CS4001NI Programming

30% Individual Coursework

2023-24 Autumn

Student Name: Shashwat Shakya

London Met ID: 23048469

College ID: np01cp4a230242

Assignment Due Date: Friday, May 10, 2024

Assignment Submission Date: Thursday, May 9, 2024

Table of Contents

Table of Figures:

1. Introduction

The main objective of this coursework is to develop a Graphical User Interface (GUI) in a separate class in the previous part of the coursework and add the details of Tutors and Lecturers in an ArrayList and perform various tests like setting salary of tutor, grading assignments and removing tutors. This program is developed by using concepts of Object-Oriented Programming (OOP) like Encapsulation for bundling data and methods so that they are not accessible from other classes, Abstraction to hide implementation details of program and show only the necessary details, Inheritance to inherit the attributes and methods to reduce code and reuse code.

Overall, this coursework helps us to understand Java GUI, Java OOP concepts and Java in general a lot.

1.1 Tools and Applications used to develop this coursework

BlueJ

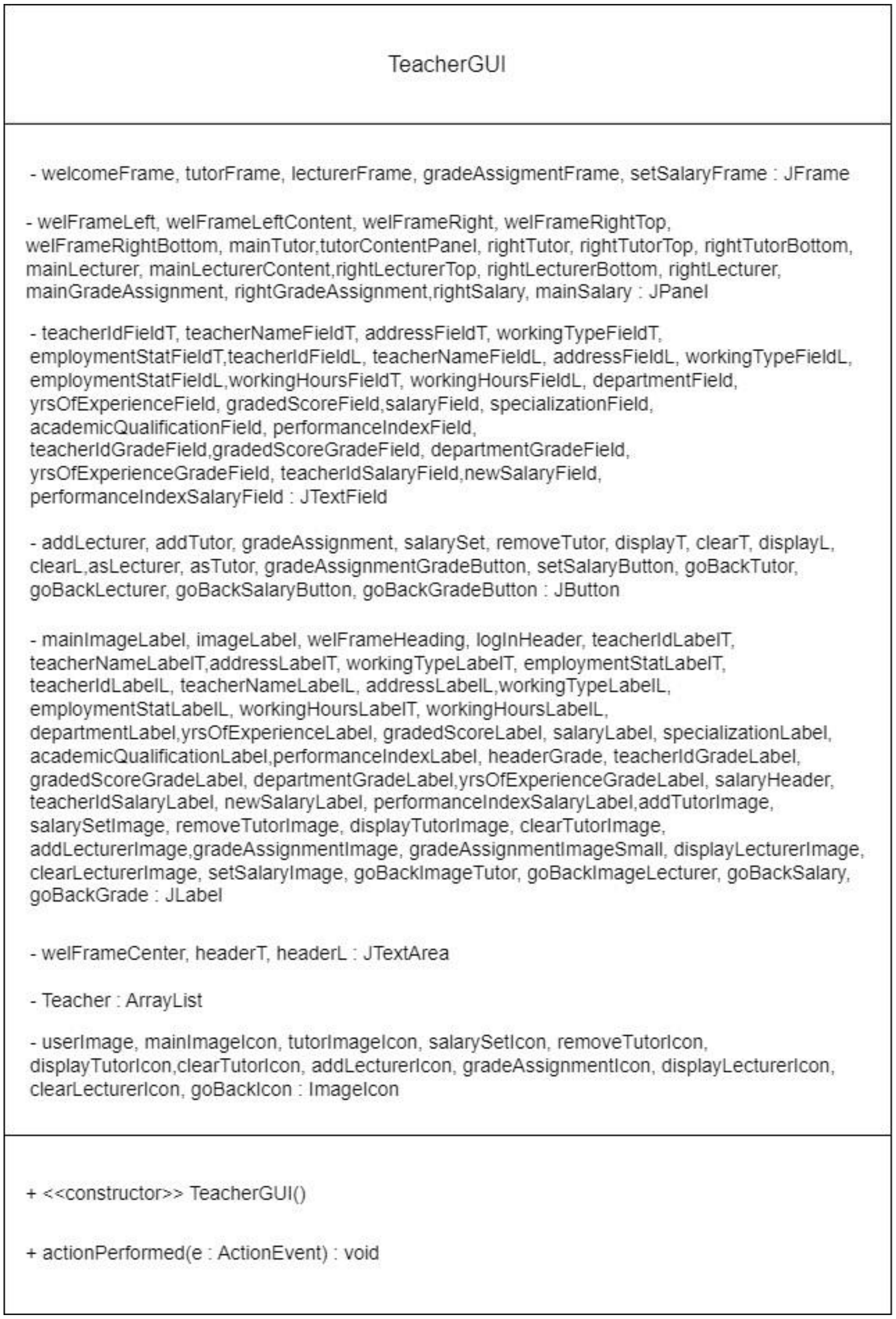
BlueJ is a development environment that makes it simple and quick to create Java programs. Compared to professional settings, BlueJ's UI is purposefully smaller and more straightforward. You can interact with items with BlueJ. They have several uses, including value inspection, method calls, parameter passing, and more. Java expressions can also be called directly without compilation. Windows, Mac OS X, Linux, and other platforms that support Java can all run BlueJ (BlueJ, 1999).

MS Word

MS Word is a word processing software which allows users to create, edit, format, and save documents such as letters, resumes, reports, and many more. It offers many features including spell check, grammar check, tables, templates, and others to help reporting be easier and accurate.

2. Class Diagram

2.1 Class Diagram of TeacherGUI Class



TeacherGUI

- welcomeFrame, tutorFrame, lecturerFrame, gradeAssigmentFrame, setSalaryFrame : JFrame

- welFrameLeft, welFrameLeftContent, welFrameRight, welFrameRightTop, welFrameRightBottom, mainTutor,tutorContentPanel, rightTutor, rightTutorTop, rightTutorBottom, mainLecturer, mainLecturerContent,rightLecturerTop, rightLecturerBottom, rightLecturer, mainGradeAssignment, rightGradeAssignment,rightSalary, mainSalary : JPanel

- teacherIdFieldT, teacherNameFieldT, addressFieldT, workingTypeFieldT, employmentStatFieldT,teacherIdFieldL, teacherNameFieldL, addressFieldL, workingTypeFieldL, employmentStatFieldL,workingHoursFieldT, workingHoursFieldL, departmentField, yrsOfExperienceField, gradedScoreField,salaryField, specializationField, academicQualificationField, performanceIndexField, teacherIdGradeField,gradedScoreGradeField, departmentGradeField, yrsOfExperienceGradeField, teacherIdSalaryField,newSalaryField, performanceIndexSalaryField : JTextField

- addLecturer, addTutor, gradeAssignment, salarySet, removeTutor, displayT, clearT, displayL, clearL,asLecturer, asTutor, gradeAssignmentGradeButton, setSalaryButton, goBackTutor, goBackLecturer, goBackSalaryButton, goBackGradeButton : JButton

- mainImageLabel, imageLabel, welFrameHeading, logInHeader, teacherIdLabelT, teacherNameLabelT,addressLabelT, workingTypeLabelT, employmentStatLabelT, teacherIdLabelL, teacherNameLabelL, addressLabelL,workingTypeLabelL, employmentStatLabelL, workingHoursLabelT, workingHoursLabelL, departmentLabel,yrsOfExperienceLabel, gradedScoreLabel, salaryLabel, specializationLabel, academicQualificationLabel,performanceIndexLabel, headerGrade, teacherIdGradeLabel, gradedScoreGradeLabel, departmentGradeLabel,yrsOfExperienceGradeLabel, salaryHeader, teacherIdSalaryLabel, newSalaryLabel, performanceIndexSalaryLabel,addTutorImage, salarySetImage, removeTutorImage, displayTutorImage, clearTutorImage, addLecturerImage,gradeAssignmentImage, gradeAssignmentImageSmall, displayLecturerImage, clearLecturerImage, setSalaryImage, goBackImageTutor, goBackImageLecturer, goBackSalary, goBackGrade : JLabel

- welFrameCenter, headerT, headerL : JTextArea

- Teacher : ArrayList

- userImage, mainImageIcon, tutorImageIcon, salarySetIcon, removeTutorIcon, displayTutorIcon,clearTutorIcon, addLecturerIcon, gradeAssignmentIcon, displayLecturerIcon, clearLecturerIcon, goBackIcon : ImageIcon

+ <<constructor>> TeacherGUI()

+ actionPerformed(e : ActionEvent) : void

Figure 1 - Screenshot of Class Diagram of TeacherGUI

2.2 Full Class Diagram

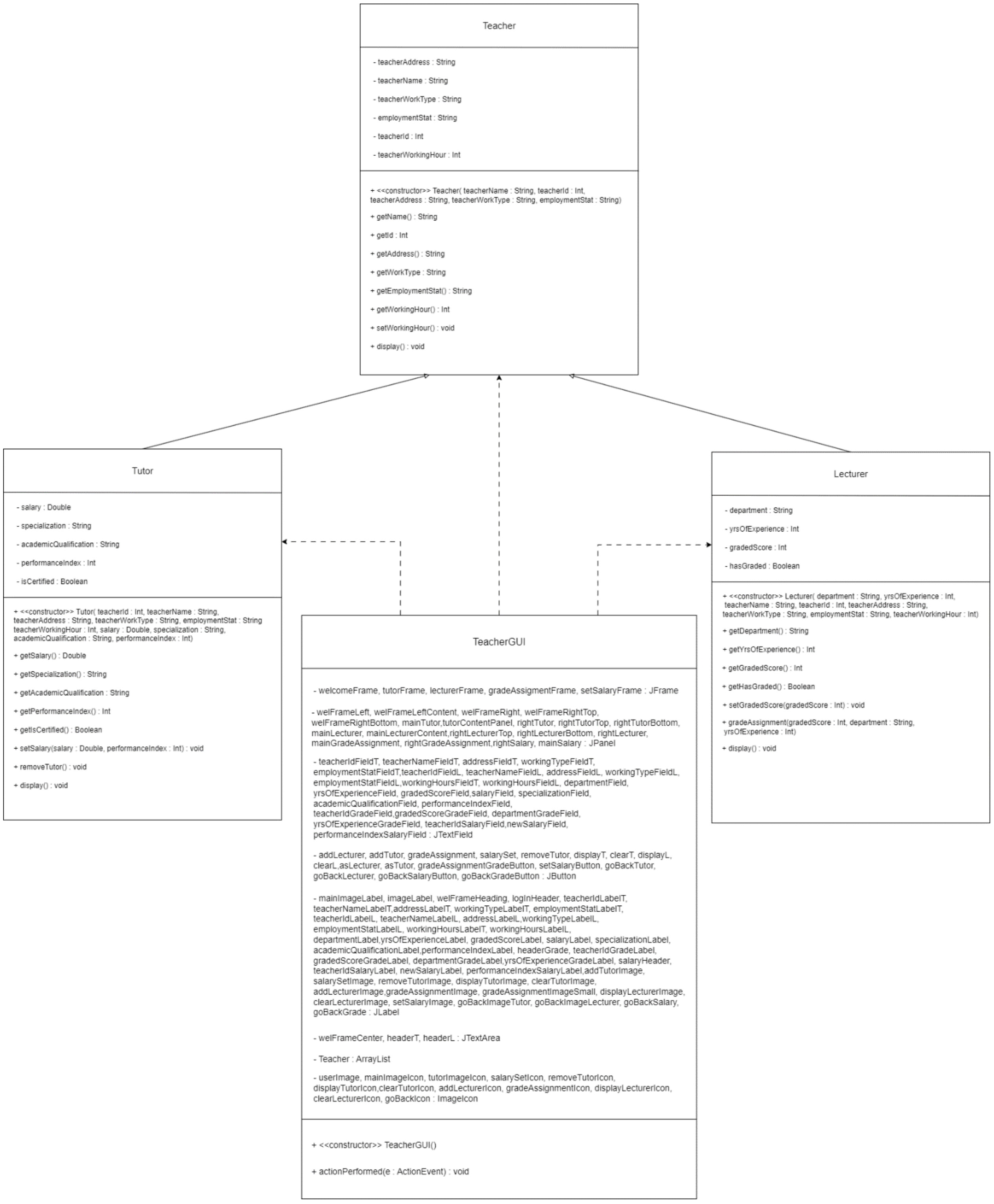

Teacher
- teacherAddress : String
- teacherName : String
- teacherWorkType : String
- employmentStat : String
- teacherId : Int
- teacherWorkingHour : Int
+ <<constructor>> Teacher(teacherName : String, teacherId : Int, teacherAddress : String, teacherWorkType : String, employmentStat : String)
+ getName() : String
+ getId : Int
+ getAddress() : String
+ getWorkType : String
+ getEmploymentStat() : String
+ getWorkingHour() : Int
+ setWorkingHour() : void
+ display() : void
Tutor
- salary : Double
- specialization : String
- academicQualification : String
- performanceIndex : Int
- isCertified : Boolean
+ <<constructor>> Tutor(teacherId : Int, teacherName : String, teacherAddress : String, teacherWorkType : String, employmentStat : String teacherWorkingHour : Int, salary : Double, specialization : String, academicQualification : String, performanceIndex : Int)
+ getSalary() : Double
+ getSpecialization() : String
+ getAcademicQualification : String
+ getPerformanceIndex() : Int
+ getIsCertified() : Boolean
+ setSalary(salary : Double, performanceIndex : Int) : void
+ removeTutor() : void
+ display() : void
Lecturer
- department : String
- yrsOfExperience : Int
- gradedScore : Int
- hasGraded : Boolean
+ <<constructor>> Lecturer(department : String, yrsOfExperience : Int, teacherName : String, teacherId : Int, teacherAddress : String, teacherWorkType : String, employmentStat : String, teacherWorkingHour : Int)
+ getDepartment() : String
+ getYrsOfExperience() : Int
+ getGradedScore() : Int
+ getHasGraded() : Boolean
+ setGradedScore(gradedScore : Int) : void
+ gradeAssignment(gradedScore : Int, department : String, yrsOfExperience : Int)
+ display() : void
TeacherGUI
- welcomeFrame, tutorFrame, lecturerFrame, gradeAssigmentFrame, setSalaryFrame : JFrame
- welFrameLeft, welFrameLeftContent, welFrameRight, welFrameRightTop, welFrameRightBottom, mainTutor,tutorContentPanel, rightTutor, rightTutorTop, rightTutorBottom, mainLecturer, mainLecturerContent,rightLecturerTop, rightLecturerBottom, rightLecturer, mainGradeAssignment, rightGradeAssignment,rightSalary, mainSalary : JPanel
- teacherIdFieldT, teacherNameFieldT, addressFieldT, workingTypeFieldT, employmentStatFieldT,teacherIdFieldL, teacherNameFieldL, addressFieldL, workingTypeFieldL, employmentStatFieldL,workingHoursFieldT, workingHoursFieldL, departmentField, yrsOfExperienceField, gradedScoreField,salaryField, specializationField, academicQualificationField, performanceIndexField, teacherIdGradeField,gradedScoreGradeField, departmentGradeField, yrsOfExperienceGradeField, teacherIdSalaryField,newSalaryField, performanceIndexSalaryField : JTextField
- addLecturer, addTutor, gradeAssignment, salarySet, removeTutor, displayT, clearT, displayL, clearL,asLecturer, asTutor, gradeAssignmentGradeButton, setSalaryButton, goBackTutor, goBackLecturer, goBackSalaryButton, goBackGradeButton : JButton
- mainImageLabel, imageLabel, welFrameHeading, logInHeader, teacherIdLabelT, teacherNameLabelT,addressLabelT, workingTypeLabelT, employmentStatLabelT, teacherIdLabelL, teacherNameLabelL, addressLabelL,workingTypeLabelL, employmentStatLabelL, workingHoursLabelT, workingHoursLabelL, departmentLabel,yrsOfExperienceLabel, gradedScoreLabel, salaryLabel, specializationLabel, academicQualificationLabel,performanceIndexLabel, headerGrade, teacherIdGradeLabel, gradedScoreGradeLabel, departmentGradeLabel,yrsOfExperienceGradeLabel, salaryHeader, teacherIdSalaryLabel, newSalaryLabel, performanceIndexSalaryLabel,addTutorImage, salarySetImage, removeTutorImage, displayTutorImage, clearTutorImage, addLecturerImage,gradeAssignmentImage, gradeAssignmentImageSmall, displayLecturerImage, clearLecturerImage, setSalaryImage, goBackImageTutor, goBackImageLecturer, goBackSalary, goBackGrade : JLabel
- welFrameCenter, headerT, headerL : JTextArea
- Teacher : ArrayList
- userImage, mainImageIcon, tutorImageIcon, salarySetIcon, removeTutorIcon, displayTutorIcon,clearTutorIcon, addLecturerIcon, gradeAssignmentIcon, displayLecturerIcon, clearLecturerIcon, goBackIcon : ImageIcon
+ <<constructor>> TeacherGUI()
+ actionPerformed(e : ActionEvent) : void

Figure 2 - Screenshot of Full Class Diagram

3. Pseudocode

3.1 Pseudocode of TeacherGUI

```
IMPORT ArrayList from package named java util
IMPORT JButton from package named Swing
IMPORT JFrame from package named Swing
IMPORT JLabel from package named Swing
IMPORT JPanel from package named Swing
IMPORT JTextField from package named Swing
IMPORT JOptionPane from package named Swing
IMPORT JTextArea from package named Swing
IMPORT Dimension from package named Awt
IMPORT ActionEvent from package named Awt
IMPORT ActionListener from package named Awt
IMPORT MouseListener from package named Awt
IMPORT MouseAdapter from package named Awt
IMPORT BorderLayout from package named Awt
IMPORT Color from package named Awt
IMPORT FlowLayout from package named Awt
IMPORT Font from package named Awt

CREATE a class named TeacherGUI which implements an interface called
ActionListener
DO
    DECLARE a JFrame named "welcomeFrame" using a private access modifier
    DECLARE a JFrame named "tutorFrame" using a private access modifier
    DECLARE a JFrame named "lecturerFrame" using a private access modifier
    DECLARE a JFrame named "gradeAssigmentFrame" using a private access
    modifier
    DECLARE a JFrame named "setSalaryFrame" using a private access modifier
```

DECLARE a JPanel named "welFrameLeft" using a private access modifier
DECLARE a JPanel named "welFrameLeftContent" using a private access modifier
DECLARE a JPanel named "welFrameRight" using a private access modifier
DECLARE a JPanel named "welFrameRightTop" using a private access modifier
DECLARE a JPanel named "welFrameRightBottom" using a private access modifier
DECLARE a JPanel named "mainTutor" using a private access modifier
DECLARE a JPanel named "tutorContentPanel" using a private access modifier
DECLARE a JPanel named "rightTutor" using a private access modifier
DECLARE a JPanel named "rightTutorTop" using a private access modifier
DECLARE a JPanel named "rightTutorBottom" using a private access modifier
DECLARE a JPanel named "mainLecturer" using a private access modifier
DECLARE a JPanel named "mainLecturerContent" using a private access modifier
DECLARE a JPanel named "rightLecturerTop" using a private access modifier
DECLARE a JPanel named "rightLecturerBottom" using a private access modifier
DECLARE a JPanel named "rightLecturer" using a private access modifier
DECLARE a JPanel named "mainGradeAssignment" using a private access modifier
DECLARE a JPanel named "rightGradeAssignment" using a private access modifier
DECLARE a JPanel named "rightSalary" using a private access modifier
DECLARE a JPanel named "mainSalary" using a private access modifier

DECLARE a JTextField named "teacherIdFieldT" using a private access modifier
DECLARE a JTextField named "teacherNameFieldT" using a private access modifier
DECLARE a JTextField named "addressFieldT" using a private access modifier

DECLARE a JTextField named "workingTypeFieldT" using a private access modifier

DECLARE a JTextField named "employmentStatFieldT" using a private access modifier

DECLARE a JTextField named "teacherIdFieldL" using a private access modifier

DECLARE a JTextField named "teacherNameFieldL" using a private access modifier

DECLARE a JTextField named "addressFieldL" using a private access modifier

DECLARE a JTextField named "workingTypeFieldL" using a private access modifier

DECLARE a JTextField named "employmentStatFieldL" using a private access modifier

DECLARE a JTextField named "workingHoursFieldT" using a private access modifier

DECLARE a JTextField named "workingHoursFieldL" using a private access modifier

DECLARE a JTextField named "departmentField" using a private access modifier

DECLARE a JTextField named "yrsOfExperienceField" using a private access modifier

DECLARE a JTextField named "gradedScoreField" using a private access modifier

DECLARE a JTextField named "salaryField" using a private access modifier

DECLARE a JTextField named "specializationField" using a private access modifier

DECLARE a JTextField named "academicQualificationField" using a private access modifier

DECLARE a JTextField named "performanceIndexField" using a private access modifier

DECLARE a JTextField named "teacherIdGradeField" using a private access modifier

DECLARE a JTextField named "gradedScoreGradeField" using a private access modifier
DECLARE a JTextField named "departmentGradeField" using a private access modifier
DECLARE a JTextField named "yrsOfExperienceGradeField" using a private access modifier
DECLARE a JTextField named "teacherIdSalaryField" using a private access modifier
DECLARE a JTextField named "newSalaryField" using a private access modifier
DECLARE a JTextField named "performanceIndexSalaryField" using a private access modifier

DECLARE a JButton named "addLecturer" using a private access modifier
DECLARE a JButton named "addTutor" using a private access modifier
DECLARE a JButton named "gradeAssignment" using a private access modifier
DECLARE a JButton named "salarySet" using a private access modifier
DECLARE a JButton named "removeTutor" using a private access modifier
DECLARE a JButton named "displayT" using a private access modifier
DECLARE a JButton named "clearT" using a private access modifier
DECLARE a JButton named "displayL" using a private access modifier
DECLARE a JButton named "clearL" using a private access modifier
DECLARE a JButton named "asLecturer" using a private access modifier
DECLARE a JButton named "asTutor" using a private access modifier
DECLARE a JButton named "gradeAssignmentGradeButton" using a private access modifier
DECLARE a JButton named "setSalaryButton" using a private access modifier
DECLARE a JButton named "goBackTutor" using a private access modifier
DECLARE a JButton named "goBackLecturer" using a private access modifier
DECLARE a JButton named "goBackSalaryButton" using a private access modifier

DECLARE a JButton named "goBackGradeButton" using a private access modifier

DECLARE a JLabel named "mainImageLabel" using a private access modifier
DECLARE a JLabel named "imageLabel" using a private access modifier
DECLARE a JLabel named "welFrameHeading" using a private access modifier
DECLARE a JLabel named "logInHeader" using a private access modifier
DECLARE a JLabel named "teacherIdLabelT" using a private access modifier
DECLARE a JLabel named "teacherNameLabelT" using a private access modifier
DECLARE a JLabel named "addressLabelT" using a private access modifier
DECLARE a JLabel named "workingTypeLabelT" using a private access modifier
DECLARE a JLabel named "employmentStatLabelT" using a private access modifier
DECLARE a JLabel named "teacherIdLabelL" using a private access modifier
DECLARE a JLabel named "teacherNameLabelL" using a private access modifier
DECLARE a JLabel named "addressLabelL" using a private access modifier
DECLARE a JLabel named "workingTypeLabelL" using a private access modifier
DECLARE a JLabel named "employmentStatLabelL" using a private access modifier
DECLARE a JLabel named "workingHoursLabelT" using a private access modifier
DECLARE a JLabel named "workingHoursLabelL" using a private access modifier
DECLARE a JLabel named "departmentLabel" using a private access modifier
DECLARE a JLabel named "yrsOfExperienceLabel" using a private access modifier
DECLARE a JLabel named "gradedScoreLabel" using a private access modifier
DECLARE a JLabel named "salaryLabel" using a private access modifier
DECLARE a JLabel named "specializationLabel" using a private access modifier

DECLARE a JLabel named "academicQualificationLabel" using a private access modifier
DECLARE a JLabel named "performanceIndexLabel" using a private access modifier
DECLARE a JLabel named "headerGrade" using a private access modifier
DECLARE a JLabel named "teacherIdGradeLabel" using a private access modifier
DECLARE a JLabel named "gradedScoreGradeLabel" using a private access modifier
DECLARE a JLabel named "departmentGradeLabel" using a private access modifier
DECLARE a JLabel named "yrsOfExperienceGradeLabel" using a private access modifier
DECLARE a JLabel named "salaryHeader" using a private access modifier
DECLARE a JLabel named "teacherIdSalaryLabel" using a private access modifier
DECLARE a JLabel named "newSalaryLabel" using a private access modifier
DECLARE a JLabel named "performanceIndexSalaryLabel" using a private access modifier
DECLARE a JLabel named "addTutorImage" using a private access modifier
DECLARE a JLabel named "salarySetImage" using a private access modifier
DECLARE a JLabel named "removeTutorImage" using a private access modifier
DECLARE a JLabel named "displayTutorImage" using a private access modifier
DECLARE a JLabel named "clearTutorImage" using a private access modifier
DECLARE a JLabel named "addLecturerImage" using a private access modifier
DECLARE a JLabel named "gradeAssignmentImage" using a private access modifier
DECLARE a JLabel named "gradeAssignmentImageSmall" using a private access modifier
DECLARE a JLabel named "displayLecturerImage" using a private access modifier

DECLARE a JLabel named "clearLecturerImage" using a private access modifier
DECLARE a JLabel named "setSalaryImage" using a private access modifier
DECLARE a JLabel named "goBackImageTutor" using a private access modifier
DECLARE a JLabel named "goBackImageLecturer" using a private access modifier
DECLARE a JLabel named "goBackSalary" using a private access modifier
DECLARE a JLabel named "goBackGrade" using a private access modifier

DECLARE a JTextArea named "welFrameCenter" using a private access modifier
DECLARE a JTextArea named "headerT" using a private access modifier
DECLARE a JTextArea named "headerL" using a private access modifier

