

**Module Code & Module Title:**

**CS4001NT Programming**

**Assessment Weightage & Type:**

**30% Individual Coursework**

**Year and Semester:**

**2023-24 Autumn**

**Student Name: Anmol Poudyal**

**London Met ID: 23049190**

**College ID:np05cp4a230006@iic.edu.np**

**Assignment Due Date:10 May 2024**

**Word Count:6777**

*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 mark of zero will be awarded.*

Table of Contents

[Table of figure 3](#_Toc166225552)

[Table of table 5](#_Toc166225553)

[1 Introduction 1](#_Toc166225554)

[1.1 Introduction to Java 1](#_Toc166225555)

[1.2 Introduction to BLUE-J 2](#_Toc166225556)

[1.3 Introduction to GUI 2](#_Toc166225557)

[1.4 Introduction to Microsoft Word 3](#_Toc166225558)

[1.5 Introduction to Project 3](#_Toc166225559)

[2 Class diagram 4](#_Toc166225560)

[Class diagram of the TeacherGUI class 5](#_Toc166225561)

[3 Pseudocode 6](#_Toc166225562)

[4 Method description 18](#_Toc166225563)

[5 Testing 22](#_Toc166225564)

[5.1 Testing 1 : Compiling & Running using command prompt 22](#_Toc166225565)

[5.2 Testing 2 : Adding objects of Tutor and Lecturer, grade assignments amount from lecturer, setting the salary and removing the tutor 25](#_Toc166225566)

[Testing 2.3: Grade Assignments from Lecturer 29](#_Toc166225567)

[5.3 Test 3 37](#_Toc166225568)

[6 Error Detection and Correction 45](#_Toc166225569)

[6.1 Syntax error 45](#_Toc166225570)

[6.2 Logical error 46](#_Toc166225571)

[6.3 Semantic error 48](#_Toc166225572)

[7 Conclusion 49](#_Toc166225573)

[8 References 50](#_Toc166225574)

[9 Appendix 50](#_Toc166225575)

[10 Plagiarism report 75](#_Toc166225576)

## Table of figure

[Figure 1: JAVA Logo 1](#_Toc166225632)

[Figure 2: BlueJ logo 2](#_Toc166225633)

[Figure 3: Microsoft Word logo 3](#_Toc166225634)

[Figure 4: class diagram of TeacherGUI 5](#_Toc166225635)

[Figure 5: calss diagram in BLUEJ 6](#_Toc166225636)

[Figure 6: Reaching to the file through path 22](#_Toc166225637)

[Figure 7: successful compilation 23](#_Toc166225638)

[Figure 8: Running the program after comilation 23](#_Toc166225639)

[Figure 9 Displayed GUI after running 24](#_Toc166225640)

[Figure 10: Entering the info of lecturer 26](#_Toc166225641)

[Figure 11: after the tutor button was clicked the oject got stored in list 26](#_Toc166225642)

[Figure 12: displaying in terminal after adding the object of lecturer 27](#_Toc166225643)

[Figure 13: adding tutor details to create tutor object 28](#_Toc166225644)

[Figure 14: after adding tutor 29](#_Toc166225645)

[Figure 15: Add tutor displayed tutor in terminal 29](#_Toc166225646)

[Figure 16: inputs for the Grade assignment of teacherID 1 30](#_Toc166225647)

[Figure 17: Displayed after grading assignment 31](#_Toc166225648)

[Figure 18: message displayed after grading assignment in CLI 31](#_Toc166225649)

[Figure 19: after adding tutor for setting salary 32](#_Toc166225650)

[Figure 20: inputs for the seting salary of teacher id 3 33](#_Toc166225651)

[Figure 21: dialogue box after setting salary 33](#_Toc166225652)

[Figure 22: adding new tutor for removing 34](#_Toc166225653)

[Figure 23: Before removing tutor 35](#_Toc166225654)

[Figure 24: Entering text field for removing tutor 35](#_Toc166225655)

[Figure 25: after remove tutor button was clicked 36](#_Toc166225656)

[Figure 26: in terminal after removing tutor 36](#_Toc166225657)

[Figure 27: adding tutor with teach id 1 37](#_Toc166225658)

[Figure 28: Output in dialogue after entering same teacher id to add lecturer 38](#_Toc166225659)

[Figure 29: info of ID 1 as a lecturer 39](#_Toc166225660)

[Figure 30: trying to remove lecturer with id 1 in tutor method 39](#_Toc166225661)

[Figure 31: Error message after entering lecturer id in remove tutor 40](#_Toc166225662)

[Figure 32: teacher info of lecturer with teacher id 1 41](#_Toc166225663)

[Figure 33: entering info of lecturer in set salary 41](#_Toc166225664)

[Figure 34: resu;t after setting salary for lecturer id 42](#_Toc166225665)

[Figure 35: createing object of tutor 43](#_Toc166225666)

[Figure 36: filling the fields of grade assignment with tutor id 44](#_Toc166225667)

[Figure 37: error message showing only lecturers can grade assignment 44](#_Toc166225668)

[Figure 38: syntax error missing part 45](#_Toc166225669)

[Figure 39 Syntax errror solved 46](#_Toc166225670)

[Figure 40: Logical error example 47](#_Toc166225671)

[Figure 41: semantic error detection 48](#_Toc166225672)

[Figure 42: correction of Semantic error 49](#_Toc166225673)

# Table of table

[Table 1: method description of the TeacherGUI class 18](#_Toc166225717)

[Table 2: test 1 compiling and running 22](#_Toc166225718)

[Table 3: Testing 1 adding Lecturer 25](#_Toc166225719)

[Table 4: test 2 Adding tutor 27](#_Toc166225720)

[Table 5: Testing 3 Grading assignment 29](#_Toc166225721)

[Table 6: testing 4 Setting salary for tutor 31](#_Toc166225722)

[Table 7: test 5 removing Tutor 34](#_Toc166225723)

[Table 8: test 6 add teacher with repeating id 37](#_Toc166225724)

[Table 9 Test 7 removing tutor 38](#_Toc166225725)

[Table 10: test 8 setting salary for lecturer 40](#_Toc166225726)

[Table 11: test 9 to grade assignment on a tutor object 42](#_Toc166225727)

# Introduction

## Introduction to Java

Java is one of the most popular and widely used object orienteed programming language and software platform developed by James Gosling of sun microsystem which later was aquired by oracal in 1991. Java rules and syntax are based on C and C++ languages. To create a java program you need to download JDK that is java developnment kit which is available for mac-os,linux,windows etc. The java software platform also consists of JVM (java virtual machine ) which interprets the java bytecode after comilation. (IBM, 2023). It has been popular among developers for 20 years and still is. It is a fast, secure, reliable programming language for building varieties of things from mobile apps and enterprise software to big data applications and server-side technologies. (AWS, 2023).



Figure 1: JAVA Logo

## Introduction to BLUE-J

Blue-J is most popular Java developnment environment especially designed for beginners. Its is wildely popular amoung college and university students as it is free and easy to use. The main character feature of blue-j are it is portable which means you can runs in wide varities of operating systems, it is simple as it uses smaller and simpler interfaces, it also interactive as you can invoke java expression without even comipling. (BLUEJ, 2024).

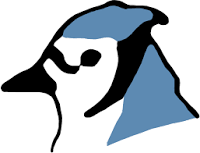


Figure 2: BlueJ logo

## Introduction to GUI

A GUI or a graphical user interface is a user friendly digital interface where a user can interact with graphical components like icons, buttons, menus etc. GUI graphics are tied to real-world objects to represent their purpose and help users understand the functions available to them. (Juviler, April 03, 2024)

Java also supports development of the GUI based programmes. There are two tools which are frewuently being used in java to develop GUI based programmes are Swing and JavaFX.

## Introduction to Microsoft Word

Microsoft word or msword is one of most used word processor developed by Microsoft in 1983. In this course work I have used microsoft word for the documentation purpose.

A blue logo with white text

Description automatically generated

Figure 3: Microsoft Word logo

## Introduction to Project

In this course work we were asked to develop a GUI for the teacher recored system from the previous course work. For building the GUI I have used BLUEJ as my IDE and have used microsoftsoft word for documentation. In this course work I have used varitires of java swing components as well as AWT.

