CS4001NI Programming

30% Individual Coursework

2023/24 Spring

Student Name: Dikshyant Chapagain

London Met ID: 23049061

College ID: NP01CP4A230128

Group: C8

Assignment Due Date: Tuesday, May 10, 2024

Assignment Submission Date: Tuesday, May 10, 2024

I confirm that I understand my coursework needs to be submitted online via MySecondTeacher under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a marks of zero will be awarded.

Acknowledgement

This is our second coursework of the programming module. The outcome of our second coursework required a lot of guidance from many acquaintances and teachers.
Firstly, I would like to give my thanks to Islington College for providing this opportunity which help a lot to get experience with programmer. Also, I would also like to thank our module leader of this module with providing us with necessary guidance foe this coursework. I would also like to thank our Tutorial teacher Ujjwal Adhikari who gave more detailed information and personally pointed out my errors and how to fix them throughout this process. Lastly, I would like to show my appreciation to every author of the books and blogs in the internet which helped me a lot by providing necessary information for the completion of the coursework along with this report.

Contents

Table of Figures

Table of tables

Introduction

Java

JAVA is an object-oriented programming language which was developed by James Gosling at Sun Microsystems Inc in 1995 and was later acquired by Oracle Corporation. It is a general-purpose programming language which uses the concept of write once run anywhere\ Since Java is an object-oriented language, it is a class-based programming language which is designed to have as few implementations as possible. Some main features of Java include that it is platform independent, this is because the Java source code is compiled into bytecode generated by the compiler. The bytecode can run on any platform with supports JVM (Java Virtual Machine). Java has a vast standard library that provides users with a wide range of tools for various programming tasks which helps in developing enterprise applications, web development etc., making Java one of the most popular programming languages in the world.

(Galkward, 2023)

Figure 1 Java logo

Java Swing

Java Swig is a part of Java Foundation Classes (JFC) that is used to create window-based applications which are built on top of the Abstract Window Toolkit (AWT), Java Swing is a powerful toolkit for creating GUIs in Java. Swing components are more lightweight and efficient and platform independent. Some of the most used components include Buttons, Text fields, Label, Menus, Panels etc. Swing components can respond to events, such as mouse clicks and key presses. To handles the event, event listeners to the components. Event listeners are objects that uses ActionListener, KeyListener and other event listener interfaces.

(Rolandmack, 2023)

Abstract Window Toolkit (AWT)

An API that which contains many classes and methods to create and manage graphical user interface (GUI) applications. Java AWT was developed to provide a set of tools that is used to help develop GUI that could work on many platforms. Java AWT are implemented using each platform which makes AWT platform dependent. But the disadvantage of it is that the GUI may look different based on the platform used.

(StudyTonight, n.d.)

Blue J

Blue J is an IDE (Integrated Development Environment) used for Java programming which was primarily built to assist in education on object-oriented programming. This software helps on providing an interface for coding in Java. A main feature of Blue J is that by organizing visual representation of Java code makes programming language like Java easier to use. So, Blue J is ideal for beginners trying to get into Java. It runs in out computer with the help of JDK.

Blue J was a big factor in creation of this courswork.

Figure 2 BlueJ

Class Diagram

A class diagram is a demonstration of the relationships and source code dependencies among the classes in the Unified Modeling Language (UML). A class defines the methods and variables in an object. Class diagrams are useful in all forms of object-oriented programming (OOP). The classes in class diagram are arranged in groups that share common features. A class diagram resembles a flowchart in which the classes are shown as boxes which has 3 sections in it. The top section holds the name of the class. The middle section holds the attributes of the class and the final section hold the different methods and the relationship between the classes are shown by line or arrows which connects the boxes. (Contributer, n.d.)

Pseudo Code

Pseudo code is a method used to describe the steps of an algorithm in a way that is easier to understand for anyone looking to develop a software. Pseudocode describes the distinct steps of an algorithm in a way that it is easier to anyone to understand. Pseudocodes are used in many fields of programming. Pseudocodes does not have a particular syntax to follow but it must contain full description of the program's logic so that the developer can have full idea on how the software should be developed. Pseudo code helps the programmer to design an algorithm and flowchart of a program. Although there is not any mandatory syntax, there are some simple rules that help make pseudocode more universally understood.

(Metwalli, 2024)

About the project

This is our second coursework of the Programming module. We were required to develop a GUI interface program which is implemented for first coursework of the programming module that stores details of teacher in an ArrayList. We need to create a GUI which creates and new Lecturer and a new Tutor with different along with different methods like grading the assignment by the lecturer, set the salary of the tutor along with removing the Tutor added in the ArrayList.

Aim and Objective

The main aim of this coursework was to create a Graphical User Interface which performs the necessary tasks in the first coursework. The GUI should have a well performing program which creates Lecturer and Tutor and perform various task as specified.

The objectives of this coursework are listed below:

- To create a well-functioning java GUI
- To understand the basic concept of a java User interface program.
- To create a Java program with minimal errors and exceptions as possible
- To create a user-friendly Graphical User Interface

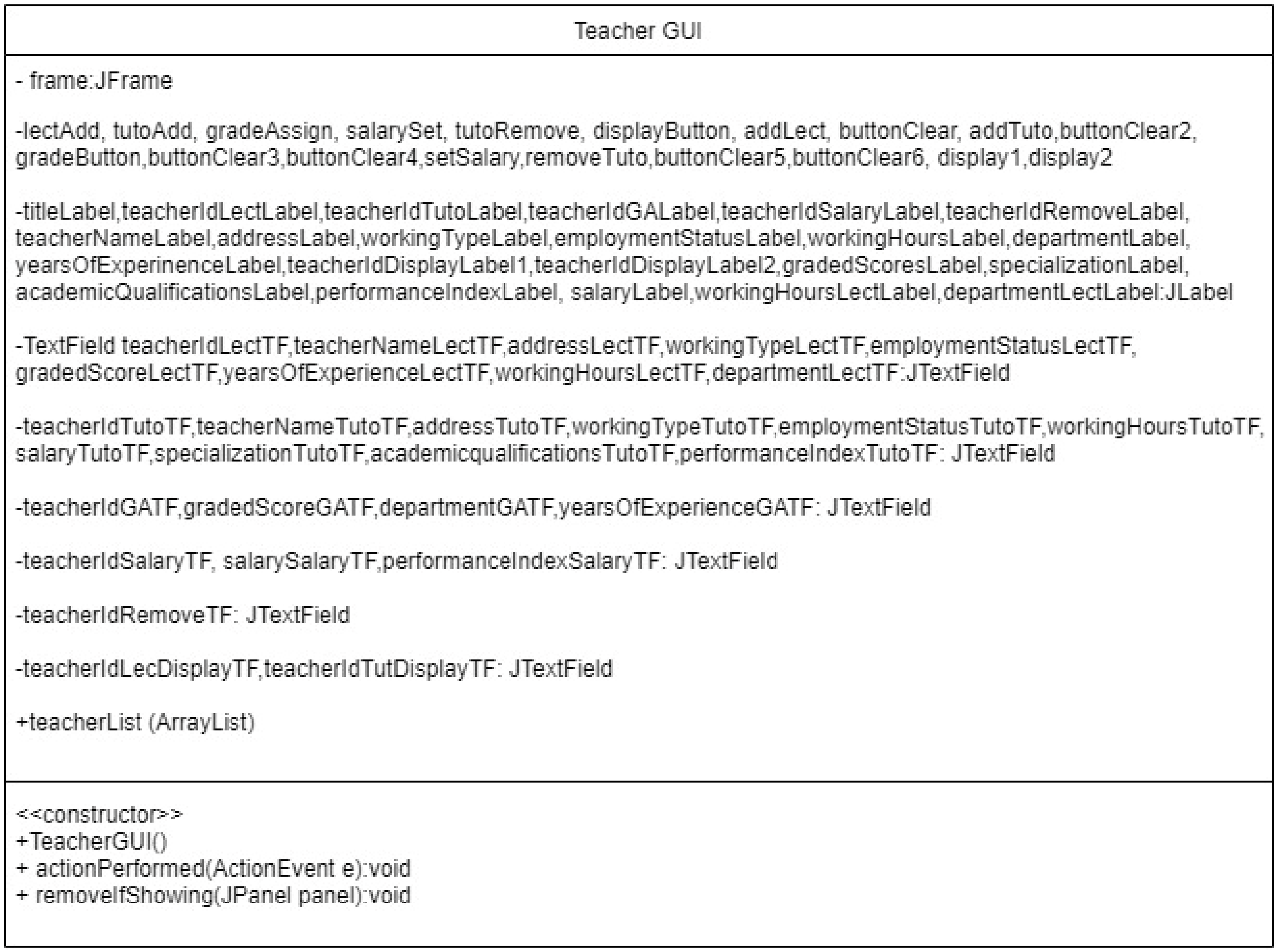
Discussion and analysis

The java program has 4 different classes in which 3 of them were created in the first coursework i.e Teacher, Lecturer and Tutor. The fourth class and the main class of the second coursework is TeacherGUI. This class implements GUI for all the previous classes before. This class has different methods including a Constructor, ActionListener, Main method etc. All the methods in this class performs its task. The GUI has many components. In this program JFrame create a new frame for the GUI. There are other components like JLabel, JTextField,Color and JButtons used in this program. In this program, the components in the frame are managed with the help of Layout manager such as GridLayout,GridBadLayout,BorderLayout. The JButton in this class has ActionListener such that clicking everybuttons on the Frame perfroms a unique action. In JTextField the user is able to write the required data which the program read with the help of the getText() method.

Using all of these components and methods a GUI which runs perfectly was created for this courswork.

Class diagram

The class diagram is represented below:



Teacher GUI

- frame:JFrame

-lectAdd, tutoAdd, gradeAssign, salarySet, tutoRemove, displayButton, addLect, buttonClear, addTuto,buttonClear2, gradeButton,buttonClear3,buttonClear4,setSalary,removeTuto,buttonClear5,buttonClear6, display1,display2

-titleLabel,teacherIdLectLabel,teacherIdTutoLabel,teacherIdGALabel,teacherIdSalaryLabel,teacherIdRemoveLabel, teacherNameLabel,addressLabel,workingTypeLabel,employmentStatusLabel,workingHoursLabel,departmentLabel, yearsOfExperinenceLabel,teacherIdDisplayLabel1,teacherIdDisplayLabel2,gradedScoresLabel,specializationLabel, academicQualificationsLabel,performanceIndexLabel, salaryLabel,workingHoursLectLabel,departmentLectLabel:JLabel

-TextField teacherIdLectTF,teacherNameLectTF,addressLectTF,workingTypeLectTF,employmentStatusLectTF, gradedScoreLectTF,yearsOfExperienceLectTF,workingHoursLectTF,departmentLectTF:JTextField

-teacherIdTutoTF,teacherNameTutoTF,addressTutoTF,workingTypeTutoTF,employmentStatusTutoTF,workingHoursTutoTF, salaryTutoTF,specializationTutoTF,academicqualificationsTutoTF,performanceIndexTutoTF: JTextField

-teacherIdGATF,gradedScoreGATF,departmentGATF,yearsOfExperienceGATF: JTextField

-teacherIdSalaryTF, salarySalaryTF,performanceIndexSalaryTF: JTextField

-teacherIdRemoveTF: JTextField

-teacherIdLecDisplayTF,teacherIdTutDisplayTF: JTextField

+teacherList (ArrayList)

<<constructor>>
+TeacherGUI()
+ actionPerformed(ActionEvent e):void
+ removeIfShowing(JPanel panel):void

Figure 3 Class Diagram

Relational diagram

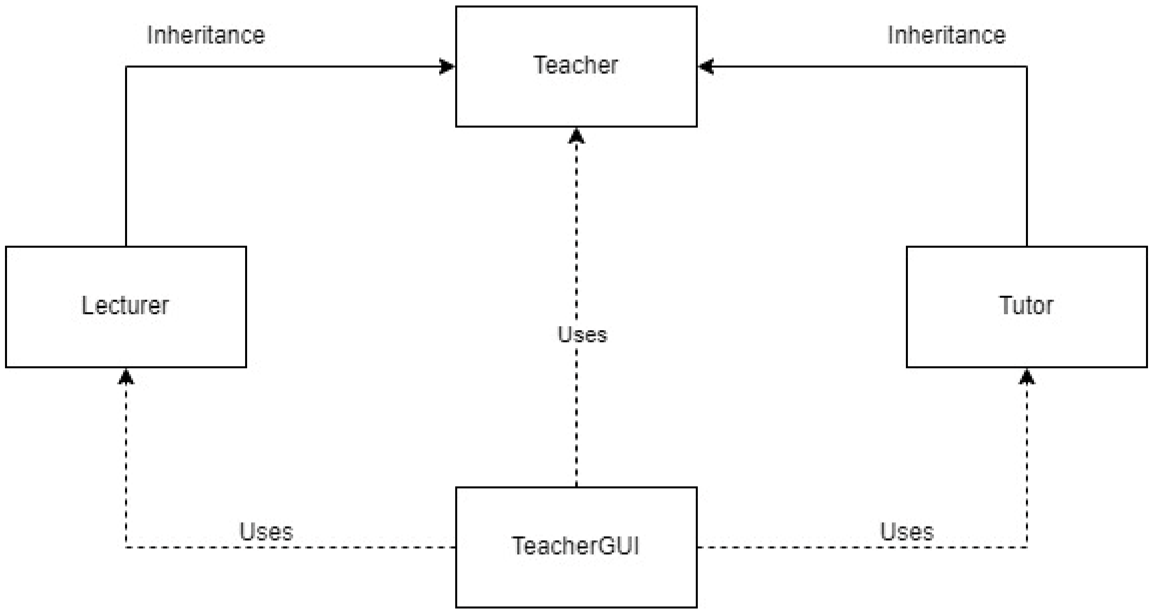


Figure 4 Relation Diagram

Pseudocode for TeacherGUI:

```
IMPORT java.util.ArrayList;

IMPORT java.awt.Color;

IMPORT javax.swing.*;

IMPORT java.awt.*;

IMPORT java.awt.event.ActionEvent;

IMPORT java.awt.event.ActionListener;



CLASS TeacherGUI

DECLARE FRAME AS JFrame

DECLARE

lectAdd, tutoAdd, gradeAssign, salarySet, tutoRemove, displayButton, addLect, buttonClear,
addTuto,buttonClear2,gradeButton,buttonClear3,buttonClear4,setSalary,removeTuto,button
Clear5,buttonClear6, display1,display2 AS JButton

DECLARE

 panel, mainContent, mainContent2, mainContent3, mainContent4,
mainContent5,mainContent6,headingPanel  AS JPANEL
```

DECLARE

titleLabel, teacherIdLectLabel,teacherIdTutoLabel,teacherIdGALabel,teacherIdSalaryLabel,teacherIdRemoveLabel,teacherNameLabel,addressLabel,workingTypeLabel,employmentStatusLabel,workingHoursLabel,departmentLabel,yearsOfExperinenceLabel,teacherIdDisplayLabel1,teacherIdDisplayLabel2,gradedScoresLabel,specializationLabel,academicQualificationsLabel,performanceIndexLabel, salaryLabel,workingHoursLectLabel,departmentLectLabe**l AS JLABEL**

DECLARE
teacherIdLectTF,teacherNameLectTF,addressLectTF,workingTypeLectTF,employmentStatusLectTF,gradedScoreLectTF,yearsOfExperienceLectTF,workingHoursLectTF,departmentLectTF **AS JTextField**

DECLARE
teacherIdTutoTF,teacherNameTutoTF,addressTutoTF,workingTypeTutoTF,employmentStatusTutoTF,workingHoursTutoTF,salaryTutoTF,specializationTutoTF,academicqualificationsTutoTF,performanceIndexTutoTF **AS JTextField**

DECLARE teacherIdGATF,gradedScoreGATF,departmentGATF,yearsOfExperienceGATF **AS JTextField**

DECLARE teacherIdSalaryTF, salarySalaryTF,performanceIndexSalaryTF **AS JTextField**

DECLARE teacherIdRemoveTF **AS JTextField**

DECLARE teacherIdLecDisplayTF,teacherIdTutDisplayTF **AS JTextField**

DECLARE teacherList as ArrayList<Teacher>

```
CREATE CONSTRUCTOR TeacherGUI()

DO

SET frame AS NEW JFrame(“name and London met ID”)

SET frame (860,555) in Frame

SET DefaultCloseOperation (JFrame,EXIT_ON_CLOSE)

SET lectAdd AS NEW JButton(“Add a lecturer”)

SET PreferedSize of lectAdd JButton

SET tutoAdd AS NEW JButton(“Add a Tutor”)

SET PreferedSize of tutoAdd JButton

SET gradeAssign AS NEW JButton(“Assign Grades”)

SET PreferedSize of gradeAssign  JButton

SET salarySet AS NEW JButton(“Set Salary”)

SET PreferredSize of salarySet JButton

SET tutoRemove AS NEW JButton

SET PreferredSize of tutoRemove JButton

SET displayButon AS NEW JButton

SET PreferredSize of displayButon JButton
```

```
**SET** panel **AS NEW** JPanel with GridLayout

**ADD** lectAdd to panel

**ADD** gradeAssign to panel

**ADD** tutoAdd to panel

**ADD** salarySet to panel

**ADD** tutoRemove to panel

**ADD** displayButton to panel

**SET** backgroundcolor of the buttons to cornflower blue

**SET** font of the buttons to “Arial”,Bold, 12

**SET** Border of the buttons to WHITE

**SET** Foreground of the buttons to WHITE


**SET** headingPanel **AS NEW** JPanel

**SET** Background of the headingPanel **AS** cornflower blue

**SET** a EmptyBorder to the headingPanel

**SET** headingPanel **AS NEW** JLabel

**SET** Foreground of the Label to WHITE
```

```
**SET** Font of the Label to Arial,Bold,19

**SET** EmptyBorder

**ADD**  headingLabel to headingPanel

**ADD** headingPanel to the frame
```

```
SET mainContent AS NEW Jpanel with GridBagLayout

CREATE NEW GridBagContraints object  as constraints

SET  the fill property of constraints to GridBagConstraints.HORIZONTAL;

SET the insets property of constraints

SET titleLabel AS new JLabel(“Add a Lecturer”)

SET EmptyBorder to the TitleLabel

SET Font of the titleLabel to Arial, FontBold,26
SET teacherIdLectLabel AS NEW JLabel(“Teacher Id”)

SET  teacherIdLectTF AS NEW  JTextField (10)

SET teacherNameLectLabel AS NEW JLabel(“Teacher Name”)

SET  teacherNameLectTF AS NEW  JTextField (10)

SET addressLabel AS NEW JLabel(“Address”)

SET  addressLectTF AS NEW  JTextField (10)

SET workingTypeLabel AS NEW JLabel(“Working Type”)

SET  workingTypeLectTF AS NEW  JTextField (10)

SET employmentStatusLabel AS NEW JLabel(“Employment Status”)

SET  employementStatusLectTF AS NEW  JTextField (10)

SET gradedScoreLabel AS NEW JLabel(“Graded Score”)

SET  gradedScoreLectTF AS NEW  JTextField (10)

SET yearsOfExperienceLectLabel AS NEW JLabel(“Years of Experience”)

SET  yearsOfExperienceLectTF AS NEW  JTextField (10)

SET departmentLectLabel AS NEW JLabel(“Teacher Id”)

SET  departmentLectTF AS NEW  JTextField (10)

SET addLect AS NEW JButton(“Add Lecturer”)
SET  PreferredSize of adLect JButton



SET buttonClear AS NEW JButton(“Clear”)

SET  PreferredSize of adLect JButton

SET Background of addLect to cornflower blue

SET Foreground of addLect to WHITE

SET Background of buttonClear to cornflower blue

SET Foreground of buttonClear to WHITE



ADD  ActionListener to addLect

CREATE NEW ActionListener

CREATE METHOD actionPerformed(ActionEvent e)

DO

TRY

        IF ANY TextFields are Empty

                SHOW popup as WARNING as “Empty Fields Found!! Please fill all the fields”

        END IF
```

```
ELSE

    SET Variables AS values obtained from the TextFields

    SET Boolean found as false

    FOR EACH teacher object in teacherList

        DO

            IF teacherId equals to teacherId from the ArrayList

            SET found AS true

            END LOOP

            END IF

END DO

IF found IS true

    DO

        SHOW popup dialog as This teacher Id alreay exists add another one"

    END DO

END IF

ELSE

    DO
```

```
        CREATE lecturer object of Lecturer Class

        ADD lecturer to the ArrayList

        SHOW popup which shows the entered values

    END DO

END TRY

    CATCH NumberFormatException e1

        SHOW popup as "Please enter valid input values"

    END CATCH

END DO


ADD  ActionListener to buttonClear

CREATE NEW ActionListener

CREATE METHOD actionPerformed(ActionEvent e)

DO

    SHOW popup message as "Clear all fields?" as WARNING MESSAGE

    SET the Texts in the TextField AS empty

END DO
```

SET constraints.gridx and constraints.gridy to the compoenents

ADD the components to the mainContent Panel

SET mainContent2 **AS NEW** Jpanel with GridBagLayout

CREATE NEW GridBagContraints object as constraints1

SET the fill property of constraints to GridBagConstraints.HORIZONTAL;

SET the insets property of constraints1

SET titleLabel 2 **AS** new JLabel(“Add a Tutor”)

SET EmptyBorder to the TitleLabel

SET Font of the titleLabel to Arial, FontBold,26

SET teacherIdTutoLabel **AS NEW** JLabel(“Teacher Id”)

SET teacherIdTutoTF **AS NEW** JTextField (10)

SET teacherNameLabel **AS NEW** JLabel(“Teacher Name”)

SET teacherNameTutoTF **AS NEW** JTextField (10)

