

FACULTY OF ENGINEERING AND TECHNOLOGY BACHELOR OF TECHNOLOGY

Advance Java Programming (AJP) (203105316)

V SEMESTER

Computer Science & Engineering Department





**CERTIFICATE**

*This is to certify that*

*Mr.* **VARIA DHRUV PRAFULBHAI** *with Enrollment No.* **210303105821** has *successfully completed his laboratory experiments in the subject (with Code)* **Advance Java Programming (****203105316)** *from the department of* **Computer Science and Engineering** *during the academic year* ***2022-2023.***



**Date of Submission …..…………… Staff In charge …..……………**

**Head of Department …..……………**

**TABLE OF CONTENT**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **S**  **r. N**  **o** | **Experiment Title** | **Page No** | | **Date of Start** | **Date of Completing** | **Sign** | **Marks**  **(Out of 10)** |
| **From** | **To** |
| 1 | Write a program to create registration form the student using AWT. |  |  |  |  |  |  |
| 2 | Write a program to create calculator using Swing. |  |  |  |  |  |  |
| 3 | Implement JDBC by connecting with database and execute Prepared Statement. |  |  |  |  |  |  |
| 4 | Implement JDBC by connecting with database and execute Callable Statement. |  |  |  |  |  |  |
| 5 | Implement chat application using java.net |  |  |  |  |  |  |
| 6 | Implement anyone sorting algorithm using TCP/UDP on Server application and Give Input on Client side and client should sorted output from server and display sorted on input side. |  |  |  |  |  |  |
| 7 | Implement Student information system using JDBC and RMI |  |  |  |  |  |  |
| 8 | Call remote procedure from a JVM to another JVM by implementing RMI. |  |  |  |  |  |  |
| 9 | Make a simple calculator using RMI |  |  |  |  |  |  |
| 10 | Study the functionalities of NetBeans and Connect to the Apache server. |  |  |  |  |  |  |
| 11 | Implement a simple Servlet application. Create directory structure, create references for web containers, create necessary web.xml and other config files and execute |  |  |  |  |  |  |
| 12 | Create registration form of student using Servlet and JDBC |  |  |  |  |  |  |
| 13 | Create a JSP page that is a student registration form. Perform server-side validation using JSP. |  |  |  |  |  |  |
| 14 | Create a custom tag using JSP tag extension / library. |  |  |  |  |  |  |
| 15 | Create user interface of a student registration and login using JSF. |  |  |  |  |  |  |
| 16 | Transfer all the Business Logic to the EJB of practical 10. |  |  |  |  |  |  |
| 17 | Create database and Implement JPA to provide persistence to practical 10. |  |  |  |  |  |  |

**Practical : 1**

*Write a program to create registration form for the student using AWT.*

import java.awt.\*;

import java.awt.event.\*;

public class pform

{

    Label lblTitle,lblName,lblFather,lblAge,lblGender,lblCourse,lblHobbies,lblAddress;

    TextField txtName,txtFather,txtAge;

    TextArea txtAddress;

    Checkbox checkMale, checkFemale, checkOthers, Hobbies1,Hobbies2,Hobbies3,Hobbies4;

    CheckboxGroup cbg;

    Choice Course;

    Button btnSave,btnClear;

    pform()

    {

        Frame a = new Frame("User Registrarion Form");

        a.setSize(1000,800);

        a.setLayout(null);

        a.setVisible(true);

        Color formColor = new Color(168, 131, 200);

        a.setBackground(formColor);

        Font titleFont = new Font("arial", Font.BOLD, 25);

        Font labelFont =new  Font("arial", Font.PLAIN, 18);

        Font textFont =new  Font("arial", Font.PLAIN, 15);

        lblTitle=new Label("Registration Form");

        lblTitle.setBounds(250,40,300,50);

        lblTitle.setFont(titleFont);

        lblTitle.setForeground(Color.BLUE);

        a.add(lblTitle);

        lblName=new Label("Name");

        lblName.setBounds(250,100,150,30);

        lblName.setFont(labelFont);

        lblName.setForeground(Color.WHITE);

        a.add(lblName);

        txtName=new TextField();

        txtName.setBounds(400,100,400,30);

        txtName.setFont(textFont);

        a.add(txtName);

        lblFather=new Label("Father Name");

        lblFather.setBounds(250,150,150,30);

        lblFather.setFont(labelFont);

        lblFather.setForeground(Color.WHITE);

        a.add(lblFather);

        txtFather=new TextField();

        txtFather.setBounds(400,150,400,30);

        txtFather.setFont(textFont);

        a.add(txtFather);

        lblAge=new Label("Age");

        lblAge.setBounds(250,200,150,30);

        lblAge.setFont(labelFont);

        lblAge.setForeground(Color.WHITE);

        a.add(lblAge);

        txtAge=new TextField();

        txtAge.setBounds(400,200,400,30);

        txtAge.setFont(textFont);

        a.add(txtAge);

        lblGender=new Label("Gender");

        lblGender.setBounds(250,250,150,30);

        lblGender.setFont(labelFont);

        lblGender.setForeground(Color.WHITE);

        a.add(lblGender);

        cbg = new CheckboxGroup();

        checkMale = new Checkbox("Male",cbg,true);

        checkMale.setBounds(400,250, 100, 30);

        checkMale.setFont(labelFont);

        checkMale.setForeground(Color.WHITE);

        a.add(checkMale);

        checkFemale = new Checkbox("Female",cbg,false);

        checkFemale.setBounds(500,250, 100, 30);

        checkFemale.setFont(labelFont);

        checkFemale.setForeground(Color.WHITE);

        a.add(checkFemale);

        checkOthers = new Checkbox("Others",cbg,false);

        checkOthers.setBounds(500,250, 100, 30);

        checkOthers.setFont(labelFont);

        checkOthers.setForeground(Color.WHITE);

        a.add(checkOthers);

        lblCourse=new Label("Course");

        lblCourse.setBounds(250,300,150,30);

        lblCourse.setFont(labelFont);

        lblCourse.setForeground(Color.WHITE);

        a.add(lblCourse);

        Course= new Choice();

        Course.setFont(labelFont);

        Course.setBounds(400, 300, 400, 50);

        Course.add("C");

        Course.add("C++");

        Course.add("Java");

        Course.add("C#");

        Course.add("Python");

        a.add(Course);

        lblHobbies=new Label("Hobbies");

        lblHobbies.setBounds(250,350,150,30);

        lblHobbies.setFont(labelFont);

        lblHobbies.setForeground(Color.WHITE);

        a.add(lblHobbies);

        Hobbies1=new Checkbox("Drawing");

        Hobbies1.setBounds(400, 350, 100, 50);

        Hobbies1.setFont(labelFont);

        Hobbies1.setForeground(Color.WHITE);

        a.add(Hobbies1);

        Hobbies2=new Checkbox("Singing");

        Hobbies2.setBounds(500, 350, 100, 50);

        Hobbies2.setFont(labelFont);

        Hobbies2.setForeground(Color.WHITE);

        a.add(Hobbies2);

        Hobbies3=new Checkbox("Music");

        Hobbies3.setBounds(600, 350, 100, 50);

        Hobbies3.setFont(labelFont);