DECLARE an ImageIcon named "userImage" using a private access modifier
DECLARE an ImageIcon named "mainImageIcon" using a private access modifier
DECLARE an ImageIcon named "tutorImageIcon" using a private access modifier
DECLARE an ImageIcon named "salarySetIcon" using a private access modifier
DECLARE an ImageIcon named "removeTutorIcon" using a private access modifier
DECLARE an ImageIcon named "displayTutorIcon" using a private access modifier
DECLARE an ImageIcon named "clearTutorIcon" using a private access modifier
DECLARE an ImageIcon named "addLecturerIcon" using a private access modifier
DECLARE an ImageIcon named "gradeAssignmentIcon" using a private access modifier
DECLARE an ImageIcon named "displayLecturerIcon" using a private access modifier

```
DECLARE an ImageIcon named "clearLecturerIcon" using a private access modifier
DECLARE an ImageIcon named "goBackIcon" using a private access modifier

DECLARE an ArrayList named Teacher which stores objects of class Teacher

CREATE a constructor named TeacherGUI with no parameters
DO
    CREATE a new JFrame named welcomeFrame titled "Welcome"
    CREATE a new JFrame named tutorFrame titled "Tutors"
    CREATE a new JFrame named lecturerFrame titled "Lecturers"
    CREATE a new JFrame named gradeAssigmentFrame titled "Grade Assigments"
    CREATE a new JFrame named setSalaryFrame titled "Set Salary"

    DECLARE secondaryColor as COLOR initialized with RGB values 44 as Red, 62 as Blue, and 80 as Green
    DECLARE primaryColor as COLOR initialized with RGB values 245 as Red, 245 as Blue, and 245 as Green
    DECLARE emptyBorder as BORDER initialized with 0, 0, 0, 0 values
    DECLARE buttonCursor as CURSOR initialized with Hand cursor value
    DECLARE mainFont as FONT initialized with the font family Cambria, as PLAIN font with size 19
    DECLARE headerFont as FONT initialized with the font family Cambria, as BOLD font with size 22
    DECLARE homePageHeader as FONT initialized with the font style Cambria, as BOLD font with size 50

    SET mainImageIcon TO a new IMAGEICON loading the resource "Icons/coverMain.png" using getClass() method
    SET mainImageLabel TO a new JLabel with mainImageIcon as parameter
```

```
DECLARE welFrameCenter AS JTextArea initialized with an empty string
SET the bounds of welFrameCenter to 150 in the x-axis, 60 in the y-axis,
600 as width, 200 as height
SET welFrameCenter as not editable
SET welFrameCenter to wrap lines
SET welFrameCenter to wrap words at whitespace
SET the background color of welFrameCenter TO primaryColor
SET the foreground color of welFrameCenter TO secondaryColor
SET the font of welFrameCenter TO testFont

SET welFrameHeading TO a new JLabel with text "Welcome to the Home
Page"
SET bounds of welFrameHeading TO 50 to the x-axis, 400 to the y-axis,
700 as width and 200 as height
SET font of welFrameHeading TO homePageHeader
SET foreground color of welFrameHeading TO secondaryColor

SET logInHeader TO a new JLabel with text "Log In As:"
SET bounds of logInHeader TO 100 to the x-axis, 0 to the y-axis, 400 as
width, 60 as height
SET font of logInHeader TO homePageHeader
SET foreground color of logInHeader TO primaryColor

SET asLecturer TO a new JButton with text "Lecturer"
SET bounds of asLecturer TO (125, 150, 150, 40)
SET preferred size of asLecturer TO (150, 40)
SET background color of asLecturer TO secondaryColor
SET foreground color of asLecturer TO primaryColor
SET focusable property of asLecturer TO false
SET font of asLecturer TO mainFont
```

```
ADD a MouseListener to asLecturer WHEN mouse enters the component
mouseEntered method is triggered
DO
    SET background color of asLecturer TO primaryColor
    SET foreground color of asLecturer TO secondaryColor
END DO
WHEN mouse exits the component mouseExited method is triggered
DO
    SET background color of asLecturer TO secondaryColor
    SET foreground color of asLecturer TO primaryColor
END DO
ADD an ActionListener to asLecturer WHEN event occurs trigger
actionPerformed method
DO
    SET visibility of lecturerFrame TO true
    DISPOSE welcomeFrame
END DO

SET asTutor TO a new JButton WITH text "Tutor"
SET bounds of asTutor TO (125, 220, 150, 40)
SET preferred size of asTutor TO (150, 40)
SET background color of asTutor TO secondaryColor
SET foreground color of asTutor TO primaryColor
SET focusable property of asTutor TO false
SET font of asTutor TO mainFont
ADD a MouseListener to asTutor WHEN mouse enters the component
mouseEntered method is triggered
DO
    SET background color of asTutor TO primaryColor
    SET foreground color of asTutor TO secondaryColor
END DO
```

```
WHEN mouse exits the component mouseExited method is triggered
DO
        SET background color of asTutor TO secondaryColor
        SET foreground color of asTutor TO primaryColor
END DO
ADD an ActionListener to asTutor WHEN event occurs trigger
actionPerformed method
DO
        SET visibility of tutorFrame TO true
        DISPOSE welcomeFrame
END DO

SET userImage TO a new IMAGEICON loading the resource
"Icons/userIcon.png" using getClass() method
SET imageLabel TO a new JLabel with userImage AS parameter
SET bounds of imageLabel TO 0 to the x-axis, 0 to the y-axis, 400 as
width, 200 as height

SET welFrameRightTop TO a new JPanel WITH FlowLayout CENTERED,
HORIZONTAL GAP as 100, and VERTICAL GAP as 190
SET background color of welFrameRightTop TO secondaryColor
ADD imageLabel TO welFrameRightTop

SET welFrameRightBottom TO a new JPanel WITH layout SET to null
SET background color of welFrameRightBottom TO secondaryColor
ADD logInHeader TO welFrameRightBottom
ADD asLecturer TO welFrameRightBottom
ADD asTutor TO welFrameRightBottom

CREATE welFrameLeftContent AS a new JPanel
SET layout of welFrameLeftContent TO null
SET preferred size of welFrameLeftContent TO 800 as width and 900 as height
SET background color of welFrameLeftContent TO primaryColor
ADD mainImageLabel TO welFrameLeftContent
ADD welFrameHeading TO welFrameLeftContent
ADD welFrameCenter TO welFrameLeftContent

CREATE welFrameLeft AS a new JPanel
SET layout of welFrameLeft TO FlowLayout CENTERED, HORIZONTAL GAP as 0, and VERTICAL GAP as 0
SET background color of welFrameLeft TO primaryColor
ADD welFrameLeftContent TO welFrameLeft

CREATE welFrameRight AS a new JPanel
SET layout of welFrameRight TO GridLayout with 2 rows and 0 columns
SET background color of welFrameRight TO secondaryColor
ADD welFrameRightTop TO welFrameRight
ADD welFrameRightBottom TO welFrameRight
SET bounds of welFrameRight TO 800 to the x-axis, 0 to the y-axis, 400 as width, 900 as height

SET Extended State of welcomeFrame TO MAXIMIZED
SET size of welcomeFrame TO 1200 as width, 900 as height
SET default close operation of welcomeFrame TO EXIT ON CLOSE
SET visibility of welcomeFrame TO TRUE
SET layout of welcomeFrame TO BorderLayout
SET resizable property of welcomeFrame TO TRUE
ADD welFrameLeft TO welcomeFrame AT BorderLayout.CENTER
ADD welFrameRight TO welcomeFrame AT BorderLayout.EAST
```

```
SET headerT TO a new JTextArea with text "Add, Remove and SET Salary for Tutors"
SET editable property of headerT TO false
SET lineWrap property of headerT TO true
SET wrapStyleWord property of headerT TO true
SET background color of headerT TO secondaryColor
SET foreground color of headerT TO primaryColor
SET font of headerT TO homePageHeader
SET preferred size of headerT TO 400 as width, 200 as height

SET teacherIdLabelT TO a new JLabel with text "Teacher ID:"
SET bounds of teacherIdLabelT TO 200 to the x-axis, 30 to the y-axis, 100 as width, 40 as height
SET foreground color of teacherIdLabelT TO secondaryColor
SET font of teacherIdLabelT TO mainFont

SET teacherNameLabelT TO a new JLabel with text "Teacher Name:"
SET bounds of teacherNameLabelT TO 200 to the x-axis, 110 to the y-axis, 190 as height, 40 as width
SET foreground color of teacherNameLabelT TO secondaryColor
SET font of teacherNameLabelT TO mainFont

SET addressLabelT TO a new JLabel with text "Address:"
SET bounds of addressLabelT TO 200 to the x-axis, 190 to the y-axis, 190 as height, 40 as width
SET foreground color of addressLabelT TO secondaryColor
SET font of addressLabelT TO mainFont

SET workingTypeLabelT TO a new JLabel with text "Working Type:"
SET bounds of workingTypeLabelT TO 200 to the x-axis, 270 to the y-axis, 190 as height, 40 as width
SET foreground color of workingTypeLabelT TO secondaryColor
SET font of workingTypeLabelT TO mainFont

SET employmentStatLabelT TO a new JLabel with text "Employment
Status:"
SET bounds of employmentStatLabelT TO 200 to the x-axis, 350 to the y-
axis, 190 as height, 40 as width
SET foreground color of employmentStatLabelT TO secondaryColor
SET font of employmentStatLabelT TO mainFont

SET workingHoursLabelT TO a new JLabel with text "Working Hours:"
SET bounds of workingHoursLabelT TO 200 to the x-axis, 430 to the y-
axis, 190 as height, and 40 as width
SET foreground color of workingHoursLabelT TO secondaryColor
SET font of workingHoursLabelT TO mainFont

SET salaryLabel TO a new JLabel with text "Salary:"
SET bounds of salaryLabel TO 200 to the x-axis, 510 to the y-axis, 190 as
height, and 40 as width
SET foreground color of salaryLabel TO secondaryColor
SET font of salaryLabel TO mainFont

SET specializationLabel TO a new JLabel with text "Specialization:"
SET bounds of specializationLabel TO 200 in the x-axis, 590 in the y-axis,
190 as width, and 40 as height
SET foreground color of specializationLabel TO secondaryColor
SET font of specializationLabel TO mainFont

SET academicQualificationLabel TO a new JLabel with text "Academic
Qualification:"
```

```
SET bounds of academicQualificationLabel TO 200 in the x-axis, 670 in the y-axis, 190 as width, and 40 as height
SET foreground color of academicQualificationLabel TO secondaryColor
SET font of academicQualificationLabel TO mainFont

SET performanceIndexLabel TO a new JLabel with text "Performance Index:"
SET bounds of performanceIndexLabel TO 200 in the x-axis, 750 in the y-axis, 190 as width, and 40 as height
SET foreground color of performanceIndexLabel TO secondaryColor
SET font of performanceIndexLabel TO mainFont

SET teacherIdFieldT TO a new JTextField
SET bounds of teacherIdFieldT TO 200 in the x-axis, 65 in the y-axis, 400 as width, and 35 as height
SET background color of teacherIdFieldT TO primaryColor
SET foreground color of teacherIdFieldT TO secondaryColor
SET horizontal alignment of teacherIdFieldT TO CENTER
SET font of teacherIdFieldT TO mainFont

SET teacherNameFieldT TO a new JTextField
SET bounds of teacherNameFieldT TO 200 in the x-axis, 145 in the y-axis, 400 as width, and 35 as height
SET background color of teacherNameFieldT TO primaryColor
SET foreground color of teacherNameFieldT TO secondaryColor
SET horizontal alignment of teacherNameFieldT TO CENTER
SET font of teacherNameFieldT TO mainFont

SET addressFieldT TO a new JTextField
SET bounds of addressFieldT TO 200 in the x-axis, 225 in the y-axis, 400 as width, and 35 as height
SET background color of addressFieldT TO primaryColor
SET foreground color of addressFieldT TO secondaryColor
SET horizontal alignment of addressFieldT TO CENTER
SET font of addressFieldT TO mainFont

SET workingTypeFieldT TO a new JTextField
SET bounds of workingTypeFieldT TO 200 in the x-axis, 305 in the y-axis,
400 as width, and 35 as height
SET background color of workingTypeFieldT TO primaryColor
SET foreground color of workingTypeFieldT TO secondaryColor
SET horizontal alignment of workingTypeFieldT TO CENTER
SET font of workingTypeFieldT TO mainFont

SET employmentStatFieldT TO a new JTextField
SET bounds of employmentStatFieldT TO 200 in the x-axis, 385 in the y-
axis, 400 as width, and 35 as height
SET background color of employmentStatFieldT TO primaryColor
SET foreground color of employmentStatFieldT TO secondaryColor
SET horizontal alignment of employmentStatFieldT TO CENTER
SET font of employmentStatFieldT TO mainFont

SET workingHoursFieldT TO a new JTextField
SET bounds of workingHoursFieldT TO 200 in the x-axis, 465 in the y-
axis, 400 as width, and 35 as height
SET background color of workingHoursFieldT TO primaryColor
SET foreground color of workingHoursFieldT TO secondaryColor
SET horizontal alignment of workingHoursFieldT TO CENTER
SET font of workingHoursFieldT TO mainFont

SET salaryField TO a new JTextField
```

```
**SET** bounds of salaryField **TO** 200 in the x-axis, 545 in the y-axis, 400 as width, and 35 as height
**SET** background color of salaryField **TO** primaryColor
**SET** foreground color of salaryField **TO** secondaryColor
**SET** horizontal alignment of salaryField **TO** CENTER
**SET** font of salaryField **TO** mainFont

**SET** specializationField **TO** a new JTextField
**SET** bounds of specializationField **TO** 200 in the x-axis, 625 in the y-axis, 400 as width, and 35 as height
**SET** background color of specializationField **TO** primaryColor
**SET** foreground color of specializationField **TO** secondaryColor
**SET** horizontal alignment of specializationField **TO** CENTER
**SET** font of specializationField **TO** mainFont

**SET** academicQualificationField **TO** a new JTextField
**SET** bounds of academicQualificationField **TO** 200 in the x-axis, 705 in the y-axis, 400 as width, and 35 as height
**SET** background color of academicQualificationField **TO** primaryColor
**SET** foreground color of academicQualificationField **TO** secondaryColor
**SET** horizontal alignment of academicQualificationField **TO** CENTER
**SET** font of academicQualificationField **TO** mainFont

**SET** performanceIndexField **TO** a new JTextField
**SET** bounds of performanceIndexField **TO** 200 in the x-axis, 785 in the y-axis, 400 as width, and 35 as height
**SET** background color of performanceIndexField **TO** primaryColor
**SET** foreground color of performanceIndexField **TO** secondaryColor
**SET** horizontal alignment of performanceIndexField **TO** CENTER
**SET** font of performanceIndexField **TO** mainFont
```

```
SET tutorImageIcon TO a new IMAGEICON loading the resource "Icons/userIconMain.png" using getClass() method
SET addTutorImage TO a new JLabel with tutorImageIcon
SET bounds of addTutorImage TO 250 in the x-axis, 20 in the y-axis, 40 as width, and 40 as height

SET salarySetIcon TO a new IMAGEICON loading the resource "Icons/salary.png" using getClass() method
SET salarySetImage TO a new JLabel with salarySetIcon
SET bounds of salarySetImage TO 250 in the x-axis, 100 in the y-axis, 40 as width, and 40 as height

SET removeTutorIcon TO a new IMAGEICON loading the resource "Icons/remove.png" using getClass() method
SET removeTutorImage TO a new JLabel with removeTutorIcon
SET bounds of removeTutorImage TO 250 in the x-axis, 180 in the y-axis, 40 as width, and 40 as height

SET displayTutorIcon TO a new IMAGEICON loading the resource "Icons/display.png" using getClass() method
SET displayTutorImage TO a new JLabel with displayTutorIcon
SET bounds of displayTutorImage TO 250 in the x-axis, 260 in the y-axis, 40 as width, and 40 as height

SET clearTutorIcon TO a new IMAGEICON loading the resource "Icons/clear.png" using getClass() method
SET clearTutorImage TO a new JLabel with clearTutorIcon
SET bounds of clearTutorImage TO 250 in the x-axis, 340 in the y-axis, 40 as width, and 40 as height
SET goBackIcon TO a new IMAGEICON loading the resource
"Icons/goBack.png" using getClass() method
SET goBackImageTutor TO a new JLabel with goBackIcon
SET bounds of goBackImageTutor TO 250 in the x-axis, 420 in the y-axis,
40 as width, and 40 as height


SET JButton addTutor with label "Add Tutor"
SET bounds of addTutor TO 100 in the x-axis, 20 in the y-axis, 150 as
width, and 40 as height
SET focusable property of addTutor TO false
SET border property of addTutor TO emptyBorder
SET cursor property of addTutor TO buttonCursor
SET background color of addTutor TO secondaryColor
SET foreground color of addTutor TO primaryColor
SET font of addTutor TO mainFont
ADD an ActionListener to addTutor

SET JButton salarySet with label "Set Salary"
SET bounds of salarySet TO 100 in the x-axis, 100 in the y-axis, 150 as
width, and 40 as height
SET focusable property of salarySet TO false
SET border property of salarySet TO emptyBorder
SET cursor property of salarySet TO buttonCursor
SET background color of salarySet TO secondaryColor
SET foreground color of salarySet TO primaryColor
SET font of salarySet TO mainFont
ADD an ActionListener to salarySet WHEN actionPerformed event occurs
DO
    SET visibility of setSalaryFrame TO TRUE
END DO
```

```
SET JButton removeTutor with label "Remove Tutor"
SET bounds of removeTutor TO 100 in the x-axis, 180 in the y-axis, 150
as width, and 40 as height
SET focusable property of removeTutor TO false
SET border property of removeTutor TO emptyBorder
SET cursor property of removeTutor TO buttonCursor
SET background color of removeTutor TO secondaryColor
SET foreground color of removeTutor TO primaryColor
SET font of removeTutor TO mainFont
ADD an ActionListener to removeTutor

SET JButton displayT with label "Display"
SET bounds of displayT TO 100 in the x-axis, 260 in the y-axis, 150 as
width, and 40 as height
SET focusable property of displayT TO false
SET border property of displayT TO emptyBorder
SET cursor property of displayT TO buttonCursor
SET background color of displayT TO secondaryColor
SET foreground color of displayT TO primaryColor
SET font of displayT TO mainFont
ADD an ActionListener to displayT

SET JButton clearT with label "Clear"
SET bounds of clearT TO 100 in the x-axis, 340 in the y-axis, 150 as
width, and 40 as height
SET focusable property of clearT TO false
SET cursor property of clearT TO buttonCursor
SET border property of clearT TO emptyBorder
SET background color of clearT TO secondaryColor
SET foreground color of clearT TO primaryColor
SET font of clearT TO mainFont
ADD an ActionListener to clearT

SET JButton goBackTutor with label "Go Back"
SET bounds of goBackTutor TO 100 in the x-axis, 420 in the y-axis, 150
as width, and 40 as height
SET focusable property of goBackTutor TO false
SET cursor property of goBackTutor TO buttonCursor
SET border property of goBackTutor TO emptyBorder
SET background color of goBackTutor TO secondaryColor
SET foreground color of goBackTutor TO primaryColor
SET font of goBackTutor TO mainFont
ADD an ActionListener to goBackTutor WHEN an event occurs trigger
actionPerformed method
DO
    SET visibility of tutorFrame TO FALSE
    SET visibility of welcomeFrame to TRUE
END DO

SET tutorContentPanel TO a JPanel with null layout
SET background color of tutorContentPanel TO primaryColor
SET preferred size of tutorContentPanel TO 800 as width, 900 as height

SET rightTutorTop TO a JPanel with FlowLayout centered horizontally,
with horizontal gap 20 and vertical gap 200
SET background color of rightTutorTop TO secondaryColor

SET rightTutorBottom TO a JPanel with null layout
SET background color of rightTutorBottom TO secondaryColor

SET mainTutor TO a JPanel with FlowLayout
SET background color of mainTutor TO primaryColor

SET rightTutor TO a JPanel with GridLayout, 2 rows and 0 columns
SET background color of rightTutor TO secondaryColor

ADD teacherIdLabelT TO tutorContentPanel
ADD teacherNameLabelT TO tutorContentPanel
ADD addressLabelT TO tutorContentPanel
ADD workingTypeLabelT TO tutorContentPanel
ADD employmentStatLabelT TO tutorContentPanel
ADD workingHoursLabelT TO tutorContentPanel
ADD salaryLabel TO tutorContentPanel
ADD specializationLabel TO tutorContentPanel
ADD academicQualificationLabel TO tutorContentPanel
ADD performanceIndexLabel TO tutorContentPanel

ADD teacherIdFieldT TO tutorContentPanel
ADD teacherNameFieldT TO tutorContentPanel
ADD addressFieldT TO tutorContentPanel
ADD workingTypeFieldT TO tutorContentPanel
ADD employmentStatFieldT TO tutorContentPanel
ADD workingHoursFieldT TO tutorContentPanel
ADD performanceIndexField TO tutorContentPanel
ADD academicQualificationField TO tutorContentPanel
ADD specializationField TO tutorContentPanel
ADD salaryField TO tutorContentPanel

ADD headerT TO rightTutorTop

ADD addTutorImage TO rightTutorBottom
ADD salarySetImage TO rightTutorBottom
ADD removeTutorImage TO rightTutorBottom
ADD displayTutorImage TO rightTutorBottom
ADD clearTutorImage TO rightTutorBottom
ADD goBackImageTutor TO rightTutorBottom

ADD addTutor TO rightTutorBottom
ADD salarySet TO rightTutorBottom
ADD removeTutor TO rightTutorBottom
ADD displayT TO rightTutorBottom
ADD clearT TO rightTutorBottom
ADD goBackTutor TO rightTutorBottom

ADD rightTutorTop TO rightTutor
ADD rightTutorBottom TO rightTutor

ADD tutorContentPanel TO mainTutor

SET Extended State of tutorFrame TO MAXIMIZED BOTH
SET size of tutorFrame TO 1200 as width, 900 as height
SET default close operation of tutorFrame TO HIDE ON CLOSE
SET layout of tutorFrame TO BorderLayout
ADD mainTutor TO tutorFrame AT CENTER of Border Layout
ADD rightTutor TO tutorFrame AT EAST of Border Layout

SET headerL TO a new JTextArea with text "Add Lecturers and Grade
Assignments"
SET editable property of headerL TO false
SET lineWrap property of headerL TO true
SET wrapStyleWord property of headerL TO true
SET background color of headerL TO secondaryColor
SET foreground color of headerL TO primaryColor
SET font of headerL TO homePageHeader
SET preferred size of headerL TO 400 as width, 200 as height

SET teacherIdLabelL TO a new JLabel with text "Teacher ID:"
SET bounds of teacherIdLabelL TO 200 in the x-axis, 50 in the y-axis, 190 as width, and 40 as height
SET foreground color of teacherIdLabelL TO secondaryColor
SET font of teacherIdLabelL TO mainFont

SET teacherNameLabelL TO a new JLabel with text "Teacher Name: "
SET bounds of teacherNameLabelL TO 200 in the x-axis, 130 in the y-axis, 190 as width, and 40 as height
SET foreground color of teacherNameLabelL TO secondaryColor
SET font of teacherNameLabelL TO mainFont

SET addressLabelL TO a new JLabel with text "Address:"
SET bounds of addressLabelL TO 200 in the x-axis, 210 in the y-axis, 190 as width, and 40 as height
SET foreground color of addressLabelL TO secondaryColor
SET font of addressLabelL TO mainFont

SET workingTypeLabelL TO a new JLabel with text "Working Type:"
SET bounds of workingTypeLabelL TO 200 in the x-axis, 290 in the y-axis, 190 as width, and 40 as height
SET foreground color of workingTypeLabelL TO secondaryColor
SET font of workingTypeLabelL TO mainFont

SET employmentStatLabelL TO a new JLabel with text "Employment Status:"
SET bounds of employmentStatLabelL TO 200 in the x-axis, 370 in the y-axis, 190 as width, and 40 as height
SET foreground color of employmentStatLabelL TO secondaryColor
SET font of employmentStatLabelL TO mainFont

SET yrsOfExperienceLabel TO a new JLabel with text "Years of Experience:"
SET bounds of yrsOfExperienceLabel TO 200 in the x-axis, 450 in the y-axis, 190 as width, and 40 as height
SET foreground color of yrsOfExperienceLabel TO secondaryColor
SET font of yrsOfExperienceLabel TO mainFont

SET gradedScoreLabel TO a new JLabel with text "Graded Score:"
SET bounds of gradedScoreLabel TO 200 in the x-axis, 530 in the y-axis, 190 as width, and 40 as height
SET foreground color of gradedScoreLabel TO secondaryColor
SET font of gradedScoreLabel TO mainFont

SET departmentLabel TO a new JLabel with text "Department:"
SET bounds of departmentLabel TO 200 in the x-axis, 610 in the y-axis, 190 as width, and 40 as height
SET foreground color of departmentLabel TO secondaryColor
SET font of departmentLabel TO mainFont

SET workingHoursLabelL TO a new JLabel with text "Working Hours:"
SET bounds of workingHoursLabelL TO 200 in the x-axis, 690 in the y-axis, 190 as width, and 40 as height
SET foreground color of workingHoursLabelL TO secondaryColor
SET font of workingHoursLabelL TO mainFont

SET teacherIdFieldL TO a new JTextField
SET bounds of teacherIdFieldL TO 200 in the x-axis, 85 in the y-axis, 400 as width, and 40 as height
SET background color of teacherIdFieldL TO primaryColor
SET foreground color of teacherIdFieldL TO secondaryColor
SET horizontal alignment of teacherIdFieldL TO CENTER
SET font of teacherIdFieldL TO mainFont

SET teacherNameFieldL TO a new JTextField
SET bounds of teacherNameFieldL TO 200 in the x-axis, 165 in the y-axis,
400 as width, and 40 as height
SET background color of teacherNameFieldL TO primaryColor
SET foreground color of teacherNameFieldL TO secondaryColor
SET horizontal alignment of teacherNameFieldL TO CENTER
SET font of teacherNameFieldL TO mainFont

SET addressFieldL TO a new JTextField
SET bounds of addressFieldL TO 200 in the x-axis, 245 in the y-axis, 400
as width, and 40 as height
SET background color of addressFieldL TO primaryColor
SET foreground color of addressFieldL TO secondaryColor
SET horizontal alignment of addressFieldL TO CENTER
SET font of addressFieldL TO mainFont

SET workingTypeFieldL TO a new JTextField
SET bounds of workingTypeFieldL TO 200 in the x-axis, 325 in the y-axis,
400 as width, and 40 as height
SET background color of workingTypeFieldL TO primaryColor
SET foreground color of workingTypeFieldL TO secondaryColor
SET horizontal alignment of workingTypeFieldL TO CENTER
SET font of workingTypeFieldL TO mainFont

SET employmentStatFieldL TO a new JTextField
SET bounds of employmentStatFieldL TO 200 in the x-axis, 405 in the y-
axis, 400 as width, and 40 as height
SET background color of employmentStatFieldL TO primaryColor
SET foreground color of employmentStatFieldL TO secondaryColor
SET horizontal alignment of employmentStatFieldL TO CENTER
SET font of employmentStatFieldL TO mainFont

SET yrsOfExperienceField TO a new JTextField
SET bounds of yrsOfExperienceField TO 200 in the x-axis, 485 in the y-
axis, 400 as width, and 40 as height
SET background color of yrsOfExperienceField TO primaryColor
SET foreground color of yrsOfExperienceField TO secondaryColor
SET horizontal alignment of yrsOfExperienceField TO CENTER
SET font of yrsOfExperienceField TO mainFont

SET gradedScoreField TO a new JTextField
SET bounds of gradedScoreField TO 200 in the x-axis, 565 in the y-axis,
400 as width, and 40 as height
SET background color of gradedScoreField TO primaryColor
SET foreground color of gradedScoreField TO secondaryColor
SET horizontal alignment of gradedScoreField TO CENTER
SET font of gradedScoreField TO mainFont

SET departmentField TO a new JTextField
SET bounds of departmentField TO 200 in the x-axis, 645 in the y-axis,
400 as width, and 40 as height
SET background color of departmentField TO primaryColor
SET foreground color of departmentField TO secondaryColor
SET horizontal alignment of departmentField TO CENTER
SET font of departmentField TO mainFont
SET workingHoursFieldL TO a new JTextField
SET bounds of workingHoursFieldL TO 200 in the x-axis, 725 in the y-axis, 400 as width, and 40 as height
SET background color of workingHoursFieldL TO primaryColor
SET foreground color of workingHoursFieldL TO secondaryColor
SET horizontal alignment of workingHoursFieldL TO CENTER
SET font of workingHoursFieldL TO mainFont

SET addLecturerIcon TO a new IMAGEICON loading the resource "Icons/userIconMain.png"
SET addLecturerImage TO a new JLabel with addLecturerIcon
SET bounds of addLecturerImage TO 250 in the x-axis, 20 in the y-axis, 40 as width, and 40 as height

SET gradeAssignmentIcon TO a new IMAGEICON with the resource "Icons/gradeAssignment.png"
SET gradeAssignmentImage TO a new JLabel with gradeAssignmentIcon
SET bounds of gradeAssignmentImage TO 250 in the x-axis, 100 in the y-axis, 40 as width, and 40 as height

SET displayLecturerIcon TO a new IMAGEICON loading the resource "Icons/display.png"
SET displayLecturerImage TO a new JLabel with displayLecturerIcon
SET bounds of displayLecturerImage TO 250 in the x-axis, 180 in the y-axis, 40 as width, and 40 as height

SET clearLecturerIcon TO a new IMAGEICON loading the resource "Icons/clear.png"
SET clearLecturerImage TO a new JLabel with clearLecturerIcon
SET bounds of clearLecturerImage TO 250 in the x-axis, 260 in the y-axis, 40 as width, and 40 as height
SET goBackImageLecturer TO a new JLabel with goBackIcon
SET bounds of goBackImageLecturer TO 250 in the x-axis, 340 in the y-
axis, 40 as width, and 40 as height

SET addLecturer TO a new JButton with label "Add Lecturer"
SET bounds of addLecturer TO 100 in the x-axis, 20 in the y-axis, 150 as
width, and 40 as height
SET focusable property of addLecturer TO false
SET border property of addLecturer TO emptyBorder
SET cursor property of addLecturer TO buttonCursor
SET background color of addLecturer TO secondaryColor
SET foreground color of addLecturer TO primaryColor
SET font of addLecturer TO mainFont
ADD an ActionListener to addLecturer

SET gradeAssignment TO a new JButton with label "Grade Assignment"
SET bounds of gradeAssignment TO 60 in the x-axis, 100 in the y-axis,
190 as width, and 40 as height
SET focusable property of gradeAssignment TO false
SET border property of gradeAssignment TO emptyBorder
SET cursor property of gradeAssignment TO buttonCursor
SET background color of gradeAssignment TO secondaryColor
SET foreground color of gradeAssignment TO primaryColor
SET font of gradeAssignment TO mainFont
ADD an ActionListener to gradeAssignment WHEN an event occurs
trigger actionPerformed method
DO
    SET visibility of gradeAssignmentFrame TO TRUE
END DO
```

```
SET displayL TO a new JButton with label "Display"
SET bounds of displayL TO 100 in the x-axis, 180 in the y-axis, 150 as
width, and 40 as height
SET focusable property of displayL TO false
SET border property of displayL TO emptyBorder
SET cursor property of displayL TO buttonCursor
SET background color of displayL TO secondaryColor
SET foreground color of displayL TO primaryColor
SET font of displayL TO mainFont
ADD an ActionListener to displayL

SET clearL TO a new JButton with label "Clear"
SET bounds of clearL TO 100 in the x-axis, 260 in the y-axis, 150 as
width, and 40 as height
SET focusable property of clearL TO false
SET border property of clearL TO emptyBorder
SET cursor property of clearL TO buttonCursor
SET background color of clearL TO secondaryColor
SET foreground color of clearL TO primaryColor
SET font of clearL TO mainFont
ADD an ActionListener to clearL

SET goBackLecturer TO a new JButton with label "Go Back"
SET bounds of goBackLecturer TO 100 in the x-axis, 340 in the y-axis,
150 as width, and 40 as height
SET focusable property of goBackLecturer TO false
SET cursor property of goBackLecturer TO buttonCursor
SET border property of goBackLecturer TO emptyBorder
SET background color of goBackLecturer TO secondaryColor
SET foreground color of goBackLecturer TO primaryColor
SET font of goBackLecturer TO mainFont
```

```
ADD an ActionListener to goBackLecturer WHEN an event occurs trigger
actionPerformed method
DO
        SET visibility of lecturerFrame TO FALSE
        SET visibility of welcomeFrame to TRUE
END DO

SET mainLecturerContent to a new JPanel with a null layout
SET the background color of mainLecturerContent TO primaryColor
SET the preferred size of mainLecturerContent TO 800 as width, 900 as
height

SET rightLecturerTop to a new JPanel with a FlowLayout centered
horizontally, with a horizontal gap of 20 and a vertical gap of 200
SET the background color of rightLecturerTop TO secondaryColor

SET rightLecturerBottom to a new JPanel with a null layout
SET the background color of rightLecturerBottom TO secondaryColor

SET mainLecturer a new JPanel with a FlowLayout
SET the background color of mainLecturer TO primaryColor

SET rightLecturer a new JPanel with a GridLayout of 2 rows and 0
columns
SET the background color of rightLecturer TO secondaryColor

ADD teacherIdLabelL TO mainLecturerContent
ADD teacherNameLabelL TO mainLecturerContent
ADD addressLabelL TO mainLecturerContent
ADD workingTypeLabelL TO mainLecturerContent
ADD employmentStatLabelL TO mainLecturerContent
ADD yrsOfExperienceLabel TO mainLecturerContent
ADD gradedScoreLabel TO mainLecturerContent
ADD departmentLabel TO mainLecturerContent
ADD workingHoursLabelL TO mainLecturerContent

ADD teacherIdFieldL TO mainLecturerContent
ADD teacherNameFieldL TO mainLecturerContent
ADD addressFieldL TO mainLecturerContent
ADD workingTypeFieldL TO mainLecturerContent
ADD employmentStatFieldL TO mainLecturerContent
ADD yrsOfExperienceField TO mainLecturerContent
ADD gradedScoreField TO mainLecturerContent
ADD departmentField TO mainLecturerContent
ADD workingHoursFieldL TO mainLecturerContent

ADD addLecturerImage TO rightLecturerBottom
ADD gradeAssignmentImage TO rightLecturerBottom
ADD displayLecturerImage TO rightLecturerBottom
ADD clearLecturerImage TO rightLecturerBottom
ADD goBackImageLecturer TO rightLecturerBottom

ADD addLecturer TO rightLecturerBottom
ADD gradeAssignment TO rightLecturerBottom
ADD displayL TO rightLecturerBottom
ADD clearL TO rightLecturerBottom
ADD goBackLecturer TO rightLecturerBottom

ADD headerL TO rightLecturerTop

ADD mainLecturerContent TO mainLecturer
ADD rightLecturerTop TO rightLecturer
ADD rightLecturerBottom TO rightLecturer

SET extendedState of lecturerFrame TO JFrame.MAXIMIZED_BOTH
SET size of lecturerFrame TO 1200 as width and 900 as height
SET defaultCloseOperation of lecturerFrame TO
JFrame.HIDE_ON_CLOSE
SET layout of lecturerFrame TO new BorderLayout()
ADD mainLecturer TO lecturerFrame at BorderLayout.CENTER
ADD rightLecturer TO lecturerFrame at BorderLayout.EAST

SET headerGrade TO a new JLabel with text "Grade Assignments"
SET bounds of headerGrade TO 50 in the x-axis, 100 in the y-axis, 250 as
width, and 40 as height
SET foreground color of headerGrade TO primaryColor
SET font of headerGrade TO headerFont

SET teacherIdGradeLabel TO a new JLabel with text "Teacher ID:"
SET bounds of teacherIdGradeLabel TO 50 in the x-axis, 50 in the y-axis,
190 as width, and 40 as height
SET foreground color of teacherIdGradeLabel TO secondaryColor
SET font of teacherIdGradeLabel TO mainFont

SET gradedScoreGradeLabel TO a new JLabel with text "Graded Score:"
SET bounds of gradedScoreGradeLabel TO 50 in the x-axis, 130 in the y-
axis, 190 as width, and 40 as height
SET foreground color of gradedScoreGradeLabel TO secondaryColor
SET font of gradedScoreGradeLabel TO mainFont

SET departmentGradeLabel TO a new JLabel with text "Department:"
SET bounds of departmentGradeLabel TO 50 in the x-axis, 210 in the y-axis, 190 as width, and 40 as height
SET foreground color of departmentGradeLabel TO secondaryColor
SET font of departmentGradeLabel TO mainFont

SET yrsOfExperienceGradeLabel TO a new JLabel with text "Years of Experience:"
SET bounds of yrsOfExperienceGradeLabel TO 50 in the x-axis, 290 in the y-axis, 190 as width, and 40 as height
SET foreground color of yrsOfExperienceGradeLabel TO secondaryColor
SET font of yrsOfExperienceGradeLabel TO mainFont

SET teacherIdGradeField TO a new JTextField
SET bounds of teacherIdGradeField TO 50 in the x-axis, 85 in the y-axis, 200 as width, and 40 as height
SET background color of teacherIdGradeField TO primaryColor
SET foreground color of teacherIdGradeField TO secondaryColor
SET horizontal alignment of teacherIdGradeField TO CENTER
SET font of teacherIdGradeField TO mainFont

SET gradedScoreGradeField TO a new JTextField
SET bounds of gradedScoreGradeField TO 50 in the x-axis, 165 in the y-axis, 200 as width, and 40 as height
SET background color of gradedScoreGradeField TO primaryColor
SET foreground color of gradedScoreGradeField TO secondaryColor
SET horizontal alignment of gradedScoreGradeField TO CENTER
SET font of gradedScoreGradeField TO mainFont

SET departmentGradeField TO a new JTextField
SET bounds of departmentGradeField TO 50 in the x-axis, 245 in the y-axis, 200 as width, and 40 as height
```

```
**SET** background color of departmentGradeField **TO** primaryColor
**SET** foreground color of departmentGradeField **TO** secondaryColor
**SET** horizontal alignment of departmentGradeField **TO** CENTER
**SET** font of departmentGradeField **TO** mainFont

**SET** yrsOfExperienceGradeField **TO** a new JTextField
**SET** bounds of yrsOfExperienceGradeField **TO** 50 in the x-axis, 325 in the
y-axis, 200 as width, and 40 as height
**SET** background color of yrsOfExperienceGradeField **TO** primaryColor
**SET** foreground color of yrsOfExperienceGradeField **TO** secondaryColor
**SET** horizontal alignment of yrsOfExperienceGradeField **TO** CENTER
**SET** font of yrsOfExperienceGradeField **TO** mainFont

**SET** gradeAssignmentImageSmall **TO** a new JLabel with
gradeAssignmentIcon
**SET** bounds of gradeAssignmentImageSmall **TO** 220 in the x-axis, 185 in
the y-axis, 40 as width, and 40 as height

**SET** goBackGrade **TO** a new JLabel with goBackIcon
**SET** bounds of goBackGrade **TO** 220 in the x-axis, 265 in the y-axis, 40
as width, and 40 as height

**SET** gradeAssignmentGradeButton **TO** a new JButton with text "Grade
Assignment"
**SET** bounds of gradeAssignmentGradeButton **TO** 30 in the x-axis, 185 in
the y-axis, 190 as width, and 40 as height
**SET** focusable property of gradeAssignmentGradeButton **TO** false
**SET** border of gradeAssignmentGradeButton **TO** emptyBorder
**SET** background color of gradeAssignmentGradeButton **TO**
secondaryColor
**SET** foreground color of gradeAssignmentGradeButton **TO** primaryColor
```

```
**SET** font of gradeAssignmentGradeButton **TO** mainFont
**ADD** actionListener to gradeAssignmentGradeButton

**SET** goBackGradeButton **TO** a new JButton with text "Go Back"
**SET** bounds of goBackGradeButton **TO** 30 in the x-axis, 265 in the y-axis,
190 as width, and 40 as height
**SET** focusable property of goBackGradeButton **TO** false
**SET** border of goBackGradeButton **TO** emptyBorder
**SET** background color of goBackGradeButton **TO** secondaryColor
**SET** foreground color of goBackGradeButton **TO** primaryColor
**SET** font of goBackGradeButton **TO** mainFont
**ADD** actionListener to goBackGradeButton **WHEN** actionPerformed event
occurs
**DO**
        **SET** visibility of gradeAssignmentFrame **TO** FALSE
**END DO**

**SET** mainGradeAssignment **TO** a new JPanel
**SET** bounds of mainGradeAssignment **TO** 0 in the x-axis, 0 in the y-axis,
300 as width, and 450 as height
**SET** background color of mainGradeAssignment **TO** primaryColor
**SET** layout of mainGradeAssignment **TO** null


**SET** rightGradeAssignment **TO** a new JPanel
**SET** bounds of rightGradeAssignment **TO** 300 in the x-axis, 0 in the y-
axis, 300 as width, and 450 as height
**SET** background color of rightGradeAssignment **TO** secondaryColor
**SET** layout of rightGradeAssignment **TO** null

**ADD** teacherIdGradeLabel **TO** mainGradeAssignment
```

```
ADD gradedScoreGradeLabel TO mainGradeAssignment
ADD departmentGradeLabel TO mainGradeAssignment
ADD yrsOfExperienceGradeLabel TO mainGradeAssignment
ADD teacherIdGradeField TO mainGradeAssignment
ADD gradedScoreGradeField TO mainGradeAssignment
ADD departmentGradeField TO mainGradeAssignment
ADD yrsOfExperienceGradeField TO mainGradeAssignment

ADD gradeAssignmentImageSmall TO rightGradeAssignment
ADD goBackGrade TO rightGradeAssignment
ADD headerGrade TO rightGradeAssignment
ADD gradeAssignmentGradeButton TO rightGradeAssignment
ADD goBackGradeButton TO rightGradeAssignment

SET size of gradeAssigmentFrame TO 600 as width, 450 as height
SET default close operation of gradeAssigmentFrame TO HIDE ON
CLOSE
SET layout of gradeAssigmentFrame TO null
SET resizable property of gradeAssigmentFrame TO false
ADD mainGradeAssignment to gradeAssigmentFrame
ADD rightGradeAssignment to gradeAssigmentFrame

SET salaryHeader TO a new JLabel with text "Set Salary"
SET bounds of salaryHeader TO 90 in the x-axis, 100 in the y-axis, 250 as
width, 40 as height
SET foreground color of salaryHeader TO primaryColor
SET font of salaryHeader TO headerFont

SET teacherIdSalaryLabel TO a new JLabel with text "Teacher ID:"
SET bounds of teacherIdSalaryLabel TO 50 in the x-axis, 70 in the y-axis,
190 as width, 40 as height
```

```
SET foreground color of teacherIdSalaryLabel TO secondaryColor
SET font of teacherIdSalaryLabel TO mainFont

SET newSalaryLabel TO a new JLabel with text "Salary:"
SET bounds of newSalaryLabel TO 50 in the x-axis, 150 in the y-axis, 190 as width, 40 as height
SET foreground color of newSalaryLabel TO secondaryColor
SET font of newSalaryLabel TO mainFont

SET performanceIndexSalaryLabel TO a new JLabel with text "Performance Index:"
SET bounds of performanceIndexSalaryLabel TO 50 in the x-axis, 230 in the y-axis, 190 as width, 40 as height
SET foreground color of performanceIndexSalaryLabel TO secondaryColor
SET font of performanceIndexSalaryLabel TO mainFont

SET setSalaryImage TO a new JLabel with salarySetIcon
SET bounds of setSalaryImage TO 180 in the x-axis, 185 in the y-axis, 40 as width, 40 as height

SET goBackSalary TO a new JLabel with goBackIcon
SET bounds of goBackSalary TO 180 in the x-axis, 265 in the y-axis, 40 as width, 40 as height

SET teacherIdSalaryField TO a new JTextField
SET bounds of teacherIdSalaryField TO 50 in the x-axis, 105 in the y-axis, 200 as width, 40 as height
SET background color of teacherIdSalaryField TO primaryColor
SET foreground color of teacherIdSalaryField TO secondaryColor
SET horizontal alignment of teacherIdSalaryField TO CENTER
SET font of teacherIdSalaryField TO mainFont

SET newSalaryField TO a new JTextField
SET bounds of newSalaryField TO 50 in the x-axis, 185 in the y-axis, 200
as width, 40 as height
SET background color of newSalaryField TO primaryColor
SET foreground color of newSalaryField TO secondaryColor
SET horizontal alignment of newSalaryField TO CENTER
SET font of newSalaryField TO mainFont

SET performanceIndexSalaryField TO a new JTextField
SET bounds of performanceIndexSalaryField TO 50 in the x-axis, 265 in
the y-axis, 200 as width, 40 as height
SET background color of performanceIndexSalaryField TO primaryColor
SET foreground color of performanceIndexSalaryField TO secondaryColor
SET horizontal alignment of performanceIndexSalaryField TO CENTER
SET font of performanceIndexSalaryField TO mainFont

SET setSalaryButton TO a new JButton with text "Set Salary"
SET bounds of setSalaryButton TO 30 in the x-axis, 185 in the y-axis, 150
as width, 40 as height
SET focusable property of setSalaryButton TO false
SET border of setSalaryButton TO emptyBorder
SET cursor of setSalaryButton TO buttonCursor
SET background color of setSalaryButton TO secondaryColor
SET foreground color of setSalaryButton TO primaryColor
SET font of setSalaryButton TO mainFont

SET goBackSalaryButton TO a new JButton with text "Go Back"
SET bounds of goBackSalaryButton TO 30 in the x-axis, 265 in the y-axis,
150 as width, 40 as height
```

```
SET focusable property of goBackSalaryButton TO false
SET border of goBackSalaryButton TO emptyBorder
SET cursor of goBackSalaryButton TO buttonCursor
SET background color of goBackSalaryButton TO secondaryColor
SET foreground color of goBackSalaryButton TO primaryColor
SET font of goBackSalaryButton TO mainFont
SET action listener of goBackSalaryButton TO dispose setSalaryFrame

SET mainSalary TO a new JPanel
SET background color of mainSalary TO primaryColor
SET bounds of mainSalary TO 0 in the x-axis, 0 in the y-axis, 300 as
width, 450 as height
SET layout of mainSalary TO null

SET rightSalary TO a new JPanel
SET background color of rightSalary TO secondaryColor
SET bounds of rightSalary TO 300 in the x-axis, 0 in the y-axis, 300 as
width, 450 as height
SET layout of rightSalary TO null

ADD teacherIdSalaryLabel TO mainSalary
ADD newSalaryLabel TO mainSalary
ADD performanceIndexSalaryLabel TO mainSalary

ADD teacherIdSalaryField TO mainSalary
ADD newSalaryField TO mainSalary
ADD performanceIndexSalaryField TO mainSalary

ADD setSalaryImage TO rightSalary
ADD setSalaryButton TO rightSalary
ADD salaryHeader TO rightSalary
ADD goBackSalaryButton TO rightSalary
ADD goBackSalary TO rightSalary

SET size of setSalaryFrame TO 600 as width, 450 as height
```

```
        SET default close operation of setSalaryFrame TO
        JFrame.HIDE_ON_CLOSE
        SET layout of setSalaryFrame TO null
        SET resizable property of setSalaryFrame TO false
END DO

CREATE a method named actionPerformed with return type void with parameter
ActionEvent e
DO
        IF source of e is equal to addTutor THEN
        DO
                DECLARE a variable named teacherName as String and ASSIGN
                the value of teacherNameFieldT using the getText() method
                DECLARE a variable named address as String and ASSIGN the
                value of addressFieldT using the getText() method
                DECLARE a variable named workingType as String and ASSIGN
                the value of tworkingTypeFieldT using the getText() method
                DECLARE a variable named employmentStat as String and
                ASSIGN the value of employmentStatFieldT using the getText()
                method
                DECLARE a variable named academicQualification as String and
                ASSIGN the value of academicQualificationField using the
                getText() method
                DECLARE a variable named specialization as String and ASSIGN
                the value of specializationField using the getText() method

                IF teacherName is empty OR address is empty OR workingType is
                empty OR employmentStat is empty OR academicQualification is
                empty OR specialization is empty THEN
                DO
        DISPLAY an error message stating that the fields are empty
        using JOptionPane
    END DO
    ELSE
    DO
        TRY
        DO
            DECLARE a variable named teacherID and ASSIGN
            the value of teacherIdFieldT by parsing it as an
            Integer and using getText() method
            DECLARE a variable named salary and ASSIGN the
            value of salaryField by parsing it as a Double and
            using getText() method
            DECLARE a variable named performanceIndex and
            ASSIGN the value of performanceIndexField by
            parsing it as an Integer and using getText() method
            DECLARE a variable named workingHours and
            ASSIGN the value of workingHoursFieldT by parsing
            it as an Integer and using getText() method

            IF salary is less than 0 THEN
            DO
                DISPLAY an error message stating that salary
                cannot be less than 0 using JOptionPane
            END DO
            ELSE IF performanceIndex is less than 0 OR
            performanceIndex is greater than 10 THEN
            DO
                DISPLAY an error message stating that
                performanceIndex must be greater than 0 and
                less than 10 using JOptionPane
```

```
END DO


ELSE IF workingHours is less than 20 OR
workingHours is greater than 70 THEN
DO
    DISPLAY an error message stating that
    working hours must be greater than 20 and
    less than 70 using JOptionPane
END DO
ELSE
DO
    CREATE an object named tutors by using
    'new' keyword and initializing the constructor of
    Tutor by passing teacherID, teacherName,
    address, workingType, employmentStat,
    workingHours, salary, specialization,
    academicQualification, performanceIndex as
    arguments

    DECLARE a variable named "isAdded" and
    ASSIGN a boolean value false to it
    IF size of ArrayList Teacher is greater than 0
    THEN
    DO
        FOR an object of Teacher named
        tutorObj in ArrayList Teacher
        DO
            IF tutorObj is an instance of Tutor
            and teacherID is equal to teacher
            ID of tutorObj THEN
```

```
                DO
                    SET isAdded to true
                    BREAK the loop
                END DO
                END IF
            END DO
            END FOR

            IF isAdded is true THEN
            DO
                DISPLAY an error message
                stating tutor with the teacherID
                already exists using JOptionPane
            END DO
            ELSE
            DO
                ADD an object named tutor to the
                ArrayList Teacher
                DISPLAY a message stating tutor
                added using JOptionPane
            END DO
            END IF
        END DO
        ELSE
        DO
            ADD an object named tutor to the
            ArrayList Teacher
            DISPLAY a message stating tutor added
            using JOptionPane
        END DO
        END IF
```

```
                END DO
                END IF
            END DO
            CATCH a NumberFormatException exp
            DO
                DISPLAY a message stating that Teacher ID, Working
                Hours, Performance Index must be and Integer using
                JOptionPane
            END DO
        END DO
        END IF
    END DO
    ELSE IF source of e is addLecturer THEN
    DO
        DECLARE a variable named teacherName as String and ASSIGN
        the value of teacherNameFieldL using the getText() method
        DECLARE a variable named address as String and ASSIGN the
        value of addressFieldL using the getText() method
        DECLARE a variable named workingType as String and ASSIGN
        the value of tworkingTypeFieldL using the getText() method
        DECLARE a variable named employmentStat as String and
        ASSIGN the value of employmentStatFieldL using the getText()
        method
        DECLARE a variable named department as String and ASSIGN
        the value of department¿ using the getText() method
        IF teacherName is empty OR address is empty OR workingType is
        empty OR employmentStat is empty OR department is empty
        THEN
        DO
            DISPLAY an error message stating that fields are empty
            using JOptionPane
```

```
END DO
ELSE
DO
    TRY
    DO
        DECLARE a variable named teacherID and ASSIGN
        the value of teacherIdFieldL by parsing it as an
        Integer and using getText() method
        DECLARE a variable named yrsOfExperience and
        ASSIGN the value of yrsOfExperienceField by parsing
        it as a Integer and using getText() method
        DECLARE a variable named gradedScore and
        ASSIGN the value of gradedScoreField by parsing it
        as an Integer and using getText() method
        DECLARE a variable named workingHours and
        ASSIGN the value of workingHoursFieldL by parsing it
        as an Integer and using getText() method

        IF yrsOfExperience less than 5 OR yrsOfExperience
        is greater than 30 THEN
        DO
            DISPLAY an error message stating that Years
            of Experience must be greater than 5 and less
            than 30
        END DO
        ELSE IF gradedScore is less than 0 OR gradedScore
        is greater than 100 THEN
        DO
            DISPLAY an error message stating that
            gradedScore must be greater than 0 and less
            than 100 using JOptionPane
```

```
END DO
ELSE IF workingHours is greater than 50 and less
than 0
DO
    DISPLAY an error message stating that
    workingHours must be greater than 0 and less
    than 50
END DO
ELSE
DO
    CREATE an object named lecturers using
    ‘new’ keyword and initializing the constructor of
    Lecturer by passing department,
    yrsOfExperience, teacherName, teacherID,
    address, workingType, employmentStat,
    workingHours as arguments

    DECLARE a variable named isAdded and
    ASSIGN a boolean value false to it

    IF size of the ArrayList is greater than 0 THEN
    DO
        FOR an object of Teacher named
        lecturerObj in the ArrayList Teacher
        DO
            IF lecturerObj is an instance of
            Lecturer AND teacherID is equal
            to teacher ID of lecturerObj
            THEN
            DO
                SET isAdded to true
                BREAK the loop
            END DO
            END IF
        END DO
        END FOR

        IF isAdded is equal to true THEN
        DO
            DISPLAY an error message
            stating that the lecturer has
            already been added
        END DO
        ELSE
        DO
            ADD the object named lecturers
            to the ArrayList Teacher
            SET the gradedScore to the
            object lecturers gradedScore
            DISPLAY a message stating that
            the lecturer has been added
            using JOptionPane
        END DO
        END IF
    END DO
    ELSE
    DO
        ADD the object named lecturers to the
        ArrayList Teacher
        SET the gradedScore to the object
        lecturers gradedScore
```

```
                    DISPLAY a message stating that the
                    lecturer has been added using
                    JOptionPane
                END DO
                END IF
            END DO
            END IF
        END DO
        CATCH
        DO
            DISPLAY a message stating that Teacher Id, years of
            experience and graded score must be an integer
        END DO
    END DO
    ELSE IF source of e is removeTutor THEN
    DO
        TRY
        DO
            DECLARE a variable named teacherID and ASSIGN
            the value of teacherIdFieldT by parsing it as an
            Integer and using getText() method

            DECLARE a variable named tutorRemoved and
            ASSIGN a boolean value false to it

            IF size of the ArrayList is greater than 0 THEN
            DO
                FOR an object named tutorObj in the ArrayList
                Teacher
                DO
                IF tutorObj is an instance of Tutor and
                teacherID is equal to teacherID of
                tutorObj THEN
                DO
                    REMOVE the object tutorObj
                    from the ArrayList Teacher
                    DISPLAY a message stating that
                    the tutor has been removed using
                    JOptionPane
                    SET tutorRemoved to true
                    BREAK the loop
                END DO
                END IF
            END DO
            END FOR

            IF tutorRemoved is false THEN
            DO
                DISPLAY an error message stating that
                the teacher with that teacherID does not
                exists using JOptionPane
            END DO
            END IF
        END DO
        ELSE
        DO
            DISPLAY an error message stating that tutor
            has not been added
        END DO
        END IF
    END DO
```

```
        CATCH NumberFormatException exp
        DO
            DISPLAY an error message stating that teacher ID
            should be an integer using JOptionPane
        END DO
    END DO
    END IF
    ELSE IF source of e is gradeAssignmentGradeButton THEN
    DO
        DECLARE a variable named department and ASSIGN the
        value of departmentGradeField using getText() method
        IF department is empty THEN
        DO
            DISPLAY an error message stating that the fields are
            empty using JOptionPane
        END DO
        ELSE
        DO
            TRY
            DO
                DECLARE a variable named teacherID and
                ASSIGN the value of teacherIdGradeField by
                parsing it as an Integer and using getText()
                method
                DECLARE a variable named gradedScore and
                ASSIGN the value of gradedScoreGradeField
                by parsing it as an Integer and using getText()
                method
                DECLARE a variable named yrsOfExperience
                and ASSIGN the value of
```

```
yrsOfExperienceGradeField by parsing it as a
Integer and using getText() method

DECLARE a variable named lecturerFound
and ASSIGN a boolean value false to it

IF size of the ArrayList Teacher is greater than
0 THEN
DO
        FOR an object named obj in the
        ArrayList Teacher
        DO
                IF obj is an instance of Lecturer
                and teacherID is equal to
                teacherID of the object obj THEN
                DO
                        SET lecturerFound to true
                        CREATE an object named
                        lecturerObj of Lecturer
                        Class and ASSIGN the
                        object obj after
                        downcasting
                        IF yrsOfExperience is less
                        than 5 OR
                        yrsOfExperience is greater
                        than 80 THEN
                        DO
                                DISPLAY an error
                                message stating
                                years of experience
                                must be greater
        than 5 and less
        than 80
END DO
ELSE IF department is not
equal to department of
lecturerObj THEN
DO
        DISPLAY an error
        message stating
        that the department
        must be same
END DO
ELSE IF gradedScore is
greater than 100 and less
than 0 THEN
DO
        DISPLAY an error
        message stating
        that graded score
        must be greater
        than 0 and less
        than 100
END DO
ELSE
DO
        DECLARE a
        variable named
        grade
        IF gradedScore is
        greater than or
        equal to 70 THEN
```

```
DO
    ASSIGN
    grade ‘A’ to
    variable
    grade
END DO
ELSE IF
gradedScore is
greater than or
equal to 60 THEN
DO
    ASSIGN
    grade ‘B’ to
    variable
    grade
END DO
ELSE IF
gradedScore is
greater than or
equal to 50 THEN
DO
    ASSIGN
    grade ‘C’ to
    variable
    grade
END DO
ELSE IF
gradedScore is
greater than or
equal to 40 THEN
DO
```

```
                        **ASSIGN**
                        grade ‘D’ to
                        variable
                        grade
                    **END DO**
                    **ELSE**
                    **DO**
                        **ASSIGN**
                        grade ‘E’ to
                        variable
                        grade
                    **END DO**
                    **END IF**

                    **CALL** a method
                    named
                    gradeAssignment
                    using lecturerObj
                    and pass
                    gradedScore,
                    department,
                    yrsOfExperience as
                    arguments
                    **DISPLAY** the grade
                    using JOptionPane
                    **BREAK** the loop
                **END DO**
                **END IF**
            **END DO**
            **END IF**
        **END DO**
```

```
                    END FOR
                    IF lecturerFound is false THEN
                    DO
                        DISPLAY an error message
                        stating that the lecturer was not
                        found using JOptionPane
                    END DO
                    END IF
                END DO
                ELSE
                DO
                    DISPLAY an error message stating
                    lecturer has not been added using
                    JOptionPane
                END DO
                END IF
            END DO
            CATCH NumberFormatException exp
            DO
                DISPLAY an error message stating that
                TeacherID, Graded Score and Years of
                Experience must be an integer
            END DO
        END DO
        END IF
    END DO
    ELSE IF source of e is setSalaryButton
    DO
        TRY
        DO
```

```
DECLARE a variable named teacherID and ASSIGN
the value of teacherIdSalaryField by parsing it as an
Integer and using getText() method
DECLARE a variable named salary and ASSIGN the
value of newSalaryField by parsing it as an Integer
and using getText() method
DECLARE a variable named performanceIndex and
ASSIGN the value of performanceIndexSalaryField by
parsing it as an Integer and using getText() method

IF size of the ArrayList is greater than 0 THEN
DO
    FOR an object of Teacher class named obj in
    the ArrayList Teacher
    DO
        IF teacherID is equal to the teacherID of
        obj AND obj is an instance of Tutor
        THEN
        DO
            CREATE an object of Tutor class
            and ASSIGN the object obj after
            downcasting
            IF performanceIndex is less than
            5 and performanceIndex greater
            than 10
            DO
                DISPLAY an error
                message stating that
                performanceIndex must be
                greater than 5 and less
                than 10 using
                JOptionPane
        **END DO**
        **ELSE IF** workingHours of
        tutorObj is less than 20 **THEN**
        **DO**
                **DISPLAY** an error
                message stating that
                working hours must be
                greater than 20 using
                JOptionPane
        **END DO**
        **ELSE**
        **DO**
                **CALL** a method named
                setSalary using tutorObj
                and pass salary and
                performanceIndex as
                arguments
                **DISPLAY** the new salary
                and new
                performanceIndex using
                getter methods and
                JOptionPane
        **END DO**
        **END IF**
**END DO**
**ELSE**
**DO**
```

```
                    DISPLAY an error message
                    stating that tutor with that ID does
                    not exist
                END DO
                END IF
            END DO
            END FOR
        END DO
        ELSE
        DO
            DISPLAY an error message stating that tutor
            with that ID does not exist
        END DO
        END IF
    END DO
    CATCH NumberFormatException exp
    DO
        DISPLAY a message stating tutor has not been
        added using JOptionPane
    END DO
END DO
ELSE IF source of e is clearL THEN
DO
    DECLARE a variable named clear as integer and ASSIGN
    the value after asking the user to select YES, NO, CANCEL
    IF clear is equal to YES THEN
    DO
        CLEAR the text of teacherIdFieldL
        CLEAR the text of teacherNameFieldL
        CLEAR the text of addressFieldL
        CLEAR the text of workingTypeFieldL
```

```
                **CLEAR** the text of employmentStatFieldL
                **CLEAR** the text of gradedScoreField
                **CLEAR** the text of yrsOfExperienceField
                **CLEAR** the text of departmentField
                **CLEAR** the text of workingHoursFieldL
        **END DO**
        **END IF**
**END DO**
**ELSE IF** source of e is clearL **THEN**
**DO**
        **DECLARE** a variable named clear as integer and **ASSIGN**
        the value after asking the user to select **YES, NO, CANCEL**
        **IF** clear is equal to **YES THEN**
        **DO**
                **CLEAR** the text of teacherIdFieldT
                **CLEAR** the text of teacherNameFieldT
                **CLEAR** the text of addressFieldT
                **CLEAR** the text of workingTypeFieldT
                **CLEAR** the text of employmentStatFieldT
                **CLEAR** the text of workingHoursFieldT
                **CLEAR** the text of salaryField
                **CLEAR** the text of specializationField
                **CLEAR** the text of academicQualificationField
                **CLEAR** the text of performanceIndexField
        **END DO**
        **END IF**
**END DO**
**ELSE IF** source of e is displayT **THEN**
**DO**
        **IF** size of the ArrayList is greater than 0 **THEN**
        **DO**
```

```
            FOR an Teacher object named obj iterating in the
            ArrayList named Teacher
            DO
                IF obj is an instance of Tutor THEN
                DO
                    CREATE a Tutor object named tutorObj
                    and ASSIGN the object obj after
                    downcasting
                    DISPLAY teacher ID, name, address,
                    working type, employment status,
                    working hours, salary, specialization,
                    academic qualification, and performance
                    index using accessor methods in a
                    JOptionPane
                END DO
                END IF
            END DO
            END FOR
        END DO
        ELSE
        DO
            DISPLAY an error message stating that the tutor has
            not been added
        END DO
        END IF
    END DO
    ELSE IF source of e is display THEN
    DO
        IF size of the ArrayList is greater than 0 THEN
        DO
```

```
                    FOR an Teacher object named obj iterating in the
                    ArrayList named Teacher
                    DO
                        IF obj is an instance of Lecturer THEN
                        DO
                            CREATE a Lecturer object named
                            lecturerObj and ASSIGN the object obj
                            after downcasting
                            DISPLAY teacher ID, name, address,
                            working type, employment status,
                            working hours, graded score, and years
                            of experience using accessor methods
                            in a JOptionPane
                        END DO
                        END IF
                    END DO
                    END FOR
                END DO
                ELSE
                DO
                    DISPLAY an error message stating that lecturer has
                    not been added
                END DO
                END IF
            END DO
            END IF
        END DO
        END IF
    END DO
```

```
**CREATE** a method named main and pass String[] and args as parameters
**DO**
    **CREATE** a new TeacherGUI object
**END DO**
```

