# FACULTY OF ENGINEERING AND TECHNOLOGY

# BACHELOR OF TECHNOLOGY

# ADVANCED JAVA PROGRAMMING

# (203105317)

## V SEMESTER

## Computer Science & Engineering

## Department

# CERTIFICATE

This is to certify that

Mr./Ms **KHUSHI RANJAN .** with enrollment no. **200303105149**

. has successfully completed his/her laboratory experiments in the **ADVANCED JAVA PROGRAMMING (203105317)** from the department of **COMPUTER SCIENCE AND ENGINEERING** during the academic year **2022-23.**

Date of Submission:..... Staff In charge:.....

Head Of Department:.....

# PRACTICAL 1

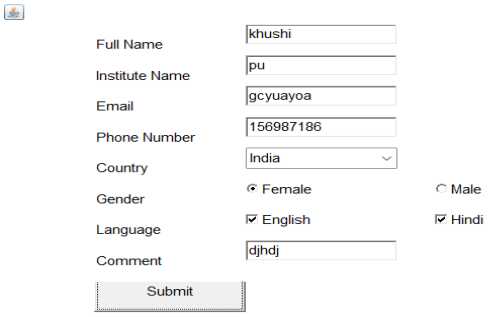
AIM:write a program to create a registration form for the student using AWT.

INPUT:

```
import java.awt.*;
import javax.swing.*;
public class RegistrationForm extends Frame
{
        RegistrationForm()
   {
      Frame fm = new Frame();
      Label lb1 = new Label("Full Name");
        lb1.setBounds(80,30,120,40);
      Label lb2= new Label("Institute Name");
        lb2.setBounds(80,60,120,40);
      Label lb3= new Label("Email");
        lb3.setBounds(80,90,120,40);
      Label lb4= new Label("Phone Number");
        lb4.setBounds(80,120,120,40);
        Label lb5= new Label("Country");
        lb5.setBounds(80,150,120,40);
        Label lb6= new Label("Gender");
        lb6.setBounds(80,180,120,40);
        Label lb7= new Label("Language");
        lb7.setBounds(80,210,120,40);
```

```
   Label lb8= new Label("Comment");
   lb8.setBounds(80,240,120,40);
fm.add(lb1);
fm.add(lb2);
fm.add(lb3);
fm.add(lb4);
fm.add(lb5);
   fm.add(lb6);
   fm.add(lb7);
   fm.add(lb8);


TextField t1,t2,t3,t4,t8;
t1=new TextField();
   t1.setBounds(200,30,120,20);
t2=new TextField();
   t2.setBounds(200,60,120,20);
t3=new TextField();
   t3.setBounds(200,90,120,20);
t4=new TextField();
   t4.setBounds(200,120,120,20);
   t8=new TextField();
   t8.setBounds(200,240,120,20);
fm.add(t1);
fm.add(t2);
fm.add(t3);
fm.add(t4);
   fm.add(t8);
   Choice c = new Choice();
   c.setBounds(200,150,120,20);
c.add("India");
c.add("Nepal");
c.add("Canada");
   c.add("Paris");
```

```
        fm.add(c);
        CheckboxGroup cbg = new CheckboxGroup();
   Checkbox checkBox1 = new Checkbox("Female", cbg, false);
            checkBox1.setBounds(200,180,120,20);
       fm.add(checkBox1);
       Checkbox checkBox2 = new Checkbox("Male", cbg, false);
      checkBox2.setBounds(350,180,120,20);
       fm.add(checkBox2);
        Checkbox checkbox1=new Checkbox("English");
      checkbox1.setBounds(200,210,120,20);
        fm.add(checkbox1);
      Checkbox checkbox2=new Checkbox("Hindi");
      checkbox2.setBounds(350,210,120,20);
        fm.add(checkbox2);
        Button b = new Button("Submit");
            b.setBounds(80,270,120,40);
            fm.add(b);
      fm.setLayout(null);
      fm.setVisible(true);
       fm.setSize(800,800);
   }

   public static void main(String args[]) {
      RegistrationForm rf = new RegistrationForm();
   }
}
```




# PRACTICAL 2

AIM:write a program to create a calculator using swing.

INPUT:

For the ADD button:

```
   private void jButton1ActionPerformed(java.awt.event.ActionEvent
evt) {
        // TODO add your handling code here:
        int a,b,c;
        a=Integer.parseInt(jTextField1.getText());
         b=Integer.parseInt(jTextField2.getText());
         c=a+b;
         jTextField3.setText(""+c);
   }
```

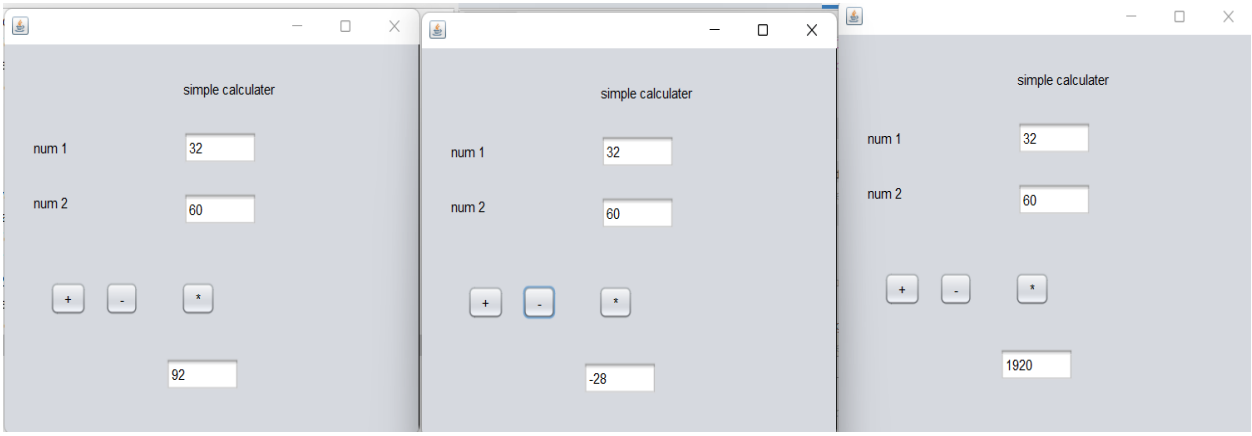
For the substitution button :

```
   private void jButton2ActionPerformed(java.awt.event.ActionEvent
evt) {
       // TODO add your handling code here:
            int a,b,c;
       a=Integer.parseInt(jTextField1.getText());
        b=Integer.parseInt(jTextField2.getText());
        c=a-b;
        jTextField3.setText(""+c);
   }
```

For the multiplying button :

```
   private void jButton3ActionPerformed(java.awt.event.ActionEvent
evt) {
       // TODO add your handling code here:
       int a,b,c;
       a=Integer.parseInt(jTextField1.getText());
       b=Integer.parseInt(jTextField2.getText());
        c=a*b;
        jTextField3.setText(""+c);
   }
```

OUTPUT:




# PRACTICAL 3

AIM:implement JDBC by connecting with the database and execute prepared statements.

