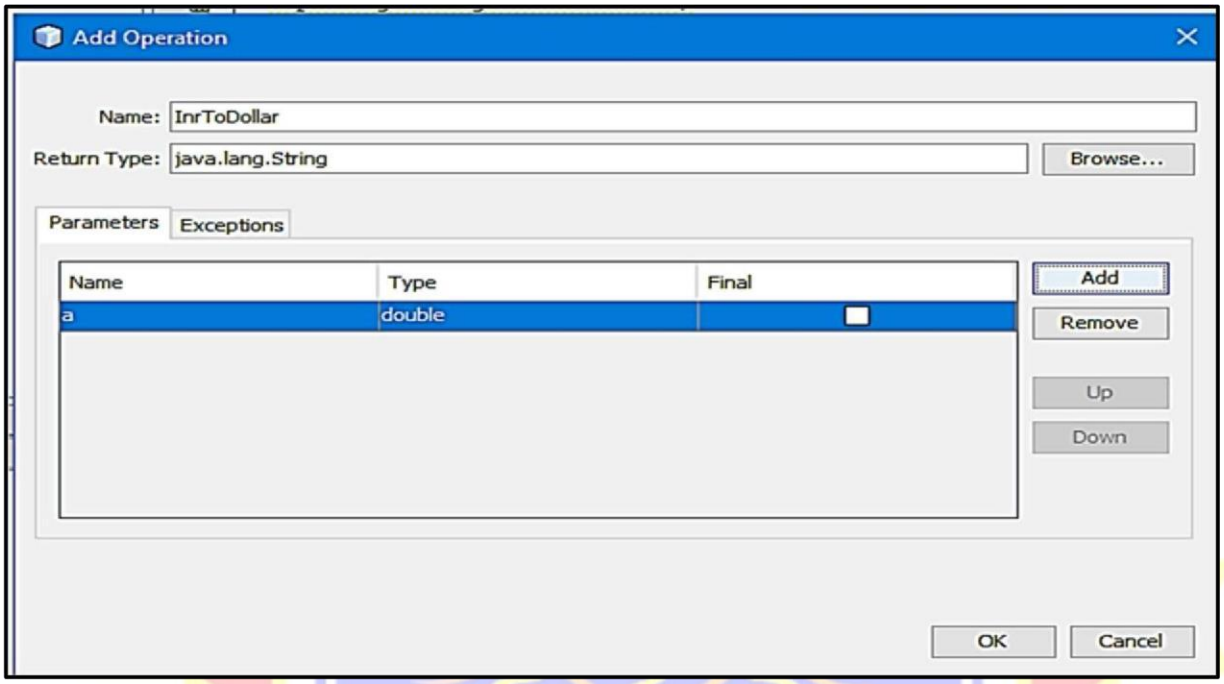
8] Now right click any where and click on insert code and select Add Web Service Operation :



9] The following window would appear :



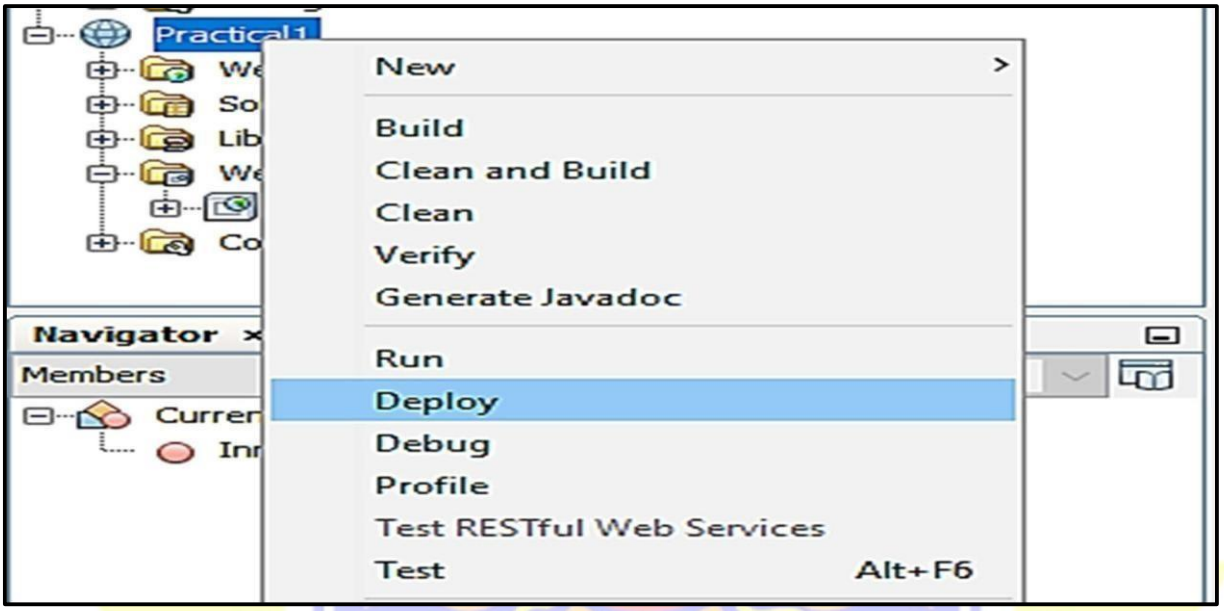
10] Just give name to the method or operation and click on add button to add parameters to the method as here we are converting dollar to rupees we should need only one parameter. give the name to the variable select data type as double and click on ok as shown below:



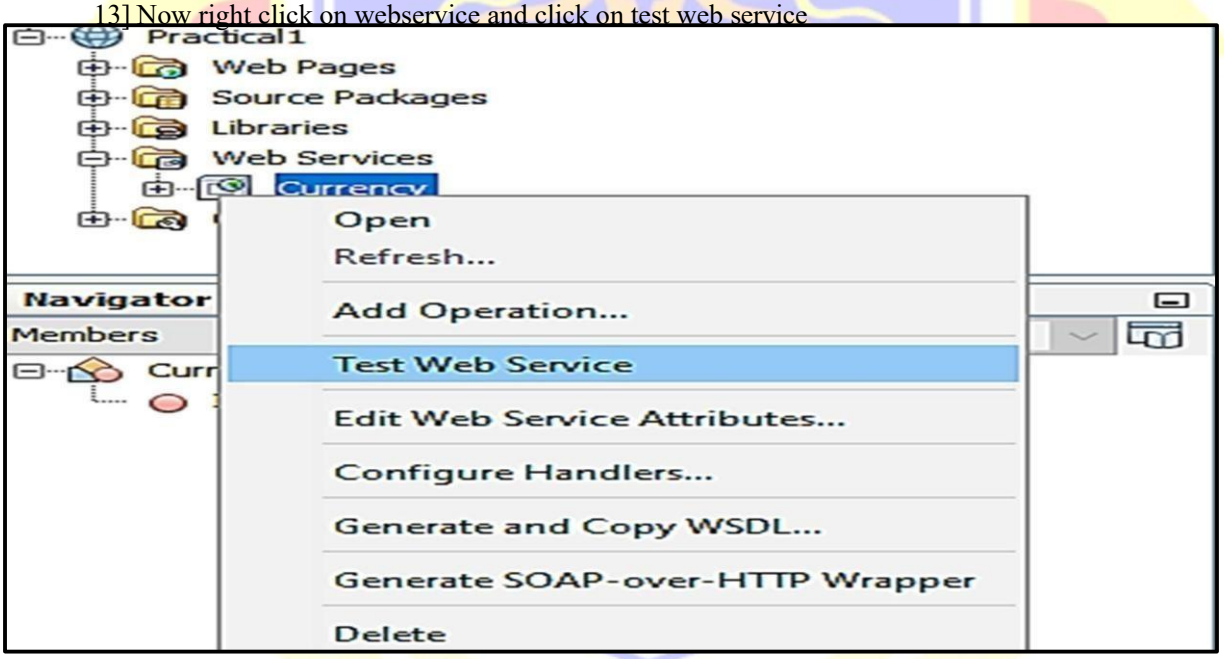
11] After clicking ok code will autogenerate, make changes in that code as mentioned below:

```
package server;

import javax.jws.WebService;
import javax.jws.WebMethod;
import javax.jws.WebParam;

@WebService(serviceName = "Currency")
public class Currency {

    @WebMethod(operationName = "InrToDollar")
    public String InrToDollar(@WebParam(name = "a") double a) {
        return "The Indian Rupees "+a+" in Dollars is"+(a/91.56);
    }

}
```

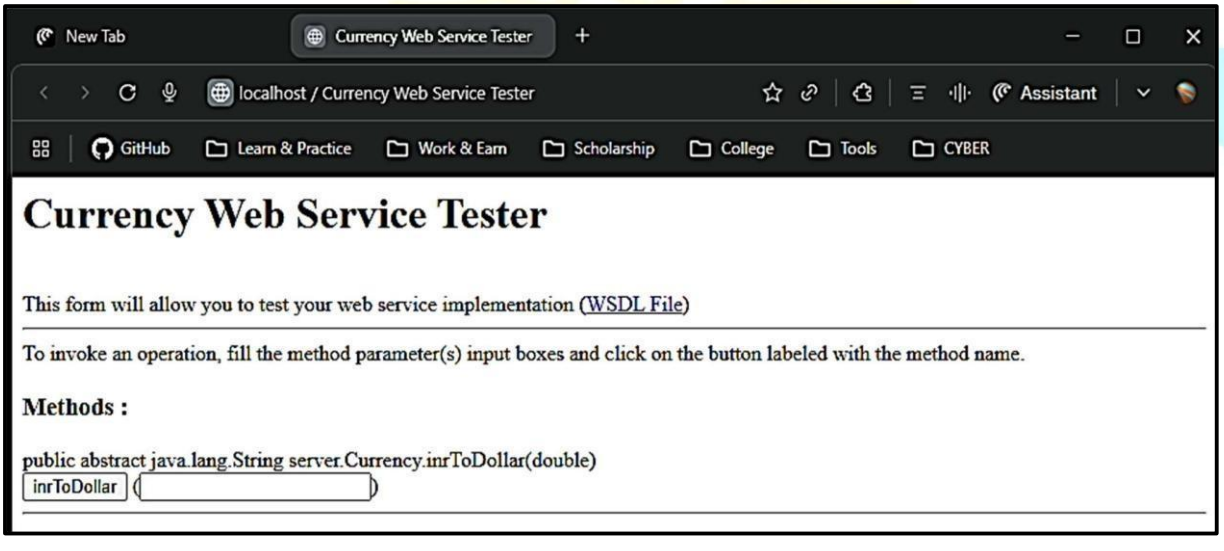
12] Now our web service is ready now right click on project and click on deploy:

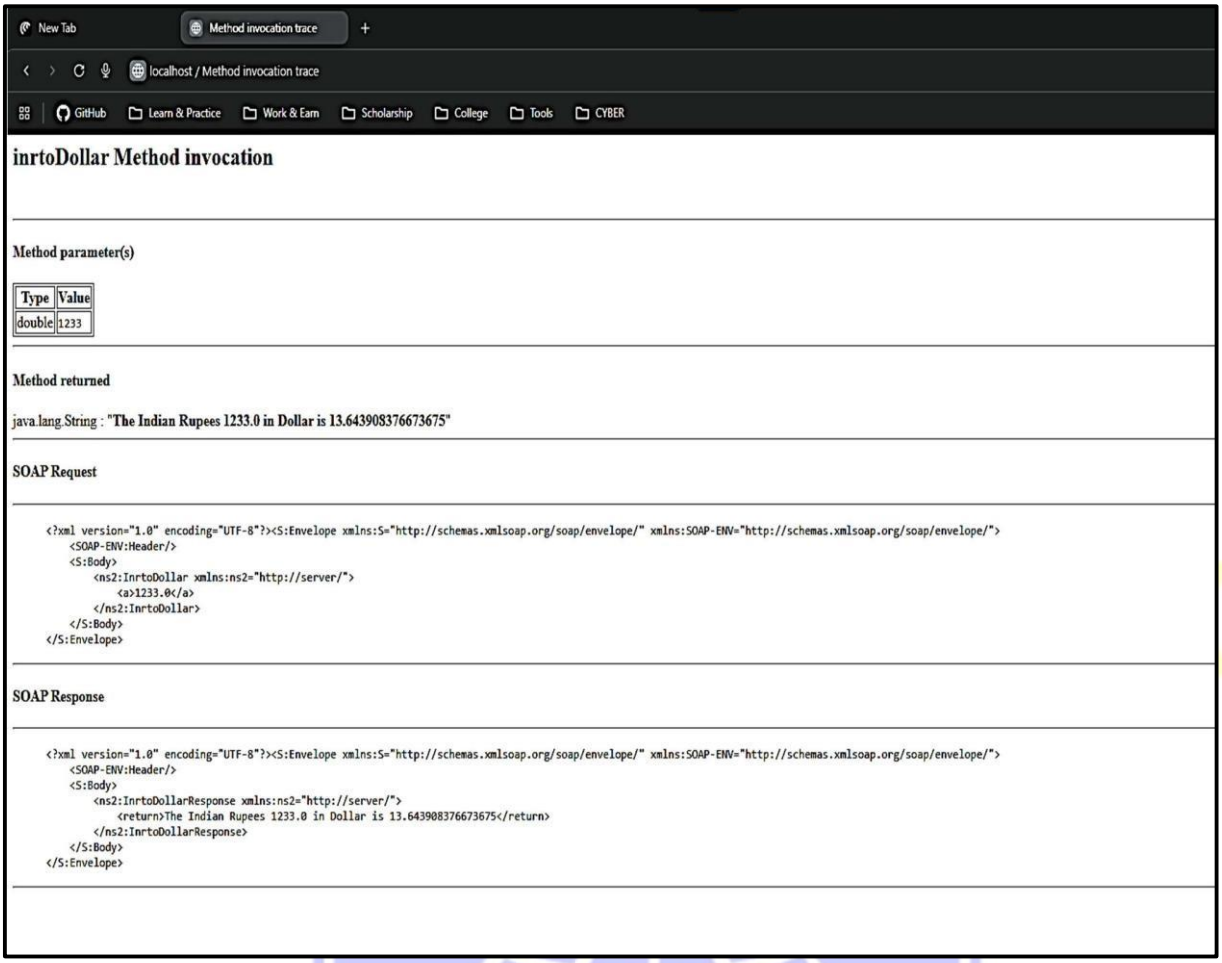


13] Now right click on webservice and click on test web service



Output:

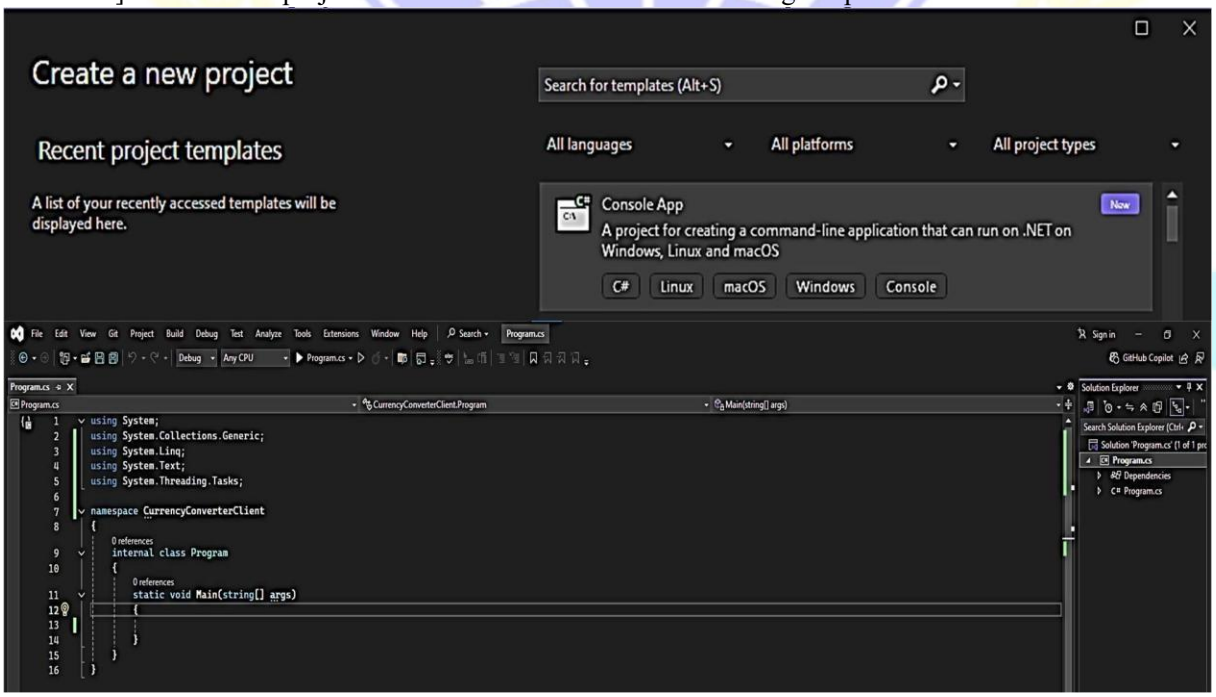




So this is how we created our web service and deployed it

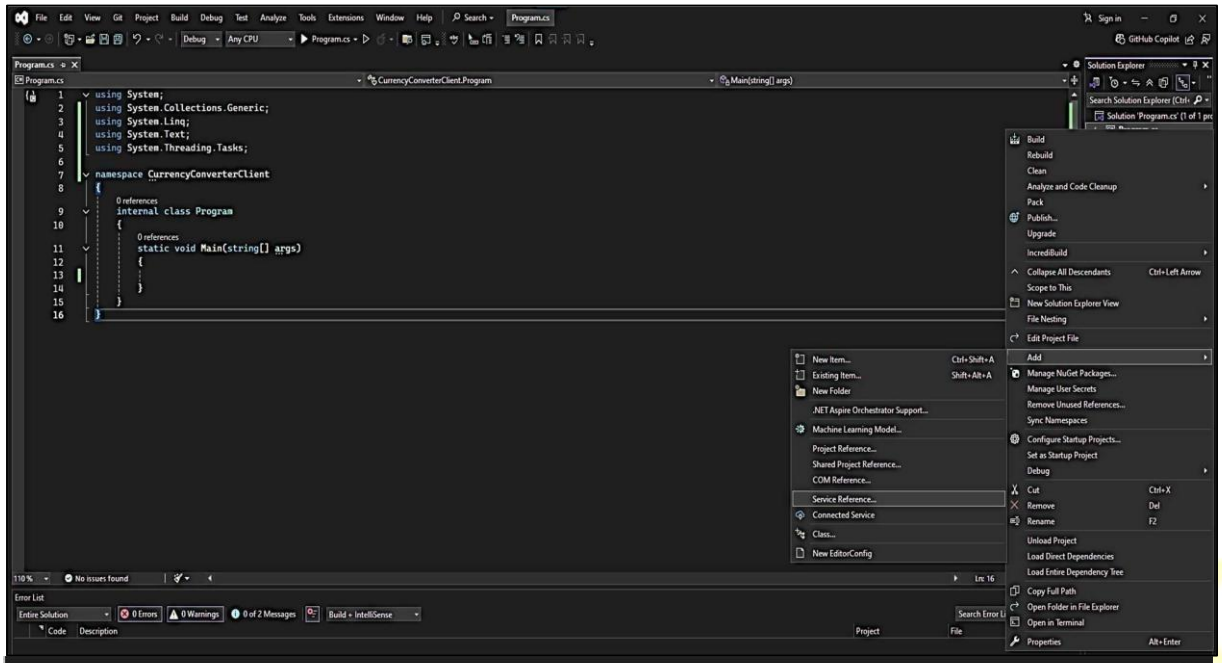
Now Calling the Service through .NET

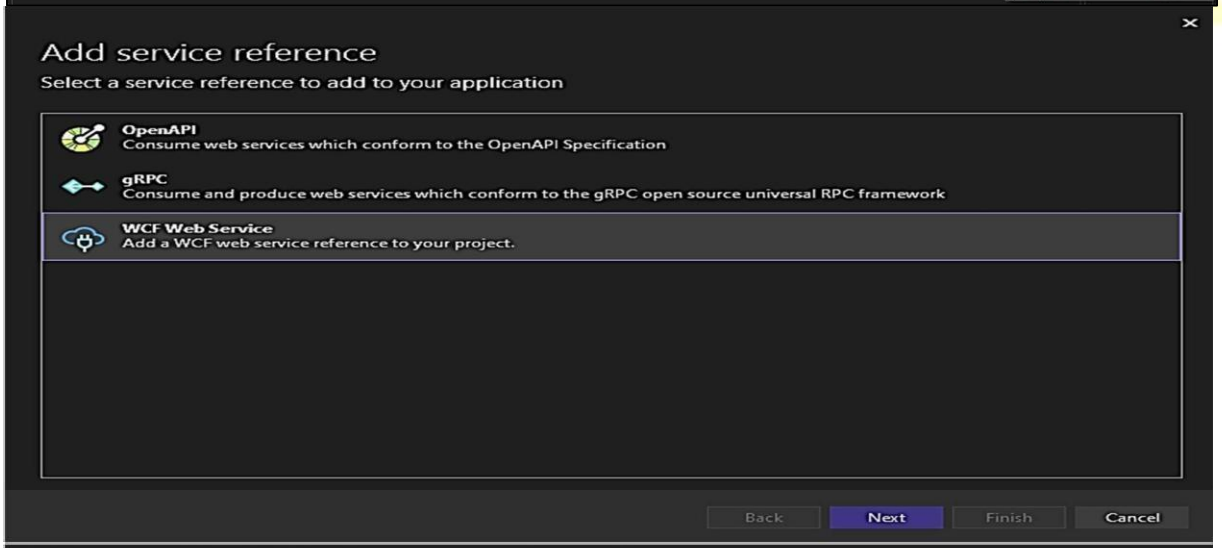
14] Create a new project in Visual Studio and use the following template :

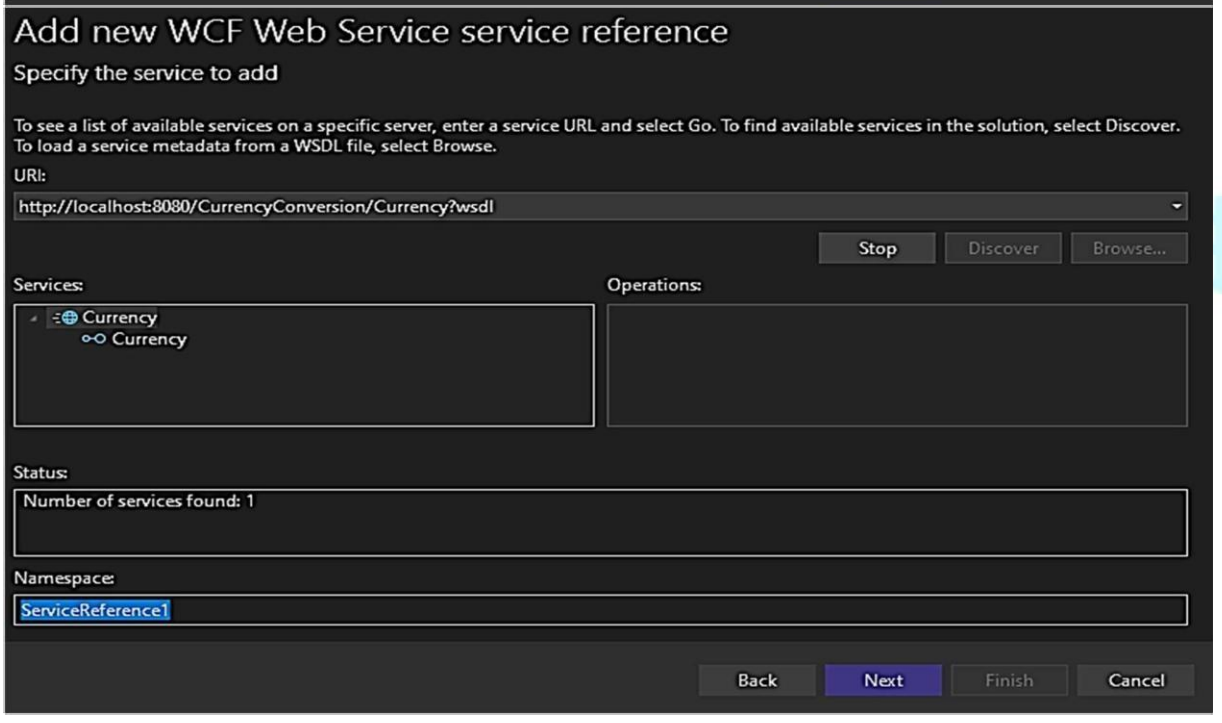


15] Click on Project > Add Service Reference

Note : the service from earlier must be hosted for Service Reference to be detected


using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace CurrencyConverterClient
internal class Program
static void Main(string[] args)
Add
Service Reference...
Build
Rebuild
Clean
Publish...
Upgrade
Manage NuGet Packages...
Manage User Secrets
Sync Namespaces
Configure Startup Projects...
Set as Startup Project
Debug
Cut
Remove
Rename
Unload Project
Load Direct Dependencies
Load Entire Dependency Tree
Copy Full Path
Open Folder in File Explorer
Open in Terminal
Properties

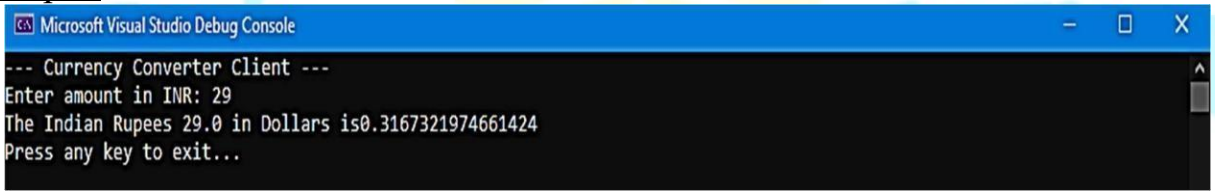

Add service reference
Select a service reference to add to your application
OpenAPI
Consume web services which conform to the OpenAPI Specification
gRPC
Consume and produce web services which conform to the gRPC open source universal RPC framework
WCF Web Service
Add a WCF web service reference to your project.
Back
Next
Finish
Cancel


Add new WCF Web Service service reference
Specify the service to add
To see a list of available services on a specific server, enter a service URL and select Go. To find available services in the solution, select Discover.
To load a service metadata from a WSDL file, select Browse.
URI:
http://localhost:8080/CurrencyConversion/Currency?wsdl
Stop
Discover
Browse...
Services:
Operations:
Currency
Currency
Status:
Number of services found: 1
Namespace:
ServiceReference1
Back
Next
Finish
Cancel

16] Modify the code:

```
using ServiceReference1;  using
System; using
System.Threading.Tasks;

namespace Currency_Converter
{
   internal class Program
   {

       static async Task Main(string[] args)
       {
CurrencyClient client = new CurrencyClient();

Console.WriteLine("--- Currency Converter Client ---");
Console.Write("Enter amount in INR: ");

          string input = Console.ReadLine();

          if (double.TryParse(input, out double inr))
          {              try
             {

                var response = await client.InrToDollarAsync(inr);


Console.WriteLine(response.Body.@return);
             }
             catch (Exception ex)
             {
Console.WriteLine("Error: " + ex.Message);
             }
          }
Console.WriteLine("Press any key to exit...");
Console.ReadKey();
       }
   } }
```

Output:



Learning Outcomes:

1. Ability to create SOAP web service.
2. Ability to consume web service in .NET.

Course Outcomes:

1. Understand cloud-based service communication.
2. Develop interoperable applications.

Conclusion:

Successfully created and consumed currency converter web service using Java and .NET.

