Object Oriented Programming

COMP6699001 / Jude Joseph Lamug Martinez

Final Project

By : Christopher Owen / 2502019180 / L2AC

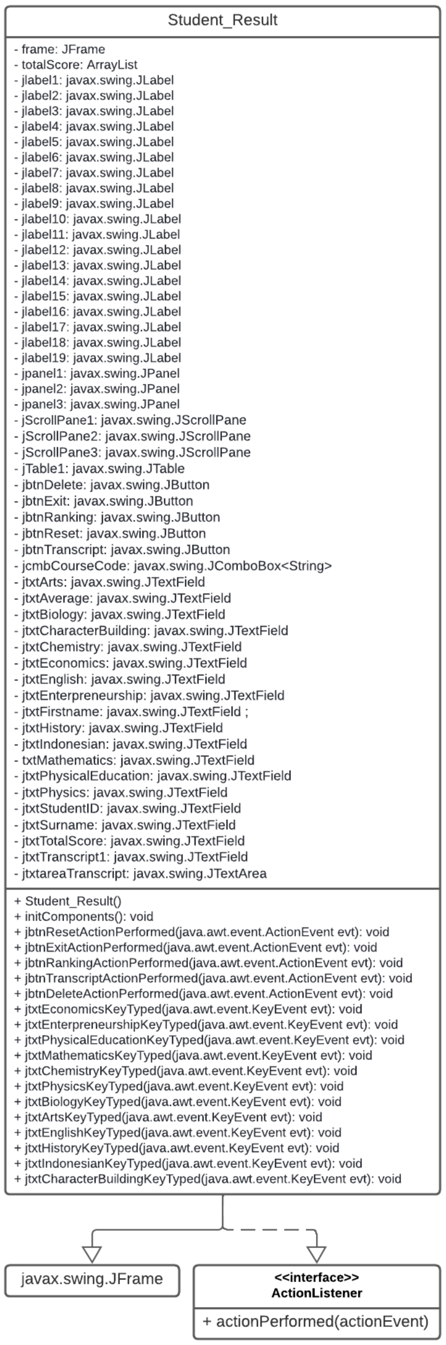
Table of Contents

Project Report: Student Result Recording System

I. Program Description

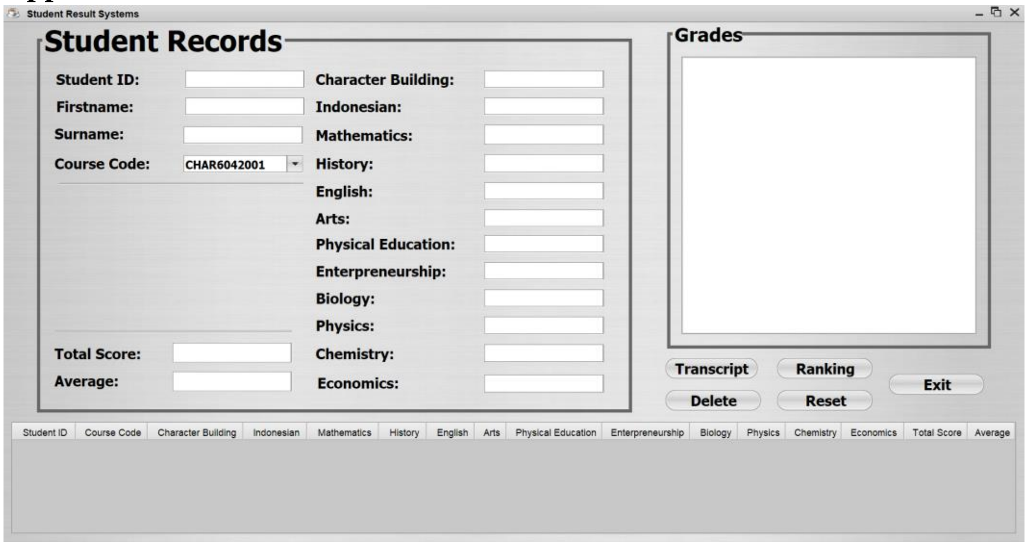
Student Result Recording System is a simple grading system for teachers or lecturers to input student's grades, rank, and transcript them. Using NetBeans IDE, the design of the application is fairly minimal and user friendly. The application uses javax.swing.JFrame for the interface. With this application, teachers can be facilitated and shorten their time in compiling student's grades.

II. Class Diagram



```
Student_Result
- frame: JFrame
- totalScore: ArrayList
- jlabel1: javax.swing.JLabel
- jlabel2: javax.swing.JLabel
- jlabel3: javax.swing.JLabel
- jlabel4: javax.swing.JLabel
- jlabel5: javax.swing.JLabel
- jlabel6: javax.swing.JLabel
- jlabel7: javax.swing.JLabel
- jlabel8: javax.swing.JLabel
- jlabel9: javax.swing.JLabel
- jlabel10: javax.swing.JLabel
- jlabel11: javax.swing.JLabel
- jlabel12: javax.swing.JLabel
- jlabel13: javax.swing.JLabel
- jlabel14: javax.swing.JLabel
- jlabel15: javax.swing.JLabel
- jlabel16: javax.swing.JLabel
- jlabel17: javax.swing.JLabel
- jlabel18: javax.swing.JLabel
- jlabel19: javax.swing.JLabel
- jpanel1: javax.swing.JPanel
- jpanel2: javax.swing.JPanel
- jpanel3: javax.swing.JPanel
- jScrollPane1: javax.swing.JScrollPane
- jScrollPane2: javax.swing.JScrollPane
- jScrollPane3: javax.swing.JScrollPane
- jTable1: javax.swing.JTable
- jbtnDelete: javax.swing.JButton
- jbtnExit: javax.swing.JButton
- jbtnRanking: javax.swing.JButton
- jbtnReset: javax.swing.JButton
- jbtnTranscript: javax.swing.JButton
- jcmbCourseCode: javax.swing.JComboBox<String>
- jtxtArts: javax.swing.JTextField
- jtxtAverage: javax.swing.JTextField
- jtxtBiology: javax.swing.JTextField
- jtxtCharacterBuilding: javax.swing.JTextField
- jtxtChemistry: javax.swing.JTextField
- jtxtEconomics: javax.swing.JTextField
- jtxtEnglish: javax.swing.JTextField
- jtxtEnterpreneurship: javax.swing.JTextField
- jtxtFirstname: javax.swing.JTextField ;
- jtxtHistory: javax.swing.JTextField
- jtxtIndonesian: javax.swing.JTextField
- txtMathematics: javax.swing.JTextField
- jtxtPhysicalEducation: javax.swing.JTextField
- jtxtPhysics: javax.swing.JTextField
- jtxtStudentID: javax.swing.JTextField
- jtxtSurname: javax.swing.JTextField
- jtxtTotalScore: javax.swing.JTextField
- jtxtTranscript1: javax.swing.JTextField
- jtxtareaTranscript: javax.swing.JTextArea
+ Student_Result()
+ initComponents(): void
+ jbtnResetActionPerformed(java.awt.event.ActionEvent evt): void
+ jbtnExitActionPerformed(java.awt.event.ActionEvent evt): void
+ jbtnRankingActionPerformed(java.awt.event.ActionEvent evt): void
+ jbtnTranscriptActionPerformed(java.awt.event.ActionEvent evt): void
+ jbtnDeleteActionPerformed(java.awt.event.ActionEvent evt): void
+ jtxtEconomicsKeyTyped(java.awt.event.KeyEvent evt): void
+ jtxtEnterpreneurshipKeyTyped(java.awt.event.KeyEvent evt): void
+ jtxtPhysicalEducationKeyTyped(java.awt.event.KeyEvent evt): void
+ jtxtMathematicsKeyTyped(java.awt.event.KeyEvent evt): void
+ jtxtChemistryKeyTyped(java.awt.event.KeyEvent evt): void
+ jtxtPhysicsKeyTyped(java.awt.event.KeyEvent evt): void
+ jtxtBiologyKeyTyped(java.awt.event.KeyEvent evt): void
+ jtxtArtsKeyTyped(java.awt.event.KeyEvent evt): void
+ jtxtEnglishKeyTyped(java.awt.event.KeyEvent evt): void
+ jtxtHistoryKeyTyped(java.awt.event.KeyEvent evt): void
+ jtxtIndonesianKeyTyped(java.awt.event.KeyEvent evt): void
+ jtxtCharacterBuildingKeyTyped(java.awt.event.KeyEvent evt): void
```

III. Application Flow

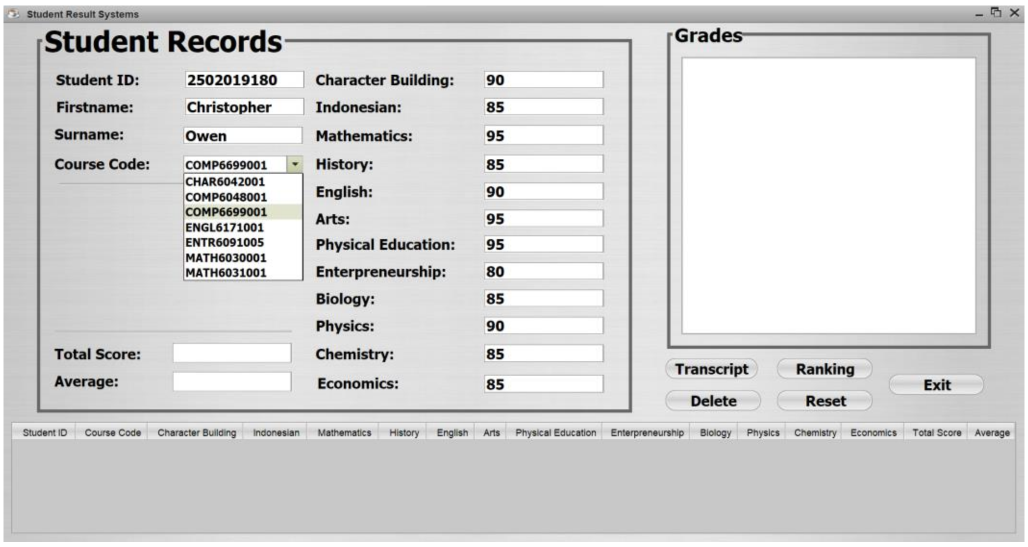

Student Result Systems

Student Records

Student ID:		Character Building:	
Firstname:		Indonesian:	
Surname:		Mathematics:	
Course Code:	CHAR6042001	History:	
		English:	
		Arts:	
		Physical Education:	
		Enterpreneurship:	
		Biology:	
		Physics:	
Total Score:		Chemistry:	
Average:		Economics:	

Student ID	Course Code	Character Building	Indonesian	Mathematics	History	English	Arts	Physical Education	Enterpreneurship	Biology	Physics	Chemistry	Economics	Total Score	Average

Program run, first appearance.

