Practical Journal

In

Advanced Cloud Computing

Submitted by

25FMSCIT010 – Harshal Khade

for the award of the Degree of

MASTERS OF SCIENCE

(INFORMATION TECHNOLOGY)

PART-1

DEPARTMENT OF INFORMATION TECHNOLOGY

KISHINCHAND CHELLARAM COLLEGE

(Affiliated to University of HSNC)

MUMBAI, 400020

2025-26

KISHINCHAND CHELLARAM COLLEGE

CHURCHGATE, MUMBAI – 400 020.

DEPARTMENT OF INFORMATION TECHNOLOGY

M.Sc. I.T. PART- I

CERTIFICATE

This is to certify that the Practical conducted by, Mr. **Harshal Khade** for M.Sc.(IT) Part- I Semester- I, Seat No: **25FMSCIT010** at Kishinchand Chellaram College in partial fulfilment for the MASTERS OF SCIENCE (INFORMATION TECHNOLOGY). Degree Examination for semester I has been periodically examined and signed, and the course of term work has been satisfactorily carried out for the year 2025 - 2026. This Practical journal had not been submitted for any other examination and does not form part of any other course undergone by the candidate.

Signature	**Signature**	**Signature**
Lecturer-In-Charge	**External Examiner**	**Course Coordination**
Guided By	**Examined By**	**Certified BY**

College Stamp

SUBJECT CODE- BIT511D

Advanced Cloud Computing

KISHINCHAND CHELLARAM COLLEGE
Churchgate, Mumbai – 400 020

INDEX

Prac. No.	Date	Topic	Sign
1		Implementing Client Server communication model using TCP	
2		Client Server based Program using UDP	
3		Implementing object communication using RMI	
4		Multicast Socket	
5		Hosting multiple websites on a single Apache server inside a Virtual Machine using Virtual Hosts, and Accessing them from host machine	
6		SSH Key Login	
7		Network & Compliance Hardening for a Linux Server	
Assign -ment			
1		Implementation of Web Service	

Practical 1

Aim: Write a program for implementing Client Server communication model using TCP.

Code:

tcpServerPrime.java -

```
import java.net.*;
import java.io.*;
class tcpServerPrime {
   public static void main(String args[]) {
      try {
         ServerSocket ss=new ServerSocket(8001);
         System.out.println("Server Started. ........");
         Socket s=ss.accept();
         DataInputStream in = new DataInputStream(s.getInputStream());
         int x=in.readInt();
         DataOutputStream otc=new
         DataOutputStream(s.getOutputStream());
         int y=x/2;
         if(x==1 || x==2 || x==3) {
            otc.writeUTF(x + "is Prime");
            System.exit(0);
         }
         for(int i=2;i<=y;i++) {
            if(x % i != 0) {
               otc.writeUTF(x + "is Prime");
            } else {
               otc.writeUTF(x + "is not Prime");
            }
         }
      } catch(Exception e) {
         System.out.println(e.toString());
      }
   }
}
```

tcpClientPrime.java -

```
import java.net.*;
import java.io.*;
class tcpClientPrime {
   public static void main(String args[]) {
```

```
        try {
            Socket cs = new Socket("LocalHost", 8001);
            BufferedReader infu = new BufferedReader(new InputStreamReader(System.in));
            System.out.println("Enter a number: ");
            int a = Integer.parseInt(infu.readLine());
            DataOutputStream out = new DataOutputStream(cs.getOutputStream());
            out.writeInt(a);
            DataInputStream in = new DataInputStream(cs.getInputStream());
            System.out.println(in.readUTF());
            cs.close();
        } catch (Exception e) {
            System.out.println(e.toString());
        }
    }
}
```

Output –

```
C:\Users\harsh\OneDrive\Documents\MSCPART1SEM1\ACC>java tcpServerPrime
Server Started.

C:\Users\harsh\OneDrive\Documents\MSCPART1SEM1\ACC>java tcpServerPrime
Server Started.

C:\Users\harsh\OneDrive\Documents\MSCPART1SEM1\ACC>java tcpServerPrime
Server Started.
```

```
C:\Users\harsh\OneDrive\Documents\MSCPART1SEM1\ACC>java tcpClientPrime
Enter a number:
3
3is Prime

C:\Users\harsh\OneDrive\Documents\MSCPART1SEM1\ACC>java tcpClientPrime
Enter a number:
10
10is not Prime
```

Practical 2

Aim: Client-Server based Program using UDP to find if the number entered is even or odd number.

Code:

udpServerEO.java -

```
import java.io.*;
import java.net.*;
public class udpServerEO {
   public static void main(String args[]) {
      try {
         DatagramSocket ds = new DatagramSocket(2000);
         byte b[] = new byte[1024];
         DatagramPacket dp = new DatagramPacket(b, b.length);
         ds.receive(dp);
         String str = new String(dp.getData(), 0, dp.getLength());
         System.out.println(str);
         int a = Integer.parseInt(str);
         String s = new String();
         if (a % 2 == 0)
            s = "Number is even";
         else
            s = "Number is odd";
         byte b1[] = new byte[1024];
         b1 = s.getBytes();
         DatagramPacket dp1 = new DatagramPacket(b1, b1.length, InetAddress.getLocalHost(),
1000);
         ds.send(dp1);
      } catch (Exception e) {
         e.printStackTrace();
      }
   }
}
```

udpClientEO.java -

```
import java.io.*;
import java.net.*;
public class udpClientEO {
   public static void main(String args[]) {
      try {
         DatagramSocket ds = new DatagramSocket(1000);
```

```
            BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
            System.out.println("Enter a number : ");
            String num = br.readLine();
            byte b[] = new byte[1024];
            b = num.getBytes();
            DatagramPacket dp = new DatagramPacket(b, b.length, InetAddress.getLocalHost(), 2000);
            ds.send(dp);
            byte b1[] = new byte[1024];
            DatagramPacket dp1 = new DatagramPacket(b1, b1.length);
            ds.receive(dp1);
            String str = new String(dp1.getData(), 0, dp1.getLength());
            System.out.println(str);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Output –

```
C:\Users\harsh\OneDrive\Documents\MSCPART1SEM1\ACC>java udpServerEO
12

C:\Users\harsh\OneDrive\Documents\MSCPART1SEM1\ACC>java udpServerEO
19
```

```
C:\Users\harsh\OneDrive\Documents\MSCPART1SEM1\ACC>java udpClientEO
Enter a number :
12
Number is even

C:\Users\harsh\OneDrive\Documents\MSCPART1SEM1\ACC>java udpClientEO
Enter a number :
19
Number is odd
```

Practical 3

Aim: Write a program to show the object communication using RMI to display current date and time.

Code:

InterDate.java -

```
import java.rmi.*;
public interface InterDate extends Remote {
   public String display()
         throws Exception;
}
```

ServerDate.java -

```
import java.rmi.*;
import java.rmi.server.*;
import java.util.*;
public class ServerDate extends UnicastRemoteObject implements
      InterDate {
   public ServerDate() throws Exception { }
   public String display() throws Exception {
      String str = "";
      Date d = new Date();
      str = d.toString();
      return str;
   }
   public static void main(String args[])
         throws Exception {
      ServerDate s1 = new ServerDate();
      Naming.bind("DS", s1);
      System.out.println("Object registered.... ");
   }
}
```

ClientDate.java -

```
import java.rmi.*;
import java.io.*;
public class ClientDate {
   public static void main(String args[])
         throws Exception {
      String s1;
      InterDate h1 = (InterDate) Naming.lookup("DS");
      s1 = h1.display();
```

```
        System.out.println(s1);
    }
}
```

Output -

```
D:\ACC>start rmiregistry

D:\ACC>java ServerDate
Object registered.....
```

```
D:\ACC>java ClientDate
Sun Sep 14 21:55:47 IST 2025

D:\ACC>
```

Practical 4

Aim: Multicast Socket Example.

Code:

BroadcastServer.java -

```
import java.net.*;
import java.io.*;
import java.util.*;
public class BroadcastServer {
   public static final int PORT = 1234;
   public static void main(String args[])
         throws Exception {
      MulticastSocket socket;
      DatagramPacket packet;
      InetAddress address;
      address = InetAddress.getByName("239.1.2.3");
      socket = new MulticastSocket();
      socket.joinGroup(address);
      byte[] data = null;
      for (;;) {
         Thread.sleep(10000);
         System.out.println("Sending");
         String str = ("This is neha calling...");
         data = str.getBytes();
         packet = new DatagramPacket(data, str.length(), address, PORT);
         socket.send(packet);
      }
   }
}
```

BroadcastClient.java -

```
import java.net.*;
import java.io.*;
public class BroadcastClient {
   public static final int PORT = 1234;
   public static void main(String args[])
         throws Exception {
      MulticastSocket socket;
      DatagramPacket packet;
      InetAddress address;
      address = InetAddress.getByName("239.1.2.3");
```

```
        socket = new MulticastSocket(PORT);
        socket.joinGroup(address);
        byte[] data = new byte[100];
        packet = new DatagramPacket(data, data.length);
        for (;;) {
           socket.receive(packet);
           String str = new String(packet.getData());
           System.out.println("Message received from" +
                 packet.getAddress() + "Message is:" + str);
        }
     }
}
```

Output -

```
D:\Java\jdk-24\bin>javac BroadcastServer.java
Note: BroadcastServer.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.

