Java Programming Open Ended

A Open Ended Experiment Submitted to

Amity University, Uttar Pradesh

in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in

Computer Science & Engineering

STUDENT INFORMATION SYSTEM

Student Name : Ayush Agarwal	Faculty Name: Mr. Ankur Chaudhary
Enrollment No: A2305216324	Designation : Asst. Professor

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

AMITY SCHOOL OF ENGINEERING AND TECHNOLOGY

AMITY UNIVERSITY UTTAR PRADESH, NOIDA (U.P.)

APRIL, 2018

/home/lucifer/NetBeansProjects/sis/src/sis/Sis.java

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package sis;

import sis.gui.MainWindow;

/**
 *
 * @author lucifer
 */
public class Sis {
    public MainWindow mainAppWindow;
    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here
        Sis s = new Sis();
        s.init();
    }
    public void init(){
        mainAppWindow = new MainWindow();
    }
}
```

/home/lucifer/NetBeansProjects/sis/src/sis/config.java

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package sis;

/**
 *
 * @author lucifer
 */
public class config {
    final public String applicationTitle = "Student Information System";

    final public int WindowHeight = 400;
    final public int WindowWidth = 400;

    final public boolean isResizable = false;

    final public String DatabaseName = "sis";
    final public String DatabaseUser = "postgres";
    final public String DatabasePass = "password";
    final public String DatabaseHost = "localhost";
    final public String DatabasePort = "5432";

}
```

/home/lucifer/NetBeansProjects/sis/src/sis/gui/views.java

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package sis.gui;

import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.util.ArrayList;
import java.util.Dictionary;
import java.util.Hashtable;
import java.util.logging.Level;
import java.util.logging.Logger;
import sis.config;
import sis.utils.Exceptions.*;
import sis.utils.studentUtils;
import sis.utils.validations;
/**
 *
 * @author lucifer
 */
class StudentView extends JFrame{
    config c = new config();
    SpringLayout layout = new SpringLayout();

    final protected String nameFieldText = "Enter Student Name";
    final protected String enrollFieldText = "Enter Enrollment";
    final protected String dobFieldText = "DD/MM/YYYY";

    public StudentView(){
        setTitle("Add Student");
        setSize(400, 200);
        setResizable(c.isResizable);
        setLayout(layout);
        setJMenuBar(new exitOnlyMenuBar(this).generateMenu());
        initializeStudentFields();
        setVisible(true);
    }

    protected void initializeStudentFields(){
        Frame parent = this;
        JPanel p = new JPanel(new SpringLayout());
        p.setBackground(Color.white);
        JLabel name = new JLabel("Student Name: ");
        JLabel enroll =  new JLabel("Enrollment: ");
        JLabel dob = new JLabel("Date of Birth");

        JTextField nameField = new JTextField(nameFieldText);
        nameField.setName("name");
        JTextField enrollField = new JTextField(enrollFieldText);
        enrollField.setName("enroll");
        JTextField dobField = new JTextField(dobFieldText);
        dobField.setName("dob");

        JButton submit = new JButton("Submit");
        JButton reset = new JButton("Reset");

        name.setLabelFor(nameField);
        enroll.setLabelFor(enrollField);
        dob.setLabelFor(dobField);

        submit.addActionListener(new ActionListener(){
            @Override
            public void actionPerformed(ActionEvent e){
                validations v = new validations();
                ErrorBox eb = new ErrorBox();
                if(!v.validate(nameField.getText(), "name")){
                    eb.addMsg(v.getMsg());
                }
                if(!v.validate(enrollField.getText(), "enroll")){
                    eb.addMsg(v.getMsg());
                }
                if(!v.validate(dobField.getText(), "dob")){
                    eb.addMsg(v.getMsg());
                }
                if(eb.is_set){
                    eb.showBox();
                }else{
                    eb.clear();
                    studentUtils s = new studentUtils();

                    try{
                        s.createNewStudent(
                            nameField.getText(),
                            enrollField.getText(),
                            dobField.getText()
                        );
                        eb.setType("success");
                        eb.addMsg("Added Successfully!");
                        eb.showBox();
                        parent.dispose();
                    }catch(EnrollmentExistsException error){
                        eb.addMsg("Enrollment Exists in Database!");
                        eb.showBox();
                    }catch(DateParsingException error){
                        eb.addMsg("Unable to Process Date!");
                        eb.showBox();
                    }catch(DatabaseException error){
                        eb.addMsg("Database Error Occured : " + error.msg);
                        eb.showBox();
                    }
                }
            }
        });

        reset.addActionListener(new ActionListener(){
            @Override
            public void actionPerformed(ActionEvent e){
                nameField.setText(nameFieldText);
                enrollField.setText(enrollFieldText);
                dobField.setText(dobFieldText);
            }
        });

        p.add(name);
        p.add(nameField);
        p.add(enroll);
        p.add(enrollField);
        p.add(dob);
        p.add(dobField);
        p.add(reset);
        p.add(submit);

        SpringUtilities.makeCompactGrid(p, 4, 2, 6, 6, 5, 5);

        setContentPane(p);
    }
}

class StudentsListView extends JFrame{
    config c = new config();
    JPanel mainPanel = new JPanel();
    JPanel headerPanel;
    JScrollPane studentListPanel = new JScrollPane();

    final String[] tableColumns = {
            "id",
            "Name",
            "Enrollment Number",
            "Date of Birth"
        };

    public StudentsListView(){
        setTitle("Students List");
        setSize(c.WindowHeight, c.WindowWidth);
        setResizable(c.isResizable);
        setJMenuBar(new studentListMenuBar(this).generateMenu());
        setVisible(true);
        mainPanel.setLayout(new GridLayout(2,1));
        initializeLayout();
        studentUtils s = new studentUtils();
        try {
            generateStudentList(s.getStudents());
        } catch (DatabaseException ex) {
            System.out.println(ex.getMessage());
        }
    }

    private void initializeLayout(){
        headerPanel = generateHeaderPanel();
        mainPanel.add(headerPanel);
        mainPanel.add(studentListPanel);
        add(mainPanel);
    }

    private JPanel generateHeaderPanel(){
        JPanel j = new JPanel();