Viva Questions:

1. What is SOAP?
2. What is WSDL?
3. What is platform independence?

For Faculty use:

Correction Parameters	Formative Assessment[40%]	Timely Completion of Practical[40%]	Attendance Learning Attitude[20%]

Practical 2

Aim: Create a Simple SOAP service. Steps:

Tools:

- NetBeans
- Java

Learning Objectives:

1. To understand SOAP architecture.
2. To learn how to create web methods.
3. To deploy web service on server.
4. To test SOAP service using browser.

Theory:

SOAP Web Service is a protocol-based service that allows communication between applications over the internet. It uses XML-based messaging format and works over HTTP protocol.

Main components of SOAP:

- SOAP Envelope
- SOAP Header
- SOAP Body
- WSDL (Web Service Description Language)

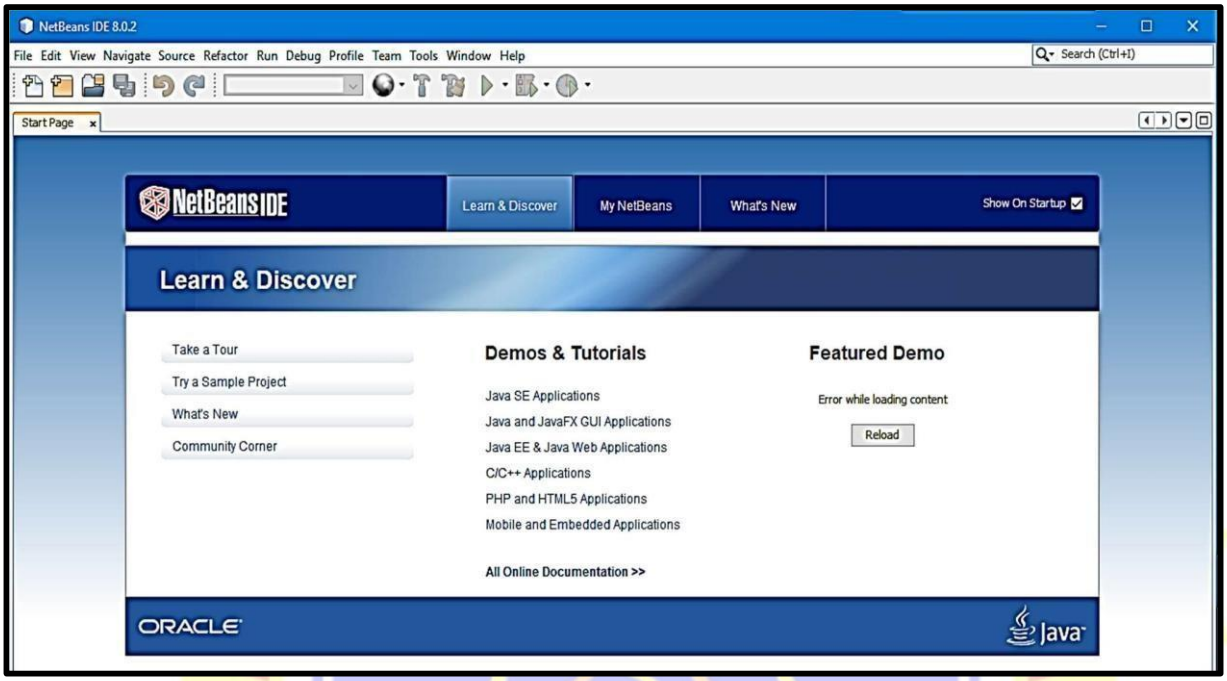
In this practical:

- A Java Web Application is created.
- A Web Service class is added.
- A method is defined inside the service.
- The application is deployed on server.
- The service is tested via generated WSDL file.

SOAP services are highly secure and reliable but slightly heavier than REST services.

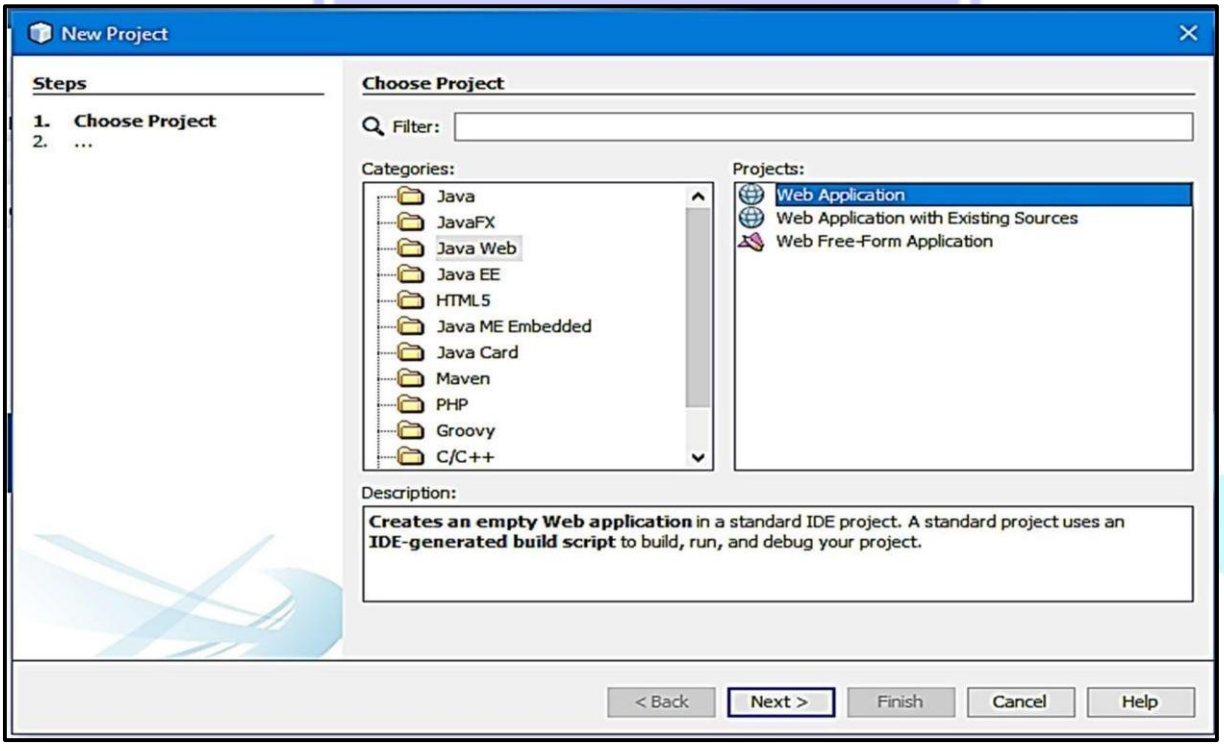
Steps:

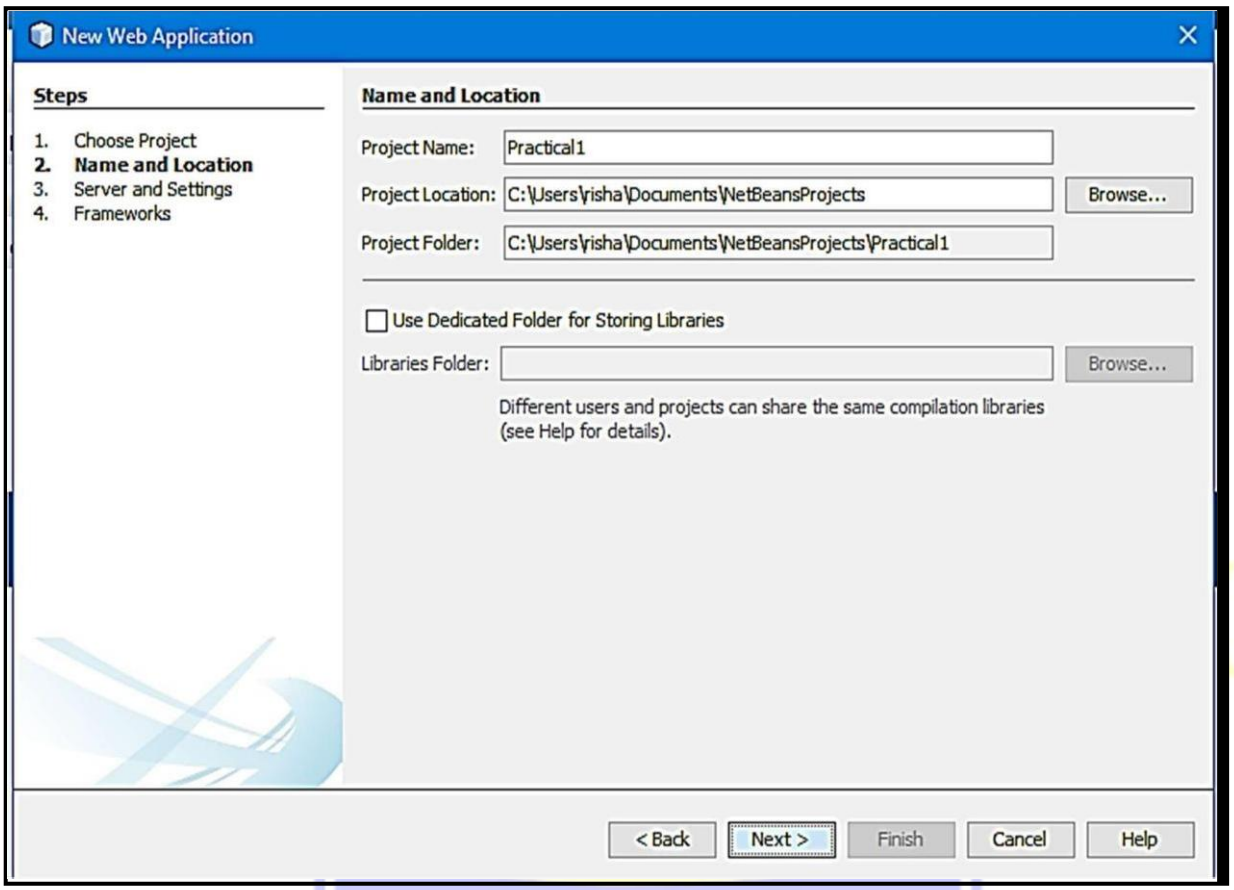
1] Open the NetBeans, and you will get the following screen. Close the start page.



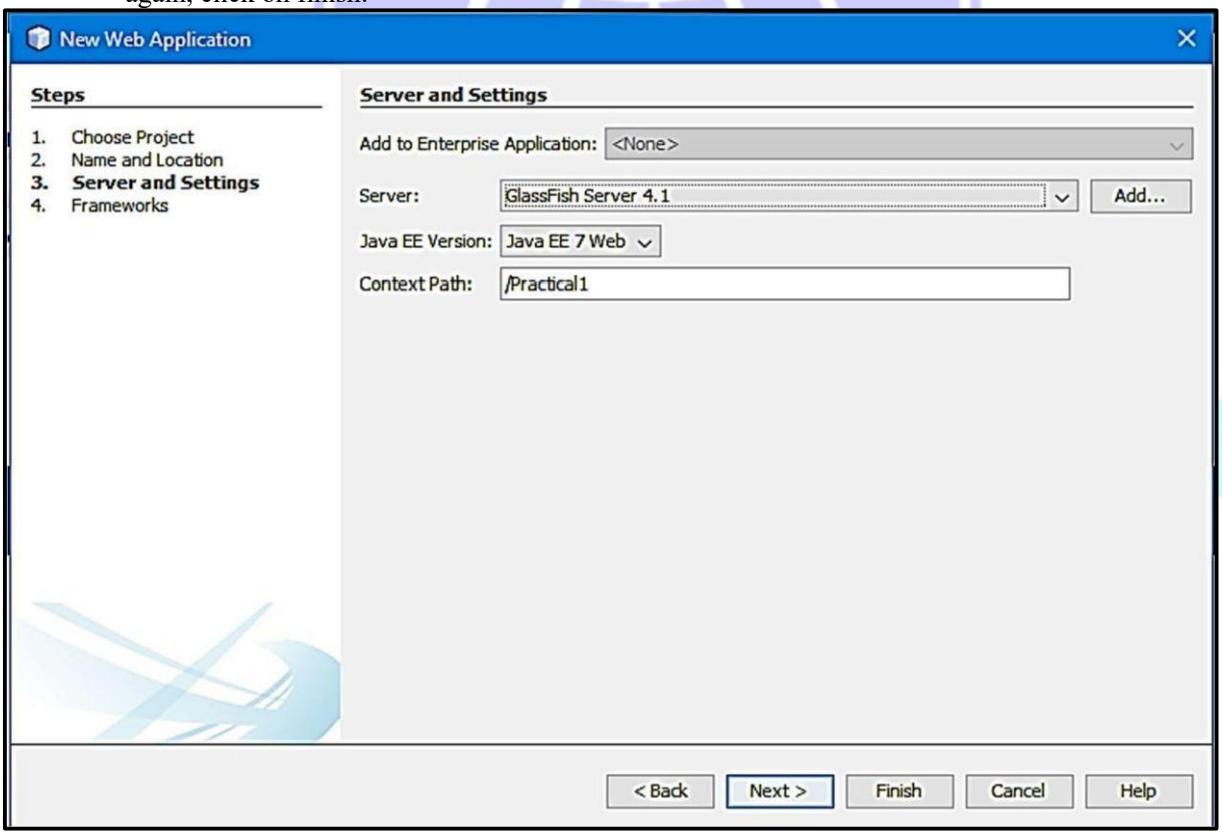
2] Now click on the file tab and click on new project you will get the following screen:

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

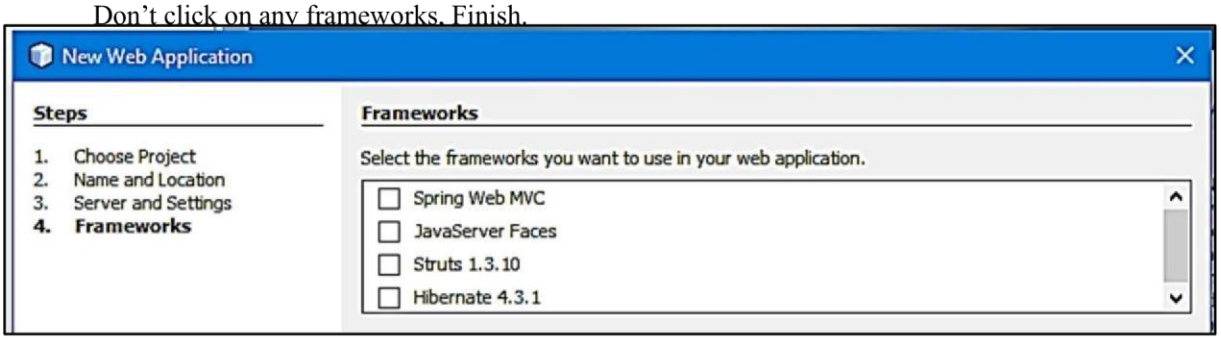




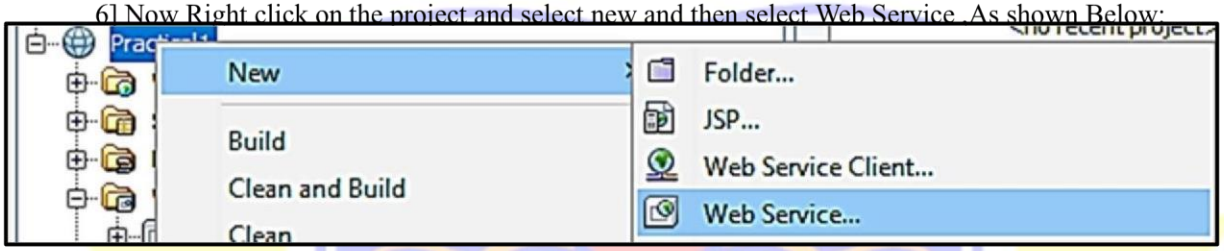
4] Now Give name to the Project Name, and click on next. You will get the following window then again, click on finish:



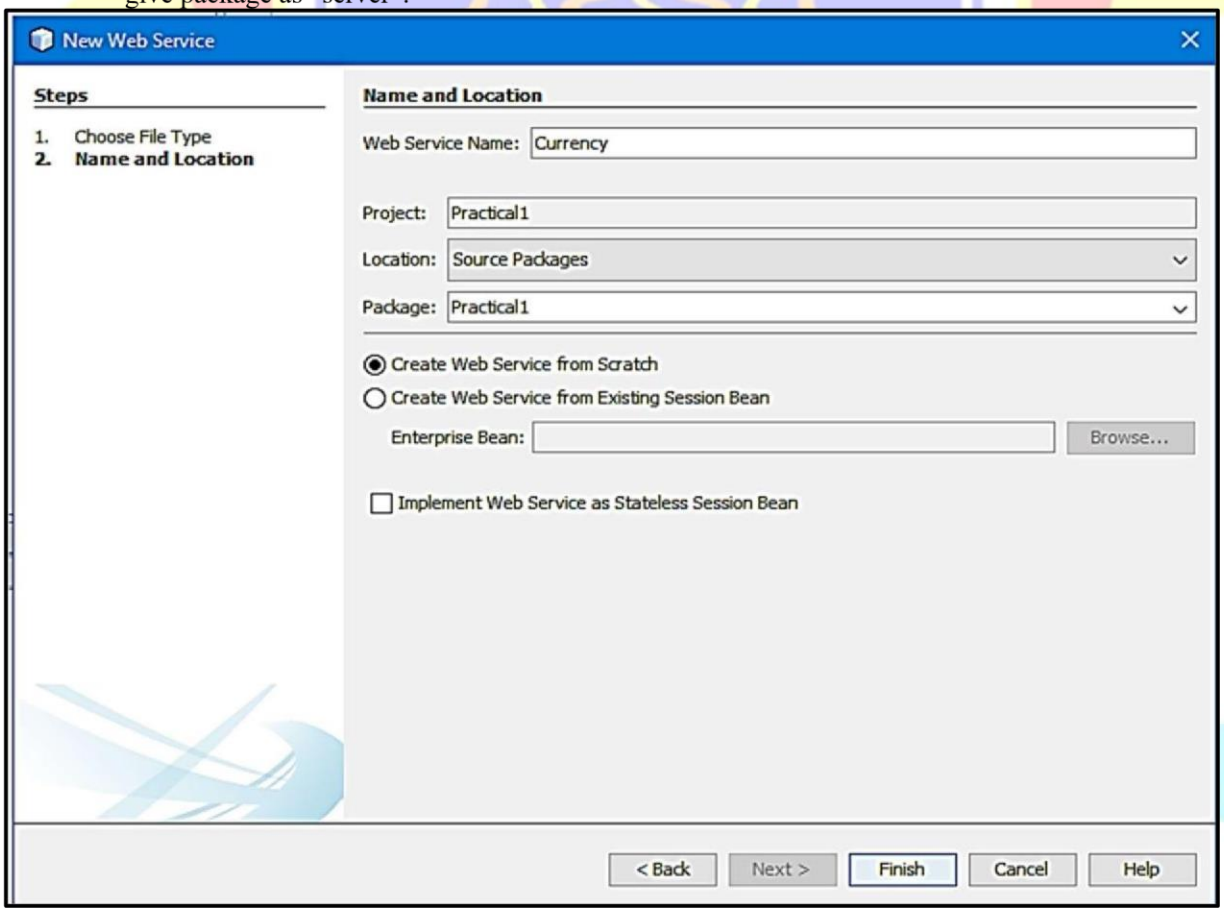
Don't click on any frameworks. Finish.



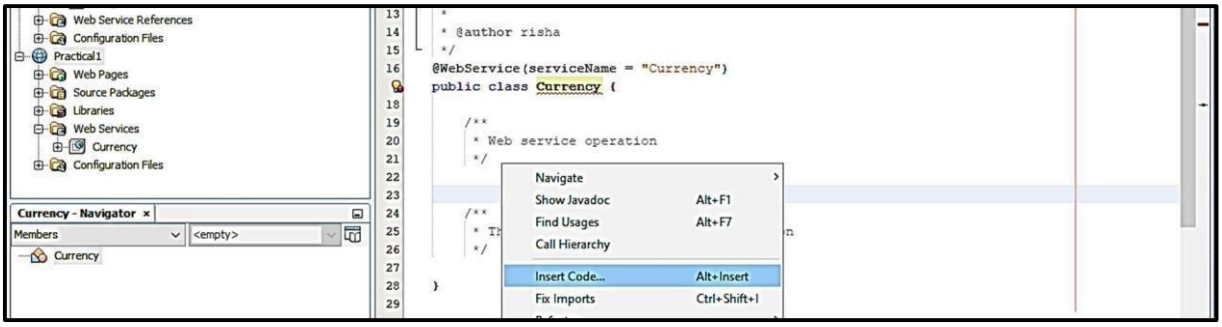
6] Now Right click on the project and select new and then select Web Service. As shown Below:



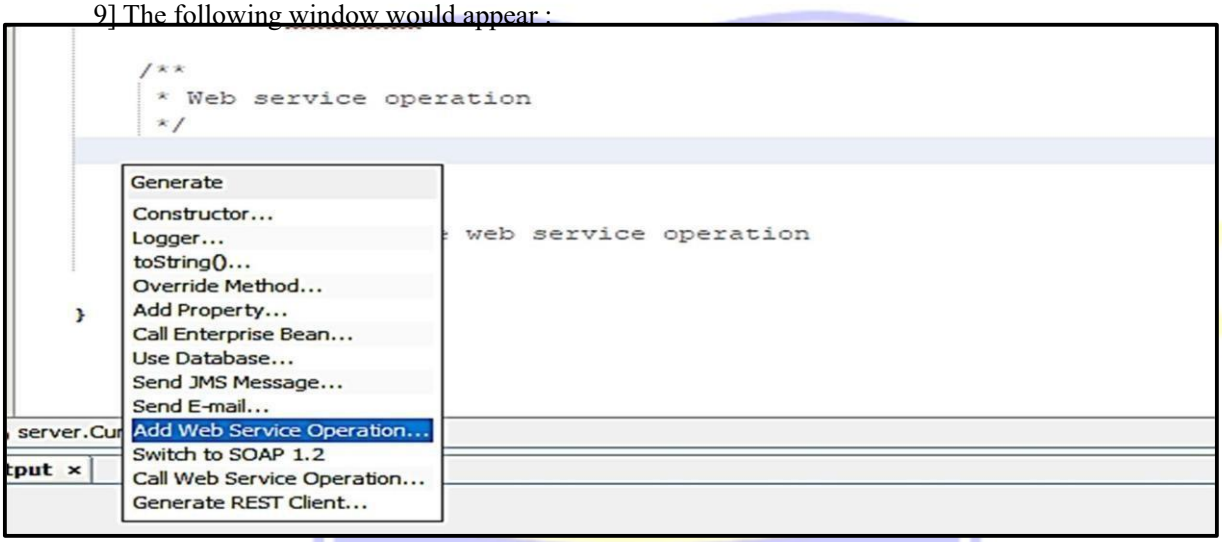
7] After clicking Web Service following window should appear , now give name to web service and give package as "server".



