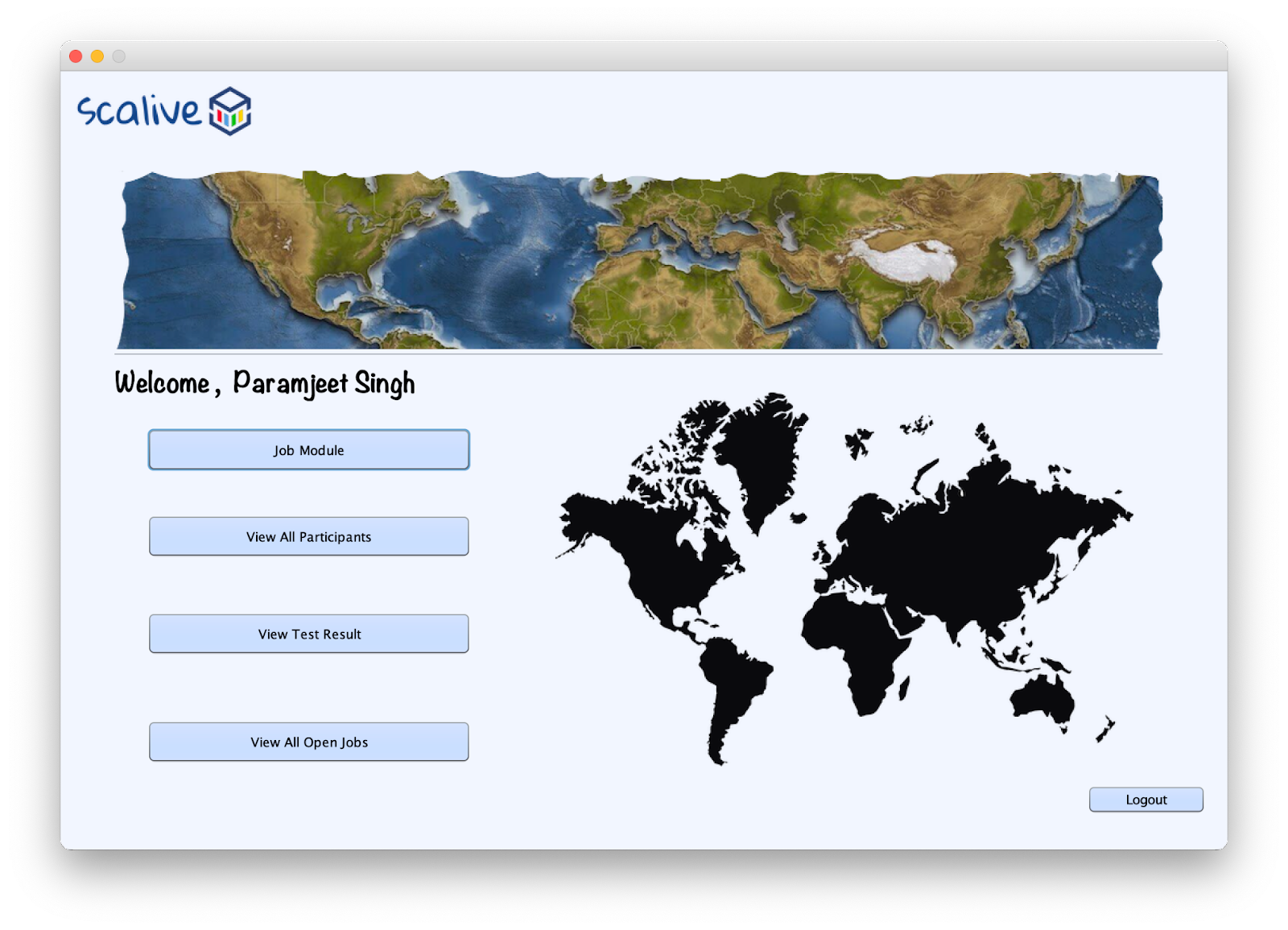
Java SE Project

Training And Placement Application

***Hr Job Module Documentation***

1. **HrOptionsFrame –**



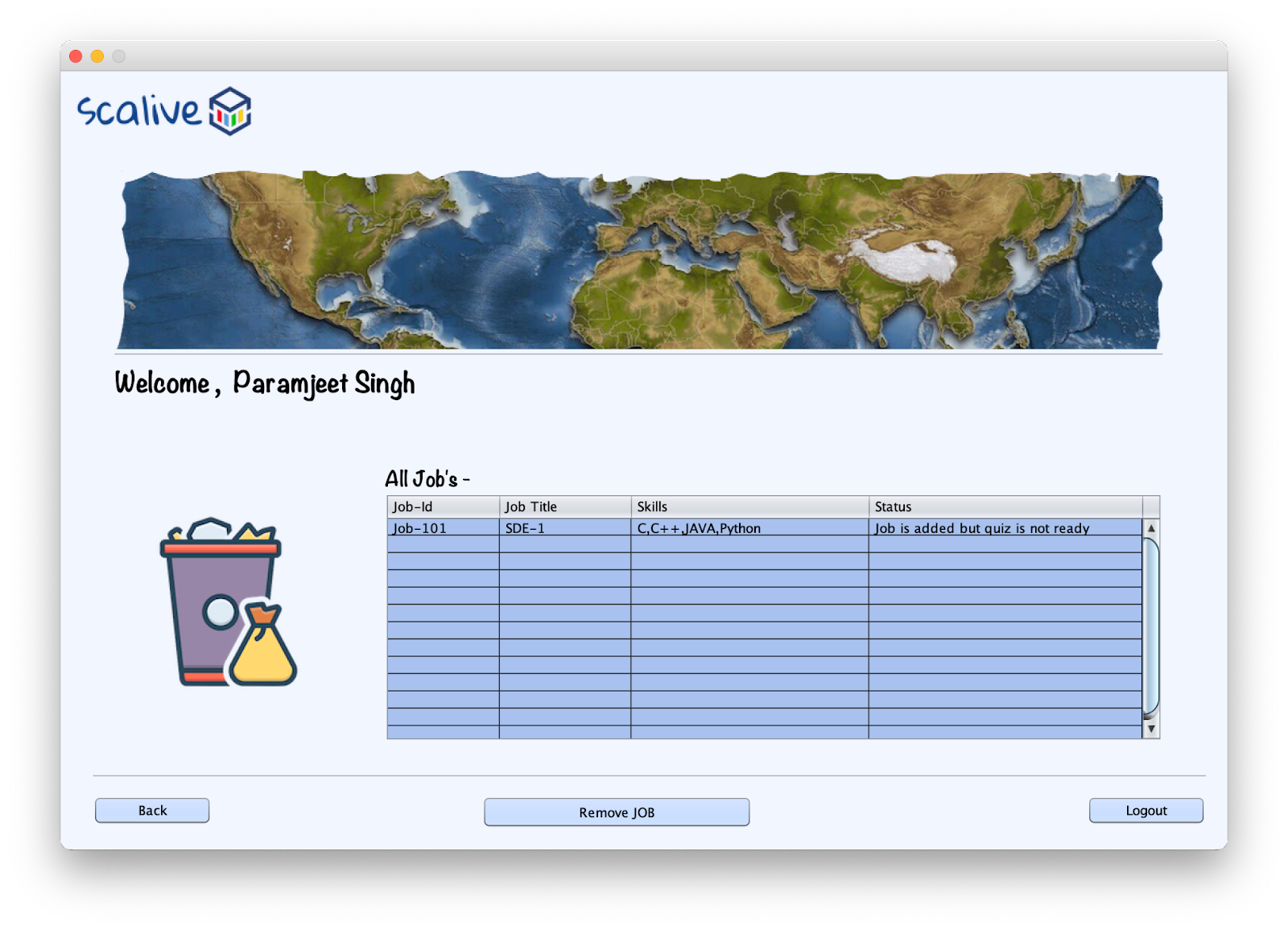
1. **HrJobOptionsFrame**



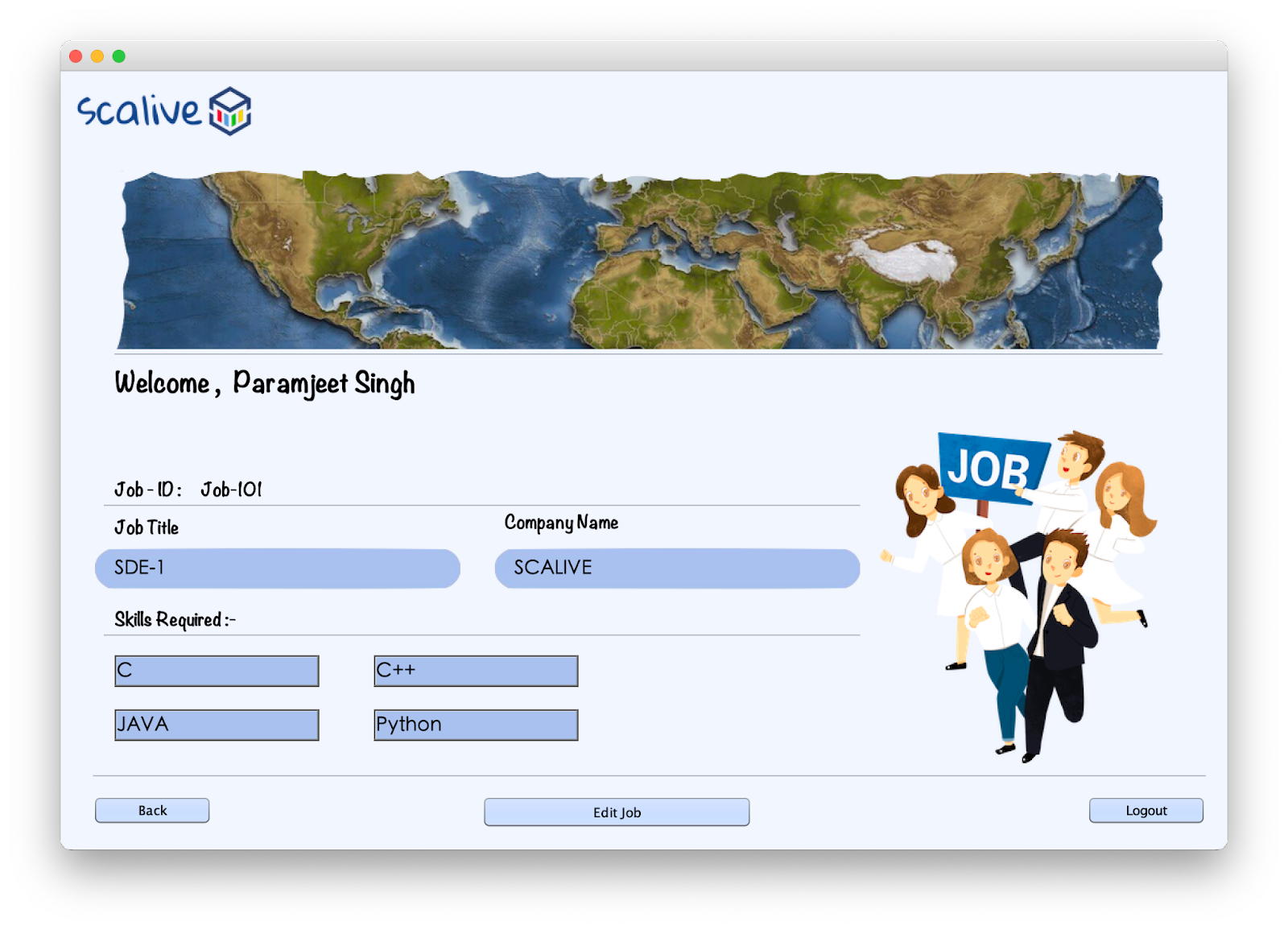
1. **HrAddNewJobFrame**



1. **HrRemoveJobFrame**

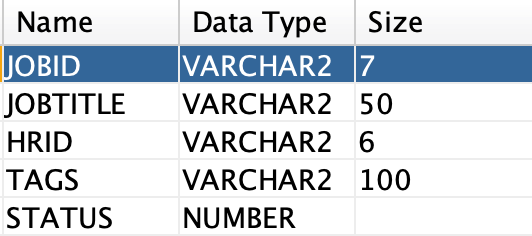


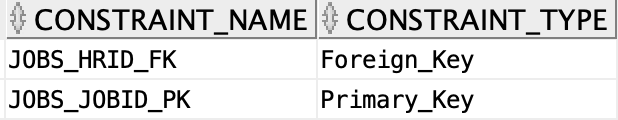
1. **HrEditJobFrame**

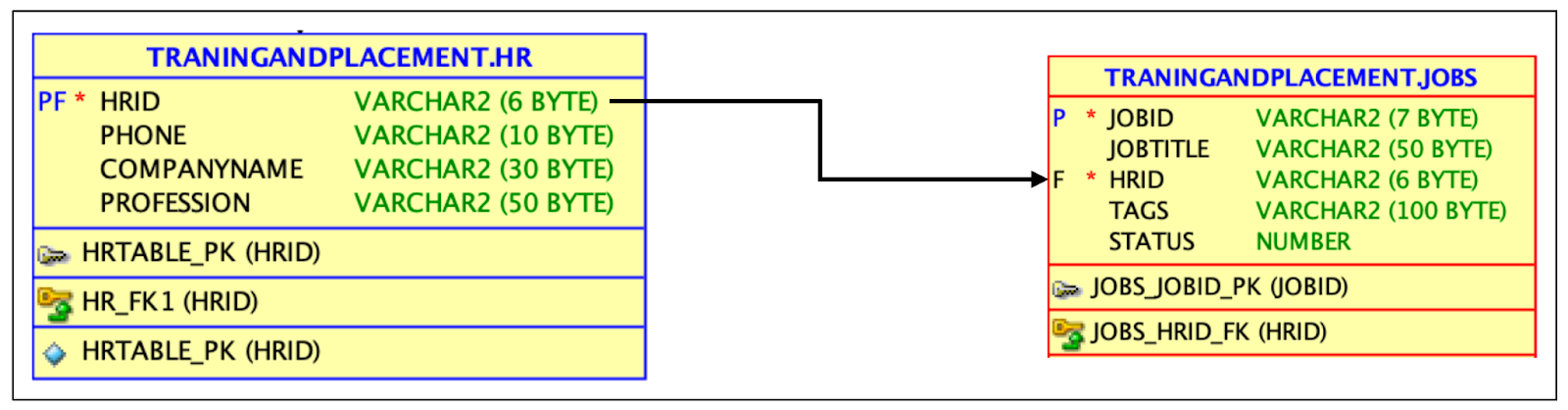


1. **Create a table for JOBS –**