        JLabel searchNameText = new JLabel("Search By Name:");
        searchNameText.setPreferredSize(new Dimension(c.WindowHeight, 20));
        JTextField searchNameField = new JTextField("");
        searchNameField.setPreferredSize(new Dimension(c.WindowHeight, 20));
        searchNameText.setLabelFor(searchNameField);


        JLabel searchEnrollText = new JLabel("Search By Enrollment:");
        searchEnrollText.setPreferredSize(new Dimension(c.WindowHeight, 20));
        JTextField searchEnrollField = new JTextField("");
        searchEnrollField.setPreferredSize(new Dimension(c.WindowHeight, 20));
        searchEnrollText.setLabelFor(searchEnrollField);

        JButton searchBtn = new JButton("Search");

        searchBtn.addActionListener(new ActionListener(){
           @Override
           public void actionPerformed(ActionEvent e){
               ErrorBox eb = new ErrorBox();
               if(searchNameField.getText().length() == 0 && searchEnrollField.getText().length() == 0){
                   eb.addMsg("Please provide either name or enrollment!");
               }else{
                   if(searchEnrollField.getText().length() != 0){
                       validations v = new validations();
                       if(!v.validate(searchEnrollField.getText(),"enroll")){
                           eb.addMsg("Enter a valid enrollment number!" + v.getMsg());
                       }else{
                           studentUtils s = new studentUtils();
                           try {
                               Dictionary res = s.searchStudentByEnrollment(searchEnrollField.getText());
                               generateStudentList(res);
                           } catch (DatabaseException ex) {
                               eb.addMsg("Database Error: " + ex.msg);
                           } catch (EnrollmentNotFoundException ex) {
                               eb.addMsg("Enrollment Not Found!");
                           }
                       }
                   }else{
                       validations v = new validations();
                       if(!v.validate(searchNameField.getText(),"name_not_full")){
                           eb.addMsg("Enter a valid name!" + v.getMsg());
                       }else{
                           studentUtils s = new studentUtils();
                           try {
                               ArrayList<Dictionary> res = s.searchStudentsByName(searchNameField.getText());
                               generateStudentList(res);
                           } catch (DatabaseException ex) {
                               eb.addMsg("Database Error: " + ex.msg);
                           }
                       }
                   }
               }
               if(eb.is_set)
```

```
                   eb.showBox();
           }
        });

        j.add(searchNameText);
        j.add(searchNameField);
        j.add(searchEnrollText);
        j.add(searchEnrollField);
        j.add(searchBtn);
        j.setSize(c.WindowWidth, 150);
        j.setLayout(new GridLayout(0,1));
        return j;
    }

    private void generateStudentList(Dictionary student){

        String[][] data = {
            {
                student.get("id").toString(),
                student.get("name").toString(),
                student.get("enroll").toString(),
                student.get("dob").toString(),
            },
        };

        JTable studentTable = new JTable(data, tableColumns);

        JScrollPane jp = new JScrollPane(studentTable);
        updateStudentListPanel(jp);

    }

    private void generateStudentList(ArrayList<Dictionary> students){

        String[][] data = new String[students.size()][4];

        int i=0;
        for(Dictionary student:students){
            data[i][0] = student.get("id").toString();
            data[i][1] = student.get("name").toString();
            data[i][2] = student.get("enroll").toString();
            data[i][3] = student.get("dob").toString();
            i++;
        }

        JTable studentTable = new JTable(data, tableColumns);

        JScrollPane jp = new JScrollPane(studentTable);
        updateStudentListPanel(jp);

    }

    private void updateStudentListPanel(JScrollPane jp){
        mainPanel.remove(studentListPanel);
        studentListPanel = jp;
        studentListPanel.setBounds(0, 170, c.WindowWidth, c.WindowHeight-170);
        mainPanel.add(studentListPanel);
        //mainPanel.revalidate();

    }

}

class ErrorBox{
    public boolean is_set;
    String[] msg;
    JPanel jp;
    JDialog jd;
    private int currentHeight = 50;
    private Color textColor = Color.white;
    ErrorBox(){
        init();
        jp.setBackground(Color.red);
        textColor = Color.white;
    }
    ErrorBox(String type){
        init();
        setType(type);
    }
    private void init(){
        is_set = false;
        jp = new JPanel();
        jp.setLayout(new FlowLayout());
    }
    public void setType(String type){
        if(type == "success"){
            jp.setBackground(Color.GREEN);
            textColor = Color.BLACK;
        }
    }
    public void showBox(){
        jd = new JDialog();
        jd.add(jp);
        jd.setVisible(true);
        jd.setSize(400,currentHeight);
    }
    public void addMsg(String msg){
        is_set = true;
        JLabel j = new JLabel(msg);
        j.setForeground(textColor);
        currentHeight += 50;
        jp.add(j);
    }
    public void clear(){
       jp.removeAll();
    }
}

class DeleteStudentView extends JFrame{


    DeleteStudentView(){
        setTitle("Delete a Student");
        setSize(300, 100);
        setVisible(true);

        setLayout(new GridLayout(1,1));

        JPanel j = new JPanel();
        j.setBackground(Color.white);
        j.setLayout(new GridLayout(3,1));
        j.setSize(300, 100);

        JLabel title = new JLabel("Enter Enrollment Number:");
        JTextField enrollField = new JTextField("");
        JButton submit = new JButton("Search");

        submit.addActionListener(new ActionListener(){

            @Override
            public void actionPerformed(ActionEvent e){
                validations v = new validations();
                if(!v.validate(enrollField.getText(), "enroll")){
                    ErrorBox eb = new ErrorBox();
                    eb.addMsg("Please enter a valid enrollment number!");
                    eb.showBox();

                }else{
                    studentUtils s = new studentUtils();
                    try{
                        Dictionary student =
s.searchStudentByEnrollment(enrollField.getText());

                        remove(j);
                        add(getDeleteStudentPanel(student));
                        setSize(300, 300);

