```
/*
 * 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.
 */

/*  CS 532 - Project 2 (Spring 2018)
*   1. Shriram Suryawanshi
*   2. Vinen Furtado
*/

package minipods;

import java.awt.event.KeyEvent;
import java.sql.*;
import oracle.jdbc.*;
import java.awt.*;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.table.DefaultTableModel;
import oracle.jdbc.pool.OracleDataSource;

/**
 *
 * @author shree
 */
public class Home extends javax.swing.JFrame {

    /**
     * Creates new form Home
     */
    public String user;
    public String pass;
    Connection conn;

    public Home() {
        initComponents();
    }

    public Home(String user1, String pass1) {
        initComponents();

        // @@ - getting username & password from Login class
```

```
    Login loginvar = new Login();
    user = user1;
    pass = pass1;
}

/**
 * This method is called from within the constructor to initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is always
 * regenerated by the Form Editor.
 */
@SuppressWarnings("unchecked")
// <editor-fold defaultstate="collapsed" desc="Generated Code">
private void initComponents() {  // @@ autogenerated GUI code has been removed to reduce code size in the hard copy     }//
</editor-fold>


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

    // @@ - show P_ShowAllStudents panel, hide all others
    P_Default_Students.setVisible(false);
    P_ShowAllStudents.setVisible(true);
    P_AddNewStudent.setVisible(false);
    P_FindStudent.setVisible(false);
    P_DeleteStudent.setVisible(false);

    // @@ - hiding table, in case exceptions
    P_Table_ShowAllStudents.setVisible(false);

    // @@ - fecthing the data using procedure and display on GUI
    try {

        // @@ - connect to DB
        OracleDataSource ds = new oracle.jdbc.pool.OracleDataSource();
        //ds.setURL("jdbc:oracle:thin:@localhost:1521:orcl");
        //Connection conn = ds.getConnection("shree", "shree");
        ds.setURL("jdbc:oracle:thin:@castor.cc.binghamton.edu:1521:ACAD111");
        conn = ds.getConnection(user, pass);

        // @@ - fetching result from DB using procedure
        CallableStatement call = conn.prepareCall("begin SQLPackage.display_students(?); end;");
        call.registerOutParameter(1, OracleTypes.CURSOR);
        call.execute();
```

```
        ResultSet rs = (ResultSet) call.getObject(1);

        // @@ - inserting data into the table on GUI
        DefaultTableModel model = (DefaultTableModel) TBL_ShowAllStudents.getModel();
        model.setRowCount(0);
        TBL_ShowAllStudents.setModel(model);

        int cols = TBL_ShowAllStudents.getColumnCount();

        while (rs.next()) {
            Object[] obj = new Object[cols];
            for (int i = 0; i < cols; i++) {
                obj[i] = rs.getObject(i + 1);
            }
            model.addRow(obj);
        }

        TBL_ShowAllStudents.setModel(model);
        TBL_ShowAllStudents.getTableHeader().setFont(new Font("Monteserrat", Font.BOLD, 14));
        P_Table_ShowAllStudents.setVisible(true);

        L_Message_ShowAllStudent.setText("Displaying all available students - ");
        L_Message_ShowAllStudent.setForeground(Color.BLACK);

        conn.close();

    } catch (SQLException ex) {

        // @@ - show errors on panel -
        L_Message_ShowAllStudent.setText("SQL Exception : " + ex);
        L_Message_ShowAllStudent.setForeground(Color.RED);

        // @@ - handling too many session error by killing conn
        try {
            conn.close();
        } catch (SQLException ex1) {
            Logger.getLogger(Login.class.getName()).log(Level.FINE, null, ex1);
        }

    } catch (Exception e) {

        // @@ - show errors on panel -
        L_Message_ShowAllStudent.setText("Exception : " + e);
```

```
        L_Message_ShowAllStudent.setForeground(Color.RED);

        // @@ - handling too many session error by killing conn
        try {
            conn.close();
        } catch (SQLException ex1) {
            Logger.getLogger(Login.class.getName()).log(Level.FINE, null, ex1);
        }
    }



}


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

    // @@ - show P_AddNewStudents panel, hide all others
    P_Default_Students.setVisible(false);
    P_ShowAllStudents.setVisible(false);
    P_AddNewStudent.setVisible(true);
    P_FindStudent.setVisible(false);
    P_DeleteStudent.setVisible(false);

    // @@ - resetting the panel (textbox values, selections etc.)
    L_Error_AddNewStudent.setVisible(false);
    L_Message_AddNewStudent.setVisible(false);
    T_FirstName.setText("Enter firstname here");
    T_LastName.setText("Enter lastname here");
    T_GPA.setText("Enter gpa here");
    T_Email.setText("Enter email id here");
    RBG_Students.clearSelection();

}


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

    // @@ - Performing validation on the details provided
    if (T_FirstName.getText().equals("") || T_FirstName.getText().equals("Enter firstname here")) {
        L_Error_AddNewStudent.setText("Please provide valid firstname!");
        L_Error_AddNewStudent.setVisible(true);

    } else if (T_LastName.getText().equals("") || T_LastName.getText().equals("Enter lastname here")) {
```

```
    L_Error_AddNewStudent.setText("Please provide valid lastname!");
    L_Error_AddNewStudent.setVisible(true);

} else if ((!(T_GPA.getText().equals("") || T_GPA.getText().equals("Enter gpa here"))) && (Float.parseFloat(T_GPA.getText()) < 0 ||
Float.parseFloat(T_GPA.getText()) > 4.0)) {
    // @@ - allowing null values for gpa
    L_Error_AddNewStudent.setText("Please provide valid gpa!");
    L_Error_AddNewStudent.setVisible(true);

} else if (T_Email.getText().equals("") || T_Email.getText().equals("Enter email id here") || (!T_Email.getText().contains("@"))) {
    L_Error_AddNewStudent.setText("Please provide valid email id!");
    L_Error_AddNewStudent.setVisible(true);

} else if (RB_Freshman.isSelected() == false && RB_Sophomore.isSelected() == false && RB_Junior.isSelected() == false &&
RB_Senior.isSelected() == false && RB_Graduate.isSelected() == false) {
    L_Error_AddNewStudent.setText("Please select valid status!");
    L_Error_AddNewStudent.setVisible(true);

} else {

    L_Error_AddNewStudent.setVisible(false);

    try {

        int SID = 0;
        String sid = "B";

        // @@ - connect to DB
        OracleDataSource ds = new oracle.jdbc.pool.OracleDataSource();
        //ds.setURL("jdbc:oracle:thin:@localhost:1521:orcl");
        //Connection conn = ds.getConnection("shree", "shree");
        ds.setURL("jdbc:oracle:thin:@castor.cc.binghamton.edu:1521:ACAD111");
        conn = ds.getConnection(user, pass);

        Statement stmt = conn.createStatement();

        ResultSet rs;
        rs = stmt.executeQuery("SELECT MAX(sid) FROM students");

        while (rs.next()) {
            SID = Integer.parseInt(rs.getString(1).substring(1));
        }
```

```
// @@ - sid reached to 999, then do nothing, else create new sid
if (SID >= 999) {
   L_Error_AddNewStudent.setText("Maximum SID (999) reached!");
   L_Error_AddNewStudent.setVisible(true);
} else {

   SID++;

   if (SID < 10) {
      sid = sid + "00" + SID;
   } else if (SID < 99) {
      sid = sid + "0" + SID;
   } else if (SID < 999) {
      sid = sid + SID;
   }

   // @@ - getting the student's details from the GUI
   String firstname = T_FirstName.getText();
   String lastname = T_LastName.getText();
   String email = T_Email.getText();
   String status = "";

   if (RB_Freshman.isSelected()) {
      status = "freshman";
   } else if (RB_Sophomore.isSelected()) {
      status = "sophomore";
   } else if (RB_Junior.isSelected()) {
      status = "junior";
   } else if (RB_Senior.isSelected()) {
      status = "senior";
   } else if (RB_Graduate.isSelected()) {
      status = "graduate";
   }

   // @@ - calling procedure to add student
   CallableStatement call = conn.prepareCall("begin SQLPackage.add_student(?, ?, ?, ?, ?, ?); end;");
   call.setString(1, sid);
   call.setString(2, firstname);
   call.setString(3, lastname);
   call.setString(4, status);
   call.setString(6, email);

   // @@ - handling the null gpa condition
```

```
        if (T_GPA.getText().equals("") || T_GPA.getText().equals("Enter gpa here")) {
            call.setString(5, "");
        } else {
            Float gpa = Float.parseFloat(T_GPA.getText());
            call.setFloat(5, gpa);
        }

        int temp = call.executeUpdate();

        if (temp == 1) {
            L_Message_AddNewStudent.setText("Congratulations!! Student added. Your B-Number (sid) is - " + sid);
            L_Message_AddNewStudent.setForeground(Color.BLACK);
            L_Message_AddNewStudent.setVisible(true);

            T_FirstName.setText("Enter firstname here");
            T_LastName.setText("Enter lastname here");
            T_GPA.setText("Enter gpa here");
            T_Email.setText("Enter email id here");
            RBG_Students.clearSelection();
        }
    }

    conn.close();

} catch (SQLException ex) {

    // @@ - if same email found, show error on GUI instead of long oracle excpetion
    if (ex.toString().contains("java.sql.SQLIntegrityConstraintViolationException: ORA-00001: unique constrain")) {
        L_Error_AddNewStudent.setText("SQL Exception : Email id should be unique!");
        L_Error_AddNewStudent.setVisible(true);
    } else {

        // @@ - show other exceptions on the GUI
        L_Message_AddNewStudent.setText("SQL Exception : " + ex);
        L_Message_AddNewStudent.setForeground(Color.RED);
        L_Message_AddNewStudent.setVisible(true);
    }