D:\Java\jdk-24\bin>java BroadcastServer
Sending
Sending
Sending
```

```
D:\Java\jdk-24\bin>javac BroadcastClient.java
Note: BroadcastClient.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.

D:\Java\jdk-24\bin>java BroadcastClient
Message received from/192.168.0.106Message is:This is neha calling...
Message received from/192.168.0.106Message is:This is neha calling...
Message received from/192.168.0.106Message is:This is neha calling...
Message received from/192.168.0.106Message is:This is neha calling...
```

Practical 5

Aim: To configure and host multiple websites on a single Apache server inside a Virtual Machine using Virtual Hosts, and access them from a host machine by mapping domain names to the VM's IP address.

Apache supports Virtual Hosting, which allows a single web server to serve multiple domain names using different configuration files. Each website is stored in a different directory and mapped using a unique domain like example.com, test.com. This simulates shared hosting on production servers.

Step 1: Install Apache (If Not Installed)

sudo apt update

sudo apt install apache2 –y

```
ubuntu@mitali:~$ sudo apt update
[sudo] password for ubuntu:
Get:1 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Hit:2 http://archive.ubuntu.com/ubuntu noble InRelease
Get:3 http://archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [1
340 kB]
Get:5 http://archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:6 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [2
19 kB]
```

```
ubuntu@mitali:~$ sudo apt install apache2 -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils libapr1t64 libaprutil1-dbd-sqlite3 l
ibaprutil1-ldap
  libaprutil1t64 ssl-cert
Suggested packages:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom www-browser ufw
The following NEW packages will be installed:
  apache2 apache2-bin apache2-data apache2-utils libapr1t64 libaprutil1-dbd-s
qlite3 libaprutil1-ldap
  libaprutil1t64 ssl-cert
0 upgraded, 9 newly installed, 0 to remove and 284 not upgraded.
Need to get 1920 kB of archives.
```

Start and enable Apache:

sudo systemctl start apache2

sudo systemctl enable apache2

```
ubuntu@mitali:~$ sudo service apache2 start
 * Starting Apache httpd web server apache2
 *
```

```
ubuntu@mitali:~$ sudo systemctl enable apache2
Synchronizing state of apache2.service with SysV service script with /usr/lib
/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable apache2
ubuntu@mitali:~$
```

```
ubuntu@mitali:~$ sudo service apache2 status
 * apache2 is running
ubuntu@mitali:~$
```

Step 2: Create Website Folders and HTML Pages

sudo mkdir -p /var/www/example.com

sudo mkdir -p /var/www/test.com

Add simple HTML files:

echo "<h1>Hello from example.com</h1>" | sudo tee /var/www/example.com/index.html

echo "<h1>Hello from test.com</h1>" | sudo tee /var/www/test.com/index.html

```
ubuntu@mitali:~$ sudo mkdir -p /var/www/example.com
ubuntu@mitali:~$ sudo mkdir -p /var/www/test.com
ubuntu@mitali:~$ echo "<h1>Hello from example.com</h1>" | sudo tee /var/www/e
xample.com/index.html
<h1>Hello from example.com</h1>
ubuntu@mitali:~$ echo "<h1>Hello from test.com</h1>" | sudo tee /var/www/test
.com/index.html
<h1>Hello from test.com</h1>
ubuntu@mitali:~$
```

Step 3: Set Permissions

sudo chown -R www-data:www-data /var/www/example.com

sudo chown -R www-data:www-data /var/www/test.com

sudo chmod -R 755 /var/www

chown = **change ownership**

-R = apply changes **recursively** (to all files and folders inside)

www-data:www-data = set the **owner and group** to Apache's default user (www-data)

/var/www/example.com = the folder you are changing

```
ubuntu@mitali:~$ sudo chown -R www-data:www-data /var/www/example.com
ubuntu@mitali:~$ sudo chown -R www-data:www-data /var/www/test.com
ubuntu@mitali:~$ sudo chmod -R 755 /var/www
ubuntu@mitali:~$
```

Step 4: Create Apache Virtual Host Config Files (ctrl + O SAVE ,Enter,Ctrl +X/Z)

1. example.com Virtual Host

sudo nano /etc/apache2/sites-available/example.com.conf

```
<VirtualHost *:80>
    ServerName example.com
    DocumentRoot /var/www/example.com
    <Directory /var/www/example.com>
        Options Indexes FollowSymLinks
        AllowOverride All
```

```
    Require all granted
  </Directory>
  ErrorLog ${APACHE_LOG_DIR}/example.com-error.log
  CustomLog ${APACHE_LOG_DIR}/example.com-access.log combined
</VirtualHost>
```

2. test.com Virtual Host

sudo nano /etc/apache2/sites-available/test.com.conf

```
<VirtualHost *:80>
  ServerName test.com
  DocumentRoot /var/www/test.com
  <Directory /var/www/test.com>
    Options Indexes FollowSymLinks
    AllowOverride All
    Require all granted
  </Directory>
  ErrorLog ${APACHE_LOG_DIR}/test.com-error.log
  CustomLog ${APACHE_LOG_DIR}/test.com-access.log combined
</VirtualHost>
```

```
ubuntu@mitali:~$ sudo nano /etc/apache2/sites-available/example.com.conf
ubuntu@mitali:~$ sudo nano /etc/apache2/sites-available/test.com.conf
ubuntu@mitali:~$
```

🔹 **Step 5: Enable the Sites and Reload Apache**

sudo a2ensite example.com.conf
sudo a2ensite test.com.conf
sudo systemctl reload apache2
Check config: sudo apache2ctl configtest
Output : Syntax OK

```
ubuntu@mitali:~$ sudo a2ensite example.com.conf
Enabling site example.com.
To activate the new configuration, you need to run:
  service apache2 reload
ubuntu@mitali:~$ sudo a2ensite test.com.conf
Enabling site test.com.
To activate the new configuration, you need to run:
  service apache2 reload
```

```
ubuntu@mitali:~$ sudo service apache2 reload
 * Reloading Apache httpd web server apache2
 *
ubuntu@mitali:~$ sudo apache2ctl configtest
Syntax OK
ubuntu@mitali:~$
```

Step 6: Get Your VM's IP Address

ip a

```
ubuntu@mitali:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group def
ault qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet 10.255.255.254/32 brd 10.255.255.254 scope global lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group d
efault qlen 1000
    link/ether 00:15:5d:02:5f:e7 brd ff:ff:ff:ff:ff:ff
    inet 172.24.20.68/20 brd 172.24.31.255 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::215:5dff:fe02:5fe7/64 scope link
       valid_lft forever preferred_lft forever
ubuntu@mitali:~$
```

Step 7: Map Domains in Host OS (hosts file)

On Windows:

1. Open Notepad **as Administrator**
2. Open file: C:\Windows\System32\drivers\etc\hosts
3. Add at the bottom:

192.168.68.128 example.com

192. 168.68.128 test.com

4. Save and exit.

On Linux/macOS:

sudo nano /etc/hosts

Add:

192.168.56.101 example.com

192.168.56.101 test.com

```
  GNU nano 7.2                        /etc/hosts
# This file was automatically generated by WSL.
# [network]
# generateHosts = false
127.0.0.1       localhost
127.0.1.1       mitali.localdomain      mitali
172.24.20.68    example.com
172.24.20.68    test.com
```

Step 8: Test in Browser (from Host OS)

- Open your browser
- Visit: http://example.com

Output: Hello from example.com

- Visit: http://test.com

Output: Hello from test.com

Practical 6

Aim: SSH Key Login.

On a cloud server (like AWS EC2), if **anyone knows the password**, they can login and **hack the server**. So we do 3 things:

1. **Create a new admin user** (instead of using root)
2. **Use SSH Key Login** (so no one can hack using passwords)
3. **Disable Root Login** (so no one can target root)

This is the **basic cloud security rule**.

Step 1: Create a new admin user

Run this command:

sudo adduser adminuser

What happens?

- Computer asks: Enter password → **Create any password**
- Then it asks name, phone etc → **Just press Enter for all**

Now give this user permission to run admin commands:

sudo usermod -aG sudo adminuser

Meaning:

We are making adminuser part of the **sudo group**, which means this user can run commands like an administrator.

Result:

We now have a **safe admin user** who will manage the server.

We will **stop using the root user**.