                    }catch(EnrollmentNotFoundException ex){
                        ErrorBox eb = new ErrorBox();
                        eb.addMsg("Enrollment Not Found! Please Recheck!");
                        eb.showBox();
                    } catch (DatabaseException ex) {
                        ErrorBox eb = new ErrorBox();
                        eb.addMsg("Database Error Occured!");
                        eb.showBox();
                    }
                }
            }

        });

        j.add(title);
        j.add(enrollField);
        j.add(submit);

        add(j);

    }

    JPanel getDeleteStudentPanel(Dictionary student){
        JPanel jp = new JPanel();

        jp.setLayout(new GridLayout(5,1));
        jp.setSize(300, 300);

        jp.add(new JLabel("Enrollment: " + student.get("enroll").toString()));
        jp.add(new JLabel("Name: " + student.get("name").toString()));
        jp.add(new JLabel("Date Of Birth: " + student.get("dob").toString()));

        JButton deleteBtn = new JButton("Delete");
        JButton cancelBtn = new JButton("Cancel");
        jp.add(deleteBtn);
        jp.add(cancelBtn);
        deleteBtn.addActionListener(new ActionListener(){
            @Override
            public void actionPerformed(ActionEvent e){
                studentUtils s = new studentUtils();
                try{

if(s.deleteStudentByEnrollment(student.get("enroll").toString())){
                        ErrorBox eb = new ErrorBox("success");
                        eb.addMsg("Deleted Successfully!");
                        eb.showBox();
                        dispose();
                    }
                }catch(EnrollmentNotFoundException ex){
                    ErrorBox eb = new ErrorBox();
                    eb.addMsg("Enrollment Number Doesn’t exist in database!");
                    eb.showBox();
                }catch(DatabaseException ex){
                    ErrorBox eb = new ErrorBox();
                    eb.addMsg("Database Error Occured: " + ex.getMessage());
                    eb.showBox();
                }
            }
        });
        cancelBtn.addActionListener(new ActionListener(){
            @Override
            public void actionPerformed(ActionEvent e){
                dispose();
            }
        });
        jp.setVisible(true);
        return jp;
    }

}

class EditStudentView extends JFrame{


    EditStudentView(){
        setTitle("Edit a Student");

        setSize(300, 100);
        setVisible(true);

        setLayout(new GridLayout(1,1));

        JPanel j = new JPanel();
        j.setBackground(Color.white);
        j.setLayout(new GridLayout(3,1));
        j.setSize(300, 100);

        JLabel title = new JLabel("Enter Enrollment Number:");
        JTextField enrollField = new JTextField("");
        JButton submit = new JButton("Search");

        submit.addActionListener(new ActionListener(){

            @Override
            public void actionPerformed(ActionEvent e){
                validations v = new validations();
                if(!v.validate(enrollField.getText(), "enroll")){
                    ErrorBox eb = new ErrorBox();
                    eb.addMsg("Please enter a valid enrollment number!");
                    eb.showBox();

                }else{
                    studentUtils s = new studentUtils();
                    try{
                        Dictionary student = s.searchStudentByEnrollment(enrollField.getText());
                        remove(j);
                        add(getEditStudentPanel(student));
                        setSize(400, 200);

                    }catch(EnrollmentNotFoundException ex){
                        ErrorBox eb = new ErrorBox();
                        eb.addMsg("Enrollment Not Found! Please Recheck!");
                        eb.showBox();
                    } catch (DatabaseException ex) {
                        ErrorBox eb = new ErrorBox();
                        eb.addMsg("Database Error Occured!");
                        eb.showBox();
                    }
                }
            }

        });

        j.add(title);
        j.add(enrollField);
        j.add(submit);

        add(j);

    }

    JPanel getEditStudentPanel(Dictionary student){
        JPanel jp = new JPanel();
        jp.setLayout(new GridLayout(4,2));
        jp.setSize(400, 200);

        jp.add(new JLabel("Enrollment"));
```

```java
        JTextField enrollField = new JTextField(student.get("enroll").toString());
        jp.add(enrollField);

        jp.add(new JLabel("Name"));
        JTextField nameField = new JTextField(student.get("name").toString());
        jp.add(nameField);

        jp.add(new JLabel("Date Of Birth"));
        JTextField dobField = new JTextField(student.get("dob").toString());
        jp.add(dobField);

        JButton submitBtn = new JButton("Submit");
        JButton cancelBtn = new JButton("Cancel");

        jp.add(submitBtn);
        jp.add(cancelBtn);

        submitBtn.addActionListener(new ActionListener(){
            @Override
            public void actionPerformed(ActionEvent e){
                validations v = new validations();
                ErrorBox eb = new ErrorBox();
                if(!v.validate(nameField.getText(), "name")){
                    eb.addMsg(v.getMsg());
                }
                if(!v.validate(enrollField.getText(), "enroll")){
                    eb.addMsg(v.getMsg());
                }
                if(!v.validate(dobField.getText(), "dob-db")){
                    eb.addMsg(v.getMsg());
                }
                if(eb.is_set){
                    eb.showBox();
                }else{
                    eb.clear();
                    studentUtils s = new studentUtils();

                    Dictionary studentNew = new Hashtable();
                    studentNew.put("name", nameField.getText());
                    studentNew.put("enroll", enrollField.getText());
                    studentNew.put("dob", dobField.getText());
                    studentNew.put("id", student.get("id"));
                    try{
                        if(s.editStudent(student, studentNew)){
                            ErrorBox ebS = new ErrorBox("success");
                            ebS.addMsg("Updated Successfully!");
                            ebS.showBox();
                            dispose();
                        }
                    }catch(DatabaseException ex){
                        eb.addMsg("Database Error Occured: " + ex.getMessage());
                        eb.showBox();
                    } catch (EnrollmentExistsException ex) {
                        eb.addMsg("Enrollment Already in Database!");
                        eb.showBox();
                    }
                }
            }
        });

        cancelBtn.addActionListener(new ActionListener(){
            @Override
            public void actionPerformed(ActionEvent e){
                dispose();
            }
        });

        return jp;
    }

}
```

/home/lucifer/NetBeansProjects/sis/src/sis/gui/SpringUtilities.java

```
/*
 * Copyright (c) 1995, 2008, Oracle and/or its affiliates. All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 *
 *   - Redistributions of source code must retain the above copyright
 *     notice, this list of conditions and the following disclaimer.
 *
 *   - Redistributions in binary form must reproduce the above copyright
 *     notice, this list of conditions and the following disclaimer in the
 *     documentation and/or other materials provided with the distribution.
 *
 *   - Neither the name of Oracle or the names of its
 *     contributors may be used to endorse or promote products derived
 *     from this software without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
 * IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,
 * THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
 * PURPOSE ARE DISCLAIMED.  IN NO EVENT SHALL THE COPYRIGHT OWNER OR
 * CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
 * EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
 * PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
 * PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
 * LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
 * NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
 * SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 */

package sis.gui;

import javax.swing.*;
import javax.swing.SpringLayout;
import java.awt.*;