INPUT:

```
import java.sql.*;
import java.sql.DriverManager;
import java.sql.Connection;
```

For insert button:

```
 private void T4ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
      try {
```

```
        int a=Integer.parseInt(jTextField1.getText());
        String n=jTextField2.getText();
         String p=jTextField3.getText();
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection
conn=DriverManager.getConnection("jdbc:mysql://localhost:3306/
student","root","Khushivyas@3113");

        System.out.println("Established");
        String query ="insert into
khushi(ID,NAME,PASSWORD)"+"values(?,?,?)";
        PreparedStatement
preparedStmt=conn.prepareStatement(query);
        preparedStmt.setInt(1,a);
        preparedStmt.setString(2,n);
        preparedStmt.setString(3,p);
        int i=preparedStmt.executeUpdate();
        T4.setText(i+"record inserted");
        conn.close();



    }
    catch(Exception e){
      System.out.println(e);
    }

  }
```

For clear button:

```
 private void jButton2ActionPerformed(java.awt.event.ActionEvent
evt) {
        // TODO add your handling code here:
            jTextField1.setText(" ");
     jTextField2.setText("");
       jTextField3.setText("");

    }
```

output:

# PRACTICAL – 4

**AIM :- Implement JDBC by connecting with database and execute Callable Statement.**

**CODE :-**

```
import java.sql.DriverManager;

import java.sql.Connection;

import java.sql.CallableStatement;

import java.sql.PreparedStatement;

import java.sql.SQLException;

import static java.time.Clock.system;

public class Callablestatement extends javax.swing.JFrame {

          public Callablestatement() {

       initComponents();
```

```
        }
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
jTextField1.setText("");
jTextField2.setText("");
jTextField3.setText("");// TODO add your handling code here:
        }

        private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

        try
        {
        int a=Integer.parseInt(jTextField1.getText());
        String n=jTextField2.getText();
        String p=jTextField3.getText();
      Class.forName("com.mysql.cj.jdbc.Driver");
      System.out.println("done");
        String query="insert into userdetail(id,name,password)" + "values(?,?,?)";
      Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/ritik","root","");
      System.out.println("Connection Established");
      CallableStatement stmt=con.prepareCall("{call INSERT_userdetail(?,?,?)}");
      stmt.setInt(1,a);
      stmt.setString(2,n);
      stmt.setString(3,p);
```

```
            int i=stmt.executeUpdate();

        System.out.println(i+" records inserted");

        con.close();

            }

    catch(SQLException | ClassNotFoundException e)

            {

        System.out.println(e);

            }

            }


            private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {


int a = Integer.parseInt(jTextField1.getText());

            String n=jTextField2.getText();

            String p=jTextField3.getText();

            try {

            Class.forName("com.mysql.cj.jdbc.Driver");

            Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/ritik","root","");

//            String query="update userdetail SET password=? WHERE id=?";

            CallableStatement stmt=conn.prepareCall("{call UPDATE_userdetail(?,?)}");




            stmt.setString(1, p);
```

```
        stmt.setInt(2, a);

            int i = stmt.executeUpdate();

        jLabel5.setText(i+"record update");

        conn.close();

        }

    catch(Exception e)

        {

        System.out.println(e);

        }

        }


        private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {


        String n=jTextField2.getText();


        try {

        Class.forName("com.mysql.cj.jdbc.Driver");

        Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/ritik","root","");

//          String query="update userdetail SET password=? WHERE id=?";

        CallableStatement stmt=conn.prepareCall("{call DELETE_userdetail(?)}");




        stmt.setString(1, n);
```

```
        int i = stmt.executeUpdate();
        jLabel5.setText(i+"record DELETED");
        conn.close();
        }
  catch(Exception e)
        {
        System.out.println(e);
        }
        }
        public static void main(String args[]) {
      java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new Callablestatement().setVisible(true);
        }
        });
        }
```

## Output:

SELECT * FROM userdetail ... ×

Max. rows: 100 Fetched Rows: 3

| # | id | name | password |
|---|---|---|---|
| 1 | 1 | abc | abc |
| 2 | 2 | cgtyviy | abc929 |
| 3 | 3 | cdsc | cgtyviy 929ertyhui |

SELECT * FROM userdetail ... ×

Max. rows: 100 Fetched Rows: 3

| # | id | name | password |
|---|---|---|---|
| 1 | 1 | abc | iop |
| 2 | 2 | cgtyviy | abc929 |
| 3 | 3 | cdsc | abc929ertyhui |

SELECT * FROM userdetail ... ×

Max. rows: 100 Fetched Rows: 1

| # | id | name | password |
|---|---|---|---|
| 1 | 1 | abc | iop |

# PRACTICAL 5

AIM:implement chat application using java.net.

INPUT:

From client side :

```
import java.net.*;
import java.io.*;

public class client {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here
        try{
            InetAddress ip=InetAddress.getLocalHost();
            Socket s=new Socket(ip,4021);

                BufferedReader br=new BufferedReader(new
InputStreamReader(s.getInputStream()));
                 BufferedReader kb=new BufferedReader(new
InputStreamReader(System.in));
```

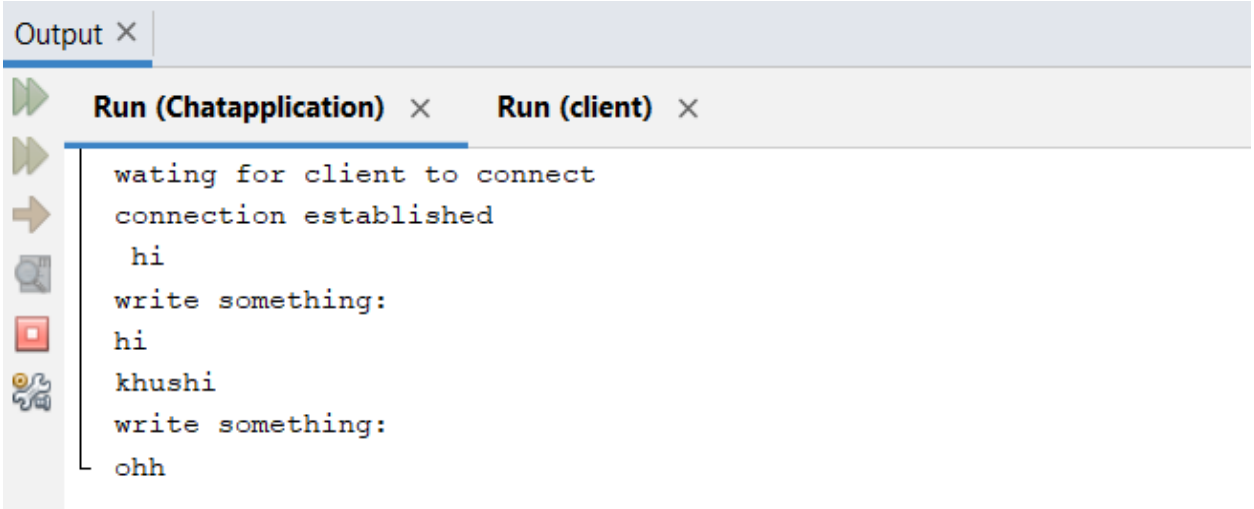
```
            PrintStream ps= new
PrintStream(s.getOutputStream());

            System.out.println("write something");
            String str=kb.readLine();

            String str1;
            while(!(str.equals("exit"))){
               ps.println(str);
               str1=br.readLine();
               System.out.println(str1);
               System.out.println("write something");
               str=kb.readLine();

            }
            ps.close();
            br.close();
            kb.close();
            s.close();
            System.out.println("client program ended");
      }
      catch(IOException e){
            System.out.println(e);
      }
   }

}
```