```
ubuntu@mitali:~$ sudo adduser adminuser
info: Adding user `adminuser' ...
info: Selecting UID/GID from range 1000 to 59999 ...
info: Adding new group `adminuser' (1002) ...
info: Adding new user `adminuser' (1002) with group `adminuser (1002)' ...
info: Creating home directory `/home/adminuser' ...
info: Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for adminuser
Enter the new value, or press ENTER for the default
        Full Name []: Admin
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] y
info: Adding new user `adminuser' to supplemental / extra groups `users' ...
info: Adding user `adminuser' to group `users' ...
ubuntu@mitali:~$
ubuntu@mitali:~$ sudo usermod -aG sudo adminuser
ubuntu@mitali:~$
```

If You Are On Windows

Step 1: Download the Tool (for Windows)
MobaXterm
This is the tool that makes SSH easy (no ssh-copy-id needed).
Download Link: https://mobaxterm.mobatek.net/download.html
Click **"MobaXterm Home Edition → Installer Edition"**
Then install it normally.

This is on linux:

Step 1 — Make Sure SSH Server Is Installed
Inside your **VM terminal** run:
sudo apt update
sudo apt install openssh-server -y

```
ubuntu@mitali:~$ sudo apt install openssh-server -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  openssh-client openssh-sftp-server ssh
Suggested packages:
  keychain libpam-ssh monkeysphere molly-guard ufw
The following packages will be upgraded:
  openssh-client openssh-server openssh-sftp-server ssh
4 upgraded, 0 newly installed, 0 to remove and 280 not upgraded.
Need to get 1457 kB of archives.
```

Now start + enable SSH:
sudo systemctl enable ssh
sudo systemctl start ssh
Check status:
sudo systemctl status ssh
If it says **active (running)**

```
ubuntu@mitali:~$ sudo systemctl enable ssh
Synchronizing state of ssh.service with SysV service script with /usr/lib/sys
temd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable ssh
ubuntu@mitali:~$
```

```
ubuntu@mitali:~$ sudo service ssh start
 * Starting OpenBSD Secure Shell server sshd                          [ OK ]
ubuntu@mitali:~$
ubuntu@mitali:~$ sudo service ssh status
 * sshd is running
ubuntu@mitali:~$
```

Step 3 — Allow SSH in the Firewall (If Needed)

Inside VM:

sudo ufw allow OpenSSH

sudo ufw enable

sudo ufw status

```
ubuntu@mitali:~$ sudo apt install ufw
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  iptables libip4tc2 libip6tc2 libnetfilter-conntrack3 libnfnetlink0
  libnftables1 libnftnl11 nftables
Suggested packages:
  firewalld
The following NEW packages will be installed:
  iptables libip4tc2 libip6tc2 libnetfilter-conntrack3 libnfnetlink0
  libnftables1 libnftnl11 nftables ufw
0 upgraded, 9 newly installed, 0 to remove and 280 not upgraded.
```

```
ubuntu@mitali:~$ sudo ufw allow OpenSSH
Rules updated
Rules updated (v6)
ubuntu@mitali:~$
ubuntu@mitali:~$ sudo ufw enable
Firewall is active and enabled on system startup
ubuntu@mitali:~$
ubuntu@mitali:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
OpenSSH                    ALLOW       Anywhere
OpenSSH (v6)               ALLOW       Anywhere (v6)
```

Test SSH Again from Host (MobaXterm / CMD)

ssh adminuser@192.168.x.x (In session tool)

```
ubuntu@mitali:~$ ssh adminuser@172.24.20.68
The authenticity of host '172.24.20.68 (172.24.20.68)' can't be established.
ED25519 key fingerprint is SHA256:aZhC7Sy2rqy2mcguSrHzHAI7L6NQNIkXnTh9DsMVT3A
.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:1: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '172.24.20.68' (ED25519) to the list of known host
s.
adminuser@172.24.20.68's password:
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.6.87.2-microsoft-standard-WSL2 x86
_64)
```

Step 4 — Copy Your Public Key to the Server

From your **host machine** (MobaXterm terminal):

ssh-copy-id adminuser@<server-ip>

Example:

ssh-copy-id adminuser@192.168.1.45

You will be asked for **adminuser password (the one you created)**.

```
adminuser@mitali:~$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/adminuser/.ssh/id_rsa):
Created directory '/home/adminuser/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/adminuser/.ssh/id_rsa
Your public key has been saved in /home/adminuser/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:qNC7EgQ33XQGh7jWe0wbFM7fUR0lIl39YF0bHyNPcNk adminuser@mitali
The key's randomart image is:
+---[RSA 3072]----+
|    . +oo=....+=@O|
|. o o o*.  ..oBoE|
| o . o .o   .. +o|
|  ..o ..o. . .  .|
| .... .+So. .    |
|  .. o. +        |
|   .o  .         |
|  .  .           |
|   ..            |
+----[SHA256]-----+
adminuser@mitali:~$
adminuser@mitali:~$ ssh-copy-id adminuser@172.24.20.68
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/adminuse
r/.ssh/id_rsa.pub"
The authenticity of host '172.24.20.68 (172.24.20.68)' can't be established.
ED25519 key fingerprint is SHA256:aZhC7Sy2rqy2mcguSrHzHAI7L6NQNIkXnTh9DsMVT3A
.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filt
er out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are pro
mpted now it is to install the new keys
adminuser@172.24.20.68's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'adminuser@172.24.20.68'"
and check to make sure that only the key(s) you wanted were added.

adminuser@mitali:~$
```

After success, test login:

ssh adminuser@192.168.1.45

If it logs in **without asking password → success** 🎯

```
adminuser@mitali:~$ ssh 'adminuser@172.24.20.68'
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.6.87.2-microsoft-standard-WSL2 x86
_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro
```

Step 6: Disable Root Login & Password Login

Inside VM, open the SSH config file:

sudo nano /etc/ssh/sshd_config

Change (or add) the following lines:

PermitRootLogin no

PasswordAuthentication no

Save & exit:

CTRL + O → Enter → CTRL + X

Restart SSH:

sudo systemctl restart ssh

```
adminuser@mitali:~$ sudo nano /etc/ssh/sshd_config
```

```
adminuser@mitali:~$ sudo service ssh restart
 * Restarting OpenBSD Secure Shell server sshd                    [ OK ]
adminuser@mitali:~$
```

Practical 7

Aim: Network & Compliance Hardening for a Linux Server.

To secure a Linux server by:

- Allowing only required network ports,
- Monitoring if system files are changed,
- Preventing any user from overusing system resources.

PART 1 - Firewall Configuration Using UFW

Step 1: Install UFW

sudo apt install ufw -y

```
adminuser@mitali:~$ sudo apt install ufw -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ufw is already the newest version (0.36.2-6).
0 upgraded, 0 newly installed, 0 to remove and 280 not upgraded.
adminuser@mitali:~$
```

Step 2: Set Default Rules

- **Block all incoming requests** (safer)
- **Allow outgoing traffic**

sudo ufw default deny incoming

sudo ufw default allow outgoing

```
adminuser@mitali:~$ sudo ufw default deny incoming
Default incoming policy changed to 'deny'
(be sure to update your rules accordingly)
adminuser@mitali:~$
adminuser@mitali:~$ sudo ufw default allow outgoing
Default outgoing policy changed to 'allow'
(be sure to update your rules accordingly)
adminuser@mitali:~$
```

Step 3: Allow Only Important Ports

Port	*Service*	*Why?*
22	SSH	For remote login
443	HTTPS	For secure websites
80	HTTP	We will **deny** (to practice control)

sudo ufw allow 22/tcp

sudo ufw allow 443/tcp

sudo ufw deny 80/tcp

```
adminuser@mitali:~$ sudo ufw allow 22/tcp
Rule added
Rule added (v6)
adminuser@mitali:~$ sudo ufw allow 443/tcp
Rule added
Rule added (v6)
adminuser@mitali:~$ sudo ufw deny 80/tcp
Rule added
Rule added (v6)
adminuser@mitali:~$
```

Step 4: Enable and Check Firewall

sudo ufw enable

sudo ufw status

```
ubuntu@ubuntu:~$ sudo ufw enable
Firewall is active and enabled on system startup
ubuntu@ubuntu:~$
ubuntu@ubuntu:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW       Anywhere
443/tcp                    ALLOW       Anywhere
80/tcp                     DENY        Anywhere
22/tcp (v6)                ALLOW       Anywhere (v6)
443/tcp (v6)               ALLOW       Anywhere (v6)
80/tcp (v6)                DENY        Anywhere (v6)

ubuntu@ubuntu:~$
```

Outcome:

Only allowed services can be accessed ,reduces hacking attacks.

PART 2 - Compliance Logging Using auditd

Step 1: Install auditd

sudo apt install auditd -y

```
adminuser@mitali:~$ sudo apt install auditd -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libauparse0t64
Suggested packages:
  audispd-plugins
The following NEW packages will be installed:
  auditd libauparse0t64
0 upgraded, 2 newly installed, 0 to remove and 280 not upgraded.
```

Step 2: Start the audit service

sudo systemctl start auditd

Step 3: Monitor Important File (User Account File)

sudo auditctl -w /etc/passwd -p wa -k passwd_changes

- -w → watch file
- -p wa → watch for Write and Attribute changes
- -k passwd_changes → give it a name (tag)

Step 4: View Logs When Someone Modifies Users

sudo ausearch -k passwd_changes

```
ubuntu@ubuntu:~$ sudo systemctl start auditd
ubuntu@ubuntu:~$ sudo auditctl -w /etc/passwd -p wa -k passwd_changes
ubuntu@ubuntu:~$ sudo ausearch -k passwdchanges
<no matches>
ubuntu@ubuntu:~$ sudo ausearch -k passwd_changes
----
time->Sat Nov 29 13:45:29 2025
type=PROCTITLE msg=audit(1764423929.301:99): proctitle=617564697463746C002D77002F6574632F
706173737764002D70007761002D6B007061737377645F6368616E676573
type=SYSCALL msg=audit(1764423929.301:99): arch=c000003e syscall=44 success=yes exit=1084
 a0=4 a1=7fff54be0290 a2=43c a3=0 items=0 ppid=6906 pid=6907 auid=1000 uid=0 gid=0 euid=0
 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts1 ses=2 comm="auditctl" exe="/usr/sbin/audit
ctl" subj=unconfined key=(null)
type=CONFIG_CHANGE msg=audit(1764423929.301:99): auid=1000 ses=2 subj=unconfined op=add_r
ule key="passwd_changes" list=4 res=1
ubuntu@ubuntu:~$
```

Outcome:

If any user is added, modified, or deleted → It gets recorded → Useful in investigations.

How to Verify auditd is Working

Step 1 - Make a Small Change (So audit logs get created)

We will **add a temporary user** (just for testing):

sudo adduser testuser

Enter a password → You can give any temporary password.

This action **modifies /etc/passwd**, so auditd will record it.

Step 2 — Now Check the audit Logs

Run: sudo ausearch -k passwd_changes

```
type=CWD msg=audit(1764424111.606:128): cwd="/home/ubuntu"
type=SYSCALL msg=audit(1764424111.606:128): arch=c000003e syscall=82 success=yes exit=0 a
0=7ffd0d256b00 a1=61c0bf02fea0 a2=7ffd0d256a70 a3=100 items=9 ppid=6922 pid=6947 auid=100
0 uid=0 gid=0 euid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts1 ses=2 comm="chfn" exe=
"/usr/bin/chfn" subj=unconfined key="passwd_changes"
```

This means auditd **caught and logged** the change.

To view in readable form, use:

sudo aureport -f -i

This will show logs in **human-readable** format (file names instead of inode numbers).