/**
 * A 1.4 file that provides utility methods for
 * creating form- or grid-style layouts with SpringLayout.
 * These utilities are used by several programs, such as
 * SpringBox and SpringCompactGrid.
 */
public class SpringUtilities {
    /**
     * A debugging utility that prints to stdout the component’s
     * minimum, preferred, and maximum sizes.
     */
    public static void printSizes(Component c) {
        System.out.println("minimumSize = " + c.getMinimumSize());
        System.out.println("preferredSize = " + c.getPreferredSize());
        System.out.println("maximumSize = " + c.getMaximumSize());
    }

    /**
     * Aligns the first <code>rows</code> * <code>cols</code>
     * components of <code>parent</code> in
     * a grid. Each component is as big as the maximum
     * preferred width and height of the components.
     * The parent is made just big enough to fit them all.
     *
     * @param rows number of rows
     * @param cols number of columns
     * @param initialX x location to start the grid at
     * @param initialY y location to start the grid at
     * @param xPad x padding between cells
     * @param yPad y padding between cells
     */
    public static void makeGrid(Container parent,
                                int rows, int cols,
                                int initialX, int initialY,
                                int xPad, int yPad) {
        SpringLayout layout;
        try {
            layout = (SpringLayout)parent.getLayout();
        } catch (ClassCastException exc) {
            System.err.println("The first argument to makeGrid must use SpringLayout.");
            return;
        }

        Spring xPadSpring = Spring.constant(xPad);
        Spring yPadSpring = Spring.constant(yPad);
        Spring initialXSpring = Spring.constant(initialX);
        Spring initialYSpring = Spring.constant(initialY);
        int max = rows * cols;

        //Calculate Springs that are the max of the width/height so that all
        //cells have the same size.
        Spring maxWidthSpring = layout.getConstraints(parent.getComponent(0)).
                                    getWidth();
        Spring maxHeightSpring = layout.getConstraints(parent.getComponent(0)).
                                    getWidth();
        for (int i = 1; i < max; i++) {
            SpringLayout.Constraints cons = layout.getConstraints(
                                            parent.getComponent(i));

            maxWidthSpring = Spring.max(maxWidthSpring, cons.getWidth());
            maxHeightSpring = Spring.max(maxHeightSpring, cons.getHeight());
        }

        //Apply the new width/height Spring. This forces all the
        //components to have the same size.
        for (int i = 0; i < max; i++) {
            SpringLayout.Constraints cons = layout.getConstraints(
                                            parent.getComponent(i));

            cons.setWidth(maxWidthSpring);
            cons.setHeight(maxHeightSpring);
        }

        //Then adjust the x/y constraints of all the cells so that they
        //are aligned in a grid.
        SpringLayout.Constraints lastCons = null;
        SpringLayout.Constraints lastRowCons = null;
        for (int i = 0; i < max; i++) {
            SpringLayout.Constraints cons = layout.getConstraints(
                                                 parent.getComponent(i));
            if (i % cols == 0) { //start of new row
                lastRowCons = lastCons;
                cons.setX(initialXSpring);
            } else { //x position depends on previous component
                cons.setX(Spring.sum(lastCons.getConstraint(SpringLayout.EAST),
                                     xPadSpring));
            }

            if (i / cols == 0) { //first row
                cons.setY(initialYSpring);
            } else { //y position depends on previous row

cons.setY(Spring.sum(lastRowCons.getConstraint(SpringLayout.SOUTH),
                                     yPadSpring));
            }
            lastCons = cons;
        }

        //Set the parent's size.
        SpringLayout.Constraints pCons = layout.getConstraints(parent);
        pCons.setConstraint(SpringLayout.SOUTH,
                            Spring.sum(
                                Spring.constant(yPad),
                                lastCons.getConstraint(SpringLayout.SOUTH)));
        pCons.setConstraint(SpringLayout.EAST,
                            Spring.sum(
                                Spring.constant(xPad),
                                lastCons.getConstraint(SpringLayout.EAST)));
    }

    /* Used by makeCompactGrid. */
    private static SpringLayout.Constraints getConstraintsForCell(
                                                int row, int col,
                                                Container parent,
                                                int cols) {
        SpringLayout layout = (SpringLayout) parent.getLayout();
        Component c = parent.getComponent(row * cols + col);
        return layout.getConstraints(c);
    }

    /**
     * Aligns the first <code>rows</code> * <code>cols</code>
     * components of <code>parent</code> in
     * a grid. Each component in a column is as wide as the maximum
     * preferred width of the components in that column;
     * height is similarly determined for each row.
     * The parent is made just big enough to fit them all.
     *
     * @param rows number of rows
     * @param cols number of columns
     * @param initialX x location to start the grid at
     * @param initialY y location to start the grid at
     * @param xPad x padding between cells
     * @param yPad y padding between cells
     */
    public static void makeCompactGrid(Container parent,
                                       int rows, int cols,
                                       int initialX, int initialY,
                                       int xPad, int yPad) {
        SpringLayout layout;
        try {
            layout = (SpringLayout)parent.getLayout();
        } catch (ClassCastException exc) {
            System.err.println("The first argument to makeCompactGrid must use SpringLayout.");
            return;
        }