4. Method Descriptions

4.1 Methods of TeacherGUI class

Methods	Descriptions
actionPerformed(Action Event e)	It is a method which is a part of ActionListener Interface which gets invoked when the button registered to this method is pressed. In this method the code is written which is to be executed once pressed. It takes an object named ActionEvent as a parameter that contains the information about the occurring event
getSource()	It is a method used for event handling which is used to identify source or the component that triggered the event.
mouseEntered(MouseEvent ev)	It is a method which is a part of MouseListener Interface which gets invoked when the cursor enters the area of button component.
mouseExited(MouseEvent ev)	It is a method which is a part of MouseListener Interface which gets invoked when the cursor exits the area of button component.

4.2 Description of JButton events

When any of these buttons are pressed the getSource() method gets the source of the button press and triggers the actionPerformed(ActionEvent e) which handles events.

- **addTutor**

  When this button is pressed the values from the text fields are retrieved such as the tutor's name, address, employment details, academic qualifications, and specialization. It checks if any of these fields are empty, displaying an error message if so. If all fields are filled, it proceeds to parse integer and double inputs for tutor ID, salary, performance index, and working hours, respectively. It then validates these inputs: ensuring salary is non-negative, performance index is between 0 and 10, and working hours are between 20 and 70. If validations pass, a new Tutor object is created and added to a list of teachers, provided there's no existing tutor with the same ID. Error messages are shown for invalid inputs or existing tutor IDs. Exception handling is included to catch any non-integer or non-double inputs, prompting the user to enter valid numeric values.

- **addLecturer**

  When this button is pressed values are retrieved from all JTextField such as teacher name, address, and employment details. It then validates these inputs, ensuring none are empty. If any field is empty, an error message is displayed. If all fields are filled, the code proceeds to parse integer inputs for teacher ID, years of experience, graded score, and working hours. It checks if years of experience are between 5 and 30, graded score is between 0 and 100, and working hours are between 0 and 50. If all validations pass, a new Lecturer object is created and added to a list of teachers, provided there are no existing lecturers with the same ID. Error messages are shown for invalid inputs or existing lecturer IDs. Finally, exceptions are caught for any non-integer inputs in the try-catch block, prompting the user to enter valid integers.

- **removeTutor**

When this button is pressed the value of teacher ID from a text field is retrieved. Subsequently, it initializes a boolean variable, tutorRemoved, to track the success of the removal operation. If the list of teachers contains entries, it iterates through each teacher object. Upon finding a tutor object with a matching ID, it removes that tutor from the list, sets tutorRemoved to true, and displays a success message. If no tutor is removed, it indicates that the provided ID does not correspond to any existing tutor and prompts an error message. Additionally, it handles scenarios where the list of teachers is empty, notifying the user that no tutors have been added. Exception handling ensures that the program can manage cases where the provided ID cannot be parsed as an integer, displaying an appropriate error message in such instances.

- **gradeAssignmentGradeButton**

When this button is pressed it checks if the department field is empty, displaying an error message if it is. Then, it attempts to parse integers from input fields representing teacher ID, graded score, and years of experience. It iterates through a list of teachers, checking if the entered teacher ID corresponds to a lecturer and verifying certain conditions such as years of experience, department match, and the validity of the graded score. If all conditions are met, it grades the assignment based on the score and displays the grade. Error messages are shown for various exceptional cases like incorrect input format or when a lecturer is not found.

- **setSalaryButton**

When this button is pressed it parses input fields for teacher ID, new salary, and performance index. Then, it iterates through a list of teachers, checking if the entered teacher ID corresponds to a tutor. If found, it validates the performance index and working hours, displaying error messages if conditions aren't met. If validations pass, it updates the tutor's salary and displays the new salary and performance index. Error messages are shown for invalid input formats or when a tutor with the given ID isn't found.

- **clearL** and **clearT**

When these buttons are pressed the user is prompted to select an option whether to clear the fields or not, if user selects yes option then the text fields are cleared and if no or cancel is selected then no changes happen.

- **displayT**

When this button is pressed it checks if there are any tutors present in the list. If found, it retrieves and displays their ID, name, address, work type, employment status, working hours, salary, specialization, academic qualification, and performance index. If no tutors are present, it displays an error message indicating that no tutor has been added.

- **displayL**

When this button is pressed it checks if there are any teachers in the list. If found, it iterates through each teacher and checks if they are instances of the Lecturer class. If a lecturer is found, it retrieves and displays their ID, name, address, work type, employment status, working hours, graded score, and years of experience. If no lecturers are present in the list, it displays an error message indicating that no lecturer has been added.

5. Testing

5.1 Test 1 – Running the compiled program using Command Prompt

Test No.	1
Objective:	Run the compiled program using command prompt
Action:	Open command prompt, go the directory in which the project is stored and write these commands: • cd project file directory • java TeacherGUI.java
Expected Results:	The program should run and open
Actual Results:	The program opened
Conclusion:	The test was successful.

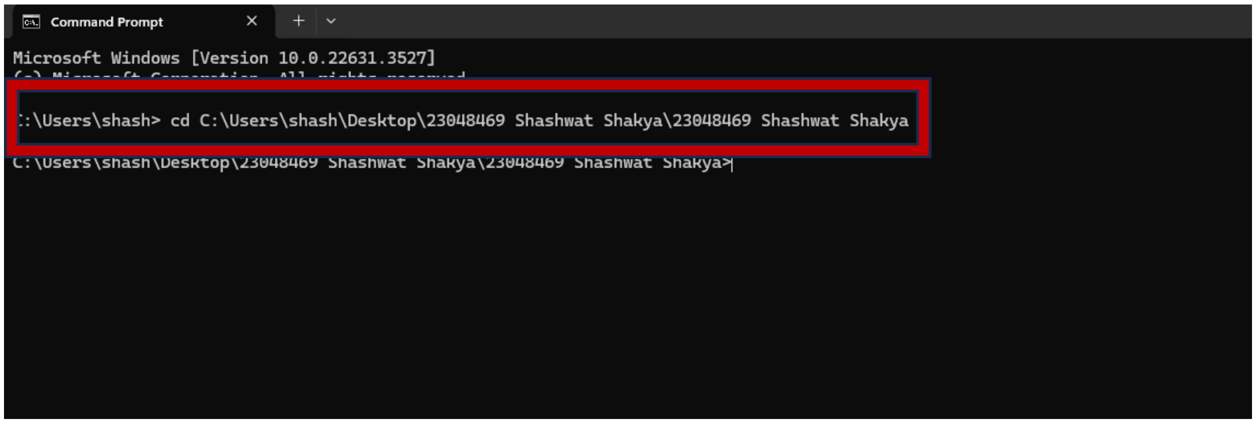
Results:

```
Microsoft Windows [Version 10.0.22631.3527]
(c) Microsoft Corporation. All rights reserved.

C:\Users\shash>
```

Figure 3 - Screenshot of Command Prompt

Opening Command Prompt



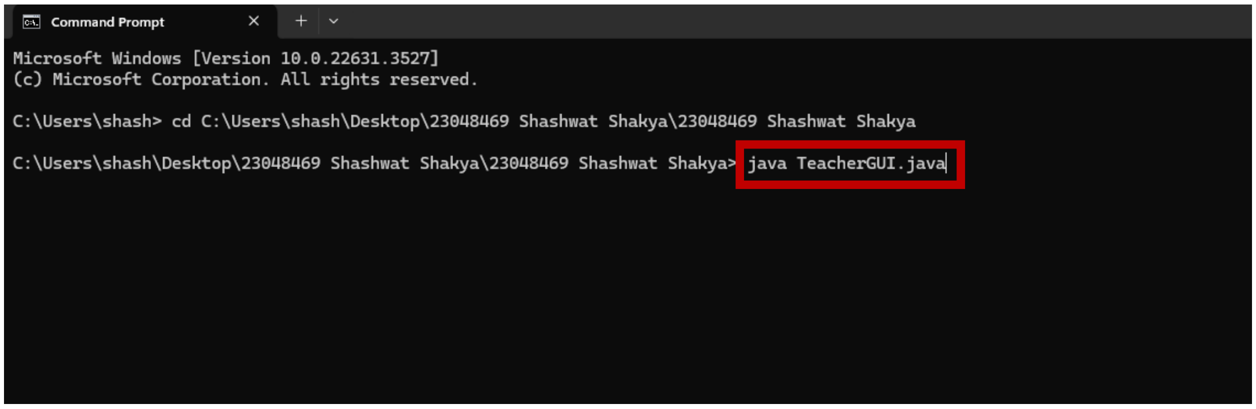
```
Microsoft Windows [Version 10.0.22631.3527]
(c) Microsoft Corporation. All rights reserved.

:\Users\shash> cd C:\Users\shash\Desktop\23048469 Shashwat Shakya\23048469 Shashwat Shakya

C:\Users\shash\Desktop\23048469 Shashwat Shakya\23048469 Shashwat Shakya>
```

Figure 4 - Screenshot of changing file directory

Changing the directory to the project folder



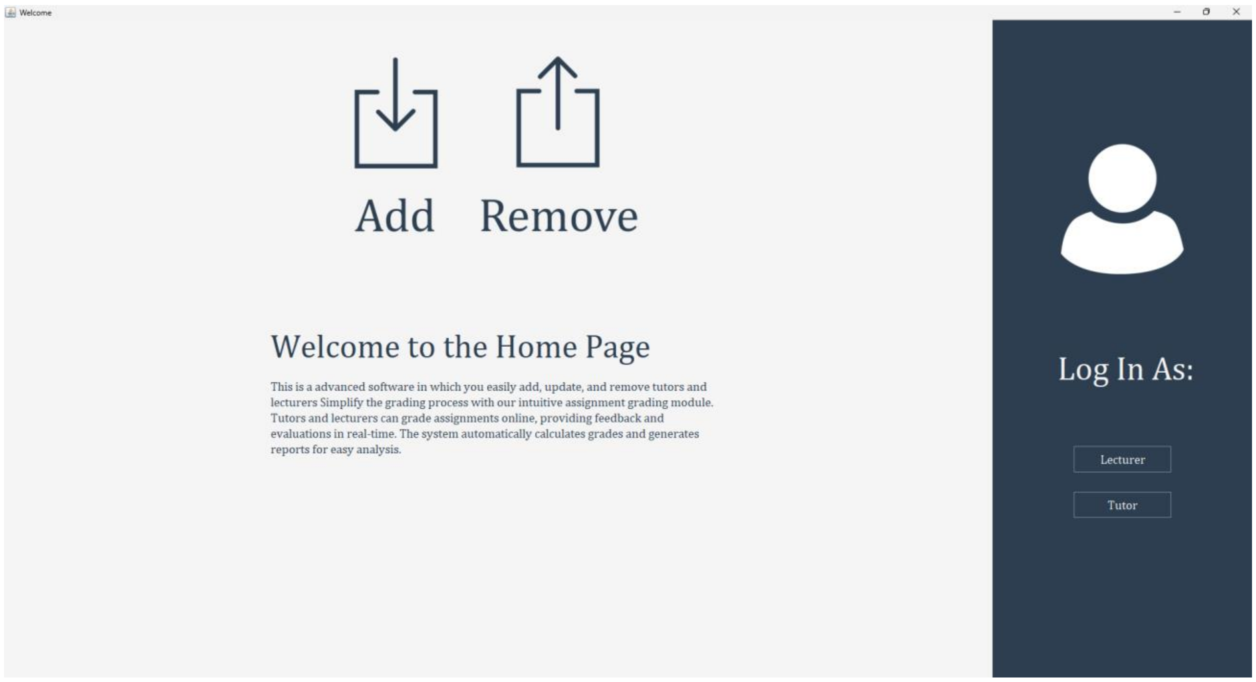
```
Microsoft Windows [Version 10.0.22631.3527]
(c) Microsoft Corporation. All rights reserved.

C:\Users\shash> cd C:\Users\shash\Desktop\23048469 Shashwat Shakya\23048469 Shashwat Shakya

C:\Users\shash\Desktop\23048469 Shashwat Shakya\23048469 Shashwat Shakya> java TeacherGUI.java
```

Figure 5 - Screenshot of entering the command

Entering the command “java TeacherGUI.java”.