From server side :

```
import java.net.*;
import java.io.*;

public class Chatapplication {

   public static void main(String[] args) {
         try{
         ServerSocket ss=new ServerSocket(4021);
         System.out.println("wating for client to connect ");
         Socket s=ss.accept();
            System.out.println("connection established");

            BufferedReader br=new BufferedReader(new
InputStreamReader(s.getInputStream()));
             BufferedReader kb=new BufferedReader(new
InputStreamReader(System.in));
             PrintStream ps= new
PrintStream(s.getOutputStream());

             String str,str1;
             str=br.readLine();
             while(str!=null)
             {
                System.out.println(str);
                System.out.println("write something:");
                str1=kb.readLine();
                ps.println(str1);

                str=br.readLine();
             }
```

```
            ps.close();
            br.close();
            kb.close();
            s.close();
            ss.close();



        }
            catch(IOException e){
                System.out.println(e);

            }
    }
}
```

OUTPUT:



# PRACTICAL 6

AIM:implement any sorting algorithm using TCP/UDP on server application and give input on client side and client should sort output from server and  display sorted on input side.

## INPUT:

SERVER.java

```
import java.net.*;

import java.io.*;

public class Server {

        public static void main(String[] args) {

        try

        {

        ServerSocket s1=new ServerSocket(12345);

        System.out.println("Server Started");

        Socket s=s1.accept();
```

```
PrintWriter p=new PrintWriter(s.getOutputStream());
BufferedReader in=new BufferedReader(new InputStreamReader(s.getInputStream()));
String num=in.readLine();
int n =Integer.parseInt(num);
System.out.println("Client want to sort"+n+"numbers");
String sarr[]=new String[n];
int arr[]=new int[n];
int swap,c,d;
System.out.println("received numbers::\n");
for(int i=0;i<n;i++)
{
   sarr[i]=in.readLine();
   arr[i]=Integer.parseInt(sarr[i]);
   System.out.println("no."+i+"="+arr[i]);
}
for(c=0;c<(n-1);c++)
{
   for(d=0;d<n-c-1;d++)
   {
      if(arr[d]>arr[d+1])
      {
         swap=arr[d];
         arr[d]=arr[d+1];
```

```
                arr[d+1]=swap;
            }
        }
    }
    System.out.println("\nSorted list of numbers");
    String sendarr=new String();
    for(c=0;c<n;c++)
    {
        sendarr+="\nnum("+c+")="+arr[c];
    }
    System.out.println(sendarr);
    p.print(sendarr);
    p.flush();
    s.close();
    }
  catch(Exception e)
    {
    System.out.println(e);
    }
    }

}
```

CLIENT.java

```
import java.net.*;
import java.io.*;
public class Client {
        public static void main(String[] args) {
        try
        {
        Socket s=new Socket("localhost",12345);
        PrintWriter p=new PrintWriter(s.getOutputStream());
        BufferedReader in=new BufferedReader(new InputStreamReader(s.getInputStream()));
        BufferedReader ink=new BufferedReader(new InputStreamReader(System.in));
        System.out.println("How many numbers to sort?");
        int num=Integer.parseInt(ink.readLine());
        p.println(num);
        p.flush();
        System.out.println("Enter"+num+"numbers to sort:");
        String sarr[]=new String[num];
        for(int i=0;i<num;i++)
        {
           System.out.print("no."+i+"=");
           sarr[i]=ink.readLine();
           p.println(sarr[i]);
```

```
            p.flush();
        }
        String res;
        System.out.println("\nSorted array::\n");
        while((res=in.readLine())!=null)
        {
            System.out.println(res);
        }
        s.close();
        }
    catch(Exception e)
        {
        System.out.println(e);
        }
        }

}
```

## Output:

```
Output ×
Practical6 (run) ×    Practical6 (run) #2 ×
run:
Server Started
Client want to sort5numbers
received numbers::

no.0=1
no.1=8
no.2=5
no.3=6
no.4=2

Sorted list of numbers

num(0)=1
num(1)=2
num(2)=5
num(3)=6
num(4)=8
BUILD SUCCESSFUL (total time: 23 seconds)
```

```
Output ×
Practical6 (run) ×    Practical6 (run) #2 ×
run:
How many numbers to sort?
5
Enter5numbers to sort:
no.0=1
no.1=8
no.2=5
no.3=6
no.4=2

Sorted array::


num(0)=1
num(1)=2
num(2)=5
num(3)=6
num(4)=8
BUILD SUCCESSFUL (total time: 17 seconds)
```

# PRACTICAL – 7

**AIM :- Implement Student information system using JDBC and RMI.**

# PRACTICAL – 8

**AIM :- Call remote procedure from a jvm to another jvm by implementing RMI.**

# PRACTICAL 9

AIM:Make a simple calculator using RMI.

Input:

(Cal.java): -

```
importjava.rmi.*;

public interface Cal extends Remote

{

	publicint add(inta,int b) throws RemoteException;

	publicint sub(inta,int b) throws RemoteException;

	publicintmul(inta,int b) throws RemoteException;

	publicint div(inta,int b) throws RemoteException;

}
```

(democal.java): -

```
importjava.rmi.*;

importjava.rmi.server.*;

public class democal extends UnicastRemoteObject implements Cal

{

	democal()throws RemoteException

	{

			super();

	}

	publicint add(inta,int b)

	{

			int c;

			c=a+b;
```

```
                return c;

        }

        publicint sub(inta,int b)

        {

                int c;

                c=a-b;

                return c;

        }

        publicintmul(inta,int b)

        {

                int c;

                c=a*b;

                return c;

        }

        publicint div(inta,int b)

        {

                int c;

                c=a/b;

                return c;

        }

}
```

(servercal.java): -

```
importjava.rmi.*;

importjava.rmi.registry.*;

public class servercal

{

	public static void main(String args[])

	{

			try

			{

				Cal stub= new democal();

				Naming.rebind("rmi://localhost:5000/ritul",stub);

			}

			catch(Exception e)

			{

				System.out.println(e);

			}

	}

}
```