        //Align all cells in each column and make them the same width.
        Spring x = Spring.constant(initialX);
        for (int c = 0; c < cols; c++) {
            Spring width = Spring.constant(0);
            for (int r = 0; r < rows; r++) {
                width = Spring.max(width,
                                   getConstraintsForCell(r, c, parent, cols).
                                       getWidth());
            }
            for (int r = 0; r < rows; r++) {
                SpringLayout.Constraints constraints =
                        getConstraintsForCell(r, c, parent, cols);
                constraints.setX(x);
                constraints.setWidth(width);
            }
            x = Spring.sum(x, Spring.sum(width, Spring.constant(xPad)));
        }

        //Align all cells in each row and make them the same height.
        Spring y = Spring.constant(initialY);
        for (int r = 0; r < rows; r++) {
            Spring height = Spring.constant(0);
            for (int c = 0; c < cols; c++) {
                height = Spring.max(height,
                                    getConstraintsForCell(r, c, parent, cols).
                                        getHeight());
            }
            for (int c = 0; c < cols; c++) {
                SpringLayout.Constraints constraints =
                        getConstraintsForCell(r, c, parent, cols);
                constraints.setY(y);
                constraints.setHeight(height);
            }
            y = Spring.sum(y, Spring.sum(height, Spring.constant(yPad)));
        }

        //Set the parent’s size.
        SpringLayout.Constraints pCons = layout.getConstraints(parent);
        pCons.setConstraint(SpringLayout.SOUTH, y);
        pCons.setConstraint(SpringLayout.EAST, x);
    }
}
```

/home/lucifer/NetBeansProjects/sis/src/sis/gui/listeners.java

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package sis.gui;
import java.awt.*;
import java.awt.event.*;
/**
 *
 * @author lucifer
 */
class ExitListener implements ActionListener{
    private Frame parentFrame = null;
    ExitListener(Frame parent){
        parentFrame = parent;
    }
    ExitListener(){}
    @Override
    public void actionPerformed(ActionEvent e){
        if (parentFrame == null)
            System.exit(0);
        else
            parentFrame.dispose();
    }
}

class addStudentListener implements ActionListener{

    @Override
    public void actionPerformed(ActionEvent e){
        StudentView sv = new StudentView();
    }
}

class studentListListener implements ActionListener{

    @Override
    public void actionPerformed(ActionEvent e){
        StudentsListView sv = new StudentsListView();
    }
}

class EditStudentListener implements ActionListener{

    @Override
    public void actionPerformed(ActionEvent e){
        EditStudentView ev = new EditStudentView();
    }
}

class DeleteStudentListener implements ActionListener{
    @Override
    public void actionPerformed(ActionEvent e){
        DeleteStudentView dv = new DeleteStudentView();
    }
}
```

/home/lucifer/NetBeansProjects/sis/src/sis/config.java

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package sis;

/**
 *
 * @author lucifer
 */
public class config {
    final public String applicationTitle = "Student Information System";

    final public int WindowHeight = 400;
    final public int WindowWidth = 400;

    final public boolean isResizable = false;

    final public String DatabaseName = "sis";
    final public String DatabaseUser = "postgres";
    final public String DatabasePass = "@yush7592";
    final public String DatabaseHost = "localhost";
    final public String DatabasePort = "5432";

}
```

/home/lucifer/NetBeansProjects/sis/src/sis/gui/listeners.java

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package sis.gui;
import java.awt.*;
import java.awt.event.*;
/**
 *
 * @author lucifer
 */
class ExitListener implements ActionListener{
    private Frame parentFrame = null;
    ExitListener(Frame parent){
        parentFrame = parent;
    }
    ExitListener(){}
    @Override
    public void actionPerformed(ActionEvent e){
        if (parentFrame == null)
            System.exit(0);
        else
            parentFrame.dispose();
    }
}

class addStudentListener implements ActionListener{

    @Override
    public void actionPerformed(ActionEvent e){
        StudentView sv = new StudentView();
    }
}

class studentListListener implements ActionListener{

    @Override
    public void actionPerformed(ActionEvent e){
        StudentsListView sv = new StudentsListView();
    }
}

class EditStudentListener implements ActionListener{

    @Override
    public void actionPerformed(ActionEvent e){
        EditStudentView ev = new EditStudentView();
    }
}

class DeleteStudentListener implements ActionListener{
    @Override
    public void actionPerformed(ActionEvent e){
        DeleteStudentView dv = new DeleteStudentView();
    }
}
```

/home/lucifer/NetBeansProjects/sis/src/sis/utils/validations.java

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package sis.utils;

import java.util.regex.*;
/**
 *
 * @author lucifer
 */

public class validations {
    String msg = new String("Unknown");

    Pattern namePattern = Pattern.compile("[a-zA-Z]{1,}\\s[a-zA-Z]{1,}");
    Pattern nameNotFullPattern = Pattern.compile("^\\s*[a-zA-Z]{1,}(\\s[a-zA-Z]{0,}){0,1}\\s*$");
    Pattern enrollPattern = Pattern.compile("[A-Z][0-9]{10,11}");
    Pattern datePattern = Pattern.compile("^\\s*(3[01]|[12][0-9]|0?[1-9])/(1[012]|0?[1-9])/((?:19|20)\\d{2})\\s*$");
    Pattern dateDbPattern = Pattern.compile("^\\s*((?:19|20)\\d{2})-(1[012]|0?[1-9])-(3[01]|[12][0-9]|0?[1-9])\\s*$");
    public boolean validate(String str, String type){
        if(type == "name"){
            if(str.length() == 0){
                msg = new String("Name is Empty");
                return false;
            }
            if(!namePattern.matcher(str).matches()){
                msg = new String("Only Alphabets and a single space is allowed!");
                return false;
            }
        }
        if(type == "name_not_full"){
            if(str.length() == 0){
                msg = new String("Name is Empty");
                return false;
            }
            if(!nameNotFullPattern.matcher(str).matches()){
                msg = new String("Only Alphabets and a space is allowed!");
                return false;
            }
        }
        if(type == "enroll"){
            if(str.length() == 0){
                msg = new String("Enrollment is Empty");
                return false;
            }
            if(!enrollPattern.matcher(str).matches()){
                msg = new String("Please enter a valid enrollment number");
                return false;
            }
        }
        if(type == "dob"){
            if(str.length() == 0){
                msg = new String("DOB is Empty");
                return false;
            }
            if(!datePattern.matcher(str).matches()){
                msg = new String("Please enter a valid date!");
                return false;
            }
        }
        if(type == "dob-db"){
            if(str.length() == 0){
                msg = new String("DOB is Empty");
                return false;
            }
            if(!dateDbPattern.matcher(str).matches()){
                msg = new String("Please enter a valid date!");
                return false;
            }
        }