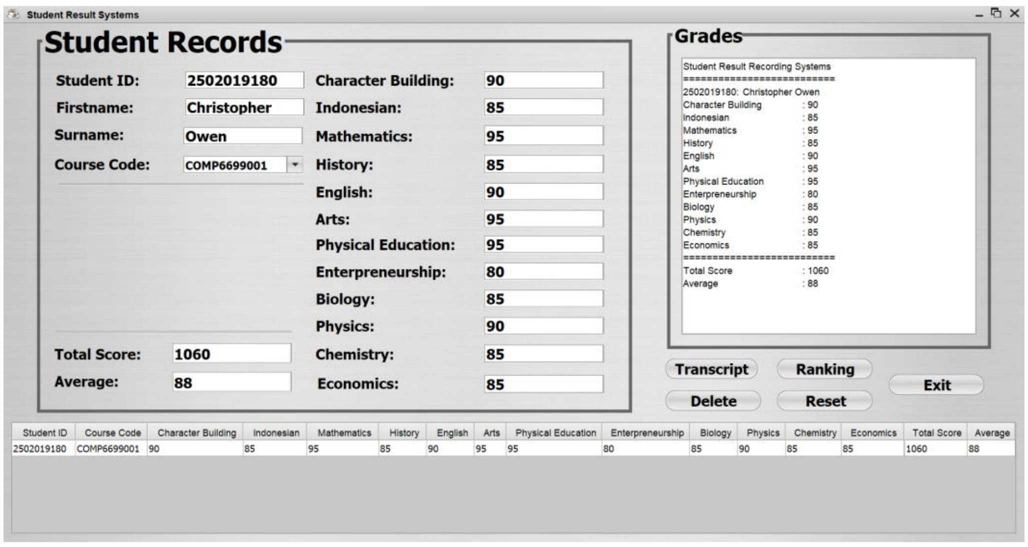

Student Result Systems

Student Records

Student ID:	2502019180	Character Building:	90
Firstname:	Christopher	Indonesian:	85
Surname:	Owen	Mathematics:	95
Course Code:	COMP6699001	History:	85
		English:	90
		Arts:	95
		Physical Education:	95
		Enterpreneurship:	80
		Biology:	85
		Physics:	90
Total Score:		Chemistry:	85
Average:		Economics:	85

Student ID	Course Code	Character Building	Indonesian	Mathematics	History	English	Arts	Physical Education	Enterpreneurship	Biology	Physics	Chemistry	Economics	Total Score	Average

Input Student ID, Firstname, Surname, scores, and choose Course Code.


Student Result Systems

Student Records

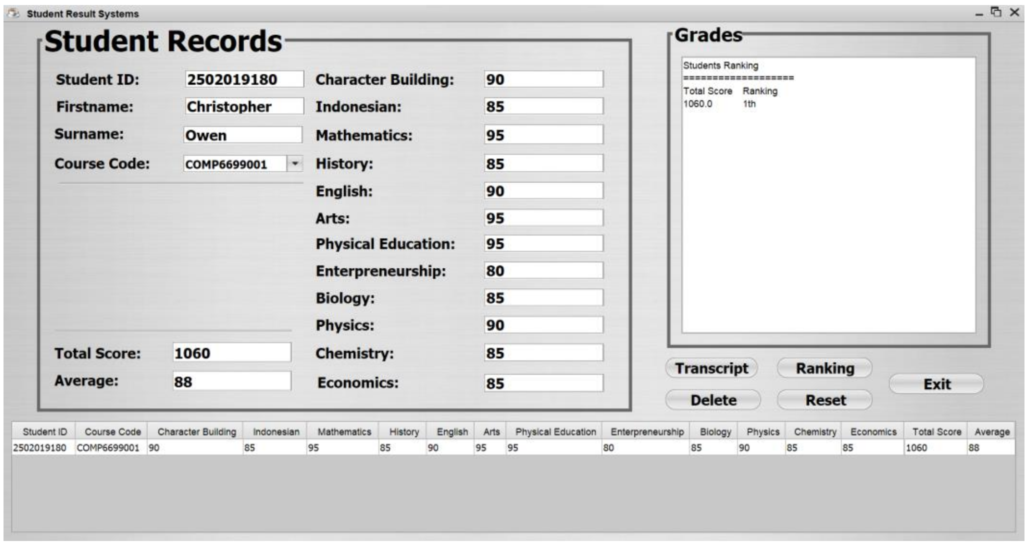
Student ID:	2502019180	Character Building:	90
Firstname:	Christopher	Indonesian:	85
Surname:	Owen	Mathematics:	95
Course Code:	COMP6699001	History:	85
		English:	90
		Arts:	95
		Physical Education:	95
		Enterpreneurship:	80
		Biology:	85
		Physics:	90
Total Score:	1060	Chemistry:	85
Average:	88	Economics:	85

Grades

```
Student Result Recording Systems
==========================
2502019180: Christopher Owen
Character Building      : 90
Indonesian              : 85
Mathematics             : 95
History                 : 85
English                 : 90
Arts                    : 95
Physical Education      : 95
Enterpreneurship        : 80
Biology                 : 85
Physics                 : 90
Chemistry               : 85
Economics               : 85
==========================
Total Score             : 1060
Average                 : 88
```

Student ID	Course Code	Character Building	Indonesian	Mathematics	History	English	Arts	Physical Education	Enterpreneurship	Biology	Physics	Chemistry	Economics	Total Score	Average
2502019180	COMP6699001	90	85	95	85	90	95	95	80	85	90	85	85	1060	88

Click Transcript to input everything to the table and transcript to Grades panel.

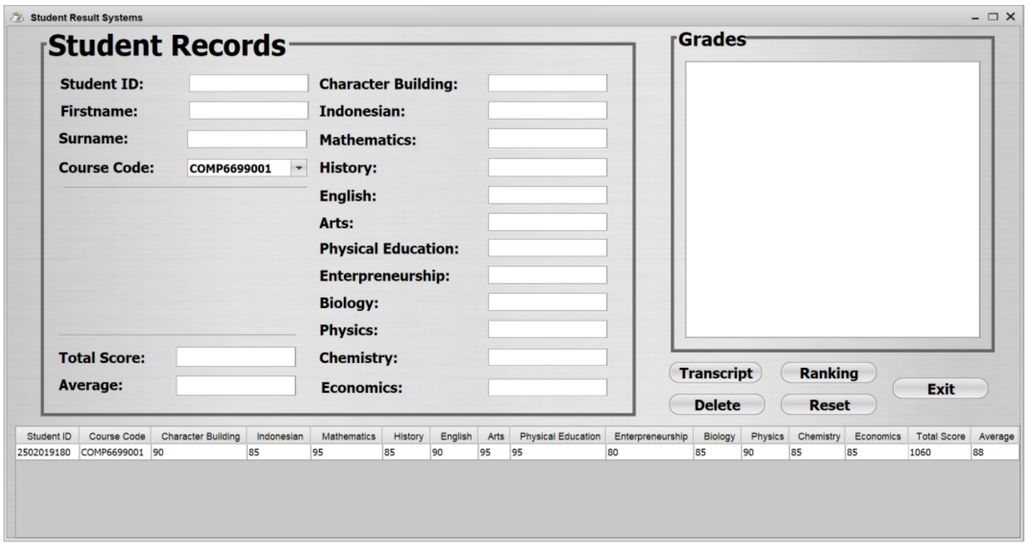


Student ID:	2502019180	Character Building:	90
Firstname:	Christopher	Indonesian:	85
Surname:	Owen	Mathematics:	95
Course Code:	COMP6699001	History:	85
		English:	90
		Arts:	95
		Physical Education:	95
		Enterpreneurship:	80
		Biology:	85
		Physics:	90
Total Score:	1060	Chemistry:	85
Average:	88	Economics:	85

```
Students Ranking
==================
Total Score    Ranking
1060.0         1th
```

Student ID	Course Code	Character Building	Indonesian	Mathematics	History	English	Arts	Physical Education	Enterpreneurship	Biology	Physics	Chemistry	Economics	Total Score	Average
2502019180	COMP6699001	90	85	95	85	90	95	95	80	85	90	85	85	1060	88

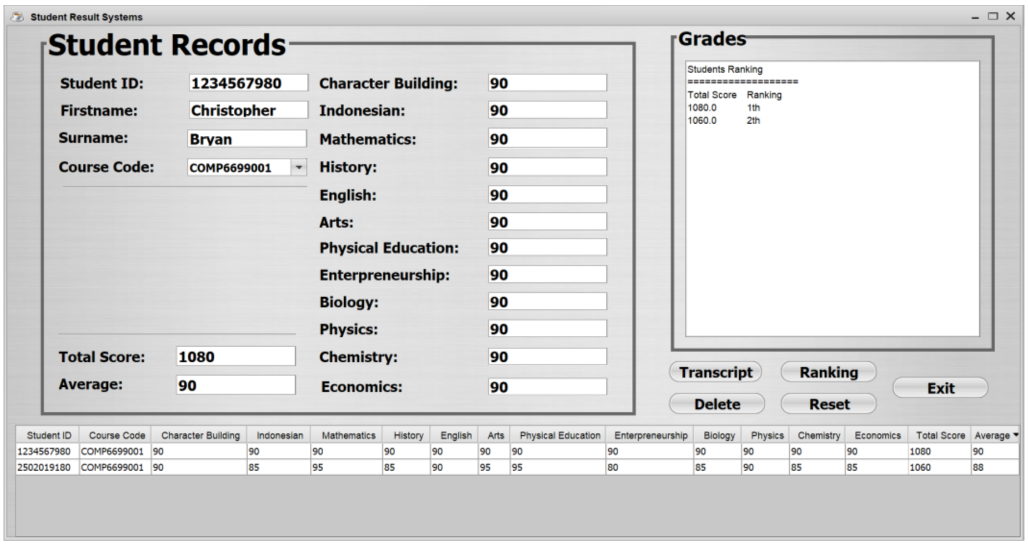
Click Ranking to view the ranks in the Grades panel



Student ID:		Character Building:	
Firstname:		Indonesian:	
Surname:		Mathematics:	
Course Code:	COMP6699001	History:	
		English:	
		Arts:	
		Physical Education:	
		Enterpreneurship:	
		Biology:	
		Physics:	
Total Score:		Chemistry:	
Average:		Economics:	

Student ID	Course Code	Character Building	Indonesian	Mathematics	History	English	Arts	Physical Education	Enterpreneurship	Biology	Physics	Chemistry	Economics	Total Score	Average
2502019180	COMP6699001	90	85	95	85	90	95	95	80	85	90	85	85	1060	88

Click Reset to clear all inputs and Grades panel.