For Job’s functionality, we have to create a table in the database by name of “**JOBS**”. Its column names and data type are shown below snapshot :







1. **POJO ( Plain Old Java Object ) for JOB’s functionality -**

1. create a class inside the package by the name of **“JobPojo.java”**

So the setter and getter are depends on the no of variables that we have created in the class.

In our scenario, we have 5 variables by the name:-

private String jobId;

     private String jobTitle;

    private String hrId;

     private String tags;

     private int status;

So we have to create 5 setters and 5 getters in our POJO class -

public String getJobId( )

public void setJobId( String jobId )

public String getJobTitle( )

public void setJobTitle( String jobTitle )

public String getHrid( )

public void setHrid( String hrid )

public String getTags( )

public void setTags( String tags )

public int getStatus( )

public void setStatus( int status )

1. **DAO ( Data Access Object ) for JOB’s functionality –**

In the **HrDAO** class define a method called **getCompanyNameById()** which should accept an **HrID** as argument and return the name of the company of Hr.

**public static String getCompanyNameById(String hrId)throws SQLException{**

**Connection conn=DBConnection.getConnection();**

**PreparedStatement ps=conn.prepareStatement("select companyname from hr where hrid=?");**

**ps.setString(1, hrId);**

**ResultSet rs=ps.executeQuery();**

**rs.next();**

**return rs.getString(1);**

**}**  
 Now create another class by the name of **“JobDAO.java”**and define following methods in it:

▶︎ **Method for generating new JobId**

So we have to create a method called **“getJobId(  )”** this method generates a new jobid automatically for a new job.