        return true;
    }

    public String getMsg(){
        return msg;
    }
}
```

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package sis.utils;

import java.text.ParseException;
import java.util.Date;
import java.util.Dictionary;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Hashtable;

import sis.utils.Exceptions.*;
/**
 *
 * @author lucifer
 */
public class studentUtils {

    public boolean createNewStudent(String name, String enroll, String dob) throws EnrollmentExistsException, DateParsingException, DatabaseInitializationException, DatabaseException{
        StudentDb database = new StudentDb();
        try{
            database.connect();
            if(!database.enrollmentExists(enroll)){
                Date dateofbirth = new SimpleDateFormat("dd/MM/yyyy").parse(dob);
                database.addNewStudent(name, enroll, dateofbirth);
                return true;
            }else{
                    throw new EnrollmentExistsException();
            }
        }catch(ParseException e){
            throw new DateParsingException();
        }catch(DatabaseInitializationException e){
            throw e;
        }catch(DatabaseException e){
            throw e;
        }
    }

    public Dictionary searchStudentByEnrollment(String enroll) throws DatabaseException, EnrollmentNotFoundException{
        Dictionary student;
        StudentDb s = new StudentDb();
        try{
            s.connect();
            student = s.getStudentByEnrollment(enroll);
        }catch(EnrollmentNotFoundException e){
            throw e;
        }catch(DatabaseException e){
            throw e;
        }
        return student;
    }
    public ArrayList<Dictionary> getStudents() throws DatabaseException{
        ArrayList<Dictionary> students;
        StudentDb s = new StudentDb();
        try{
            s.connect();
            students = s.getStudents();
            return students;
        }catch(DatabaseException e){
            throw e;
        }
    }
    public ArrayList<Dictionary> searchStudentsByName(String name) throws DatabaseException{
        ArrayList<Dictionary> students;
        StudentDb s = new StudentDb();
        try{
            s.connect();
            students = s.getStudentsByName(name);
            return students;
        }catch(DatabaseException e){
            throw e;
        }
    }

    public boolean deleteStudentByEnrollment(String enroll) throws DatabaseException, EnrollmentNotFoundException{
        try{
            searchStudentByEnrollment(enroll);

            StudentDb s = new StudentDb();

            try{
                s.connect();
                if(s.deleteStudentByEnrollment(enroll)){
                    return true;
                }else{
                    return false;
                }
            }catch(DatabaseException ex){
                throw ex;
            }
        }catch(EnrollmentNotFoundException ex){
            throw ex;
        }
    }
    public boolean editStudent(Dictionary student, Dictionary studentNew) throws DatabaseException, EnrollmentExistsException{
        try{
            StudentDb s = new StudentDb();
            s.connect();
            s.editStudent(student, studentNew);
            return true;
        }catch(DatabaseException ex){
                throw ex;
        }
    }
```

```
}
```

/home/lucifer/NetBeansProjects/sis/src/sis/utils/db.java

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package sis.utils;

import java.sql.*;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Dictionary;
import java.util.Hashtable;
import java.util.logging.Level;
import java.util.logging.Logger;

import sis.config;
import sis.utils.Exceptions.*;
/**
 *
 * @author lucifer
 */
public class db {
    protected Connection con = null;
    protected config c = new config();

    private String CreateStudentTableSQL = "CREATE TABLE IF NOT EXISTS students("
            + "id SERIAL PRIMARY KEY,"
            + "name VARCHAR(255) NOT NULL,"
            + "enrollment VARCHAR(12) NOT NULL UNIQUE,"
            + "dob DATE NOT NULL"
            + ");";

    protected String AddNewStudentSQL = "INSERT INTO students(name, enrollment, dob) VALUES(?, ?, ?) RETURNING id;";

    public boolean connect() throws DatabaseException{
        if(isConnected()){
            return true;
        }
        try{
            con = DriverManager.getConnection(getConnectionString(), c.DatabaseUser, c.DatabasePass);
            System.out.println("Connected to Database!");
            try{
                initialize();
                return true;
            }catch(DatabaseInitializationException e){
                con.close();
                con =  null;
                throw e;
            }
        }catch(SQLException e){
            System.out.println(e.getMessage());
            throw new DatabaseException(e.getMessage());
        }
    }

    private String getConnectionString(){
        return "jdbc:postgresql://"+c.DatabaseHost+":"+c.DatabasePort+"/"+c.DatabaseName;
    }
    private boolean initialize() throws DatabaseInitializationException{
        if(isConnected()){
            try{
                PreparedStatement st = con.prepareStatement(CreateStudentTableSQL);
                st.execute();
                System.out.println("Database Initialized");
                return true;
            }catch(SQLException e){
                System.out.println("Unable to Initialize the database!");
                System.out.println(e.getMessage());
                throw new DatabaseInitializationException();
            }
        }
        return false;
    }

    public boolean isConnected(){
        if(con!=null)
            return true;
        else
            return false;
    }
}


class StudentDb extends db{

    public int addNewStudent(String name, String enroll, java.util.Date dob) throws DatabaseException{
        try{
            PreparedStatement st = con.prepareStatement(AddNewStudentSQL);
            st.setString(1, name);
            st.setString(2, enroll);
            st.setDate(3, new java.sql.Date(dob.getTime()));
            ResultSet res = st.executeQuery();
            res.next();
            int id = res.getInt("id");
            System.out.println("Inserted Student with id " + id);
            return id;
        }catch(SQLException e){
            System.out.println(e.getMessage());
            throw new DatabaseException(e.getMessage());
        }
    }

    public boolean enrollmentExists(String enroll) throws DatabaseException{
        try{
            PreparedStatement st = con.prepareStatement("SELECT COUNT(id) AS total FROM students WHERE enrollment=?");
            st.setString(1, enroll);
            ResultSet res = st.executeQuery();
            res.next();
            if(res.getInt(1) == 1)
                return true;
            else
                return false;
        }catch(SQLException e){
            throw new DatabaseException(e.getMessage());
        }
    }

    public Dictionary getStudentByEnrollment(String enroll) throws EnrollmentNotFoundException, DatabaseException{
        Dictionary student = new Hashtable();
        try{
            PreparedStatement st = con.prepareStatement("SELECT id, name, enrollment, dob FROM students WHERE enrollment = ? LIMIT 1;");
            st.setString(1, enroll);