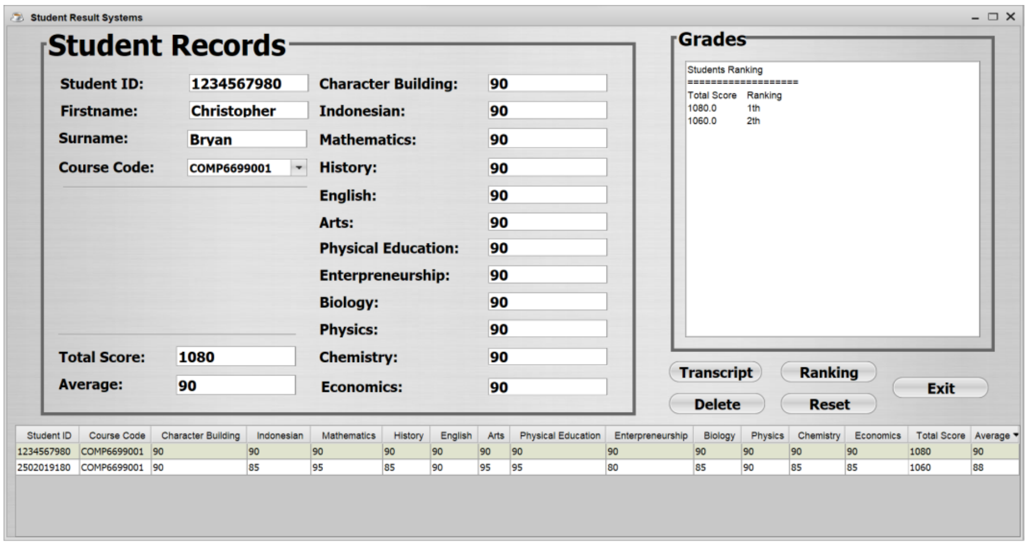


Student ID:	1234567980	Character Building:	90
Firstname:	Christopher	Indonesian:	90
Surname:	Bryan	Mathematics:	90
Course Code:	COMP6699001	History:	90
		English:	90
		Arts:	90
		Physical Education:	90
		Enterpreneurship:	90
		Biology:	90
		Physics:	90
Total Score:	1080	Chemistry:	90
Average:	90	Economics:	90

```
Students Ranking
==================
Total Score    Ranking
1080.0         1th
1060.0         2th
```

Student ID	Course Code	Character Building	Indonesian	Mathematics	History	English	Arts	Physical Education	Enterpreneurship	Biology	Physics	Chemistry	Economics	Total Score	Average
1234567980	COMP6699001	90	90	90	90	90	90	90	90	90	90	90	90	1080	90
2502019180	COMP6699001	90	85	95	85	90	95	95	80	85	90	85	85	1060	88

After inputting everything for the second person, click Transcript to add into table and click Ranking to view the rank between them.


Student Result Systems

Student Records

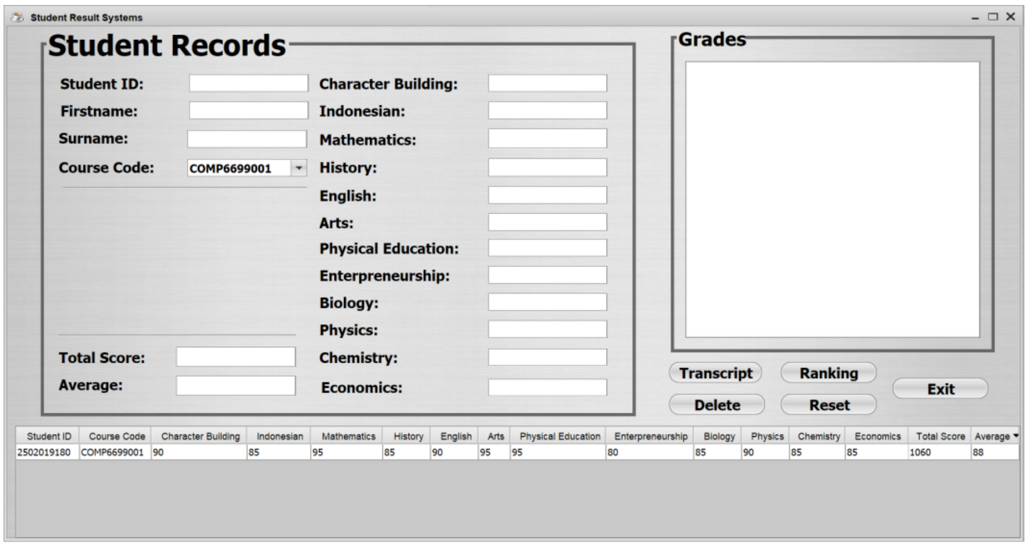
Student ID:	1234567980	Character Building:	90
Firstname:	Christopher	Indonesian:	90
Surname:	Bryan	Mathematics:	90
Course Code:	COMP6699001	History:	90
		English:	90
		Arts:	90
		Physical Education:	90
		Enterpreneurship:	90
		Biology:	90
		Physics:	90
Total Score:	1080	Chemistry:	90
Average:	90	Economics:	90

Grades

```
Students Ranking
====================
Total Score   Ranking
1080.0        1th
1060.0        2th
```

Student ID	Course Code	Character Building	Indonesian	Mathematics	History	English	Arts	Physical Education	Enterpreneurship	Biology	Physics	Chemistry	Economics	Total Score	Average
1234567980	COMP6699001	90	90	90	90	90	90	90	90	90	90	90	90	1080	90
2502019180	COMP6699001	90	85	95	85	90	95	95	80	85	90	85	85	1060	88

Select a row in the table to delete.

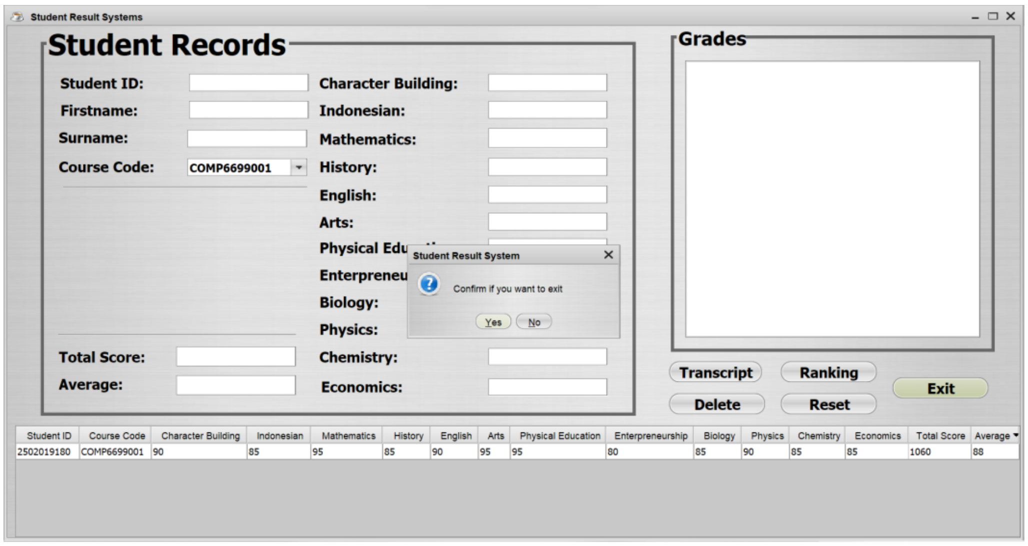

Student Result Systems

Student Records

Student ID:		Character Building:	
Firstname:		Indonesian:	
Surname:		Mathematics:	
Course Code:	COMP6699001	History:	
		English:	
		Arts:	
		Physical Education:	
		Enterpreneurship:	
		Biology:	
		Physics:	
Total Score:		Chemistry:	
Average:		Economics:	

Student ID	Course Code	Character Building	Indonesian	Mathematics	History	English	Arts	Physical Education	Enterpreneurship	Biology	Physics	Chemistry	Economics	Total Score	Average
2502019180	COMP6699001	90	85	95	85	90	95	95	80	85	90	85	85	1060	88

Click Delete to delete a row in the table


Student Result Systems

Student Records

Student ID:		Character Building:	
Firstname:		Indonesian:	
Surname:		Mathematics:	
Course Code:	COMP6699001	History:	
		English:	
		Arts:	
		Physical Edu	
		Enterpreneu	
		Biology:	
		Physics:	
Total Score:		Chemistry:	
Average:		Economics:	

Student Result System

Confirm if you want to exit

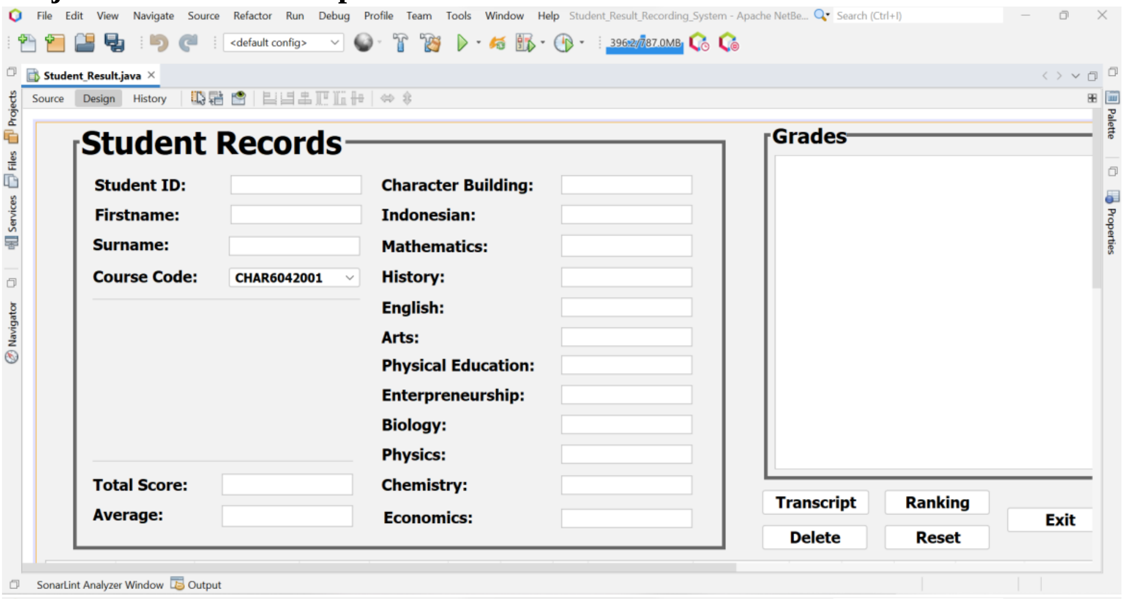
Student ID	Course Code	Character Building	Indonesian	Mathematics	History	English	Arts	Physical Education	Enterpreneurship	Biology	Physics	Chemistry	Economics	Total Score	Average
2502019180	COMP6699001	90	85	95	85	90	95	95	80	85	90	85	85	1060	88