```
ubuntu@ubuntu:~$ sudo aureport -f -i

File Report
===============================================
# date time file syscall success exe auid event
===============================================
1. 11/29/2025 13:48:15 /etc/passwd openat yes /usr/sbin/useradd ubuntu 121
2. 11/29/2025 13:48:15 /etc/ rename yes /usr/sbin/useradd ubuntu 123
3. 11/29/2025 13:48:31 /etc/passwd openat yes /usr/bin/chfn ubuntu 127
4. 11/29/2025 13:48:31 /etc/ rename yes /usr/bin/chfn ubuntu 128
ubuntu@ubuntu:~$
```

sudo lastlog

(to check the last log)

```
testuser                                         **Never logged in**
```

Step 1: Open Limits Configuration File

sudo nano /etc/security/limits.conf

Step 2: Add These Lines at Bottom

* hard nproc 200

* soft nofile 100

Meaning

Rule	Controls	Effect
nproc	Number of running processes	Stops users from flooding the system
nofile	Number of files a program can open	Prevents crashes / overload

Step 3: Check Limits

ulimit -a

```
ubuntu@ubuntu:~$ sudo nano /etc/security/limits.conf
ubuntu@ubuntu:~$ ulimit -a
real-time non-blocking time  (microseconds, -R) unlimited
core file size              (blocks, -c) 0
data seg size               (kbytes, -d) unlimited
scheduling priority                 (-e) 0
file size                   (blocks, -f) unlimited
pending signals                     (-i) 7324
max locked memory           (kbytes, -l) 245692
max memory size             (kbytes, -m) unlimited
open files                          (-n) 1024
pipe size                (512 bytes, -p) 8
POSIX message queues         (bytes, -q) 819200
real-time priority                  (-r) 0
stack size                  (kbytes, -s) 8192
cpu time                   (seconds, -t) unlimited
max user processes                  (-u) 7324
virtual memory              (kbytes, -v) unlimited
file locks                          (-x) unlimited
ubuntu@ubuntu:~$
```

Outcome:
No single user/application can misbehave , prevents system slowdown or high cloud bills.

PART 3 Checking Open Network Ports (Monitoring Active Services)

Aim: To identify which network ports and services are currently running on the Linux server.
Why This Is Important
Every open port is a **possible entry point for attackers**.
Only necessary services should be running.
Command Used: sudo ss -tuln
(Alternative command): sudo netstat -tuln
Explanation of Options

Option	Meaning
-t	TCP connections
-u	UDP connections
-l	Listening services
-n	Show port numbers

Output

```
ubuntu@ubuntu:~$ sudo ss -tuln
Netid   State    Recv-Q   Send-Q     Local Address:Port      Peer Address:Port  Process
udp     UNCONN   0        0             127.0.0.54:53            0.0.0.0:*
udp     UNCONN   0        0          127.0.0.53%lo:53            0.0.0.0:*
udp     UNCONN   0        0                0.0.0.0:5353          0.0.0.0:*
udp     UNCONN   0        0                0.0.0.0:56732         0.0.0.0:*
udp     UNCONN   0        0                   [::]:5353             [::]:*
udp     UNCONN   0        0                   [::]:46663            [::]:*
tcp     LISTEN   0        4096       127.0.0.53%lo:53            0.0.0.0:*
tcp     LISTEN   0        4096          127.0.0.54:53            0.0.0.0:*
tcp     LISTEN   0        4096           127.0.0.1:631           0.0.0.0:*
tcp     LISTEN   0        4096               [::1]:631             [::]:*
ubuntu@ubuntu:~$
```

Shows:

- Open ports (22, 443, etc.)
- Services listening on those ports

One can verify that **only authorized services and ports are active**, reducing security risks.

PART 4 Enable Automatic Security Updates

Aim: To ensure the system automatically installs critical security updates.
Why This Is Important
Unpatched systems are vulnerable to known attacks.
Automatic updates keep the server **safe without manual work**.

Step 1: Install Unattended Upgrades

sudo apt install unattended-upgrades -y

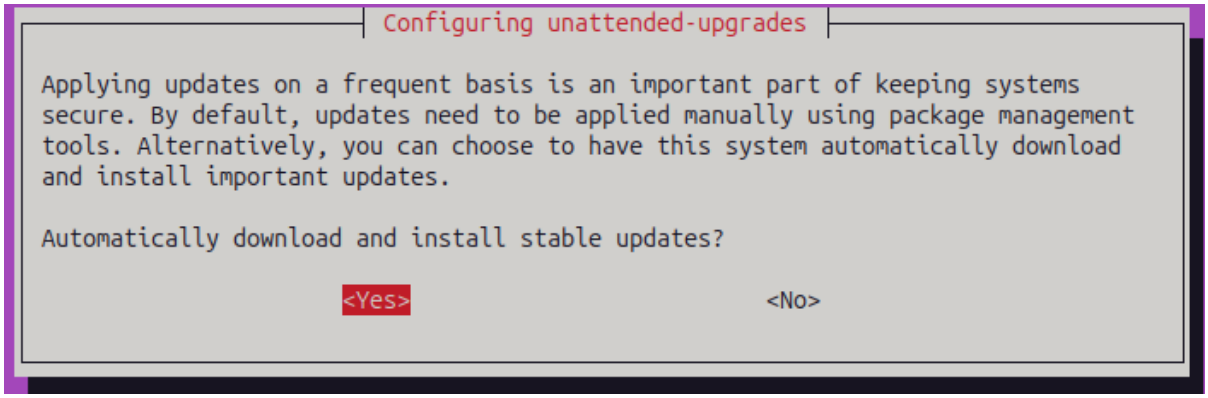
```
ubuntu@ubuntu:~$ sudo apt install unattended-upgrades -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
unattended-upgrades is already the newest version (2.9.1+nmu4ubuntu1).
unattended-upgrades set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 211 not upgraded.
ubuntu@ubuntu:~$
```

Step 2: Enable Automatic Updates

sudo dpkg-reconfigure unattended-upgrades

Select **Yes** when prompted.

```
ubuntu@ubuntu:~$ sudo dpkg-reconfigure unattended-upgrades
```



Outcome

The system will automatically install important security updates, protecting it from new vulnerabilities.

PART 5 Disable Unnecessary Services

Aim: To reduce attack surface by stopping and disabling unused services.

Why This Is Important

Unneeded services consume resources and increase security risks.

Step 1: List All Services

systemctl list-unit-files --type=service

```
UNIT FILE                                  STATE          PRESET
accounts-daemon.service                    enabled        enabled
alsa-restore.service                       static         -
alsa-state.service                         static         -
alsa-utils.service                         masked         enabled
anacron.service                            enabled        enabled
apparmor.service                           enabled        enabled
apport-autoreport.service                  static         -
apport-coredump-hook@.service              static         -
```

Step 2: Stop an Unused Service (Example: Bluetooth)

sudo systemctl stop Bluetooth