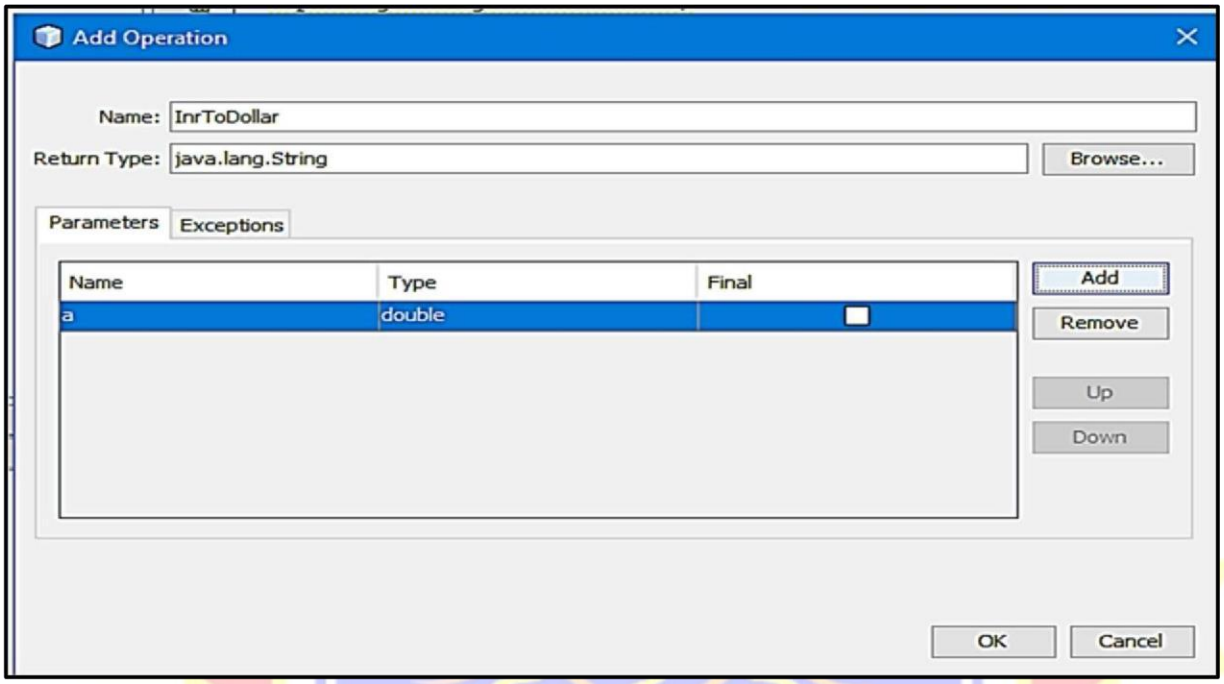
8] Now right click anywhere and click on insert code and select Add Web Service Operation :



9] The following window would appear :



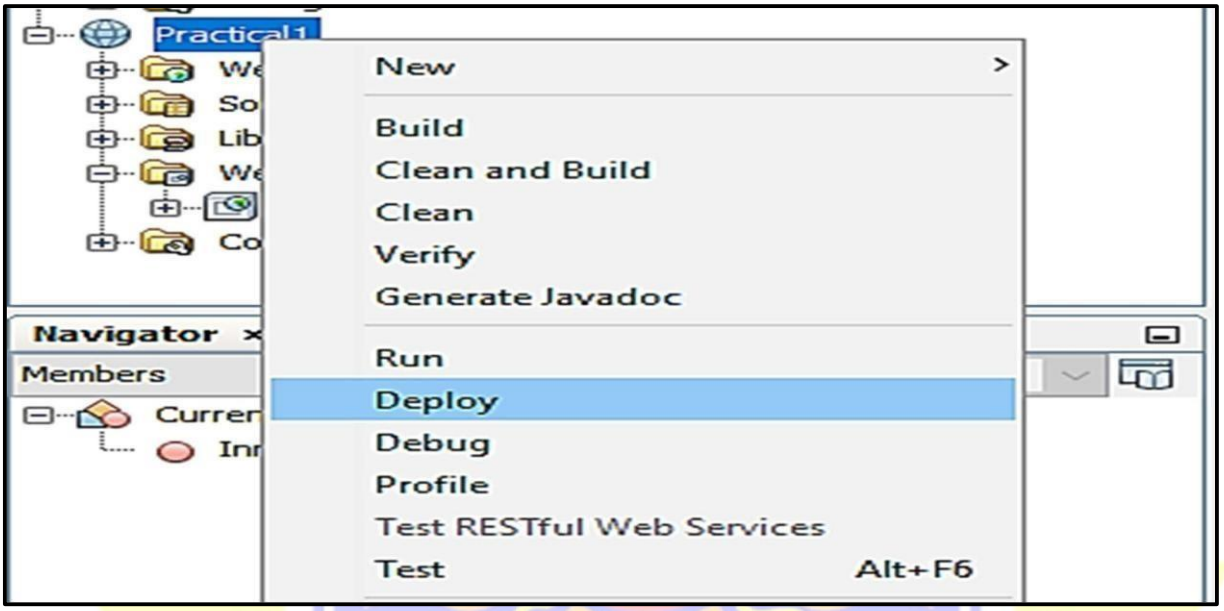
10] Just give name to the method or operation and click on add button to add parameters to the method as here we are converting dollar to rupees we should need only one parameter. give the name to the variable select data type as double and click on ok as shown below:



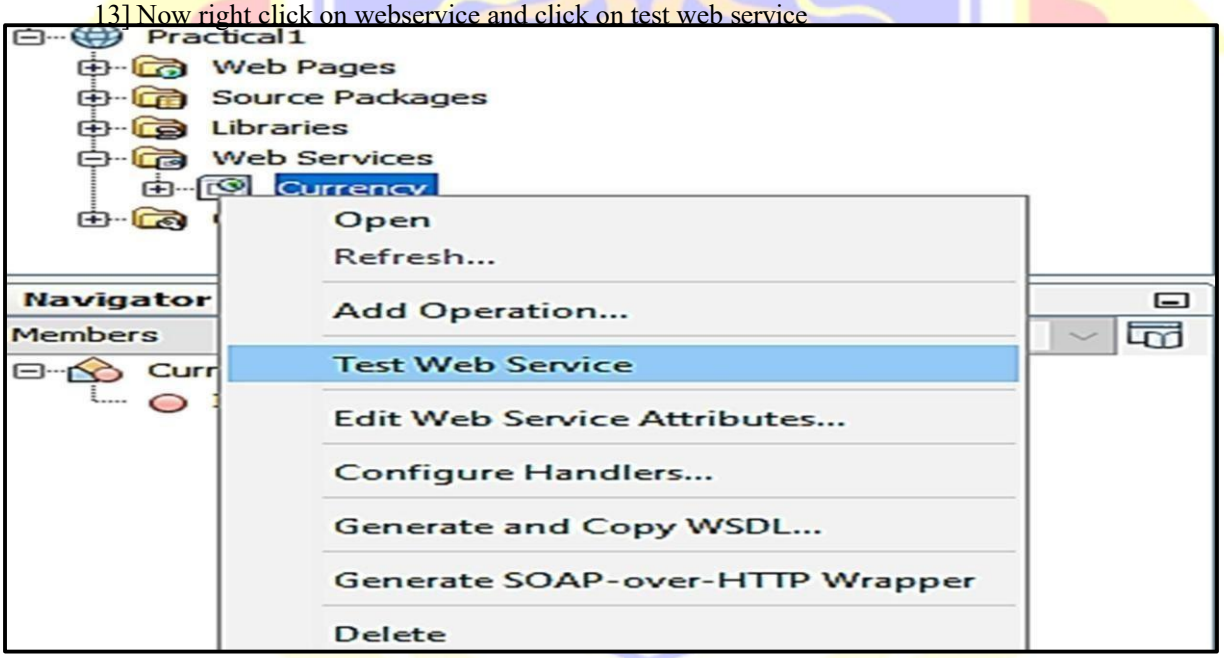
11] After clicking ok code will autogenerate, make changes in that code as mentioned below:

```
package server;

import javax.jws.WebService;
import javax.jws.WebMethod;
import javax.jws.WebParam;

@WebService(serviceName = "Currency")
public class Currency {

    @WebMethod(operationName = "InrToDollar")
    public String InrToDollar(@WebParam(name = "a") double a) {
        return "The Indian Rupees "+a+" in Dollars is"+(a/91.56);
    }

}
```

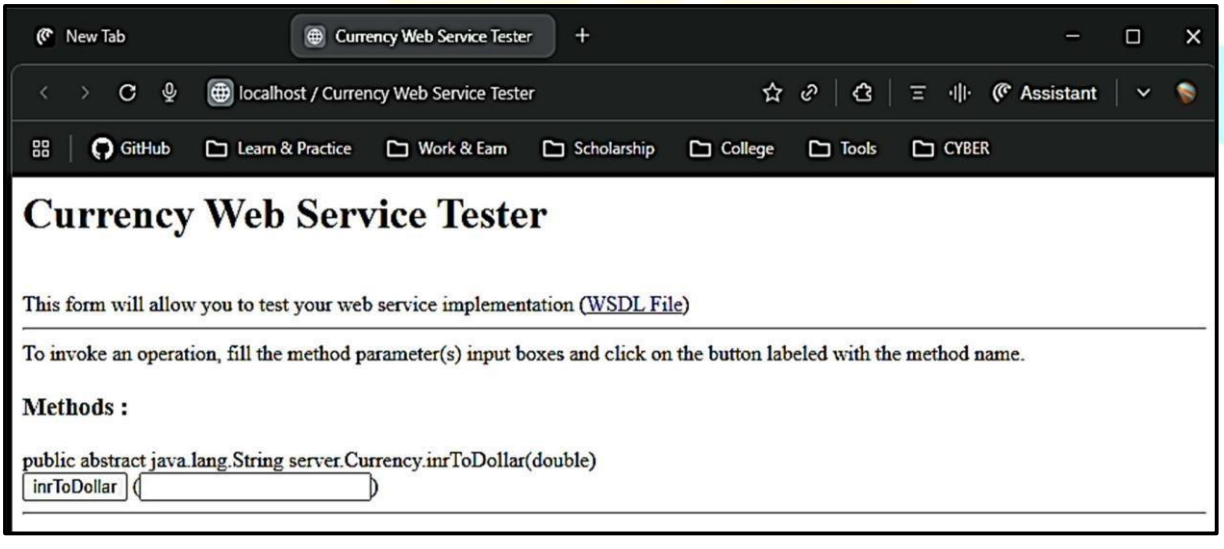
12] Now our web service is ready now right click on project and click on deploy:

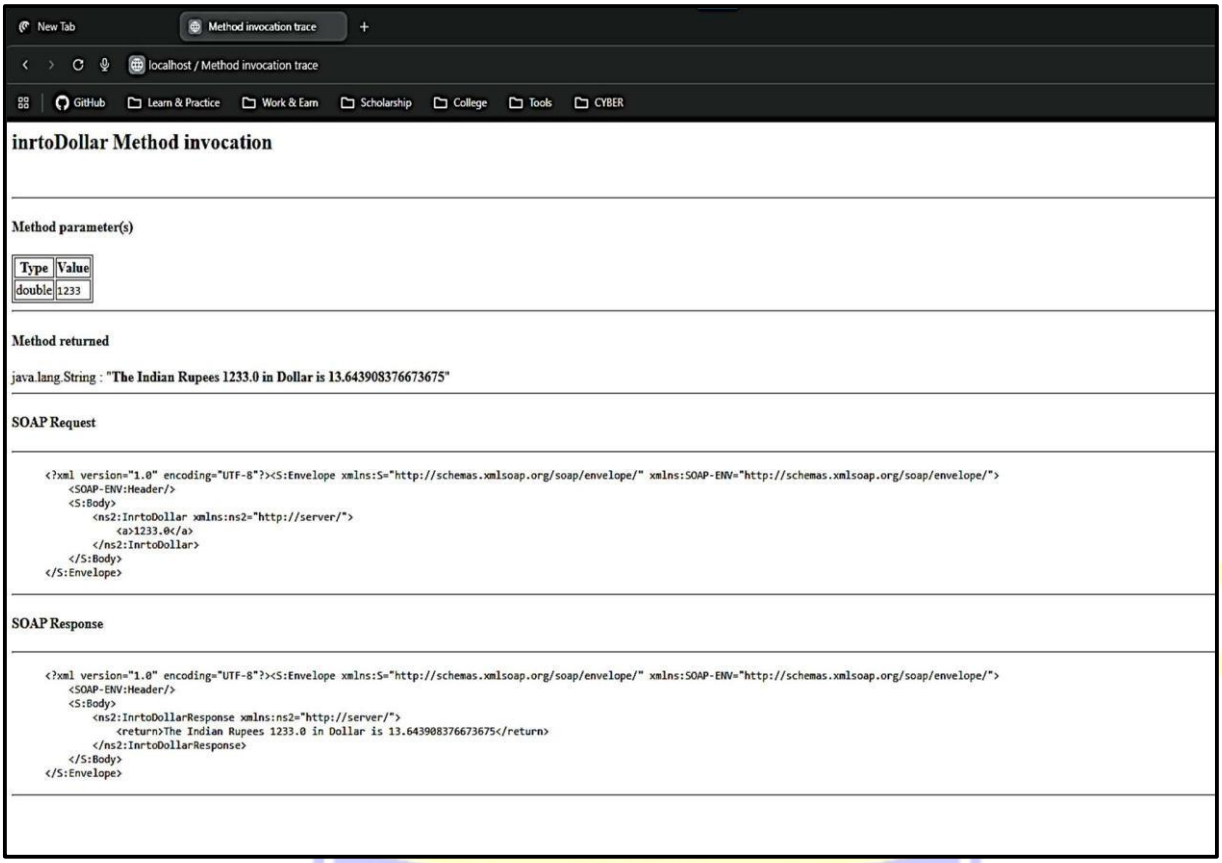


13] Now right click on webservice and click on test web service



Output:





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

Learning Outcomes:

1. Ability to create SOAP-based services.
2. Understanding of WSDL structure.

Course Outcomes:

1. Apply web service development techniques.
2. Understand service-oriented architecture.

Conclusion:

Successfully developed and deployed a SOAP web service.

Viva Questions:

1. What is SOAP Envelope?
2. What is WSDL?
3. Difference between SOAP and REST?

For Faculty use:

Correction Parameters	Formative Assessment[40%]	Timely Completion of Practical[40%]	Attendance Learning Attitude[20%]

Practical 3

Aim: Create a Simple REST Service.
Steps:

Tools & Technologies Used:

- NetBeans
- Java
- JAX-RS

Learning Objectives:

1. To understand REST architecture.
2. To learn HTTP methods usage.
3. To create RESTful API using JAXRS.
4. To test REST services through browser.

Theory:

REST (Representational State Transfer) is a lightweight web service architecture. It uses HTTP methods like:

- GET – Retrieve data
- POST – Insert data
- PUT – Update data
- DELETE – Delete data

REST services are faster and simpler compared to SOAP. They use JSON or XML format.

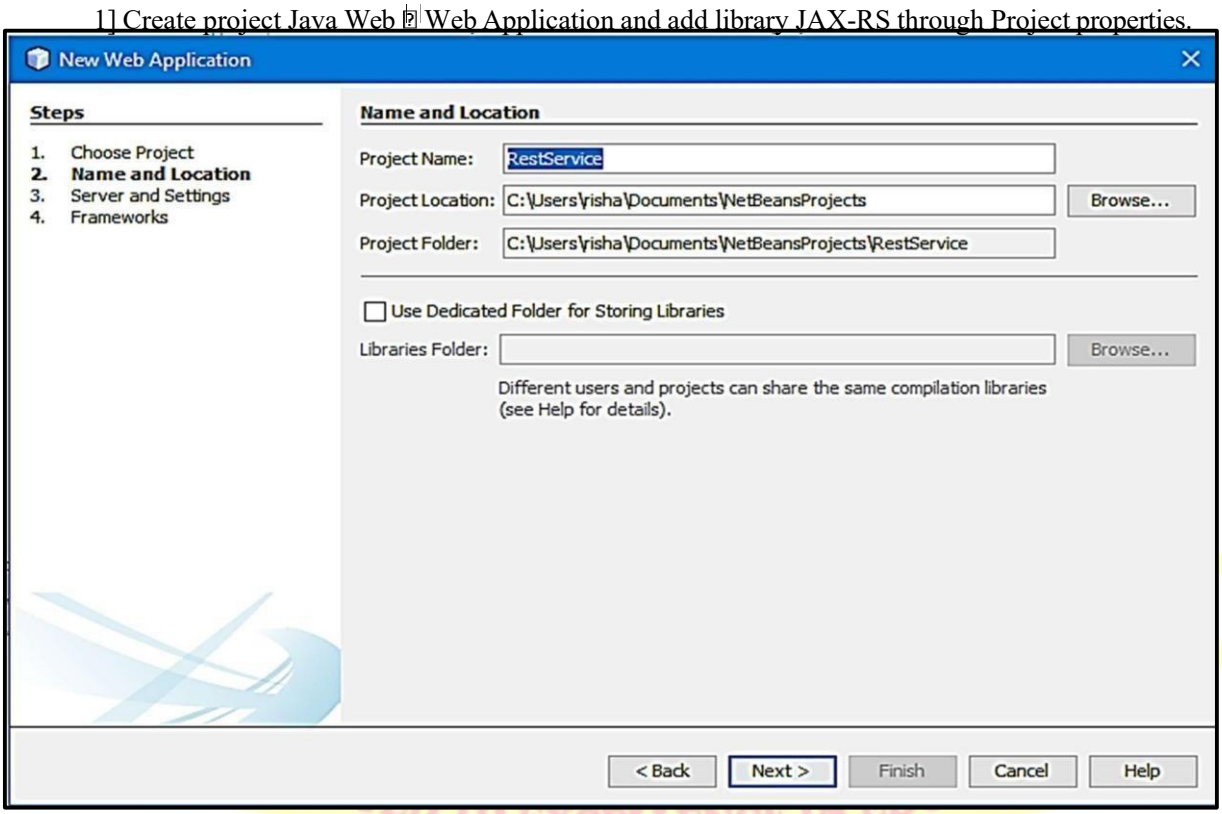
In this practical:

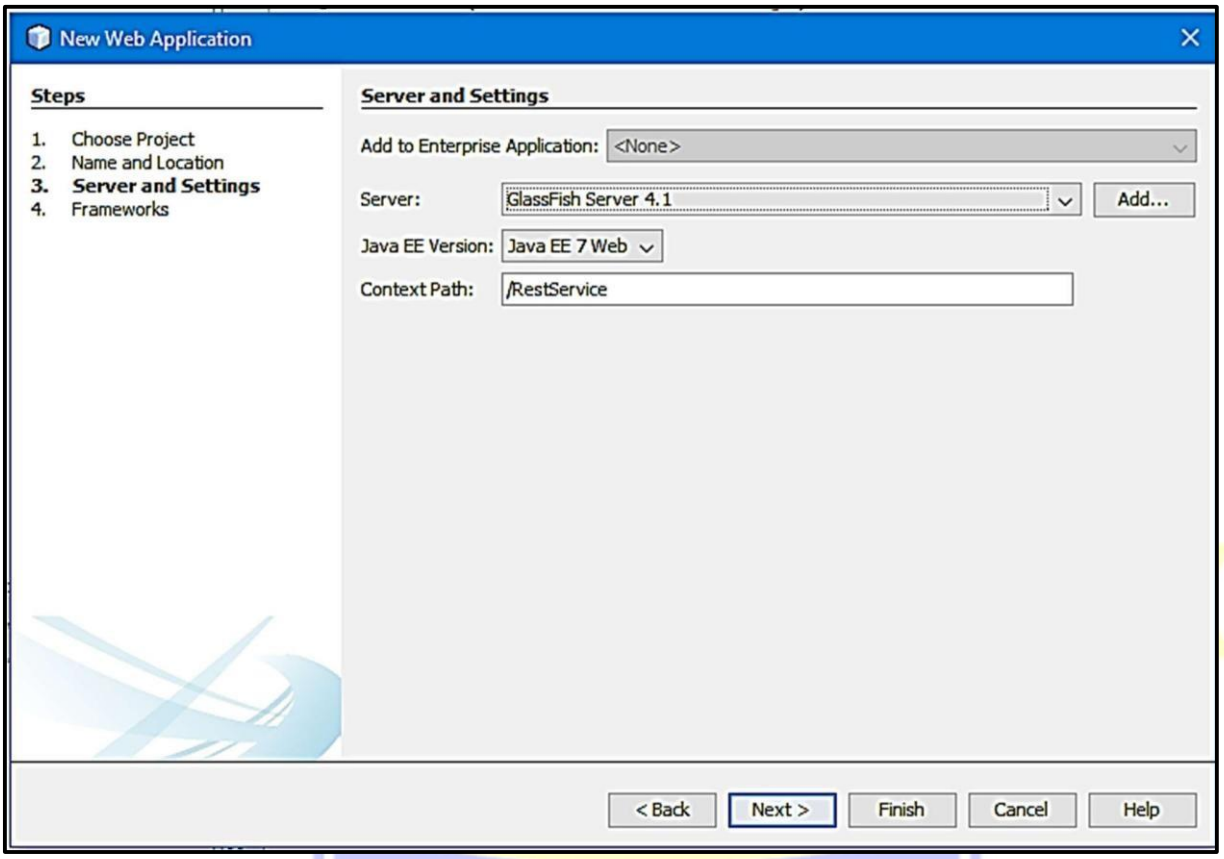
- A REST application class is created.
- A resource class is defined.
- A method is created to convert miles into kilometers.
- The service is accessed using URL.

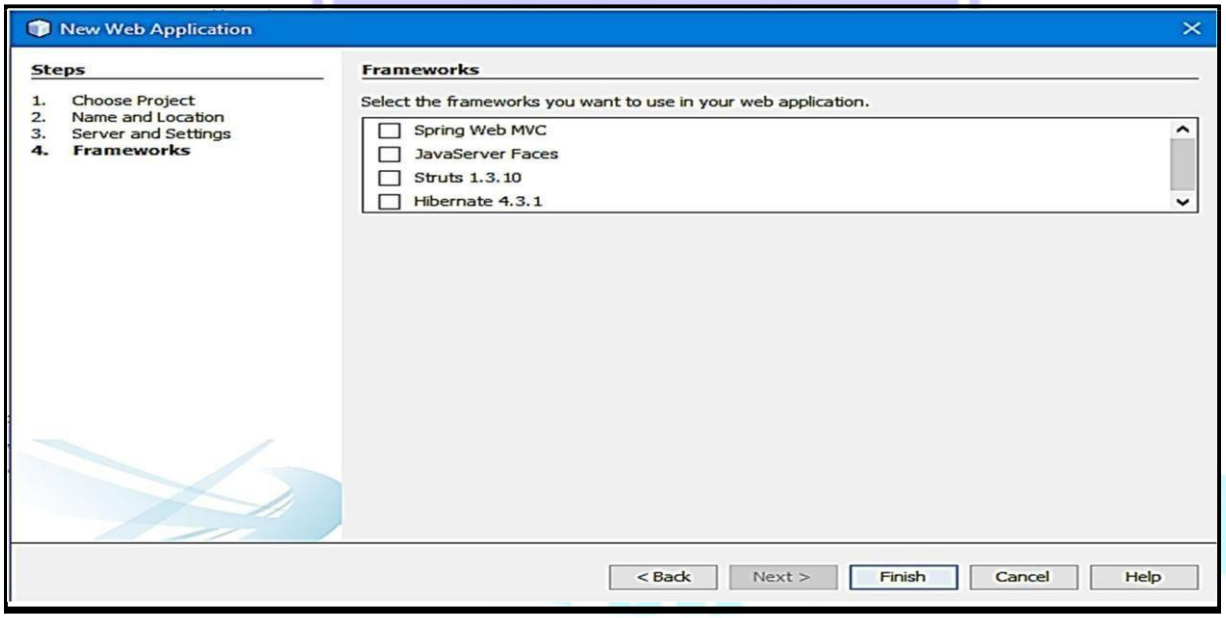
REST services are widely used in modern cloud applications.