        Hobbies3.setForeground(Color.WHITE);

        a.add(Hobbies3);

        Hobbies4=new Checkbox("Others");

        Hobbies4.setBounds(700, 350, 100, 50);

        Hobbies4.setFont(labelFont);

        Hobbies4.setForeground(Color.WHITE);

        a.add(Hobbies4);

        lblAddress=new Label("Address");

        lblAddress.setBounds(250,400,150,30);

        lblAddress.setFont(labelFont);

        lblAddress.setForeground(Color.WHITE);

        a.add(lblAddress);

        txtAddress=new TextArea(10,30);

        txtAddress.setBounds(400,400,400,100);

        txtAddress.setFont(labelFont);

        a.add(txtAddress);

        btnSave=new Button("Save Details");

        btnSave.setBounds(400,530,150,30);

        btnSave.setFont(labelFont);

        btnSave.setBackground(Color.GREEN);

        btnSave.setForeground(Color.WHITE);

        a.add(btnSave);

        btnClear=new Button("Clear All");

        btnClear.setBounds(560,530,150,30);

        btnClear.setFont(labelFont);

        btnClear.setBackground(Color.RED);

        btnClear.setForeground(Color.WHITE);

        a.add(btnClear);

        // Close Button Code

        this.addWindowListener(new WindowAdapter()

        {

            public void windowClosing(WindowEvent we)

            {

                System.exit(0);

            }

        });

    }

    private void addWindowListener(WindowAdapter windowAdapter) {

    }

    public static void main(String[]args)

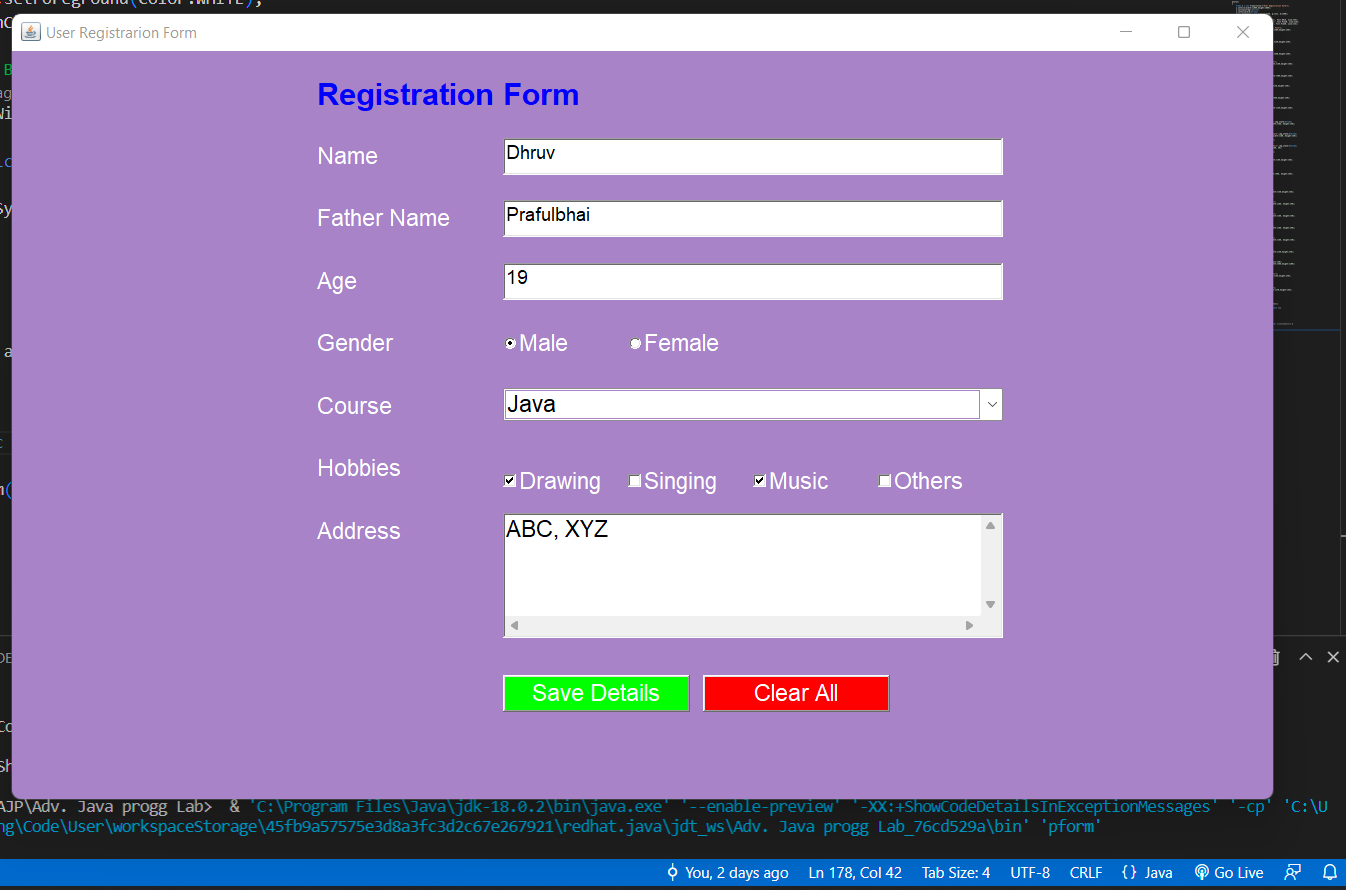
    {

        new pform();

    }

}

**Output :-**

****

**Practical : 2**

*Write a program to create calculator using Swing*

import javax.swing.\*;

import java.awt.event.\*;

public class p2calcswing implements ActionListener

{

    JFrame f;

    JTextField t;

    JButton b1,b2,b3,b4,b5,b6,b7,b8,b9,b0,bdiv,bmul,bsub,badd,bdec,beq,bdel,bclr;

    static double a=0,b=0,result=0;

    static int operator=0;

    p2calcswing()

    {

        f =new JFrame("Calculator");

        t =new JTextField();

        b1=new JButton("1");

        b2=new JButton("2");

        b3=new JButton("3");

        b4=new JButton("4");

        b5=new JButton("5");

        b6=new JButton("6");

        b7=new JButton("7");

        b8=new JButton("8");

        b9=new JButton("9");

        b0=new JButton("0");

        bdiv=new JButton("/");

        bmul=new JButton("\*");

        bsub=new JButton("-");

        badd=new JButton("+");

        bdec=new JButton(".");

        beq=new JButton("=");

        bdel=new JButton("Delete");

        bclr=new JButton("Clear");

        t.setBounds(30,40,280,30);

        b7.setBounds(40,100,50,40);

        b8.setBounds(110,100,50,40);

        b9.setBounds(180,100,50,40);

        bdiv.setBounds(250,100,50,40);

        b4.setBounds(40,170,50,40);

        b5.setBounds(110,170,50,40);

        b6.setBounds(180,170,50,40);

        bmul.setBounds(250,170,50,40);

        b1.setBounds(40,240,50,40);

        b2.setBounds(110,240,50,40);

        b3.setBounds(180,240,50,40);

        bsub.setBounds(250,240,50,40);

        bdec.setBounds(40,310,50,40);

        b0.setBounds(110,310,50,40);