```
ubuntu@ubuntu:~$ sudo systemctl stop Bluetooth
Failed to stop Bluetooth.service: Unit Bluetooth.service not loaded.
```

Step 3: Disable the Service Permanently

sudo systemctl disable bluetooth

(Service will not start after reboot)

Outcome

Only required services remain active → improved security and performance.

Assignment:1

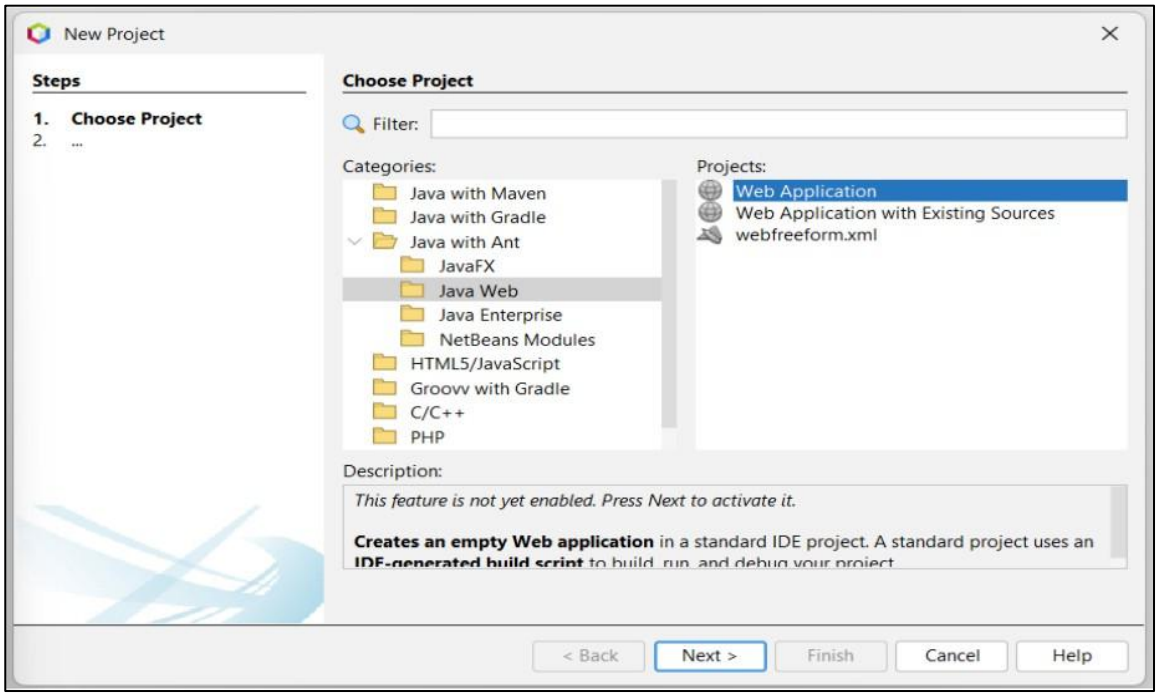
Aim: Show the implementation of web services.

What Are Web Services?

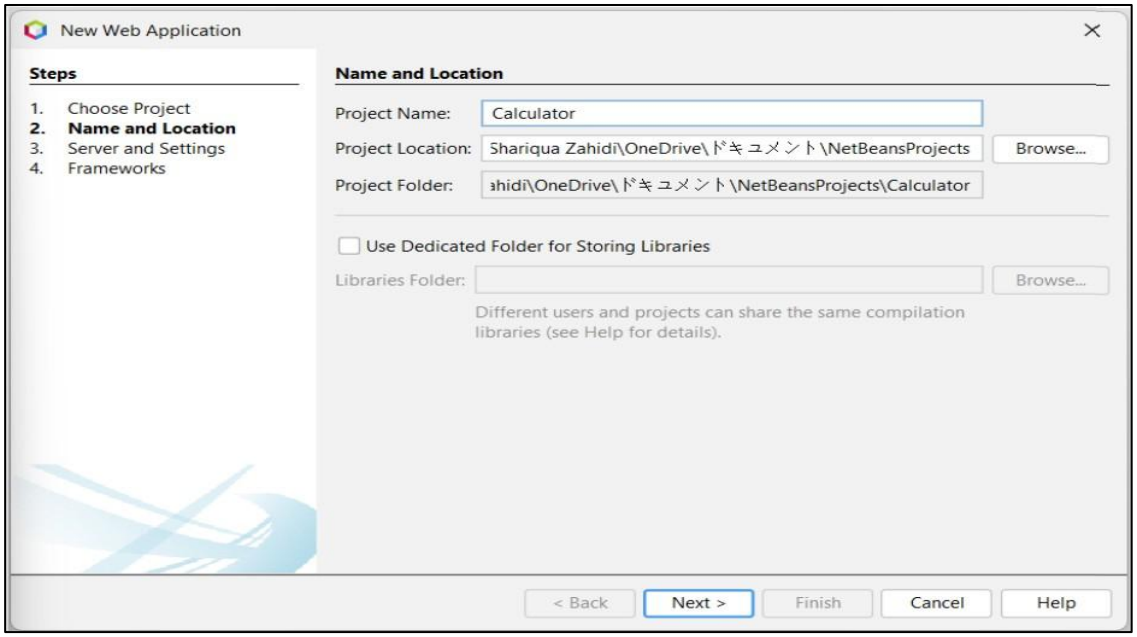
Web services are client and server applications that communicate over the World Wide Web's (WWW) HyperText Transfer Protocol (HTTP). As described by the World Wide Web Consortium (W3C), web services provide a standard means of interoperating between software applications running on a variety of platforms and frameworks. Web services are characterized by their great interoperability and extensibility, as well as their machine-process able descriptions, thanks to the use of XML. Web services can be combined in a loosely coupled way to achieve complex operations. Programs providing simple services can interact with each other to deliver sophisticated added-value services.