    // @@ - handling too many session error by killing conn
    try {
        conn.close();
    } catch (SQLException ex1) {
        Logger.getLogger(Login.class.getName()).log(Level.FINE, null, ex1);
```

```
            }

        } catch (HeadlessException | NumberFormatException e) {

            // @@ - show other exceptions on the GUI
            L_Message_AddNewStudent.setText("Exception : " + e);
            L_Message_AddNewStudent.setForeground(Color.RED);
            L_Message_AddNewStudent.setVisible(true);

            // @@ - handling too many session error by killing conn
            try {
                conn.close();
            } catch (SQLException ex1) {
                Logger.getLogger(Login.class.getName()).log(Level.FINE, null, ex1);
            }
        }
    }
}


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

    // @@ - clearing all the selected/entered values, errors on Cancel
    L_Error_AddNewStudent.setVisible(false);
    L_Message_AddNewStudent.setVisible(false);
    T_FirstName.setText("Enter firstname here");
    T_LastName.setText("Enter lastname here");
    T_GPA.setText("Enter gpa here");
    T_Email.setText("Enter email id here");
    RBG_Students.clearSelection();
}


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

    // @@ - show P_FindStudent panel, hide all others
    P_Default_Students.setVisible(false);
    P_ShowAllStudents.setVisible(false);
    P_AddNewStudent.setVisible(false);
    P_FindStudent.setVisible(true);
    P_DeleteStudent.setVisible(false);

    // @@ -  hiding the internal panels, labels from find student panel
```

```
    P_StudentDetails_FindStudent.setVisible(false);
    P_ClassDetails_FindStudents.setVisible(false);
    T_sid_FindStudent.setText("Enter (B-Number) sid here");
    L_Error_FindStudent.setVisible(false);
}


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

    // @@ - show P_DeleteStudent panel, hide all others
    P_Default_Students.setVisible(false);
    P_ShowAllStudents.setVisible(false);
    P_AddNewStudent.setVisible(false);
    P_FindStudent.setVisible(false);
    P_DeleteStudent.setVisible(true);

    // @@ -  hiding labels
    T_sid_DeleteStudent.setText("Enter (B-Number) sid here");
    L_Error_DeleteStudent.setVisible(false);
    L_Message_DeleteStudent.setVisible(false);
}


private void T_FirstNameFocusGained(java.awt.event.FocusEvent evt) {

    // @@ - clearing the default text on focus
    if (T_FirstName.getText().equals("Enter firstname here")) {
        T_FirstName.setText("");

    }
    L_Error_AddNewStudent.setVisible(false);
    L_Message_AddNewStudent.setVisible(false);
}


private void T_FirstNameFocusLost(java.awt.event.FocusEvent evt) {

    // @@ - resetting the defualt text if the firstname is not entered
    if (T_FirstName.getText().equals("")) {
        T_FirstName.setText("Enter firstname here");
    }
}
```

```
private void T_LastNameFocusGained(java.awt.event.FocusEvent evt) {

    // @@ - clearing the default text on focus
    if (T_LastName.getText().equals("Enter lastname here")) {
        T_LastName.setText("");

    }
    L_Error_AddNewStudent.setVisible(false);
    L_Message_AddNewStudent.setVisible(false);
}


private void T_LastNameFocusLost(java.awt.event.FocusEvent evt) {

    // @@ - resetting the defualt text if the lastname is not entered
    if (T_LastName.getText().equals("")) {
        T_LastName.setText("Enter lastname here");
    }
}


private void T_GPAFocusGained(java.awt.event.FocusEvent evt) {

    // @@ - clearing the default text on focus
    if (T_GPA.getText().equals("Enter gpa here")) {
        T_GPA.setText("");

    }
    L_Error_AddNewStudent.setVisible(false);
    L_Message_AddNewStudent.setVisible(false);
}


private void T_GPAFocusLost(java.awt.event.FocusEvent evt) {

    // @@ - resetting the defualt text if the gpa is not entered
    if (T_GPA.getText().equals("")) {
        T_GPA.setText("Enter gpa here");
    }
}
```

```
private void T_EmailFocusGained(java.awt.event.FocusEvent evt) {

    // @@ - clearing the default text on focus
    if (T_Email.getText().equals("Enter email id here")) {
        T_Email.setText("");
    }
    L_Error_AddNewStudent.setVisible(false);
    L_Message_AddNewStudent.setVisible(false);
}



private void T_EmailFocusLost(java.awt.event.FocusEvent evt) {

    // @@ - resetting the defualt text if the email is not entered
    if (T_Email.getText().equals("")) {
        T_Email.setText("Enter email id here");
    }
}



private void T_GPAKeyTyped(java.awt.event.KeyEvent evt) {

    // @@ - allowing only numbers and only one "." in the gpa textbox
    char temp = evt.getKeyChar();

    if (Character.isDigit(temp) || temp == KeyEvent.VK_PERIOD) {

        if (temp == KeyEvent.VK_PERIOD && T_GPA.getText().contains(".")) {
            evt.consume();
        }

    } else {
        evt.consume();
    }
}



private void T_sid_FindStudentFocusGained(java.awt.event.FocusEvent evt) {

    // @@ - clearing the default text on focus
    if (T_sid_FindStudent.getText().equals("Enter (B-Number) sid here")) {
        T_sid_FindStudent.setText("");
    }
```

```
    L_Error_FindStudent.setVisible(false);
}



private void T_sid_FindStudentFocusLost(java.awt.event.FocusEvent evt) {

    // @@ - no value provided, resetting the text
    if (T_sid_FindStudent.getText().equals("")) {
        T_sid_FindStudent.setText("Enter (B-Number) sid here");
    }
}



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

    // @@ -  hiding the internal panels, labels from find student panel
    P_StudentDetails_FindStudent.setVisible(false);
    P_ClassDetails_FindStudents.setVisible(false);
    T_sid_FindStudent.setText("Enter (B-Number) sid here");
    L_Error_FindStudent.setVisible(false);
}



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

    P_StudentDetails_FindStudent.setVisible(false);
    P_ClassDetails_FindStudents.setVisible(false);

    // @@ - perform sid validations first, if passes then fetch student details using procedure along with his classes, show errors
    if ((!T_sid_FindStudent.getText().matches("B[0-9]+")) || (T_sid_FindStudent.getText().length() < 4) || (T_sid_FindStudent.getText().equals("Enter
    (B-Number) sid here"))) {
        L_Error_FindStudent.setVisible(true);
        P_StudentDetails_FindStudent.setVisible(false);
        P_ClassDetails_FindStudents.setVisible(false);
    } else {

        L_Error_FindStudent.setVisible(false);

        // @@ - validation pass, fetech the data and show on GUI
        try {

            String sid = T_sid_FindStudent.getText();
```

```
// @@ - connect to DB
OracleDataSource ds = new oracle.jdbc.pool.OracleDataSource();
//ds.setURL("jdbc:oracle:thin:@localhost:1521:orcl");
//Connection conn = ds.getConnection("shree", "shree");
ds.setURL("jdbc:oracle:thin:@castor.cc.binghamton.edu:1521:ACAD111");
conn = ds.getConnection(user, pass);

// @@ - calling procedure to  student
CallableStatement call = conn.prepareCall("begin SQLPackage.find_student(?, ?); end;");
call.registerOutParameter(1, OracleTypes.CURSOR);
call.setString(2, sid);
call.execute();

ResultSet rs = (ResultSet) call.getObject(1);

// @@ - student is not enrolled in any classes, if column count in rs is only 3
if (rs.getMetaData().getColumnCount() == 3) {

  // @@ - showing student details
  P_StudentDetails_FindStudent.setVisible(true);
  L_Message_Student_FindStudent.setText("Student Details - ");
  L_Message_Student_FindStudent.setForeground(Color.BLACK);
  L_SID_FindStudent.setVisible(true);
  L_Lastname_FindStudent.setVisible(true);
  L_Status_FindStudent.setVisible(true);

  while (rs.next()) {
    L_SID_Value_FindStudent.setText(rs.getString(1));
    L_Lastname_Value_FindStudent.setText(rs.getString(2));
    L_Status_Value_FindStudent.setText(rs.getString(3));
  }

  P_ClassDetails_FindStudents.setVisible(true);
  L_Message_Classes_FindStudent.setText("The student has not taken any course.");
  L_Message_Classes_FindStudent.setForeground(Color.RED);
  TBL_Classes_FindStudent.setVisible(false);

  P_StudentDetails_FindStudent.setVisible(true);
  P_ClassDetails_FindStudents.setVisible(true);

} else {

  // @@ - showing student details
```

```
            P_StudentDetails_FindStudent.setVisible(true);
            L_Message_Student_FindStudent.setText("Student Details - ");
            L_Message_Student_FindStudent.setForeground(Color.BLACK);
            L_SID_FindStudent.setVisible(true);
            L_Lastname_FindStudent.setVisible(true);
            L_Status_FindStudent.setVisible(true);

            // @@ - showing classes details
            P_ClassDetails_FindStudents.setVisible(true);
            L_Message_Classes_FindStudent.setText("Classes Enrolled - ");
            L_Message_Classes_FindStudent.setForeground(Color.BLACK);
            TBL_Classes_FindStudent.setVisible(true);

            // @@ - inserting data into the table on GUI
            DefaultTableModel model = (DefaultTableModel) TBL_Classes_FindStudent.getModel();
            model.setRowCount(0);
            TBL_Classes_FindStudent.setModel(model);

            int cols = TBL_Classes_FindStudent.getColumnCount();

            while (rs.next()) {
               Object[] obj = new Object[cols];

               L_SID_Value_FindStudent.setText(rs.getString(1));
               L_Lastname_Value_FindStudent.setText(rs.getString(2));
               L_Status_Value_FindStudent.setText(rs.getString(3));

               for (int i = 0; i < cols; i++) {
                  obj[i] = rs.getObject(i + 4);
               }
               model.addRow(obj);
            }

            P_StudentDetails_FindStudent.setVisible(true);