# Class diagram

A class diagram is a diagram or a representation of a class that is used to view software in high level of abstraction. It also describes the relationship between different classes in a software (Agile, 2023)

## Class diagram of the TeacherGUI class



Figure 4: class diagram of TeacherGUI

A diagram of a teacher

Description automatically generated

Figure 5: calss diagram in BLUEJ

# Pseudocode

**Fucnction** TeacherGUI()

**Create** object of Jframe with name “Frame”

**Set**  size of the frame(width 1000, height 750)

**Set** location to middle of the screen

**Set** closeoperation to close progrm on exit

**Set** the layout of the frame null

**Create** object of gridlayout named as lecturegridlayout

**Set** columns and rows to be (2,9)

**Create** object of gridlayout named as gradeassignment layout

**Set** columns and rows (2,4)

**Create** object of gridlayout named as tutogridlayout

**Set** columns and rows(2,10)

**Create** object of gridlayout named as Setsalagridlayout

**Set** columns and rows(2,3)

**Create** object of boarder named as teacherinput border, gradeassignment border, tutor panelborder, setsalaryborder, removetutboarder

**Create** object of Jlable named as toplable

**ADD** toplable to frame

**Set** location and size to be (400,10,250,50)

**Set** font of the toplable to be verdan in bolad with size 25px

**Create** object of Jpanel named as teacherJpanel

**ADD** teacherJPanel to the frame

**SET** the bounds of teacherJPanel to (70, 70, 400, 200)

**SET** the layout of teacherJPanel to lectureGridLayout

**SET** the border of teacherJPanel to teacherinputBorder

**SET** the background color of teacherJPanel to GRAY

**CREATE** object of JLabel named teacherId with text "Teacher ID"

**ADD** teacherId to teacherJPanel

**CREATE** a JTextField named teacherIdTf

**ADD** teacherIdTf to teacherJPanel

**CREATE** a JLabel named workingHr with text "Working Hour"

**ADD** workingHr to teacherJPanel

**CREATE** a JTextField named workingHrTf

**ADD** workingHrTf to teacherJPanel

**CREATE** a JLabel named teacherName with text "Teacher Name"

**ADD** teacherName to teacherJPanel

**CREATE** a JTextField named teacherNameTf

**ADD** teacherNameTf to teacherJPanel

**CREATE** a JLabel named address with text "Address"

**ADD** address to teacherJPanel

**CREATE** a JTextField named addressTf

**ADD** addressTf to teacherJPanel

**CREATE** a JLabel named workType with text "Work Type"

**ADD** workType to teacherJPanel

**CREATE** a JTextField named worktypeTf

**ADD** worktypeTf to teacherJPanel

**CREATE** a JLabel named employeeStatus with text "Employee Status"

**ADD** employeeStatus to teacherJPanel

**CREATE** a JTextField named employeeStatusTf

**ADD** employeeStatusTf to teacherJPanel

**CREATE** a JButton named addlecturer with text "Add lecturer"

**ADD** addlecturer to frame

**SET** bounds of addlecturer button to (190, 300, 150, 30)

**Register** addlecturer button with action listner

**Function** action performed(ActionEvent e)

**Call** method addlecturer()

**CREATE** a JPanel named gradeassignmentpanel

**ADD** gradeassignmentpanel to frame

**SET** location and size of gradeassignmentpanel to (70, 350, 400, 300)

**SET** border of gradeassignmentpanel to gradeAssignmBorder

**SET** layout of gradeassignmentpanel to gradeassignmenLayout

**CREATE** JLabel teacherIdGA with text "Teacher ID:"

**ADD** teacherIdGA to gradeassignmentpanel

**CREATE** JTextField teacherIdGATextField

**ADD** teacherIdGATextField to gradeassignmentpanel

**CREATE** JLabel gradedScoreGA with text "Graded score"

**ADD** gradedScoreGA to gradeassignmentpanel

**CREATE** JTextField gradedScoreGATextField

**ADD** gradedScoreGATextField to gradeassignmentpanel

**CREATE** JLabel departmentGA with text "Department"

**ADD** departmentGA to gradeassignmentpanel

**CREATE** JTextField departmentGATextField

**ADD** departmentGATextField to gradeassignmentpanel

**CREATE** JLabel yearsofexpGA with text "Years of exp"

**ADD** yearsofexpGA to gradeassignmentpanel

**CREATE** JTextField yearsOfexpGATextField

**ADD** yearsOfexpGATextField to gradeassignmentpanel

**CREATE** JButton gradeAssignment with text "Grade assignment"

**ADD** gradeAssignment to frame

**SET** bounds of gradeAssignment to (190, 650, 150, 30)

**Register** gradeAssignment button with action listner

**Function** actionPerformed(ActionEvent e)

**Call** the function gradeAssignment()

**CREATE** JPanel tutorPanel

**ADD** tutorPanel to frame

**SET** bounds of tutorPanel to (550, 75, 400, 200)

**SET** border of tutorPanel to tutorpanelborder

**SET** layout of tutorPanel to tutoGridLayout

**SET** background color of tutorPanel to gray

**CREATE** JLabel teacherIdJLabelTU with text "Teacher Id"

**ADD** teacherIdJLabelTU to tutorPanel

**CREATE** JTextField teacheridTU

**ADD** teacheridTU to tutorPanel

**CREATE** JLabel teacherNameLabelTU with text "Teacher Name"

**ADD** teacherNameLabelTU to tutorPanel

**CREATE** JTextField teacherNameTU

**ADD** teacherNameTU to tutorPanel

**CREATE** JLabel addressJLabelTU with text "Address"

**ADD** addressJLabelTU to tutorPanel

**CREATE** JTextField addressTU

**ADD** addressTU to tutorPanel

**CREATE** JLabel workingtypeJLabelTU with text "Working type"

**ADD** workingtypeJLabelTU to tutorPanel

**CREATE** JTextField workTypeTU

**ADD** workTypeTU to tutorPanel

**CREATE** JLabel employeeStatusJlableTU with text "Employeement Status"

**ADD** employeeStatusJlableTU to tutorPanel

**CREATE** JTextField employementStatusTU

**ADD** employementStatusTU to tutorPanel

**CREATE** JLabel workinHrJLabelTU with text "Working Hr"

**ADD** workinHrJLabelTU to tutorPanel

**CREATE** JTextField workingHrTU

**ADD** workingHrTU to tutorPanel

**CREATE** JLabel salaryJLabelTU with text "Salary"

**ADD** salaryJLabelTU to tutorPanel

**CREATE** JTextField salaryTU

**ADD** salaryTU to tutorPanel

**CREATE** JLabel specializationtJLabelTU with text "Specialization"

**ADD** specializationtJLabelTU to tutorPanel

**CREATE** JTextField specializationTU

**ADD** specializationTU to tutorPanel

**CREATE** JLabel academicqualificationJLabelTU with text "Academic Qualification"

**ADD** academicqualificationJLabelTU to tutorPanel

**CREATE** JTextField acedemicQualificationTU

**ADD** acedemicQualificationTU to tutorPanel

**CREATE** JLabel performanceIndexJLabelTU with text "Performance Index"

**ADD** performanceIndexJLabelTU to tutorPanel

**CREATE** JTextField performanceIndexTU

**ADD** performanceIndexTU to tutorPanel

**CREATE** JButton addTutorButton with text "ADD Tutor"

**ADD** addTutorButton to frame

**SET** bounds of addTutorButton to (700, 300, 150, 30)

**Register** addtutorbutton with action listner

**Function** actionPerformed(ActionEvent e)

**Call** function addtutor()

**CREATE** JPanel setsalaryJPanel

**ADD** setsalaryJPanel to frame

**SET** bounds of setsalaryJPanel to (500, 380, 200, 190)

**SET** border of setsalaryJPanel to setsalaryBorder

**SET** layout of setsalaryJPanel to setSalaGridLayout

**CREATE** JLabel teacheriDJPanelSalary with text "Teacher ID"

**ADD** teacheriDJPanelSalary to setsalaryJPanel

**CREATE** JTextField teacherIdSalary

**ADD** teacherIdSalary to setsalaryJPanel

**CREATE** JLabel salarylJLabelsalary with text "Salary"

**ADD** salarylJLabelsalary to setsalaryJPanel

**CREATE** JTextField salarySalary

**ADD** salarySalary to setsalaryJPanel

**CREATE** JLabel performanceIndexJLabelSalary with text "Performance index"

**ADD** performanceIndexJLabelSalary to setsalaryJPanel

**CREATE** JTextField performanceIndexSalary

**ADD** performanceIndexSalary to setsalaryJPanel

**CREATE** JButton setsalary with text "SET SALARY"

**ADD** setsalary to frame

**SET** bounds of setsalary to (530, 580, 150, 30)

**Register** ActionListener to setsalary:

**Function** actionPerformed(ActionEvent e)

**CALL** setSalary()

**CREATE** GridLayout removeGridLayout

**SET** columns of removeGridLayout to 2

**SET** rows of removeGridLayout to 1

**CREATE** JPanel removeTutJPanel

**ADD** removeTutJPanel to frame

**SET** bounds of removeTutJPanel to (710, 410, 200, 100)

**SET** border of removeTutJPanel to removetutBorder

**SET** layout of removeTutJPanel to removeGridLayout

**CREATE** JLabel teacheridJPanelTutor with text "Teacher ID"

**ADD** teacheridJPanelTutor to removeTutJPanel

**CREATE** JTextField teacehrIDTutor

**ADD** teacehrIDTutor to removeTutJPanel

**CREATE** JButton removeTutor with text "Remove tutor"

**ADD** removeTutor to frame

**SET** bounds of removeTutor to (730, 510, 150, 50)

**Register** ActionListener to removeTutor:

**Function** actionPerformed(ActionEvent e)

**CALL** removeTutor()

**CREATE** JButton display with text "Display"

**ADD** display to frame

**SET** bounds of display to (600, 650, 150, 30)

**SET** frame visibility to true

**CREATE** JButton clear with text "Clear"

**ADD** clear to frame

**SET** bounds of clear to (750, 650, 150, 30)

**ADD** ActionListener to clear:

**Function** actionPerformed(ActionEvent e)

**CALL** clearAllTextFields()

**Function** addTutor

**TRY**

**CONVERT** getTextproper(teacheridTU) TO INTEGER, STORE AS teacherID

**GET** getTextproper(teacheridTU), STORE AS teacherName

**GET** getTextproper(addressTU), STORE AS address

**GET** getTextproper(workTypeTU), STORE AS workType

**GET** getTextproper(employementStatusTU), STORE AS employeeStatus

**CONVERT** getTextproper(workingHrTU) TO INTEGER, STORE AS workingHr

**CONVERT** getTextproper(salaryTU) TO INTEGER, STORE AS salary

**GET** getTextproper(specializationTU), STORE AS specialization

**GET** getTextproper(acedemicQualificationTU), STORE AS academicQualifications

**CONVERT** getTextproper(performanceIndexTU) TO INTEGER, STORE AS

performanceIndex

**IF** (teacherID is not unique)

**DISPLAY** JOptionPane error message "Id already exist"

**RETURN**

**CREATE** new Tutor object with obtained data

**ADD** Tutor object to teacherListArray

**DISPLAY** JOptionPane message "tutor added successfully.

**CATCH** NumberFormatException:

**DISPLAY** JOptionPane error message "Please enter the teacher ID, years of

experience, and working hours in numbers only"