Job id format - **‘Job-101’**

**public static int getJobId( ) throws SQLException**

**{**

**Connection conn=DBConnection.getConnection();**

**Statement st=conn.createStatement();**

**ResultSet rs=st.executeQuery("Select max(jobid) from jobs");**

**int jobId=101;**

**rs.next();**

**String strid=rs.getString(1);**

**if(strid!=null){**

**String id=strid.substring(3);**

**jobId=Integer.parseInt(id)+1;**

**}**

**return jobId;**

**}**

▶︎ **Method for Adding new job**

So we have to create a method called **“addNewJob( JobPojo job )”** *( And pass the POJO object as arguments we have created in step 3)* in which the job details are already passed by **HrAddNewJobFrame** And insert into the **JOBS Table**

**public static boolean addNewJob(JobPojo job)throws SQLException{**

**Connection conn=DBConnection.getConnection();**

**PreparedStatement ps=conn.prepareStatement("insert into jobs values(?,?,?,?,?)");**

**ps.setString(1, job.getJobId());**

**ps.setString(2,job.getJobTitle());**

**ps.setString(3, job.getHrId());**

**ps.setString(4, job.getTags());**

**ps.setInt(5, job.getStatus());**

**return 1==ps.executeUpdate();**

**}**

1. **Button Code & functionality -**

▶︎ **HrAddNewJobFrame’s add new Job button’s -**

**public class HrAddNewJobFrame extends javax.swing.JFrame {**

**public HrAddNewJobFrame() {**

**initComponents();**

**this.setLocationRelativeTo(null);**

**lblName.setText(CurrentUser.name);**

**loadData();**

**}**

**private void loadData(){**

**try{**

**lblJobId.setText("Job-"+JobDAO.getNewJobId());**

**if(txtCompanyName.getText().trim().isEmpty())**

**txtCompanyName.setText(HrDAO.getCompanyNameById(CurrentUser.getId()));**

**}**

**catch(SQLException ex){**

**JOptionPane.showMessageDialog(null,"DB Error","Error",JOptionPane.ERROR\_MESSAGE);**

**ex.printStackTrace();**

**}**

**}**

**private boolean validateInputs(){**

**if(txtJobTitle.getText().trim().isEmpty()||txtSkill1.getText().trim().isEmpty()||txtSkill2.getText().trim().isEmpty()||txtSkill3.getText().trim().isEmpty()||txtSkill4.getText().trim().isEmpty())**

**return false;**

**return true;**

**}**

**private void btnAddJobActionPerformed(java.awt.event.ActionEvent evt) {**

**if(validateInputs()==false){**

**JOptionPane.showMessageDialog(null,"Please fill all the fields","Error!",JOptionPane.ERROR\_MESSAGE);**

**return;**

**}**

**JobPojo job=new JobPojo();**

**job.setJobId(lblJobId.getText().trim());**

**job.setJobTitle(txtJobTitle.getText().trim());**

**job.setHrId(CurrentUser.getId());**

**job.setTags(txtSkill1.getText().trim()+","+txtSkill2.getText().trim()+","+txtSkill3.getText().trim()+","+txtSkill4.getText().trim());**

**job.setStatus(0);**

**try{**

**boolean result=JobDAO.addNewJob(job);**

**if(result){**

**JOptionPane.showMessageDialog(null,"Job Added Successfully!","Success!",JOptionPane.INFORMATION\_MESSAGE);**

**clearText();**

**loadData();**

**return;**

**}**

**JOptionPane.showMessageDialog(null,"Could Not Add The Job!","Failure!",JOptionPane.ERROR\_MESSAGE);**

**}**

**catch(SQLException ex){**

**JOptionPane.showMessageDialog(null,"DB Error","Error",JOptionPane.ERROR\_MESSAGE);**

**ex.printStackTrace();**

**}**

**}**

**private void clearText(){**

**txtJobTitle.setText("");**

**txtSkill1.setText("");**

**txtSkill2.setText("");**

**txtSkill3.setText("");**

**txtSkill4.setText("");**

**}**

**}**