SET workingTypeLabel **AS NEW** JLabel(“Working Type”)

```
SET  teacherIdLectTF AS NEW  JTextField (10)

SET employmentStatusLabel AS NEW JLabel(“EmploymentStatus”)

SET  employmentStatusTutoTF AS NEW  JTextField (10)

SET workingHoursLableAS NEW JLabel(“Working Hours”)

SET  workingHoursTutoTF AS NEW  JTextField (10)

SET academicQualificationsLabel AS NEW JLabel(“Academic Qualifications”)

SET academicQualificationsTutoTF AS NEW  JTextField (10)

SET performanceIndexLabel AS NEW JLabel(“Performance Index”)

SET performanceeIndexTutoTF AS NEW  JTextField (10)

SET addTuto AS NEW JButton

SET PrefeferredSize for addTuto Button

SET buttonClear1 AS NEW JButton

SET PrefeferredSize for addTuto Button

SET Background of addTuto as cornflower blue

SET  Foreground of addTuto as WHITE

SET Background of buttonClear1 as cornflower blue

SET  Foreground of buttonClear as WHITE
```

```
**ADD** ActionListener to addTuto

**CREATE NEW** ActionListener

**CREATE METHOD** actionPerformed(ActionEvent e)

**DO**

**TRY**

    **IF** ANY TextFields are Empty

    **SHOW** popup as WARNING as “Empty Fields Found!! Please fill all the fields”

    **END IF**

    **ELSE**

        **SET** Variables **AS** values obtained from the TextFields

        **SET** Boolean found as false

        **FOR EACH** teacher object in teacherList

        **DO**

            **IF** teacherId equals to teacherId from the ArrayList **AND** teacher object
        isinstanceof Tutor class
```

```
                    SET found AS true

                    SHOW  popup message "A tutor with id " + teacherId + " exists"

                    END LOOP

                    END IF

                END DO

            IF found IS False

            DO

                CREATE  a tutor object of Tutor Class

                ADD tutor object to the ArrayList

                Show popup showing the entered values

            END DO

        END TRY

            CATCH NumberFormatException e1

                SHOW popup as "Please enter valid input values"

            END CATCH

        END DO
```

```
ADD  ActionListener to buttonClear2

CREATE NEW ActionListener

CREATE METHOD actionPerformed(ActionEvent e)

DO

        SHOW popup message as "Clear all fields?" as WARNING MESSAGE

        SET the Texts in the TextField AS empty

END DO

SET constraints1.gridx and constraints1.gridy to the components

ADD the components to the mainContent2 Panel
```

```
SET mainContent3 AS NEW Jpanel with GridBagLayout

CREATE NEW GridBagContraints object  as constraints2

SET  the fill property of constraints2 to GridBagConstraints.HORIZONTAL;

SET the insets property of constraints2


SET titleLabel3 AS new JLabel("Grade Assignment")

SET EmptyBorder to the TitleLabel

SET Font of the titleLabel to Arial, FontBold,22

SET teacherIdGALabel AS NEW JLabel("Teacher Id")

SET  teacherIdGATF AS NEW  JTextField (10)

SET gradedScoresLabel AS NEW JLabel("Graded Score")

SET  gradedScoreGATF AS NEW  JTextField (10)

SET departmentLabel AS NEW JLabel("Department")

SET  departmentGATFAS NEW  JTextField (10)

SET yearsOfExperienceLabel AS NEW JLabel("Years of Expereince")

SET  yearsOfExperienceGATF AS NEW  JTextField (10)
SET gradeButton AS NEW JLabel(“Grade”)

SET PreferredSize for gradeButton

SET buttonClear2 AS NEW JLabel(“Grade”)

SET PreferredSize for buttonClear2

SET Background  of gradeButton AS cornflower blue

SET Foregorund  of gradeButton AS WHITE

SET Background  of buttonClear2 AS cornflower blue

SET Foregorund  of buttonClear2 AS WHITE
```

```
ADD  ActionListener to gradeButton

CREATE NEW ActionListener

CREATE METHOD actionPerformed(ActionEvent e)

DO

TRY

    IF ANY TextFields are Empty

        SHOW popup as WARNING as “Empty Fields Found!! Please fill all the fields”

    END IF

    ELSE

        SET Variables AS values obtained from the TextFields

        SET Boolean found as false

        FOR EACH teacher object in teacherList

            DO

                IF teacherId equals to teacherId from the ArrayList

                SET found AS true
```

```
                    IF teacher object is an instanceof Lecturer
               CAST teacher to a Lecturer type
               SET Variables AS values obtained from the TextFields
          CALL the gradeAssignment method form the lecturer class
          SHOW a popup which shows the entered information
END IF
ELSE
     SHOW a popup as "Teacher with this ID is not a Lecturer"
END ELSE
END LOOP
IF FOUND AS false
     SHOW a popup as “Teacher with this ID not found"
END IF
END ELSE
END TRY
CATCH NumberFormatException e1
     SHOW popup as "Please enter valid input values"
```

```
END CATCH

END DO



ADD  ActionListener to buttonClear3

CREATE NEW ActionListener

CREATE METHOD actionPerformed(ActionEvent e)

DO

        SHOW popup message as "Clear all fields?" as WARNING MESSAGE

        SET the Texts in the TextField AS empty

END DO

SET constraints2.gridx and constraints2.gridy to the components

ADD the components to the mainContent3 Panel
```

```
SET mainContent4 AS NEW Jpanel with GridBagLayout

CREATE NEW GridBagContraints object  as constraints3

SET  the fill property of constraints3 to GridBagConstraints.HORIZONTAL;

SET the insets property of constraints3

SET titleLabel4 AS new JLabel("Set Salary")

SET EmptyBorder to the TitleLabel

SET Font of the titleLabel to Arial, Font.Bold,26

SET teacherIdSalaryLabel AS NEW JLabel("Teacher Id")

SET  teacherIdSalaryTF AS NEW  JTextField (10)

SET performanceIndexLabel AS NEW JLabel("Performance Index")

SET  performanceIndexSalaryTF AS NEW  JTextField (10)

SET salaryLabel AS NEW JLabel("Salary")

SET  salarySalaryTF AS NEW  JTextField (10)

SET setSalary AS NEW JButton("Set")

SET PreferredSize for gradeButton

SET buttonClear4 AS NEW JLabel("Clear")
SET PreferredSize for buttonClear4

SET Background for setSalary as cornflower blue

SET Foreground for setSalary as WHITE

SET Background for buttonClear4 as cornflower blue

SET Foreground for buttonClear4 as WHITE



ADD  ActionListener to setSalary

CREATE NEW ActionListener

CREATE METHOD actionPerformed(ActionEvent e)

DO

TRY

    IF ANY TextFields are Empty

        SHOW popup as WARNING as “Empty Fields Found!! Please fill all the fields”

    END IF

    ELSE

        SET Variables AS values obtained from the TextFields
```

```
                **SET** Boolean found as false

                **FOR EACH** teacher object in teacherList

        **DO**

                **IF** teacherId equals to teacherId from the ArrayList AND teacher is an instance
                of Tutor

                        **SET** found **AS** true

                        **CAST** teacher to a Tutor type



                        **SET** Variables **AS** values obtained from the TextFields

                        **CALL** the setSalary method form the Tutor class

                        **SHOW** a popup which shows the entered information

                **END IF**

**ELSE**

        **SHOW a** popup as "Teacher with this ID is not a tutor"

**END ELSE**

**END LOOP**



**END ELSE**
```

```
END TRY

CATCH NumberFormatException e1

    SHOW popup as "Please enter valid input values"

END CATCH

END DO




ADD  ActionListener to buttonClear4

CREATE NEW ActionListener

CREATE METHOD actionPerformed(ActionEvent e)

DO

    SHOW popup message as "Clear all fields?" as WARNING MESSAGE

    SET the Texts in the TextField AS empty

END DO

SET constraints3.gridx and constraints3.gridy to the components

ADD the components to the mainContent4 Panel
```

```
SET mainContent5 AS NEW Jpanel with GridBagLayout

CREATE NEW GridBagContraints object  as constraints4

SET  the fill property of constraints4 to GridBagConstraints.HORIZONTAL;

SET the insets property of constraints4



SET titleLabel5 AS new JLabel(“Remove Tutor”)

SET EmptyBorder to the TitleLabel

SET Font of the titleLabel to Arial, FontBold,24

SET removeTuto AS NEW JButton(Remove)

SET PreferredSize for removeTuto

SET Background for remove as cornflower blue

SET Foreground for remove as WHITE

SET Background for buttonClear5 as cornflower blue

SET Foreground for buttonClear5 as WHITE
SET teacherIdRemoveLabel AS NEW JLabel("Teacher Id")

SET  teacherIdRemoveTF AS NEW  JTextField (10)



ADD  ActionListener to removeTuto

CREATE NEW ActionListener

CREATE METHOD actionPerformed(ActionEvent e)

DO

TRY

    IF ANY TextFields are Empty

        SHOW popup as WARNING as "Empty Fields Found!! Please fill all the fields"

    END IF

    ELSE

        SET Variables AS values obtained from the TextFields

        SET Boolean found as false

        FOR EACH teacher object in arraylist

            IF teacherID equals to teacher.getteacherId() AND teacher is instance of
        Tutor
```

```
                    **SET FOUND AS** true
                    **CAST** teacher to a Tutor type
                    **REMOVE** tutor from the teacherList arrayList
                    **SHOW** Popup Message as (Tutor Remove)
                **END LOOP**
    **END IF**
**IF FOUND** as false
        **SHOW** popup Message as "Tutor with that Id not found"
**END IF**
**END ELSE**
**END TRY**
**CATCH** NumberFormatException e1
        **SHOW** popup Message as “Enter valid values”
**END CATCH**
**END DO**
```

```
ADD ActionListener to buttonClear5

CREATE NEW ActionListener

CREATE METHOD actionPerformed(ActionEvent e)

DO

    SHOW popup message as "Clear all fields?" as WARNING MESSAGE

    SET the Texts in the TextField AS empty

END DO

SET constraints4.gridx and constraints4.gridy to the components

ADD the components to the mainContent5 Panel
```

```
**SET** mainContent6 **AS NEW** Jpanel with GridBagLayout

**CREATE NEW** GridBagContraints object  as constraints5

**SET**  the fill property of constraints5 to GridBagConstraints.HORIZONTAL;

**SET** the insets property of constraints5


**SET** titleLabel6 **AS** new JLabel(“Display”)

**SET** EmptyBorder to the TitleLabel

**SET** Font of the titleLabel to Arial, FontBold,24


**SET** teacherIdDisplayLabel1**AS NEW** JLabel(“Teacher Id(Lecturer)”)

**SET**  teacherIdLecDisplayTF **AS NEW**  JTextField (10)


**SET** teacherIdDisplayLabel2**AS NEW** JLabel(“Teacher Id(Lecturer)”)

**SET**  teacherIdTutDisplayTF **AS NEW**  JTextField (10)

**SET** display1 **AS NEW** JButton(“Display Lecturer”)

**SET** PreferredSize for display1

**SET** display2 **AS NEW** JButton(“Display Tutor”)
```

SET PreferredSize for display2

SET buttonClear6 **AS NEW** JButton(“Clear”)

SET PreferredSize for buttonClear6

SET constraints5.gridx and constraints5.gridy to the compoenents

ADD the components to the mainContent6 Panel

SET Background for display1 as cornflower blue

SET Foreground for display1 as WHITE

SET Background for buttonClear5 as cornflower blue

SET Background for display2 as cornflower blue

SET Foreground for display2 as WHITE

SET Background for buttonClear6 as cornflower blue

SET Foreground for buttonClear6 as WHITE

```
ADD  ActionListener to display1

CREATE NEW ActionListener

CREATE METHOD actionPerformed(ActionEvent e)

DO

TRY

    IF ANY TextFields are Empty

        SHOW popup as WARNING as “Empty Fields Found!! Please fill all the fields”

    END IF

    ELSE

        SET Variables AS values obtained from the TextFields

        SET Boolean found as false

        FOR EACH teacher object in arraylist

            IF teacher an instance of Lecturer AND teacherId is equl to
        teacher.getteacherID()

            SET FOUND AS true

            CAST teacher to a Lecturer type

            SET teacherInfo to the strDIsplay() of the lecture object
```

```
                **SHOW** Popup Message which shows the lecturer information as
            INFORMATION MESSAGE

            END LOOP

    END IF

IF FOUND as false

        SHOW popup Message as "Techer not found"

END IF

END ELSE

END TRY

CATCH NumberFormatException e1

        SHOW popup Message as “Enter valid values”

END CATCH

END DO
```

```
ADD  ActionListener to display2

CREATE NEW ActionListener

CREATE METHOD actionPerformed(ActionEvent e)

DO

TRY

    IF ANY TextFields are Empty

        SHOW popup as WARNING as “Empty Fields Found!! Please fill all the fields”

    END IF

    ELSE

        SET Variables AS values obtained from the TextFields

        SET Boolean found as false

        FOR EACH teacher object in arraylist

        IF teacher an instance of Tutor AND teacherId is equl to teacher.getteacherID()

            SET FOUND AS true

            CAST teacher to a Tutor type

            SET teacherInfo to the strDIsplay() of the lecture object

            SHOW Popup Message which shows the lecturer information as
        INFORMATION MESSAGE
```

```
            END LOOP

    END IF

IF FOUND as false

      SHOW popup Message as "Techer not found"

END IF

END ELSE

END TRY

CATCH NumberFormatException e1

      SHOW popup Message as “Enter valid values”

END CATCH

END DO
```

```
ADD  ActionListener to buttonClear6

CREATE NEW ActionListener

CREATE METHOD actionPerformed(ActionEvent e)

DO

    SHOW popup message as "Clear all fields?" as WARNING MESSAGE

    SET the Texts in the TextField AS empty

END DO

SET constraints5.gridx and constraints5.gridy to the components

ADD the components to the mainContent6 Panel
ADD  ActionListener to lectAdd

CREATE NEW ActionListener

CREATE METHOD actionPerformed(ActionEvent e)

DO

removeIfShowing(mainContent2)

removeIfShowing(mainContent3)

removeIfShowing(mainContent4)

removeIfShowing(mainContent5)

removeIfShowing(mainContent6)

IF mainContent is not showing

    DO

        Add mainContent panel to the frame

        SET Background as WHITE

END IF

Perform revalidate() on frame

Perform repaint() on frame

END DO
```

```
CREATE METHOD removeIfShowing with Jpanel panel as parameter

DO

IF panel is showing

DO

Remove the panel

END DO

END IF

END DO
```

```
ADD  ActionListener to tutoAdd

CREATE NEW ActionListener

CREATE METHOD actionPerformed(ActionEvent e)

DO

removeIfShowing(mainContent)

removeIfShowing(mainContent3)
removeIfShowing(mainContent4)

removeIfShowing(mainContent5)

removeIfShowing(mainContent6)

IF mainContent2 is not showing

    DO

        Add mainContent2 panel to the frame

        SET Background as WHITE

END IF

Perform revalidate() on frame

Perform repaint() on frame

END DO

CREATE METHOD removeIfShowing with Jpanel panel as parameter

DO

IF panel is showing

DO

Remove the panel

END DO
END IF
END DO

ADD ActionListener to gradeAssign
CREATE NEW ActionListener
CREATE METHOD actionPerformed(ActionEvent e)
DO
removeIfShowing(mainContent)
removeIfShowing(mainContent2)
removeIfShowing(mainContent4)
removeIfShowing(mainContent5)
removeIfShowing(mainContent6)
IF mainContent3 is not showing
```

```
        DO
                Add mainContent3 panel to the frame
                SET Background as WHITE
END IF
Perform revalidate() on frame
Perform repaint() on frame
END DO
CREATE METHOD removeIfShowing() with Jpanel panel as parameter
DO
IF panel is showing
DO
Remove the panel
END DO
END IF
END DO
```

```
ADD  ActionListener to salarySet

CREATE NEW ActionListener

CREATE METHOD actionPerformed(ActionEvent e)

DO

removeIfShowing(mainContent)

removeIfShowing(mainContent2)

removeIfShowing(mainContent3)

removeIfShowing(mainContent5)

removeIfShowing(mainContent6)

IF mainContent4 is not showing

    DO

        Add mainContent4 panel to the frame

        SET Background as WHITE

END IF

Perform revalidate() on frame

Perform repaint() on frame

END DO
```

```
CREATE METHOD removeIfShowing with Jpanel panel as parameter
DO
IF panel is showing
DO
Remove the panel
END DO
END IF
END DO


ADD  ActionListener to tutoRemove
CREATE NEW ActionListener
CREATE METHOD actionPerformed(ActionEvent e)
DO
removeIfShowing(mainContent)
removeIfShowing(mainContent2)
removeIfShowing(mainContent3)
removeIfShowing(mainContent4)
removeIfShowing(mainContent6)

IF mainContent5 is not showing

        DO

                Add mainContent5 panel to the frame

                SET Background as WHITE

END IF

Perform revalidate() on frame

Perform repaint() on frame

END DO

CREATE METHOD removeIfShowing with Jpanel panel as parameter

DO

IF panel is showing

DO

Remove the panel

END DO

END IF

END DO
```

```
ADD ActionListener to displayButton

CREATE NEW ActionListener

CREATE METHOD actionPerformed(ActionEvent e)

DO

removeIfShowing(mainContent)

removeIfShowing(mainContent2)

removeIfShowing(mainContent3)

removeIfShowing(mainContent4)

removeIfShowing(mainContent5)

IF mainContent6 is not showing

    DO

        Add mainContent6 panel to the frame

        SET Background as WHITE

END IF

Perform revalidate() on frame

Perform repaint() on frame

END DO
```

```
CREATE METHOD removeIfShowing with Jpanel panel as parameter

DO

IF panel is showing

DO

Remove the panel

END DO

END IF

END DO



SET Visibility of mainContent to ture

ADD panel Jpanel to the frame

ADD mainContent Jpanel to the frame

SET Background of mainContent to WHITE

SET Visibility of frame to true

END DO
```

```
CREATE METHOD main(String[] args)

DO

CREATE  NEW  TeacherGUI();

END DO

END DO
```

Method description

Method Name: TeacherGUI()
This is the constructor of the TeacherGUI class. The constructor initializes the components of the graphical user interface and sets up the l layout for the UI. It creates a JFrame, sets up its title, size and default close operation. It also creates several buttons which changes the JFrame according to the user's needs. The buttons are arranged using Gridlayout. It also creates several JPanels in which the components in each panels are arranged with GridBagLayout and several JButton and JTextFields are aaded to it for entering informations.

Method Name: actionPerformed(ActionEvent e)
This method sets the action listeners for the buttons which are initialized in the constructor where the buttons on the side of the JFrame changes the layout of the Panel based on the required information to be entered by the user. The Action Listener in the Add Lecturer creates a new object of the lecturer class and adds that object to an ArrayList and validates the input entered by the user are valid or not. If not, appropriate message will be shown indicating that the input entered by the user is wrong. The action listener for the "Grade" grades an assignment if the object added in the ArrayList is a lecturer. If not, an appropriate message will be shown to the user. If lecturer, then the method to grade the Assignment will be called from the Teacher class.

The action listener for Add Tutor takes valid input from the user and then creates an object of the Tutor class and then adds the object to the ArrayList. If not valid, then an appropriate message will be shown to the user. The action listener to Set Salary calls the method to set salary from the tutor class if the object in the ArrayList is an object of the Tutor class. The action listener for the Remove button removes the tutor object from the ArrayList. The action listener in the clear button clears the values in the textfield at once and the Action Listener for display button shows the relevant information about the lecturer and the tutor object.

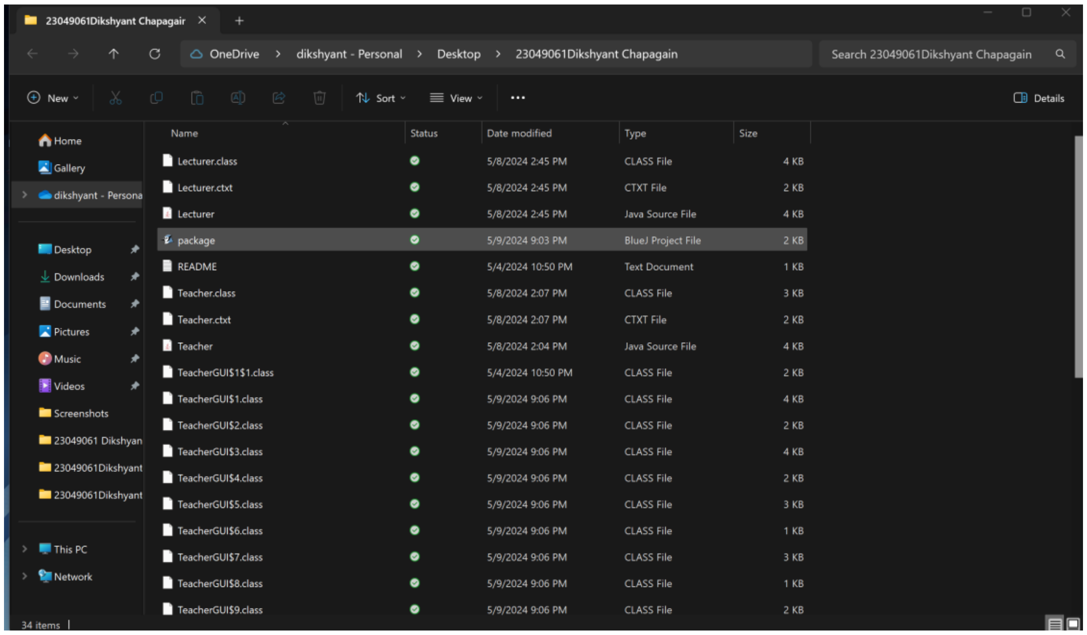
Method name : main(String[] args)
This is the main method for the TeacherGUI class. This method creates an instance of the "Teacher GUI" class by calling the constructer. This initializes the GUI components and displays the GUI on the screen for the user to interact with.

Testing

Test that the program can be compiled and run using the command prompt.

Test No	1
Objective	Test that the program can be compiled and run using the command prompt
Action	• The command Prompt was opened • The directory was changed to where the java program was stored • Javac along with the name of java file was typed • Java along with the name of java file was typed
Expected Result	The program should start
Actual Result	The program was successfully started
Conclusion	The Test was sucessful

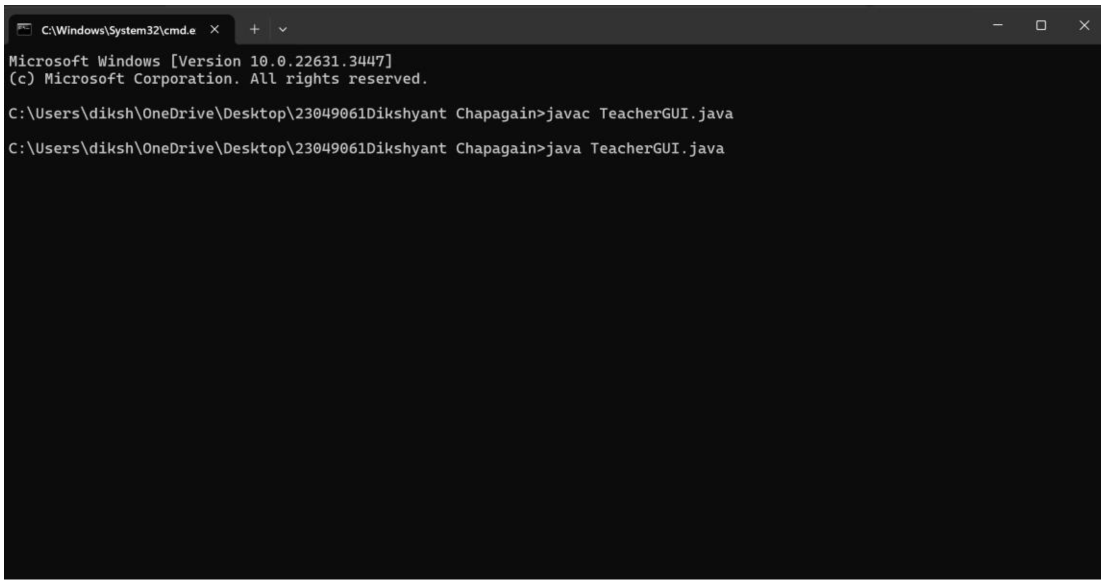
Table 1 Test 1



Name	Status	Date modified	Type	Size
Lecturer.class	✓	5/8/2024 2:45 PM	CLASS File	4 KB
Lecturer.ctxt	✓	5/8/2024 2:45 PM	CTXT File	2 KB
Lecturer	✓	5/8/2024 2:45 PM	Java Source File	4 KB
package	✓	5/9/2024 9:03 PM	BlueJ Project File	2 KB
README	✓	5/4/2024 10:50 PM	Text Document	1 KB
Teacher.class	✓	5/8/2024 2:07 PM	CLASS File	3 KB
Teacher.ctxt	✓	5/8/2024 2:07 PM	CTXT File	2 KB
Teacher	✓	5/8/2024 2:04 PM	Java Source File	4 KB
TeacherGUI$1$1.class	✓	5/4/2024 10:50 PM	CLASS File	2 KB
TeacherGUI$1.class	✓	5/9/2024 9:06 PM	CLASS File	4 KB
TeacherGUI$2.class	✓	5/9/2024 9:06 PM	CLASS File	2 KB
TeacherGUI$3.class	✓	5/9/2024 9:06 PM	CLASS File	4 KB
TeacherGUI$4.class	✓	5/9/2024 9:06 PM	CLASS File	2 KB
TeacherGUI$5.class	✓	5/9/2024 9:06 PM	CLASS File	3 KB
TeacherGUI$6.class	✓	5/9/2024 9:06 PM	CLASS File	1 KB
TeacherGUI$7.class	✓	5/9/2024 9:06 PM	CLASS File	3 KB
TeacherGUI$8.class	✓	5/9/2024 9:06 PM	CLASS File	1 KB
TeacherGUI$9.class	✓	5/9/2024 9:06 PM	CLASS File	2 KB

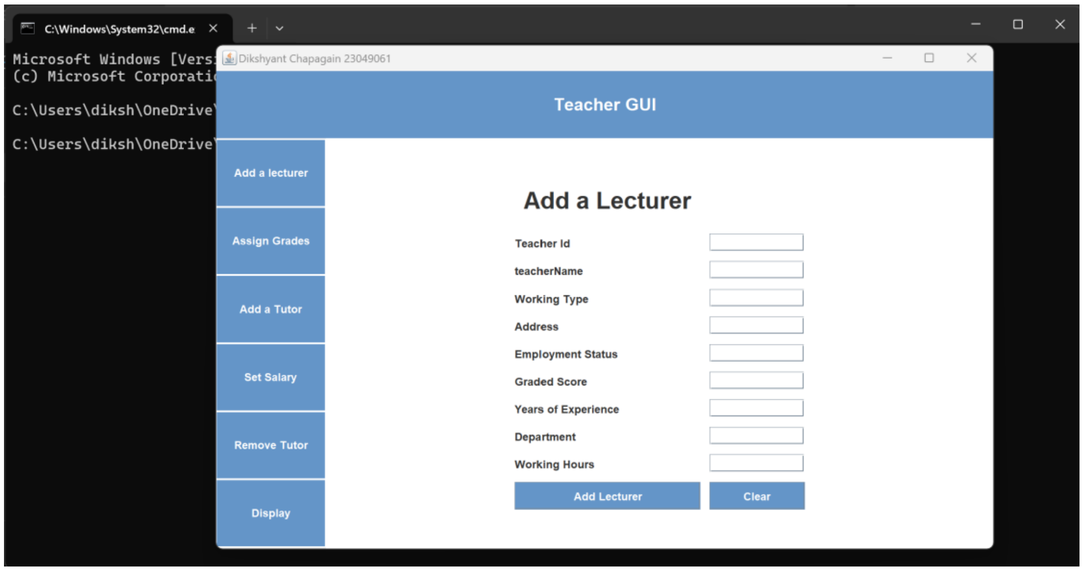
Figure 5 Test 1: Folder containing javafile

Figure 6 Test 1: Running program through command prompt!



```
Microsoft Windows [Version 10.0.22631.3447]
(c) Microsoft Corporation. All rights reserved.

C:\Users\diksh\OneDrive\Desktop\23049061Dikshyant Chapagain>javac TeacherGUI.java

C:\Users\diksh\OneDrive\Desktop\23049061Dikshyant Chapagain>java TeacherGUI.java
```

Figure 7 Test 1: Running program through command prompt



```
Microsoft Windows [Vers
(c) Microsoft Corporati

C:\Users\diksh\OneDrive\

C:\Users\diksh\OneDrive\
```