**CATCH** IllegalArgumentException

**DISPLAY** JOptionPane error message "Fill all the fields"

**DEFINE** convertToInt method taking a String parameter text

**CONVERT** text TO INTEGER, STORE AS intValue

**RETURN** intValue

**Function** getTextproper method taking a JTextField parameter tf:

**GET** text from tf, **TRIM** whitespace, **STORE** as text

**IF** (text is empty)

**THROW** IllegalArgumentException

**RETURN** text

**FUNCTION** isIdUnique method taking an integer parameter id:

**FOR** each teacher in teacherListArray

**IF** teacher's teacherId is equal to id

**RETURN** false

**RETURN true**

**FUNCTION** gradeAssignment method:

**TRY:**

**GET** teacherId from getTextproper(teacherIdGATextField)

**GET** gradedScore from getTextproper(gradedScoreGATextField)

**GET** department from getTextproper(departmentGATextField)

**GET** yearsOfExp from getTextproper(yearsOfexpGATextField)

**FIND** teacher with given teacherId

**IF** teacher is null:

**SHOW** error message "Teacher with ID {teacherId} not found."

**RETURN**

**IF** yearsOfExp < 5:

**SHOW** error message "need to have more than 5 yrs of experience"

**RETURN**

**IF** teacher is not an instance of Lecturer:

**SHOW** error message "Only lecturers can grade assignments."

**RETURN**

**CAST** teacher to Lecturer

**ASK** user for confirmation

**IF** user confirms:

**SHOW** message "Assignment graded successfully."

**CALL** lecturer.gradeAssignment(gradedScore, department, yearsOfExp)

**RETURN**

**CATCH** NumberFormatException:

**SHOW** error message "Only enter valid numeric values"

**CATCH** IllegalArgumentException:

**SHOW** error message "Fill all the fields"

**FUNCTION** getTeacherById method taking int id:

**FOR** each Teacher in teacherListArray:

**IF** teacher.getTeacherId() equals id:

**RETURN** teacher

**RETURN** null

**FUNCTION** setSalary

**DECLARE** teacherId as INTEGER

**DECLARE** salary as INTEGER

**DECLARE** performanceIndex as INTEGER

**DECLARE** tutor as Tutor

**TRY**

**SET** teacherId = **CONVERT** input(teacherIdSalary) TO INTEGER

**SET** salary = **CONVERT** input(salarySalary) TO INTEGER

**SET** performanceIndex = **CONVERT** input(performanceIndexSalary) TO INTEGER

**Find** the tutor in the teacherArrayList based on teacherId

**SET** tutor = NULL

**FOR** EACH teacher IN teacherListArray

**IF** teacher IS INSTANCE OF Tutor AND teacher.getTeacherId() EQUALS

teacherId THEN

**SET** tutor = teacher

**BREAK**

**END** **IF**

**END** **FOR**

**IF** tutor IS NULL THEN

**DISPLAY** "Tutor not found with ID: " + teacherId, "Error"

**RETURN**

**IF** performanceIndex LESS THAN 5 OR tutor.getWorkingHr() LESS THAN OR

EQUAL TO 20 THEN

**DISPLAY** "cannot set salary performance index must be greater than 5 and

working hours must be greater than 20", "Error"

**RETURN**

**DISPLAY** "Are you sure? "

**IF** USER CLICKS OK THEN

**DISPLAY** "Salary set successfully.", "Success"

**CALL** tutor.setSalary(salary, performanceIndex)

**RETURN**

**CALL** tutor.setSalary(salary, performanceIndex)

**CATCH** NumberFormatException

**DISPLAY** "Error: Invalid input. Please enter numeric values for salary and

performance index.", "Error"

**CATCH** IllegalArgumentException

**DISPLAY** "Please enter all fields", "Error"

**END**

**FUNCTION** removeTutor

**CREATE** teacherId as INTEGER

**CREATE** tutor as Tutor

**TRY**

**SET** teacherId = **CONVERT** input(teacehrIDTutor) TO INTEGER

**FOR** **EACH** teacher **IN** teacherListArray

**IF** teacher IS INSTANCE OF Tutor AND teacher.getTeacherId() EQUALS

teacherId **THEN**

**SET** tutor = teacher

**BREAK**

**END** IF

**END** **FOR**

**IF** tutor IS NULL THEN

**DISPLAY** "Tutor not found with ID: " + teacherId, "Error"

**RETURN**

**IF** tutor.getIsCertified() EQUALS TRUE THEN

**DISPLAY** "Cannot remove a certified tutor", "Error"

**RETURN**

**DISPLAY** "Are you sure? "

**IF** USER CLICKS OK THEN

**CALL** tutor.removeTutor()

**DISPLAY** "Tutor removed successfully.", "Success"

**END** **IF**

**CATCH** NumberFormatException

**DISPLAY** "Error: Invalid input. Please enter valid teacher id", "Error"

**CATCH** IllegalArgumentException

**DISPLAY** "Please enter all fields", "Error"

**END**

**FUNCTION** clearAllTextFields

**SET** teacherIdTf TO ""

**SET** workingHrTf TO ""

**SET** teacherNameTf TO ""

**SET** addressTf TO ""

**SET** worktypeTf TO ""

**SET** employeeStatusTf TO ""

**SET** gradedScoreTf TO ""

**SET** yearsOfExperinceTf TO ""

**SET** dempartmentTF TO ""

**SET** teacherIdGATextField TO ""

**SET** gradedScoreGATextField TO ""

**SET** departmentGATextField TO ""

**SET** yearsOfexpGATextField TO ""

**SET** teacheridTU TO ""

**SET** teacherNameTU TO ""

**SET** addressTU TO ""

**SET** workTypeTU TO ""

**SET** employementStatusTU TO ""

**SET** workingHrTU TO ""

**SET** salaryTU TO ""

**SET** specializationTU TO ""

**SET** acedemicQualificationTU TO ""

**SET** performanceIndexTU TO ""

**SET** teacherIdSalary TO ""

**SET** salarySalary TO ""

**SET** performanceIndexSalary TO ""

**SET** teacehrIDTutor TO ""

**END**

**FUNCTION** main

**CREATE** TeacherGUI

**END** PROCEDURE

# Method description

This method description is the explanation of the implementation of the different functions of the entire program.theri are total of 12 methods

Table 1: method description of the TeacherGUI class

|  |  |
| --- | --- |
| Methods | Description |
| addLectureInfo() | The function of this method is to collect inputs from GUI text fields, validate it for the correctness purpose and after validation it creates object of Lecturer which is then stored in array named as teacherListArray. |
| addTutor() | The function of this method is also to collect inputs form the GUI text fields and use those text fields data after validation to create an object of Tutor and after creating the object is then stored in Array list that goes by the name teacherListArray |
| convertToInt() | This function basically returns value in integer. It takes string value from the parameter and converts the given string value into integer using Integer.parsInt() from the Integer class of java. |
| getTextproper() | This function basically takes parameter in string and also returns String. The main function of this method is to take String and remove any empty spaces it has around the text. |
| isIdUnique() | This function return type is Boolean which means it only returns either true or false. The purpose of this function is to check if the given teacher id is unique within the array list |
| gradeAssignment() | This method basically helps with grading on assignment by Lecturer. It validates the input, checks if the teacher is a lecturer with sufficient experience, prompts for confirmation,then calls the grade Assignment() method of the   Lecturer object to perform the actual grading. It also handles various error scenarios and displays appropriate messages to the user |
| getTeacherById | The function of this method is to help retrieve a teacher object from the teacherListArray ArrayList based on the provided teacher ID. The return type of this function is Teacher as it returns the object of Teacher. |
| setSalary() | The setSalary() method in the TeacherGUI class is responsible for setting the salary of a tutor based on their performance index and working hours. It also validates the input, checks if the tutor meets the criteria for setting the salary, prompts for confirmation, and then calls the setSalary() method of the Tutor object to set the salary. It also handles various error scenarios and displays appropriate messages to the user. |
| removeTutor() | The function removetutor() facilitates the process of removing a tutor from the teacherListArray ArrayList. It validates the input, checks if the tutor is certified (and cannot be removed), prompts for confirmation, and then calls the removeTutor() method of the Tutor object to perform the removal. It also handles various error scenarios and displays appropriate messages to the user. |
| clearAllTextFields() | This method is typically called when the user clicks a "Clear" button or when the application needs to reset the input fields for any reason. In the TeacherGUI class, The clearAllTextFields() method is called when the user clicks the "Clear" button, which is set up with an ActionListener that calls this method. |
| Display() | The display() method in the TeacherGUI class is responsible for displaying the information of all the teachers (lecturers and tutors) stored in the teacherListArray ArrayList. |
| Main() | The main() method in the TeacherGUI class is the entry point of the Java application. It is a static method that is responsible for creating an instance of the TeacherGUI class and starting the graphical user interface (GUI) |
| actionPerformed() | It displays all the action initialize in the respective method |