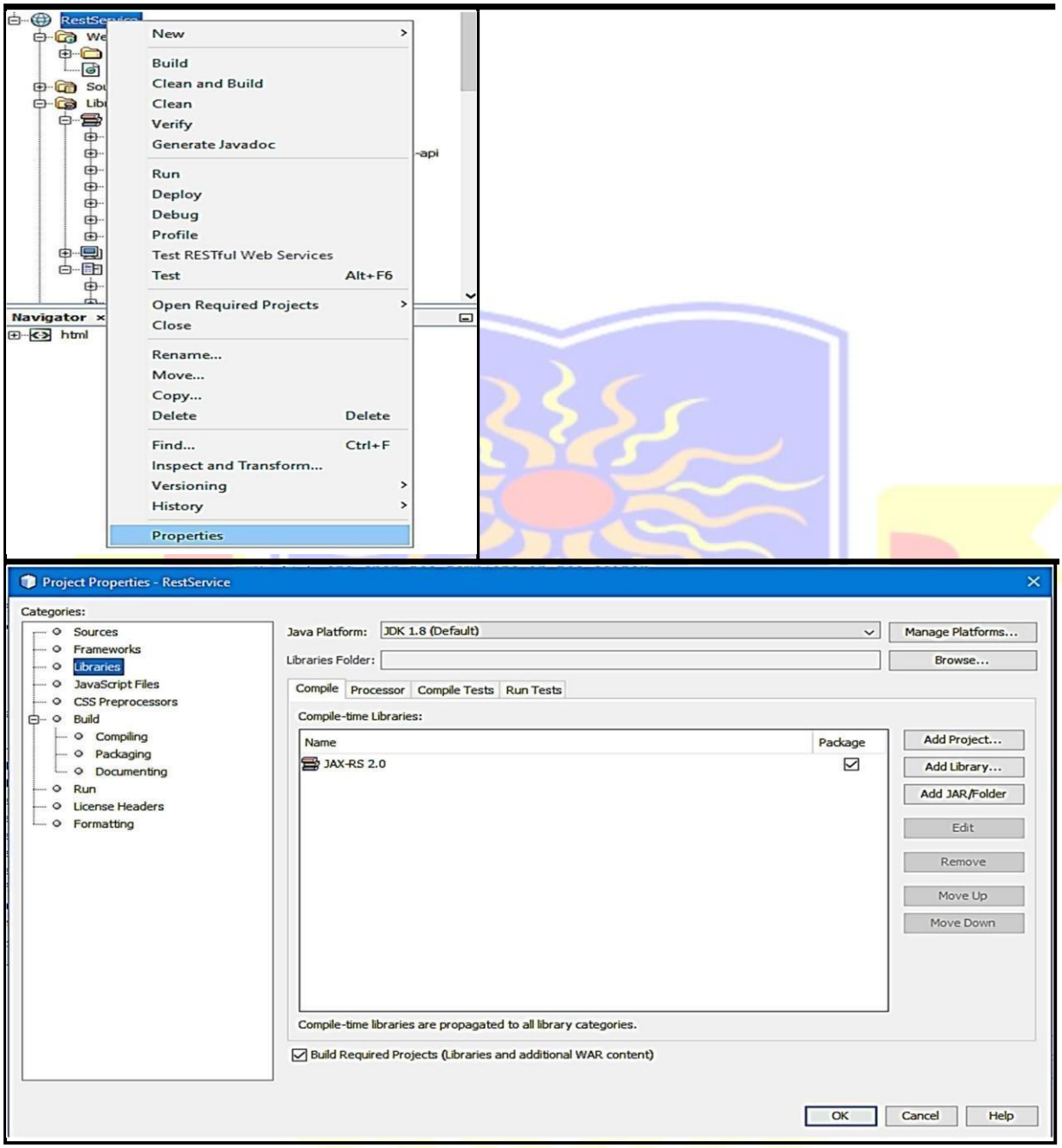
Steps:

1] Create project Java Web Web Application and add library JAX-RS through Project properties.

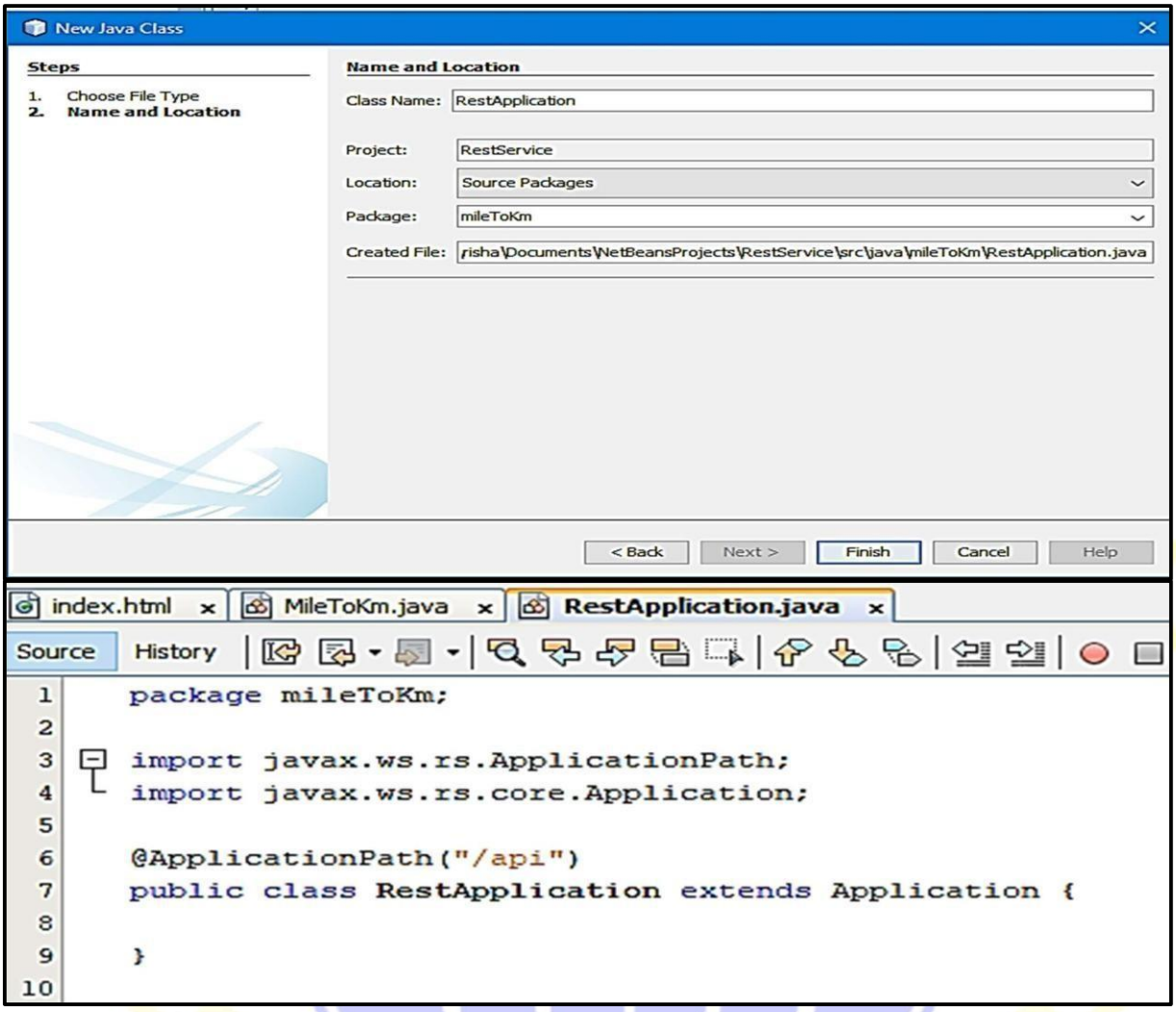



New Web Application
Steps
1. Choose Project
2. Name and Location
3. Server and Settings
4. Frameworks
Server and Settings
Add to Enterprise Application: <None>
Server: GlassFish Server 4.1
Add...
Java EE Version: Java EE 7 Web
Context Path: /RestService
< Back
Next >
Finish
Cancel
Help


New Web Application
Steps
1. Choose Project
2. Name and Location
3. Server and Settings
4. Frameworks
Frameworks
Select the frameworks you want to use in your web application.
Spring Web MVC
JavaServer Faces
Struts 1.3.10
Hibernate 4.3.1
< Back
Next >
Finish
Cancel
Help



2] Create a Java Class named RestApplication.

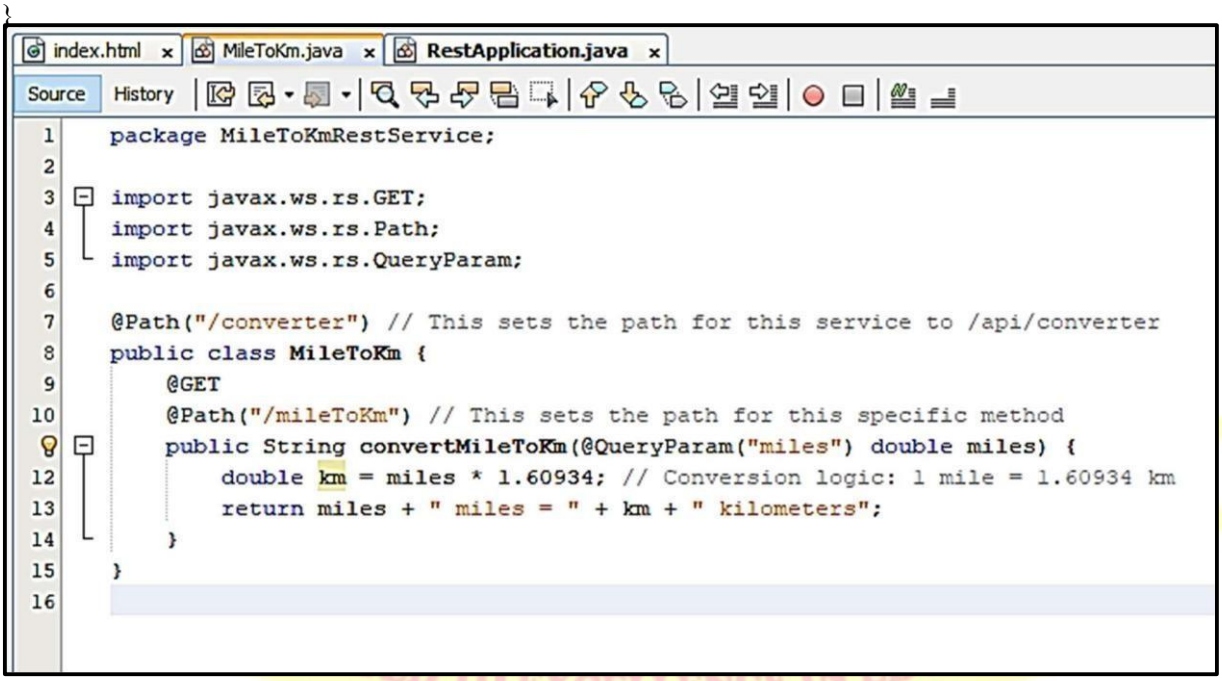



```
package mileToKm;

import javax.ws.rs.ApplicationPath;
import javax.ws.rs.core.Application;

@ApplicationPath("/api")
public class RestApplication extends Application {

}
```

3] Then create another Java class named MileToKmRestService.

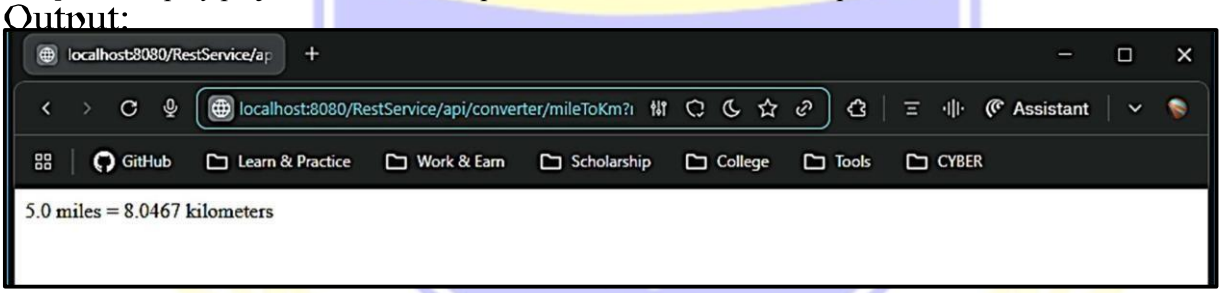
```
package MileToKmRestService;

import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.QueryParam;

@Path("/converter") // This sets the path for this service to /api/converter  public
class MileToKm {
   @GET
   @Path("/mileToKm") // This sets the path for this specific method     public
String convertMileToKm(@QueryParam("miles") double miles)
```

```
{       double km = miles * 1.60934; // Conversion logic: 1 mile = 1.60934
km
      return miles + " miles = " + km + " kilometers";
   }
}
```



4] Now Deploy project and Go to http://localhost:8080/RestService/api/converter/mileToKm?miles=5

Output:



Learning Outcomes:

1. Ability to build REST APIs.
2. Understanding of HTTP-based communication.

Course Outcomes:

1. Develop cloud-based REST services.
2. Understand API-based application design.

Conclusion:

Successfully created and tested RESTful web service.

Viva Questions:

1. What is REST?
2. What are HTTP methods?
3. Difference between SOAP and REST?

For Faculty use:

Correction Parameters	Formative Assessment[40%]	Timely Completion of Practical[40%]	Attendance Learning Attitude[20%]

Practical 4

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

Tools & Technologies Used:

- NetBeans
- JSP
- Google Cloud Console
- JavaScript

Learning Objectives:

1. To understand API integration.
2. To learn how to generate API keys.
3. To integrate third-party cloud services.
4. To display map using latitude and longitude.

Theory:

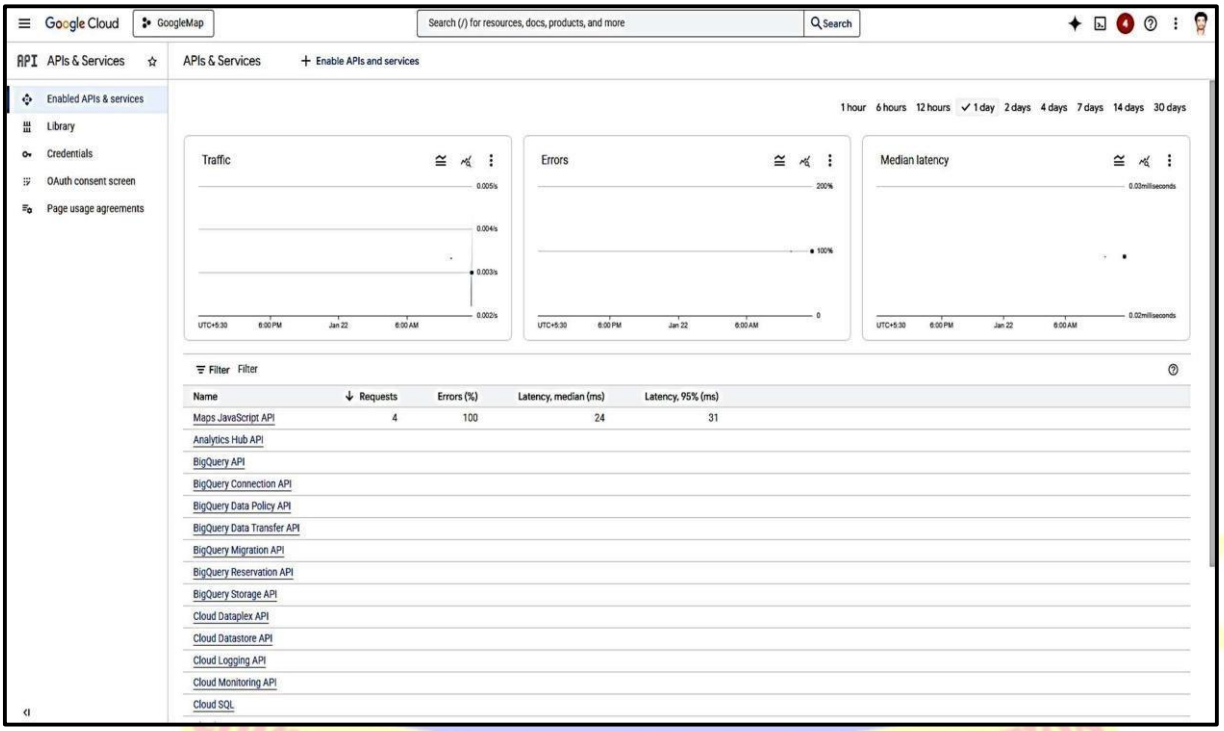
API (Application Programming Interface) allows applications to interact with external services. Google Maps API provides mapping services including:

- Location search
- Map display
- Marker placement Steps involved:
- Enable API in Google Cloud Console.
- Generate API key.
- Embed script in JSP.
- Use JavaScript to initialize map.

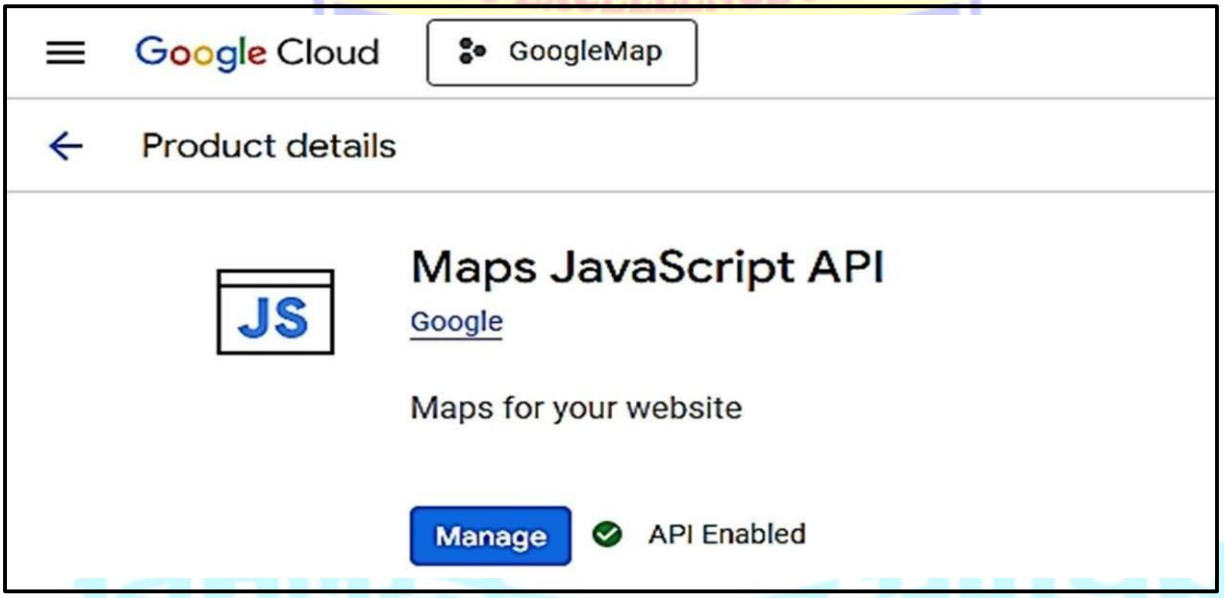
This practical demonstrates real-world cloud service integration.

Steps:

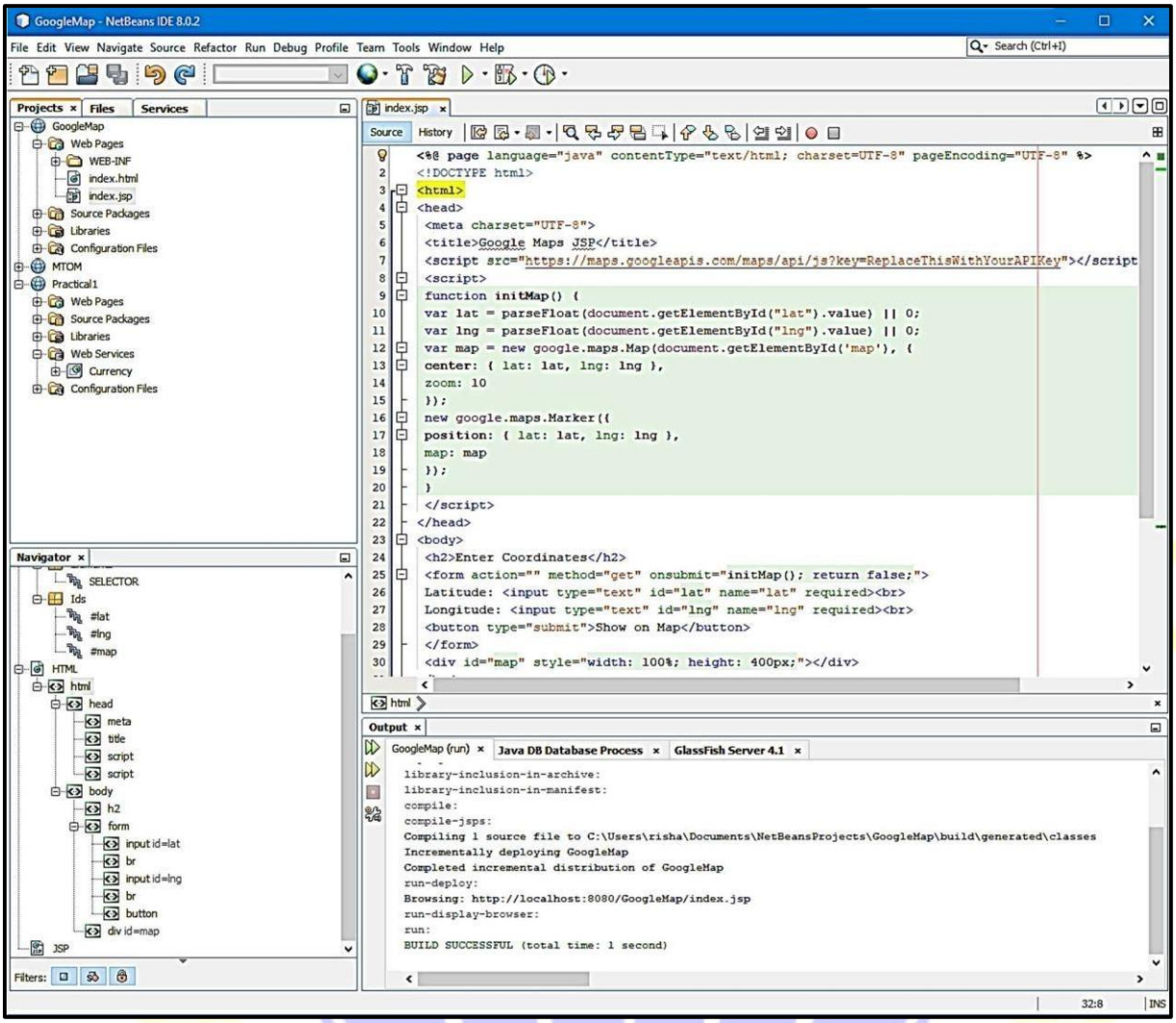
1] Visit console.cloud.google.com, go to APIs and services and select Enabled APIs and services.



2] Enable the following API :



3] Now create a Web Application in NetBeans and type the following in index.jsp file.