            P_ClassDetails_FindStudents.setVisible(true);
            TBL_Classes_FindStudent.setModel(model);
            TBL_Classes_FindStudent.getTableHeader().setFont(new Font("Monteserrat", Font.BOLD, 14));
         }

         conn.close();

      } catch (SQLException ex) {
```

```
            // @@ - show excptions on GUI (avoiding long error text thrown by oracle)
            if (ex.toString().contains("The sid is invalid.")) {
               L_Error_FindStudent.setText("The sid is invalid.");
            } else {
               L_Error_FindStudent.setText(ex.toString());
            }
            L_Error_FindStudent.setVisible(true);

            // @@ - handling too many session error by killing conn
            try {
               conn.close();
            } catch (SQLException ex1) {
               Logger.getLogger(Login.class.getName()).log(Level.FINE, null, ex1);
            }

         } catch (HeadlessException | NumberFormatException e) {

            // @@ - show excptions on GUI
            L_Error_FindStudent.setText(e.toString());
            L_Error_FindStudent.setVisible(true);

            // @@ - handling too many session error by killing conn
            try {
               conn.close();
            } catch (SQLException ex1) {
               Logger.getLogger(Login.class.getName()).log(Level.FINE, null, ex1);
            }
         }

      }
   }


   private void B_ExitStudentActionPerformed(java.awt.event.ActionEvent evt) {
      System.exit(0);
   }


   private void T_sid_DeleteStudentFocusGained(java.awt.event.FocusEvent evt) {

      // @@ - clearing the default text on focus
      if (T_sid_DeleteStudent.getText().equals("Enter (B-Number) sid here")) {
         T_sid_DeleteStudent.setText("");
```

```
    }
    L_Error_DeleteStudent.setVisible(false);
    L_Message_DeleteStudent.setVisible(false);
}



private void T_sid_DeleteStudentFocusLost(java.awt.event.FocusEvent evt) {

    // @@ - no value provided, resetting the text
    if (T_sid_DeleteStudent.getText().equals("")) {
        T_sid_DeleteStudent.setText("Enter (B-Number) sid here");
    }
}



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

    // @@ - perform sid validations first, if passes then delete student
    if ((!T_sid_DeleteStudent.getText().matches("B[0-9]+")) || (T_sid_DeleteStudent.getText().length() < 4) ||
    (T_sid_DeleteStudent.getText().equals("Enter (B-Number) sid here"))) {
        L_Error_DeleteStudent.setVisible(true);
        L_Message_DeleteStudent.setVisible(false);
    } else {

        L_Error_DeleteStudent.setVisible(false);

        try {

            String sid = T_sid_DeleteStudent.getText();

            // @@ - connect to DB
            OracleDataSource ds = new oracle.jdbc.pool.OracleDataSource();
            //ds.setURL("jdbc:oracle:thin:@localhost:1521:orcl");
            //Connection conn = ds.getConnection("shree", "shree");
            ds.setURL("jdbc:oracle:thin:@castor.cc.binghamton.edu:1521:ACAD111");
            conn = ds.getConnection(user, pass);

            // @@ - calling procedure to delete student
            CallableStatement call = conn.prepareCall("begin SQLPackage.delete_student(?); end;");
            call.setString(1, sid);
            int temp = call.executeUpdate();

            if (temp == 1) {
```

```
            L_Message_DeleteStudent.setText("Student having sid - " + sid + " has been deleted successfully!");
            L_Message_DeleteStudent.setVisible(true);
            T_sid_DeleteStudent.setText("Enter (B-Number) sid here");
        }

        conn.close();

    } catch (SQLException ex) {

        // @@ - show excptions on GUI (avoiding long error text thrown by oracle)
        if (ex.toString().contains("The sid is invalid.")) {
            L_Error_DeleteStudent.setText("The sid is invalid.");
        } else {
            L_Error_DeleteStudent.setText(ex.toString());
        }
        L_Error_DeleteStudent.setVisible(true);

        // @@ - handling too many session error by killing conn
        try {
            conn.close();
        } catch (SQLException ex1) {
            Logger.getLogger(Login.class.getName()).log(Level.FINE, null, ex1);
        }

    } catch (HeadlessException | NumberFormatException e) {

        // @@ - show excptions on GUI
        L_Error_DeleteStudent.setText(e.toString());
        L_Error_DeleteStudent.setVisible(true);

        // @@ - handling too many session error by killing conn
        try {
            conn.close();
        } catch (SQLException ex1) {
            Logger.getLogger(Login.class.getName()).log(Level.FINE, null, ex1);
        }
    }
  }
}


private void B_Cancel_DeleteStudentActionPerformed(java.awt.event.ActionEvent evt) {
```

```
    // @@ -  hiding labels
    T_sid_DeleteStudent.setText("Enter (B-Number) sid here");
    L_Error_DeleteStudent.setVisible(false);
}



private void TabsStateChanged(javax.swing.event.ChangeEvent evt) {

    // @@ - tab selection changed, hide panels depending on Tab selected
    if (Tabs.getSelectedIndex() == 0) {
        P_Default_Students.setVisible(true);
        P_ShowAllStudents.setVisible(false);
        P_AddNewStudent.setVisible(false);
        P_FindStudent.setVisible(false);
        P_DeleteStudent.setVisible(false);

    } else if (Tabs.getSelectedIndex() == 1) {
        P_Default_Courses.setVisible(true);
        P_ShowAllCourses.setVisible(false);
        P_ShowAllPrerequisites.setVisible(false);
        P_FindCourse.setVisible(false);

    } else if (Tabs.getSelectedIndex() == 2) {
        P_Default_Classes.setVisible(true);
        P_ShowAllClasses.setVisible(false);
        P_FindClass.setVisible(false);

    } else if (Tabs.getSelectedIndex() == 3) {
        P_Default_Enrollments.setVisible(true);
        P_ShowAllEnrollments.setVisible(false);
        P_EnrollStudent.setVisible(false);
        P_DropEnrollment.setVisible(false);

    } else if (Tabs.getSelectedIndex() == 4) {
        P_Default_Logs.setVisible(true);
        P_ShowAllLogs.setVisible(false);
    }

}



private void B_ShowAllCoursesActionPerformed(java.awt.event.ActionEvent evt) {
```

```
// @@ - show only P_ShowAllCourses panel
P_Default_Courses.setVisible(false);
P_ShowAllCourses.setVisible(true);
P_ShowAllPrerequisites.setVisible(false);
P_FindCourse.setVisible(false);

// @@ - hiding a table
P_Table_ShowAllCourses.setVisible(false);

// @@ - fecthing the data using procedure and display on GUI
try {

   // @@ - connect to DB
   OracleDataSource ds = new oracle.jdbc.pool.OracleDataSource();
   //ds.setURL("jdbc:oracle:thin:@localhost:1521:orcl");
   //Connection conn = ds.getConnection("shree", "shree");
   ds.setURL("jdbc:oracle:thin:@castor.cc.binghamton.edu:1521:ACAD111");
   conn = ds.getConnection(user, pass);

   // @@ - fetching result from DB usnig procedure
   CallableStatement call = conn.prepareCall("begin SQLPackage.display_courses(?); end;");
   call.registerOutParameter(1, OracleTypes.CURSOR);
   call.execute();

   ResultSet rs = (ResultSet) call.getObject(1);

   // @@ - inserting data into the table on GUI
   DefaultTableModel model = (DefaultTableModel) TBL_ShowAllCourses.getModel();
   model.setRowCount(0);
   TBL_ShowAllCourses.setModel(model);

   int cols = TBL_ShowAllCourses.getColumnCount();

   while (rs.next()) {
      Object[] obj = new Object[cols];
      for (int i = 0; i < cols; i++) {
         obj[i] = rs.getObject(i + 1);
      }
      model.addRow(obj);
   }

   TBL_ShowAllCourses.setModel(model);
   TBL_ShowAllCourses.getTableHeader().setFont(new Font("Monteserrat", Font.BOLD, 14));
```

```
            P_Table_ShowAllCourses.setVisible(true);

            L_Message_ShowAllCourses.setText("Displaying all available courses - ");
            L_Message_ShowAllCourses.setForeground(Color.BLACK);

            conn.close();

        } catch (SQLException ex) {

            // @@ - show errors on panel -
            L_Message_ShowAllCourses.setText("SQL Exception : " + ex);
            L_Message_ShowAllCourses.setForeground(Color.RED);

            // @@ - handling too many session error by killing conn
            try {
                conn.close();
            } catch (SQLException ex1) {
                Logger.getLogger(Login.class.getName()).log(Level.FINE, null, ex1);
            }

        } catch (Exception e) {

            // @@ - show errors on panel -
            L_Message_ShowAllCourses.setText("Exception : " + e);
            L_Message_ShowAllCourses.setForeground(Color.RED);

            // @@ - handling too many session error by killing conn
            try {
                conn.close();
            } catch (SQLException ex1) {
                Logger.getLogger(Login.class.getName()).log(Level.FINE, null, ex1);
            }
        }
    }



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

        // @@ - show only prerequisites panel
        P_Default_Courses.setVisible(false);
        P_ShowAllCourses.setVisible(false);
        P_ShowAllPrerequisites.setVisible(true);
        P_FindCourse.setVisible(false);
```

```
// @@ - hiding a table
P_Table_ShowAllPre.setVisible(false);

// @@ - fecthing the data using procedure and display on GUI
try {

   // @@ - connect to DB
   OracleDataSource ds = new oracle.jdbc.pool.OracleDataSource();
   //ds.setURL("jdbc:oracle:thin:@localhost:1521:orcl");
   //Connection conn = ds.getConnection("shree", "shree");
   ds.setURL("jdbc:oracle:thin:@castor.cc.binghamton.edu:1521:ACAD111");
   conn = ds.getConnection(user, pass);

   // @@ - fetching result from DB usnig procedure
   CallableStatement call = conn.prepareCall("begin SQLPackage.display_prerequisites(?); end;");