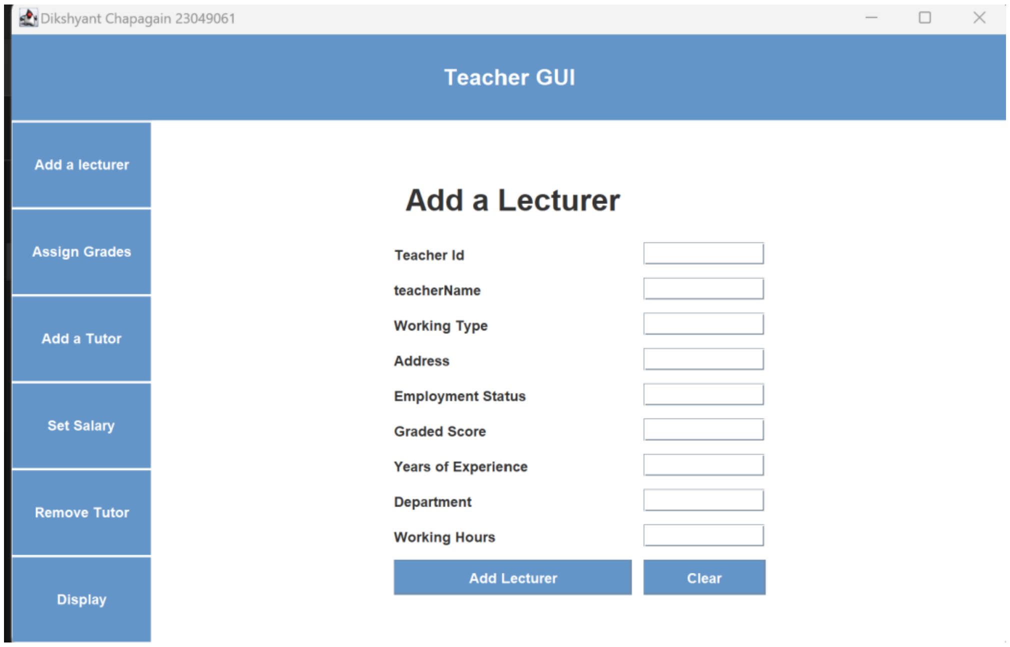
Figure 8Test 1: Running the program

Test 2: Evidence should be shown of:

a. Add a Lecturer

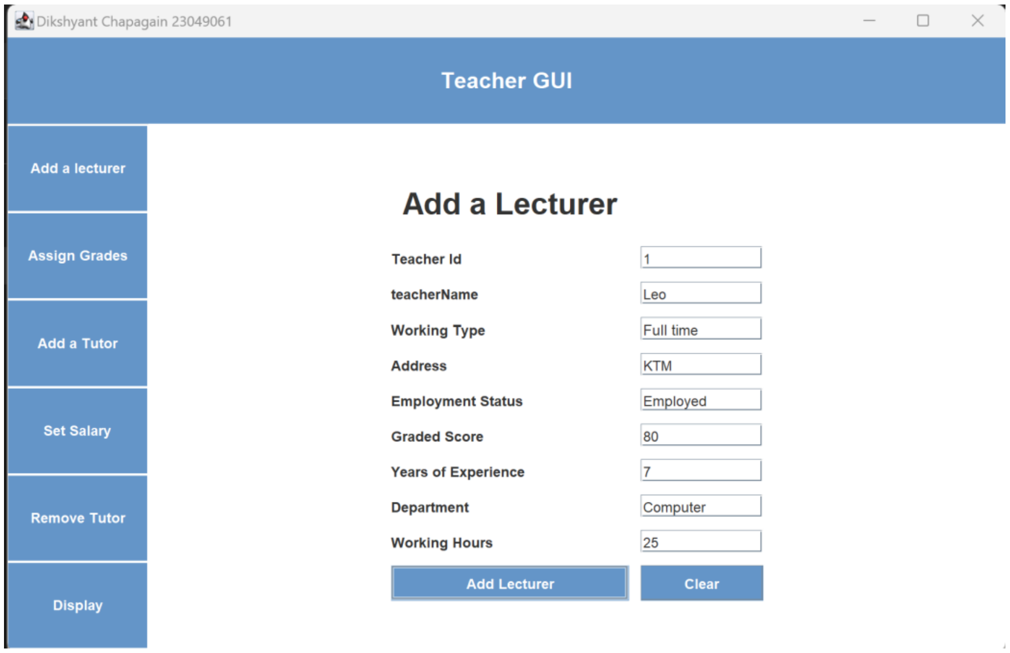
Test No	2.1
Objective	To show evidence of adding a Lecturer
Action	• The button add Lecturer was clicked. • The necessary values for teacherId, teacherName,WorkingType,Address,Employment Status, GradedScore, Years of Experience,Department and working hours were enterted • The button Add Lecturer was clicked • The Display Button was clicked • And the teacherId of the lecturer was entered in the teacherId (Lecturer) textfield and “Display Lecturer” Button was clicked.
Expected Result	• A popup showing the message “Lecturer Added with Id 1should be shown” after clicking the Add Lecturer Button • After clicking “Display Lecturer” button the information related to the object should be shown in a popup
Actual Result	• A popup showing the message “Lecturer Added with Id 1 was shown” along with the entered value after clicking the Add Lecturer Button • After clicking “Display Lecturer” button the information related to the object was shown in a popup
Conclusion	The Test was successful

Table 2 Test 2.1



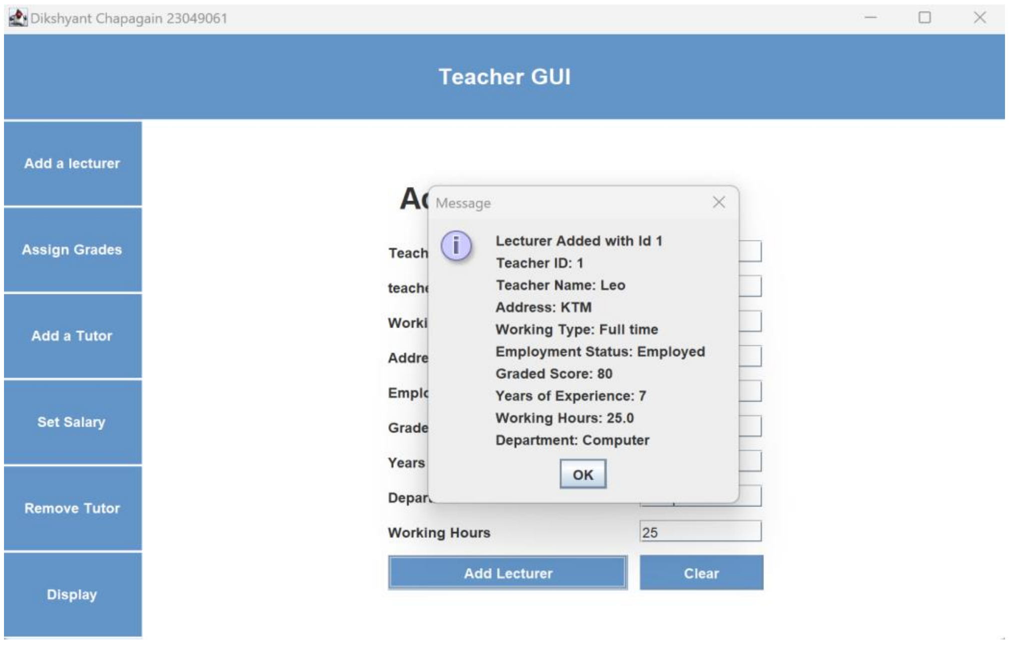
Teacher Id	
teacherName	
Working Type	
Address	
Employment Status	
Graded Score	
Years of Experience	
Department	
Working Hours	

Figure 9 Test 2.1.1: clicking the Add a lecturer button and it shows Add a Lecturer Panel



Teacher Id	1
teacherName	Leo
Working Type	Full time
Address	KTM
Employment Status	Employed
Graded Score	80
Years of Experience	7
Department	Computer
Working Hours	25

Figure 10 Test2.1.2 Entering the necessary values in the text field.



Lecturer Added with Id 1
Teacher ID: 1
Teacher Name: Leo
Address: KTM
Working Type: Full time
Employment Status: Employed
Graded Score: 80
Years of Experience: 7
Working Hours: 25.0
Department: Computer

Figure 11 Test 2.1.3 Popup which shows the entered values by the user

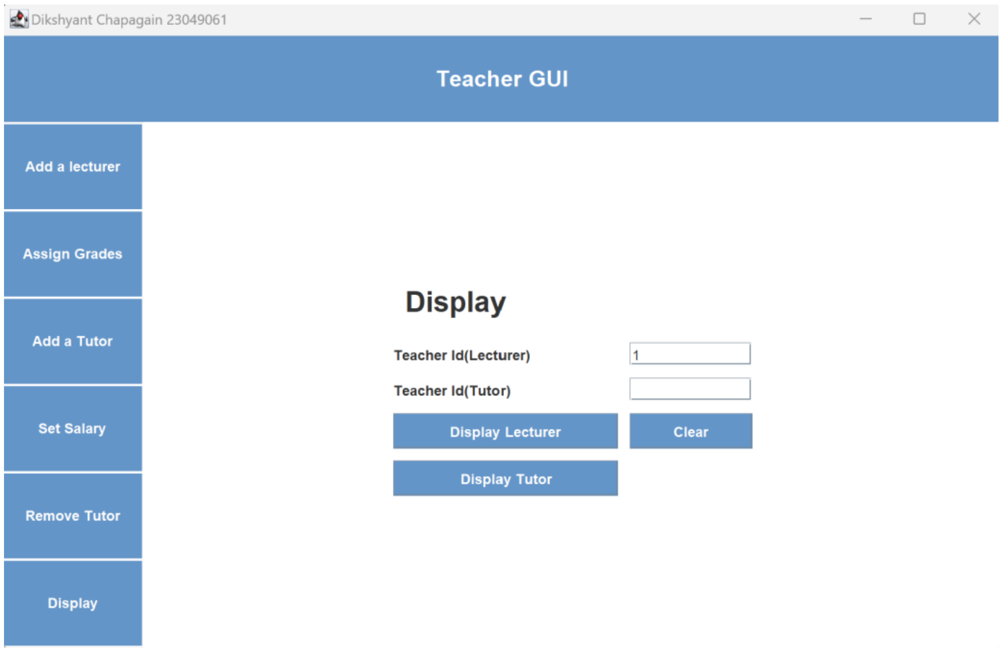
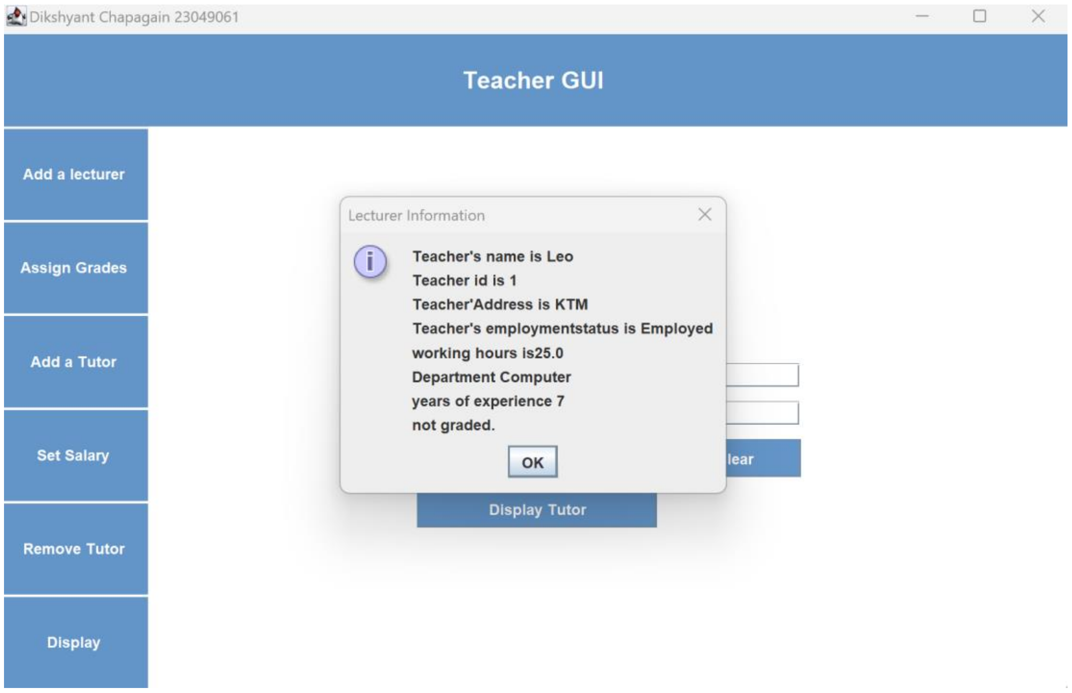


Figure 12 Test 2.1.4pressing the display button and entering the teacher Id of Leo and pressing the Display Lecturer button



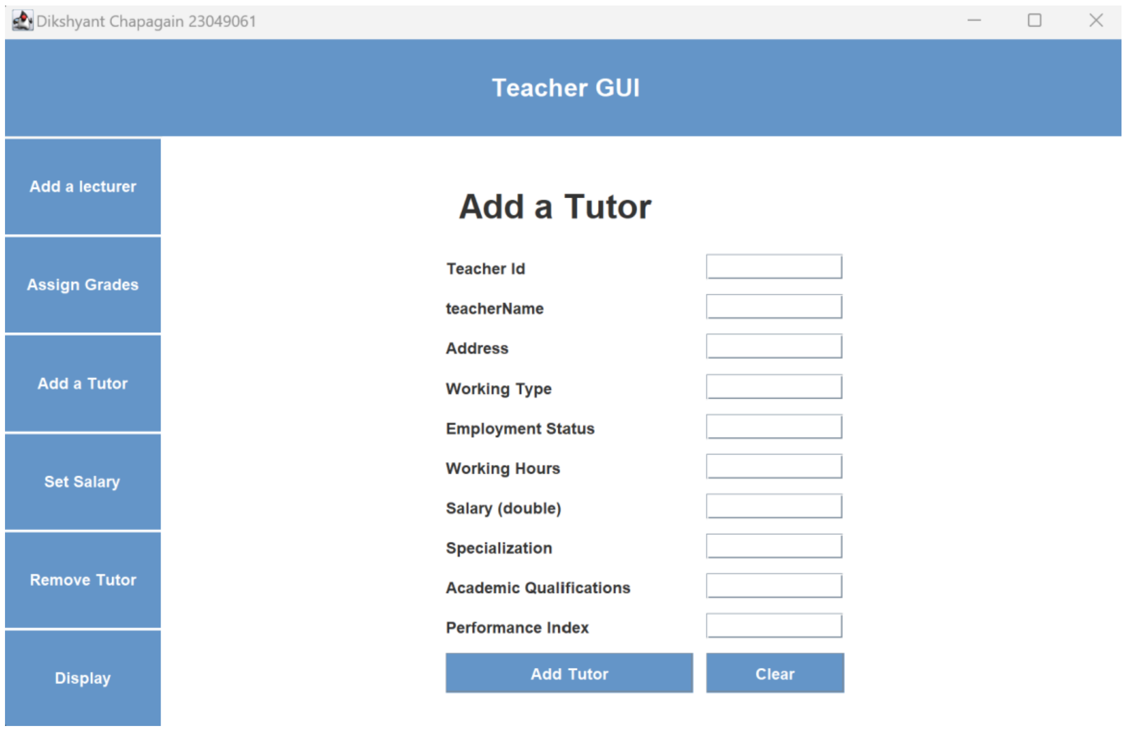
```
Teacher's name is Leo
Teacher id is 1
Teacher'Address is KTM
Teacher's employmentstatus is Employed
working hours is25.0
Department Computer
years of experience 7
not graded.
```

Figure 13 2.1.5 The details of the Lecturer was shown

b.Add a Tutor

Test No	2.2
Objective	To show evidence of adding a Tutor
Action	• The button add Tutor was clicked. • The necessary values for teacherId, teacherName, Address, WorkingType Employment Status, working hours ,Salary, Specialization, Academic Qualification and Performance Index was enterted • The button Add Tutor was clicked • The Display Button was clicked • And the teacherId of the lecturer was entered in the teacherId (Tutor) textfield and “Display Tutor” Button was clicked.
Expected Result	• A popup showing the message “Lecturer Added with Id 1should be shown” after clicking the Add Lecturer Button along with the entered values • After clicking “Display Lecturer” button the necessary Values of the tutor will be shown
Actual Result	• A popup showing the message “Lecturer Added with Id “1was shown after clicking the Add Lecturer Button along with the entered values • After clicking “Display Lecturer” button the necessary Values of the tutor was shown
Conclusion	The Test was successful

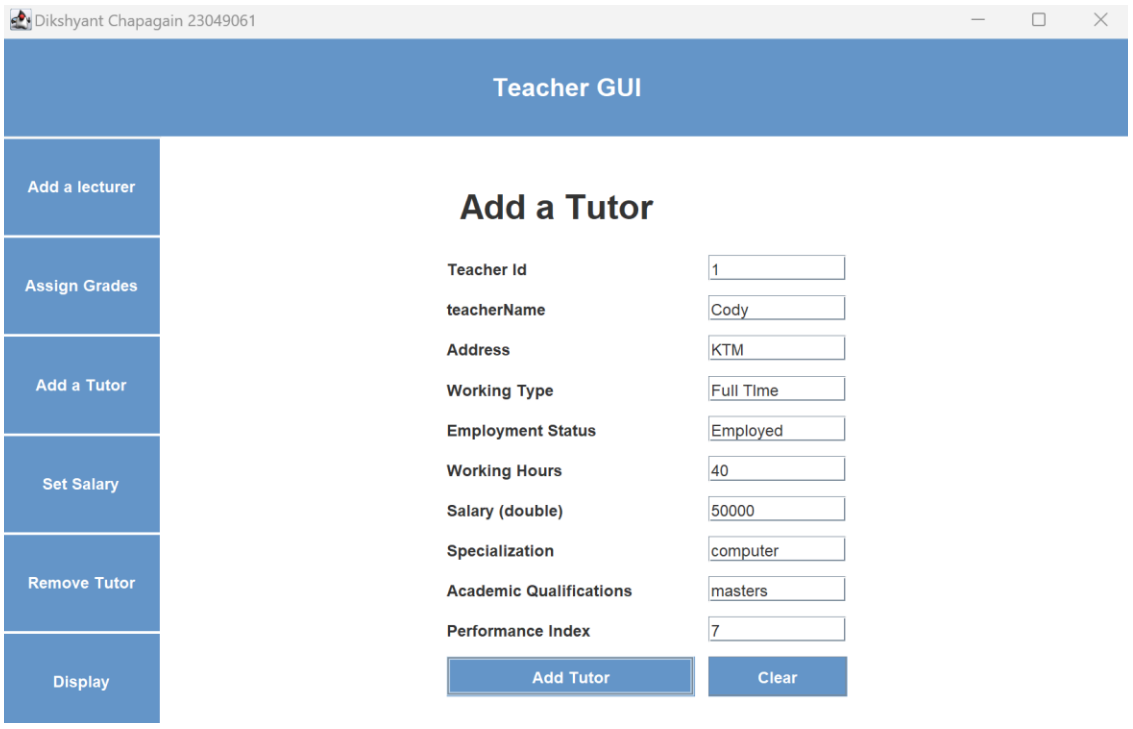
Table 3 2.2



Add a Tutor

Teacher Id	
teacherName	
Address	
Working Type	
Employment Status	
Working Hours	
Salary (double)	
Specialization	
Academic Qualifications	
Performance Index	

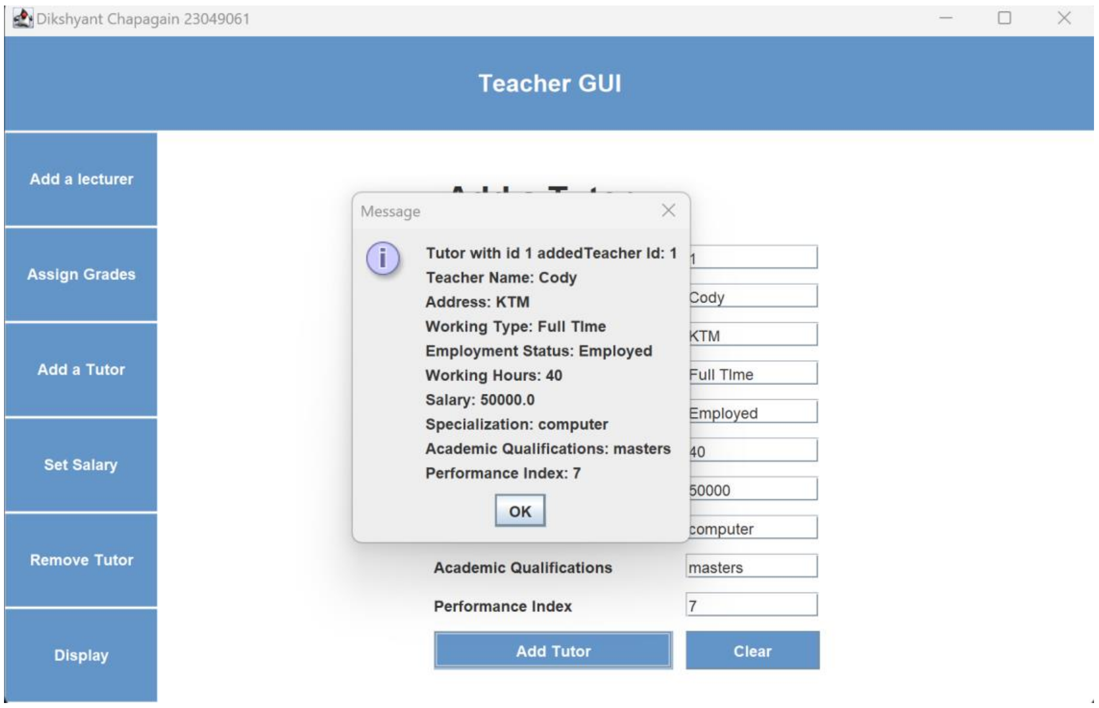
Figure 14 Test 2.2.1Pressing Add a Tutor Button and the Add a Tutor panel shows



Add a Tutor

Teacher Id	1
teacherName	Cody
Address	KTM
Working Type	Full TIme
Employment Status	Employed
Working Hours	40
Salary (double)	50000
Specialization	computer
Academic Qualifications	masters
Performance Index	7

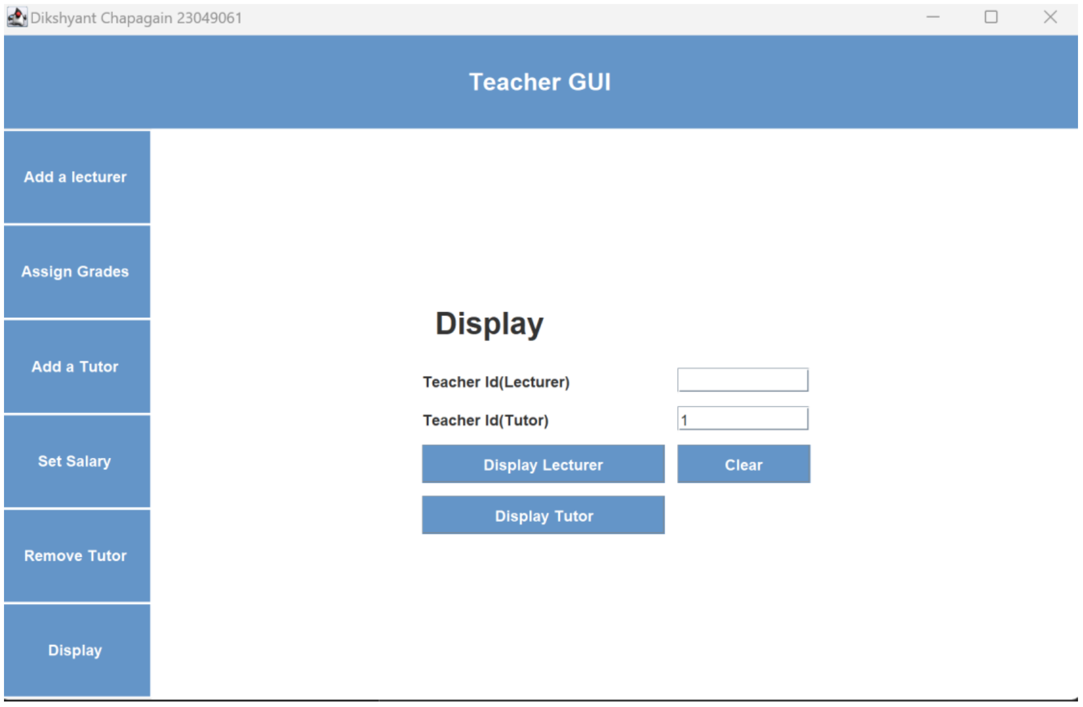
Figure 15 Test 2.2.2Entering the necessary values in the textfields



Tutor with id 1 addedTeacher Id: 1
Teacher Name: Cody
Address: KTM
Working Type: Full TIme
Employment Status: Employed
Working Hours: 40
Salary: 50000.0
Specialization: computer
Academic Qualifications: masters
Performance Index: 7

	1
	Cody
	KTM
	Full TIme
	Employed
	40
	50000
	computer
Academic Qualifications	masters
Performance Index	7

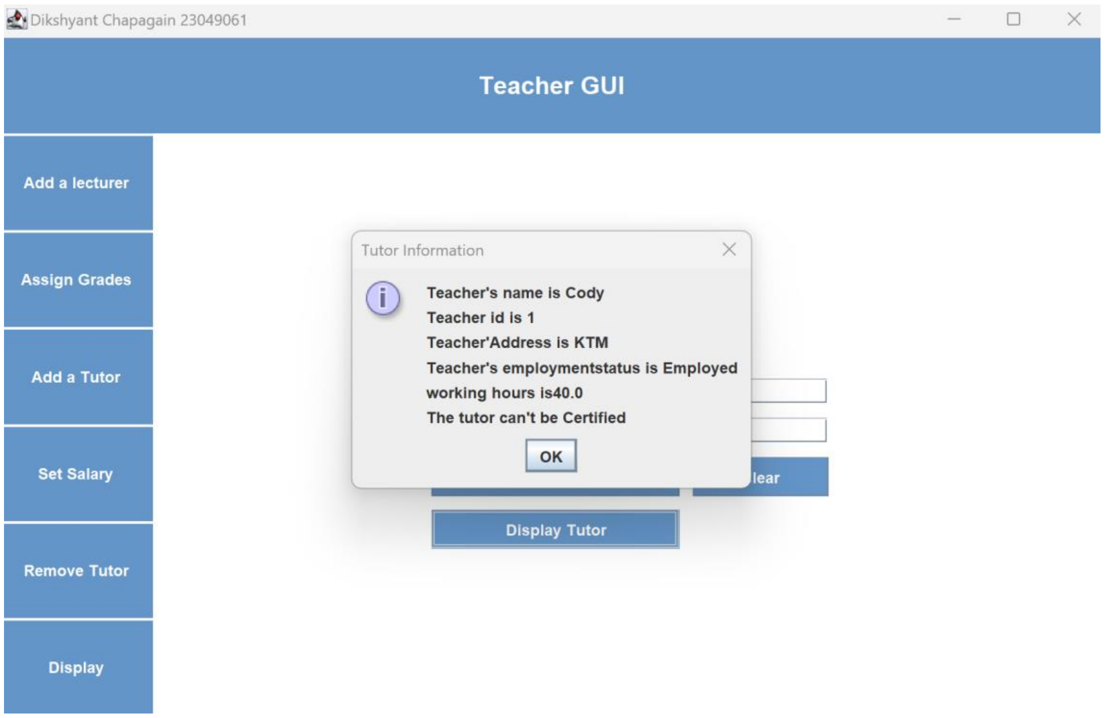
Figure 16 Test 2.2.3 Pressing the Add Tutor and a popup shows which contains the entered values by the user