Welcome to the Home Page

This is a advanced software in which you easily add, update, and remove tutors and lecturers Simplify the grading process with our intuitive assignment grading module. Tutors and lecturers can grade assignments online, providing feedback and evaluations in real-time. The system automatically calculates grades and generates reports for easy analysis.

Figure 6 - Screenshot of GUI

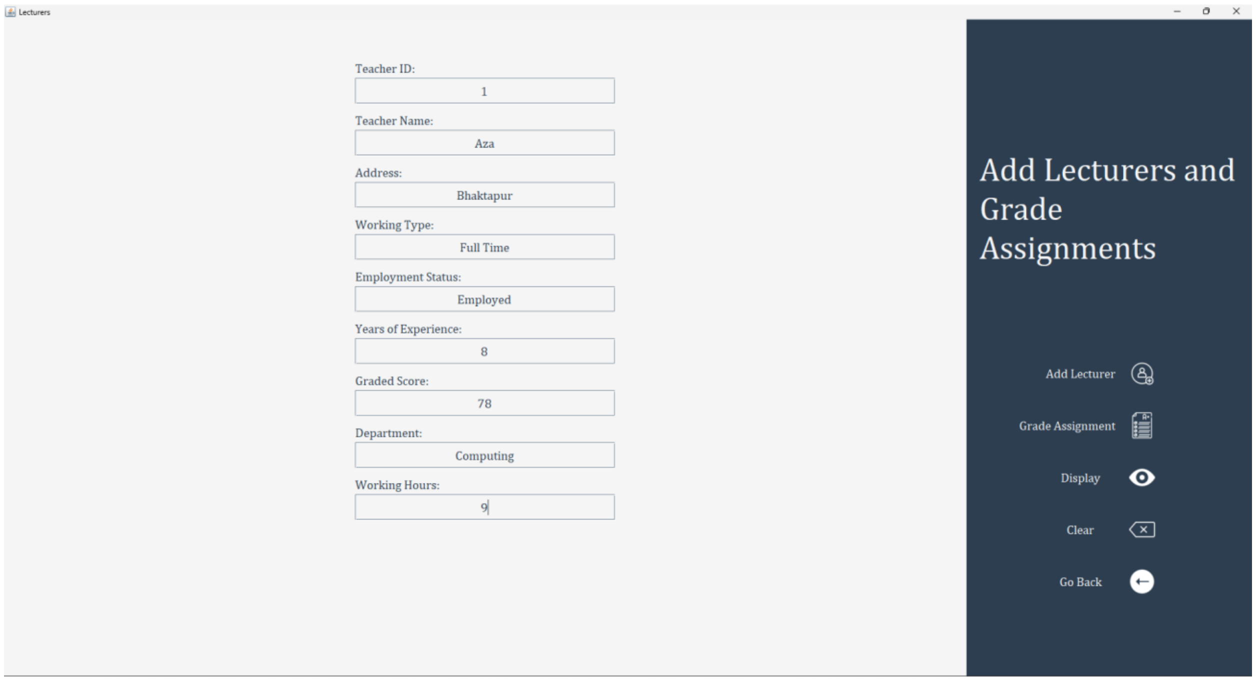
Program opened.

5.2 Various Tests of the GUI and Logic of the program

5.2.1 Test 1 – Adding the Lecturer

Test No.	2.1
Objective:	Adding a Lecturer using proper values
Action:	Entering the values for the fields: • Teacher ID – 1 • Teacher Name – “Aza” • Address – “Bhaktapur” • Working Type – “Full Time” • Employment Status – “Employed” • Years of Experience – 8 • Graded Score – 78 • Department – “Computing” • Working Hours – 9
Expected Results:	The Lecturer should be added
Actual Results:	The Lecturer is added
Conclusion:	The test was successful.

Results:



Lecturers

Teacher ID: 1

Teacher Name: Aza

Address: Bhaktapur

Working Type: Full Time

Employment Status: Employed

Years of Experience: 8

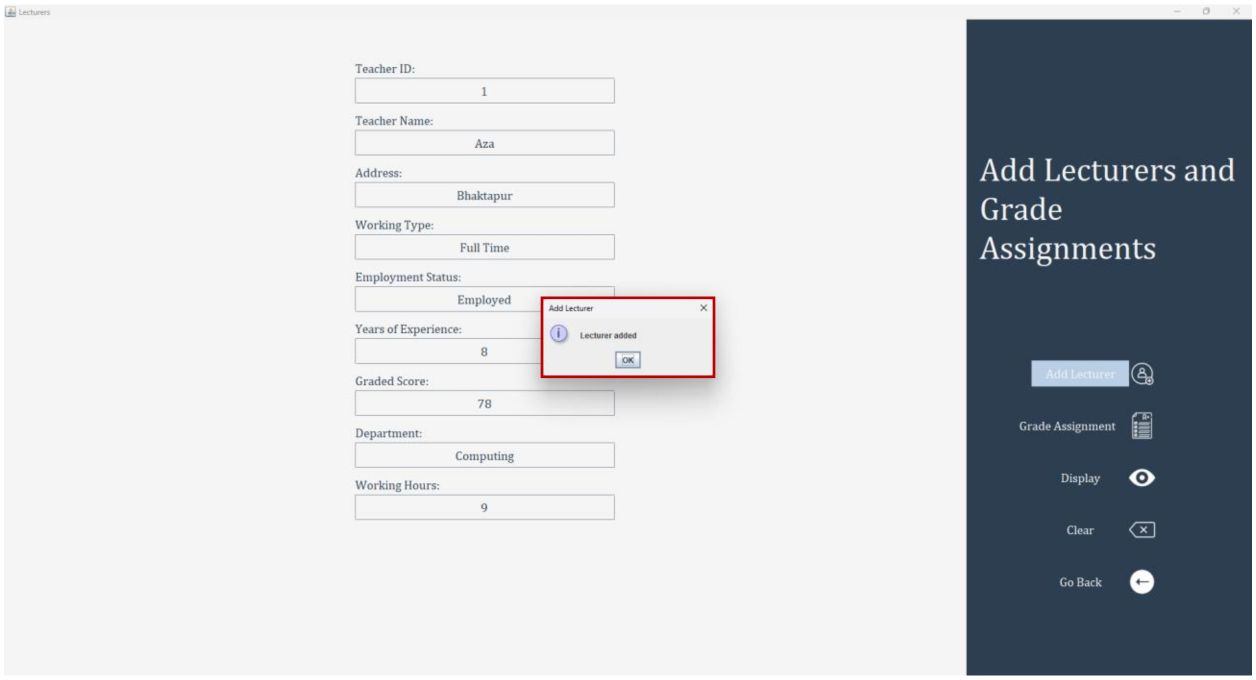
Graded Score: 78

Department: Computing

Working Hours: 9

Figure 7 - Screenshot of entering values

Enter the values in their respective fields.



Lecturers

Teacher ID: 1

Teacher Name: Aza

Address: Bhaktapur

Working Type: Full Time

Employment Status: Employed

Years of Experience: 8

Graded Score: 78

Department: Computing

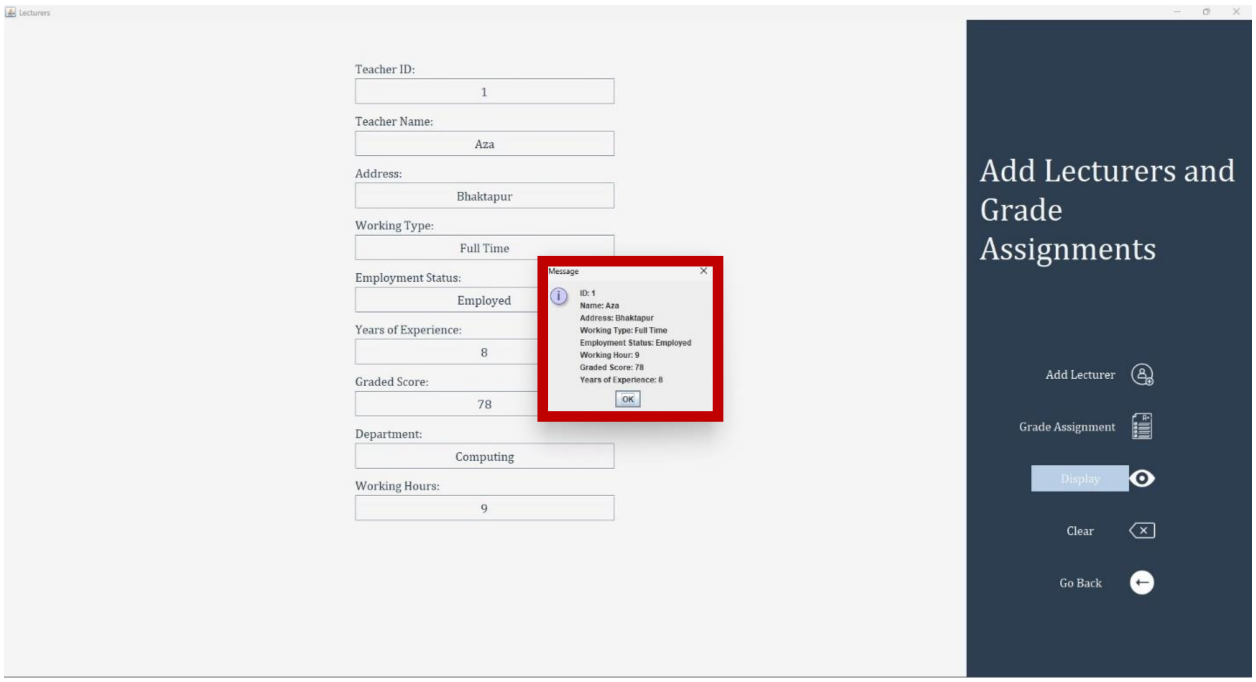
Working Hours: 9

Add Lecturer

Lecturer added

Figure 8 - Screenshot of adding lecturer

The Lecturer was successfully added.


Teacher ID: 1
Teacher Name: Aza
Address: Bhaktapur
Working Type: Full Time
Employment Status: Employed
Years of Experience: 8
Graded Score: 78
Department: Computing
Working Hours: 9

```
Message
ID: 1
Name: Aza
Address: Bhaktapur
Working Type: Full Time
Employment Status: Employed
Working Hour: 9
Graded Score: 78
Years of Experience: 8
```

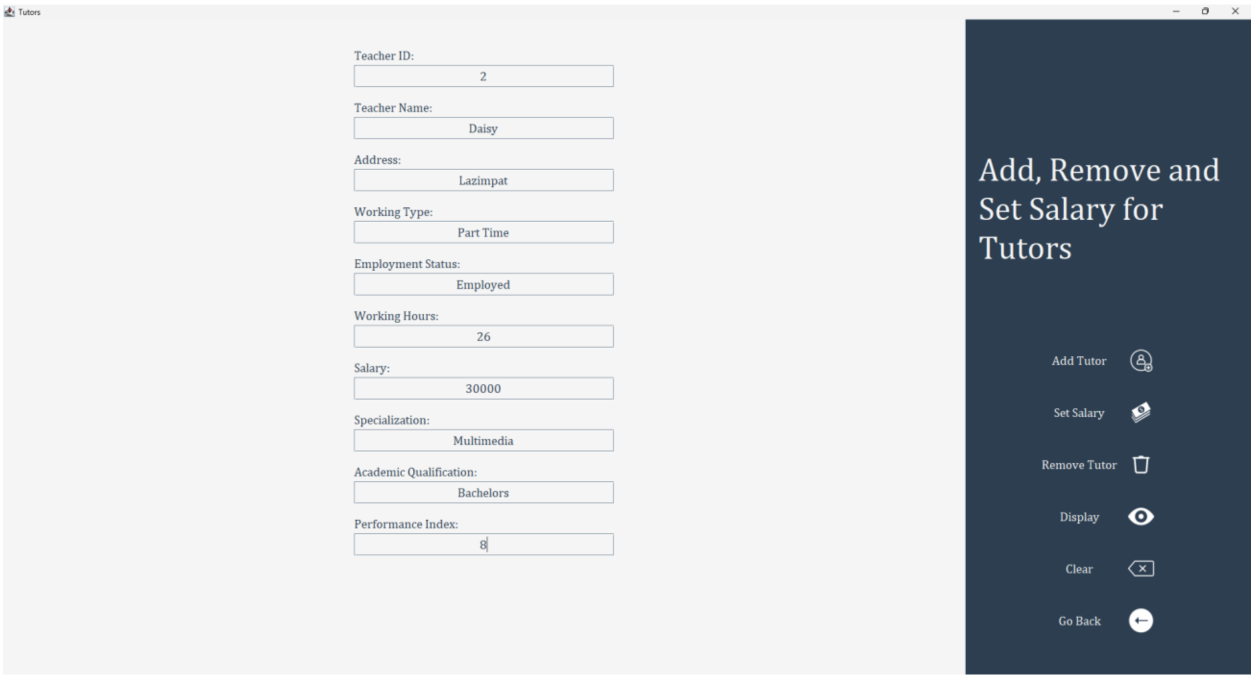
Figure 9 - Screenshot of details of lecturer

Displaying the details of lecturer

5.2.2 Test 2 – Adding a Tutor

Test No.	2.2
Objective:	Adding a Tutor using proper values
Action:	Entering the values for the fields: • Teacher ID – 2 • Teacher Name – “Daisy” • Address – “Lazimpat” • Working Type – “Part Time” • Employment Status – “Employed” • Working Hours – 26 • Salary – 50000 • Specialization – “Multimedia” • Academic Qualification – “Bachelors” • Performance Index – 8
Expected Results:	The Tutor should be added
Actual Results:	The Tutor is added
Conclusion:	The test was successful.

Results:


Tutors

Teacher ID:
2

Teacher Name:
Daisy

Address:
Lazimpat

Working Type:
Part Time

Employment Status:
Employed

Working Hours:
26

Salary:
30000

Specialization:
Multimedia

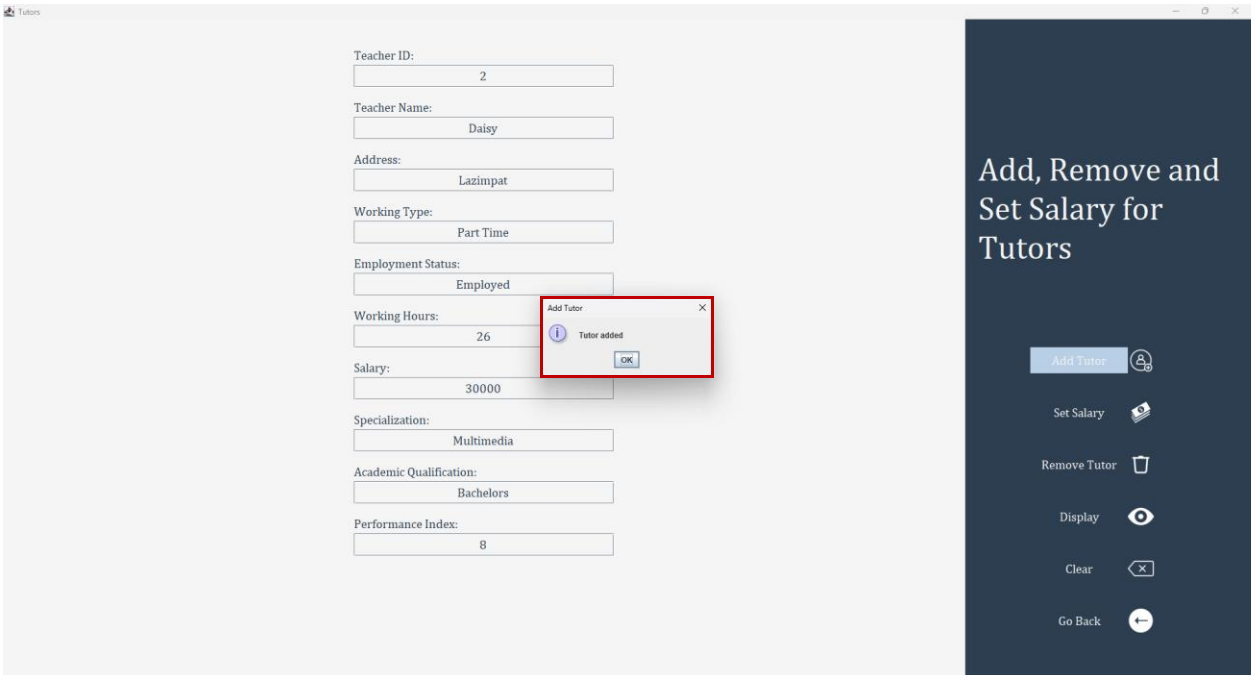
Academic Qualification:
Bachelors

Performance Index:
8

Add, Remove and Set Salary for Tutors

Figure 10 - Screenshot of entering values

Entering values in their respective fields.


Tutors

Teacher ID:
2

Teacher Name:
Daisy

Address:
Lazimpat

Working Type:
Part Time

Employment Status:
Employed

Working Hours:
26

Salary:
30000

Specialization:
Multimedia

Academic Qualification:
Bachelors

Performance Index:
8

Add Tutor

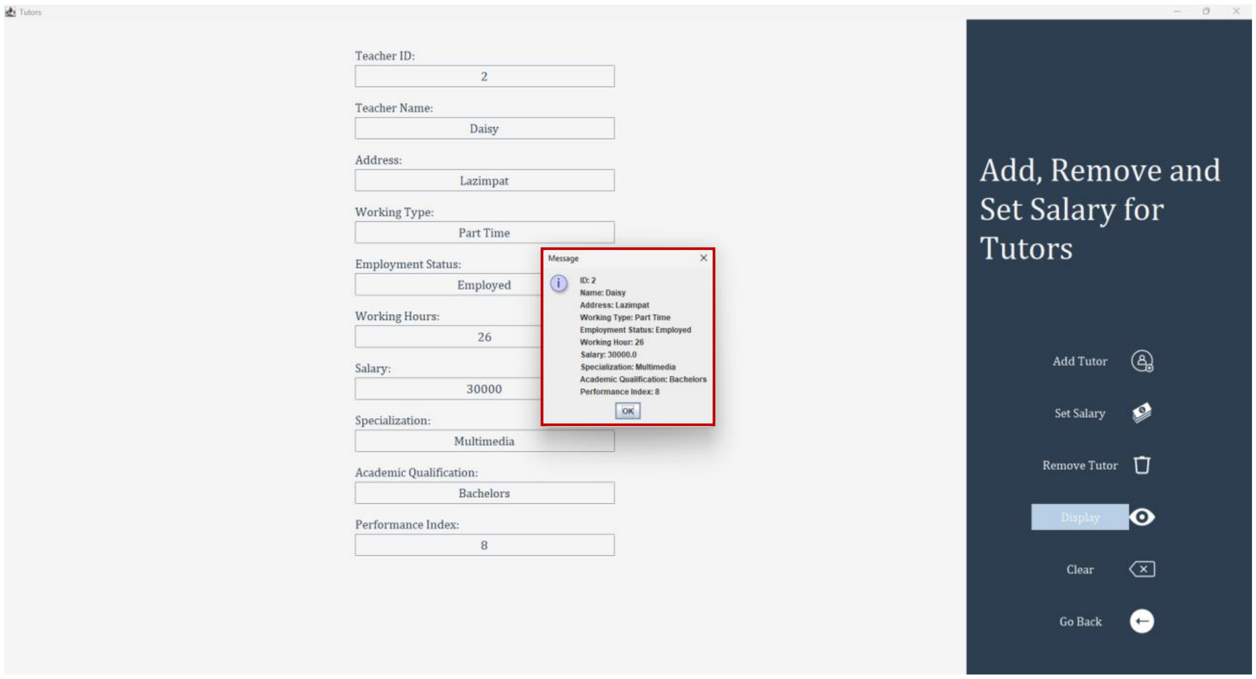
Tutor added

OK

Add, Remove and Set Salary for Tutors

Figure 11 - Screenshot of adding tutor

Successfully added a Tutor.


Tutors

Teacher ID: 2

Teacher Name: Daisy

Address: Lazimpat

Working Type: Part Time

Employment Status: Employed

Working Hours: 26

Salary: 30000

Specialization: Multimedia

Academic Qualification: Bachelors

Performance Index: 8

Message

```
ID: 2
Name: Daisy
Address: Lazimpat
Working Type: Part Time
Employment Status: Employed
Working Hour: 26
Salary: 30000.0
Specialization: Multimedia
Academic Qualification: Bachelors
Performance Index: 8
```

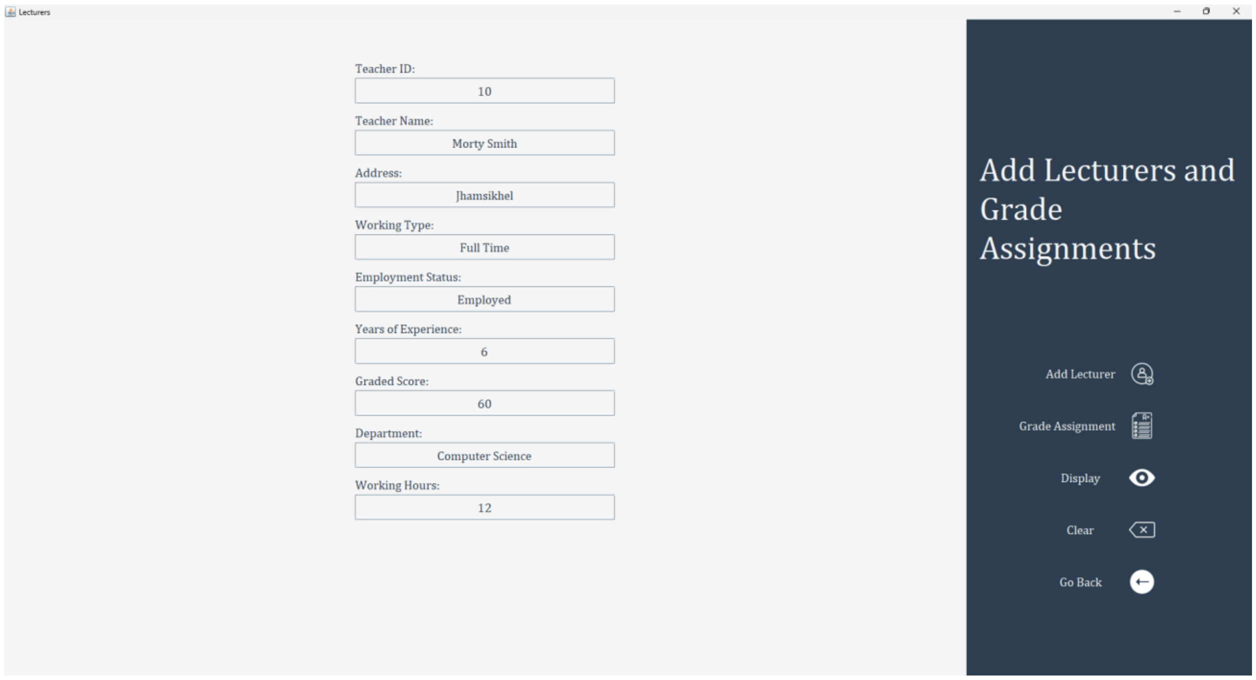
Figure 12 - Screenshot of displaying lecturer

Displaying Tutor information.

5.2.3 Test 3 – Grade Assignment

Test No.	2.3
Objective:	Grading assignments
Action:	Adding another lecturer and grading assignment Entering the values for the fields: • Teacher ID – 10 • Teacher Name – “Morty Smith” • Address – “Jhamsikhel” • Working Type – “Full Time” • Employment Status – “Employed” • Years of Experience – 6 • Graded Score – 60 • Department – “Computer Science” • Working Hours – 12 Entering these values in the grade assignment frame: • Teacher ID – 10 • Graded Score – 65 • Department – Computer Science • Years of Experience - 12
Expected Results:	The Lecturer should be added and grade should be displayed
Actual Results:	The Lecturer is added and grade is displayed
Conclusion:	The test was successful.

Results:


Lecturers

Teacher ID: 10

Teacher Name: Morty Smith

Address: Jhamsikhel

Working Type: Full Time

Employment Status: Employed

Years of Experience: 6

Graded Score: 60

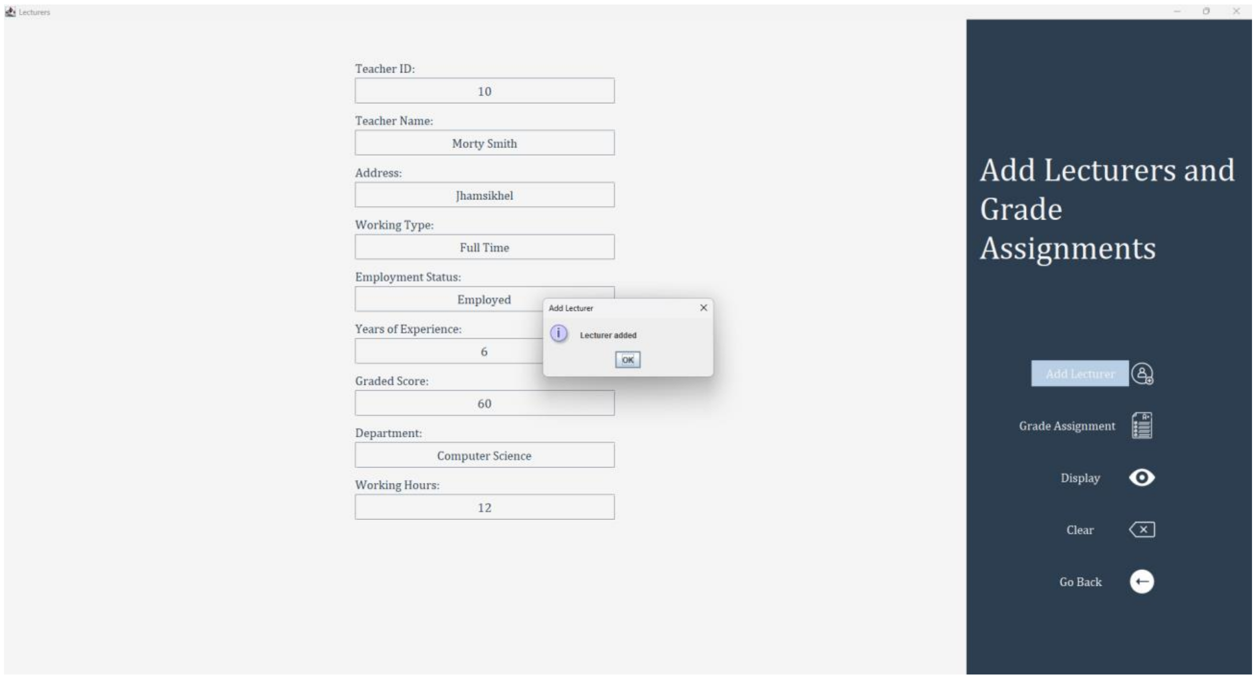
Department: Computer Science

Working Hours: 12

Add Lecturers and Grade Assignments

Figure 13 - Screenshot of entering values

Entering values in the fields to add a lecturer


Lecturers

Teacher ID: 10

Teacher Name: Morty Smith

Address: Jhamsikhel

Working Type: Full Time

Employment Status: Employed

Years of Experience: 6

Graded Score: 60

Department: Computer Science

Working Hours: 12

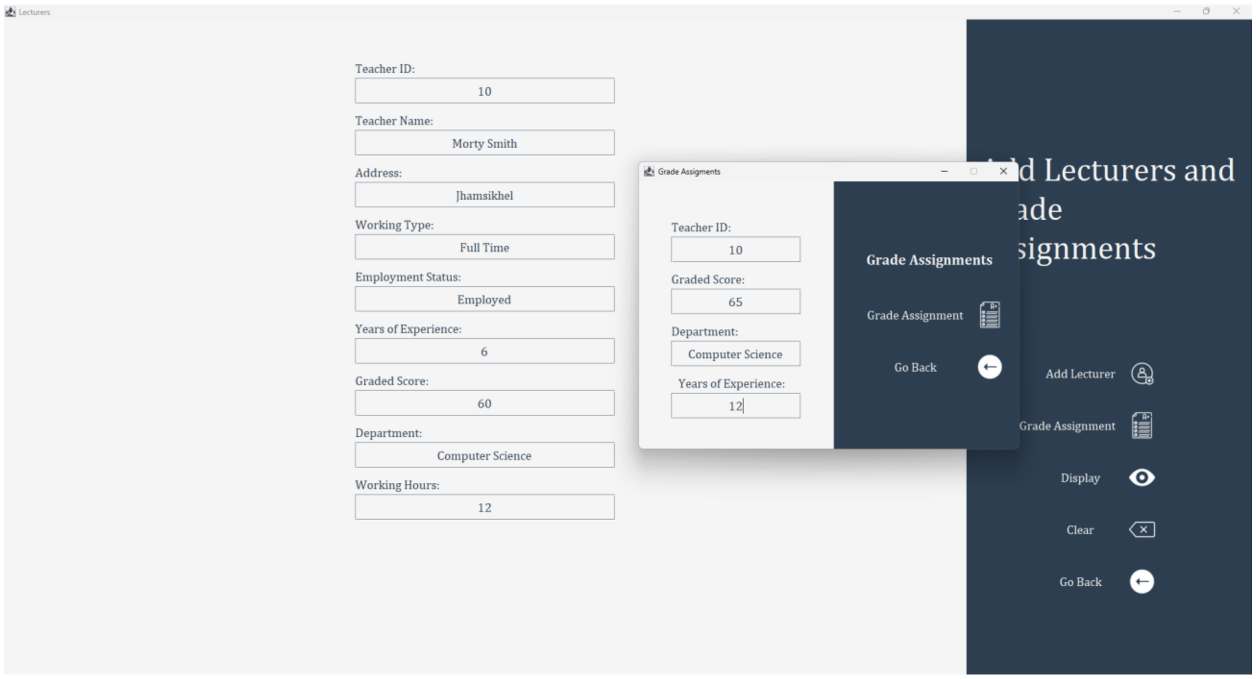
Add Lecturer

Lecturer added

Add Lecturers and Grade Assignments

Figure 14 - Screenshot of adding lecturer

Lecturer Added

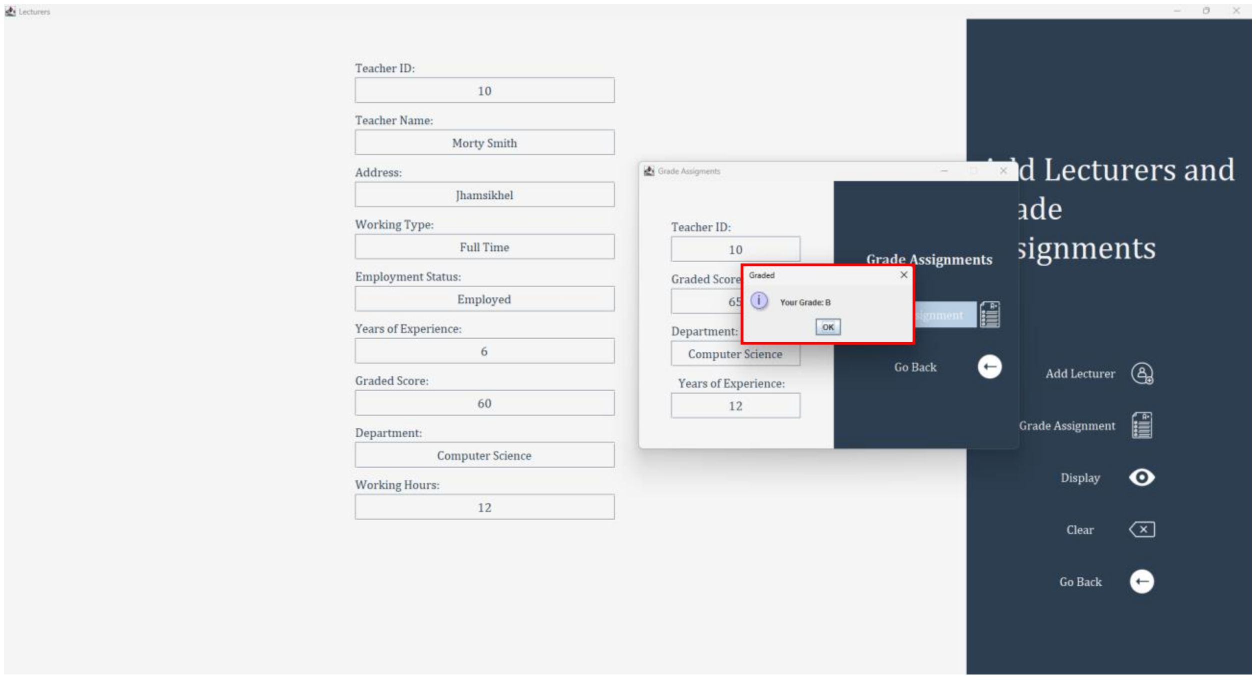


Teacher ID: 10
Teacher Name: Morty Smith
Address: Jhamsikhel
Working Type: Full Time
Employment Status: Employed
Years of Experience: 6
Graded Score: 60
Department: Computer Science
Working Hours: 12

Teacher ID: 10
Graded Score: 65
Department: Computer Science
Years of Experience: 12

Figure 15 - Screenshot of entering values to grade assignment

Entering values to grade assignment, Teacher ID, and Department should be same to be able to grade assignments.