   call.registerOutParameter(1, OracleTypes.CURSOR);
   call.execute();

   ResultSet rs = (ResultSet) call.getObject(1);

   // @@ - inserting data into the table on GUI
   DefaultTableModel model = (DefaultTableModel) TBL_ShowAllPre.getModel();
   model.setRowCount(0);
   TBL_ShowAllPre.setModel(model);

   int cols = TBL_ShowAllPre.getColumnCount();

   while (rs.next()) {
      Object[] obj = new Object[cols];
      for (int i = 0; i < cols; i++) {
         obj[i] = rs.getObject(i + 1);
      }
      model.addRow(obj);
   }

   TBL_ShowAllPre.setModel(model);
   TBL_ShowAllPre.getTableHeader().setFont(new Font("Monteserrat", Font.BOLD, 14));
   P_Table_ShowAllPre.setVisible(true);

   L_Message_ShowAllPre.setText("Displaying all available prerequisites - ");
   L_Message_ShowAllPre.setForeground(Color.BLACK);
```

```
            conn.close();

        } catch (SQLException ex) {

            // @@ - show errors on panel -
            L_Message_ShowAllPre.setText("SQL Exception : " + ex);
            L_Message_ShowAllPre.setForeground(Color.RED);

            // @@ - handling too many session error by killing conn
            try {
                conn.close();
            } catch (SQLException ex1) {
                Logger.getLogger(Login.class.getName()).log(Level.FINE, null, ex1);
            }

        } catch (Exception e) {

            // @@ - show errors on panel -
            L_Message_ShowAllPre.setText("Exception : " + e);
            L_Message_ShowAllPre.setForeground(Color.RED);

            // @@ - handling too many session error by killing conn
            try {
                conn.close();
            } catch (SQLException ex1) {
                Logger.getLogger(Login.class.getName()).log(Level.FINE, null, ex1);
            }
        }

    }



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

        // @@ - show only find course panel
        P_Default_Courses.setVisible(false);
        P_ShowAllCourses.setVisible(false);
        P_ShowAllPrerequisites.setVisible(false);
        P_FindCourse.setVisible(true);

        // @@ - resetting the find course panel
        T_Dept_FindCourse.setText("Enter department code here");
        T_Course_FindCourse.setText("Enter course no. here");
```

```
    L_Error_FindCourse.setVisible(false);
    P_ShowPre_FindCourse.setVisible(false);
    L_Message_FindCourse.setVisible(false);
}


private void B_ExitCoursesActionPerformed(java.awt.event.ActionEvent evt) {
    System.exit(0);
}


private void T_Dept_FindCourseFocusGained(java.awt.event.FocusEvent evt) {

    // @@ - clearing the default text on focus
    if (T_Dept_FindCourse.getText().equals("Enter department code here")) {
        T_Dept_FindCourse.setText("");
    }
    L_Error_FindCourse.setVisible(false);
}


private void T_Dept_FindCourseFocusLost(java.awt.event.FocusEvent evt) {

    // @@ - no value provided, resetting the text
    if (T_Dept_FindCourse.getText().equals("")) {
        T_Dept_FindCourse.setText("Enter department code here");
    }
}


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

    // @@ - performing validation on details entered, if pass, fetch details, ele report errors
    if (T_Dept_FindCourse.getText().equals("") || T_Dept_FindCourse.getText().equals("Enter department code here") ||
    T_Dept_FindCourse.getText().length() > 4) {
        L_Error_FindCourse.setText("Please provide valid department code!");
        L_Error_FindCourse.setVisible(true);
        P_ShowPre_FindCourse.setVisible(false);

    } else if (T_Course_FindCourse.getText().equals("") || T_Course_FindCourse.getText().equals("Enter course no. here") ||
    (!T_Course_FindCourse.getText().matches("[0-9][0-9][0-9]"))) {
        L_Error_FindCourse.setText("Please provide valid numeric 3 digit course number!");
        L_Error_FindCourse.setVisible(true);
```

```
        P_ShowPre_FindCourse.setVisible(false);
    } else {

        L_Error_FindCourse.setVisible(false);

        // @@ - fecthing the data using procedure and display on GUI
        try {

            String dept = T_Dept_FindCourse.getText().toUpperCase();
            Integer course = Integer.parseInt(T_Course_FindCourse.getText());

            // @@ - connect to DB
            OracleDataSource ds = new oracle.jdbc.pool.OracleDataSource();
            //ds.setURL("jdbc:oracle:thin:@localhost:1521:orcl");
            //Connection conn = ds.getConnection("shree", "shree");
            ds.setURL("jdbc:oracle:thin:@castor.cc.binghamton.edu:1521:ACAD111");
            conn = ds.getConnection(user, pass);

            // @@ - fetching result from DB usnig procedure
            CallableStatement call = conn.prepareCall("begin SQLPackage.find_course(?, ?, ?); end;");
            call.registerOutParameter(1, OracleTypes.CURSOR);
            call.setString(2, dept);
            call.setInt(3, course);
            call.execute();

            ResultSet rs = (ResultSet) call.getObject(1);

            // @@ - inserting data into the table on GUI
            DefaultTableModel model = (DefaultTableModel) TBL_ShowPre_FindCourse.getModel();
            model.setRowCount(0);
            TBL_ShowPre_FindCourse.setModel(model);

            int cols = TBL_ShowPre_FindCourse.getColumnCount();

            int rows = 0;

            while (rs.next()) {
                Object[] obj = new Object[cols];
                for (int i = 0; i < cols; i++) {
                    obj[i] = rs.getObject(i + 1);
                }
                model.addRow(obj);
                rows++;
```

```
        }

        if (rows != 0) {
            TBL_ShowPre_FindCourse.setModel(model);
            TBL_ShowPre_FindCourse.getTableHeader().setFont(new Font("Monteserrat", Font.BOLD, 14));
            P_ShowPre_FindCourse.setVisible(true);

            L_Message_FindCourse.setText("Prerequisites courses - ");
            L_Message_FindCourse.setForeground(Color.BLACK);
            L_Message_FindCourse.setVisible(true);
        } else {
            L_Message_FindCourse.setText("There is no prerequisites courses for this course!");
            L_Message_FindCourse.setForeground(Color.BLACK);
            L_Message_FindCourse.setVisible(true);
            P_ShowPre_FindCourse.setVisible(false);
        }

        conn.close();

    } catch (SQLException ex) {

        // @@ - show errors on panel
        if (ex.toString().contains("Course not found!")) {
            L_Message_FindCourse.setText("Course not found! Please make sure provided department code and course no. combination is correct!");
        } else {
            L_Message_FindCourse.setText("SQL Exception : " + ex);
        }
        L_Message_FindCourse.setForeground(Color.RED);
        L_Message_FindCourse.setVisible(true);
        P_ShowPre_FindCourse.setVisible(false);

        // @@ - handling too many session error by killing conn
        try {
            conn.close();
        } catch (SQLException ex1) {
            Logger.getLogger(Login.class.getName()).log(Level.FINE, null, ex1);
        }

    } catch (Exception e) {

        // @@ - show errors on panel
        L_Message_FindCourse.setText("Exception : " + e);
        L_Message_FindCourse.setForeground(Color.RED);
```

```
            L_Message_FindCourse.setVisible(true);
            P_ShowPre_FindCourse.setVisible(false);

            // @@ - handling too many session error by killing conn
            try {
               conn.close();
            } catch (SQLException ex1) {
               Logger.getLogger(Login.class.getName()).log(Level.FINE, null, ex1);
            }
         }
      }



}


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

      // @@ - resetting the find course panel
      T_Dept_FindCourse.setText("Enter department code here");
      T_Course_FindCourse.setText("Enter course no. here");
      L_Error_FindCourse.setVisible(false);
      P_ShowPre_FindCourse.setVisible(false);
      L_Message_FindCourse.setVisible(false);
   }


   private void T_Course_FindCourseFocusGained(java.awt.event.FocusEvent evt) {

      // @@ - clearing the default text on focus
      if (T_Course_FindCourse.getText().equals("Enter course no. here")) {
         T_Course_FindCourse.setText("");
      }
      L_Error_FindCourse.setVisible(false);
   }


   private void T_Course_FindCourseFocusLost(java.awt.event.FocusEvent evt) {

      // @@ - no value provided, resetting the text
      if (T_Course_FindCourse.getText().equals("")) {
         T_Course_FindCourse.setText("Enter course no. here");
      }
```

```
}


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

    // @@ - show only show all classes panel
    P_Default_Classes.setVisible(false);
    P_ShowAllClasses.setVisible(true);
    P_FindClass.setVisible(false);

    // @@ - hiding a table
    P_Table_ShowAllClasses.setVisible(false);

    // @@ - fecthing the data using procedure and display on GUI
    try {

        // @@ - connect to DB
        OracleDataSource ds = new oracle.jdbc.pool.OracleDataSource();
        //ds.setURL("jdbc:oracle:thin:@localhost:1521:orcl");
        //Connection conn = ds.getConnection("shree", "shree");
        ds.setURL("jdbc:oracle:thin:@castor.cc.binghamton.edu:1521:ACAD111");
        conn = ds.getConnection(user, pass);

        // @@ - fetching result from DB usnig procedure
        CallableStatement call = conn.prepareCall("begin SQLPackage.display_classes(?); end;");
        call.registerOutParameter(1, OracleTypes.CURSOR);
        call.execute();

        ResultSet rs = (ResultSet) call.getObject(1);

        // @@ - inserting data into the table on GUI