Display

Teacher Id(Lecturer)	
Teacher Id(Tutor)	1

Figure 17 Test 2.2.4Adding the teacher Id of Cody and pressing DisplayTutor



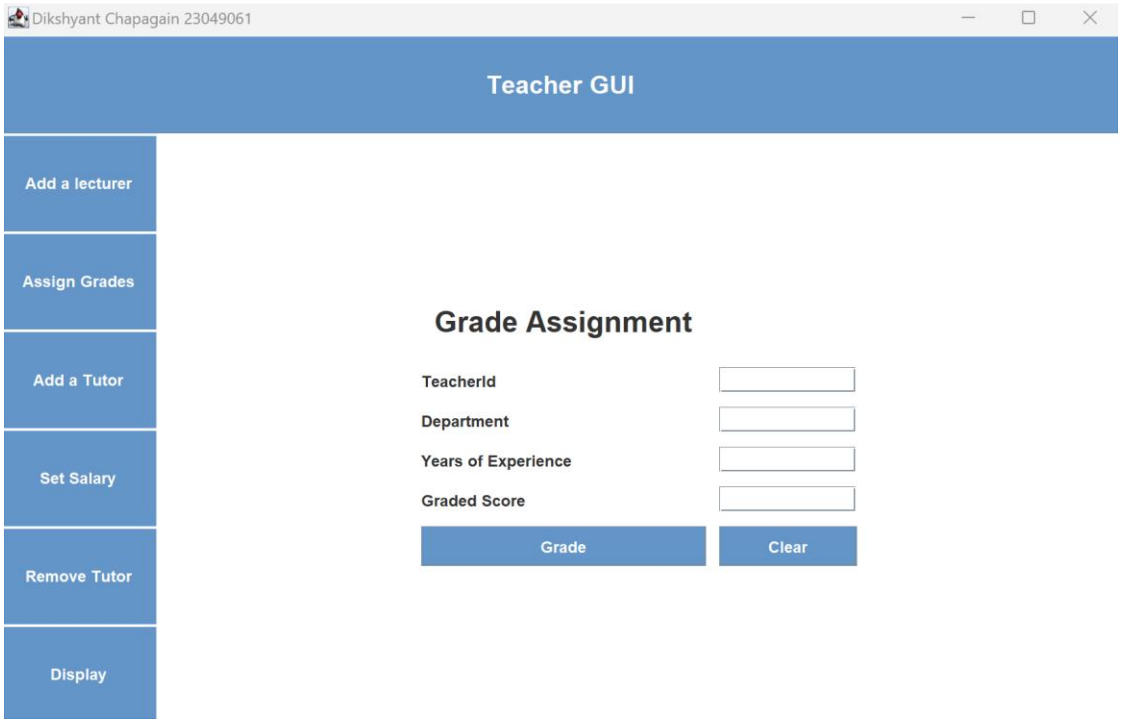
```
Teacher's name is Cody
Teacher id is 1
Teacher'Address is KTM
Teacher's employmentstatus is Employed
working hours is40.0
The tutor can't be Certified
```

Figure 18 Test 2.2.5Information of the Tutor is shown

c.Grade Assignments from Lecturer

Test No	2.3
Objective	To show evidence of Grading Assignments from Lecturer
Action	• The button Assign Grades was clicked. • The necessary values for teacherId, Department ,Years of experience Graded Score was enterted • The button Grade was clicked
Expected Result	• A popup showing the entered values should be shown after clicking the Grade Button • The grade according to the Graded Score should appear on the terminal
Actual Result	• A popup showing the entered values was shown after clicking the Grade Button • The grade according to the Graded Score appeared on the terminal
Conclusion	The Test was successful

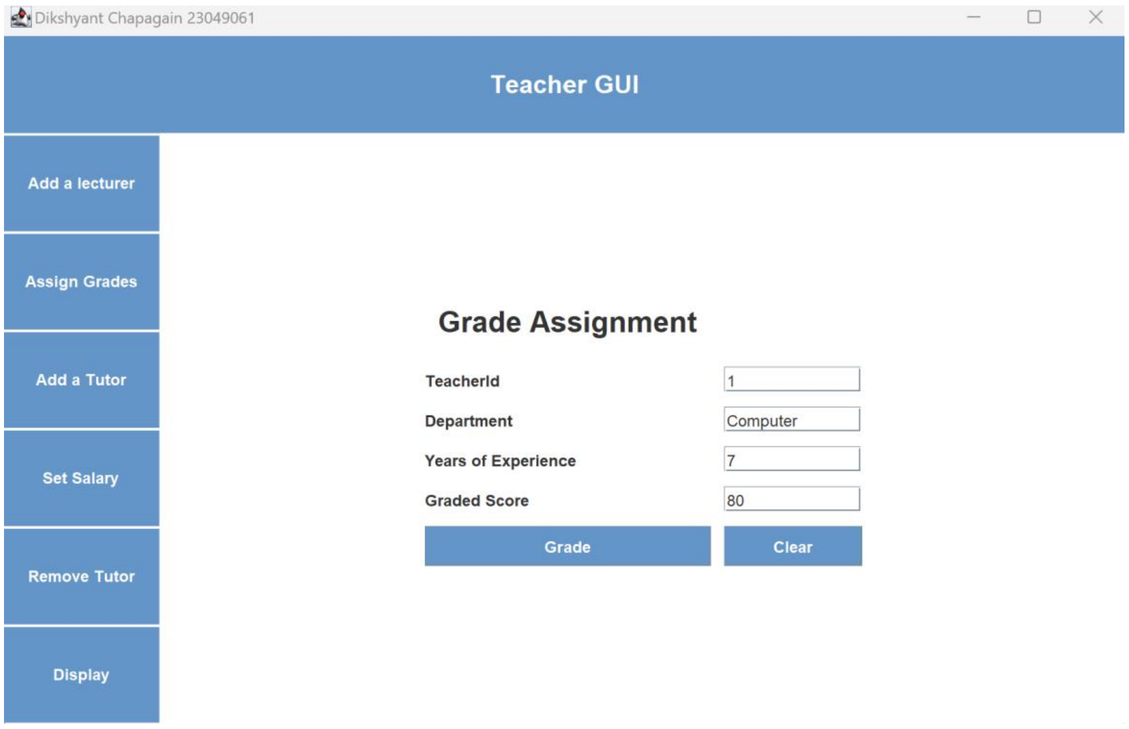
Table 4 Test 2.3



Grade Assignment

TeacherId	
Department	
Years of Experience	
Graded Score	

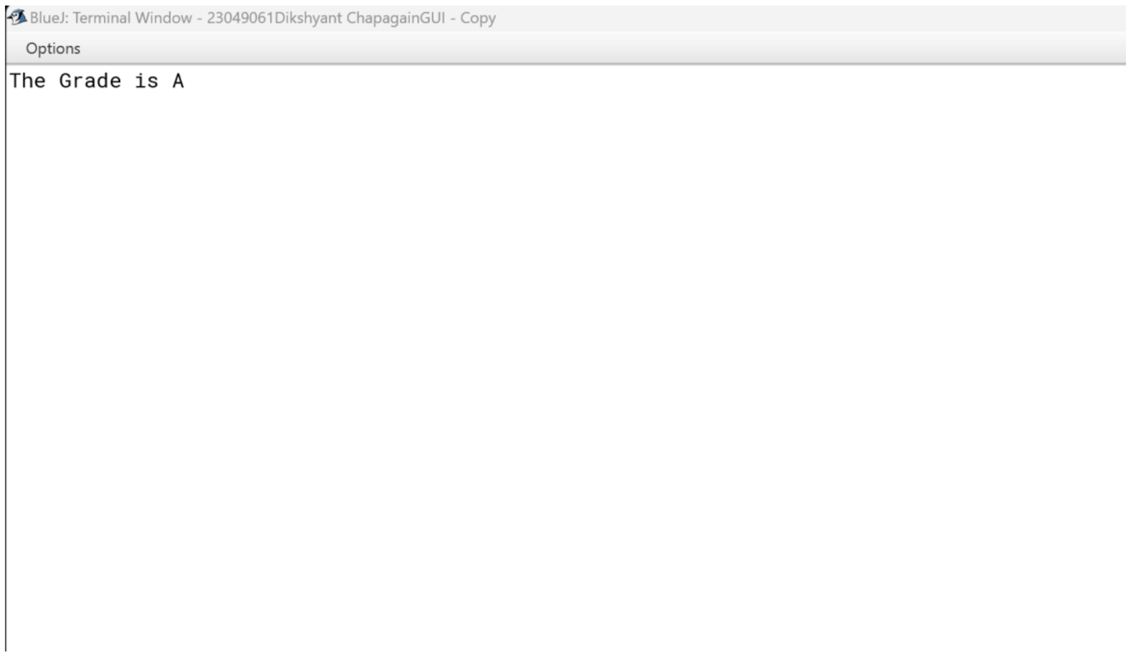
Figure 19 Test 2.3.1clicking the Assign Grades and Grade Assignment panel shows up



Grade Assignment

TeacherId	1
Department	Computer
Years of Experience	7
Graded Score	80

Figure 20 Test 2.3.2 Entering the Id of Leo and adding necessary values


BlueJ: Terminal Window - 23049061Dikshyant ChapagainGUI - Copy
Options

```
The Grade is A
```

Figure 21 Test 2.3.3The graded score in the terminal

d.Set the salary

Test No	2.4
Objective	To show evidence of Setting the Salary
Action	• The button add Tutor was clicked. • The necessary values for PerformanceIndex, TeacherId and new Salary was enterted • The button Set was clicked • The Display Button was clicked • And the teacherId of the Tutor was entered in the teacherId (Tutor) textfield and “Display Tutor” Button was clicked.
Expected Result	• A popup showing the entered values should be shown after clicking the Set Button • After clicking “Display Tutor” button the updated Salary should be shown
Actual Result	• A popup showing the entered values was shown after clicking the Set Button After clicking “Display Tutor” button the updated Salary was shown
Conclusion	The Test was successful

Table 5 Test 2.4

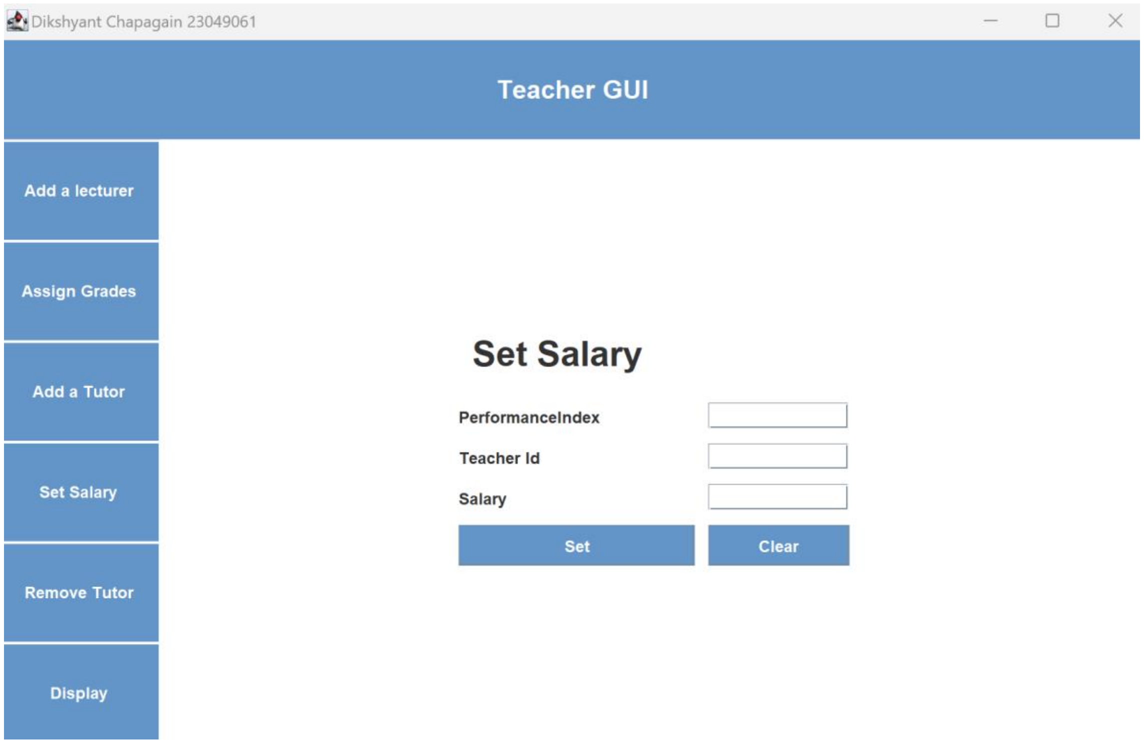
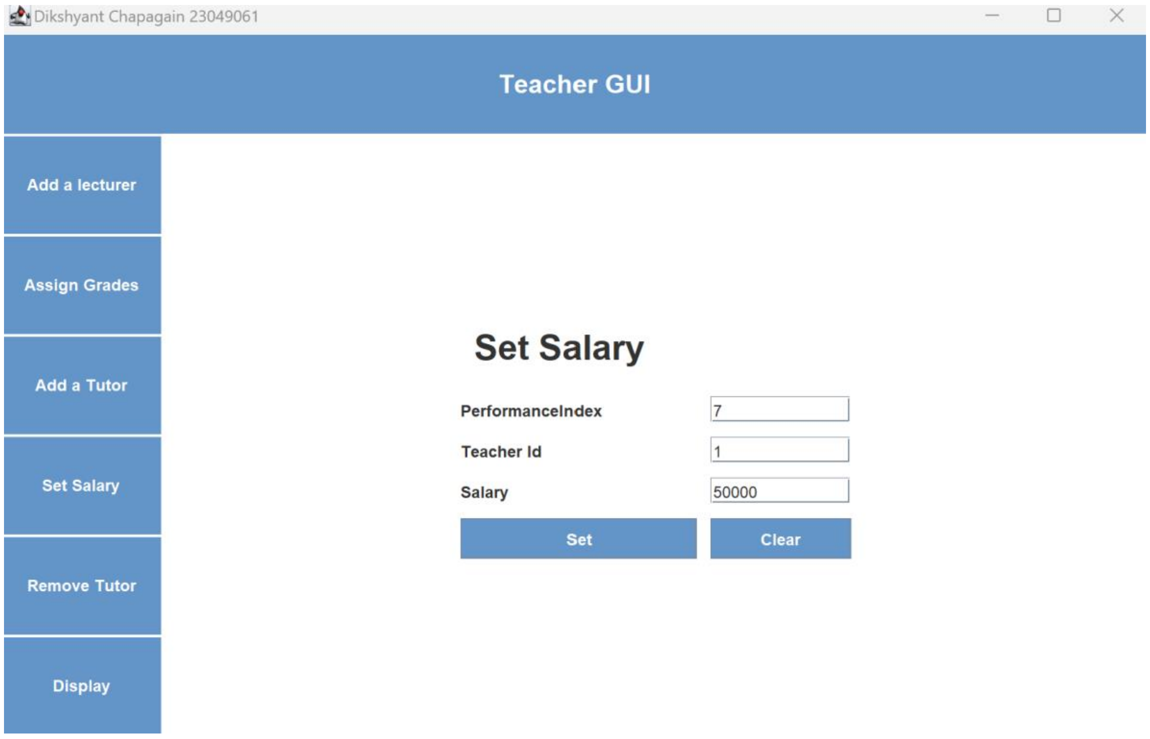
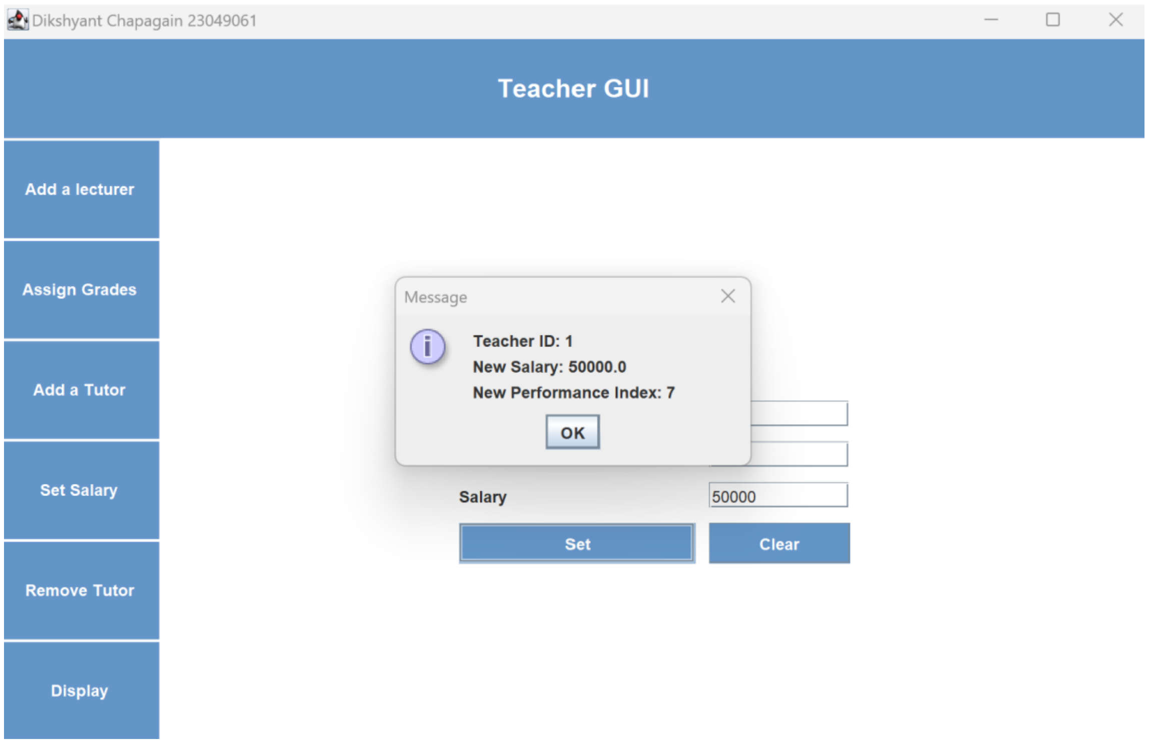


Figure 22 Test 2.4.1Clicking the Set Salary button and Set Salary panel shows up



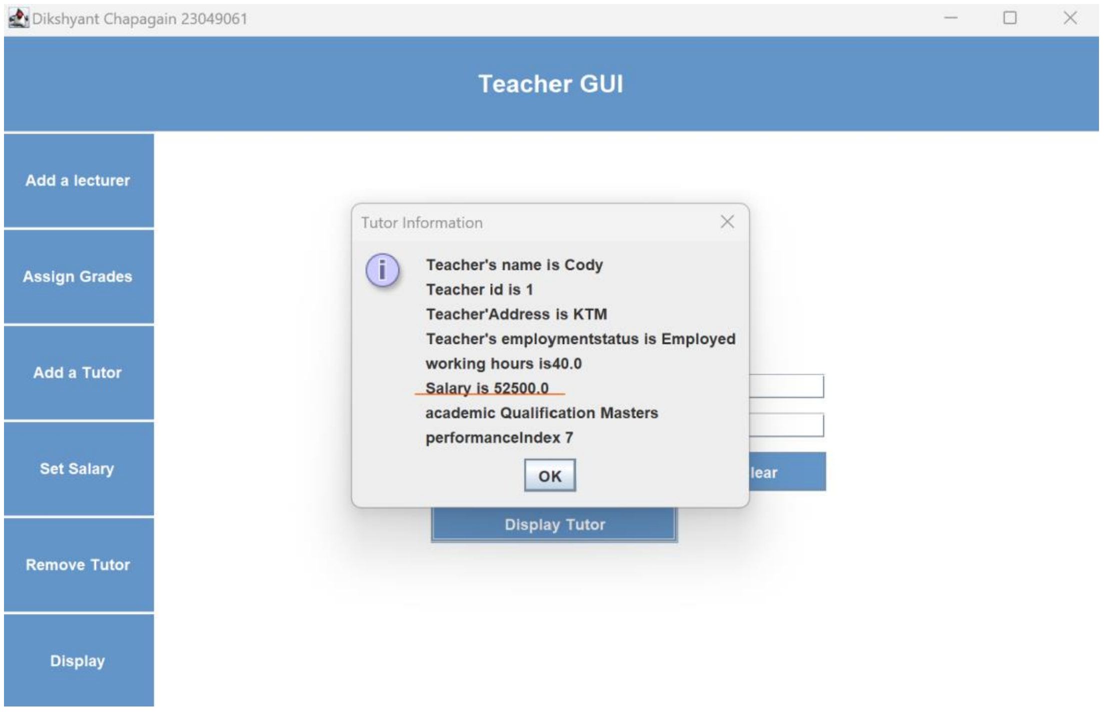
PerformanceIndex 7
Teacher Id 1
Salary 50000

Figure 23 Test 2.4.2Entering necessary values



Teacher ID: 1
New Salary: 50000.0
New Performance Index: 7

Figure 24 Test 2.4.2 popup showing the entered values



Teacher's name is Cody
Teacher id is 1
Teacher'Address is KTM
Teacher's employmentstatus is Employed
working hours is40.0
Salary is 52500.0
academic Qualification Masters
performanceIndex 7

Figure 25 Test 2.4.3Updated salary when displaying the Tutor

e.Remove the tutor

Test No	2.5
Objective	To show evidence of removing a Tutor
Action	• The button Remove Tutor was clicked. • The necessary values for teacherId was entered • The button Remove was clicked • The Display Button was clicked • And the teacherId of the lecturer was entered in the “teacherId (Tutor) “Text Field and “Display Tutor” Button was clicked.
Expected Result	• A popup showing the message “Tutor Removed Should be shown • After clicking “Display Lecturer” button appropriate message stating that teacher cannot be found should be shown
Actual Result	• A popup showing the message “Tutor Removed.” was shown. • After clicking “Display Lecturer” button appropriate message stating that teacher cannot be found was shown
Conclusion	The Test was successful

Table 6 Test 2.5

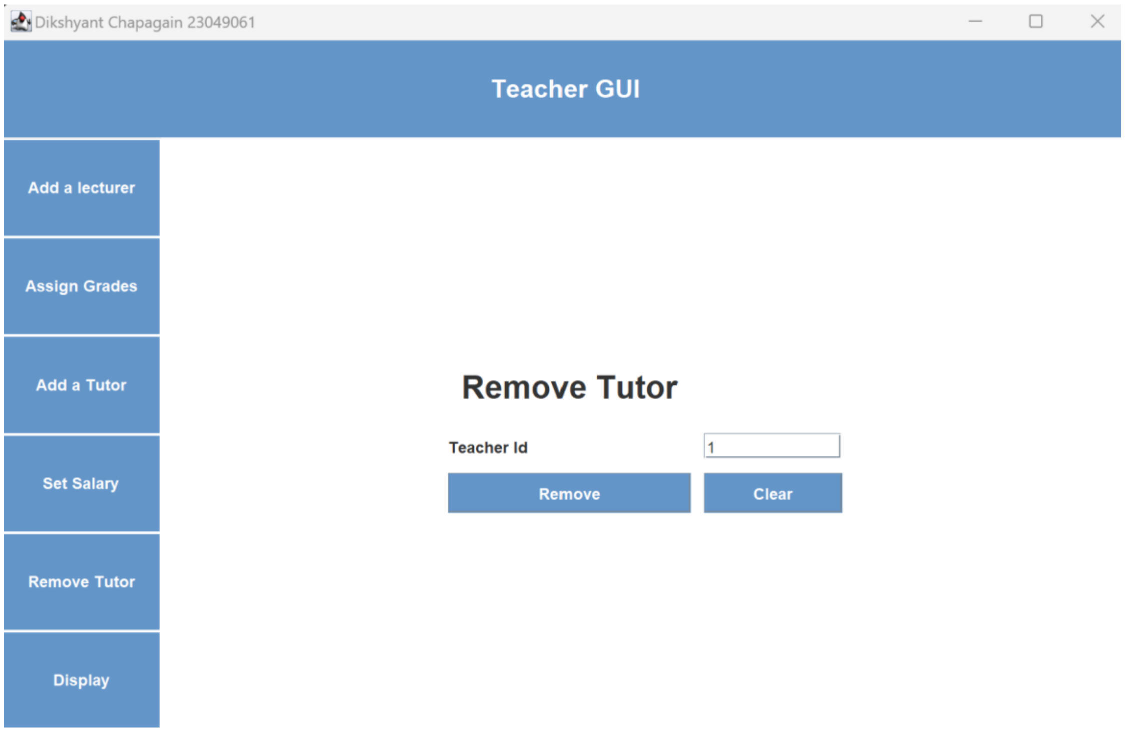
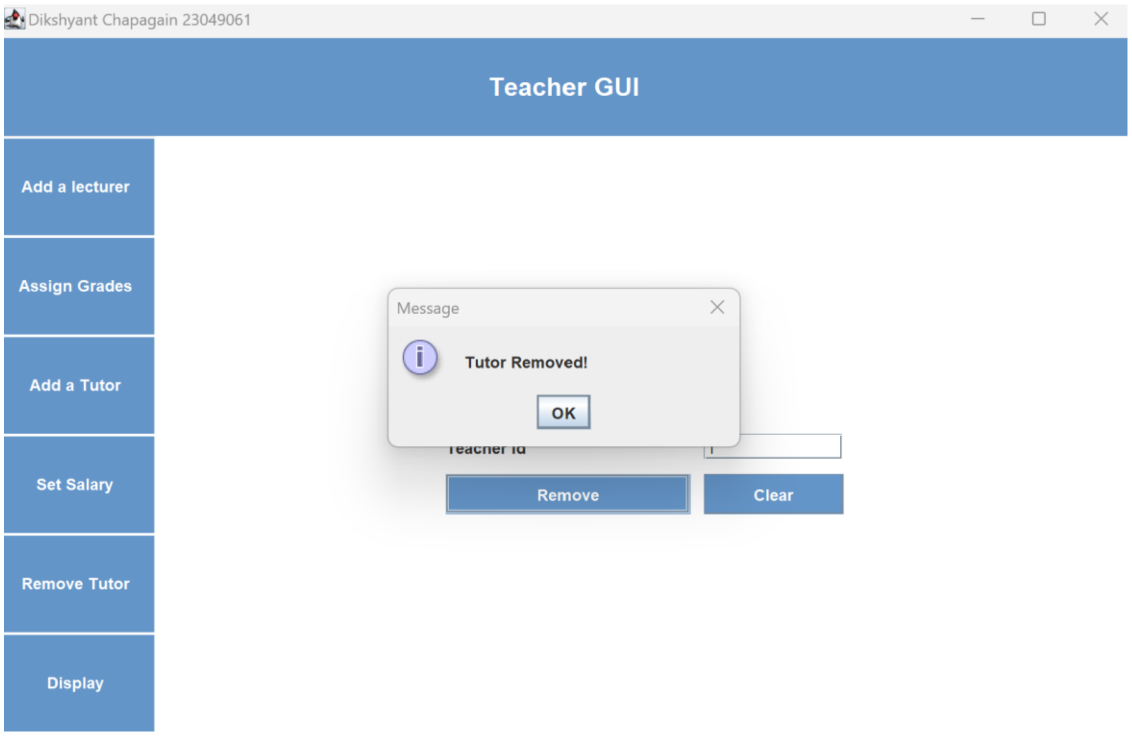


Figure 26 Test 2.5.1Clicking the Remove Tutor and Remove Tutor panel shows



Tutor Removed!