# Testing

## Testing 1 : Compiling & Running using command prompt

Table 2: test 1 compiling and running

|  |  |
| --- | --- |
| Objective | Testing weather the program can be compiled and run using command prompt (Terminal) |
| Action | Opened command prompt run command to reach to that file location using path  Feed command javac TeacherGui.java for compiling and java TeacherGui.java to run the file |
| Expected result | The program must compile and the Guimust be displayed on the screen |
| Actual result | The program compiled successfully and the after running the Gui window poped up on the screen |
| Conclusion | Test was successful |

**Evidence:**

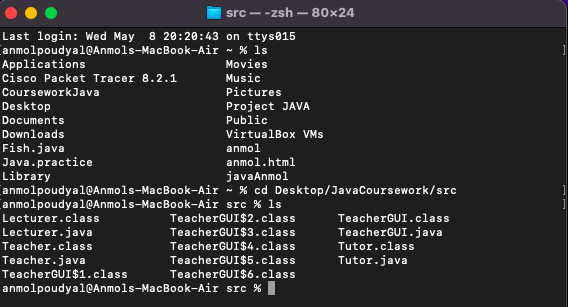


Figure 6: Reaching to the file through path

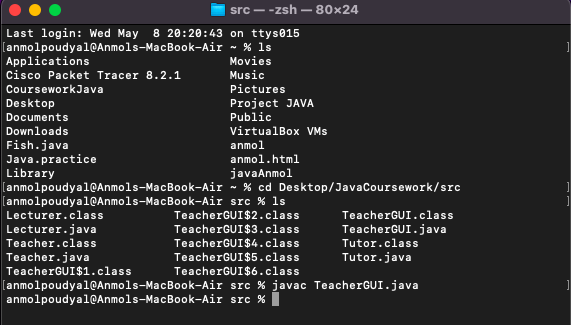


Figure 7: successful compilation



Figure 8: Running the program after comilation

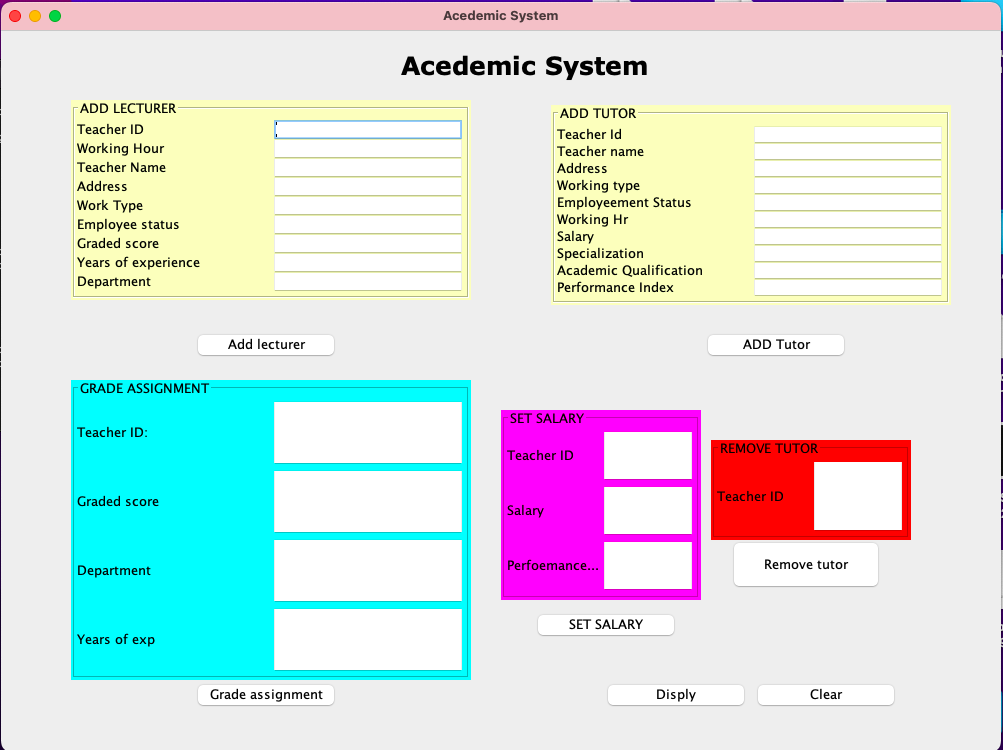


Figure 9 Displayed GUI after running

## Testing 2 : Adding objects of Tutor and Lecturer, grade assignments amount from lecturer, setting the salary and removing the tutor

**Testing 2.1: Adding Lecturer**

Table 3: Testing 1 adding Lecturer

|  |  |
| --- | --- |
| Objective | To add lecturer into the |
| Action | * The add lecture text field was filled with details :teacher id =1, workinghr=12 teacher name = anmol, address = urlabari, work type = GTA, employee status = acyive, graded score = 0, years of exp =12 and department = BIT and then the add tutor button was clicked |
| Expected result | The tutor must be successfully added and the dilogue box pops up on the screen |
| Actual result | The tutor was successfully added and the dilogue box pops up on the screen |
| Conclusion | The test was successful |

A screenshot of a computer

Description automatically generated

Figure 10: Entering the info of lecturer

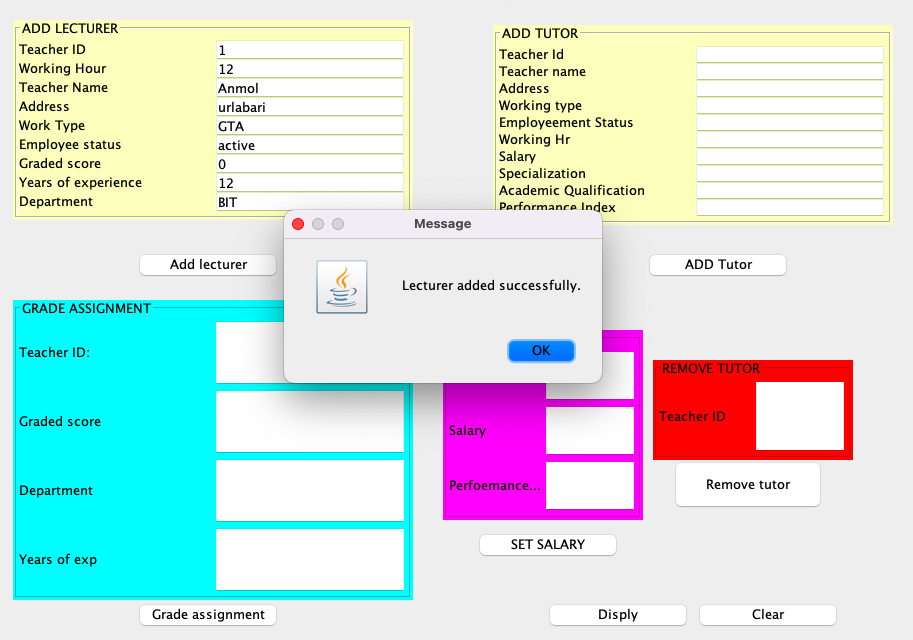


Figure 11: after the tutor button was clicked the oject got stored in list

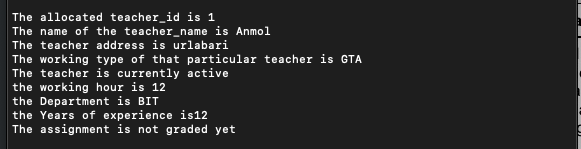


Figure 12: displaying in terminal after adding the object of lecturer

**Testing 2.2: Adding tutor**

Table 4: test 2 Adding tutor

|  |  |
| --- | --- |
| Objective | To add tutor in the syste |
| Action | The tutor text field was filled with detials teacher id =2,  teacher name= sabin,  address = urlabari,  work type = GTA,  employee status = active,  working hr =14,  slary =10000,  specialization = database,  academic qualification = MIT performance index=12 and the Grade assignment buttons was clicked |
| Expected result | The tutor must be added successfully and the GUI must display dialogue box informing user that the performed action was successful |
| Actual Result | The tutor was added successfully and the dialogue box also appeared informing the user about adding the user successfully |
| Connclusion | The test was Successful |