        DefaultTableModel model = (DefaultTableModel) TBL_ShowAllClasses.getModel();
        model.setRowCount(0);
        TBL_ShowAllClasses.setModel(model);

        int cols = TBL_ShowAllClasses.getColumnCount();

        while (rs.next()) {
            Object[] obj = new Object[cols];
            for (int i = 0; i < cols; i++) {
                obj[i] = rs.getObject(i + 1);
            }
            model.addRow(obj);
```

```
        }

        TBL_ShowAllClasses.setModel(model);
        TBL_ShowAllClasses.getTableHeader().setFont(new Font("Monteserrat", Font.BOLD, 14));
        P_Table_ShowAllClasses.setVisible(true);

        L_Message_ShowAllClasses.setText("Displaying all available classes - ");
        L_Message_ShowAllClasses.setForeground(Color.BLACK);

        conn.close();

    } catch (SQLException ex) {

        // @@ - show errors on panel -
        L_Message_ShowAllClasses.setText("SQL Exception : " + ex);
        L_Message_ShowAllClasses.setForeground(Color.RED);

        // @@ - handling too many session error by killing conn
        try {
            conn.close();
        } catch (SQLException ex1) {
            Logger.getLogger(Login.class.getName()).log(Level.FINE, null, ex1);
        }

    } catch (Exception e) {

        // @@ - show errors on panel -
        L_Message_ShowAllClasses.setText("Exception : " + e);
        L_Message_ShowAllClasses.setForeground(Color.RED);

        // @@ - handling too many session error by killing conn
        try {
            conn.close();
        } catch (SQLException ex1) {
            Logger.getLogger(Login.class.getName()).log(Level.FINE, null, ex1);
        }
    }
}


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

    // @@ - show only show all classes panel
```

```
    P_Default_Classes.setVisible(false);
    P_ShowAllClasses.setVisible(false);
    P_FindClass.setVisible(true);

    // @@ - resetting the panel
    L_Error_FindClass.setVisible(false);
    P_ClassDetails_FindClass.setVisible(false);
    P_StudentDetails_FindClass.setVisible(false);
    T_classid_FindClass.setText("Enter classid here");
}


private void B_ExitClassesActionPerformed(java.awt.event.ActionEvent evt) {
    System.exit(0);
}


private void T_classid_FindClassFocusGained(java.awt.event.FocusEvent evt) {

    // @@ - clearing the default text on focus
    if (T_classid_FindClass.getText().equals("Enter classid here")) {
        T_classid_FindClass.setText("");
    }
    L_Error_FindClass.setVisible(false);
}


private void T_classid_FindClassFocusLost(java.awt.event.FocusEvent evt) {

    // @@ - no value provided, resetting the text
    if (T_classid_FindClass.getText().equals("")) {
        T_classid_FindClass.setText("Enter classid here");
    }
}


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

    // @@ - hiding the panels
    P_ClassDetails_FindClass.setVisible(false);
    P_StudentDetails_FindClass.setVisible(false);
```

```
// @@ - perform classid validations first, if passes then fetch class details using procedure along with enrolled students, show errors
if ((!T_classid_FindClass.getText().matches("c[0-9]+")) || (T_classid_FindClass.getText().equals("Enter classid here"))) {
   L_Error_FindClass.setVisible(true);
   P_ClassDetails_FindClass.setVisible(false);
   P_StudentDetails_FindClass.setVisible(false);
} else {

   L_Error_FindClass.setVisible(false);

   // @@ - validation pass, fetech the data and show on GUI
   try {

      String cid = T_classid_FindClass.getText();

      // @@ - connect to DB
      OracleDataSource ds = new oracle.jdbc.pool.OracleDataSource();
      //ds.setURL("jdbc:oracle:thin:@localhost:1521:orcl");
      //Connection conn = ds.getConnection("shree", "shree");
      ds.setURL("jdbc:oracle:thin:@castor.cc.binghamton.edu:1521:ACAD111");
      conn = ds.getConnection(user, pass);

      // @@ - calling procedure to  student
      CallableStatement call = conn.prepareCall("begin SQLPackage.find_class(?, ?); end;");
      call.registerOutParameter(1, OracleTypes.CURSOR);
      call.setString(2, cid);
      call.execute();

      ResultSet rs = (ResultSet) call.getObject(1);

      // @@ - student is not enrolled in any classes, if column count in rs is only 3
      if (rs.getMetaData().getColumnCount() == 4) {

         // @@ - showing student details
         P_ClassDetails_FindClass.setVisible(true);
         L_Message_Class_FindClass.setText("Class Details - ");
         L_Message_Class_FindClass.setForeground(Color.BLACK);
         L_Classid_FindClass.setVisible(true);
         L_Title_FindClass.setVisible(true);
         L_Semester_FindClass.setVisible(true);
         L_Year_FindClass.setVisible(true);

         while (rs.next()) {
            L_Classid_Value_FindClass.setText(rs.getString(1));
```

```
            L_Title_Value_FindClass.setText(rs.getString(2));
            L_Semester_Value_FindClass.setText(rs.getString(3));
            L_Year_Value_FindClass.setText(rs.getString(4));
        }

        P_StudentDetails_FindClass.setVisible(true);
        L_Message_Students_FindClass.setText("No student is enrolled in the class.");
        L_Message_Students_FindClass.setForeground(Color.RED);
        TBL_Students_FindClass.setVisible(false);

    } else {

        // @@ - showing student details
        P_ClassDetails_FindClass.setVisible(true);
        L_Message_Class_FindClass.setText("Class Details - ");
        L_Message_Class_FindClass.setForeground(Color.BLACK);
        L_Classid_FindClass.setVisible(true);
        L_Title_FindClass.setVisible(true);
        L_Semester_FindClass.setVisible(true);
        L_Year_FindClass.setVisible(true);

        // @@ - showing classes details
        P_StudentDetails_FindClass.setVisible(true);
        L_Message_Students_FindClass.setText("Enrolled Students - ");
        L_Message_Students_FindClass.setForeground(Color.BLACK);
        TBL_Students_FindClass.setVisible(true);

        // @@ - inserting data into the table on GUI
        DefaultTableModel model = (DefaultTableModel) TBL_Students_FindClass.getModel();
        model.setRowCount(0);
        TBL_Students_FindClass.setModel(model);

        int cols = TBL_Students_FindClass.getColumnCount();

        while (rs.next()) {
            Object[] obj = new Object[cols];

            L_Classid_Value_FindClass.setText(rs.getString(1));
            L_Title_Value_FindClass.setText(rs.getString(2));
            L_Semester_Value_FindClass.setText(rs.getString(3));
            L_Year_Value_FindClass.setText(rs.getString(4));

            for (int i = 0; i < cols; i++) {
```

```
                obj[i] = rs.getObject(i + 5);
            }
            model.addRow(obj);
        }

        TBL_Students_FindClass.setModel(model);
        TBL_Students_FindClass.getTableHeader().setFont(new Font("Monteserrat", Font.BOLD, 14));
    }

    conn.close();

} catch (SQLException ex) {

    // @@ - show excptions on GUI (avoiding long error text thrown by oracle)
    if (ex.toString().contains("The cid is invalid.")) {
        L_Error_FindClass.setText("The cid is invalid.");
    } else {
        L_Error_FindClass.setText(ex.toString());
    }
    L_Error_FindClass.setVisible(true);

    // @@ - handling too many session error by killing conn
    try {
        conn.close();
    } catch (SQLException ex1) {
        Logger.getLogger(Login.class.getName()).log(Level.FINE, null, ex1);
    }

} catch (HeadlessException | NumberFormatException e) {

    // @@ - show excptions on GUI
    L_Error_FindClass.setText(e.toString());
    L_Error_FindClass.setVisible(true);

    // @@ - handling too many session error by killing conn
    try {
        conn.close();
    } catch (SQLException ex1) {
        Logger.getLogger(Login.class.getName()).log(Level.FINE, null, ex1);
    }
  }
 }
}
```

```
private void B_Cancel_FindClassActionPerformed(java.awt.event.ActionEvent evt) {

    // @@ - resetting the panel
    L_Error_FindClass.setVisible(false);
    P_ClassDetails_FindClass.setVisible(false);
    P_StudentDetails_FindClass.setVisible(false);
    T_classid_FindClass.setText("Enter classid here");
}


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

    // @@ - show only show all enrollment panel
    P_Default_Enrollments.setVisible(false);
    P_ShowAllEnrollments.setVisible(true);
    P_EnrollStudent.setVisible(false);
    P_DropEnrollment.setVisible(false);

    // @@ - hiding a table
    P_Table_ShowAllEnrollments.setVisible(false);

    // @@ - fecthing the data using procedure and display on GUI
    try {

        // @@ - connect to DB
        OracleDataSource ds = new oracle.jdbc.pool.OracleDataSource();
        //ds.setURL("jdbc:oracle:thin:@localhost:1521:orcl");
        //Connection conn = ds.getConnection("shree", "shree");
        ds.setURL("jdbc:oracle:thin:@castor.cc.binghamton.edu:1521:ACAD111");
        conn = ds.getConnection(user, pass);

        // @@ - fetching result from DB usnig procedure
        CallableStatement call = conn.prepareCall("begin SQLPackage.display_enrollments(?); end;");
        call.registerOutParameter(1, OracleTypes.CURSOR);
        call.execute();

        ResultSet rs = (ResultSet) call.getObject(1);

        // @@ - inserting data into the table on GUI
        DefaultTableModel model = (DefaultTableModel) TBL_ShowAllEnrollments.getModel();
        model.setRowCount(0);
```

```
        TBL_ShowAllEnrollments.setModel(model);

        int cols = TBL_ShowAllEnrollments.getColumnCount();

        while (rs.next()) {
            Object[] obj = new Object[cols];
            for (int i = 0; i < cols; i++) {
                obj[i] = rs.getObject(i + 1);
            }
            model.addRow(obj);
        }