        beq.setBounds(180,310,50,40);

        badd.setBounds(250,310,50,40);

        bdel.setBounds(60,380,100,40);

        bclr.setBounds(180,380,100,40);

        f.add(t);

        f.add(b7);

        f.add(b8);

        f.add(b9);

        f.add(bdiv);

        f.add(b4);

        f.add(b5);

        f.add(b6);

        f.add(bmul);

        f.add(b1);

        f.add(b2);

        f.add(b3);

        f.add(bsub);

        f.add(bdec);

        f.add(b0);

        f.add(beq);

        f.add(badd);

        f.add(bdel);

        f.add(bclr);

        f.setLayout(null);

        f.setVisible(true);

        f.setSize(350,500);

        f.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

        f.setResizable(false);

        b1.addActionListener(this);

        b2.addActionListener(this);

        b3.addActionListener(this);

        b4.addActionListener(this);

        b5.addActionListener(this);

        b6.addActionListener(this);

        b7.addActionListener(this);

        b8.addActionListener(this);

        b9.addActionListener(this);

        b0.addActionListener(this);

        badd.addActionListener(this);

        bdiv.addActionListener(this);

        bmul.addActionListener(this);

        bsub.addActionListener(this);

        bdec.addActionListener(this);

        beq.addActionListener(this);

        bdel.addActionListener(this);

        bclr.addActionListener(this);

    }

    public void actionPerformed(ActionEvent e)

    {

        if(e.getSource()==b1)

        t.setText(t.getText().concat("1"));

        if(e.getSource()==b2)

        t.setText(t.getText().concat("2"));

        if(e.getSource()==b3)

        t.setText(t.getText().concat("3"));

        if(e.getSource()==b4)

        t.setText(t.getText().concat("4"));

        if(e.getSource()==b5)

        t.setText(t.getText().concat("5"));

        if(e.getSource()==b6)

        t.setText(t.getText().concat("6"));

        if(e.getSource()==b7)

        t.setText(t.getText().concat("7"));

        if(e.getSource()==b8)

        t.setText(t.getText().concat("8"));

        if(e.getSource()==b9)

        t.setText(t.getText().concat("9"));

        if(e.getSource()==b0)

        t.setText(t.getText().concat("0"));

        if(e.getSource()==bdec)

        t.setText(t.getText().concat("."));

        if(e.getSource()==badd)

        {

            a=Double.parseDouble(t.getText());

            operator=1;

            t.setText("");

        }

        if(e.getSource()==bsub)

        {

            a=Double.parseDouble(t.getText());

            operator=2;

            t.setText("");

        }

        if(e.getSource()==bmul)

        {

            a=Double.parseDouble(t.getText());

            operator=3;

            t.setText("");

        }

        if(e.getSource()==bdiv)

        {

            a=Double.parseDouble(t.getText());

            operator=4;

            t.setText("");

        }

        if(e.getSource()==beq)

        {

            b=Double.parseDouble(t.getText());

        switch(operator)

        {

            case 1: result=a+b;

            break;

            case 2: result=a-b;

            break;

            case 3: result=a\*b;

            break;

            case 4: result=a/b;

            break;

            default: result=0;

        }

            t.setText(""+result);

        }

        if(e.getSource()==bclr)

        t.setText("");

        if(e.getSource()==bdel)

        {

        String s=t.getText();

        t.setText("");

        for(int i=0;i<s.length()-1;i++)

        t.setText(t.getText()+s.charAt(i));

        }

    }

    public static void main(String...s)

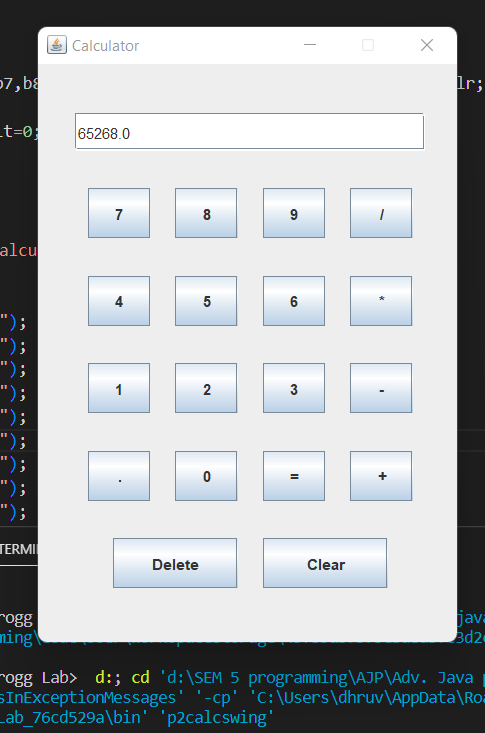
    {

        new p2calcswing();

    }

}

**Output :-**



**Practical : 3**

*Implement JDBC by connecting with database and execute Prepared Statement*

import java.sql.\*;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.SQLException;

public class NewJFrame extends javax.swing.JFrame {

    /\*\*

     \* Creates new form NewJFrame

     \*/

    public NewJFrame() {

        initComponents();

    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_jButton1ActionPerformed

        try {

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

                String n = jTextField2.getText();

                String p = jTextField3.getText();

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

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

                System.out.println("Connection Established");

                String query = " insert into user(id,name,password)"+"values(?,?,?)";

                 PreparedStatement preparedStmt = con.prepareStatement(query);

                 preparedStmt.setInt(1, a);

                 preparedStmt.setString(2, n);

                 preparedStmt.setString(3, p);

                int i = preparedStmt.executeUpdate();

                jLabel5.setText(1 + "records inserted");

                con.close();

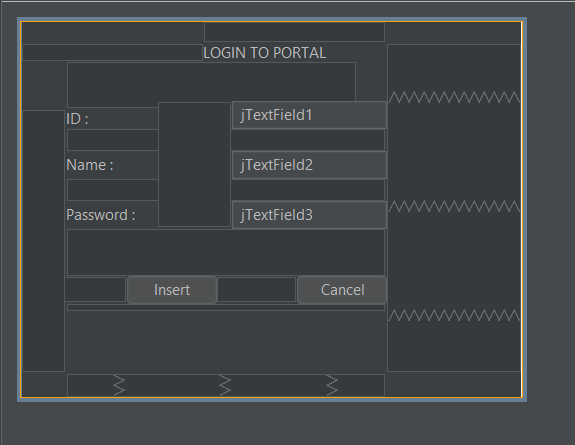
        } catch (Exception e) {

            System.out.println("e");

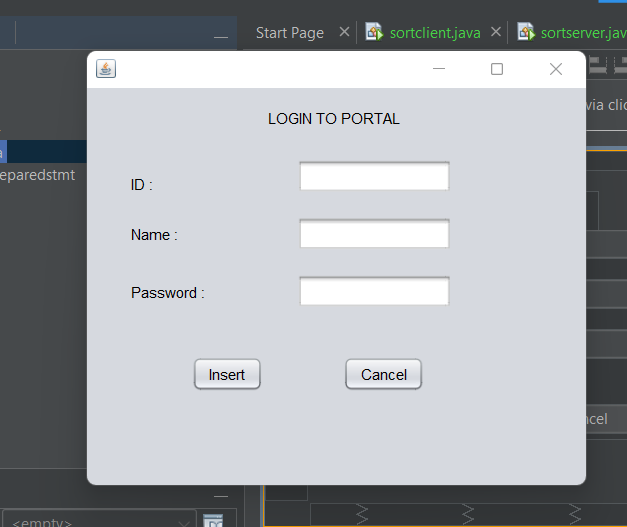
        }

    }

*Configuration & Design of Swing components using Apache NETBEANS IDE :-*



**Output :-**



**Practical : 4**

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

import java.sql.\*;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.SQLException;

public class NewJFrame extends javax.swing.JFrame {

    public NewJFrame() {

        initComponents();