**Evidence**

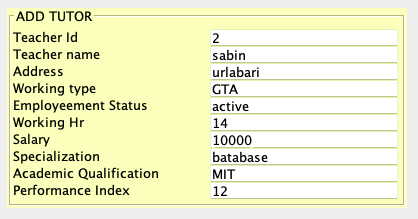


Figure 13: adding tutor details to create tutor object

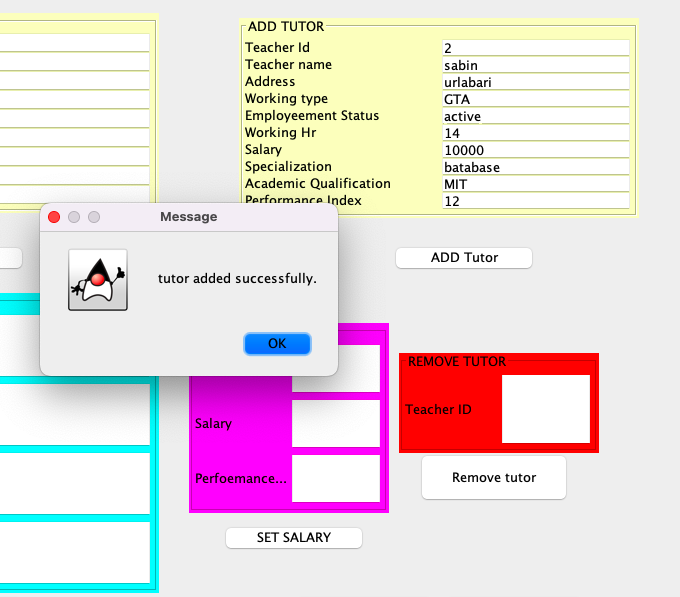


Figure 14: after adding tutor

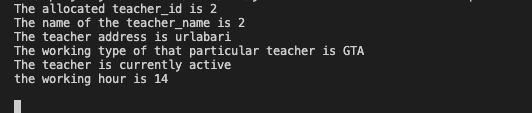


Figure 15: Add tutor displayed tutor in terminal

## Testing 2.3: Grade Assignments from Lecturer

Table 5: Testing 3 Grading assignment

|  |  |
| --- | --- |
| Objective | TO Grade Assignments from Lecturer |
| Action | The tutor text field was filled with details of the teacher we want to grade assignment with:  teacher\_id = 1  GradedScore = 50  Department = BIT  Years Of exp = 12 and the Grade assignment button was added. |
| Expected result | The message dialogue was poped and the assignment graded message must be displayed in CLI |
| Actual result | The message was displayed through dialogu box and also displayed msg in CLI |
| Conclusion | Test was successful. |

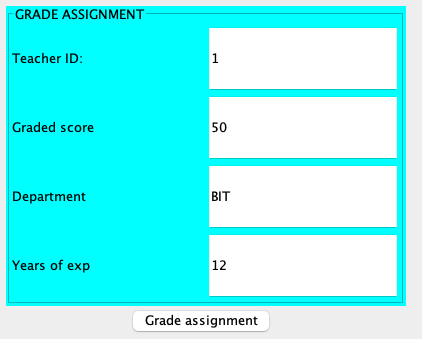


Figure 16: inputs for the Grade assignment of teacherID 1

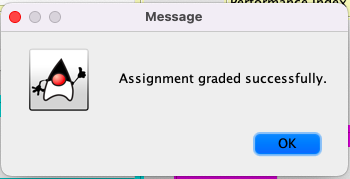
****

Figure 17: Displayed after grading assignment



Figure 18: message displayed after grading assignment in CLI

Testing 2.4: **Set the salary**

Table 6: testing 4 Setting salary for tutor

|  |  |
| --- | --- |
| Objective | To set the salary for the added tutor |
| Action | All the text field was filled with teacher id 3  Salary = 25000  Performanceindex = 12  Before that add a tutor with following details  teacher id =3,  teacher name= sabin,  address = urlabari,  work type = GTA,  employee status = active,  working hr =22,  slary =10000,  specialization = database,  academic qualification = MIT performance index=12 |
| Expected result | The tutor must be declared. Salary and the message dialogue appears along with the information on the CLI(Terminal window) |
| Actual result | The tutor salary was set along with the message dialogue about the setting og the salary and message was displayed on CLI |
| Conclusion | The test was successful |

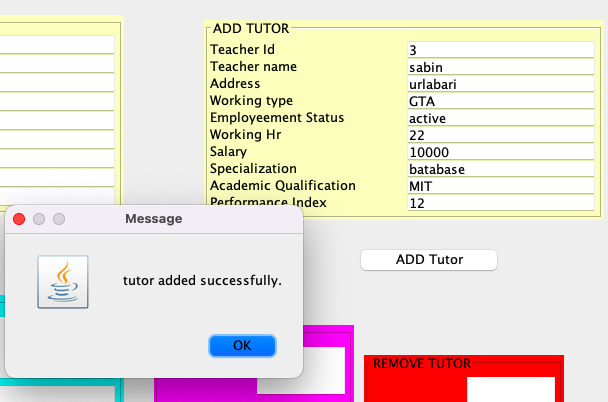


Figure 19: after adding tutor for setting salary

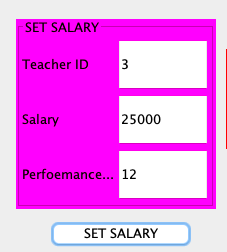


Figure 20: inputs for the seting salary of teacher id 3

A screenshot of a computer

Description automatically generated

Figure 21: dialogue box after setting salary

**Test 2.5 Remove tutor**

Table 7: test 5 removing Tutor

|  |  |
| --- | --- |
| Objective | To remove tutor(object of the tutor) |
| Action | The text field was filled with data including the teacher id = 3 which exists from test 2 |
| Expected result | The tutor with the above teacher Id must be removed |
| Actual result | The tutor with the id 3 was removed |
| Conclusion | The test was successful |

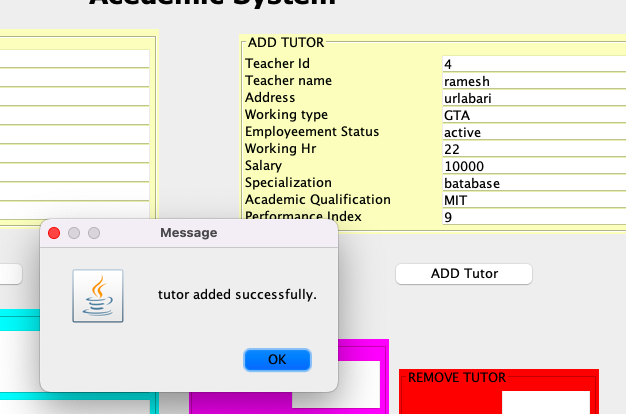


Figure 22: adding new tutor for removing

A screenshot of a computer

Description automatically generated

Figure 23: Before removing tutor

A red rectangular object with white squares and black text

Description automatically generated

Figure 24: Entering text field for removing tutor

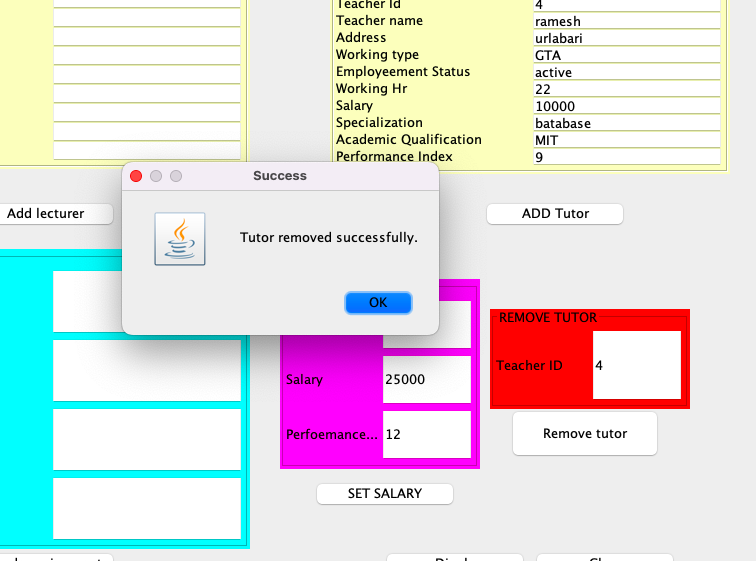


Figure 25: after remove tutor button was clicked

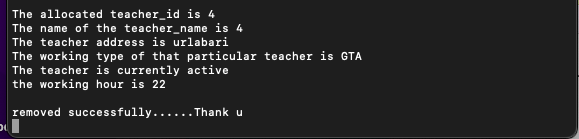


Figure 26: in terminal after removing tutor

## Test 3

**Testing 3.1 Try to add teacher repeating teacher id**

Table 8: test 6 add teacher with repeating id

|  |  |
| --- | --- |
| Objective | Try to add teacher repeating teacher id |
| Action | Giving input of same ID number here we have set teacher Id to be declared 2 times |
| Expected result | Error message must be shown as soon as user hits the button. |
| Actual result | Error message was displayed as soon as user clicked the button |
| conclusion | The test was successful |