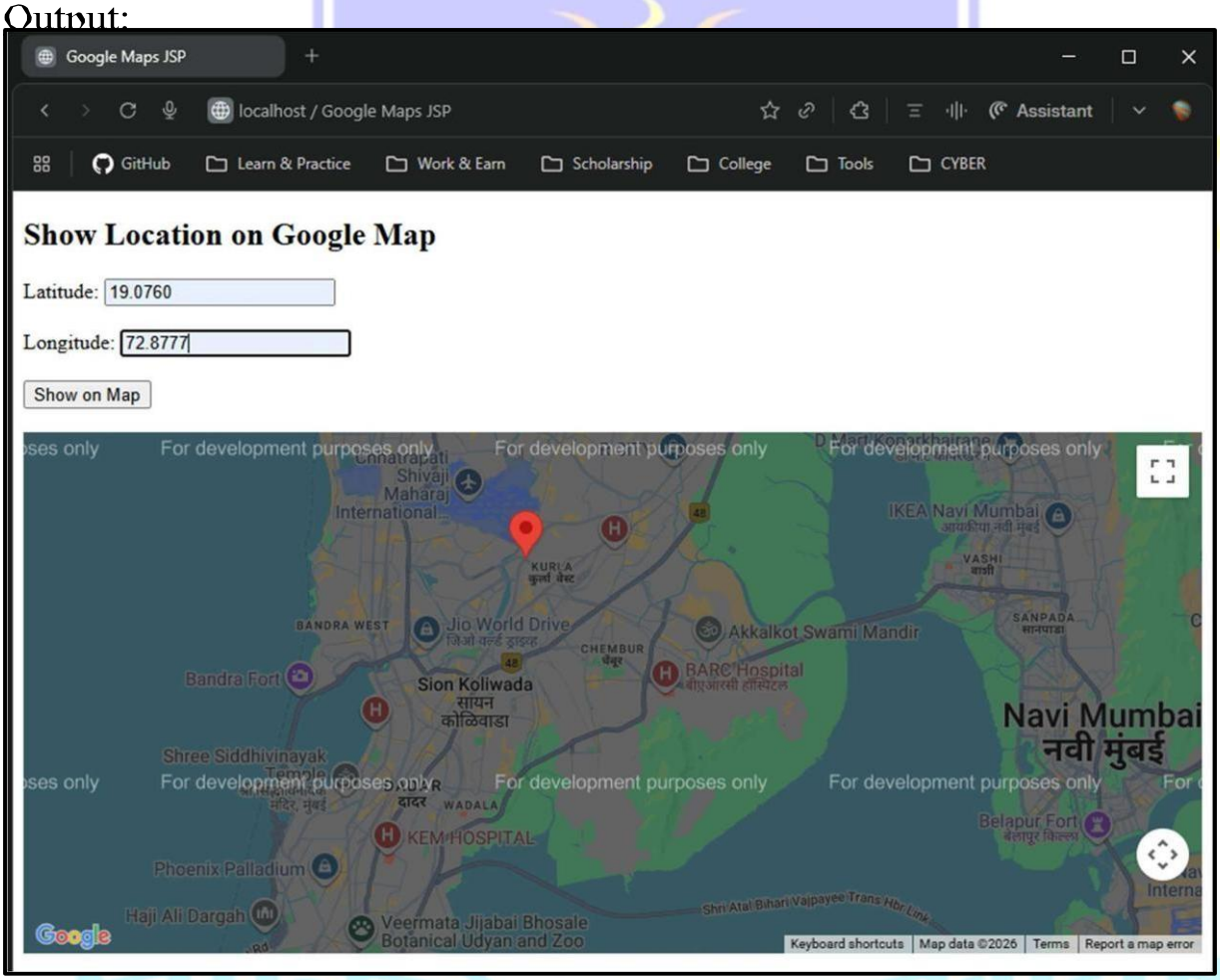



```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Google Maps JSP</title>
<script src="https://maps.googleapis.com/maps/api/js?key=ReplaceThisWithYourAPIKey"></script>
<script> function
initMap() {
 var lat = parseFloat(document.getElementById("lat").value) ||
0;  var lng =
parseFloat(document.getElementById("lng").value) || 0;
 var map = new google.maps.Map(document.getElementById('map'), {  center:
{ lat: lat, lng: lng },
 zoom: 10
 });
 new google.maps.Marker({
position: { lat: lat, lng: lng },  map:
map
 });
 }
</script>
```

```
</head>
<body>
<h2>Enter Coordinates</h2>
<form action="" method="get" onsubmit="initMap(); return false;">
Latitude: <input type="text" id="lat" name="lat" required><br>
Longitude: <input type="text" id="lng" name="lng" required><br>
<button type="submit">Show on Map</button>
</form>
<div id="map" style="width: 100%; height: 400px;"></div>
</body>
</html>
```

4] Visit the index.jsp file on browser and enter some random co-ordinates to test the API

Output:



Learning Outcomes:

1. Ability to integrate external APIs.
2. Understanding of cloud service consumption.

Conclusion:

Successfully integrated Google Maps API in web application.

Course Outcomes:

1. Understand how to integrate third-party cloud APIs into applications.
2. Develop skills to work with real-time cloud services.
3. Apply API-based communication in web development.
4. Enhance knowledge of cloud service consumption.

Viva Questions:

1. What is an API?
2. What is REST API?
3. Why do we need an API key?
4. What is the difference between SOAP and REST API?
5. What is Google Cloud Console used for?

For Faculty use:

Correction Parameters	Formative Assessment[40%]	Timely Completion of Practical[40%]	Attendance Learning Attitude[20%]

Practical 5

Aim: Installation and Configuration of virtualization using KVM.

Tools & Technologies Used:

- Ubuntu Operating System
- VMware Workstation
- KVM (Kernel-based Virtual Machine)
- Virtual Machine Manager
- Windows OS (Host System)

Learning Objectives:

1. To understand the concept of virtualization.
2. To learn installation of Ubuntu OS.
3. To configure a virtual machine environment.
4. To allocate system resources (RAM, CPU, Disk) for VM.
5. To understand cloud infrastructure basics.

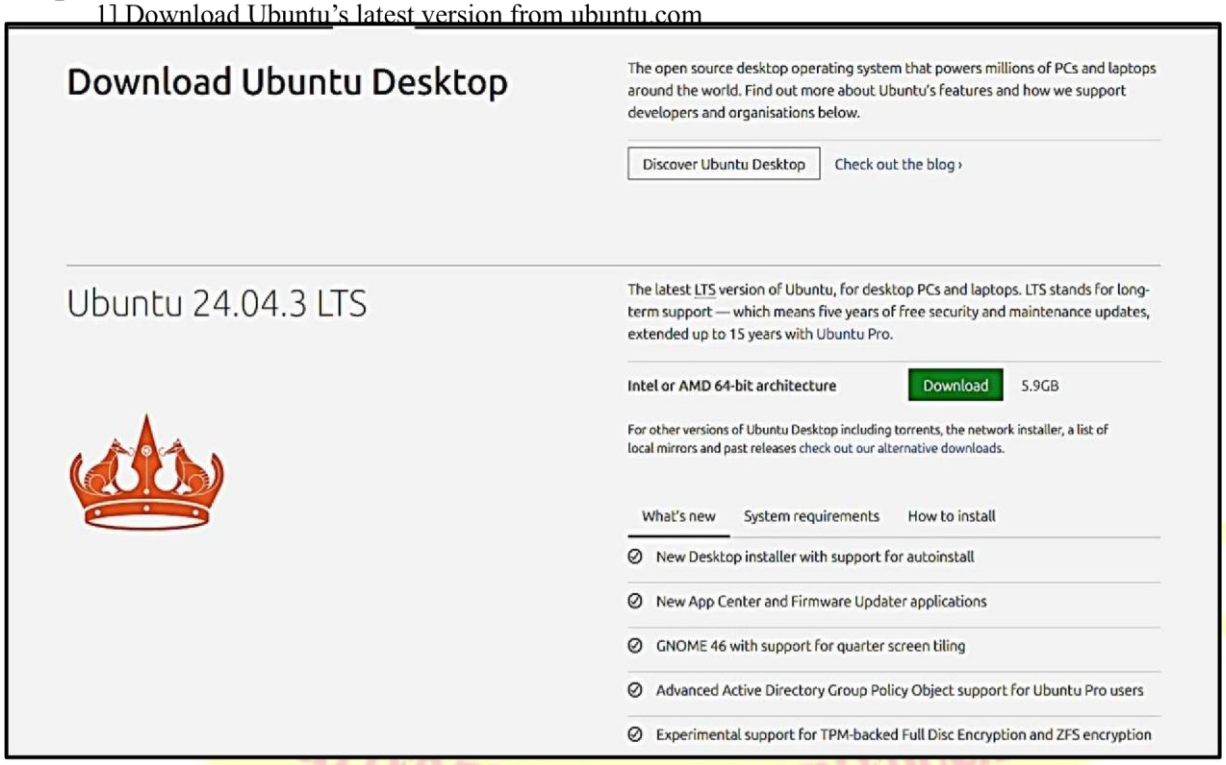
Theory:

Virtualization is a technology that allows multiple operating systems to run on a single physical machine.
KVM (Kernel-based Virtual Machine) is a virtualization solution for Linux systems.
Advantages:

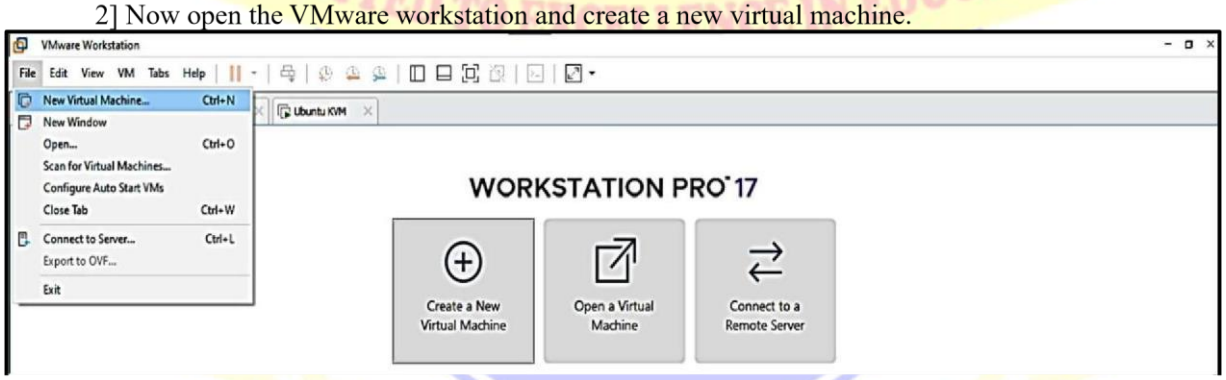
- Better resource utilization
- Cost reduction
- Isolation between systems In this practical:
- Ubuntu OS is downloaded.
- Virtual machine is created.
- RAM, disk, processor allocated.
- OS installed and configured.

Steps:

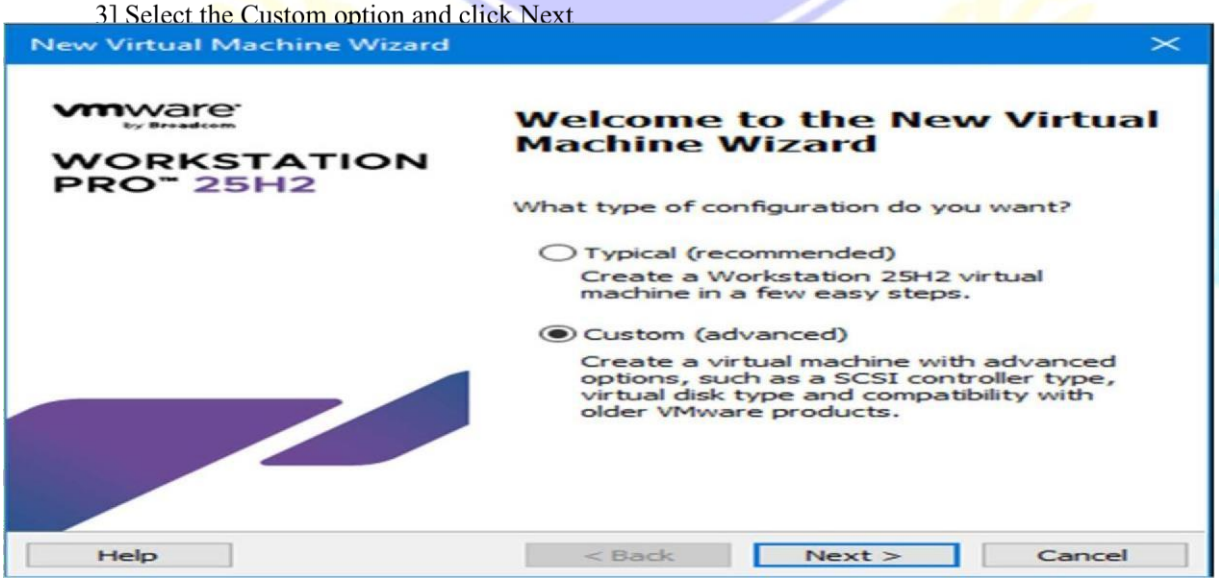
1] Download Ubuntu's latest version from ubuntu.com



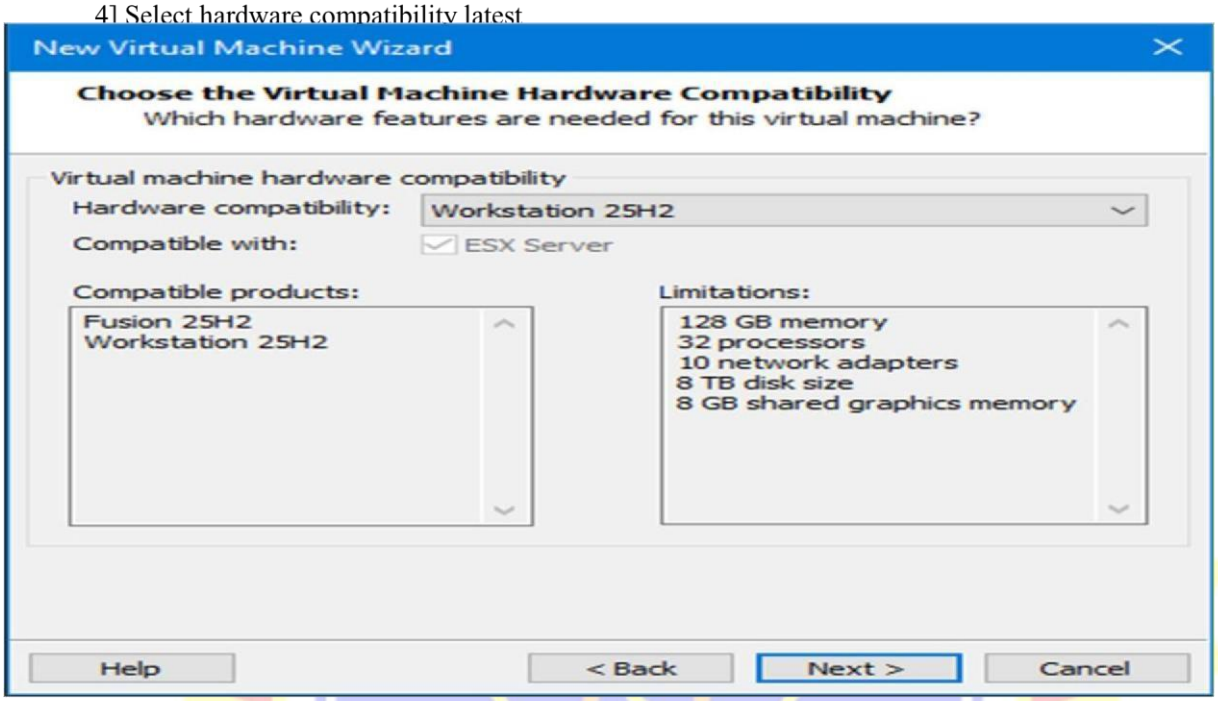
2] Now open the VMware workstation and create a new virtual machine.



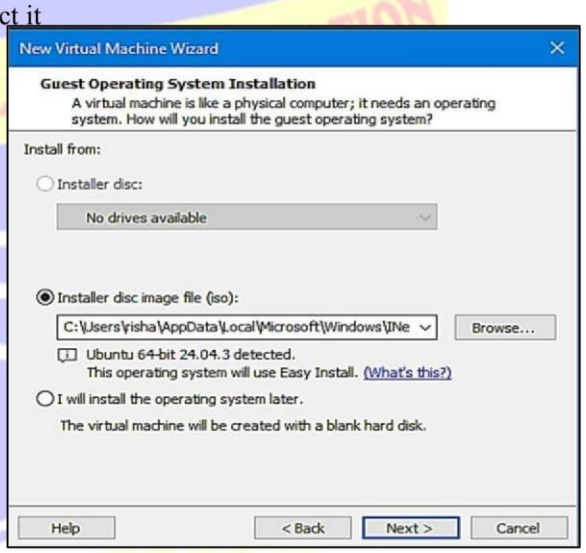
3] Select the Custom option and click Next



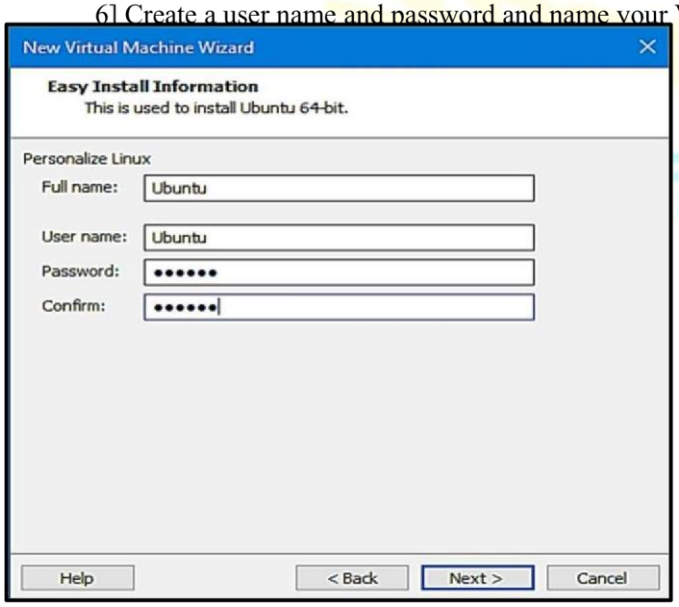
4] Select hardware compatibility latest

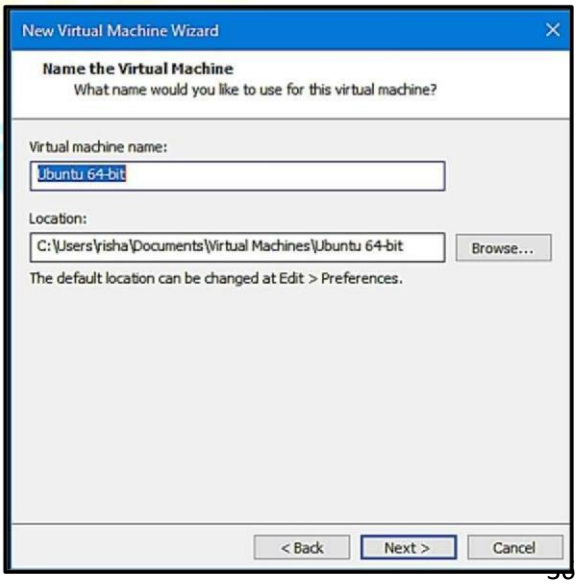


5] Browse the file in Ubuntu's file manager and select it

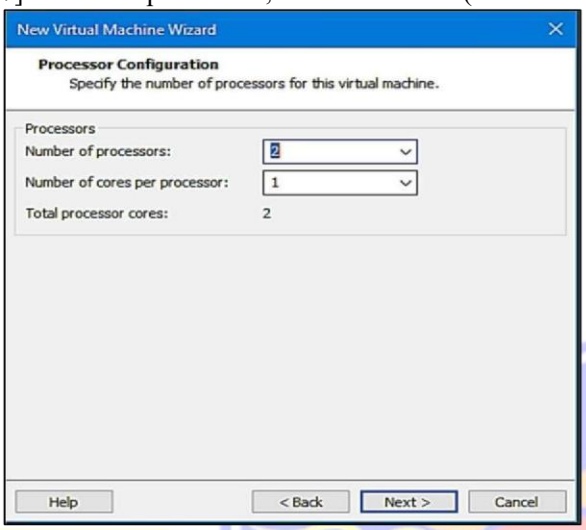


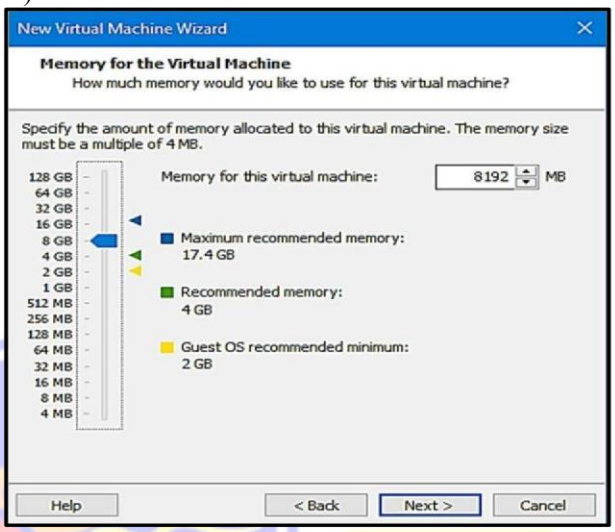
6] Create a user name and password and name your Virtual Machine.



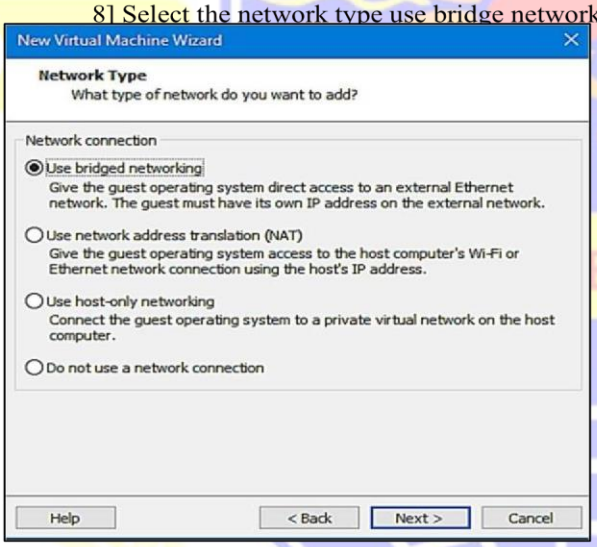


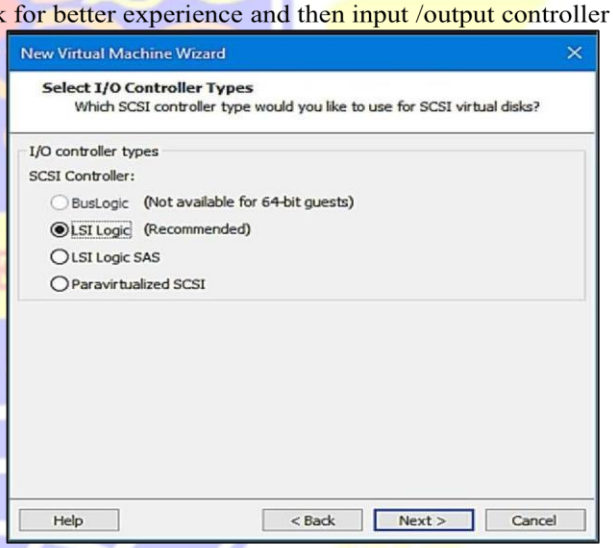
7] Select the processors, cores and RAM (atleast 2GBs)



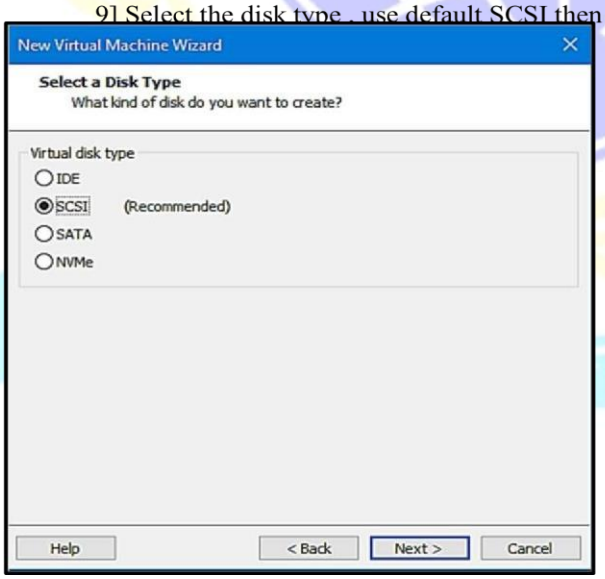


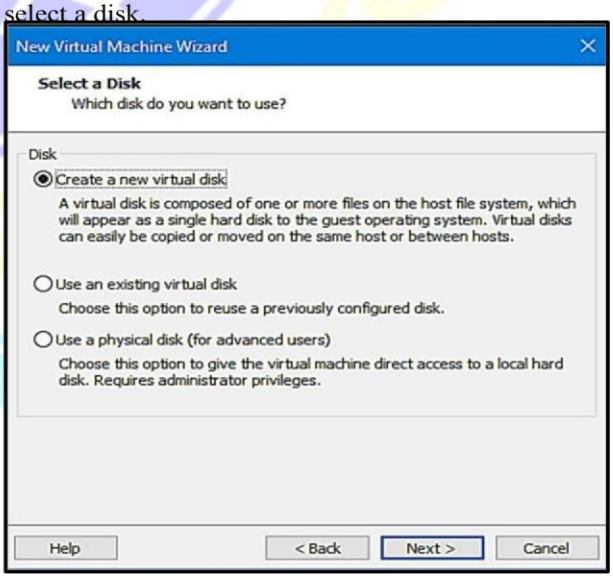
8] Select the network type use bridge network for better experience and then input /output controller



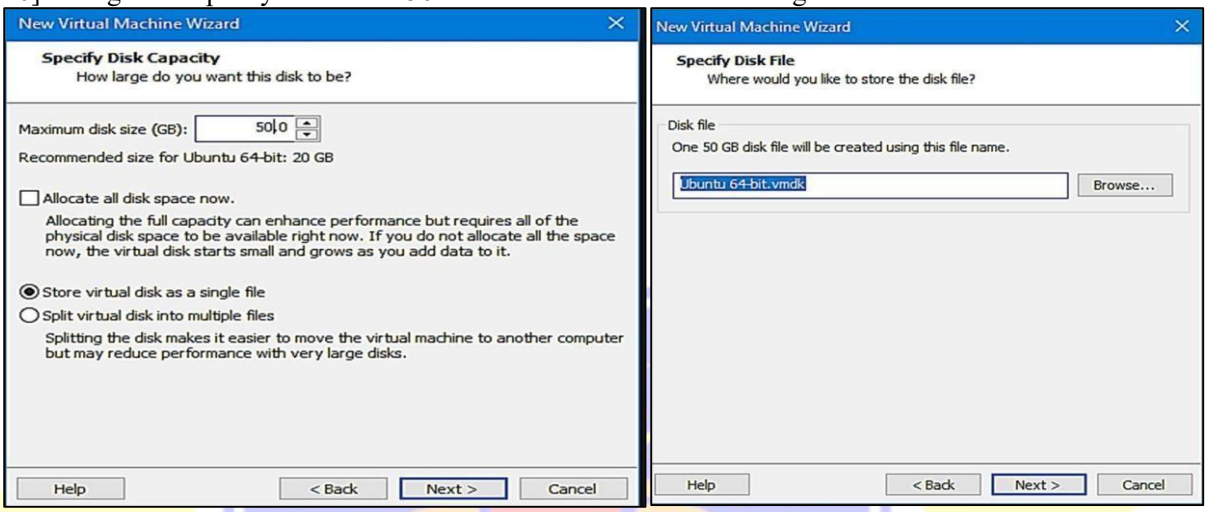


9] Select the disk type , use default SCSI then select a disk.