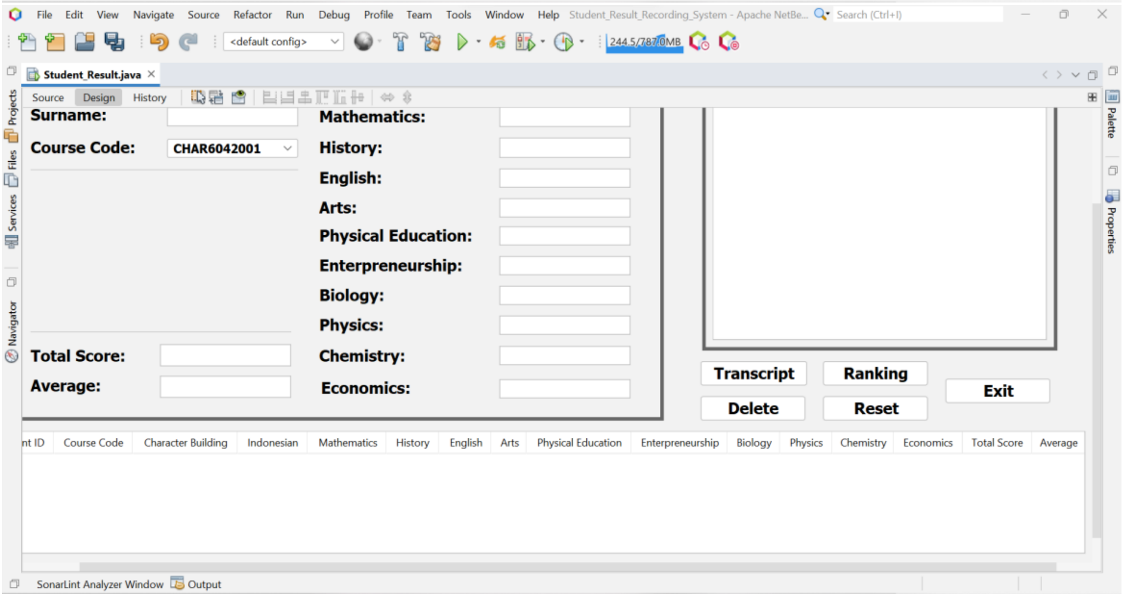
Click Exit to exit the application.

IV. Lessons that Have Been Learned

In this project, I have learned new Java libraries that can be very useful in making an application based on pure Java, such as java.awt.Component, java.awt.event.KeyEvent, java.util.ArrayList, and many more. Along with their classes, I learned new implementations of Java libraries that make the application more interactive. Using NetBeans as an IDE is one of the benefits of making the interface for the application. With this project, I get to learn, understand, and improve myself on Java and OOP.

V. Project Technical Description





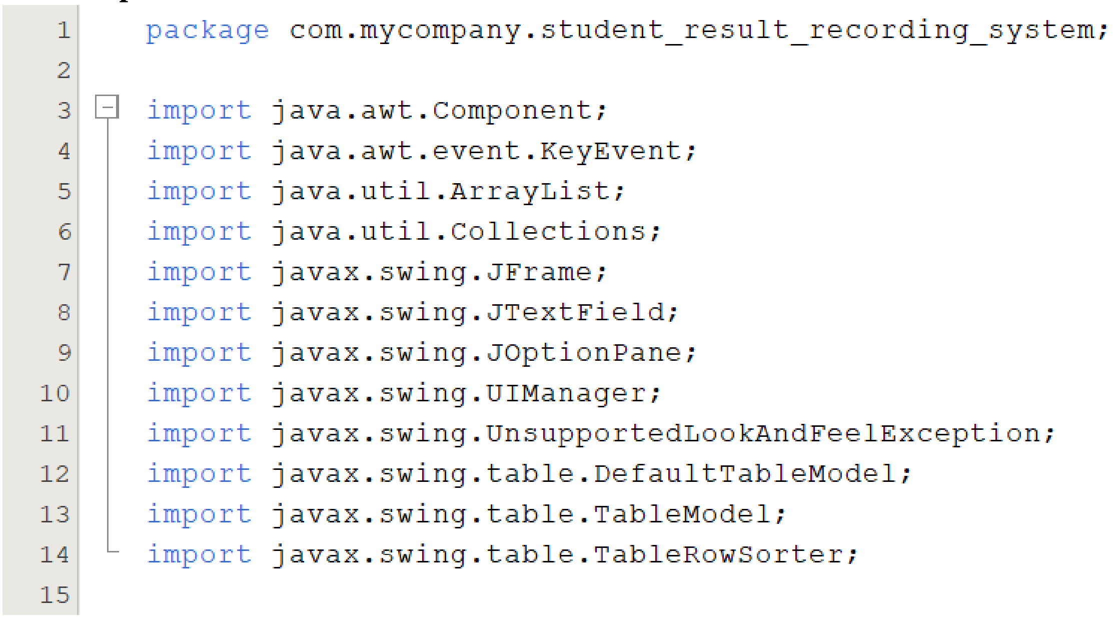
In this project, I used NetBeans as an IDE because I can easily design the interface of the application system using built-in JFrame from JDK. Using its palettes and properties, I can easily position, resize, and name all the constructors visually based on how I want them to look.

javax.swing.JFrame

JFrame is a class imported from java.awt and the extension of java.awt.Frame. It has the constructors and methods in making an API based on pure Java. In this project, I use this

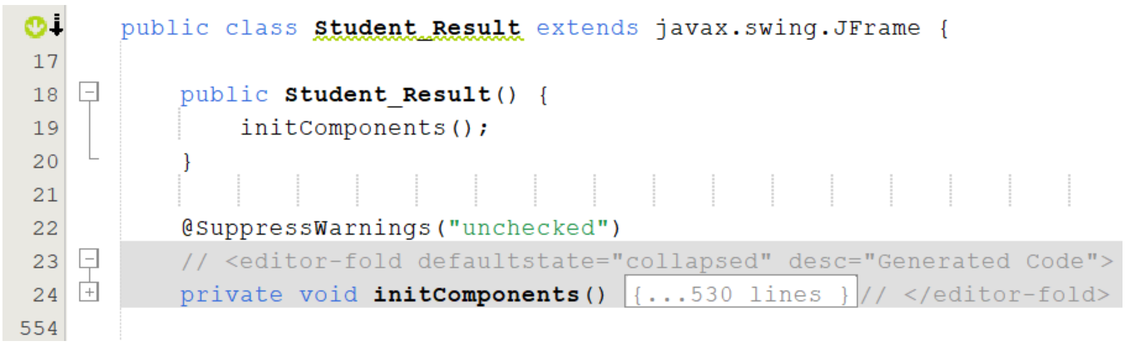
API for the simplicity and showcasing Java libraries in making a useful application using only Java.

VI. Code Explanation



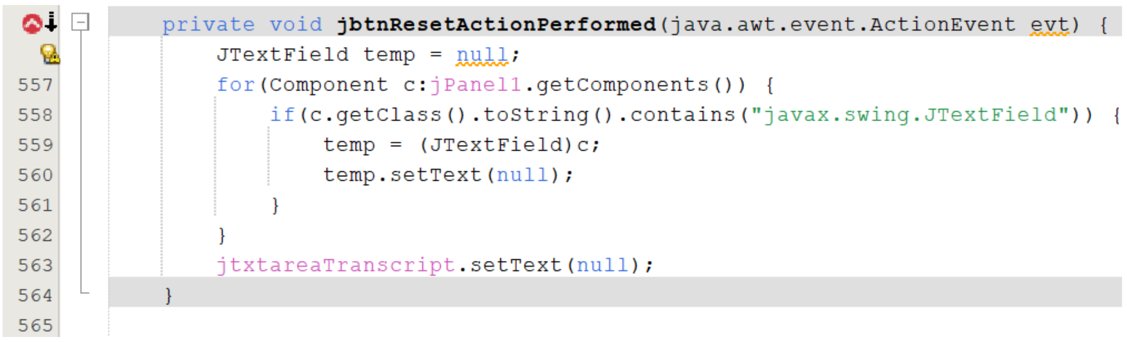
```java
package com.mycompany.student_result_recording_system;

import java.awt.Component;
import java.awt.event.KeyEvent;
import java.util.ArrayList;
import java.util.Collections;
import javax.swing.JFrame;
import javax.swing.JTextField;
import javax.swing.JOptionPane;
import javax.swing.UIManager;
import javax.swing.UnsupportedLookAndFeelException;
import javax.swing.table.DefaultTableModel;
import javax.swing.table.TableModel;
import javax.swing.table.TableRowSorter;

```

Here are the Java libraries that are used in the application.
java.awt.Component will help other libraries to be called.
java.awt.event.KetEvent will be the library that is used for consuming characters that is not erased.
java.util.ArrayList is used for declaring and using arraylists within the code.
java.util.Collections is used for sorting arraylists in ascending or descending order.
javax.swing.JFrame provides all constructors and methods to build the layout of the interface.
javax.swing.JTextField provides necessary constructors and methods for accessing the JTextField
javax.swing.JOptionPane provides necessary constructors and methods in making options for the user to proceed.
javax.swing.UIManager provides the ability to decorate the interface as the user's choice.
javax.swing.UnsupportedLookAndFeelException allows the user to use external decorative interfaces to be used.
javax.swing.table.DefaultTableModel and javax.swing.table.TableModel provides the elements for designing a table.
javax.swing.table.TableRowSorter provides the ability to sort the table in ascending or descending order.