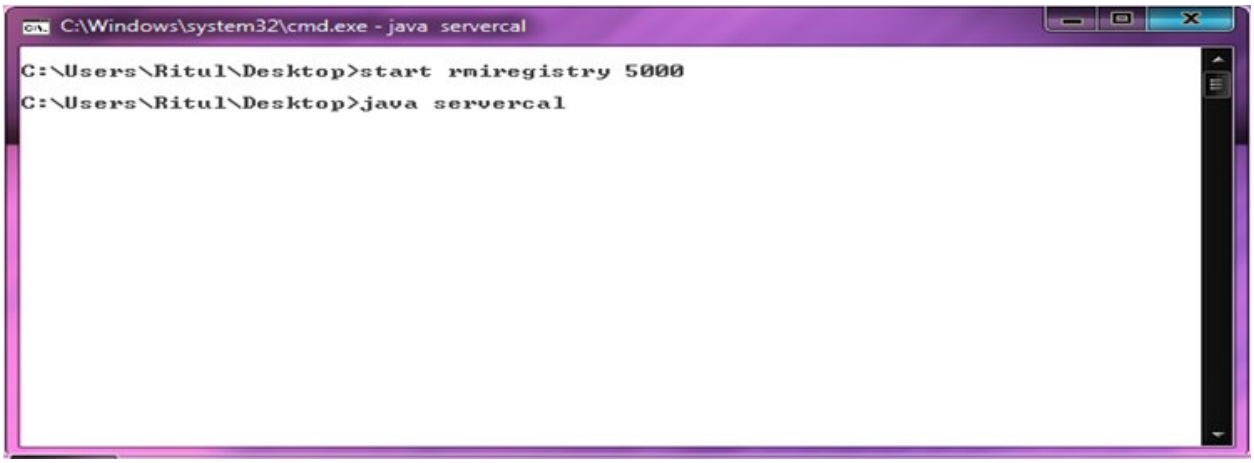
(clientcal.java): -

```
importjava.rmi.*;

public class clientcal
```

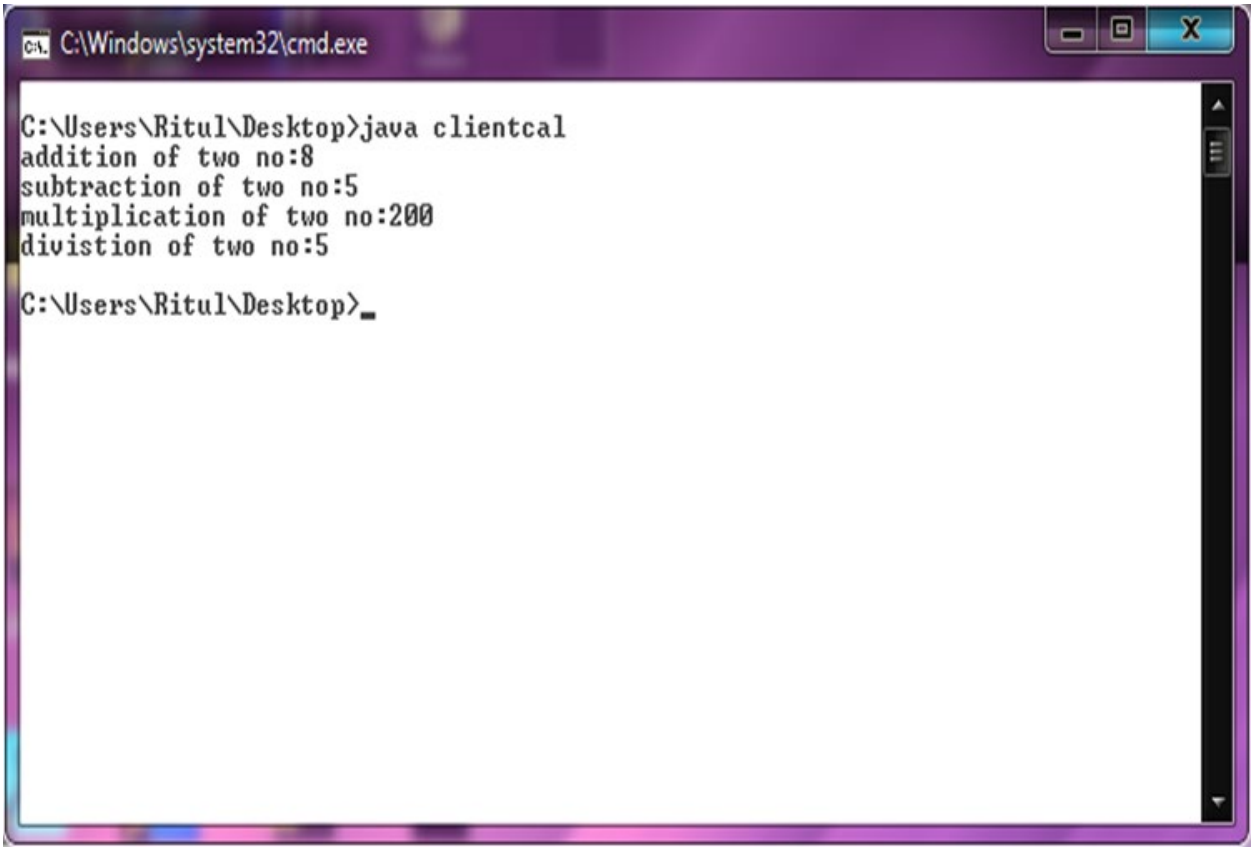
```
{
        public static void main(String args[])
        {
                try
                {
                        Cal stub=(Cal)Naming.lookup("rmi://localhost:5000/ritul");
                        System.out.println ("addition of two no:"+(stub.add(4,4)));
                        System.out.println ("subtraction of two no:"+(stub.sub(10,5)));
                        System.out.println ("multiplication of two no:"+(stub.mul(10,20)));
                        System.out.println ("divistion of two no:"+(stub.div(25,5)));
                }
                catch(Exception e)
                {
                        System.out.println(e);
                }
        }
}
```

Output: - (Windows)

Server Side


C:\Windows\system32\cmd.exe - java servercal
C:\Users\Ritul\Desktop>start rmiregistry 5000
C:\Users\Ritul\Desktop>java servercal

Client Side


C:\Windows\system32\cmd.exe
C:\Users\Ritul\Desktop>java clientcal
addition of two no:8
subtraction of two no:5
multiplication of two no:200
divistion of two no:5
C:\Users\Ritul\Desktop>

# PRACTICAL 10

AIM:Study the functionalities of Eclipse/NetBeans and Connect to the Glassfish / Apache server.

# PRACTICAL 11

AIM:Implement a simple Servlet application. Create directory structure, create references for web containers, create necessary web.xml and other config files and execute.

# PRACTICAL 12

AIM:Create registration form of student using Servlet & JDBC.