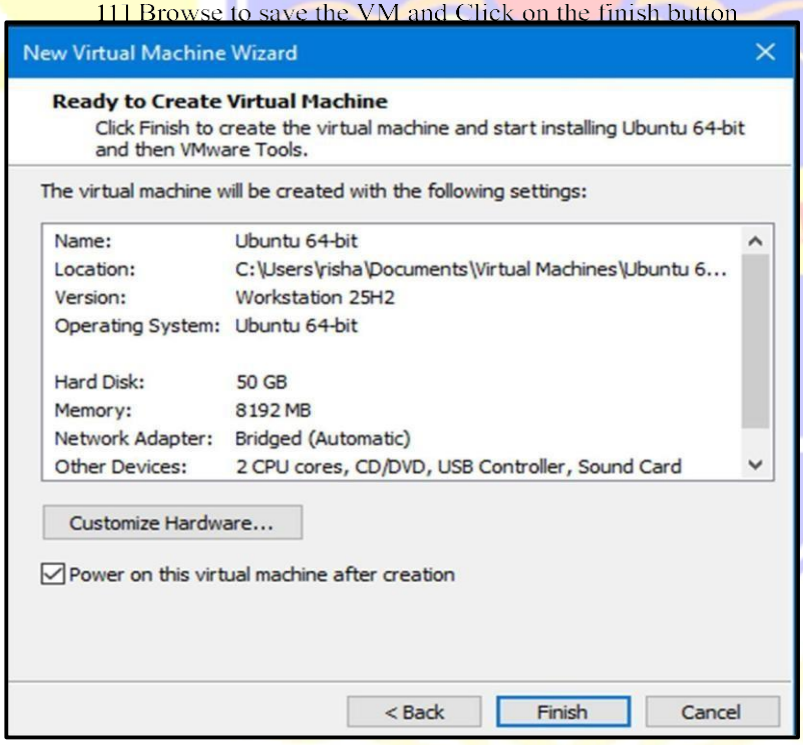




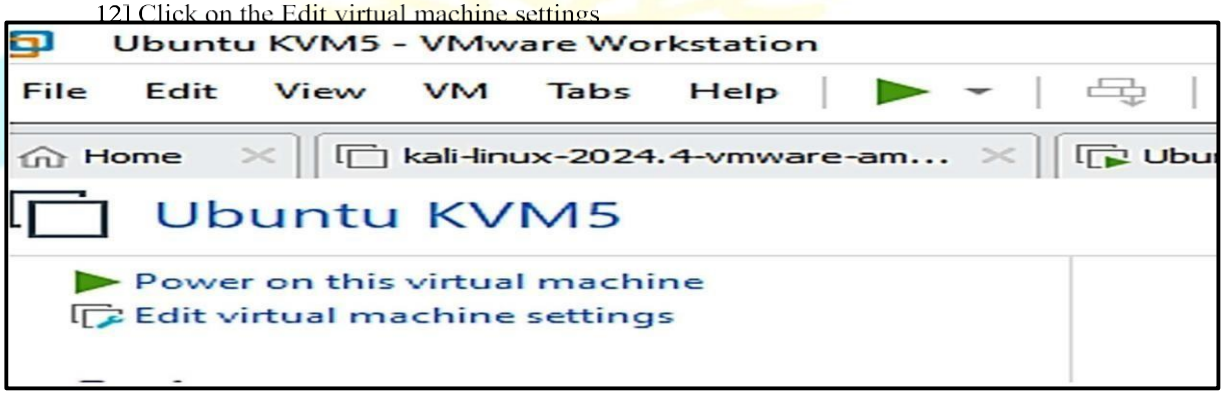
10] Change the capacity of the disk 50GB or Store virtual disk as a single file



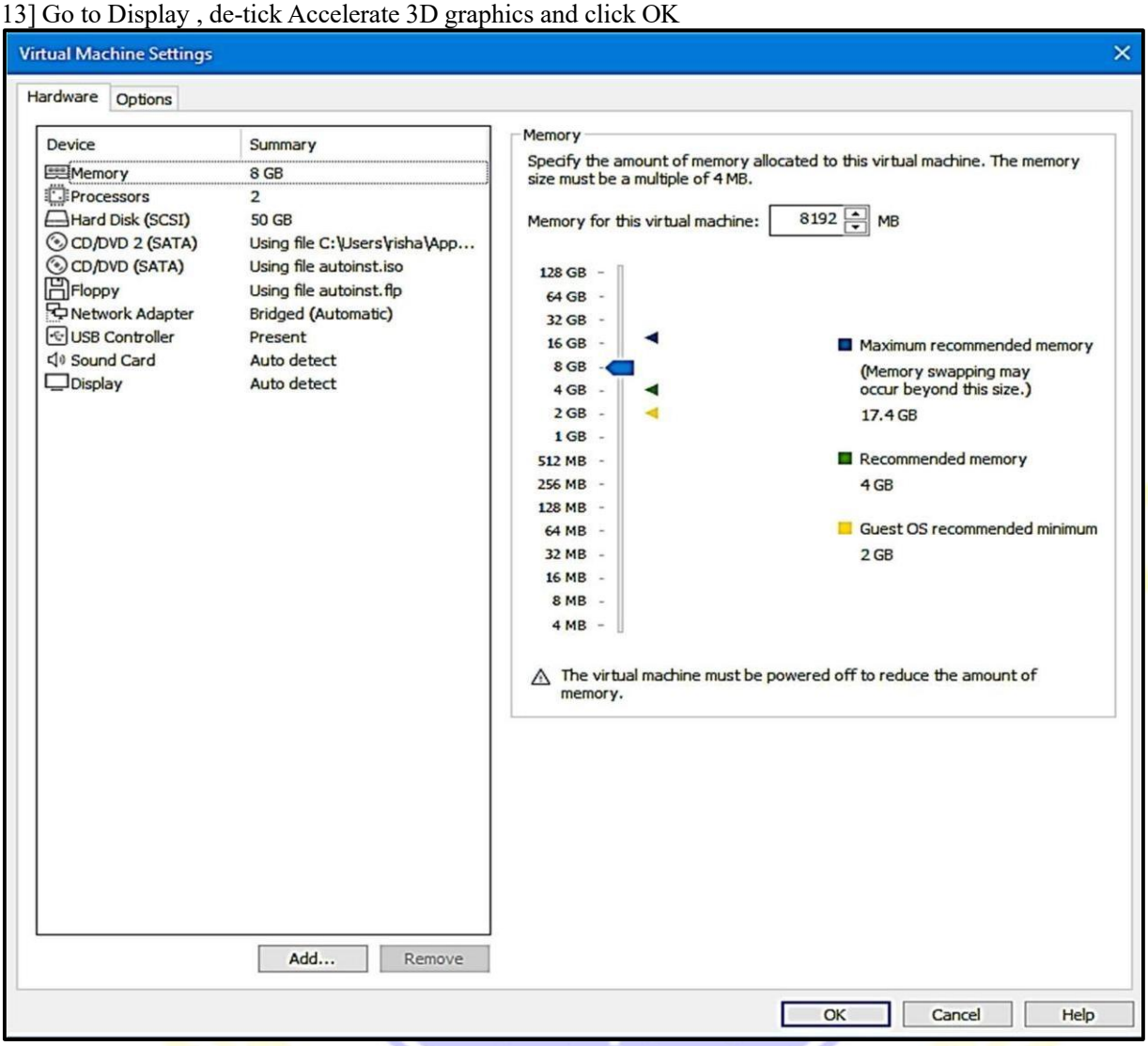
11] Browse to save the VM and Click on the finish button



12] Click on the Edit virtual machine settings

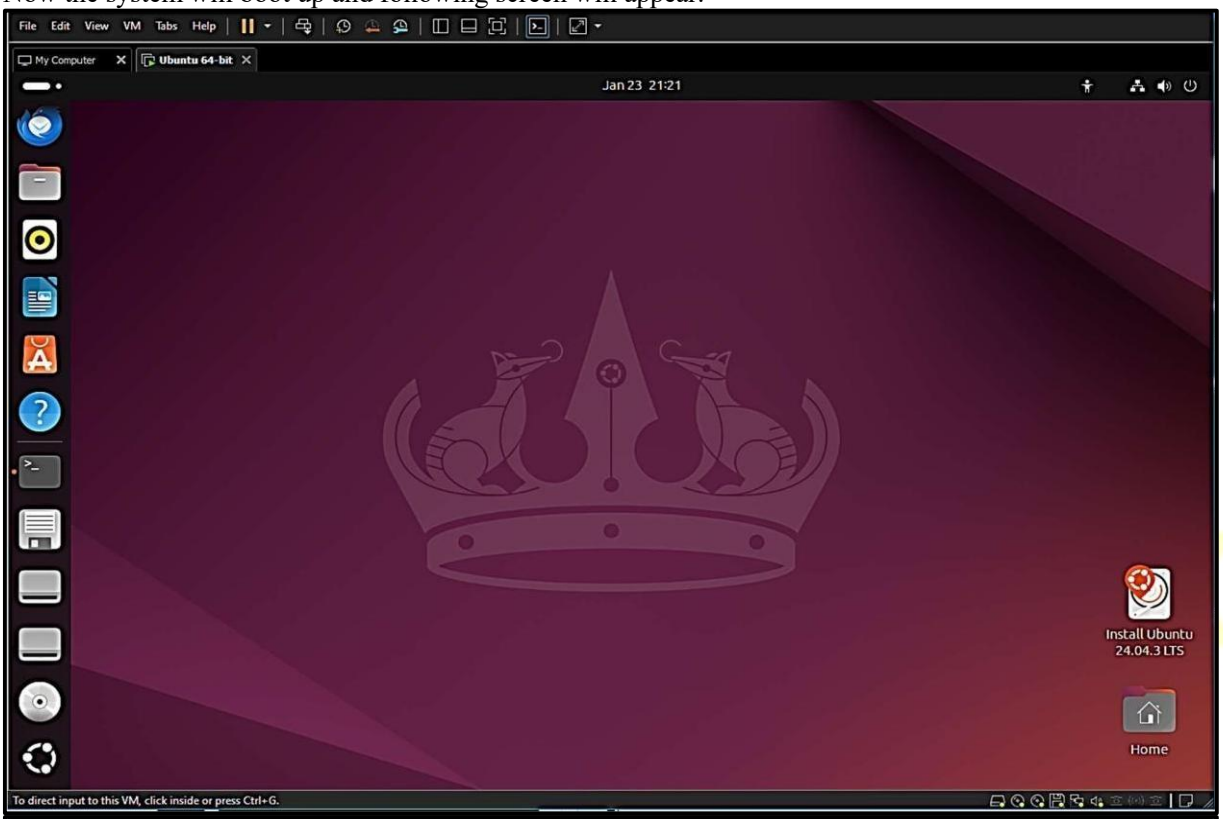


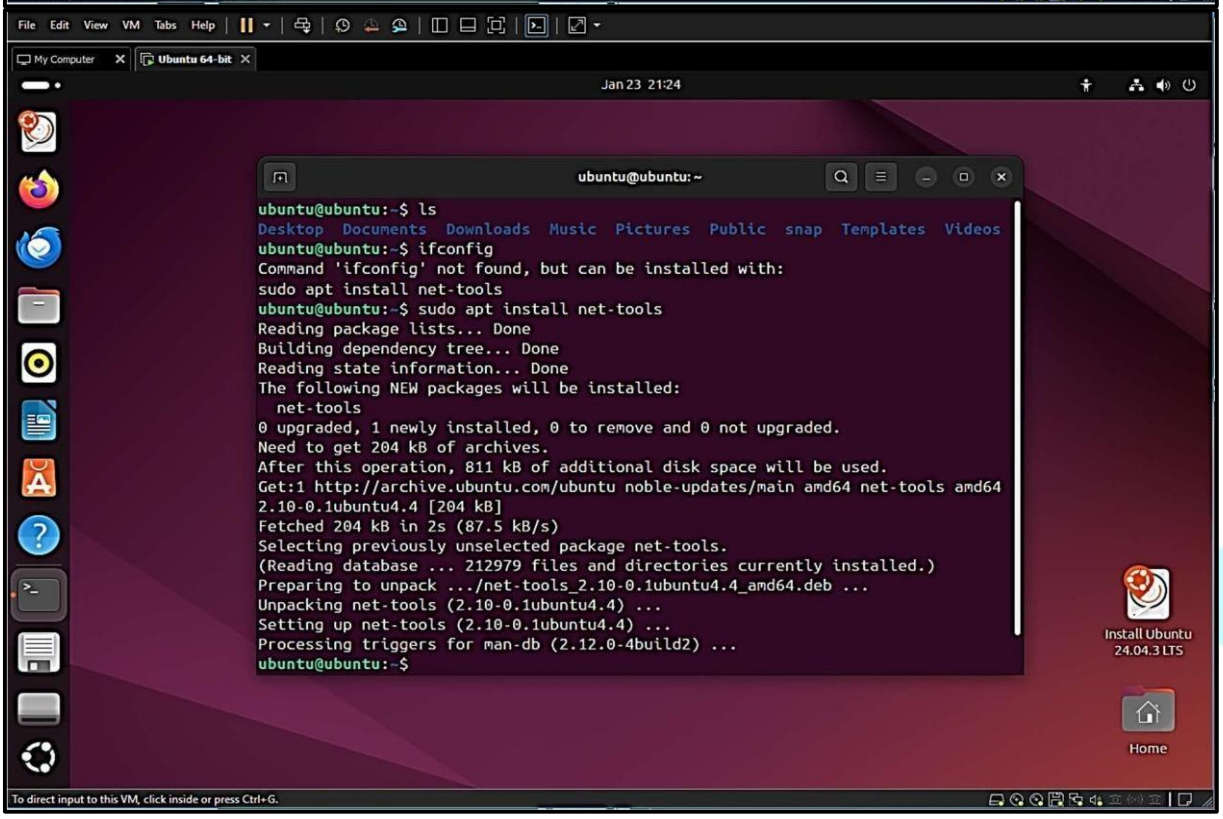
13] Go to Display , de-tick Accelerate 3D graphics and click OK



14] Select Try or Install Ubuntu

Now the system will boot up and following screen will appear:





Learning Outcomes:

1. Able to understand the concept of virtualization and its importance in cloud computing.
2. Able to install and configure Ubuntu operating system in a virtual machine.
3. Capable of allocating and managing system resources like RAM, CPU, and storage in a virtual environment.
4. Understand the working of hypervisors and virtual machine architecture.

Course Outcomes:

1. Understand virtualization in cloud computing.
2. Learn infrastructure setup and configuration.
3. Apply virtual machine technology for system management.
4. Understand role of hypervisors in cloud.

Conclusion:

Successfully installed and configured virtual machine using Ubuntu.

Viva Questions:

1. What is virtualization?
2. What is KVM?
3. What is hypervisor?
4. What is the difference between Type 1 and Type 2 hypervisor?
5. Why is virtualization important in cloud computing?

For Faculty use:

Correction Parameters	Formative Assessment[40%]	Timely Completion of Practical[40%]	Attendance Learning Attitude[20%]

Practical 6

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

Learning Objectives:

1. To understand MTOM concept.
2. To transfer binary data using SOAP.
3. To upload and download files through web service.

Tools & Technologies Used:

- NetBeans IDE
- Java
- SOAP Web Service
- Apache Tomcat Server
- MTOM (Message Transmission Optimization Mechanism)
- JSP (Java Server Pages)

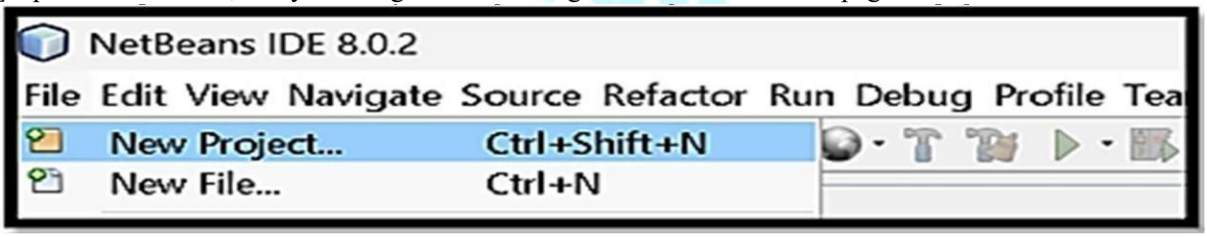
Theory:

MTOM (Message Transmission Optimization Mechanism) is used to efficiently send binary data in SOAP messages.
Normally SOAP encodes binary data in base64 which increases size. MTOM optimizes this process.
In this practical:

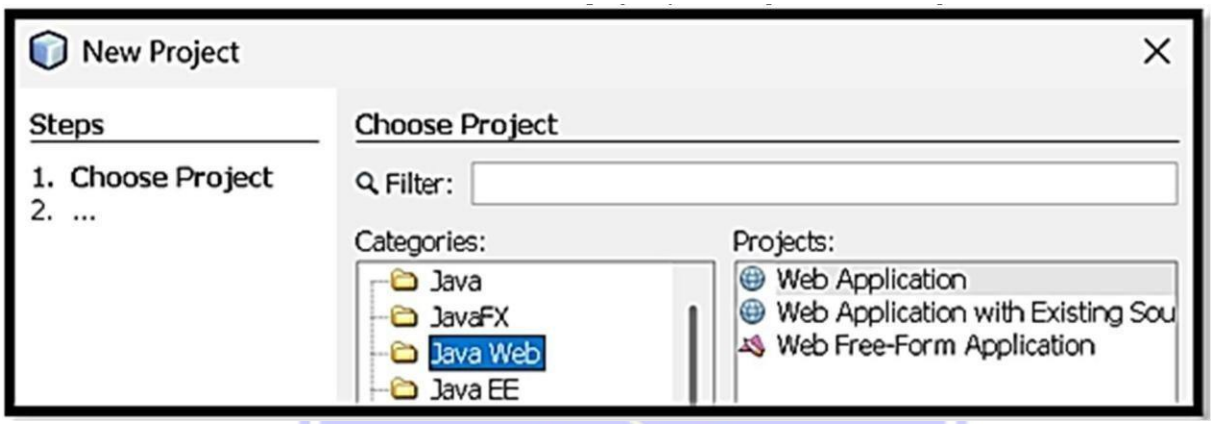
- Web service is created.
- MTOM enabled.
- Image uploaded to server.
- Image downloaded from server.

Steps:

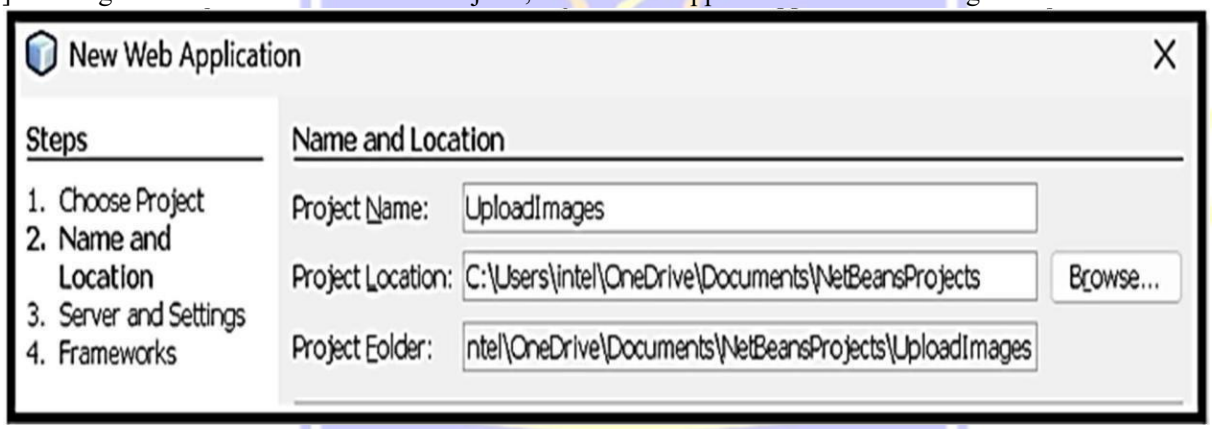
1] Open the NetBeans, and you will get the following screen. Close the start page.



