**WRITING THE CODE FOR BUTTON Button.png IN AddEmployeeFrame**

1. When the user will click the **Add Emp** button then it will add the record of new employee in the **Employee** table.

2. Following are it's important points:

**a. It will first validate whether all the data has been properly filled all or not.**

**b. If not , then it will generate an error message and return.**

**c. Otherwise , it will create an Employee pojo object , fill all the values in it and pass it to the method addEmployee( ) of the EmployeeDAO**

**d. If the method addEmployee( ) returned true then it will display the message "Record inserted successfully" , otherwise it will display an error message.**

**f. It will also handle NumberFormatException as well as any SQLException that will be thrown by the method addEmployee( )**

3. Based upon the above facts following is it's code:

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

***if(validateInput()==false){***

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

***return;***

***}***

***try***

***{***

***int eno=Integer.parseInt(txtEmpNo.getText());***

***String ename=txtEmpName.getText();***

***double sal=Double.parseDouble(txtSal.getText());***

***Employee e=new Employee();***

***e.setEmpNo(eno);***

***e.setEname(ename);***

***e.setSal(sal);***

***boolean result=EmpDAO.addEmployee(e);***

***if(result==true)***

***JOptionPane.showMessageDialog(null,"Record Successfully Added To The Database!!", "Success!!", JOptionPane.INFORMATION\_MESSAGE);***

***else***

***JOptionPane.showMessageDialog(null,"Could Not Add Employee To The Database!!", "Error!!", JOptionPane.ERROR\_MESSAGE);***

***}***

***catch(NumberFormatException ex){***

***JOptionPane.showMessageDialog(null,"Please provide numeric input for eno and sal!!", "Error!!", JOptionPane.ERROR\_MESSAGE);***

***}***

***catch(SQLException ex){***

***JOptionPane.showMessageDialog(null,"Problem In The Database!!", "Error!!", JOptionPane.ERROR\_MESSAGE);***

***System.out.println("SQLException is "+ex);***

***}***

***}***

**WRITING THE CODE FOR BUTTON Button.png IN SearchEmployeeFrame**

1. When the user will click the **Search** button then it will find and display the record of the employee from the **Employee** table

2. Following are it's important points:

**a. It will first validate whether employee number has been given or not.**

**b. If not , then it will generate an error message and return.**

**c. Otherwise , it will call the method findEmployeeById( ) method of EmployeeDAO passing it the Employee Number as argument**

**d. If the method findEmployeeById( ) returned Employee pojo object then it will display the details of the employee i.e. his/her name and salary , otherwise it will display the message "record not found".**

**e. It will also handle NumberFormatException as well as any SQLException that will be thrown by the method findEmployeeById( )**

3. Based upon the above facts following is it's code:

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

***String str=txtEmpNo.getText();***

***if(str.isEmpty()){***

***JOptionPane.showMessageDialog(null,"Please fill empno", "Empty Values!!", JOptionPane.ERROR\_MESSAGE);***

***return;***

***}***

***try***

***{***

***int eno=Integer.parseInt(str);***

***Employee e=EmpDAO.findEmployeeById(eno);***

***if(e==null)***

***JOptionPane.showMessageDialog(null,"No Record Of EmpNo "+eno+" Present", "Not Found!!", JOptionPane.INFORMATION\_MESSAGE);***

***else***

***{***

***txtEmpName.setText(e.getEname());***

***txtSal.setText(String.valueOf(e.getSal()));***

***}***

***}***

***catch(NumberFormatException ex){***

***JOptionPane.showMessageDialog(null,"Please provide numeric input for eno!!", "Error!!", JOptionPane.ERROR\_MESSAGE);***

***}***

***catch(SQLException ex){***

***JOptionPane.showMessageDialog(null,"Problem In The Database!!", "Error!!", JOptionPane.ERROR\_MESSAGE);***

***System.out.println("SQLException is "+ex);***

***}***

***}***

**WRITING THE CODE FOR BUTTON Button.png IN ViewAllEmployeeFrame**

1. When the user will click the **Show All Records** button then it will pull and display the records of all the Employees from the **Employees** table

2. Following are it's important points:

**a. It will call the method getAllEmployees( ) of the EmployeeDAO class**

**b. It will then check whether the list returned by the method getAllEmployees( ) is empty or not. If it is empty then it will display the message "No record found"**

**c. Otherwise , it will iterate over the list of Employee objects in the list and display each object's data in the Text Area**

**d. It will also handle SQLException that will be thrown by the method getAllEmployees( )**

3. Based upon the above facts following is it's code:

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

***try***

***{***

***ArrayList<Employee> empList=EmpDAO.getAllEmployees();***

***if(empList.size()==0){***

***JOptionPane.showMessageDialog(null,"No Employee Present!!", "Empty Values!!", JOptionPane.ERROR\_MESSAGE);***

***return;***

***}***

***String str=new String();***

***txtAllEmp.setText("");***

***str="EMPNO \t ENAME \t SALARY\n\n";***

***for(Employee e: empList){***

***str=str+e.getEmpNo()+"\t"+e.getEname()+"\t"+e.getSal()+"\n";***

***}***

***txtAllEmp.setText(str);***

***}***

***catch(SQLException ex){***

***JOptionPane.showMessageDialog(null,"Problem In The Database!!", "Error!!", JOptionPane.ERROR\_MESSAGE);***

***System.out.println("SQLException is "+ex);***

***}***

***}***

Based upon the above understanding , students are advised to implement the functionality of remaining 2 Frames i.e **UpdateEmployeeFrame** and **DeleteEmployeeFrame** themselves.