    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_jButton1ActionPerformed

        try

        {

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

                String n = jTextField2.getText();

                String p = jTextField3.getText();

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

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

            System.out.println("Connection Established");

            CallableStatement stmt=con.prepareCall("{call INSERT\_dhruv(?,?,?)}");

            stmt.setInt(1,a);

            stmt.setString(2,n);

            stmt.setString(3, p);

            int i=stmt.executeUpdate();

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

                }

            catch(SQLException | ClassNotFoundException e)

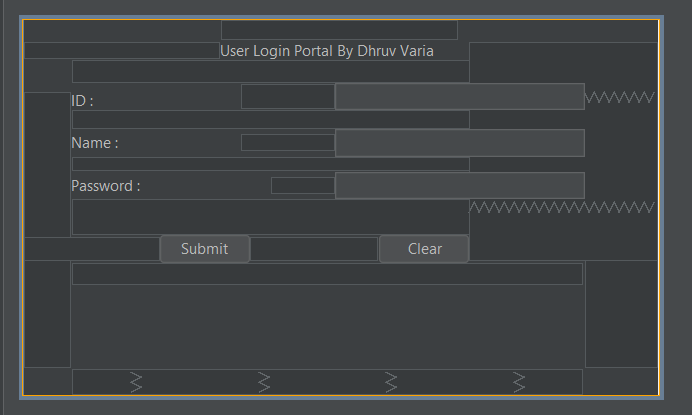
            {

                System.out.println(e);

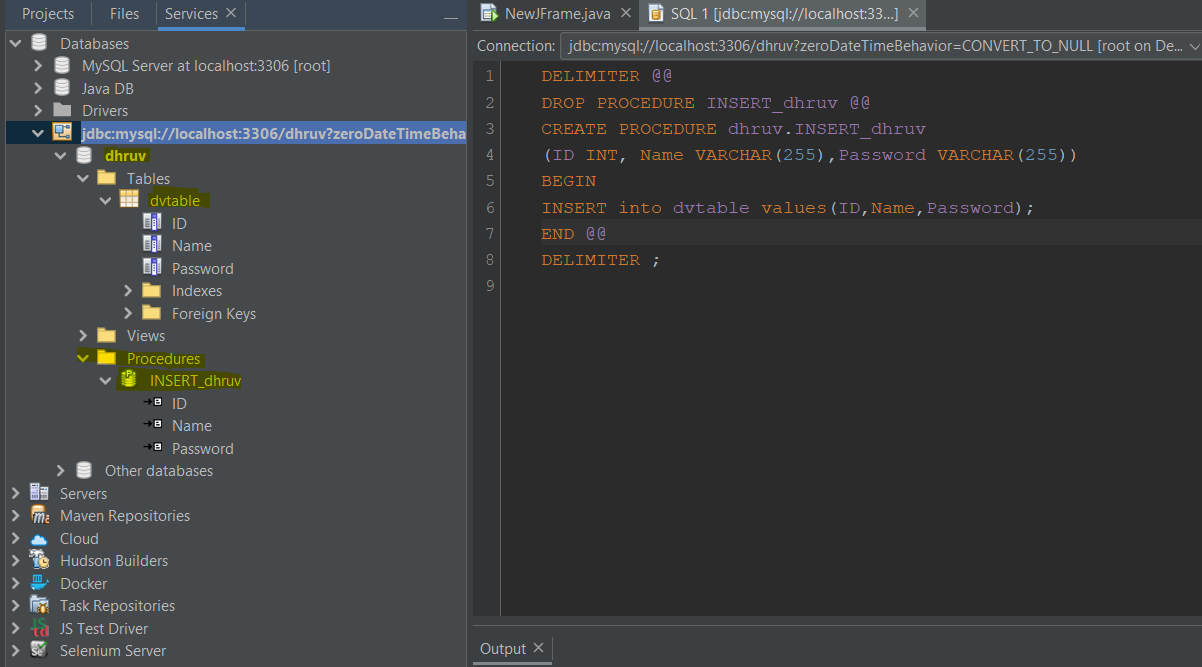
            }

    }

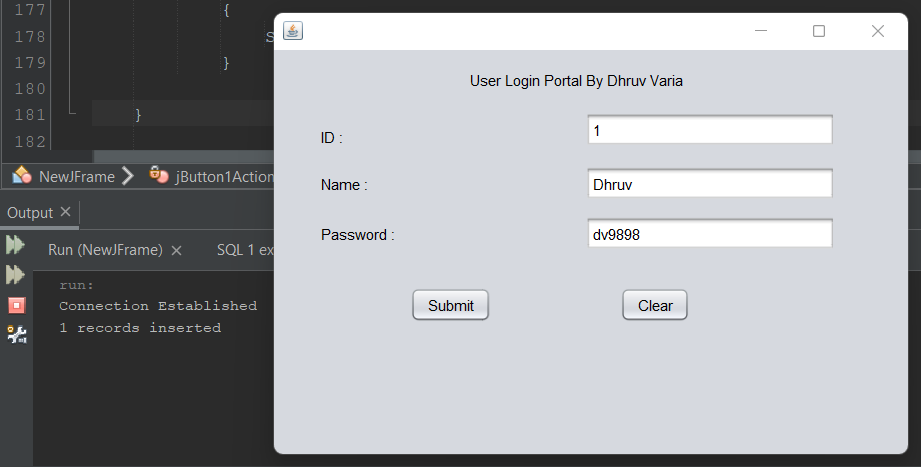
*Configuration & Design of Swing components using Apache NETBEANS IDE :-*



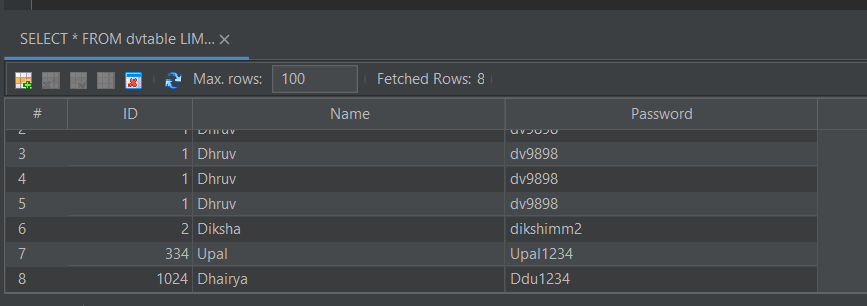
*Configuring JDBC DB Type 4 Driver and create DB, Procedure for callable statement.*