Create a calculator web service

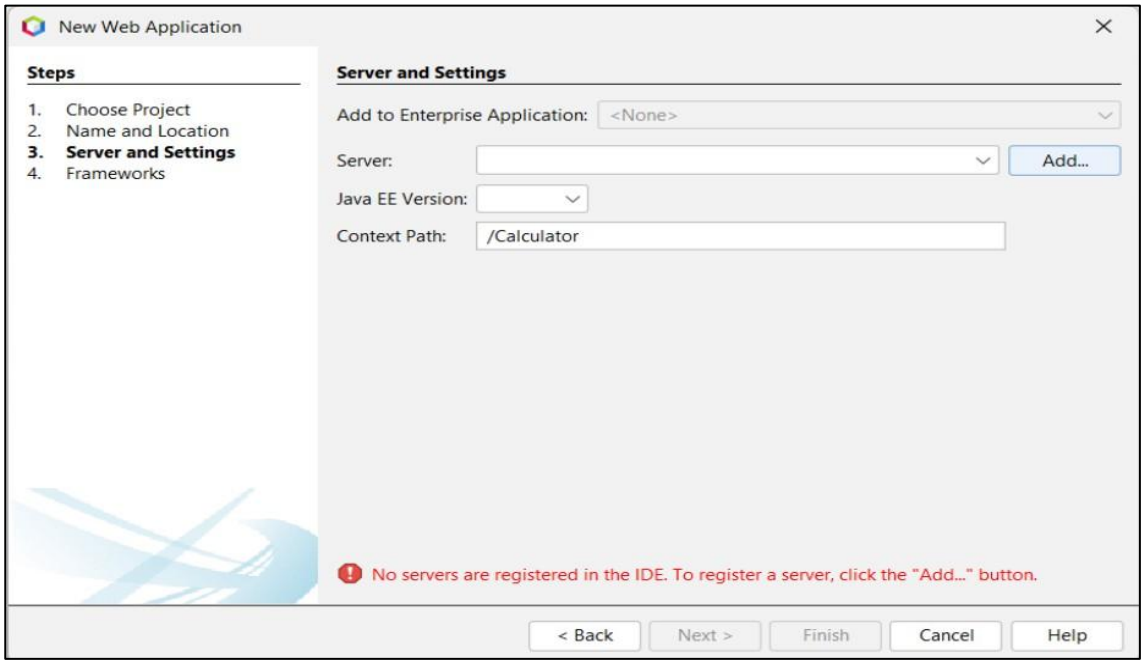
1. Choose File > New Project. Select Web Application from the Java Web

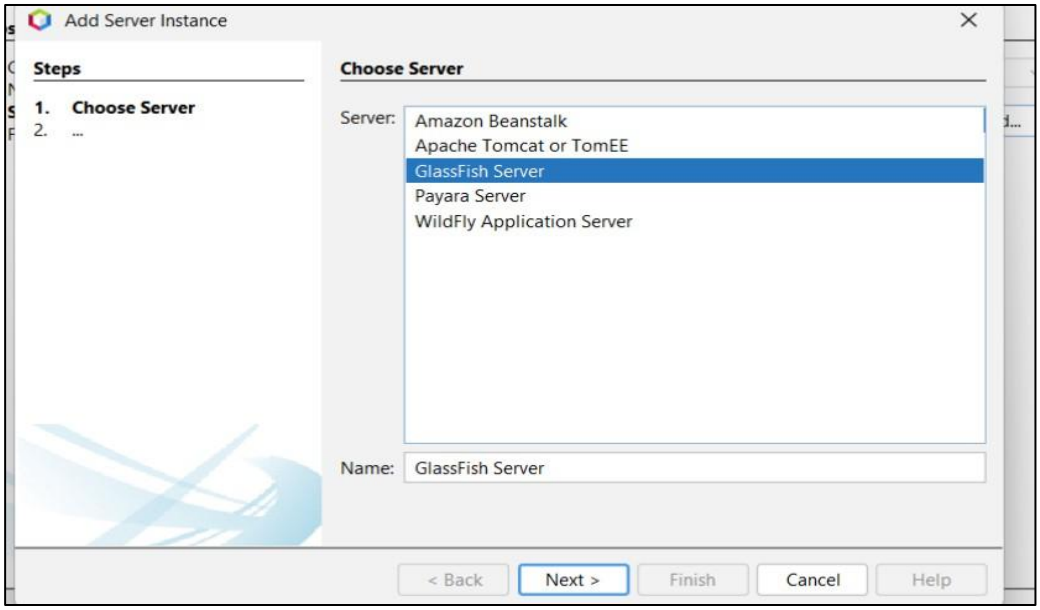


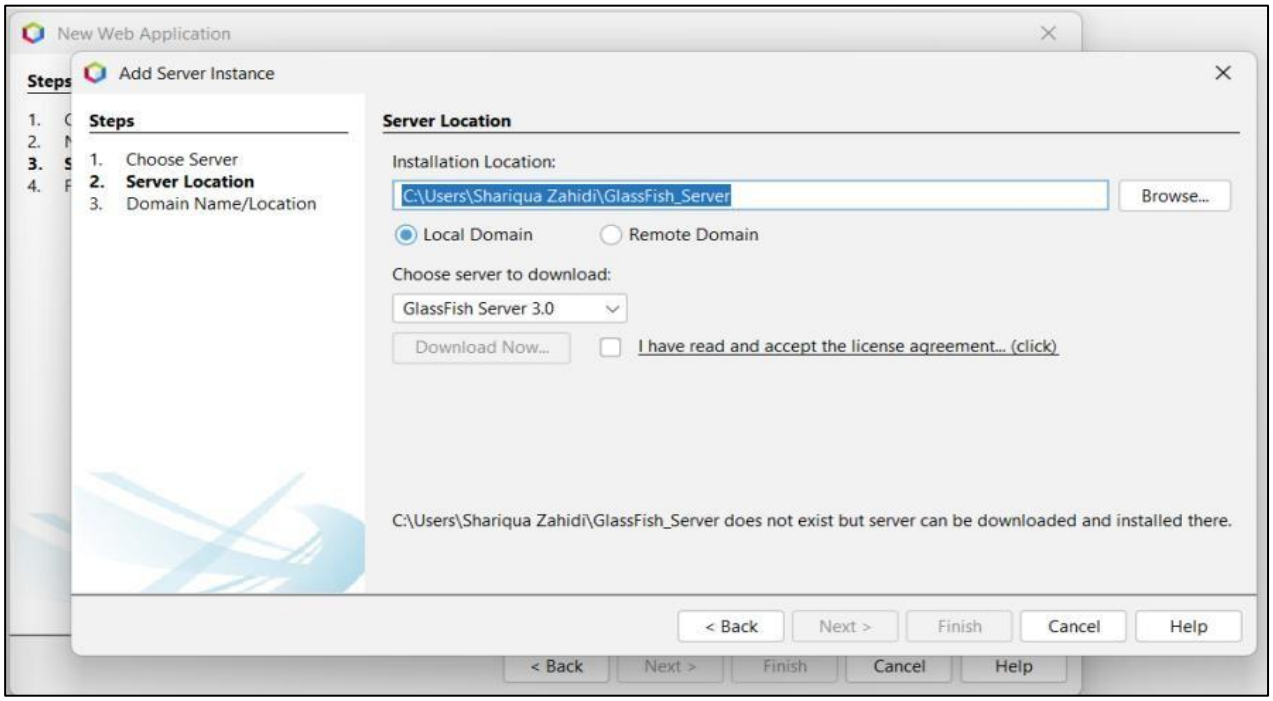
2. Name the project Calculator. Select a location for the project. Click Next.

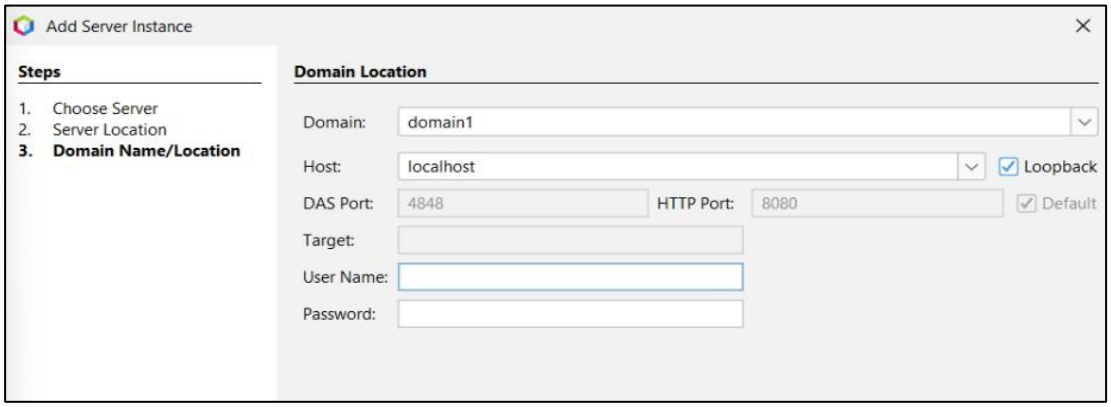


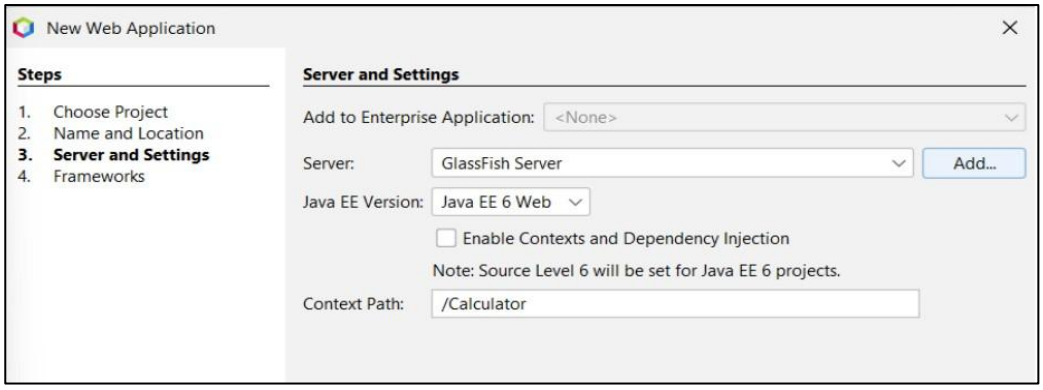
3. Add GlassFish server if not already added.

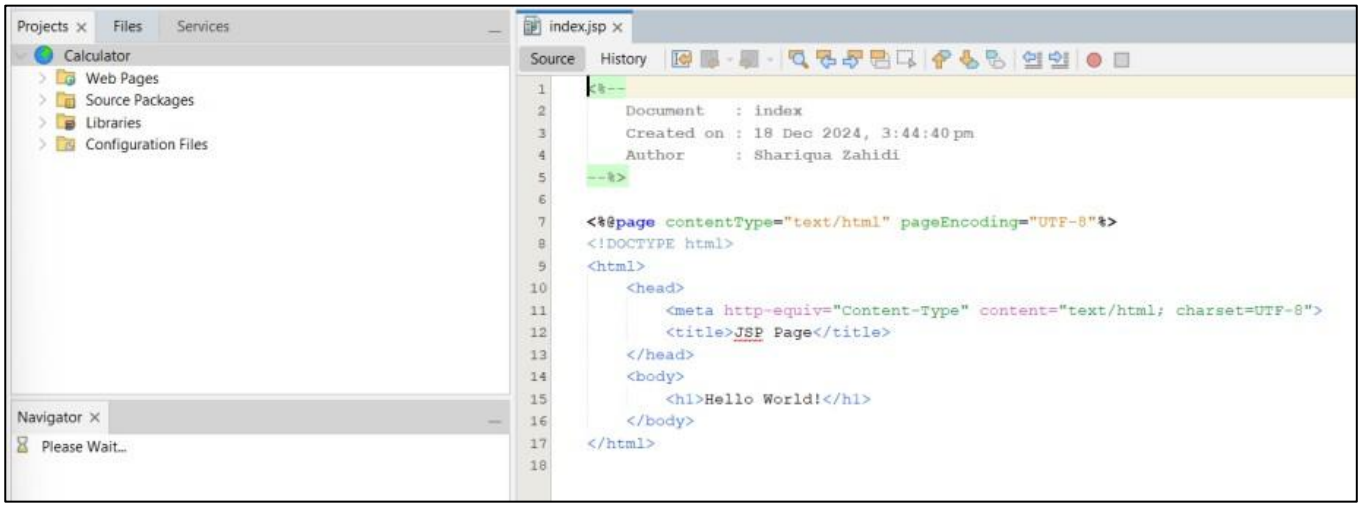



Add Server Instance
Steps
1. Choose Server
2. ...
Choose Server
Server:
Amazon Beanstalk
Apache Tomcat or TomEE
GlassFish Server
Payara Server
WildFly Application Server
Name: GlassFish Server
< Back
Next >
Finish
Cancel
Help

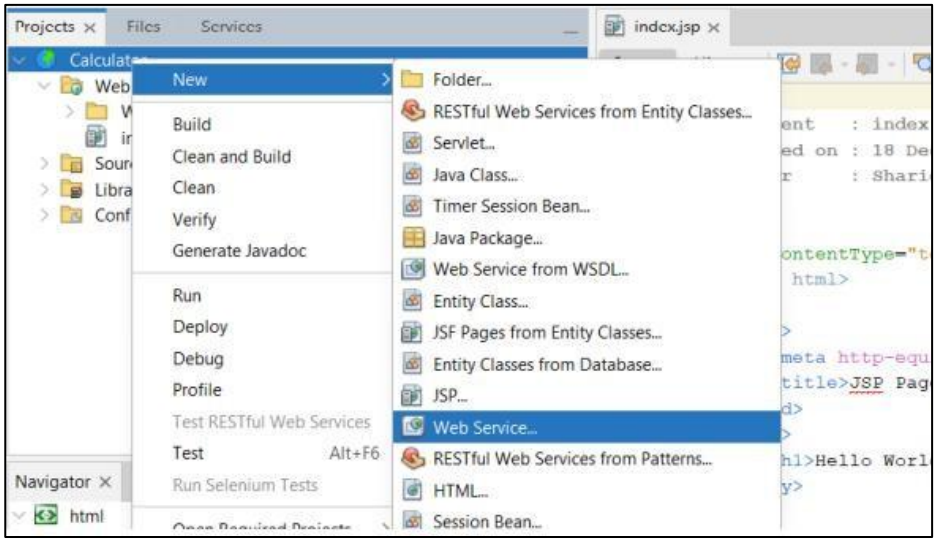

New Web Application
Add Server Instance
Steps
1. Choose Server
2. Server Location
3. Domain Name/Location
Server Location
Installation Location:
C:\Users\Shariqua Zahidi\GlassFish_Server
Browse...
Local Domain
Remote Domain
Choose server to download:
GlassFish Server 3.0
Download Now...
I have read and accept the license agreement... (click)
C:\Users\Shariqua Zahidi\GlassFish_Server does not exist but server can be downloaded and installed there.
< Back
Next >
Finish
Cancel
Help


Add Server Instance
Steps
1. Choose Server
2. Server Location
3. Domain Name/Location
Domain Location
Domain: domain1
Host: localhost
Loopback
DAS Port: 4848
HTTP Port: 8080
Default
Target:
User Name:
Password:

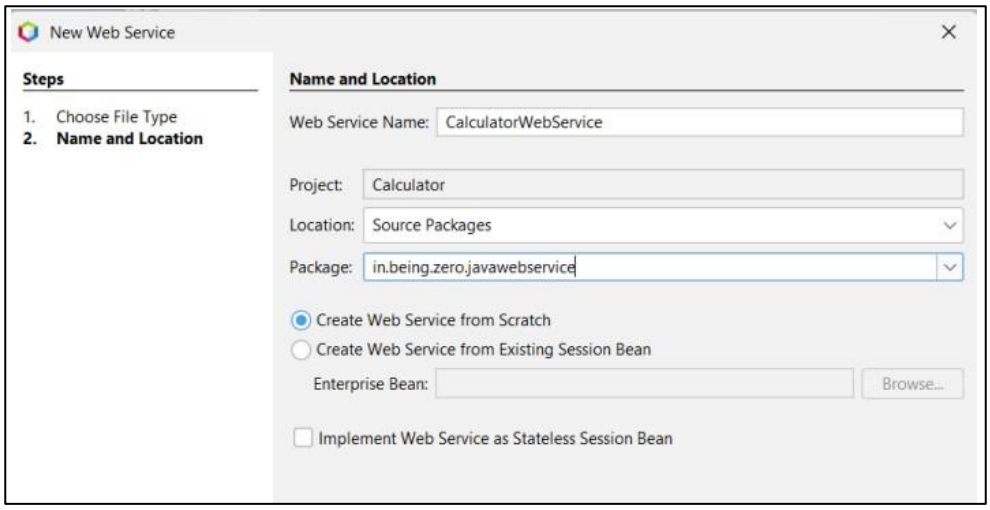




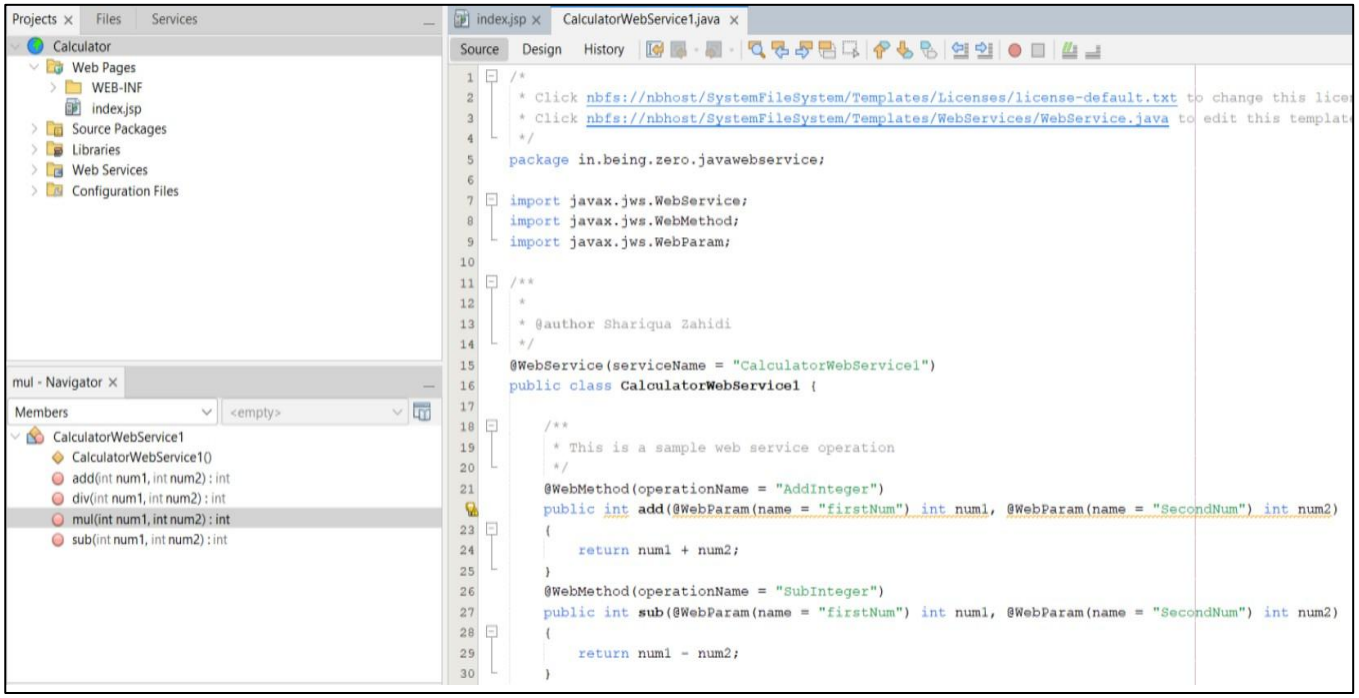
4. Right click on the Project (i.e. Calculator), click New → Web Service



5. Enter web service name and package name then click on Finish



6. A new page Web service code page should be displayed. Update the code by following the below screenshot:

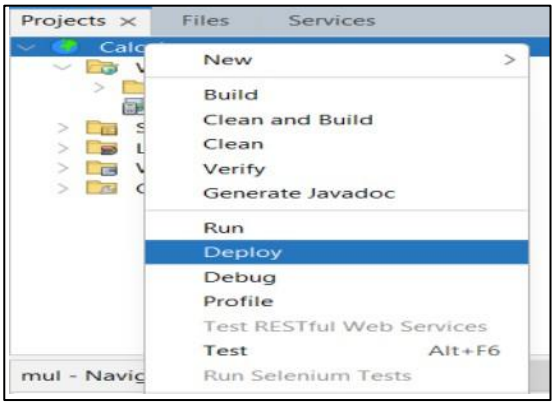


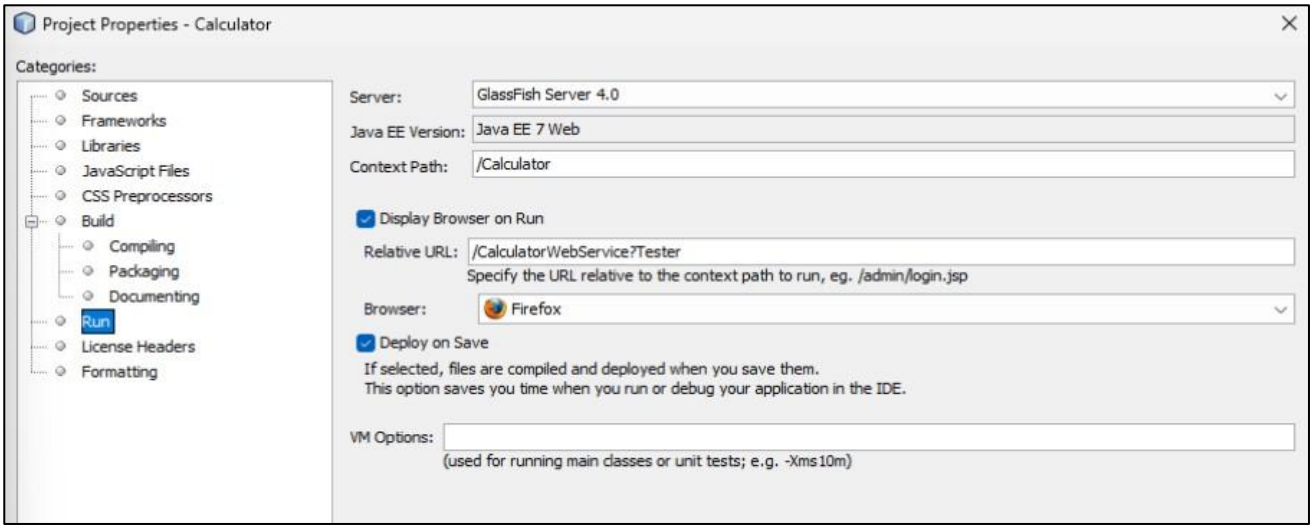
index.jsp | CalculatorWebService1.java

Source | Design | History

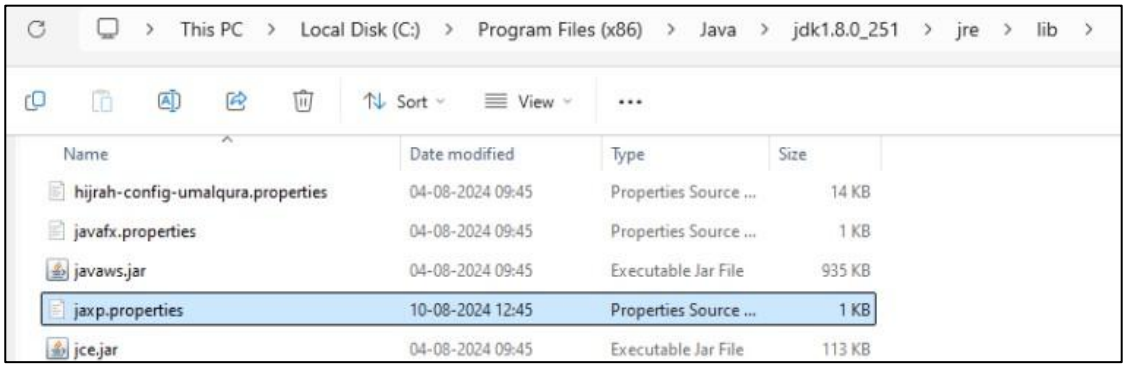
```
@WebService(serviceName = "CalculatorWebService1")
public class CalculatorWebService1 {