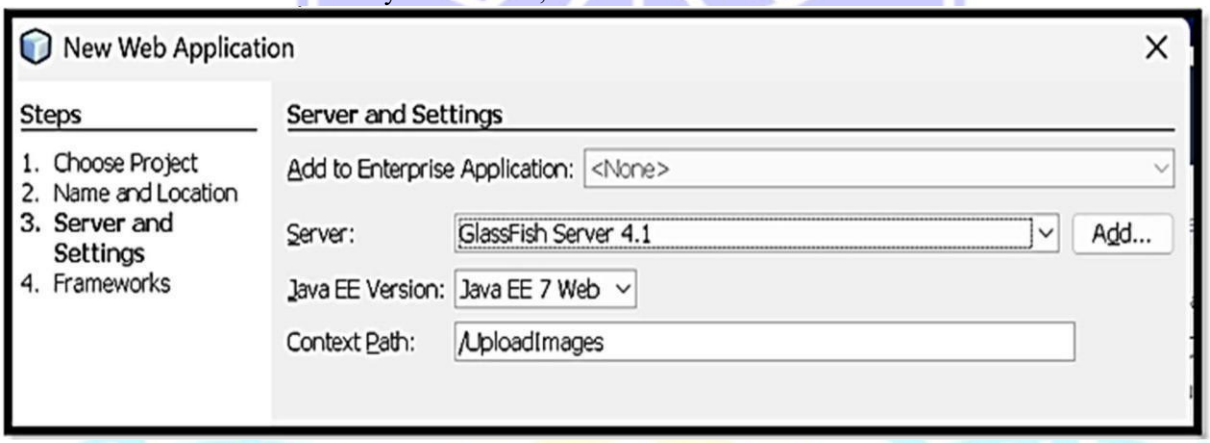
2] Now click on the file tab and click on new project you will get the following screen:



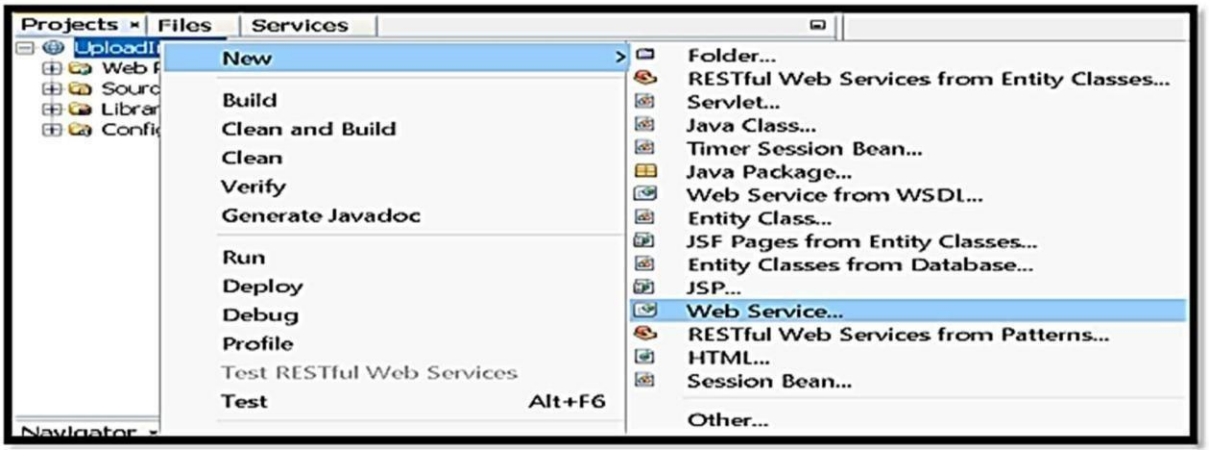
3] In Categories select Java Web and in Projects, Select Web Application. After selecting click on next.



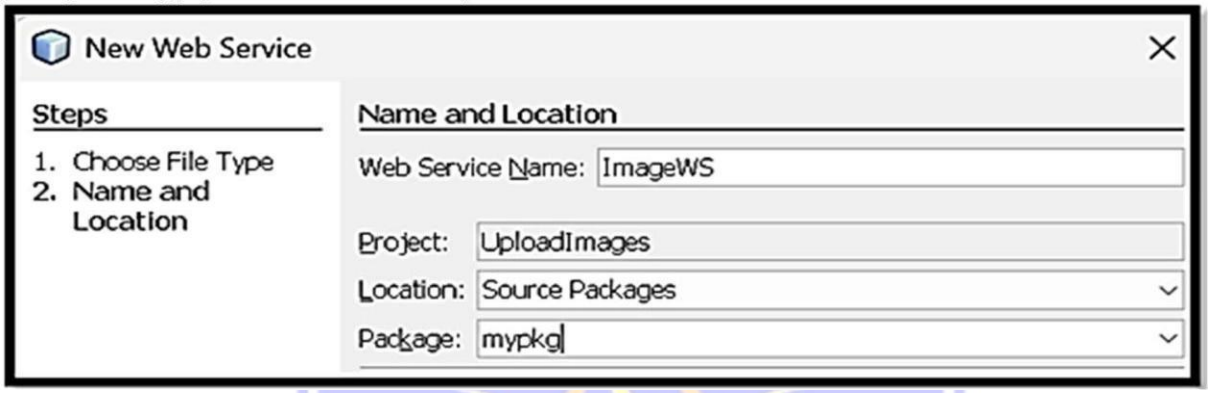
5] Now Give name to the Project Name: and click on next. you will get the following window then. again, click on finish. Don't click on any frameworks, Finish.



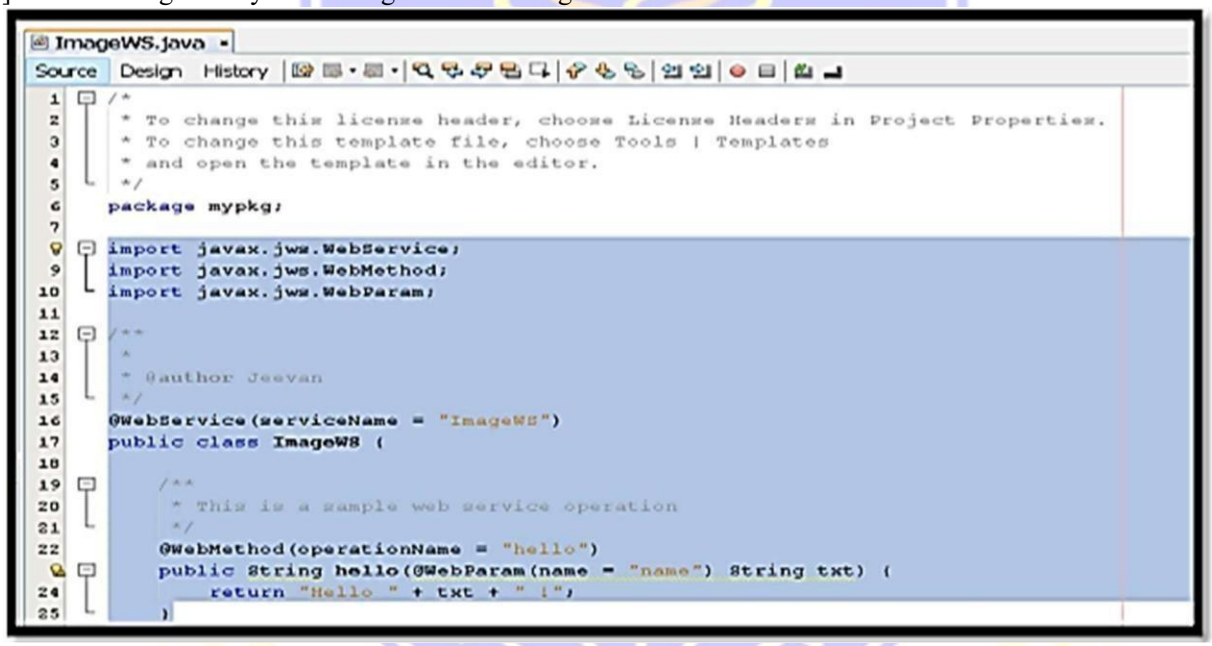
6] Now right click on the project and select new and then select Web Service, as shown Below:



7] After clicking Web Service following window should appear, now give name to web service and give package as “mypkg”. As shown in the image:



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



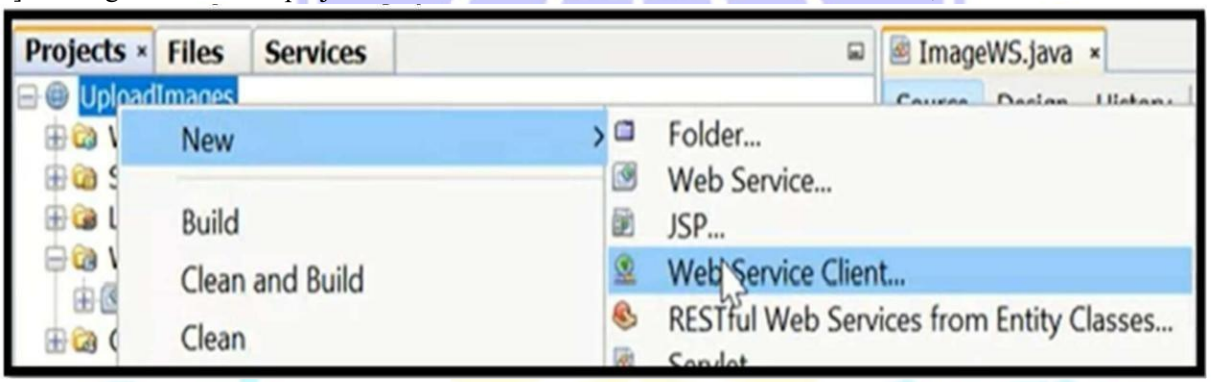
9] Now in ImageWS.java type this code: import java.io.*; import javax.jws.Oneway; import javax.jws.WebService; import javax.jws.WebMethod; import javax.jws.WebParam; import javax.xml.ws.soap.MTOM;

```
@MTOM(enabled = true, threshold = 6000)  @WebService(serviceName
= "ImageWS")
public class ImageWS {

    @WebMethod(operationName = "upload")
    @Oneway
    public void upload(@WebParam(name = "Filename") String Filename, @WebParam(name = "ImageBytes")
byte[] ImageBytes) {
        String filePath = "C:/Picture/upload/" +
Filename;         try {
FileOutputStreamfos = new FileOutputStream(filePath); BufferedOutputStreambos
= new BufferedOutputStream(fos); bos.write(ImageBytes);
bos.close();
System.out.println("Received file: " + filePath);
        } catch (Exception ex) {  ex.printStackTrace();
        }
    }
```

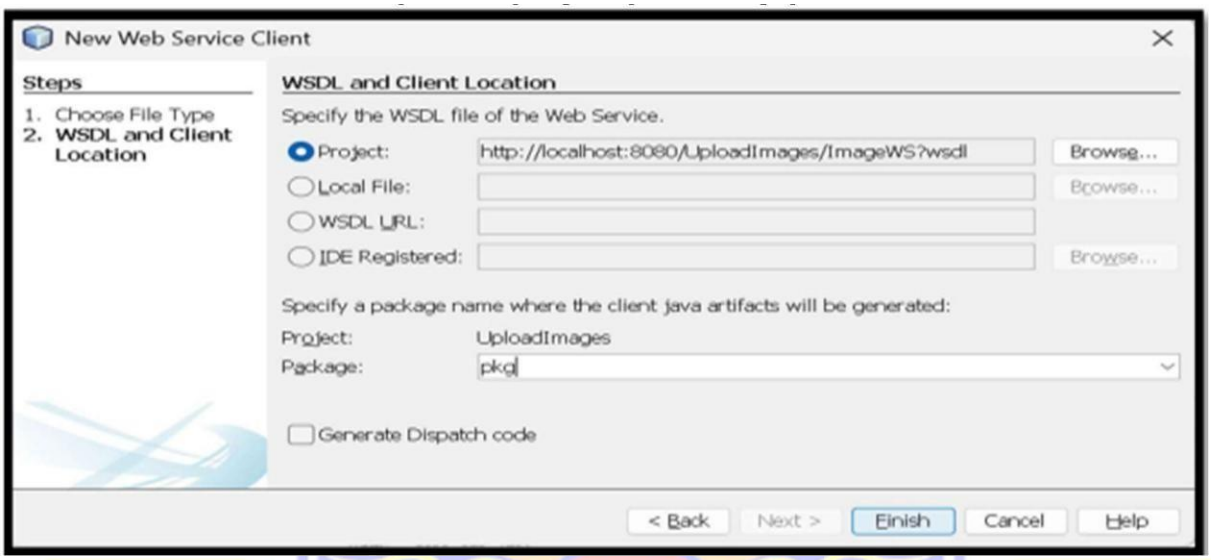
```
    @WebMethod(operationName = "download")
    public byte[] download(@WebParam(name = "Filename") String Filename) {
        String filePath = "C:/Picture/upload/" + Filename;
System.out.println("Sending file: " + filePath);          try
{
            File file = new File(filePath);
FileInputStreamfis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

10] Now run file ImageWS.java.

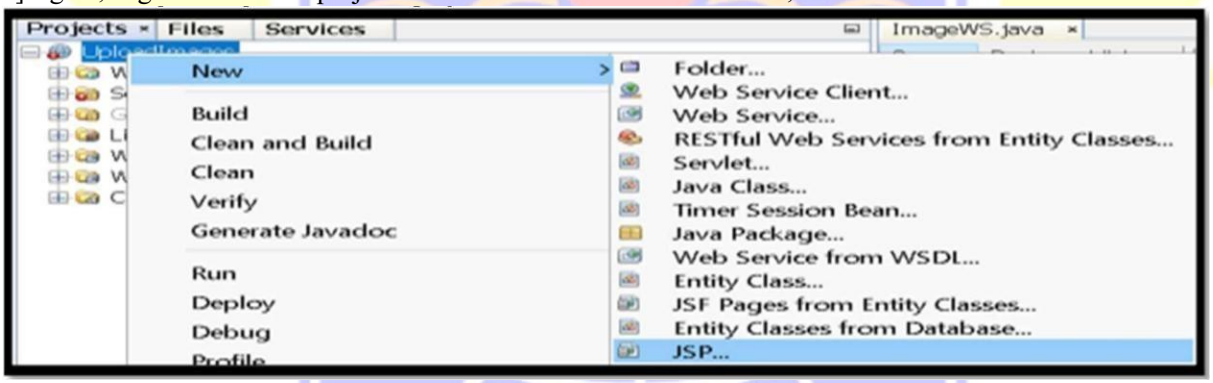
11] Now right click on the project and select new and then select Web Service Client, as shown Below:



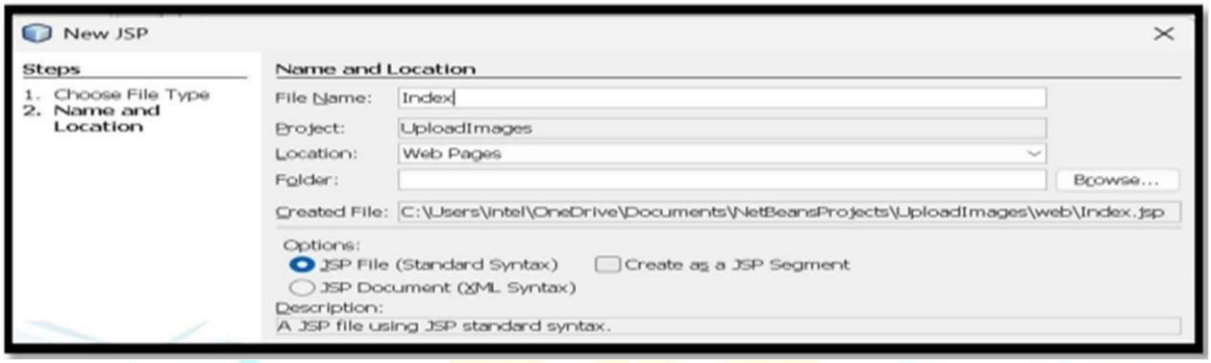
12] Then Browse select ImageWs and give package name is “pkg”.



13] Again, Right click on the project and select new and then select JSP, as shown Below:



14] Then file name is Index, as shown below:



15] Delete index.html file

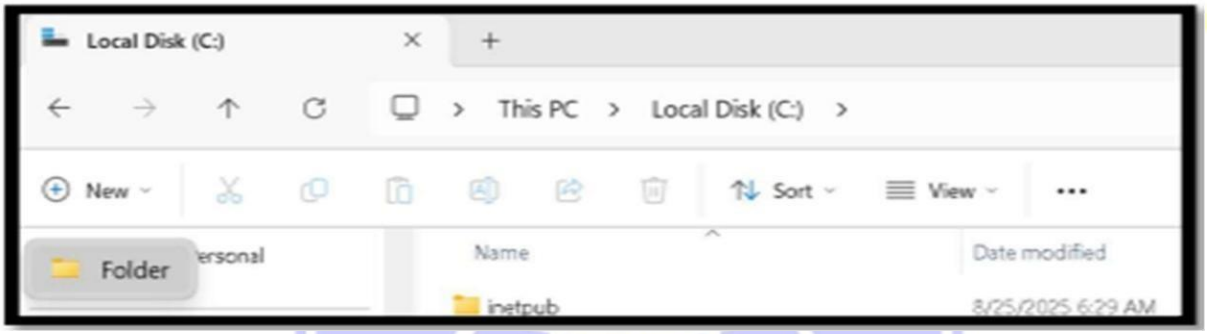
16] Now Erase all code index.jsp and type this code:

```
<%@page import="java.io.BufferedOutputStream"%>
<%@page import="java.io.FileOutputStream"%>
<%@page import="java.io.FileInputStream"%>
<%@page import="java.io.BufferedInputStream"%>
```

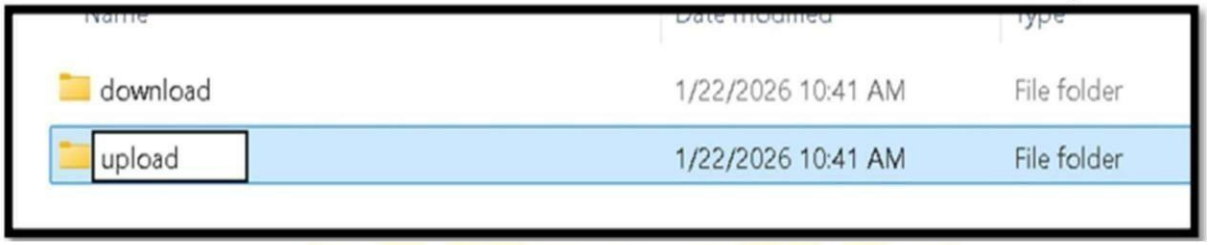
```
<%@page import="java.io.File"%>
<%@page import="javax.xml.ws.soap.MTOMFeature"%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<title>JSP Page</title>
</head>
<body>
<%-- Start Web Service Innovation (Upload) --%>
<hr/>
<%
        try {
            // Initialize Service and Port with MTOM enabled pkg.ImageWS_Service
service = new pkg.ImageWS_Service();
pkg.ImageWS port = service.getImageWSPort(new MTOMFeature(6000));

            String filePath = "C:/Picture/car.jpg";
            File file = new File(filePath);

            // Read the file into a byte array
FileInputStreamfis = new FileInputStream(file);
BufferedInputStream bis = new BufferedInputStream(fis);
String filename = file.getName();
byte[] imageBytes = new byte[(int) file.length()];

bis.read(imageBytes);  bis.close();

            // Call Web Service Upload
port.upload(filename, imageBytes);

out.println("File uploaded: " + filePath);

        } catch(Exception ex) {
ex.printStackTrace();
        }
    %>

<%-- End Web Service Innovation --%>
<hr/>
<%-- Start Web Service Innovation (Download) --%>
<hr/>

<%
        try {
            // Initialize Service and Port
pkg.ImageWS_Service service = new pkg.ImageWS_Service();
pkg.ImageWS port = service.getImageWSPort();

            String filename = "car.jpg";
            String filePath = "C:/Picture/download/" + filename;

            // Call Web Service Download
```

```
byte[] fileBytes = port.download(filename);

            // Write the downloaded bytes to a file
FileOutputStreamfos = new FileOutputStream(filePath);
BufferedOutputStreambos = new BufferedOutputStream(fos);

bos.write(fileBytes);
bos.close();
out.println("File download: " + filePath);

        } catch(Exception ex) {
ex.printStackTrace();
        }
    %>
</body>
</html>
```

17] Now make new folder in Local Disk, and rename it Picture, as shown below:

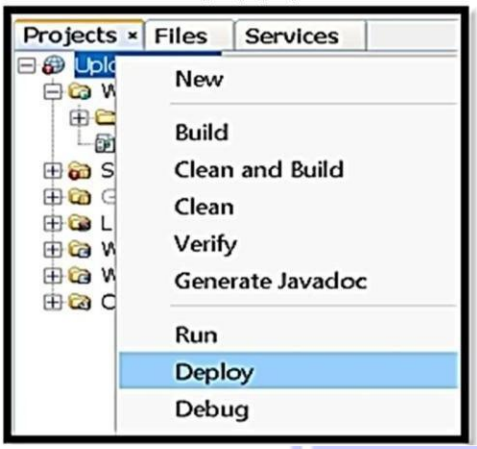


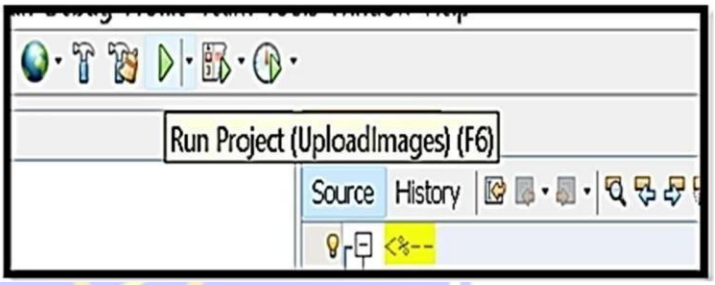
18] Now make new two folder inside in Picture folder name and rename it download and upload:



19] In picture folder paste one jpg image, as shown below:

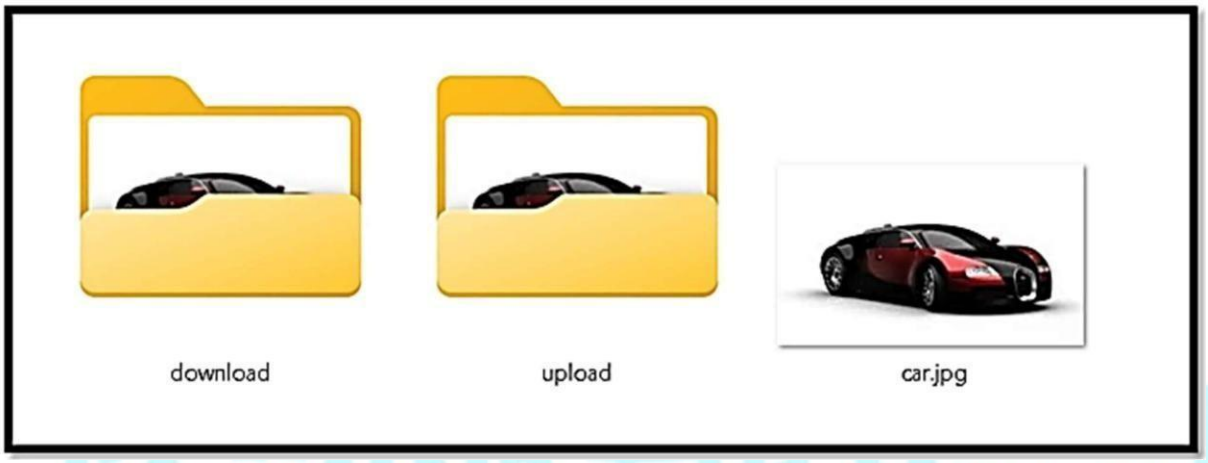
20 Now deploy project and run index.jsp file:


Projects
Files
Services
New
Build
Clean and Build
Clean
Verify
Generate Javadoc
Run
Deploy
Debug


Run Project (UploadImages) (F6)
Source
History

Output:


TODO supply a title
JSP Page
localhost:8080/UploadImages/
<
File uploaded: C:/Picture/car.jpg
File download: C:/Picture/download/car.jpg


download
upload
car.jpg

Learning Outcomes:

1. Able to understand the concept of MTOM (Message Transmission Optimization Mechanism).
2. Able to transmit binary data (images/files) using SOAP web services.
3. Capable of implementing file upload and download functionality in a distributed system.
4. Understand how MTOM improves performance compared to normal SOAP encoding.
5. Able to establish client-server communication for file transfer.

Course Outcomes:

1. Understand binary data transmission in web services.
2. Apply MTOM for optimized SOAP communication.
3. Develop file transfer applications using web services.
4. Understand message optimization in distributed systems.

Conclusion:

Successfully implemented MTOM-based file transfer system.

Viva Questions:

1. What is MTOM?
2. Why is MTOM used in SOAP services?
3. What happens if MTOM is not enabled?
4. What is binary data?
5. What is the difference between normal SOAP and MTOM SOAP?

For Faculty use:

Correction Parameters	Formative Assessment[40%]	Timely Completion of Practical[40%]	Attendance Learning Attitude[20%]

Practical 7

Aim: 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.

Tools & Technologies Used:

- Visual Studio Code (VS Code)
- Java Development Kit (JDK) • AWS Flow Framework
- Command Prompt / Terminal
- Java Compiler (javac)

Learning Objectives:

1. To understand workflow concept.
2. To define activity interface.
3. To implement workflow logic.
4. To execute workflow using worker.

Theory:

AWS Flow Framework helps developers build distributed and asynchronous applications.
Components:

- Workflow – Defines coordination logic
- Activity – Performs task
- Worker – Executes workflow

In this practical:

- Activity prints “Hello World”
- Workflow calls activity
- Worker runs program

This demonstrates distributed cloud application design.

AWS Flow Framework is a programming framework provided by Amazon Web Services that helps developers build distributed and asynchronous applications using workflows and activities. A workflow defines the coordination logic, while activities perform individual tasks. Workers are responsible for executing workflows and activities. In this practical, a simple workflow is

implemented that invokes an activity to print a message on the console, demonstrating the basic usage of the AWS Flow Framework.

ALGORITHM

1. Define an activity interface.
2. Implement the activity to print “Hello World”.
3. Define a workflow interface.
4. Implement workflow logic to call the activity.
5. Create a worker program to execute the workflow.
6. Execute the program and verify output.

Steps:

1] Download VS Code and install JDK.

2] Create a Folder AWS with 5 java files (HelloActivity.java, HelloAtivityImpl.java, HelloWorkflow.java, HelloWorkflowImpl.java, Worker.java)

HelloActivity.java