**Output :-**



**SQL Database Table Records :-**



**Practical : 5**

*Implement chat application using java.net.*

**Java Class 1 : Server :-**

import java.net.\*;

import java.io.\*;

public class server

{

    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();//read frm client

              while(str!=null)

               {

                    System.out.println(str);//reads frm client n writes on server

                    System.out.print("Write something  : ");

                    str1=kb.readLine();

                    ps.println(str1);

                    str=br.readLine();//reads frm client

               }

            ps.close();

            br.close();

            kb.close();

            s.close();

            ss.close();

        }

        catch(IOException e)

        {

            System.out.println(e);

        }

    }

}

**Java Class 2 : Client :-**

import java.net.\*;

import java.io.\*;

public class client

{

    public static void main(String[] args)

    {

        try{

           InetAddress ip= InetAddress.getLocalHost();

            Socket s=new Socket(ip,4021);

            //"192.168.1.47"

            BufferedReader br=new BufferedReader(new InputStreamReader(s.getInputStream()));

            BufferedReader kb=new BufferedReader(new InputStreamReader(System.in));

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

            System.out.print("write something : ");

            String str=kb.readLine();

            String str1;

               while(!(str.equals("exit")))

               {

                    ps.println(str);

                    str1=br.readLine();

                    System.out.println(str1);

                    System.out.print("write something : ");

                    str=kb.readLine();

          }

            ps.close();

            br.close();

            kb.close();

            s.close();

            System.out.println("client program ended");

        }

        catch(IOException e)

        {

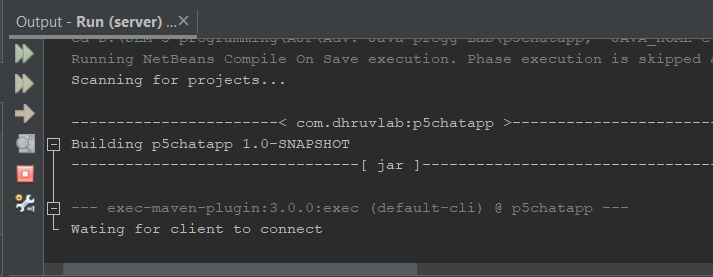
            System.out.println(e);

        }

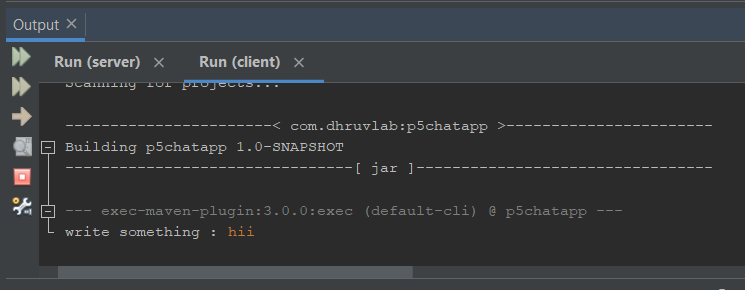
    }

}

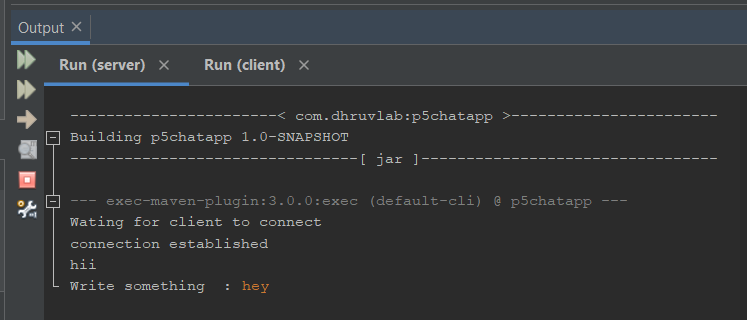
**Output :** Running Server :-



Running Client and sending Message:-



Receiving message and responding to client :-



**Practical : 6**

*Implement anyone sorting algorithm using TCP/UDP on Server application and Give Input on Client side and client should sorted output from server and display sorted on input side.*

**Java Class 1 : Server :-**

import java.net.\*;

import java.io.\*;

public class sortserver {

    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.println(sendarr);

        p.flush();

        s.close();

        }

        catch(IOException | NumberFormatException e)

        {

            System.out.println(e);

        }

    }

}

**Java Class 2 : Client :-**

import java.net.\*;

import java.io.\*;

public class sortclient

{

    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)

        {

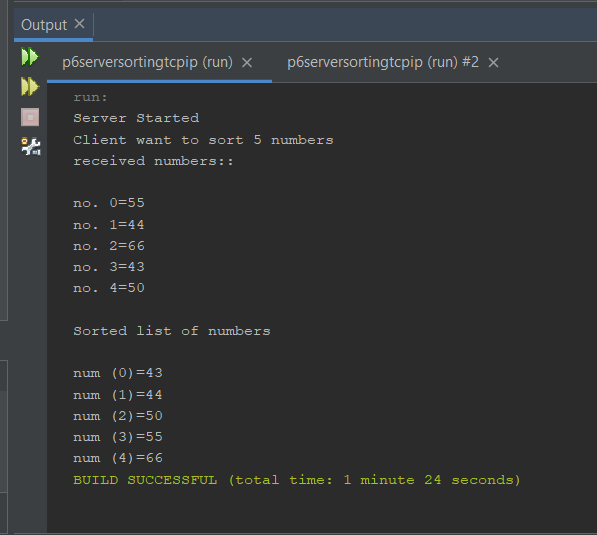
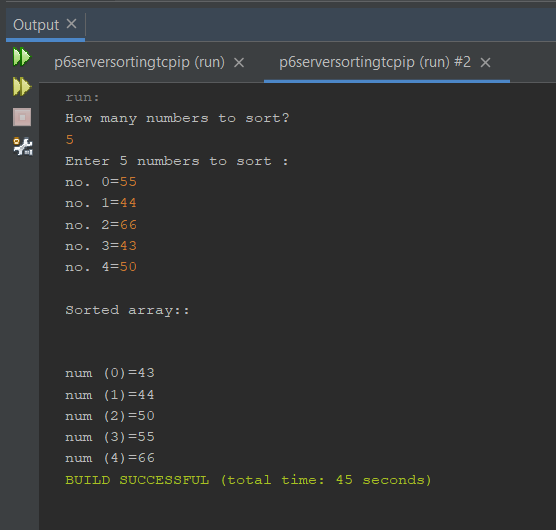
            System.out.println(e);

        }

    }

}

**Server Output :- Client Output :-**

**Practical : 7**

*Implement Student information system using JDBC and RMI*

**Code :**

**StudentI.java**

import java.rmi.Remote;

import java.sql.\*;

import java.util.\*;

public interface StudentI extends Remote

{

public abstract ArrayList insert(int id,String name,String branch,int atd) throws Exception;

}

**StudentC.java**

import java.sql.\*;

import java.rmi.\*;

import java.rmi.server.\*;

import java.util.\*;

public class StudentC extends UnicastRemoteObject implements StudentI

{

public StudentC() throws Exception

{

super();

}

public ArrayList insert(int id,String name,String branch,int atd) throws Exception

{

ArrayList ar=new ArrayList();

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

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

Statement stmt=con.createStatement();

stmt.executeUpdate("insert into student values("+id+",'"+name+"','"+branch+"',"+atd+")");