Figure 27 Test 2.5.2 Adding the teacher id of cody and pressing Remove

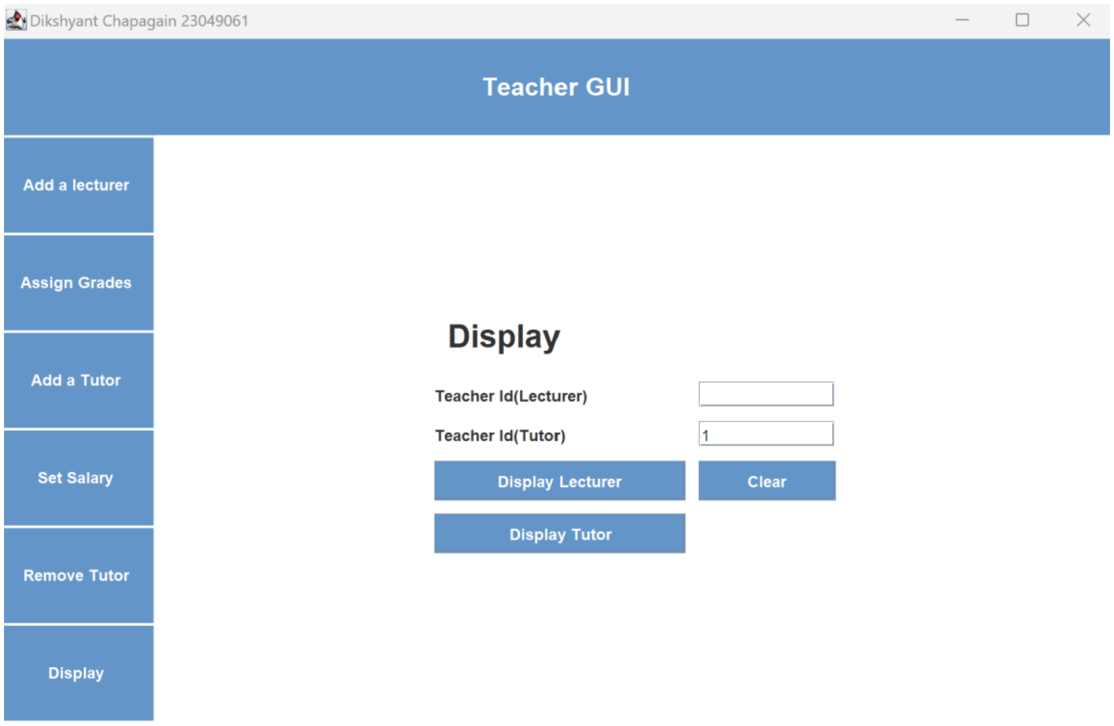
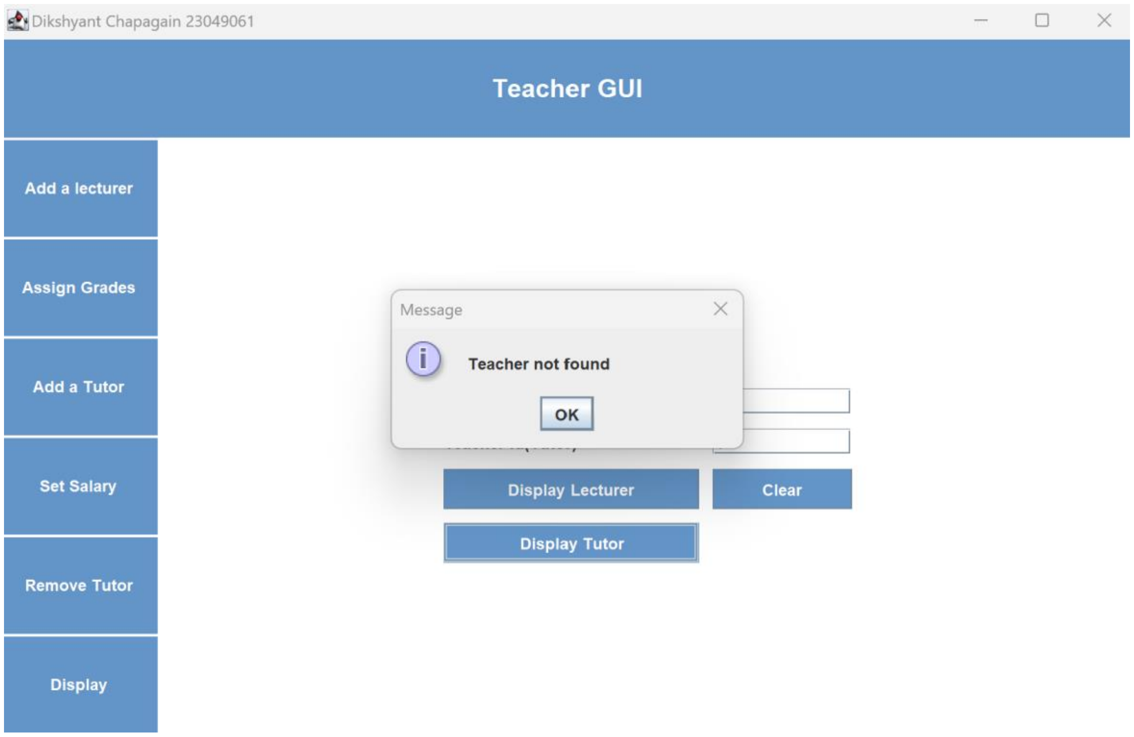


Figure 28 Test 2.5.3 entering Cody's teacher id in display panel



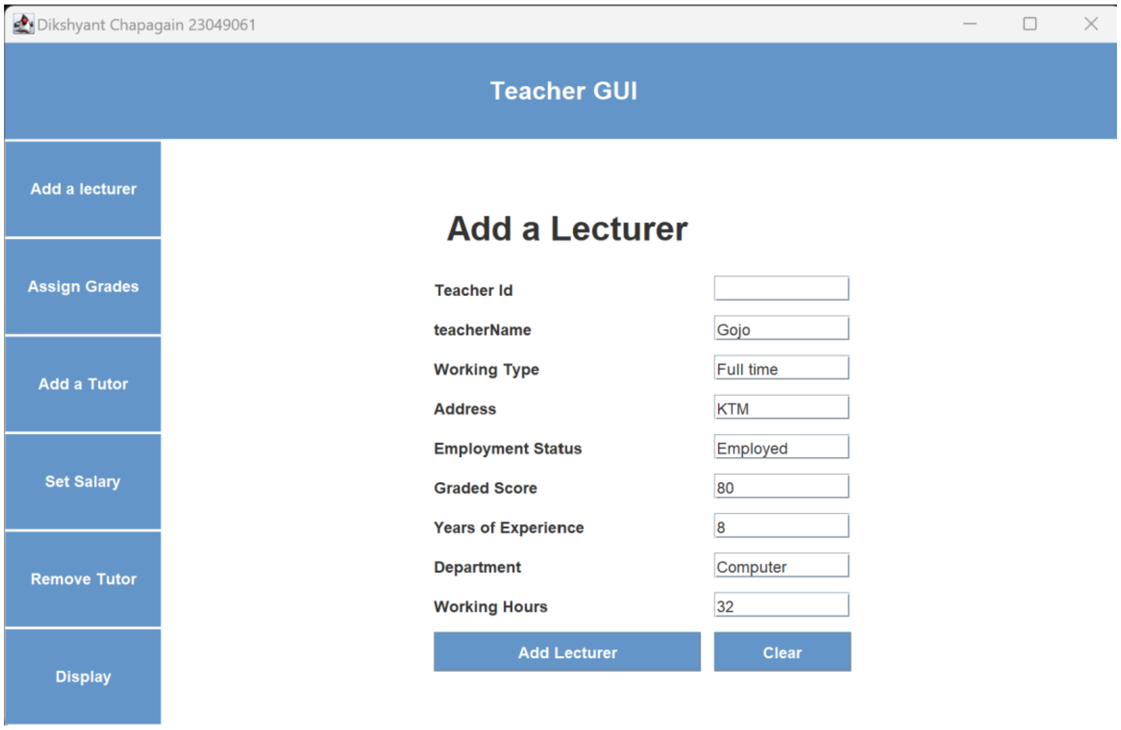
Teacher not found

Figure 29 Test 2.5.4 techer not found since it is removed

3. Test that appropriate dialog boxes appear when unsuitable values are entered for the Teacher ID

Test No	3
Objective	To test appropriate Dialog box when unsuitable values are entered for teacher ID
Action	• The button add Lecturer was clicked. • String values were entered instead of an integer • No value was entered
Expected Result	• A popup showing the message “Please enter valid input values” should appear after passing String value. • A popup showing the message Empty Fields Found!! Please fill all the fields " should appear
Actual Result	• A popup showing the message “Please enter valid input values” appeared after passing String value. A popup showing the message Empty Fields Found!! Please fill all the fields “appeared
Conclusion	The Test was successful

Table 7Test 3

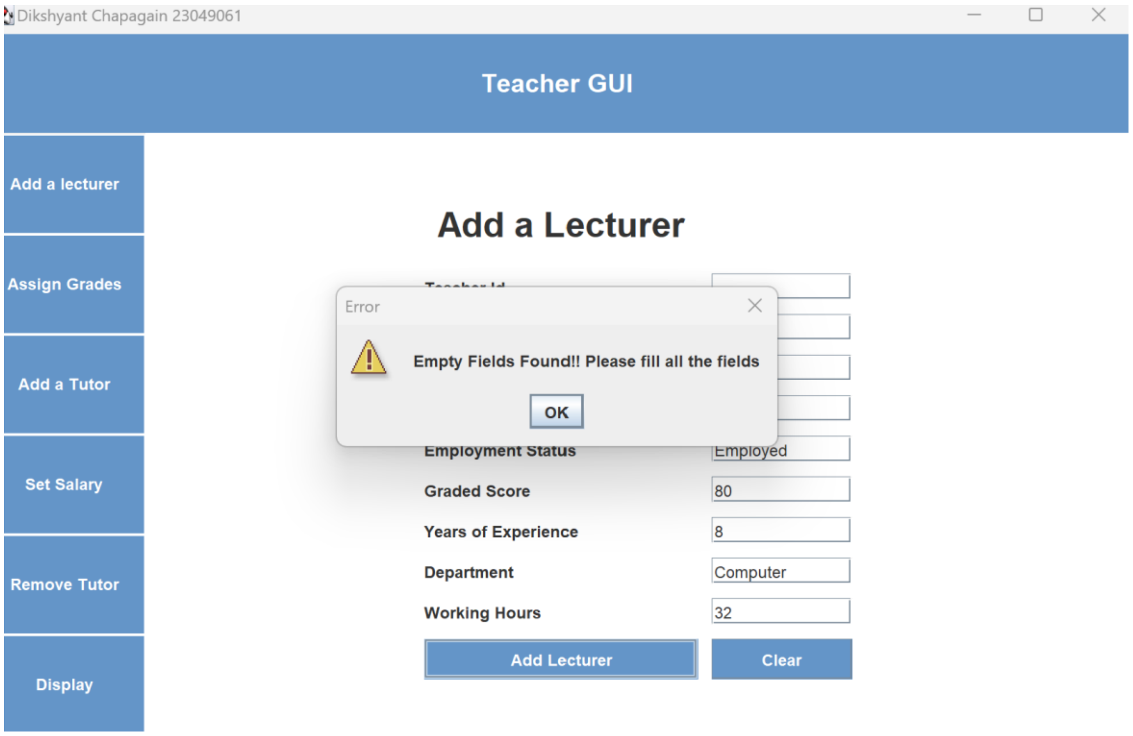


Dikshyant Chapagain 23049061

Add a Lecturer

Teacher Id	
teacherName	Gojo
Working Type	Full time
Address	KTM
Employment Status	Employed
Graded Score	80
Years of Experience	8
Department	Computer
Working Hours	32

Figure 30 Test 3.1: Entering no values in Add a Lecturer Panel



Dikshyant Chapagain 23049061

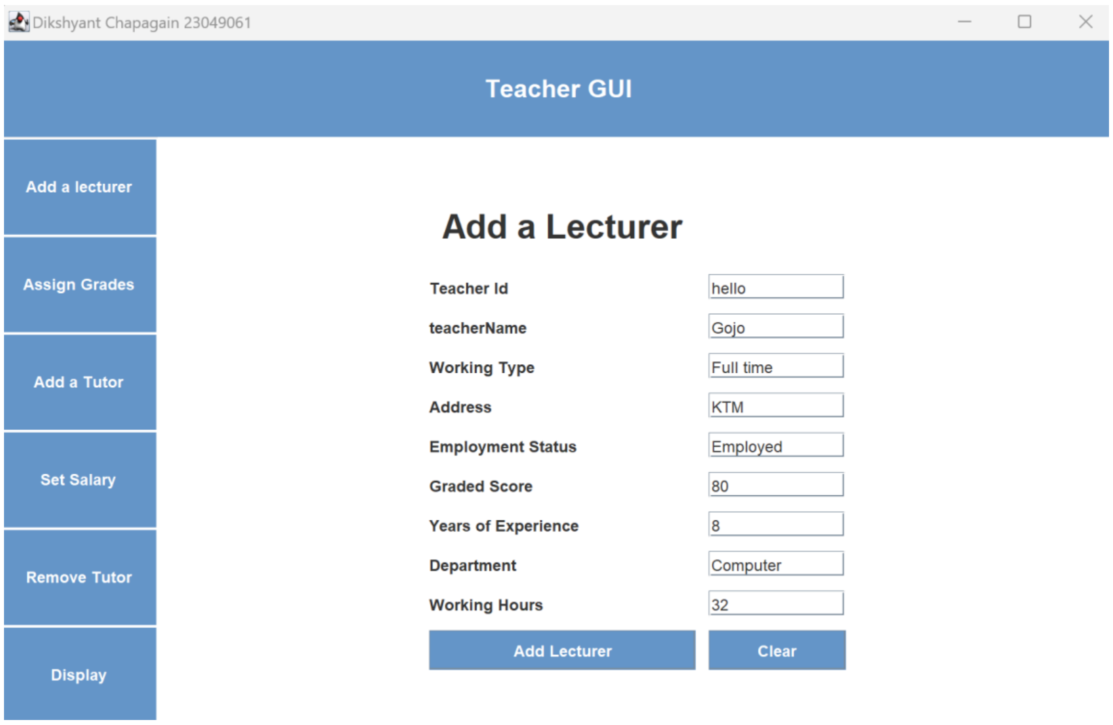
Add a Lecturer

Error

Empty Fields Found!! Please fill all the fields

Employment Status	Employed
Graded Score	80
Years of Experience	8
Department	Computer
Working Hours	32

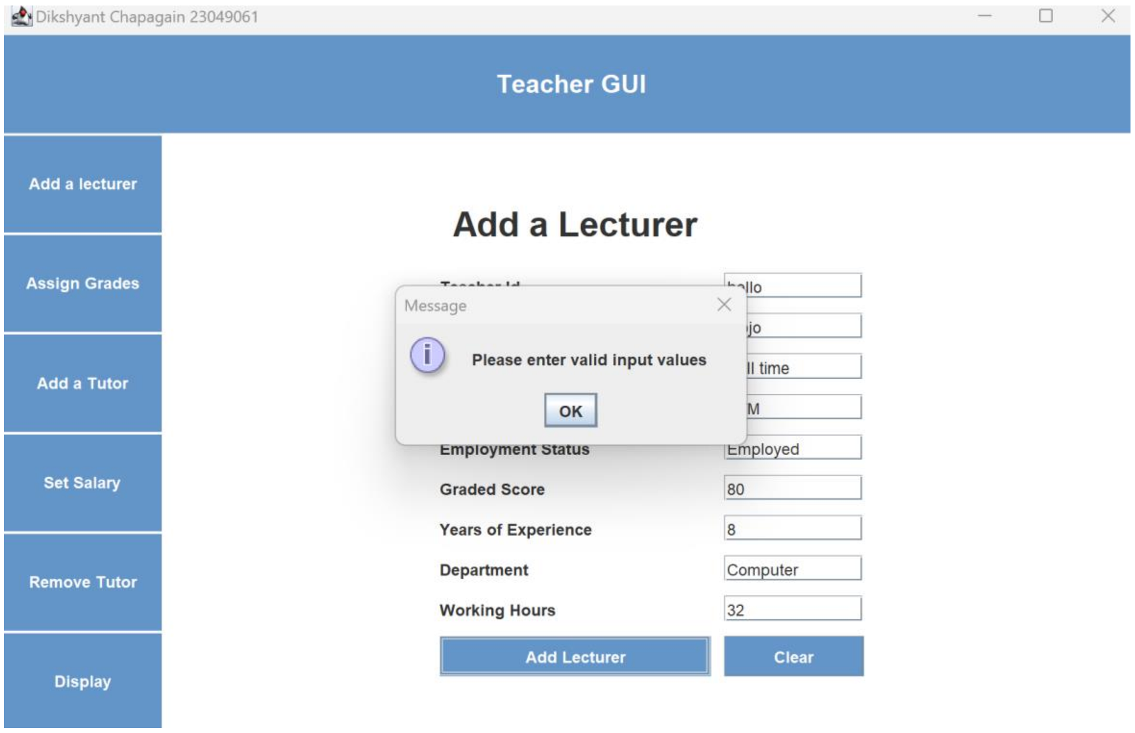
Figure 31 Test 3.2 Appropiate message display



Add a Lecturer

Teacher Id	hello
teacherName	Gojo
Working Type	Full time
Address	KTM
Employment Status	Employed
Graded Score	80
Years of Experience	8
Department	Computer
Working Hours	32

Figure 32 Test 3.3 Adding a string instead of integer


Dikshyant Chapagain 23049061

Teacher GUI

Add a Lecturer

Teacher Id	hello
	jo
	ll time
	M
Employment Status	Employed
Graded Score	80
Years of Experience	8
Department	Computer
Working Hours	32

Message

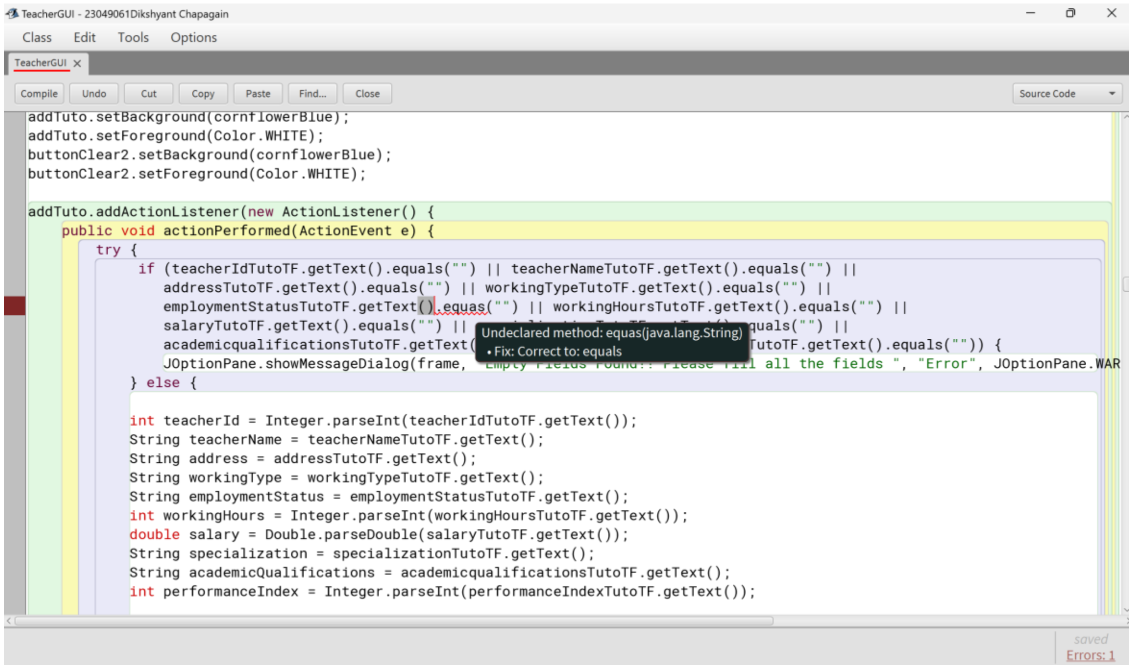
Please enter valid input values

Figure 33 Test 3.4 popup showing appropiate message

Error and Debugging

Error 1(Syntax Error):

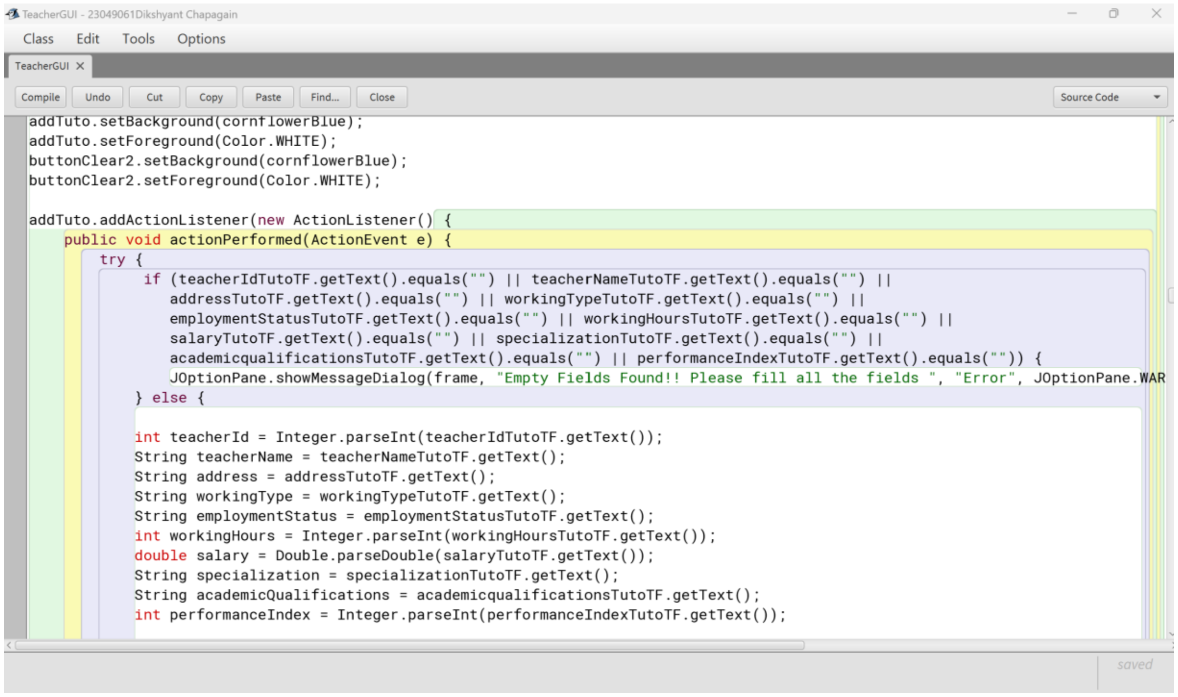
Error	Undeclared method equas(java.land.String)
Cause	Spelling error of the “equals” method to “equas”
Fix	“equas” was changed to “equals”


TeacherGUI - 23049061Dikshyant Chapagain
Class Edit Tools Options
TeacherGUI
Compile Undo Cut Copy Paste Find... Close
Source Code
saved
Errors: 1

```
addTuto.setBackground(cornflowerBlue);
addTuto.setForeground(Color.WHITE);
buttonClear2.setBackground(cornflowerBlue);
buttonClear2.setForeground(Color.WHITE);

addTuto.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        try {
             if (teacherIdTutoTF.getText().equals("") || teacherNameTutoTF.getText().equals("") ||
                addressTutoTF.getText().equals("") || workingTypeTutoTF.getText().equals("") ||
                employmentStatusTutoTF.getText().equas("") || workingHoursTutoTF.getText().equals("") ||
                salaryTutoTF.getText().equals("") ||                              uals("") ||
                academicqualificationsTutoTF.getText(                           TutoTF.getText().equals("")) {
                JOptionPane.showMessageDialog(frame, "Empty Fields Found!! Please fill all the fields ", "Error", JOptionPane.WAR
           } else {

           int teacherId = Integer.parseInt(teacherIdTutoTF.getText());
           String teacherName = teacherNameTutoTF.getText();
           String address = addressTutoTF.getText();
           String workingType = workingTypeTutoTF.getText();
           String employmentStatus = employmentStatusTutoTF.getText();
           int workingHours = Integer.parseInt(workingHoursTutoTF.getText());
           double salary = Double.parseDouble(salaryTutoTF.getText());
           String specialization = specializationTutoTF.getText();
           String academicQualifications = academicqualificationsTutoTF.getText();
           int performanceIndex = Integer.parseInt(performanceIndexTutoTF.getText());
```