        TBL_ShowAllEnrollments.setModel(model);
        TBL_ShowAllEnrollments.getTableHeader().setFont(new Font("Monteserrat", Font.BOLD, 14));
        P_Table_ShowAllEnrollments.setVisible(true);

        L_Message_ShowAllEnrollments.setText("Displaying all available enrollments - ");
        L_Message_ShowAllEnrollments.setForeground(Color.BLACK);

        conn.close();

    } catch (SQLException ex) {

        // @@ - show errors on panel -
        L_Message_ShowAllEnrollments.setText("SQL Exception : " + ex);
        L_Message_ShowAllEnrollments.setForeground(Color.RED);

        // @@ - handling too many session error by killing conn
        try {
            conn.close();
        } catch (SQLException ex1) {
            Logger.getLogger(Login.class.getName()).log(Level.FINE, null, ex1);
        }

    } catch (Exception e) {

        // @@ - show errors on panel -
        L_Message_ShowAllEnrollments.setText("Exception : " + e);
        L_Message_ShowAllEnrollments.setForeground(Color.RED);

        // @@ - handling too many session error by killing conn
        try {
            conn.close();
```

```
        } catch (SQLException ex1) {
            Logger.getLogger(Login.class.getName()).log(Level.FINE, null, ex1);
        }
    }

}


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

    // @@ - show only enrollment panel
    P_Default_Enrollments.setVisible(false);
    P_ShowAllEnrollments.setVisible(false);
    P_EnrollStudent.setVisible(true);
    //P_DropEnrollment.setVisible(false);

    // @@ - resetting the panel
    T_sid_EnrollStudent.setText("Enter sid here");
    T_cid_EnrollStudent.setText("Enter classid here");
    L_Error_EnrollStudent.setVisible(false);
    L_Message_EnrollStudent.setVisible(false);
}


private void B_ExitEnrollmentsActionPerformed(java.awt.event.ActionEvent evt) {
    System.exit(0);
}


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

    // @@ - show only enrollment panel
    P_Default_Enrollments.setVisible(false);
    P_ShowAllEnrollments.setVisible(false);
    P_EnrollStudent.setVisible(false);
    P_DropEnrollment.setVisible(true);

    // @@ - resetting the panel
    T_sid_DropEnrollment.setText("Enter sid here");
    T_cid_DropEnrollment.setText("Enter classid here");
    L_Error_DropEnrollment.setVisible(false);
    L_Message_DropEnrollment.setVisible(false);
}
```

```
private void T_sid_EnrollStudentFocusGained(java.awt.event.FocusEvent evt) {

    // @@ - clearing the default text on focus
    if (T_sid_EnrollStudent.getText().equals("Enter sid here")) {
        T_sid_EnrollStudent.setText("");
    }
    L_Error_EnrollStudent.setVisible(false);
}


private void T_sid_EnrollStudentFocusLost(java.awt.event.FocusEvent evt) {

    // @@ - no value provided, resetting the text
    if (T_sid_EnrollStudent.getText().equals("")) {
        T_sid_EnrollStudent.setText("Enter sid here");
    }
}


private void T_cid_EnrollStudentFocusGained(java.awt.event.FocusEvent evt) {

    // @@ - clearing the default text on focus
    if (T_cid_EnrollStudent.getText().equals("Enter classid here")) {
        T_cid_EnrollStudent.setText("");
    }
    L_Error_EnrollStudent.setVisible(false);
}


private void T_cid_EnrollStudentFocusLost(java.awt.event.FocusEvent evt) {

    // @@ - no value provided, resetting the text
    if (T_cid_EnrollStudent.getText().equals("")) {
        T_cid_EnrollStudent.setText("Enter classid here");
    }
}


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

    // @@ - performing validation on details entered, if pass, fetch details, else report errors
```

```
if ((!T_sid_EnrollStudent.getText().matches("B[0-9]+")) || (T_sid_EnrollStudent.getText().length() < 4) ||
(T_sid_EnrollStudent.getText().equals("Enter (B-Number) sid here"))) {
    L_Error_EnrollStudent.setText("Please provide valid sid (sid should be of length 4, and it starts with B)!");
    L_Error_EnrollStudent.setVisible(true);
    L_Message_EnrollStudent.setVisible(false);

} else if ((!T_cid_EnrollStudent.getText().matches("c[0-9]+")) || (T_cid_EnrollStudent.getText().equals("Enter classid here"))) {
    L_Error_EnrollStudent.setText("Please provide valid classid (cid should be of length 5, and it starts with c)!");
    L_Error_EnrollStudent.setVisible(true);
    L_Message_EnrollStudent.setVisible(false);

} else {

    L_Error_EnrollStudent.setVisible(false);

    // @@ - enrolling student to class, showing exceptions
    try {

        String sid = T_sid_EnrollStudent.getText();
        String cid = T_cid_EnrollStudent.getText();

        // @@ - connect to DB
        OracleDataSource ds = new oracle.jdbc.pool.OracleDataSource();
        //ds.setURL("jdbc:oracle:thin:@localhost:1521:orcl");
        //Connection conn = ds.getConnection("shree", "shree");
        ds.setURL("jdbc:oracle:thin:@castor.cc.binghamton.edu:1521:ACAD111");
        conn = ds.getConnection(user, pass);

        // @@ - fetching result from DB usnig procedure
        CallableStatement call = conn.prepareCall("begin SQLPackage.enroll_student(?, ?); end;");
        call.setString(1, sid);
        call.setString(2, cid);
        Integer chk = call.executeUpdate();

        L_Message_EnrollStudent.setText("Enrollment successful!");
        L_Message_EnrollStudent.setForeground(Color.BLACK);
        L_Message_EnrollStudent.setVisible(true);

        conn.close();

    } catch (SQLException ex) {

        // @@ - show errors on panel
```

```
            if (ex.toString().contains("The sid is invalid.")) {
                L_Message_EnrollStudent.setText("The sid is invalid.");
                L_Message_EnrollStudent.setForeground(Color.RED);

            } else if (ex.toString().contains("The classid is invalid.")) {
                L_Message_EnrollStudent.setText("The classid is invalid.");
                L_Message_EnrollStudent.setForeground(Color.RED);

            } else if (ex.toString().contains("The class is closed.")) {
                L_Message_EnrollStudent.setText("The class is closed.");
                L_Message_EnrollStudent.setForeground(Color.RED);

            } else if (ex.toString().contains("The student is already in the class.")) {
                L_Message_EnrollStudent.setText("The student is already in the class.");
                L_Message_EnrollStudent.setForeground(Color.RED);

            } else if (ex.toString().contains("You are overloaded.")) {
                L_Message_EnrollStudent.setText("You are overloaded. Enrollment successful!");
                L_Message_EnrollStudent.setForeground(Color.BLACK);

            } else if (ex.toString().contains("Students cannot be enrolled in more than three classes in the same semester.")) {
                L_Message_EnrollStudent.setText("Students cannot be enrolled in more than three classes in the same semester.");
                L_Message_EnrollStudent.setForeground(Color.RED);

            } else if (ex.toString().contains("Prerequisite courses have not been completed.")) {
                L_Message_EnrollStudent.setText("Prerequisite courses have not been completed.");
                L_Message_EnrollStudent.setForeground(Color.RED);

            } else {
                L_Message_EnrollStudent.setText("SQL Exception : " + ex);
                L_Message_EnrollStudent.setForeground(Color.RED);
            }

            L_Message_EnrollStudent.setVisible(true);

            // @@ - handling too many session error by killing conn
            try {
                conn.close();
            } catch (SQLException ex1) {
                Logger.getLogger(Login.class.getName()).log(Level.FINE, null, ex1);
            }

        } catch (Exception e) {
```

```
            // @@ - show errors on panel
            L_Message_EnrollStudent.setText("Exception : " + e);
            L_Message_EnrollStudent.setForeground(Color.RED);
            L_Message_EnrollStudent.setVisible(true);

            // @@ - handling too many session error by killing conn
            try {
               conn.close();
            } catch (SQLException ex1) {
               Logger.getLogger(Login.class.getName()).log(Level.FINE, null, ex1);
            }
         }
      }
   }


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

      // @@ - resetting the panel
      T_sid_EnrollStudent.setText("Enter sid here");
      T_cid_EnrollStudent.setText("Enter classid here");
      L_Error_EnrollStudent.setVisible(false);
      L_Message_EnrollStudent.setVisible(false);
   }


   private void T_sid_DropEnrollmentFocusGained(java.awt.event.FocusEvent evt) {

      // @@ - clearing the default text on focus
      if (T_sid_DropEnrollment.getText().equals("Enter sid here")) {
         T_sid_DropEnrollment.setText("");
      }
      L_Error_DropEnrollment.setVisible(false);
   }


   private void T_sid_DropEnrollmentFocusLost(java.awt.event.FocusEvent evt) {

      // @@ - no value provided, resetting the text
      if (T_sid_DropEnrollment.getText().equals("")) {
         T_sid_DropEnrollment.setText("Enter sid here");
      }
```

```
}


private void T_cid_DropEnrollmentFocusGained(java.awt.event.FocusEvent evt) {

    // @@ - clearing the default text on focus
    if (T_cid_DropEnrollment.getText().equals("Enter classid here")) {
        T_cid_DropEnrollment.setText("");
    }
    L_Error_DropEnrollment.setVisible(false);
}


private void T_cid_DropEnrollmentFocusLost(java.awt.event.FocusEvent evt) {

    // @@ - no value provided, resetting the text
    if (T_cid_DropEnrollment.getText().equals("")) {
        T_cid_DropEnrollment.setText("Enter classid here");
    }
}


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