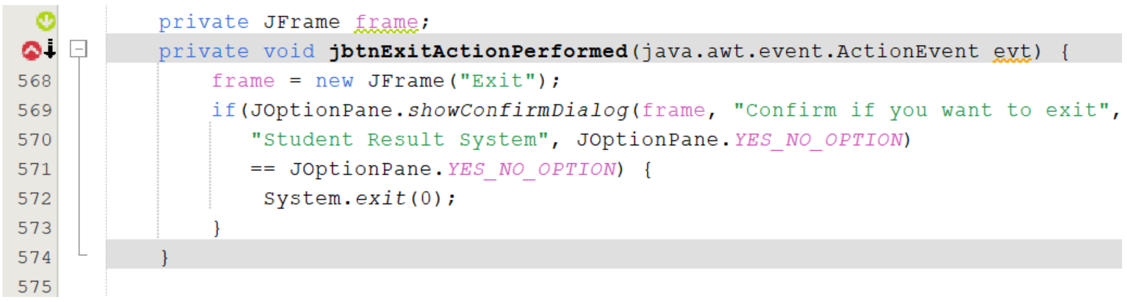
```java
public class Student_Result extends javax.swing.JFrame {

    public Student_Result() {
        initComponents();
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {...530 lines }// </editor-fold>
```

The Student_Result class extends javax.swing.JFrame class form the library inheriting all constructors and methods within the class. The Student_Result function will call initComponents function which is a generated code from designing the layout of the interface using the design feature from NetBeans. The initComponents function contains all the initialization, declarations, positions, sizes, fonts, groups, and spaces of the constructors and methods that are already designed



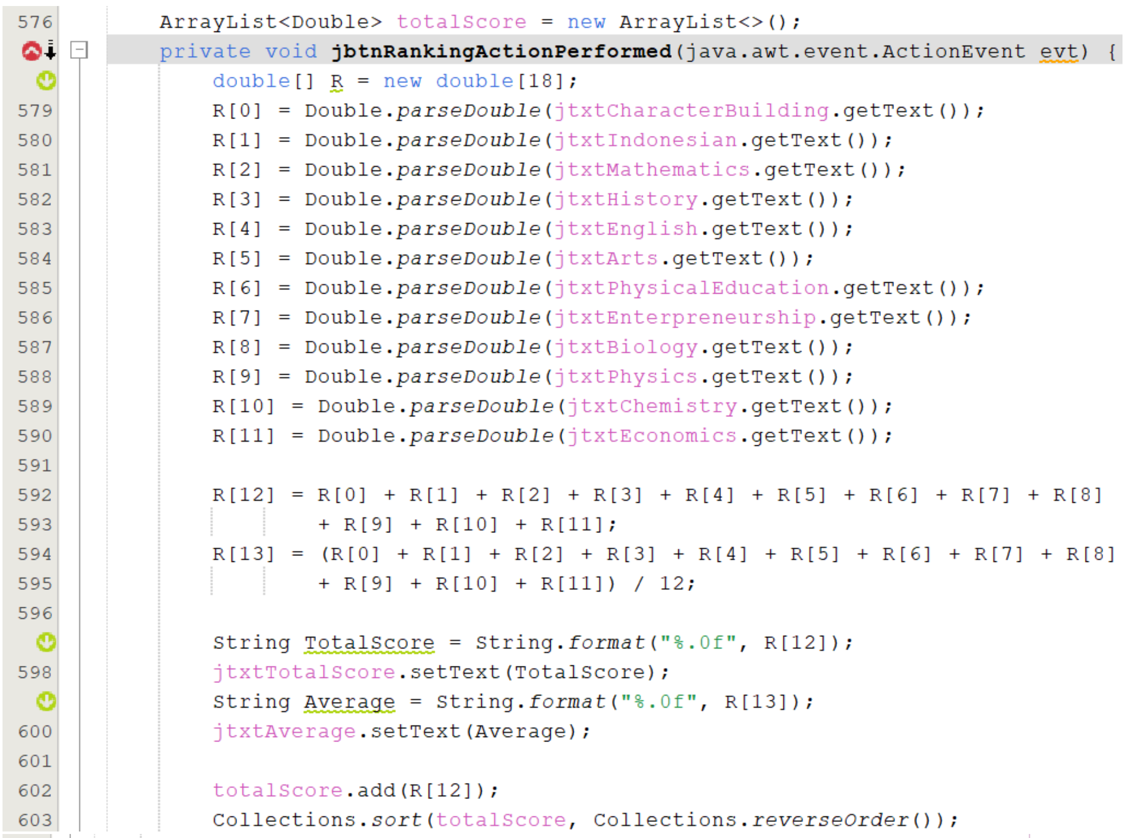
```java
    private void jbtnResetActionPerformed(java.awt.event.ActionEvent evt) {
        JTextField temp = null;
        for(Component c:jPanel1.getComponents()) {
            if(c.getClass().toString().contains("javax.swing.JTextField")) {
                temp = (JTextField)c;
                temp.setText(null);
            }
        }
        jtxtareaTranscript.setText(null);
    }
```

jbtnResetActionPerformed function indicates the functionality of the Reset button in the application. First, it will declare temp as null under the JTextField constructor. Then, all JTextField in jPanel1 will become null or cleared from any inputted text. Also, it will clear jtxtareaTranscript from any printed text.

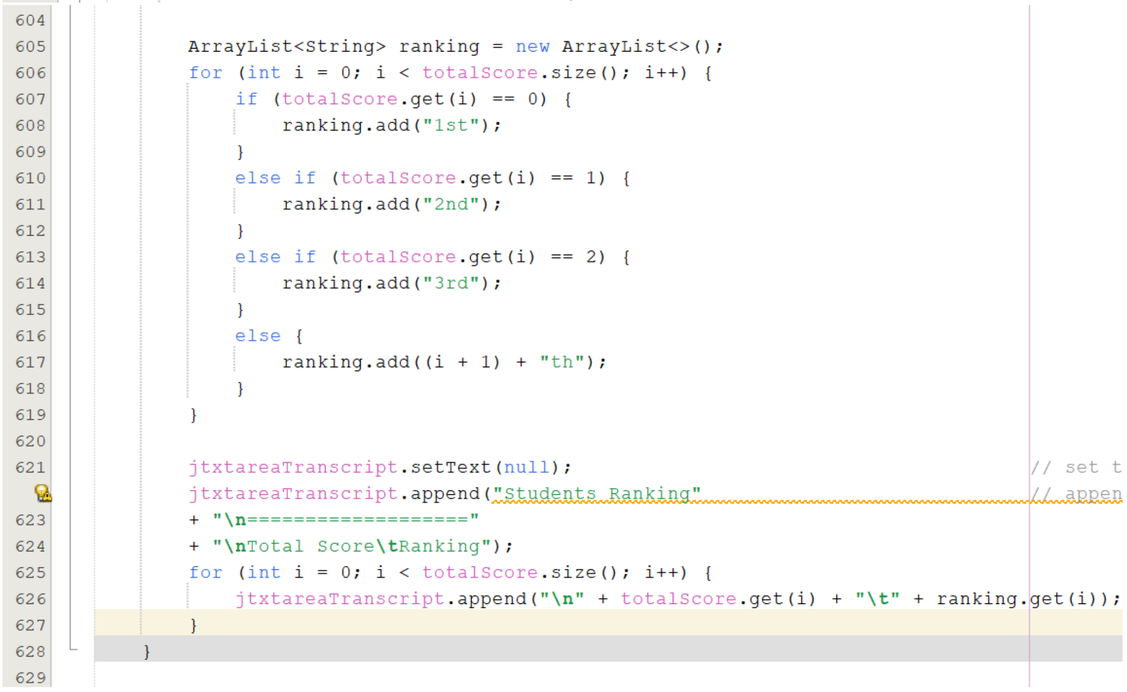


```java
    private JFrame frame;
    private void jbtnExitActionPerformed(java.awt.event.ActionEvent evt) {
        frame = new JFrame("Exit");
        if(JOptionPane.showConfirmDialog(frame, "Confirm if you want to exit",
           "Student Result System", JOptionPane.YES_NO_OPTION)
           == JOptionPane.YES_NO_OPTION) {
            System.exit(0);
        }
    }
```

jbtnExitActionPerformed function indicates the functionality of the Exit button in the application. First, it will initiate a new JFrame named Exit. Then, it will show a confirmation message using JOptionPane titled Student Result System that contains Confirm if you want to exit message, along with yes or no option from JOptionPane constructor. If it returns true or the user click yes, it will exit the application. If the user chooses no, it will return false and continue the application.

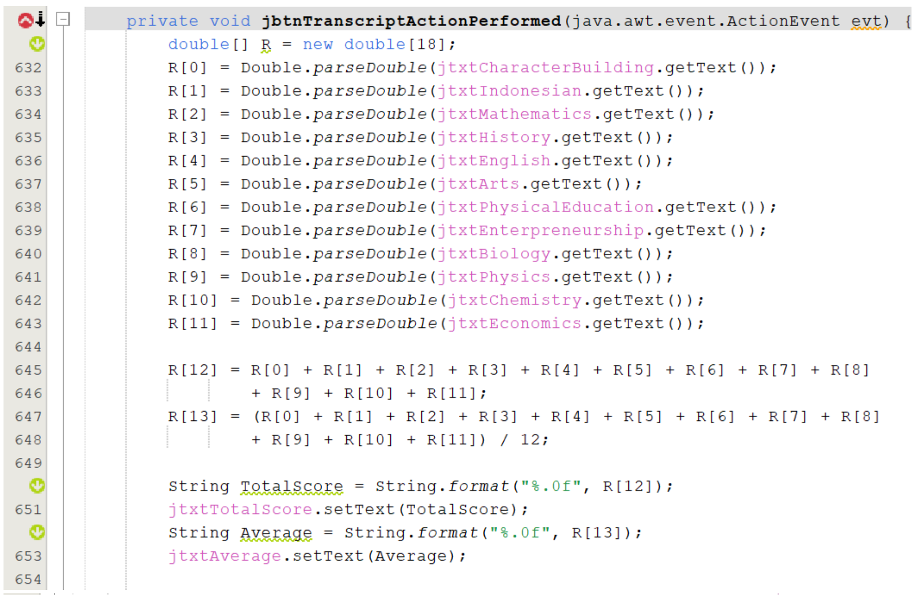


```java
    ArrayList<Double> totalScore = new ArrayList<>();
    private void jbtnRankingActionPerformed(java.awt.event.ActionEvent evt) {
        double[] R = new double[18];
        R[0] = Double.parseDouble(jtxtCharacterBuilding.getText());
        R[1] = Double.parseDouble(jtxtIndonesian.getText());
        R[2] = Double.parseDouble(jtxtMathematics.getText());
        R[3] = Double.parseDouble(jtxtHistory.getText());
        R[4] = Double.parseDouble(jtxtEnglish.getText());
        R[5] = Double.parseDouble(jtxtArts.getText());
        R[6] = Double.parseDouble(jtxtPhysicalEducation.getText());
        R[7] = Double.parseDouble(jtxtEnterpreneurship.getText());
        R[8] = Double.parseDouble(jtxtBiology.getText());
        R[9] = Double.parseDouble(jtxtPhysics.getText());
        R[10] = Double.parseDouble(jtxtChemistry.getText());
        R[11] = Double.parseDouble(jtxtEconomics.getText());

        R[12] = R[0] + R[1] + R[2] + R[3] + R[4] + R[5] + R[6] + R[7] + R[8]
                + R[9] + R[10] + R[11];
        R[13] = (R[0] + R[1] + R[2] + R[3] + R[4] + R[5] + R[6] + R[7] + R[8]
                + R[9] + R[10] + R[11]) / 12;

        String TotalScore = String.format("%.0f", R[12]);
        jtxtTotalScore.setText(TotalScore);
        String Average = String.format("%.0f", R[13]);
        jtxtAverage.setText(Average);

        totalScore.add(R[12]);
        Collections.sort(totalScore, Collections.reverseOrder());
```