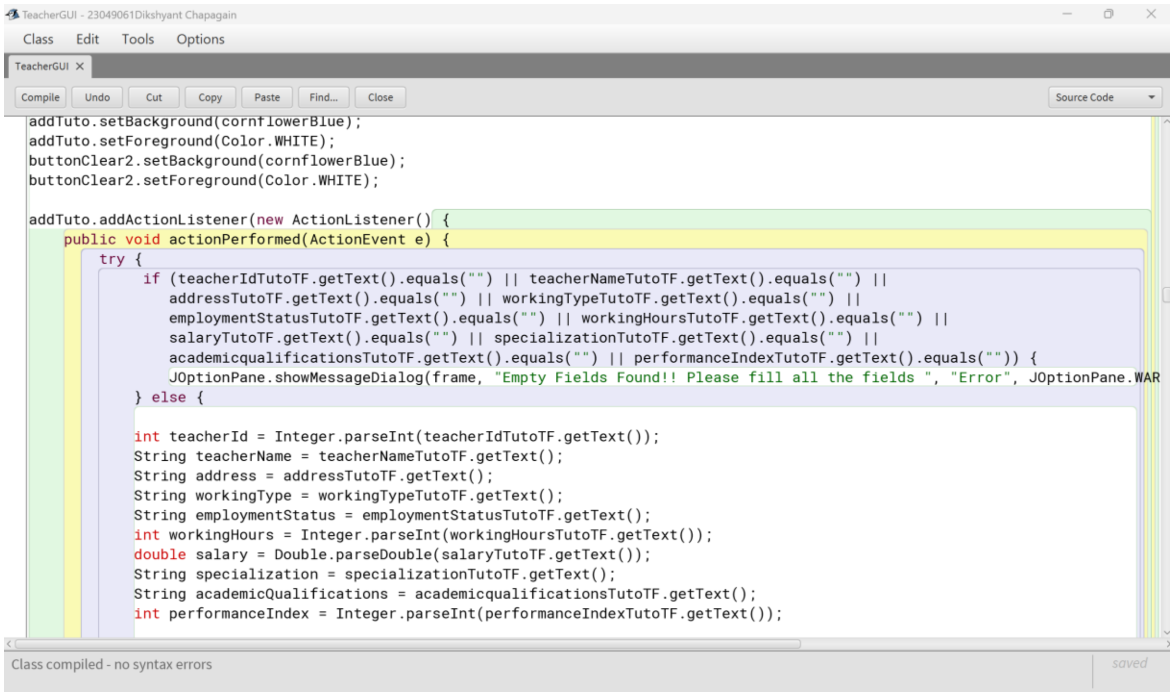
```
Undeclared method: equas(java.lang.String)
• Fix: Correct to: equals
```

Figure 34 Syntax Error


TeacherGUI - 23049061Dikshyant Chapagain
Class Edit Tools Options
TeacherGUI
Compile Undo Cut Copy Paste Find... Close
Source Code
saved

```
addTuto.setBackground(cornflowerBlue);
addTuto.setForeground(Color.WHITE);
buttonClear2.setBackground(cornflowerBlue);
buttonClear2.setForeground(Color.WHITE);

addTuto.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        try {
             if (teacherIdTutoTF.getText().equals("") || teacherNameTutoTF.getText().equals("") ||
                addressTutoTF.getText().equals("") || workingTypeTutoTF.getText().equals("") ||
                employmentStatusTutoTF.getText().equals("") || workingHoursTutoTF.getText().equals("") ||
                salaryTutoTF.getText().equals("") || specializationTutoTF.getText().equals("") ||
                academicqualificationsTutoTF.getText().equals("") || performanceIndexTutoTF.getText().equals("")) {
                JOptionPane.showMessageDialog(frame, "Empty Fields Found!! Please fill all the fields ", "Error", JOptionPane.WAR
            } else {

            int teacherId = Integer.parseInt(teacherIdTutoTF.getText());
            String teacherName = teacherNameTutoTF.getText();
            String address = addressTutoTF.getText();
            String workingType = workingTypeTutoTF.getText();
            String employmentStatus = employmentStatusTutoTF.getText();
            int workingHours = Integer.parseInt(workingHoursTutoTF.getText());
            double salary = Double.parseDouble(salaryTutoTF.getText());
            String specialization = specializationTutoTF.getText();
            String academicQualifications = academicqualificationsTutoTF.getText();
            int performanceIndex = Integer.parseInt(performanceIndexTutoTF.getText());
```

Figure 35Fixing Syntax Error


TeacherGUI - 23049061Dikshyant Chapagain
Class Edit Tools Options
TeacherGUI
Compile Undo Cut Copy Paste Find... Close
Source Code
saved

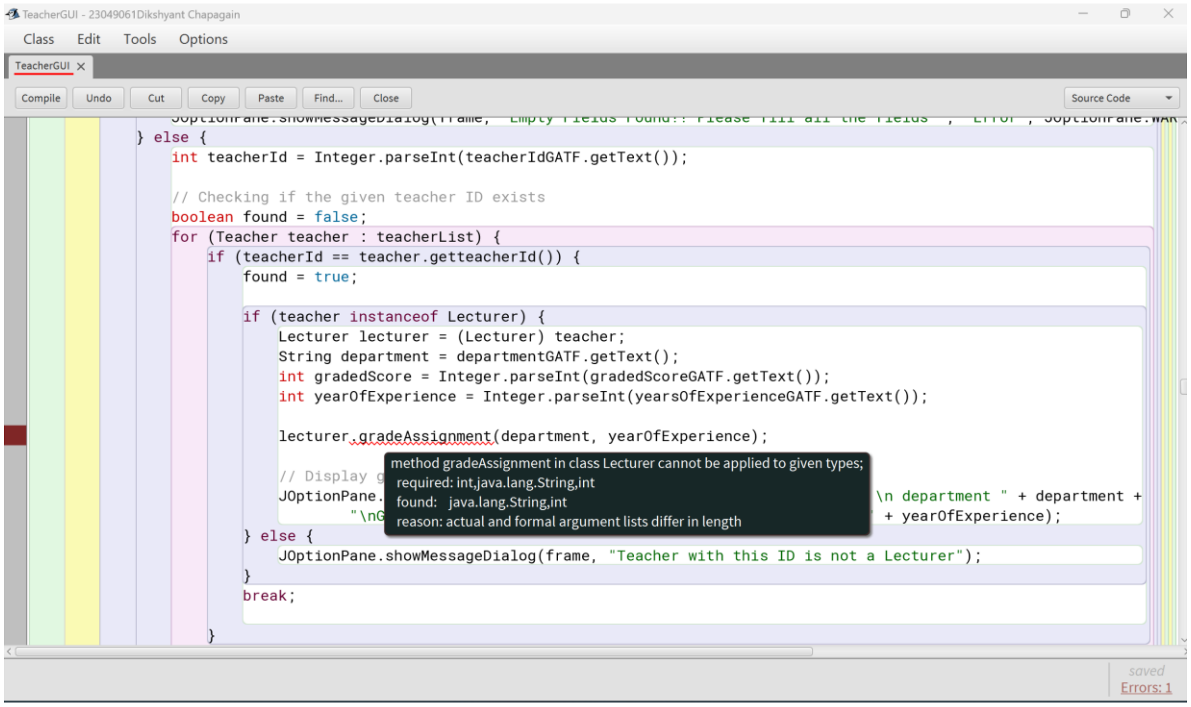
```
addTuto.setBackground(cornflowerBlue);
addTuto.setForeground(Color.WHITE);
buttonClear2.setBackground(cornflowerBlue);
buttonClear2.setForeground(Color.WHITE);

addTuto.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        try {
             if (teacherIdTutoTF.getText().equals("") || teacherNameTutoTF.getText().equals("") ||
                addressTutoTF.getText().equals("") || workingTypeTutoTF.getText().equals("") ||
                employmentStatusTutoTF.getText().equals("") || workingHoursTutoTF.getText().equals("") ||
                salaryTutoTF.getText().equals("") || specializationTutoTF.getText().equals("") ||
                academicqualificationsTutoTF.getText().equals("") || performanceIndexTutoTF.getText().equals("")) {
                JOptionPane.showMessageDialog(frame, "Empty Fields Found!! Please fill all the fields ", "Error", JOptionPane.WAR
            } else {

            int teacherId = Integer.parseInt(teacherIdTutoTF.getText());
            String teacherName = teacherNameTutoTF.getText();
            String address = addressTutoTF.getText();
            String workingType = workingTypeTutoTF.getText();
            String employmentStatus = employmentStatusTutoTF.getText();
            int workingHours = Integer.parseInt(workingHoursTutoTF.getText());
            double salary = Double.parseDouble(salaryTutoTF.getText());
            String specialization = specializationTutoTF.getText();
            String academicQualifications = academicqualificationsTutoTF.getText();
            int performanceIndex = Integer.parseInt(performanceIndexTutoTF.getText());
```

Class compiled - no syntax errors

Figure 36Syntax Error fixed

Error 2 (Semantic error):

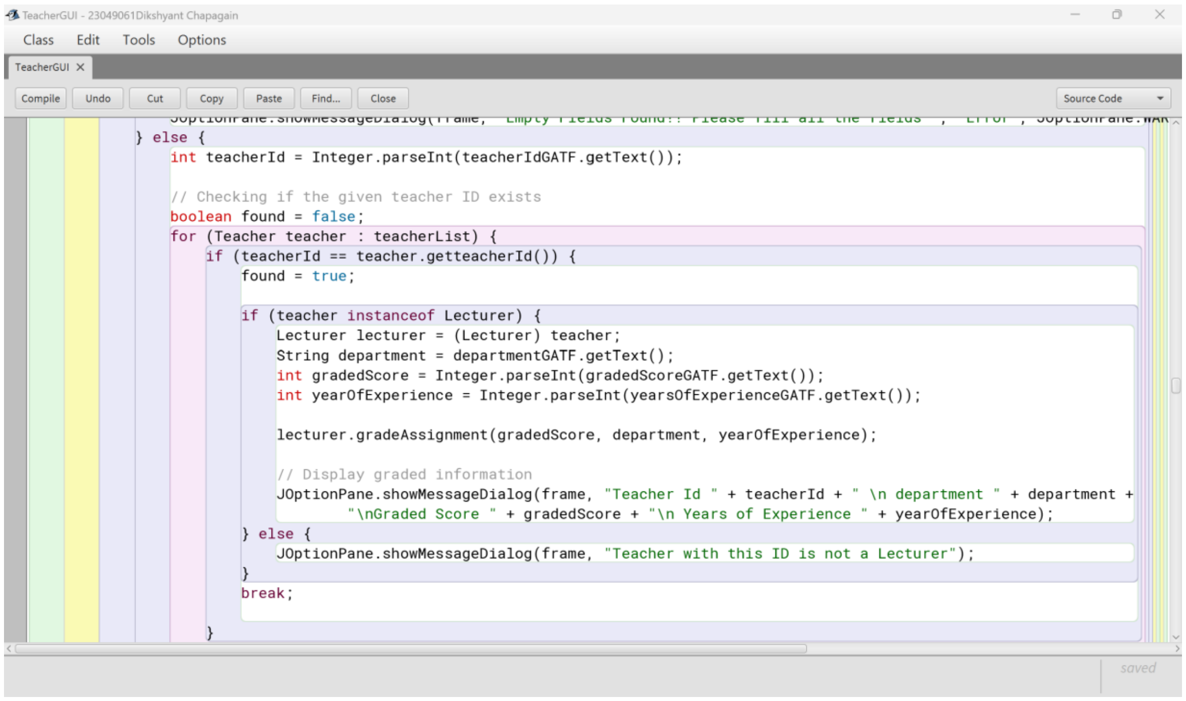
Error	Method gradeAssignment in class Lecturer cannot be applied to given types
Cause	Wrong number of arguments were passed in the method ie gradedScore was missing
Fix	gradedScore was passed in the method



```
        JOptionPane.showMessageDialog(frame, "Empty Fields Found!! Please fill all the fields", "Error", JOptionPane.WAR
    } else {
        int teacherId = Integer.parseInt(teacherIdGATF.getText());

        // Checking if the given teacher ID exists
        boolean found = false;
        for (Teacher teacher : teacherList) {
            if (teacherId == teacher.getteacherId()) {
                found = true;

                if (teacher instanceof Lecturer) {
                    Lecturer lecturer = (Lecturer) teacher;
                    String department = departmentGATF.getText();
                    int gradedScore = Integer.parseInt(gradedScoreGATF.getText());
                    int yearOfExperience = Integer.parseInt(yearsOfExperienceGATF.getText());

                    lecturer.gradeAssignment(department, yearOfExperience);

                    // Display g
                    JOptionPane.                                              \n department " + department +
                            "\nG                                                + yearOfExperience);
                } else {
                    JOptionPane.showMessageDialog(frame, "Teacher with this ID is not a Lecturer");
                }
                break;
            }
```

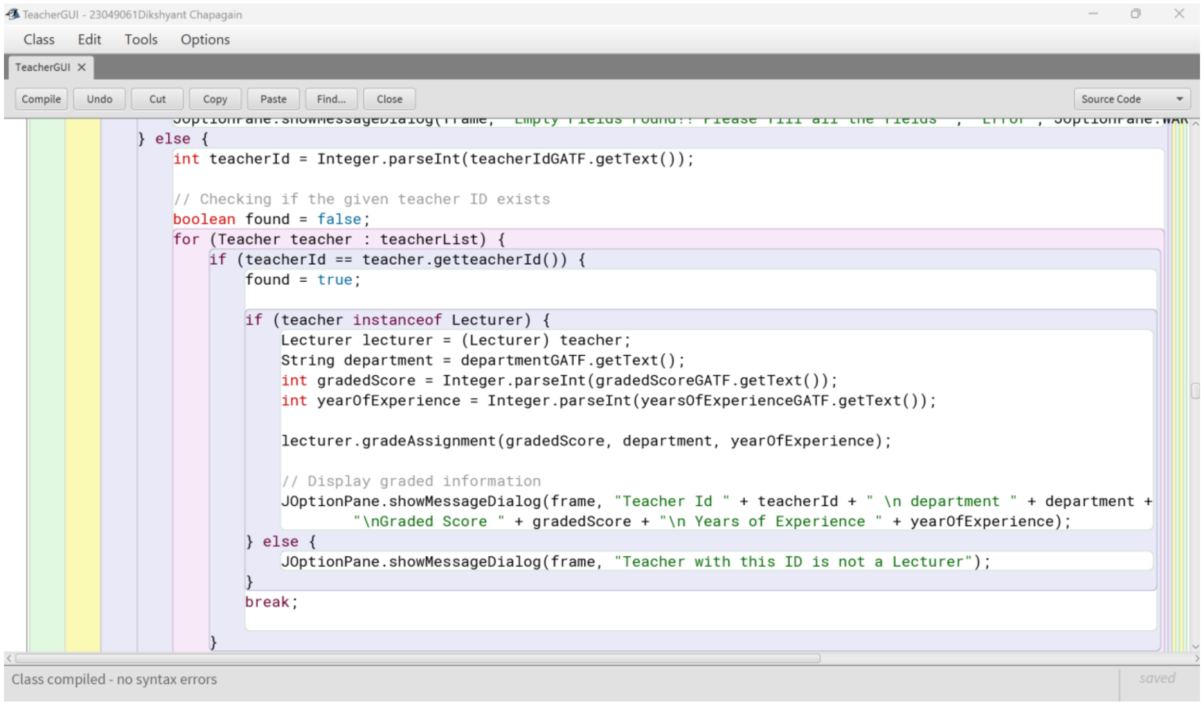
```
method gradeAssignment in class Lecturer cannot be applied to given types;
  required: int,java.lang.String,int
  found:    java.lang.String,int
  reason: actual and formal argument lists differ in length
```

Figure 37Semantic error


TeacherGUI - 23049061Dikshyant Chapagain
Class Edit Tools Options
TeacherGUI
Compile Undo Cut Copy Paste Find... Close
Source Code
saved

```
} else {
    int teacherId = Integer.parseInt(teacherIdGATF.getText());

    // Checking if the given teacher ID exists
    boolean found = false;
    for (Teacher teacher : teacherList) {
        if (teacherId == teacher.getteacherId()) {
            found = true;

            if (teacher instanceof Lecturer) {
                Lecturer lecturer = (Lecturer) teacher;
                String department = departmentGATF.getText();
                int gradedScore = Integer.parseInt(gradedScoreGATF.getText());
                int yearOfExperience = Integer.parseInt(yearsOfExperienceGATF.getText());

                lecturer.gradeAssignment(gradedScore, department, yearOfExperience);

                // Display graded information
                JOptionPane.showMessageDialog(frame, "Teacher Id " + teacherId + " \n department " + department +
                        "\nGraded Score " + gradedScore + "\n Years of Experience " + yearOfExperience);
            } else {
                JOptionPane.showMessageDialog(frame, "Teacher with this ID is not a Lecturer");
            }
            break;
        }
```

Figure 38fixing Semantic error


TeacherGUI - 23049061Dikshyant Chapagain
Class Edit Tools Options
TeacherGUI
Compile Undo Cut Copy Paste Find... Close
Source Code
saved

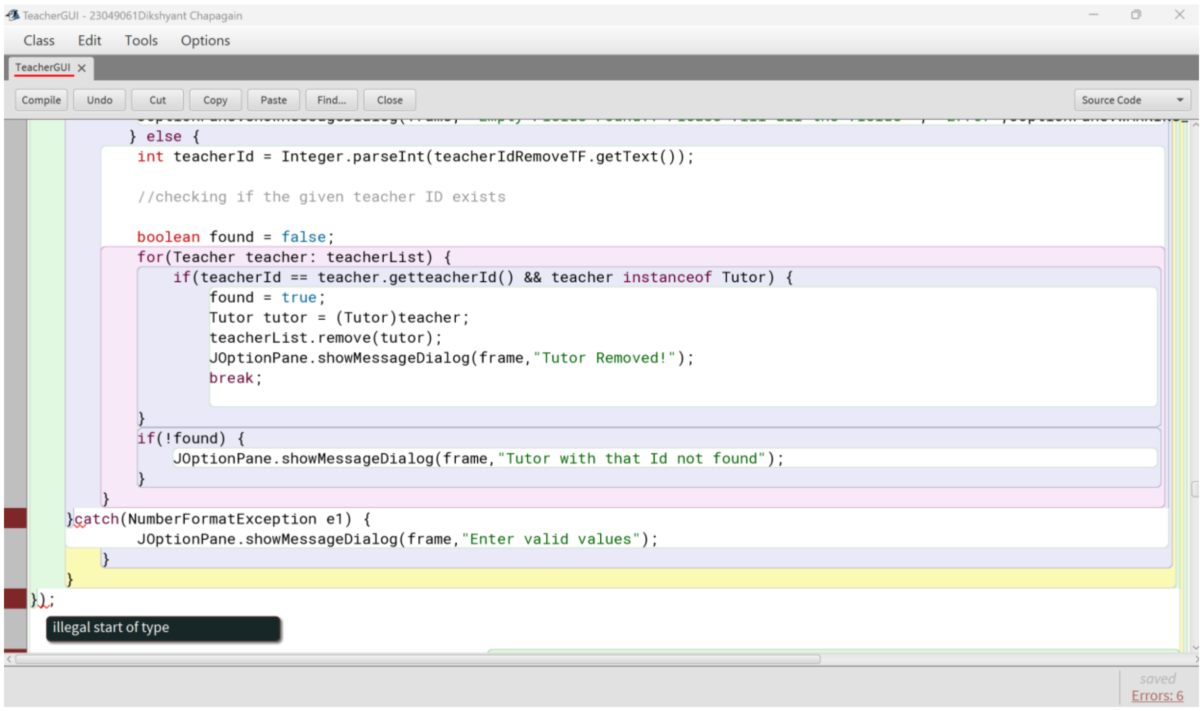
```
} else {
    int teacherId = Integer.parseInt(teacherIdGATF.getText());

    // Checking if the given teacher ID exists
    boolean found = false;
    for (Teacher teacher : teacherList) {
        if (teacherId == teacher.getteacherId()) {
            found = true;

            if (teacher instanceof Lecturer) {
                Lecturer lecturer = (Lecturer) teacher;
                String department = departmentGATF.getText();
                int gradedScore = Integer.parseInt(gradedScoreGATF.getText());
                int yearOfExperience = Integer.parseInt(yearsOfExperienceGATF.getText());

                lecturer.gradeAssignment(gradedScore, department, yearOfExperience);

                // Display graded information
                JOptionPane.showMessageDialog(frame, "Teacher Id " + teacherId + " \n department " + department +
                        "\nGraded Score " + gradedScore + "\n Years of Experience " + yearOfExperience);
            } else {
                JOptionPane.showMessageDialog(frame, "Teacher with this ID is not a Lecturer");
            }
            break;
        }
```

Class compiled - no syntax errors

Figure 39Semantic error fixed

Error 3 (Logical error)

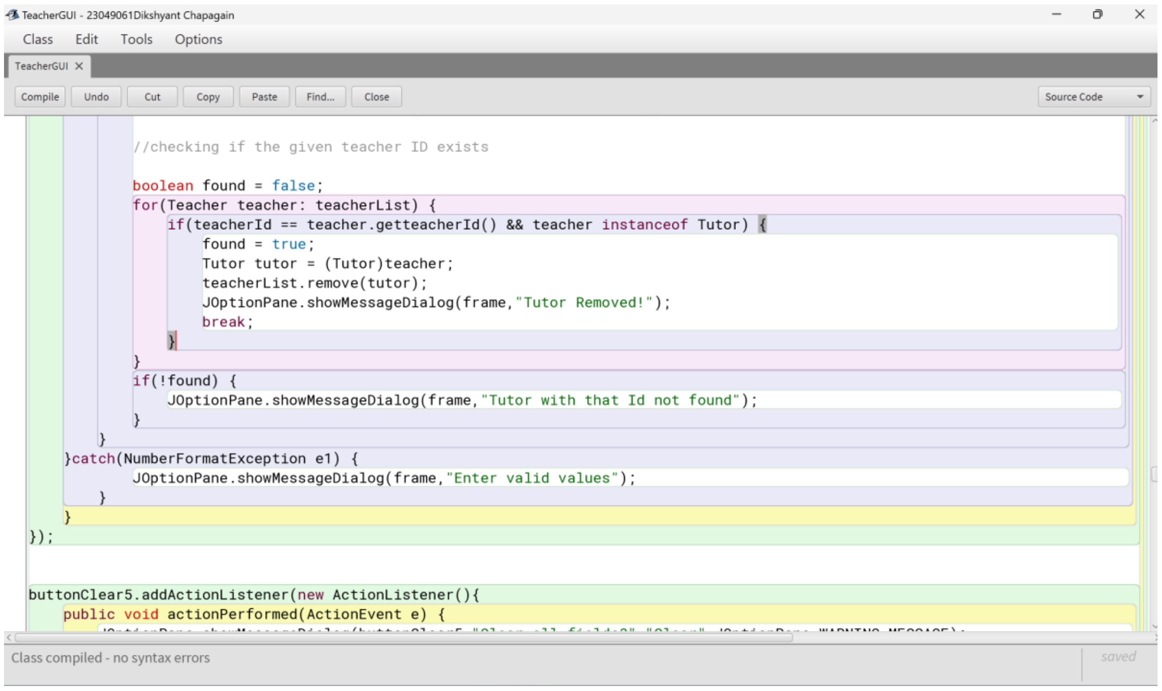
Error	Illegal start of type
Cause	If statement was not closed properly
Fix	If statement was closed



```
        } else {
            int teacherId = Integer.parseInt(teacherIdRemoveTF.getText());

            //checking if the given teacher ID exists

            boolean found = false;
            for(Teacher teacher: teacherList) {
                if(teacherId == teacher.getteacherId() && teacher instanceof Tutor) {
                    found = true;
                    Tutor tutor = (Tutor)teacher;
                    teacherList.remove(tutor);
                    JOptionPane.showMessageDialog(frame,"Tutor Removed!");
                    break;

                }
            if(!found) {
                JOptionPane.showMessageDialog(frame,"Tutor with that Id not found");
            }
        }
    }catch(NumberFormatException e1) {
        JOptionPane.showMessageDialog(frame,"Enter valid values");
    }
}
});
```

illegal start of type

Figure 40 Logical error


TeacherGUI - 23049061Dikshyant Chapagain
Class Edit Tools Options
TeacherGUI
Compile Undo Cut Copy Paste Find... Close
Source Code

```
                //checking if the given teacher ID exists

                boolean found = false;
                for(Teacher teacher: teacherList) {
                    if(teacherId == teacher.getteacherId() && teacher instanceof Tutor) {
                        found = true;
                        Tutor tutor = (Tutor)teacher;
                        teacherList.remove(tutor);
                        JOptionPane.showMessageDialog(frame,"Tutor Removed!");
                        break;
                    }
                }
                if(!found) {
                    JOptionPane.showMessageDialog(frame,"Tutor with that Id not found");
                }
            }
        }catch(NumberFormatException e1) {
                JOptionPane.showMessageDialog(frame,"Enter valid values");
            }
        }
});


buttonClear5.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent e) {
```

Class compiled - no syntax errors

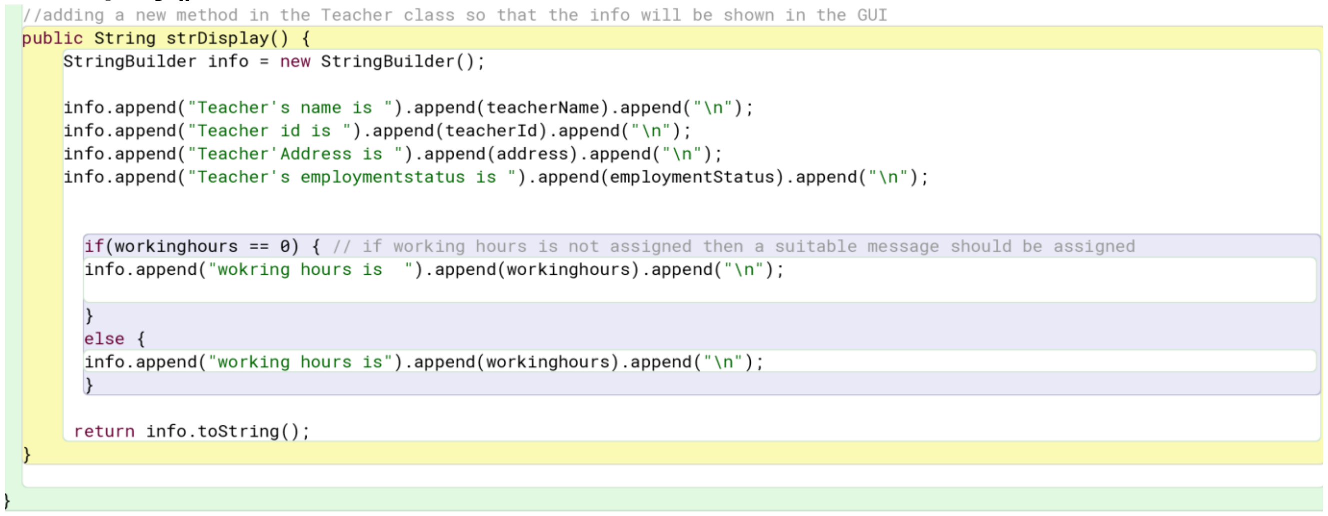
saved

Figure 41logical error fix

Changes made to the code of the previous coursework.