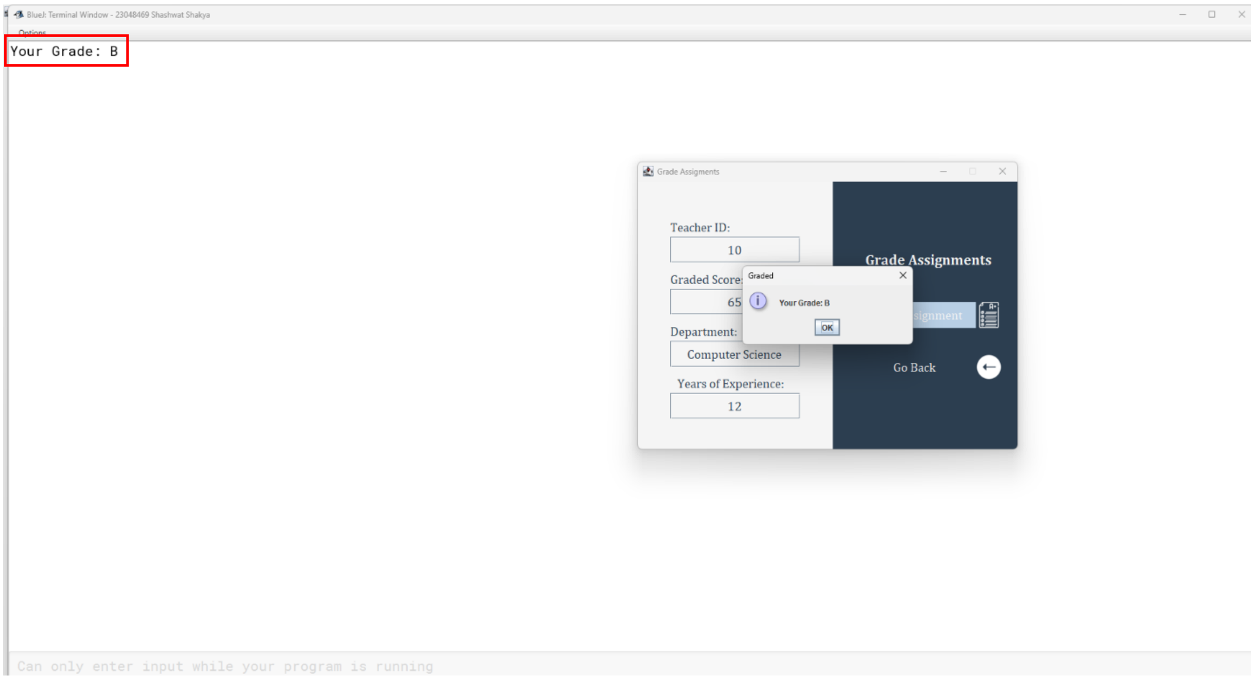
Teacher ID: 10
Teacher Name: Morty Smith
Address: Jhamsikhel
Working Type: Full Time
Employment Status: Employed
Years of Experience: 6
Graded Score: 60
Department: Computer Science
Working Hours: 12

Teacher ID: 10
Graded Score: 65
Department: Computer Science
Years of Experience: 12

Your Grade: B

Figure 16 - Screenshot of grade displayed

Grade is displayed in a JOptionPane



```
Your Grade: B
```

Teacher ID: 10

Graded Score: 65

Department: Computer Science

Years of Experience: 12

Your Grade: B

BlueJ: Terminal Window - 23048469 Shashwat Shakya
Options
Grade Assigments
Grade Assignments
Graded
OK
Go Back
Can only enter input while your program is running

Figure 17 - Screenshot of grade displayed in the terminal and JOptionPane

Grade is displayed in the terminal and in the JOptionPane.

5.2.4 Test 4 – Set Salary

Test No.	2.2
Objective:	Adding a Tutor using proper values
Action:	Entering the values for the fields: • Teacher ID – 15 • Teacher Name – “Rick Sanchez” • Address – “Lainchaur” • Working Type – “Full Time” • Employment Status – “Employed” • Working Hours – 23 • Salary – 40000 • Specialization – “Cyber Security” • Academic Qualification – “Masters” • Performance Index – 9 Entering the values for the fields in the set salary frame: • Teacher ID – 15 • Salary – 50000 • Performance Index - 10
Expected Results:	The Tutor should be added and salary should increase according to the performance index
Actual Results:	The Tutor is added and salary is increased according to the performance index
Conclusion:	The test was successful.

Results:

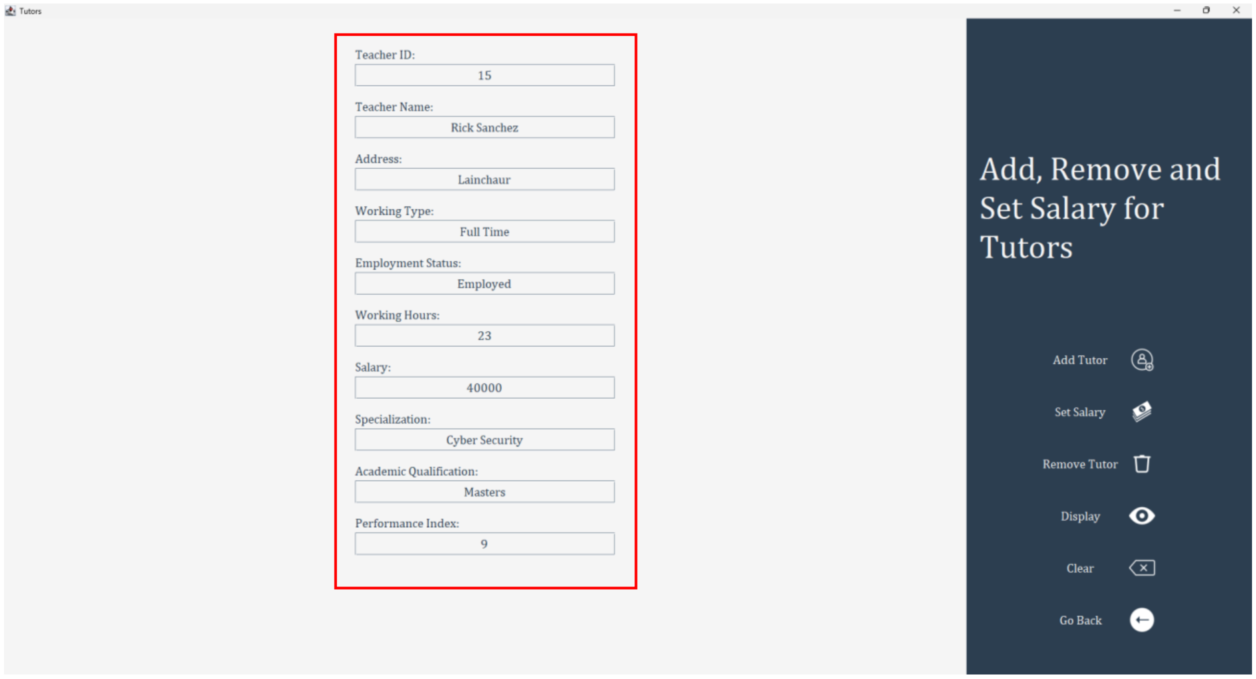
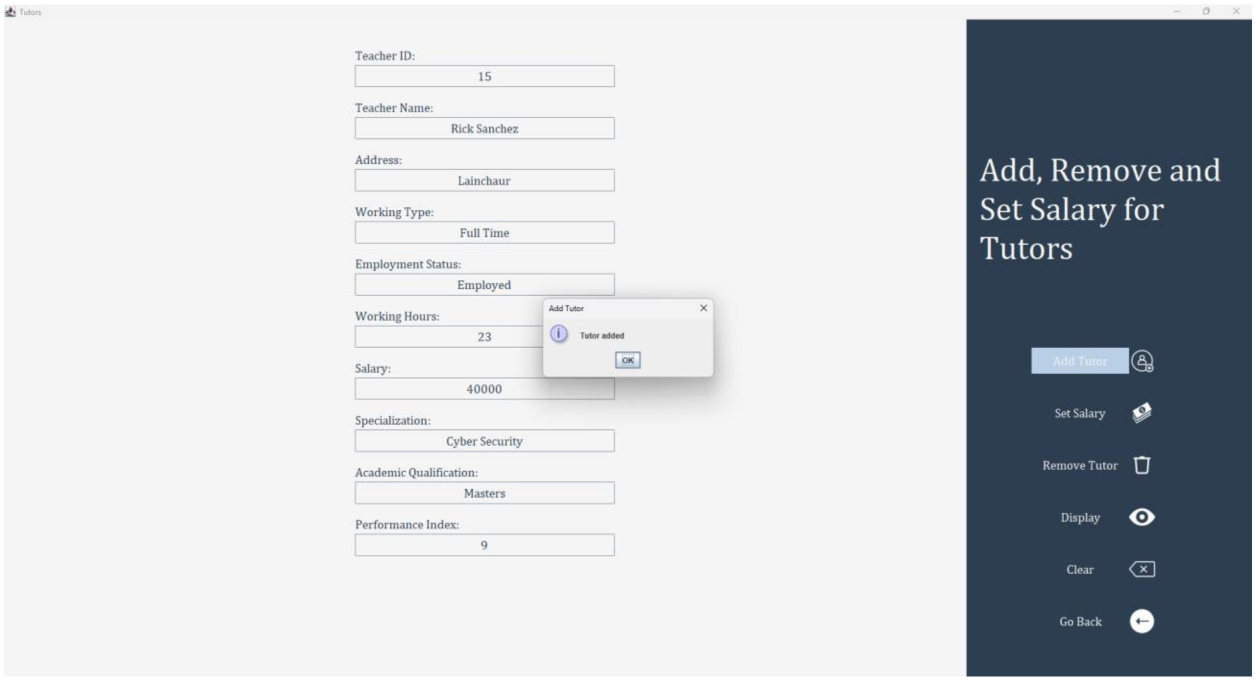

Teacher ID: 15
Teacher Name: Rick Sanchez
Address: Lainchaur
Working Type: Full Time
Employment Status: Employed
Working Hours: 23
Salary: 40000
Specialization: Cyber Security
Academic Qualification: Masters
Performance Index: 9

Figure 18 - Screenshot of entering values

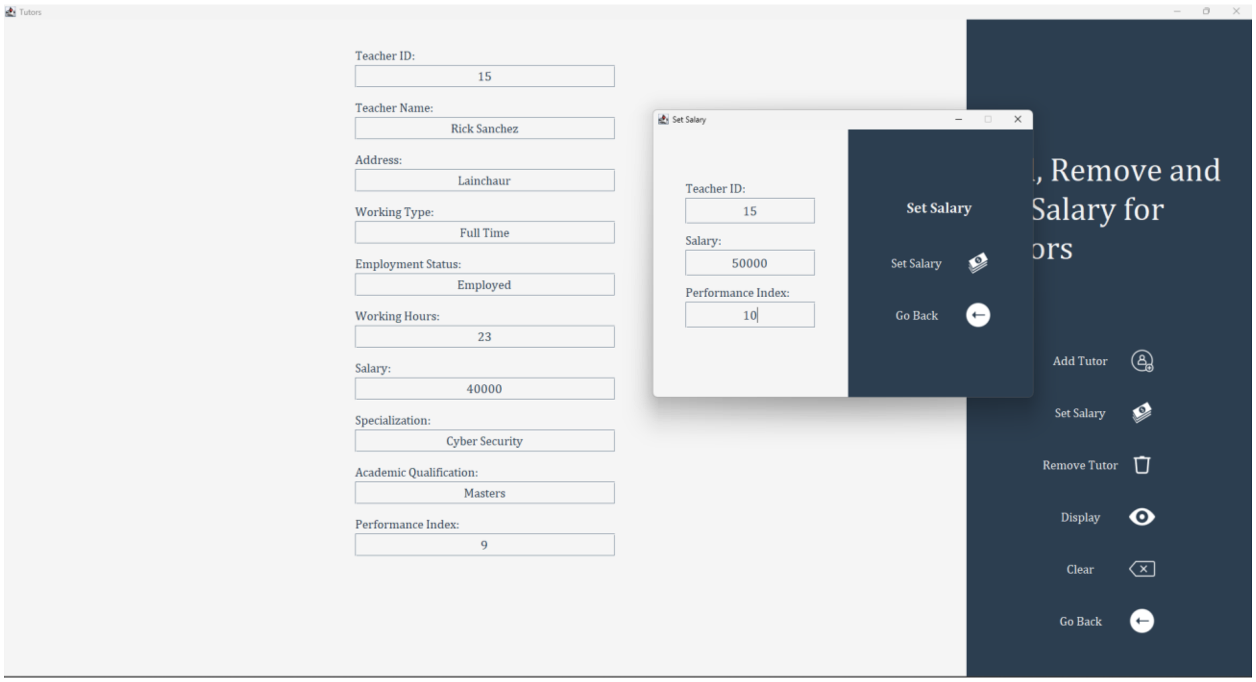
Entering the values to add another Tutor


Teacher ID: 15
Teacher Name: Rick Sanchez
Address: Lainchaur
Working Type: Full Time
Employment Status: Employed
Working Hours: 23
Salary: 40000
Specialization: Cyber Security
Academic Qualification: Masters
Performance Index: 9

Add Tutor
Tutor added

Figure 19 - Screenshot of adding tutor

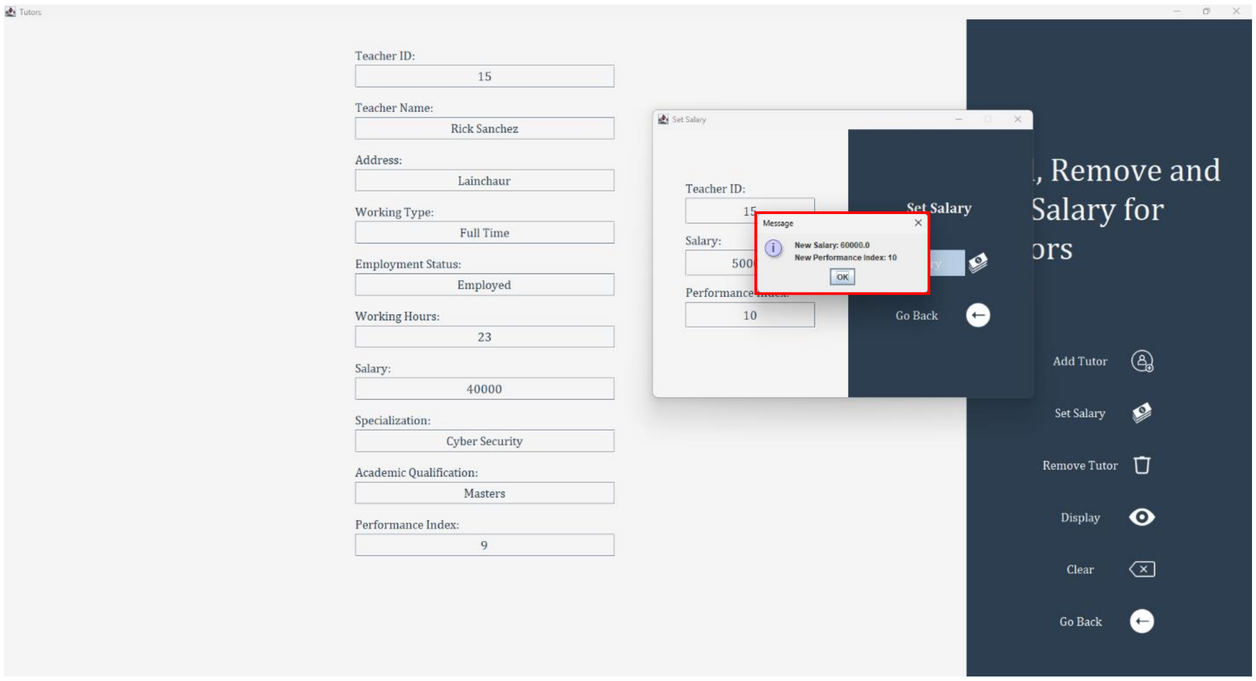
Tutor added


Teacher ID: 15
Teacher Name: Rick Sanchez
Address: Lainchaur
Working Type: Full Time
Employment Status: Employed
Working Hours: 23
Salary: 40000
Specialization: Cyber Security
Academic Qualification: Masters
Performance Index: 9

Set Salary
Teacher ID: 15
Salary: 50000
Performance Index: 10

Figure 20 - Screenshot of entering values to set salary

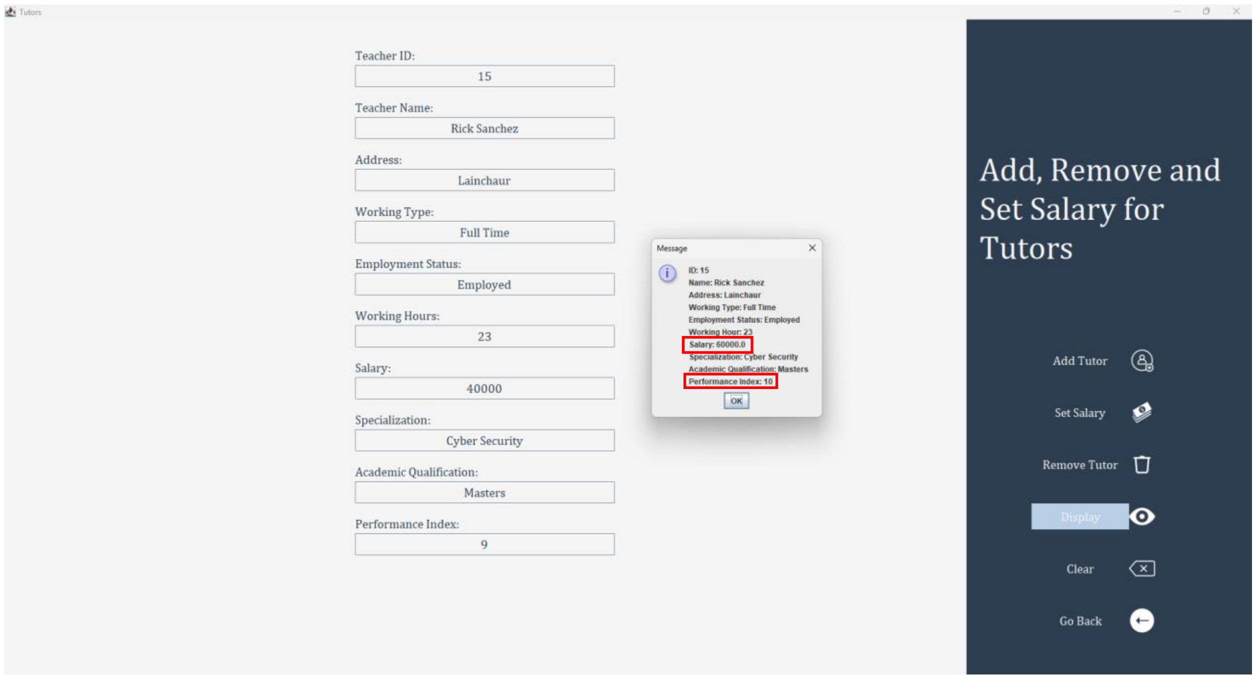
Entering the values to set new salary, Teacher ID should be same


Teacher ID: 15
Teacher Name: Rick Sanchez
Address: Lainchaur
Working Type: Full Time
Employment Status: Employed
Working Hours: 23
Salary: 40000
Specialization: Cyber Security
Academic Qualification: Masters
Performance Index: 9

Message
New Salary: 60000.0
New Performance Index: 10

Figure 21 - Screenshot of new salary and performance index being displayed

New salary and new performance index are displayed in a JOptionPane



Teacher ID: 15

Teacher Name: Rick Sanchez

Address: Lainchaur

Working Type: Full Time

Employment Status: Employed

Working Hours: 23

Salary: 40000

Specialization: Cyber Security

Academic Qualification: Masters

Performance Index: 9

```
ID: 15
Name: Rick Sanchez
Address: Lainchaur
Working Type: Full Time
Employment Status: Employed
Working Hour: 23
Salary: 60000.0
Specialization: Cyber Security
Academic Qualification: Masters
Performance Index: 10
```

Figure 22 - Screenshot of details of tutor

The salary and performance index are updated in the ArrayList as well.

5.2.5 Test 5 – Removing Tutor

Test No.	2.5
Objective:	Removing a tutor
Action:	Entering the values for the fields: • Teacher ID – 2
Expected Results:	The Tutor should be removed
Actual Results:	The Tutor is removed
Conclusion:	The test was successful.

<u>Results:</u>

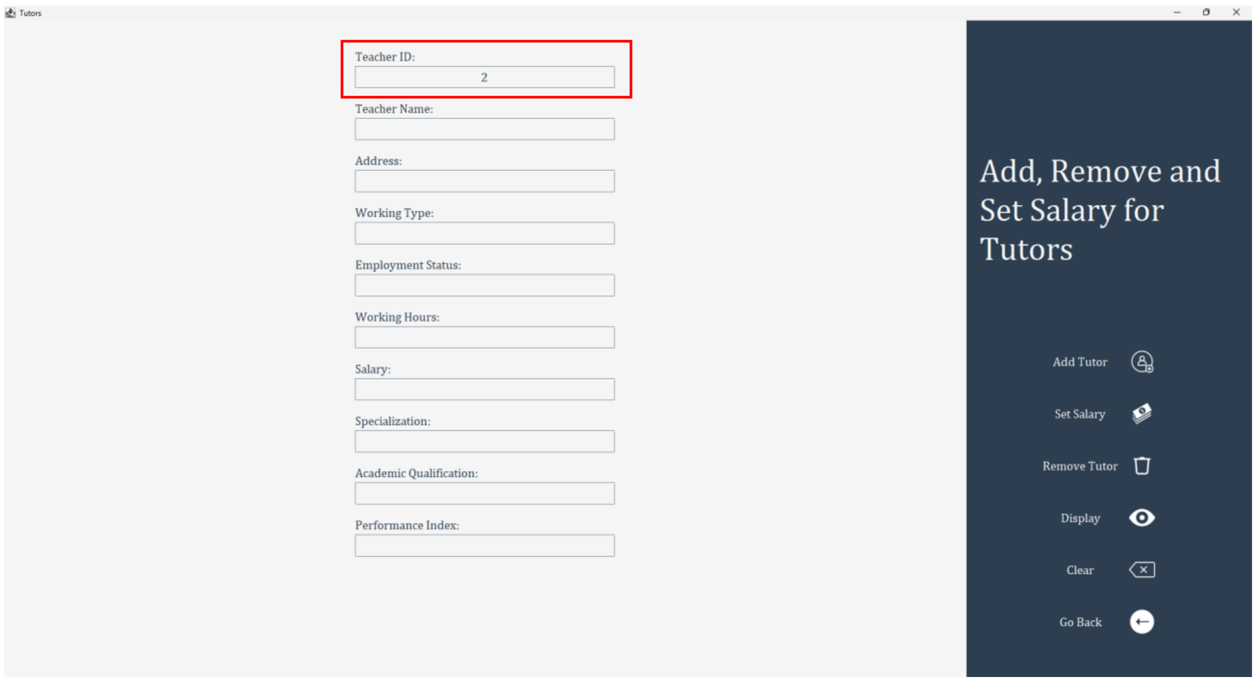


Figure 23 - Screenshot of entering value

Entering the teacher ID to remove

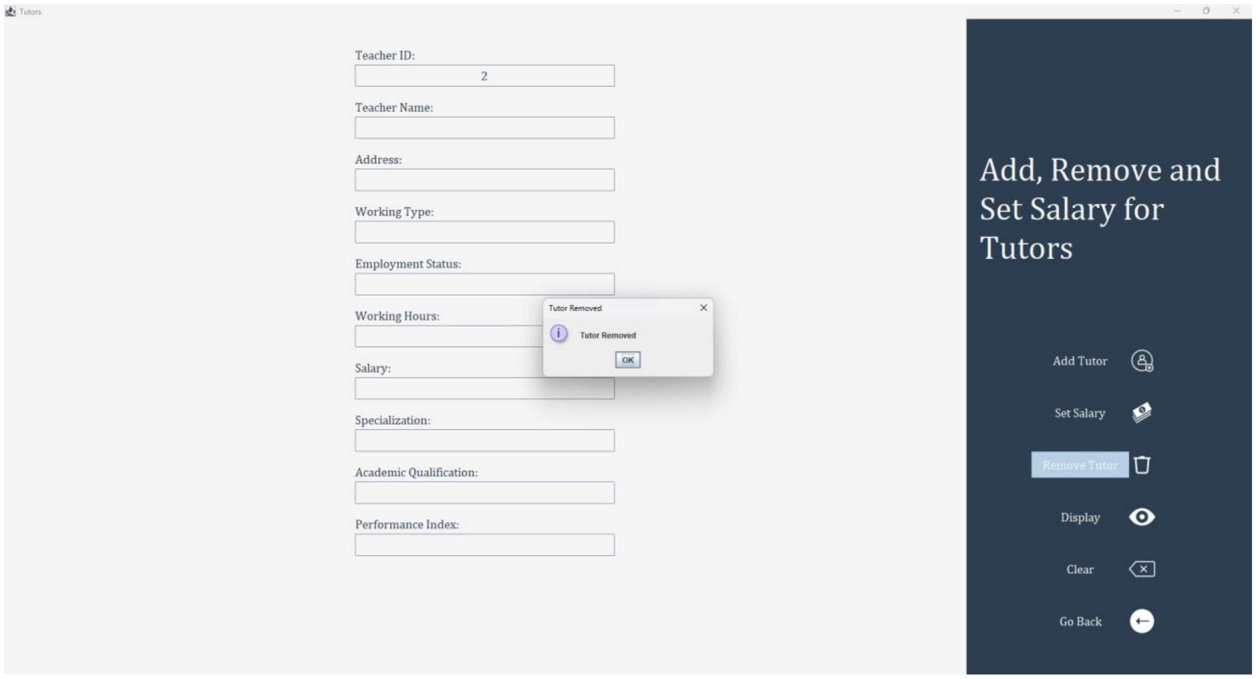


Figure 24 - Screenshot of removing tutor

Tutor Removed

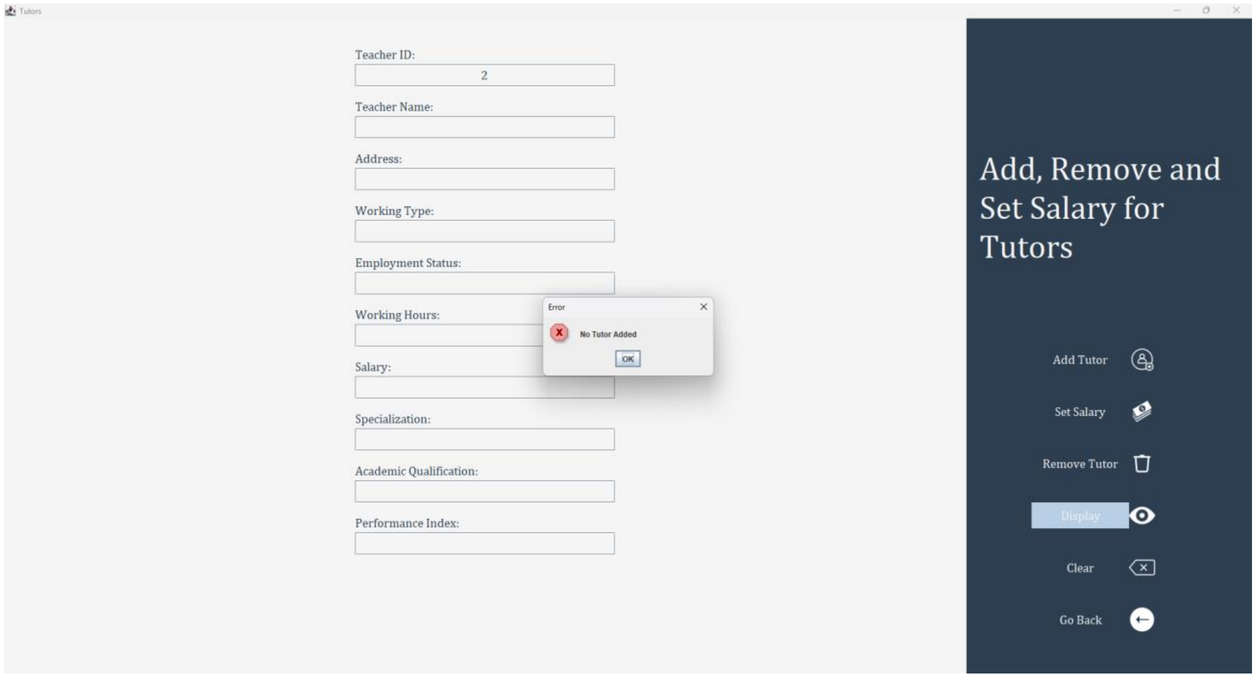
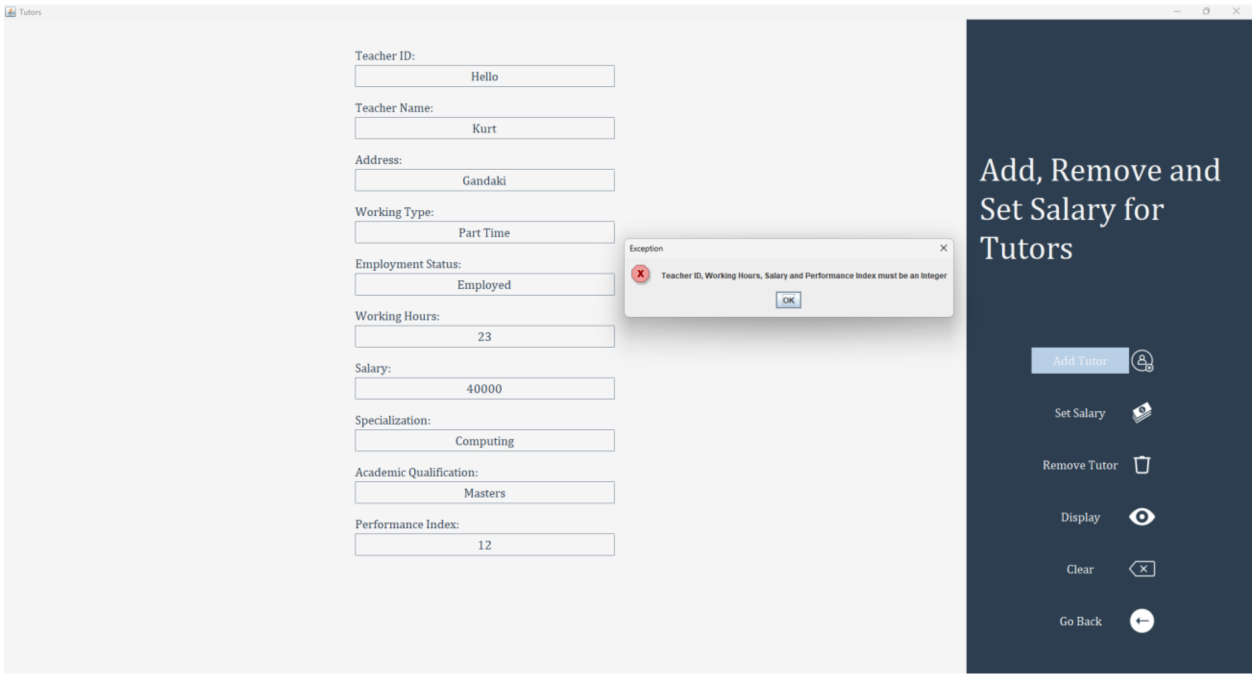


Figure 25 - Screenshot of no tutor shown

Checking if it is removed or not

5.3 Test 3 – Error when inappropriate values are entered.

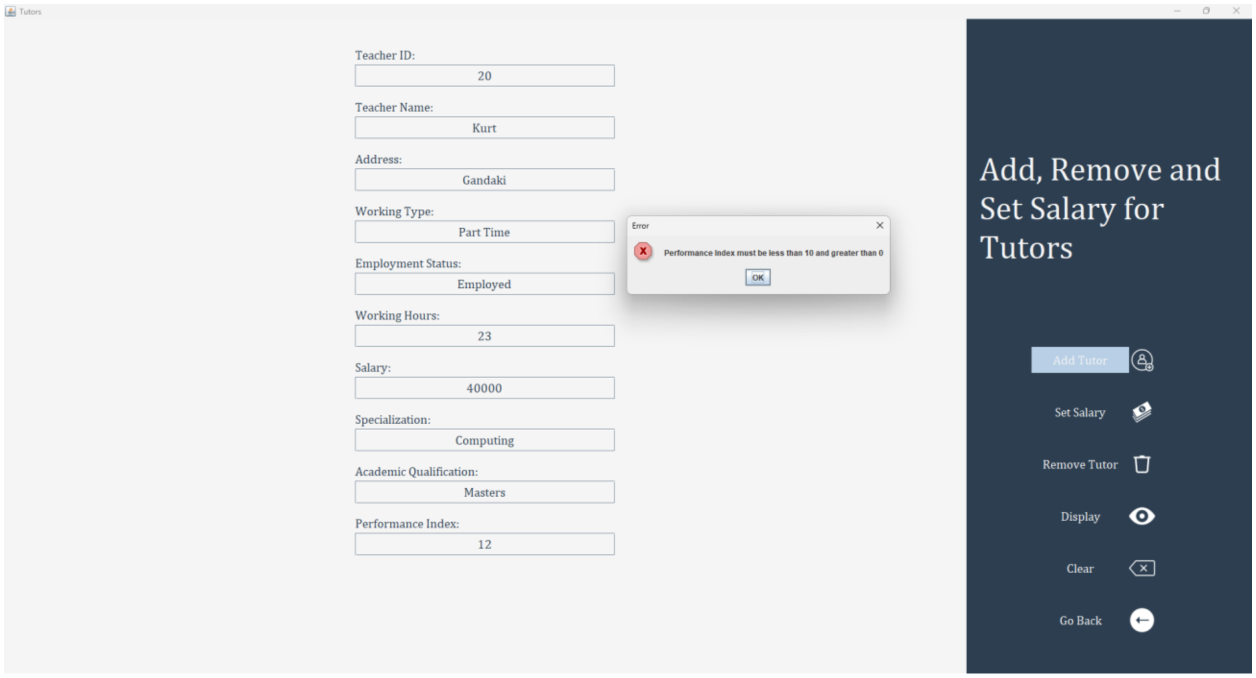
Test No.	3
Objective:	Testing whether appropriate dialog boxes are displayed when unsuitable values are entered.
Action:	Entering these values in the text fields: • Teacher ID – “Hello” • Teacher Name – “Kurt” • Address – “Gandaki” • Working Type – “Part Time” • Employment Status – “Employed” • Working Hours – 23 • Salary – 40000 • Specialization – Computing • Academic Qualification – “Masters” • Performance Index - 12
Expected Results:	Dialog box should appear stating that there is an error
Actual Results:	Dialog box should appeared stating that there is an error
Conclusion:	The test was successful.