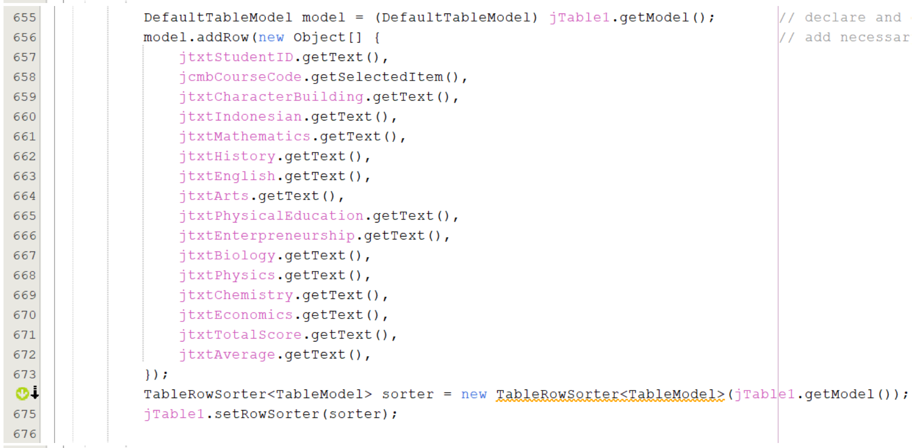


```java

        ArrayList<String> ranking = new ArrayList<>();
        for (int i = 0; i < totalScore.size(); i++) {
            if (totalScore.get(i) == 0) {
                ranking.add("1st");
            }
            else if (totalScore.get(i) == 1) {
                ranking.add("2nd");
            }
            else if (totalScore.get(i) == 2) {
                ranking.add("3rd");
            }
            else {
                ranking.add((i + 1) + "th");
            }
        }

        jtxtareaTranscript.setText(null);                                   // set t
        jtxtareaTranscript.append("Students Ranking"                        // appen
        + "\n==================="
        + "\nTotal Score\tRanking");
        for (int i = 0; i < totalScore.size(); i++) {
            jtxtareaTranscript.append("\n" + totalScore.get(i) + "\t" + ranking.get(i));
        }
    }
```

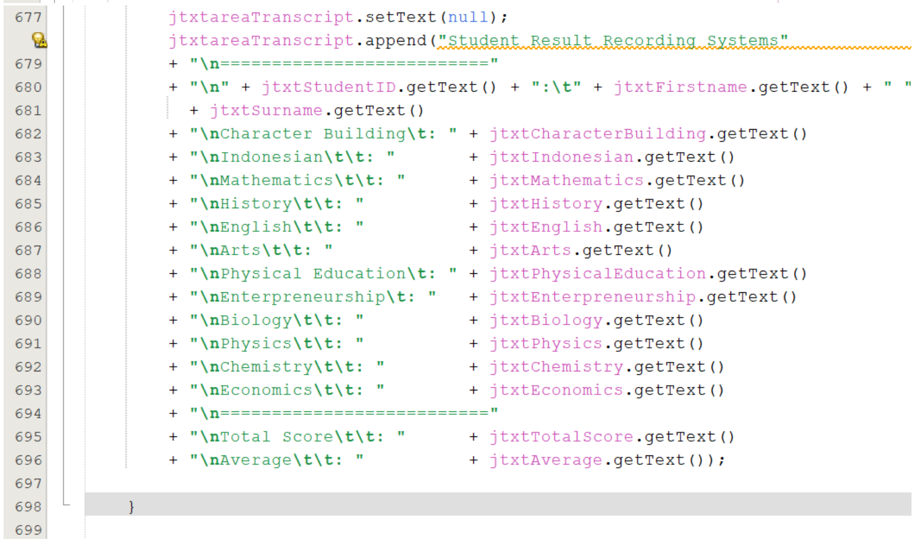
jbtnRankingActionPerformed function indicates the functionality of the Ranking button in the application. First, it will declare an array as a container for the scores, total score, and average to be inputted. Then, adding the total score to the newly declared arraylist and sort it in descending order. Then, declare a new arraylist containing the ranking descriptions of the total score array list. Finally, the total score and the ranking descriptions will be printed in the jtxtareaTranscript.



```
    private void jbtnTranscriptActionPerformed(java.awt.event.ActionEvent evt) {
        double[] R = new double[18];
        R[0] = Double.parseDouble(jtxtCharacterBuilding.getText());
        R[1] = Double.parseDouble(jtxtIndonesian.getText());
        R[2] = Double.parseDouble(jtxtMathematics.getText());
        R[3] = Double.parseDouble(jtxtHistory.getText());
        R[4] = Double.parseDouble(jtxtEnglish.getText());
        R[5] = Double.parseDouble(jtxtArts.getText());
        R[6] = Double.parseDouble(jtxtPhysicalEducation.getText());
        R[7] = Double.parseDouble(jtxtEnterpreneurship.getText());
        R[8] = Double.parseDouble(jtxtBiology.getText());
        R[9] = Double.parseDouble(jtxtPhysics.getText());
        R[10] = Double.parseDouble(jtxtChemistry.getText());
        R[11] = Double.parseDouble(jtxtEconomics.getText());

        R[12] = R[0] + R[1] + R[2] + R[3] + R[4] + R[5] + R[6] + R[7] + R[8]
                + R[9] + R[10] + R[11];
        R[13] = (R[0] + R[1] + R[2] + R[3] + R[4] + R[5] + R[6] + R[7] + R[8]
                + R[9] + R[10] + R[11]) / 12;

        String TotalScore = String.format("%.0f", R[12]);
        jtxtTotalScore.setText(TotalScore);
        String Average = String.format("%.0f", R[13]);
        jtxtAverage.setText(Average);

```

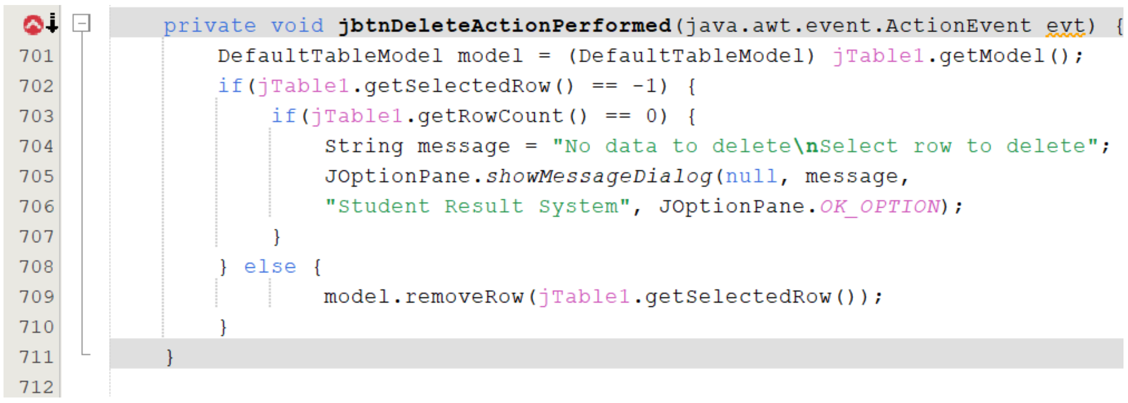


```
        DefaultTableModel model = (DefaultTableModel) jTable1.getModel();       // declare and
        model.addRow(new Object[] {                                             // add necessar
            jtxtStudentID.getText(),
            jcmbCourseCode.getSelectedItem(),
            jtxtCharacterBuilding.getText(),
            jtxtIndonesian.getText(),
            jtxtMathematics.getText(),
            jtxtHistory.getText(),
            jtxtEnglish.getText(),
            jtxtArts.getText(),
            jtxtPhysicalEducation.getText(),
            jtxtEnterpreneurship.getText(),
            jtxtBiology.getText(),
            jtxtPhysics.getText(),
            jtxtChemistry.getText(),
            jtxtEconomics.getText(),
            jtxtTotalScore.getText(),
            jtxtAverage.getText(),
        });
        TableRowSorter<TableModel> sorter = new TableRowSorter<TableModel>(jTable1.getModel());
        jTable1.setRowSorter(sorter);