```
public interface HelloActivity {
    void printHello();
}
```

HelloActivityImpl.java

```
public class HelloActivityImpl implements
HelloActivity {     public void printHello() {
System.out.println("Hello World from AWS Flow Activity");
    }
}
```

HelloWorkflow.java

```
public interface HelloWorkflow {
    void start();
}
```

HelloWorkflowImpl.java

```
public class HelloWorkflowImpl implements HelloWorkflow {

HelloActivity activity = new HelloActivityImpl();

    public void start() {
activity.printHello();
    }
}
```

Worker.java

```
public class Worker {
    public static void main(String[] args) {
HelloWorkflow workflow = new HelloWorkflowImpl();
workflow.start();
    }
}
```

3] Open Terminal or press Ctrl+ ~ and then enter the following command to compile the java files: javac *.java

java Worker

Output:

Learning Outcomes:

1. Able to understand the concept of workflow-based programming in cloud computing.
2. Able to define and implement activity and workflow interfaces in Java.
3. Capable of coordinating tasks using workflow and activity model.
4. Understand the role of worker programs in executing workflows.
5. Gain knowledge of distributed and asynchronous application development

Course Outcomes:

1. Understand workflow-based cloud application development.
2. Learn distributed application concepts.
3. Apply AWS Flow Framework concepts in Java programs.
4. Understand coordination between workflow and activities.

Conclusion:

Successfully implemented AWS workflow-based application.

Viva Questions:

1. What is AWS Flow Framework?
2. What is workflow in AWS?
3. What is activity in AWS?
4. What is worker program?
5. What is distributed computing?

For Faculty use:

Correction Parameters	Formative Assessment[40%]	Timely Completion of Practical[40%]	Attendance Learning Attitude[20%]

Practical 8:

Aim: Implementation of OpenStack with user and private network creation.

Tools & Technologies Used:

1. OpenStack
2. Horizon Dashboard
3. OpenMetal Cloud Platform
4. SSH (Secure Shell)
5. Linux Terminal
6. Neutron (OpenStack Networking Service)
7. Virtual Machine Instance

Learning Objectives:

1. To understand the architecture of OpenStack cloud platform.
2. To create and manage private networks in OpenStack.
3. To configure subnet and router in cloud environment.
4. To implement Infrastructure as a Service (IaaS) concept.
5. To learn secure access using SSH key generation.

Theory:

OpenStack is an open-source cloud computing platform that provides Infrastructure as a Service (IaaS). It allows users to create and manage virtual machines, storage, and networking resources in a cloud environment.

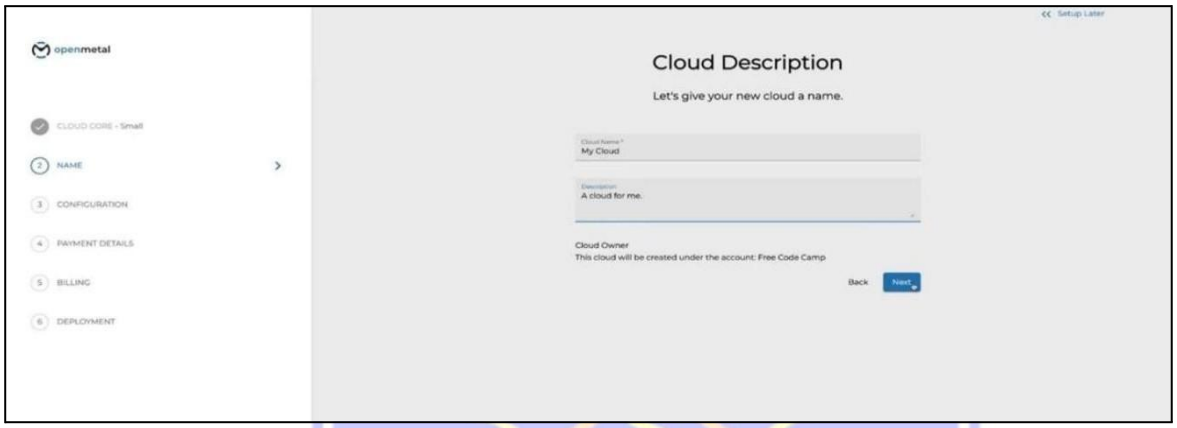
In OpenStack, networking is handled by the Neutron service, which allows users to create private networks, subnets, and routers. Private networks are used to enhance security and restrict unauthorized access.

Horizon is the web-based dashboard used to manage OpenStack resources such as networks, instances, users, and routers.

Procedure:

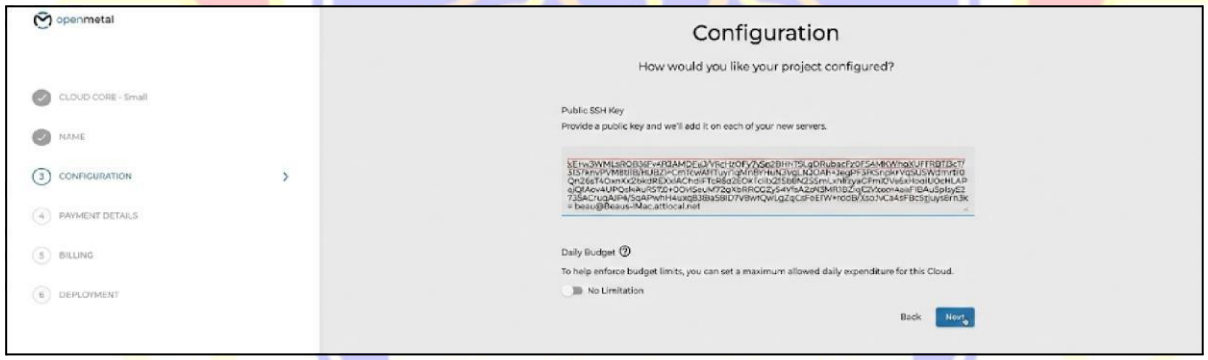
Note- Private networks should be used where possible. Only expose the portions of your cloud to a public network when needed.

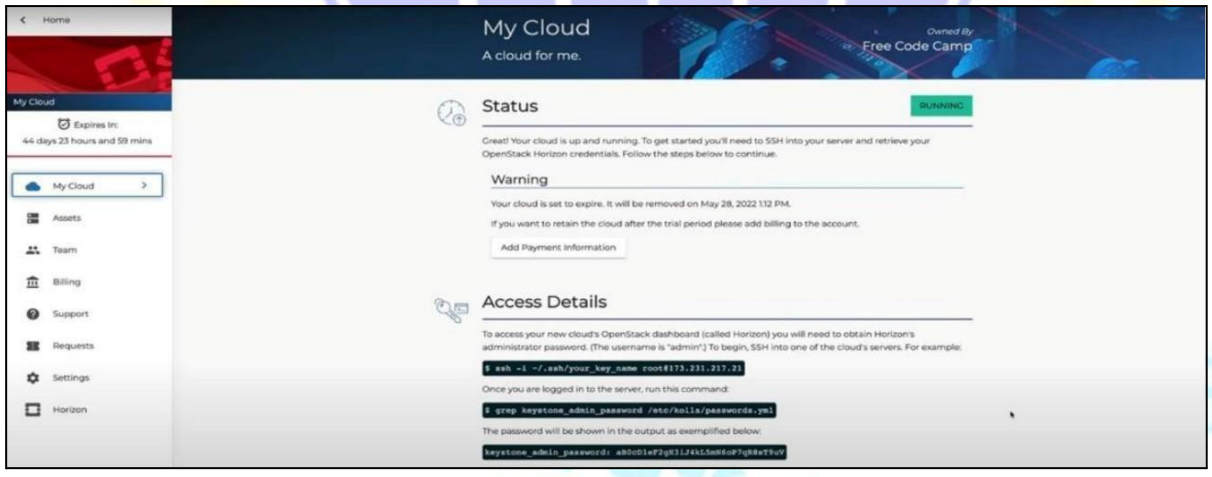
First you can create account, openmetal.io



Follow this command in terminal :

1. ssh-keygen
2. ssh ls
3. ssh vim id_rsa.public

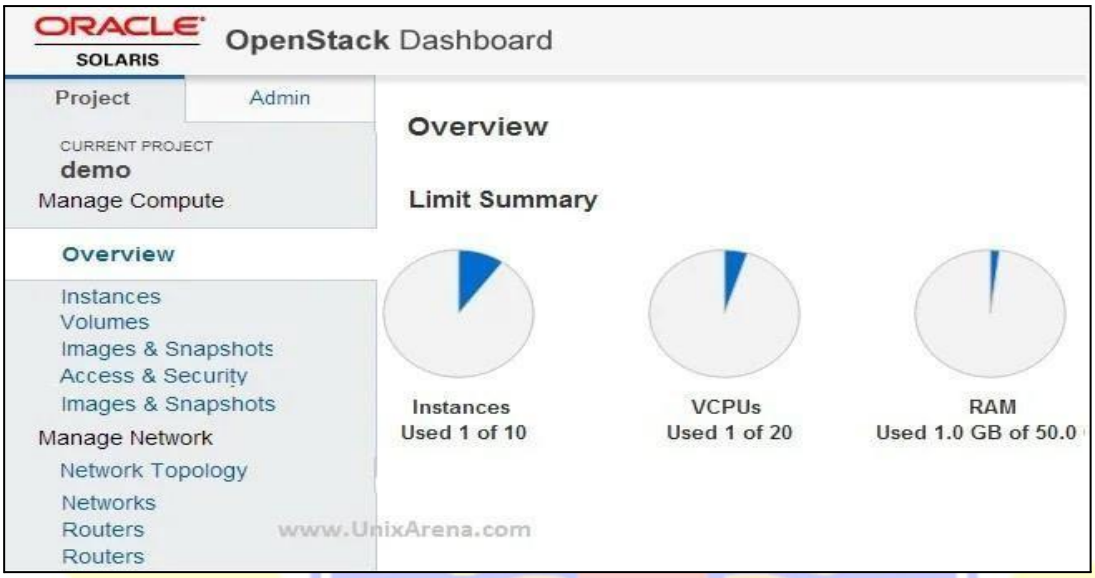




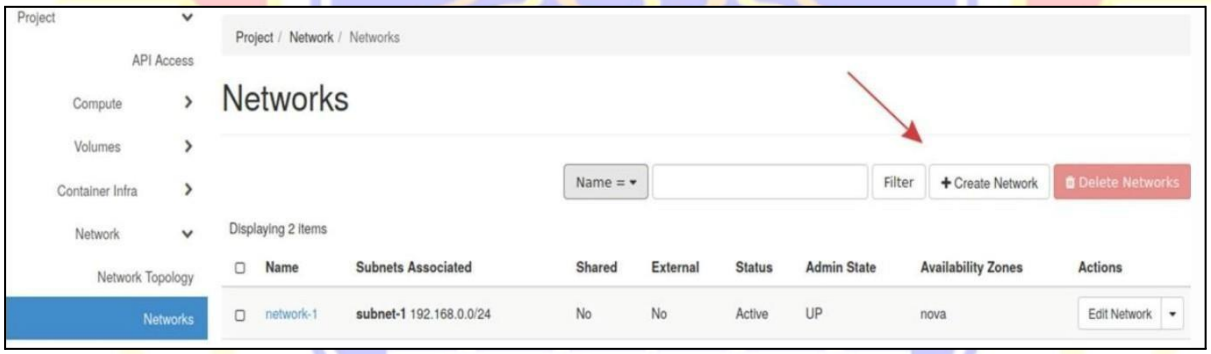
Create a network:

Step 1: Getting started In Horizon, notice the sections on the left called Project and Identity.

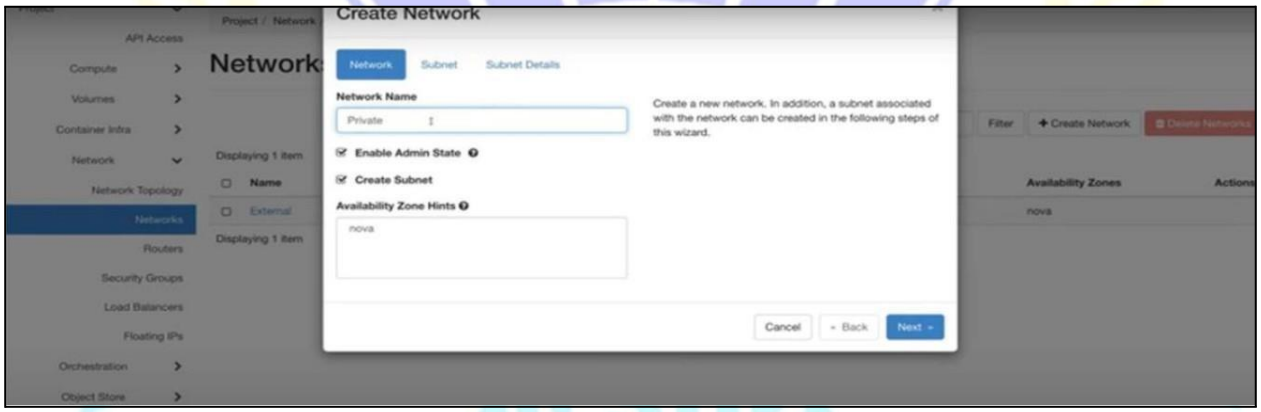
Components of an OpenStack cloud, like a private network, are created through the Project tab.



To create a network, under Project on the left, find Network, then navigate to Networks. Finally, locate the Create Network button near the top right.

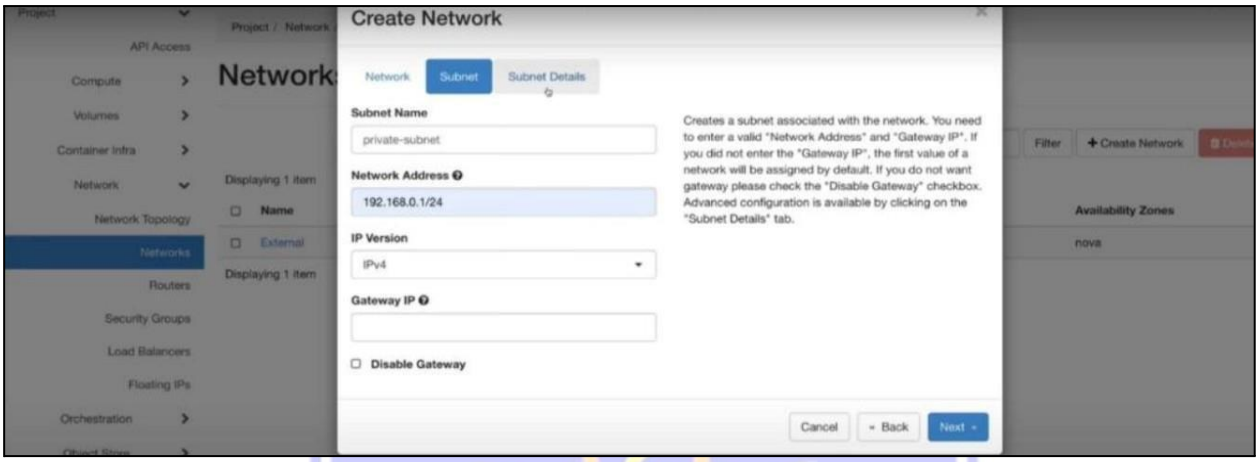


Step 2: Create network

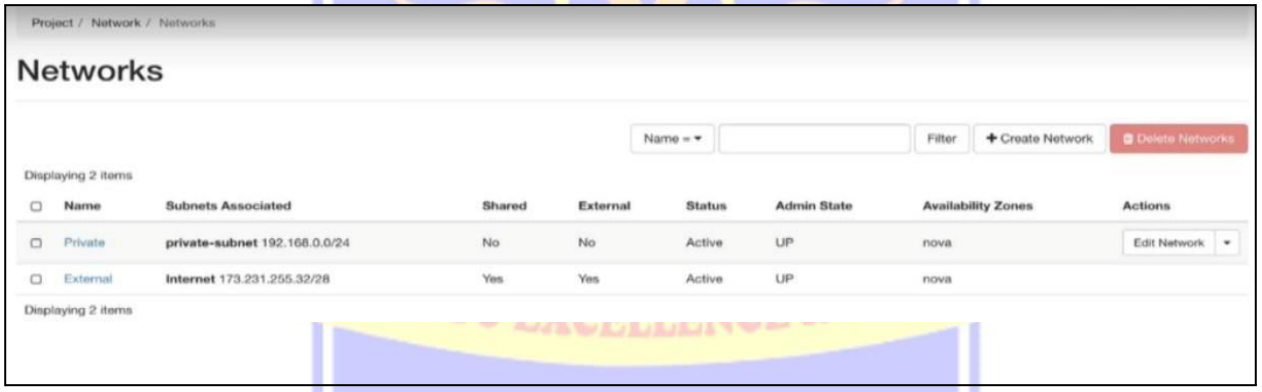


Step 3: Create subnet

- Subnet Name -- Specify a name for the subnet, this example is called subnet-1
- Network Address--Choose a network in CIDR notation. This example uses 1G2.168.0.1/24.
- Gateway IP --Optionally choose the gateway IP forthis network.Ifthe gateway IP is not filled out, one will be chosen by the Neutron service

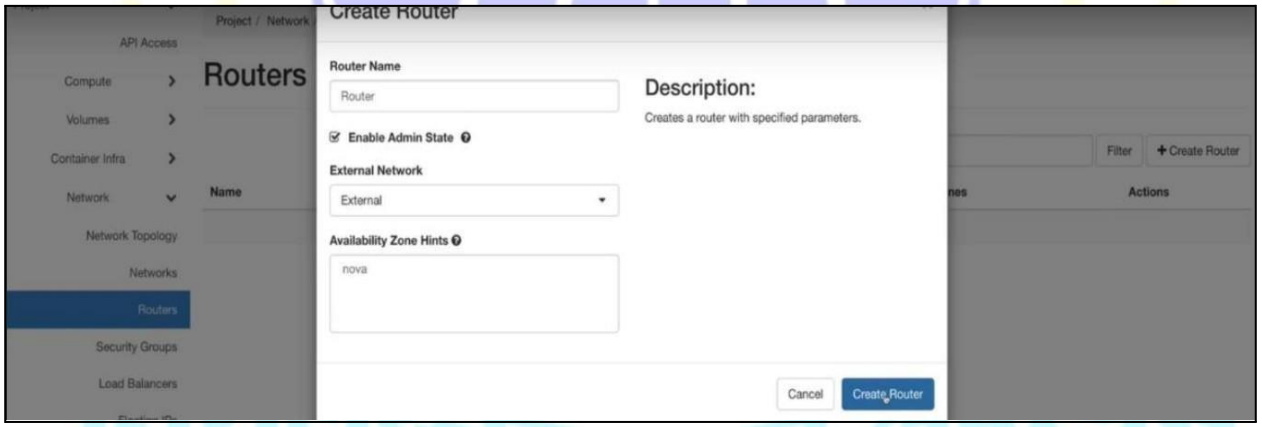


Step 4: Confirm network creation

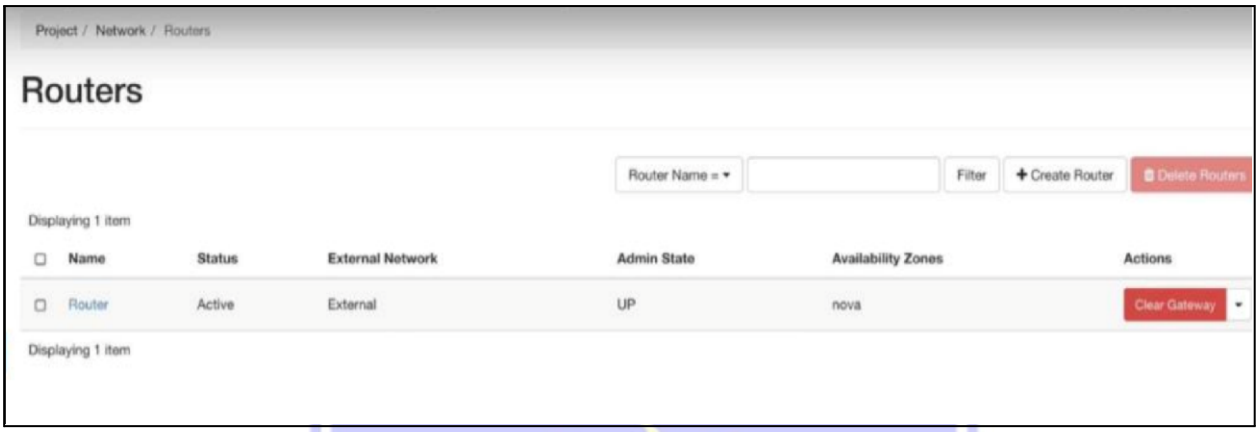


Create a Router

1] To create a new router, click the Create Router button near the top right.

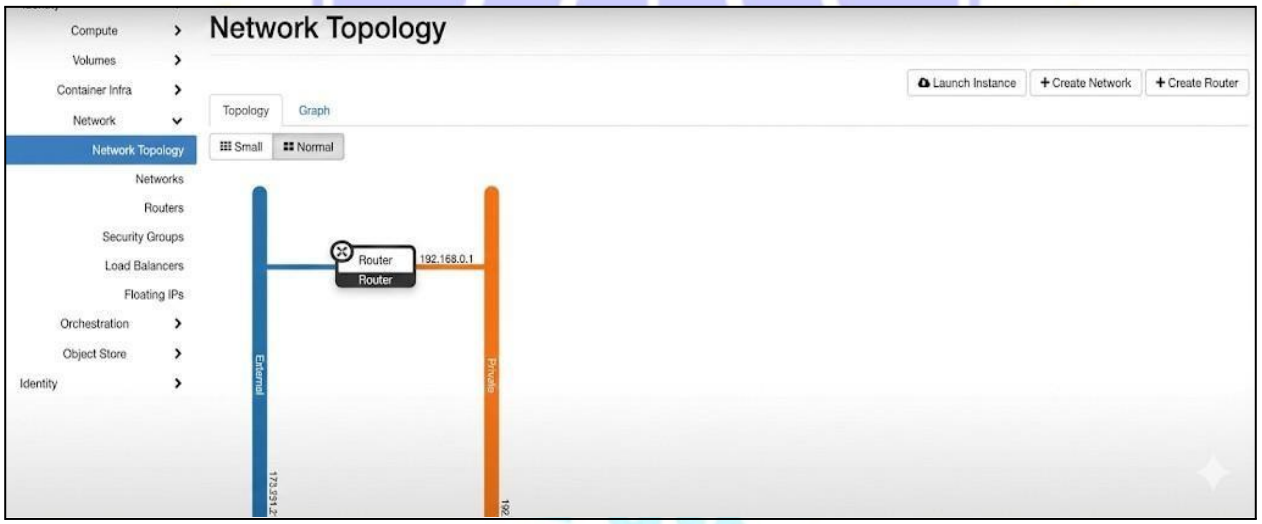


2] Confirm the router has been created Once created it will show in the list of routers, under private ->Routers.



3] Connect router to private network

4] Confirm the networks are connected



Learning Outcomes:

1. Able to understand the working of OpenStack cloud platform.
2. Able to create and configure private network and subnet.
3. Capable of creating and managing routers in cloud environment.
4. Understand the concept of Infrastructure as a Service (IaaS).
5. Able to implement secure access using SSH keys.
6. Gain practical knowledge of cloud networking setup.

Course Outcomes:

1. Understand cloud computing architecture and services.
2. Implement cloud-based virtual infrastructure.
3. Configure networking components in cloud platform.
4. Apply security practices in cloud environment.

Conclusion:

OpenStack was successfully implemented. Private network, subnet, and router were created and configured properly. The experiment helped in understanding cloud infrastructure and networking concepts practically.

Viva Questions:

1. What is OpenStack?
2. What is IaaS in cloud computing?
3. What is the purpose of Neutron service?
4. What is CIDR notation?
5. Difference between public and private network?

For Faculty use:

Correction Parameters	Formative Assessment[40%]	Timely Completion of Practical[40%]	Attendance Learning Attitude[20%]