    // @@ - performing validation on details entered, if pass, fetch details, else report errors
    if ((!T_sid_DropEnrollment.getText().matches("B[0-9]+")) || (T_sid_DropEnrollment.getText().length() < 4) ||
    (T_sid_DropEnrollment.getText().equals("Enter (B-Number) sid here"))) {
        L_Error_DropEnrollment.setText("Please provide valid sid (sid should be of length 4, and it starts with B)!");
        L_Error_DropEnrollment.setVisible(true);
        L_Message_DropEnrollment.setVisible(false);

    } else if ((!T_cid_DropEnrollment.getText().matches("c[0-9]+")) || (T_cid_DropEnrollment.getText().equals("Enter classid here"))) {
        L_Error_DropEnrollment.setText("Please provide valid classid (cid should be of length 5, and it starts with c)!");
        L_Error_DropEnrollment.setVisible(true);
        L_Message_DropEnrollment.setVisible(false);

    } else {

        L_Error_DropEnrollment.setVisible(false);

        // @@ - enrolling student to class, showing exceptions
        try {
```

```
            String sid = T_sid_DropEnrollment.getText();
            String cid = T_cid_DropEnrollment.getText();

            // @@ - connect to DB
            OracleDataSource ds = new oracle.jdbc.pool.OracleDataSource();
            //ds.setURL("jdbc:oracle:thin:@localhost:1521:orcl");
            //Connection conn = ds.getConnection("shree", "shree");
            ds.setURL("jdbc:oracle:thin:@castor.cc.binghamton.edu:1521:ACAD111");
            conn = ds.getConnection(user, pass);

            // @@ - fetching result from DB usnig procedure
            CallableStatement call = conn.prepareCall("begin SQLPackage.drop_enrollment(?, ?); end;");
            call.setString(1, sid);
            call.setString(2, cid);
            Integer chk = call.executeUpdate();

            L_Message_DropEnrollment.setText("Enrollment dropped!");
            L_Message_DropEnrollment.setForeground(Color.BLACK);
            L_Message_DropEnrollment.setVisible(true);

            conn.close();

        } catch (SQLException ex) {

            // @@ - show errors on panel
            if (ex.toString().contains("The sid is invalid.")) {
                L_Message_DropEnrollment.setText("The sid is invalid.");
                L_Message_DropEnrollment.setForeground(Color.RED);

            } else if (ex.toString().contains("The classid is invalid.")) {
                L_Message_DropEnrollment.setText("The classid is invalid.");
                L_Message_DropEnrollment.setForeground(Color.RED);

            } else if (ex.toString().contains("The student is not enrolled in the class.")) {
                L_Message_DropEnrollment.setText("The student is not enrolled in the class.");
                L_Message_DropEnrollment.setForeground(Color.RED);

            } else if (ex.toString().contains("The drop is not permitted because another class uses it as a prerequisite.")) {
                L_Message_DropEnrollment.setText("The drop is not permitted because another class uses it as a prerequisite.");
                L_Message_DropEnrollment.setForeground(Color.RED);

            } else if (ex.toString().contains("This student is not enrolled in any classes.")) {
                L_Message_DropEnrollment.setText("Enrollment dropped! This student is not enrolled in any classes.");
```

```
            L_Message_DropEnrollment.setForeground(Color.BLACK);

        } else if (ex.toString().contains("The class now has no students..")) {
            L_Message_DropEnrollment.setText("Enrollment dropped! The class now has no students.");
            L_Message_DropEnrollment.setForeground(Color.BLACK);

        } else {
            L_Message_DropEnrollment.setText("SQL Exception : " + ex);
            L_Message_DropEnrollment.setForeground(Color.RED);
        }

        L_Message_DropEnrollment.setVisible(true);

        // @@ - handling too many session error by killing conn
        try {
            conn.close();
        } catch (SQLException ex1) {
            Logger.getLogger(Login.class.getName()).log(Level.FINE, null, ex1);
        }

    } catch (Exception e) {

        // @@ - show errors on panel
        L_Message_DropEnrollment.setText("Exception : " + e);
        L_Message_DropEnrollment.setForeground(Color.RED);
        L_Message_DropEnrollment.setVisible(true);

        // @@ - handling too many session error by killing conn
        try {
            conn.close();
        } catch (SQLException ex1) {
            Logger.getLogger(Login.class.getName()).log(Level.FINE, null, ex1);
        }
    }
  }
}


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

  // @@ - resetting the panel
  T_sid_DropEnrollment.setText("Enter sid here");
  T_cid_DropEnrollment.setText("Enter classid here");
```

```
        L_Error_DropEnrollment.setVisible(false);
        L_Message_DropEnrollment.setVisible(false);
    }



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

        // @@ - show only show all logs panel
        P_Default_Logs.setVisible(false);
        P_ShowAllLogs.setVisible(true);

        // @@ - hiding a table
        P_Table_ShowAllLogs.setVisible(false);

        // @@ - fecthing the data using procedure and display on GUI
        try {

            // @@ - connect to DB
            OracleDataSource ds = new oracle.jdbc.pool.OracleDataSource();
            //ds.setURL("jdbc:oracle:thin:@localhost:1521:orcl");
            //Connection conn = ds.getConnection("shree", "shree");
            ds.setURL("jdbc:oracle:thin:@castor.cc.binghamton.edu:1521:ACAD111");
            conn = ds.getConnection(user, pass);

            // @@ - fetching result from DB usnig procedure
            CallableStatement call = conn.prepareCall("begin SQLPackage.display_logs(?); end;");
            call.registerOutParameter(1, OracleTypes.CURSOR);
            call.execute();

            ResultSet rs = (ResultSet) call.getObject(1);

            // @@ - inserting data into the table on GUI
            DefaultTableModel model = (DefaultTableModel) TBL_ShowAllLogs.getModel();
            model.setRowCount(0);
            TBL_ShowAllLogs.setModel(model);

            int cols = TBL_ShowAllLogs.getColumnCount();

            while (rs.next()) {
                Object[] obj = new Object[cols];
                for (int i = 0; i < cols; i++) {
                    obj[i] = rs.getObject(i + 1);
                }
```

```
            model.addRow(obj);
        }

        TBL_ShowAllLogs.setModel(model);
        TBL_ShowAllLogs.getTableHeader().setFont(new Font("Monteserrat", Font.BOLD, 14));
        P_Table_ShowAllLogs.setVisible(true);

        L_Message_ShowAllLogs.setText("Displaying all available logs - ");
        L_Message_ShowAllLogs.setForeground(Color.BLACK);

        conn.close();

    } catch (SQLException ex) {

        // @@ - show errors on panel -
        L_Message_ShowAllLogs.setText("SQL Exception : " + ex);
        L_Message_ShowAllLogs.setForeground(Color.RED);

        // @@ - handling too many session error by killing conn
        try {
            conn.close();
        } catch (SQLException ex1) {
            Logger.getLogger(Login.class.getName()).log(Level.FINE, null, ex1);
        }

    } catch (Exception e) {

        // @@ - show errors on panel -
        L_Message_ShowAllLogs.setText("Exception : " + e);
        L_Message_ShowAllLogs.setForeground(Color.RED);

        // @@ - handling too many session error by killing conn
        try {
            conn.close();
        } catch (SQLException ex1) {
            Logger.getLogger(Login.class.getName()).log(Level.FINE, null, ex1);
        }
    }
}



private void B_ExitLogsActionPerformed(java.awt.event.ActionEvent evt) {
    System.exit(0);
```

```
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
         * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;

                }
            }
        } catch (ClassNotFoundException ex) {
            java.util.logging.Logger.getLogger(Home.class
                    .getName()).log(java.util.logging.Level.SEVERE, null, ex);

        } catch (InstantiationException ex) {
            java.util.logging.Logger.getLogger(Home.class
                    .getName()).log(java.util.logging.Level.SEVERE, null, ex);

        } catch (IllegalAccessException ex) {
            java.util.logging.Logger.getLogger(Home.class
                    .getName()).log(java.util.logging.Level.SEVERE, null, ex);

        } catch (javax.swing.UnsupportedLookAndFeelException ex) {
            java.util.logging.Logger.getLogger(Home.class
                    .getName()).log(java.util.logging.Level.SEVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new Home().setVisible(true);
            }
        });
```

```
    }


    // Variables declaration - do not modify
    private javax.swing.JButton B_AddNewStudent;
    private javax.swing.JButton B_Add_AddNewStudent;
    private javax.swing.JButton B_Cancel_AddNewStudent;
    private javax.swing.JButton B_Cancel_DeleteStudent;
    private javax.swing.JButton B_Cancel_DropEnrollment;
    private javax.swing.JButton B_Cancel_EnrollStudent;
    private javax.swing.JButton B_Cancel_FindClass;
    private javax.swing.JButton B_Cancel_FindCourse;
    private javax.swing.JButton B_Cancel_FindStudent;
    private javax.swing.JButton B_DeleteStudent;
    private javax.swing.JButton B_Delete_DeleteStudent;
    private javax.swing.JButton B_DropEnrollment;
    private javax.swing.JButton B_Drop_DropEnrollment;
    private javax.swing.JButton B_EnrollStudent;
    private javax.swing.JButton B_Enroll_EnrollStudent;
    private javax.swing.JButton B_ExitClasses;
    private javax.swing.JButton B_ExitCourses;