Teacher ID: Hello
Teacher Name: Kurt
Address: Gandaki
Working Type: Part Time
Employment Status: Employed
Working Hours: 23
Salary: 40000
Specialization: Computing
Academic Qualification: Masters
Performance Index: 12

Exception

Teacher ID, Working Hours, Salary and Performance Index must be an Integer

When entering an invalid value in Teacher ID and performance index, a dialog box appears. After changing the teacher ID:


Teacher ID: 20
Teacher Name: Kurt
Address: Gandaki
Working Type: Part Time
Employment Status: Employed
Working Hours: 23
Salary: 40000
Specialization: Computing
Academic Qualification: Masters
Performance Index: 12

Error

Performance Index must be less than 10 and greater than 0

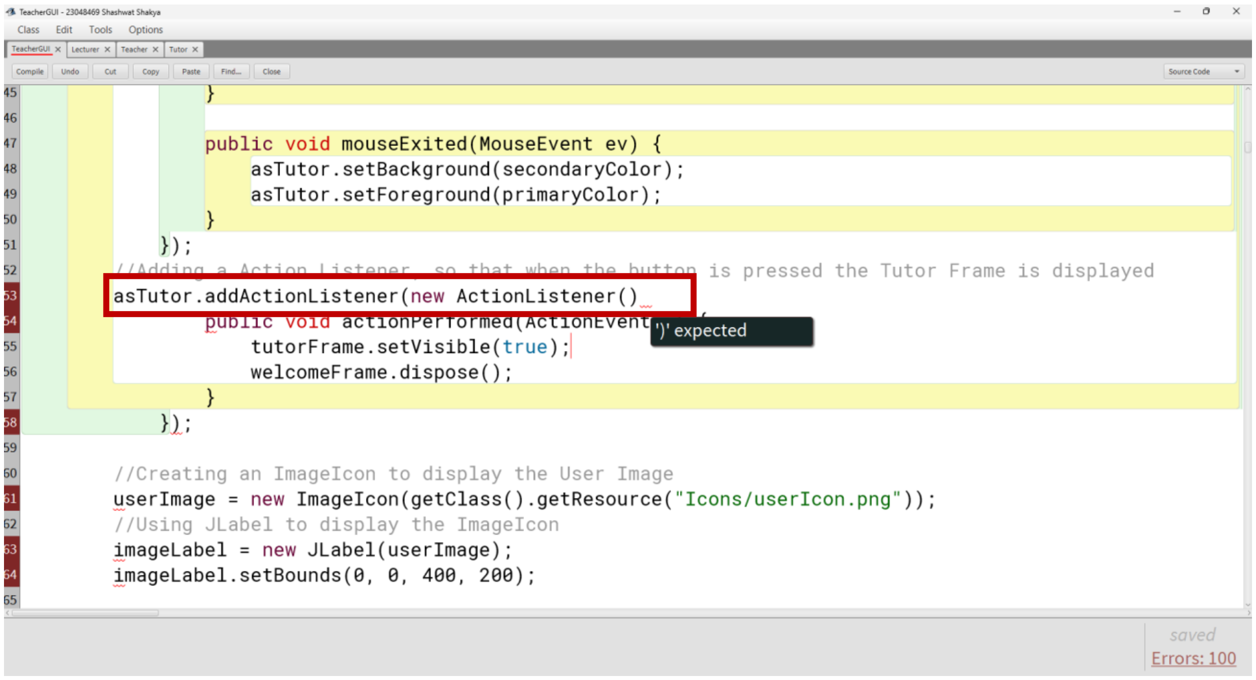
A dialog box is popped up stating performance index must be between 0 and 10

6. Error Detection and Error Correction

As a newbie I made a lot of errors whether it was in code or documentation. Errors are made by everyone especially in the beginning of something new. Here are some errors that I made and corrected along the way of the coursework:

6.1 Syntax Error

The syntax error may be due to a missing or misplaced character or punctuation in the code. Some common syntax errors include missing semicolons, parentheses, or curly braces, as well as typos in variable names or keywords. These errors prevent the code from compiling or running correctly.


TeacherGUI - 23048469 Shashwat Shakya
Class Edit Tools Options
TeacherGUI Lecturer Teacher Tutor
Compile Undo Cut Copy Paste Find... Close
Source Code
saved
Errors: 100

```
                }

                public void mouseExited(MouseEvent ev) {
                    asTutor.setBackground(secondaryColor);
                    asTutor.setForeground(primaryColor);
                }
            });
        //Adding a Action Listener, so that when the button is pressed the Tutor Frame is displayed
        asTutor.addActionListener(new ActionListener()
                public void actionPerformed(ActionEvent
                    tutorFrame.setVisible(true);
                    welcomeFrame.dispose();
                }
            });

        //Creating an ImageIcon to display the User Image
        userImage = new ImageIcon(getClass().getResource("Icons/userIcon.png"));
        //Using JLabel to display the ImageIcon
        imageLabel = new JLabel(userImage);
        imageLabel.setBounds(0, 0, 400, 200);
```

')' expected

Figure 26 - Screenshot of syntax error

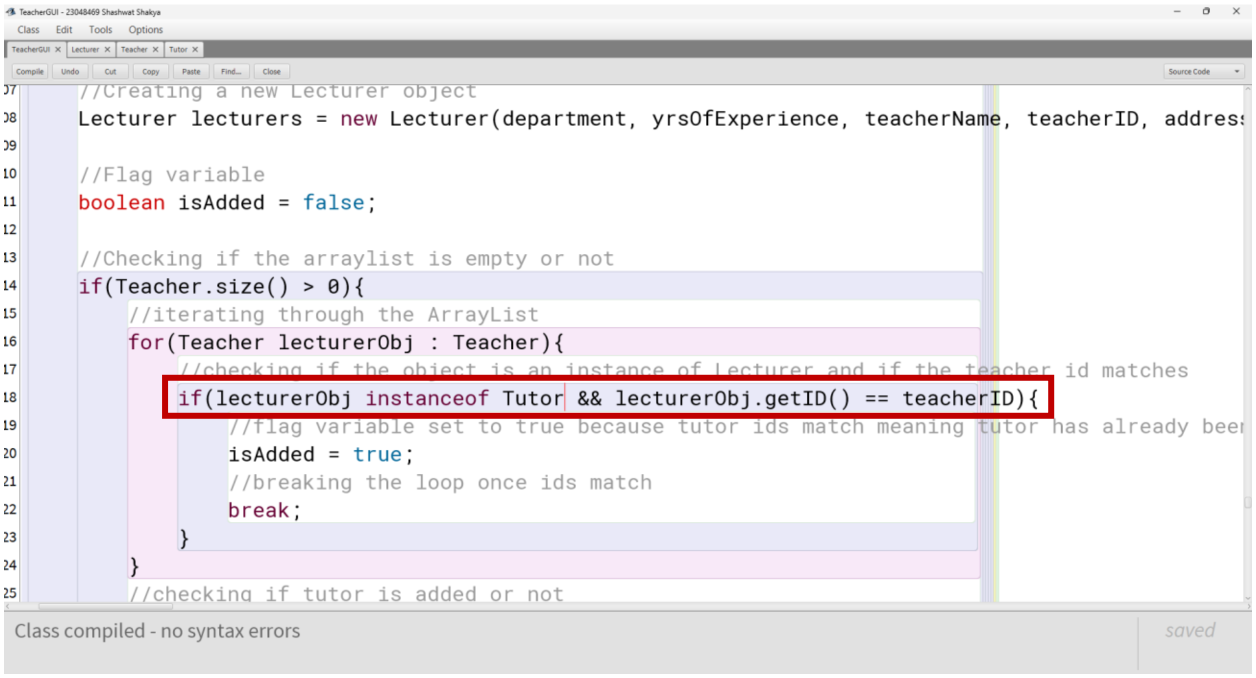
This is an example of syntax error in which it was missing a curly bracket due to this missing character the code is invalid and doesn't compile.

Correction:

To correct this error simply add a curly bracket where it was missing, and the error was fixed.

6.2 Semantic Error

Semantic Error occurs when the expected results are different or incorrect, but the code gets compiled and runs normally.



```java
//Creating a new Lecturer object
Lecturer lecturers = new Lecturer(department, yrsOfExperience, teacherName, teacherID, addres

//Flag variable
boolean isAdded = false;

//Checking if the arraylist is empty or not
if(Teacher.size() > 0){
    //iterating through the ArrayList
    for(Teacher lecturerObj : Teacher){
        //checking if the object is an instance of Lecturer and if the teacher id matches
        if(lecturerObj instanceof Tutor && lecturerObj.getID() == teacherID){
            //flag variable set to true because tutor ids match meaning tutor has already bee
            isAdded = true;
            //breaking the loop once ids match
            break;
        }
    }
    //checking if tutor is added or not
```

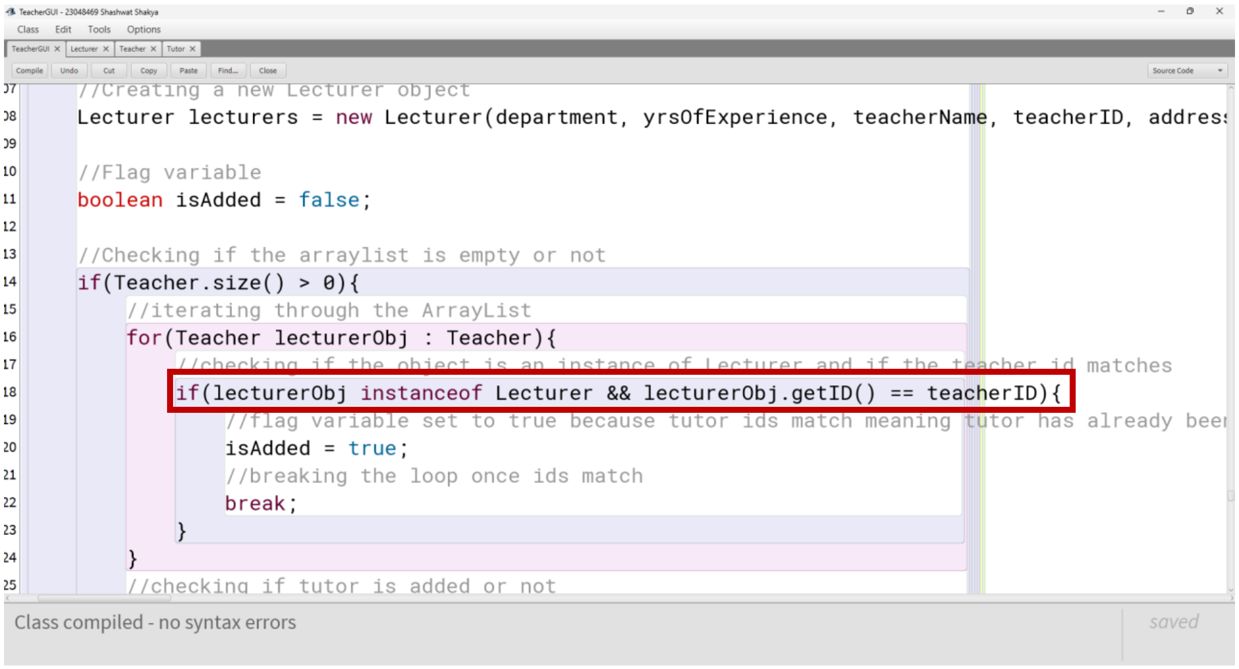
```
Class compiled - no syntax errors                                   saved
```

Figure 27 - Screenshot of semantic error

This is an example of semantic error, in the figure above the teacher object named lecturerObj is checked if it is an instance of Tutor but to add a lecturer we need to check if it is an instance of Lecturer instead. The code compiles but does not give the expected results.

Correction:

To correct this error I changed the Tutor from Lecturer so that it ensures that there are no multiple lecturers with the same ID.



TeacherGUI - 23048469 Shashwat Shakya
Class Edit Tools Options
TeacherGUI Lecturer Teacher Tutor
Compile Undo Cut Copy Paste Find... Close
Source Code

```
//Creating a new Lecturer object
Lecturer lecturers = new Lecturer(department, yrsOfExperience, teacherName, teacherID, addres

//Flag variable
boolean isAdded = false;

//Checking if the arraylist is empty or not
if(Teacher.size() > 0){
    //iterating through the ArrayList
    for(Teacher lecturerObj : Teacher){
        //checking if the object is an instance of Lecturer and if the teacher id matches
        if(lecturerObj instanceof Lecturer && lecturerObj.getID() == teacherID){
            //flag variable set to true because tutor ids match meaning tutor has already bee
            isAdded = true;
            //breaking the loop once ids match
            break;
        }
    }
    //checking if tutor is added or not
```

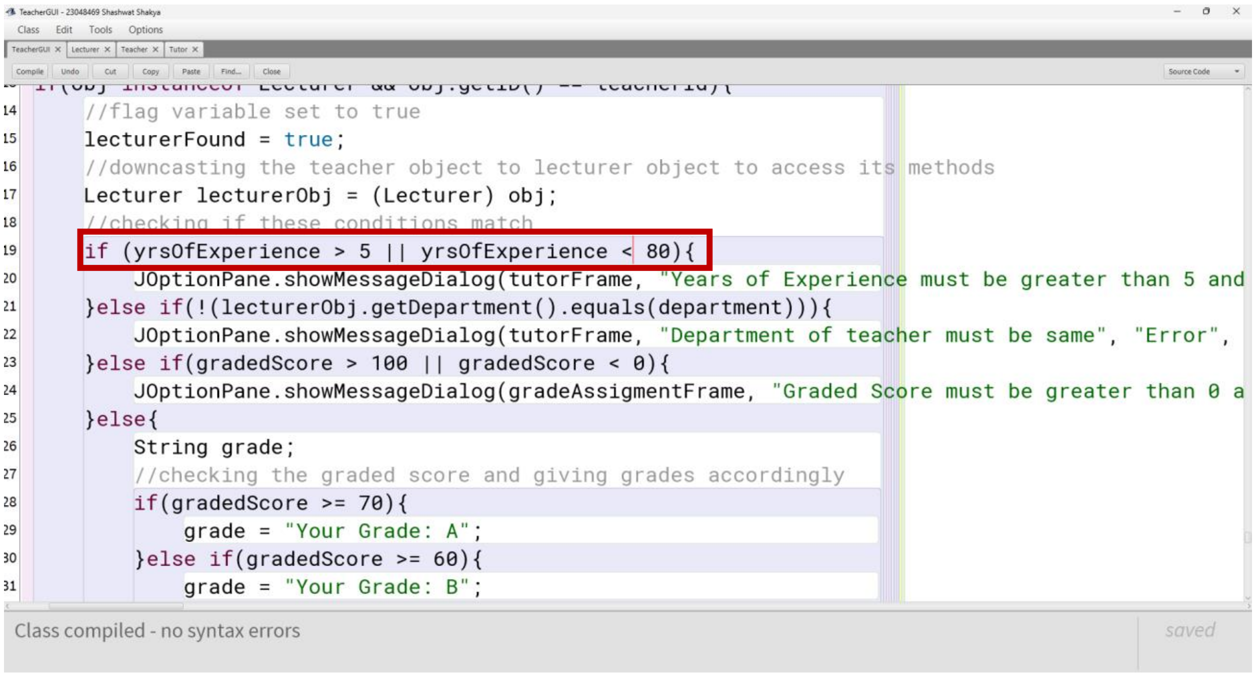
Class compiled - no syntax errors

saved

Figure 28 - Screenshot of correcting semantic error

6.3 Logical Error

Logical error occurs when the code compiles and runs without syntax errors but gives incorrect results due to flaw in the logic of the program.


TeacherGUI - 23048469 Shashwat Shakya
Class Edit Tools Options
TeacherGUI Lecturer Teacher Tutor
Compile Undo Cut Copy Paste Find... Close
Source Code

```
if(obj instanceof Lecturer && obj.getID() == teacherId){
    //flag variable set to true
    lecturerFound = true;
    //downcasting the teacher object to lecturer object to access its methods
    Lecturer lecturerObj = (Lecturer) obj;
    //checking if these conditions match
    if (yrsOfExperience > 5 || yrsOfExperience < 80){
        JOptionPane.showMessageDialog(tutorFrame, "Years of Experience must be greater than 5 and
    }else if(!(lecturerObj.getDepartment().equals(department))){
        JOptionPane.showMessageDialog(tutorFrame, "Department of teacher must be same", "Error",
    }else if(gradedScore > 100 || gradedScore < 0){
        JOptionPane.showMessageDialog(gradeAssigmentFrame, "Graded Score must be greater than 0 a
    }else{
        String grade;
        //checking the graded score and giving grades accordingly
        if(gradedScore >= 70){
            grade = "Your Grade: A";
        }else if(gradedScore >= 60){
            grade = "Your Grade: B";
```

Class compiled - no syntax errors

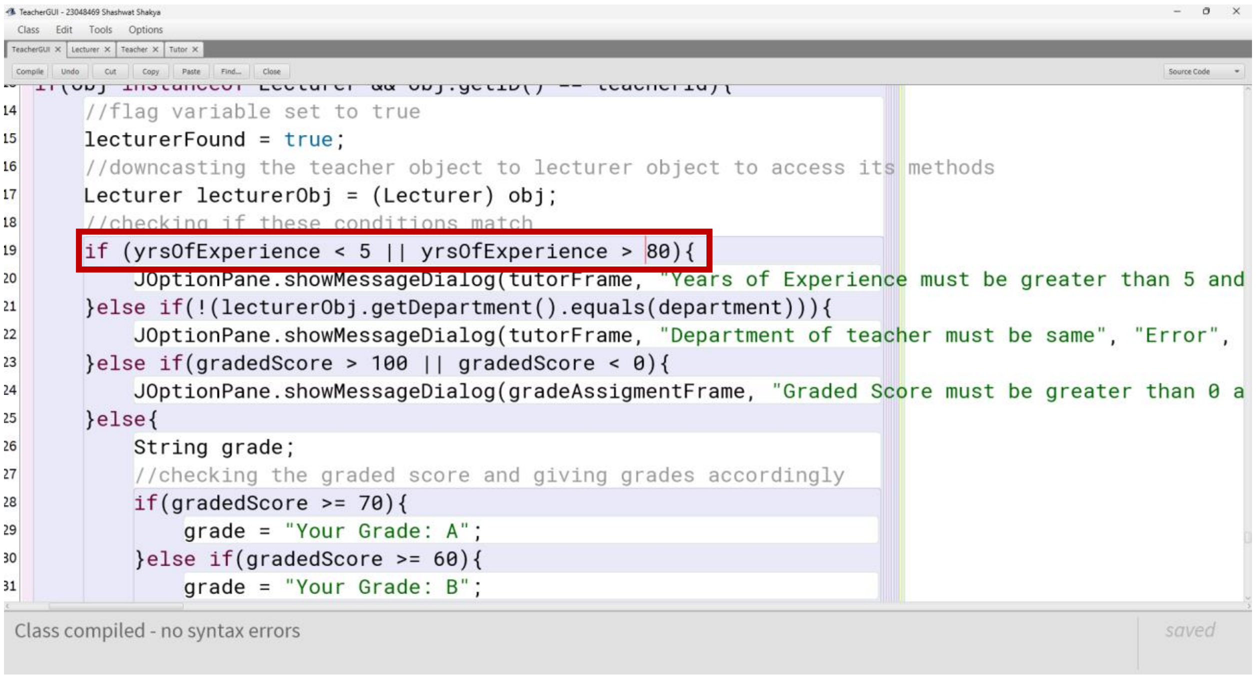
saved

Figure 29 - Screenshot of logical error

This is an example of logical error in which when years of experience is checked it should be less than 5 and greater than 80 to display an error message, but in the figure above the opposite is done which gives incorrect results.

Correction:

To correct this change the condition to years of experience less than 5 and greater than 80 display and error.



TeacherGUI - 23048469 Shashwat Shakya
Class Edit Tools Options
TeacherGUI Lecturer Teacher Tutor
Compile Undo Cut Copy Paste Find... Close
Source Code

```
        //flag variable set to true
        lecturerFound = true;
        //downcasting the teacher object to lecturer object to access its methods
        Lecturer lecturerObj = (Lecturer) obj;
        //checking if these conditions match
        if (yrsOfExperience < 5 || yrsOfExperience > 80){
            JOptionPane.showMessageDialog(tutorFrame, "Years of Experience must be greater than 5 and
        }else if(!(lecturerObj.getDepartment().equals(department))){
            JOptionPane.showMessageDialog(tutorFrame, "Department of teacher must be same", "Error",
        }else if(gradedScore > 100 || gradedScore < 0){
            JOptionPane.showMessageDialog(gradeAssigmentFrame, "Graded Score must be greater than 0 a
        }else{
            String grade;
            //checking the graded score and giving grades accordingly
            if(gradedScore >= 70){
                grade = "Your Grade: A";
            }else if(gradedScore >= 60){
                grade = "Your Grade: B";
```

Class compiled - no syntax errors

Figure 30 - Screenshot of correction of logical error

Conclusion

In conclusion this coursework has been helpful for the progress of my Java programming. This was our second coursework for the programming module, the implementation of Object-Oriented Programming and Java GUI used in this coursework has significantly enhanced my understanding and proficiency in Java. I have gained valuable insights into structuring and organizing code effectively.

Tackling this coursework has allowed me to encounter various challenges and problem-solving scenarios, thereby honing my critical thinking and debugging abilities. Type casting was a new concept that I had not learnt before, implementing it in the program was a confusing task but through the help of my tutors and friends I overcame the obstacle. The iterative process of designing, implementing, and testing code has fostered my mindset of continuous improvement and adaptability, essential traits for any aspiring programmer.

In conclusion, this coursework has been instrumental in my journey towards becoming proficient in Java programming, and I am grateful for the invaluable learning experience out tutors and lecturers have given us. They have helped me overcome the challenges and obstacles that have come along the way of this coursework.

References

BlueJ, 1999. *BlueJ.* [Online]

Available at: https://www.bluej.org/about.html

[Accessed 7 May 2024].