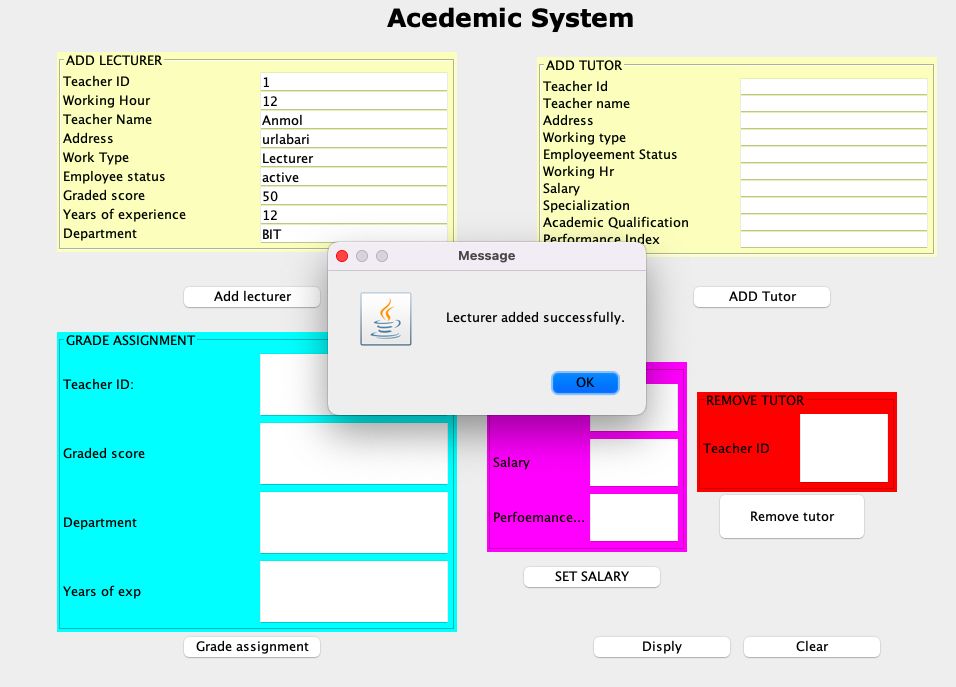


Figure 27: adding tutor with teach id 1

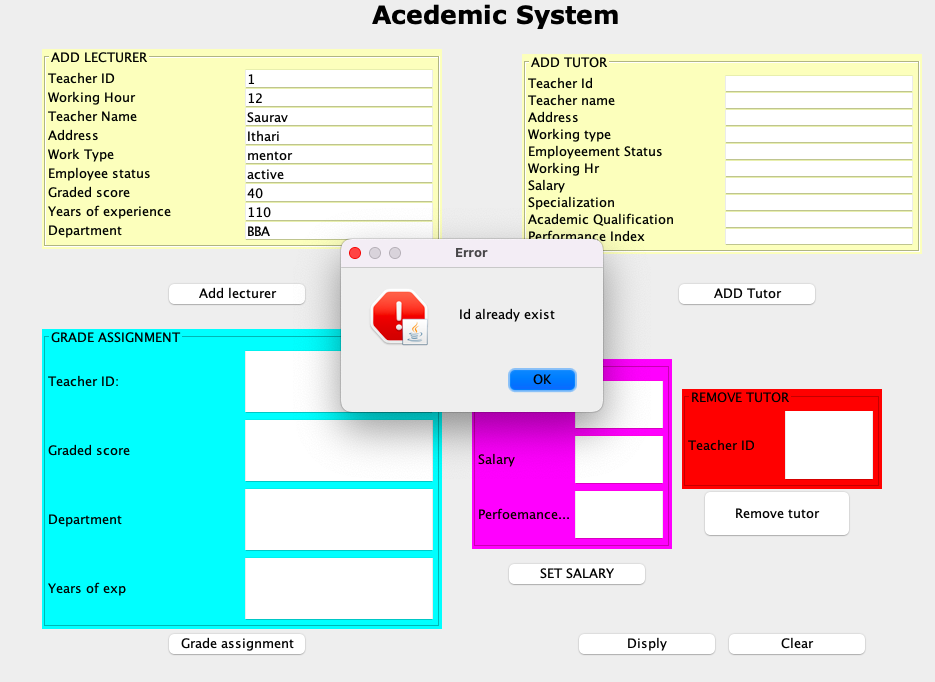


Figure 28: Output in dialogue after entering same teacher id to add lecturer

**Testing 3.2 remove tutor on a lecturer.**

Table 9 Test 7 removing tutor

|  |  |
| --- | --- |
| Objective | Try to remove tutor on a lecturer. |
| Actions | From test 1 teacher id 1 is a lecturer object adding data the teacher id of lecture calss 1 in remove tutor |
| Expected Result | Appropriate error message must be displayed |
| Actual Result | Appropriate error message was displayed |
| Conclusion | The test was successful |

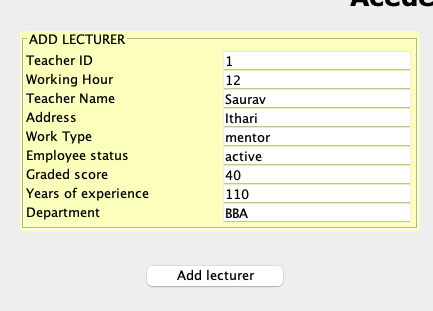


Figure 29: info of ID 1 as a lecturer

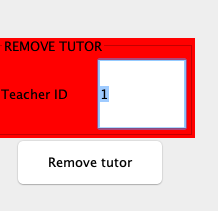


Figure 30: trying to remove lecturer with id 1 in tutor method

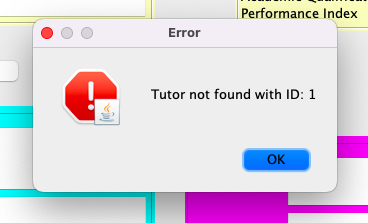


Figure 31: Error message after entering lecturer id in remove tutor

Testing 3.3 Set salary on a lecturer.

Table 10: test 8 setting salary for lecturer

|  |  |
| --- | --- |
| Objective | Try to Set salary on a lecturer. |
| Actions | From test 3.1 teacher id 1 is a lecturer object adding data the teacher id and other info in set salary  Fields entered in set salary  Teacher id =1  Salary =10000  Performance Index = 12 |
| Expected Result | Appropriate error message must be displayed |
| Actual Result | Appropriate error message was displayed |
| Conclusion | The test was successful |

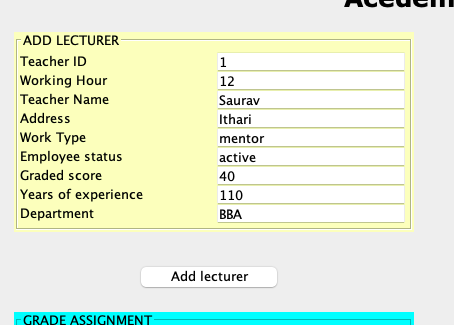


Figure 32: teacher info of lecturer with teacher id 1

A screenshot of a computer screen

Description automatically generated

Figure 33: entering info of lecturer in set salary

A screenshot of a computer

Description automatically generated

Figure 34: resu;t after setting salary for lecturer id

**Testing 3.4** Try to grade assignment on a tutor.

Table 11: test 9 to grade assignment on a tutor object

|  |  |
| --- | --- |
| Objective | Try to grade assignment on a tutor. |
| Action | Object of the tutor was created and the same tutor id was used to grade assignment |
| Expected result | Suitable message must be displayed rejecting to grade assignment by the tutor |
| Actual result | Suitable message was displayed rejecting to gradeassignment |
| Conclusion | The test was successful |

A screenshot of a computer

Description automatically generated

Figure 35: createing object of tutor

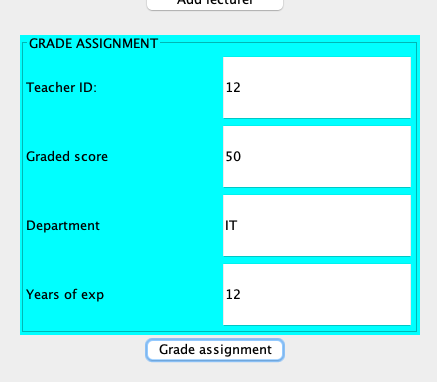


Figure 36: filling the fields of grade assignment with tutor id

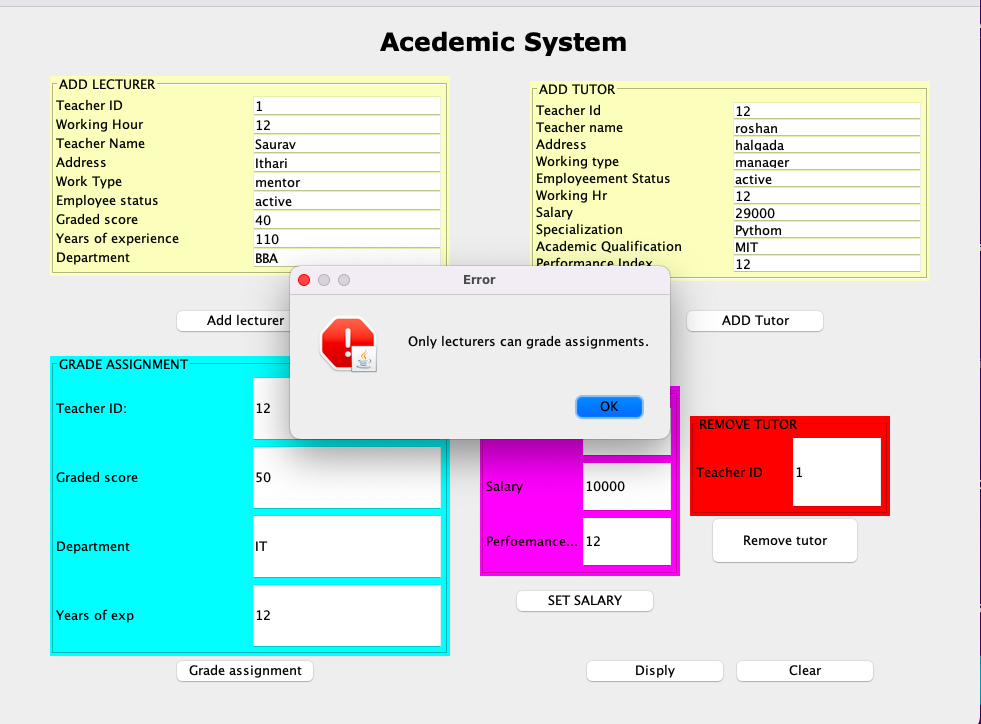


Figure 37: error message showing only lecturers can grade assignment

# Error Detection and Correction

## Syntax error

Syntax error is a language or a gramatical error especially in programming language. Syntax error actually referes to missing syntax or wrong standard structure.