            ResultSet res = st.executeQuery();
            if(res.next()){
                student.put("id", res.getString("id"));
                student.put("name", res.getString("name"));
                student.put("enroll", res.getString("enrollment"));
                student.put("dob", res.getString("dob"));
            }else{
                throw new EnrollmentNotFoundException();
            }
        }catch(SQLException e){
            throw new DatabaseException(e.getMessage());
        }
        return student;
    }

    public ArrayList<Dictionary> getStudents() throws DatabaseException{
        ArrayList<Dictionary> students = new ArrayList<Dictionary>();
        try{
            PreparedStatement st = con.prepareStatement("SELECT id, name, enrollment, dob FROM students;");

            ResultSet res = st.executeQuery();
            while(res.next()){
                Dictionary student = new Hashtable();
                student.put("id", res.getString("id"));
                student.put("name", res.getString("name"));
                student.put("enroll", res.getString("enrollment"));
                student.put("dob", res.getString("dob"));
                students.add(student);
            }
        }catch(SQLException e){
            throw new DatabaseException(e.getMessage());
        }
        return students;
    }
    public ArrayList<Dictionary> getStudentsByName(String name) throws DatabaseException{
        ArrayList<Dictionary> students = new ArrayList<Dictionary>();
        try{
            PreparedStatement st = con.prepareStatement("SELECT id, name, enrollment, dob FROM students WHERE LOWER(name) LIKE ?;");
            st.setString(1, "%"+name.toLowerCase()+"%");
            ResultSet res = st.executeQuery();
            while(res.next()){
                Dictionary student = new Hashtable();
                student.put("id", res.getString("id"));
                student.put("name", res.getString("name"));
                student.put("enroll", res.getString("enrollment"));
                student.put("dob", res.getString("dob"));
                students.add(student);
            }
        }catch(SQLException e){
            throw new DatabaseException(e.getMessage());
        }
        return students;
    }

    public boolean deleteStudentByEnrollment(String enroll) throws DatabaseException{
        try{
            PreparedStatement st = con.prepareStatement("DELETE FROM students WHERE enrollment=?;");
            st.setString(1, enroll);
            st.execute();
            try{
                getStudentByEnrollment(enroll);
                return false;
            } catch (EnrollmentNotFoundException ex) {
                return true;
            }
        }catch(SQLException e){
            throw new DatabaseException(e.getMessage());
        }
    }
    public boolean editStudent(Dictionary student, Dictionary studentNew) throws DatabaseException, EnrollmentExistsException{

        try{
            Dictionary currentData = getStudentByEnrollment(studentNew.get("enroll").toString());
            if(currentData.get("id").toString().equals(student.get("id"))){
                throw new EnrollmentNotFoundException();
            }else{
                throw new EnrollmentExistsException();
            }

        }catch(EnrollmentNotFoundException ex){
            try{
                PreparedStatement st = con.prepareStatement("UPDATE students SET name=?, enrollment=?, dob=? WHERE id=?;");

                SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");

                st.setString(1, studentNew.get("name").toString());
                st.setString(2, studentNew.get("enroll").toString());
                st.setDate(3, new
java.sql.Date(sdf.parse(studentNew.get("dob").toString()).getTime()));
                st.setInt(4, Integer.parseInt(studentNew.get("id").toString()));

                st.execute();
                return true;
            }catch(SQLException e){
                e.printStackTrace();
                throw new DatabaseException(e.getMessage());
            } catch (ParseException e) {
                return false;
            }
        }

    }
}
```

/home/lucifer/NetBeansProjects/sis/src/sis/utils/Exceptions/EnrollmentExistsException.java

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package sis.utils.Exceptions;
import java.lang.Exception;
/**
 *
 * @author lucifer
 */
public class EnrollmentExistsException extends Exception {

}
```

/home/lucifer/NetBeansProjects/sis/src/sis/utils/Exceptions/DateParsingException.java

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package sis.utils.Exceptions;

import java.lang.Exception;
/**
 *
 * @author lucifer
 */
public class DateParsingException extends Exception{

}
```

/home/lucifer/NetBeansProjects/sis/src/sis/utils/Exceptions/DatabaseInitializationException.java

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package sis.utils.Exceptions;

/**
 *
 * @author lucifer
 */
public class DatabaseInitializationException extends DatabaseException{
    public DatabaseInitializationException(){msg = "Unable to Initialize Database";}
    public DatabaseInitializationException(String msg){this.msg = msg;}
}
```

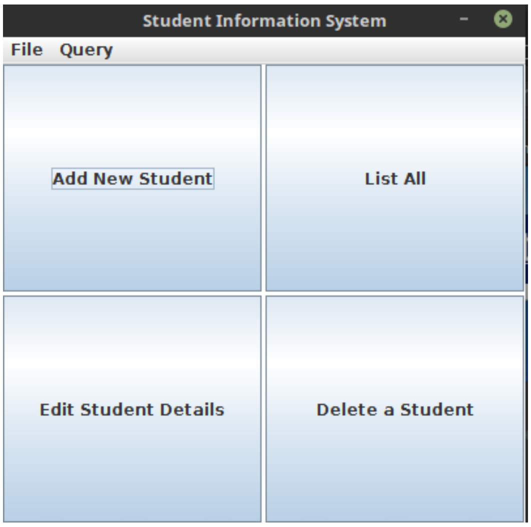
/home/lucifer/NetBeansProjects/sis/src/sis/utils/Exceptions/DatabaseException.java