Appendix

Code of TeacherGUI.java

```
import java.util.ArrayList;

import javax.swing.BorderFactory;
import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JTextField;
import javax.swing.border.Border;
import javax.swing.JOptionPane;
import javax.swing.JTextArea;

import java.awt.Dimension;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.MouseEvent;
import java.awt.event.MouseAdapter;
import java.awt.Cursor;
import java.awt.GridLayout;
import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.FlowLayout;
import java.awt.Font;
```

```
public class TeacherGUI implements ActionListener {

    private JFrame welcomeFrame, tutorFrame, lecturerFrame, gradeAssigmentFrame,
setSalaryFrame;

    private JPanel welFrameLeft, welFrameLeftContent, welFrameRight,
welFrameRightTop, welFrameRightBottom, mainTutor,tutorContentPanel, rightTutor,
rightTutorTop, rightTutorBottom, mainLecturer, mainLecturerContent,rightLecturerTop,
rightLecturerBottom, rightLecturer, mainGradeAssignment,
rightGradeAssignment,rightSalary, mainSalary;

    private JTextField teacherIdFieldT, teacherNameFieldT, addressFieldT,
workingTypeFieldT, employmentStatFieldT,teacherIdFieldL, teacherNameFieldL,
addressFieldL, workingTypeFieldL, employmentStatFieldL,workingHoursFieldT,
workingHoursFieldL, departmentField, yrsOfExperienceField,
gradedScoreField,salaryField, specializationField, academicQualificationField,
performanceIndexField, teacherIdGradeField,gradedScoreGradeField,
departmentGradeField, yrsOfExperienceGradeField,
teacherIdSalaryField,newSalaryField, performanceIndexSalaryField;

    private JButton addLecturer, addTutor, gradeAssignment, salarySet, removeTutor,
displayT, clearT, displayL, clearL,asLecturer, asTutor, gradeAssignmentGradeButton,
setSalaryButton, goBackTutor, goBackLecturer, goBackSalaryButton,
goBackGradeButton;

    private JLabel mainImageLabel, imageLabel, welFrameHeading, logInHeader,
teacherIdLabelT, teacherNameLabelT,addressLabelT, workingTypeLabelT,
employmentStatLabelT, teacherIdLabelL, teacherNameLabelL,
addressLabelL,workingTypeLabelL, employmentStatLabelL, workingHoursLabelT,
workingHoursLabelL, departmentLabel,yrsOfExperienceLabel, gradedScoreLabel,
salaryLabel, specializationLabel, academicQualificationLabel,performanceIndexLabel,
headerGrade, teacherIdGradeLabel, gradedScoreGradeLabel,
departmentGradeLabel,yrsOfExperienceGradeLabel, salaryHeader,
teacherIdSalaryLabel, newSalaryLabel, performanceIndexSalaryLabel,addTutorImage,
salarySetImage, removeTutorImage, displayTutorImage, clearTutorImage,
addLecturerImage,gradeAssignmentImage, gradeAssignmentImageSmall,
displayLecturerImage, clearLecturerImage, setSalaryImage, goBackImageTutor,
goBackImageLecturer, goBackSalary, goBackGrade;

    private JTextArea welFrameCenter, headerT, headerL;

    ImageIcon userImage, mainImageIcon, tutorImageIcon, salarySetIcon,
removeTutorIcon, displayTutorIcon,clearTutorIcon, addLecturerIcon,
gradeAssignmentIcon, displayLecturerIcon, clearLecturerIcon, goBackIcon;
```

```
ArrayList<Teacher> Teacher = new ArrayList<Teacher>();

public TeacherGUI() {
    //Creating frames for the GUI
    //Creating Frame for the welcome page
    welcomeFrame = new JFrame("Welcome");

    //Creating Frame for the Tutor page
    tutorFrame = new JFrame("Tutors");

    //Creating Frame for the Lecturer page
    lecturerFrame = new JFrame("Lecturers");

    //Creating Frame for grading assignments
    gradeAssigmentFrame = new JFrame("Grade Assigments");

    //Creating Frame for setting salary
    setSalaryFrame = new JFrame("Set Salary");

    //Creating a Color object for text color and background
    Color secondaryColor = new Color(44, 62, 80);
    //Creating a Color onject for primary background color
    Color primaryColor = new Color(245, 245, 245);

    //Creating a Border object to remove borders
    Border emptyBorder = BorderFactory.createEmptyBorder();
```

```
        //Creating a Cursor object to change the cursor to pointer
        Cursor buttonCursor = new Cursor(Cursor.HAND_CURSOR);

        //Creating a Font object for main text in the GUI
        Font mainFont = new Font("Cambria", Font.PLAIN, 19);

        //Creating a Font object for headers
        Font headerFont = new Font("Cambria", Font.BOLD, 22);

        //Creating a Font object for the larger headers
        Font homePageHeader = new Font("Cambria", Font.PLAIN, 50);

        //Code for the welcome frame GUI
        //Creating an ImageIcon for the Image used in the welcome page
        mainImageIcon = new
ImageIcon(getClass().getResource("Icons/coverMain.png"));
        //Using JLabel to display the ImageIcon
        mainImageLabel = new JLabel(mainImageIcon);
        //setting bounds of the JLabel
        mainImageLabel.setBounds(0, 0, 802, 400);

        //Creating a JTextArea to display short description about the program
        welFrameCenter = new JTextArea("This is a advanced software in which you
easily add, update, and remove tutors and lecturers Simplify the grading process with
our intuitive assignment grading module. Tutors and lecturers can grade assignments
online, providing feedback and evaluations in real-time. The system automatically
calculates grades and generates reports for easy analysis.");
        welFrameCenter.setBounds(50, 550, 700, 150);
        welFrameCenter.setEditable(false);
        welFrameCenter.setLineWrap(true);
        welFrameCenter.setWrapStyleWord(true);
        welFrameCenter.setBackground(primaryColor);
        welFrameCenter.setForeground(secondaryColor);
        welFrameCenter.setFont(mainFont);

        //Creating a JLabel to display the Main header
        welFrameHeading = new JLabel("Welcome to the Home Page");
        welFrameHeading.setBounds(50, 400, 700, 200);
        welFrameHeading.setFont(homePageHeader);
        welFrameHeading.setForeground(secondaryColor);

        //Creating a JLabel to display log in message
        logInHeader = new JLabel("Log In As:");
        logInHeader.setBounds(100, 0, 400, 60);
        logInHeader.setFont(homePageHeader);
        logInHeader.setForeground(primaryColor);

        //Creating a JButton to go to the Lecturer Frame
        asLecturer = new JButton("Lecturer");
        asLecturer.setBounds(125, 150, 150, 40);
        asLecturer.setPreferredSize(new Dimension(150, 40));
        asLecturer.setBackground(secondaryColor);
        asLecturer.setForeground(primaryColor);
        asLecturer.setFocusable(false);
        asLecturer.setFont(mainFont);
        //Adding a Mouse Listener, so that when hovering over the button the color
changes
        asLecturer.addMouseListener(new MouseAdapter() {
            public void mouseEntered(MouseEvent ev) {
```

```
            asLecturer.setBackground(primaryColor);
            asLecturer.setForeground(secondaryColor);
        }

        public void mouseExited(MouseEvent ev) {
            asLecturer.setBackground(secondaryColor);
            asLecturer.setForeground(primaryColor);
        }
    });
    //Adding a Action Listener, so that when the button is pressed the Lecturer Frame
is displayed
    asLecturer.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            lecturerFrame.setVisible(true);
            welcomeFrame.dispose();
        }
    });

    //Creating a JButton to open Tutor Frame
    asTutor = new JButton("Tutor");
    asTutor.setBounds(125, 220, 150, 40);
    asTutor.setPreferredSize(new Dimension(150, 40));
    asTutor.setBackground(secondaryColor);
    asTutor.setForeground(primaryColor);
    asTutor.setFocusable(false);
    asTutor.setFont(mainFont);
    //Adding a Mouse Listener, so that when hovering over the button the color
changes
    asTutor.addMouseListener(new MouseAdapter() {
            public void mouseEntered(MouseEvent ev) {
               asTutor.setBackground(primaryColor);
               asTutor.setForeground(secondaryColor);
            }

            public void mouseExited(MouseEvent ev) {
               asTutor.setBackground(secondaryColor);
               asTutor.setForeground(primaryColor);
            }
         });
      //Adding a Action Listener, so that when the button is pressed the Tutor Frame is
displayed
      asTutor.addActionListener(new ActionListener(){
            public void actionPerformed(ActionEvent e) {
               tutorFrame.setVisible(true);
               welcomeFrame.dispose();
            }
         });

      //Creating an ImageIcon to display the User Image
      userImage = new ImageIcon(getClass().getResource("Icons/userIcon.png"));
      //Using JLabel to display the ImageIcon
      imageLabel = new JLabel(userImage);
      imageLabel.setBounds(0, 0, 400, 200);

      //JPanel for the Top right section of the GUI
      welFrameRightTop = new JPanel(new FlowLayout(FlowLayout.CENTER, 100,
190));
      welFrameRightTop.setBackground(secondaryColor);
      welFrameRightTop.add(imageLabel);

      //JPanel for the bottom right section of the GUI
      welFrameRightBottom = new JPanel(null);
      welFrameRightBottom.setBackground(secondaryColor);
      welFrameRightBottom.add(logInHeader);
      welFrameRightBottom.add(asLecturer);
      welFrameRightBottom.add(asTutor);

      //JPanel for the main content in the left section
      welFrameLeftContent = new JPanel();
      welFrameLeftContent.setLayout(null);
      welFrameLeftContent.setPreferredSize(new Dimension(800, 900));
      welFrameLeftContent.setBackground(primaryColor);
      welFrameLeftContent.add(mainImageLabel);
      welFrameLeftContent.add(welFrameHeading);
      welFrameLeftContent.add(welFrameCenter);

      //JPanel for the left section
      welFrameLeft = new JPanel();
      welFrameLeft.setLayout(new FlowLayout(FlowLayout.CENTER, 0, 0));
      welFrameLeft.setBackground(primaryColor);
      welFrameLeft.add(welFrameLeftContent);

      //JPanel for the right section
      welFrameRight = new JPanel();
      welFrameRight.setLayout(new GridLayout(2, 0));
      welFrameRight.setBackground(secondaryColor);
      welFrameRight.add(welFrameRightTop);
      welFrameRight.add(welFrameRightBottom);
      welFrameRight.setBounds(800, 0, 400, 900);

      //welcome Frame settingsL
      //setting the extendedState to MAXIMIZED to open the GUI in fullscreen
      welcomeFrame.setExtendedState(JFrame.MAXIMIZED_BOTH);
      welcomeFrame.setSize(1200, 900);
      welcomeFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      welcomeFrame.setVisible(true);
      welcomeFrame.setLayout(new BorderLayout());
      welcomeFrame.setResizable(true);
      welcomeFrame.add(welFrameLeft, BorderLayout.CENTER);
      welcomeFrame.add(welFrameRight, BorderLayout.EAST);

      // For Tutor Frame:
      //Header for the top right panel
      headerT = new JTextArea("Add, Remove and Set Salary for Tutors");
      headerT.setEditable(false);
      headerT.setLineWrap(true);
      headerT.setWrapStyleWord(true);
      headerT.setBackground(secondaryColor);
      headerT.setForeground(primaryColor);
      headerT.setFont(homePageHeader);
      headerT.setPreferredSize(new Dimension(400, 200));

      //JLabels for Tutor JFrame
      //JLabel for Teacher ID in Tutor Frame
```

```
teacherIdLabelT = new JLabel("Teacher ID:");
teacherIdLabelT.setBounds(200, 30, 100, 40);
teacherIdLabelT.setForeground(secondaryColor);
teacherIdLabelT.setFont(mainFont);

//JLabel for Teacher Name in Tutor Frame
teacherNameLabelT = new JLabel("Teacher Name: ");
teacherNameLabelT.setBounds(200, 110, 190, 40);
teacherNameLabelT.setForeground(secondaryColor);
teacherNameLabelT.setFont(mainFont);

//JLabel for Address in Tutor Frame
addressLabelT = new JLabel("Address:");
addressLabelT.setBounds(200, 190, 190, 40);
addressLabelT.setForeground(secondaryColor);
addressLabelT.setFont(mainFont);

//JLabel for Working Type in Tutor Frame
workingTypeLabelT = new JLabel("Working Type:");
workingTypeLabelT.setBounds(200, 270, 190, 40);
workingTypeLabelT.setForeground(secondaryColor);
workingTypeLabelT.setFont(mainFont);

//JLabel for Employment Status in Tutor Frame
employmentStatLabelT = new JLabel("Employment Status:");
employmentStatLabelT.setBounds(200, 350, 190, 40);
employmentStatLabelT.setForeground(secondaryColor);
employmentStatLabelT.setFont(mainFont);
```

```
//JLabel for Working Hours in Tutor Frame
workingHoursLabelT = new JLabel("Working Hours:");
workingHoursLabelT.setBounds(200, 430, 190, 40);
workingHoursLabelT.setForeground(secondaryColor);
workingHoursLabelT.setFont(mainFont);

//JLabel for Salary ID in Tutor Frame
salaryLabel = new JLabel("Salary:");
salaryLabel.setBounds(200, 510, 190, 40);
salaryLabel.setForeground(secondaryColor);
salaryLabel.setFont(mainFont);

//JLabel for Specialization ID in Tutor Frame
specializationLabel = new JLabel("Specialization:");
specializationLabel.setBounds(200, 590, 190, 40);
specializationLabel.setForeground(secondaryColor);
specializationLabel.setFont(mainFont);

//JLabel for Academic Qualification in Tutor Frame
academicQualificationLabel = new JLabel("Academic Qualification:");
academicQualificationLabel.setBounds(200, 670, 190, 40);
academicQualificationLabel.setForeground(secondaryColor);
academicQualificationLabel.setFont(mainFont);

//JLabel for Performance Index in Tutor Frame
performanceIndexLabel = new JLabel("Performance Index:");
performanceIndexLabel.setBounds(200, 750, 190, 40);
performanceIndexLabel.setForeground(secondaryColor);
performanceIndexLabel.setFont(mainFont);

//JTextFields for Tutor Frame
//JTextField for Teacher ID in Tutor Frame
teacherIdFieldT = new JTextField();
teacherIdFieldT.setBounds(200, 65, 400, 35);
teacherIdFieldT.setBackground(primaryColor);
teacherIdFieldT.setForeground(secondaryColor);
teacherIdFieldT.setHorizontalAlignment(JTextField.CENTER);
teacherIdFieldT.setFont(mainFont);

//JTextField for Teacher Name in Tutor Frame
teacherNameFieldT = new JTextField();
teacherNameFieldT.setBounds(200, 145, 400, 35);
teacherNameFieldT.setBackground(primaryColor);
teacherNameFieldT.setForeground(secondaryColor);
teacherNameFieldT.setHorizontalAlignment(JTextField.CENTER);
teacherNameFieldT.setFont(mainFont);

//JTextField for Address in Tutor Frame
addressFieldT = new JTextField();
addressFieldT.setBounds(200, 225, 400, 35);
addressFieldT.setBackground(primaryColor);
addressFieldT.setForeground(secondaryColor);
addressFieldT.setHorizontalAlignment(JTextField.CENTER);
addressFieldT.setFont(mainFont);
```

```
//JTextField for Working Type in Tutor Frame
workingTypeFieldT = new JTextField();
workingTypeFieldT.setBounds(200, 305, 400, 35);
workingTypeFieldT.setBackground(primaryColor);
workingTypeFieldT.setForeground(secondaryColor);
workingTypeFieldT.setHorizontalAlignment(JTextField.CENTER);
workingTypeFieldT.setFont(mainFont);

//JTextField for Employment Status in Tutor Frame
employmentStatFieldT = new JTextField();
employmentStatFieldT.setBounds(200, 385, 400, 35);
employmentStatFieldT.setBackground(primaryColor);
employmentStatFieldT.setForeground(secondaryColor);
employmentStatFieldT.setHorizontalAlignment(JTextField.CENTER);
employmentStatFieldT.setFont(mainFont);

//JTextField for Working Hours in Tutor Frame
workingHoursFieldT = new JTextField();
workingHoursFieldT.setBounds(200, 465, 400, 35);
workingHoursFieldT.setBackground(primaryColor);
workingHoursFieldT.setForeground(secondaryColor);
workingHoursFieldT.setHorizontalAlignment(JTextField.CENTER);
workingHoursFieldT.setFont(mainFont);

//JTextField for Salary in Tutor Frame
salaryField = new JTextField();
salaryField.setBounds(200, 545, 400, 35);
salaryField.setBackground(primaryColor);
salaryField.setForeground(secondaryColor);
salaryField.setHorizontalAlignment(JTextField.CENTER);
salaryField.setFont(mainFont);

//JTextField for Specialization in Tutor Frame
specializationField = new JTextField();
specializationField.setBounds(200, 625, 400, 35);
specializationField.setBackground(primaryColor);
specializationField.setForeground(secondaryColor);
specializationField.setHorizontalAlignment(JTextField.CENTER);
specializationField.setFont(mainFont);

//JTextField for Academic Qualification in Tutor Frame
academicQualificationField = new JTextField();
academicQualificationField.setBounds(200, 705, 400, 35);
academicQualificationField.setBackground(primaryColor);
academicQualificationField.setForeground(secondaryColor);
academicQualificationField.setHorizontalAlignment(JTextField.CENTER);
academicQualificationField.setFont(mainFont);

//JTextField for Performance Index in Tutor Frame
performanceIndexField = new JTextField();
performanceIndexField.setBounds(200, 785, 400, 35);
performanceIndexField.setBackground(primaryColor);
performanceIndexField.setForeground(secondaryColor);
performanceIndexField.setHorizontalAlignment(JTextField.CENTER);
performanceIndexField.setFont(mainFont);
```

```
        //ImageIcon for Add Tutor Button
        tutorImageIcon = new
ImageIcon(getClass().getResource("Icons/userIconMain.png"));
        addTutorImage = new JLabel(tutorImageIcon);
        addTutorImage.setBounds(250, 0, 40, 40);

        //ImageIcon for Set Salary Button
        salarySetIcon = new ImageIcon(getClass().getResource("Icons/salary.png"));
        salarySetImage = new JLabel(salarySetIcon);
        salarySetImage.setBounds(250, 80, 40, 40);

        //ImageIcon for Remove Tutor Button
        removeTutorIcon = new ImageIcon(getClass().getResource("Icons/remove.png"));
        removeTutorImage = new JLabel(removeTutorIcon);
        removeTutorImage.setBounds(250, 160, 40, 40);

        //ImageIcon for Display Button
        displayTutorIcon = new ImageIcon(getClass().getResource("Icons/display.png"));
        displayTutorImage = new JLabel(displayTutorIcon);
        displayTutorImage.setBounds(250, 240, 40, 40);

        //ImageIcon for Clear Button
        clearTutorIcon = new ImageIcon(getClass().getResource("Icons/clear.png"));
        clearTutorImage = new JLabel(clearTutorIcon);
        clearTutorImage.setBounds(250, 320, 40, 40);

        //ImageIcon for Go Back Button
        goBackIcon = new ImageIcon(getClass().getResource("Icons/goBack.png"));
        goBackImageTutor = new JLabel(goBackIcon);
```

```
goBackImageTutor.setBounds(250, 400, 40, 40);

//JButtons for Tutor Frame
//JButton for Add Tutor in Tutor Frame
addTutor = new JButton("Add Tutor");
addTutor.setBounds(100, 0, 150, 40);
addTutor.setFocusable(false);
addTutor.setBorder(emptyBorder);
addTutor.setCursor(buttonCursor);
addTutor.setBackground(secondaryColor);
addTutor.setForeground(primaryColor);
addTutor.setFont(mainFont);
addTutor.addActionListener(this);

//JButton for Set Salary in Tutor Frame
salarySet = new JButton("Set Salary");
salarySet.setBounds(100, 80, 150, 40);
salarySet.setFocusable(false);
salarySet.setBorder(emptyBorder);
salarySet.setCursor(buttonCursor);
salarySet.setBackground(secondaryColor);
salarySet.setForeground(primaryColor);
salarySet.setFont(mainFont);
salarySet.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        setSalaryFrame.setVisible(true);
    }
});
```

```
//JButton for Remove Tutor in Tutor Frame
removeTutor = new JButton("Remove Tutor");
removeTutor.setBounds(100, 160, 150, 40);
removeTutor.setFocusable(false);
removeTutor.setBorder(emptyBorder);
removeTutor.setCursor(buttonCursor);
removeTutor.setBackground(secondaryColor);
removeTutor.setForeground(primaryColor);
removeTutor.setFont(mainFont);
removeTutor.addActionListener(this);

//JButton for Display in Tutor Frame
displayT = new JButton("Display");
displayT.setBounds(100, 240, 150, 40);
displayT.setFocusable(false);
displayT.setBorder(emptyBorder);
displayT.setCursor(buttonCursor);
displayT.setBackground(secondaryColor);
displayT.setForeground(primaryColor);
displayT.setFont(mainFont);
displayT.addActionListener(this);

//JButton for Clear in Tutor Frame
clearT = new JButton("Clear");
clearT.setBounds(100, 320, 150, 40);
clearT.setFocusable(false);
clearT.setCursor(buttonCursor);
clearT.setBorder(emptyBorder);
clearT.addActionListener(this);
clearT.setBackground(secondaryColor);
clearT.setForeground(primaryColor);
clearT.setFont(mainFont);

//JButton for Go Back in Tutor Frame
goBackTutor = new JButton("Go Back");
goBackTutor.setBounds(100, 400, 150, 40);
goBackTutor.setFocusable(false);
goBackTutor.setCursor(buttonCursor);
goBackTutor.setBorder(emptyBorder);
goBackTutor.setBackground(secondaryColor);
goBackTutor.setForeground(primaryColor);
goBackTutor.setFont(mainFont);
goBackTutor.addActionListener(new ActionListener() {
   public void actionPerformed(ActionEvent e){
      tutorFrame.setVisible(false);
      welcomeFrame.setVisible(true);
   }
});

//JPanel for the main contents
tutorContentPanel = new JPanel(null);
tutorContentPanel.setBackground(primaryColor);
tutorContentPanel.setPreferredSize(new Dimension(800, 900));

//JPanel for the Top right section for Header
```

```
rightTutorTop = new JPanel(new FlowLayout(FlowLayout.CENTER, 20, 200));
rightTutorTop.setBackground(secondaryColor);

//JPanel for the Bottom right section for JButtons
rightTutorBottom = new JPanel(null);
rightTutorBottom.setBackground(secondaryColor);

//JPanel for the left section of the GUI
mainTutor = new JPanel(new FlowLayout());
mainTutor.setBackground(primaryColor);

//JPanel for the right section of the GUI
rightTutor = new JPanel(new GridLayout(2, 0));
rightTutor.setBackground(secondaryColor);

//Adding all the JLabels
tutorContentPanel.add(teacherIdLabelT);
tutorContentPanel.add(teacherNameLabelT);
tutorContentPanel.add(addressLabelT);
tutorContentPanel.add(workingTypeLabelT);
tutorContentPanel.add(employmentStatLabelT);
tutorContentPanel.add(workingHoursLabelT);
tutorContentPanel.add(salaryLabel);
tutorContentPanel.add(specializationLabel);
tutorContentPanel.add(academicQualificationLabel);
tutorContentPanel.add(performanceIndexLabel);

//Adding all the JTextFields
```

```
tutorContentPanel.add(teacherIdFieldT);
tutorContentPanel.add(teacherNameFieldT);
tutorContentPanel.add(addressFieldT);
tutorContentPanel.add(workingTypeFieldT);
tutorContentPanel.add(employmentStatFieldT);
tutorContentPanel.add(workingHoursFieldT);
tutorContentPanel.add(performanceIndexField);
tutorContentPanel.add(academicQualificationField);
tutorContentPanel.add(specializationField);
tutorContentPanel.add(salaryField);

//Adding header in the top right JPanel
rightTutorTop.add(headerT);

//Adding JButtons in the bottom right JPanel
rightTutorBottom.add(addTutorImage);
rightTutorBottom.add(salarySetImage);
rightTutorBottom.add(removeTutorImage);
rightTutorBottom.add(displayTutorImage);
rightTutorBottom.add(clearTutorImage);
rightTutorBottom.add(goBackImageTutor);

//Adding ImageIcons in the bottom right JPanel
rightTutorBottom.add(addTutor);
rightTutorBottom.add(salarySet);
rightTutorBottom.add(removeTutor);
rightTutorBottom.add(displayT);
rightTutorBottom.add(clearT);
rightTutorBottom.add(goBackTutor);

//Adding Top JPanel and Bottom JPanel in right section
rightTutor.add(rightTutorTop);
rightTutor.add(rightTutorBottom);

//Adding the main content panel in the Left JPanel
mainTutor.add(tutorContentPanel);

//Configuring the tutor JFrame
tutorFrame.setExtendedState(JFrame.MAXIMIZED_BOTH);
tutorFrame.setSize(1200, 900);
tutorFrame.setDefaultCloseOperation(JFrame.HIDE_ON_CLOSE);
tutorFrame.setLayout(new BorderLayout());
tutorFrame.add(mainTutor, BorderLayout.CENTER);
tutorFrame.add(rightTutor, BorderLayout.EAST);

// For Lecturer Frame:
//Header for the top right JPanel
headerL = new JTextArea("Add Lecturers and Grade Assignments");
headerL.setEditable(false);
headerL.setLineWrap(true);
headerL.setWrapStyleWord(true);
headerL.setBackground(secondaryColor);
headerL.setForeground(primaryColor);
headerL.setFont(homePageHeader);
headerL.setPreferredSize(new Dimension(400, 200));
```

```
//JLabels for Lecturer JFrame
//JLabel for Teacher Id in Lecturer Frame
teacherIdLabelL = new JLabel("Teacher ID:");
teacherIdLabelL.setBounds(200, 50, 190, 40);
teacherIdLabelL.setForeground(secondaryColor);
teacherIdLabelL.setFont(mainFont);

//JLabel for Teacher Name in Lecturer Frame
teacherNameLabelL = new JLabel("Teacher Name: ");
teacherNameLabelL.setBounds(200, 130, 190, 40);
teacherNameLabelL.setForeground(secondaryColor);
teacherNameLabelL.setFont(mainFont);

//JLabel for Address in Lecturer Frame
addressLabelL = new JLabel("Address:");
addressLabelL.setBounds(200, 210, 190, 40);
addressLabelL.setForeground(secondaryColor);
addressLabelL.setFont(mainFont);

//JLabel for Working Type in Lecturer Frame
workingTypeLabelL = new JLabel("Working Type:");
workingTypeLabelL.setBounds(200, 290, 190, 40);
workingTypeLabelL.setForeground(secondaryColor);
workingTypeLabelL.setFont(mainFont);

//JLabel for Employment Status in Lecturer Frame
employmentStatLabelL = new JLabel("Employment Status:");
employmentStatLabelL.setBounds(200, 370, 190, 40);
employmentStatLabelL.setForeground(secondaryColor);
employmentStatLabelL.setFont(mainFont);

//JLabel for Years of Experience in Lecturer Frame
yrsOfExperienceLabel = new JLabel("Years of Experience:");
yrsOfExperienceLabel.setBounds(200, 450, 190, 40);
yrsOfExperienceLabel.setForeground(secondaryColor);
yrsOfExperienceLabel.setFont(mainFont);

//JLabel for Graded Score in Lecturer Frame
gradedScoreLabel = new JLabel("Graded Score:");
gradedScoreLabel.setBounds(200, 530, 190, 40);
gradedScoreLabel.setForeground(secondaryColor);
gradedScoreLabel.setFont(mainFont);

//JLabel for Department in Lecturer Frame
departmentLabel = new JLabel("Department:");
departmentLabel.setBounds(200, 610, 190, 40);
departmentLabel.setForeground(secondaryColor);
departmentLabel.setFont(mainFont);

//JLabel for Working Hours in Lecturer Frame
workingHoursLabelL = new JLabel("Working Hours:");
workingHoursLabelL.setBounds(200, 690, 190, 40);
workingHoursLabelL.setForeground(secondaryColor);
workingHoursLabelL.setFont(mainFont);

//JTextFields for the Lecturer JFrame
```

```
//JTextField for Teacher ID in Lecturer Frame
teacherIdFieldL = new JTextField();
teacherIdFieldL.setBounds(200, 85, 400, 40);
teacherIdFieldL.setBackground(primaryColor);
teacherIdFieldL.setForeground(secondaryColor);
teacherIdFieldL.setHorizontalAlignment(JTextField.CENTER);
teacherIdFieldL.setFont(mainFont);

//JTextField for Teacher Name in Lecturer Frame
teacherNameFieldL = new JTextField();
teacherNameFieldL.setBounds(200, 165, 400, 40);
teacherNameFieldL.setBackground(primaryColor);
teacherNameFieldL.setForeground(secondaryColor);
teacherNameFieldL.setHorizontalAlignment(JTextField.CENTER);
teacherNameFieldL.setFont(mainFont);

//JTextField for Address in Lecturer Frame
addressFieldL = new JTextField();
addressFieldL.setBounds(200, 245, 400, 40);
addressFieldL.setBackground(primaryColor);
addressFieldL.setForeground(secondaryColor);
addressFieldL.setHorizontalAlignment(JTextField.CENTER);
addressFieldL.setFont(mainFont);

//JTextField for Working Type in Lecturer Frame
workingTypeFieldL = new JTextField();
workingTypeFieldL.setBounds(200, 325, 400, 40);
workingTypeFieldL.setBackground(primaryColor);
workingTypeFieldL.setForeground(secondaryColor);
workingTypeFieldL.setHorizontalAlignment(JTextField.CENTER);
workingTypeFieldL.setFont(mainFont);

//JTextField for Employment Status in Lecturer Frame
employmentStatFieldL = new JTextField();
employmentStatFieldL.setBounds(200, 405, 400, 40);
employmentStatFieldL.setBackground(primaryColor);
employmentStatFieldL.setForeground(secondaryColor);
employmentStatFieldL.setHorizontalAlignment(JTextField.CENTER);
employmentStatFieldL.setFont(mainFont);

//JTextField for Years of Experience in Lecturer Frame
yrsOfExperienceField = new JTextField();
yrsOfExperienceField.setBounds(200, 485, 400, 40);
yrsOfExperienceField.setBackground(primaryColor);
yrsOfExperienceField.setForeground(secondaryColor);
yrsOfExperienceField.setHorizontalAlignment(JTextField.CENTER);
yrsOfExperienceField.setFont(mainFont);

//JTextField for Graded Score in Lecturer Frame
gradedScoreField = new JTextField();
gradedScoreField.setBounds(200, 565, 400, 40);
gradedScoreField.setBackground(primaryColor);
gradedScoreField.setForeground(secondaryColor);
gradedScoreField.setHorizontalAlignment(JTextField.CENTER);
gradedScoreField.setFont(mainFont);
```

```
        //JTextField for Department in Lecturer Frame
        departmentField = new JTextField();
        departmentField.setBounds(200, 645, 400, 40);
        departmentField.setBackground(primaryColor);
        departmentField.setForeground(secondaryColor);
        departmentField.setHorizontalAlignment(JTextField.CENTER);
        departmentField.setFont(mainFont);

        //JTextField for Working Hours in Lecturer Frame
        workingHoursFieldL = new JTextField();
        workingHoursFieldL.setBounds(200, 725, 400, 40);
        workingHoursFieldL.setBackground(primaryColor);
        workingHoursFieldL.setForeground(secondaryColor);
        workingHoursFieldL.setHorizontalAlignment(JTextField.CENTER);
        workingHoursFieldL.setFont(mainFont);

        //Adding ImageIcons in the Lecturer Frame
        //ImageIcon for Add Lecturer Button
        addLecturerIcon = new
ImageIcon(getClass().getResource("Icons/userIconMain.png"));
        addLecturerImage = new JLabel(addLecturerIcon);
        addLecturerImage.setBounds(250, 20, 40, 40);

        //ImageIcon for Grade Assignment Button
        gradeAssignmentIcon = new
ImageIcon(getClass().getResource("Icons/gradeAssignment.png"));
        gradeAssignmentImage = new JLabel(gradeAssignmentIcon);
        gradeAssignmentImage.setBounds(250, 100, 40, 40);
        //ImageIcon for Display Button
        displayLecturerIcon = new
ImageIcon(getClass().getResource("Icons/display.png"));
        displayLecturerImage = new JLabel(displayLecturerIcon);
        displayLecturerImage.setBounds(250, 180, 40, 40);

        //ImageIcon for Clear Button
        clearLecturerIcon = new ImageIcon(getClass().getResource("Icons/clear.png"));
        clearLecturerImage = new JLabel(clearLecturerIcon);
        clearLecturerImage.setBounds(250, 260, 40, 40);

        //ImageIcon for Go Back Button
        goBackImageLecturer = new JLabel(goBackIcon);
        goBackImageLecturer.setBounds(250, 340, 40, 40);

        //JButtons for the Lecturer Frame
        //JButton for Add Lecturer
        addLecturer = new JButton("Add Lecturer");
        addLecturer.setBounds(100, 20, 150, 40);
        addLecturer.setFocusable(false);
        addLecturer.setBorder(emptyBorder);
        addLecturer.setCursor(buttonCursor);
        addLecturer.setBackground(secondaryColor);
        addLecturer.setForeground(primaryColor);
        addLecturer.setFont(mainFont);
        addLecturer.addActionListener(this);

        //JButton for Grade Assignment
        gradeAssignment = new JButton("Grade Assignment");
```

```
gradeAssignment.setBounds(60, 100, 190, 40);
gradeAssignment.setFocusable(false);
gradeAssignment.setBorder(emptyBorder);
gradeAssignment.setCursor(buttonCursor);
gradeAssignment.setBackground(secondaryColor);
gradeAssignment.setForeground(primaryColor);
gradeAssignment.setFont(mainFont);
gradeAssignment.addActionListener(new ActionListener() {
   public void actionPerformed(ActionEvent e) {
      gradeAssigmentFrame.setVisible(true);
   }
});

//JButton for Display button
displayL = new JButton("Display");
displayL.setBounds(100, 180, 150, 40);
displayL.setFocusable(false);
displayL.setBorder(emptyBorder);
displayL.setCursor(buttonCursor);
displayL.setBackground(secondaryColor);
displayL.setForeground(primaryColor);
displayL.setFont(mainFont);
displayL.addActionListener(this);

//JButton for Clear button
clearL = new JButton("Clear");
clearL.setFocusable(false);
clearL.setBounds(100, 260, 150, 40);
clearL.setBorder(emptyBorder);
clearL.setCursor(buttonCursor);
clearL.addActionListener(this);
clearL.setBackground(secondaryColor);
clearL.setForeground(primaryColor);
clearL.setFont(mainFont);

//JButton for Go Back
goBackLecturer = new JButton("Go Back");
goBackLecturer.setBounds(100, 340, 150, 40);
goBackLecturer.setFocusable(false);
goBackLecturer.setCursor(buttonCursor);
goBackLecturer.setBorder(emptyBorder);
goBackLecturer.setBackground(secondaryColor);
goBackLecturer.setForeground(primaryColor);
goBackLecturer.setFont(mainFont);
goBackLecturer.addActionListener(new ActionListener() {
   public void actionPerformed(ActionEvent e){
      lecturerFrame.setVisible(false);
      welcomeFrame.setVisible(true);
   }
});

//JPanel for main content
mainLecturerContent = new JPanel(null);
mainLecturerContent.setBackground(primaryColor);
mainLecturerContent.setPreferredSize(new Dimension(800, 900));
```

```
//JPanel for the top right section
rightLecturerTop = new JPanel(new FlowLayout(FlowLayout.CENTER, 20, 200));
rightLecturerTop.setBackground(secondaryColor);

//JPanel for the bottom right section
rightLecturerBottom = new JPanel(null);
rightLecturerBottom.setBackground(secondaryColor);

//JPanel for the left section of the GUI
mainLecturer = new JPanel(new FlowLayout());
mainLecturer.setBackground(primaryColor);

//JPanel for the right section of the GUI
rightLecturer = new JPanel(new GridLayout(2, 0));
rightLecturer.setBackground(secondaryColor);

//Adding all the JLabels
mainLecturerContent.add(teacherIdLabelL);
mainLecturerContent.add(teacherNameLabelL);
mainLecturerContent.add(addressLabelL);
mainLecturerContent.add(workingTypeLabelL);
mainLecturerContent.add(employmentStatLabelL);
mainLecturerContent.add(yrsOfExperienceLabel);
mainLecturerContent.add(gradedScoreLabel);
mainLecturerContent.add(departmentLabel);
mainLecturerContent.add(workingHoursLabelL);

//Adding all the JTextFields
```

```
mainLecturerContent.add(teacherIdFieldL);
mainLecturerContent.add(teacherNameFieldL);
mainLecturerContent.add(addressFieldL);
mainLecturerContent.add(workingTypeFieldL);
mainLecturerContent.add(employmentStatFieldL);
mainLecturerContent.add(yrsOfExperienceField);
mainLecturerContent.add(gradedScoreField);
mainLecturerContent.add(departmentField);
mainLecturerContent.add(workingHoursFieldL);

//Adding all the ImageIcons
rightLecturerBottom.add(addLecturerImage);
rightLecturerBottom.add(gradeAssignmentImage);
rightLecturerBottom.add(displayLecturerImage);
rightLecturerBottom.add(clearLecturerImage);
rightLecturerBottom.add(goBackImageLecturer);

//Adding all the JButtons
rightLecturerBottom.add(addLecturer);
rightLecturerBottom.add(gradeAssignment);
rightLecturerBottom.add(displayL);
rightLecturerBottom.add(clearL);
rightLecturerBottom.add(goBackLecturer);

//Adding header in the top right section
rightLecturerTop.add(headerL);

//Adding the main content JPanel in the left section
```

```
mainLecturer.add(mainLecturerContent);

//Adding the Top JPanel and Bottom JPanel in the right section
rightLecturer.add(rightLecturerTop);
rightLecturer.add(rightLecturerBottom);

//Configuring the Lecturer Frame
lecturerFrame.setExtendedState(JFrame.MAXIMIZED_BOTH);
lecturerFrame.setSize(1200, 900);
lecturerFrame.setDefaultCloseOperation(JFrame.HIDE_ON_CLOSE);
lecturerFrame.setLayout(new BorderLayout());
lecturerFrame.add(mainLecturer, BorderLayout.CENTER);
lecturerFrame.add(rightLecturer, BorderLayout.EAST);

// for Grade Assigment Frame
//Header for the Grade Assignment Frame
headerGrade = new JLabel("Grade Assignments");
headerGrade.setBounds(50, 100, 250, 40);
headerGrade.setForeground(primaryColor);
headerGrade.setFont(headerFont);

//JLabels for Grade Assignment Frame
//JLabel for Teacher ID in Grade Assignment Frame
teacherIdGradeLabel = new JLabel("Teacher ID:");
teacherIdGradeLabel.setBounds(50, 50, 190, 40);
teacherIdGradeLabel.setForeground(secondaryColor);
teacherIdGradeLabel.setFont(mainFont);
```

```
//JLabel for Graded Score in Grade Assignment Frame
gradedScoreGradeLabel = new JLabel("Graded Score:");
gradedScoreGradeLabel.setBounds(50, 130, 190, 40);
gradedScoreGradeLabel.setForeground(secondaryColor);
gradedScoreGradeLabel.setFont(mainFont);

//JLabel for Department in Grade Assignment Frame
departmentGradeLabel = new JLabel("Department:");
departmentGradeLabel.setBounds(50, 210, 190, 40);
departmentGradeLabel.setForeground(secondaryColor);
departmentGradeLabel.setFont(mainFont);

//JLabel for Years of Experience in Grade Assignment Frame
yrsOfExperienceGradeLabel = new JLabel("Years of Experience: ");
yrsOfExperienceGradeLabel.setBounds(50, 290, 190, 40);
yrsOfExperienceGradeLabel.setBackground(primaryColor);
yrsOfExperienceGradeLabel.setForeground(secondaryColor);
yrsOfExperienceGradeLabel.setHorizontalAlignment(JTextField.CENTER);
yrsOfExperienceGradeLabel.setFont(mainFont);

//JTextFields for Grade Assignment Frame
//JTextField for Teacher ID in Grade Assignment Frame
teacherIdGradeField = new JTextField();
teacherIdGradeField.setBounds(50, 85, 200, 40);
teacherIdGradeField.setBackground(primaryColor);
teacherIdGradeField.setForeground(secondaryColor);
teacherIdGradeField.setHorizontalAlignment(JTextField.CENTER);
teacherIdGradeField.setFont(mainFont);
```

```
//JTextField for Graded Score in Grade Assignment Frame
gradedScoreGradeField = new JTextField();
gradedScoreGradeField.setBounds(50, 165, 200, 40);
gradedScoreGradeField.setBackground(primaryColor);
gradedScoreGradeField.setForeground(secondaryColor);
gradedScoreGradeField.setHorizontalAlignment(JTextField.CENTER);
gradedScoreGradeField.setFont(mainFont);

//JTextField for Department in Grade Assignment Frame
departmentGradeField = new JTextField();
departmentGradeField.setBounds(50, 245, 200, 40);
departmentGradeField.setBackground(primaryColor);
departmentGradeField.setForeground(secondaryColor);
departmentGradeField.setHorizontalAlignment(JTextField.CENTER);
departmentGradeField.setFont(mainFont);

//JTextField for Years of Experience in Grade Assignment Frame
yrsOfExperienceGradeField = new JTextField();
yrsOfExperienceGradeField.setBounds(50, 325, 200, 40);
yrsOfExperienceGradeField.setBackground(primaryColor);
yrsOfExperienceGradeField.setForeground(secondaryColor);
yrsOfExperienceGradeField.setHorizontalAlignment(JTextField.CENTER);
yrsOfExperienceGradeField.setFont(mainFont);

//JLabel for adding ImageIcon
gradeAssignmentImageSmall = new JLabel(gradeAssignmentIcon);
gradeAssignmentImageSmall.setBounds(220, 185, 40, 40);
```

```
//JLabel for adding ImageIcon
goBackGrade = new JLabel(goBackIcon);
goBackGrade.setBounds(220, 265, 40, 40);

//JButtons for Grade Assignment Frame
//JButton for Grade Assignment
gradeAssignmentGradeButton = new JButton("Grade Assignment");
gradeAssignmentGradeButton.setBounds(30, 185, 190, 40);
gradeAssignmentGradeButton.setFocusable(false);
gradeAssignmentGradeButton.setBorder(emptyBorder);
gradeAssignmentGradeButton.setBackground(secondaryColor);
gradeAssignmentGradeButton.setForeground(primaryColor);
gradeAssignmentGradeButton.setFont(mainFont);
gradeAssignmentGradeButton.addActionListener(this);

//JButton for Go Back
goBackGradeButton = new JButton("Go Back");
goBackGradeButton.setBounds(30, 265, 190, 40);
goBackGradeButton.setFocusable(false);
goBackGradeButton.setBorder(emptyBorder);
goBackGradeButton.setBackground(secondaryColor);
goBackGradeButton.setForeground(primaryColor);
goBackGradeButton.setFont(mainFont);
goBackGradeButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e){
        gradeAssigmentFrame.dispose();
    }
});

//Creating a JPanel for the left side of the frame
mainGradeAssignment = new JPanel();
mainGradeAssignment.setBounds(0, 0, 300, 450);
mainGradeAssignment.setBackground(primaryColor);
mainGradeAssignment.setLayout(null);

//creating a JPanel for the right side of the panel
rightGradeAssignment = new JPanel();
rightGradeAssignment.setBounds(300, 0, 300, 450);
rightGradeAssignment.setBackground(secondaryColor);
rightGradeAssignment.setLayout(null);

//Adding all the labels in the panel
mainGradeAssignment.add(teacherIdGradeLabel);
mainGradeAssignment.add(gradedScoreGradeLabel);
mainGradeAssignment.add(departmentGradeLabel);
mainGradeAssignment.add(yrsOfExperienceGradeLabel);

//Adding all the TextFields in the panel
mainGradeAssignment.add(teacherIdGradeField);
mainGradeAssignment.add(gradedScoreGradeField);
mainGradeAssignment.add(departmentGradeField);
mainGradeAssignment.add(yrsOfExperienceGradeField);

//Adding the ImageIcon
rightGradeAssignment.add(gradeAssignmentImageSmall);
rightGradeAssignment.add(goBackGrade);

//Adding the buttons and the header
rightGradeAssignment.add(headerGrade);
rightGradeAssignment.add(gradeAssignmentGradeButton);
rightGradeAssignment.add(goBackGradeButton);

//Configuring the Grade Assignment Frame
gradeAssigmentFrame.setSize(600, 450);
gradeAssigmentFrame.setDefaultCloseOperation(JFrame.HIDE_ON_CLOSE);
gradeAssigmentFrame.setLayout(null);
gradeAssigmentFrame.setResizable(false);
gradeAssigmentFrame.add(mainGradeAssignment);
gradeAssigmentFrame.add(rightGradeAssignment);

// for Set Salary Frame
//Header for set salary frame
salaryHeader = new JLabel("Set Salary");
salaryHeader.setBounds(90, 100, 250, 40);
salaryHeader.setForeground(primaryColor);
salaryHeader.setFont(headerFont);

//JLabel for Teacher ID
teacherIdSalaryLabel = new JLabel("Teacher ID:");
teacherIdSalaryLabel.setBounds(50, 70, 190, 40);
teacherIdSalaryLabel.setForeground(secondaryColor);
teacherIdSalaryLabel.setFont(mainFont);
```

```
//JLabel for Salary
newSalaryLabel = new JLabel("Salary:");
newSalaryLabel.setBounds(50, 150, 190, 40);
newSalaryLabel.setForeground(secondaryColor);
newSalaryLabel.setFont(mainFont);

//JLabel for Performance Index
performanceIndexSalaryLabel = new JLabel("Performance Index:");
performanceIndexSalaryLabel.setBounds(50, 230, 190, 40);
performanceIndexSalaryLabel.setForeground(secondaryColor);
performanceIndexSalaryLabel.setFont(mainFont);

//ImageIcon for Set salary button
setSalaryImage = new JLabel(salarySetIcon);
setSalaryImage.setBounds(180, 185, 40, 40);

//ImageIcon for go back button
goBackSalary = new JLabel(goBackIcon);
goBackSalary.setBounds(180, 265, 40, 40);

//JTextField for Teacher ID
teacherIdSalaryField = new JTextField();
teacherIdSalaryField.setBounds(50, 105, 200, 40);
teacherIdSalaryField.setBackground(primaryColor);
teacherIdSalaryField.setForeground(secondaryColor);
teacherIdSalaryField.setHorizontalAlignment(JTextField.CENTER);
teacherIdSalaryField.setFont(mainFont);
```

```
//JTextField for salary
newSalaryField = new JTextField();
newSalaryField.setBounds(50, 185, 200, 40);
newSalaryField.setBackground(primaryColor);
newSalaryField.setForeground(secondaryColor);
newSalaryField.setHorizontalAlignment(JTextField.CENTER);
newSalaryField.setFont(mainFont);

//JTextField for Performance Index
performanceIndexSalaryField = new JTextField();
performanceIndexSalaryField.setBounds(50, 265, 200, 40);
performanceIndexSalaryField.setBackground(primaryColor);
performanceIndexSalaryField.setForeground(secondaryColor);
performanceIndexSalaryField.setHorizontalAlignment(JTextField.CENTER);
performanceIndexSalaryField.setFont(mainFont);

//JButton for set salary
setSalaryButton = new JButton("Set Salary");
setSalaryButton.setBounds(30, 185, 150, 40);
setSalaryButton.setFocusable(false);
setSalaryButton.setBorder(emptyBorder);
setSalaryButton.setCursor(buttonCursor);
setSalaryButton.setBackground(secondaryColor);
setSalaryButton.setForeground(primaryColor);
setSalaryButton.setFont(mainFont);
setSalaryButton.addActionListener(this);

//JButton for go back
```

```
goBackSalaryButton = new JButton("Go Back");
goBackSalaryButton.setBounds(30, 265, 150, 40);
goBackSalaryButton.setFocusable(false);
goBackSalaryButton.setBorder(emptyBorder);
goBackSalaryButton.setCursor(buttonCursor);
goBackSalaryButton.setBackground(secondaryColor);
goBackSalaryButton.setForeground(primaryColor);
goBackSalaryButton.setFont(mainFont);
goBackSalaryButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e){
        setSalaryFrame.dispose();
    }
});

//left side panel
mainSalary = new JPanel();
mainSalary.setBackground(primaryColor);
mainSalary.setBounds(0, 0, 300, 450);
mainSalary.setLayout(null);

//right side panel
rightSalary = new JPanel();
rightSalary.setBackground(secondaryColor);
rightSalary.setBounds(300, 0, 300, 450);
rightSalary.setLayout(null);

//adding the labels in the panel
mainSalary.add(teacherIdSalaryLabel);
```

```
        mainSalary.add(newSalaryLabel);
        mainSalary.add(performanceIndexSalaryLabel);

        //adding the text fields in the panel
        mainSalary.add(teacherIdSalaryField);
        mainSalary.add(newSalaryField);
        mainSalary.add(performanceIndexSalaryField);

        //adding the imageicons and buttons in the panel
        rightSalary.add(setSalaryImage);
        rightSalary.add(setSalaryButton);
        rightSalary.add(salaryHeader);
        rightSalary.add(goBackSalaryButton);
        rightSalary.add(goBackSalary);

        //Configuring the set salary frame
        setSalaryFrame.setSize(600, 450);
        setSalaryFrame.setDefaultCloseOperation(JFrame.HIDE_ON_CLOSE);
        setSalaryFrame.setLayout(null);
        setSalaryFrame.setResizable(false);
        setSalaryFrame.add(mainSalary);
        setSalaryFrame.add(rightSalary);
    }

    //Method to detect an event when a button is pressed
    public void actionPerformed(ActionEvent e){
        if(e.getSource() == addTutor){
            //Getting all the text field values which are String
            String teacherName = teacherNameFieldT.getText();
            String address = addressFieldT.getText();
            String workingType = workingTypeFieldT.getText();
            String employmentStat = employmentStatFieldT.getText();
            String academicQualification = academicQualificationField.getText();
            String specialization = specializationField.getText();
            //Checking if any textfield is empty
            if(teacherName.isEmpty() || address.isEmpty() || workingType.isEmpty() ||
employmentStat.isEmpty() || academicQualification.isEmpty() ||
specialization.isEmpty()){
                JOptionPane.showMessageDialog(tutorFrame, "Fields are empty\nFill in all
the fields and try again", "Empty Fields", JOptionPane.ERROR_MESSAGE);
            }else{
                //Try Catch to handle exceptions
                try{
                    //getting all the text field values which are integers or double
                    int teacherID = Integer.parseInt(teacherIdFieldT.getText());
                    double salary = Double.parseDouble(salaryField.getText());
                    int performanceIndex = Integer.parseInt(performanceIndexField.getText());
                    int workingHours = Integer.parseInt(workingHoursFieldT.getText());

                    //Conditions to add a tutor
                    if(salary < 0){
                        JOptionPane.showMessageDialog(tutorFrame, "Salary cannot be lower
than 0", "Error", JOptionPane.ERROR_MESSAGE);
                    }else if(performanceIndex < 0 || performanceIndex > 10){
                        JOptionPane.showMessageDialog(tutorFrame, "Performance Index must
be less than 10 and greater than 0", "Error", JOptionPane.ERROR_MESSAGE);
                    }else if(workingHours < 20 || workingHours > 70){
```

```
                    JOptionPane.showMessageDialog(tutorFrame, "Working Hours must be
greater than 20 and less than 70", "Error", JOptionPane.ERROR_MESSAGE);
                }else{
                    Tutor tutors = new Tutor(teacherID, teacherName, address,
workingType, employmentStat, workingHours, salary, specialization,
academicQualification, performanceIndex);


                    //Flag variable
                    boolean isAdded = false;


                    //Checking if the Array List is empty or not
                    if(Teacher.size() > 0){
                        //Iterating through the Array List using for-each loop
                        for(Teacher tutorObj : Teacher){
                            //checking if the object is an instance of Tutor class and if the ID
match
                            if(tutorObj instanceof Tutor && teacherID == tutorObj.getID()){
                                //changing the flag variable
                                isAdded = true;
                                //breaking the loop if the tutor is already added
                                break;
                            }
                        }
                        //if tutor is already added show an error message if not added then
creating a new tutor object and adding to the Array List
                        if(isAdded == true){
                            JOptionPane.showMessageDialog(tutorFrame, "Tutor with this ID
already exists", "Error", JOptionPane.ERROR_MESSAGE);
                        }else{
                            Teacher.add(tutors);
                            JOptionPane.showMessageDialog(tutorFrame, "Tutor added", "Add
Tutor", JOptionPane.INFORMATION_MESSAGE);

                        }

                    }else{//if Array List is empty then creating a new tutor object and adding
to the Array List

                        Teacher.add(tutors);

                        JOptionPane.showMessageDialog(tutorFrame, "Tutor added", "Add
Tutor", JOptionPane.INFORMATION_MESSAGE);

                    }

                }


            }catch(NumberFormatException exp){

                JOptionPane.showMessageDialog(tutorFrame, "Teacher ID, Working
Hours, Salary and Performance Index must be an Integer", "Exception",
JOptionPane.ERROR_MESSAGE);

            }

        }


    }else if(e.getSource() == addLecturer){

        //Extracting values for JTextFields for String data type

        String teacherName = teacherNameFieldL.getText();

        String address = addressFieldL.getText();

        String workingType = workingTypeFieldL.getText();

        String employmentStat = employmentStatFieldL.getText();

        String department = departmentField.getText();

        //Checking if any of the JTextFields are empty

        if(teacherName.isEmpty() || address.isEmpty() || workingType.isEmpty() ||
employmentStat.isEmpty() || department.isEmpty()){

            JOptionPane.showMessageDialog(lecturerFrame, "Fields are empty\nFill in
all the fields and try again" , "Empty Fields", JOptionPane.ERROR_MESSAGE);

        }
        else{
            try{
                //Extracting all the values from JTextFields for Int data type
                int teacherID = Integer.parseInt(teacherIdFieldL.getText());
                int yrsOfExperience = Integer.parseInt(yrsOfExperienceField.getText());
                int gradedScore = Integer.parseInt(gradedScoreField.getText());
                int workingHours = Integer.parseInt(workingHoursFieldL.getText());


                //Displaying an error if these conditions are not met
                if(yrsOfExperience < 5 || yrsOfExperience > 30){
                    JOptionPane.showMessageDialog(tutorFrame, "Years of Experience
must be greater than 5 and less than 30", "Error", JOptionPane.ERROR_MESSAGE);
                }else if(gradedScore < 0 || gradedScore > 100){
                    JOptionPane.showMessageDialog(tutorFrame, "Graded Score must be
between 0 and 100", "Error", JOptionPane.ERROR_MESSAGE);
                }else if(workingHours < 0 || workingHours > 50){
                    JOptionPane.showMessageDialog(tutorFrame, "Working Hours must be
between 0 and 50", "Error", JOptionPane.ERROR_MESSAGE);
                }else{
                    //Creating a new Lecturer object
                    Lecturer lecturers = new Lecturer(department, yrsOfExperience,
teacherName, teacherID, address, workingType, employmentStat, workingHours);


                    //Flag variable
                    boolean isAdded = false;


                    //Checking if the arraylist is empty or not
                    if(Teacher.size() > 0){
                        //iterating through the ArrayList
                        for(Teacher lecturerObj : Teacher){
                            //checking if the object is an instance of Lecturer and if the teacher
id matches
                            if(lecturerObj instanceof Lecturer && lecturerObj.getID() ==
teacherID){
                                //flag variable set to true because tutor ids match meaning tutor
has already been added
                                isAdded = true;
                                //breaking the loop once ids match
                                break;
                            }
                        }
                        //checking if tutor is added or not
                        if(isAdded == true){
                            //Displaying an error message
                            JOptionPane.showMessageDialog(lecturerFrame, "Lecturer with
that ID is already added", "Error", JOptionPane.ERROR_MESSAGE);
                        }else{
                            //adding lecturer to the arraylist
                            Teacher.add(lecturers);
                            //setting graded score
                            lecturers.setGradedScore(gradedScore);
                            //displaying that the lecturer has been added
                            JOptionPane.showMessageDialog(lecturerFrame, "Lecturer
added", "Add Lecturer", JOptionPane.INFORMATION_MESSAGE);
                        }
                    }else{
                        //adding lecturer to the arraylist because there are no entries in the
arraylist
                        Teacher.add(lecturers);
                        //setting graded score
                        lecturers.setGradedScore(gradedScore);
                        //displaying that the lecturer has been added
                        JOptionPane.showMessageDialog(lecturerFrame, "Lecturer added",
"Add Lecturer", JOptionPane.INFORMATION_MESSAGE);
                    }
                }
            }catch(NumberFormatException exp){
                //error if format does not match
                JOptionPane.showMessageDialog(lecturerFrame, "Teacher ID, Years Of
Experience and Graded Score must be an Integer", "Exception",
JOptionPane.ERROR_MESSAGE);
            }
        }
    }else if(e.getSource() == removeTutor){
        try{
            //extracting the teacher id from jtextfield
            int teacherID = Integer.parseInt(teacherIdFieldT.getText());

            //flag variable
            boolean tutorRemoved = false;

            //checking if the arraylist is empty or not
            if(Teacher.size() > 0){
                //iterating through the arraylist
                for(Teacher tutorObj : Teacher){
                    //checking if the object is an instance of tutor or not and checking the
teacher ids
                    if(tutorObj instanceof Tutor && teacherID == tutorObj.getID()){
                        //once the teacher id match
                        //removing the object of tutor
                    Teacher.remove(tutorObj);
                    //displaying that the tutor is removed
                    JOptionPane.showMessageDialog(tutorFrame, "Tutor Removed",
"Tutor Removed", JOptionPane.INFORMATION_MESSAGE);
                    //setting the flag variable to true
                    tutorRemoved = true;
                    //breaking the loop once tutor is removed
                    break;
                 }
              }


              //if tutor ids do not match
              if(tutorRemoved == false){
                 //displaying the teacher id does not match
                 JOptionPane.showMessageDialog(tutorFrame, "Teacher with that
Teacher ID does not exist", "Error Occurred", JOptionPane.ERROR_MESSAGE);
              }
           }else{
              //if arraylist is empty then display an error message
              JOptionPane.showMessageDialog(tutorFrame, "Tutor has not been
added", "Error Occurred", JOptionPane.ERROR_MESSAGE);
           }
        }catch(NumberFormatException exp){
           //if format doesnt match show error
           JOptionPane.showMessageDialog(tutorFrame, "Teacher ID should be an
Integer", "Exception Occurred", JOptionPane.ERROR_MESSAGE);
        }


     }else if(e.getSource() == gradeAssignmentGradeButton){
        //extracting the value of department from jtextfield
        String department = departmentGradeField.getText();
        //checking if the field is empty
        if(department.isEmpty()){
            //displaying error
            JOptionPane.showMessageDialog(gradeAssigmentFrame, "Fields are
empty\nFill in all the fields and try again", "Error", JOptionPane.ERROR_MESSAGE);
        }else{
            try{
                //extracting values from jtextfields of data type int
                int teacherId = Integer.parseInt(teacherIdGradeField.getText());
                int gradedScore = Integer.parseInt(gradedScoreGradeField.getText());
                int yrsOfExperience =
Integer.parseInt(yrsOfExperienceGradeField.getText());

                //flag variable
                boolean lecturerFound = false;

                //checking if the arraylist is empty or not
                if(Teacher.size() > 0){
                    //iterating through the arraylist
                    for(Teacher obj : Teacher){
                        //checking if the object is an instance of lecturer and their ids match
                        if(obj instanceof Lecturer && obj.getID() == teacherId){
                            //flag variable set to true
                            lecturerFound = true;
                            //downcasting the teacher object to lecturer object to access its
methods
                            Lecturer lecturerObj = (Lecturer) obj;
                            //checking if these conditions match
                    if (yrsOfExperience < 5 || yrsOfExperience > 80){
                        JOptionPane.showMessageDialog(tutorFrame, "Years of
Experience must be greater than 5 and less than 80", "Error",
JOptionPane.ERROR_MESSAGE);
                    }else if(!(lecturerObj.getDepartment().equals(department))){
                        JOptionPane.showMessageDialog(tutorFrame, "Department of
teacher must be same", "Error", JOptionPane.ERROR_MESSAGE);
                    }else if(gradedScore > 100 || gradedScore < 0){
                        JOptionPane.showMessageDialog(gradeAssigmentFrame,
"Graded Score must be greater than 0 and less than 100", "Error",
JOptionPane.ERROR_MESSAGE);
                    }else{
                        String grade;
                        //checking the graded score and giving grades accordingly
                        if(gradedScore >= 70){
                            grade = "Your Grade: A";
                        }else if(gradedScore >= 60){
                            grade = "Your Grade: B";
                        }else if(gradedScore >= 50){
                            grade = "Your Grade: C";
                        }else if(gradedScore >= 40){
                            grade = "Your Grade: D";
                        }else{
                            grade = "Your Grade: E";
                        }
                        //grading the assignment using the method
                        lecturerObj.gradeAssignment(gradedScore, department,
yrsOfExperience);
                        //displaying the grade
                        JOptionPane.showMessageDialog(gradeAssigmentFrame,
grade, "Graded", JOptionPane.INFORMATION_MESSAGE);
```

```
                        //breaking the loop once graded.
                        break;
                    }
                }
            }
            //checking if flag variable is changed or not
            if(lecturerFound == false){
                //displaying error
                JOptionPane.showMessageDialog(gradeAssigmentFrame, "Lecturer
not found!", "Error", JOptionPane.ERROR_MESSAGE);
            }
        }else{
            //displaying error
            JOptionPane.showMessageDialog(gradeAssigmentFrame, "Lecturer has
not been added", "Error", JOptionPane.ERROR_MESSAGE);
        }
    }catch(NumberFormatException exp){
        //displaying error if format does not match
        JOptionPane.showMessageDialog(gradeAssigmentFrame, "Teacher ID,
Graded Score and Years of Experience must be an Integer", "Error",
JOptionPane.ERROR_MESSAGE);
    }
}


}else if(e.getSource() == setSalaryButton){
    try{
        //extracting the values from jtextfields
        int teacherId = Integer.parseInt(teacherIdSalaryField.getText());
        double salary = Double.parseDouble(newSalaryField.getText());
```

```
            int performanceIndex =
Integer.parseInt(performanceIndexSalaryField.getText());


            boolean salarySet = false;
            //checking if the arraylist is empty or not
            if(Teacher.size() > 0){
               //iterating through the arraylist
               for(Teacher obj : Teacher){
                  //checking if the object is an instance of tutor and checking if the ids
match
                  if(obj.getID() == teacherId && obj instanceof Tutor){
                     salarySet = true;
                     //downcasting the teacher object to tutor object to access its methods
                     Tutor tutorObj = (Tutor) obj;
                     //checking if the values are entered correctly
                     if(performanceIndex < 5 || performanceIndex > 10){
                        //displaying error
                        JOptionPane.showMessageDialog(setSalaryFrame, "Performance
Index must be greater than 5 and less than 10", "Error",
JOptionPane.ERROR_MESSAGE);
                     }else if(tutorObj.getWorkingHour() < 20){
                        //displaying error
                        JOptionPane.showMessageDialog(setSalaryFrame, "Working
Hours must be greater than 20", "Error", JOptionPane.ERROR_MESSAGE);
                     }
                     else{
                        //setting salary through the method
                        tutorObj.setSalary(salary, performanceIndex);
                        //displaying the new salary and new performance index
                        JOptionPane.showMessageDialog(setSalaryFrame, "New Salary: "
+ tutorObj.getSalary() + "\n" + "New Performance Index: " +
performanceIndexSalaryField.getText());

                        break;

                    }

                }

            }

            if(salarySet == false){

                //displaying error

                JOptionPane.showMessageDialog(setSalaryFrame, "Tutor with that ID
does not exist", "Error", JOptionPane.ERROR_MESSAGE);

            }


        }else{

            //displaying error

            JOptionPane.showMessageDialog(setSalaryFrame, "Tutor has not been
added", "Error", JOptionPane.ERROR_MESSAGE);

        }

    }catch(NumberFormatException exp){

        //displaying error if format does not match

        JOptionPane.showMessageDialog(setSalaryFrame, "Teacher ID, Salary and
Performance Index must be an Integer", "Error", JOptionPane.ERROR_MESSAGE);

    }

}else if(e.getSource() == clearL){

    //clearing all the textfields

    //confirming with the user if they want to clear or not

    int clear = JOptionPane.showConfirmDialog(lecturerFrame, "Are you sure you
want to clear everything", "Clear", JOptionPane.YES_NO_CANCEL_OPTION);

    //if they select yes clear the fields

    if(clear == JOptionPane.YES_OPTION){
```

```
                teacherIdFieldL.setText("");
                teacherNameFieldL.setText("");
                addressFieldL.setText("");
                workingTypeFieldL.setText("");
                employmentStatFieldL.setText("");
                gradedScoreField.setText("");
                yrsOfExperienceField.setText("");
                departmentField.setText("");
                workingHoursFieldL.setText("");
            }

        }else if(e.getSource() == clearT){
            //clearing all the textfields
            //confirming with the user if they want to clear or not
            int clear = JOptionPane.showConfirmDialog(lecturerFrame, "Are you sure you
want to clear everything", "Clear", JOptionPane.YES_NO_CANCEL_OPTION);
            //if they select yes clear the fields
            if(clear == JOptionPane.YES_OPTION){
                teacherIdFieldT.setText("");
                teacherNameFieldT.setText("");
                addressFieldT.setText("");
                workingTypeFieldT.setText("");
                employmentStatFieldT.setText("");
                workingHoursFieldT.setText("");
                salaryField.setText("");
                specializationField.setText("");
                academicQualificationField.setText("");
                performanceIndexField.setText("");
            }
        }else if(e.getSource() == displayT){
            //checking if the arraylist is empty or not
            if(Teacher.size() > 0){
                //checking if the arraylist is empty or not
                for(Teacher  obj : Teacher){
                    //checking if the object is an instance of tutor
                    if(obj instanceof Tutor){
                        //downcasting the teacher object to tutor object to access its methods
                        Tutor tutorObj = (Tutor) obj;
                        //displaying all the details
                        JOptionPane.showMessageDialog(tutorFrame, "ID: " + tutorObj.getID() +
"\n" + "Name: " + tutorObj.getName() + "\n" + "Address: " + tutorObj.getAddress() + "\n"
+ "Working Type: "  + tutorObj.getWorkType() + "\n" + "Employment Status: "  +
tutorObj.getEmploymentStat() + "\n" + "Working Hour: " + tutorObj.getWorkingHour()  +
"\n" + "Salary: " + tutorObj.getSalary()  + "\n" +  "Specialization: " +
tutorObj.getSpecialization() + "\n" + "Academic Qualification: " +
tutorObj.getAcademicQualification()  + "\n" + "Performance Index: " +
tutorObj.getPerformanceIndex());
                    }
                }
            }else{
                //displaying error is no entries found
                JOptionPane.showMessageDialog(tutorFrame, "No Tutor Added", "Error",
JOptionPane.ERROR_MESSAGE);
            }
        }else if(e.getSource() == displayL){
            //checking if the arraylist is empty or not
            if(Teacher.size() > 0){
                //checking if the arraylist is empty or not
                for(Teacher obj : Teacher){
                    //checking if the object is an instance of lecturer
                    if(obj instanceof Lecturer){
                        //downcasting the teacher object to lecturer object to access its methods
                        Lecturer lecturerObj = (Lecturer) obj;
                        //displaying all the details
                        JOptionPane.showMessageDialog(lecturerFrame, "ID: " +
lecturerObj.getID() + "\n" + "Name: " + lecturerObj.getName() + "\n" + "Address: " +
lecturerObj.getAddress() + "\n" + "Working Type: "  + lecturerObj.getWorkType() + "\n" +
"Employment Status: "  + lecturerObj.getEmploymentStat() + "\n" + "Working Hour: " +
lecturerObj.getWorkingHour()  + "\n" + "Graded Score: " + lecturerObj.getGradedScore()
+ "\n" +  "Years of Experience: " + lecturerObj.getYrsOfExperience());
                    }
                }
            }else{
                //displaying an error
                JOptionPane.showMessageDialog(lecturerFrame, "No Lecturer Added",
"Error", JOptionPane.ERROR_MESSAGE);
            }
        }
      }

      //creating a main method
      public static void main(String[] args) {
        //creating a new object of TeacherGUI
        new TeacherGUI();
      }
}
```