First.java

```
package servletdemo1;


import java.io.IOException;

import java.io.PrintWriter;

import java.sql.Connection;

import java.sql.DriverManager;


import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;
```

```
import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;


/**

 * Servlet implementation class First

 */

@WebServlet("/First")

public class First extends HttpServlet {

	private static final long serialVersionUID = 1L;


	/**

	* Default constructor.

	*/

	public First() {

	// TODO Auto-generated constructor stub

	}


	/**

	 * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

	 */

	protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {

				// TODO Auto-generated method stub

				response.getWriter().append("Served at: ").append(request.getContextPath());

				PrintWriter pw=response.getWriter();

				String name=request.getParameter("name");
```

```
                String rollno=request.getParameter("rollno");

                String mobile_no=request.getParameter("mobile_no");


                //pw.println(name);

                try

                {

                        kk obj= new kk();

                        obj.demo(name,Integer.parseInt(rollno),Integer.parseInt(mobile_no));

                        pw.println("record inserted successfully");

                        //pw.println(str);

                }catch(Exception e)

                {

                        pw.println(e.getMessage());

                }

        }


        /**

         * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

         */

        protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {

                // TODO Auto-generated method stub

                doGet(request, response);

        }


}
```

kk.java

```
package servletdemo1;

import java.beans.Statement;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;

public class kk {
	public void demo(String name,int enroll,int mobileno) {
		try {
			Class.forName("oracle.jdbc.driver.OracleDriver");
			Connection con =
DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:orcl","system","mobile1");
			PreparedStatement ps = con.prepareStatement("insert into student
values(?,?,?)");
			ps.setString(1, name);
			ps.setInt(2, enroll);
			ps.setInt(3, mobileno);
			ps.executeUpdate();
			con.close();
			//return "Connection established successfully";
		} catch (Exception e) {
			//return "Connection Failed";
```

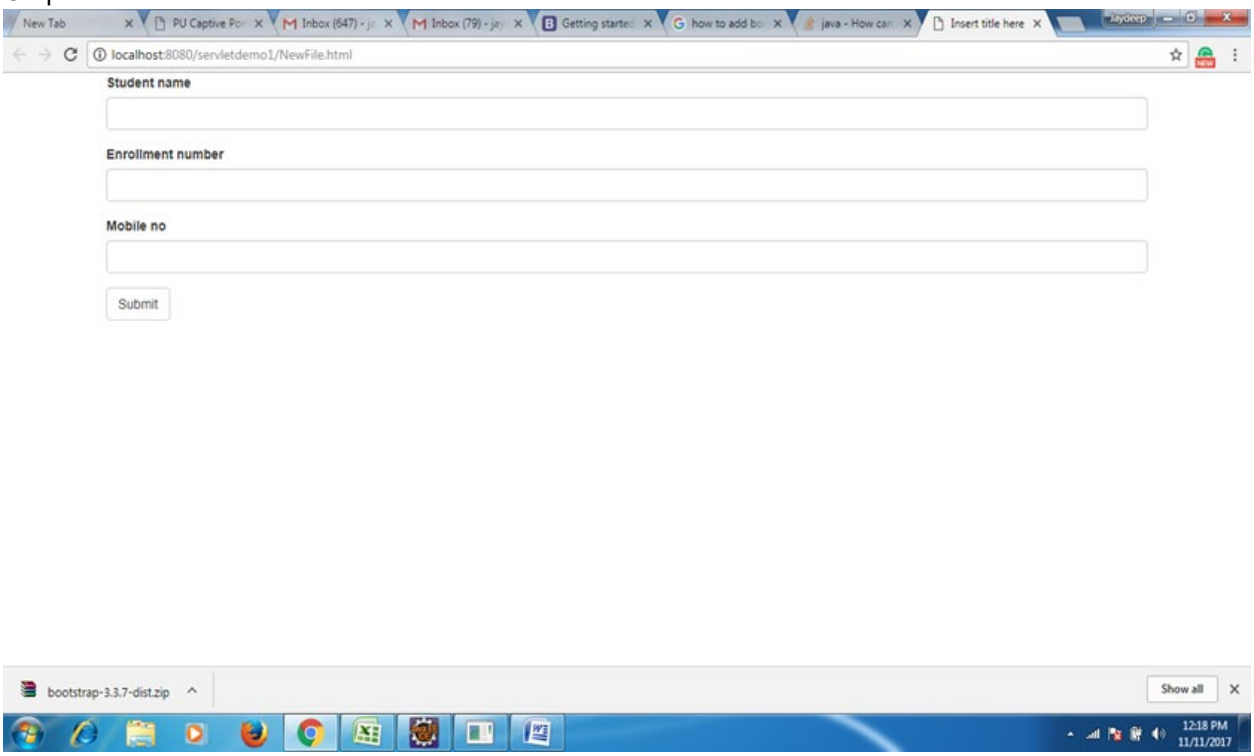
```
                    // TODO: handle exception
                }
        }
}
```

NewFile.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
<link href="bootstrap/css/bootstrap.min.css" rel="stylesheet" type="text/css" />
<script type="text/javascript" src="bootstrap/js/bootstrap.min.js"></script>
</head>
<body>
<div class="container">

<form role="form" action="First" method="get">
<div class="form-group">
<label>Student name</label>
<input type="text" class="form-control" name="name">
</div>
<div class="form-group">
<label>Enrollment number</label>
```

```
<input type="number" class="form-control" name="rollno">

</div>

<div class="form-group">

<label>Mobile no</label>

<input type="text" class="form-control" name="mobile_no">

</div>



<button type="submit" class="btn btn-default">Submit</button>

</form>

</div>

</body>

</html>
```

Output:




# PRACTICAL 13

AIM:Create a JSP page that is a student registration form. Perform server side validations using JSP.

NewFile.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
        pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
```

```
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<div class="container">


<form role="form" action="Validation" method="get">

<div class="form-group">

<label>Student name</label>

<input type="text" class="form-control" name="name">

</div>

<div class="form-group">

<label>Enrollment number</label>

<input type="number" class="form-control" name="rollno">

</div>

<div class="form-group">

<label>Mobile no</label>

<input type="text" class="form-control" name="mobile_no">

</div>



<button type="submit" class="btn btn-default">Submit</button>

</form>

</div>

</body>
```

```
</html>
```

Validation.java

```
package servletdemo1;

import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 * Servlet implementation class Validation
 */
@WebServlet("/Validation")
public class Validation extends HttpServlet {
	private static final long serialVersionUID = 1L;

	/**
	* @see HttpServlet#HttpServlet()
	*/
```

```
public Validation() {

super();

// TODO Auto-generated constructor stub

}



/**

 * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

 */

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {

                    // TODO Auto-generated method stub

                    response.getWriter().append("Served at: ").append(request.getContextPath());

                    PrintWriter pw = response.getWriter();

                    String name = request.getParameter("name");

                    String rollno = request.getParameter("rollno");

                    String mobile_no=request.getParameter("mobile_no");



                    if(!(name.isEmpty() && rollno.isEmpty() && mobile_no.isEmpty()))

                    {

                              if(rollno.chars().allMatch( Character::isDigit ) &&
mobile_no.chars().allMatch( Character::isDigit ))

                              {

                                        pw.println("Valid Input");

}

                              else

                              {
```

```
                                        pw.println("Enter numeric value in 2nd and 3rd textbox");
                                }
                        }
                        else
                        {
                                pw.println("Enter value in all the textboxs");
                        }
        }

        /**
         * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
         */
        protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
                        // TODO Auto-generated method stub
                        doGet(request, response);
        }

}
```

# PRACTICAL 14

AIM:Create a custom tag using JSP tag extension / library.