```



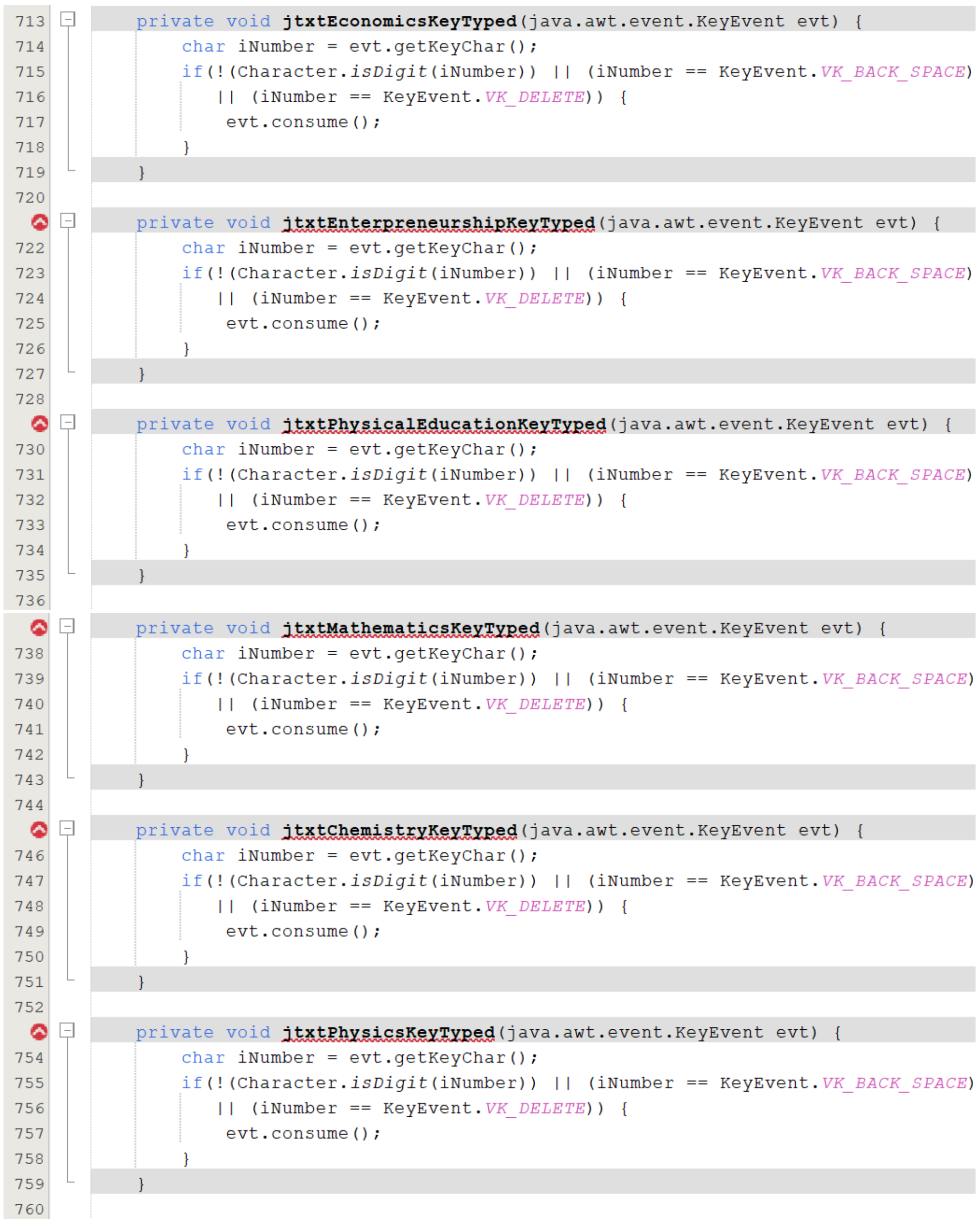
```
        jtxtareaTranscript.setText(null);
        jtxtareaTranscript.append("Student Result Recording Systems"
        + "\n========================="
        + "\n" + jtxtStudentID.getText() + ":\t" + jtxtFirstname.getText() + " "
          + jtxtSurname.getText()
        + "\nCharacter Building\t: " + jtxtCharacterBuilding.getText()
        + "\nIndonesian\t\t: "       + jtxtIndonesian.getText()
        + "\nMathematics\t\t: "      + jtxtMathematics.getText()
        + "\nHistory\t\t: "          + jtxtHistory.getText()
        + "\nEnglish\t\t: "          + jtxtEnglish.getText()
        + "\nArts\t\t: "             + jtxtArts.getText()
        + "\nPhysical Education\t: " + jtxtPhysicalEducation.getText()
        + "\nEnterpreneurship\t: "   + jtxtEnterpreneurship.getText()
        + "\nBiology\t\t: "          + jtxtBiology.getText()
        + "\nPhysics\t\t: "          + jtxtPhysics.getText()
        + "\nChemistry\t\t: "        + jtxtChemistry.getText()
        + "\nEconomics\t\t: "        + jtxtEconomics.getText()
        + "\n=========================="
        + "\nTotal Score\t\t: "      + jtxtTotalScore.getText()
        + "\nAverage\t\t: "          + jtxtAverage.getText());

    }

```

jbtnTranscriptActionPerformed function indicates the functionality of the Transcript button in the application. Similar to the ranking button, it will declare an array as a container for the scores, total score, and average to be inputted. Then, it will call the table and add a row in the table containing the components that are already inputted. Also, there is a method which the table can be sorted by column so it will ease the user to see the elements, either in ascending or descending order. Finally, it will print all the elements inputted to jtxtareaTranscript.

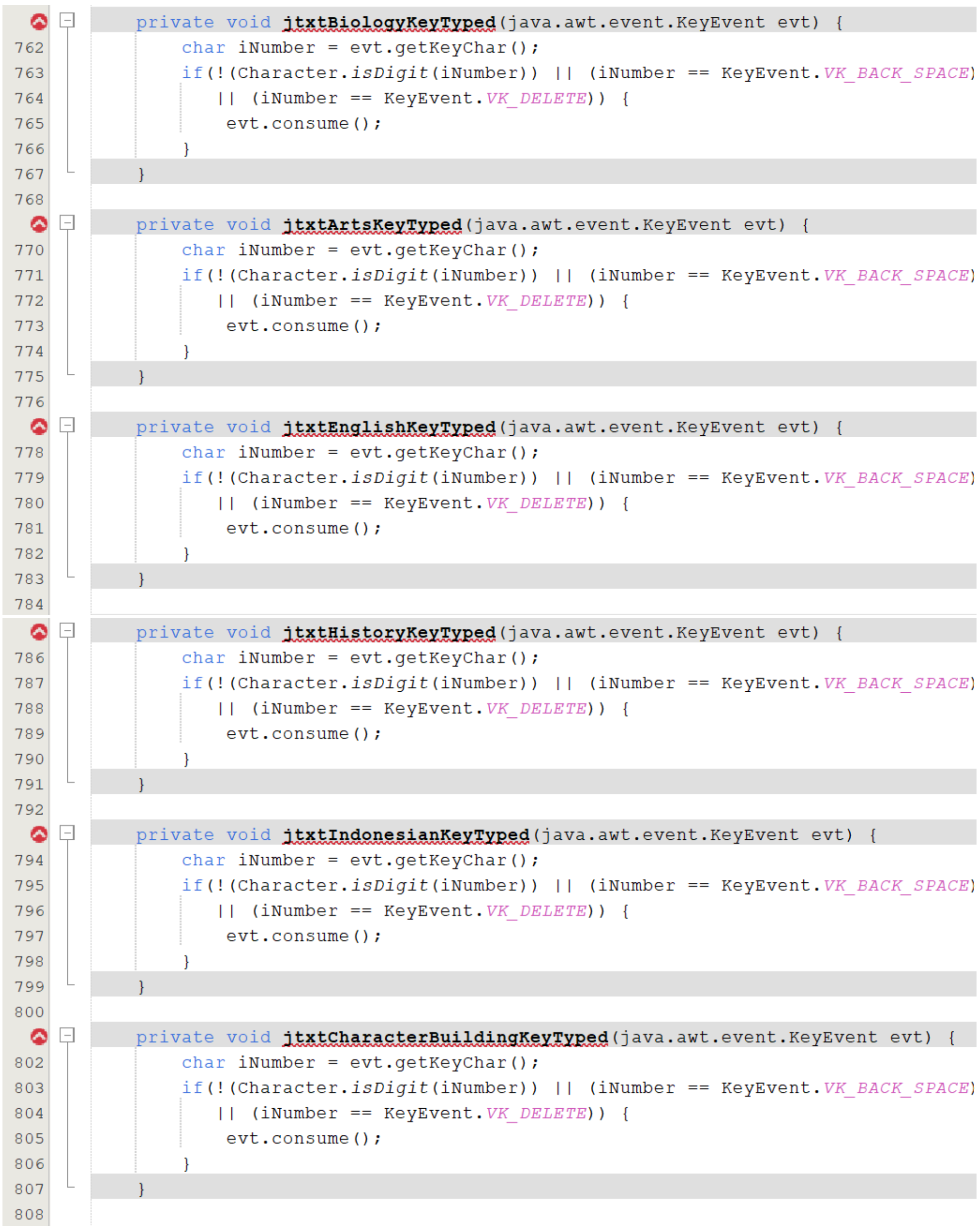


```java
    private void jbtnDeleteActionPerformed(java.awt.event.ActionEvent evt) {
        DefaultTableModel model = (DefaultTableModel) jTable1.getModel();
        if(jTable1.getSelectedRow() == -1) {
            if(jTable1.getRowCount() == 0) {
                String message = "No data to delete\nSelect row to delete";
                JOptionPane.showMessageDialog(null, message,
                "Student Result System", JOptionPane.OK_OPTION);
            }
        } else {
                model.removeRow(jTable1.getSelectedRow());
        }
    }

```

jbtnDeleteActionPerformed function indicates the functionality of the Delete button in the application. First, it will call the table and onto the condition. If the user clicked delete but did not select a row, a message will appear using JOptionPane, titled Student Result System, with the OK method. Else if the user selects a row in the table and click delete, the selected row will be removed from the table.