    /**
     * This is a sample web service operation
     */
    @WebMethod(operationName = "AddInteger")
    public int add(@WebParam(name = "firstNum") int num1, @WebParam(name = "SecondNum") int num2)
    {
        return num1 + num2;
    }
    @WebMethod(operationName = "SubInteger")
    public int sub(@WebParam(name = "firstNum") int num1, @WebParam(name = "SecondNum") int num2)
    {
        return num1 - num2;
    }
    @WebMethod(operationName = "MulInteger")
    public int mul(@WebParam(name = "firstNum") int num1, @WebParam(name = "SecondNum") int num2)
    {
        return num1 * num2;
    }
    @WebMethod(operationName = "DivInteger")
    public int div(@WebParam(name = "firstNum") int num1, @WebParam(name = "SecondNum") int num2)
    {
        return num1 / num2;
    }
}
```

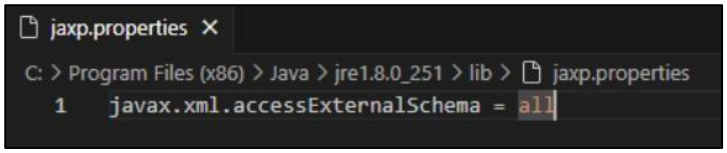
7. Again, right click on project and click on deploy option.



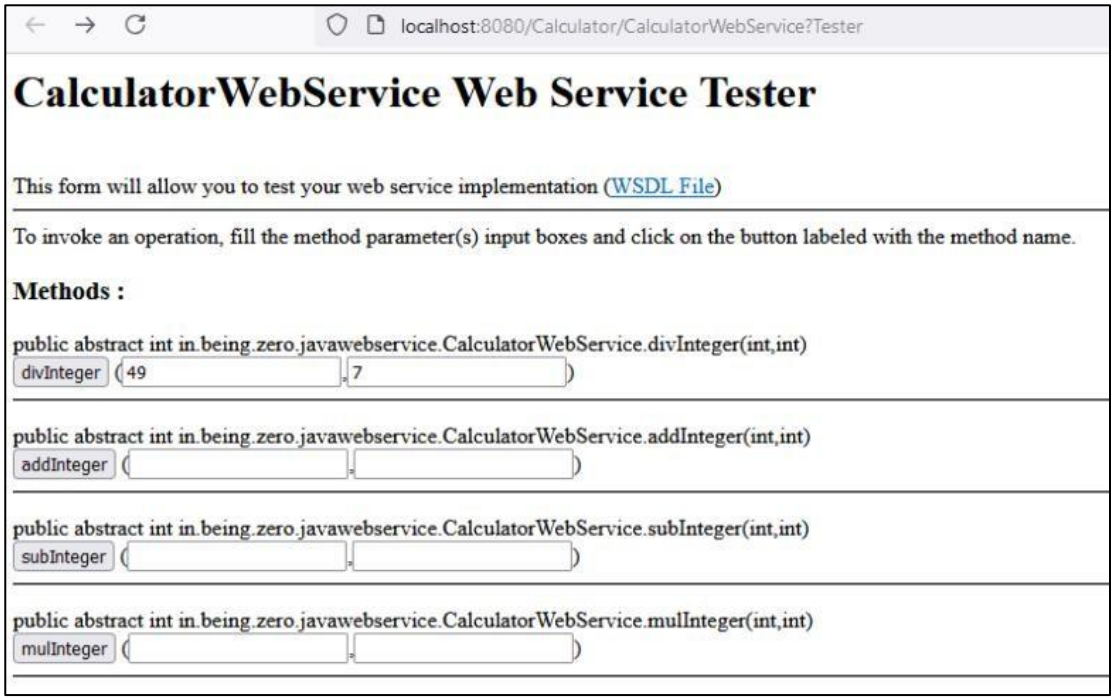


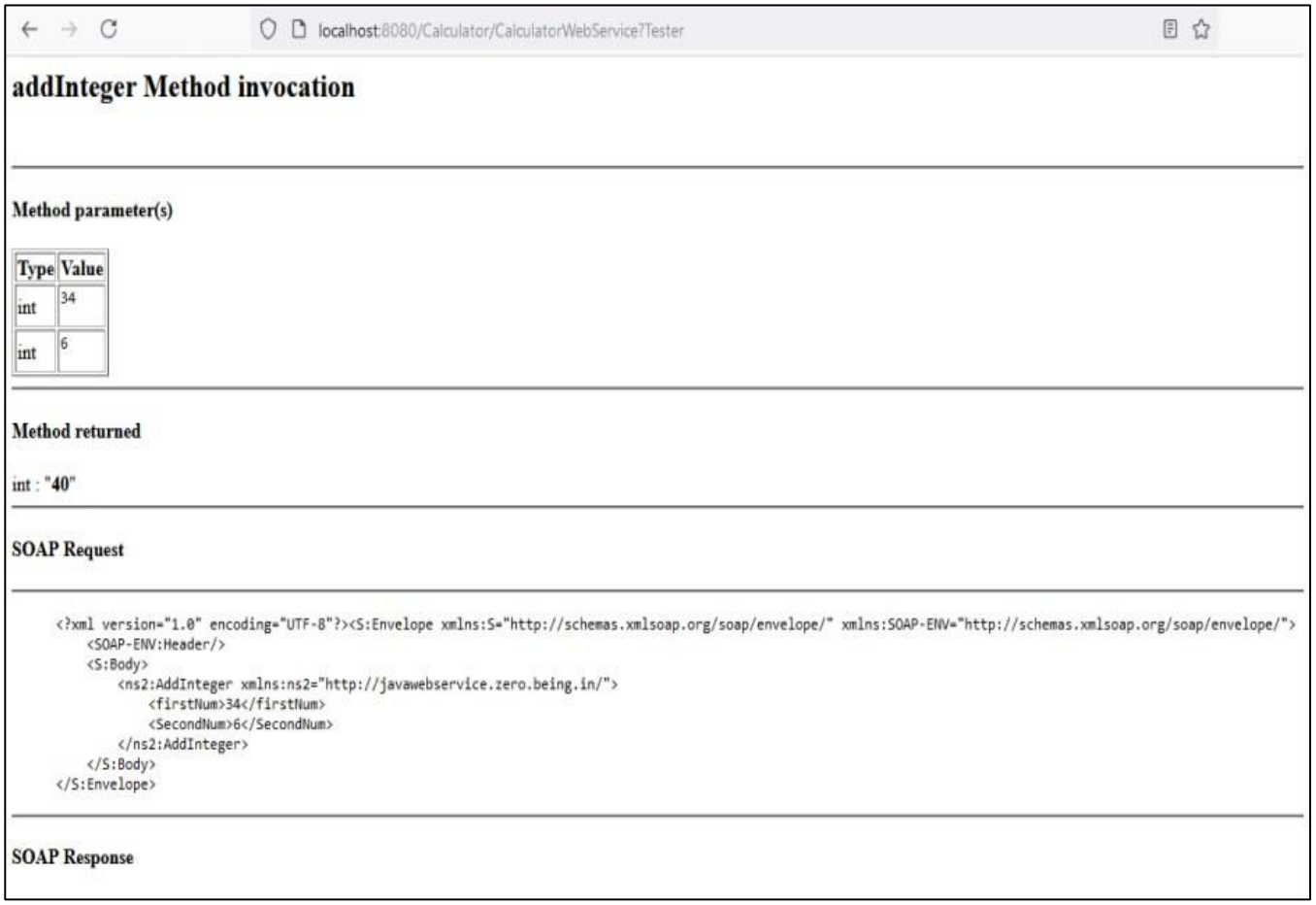
9. If deployment fails, then do the below steps





10. Again, deploy the java file




localhost:8080/Calculator/CalculatorWebService?Tester
addInteger Method invocation
Method parameter(s)
Type
Value
int
34
int
6
Method returned
int : "40"
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:AddInteger xmlns:ns2="http://javawebservice.zero.being.in/">
<firstNum>34</firstNum>
<SecondNum>6</SecondNum>
</ns2:AddInteger>
</S:Body>
</S:Envelope>
SOAP Response