While doing this coursework, few changes were made to the code of the previous coursework in the Teacher, Lecturer and Tutor class. In all of them a new method called **strDisplay()** of the return type String was introduced. This was done due to the original display method of the void return type was not able to be displayed in the GUI. So, a new method which returns a String was introduced to fix this issue. In all of them a “String Builder” object named info was created. StringBuilder is used to efficiently build strings bby appending different pieces of text together. The display data were appended to the String Builder by the **append()** method. Also, since we need to return the data into a string, the method. **toString()** was used to return the “StringBuilder” object into a string.

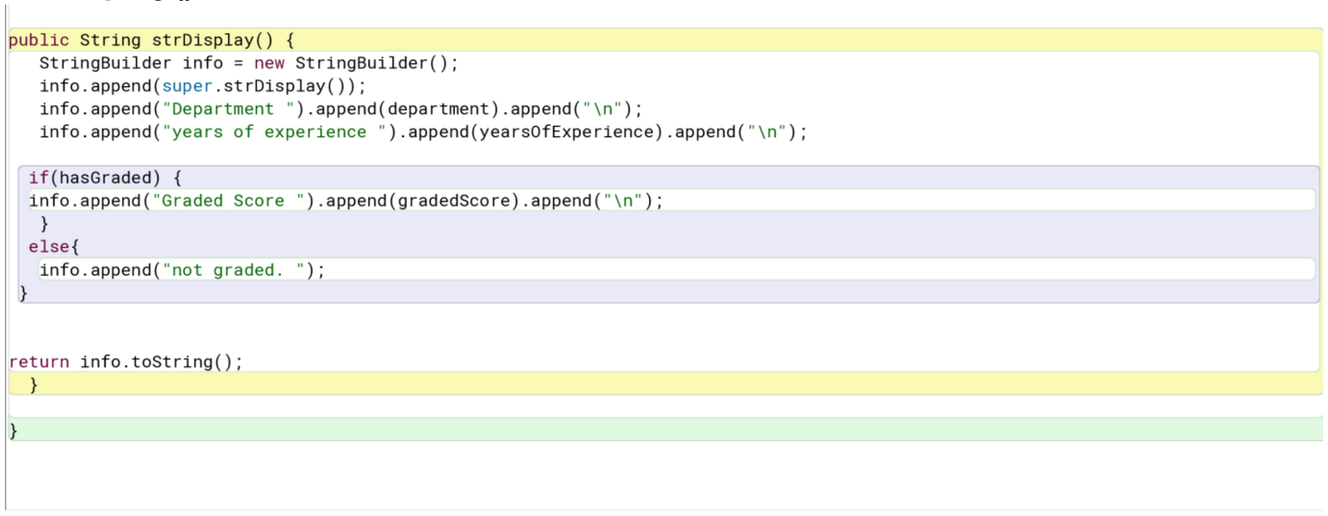
strDisplay() in the Teacher class



```
//adding a new method in the Teacher class so that the info will be shown in the GUI
public String strDisplay() {
    StringBuilder info = new StringBuilder();

    info.append("Teacher's name is ").append(teacherName).append("\n");
    info.append("Teacher id is ").append(teacherId).append("\n");
    info.append("Teacher'Address is ").append(address).append("\n");
    info.append("Teacher's employmentstatus is ").append(employmentStatus).append("\n");


      if(workinghours == 0) { // if working hours is not assigned then a suitable message should be assigned
      info.append("wokring hours is  ").append(workinghours).append("\n");

      }
      else {
      info.append("working hours is").append(workinghours).append("\n");
      }

     return info.toString();
}

}
```

Figure 42 strDisplay() in the Teacher class

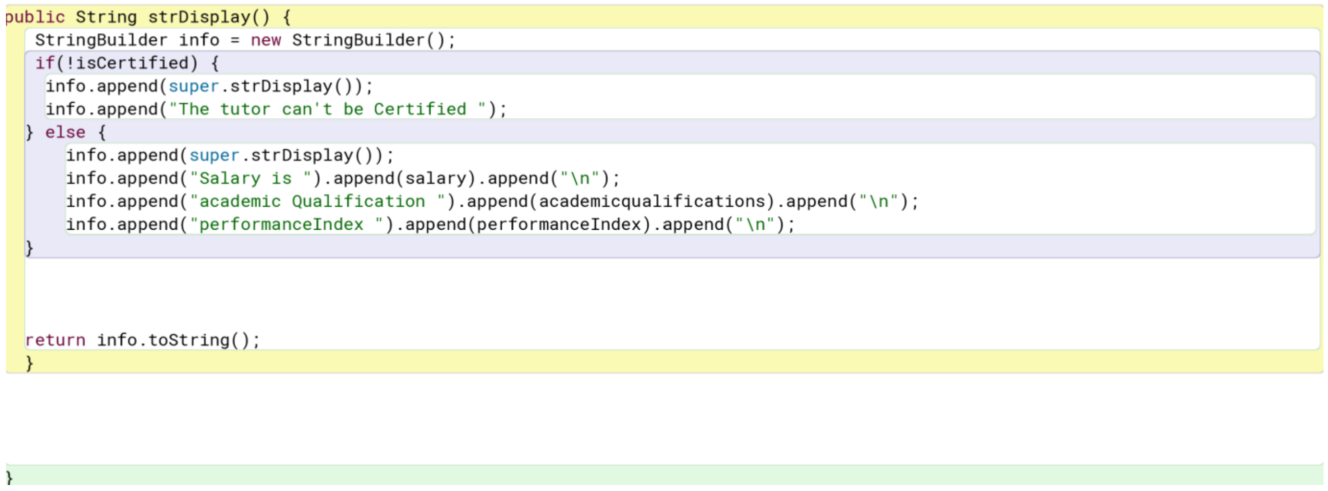
strDisplay() in the Lecturer class



```
public String strDisplay() {
   StringBuilder info = new StringBuilder();
   info.append(super.strDisplay());
   info.append("Department ").append(department).append("\n");
   info.append("years of experience ").append(yearsOfExperience).append("\n");

  if(hasGraded) {
  info.append("Graded Score ").append(gradedScore).append("\n");
   }
  else{
   info.append("not graded. ");
 }


return info.toString();
  }

}
```

Figure 43 strDisplay() in the lecturer class

strDisplay() in the Tutor class



```
public String strDisplay() {
  StringBuilder info = new StringBuilder();
  if(!isCertified) {
   info.append(super.strDisplay());
   info.append("The tutor can't be Certified ");
 } else {
     info.append(super.strDisplay());
     info.append("Salary is ").append(salary).append("\n");
     info.append("academic Qualification ").append(academicqualifications).append("\n");
     info.append("performanceIndex ").append(performanceIndex).append("\n");
 }


 return info.toString();
 }



}
```

Figure 44 strDisplay() in the Tutor class

Conclusion

This is the second coursework of the Programming module. The main objective of this second coursework was to create a Graphical User Interface by using Java. Java is an object-oriented programming language which uses the concept of classes and modules. This Java project has 4 classes in which 3 of them was created during the first coursework. The final class called TeacherGUI was created in this coursework. The main aim of this project was to develop a graphical user interface (GUI) for a system that stores details of teachers in an ArrayList.

This coursework was an interesting one. A lot of work was research was put into this coursework. A lot of hours was spent on this project. Being a computing student, I had to balance this and the coursework of the Fundamentals of Computing module. But, through learning, practicing and a lot of sleepless nights, the completion of this coursework was possible. Creating a GUI that is visually appealing was arguably the most challenging part of this project. Adding the functionality was also very tough. Writing the pseudo was quite hectic at some point. Also, the numerous times of blunders and errors also occurred. However, despite the many challenges and hardships throughout the duration of this project. I was able to finish the project on time.

This coursework helped in understanding the basics of Java Swing and give us a basic idea of how a GUI is developed. It also helped a lot in searching for solutions for problems that occurred. With the help of my teachers, seniors and the authors of many articles related to this topic, along with finishing the coursework it also help in having a basic understanding on how a programmer does his projects.

References

Contributer, T. T., n.d. *https://www.techtarget.com/.* [Online]
Available at: https://www.techtarget.com/searchapparchitecture/definition/class-diagram
[Accessed 9 5 2024].

Galkward, P., 2023. *https://medium.com/.* [Online]
Available at: https://medium.com/@priyankaaa2502/introduction-to-java-b7aadb85ae7b
[Accessed 9 5 2024].

Metwalli, S. A., 2024. *https://builtin.com/.* [Online]
Available at: https://builtin.com/data-science/pseudocode
[Accessed 9 5 2024].

Rolandmack, 2023. *ttps://medium.com.* [Online]
Available at: https://medium.com/@rolandmack63/introduction-to-java-gui-programming-with-swing-c9bedda86ee8#:~:text=Java%20Swing%20is%20a%20powerful,create%20attractive%20and%20functional%20GUIs.
[Accessed 9 5 2024].

StudyTonight, n.d. *https://www.studytonight.com/.* [Online]
Available at: https://www.studytonight.com/java/java-awt.php
[Accessed 9 5 2024].

Appendix

```
import java.util.ArrayList;

import java.awt.Color;

import javax.swing.*;

import java.awt.*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;




public class TeacherGUI  {



private JFrame frame;



private JButton lectAdd, tutoAdd, gradeAssign, salarySet, tutoRemove, displayButton,
addLect, buttonClear, addTuto,buttonClear2,

gradeButton,buttonClear3,buttonClear4,setSalary,removeTuto,buttonClear5,buttonClear6,
display1,display2;
```

```
private JPanel  panel, mainContent,mainContent2, mainContent3,
mainContent4,mainContent5,mainContent6,headingPanel;



private JLabel
titleLabel,teacherIdLectLabel,teacherIdTutoLabel,teacherIdGALabel,teacherIdSalaryLabel,te
acherIdRemoveLabel,

teacherNameLabel,addressLabel,workingTypeLabel,employmentStatusLabel,workingHoursL
abel,departmentLabel,yearsOfExperinenceLabel,

teacherIdDisplayLabel1,teacherIdDisplayLabel2,gradedScoresLabel,specializationLabel,aca
demicQualificationsLabel,performanceIndexLabel, salaryLabel,

workingHoursLectLabel,departmentLectLabel ;

//for lecturer

private JTextField
teacherIdLectTF,teacherNameLectTF,addressLectTF,workingTypeLectTF,employmentStatus
LectTF,gradedScoreLectTF,

yearsOfExperienceLectTF,workingHoursLectTF,departmentLectTF;

//for tutor

private JTextField
teacherIdTutoTF,teacherNameTutoTF,addressTutoTF,workingTypeTutoTF,employmentStatus
TutoTF,workingHoursTutoTF,

salaryTutoTF,specializationTutoTF,academicqualificationsTutoTF,performanceIndexTutoTF;

//for garde assignment
```

```
private JTextField
teacherIdGATF,gradedScoreGATF,departmentGATF,yearsOfExperienceGATF;

// set Salary

private JTextField teacherIdSalaryTF, salarySalaryTF,performanceIndexSalaryTF;

// remove

private JTextField teacherIdRemoveTF;

// display

private JTextField teacherIdLecDisplayTF,teacherIdTutDisplayTF;




//--------------------Importing Color-----------------------------------------------------


private Color white = new Color(0xFFC3C0);

private Color cornflowerBlue = new Color(100, 149, 200);



//Array List

public ArrayList<Teacher> teacherList = new ArrayList<>();
public TeacherGUI() {

frame = new JFrame("Dikshyant Chapagain 23049061");

frame.setSize(860,555);

frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);









//-----------------------------------------------BUTTONS TO CHANGE THE PANELS---------------------
--------



lectAdd = new JButton("Add a lecturer");

lectAdd.setPreferredSize(new Dimension(120,60));

tutoAdd = new JButton("Add a Tutor");

tutoAdd.setPreferredSize(new Dimension(120,60));
```

```
gradeAssign = new JButton("Assign Grades");

gradeAssign.setPreferredSize(new Dimension(120,60));

salarySet = new JButton("Set Salary");

salarySet.setPreferredSize(new Dimension(120,60));

tutoRemove = new JButton("Remove Tutor");

tutoRemove .setPreferredSize(new Dimension(120,60));

displayButton = new JButton("Display");

displayButton.setPreferredSize(new Dimension(120,60));









//adding the button to the side of the frame

panel = new JPanel(new GridLayout(6,1));

panel.add(lectAdd);

panel.add(gradeAssign);
panel.add(tutoAdd);

panel.add(salarySet);

panel.add(tutoRemove);

panel.add(displayButton);




lectAdd.setBackground(cornflowerBlue);

gradeAssign.setBackground(cornflowerBlue);

tutoAdd.setBackground(cornflowerBlue);

salarySet.setBackground(cornflowerBlue);

tutoRemove.setBackground(cornflowerBlue);

displayButton.setBackground(cornflowerBlue);



lectAdd.setFont(new Font("Arial",Font.BOLD,12));

gradeAssign.setFont(new Font("Arial",Font.BOLD,12));

tutoAdd.setFont(new Font("Arial",Font.BOLD,12));

salarySet.setFont(new Font("Arial",Font.BOLD,12));
```

```
tutoRemove.setFont(new Font("Arial",Font.BOLD,12));

displayButton.setFont(new Font("Arial",Font.BOLD,12));




lectAdd.setBorder(BorderFactory.createLineBorder(Color.WHITE));

gradeAssign.setBorder(BorderFactory.createLineBorder(Color.WHITE));

tutoAdd.setBorder(BorderFactory.createLineBorder(Color.WHITE));

salarySet.setBorder(BorderFactory.createLineBorder(Color.WHITE));

tutoRemove.setBorder(BorderFactory.createLineBorder(Color.WHITE));

displayButton.setBorder(BorderFactory.createLineBorder(Color.WHITE));




lectAdd.setForeground(Color.WHITE);

gradeAssign.setForeground(Color.WHITE);

tutoAdd.setForeground(Color.WHITE);

salarySet.setForeground(Color.WHITE);

tutoRemove.setForeground(Color.WHITE);
```

```
displayButton.setForeground(Color.WHITE);







headingPanel = new JPanel();

headingPanel.setBackground(cornflowerBlue);

headingPanel.setBorder(BorderFactory.createEmptyBorder(10,10,10,10));




JLabel headingLabel = new JLabel("Teacher GUI");

headingLabel.setForeground(Color.WHITE);

headingLabel.setFont(new Font("Arial",Font.BOLD,19));

headingLabel.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));


headingPanel.add(headingLabel);
```

```
frame.getContentPane().add(headingPanel, BorderLayout.NORTH);

//---------------------------------panel for the main content-----------------------------------------------

mainContent = new JPanel(new GridBagLayout());

GridBagConstraints constraints = new GridBagConstraints();

constraints.fill = GridBagConstraints.HORIZONTAL;

constraints.insets = new Insets(5,5,5,5);

JLabel titleLabel = new JLabel("Add a Lecturer");
```

```
titleLabel.setFont(new Font("Arial", Font.BOLD, 26));

titleLabel.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));




teacherIdLectLabel = new JLabel("Teacher Id");

teacherIdLectTF = new JTextField(10);


teacherNameLabel = new JLabel("teacherName");

teacherNameLectTF = new JTextField(10);


addressLabel = new JLabel("Address");

addressLectTF = new JTextField(10);


workingTypeLabel = new JLabel("Working Type");

workingTypeLectTF = new JTextField(10);


employmentStatusLabel = new JLabel("Employment Status");
```

```
employmentStatusLectTF = new JTextField(10);



gradedScoresLabel = new JLabel("Graded Score");

gradedScoreLectTF = new JTextField(10);



yearsOfExperinenceLabel = new JLabel("Years of Experience");

yearsOfExperienceLectTF = new JTextField(10);



workingHoursLectLabel = new JLabel("Working Hours");

workingHoursLectTF = new JTextField(10);



departmentLectLabel = new JLabel("Department");

departmentLectTF = new JTextField(10);





addLect = new JButton("Add Lecturer");

addLect.setPreferredSize(new Dimension(60,30));
```

```
buttonClear = new JButton ("Clear");

buttonClear.setPreferredSize(new Dimension(60,30));



addLect.setBackground(cornflowerBlue);

addLect.setForeground(Color.WHITE);

buttonClear.setBackground(cornflowerBlue);

buttonClear.setForeground(Color.WHITE);






addLect.addActionListener(new ActionListener() {

   public void actionPerformed(ActionEvent e) {





     try{
        if(teacherIdLectTF.getText().equals("")
||teacherNameLectTF.getText().equals("")||addressLectTF.getText().equals("")


||workingTypeLectTF.getText().equals("")||employmentStatusLectTF.getText().equals("")||

        gradedScoreLectTF.getText().equals("")||yearsOfExperienceLectTF.getText().


equals("")||workingHoursLectTF.getText().equals("")||departmentLectTF.getText().equals("")) {

            JOptionPane.showMessageDialog(frame, "Empty Fields Found!! Please fill all the
fields ", "Error",JOptionPane.WARNING_MESSAGE);

          } else {




        int teacherId = Integer.parseInt(teacherIdLectTF.getText());

        String teacherName = teacherNameLectTF.getText();

        String address = addressLectTF.getText();

        String workingType = workingTypeLectTF.getText();

        String employmentStatus = employmentStatusLectTF.getText();

        int gradedScore = Integer.parseInt(gradedScoreLectTF.getText());

        int yearsOfExperience= Integer.parseInt(yearsOfExperienceLectTF.getText());

        double workinghours = Double.parseDouble(workingHoursLectTF.getText());
```

```
    String department = departmentLectTF.getText();




    //checking if a teacher with a same Id exists

    boolean found = false;

    for(Teacher teacher: teacherList) {

        if(teacherId == teacher.getteacherId()) {

        found = true;

        break;

     }

    }

    if(found) {

        JOptionPane.showMessageDialog(frame,"This teacher Id alreay exists add another
one");

   }

    else {

        //creating a new Lecturer object
        Lecturer lecturer = new Lecturer(teacherId, teacherName, address, workingType,
employmentStatus, workinghours, department, yearsOfExperience);

        //adding lecturer to teacherlist

        teacherList.add(lecturer);

        JOptionPane.showMessageDialog(frame,"Lecturer Added with Id "+teacherId +"\n"

                + "Teacher ID: " + teacherId + "\n"

                + "Teacher Name: " + teacherName + "\n"

                + "Address: " + address + "\n"

                + "Working Type: " + workingType + "\n"

                + "Employment Status: " + employmentStatus + "\n"

                + "Graded Score: " + gradedScore + "\n"

                + "Years of Experience: " + yearsOfExperience + "\n"

                + "Working Hours: " + workinghours + "\n"

                + "Department: " + department



        );

      }

    }
```

```
        }catch(NumberFormatException e1) {

            JOptionPane.showMessageDialog(frame,"Please enter valid input values");

        }

}



});




buttonClear.addActionListener(new ActionListener(){

   public void actionPerformed(ActionEvent e) {

        JOptionPane.showMessageDialog(buttonClear,"Clear all
fields?","Clear",JOptionPane.WARNING_MESSAGE);

        teacherIdLectTF.setText("");

        teacherNameLectTF.setText("");

        addressLectTF.setText("");

        workingTypeLectTF.setText("");

        employmentStatusLectTF.setText("");

        gradedScoreLectTF.setText("");
        yearsOfExperienceLectTF.setText("");

        workingHoursLectTF.setText("");

        departmentLectTF.setText("");

          }

});



constraints.gridx = 0;

constraints.gridy = 0;

mainContent.add(titleLabel, constraints);



constraints.gridx = 0;

constraints.gridy = 1;

mainContent.add(teacherIdLectLabel, constraints);



constraints.gridx = 1;

constraints.gridy = 1;

mainContent.add(teacherIdLectTF, constraints);
```

```
constraints.gridx = 0;

constraints.gridy = 2;

mainContent.add(teacherNameLabel, constraints);



constraints.gridx = 1;

constraints.gridy = 2;

mainContent.add(teacherNameLectTF, constraints);



constraints.gridx = 0;

constraints.gridy = 3;

mainContent.add(workingTypeLabel,constraints);



constraints.gridx = 1;

constraints.gridy = 3;

mainContent.add(workingTypeLectTF,constraints);
```

```
constraints.gridx = 0;

constraints.gridy = 4;

mainContent.add(addressLabel,constraints);



constraints.gridx = 1;

constraints.gridy = 4;

mainContent.add(addressLectTF,constraints);



constraints.gridx = 0;

constraints.gridy = 5;

mainContent.add(employmentStatusLabel,constraints);



constraints.gridx = 1;

constraints.gridy = 5;

mainContent.add(employmentStatusLectTF, constraints);



constraints.gridx = 0;
constraints.gridy = 6;

mainContent.add(gradedScoresLabel,constraints);



constraints.gridx = 1;

constraints.gridy = 6;

mainContent.add(gradedScoreLectTF,constraints);



constraints.gridx = 0;

constraints.gridy = 7;

mainContent.add(yearsOfExperinenceLabel,constraints);




constraints.gridx = 1;

constraints.gridy = 7;

mainContent.add(yearsOfExperienceLectTF, constraints);



constraints.gridx = 0;
constraints.gridy = 8;

mainContent.add(departmentLectLabel,constraints);



constraints.gridx = 1;

constraints.gridy = 8;

mainContent.add(departmentLectTF,constraints);



constraints.gridx = 0;

constraints.gridy = 9;

mainContent.add(workingHoursLectLabel,constraints);



constraints.gridx = 1;

constraints.gridy = 9;

mainContent.add(workingHoursLectTF,constraints);




constraints.gridx = 0;
constraints.gridy = 10;

mainContent.add(addLect,constraints);



constraints.gridx = 1;

constraints.gridy = 10;

mainContent.add(buttonClear,constraints);









mainContent2 = new JPanel(new GridBagLayout());

GridBagConstraints constraints1 = new GridBagConstraints();

constraints1.fill = GridBagConstraints.HORIZONTAL;

constraints1.insets = new Insets(5,5,5,5);



JLabel titleLabel2 = new JLabel("Add a Tutor   ");

titleLabel2.setFont(new Font("Arial", Font.BOLD, 26));
```

```
titleLabel2.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));




teacherIdTutoLabel = new JLabel("Teacher Id");

teacherIdTutoTF = new JTextField(10);



teacherNameLabel = new JLabel("teacherName");

teacherNameTutoTF = new JTextField(10);



addressLabel = new JLabel("Address");

addressTutoTF = new JTextField(10);



workingTypeLabel = new JLabel("Working Type");

workingTypeTutoTF = new JTextField(10);



employmentStatusLabel = new JLabel("Employment Status");

employmentStatusTutoTF = new JTextField(10);
```

```
workingHoursLabel = new JLabel("Working Hours");

workingHoursTutoTF = new JTextField(10);




salaryLabel = new JLabel("Salary (double)");

salaryTutoTF = new JTextField(10);


specializationLabel = new JLabel("Specialization");

specializationTutoTF = new JTextField(10);


academicQualificationsLabel = new JLabel("Academic Qualifications");

academicqualificationsTutoTF = new JTextField(10);


performanceIndexLabel = new JLabel("Performance Index");

performanceIndexTutoTF= new JTextField(10);
```

```
addTuto = new JButton("Add Tutor");

//changing the size of the button

addTuto.setPreferredSize(new Dimension(60,30));



buttonClear2 = new JButton("Clear");

buttonClear2.setPreferredSize(new Dimension(60,30));



addTuto.setBackground(cornflowerBlue);

addTuto.setForeground(Color.WHITE);

buttonClear2.setBackground(cornflowerBlue);

buttonClear2.setForeground(Color.WHITE);



addTuto.addActionListener(new ActionListener() {

    public void actionPerformed(ActionEvent e) {

        try {

            if (teacherIdTutoTF.getText().equals("") || teacherNameTutoTF.getText().equals("") ||
        addressTutoTF.getText().equals("") || workingTypeTutoTF.getText().equals("") ||

        employmentStatusTutoTF.getText().equals("") ||
workingHoursTutoTF.getText().equals("") ||

        salaryTutoTF.getText().equals("") || specializationTutoTF.getText().equals("") ||

        academicqualificationsTutoTF.getText().equals("") ||
performanceIndexTutoTF.getText().equals("")) {

        JOptionPane.showMessageDialog(frame, "Empty Fields Found!! Please fill all the
fields ", "Error", JOptionPane.WARNING_MESSAGE);

      } else {



      int teacherId = Integer.parseInt(teacherIdTutoTF.getText());

      String teacherName = teacherNameTutoTF.getText();

      String address = addressTutoTF.getText();

      String workingType = workingTypeTutoTF.getText();

      String employmentStatus = employmentStatusTutoTF.getText();

      int workingHours = Integer.parseInt(workingHoursTutoTF.getText());

      double salary = Double.parseDouble(salaryTutoTF.getText());

      String specialization = specializationTutoTF.getText();

      String academicQualifications = academicqualificationsTutoTF.getText();
```