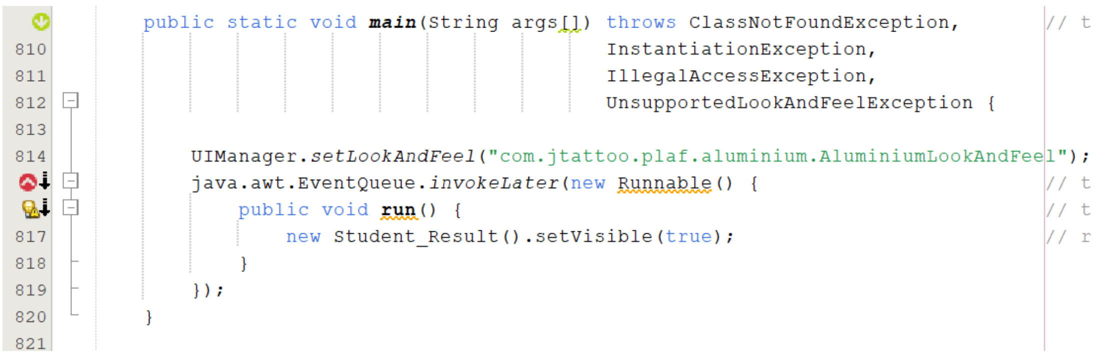


```java
    private void jtxtEconomicsKeyTyped(java.awt.event.KeyEvent evt) {
        char iNumber = evt.getKeyChar();
        if(!(Character.isDigit(iNumber)) || (iNumber == KeyEvent.VK_BACK_SPACE)
           || (iNumber == KeyEvent.VK_DELETE)) {
            evt.consume();
        }
    }

    private void jtxtEnterpreneurshipKeyTyped(java.awt.event.KeyEvent evt) {
        char iNumber = evt.getKeyChar();
        if(!(Character.isDigit(iNumber)) || (iNumber == KeyEvent.VK_BACK_SPACE)
           || (iNumber == KeyEvent.VK_DELETE)) {
            evt.consume();
        }
    }

    private void jtxtPhysicalEducationKeyTyped(java.awt.event.KeyEvent evt) {
        char iNumber = evt.getKeyChar();
        if(!(Character.isDigit(iNumber)) || (iNumber == KeyEvent.VK_BACK_SPACE)
           || (iNumber == KeyEvent.VK_DELETE)) {
            evt.consume();
        }
    }

    private void jtxtMathematicsKeyTyped(java.awt.event.KeyEvent evt) {
        char iNumber = evt.getKeyChar();
        if(!(Character.isDigit(iNumber)) || (iNumber == KeyEvent.VK_BACK_SPACE)
           || (iNumber == KeyEvent.VK_DELETE)) {
            evt.consume();
        }
    }

    private void jtxtChemistryKeyTyped(java.awt.event.KeyEvent evt) {
        char iNumber = evt.getKeyChar();
        if(!(Character.isDigit(iNumber)) || (iNumber == KeyEvent.VK_BACK_SPACE)
           || (iNumber == KeyEvent.VK_DELETE)) {
            evt.consume();
        }
    }

    private void jtxtPhysicsKeyTyped(java.awt.event.KeyEvent evt) {
        char iNumber = evt.getKeyChar();
        if(!(Character.isDigit(iNumber)) || (iNumber == KeyEvent.VK_BACK_SPACE)
           || (iNumber == KeyEvent.VK_DELETE)) {
            evt.consume();
        }
    }

```



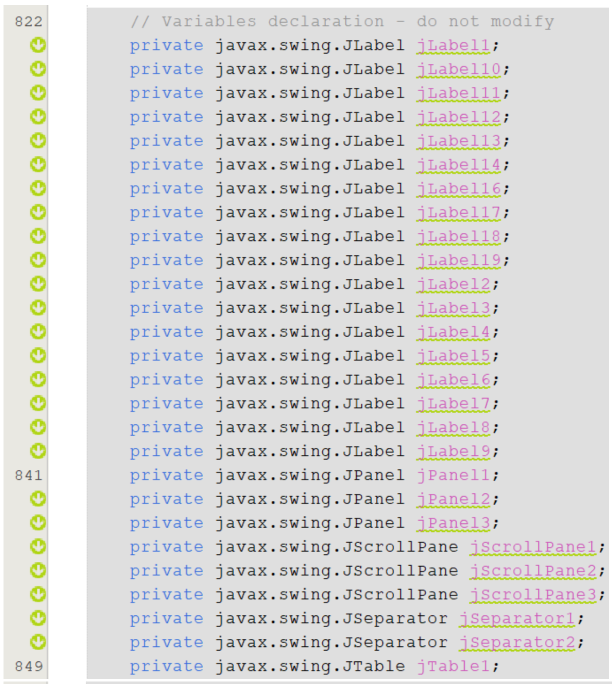
```
    private void jtxtBiologyKeyTyped(java.awt.event.KeyEvent evt) {
        char iNumber = evt.getKeyChar();
        if(!(Character.isDigit(iNumber)) || (iNumber == KeyEvent.VK_BACK_SPACE)
           || (iNumber == KeyEvent.VK_DELETE)) {
            evt.consume();
        }
    }

    private void jtxtArtsKeyTyped(java.awt.event.KeyEvent evt) {
        char iNumber = evt.getKeyChar();
        if(!(Character.isDigit(iNumber)) || (iNumber == KeyEvent.VK_BACK_SPACE)
           || (iNumber == KeyEvent.VK_DELETE)) {
            evt.consume();
        }
    }

    private void jtxtEnglishKeyTyped(java.awt.event.KeyEvent evt) {
        char iNumber = evt.getKeyChar();
        if(!(Character.isDigit(iNumber)) || (iNumber == KeyEvent.VK_BACK_SPACE)
           || (iNumber == KeyEvent.VK_DELETE)) {
            evt.consume();
        }
    }

    private void jtxtHistoryKeyTyped(java.awt.event.KeyEvent evt) {
        char iNumber = evt.getKeyChar();
        if(!(Character.isDigit(iNumber)) || (iNumber == KeyEvent.VK_BACK_SPACE)
           || (iNumber == KeyEvent.VK_DELETE)) {
            evt.consume();
        }
    }

    private void jtxtIndonesianKeyTyped(java.awt.event.KeyEvent evt) {
        char iNumber = evt.getKeyChar();
        if(!(Character.isDigit(iNumber)) || (iNumber == KeyEvent.VK_BACK_SPACE)
           || (iNumber == KeyEvent.VK_DELETE)) {
            evt.consume();
        }
    }

    private void jtxtCharacterBuildingKeyTyped(java.awt.event.KeyEvent evt) {
        char iNumber = evt.getKeyChar();
        if(!(Character.isDigit(iNumber)) || (iNumber == KeyEvent.VK_BACK_SPACE)
           || (iNumber == KeyEvent.VK_DELETE)) {
            evt.consume();
        }
    }
```

All these jtxt……KeyTyped have the same code which to eliminate a character if back space or delete is pressed, by using the consume method.

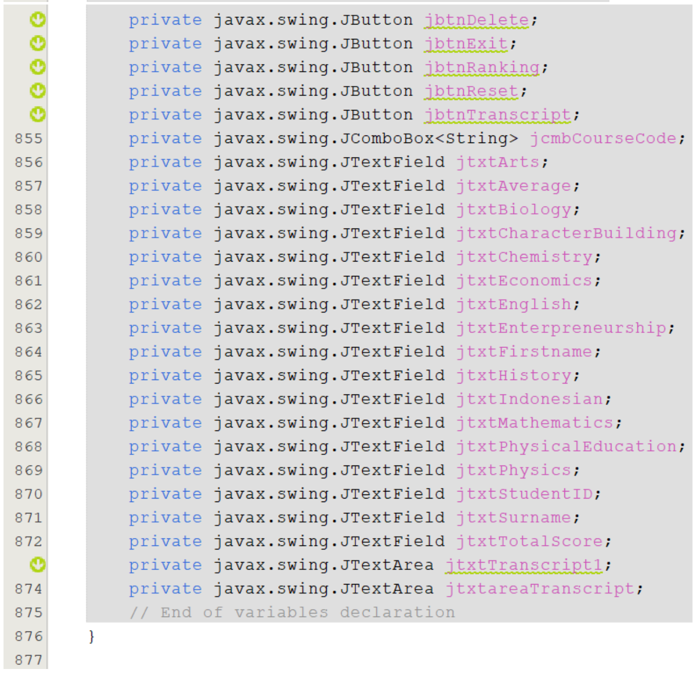


```
    public static void main(String args[]) throws ClassNotFoundException,        // t
                                                  InstantiationException,
                                                  IllegalAccessException,
                                                  UnsupportedLookAndFeelException {

        UIManager.setLookAndFeel("com.jtattoo.plaf.aluminium.AluminiumLookAndFeel");
        java.awt.EventQueue.invokeLater(new Runnable() {                          // t
            public void run() {                                                   // t
                new Student_Result().setVisible(true);                            // r
            }
        });
    }
```

The main function will set the look of the application using setLookAndFeel constructor from the UIManager library. Also, I added jtattoo which is compatible with applications that use swing. Finally, it will run and call the Student_Result and set it to be visible (true).



```
    // Variables declaration - do not modify
    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 jLabel16;
    private javax.swing.JLabel jLabel17;
    private javax.swing.JLabel jLabel18;
    private javax.swing.JLabel jLabel19;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JLabel jLabel9;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JPanel jPanel3;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JScrollPane jScrollPane2;
    private javax.swing.JScrollPane jScrollPane3;
    private javax.swing.JSeparator jSeparator1;
    private javax.swing.JSeparator jSeparator2;
    private javax.swing.JTable jTable1;
```



```
    private javax.swing.JButton jbtnDelete;
    private javax.swing.JButton jbtnExit;
    private javax.swing.JButton jbtnRanking;
    private javax.swing.JButton jbtnReset;
    private javax.swing.JButton jbtnTranscript;
    private javax.swing.JComboBox<String> jcmbCourseCode;
    private javax.swing.JTextField jtxtArts;
    private javax.swing.JTextField jtxtAverage;
    private javax.swing.JTextField jtxtBiology;
    private javax.swing.JTextField jtxtCharacterBuilding;
    private javax.swing.JTextField jtxtChemistry;
    private javax.swing.JTextField jtxtEconomics;
    private javax.swing.JTextField jtxtEnglish;
    private javax.swing.JTextField jtxtEnterpreneurship;
    private javax.swing.JTextField jtxtFirstname;
    private javax.swing.JTextField jtxtHistory;
    private javax.swing.JTextField jtxtIndonesian;
    private javax.swing.JTextField jtxtMathematics;
    private javax.swing.JTextField jtxtPhysicalEducation;
    private javax.swing.JTextField jtxtPhysics;
    private javax.swing.JTextField jtxtStudentID;
    private javax.swing.JTextField jtxtSurname;
    private javax.swing.JTextField jtxtTotalScore;
    private javax.swing.JTextArea jtxtTranscript1;
    private javax.swing.JTextArea jtxtareaTranscript;
    // End of variables declaration
}
```

These are the variable declarations which are already in the designed layout interface.

VII. Project Link

https://github.com/Own20/GitHub/tree/main/Semester%202/Object%20Oriented%20Programming%20COMP6699001/Student_Result_Recording_System

References

https://www.youtube.com/watch?v=e1Ktv9AlwjU&t=967s

https://drive.google.com/file/d/1ieoMRqWX6-QlOj3GzTK1vSEP4gvJ8NSe/view

https://stackoverflow.com/