ResultSet rs=stmt.executeQuery("select \* from student where id="+id); rs.next();

int Id=rs.getInt(1);

name=rs.getString(2);

branch=rs.getString(3);

atd=rs.getInt(4);

ar.add(new Integer(Id));

ar.add(name);

ar.add(branch);

ar.add(new Integer(atd));

con.close();

return ar;

}

}

**Server.java**

import java.rmi.\*;

public class Server

{

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

{

StudentI obj=new StudentC();

Naming.rebind("stinfo",obj);

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

}

}

**Client.java**

import java.rmi.\*;

import java.sql.\*;

import java.util.\*;

public class Client

{

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

{

Scanner scan= new Scanner(System.in);

System.out.println("Enter id,name,branch,attendance for student:");

int id=scan.nextInt();

String name=scan.next();

String branch=scan.next();

int atd=scan.nextInt();

StudentI obj=(StudentI)Naming.lookup("stinfo");

ArrayList ar=obj.insert(id,name,branch,atd);

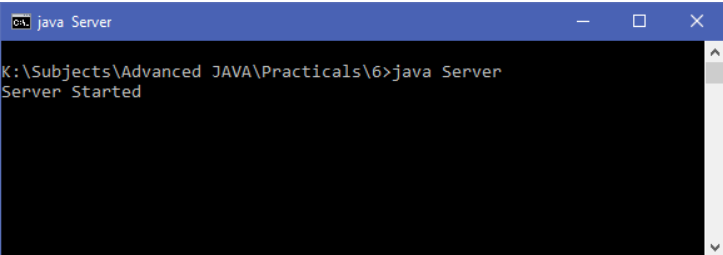
Iterator it=ar.iterator();

System.out.println("Id\tName\tBranch\tAttendance"); System.out.println(it.next()+"\t"+it.next()+"\t"+it.next()+"\t"+it.next()+"\t");

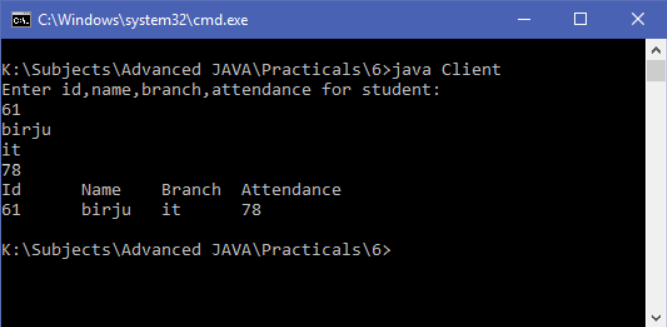
}

}

**Server Output :-**



**Client Output :-**



**Practical : 8**

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

**Code:-**

Consider a scenario, there are two applications running in different machines. Let's say Machine A and Machine B, Machine A is located in United States and Machine B in India. Machine B want to get list of all the customers of Machine A application.

1. **Create the table**

First of all, we need to create the table in the database. Here, we are using Oracle10 database.

******

1. **Create Customer class and Remote interface**

**Customer.java**

package com.company;

public class Customer implements java.io.Serializable

{

private int acc\_no;

private String firstname,lastname,email;

private float amount;

}

**Bank.java**

package com.company;

import java.rmi.\*;

import java.util.\*;

interface Bank extends Remote

{

public List getCustomers()throws RemoteException;

}

1. **Create the class that provides the implementation of Remote interface**

**BankImpl.java**

package com.javatpoint;

import java.rmi.\*;

import java.rmi.server.\*;

import java.sql.\*;

import java.util.\*;

class BankImpl extends UnicastRemoteObject implements Bank

{

BankImpl()throws RemoteException{}

public ListgetCustomers()

{

List<Customer>getCustomers();

Try

{

Class.forName("oracle.jdbc.driver.OracleDriver");

Connection con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","syste m","oracle");

PreparedStatement ps=con.prepareStatement("select \* from customer400");

ResultSet rs=ps.executeQuery();

while(rs.next())

{

Customer c=new Customer();

c.setAcc\_no(rs.getInt(1));

c.setFirstname(rs.getString(2));

c.setLastname(rs.getString(3));

c.setEmail(rs.getString(4));

c.setAmount(rs.getFloat(5));

list.add(c);

} con.close();

}

catch(Exception e)

{

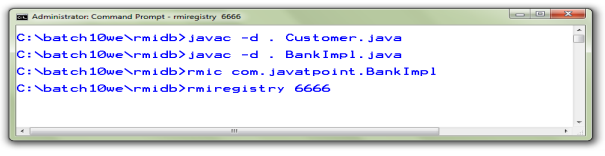
System.out.println(e);

}

return list;

}

1. **Compile the class rmic tool and start the registry service by rmiregistry tool**

****

1. **Create and run the Server**

**MyServer.java**

package com.company;

import java.rmi.\*;

public class MyServer

{

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

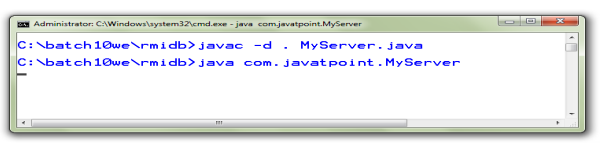
{

Remote r=new BankImpl();

Naming.rebind("rmi://localhost:6666/company",r);

}

}

****

1. **Create and run the Client**

**MyClient.java**

package com.company;

import java.util.\*;

import java.rmi.\*;

public class MyClient

{

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

{

Bank b=(Bank)Naming.lookup("rmi://localhost:6666/javatpoint");

List list=b.getCustomers();

for(Customer c:list)

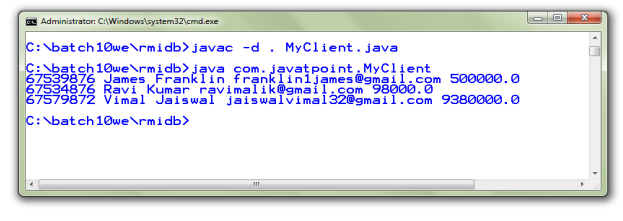
{

System.out.println(c.getAcc\_no()+" "+c.getFirstname()+" "+c.getLastname() +" "+c.getEmail()+" "+c.getAmount());

}

}

}

****

**Practical : 9**

*Make a simple calculator using RMI.*

**(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)

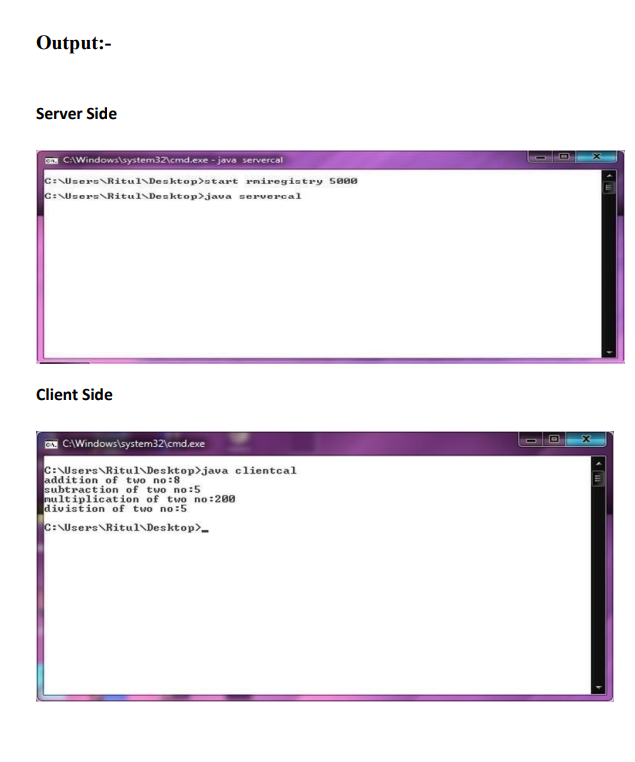
{

System.out.println(e);