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package sis.utils.Exceptions;
import java.lang.Exception;

/**
 *
 * @author lucifer
 */
public class DatabaseException extends Exception{
    public String msg;

    DatabaseException(){}
    public DatabaseException(String msg){this.msg = msg;}
}
```

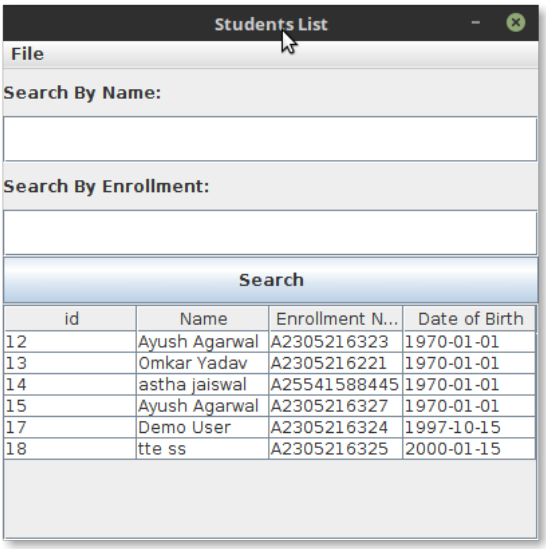
/home/lucifer/NetBeansProjects/sis/src/sis/utils/Exceptions/EnrollmentNotFoundException.java

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package sis.utils.Exceptions;

import java.lang.Exception;

/**
 *
 * @author lucifer
 */
public class EnrollmentNotFoundException extends Exception{

}
```

SCREENSHOTS

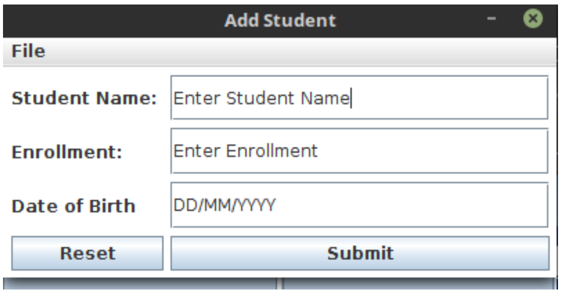


Main App Window



id	Name	Enrollment N...	Date of Birth
12	Ayush Agarwal	A2305216323	1970-01-01
13	Omkar Yadav	A2305216221	1970-01-01
14	astha jaiswal	A25541588445	1970-01-01
15	Ayush Agarwal	A2305216327	1970-01-01
17	Demo User	A2305216324	1997-10-15
18	tte ss	A2305216325	2000-01-15

List All Student View

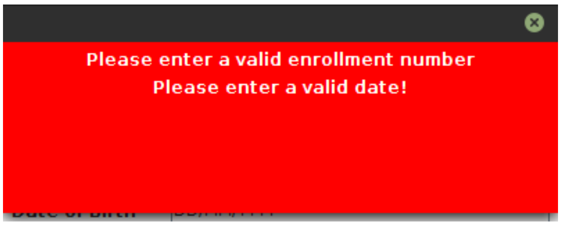


Student Name: Enter Student Name

Enrollment: Enter Enrollment

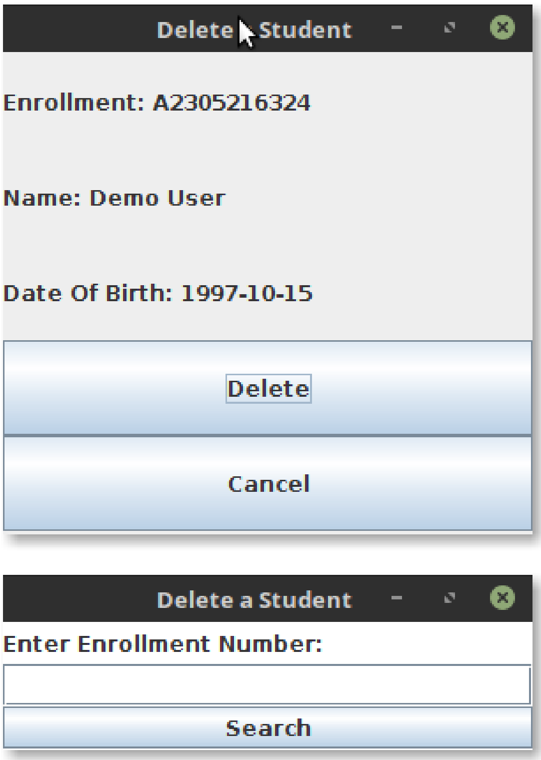
Date of Birth DD/MM/YYYY

Add New Student View



Please enter a valid enrollment number
Please enter a valid date!

Error Box

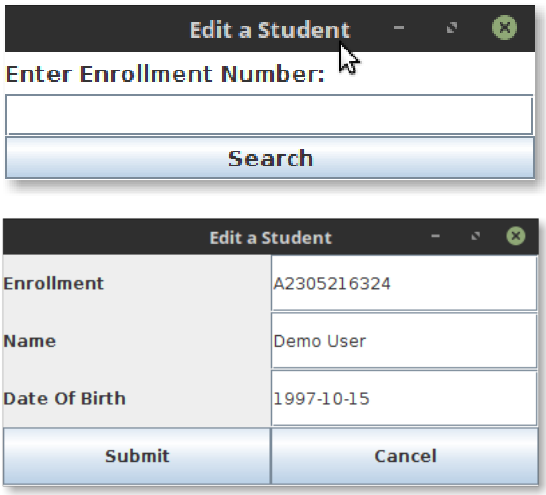


Enrollment: A2305216324

Name: Demo User

Date Of Birth: 1997-10-15

Delete Student View



Enrollment	A2305216324
Name	Demo User
Date Of Birth	1997-10-15

Edit Student Screen
1: Search 2: Edit Details