*File: MyTagHandler.java*

```
package com.javatpoint.sonoo;
```

```
import java.util.Calendar;

import javax.servlet.jsp.JspException;

import javax.servlet.jsp.JspWriter;

import javax.servlet.jsp.tagext.TagSupport;

public class MyTagHandler extends TagSupport{


public int doStartTag() throws JspException {

   JspWriter out=pageContext.getOut();//returns the instance of JspWriter

   try{

    out.print(Calendar.getInstance().getTime());//printing date and time using JspWriter

   }catch(Exception e){System.out.println(e);}

   return SKIP_BODY;//will not evaluate the body content of the tag

}

}
```

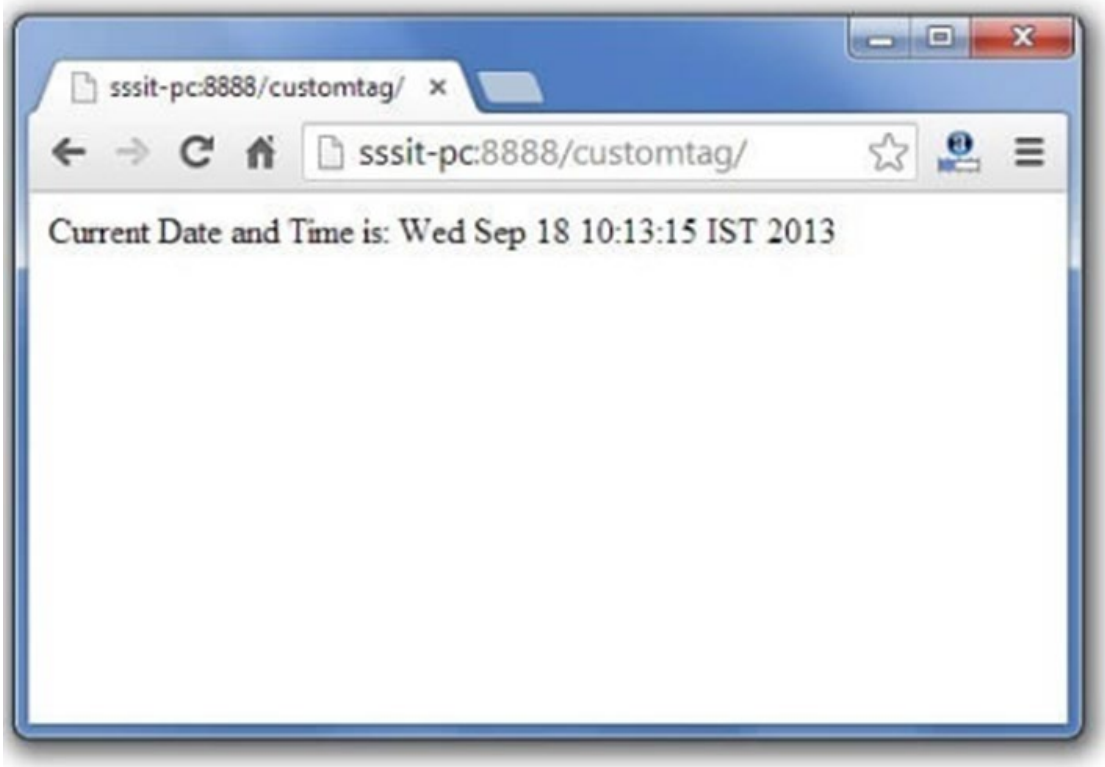
*mytags.tld*

```
<?xml version="1.0" encoding="ISO-8859-1" ?>

<!DOCTYPE taglib

      PUBLIC "-//Sun Microsystems, Inc.//DTD JSP Tag Library 1.2//EN"

   "http://java.sun.com/j2ee/dtd/web-jsptaglibrary_1_2.dtd">


<taglib>


  <tlib-version>1.0</tlib-version>

  <jsp-version>1.2</jsp-version>

  <short-name>simple</short-name>
```

```
<uri>http://tomcat.apache.org/example-taglib</uri>

<tag>
<name>today</name>
<tag-class>com.javatpoint.sonoo.MyTagHandler</tag-class>
</tag>
</taglib>
```

*index.jsp*

```
<%@ taglib uri="WEB-INF/mytags.tld" prefix="m" %>
Current Date and Time is: <m:today/>
```




# PRACTICAL 15

AIM:Create user interface of a student registration and login using JSF.

**register.xhtml**

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml"
   xmlns:h="http://java.sun.com/jsf/html"
   xmlns:f="http://java.sun.com/jsf/core">

<h:head>
   <title>Registration Page</title>
</h:head>
<h:body>
   <f:view>
         <h:form id="registerForm">
                  <table>
                           <tr>
                              <td><h:outputText value="Enter Your First Name:" /></td>
                              <td><h:inputText id="fname" value="#{user.firstName}"
                                          required="true" requiredMessage="Please enter your first name"
/></td>
                              <td><h:message for="fname" style="color:red" /></td>
                           </tr>
```

```
<tr>
   <td><h:outputText value="Enter Your Last Name:" /></td>
   <td><h:inputText id="lname" value="#{user.lastName}"
                    required="true" requiredMessage="Please enter your last name"
/></td>
   <td><h:message for="lname" style="color:red" /></td>
</tr>
<tr>
   <td><h:outputText value="Enter Your email ID:" /></td>
   <td><h:inputText id="email" value="#{user.email}"
                    required="true" requiredMessage="Please enter your email id"
/></td>
   <td><h:message for="email" style="color:red" /></td>
</tr>
<tr>
         <td><h:outputText value="Enter Password :" /></td>
   <td><h:inputSecret id="psw" value="#{user.password}"
                    required="true" requiredMessage="Please enter your password"
/></td>
   <td><h:message for="psw" style="color:red" /></td>
</tr>
<tr>
   <td />
   <td><h:commandButton value="Register" action="#{user.add}" /></td>
</tr>
<tr>
         <td><h:outputLink value="home.xhtml">Home</h:outputLink></td>
```

```
                    </tr>
                </table>
            </h:form>
    </f:view>
</h:body>
</html>
```

**success.xhtml**

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml"
    xmlns:ui="http://java.sun.com/jsf/facelets"
    xmlns:h="http://java.sun.com/jsf/html"
    xmlns:f="http://java.sun.com/jsf/core">

<h:head>
    <title>Success Page</title>
</h:head>
<h:body>
    <f:view>
            <p>Successfully logged in</p>
```

```
        <p>Hi, #{user.firstName}</p>
        <h:form>
        <p>
                <h:commandLink value="logout" action="#{user.logout}" />
            </p>
        </h:form>
    </f:view>
</h:body>
</html>
```

**unsuccess.xhtml**

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml"
    xmlns:h="http://java.sun.com/jsf/html"
    xmlns:f="http://java.sun.com/jsf/core">
<h:head>
    <title>Unsuccess Page</title>
</h:head>
<h:body>
    <f:view>
```

```
        <p>There is an error in signing up. See Server Console for error.</p>