Code of Teacher.java

```
public class Teacher{
  private String teacherAddress;
  private String teacherName;
  private String teacherWorkType;
  private String employmentStat;
  private int teacherID;
  private int teacherWorkingHour;

  /*Constructor to initialize the variables
    Using 'this' keyword to point to instance variables*/

  public Teacher(String teacherName, int teacherID, String teacherAddress, String
teacherWorkType, String employmentStat){
     this.teacherName = teacherName;
     this.teacherID = teacherID;
     this.teacherAddress = teacherAddress;
     this.teacherWorkType = teacherWorkType;
     this.employmentStat = employmentStat;
  }

  //accessor method to get name

  public String getName(){
     return this.teacherName;
  }

  //accessor method to get ID

  public int getID(){
     return this.teacherID;
  }

  //accessor method to get address

  public String getAddress(){
     return this.teacherAddress;
  }

  //accessor method to get work type

  public String getWorkType(){
     return this.teacherWorkType;
  }

  //accessor method to get employment stat

  public String getEmploymentStat(){
     return this.employmentStat;
  }

  //accessor method to get teacher working hour

  public int getWorkingHour(){
     return this.teacherWorkingHour;
  }

  //setter method to set teacher working hour

  public void setWorkingHour(int teacherWorkingHour){
     this.teacherWorkingHour = teacherWorkingHour;
    }

    //display method to display teacher details

    public void display(){
        System.out.println("Name = " + this.teacherName);
        System.out.println("ID = " + this.teacherID);
        System.out.println("Address = " + this.teacherAddress);
        System.out.println("Work Type = " + this.teacherWorkType);
        System.out.println("Employment Status = " + this.employmentStat);

        if(this.teacherWorkingHour >= 0){
            System.out.println("Working Hour = " + this.teacherWorkingHour);
        }else{
            System.out.println("Teacher working hour is not assigned");
        }
    }
}
```

Code of Lecturer.java

```
public class Lecturer extends Teacher{
   private String department;
   private int yrsOfExperience;
   private int gradedScore;
   private boolean hasGraded;

   /*constructor for lecturer class
   'super' keyword is used to inherit the constructor from parent class.*/

   public Lecturer(String department, int yrsOfExperience, String teacherName, int
teacherID, String teacherAddress, String teacherWorkType, String employmentStat, int
teacherWorkingHour){
      super(teacherName, teacherID, teacherAddress, teacherWorkType,
employmentStat);
      setWorkingHour(teacherWorkingHour);
      this.department = department;
      this.yrsOfExperience = yrsOfExperience;
      gradedScore = 0;
      hasGraded = false;
   }

   //accessor method to get department

   public String getDepartment(){
      return this.department;
   }

   //accessor method to get years of experience
```

```
    public int getYrsOfExperience(){
        return this.yrsOfExperience;
    }

    //accessor method to get graded score

    public int getGradedScore(){
        return this.gradedScore;
    }

    //accessor method to get has graded

    public boolean getHasGraded(){
        return this.hasGraded;
    }

    //mutator method to set graded score

    public void setGradedScore(int gradedScore){
        this.gradedScore = gradedScore;
    }

    //method to grade assignment
    public void gradeAssignment(int gradedScore, String department, int
yrsOfExperience){
        if(yrsOfExperience >= 5 && department.equals(this.department)){
            if(gradedScore >= 70){
                System.out.println("Your Grade: A");
            }else if(gradedScore >= 60){
                System.out.println("Your Grade: B");
            }else if(gradedScore >= 50){
```

```
            System.out.println("Your Grade: C");
         }else if(gradedScore >= 40){
            System.out.println("Your Grade: D");
         }else{
            System.out.println("Your Grade: E");
         }
         this.hasGraded = true;
      }else{
         System.out.println("Lecturer has not graded the assignment");
      }
   }

   //method to display the all the details.

   public void display(){
         super.display();
         System.out.println("Department: " + this.department);
         System.out.println("Years of experience: " + this.yrsOfExperience);
         System.out.println("Graded Score: " + this.gradedScore);
      if(this.hasGraded == false){
         System.out.println("Not Graded!");
      }
   }
}
```

Code of Tutor.java

```
public class Tutor extends Teacher{
   private double salary;
   private String specialization;
   private String academicQualification;
   private int performanceIndex;
   private boolean isCertified;

   /*constructor for lecturer class
   'super' keyword is used to inherit the constructor from parent class.*/

   public Tutor(int teacherID, String teacherName, String teacherAddress, String
teacherWorkType, String employmentStat, int teacherWorkingHour, double salary, String
specialization, String academicQualification, int performanceIndex){
      super(teacherName, teacherID, teacherAddress, teacherWorkType,
employmentStat);
      setWorkingHour(teacherWorkingHour);
      this.salary = salary;
      this.specialization = specialization;
      this.academicQualification = academicQualification;
      this.performanceIndex = performanceIndex;
      this.isCertified = false;
   }

   //accessor method to get salary

   public double getSalary(){
      return salary;
   }
```

```
//accessor method to get specialization

public String getSpecialization(){
    return specialization;
}

//accessor method to get academic qualification

public String getAcademicQualification(){
    return academicQualification;
}

//accessor method to get performance index

public int getPerformanceIndex(){
    return performanceIndex;
}

//accessor method to get the value of isCertified

public boolean getIsCertified(){
    return isCertified;
}

//mutator method to set salary of Tutor

public void setSalary(double salary, int performanceIndex){
    if(performanceIndex >= 5 && getWorkingHour() > 20){
        if(performanceIndex >= 5 && performanceIndex <= 7){
            this.salary = salary + (salary * 0.05);
        }else if(performanceIndex == 8 || performanceIndex == 9){
            this.salary = salary + (salary * 0.1);
         }else if(performanceIndex == 10){
            this.salary = salary + (salary * 0.2);
         }
         this.performanceIndex = performanceIndex;
         this.isCertified = true;
      }else{
         System.out.println("Cannot approve new salary because tutor has not been
certified.");
      }
   }

   //method to remove tutor

   public void removeTutor(){
      if(this.isCertified == false){
         this.salary = 0;
         this.performanceIndex = 0;
         this.academicQualification = "";
         this.specialization = "";
         this.isCertified = false;
      }else{
         System.out.println("Tutor is certified cannot remove");
      }
   }

   //method to display details of tutor

   public void display(){
      super.display();
      if(isCertified == true){
            System.out.println("Salary: " + this.salary);
            System.out.println("Specialization: " + this.specialization);
            System.out.println("Academic Qualifications: " + this.academicQualification);
            System.out.println("Performance Index: " + this.performanceIndex);
        }
    }
}
```