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



        // Checking if the given Id of tutor already exists

        boolean found = false;

        for (Teacher teacher : teacherList) {

            if (teacher instanceof Tutor && teacherId == teacher.getteacherId()) {

                found = true;

                JOptionPane.showMessageDialog(frame, "A tutor with id " + teacherId + "
exists");

                break;

            }

        }



        if (!found) {

            // Create a new Tutor Object

            Tutor tutor = new Tutor(teacherId, teacherName, address, workingType,
employmentStatus, workingHours, salary, specialization, academicQualifications,
performanceIndex);

            // Add the new tutor to the arrayList
        teacherList.add(tutor);

        JOptionPane.showMessageDialog(frame, "Tutor with id " + teacherId + " added"

            +  "Teacher Id: " +teacherId + "\n"

            + "Teacher Name: " + teacherName + "\n"

            + "Address: " + address + "\n"

            + "Working Type: " + workingType + "\n"

            + "Employment Status: " + employmentStatus + "\n"

            + "Working Hours: " + workingHours + "\n"

            + "Salary: " + salary + "\n"

            + "Specialization: " + specialization + "\n"

            + "Academic Qualifications: " + academicQualifications + "\n"

            + "Performance Index: " + performanceIndex);

      }



    }

   }

    catch  (NumberFormatException e1) {
          JOptionPane.showMessageDialog(frame, "Please Enter valid values");

       }

   }

});

buttonClear2.addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

      JOptionPane.showMessageDialog(buttonClear,"Clear all
fields?","Clear",JOptionPane.WARNING_MESSAGE);

       teacherIdTutoTF.setText("");

        teacherNameTutoTF.setText("");

        addressTutoTF.setText("");

        workingTypeTutoTF.setText("");

        employmentStatusTutoTF.setText("");
        workingHoursTutoTF.setText("");

        salaryTutoTF.setText("");

        specializationTutoTF.setText("");

        academicqualificationsTutoTF.setText("");

        performanceIndexTutoTF.setText("");

    }

}

);




constraints.gridx = 0;

constraints.gridy = 0;

mainContent2.add(titleLabel2, constraints);



constraints1.gridx = 0;

constraints1.gridy = 1;

mainContent2.add(teacherIdTutoLabel, constraints1);
```

```
constraints1.gridx = 1;

constraints1.gridy = 1;

mainContent2.add(teacherIdTutoTF, constraints1);



constraints1.gridx = 0;

constraints1.gridy = 2;

mainContent2.add(teacherNameLabel, constraints1);



constraints1.gridx = 1;

constraints1.gridy = 2;

mainContent2.add(teacherNameTutoTF, constraints1);



constraints1.gridx = 0;

constraints1.gridy = 3;

mainContent2.add(addressLabel,constraints1);
```

```
constraints1.gridx = 1;

constraints1.gridy = 3;

mainContent2.add(addressTutoTF,constraints1);



constraints1.gridx = 0;

constraints1.gridy = 4;

mainContent2.add(workingTypeLabel,constraints1);



constraints1.gridx = 1;

constraints1.gridy = 4;

mainContent2.add(workingTypeTutoTF,constraints1);



constraints1.gridx = 0;

constraints1.gridy = 5;

mainContent2.add(employmentStatusLabel,constraints1);



constraints1.gridx = 1;
```

```
constraints1.gridy = 5;

mainContent2.add(employmentStatusTutoTF, constraints1);



constraints1.gridx = 0;

constraints1.gridy = 6;

mainContent2.add(workingHoursLabel,constraints1);



constraints1.gridx = 1;

constraints1.gridy = 6;

mainContent2.add(workingHoursTutoTF,constraints1);



constraints1.gridx = 0;

constraints1.gridy = 7;

mainContent2.add(salaryLabel,constraints1);



constraints1.gridx = 1;

constraints1.gridy = 7;
```

```
mainContent2.add(salaryTutoTF,constraints1);



constraints1.gridx = 0;

constraints1.gridy = 8;

mainContent2.add(specializationLabel,constraints1);



constraints1.gridx = 1;

constraints1.gridy = 8;

mainContent2.add(specializationTutoTF,constraints1);



constraints1.gridx = 0;

constraints1.gridy = 9;

mainContent2.add(academicQualificationsLabel,constraints1);



constraints1.gridx = 1;

constraints1.gridy = 9;

mainContent2.add(academicqualificationsTutoTF,constraints1);
```

```
constraints1.gridx = 0;

constraints1.gridy = 10;

mainContent2.add(performanceIndexLabel,constraints1);



constraints1.gridx = 1;

constraints1.gridy = 10;

mainContent2.add(performanceIndexTutoTF,constraints1);



constraints1.gridx= 0;

constraints1.gridy = 11;

mainContent2.add(addTuto,constraints1);



constraints1.gridx=1;

constraints1.gridy=11;

mainContent2.add(buttonClear2,constraints1);
```

```
// grading the assignment

mainContent3 = new JPanel(new GridBagLayout());

GridBagConstraints constraints2 = new GridBagConstraints();

constraints2.fill = GridBagConstraints.HORIZONTAL;

constraints2.insets = new Insets(5,5,5,5);

JLabel titleLabel3 = new JLabel("Grade Assignment");

titleLabel3.setFont(new Font("Arial", Font.BOLD, 22));

titleLabel3.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));

teacherIdGALabel = new JLabel("TeacherId");
```

```
teacherIdGATF = new JTextField(10);



gradedScoresLabel = new JLabel("Graded Score");

gradedScoreGATF = new JTextField(10);



departmentLabel = new JLabel("Department");

departmentGATF = new JTextField(10);



yearsOfExperinenceLabel = new JLabel("Years of Experience");

yearsOfExperienceGATF = new JTextField(10);



gradeButton = new JButton("Grade");

gradeButton.setPreferredSize(new Dimension(60,30));



buttonClear2 = new JButton("Clear");

buttonClear2.setPreferredSize(new Dimension(60,30));
```

```
gradeButton.setBackground(cornflowerBlue);

gradeButton.setForeground(Color.WHITE);

buttonClear2.setBackground(cornflowerBlue);

buttonClear2.setForeground(Color.WHITE);




gradeButton.addActionListener(new ActionListener() {

    public void actionPerformed(ActionEvent e) {

        try {

            // Checking if any of the text fields are empty

            if (teacherIdGATF.getText().equals("") || departmentGATF.getText().equals("") ||
gradedScoreGATF.getText().equals("")

                    || yearsOfExperienceGATF.getText().equals("")) {

                JOptionPane.showMessageDialog(frame, "Empty Fields Found!! Please fill all the
fields ", "Error", JOptionPane.WARNING_MESSAGE);

            } else {

                int teacherId = Integer.parseInt(teacherIdGATF.getText());
        // Checking if the given teacher ID exists

        boolean found = false;

        for (Teacher teacher : teacherList) {

            if (teacherId == teacher.getteacherId()) {

                found = true;



                if (teacher instanceof Lecturer) {

                    Lecturer lecturer = (Lecturer) teacher;

                    String department = departmentGATF.getText();

                    int gradedScore = Integer.parseInt(gradedScoreGATF.getText());

                    int yearOfExperience =
Integer.parseInt(yearsOfExperienceGATF.getText());



                    lecturer.gradeAssignment(gradedScore, department, yearOfExperience);



                    // Display graded information

                    JOptionPane.showMessageDialog(frame, "Teacher Id " + teacherId + " \n
department " + department +
                          "\nGraded Score " + gradedScore + "\n Years of Experience " +
yearOfExperience);

                } else {

                    JOptionPane.showMessageDialog(frame, "Teacher with this ID is not a
Lecturer");

                }

                break;



            }

        }



        // If teacher is not found

        if (!found) {

            JOptionPane.showMessageDialog(frame, "Teacher with this ID not found");

        }

    }

} catch (NumberFormatException e1) {

    JOptionPane.showMessageDialog(frame, "Please enter valid values");
        }

    }

});

buttonClear2.addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

      JOptionPane.showMessageDialog(buttonClear2,"Clear all
fields?","Clear",JOptionPane.WARNING_MESSAGE);

       teacherIdGATF.setText("");

       departmentGATF.setText("");

       yearsOfExperienceGATF.setText("");

       gradedScoreGATF.setText("");



    }

}
```

```
);



constraints2.gridx = 0;

constraints2.gridy = 0;

mainContent3.add(titleLabel3,constraints2);






constraints2.gridx = 0;

constraints2.gridy = 1;

mainContent3.add(teacherIdGALabel, constraints2);



constraints2.gridx = 1;

constraints2.gridy = 1;

mainContent3.add(teacherIdGATF, constraints2);



constraints2.gridx = 0;
```

```
constraints2.gridy = 2;

mainContent3.add(departmentLabel, constraints2);



constraints2.gridx = 1;

constraints2.gridy = 2;

mainContent3.add(departmentGATF, constraints2);





constraints2.gridx = 0;

constraints2.gridy = 3;

mainContent3.add( yearsOfExperinenceLabel,constraints2);



constraints2.gridx = 1;

constraints2.gridy = 3;

mainContent3.add(yearsOfExperienceGATF,constraints2);



constraints2.gridx = 0;
```

```
constraints2.gridy = 4;

mainContent3.add(gradedScoresLabel,constraints2);



constraints2.gridx = 1;

constraints2.gridy = 4;

mainContent3.add(gradedScoreGATF,constraints2);



constraints2.gridx = 0;

constraints2.gridy = 5;

mainContent3.add(gradeButton,constraints2);



constraints2.gridx = 1;

constraints2.gridy = 5;

mainContent3.add(buttonClear2,constraints2);




//set salary of the Tutor
```

```
mainContent4 = new JPanel(new GridBagLayout());

GridBagConstraints constraints3 = new GridBagConstraints();

constraints3.fill = GridBagConstraints.HORIZONTAL;

constraints3.insets = new Insets(5,5,5,5);



JLabel titleLabel4 = new JLabel("Set Salary    ");

titleLabel4.setFont(new Font("Arial", Font.BOLD, 26));

titleLabel4.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));




teacherIdSalaryLabel = new JLabel("Teacher Id");

teacherIdSalaryTF = new JTextField(10);



performanceIndexLabel = new JLabel("PerformanceIndex");

performanceIndexSalaryTF = new JTextField(10);
```

```
salaryLabel = new JLabel("Salary ");

salarySalaryTF = new JTextField(10);



setSalary = new JButton("Set");

setSalary.setPreferredSize(new Dimension(60,30));



buttonClear4 = new JButton("Clear");

buttonClear4.setPreferredSize(new Dimension(60,30));



setSalary.setBackground(cornflowerBlue);

setSalary.setForeground(Color.WHITE);

buttonClear4.setBackground(cornflowerBlue);

buttonClear4.setForeground(Color.WHITE);



setSalary.addActionListener(new ActionListener() {

    public void actionPerformed(ActionEvent e) {

        try {
```

```
        if (teacherIdSalaryTF.getText().equals("") || salarySalaryTF.getText().equals("") ||
performanceIndexSalaryTF.getText().equals("")) {

            JOptionPane.showMessageDialog(frame, "Empty Fields Found!! Please fill all the
fields ", "Error", JOptionPane.WARNING_MESSAGE);

        } else {

            int teacherId = Integer.parseInt(teacherIdSalaryTF.getText());



            // Checking if the given teacherId exists

            boolean found = false;

            for (Teacher teacher : teacherList) {

                if (teacherId == teacher.getteacherId() && teacher instanceof Tutor) {

                    found = true;



                    Tutor tutor = (Tutor) teacher;

                    double newSalary = Double.parseDouble(salarySalaryTF.getText());

                    int newPerformanceIndex =
Integer.parseInt(performanceIndexSalaryTF.getText());

                    tutor.setSalary(newSalary, newPerformanceIndex);
                    JOptionPane.showMessageDialog(frame, "Teacher ID: " + teacherId + "\n" +
"New Salary: " + newSalary + "\n" + "New Performance Index: " + newPerformanceIndex);

                    break;

                }

            }

            if (!found) {

                // If teacher ID exists but is not a Tutor

                JOptionPane.showMessageDialog(frame, "Teacher ID does not exist or is not a
tutor");

            }

        }

    } catch (NumberFormatException e1) {

        JOptionPane.showMessageDialog(frame, "Please enter valid values");

    }

  }

});
```

```
buttonClear4.addActionListener(new ActionListener() {

   public void actionPerformed(ActionEvent e) {

        JOptionPane.showMessageDialog(buttonClear4,"Clear all
fields?","Clear",JOptionPane.WARNING_MESSAGE);

        teacherIdSalaryTF.setText("");

        salarySalaryTF.setText("");

        performanceIndexSalaryTF.setText("");

   }

});

constraints3.gridx = 0;

constraints3.gridy = 0;

mainContent4.add(titleLabel4,constraints3);




constraints3.gridx = 0;

constraints3.gridy = 1;

mainContent4.add(performanceIndexLabel,constraints3);
```

```
constraints3.gridx = 1;

constraints3.gridy = 1;

mainContent4.add(performanceIndexSalaryTF,constraints3);



constraints3.gridx = 0;

constraints3.gridy = 2;

mainContent4.add(teacherIdSalaryLabel,constraints3);



constraints3.gridx = 1;

constraints3.gridy = 2;

mainContent4.add(teacherIdSalaryTF,constraints3);



constraints3.gridx = 0;

constraints3.gridy = 3;

mainContent4.add(salaryLabel,constraints3);
```

```
constraints3.gridx = 1;

constraints3.gridy = 3;

mainContent4.add(salarySalaryTF,constraints3);



constraints3.gridx = 1;

constraints3.gridy = 4;

mainContent4.add(buttonClear4, constraints3);



constraints3.gridx = 0;

constraints3.gridy = 4;

mainContent4.add(setSalary,constraints3);



//remove the tutor

mainContent5 = new JPanel(new GridBagLayout());

GridBagConstraints constraints4 = new GridBagConstraints();

constraints4.fill = GridBagConstraints.HORIZONTAL;

constraints4.insets = new Insets(5,5,5,5);
```

```
JLabel titleLabel5 = new JLabel("Remove Tutor");

titleLabel5.setFont(new Font("Arial",Font.BOLD,24));

titleLabel5.setBorder(BorderFactory.createEmptyBorder(10,10,10,10));



removeTuto = new JButton("Remove");

removeTuto.setPreferredSize(new Dimension(90,30));

buttonClear5 = new JButton("Clear");

buttonClear5.setPreferredSize(new Dimension(90,30));



removeTuto.setBackground(cornflowerBlue);

buttonClear5.setBackground(cornflowerBlue);

removeTuto.setForeground(Color.WHITE);

buttonClear5.setForeground(Color.WHITE);



teacherIdRemoveLabel = new JLabel("Teacher Id");

teacherIdRemoveTF = new JTextField(10);
```

```
removeTuto.addActionListener(new ActionListener(){

    public void actionPerformed(ActionEvent e) {

        try {

            if(teacherIdRemoveTF.getText().equals("")) {

            JOptionPane.showMessageDialog(frame, "Empty Fields Found!! Please fill all the
fields ", "Error",JOptionPane.WARNING_MESSAGE);

           } else {

            int teacherId = Integer.parseInt(teacherIdRemoveTF.getText());



            //checking if the given teacher ID exists



            boolean found = false;

            for(Teacher teacher: teacherList) {

                if(teacherId == teacher.getteacherId() && teacher instanceof Tutor) {
                    found = true;

                    Tutor tutor = (Tutor)teacher;

                    teacherList.remove(tutor);

                    JOptionPane.showMessageDialog(frame,"Tutor Removed!");

                    break;

                  }

                }

                if(!found) {

                  JOptionPane.showMessageDialog(frame,"Tutor with that Id not found");

                }

              }

          }catch(NumberFormatException e1) {

                JOptionPane.showMessageDialog(frame,"Enter valid values");

              }

          }

      });
```

```
buttonClear5.addActionListener(new ActionListener(){

    public void actionPerformed(ActionEvent e) {

        JOptionPane.showMessageDialog(buttonClear5,"Clear all
fields?","Clear",JOptionPane.WARNING_MESSAGE);

        teacherIdRemoveTF.setText("");

    }

});




constraints4.gridx = 0;

constraints4.gridy = 0;

mainContent5.add(titleLabel5,constraints4);



constraints4.gridx = 0;

constraints4.gridy = 1;

mainContent5.add(teacherIdRemoveLabel,constraints4);
```

```
constraints4.gridx=1;

constraints4.gridy=1;

mainContent5.add(teacherIdRemoveTF,constraints4);



constraints4.gridx = 0;

constraints4.gridy = 2;

mainContent5.add(removeTuto,constraints4);



constraints4.gridx = 1;

constraints4.gridy = 2;

mainContent5.add(buttonClear5,constraints4);
```

```
//---------------------------------display--------------------------------

mainContent6 = new JPanel(new GridBagLayout());

GridBagConstraints constraints5 = new GridBagConstraints();

constraints5.fill = GridBagConstraints.HORIZONTAL;

constraints5.insets = new Insets(5,5,5,5);


JLabel titleLabel6 = new JLabel("Display            ");

titleLabel6.setFont(new Font("Arial",Font.BOLD,24));

titleLabel6.setBorder(BorderFactory.createEmptyBorder(10,10,10,10));




teacherIdDisplayLabel1 = new JLabel("Teacher Id(Lecturer)");

teacherIdLecDisplayTF = new JTextField(10);


teacherIdDisplayLabel2 = new JLabel("Teacher Id(Tutor)");
```

```
teacherIdTutDisplayTF = new JTextField(10);




display1 = new JButton("Display Lecturer");

display1.setPreferredSize(new Dimension(120,30));



display2 = new JButton("Display Tutor");

display2.setPreferredSize(new Dimension(120,30));





buttonClear6 = new JButton("Clear");

buttonClear6.setPreferredSize(new Dimension(60,30));



display1.setBackground(cornflowerBlue);



constraints5.gridx = 0;
```

```
constraints5.gridy = 0;

mainContent6.add(titleLabel6,constraints5);



constraints5.gridx = 0;

constraints5.gridy = 1;

mainContent6.add(teacherIdDisplayLabel1,constraints5);



constraints5.gridx = 1;

constraints5.gridy = 1;

mainContent6.add(teacherIdLecDisplayTF,constraints5);



constraints5.gridx = 0;

constraints5.gridy = 2;

mainContent6.add(teacherIdDisplayLabel2,constraints5);



constraints5.gridx = 1;

constraints5.gridy = 2;
```

```
mainContent6.add(teacherIdTutDisplayTF,constraints5);




constraints5.gridx = 0;

constraints5.gridy = 3;

mainContent6.add(display1,constraints5);


constraints5.gridx = 0;

constraints5.gridy = 4;

mainContent6.add(display2,constraints5);




constraints5.gridx = 1;

constraints5.gridy = 3;

mainContent6.add(buttonClear6,constraints5);


display1.setBackground(cornflowerBlue);
```

```
display1.setForeground(Color.WHITE);



display2.setBackground(cornflowerBlue);

display2.setForeground(Color.WHITE);





buttonClear6.setBackground(cornflowerBlue);

buttonClear6.setForeground(Color.WHITE);


display1.addActionListener(new ActionListener() {

    public void actionPerformed(ActionEvent e) {

        try {

            if(teacherIdLecDisplayTF.getText().equals("")) {

            JOptionPane.showMessageDialog(frame, "Empty Fields Found!! Please fill all the
fields ", "Error",JOptionPane.WARNING_MESSAGE);



            }
```

```
else {

int teacherId = Integer.parseInt(teacherIdLecDisplayTF.getText());




boolean found = false;

for (Teacher teacher : teacherList) {

    if (teacher instanceof Lecturer && teacherId == teacher.getteacherId()) {

        found = true;

        Lecturer lecturer = (Lecturer) teacher;

        String teacherInfo = lecturer.strDisplay();

        JOptionPane.showMessageDialog(frame, teacherInfo, "Lecturer Information",
JOptionPane.INFORMATION_MESSAGE);

         break;

    }




}
```

```
            if (!found) {

                JOptionPane.showMessageDialog(frame, "Teacher not found");

            }



        }

      }catch (NumberFormatException e1) {

            JOptionPane.showMessageDialog(frame, "Please enter valid integer values in the
text fields");

        }

    }

});




display2.addActionListener(new ActionListener() {

    public void actionPerformed(ActionEvent e) {

        try {

            if(teacherIdTutDisplayTF.getText().equals("")) {
```

```
        JOptionPane.showMessageDialog(frame, "Empty Fields Found!! Please fill all the
fields ", "Error",JOptionPane.WARNING_MESSAGE);



        }else{

         int teacherId = Integer.parseInt(teacherIdTutDisplayTF.getText());

         boolean found = false;

         for (Teacher teacher : teacherList) {

            if (teacher instanceof Tutor && teacherId == teacher.getteacherId()) {

                found = true;

                Tutor tutor = (Tutor) teacher;

                String teacherInfo = tutor.strDisplay();

                JOptionPane.showMessageDialog(frame, teacherInfo, "Tutor Information",
JOptionPane.INFORMATION_MESSAGE);

                break;

            }

         }



         if (!found) {
            JOptionPane.showMessageDialog(frame, "Teacher not found");

          }



        }



    }catch (NumberFormatException e1) {

          JOptionPane.showMessageDialog(frame, "Please enter valid integer values in the
text fields");

        }

    }

});







buttonClear6.addActionListener(new ActionListener(){

    public void actionPerformed(ActionEvent e) {

        JOptionPane.showMessageDialog(buttonClear6,"Clear all
fields?","Clear",JOptionPane.WARNING_MESSAGE);
```

```
        teacherIdLecDisplayTF.setText("");

        teacherIdTutDisplayTF.setText("");

    }

});



//panel to add when clicking  a buttton

lectAdd.addActionListener(new ActionListener() {

     public void actionPerformed(ActionEvent e) {

        removeIfShowing(mainContent2);

        removeIfShowing(mainContent3);

        removeIfShowing(mainContent4);

        removeIfShowing(mainContent5);

        removeIfShowing(mainContent6);



        if (!mainContent.isShowing()) {

            frame.getContentPane().add(mainContent, BorderLayout.CENTER);

            mainContent6.setBackground(Color.WHITE);
        }

        frame.revalidate();

        frame.repaint();

    }

    public void removeIfShowing(JPanel panel) {

        if (panel.isShowing()) {

            frame.getContentPane().remove(panel);

         }

    }

});

tutoAdd.addActionListener(new ActionListener(){

    public void actionPerformed(ActionEvent e) {

        removeIfShowing(mainContent);

        removeIfShowing(mainContent3);
        removeIfShowing(mainContent4);

        removeIfShowing(mainContent5);

        removeIfShowing(mainContent6);



        if (!mainContent2.isShowing()) {

            frame.getContentPane().add(mainContent2, BorderLayout.CENTER);

            mainContent2.setBackground(Color.WHITE);

        }



        frame.revalidate();

        frame.repaint();

    }



    public void removeIfShowing(JPanel panel) {

        if (panel.isShowing()) {

            frame.getContentPane().remove(panel);

         }
    }

});



gradeAssign.addActionListener(new ActionListener(){

    public void actionPerformed(ActionEvent e) {

        removeIfShowing(mainContent);

        removeIfShowing(mainContent2);

        removeIfShowing(mainContent4);

        removeIfShowing(mainContent5);

        removeIfShowing(mainContent6);



        if (!mainContent3.isShowing()) {

            frame.getContentPane().add(mainContent3, BorderLayout.CENTER);

            mainContent3.setBackground(Color.WHITE);

        }



        frame.revalidate();
        frame.repaint();

    }



    public void removeIfShowing(JPanel panel) {

        if (panel.isShowing()) {

            frame.getContentPane().remove(panel);

         }

    }

});



salarySet.addActionListener(new ActionListener(){

      public void actionPerformed(ActionEvent e) {

        removeIfShowing(mainContent);

        removeIfShowing(mainContent2);

        removeIfShowing(mainContent3);

        removeIfShowing(mainContent5);

        removeIfShowing(mainContent6);
        if (!mainContent4.isShowing()) {

            frame.getContentPane().add(mainContent4, BorderLayout.CENTER);

            mainContent4.setBackground(Color.WHITE);

        }



        frame.revalidate();

        frame.repaint();

    }



    public void removeIfShowing(JPanel panel) {

        if (panel.isShowing()) {

            frame.getContentPane().remove(panel);

         }

    }

});
```

```
tutoRemove.addActionListener(new ActionListener() {

    public void actionPerformed(ActionEvent e) {

      removeIfShowing(mainContent);

      removeIfShowing(mainContent2);

      removeIfShowing(mainContent3);

      removeIfShowing(mainContent4);

      removeIfShowing(mainContent6);


      if (!mainContent5.isShowing()) {

          frame.getContentPane().add(mainContent5, BorderLayout.CENTER);

          mainContent5.setBackground(Color.WHITE);

      }


      frame.revalidate();

      frame.repaint();

  }
```

```
    public void removeIfShowing(JPanel panel) {

        if (panel.isShowing()) {

            frame.getContentPane().remove(panel);

         }

    }

});




displayButton.addActionListener(new ActionListener() {

    public void actionPerformed(ActionEvent e) {

        removeIfShowing(mainContent);

        removeIfShowing(mainContent2);

        removeIfShowing(mainContent3);

        removeIfShowing(mainContent4);

        removeIfShowing(mainContent5);



        if (!mainContent6.isShowing()) {
            frame.getContentPane().add(mainContent6, BorderLayout.CENTER);

            mainContent6.setBackground(Color.WHITE);

        }



        frame.revalidate();

        frame.repaint();

    }



    public void removeIfShowing(JPanel panel) {

        if (panel.isShowing()) {

            frame.getContentPane().remove(panel);

         }

    }




});
```

```
mainContent.setVisible(true);

frame.getContentPane().add(panel,BorderLayout.WEST);

frame.getContentPane().add(mainContent,BorderLayout.CENTER);

mainContent.setBackground(Color.WHITE);

frame.setVisible(true);




}
```

```
// main method

public static void main(String[] args) {

    new TeacherGUI();

}




}
```