}

}

}

****

**Practical : 10**

*Study the functionalities of Eclipse/NetBeans and Connect to the Glassfish / Apache server*

**What is an IDE?**

IDE stands for “integrated development environment.” An IDE is a tool for [programmers](https://careerkarma.com/blog/software-engineer-vs-programmer/). It allows for more than just text editing. An IDE is a text editor, debugger, and compiler all in one. This makes for speedy programming with less editing time and stress.

IDEs tend to take up more CPU power and memory than the average text editor, and the sheer volume of features and capabilities can [compromise efficient text editing](https://spin.atomicobject.com/2015/12/22/ide-vs-text-editor/). Therefore, some programmers and coders choose to stick with a basic text editor. Even so, IDEs are powerful tools as you learn how to code, develop your coding style, and determine what to do with your skills.

the most commonly-chosen IDEs for coding with Java are NetBeans, Eclipse, and IDEA.

**NetBeans as Your IDE**

NetBeans

[NetBeans](https://netbeans.org/about/history.html) started in Prague as a student project to create a Java IDE. The project caught so much attention that the students successfully marketed it as a commercial project. In 1999, Sun Microsystems, the originator of Java, took a liking to NetBeans and struck a deal with the students. Sun Microsystems obtained NetBeans, and kept it open source. Oracle bought Sun Microsystems in 2010 and took over both Java and NetBeans. Oracle made NetBeans an official Java IDE.

NetBeans is more than just an IDE; it is a platform. It supports coding in HTML, Java, and CSS with a clean-cut structure designed to simplify the look of large projects.

NetBeans is ready to use out of the box. No extra plugins or extensions are necessary to increase usability. Since it is open source, there are thousands of other users that can help contribute to projects and concepts. And, [as noted by NetBeans itself](https://netbeans.org/switch/why.html), they even have a developer support package to help you fix what the open-source community cannot.

Overall, NetBeans is an amazing, open-source Java IDE that allows for optimal functionality and version control out of the box. It’s a free software supported by the same company that owns Java, which means NetBeans has fantastic Java support.

**Eclipse as Your IDE**

Eclipse

Eclipse was developed in the late 1990s in response to a rising need for an all-in-one editor. [According to Iri](https://www.iri.com/blog/iri/business/brief-history-of-eclipse/), International Business Machines (IBM) designed Eclipse to handle text editing, compiling, and debugging on a more commercial scale. It didn’t take long for IBM to make Eclipse open source and give it a royalty-free license. Doing so allowed more companies to utilize the software on a commercial scale. 

Eclipse formed the [Eclipse Foundation](https://www.eclipse.org/). It’s a nonprofit creating a vendor-neutral source for implementing, sharing, and teaching the developments of the open-source code in Eclipse. The Eclipse Foundation is still active today. It aims to establish global collaboration of individuals and companies alike to create new innovations in and improve the commercial experience of Eclipse.

Eclipse is now commonly integrated in commercial settings. Eclipse was even an original competitor for [Android](https://careerkarma.com/blog/programming-languages-android/) App developers. The IDE is capable of handling large amounts of code in order to create powerful programs, applications, and extensions to improve the Eclipse experience. Like NetBeans, Eclipse is written in Java, so it’s capable of supporting nearly all operating systems, but it doesn’t limit its users to writing in Java.

Eclipse is designed with customization in mind, running off various extensions and plugins. The idea behind Eclipse is that out of the box it functions at the most “bare necessities” level. Eclipse’s extensions and plugins provide rich functionality. That functionality makes the Eclipse experience what it is.

Therefore, Eclipse’s learning curve can be steep if you are starting from little to no experience. However, if your goal is large-scale projects or app development, this learning curve is well worth climbing. Now that we have a better grasp of what’s going on with each of these popular Java IDEs, it’s time to compare NetBeans vs Eclipse head to head.

### **NetBeans vs Eclipse: Out-of-the-Box Experience**

The first major difference between the NetBeans and Eclipse is the out-of-the-box experience.

NetBeans is ready to go out of the box, complete with nice functional elements. It comes with easy-to-understand drag-and-drop modules and handy plugins available to the user before they even click a button.

Eclipse is, on the other hand, a rather barren program to begin with. This is because Eclipse offers a fully customizable experience, but that means it can be rather difficult to learn if you’re just beginning the learning process. When you’re working with Eclipse, you must first learn how to add plugins and extensions, as well as decide which ones to install.

### **NetBeans vs Eclipse: Project Size**

Your project size will also affect your decision. NetBeans is a quick and fantastic IDE, but it’s not developed to the enterprising level that Eclipse is designed to handle. Keep your goals in mind when making your decision. NetBeans’ easier learning process may be appealing to you as you start out, especially since you’ll be starting with smaller projects. However, if your goal is to work on Android development or large scale projects, your future self will thank you for taking the time now to learn Eclipse.

## Conclusion: NetBeans vs Eclipse

Overall, both Eclipse and NetBeans are high-functioning Java IDEs. So, when comparing NetBeans or Eclipse, you simply need to determine what’s most important to you.

NetBeans is functional from the get-go and considerably easy to learn for the beginning coder. However, it can be a little rudimentary when it comes to editing larger projects.

Eclipse, on the other hand, is designed to handle larger projects in an elegant fashion. It does have a steeper learning curve, though, since it relies on the immediate adaptation of plugins and extensions to become the slick software it’s designed to be.

It’s hard to choose the wrong option, since both NetBeans and Eclipse are elegant and high functioning. So, gathering what you learned from this article, pick your favourite IDE and code away! You’re on the move to greatness, one line at a time.

**Practical : 11**

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

**Code:-**

Here, we are going to use apache tomcat server in this example.

The steps are as follows:

1.Create a directory structure

2.Create a Servlet

3.Compile the Servlet

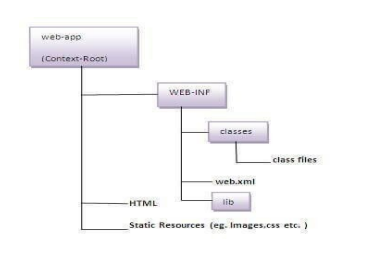
4.Create a deployment descriptor

5.Start the server and deploy the project

6.Access the servlet

1. **Create a directory structures**

The directory structure defines that where to put the different types of files so that web container may get the information and respond to the client. The Sun Microsystem defines a unique standard to be followed by all the server vendors. Let's see the directory structure that must be followed to create the servlet.