    private javax.swing.JButton B_ExitEnrollments;
    private javax.swing.JButton B_ExitLogs;
    private javax.swing.JButton B_ExitStudent;
    private javax.swing.JButton B_FindClass;
    private javax.swing.JButton B_FindCourse;
    private javax.swing.JButton B_FindStudent;
    private javax.swing.JButton B_Find_FindClass;
    private javax.swing.JButton B_Find_FindCourse;
    private javax.swing.JButton B_Find_FindStudent;
    private javax.swing.JButton B_ShowAllClasses;
    private javax.swing.JButton B_ShowAllCourses;
    private javax.swing.JButton B_ShowAllEnrollments;
    private javax.swing.JButton B_ShowAllLogs;
    private javax.swing.JButton B_ShowAllPrerequisites;
    private javax.swing.JButton B_ShowAllStudents;
    private javax.swing.JLayeredPane LP_Classes;
    private javax.swing.JLayeredPane LP_Courses;
    private javax.swing.JLayeredPane LP_Enrollments;
    private javax.swing.JLayeredPane LP_Logs;
    private javax.swing.JLayeredPane LP_Students;
    private javax.swing.JLabel L_Classid_FindClass;
    private javax.swing.JLabel L_Classid_Value_FindClass;
    private javax.swing.JLabel L_Error_AddNewStudent;
```

```
    private javax.swing.JLabel L_Error_DeleteStudent;
    private javax.swing.JLabel L_Error_DropEnrollment;
    private javax.swing.JLabel L_Error_EnrollStudent;
    private javax.swing.JLabel L_Error_FindClass;
    private javax.swing.JLabel L_Error_FindCourse;
    private javax.swing.JLabel L_Error_FindStudent;
    private javax.swing.JLabel L_Lastname_FindStudent;
    private javax.swing.JLabel L_Lastname_Value_FindStudent;
    private javax.swing.JLabel L_Message_AddNewStudent;
    private javax.swing.JLabel L_Message_Class_FindClass;
    private javax.swing.JLabel L_Message_Classes_FindStudent;
    private javax.swing.JLabel L_Message_DeleteStudent;
    private javax.swing.JLabel L_Message_DropEnrollment;
    private javax.swing.JLabel L_Message_EnrollStudent;
    private javax.swing.JLabel L_Message_FindCourse;
    private javax.swing.JLabel L_Message_ShowAllClasses;
    private javax.swing.JLabel L_Message_ShowAllCourses;
    private javax.swing.JLabel L_Message_ShowAllEnrollments;
    private javax.swing.JLabel L_Message_ShowAllLogs;
    private javax.swing.JLabel L_Message_ShowAllPre;
    private javax.swing.JLabel L_Message_ShowAllStudent;
    private javax.swing.JLabel L_Message_Student_FindStudent;
    private javax.swing.JLabel L_Message_Students_FindClass;
    private javax.swing.JLabel L_SID_FindStudent;
    private javax.swing.JLabel L_SID_Value_FindStudent;
    private javax.swing.JLabel L_Semester_FindClass;
    private javax.swing.JLabel L_Semester_Value_FindClass;
    private javax.swing.JLabel L_Status_FindStudent;
    private javax.swing.JLabel L_Status_Value_FindStudent;
    private javax.swing.JLabel L_Title_FindClass;
    private javax.swing.JLabel L_Title_Value_FindClass;
    private javax.swing.JLabel L_Year_FindClass;
    private javax.swing.JLabel L_Year_Value_FindClass;
    private javax.swing.JPanel P_AddNewStudent;
    private javax.swing.JPanel P_Button_Classes;
    private javax.swing.JPanel P_Button_Courses;
    private javax.swing.JPanel P_Button_Enrollments;
    private javax.swing.JPanel P_Button_Logs;
    private javax.swing.JPanel P_Button_Students;
    private javax.swing.JPanel P_ClassDetails_FindClass;
    private javax.swing.JPanel P_ClassDetails_FindStudents;
    private javax.swing.JPanel P_Default_Classes;
    private javax.swing.JPanel P_Default_Courses;
```

```
    private javax.swing.JPanel P_Default_Enrollments;
    private javax.swing.JPanel P_Default_Logs;
    private javax.swing.JPanel P_Default_Students;
    private javax.swing.JPanel P_DeleteStudent;
    private javax.swing.JPanel P_DropEnrollment;
    private javax.swing.JPanel P_EnrollStudent;
    private javax.swing.JPanel P_FindClass;
    private javax.swing.JPanel P_FindCourse;
    private javax.swing.JPanel P_FindStudent;
    private javax.swing.JPanel P_ManageEnrollments;
    private javax.swing.JPanel P_ManageStudents;
    private javax.swing.JPanel P_ShowAllClasses;
    private javax.swing.JPanel P_ShowAllCourses;
    private javax.swing.JPanel P_ShowAllEnrollments;
    private javax.swing.JPanel P_ShowAllLogs;
    private javax.swing.JPanel P_ShowAllPrerequisites;
    private javax.swing.JPanel P_ShowAllStudents;
    private javax.swing.JPanel P_ShowPre_FindCourse;
    private javax.swing.JPanel P_StudentDetails_FindClass;
    private javax.swing.JPanel P_StudentDetails_FindStudent;
    private javax.swing.JPanel P_Table_ShowAllClasses;
    private javax.swing.JPanel P_Table_ShowAllCourses;
    private javax.swing.JPanel P_Table_ShowAllEnrollments;
    private javax.swing.JPanel P_Table_ShowAllLogs;
    private javax.swing.JPanel P_Table_ShowAllPre;
    private javax.swing.JPanel P_Table_ShowAllStudents;
    private javax.swing.JPanel P_ViewClasses;
    private javax.swing.JPanel P_ViewCourses;
    private javax.swing.JPanel P_ViewLogs;
    private javax.swing.ButtonGroup RBG_Students;
    private javax.swing.JRadioButton RB_Freshman;
    private javax.swing.JRadioButton RB_Graduate;
    private javax.swing.JRadioButton RB_Junior;
    private javax.swing.JRadioButton RB_Senior;
    private javax.swing.JRadioButton RB_Sophomore;
    private javax.swing.JTable TBL_Classes_FindStudent;
    private javax.swing.JTable TBL_ShowAllClasses;
    private javax.swing.JTable TBL_ShowAllCourses;
    private javax.swing.JTable TBL_ShowAllEnrollments;
    private javax.swing.JTable TBL_ShowAllLogs;
    private javax.swing.JTable TBL_ShowAllPre;
    private javax.swing.JTable TBL_ShowAllStudents;
    private javax.swing.JTable TBL_ShowPre_FindCourse;
```

```
    private javax.swing.JTable TBL_Students_FindClass;
    private javax.swing.JTextField T_Course_FindCourse;
    private javax.swing.JTextField T_Dept_FindCourse;
    private javax.swing.JTextField T_Email;
    private javax.swing.JTextField T_FirstName;
    private javax.swing.JTextField T_GPA;
    private javax.swing.JTextField T_LastName;
    private javax.swing.JTextField T_cid_DropEnrollment;
    private javax.swing.JTextField T_cid_EnrollStudent;
    private javax.swing.JTextField T_classid_FindClass;
    private javax.swing.JTextField T_sid_DeleteStudent;
    private javax.swing.JTextField T_sid_DropEnrollment;
    private javax.swing.JTextField T_sid_EnrollStudent;
    private javax.swing.JTextField T_sid_FindStudent;
    private javax.swing.JPanel Tab_Classes;
    private javax.swing.JPanel Tab_Courses;
    private javax.swing.JPanel Tab_Enrollments;
    private javax.swing.JPanel Tab_Logs;
    private javax.swing.JPanel Tab_Students;
    private javax.swing.JTabbedPane Tabs;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel10;
    private javax.swing.JLabel jLabel11;
    private javax.swing.JLabel jLabel12;
    private javax.swing.JLabel jLabel13;
    private javax.swing.JLabel jLabel14;
    private javax.swing.JLabel jLabel15;
    private javax.swing.JLabel jLabel18;
    private javax.swing.JLabel jLabel19;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel20;
    private javax.swing.JLabel jLabel21;
    private javax.swing.JLabel jLabel22;
    private javax.swing.JLabel jLabel23;
    private javax.swing.JLabel jLabel24;
    private javax.swing.JLabel jLabel25;
    private javax.swing.JLabel jLabel26;
    private javax.swing.JLabel jLabel27;
    private javax.swing.JLabel jLabel28;
    private javax.swing.JLabel jLabel29;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel30;
    private javax.swing.JLabel jLabel31;
```

```
    private javax.swing.JLabel jLabel32;
    private javax.swing.JLabel jLabel33;
    private javax.swing.JLabel jLabel34;
    private javax.swing.JLabel jLabel35;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel41;
    private javax.swing.JLabel jLabel42;
    private javax.swing.JLabel jLabel43;
    private javax.swing.JLabel jLabel44;
    private javax.swing.JLabel jLabel45;
    private javax.swing.JLabel jLabel46;
    private javax.swing.JLabel jLabel47;
    private javax.swing.JLabel jLabel48;
    private javax.swing.JLabel jLabel49;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel50;
    private javax.swing.JLabel jLabel51;
    private javax.swing.JLabel jLabel52;
    private javax.swing.JLabel jLabel53;
    private javax.swing.JLabel jLabel54;
    private javax.swing.JLabel jLabel55;
    private javax.swing.JLabel jLabel56;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JLabel jLabel9;
    private javax.swing.JPanel jPanel5;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JScrollPane jScrollPane2;
    private javax.swing.JScrollPane jScrollPane3;
    private javax.swing.JScrollPane jScrollPane4;
    private javax.swing.JScrollPane jScrollPane5;
    private javax.swing.JScrollPane jScrollPane6;
    private javax.swing.JScrollPane jScrollPane7;
    private javax.swing.JScrollPane jScrollPane8;
    private javax.swing.JScrollPane jScrollPane9;
    private javax.swing.JSeparator jSeparator1;
    private javax.swing.JSeparator jSeparator11;
    private javax.swing.JSeparator jSeparator12;
    private javax.swing.JSeparator jSeparator13;
    private javax.swing.JSeparator jSeparator14;
    private javax.swing.JSeparator jSeparator15;
    private javax.swing.JSeparator jSeparator16;
```

```
    private javax.swing.JSeparator jSeparator2;
    private javax.swing.JSeparator jSeparator3;
    private javax.swing.JSeparator jSeparator4;
    private javax.swing.JSeparator jSeparator5;
    private javax.swing.JSeparator jSeparator6;
    private javax.swing.JSeparator jSeparator7;
    // End of variables declaration
}
```