Here in the program also I have encountered syntax error during the developnment of my code.

A screenshot of a computer

Description automatically generated

Figure 38: syntax error missing part

Here in the above a semicolon was missing which occurred to be a problem and was shoeing ; expected

I resolved the issue by adding a missing semicolon

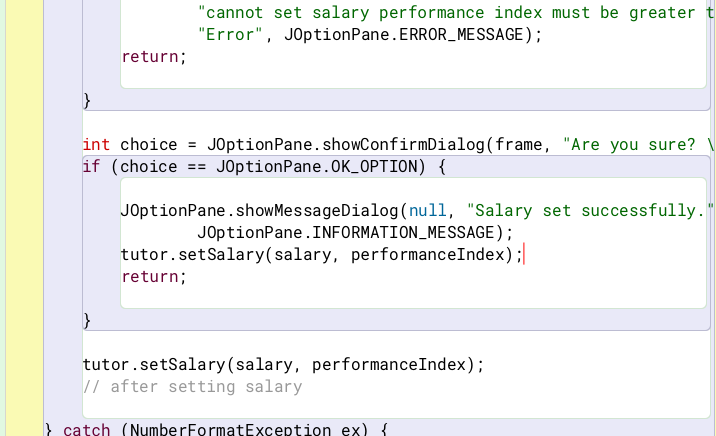


Figure 39 Syntax errror solved

## Logical error

Logical error is classified as a run time error that could cause a program to process unexpected output. It is most of the difficult error to handle. During the development of my project I had also faced lots of logical error. Here, I have error that I had faced during my project:

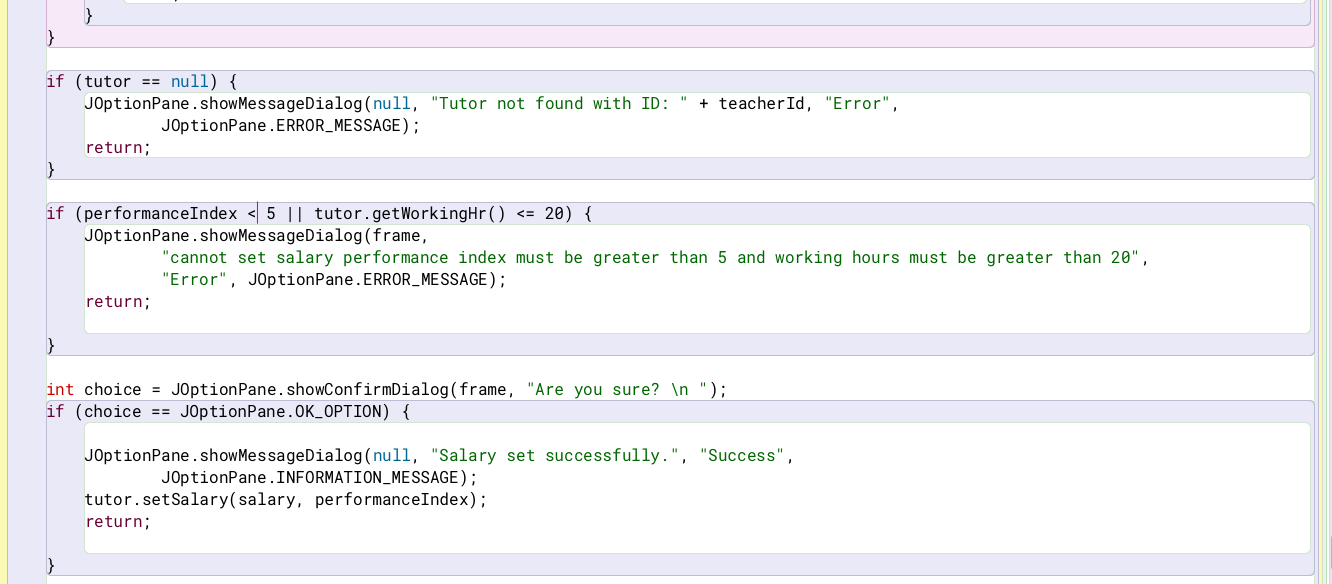




Figure 40: Logical error example

Here I had entered wrong comparision operator which didn’t allowed me to to performed my task as desired. Inorder to solve it I have changed the greater than operator

A screenshot of a computer program

Description automatically generated



## Semantic error

Semantic errors are classified as Compile time error. This type of error mostly occurs during the use of GUI as we have to deal with lots of variables and their names. During my project I came across lots of semantic errors like in the figure 41 here the teacher id is in int and we have entered String value which displays dialogue box of error

A screenshot of a computer

Description automatically generated

Figure 41: semantic error detection

The error is solved by entering the integer value for the textfield as shown in the figure below

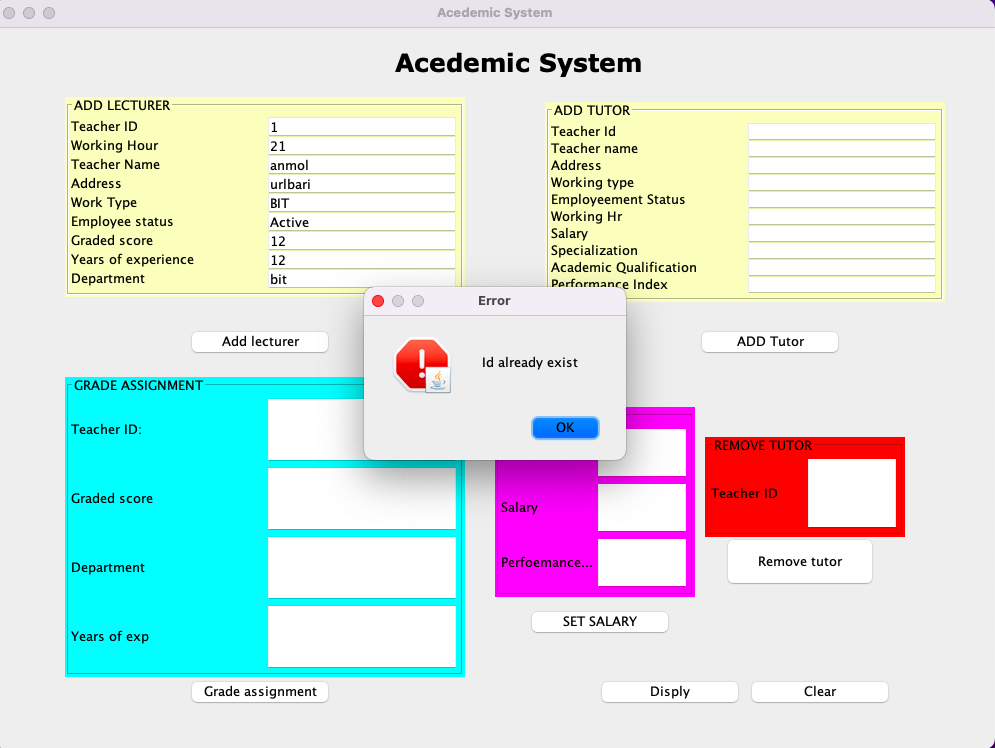


Figure 42: correction of Semantic error

# Conclusion

To conclude with this coursework was assigned to us for making GUI of a Teacher record system. During the coursework i have learned lots of things relating to the UI especially GUI. Through out the coursework I got chance to learn lots of different thing through out my research.

This project has provided valuable insights into the intricacies of event-driven programming and graphical user interface design principles. We've navigated challenges, such as managing user interactions, maintaining responsiveness, and ensuring all the validation required.

Overall it was a fun experience I also would like thank our module experties for clearing our doubts when we had one all in all it was a good exper

# References

(AWS, 2023)

(Agile, 2023)

(BLUEJ, 2024)

(IBM, 2023)

(Juviler, April 03, 2024)

(Juviler, April 03, 2024)

# Appendix

import javax.swing.BorderFactory;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JOptionPane;

import javax.swing.JPanel;

import javax.swing.JTextField;

import javax.swing.border.Border;

import java.awt.Color;

import java.awt.GridLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.ArrayList;

import java.awt.Font;

public class TeacherGUI {

private JTextField workingHrTf;

private JTextField teacherNameTf;

private JTextField teacherIdTf;

private JTextField addressTf;

private JTextField worktypeTf;

private JTextField employeeStatusTf;

private JTextField departmentTf;

private JTextField gradedScoreTf;

private JTextField yearsOfExperinceTf;

private JTextField salaryTf;

private JTextField specializationTf;

private JTextField academicQualificationTf;

private JTextField performanceIndex;

private JFrame frame;

private JButton addlecturer;

private JTextField teacherIdGATextField;

private JTextField gradedScoreGATextField;

private JTextField departmentGATextField;

private JTextField yearsOfexpGATextField;

private JTextField dempartmentTF;

JButton gradeAssignment;

// instance var for the tutor panel section

private JTextField teacheridTU;

private JTextField teacherNameTU;

private JTextField addressTU;

private JTextField workTypeTU;

private JTextField employementStatusTU;

private JTextField workingHrTU;

private JTextField salaryTU;

private JTextField specializationTU;

private JTextField acedemicQualificationTU;

private JTextField performanceIndexTU;

private JButton addTutorButton;