As you can see that the servlet class file must be in the classes folder. The web.xml file must be under the WEB-INF folder.

1. **Create a Servlet**

**DemoServlet.java**

import javax.servlet.http.\*;

import javax.servlet.\*;

import java.io.\*;

public class DemoServlet extends HttpServlet{

public void doGet(HttpServletRequest eq,HttpServletResponse res)

throws ServletException,IOException {

.res.setContentType("text/html");

PrintWriter pw=res.getWriter();

pw.println("");

.pw.println("Welcome to servlet");

pw.println("");

pw.close()

}

}

1. **Compile the servlet**

**For apache tomcat program, we have to use servlet-api.jar for compiling.**

1. **Create the deployment descriptor (web.xml file) web.xml file**

*<web-app>*

*<servlet>*

*<servlet-name>sonoojaiswal</servlet-name>*

*<servlet-class>DemoServlet</servlet-class>*

*</servlet>*

*<servlet-mapping>*

*<servlet-name>sonoojaiswal</servlet-name>*

*<url-pattern>/welcome</url-pattern>*

*</servlet-mapping>*

*</web-app>*

1. **Start the Server and deploy the project**

*To start Apache Tomcat server, double click on the startup.bat file under apache-tomcat/bin*

*directory.*

*To deploy the servlet,*

*Copy the project and paste it in the webapps folder under apache tomcat.*

*You can also create war file, and paste it inside the webapps directory. To do so, you need to*

*use jar tool to create the war file. Go inside the project directory (before the WEB-INF), then*

*write:*

*<projectfolder> jar cvf myproject.war \**

1. **To access the servlet**

*Open browser and write http://hostname:portno/contextroot/urlpatternofservlet. For*

*example:*

*http://localhost:9999/demo/welcome*

****

**Practical : 12**

*Create registration form of student using Servlet & JDBC.*

**Code :**

**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;

@WebServlet("/First")

public class First extends HttpServlet

{

private static final long serialVersionUID = 1L

public First() {

// TODO Auto-generated constructor stub

}

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());

}

}

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>

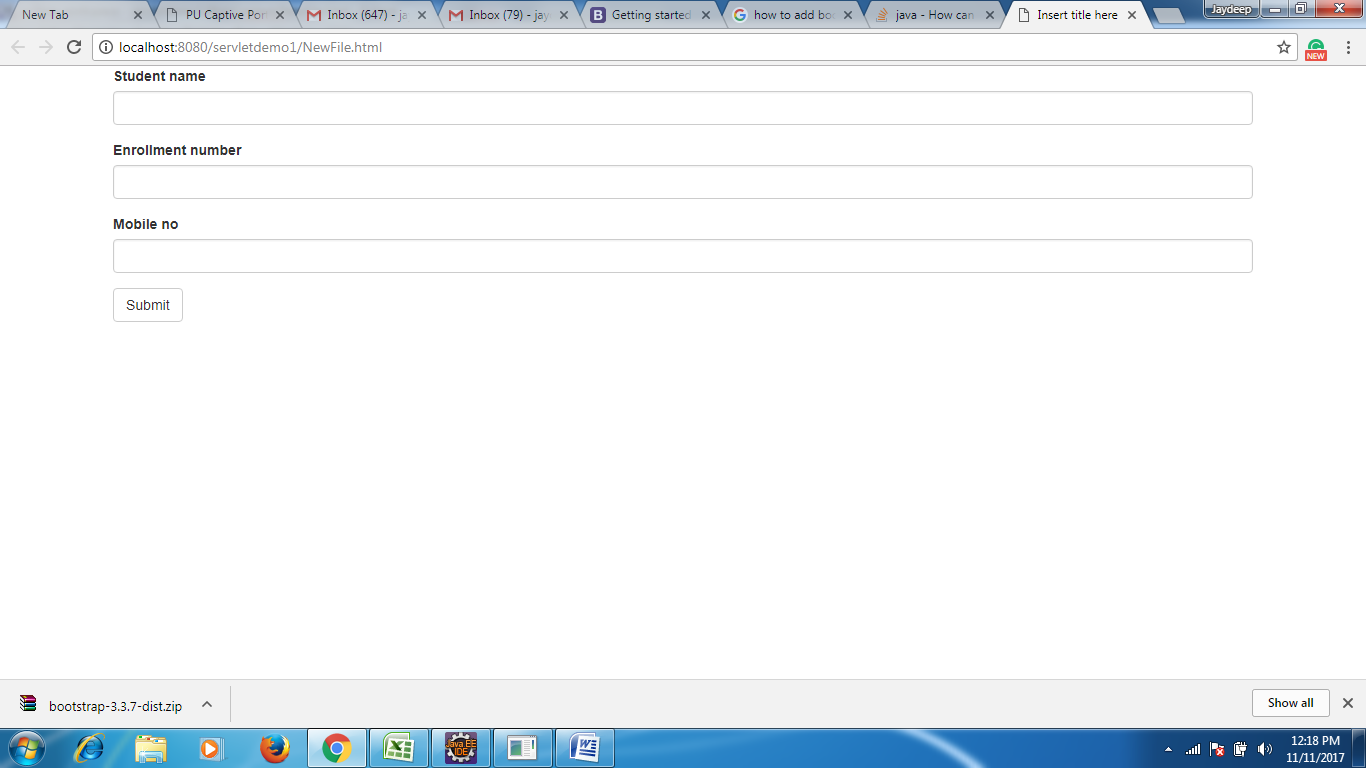
</form>

</div>

</body>

</html>

**Output:**



**Practical : 13**

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

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**

*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>

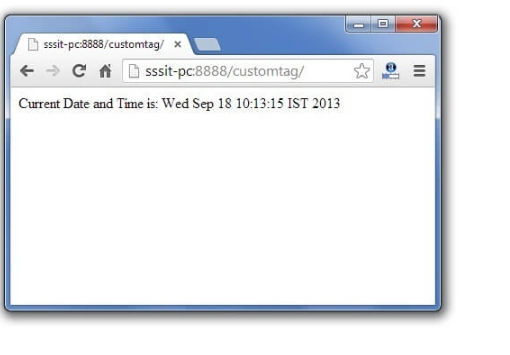
</tag>

</taglib>

*index.jsp*

<%@ taglib uri="WEB-INF/mytags.tld" prefix="m" %>

Current Date and Time is: <m:today/>



**Practical : 15**

*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>

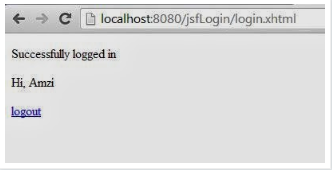
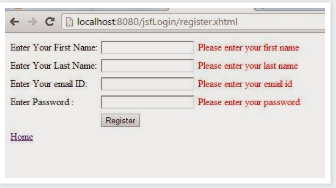
    </f:view>

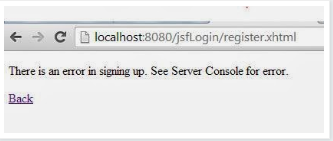
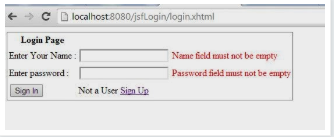
</h:body>

</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");  
    }  
}





**Practical : 16**

*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**

*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