        <h:outputLink value="register.xhtml">Back</h:outputLink>

    </f:view>

</h:body>

</html>
```

**User.java**

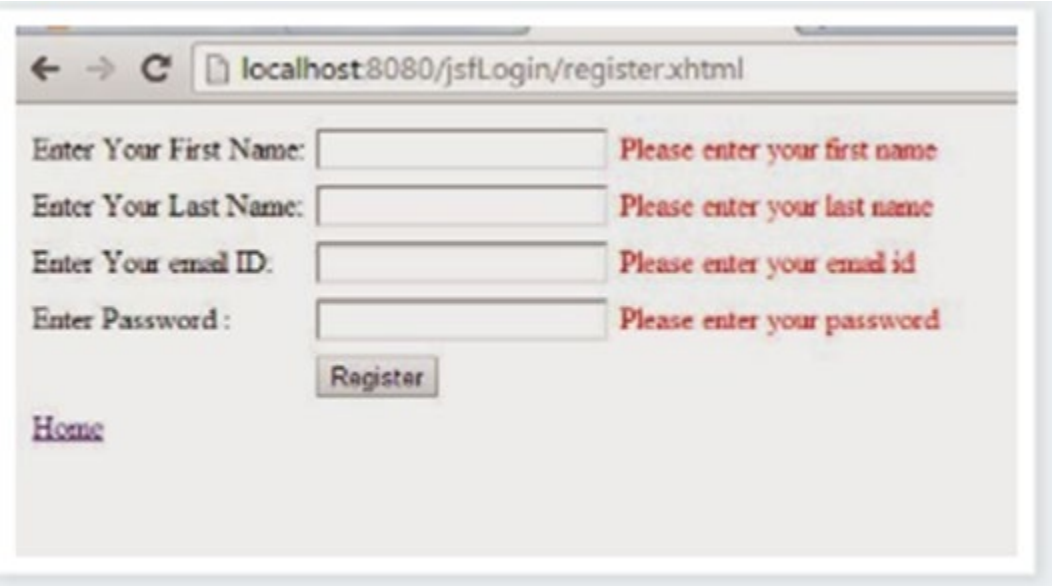
```
package com.amzi.beans;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

import javax.faces.bean.ManagedBean;
import javax.faces.bean.RequestScoped;
import javax.faces.context.FacesContext;
import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;
import javax.sql.DataSource;

@ManagedBean(name = "user")
@RequestScoped
public class User {

        private String firstName;
        private String lastName;
        private String email;
        private String password;
        private String dbPassword;
        private String dbName;
        DataSource ds;

        public User() {
                try {
                        Context ctx = new InitialContext();
                        ds = (DataSource) ctx.lookup("java:comp/env/jdbc/database");
                } catch (NamingException e) {
                        e.printStackTrace();
```

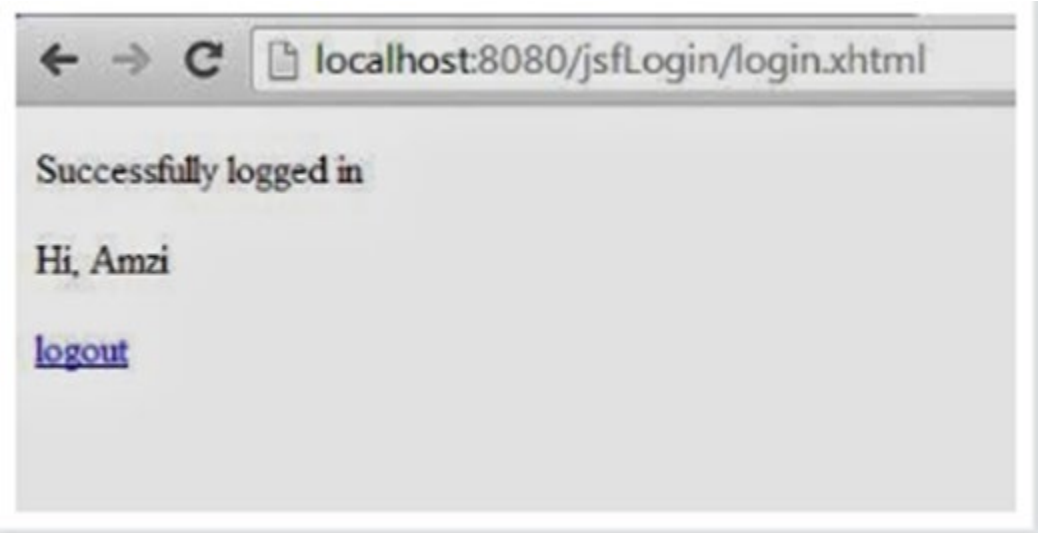
```
            }
        }

        public String getDbPassword() {
            return dbPassword;
        }

        public String getDbName() {
            return dbName;
        }

        public String getFirstName() {
            return firstName;
        }

        public void setFirstName(String name) {
            this.firstName = name;
        }

        public String getLastName() {
            return lastName;
        }

        public void setLastName(String lastName) {
            this.lastName = lastName;
        }

        public String getEmail() {
            return email;
        }

        public void setEmail(String email) {
            this.email = email;
        }

        public String getPassword() {
            return password;
        }

        public void setPassword(String password) {
            this.password = password;
        }

    public String add() {

            int i = 0;
```

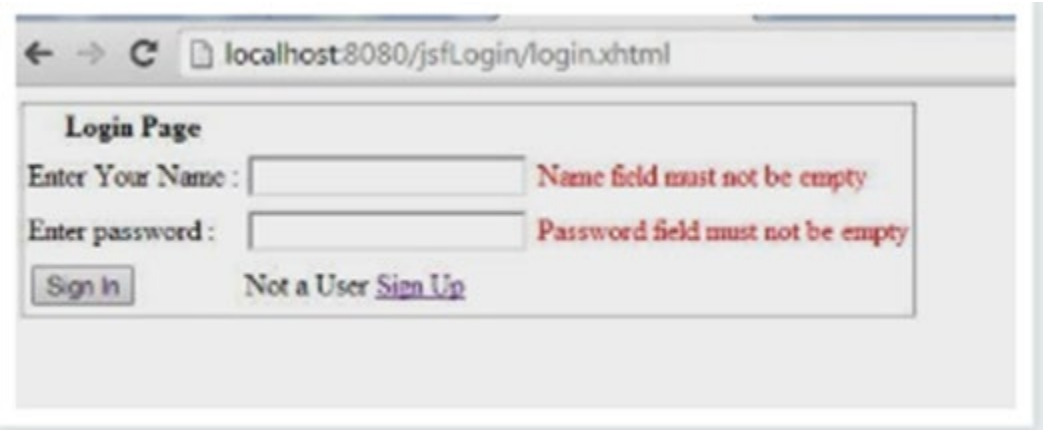
```
if (firstName != null) {
        PreparedStatement ps = null;
        Connection con = null;
        try {
                if (ds != null) {
                con = ds.getConnection();
                if (con != null) {
                        String sql = "INSERT INTO user(firstname, password, lastname,
email) VALUES(?,?,?,?)";
                        ps = con.prepareStatement(sql);
                        ps.setString(1, firstName);
                        ps.setString(2, password);
                        ps.setString(3, lastName);
                        ps.setString(4, email);
                        i = ps.executeUpdate();
                        System.out.println("Data Added Successfully");
                }
                }
        } catch (Exception e) {
           System.out.println(e);
        } finally {
                try {
                con.close();
                ps.close();
                } catch (Exception e) {
                e.printStackTrace();
```

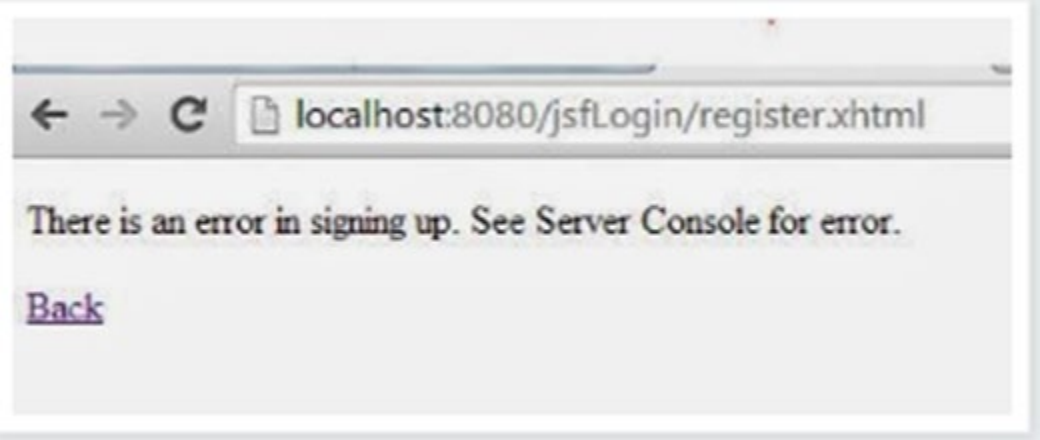
```
                        }
                }
        }
        if (i > 0) {
                return "success";
        } else
                return "unsuccess";
    }

    public void dbData(String uName) {
        if (uName != null) {
                PreparedStatement ps = null;
                Connection con = null;
                ResultSet rs = null;

                if (ds != null) {
                        try {
                        con = ds.getConnection();
                        if (con != null) {
                                String sql = "select firstname,password from user where
firstname = '"
                                        + uName + "'";
                                ps = con.prepareStatement(sql);
                                rs = ps.executeQuery();
                                rs.next();
                                dbName = rs.getString("firstname");
```

```
                    dbPassword = rs.getString("password");
                }
                } catch (SQLException sqle) {
                sqle.printStackTrace();
                }
            }
        }
    }

    public String login() {
        dbData(firstName);
        if (firstName.equals(dbName) && password.equals(dbPassword)) {
            return "output";
        } else
            return "invalid";
    }

    public void logout() {
      FacesContext.getCurrentInstance().getExternalContext()
                .invalidateSession();
        FacesContext.getCurrentInstance()
                .getApplication().getNavigationHandler()
                .handleNavigation(FacesContext.getCurrentInstance(), null, "/login.xhtml");
    }
}
```


localhost:8080/jsfLogin/register.xhtml
Enter Your First Name:
Please enter your first name
Enter Your Last Name:
Please enter your last name
Enter Your email ID:
Please enter your email id
Enter Password :
Please enter your password
Register
Home


localhost:8080/jsfLogin/login.xhtml
Successfully logged in
Hi, Amzi
logout


localhost:8080/jsfLogin/login.xhtml
Login Page
Enter Your Name :
Name field must not be empty
Enter password :
Password field must not be empty
Sign In
Not a User Sign Up




# PRACTICAL 16

AIM:Transfer all the Business Logic to the EJB of practical 10.

*AdderImplRemote.java*

```
package com.javatpoint;

import javax.ejb.Remote;


@Remote

public interface AdderImplRemote {

int add(int a,int b);

}
```

*AdderImpl.java*

```
package com.javatpoint;

import javax.ejb.Stateless;


@Stateless(mappedName="st1")

public class AdderImpl implements AdderImplRemote {

  public int add(int a,int b){

      return a+b;
```

```
    }
```

*AdderImpl.java*

```
package com.javatpoint;

import javax.naming.Context;

import javax.naming.InitialContext;


public class Test {

public static void main(String[] args)throws Exception {

    Context context=new InitialContext();

    AdderImplRemote remote=(AdderImplRemote)context.lookup("st1");

    System.out.println(remote.add(32,32));

}

}
```

Output:64

# PRACTICAL 17

AIM:Create database and Implement JPA to provide persistence to practical 10.

**Employ.java**

```
package mrbool.eclipselink.entity;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Table;
```

```
@Entity
@Table
public class Employ {

  @Id
  @GeneratedValue(strategy = GenerationType.AUTO)
  private int id;
  private String name;
  private double sal;
  private String deg;
  public Employ(int id, String name, double sal, String deg) {
        super( );
        this.id = id;
        this.name = name;
        this.sal = sal;
        this.deg = deg;
  }
  public Employ( ) {
        super();
  }
  public int getid( ) {
        return id;
  }
  public void setid(int eid) {
        this.id = id;
```

```
  }
  public String getname( ) {
        return name;
  }
  public void setname(String name) {
        this.name = name;
  }
  public double getSal ( ) {
        return sal;
  }

  public void setSal (double sal) {
        this.sal = sal;
  }
  public String getDeg( ) {
        return deg;
  }
  public void setDeg(String deg) {
        this.deg = deg;
  }
  @Override
  public String toString() {
        return "Employee [Id=" + id + ", Name=" + name + ", Salary=" + sal + ", deg=" + deg + "]";
  }
}
```

```
create database jpadb

use jpadb
```

**Persist.xml**

```
<?xml version="1.0" encoding="UTF-8"?>

<persistence version="2.0" xmlns="http://java.sun.com/xml/ns/persistence"

  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

  xsi:schemaLocation="http://java.sun.com/xml/ns/persistence

  http://java.sun.com/xml/ns/persistence/persistence_2_0.xsd">

  <persistence-unit name="Eclipselink_JPA" transaction-type="RESOURCE_LOCAL">

        <class> eclipselink.entity.Employ</class>


        <properties>

        <property name="javax.persistence.jdbc.url" value="jdbc:mysql://localhost:3306/jpadb"/>

        <property name="javax.persistence.jdbc.user" value="root"/>

        <property name="javax.persistence.jdbc.password" value="root"/>

        <property name="javax.persistence.jdbc.driver" value="com.mysql.jdbc.Driver"/>

        <property name="eclipselink.logging.level" value="FINE"/>

        <property name="eclipselink.ddl-generation" value="create-tables"/>

        </properties>

  </persistence-unit>

</persistence>
```

**CreateEmploy.java**

```
package mrbool.eclipselink.service;
```

```
import javax.persistence.EntityManager;

import javax.persistence.EntityManagerFactory;

import javax.persistence.Persist;

import eclipselink.entity.Employ;

public class CreateEmploy {

  public static void main( String[ ] args ) {

        EntityManagerFactory emfactory = Persist.createEntityManagerFactory( "Eclipselink_JPA" );

        EntityManager entitymanager = emfactory.createEntityManager( );

        entitymanager.getTransaction( ).begin( );


        Employ employee = new Employ( );

        employee.setid( 101 );

        employee.setname( "Ravi" );

        employee.setSalary( 60000 );

        employee.setDeg( "Technical Support" );

        entitymanager.persist( employee );

        entitymanager.getTransaction( ).commit( );

        entitymanager.close( );

        emfactory.close( );

  }

}

use jpadb

select * from employee
```

| Id | Name | Salary | Deg |
|---|---|---|---|
| 101 | Ravi | 60000 | Technical Support |