// instance for setsalary

private JTextField teacherIdSalary;

private JTextField salarySalary;

private JTextField performanceIndexSalary;

private JButton setsalary;

// instance remove tutor instance

private JTextField teacehrIDTutor;

private JButton removeTutor;

// instances for the button

private JButton clear;

private JButton display;

// array to store teachers object

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

// The methods are below:>>

public TeacherGUI() {

// for the main frame

this.frame = new JFrame("Acedemic System");

frame.setSize(1000, 750);

frame.setLocationRelativeTo(null);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setLayout(null);

frame.setBackground(Color.RED);

// for the layout

GridLayout lectureGridLayout = new GridLayout();

lectureGridLayout.setColumns(2);

lectureGridLayout.setRows(9);

// gradeassignment lecturer panel lay out

GridLayout gradeassignmenLayout = new GridLayout();

gradeassignmenLayout.setColumns(2);

gradeassignmenLayout.setRows(4);

GridLayout tutoGridLayout = new GridLayout();

tutoGridLayout.setColumns(2);

tutoGridLayout.setRows(10);

GridLayout setSalaGridLayout = new GridLayout();

setSalaGridLayout.setColumns(2);

setSalaGridLayout.setRows(3);

// for the boarder line in panels

Border teacherinputBorder = BorderFactory.createTitledBorder("ADD LECTURER"); // for lecturer panel

Border gradeAssignmBorder = BorderFactory.createTitledBorder("GRADE ASSIGNMENT");// for gradeassignment panel

// panel

Border tutorpanelborder = BorderFactory.createTitledBorder("ADD TUTOR");

Border setsalaryBorder = BorderFactory.createTitledBorder("SET SALARY");

Border removetutBorder = BorderFactory.createTitledBorder("REMOVE TUTOR");

// for the toplable

JLabel toplable = new JLabel("Acedemic System");

frame.add(toplable);

toplable.setBounds(400, 10, 250, 50);

toplable.setFont(new Font("Verdana", Font.BOLD, 25));

// panel for the lecturer

JPanel teacherJPanel = new JPanel();

frame.add(teacherJPanel);

teacherJPanel.setBounds(70, 70, 400, 200);

teacherJPanel.setLayout(lectureGridLayout);

teacherJPanel.setBorder(teacherinputBorder);

teacherJPanel.setBackground(Color.getHSBColor(134, 56, 50));

// components for the addlecturer panel

JLabel teacherId = new JLabel("Teacher ID");

teacherJPanel.add(teacherId);

this.teacherIdTf = new JTextField();

teacherJPanel.add(teacherIdTf);

JLabel workingHr = new JLabel("Working Hour");

teacherJPanel.add(workingHr);

this.workingHrTf = new JTextField();

teacherJPanel.add(workingHrTf);

JLabel teacherName = new JLabel("Teacher Name");

teacherJPanel.add(teacherName);

this.teacherNameTf = new JTextField();

teacherJPanel.add(teacherNameTf);

JLabel address = new JLabel("Address");

teacherJPanel.add(address);

this.addressTf = new JTextField();

teacherJPanel.add(addressTf);

JLabel workType = new JLabel("Work Type");

teacherJPanel.add(workType);

this.worktypeTf = new JTextField();

teacherJPanel.add(worktypeTf);

JLabel employeeStatus = new JLabel("Employee status");

teacherJPanel.add(employeeStatus);

this.employeeStatusTf = new JTextField();

teacherJPanel.add(employeeStatusTf);

JLabel gradedScore = new JLabel("Graded score");

teacherJPanel.add(gradedScore);

this.gradedScoreTf = new JTextField();

teacherJPanel.add(gradedScoreTf);

JLabel yearsofexp = new JLabel("Years of experience");

teacherJPanel.add(yearsofexp);

this.yearsOfExperinceTf = new JTextField();

teacherJPanel.add(yearsOfExperinceTf);

JLabel department = new JLabel("Department");

teacherJPanel.add(department);

this.dempartmentTF = new JTextField();

teacherJPanel.add(dempartmentTF);

// adding button for the add lecturer button

this.addlecturer = new JButton("Add lecturer");

frame.add(addlecturer);

addlecturer.setBounds(190, 300, 150, 30);

addlecturer.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

addLectureInfo();

}

});

// panel for grade assignment

JPanel gradeassignmentpanel = new JPanel();

frame.add(gradeassignmentpanel);

gradeassignmentpanel.setBounds(70, 350, 400, 300);

gradeassignmentpanel.setBorder(gradeAssignmBorder);

gradeassignmentpanel.setLayout(gradeassignmenLayout);

gradeassignmentpanel.setBackground(Color.cyan);

// adding components for the gradeassignment panels like,,,,gradedscore,teacher

// id,deppartmnt,years of exp

JLabel teacherIdGA = new JLabel("Teacher ID:");

gradeassignmentpanel.add(teacherIdGA);

this.teacherIdGATextField = new JTextField();

gradeassignmentpanel.add(teacherIdGATextField);

JLabel gradedScoreGA = new JLabel("Graded score");

gradeassignmentpanel.add(gradedScoreGA);

this.gradedScoreGATextField = new JTextField();

gradeassignmentpanel.add(gradedScoreGATextField);

JLabel departmentGA = new JLabel("Department");

gradeassignmentpanel.add(departmentGA);

this.departmentGATextField = new JTextField();

gradeassignmentpanel.add(departmentGATextField);

JLabel yearsofexpGA = new JLabel("Years of exp");

gradeassignmentpanel.add(yearsofexpGA);

this.yearsOfexpGATextField = new JTextField();

gradeassignmentpanel.add(yearsOfexpGATextField);

// adding button for the grade assignment

this.gradeAssignment = new JButton("Grade assignment");

frame.add(gradeAssignment);

gradeAssignment.setBounds(190, 650, 150, 30);

gradeAssignment.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

gradeAssignment();

}

});

// panel for adding tutor

JPanel tutorPanel = new JPanel();

frame.add(tutorPanel);

tutorPanel.setBounds(550, 75, 400, 200);

tutorPanel.setBorder(tutorpanelborder);

tutorPanel.setLayout(tutoGridLayout);

tutorPanel.setBackground(Color.getHSBColor(134, 56, 50));

// adding components for the tutor panel

JLabel teacherIdJLabelTU = new JLabel("Teacher Id");

tutorPanel.add(teacherIdJLabelTU);

this.teacheridTU = new JTextField();

tutorPanel.add(teacheridTU);

JLabel teacherNameLabelTU = new JLabel("Teacher name ");

tutorPanel.add(teacherNameLabelTU);

this.teacherNameTU = new JTextField();

tutorPanel.add(teacherNameTU);

JLabel addressJLabelTU = new JLabel("Address");

tutorPanel.add(addressJLabelTU);

this.addressTU = new JTextField();

tutorPanel.add(addressTU);

JLabel workingtypeJLabelTU = new JLabel("Working type");

tutorPanel.add(workingtypeJLabelTU);

this.workTypeTU = new JTextField();

tutorPanel.add(workTypeTU);

JLabel employeeStatusJlableTU = new JLabel("Employeement Status");

tutorPanel.add(employeeStatusJlableTU);

this.employementStatusTU = new JTextField();

tutorPanel.add(employementStatusTU);

JLabel workinHrJLabelTU = new JLabel("Working Hr");

tutorPanel.add(workinHrJLabelTU);

this.workingHrTU = new JTextField();

tutorPanel.add(workingHrTU);

JLabel salaryJLabelTU = new JLabel("Salary");

tutorPanel.add(salaryJLabelTU);

this.salaryTU = new JTextField();

tutorPanel.add(salaryTU);

JLabel specializationtJLabelTU = new JLabel("Specialization");

tutorPanel.add(specializationtJLabelTU);

this.specializationTU = new JTextField();

tutorPanel.add(specializationTU);

JLabel academicqualificationJLabelTU = new JLabel("Academic Qualification");

tutorPanel.add(academicqualificationJLabelTU);

this.acedemicQualificationTU = new JTextField();

tutorPanel.add(acedemicQualificationTU);

JLabel performanceIndexJLabelTU = new JLabel("Performance Index");

tutorPanel.add(performanceIndexJLabelTU);

this.performanceIndexTU = new JTextField();

tutorPanel.add(performanceIndexTU);

// button for adding the tutor

this.addTutorButton = new JButton("ADD Tutor");

frame.add(addTutorButton);

addTutorButton.setBounds(700, 300, 150, 30);

addTutorButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

addTutor();

}

});

// panel for setsalary part

JPanel setsalaryJPanel = new JPanel();

frame.add(setsalaryJPanel);

setsalaryJPanel.setBounds(500, 380, 200, 190);

setsalaryJPanel.setBorder(setsalaryBorder);

setsalaryJPanel.setLayout(setSalaGridLayout);

setsalaryJPanel.setBackground(Color.MAGENTA);

// set salary components

JLabel teacheriDJPanelSalary = new JLabel("Teacher ID");

setsalaryJPanel.add(teacheriDJPanelSalary);

this.teacherIdSalary = new JTextField();

setsalaryJPanel.add(teacherIdSalary);

JLabel salarylJLabelsalary = new JLabel("Salary");

setsalaryJPanel.add(salarylJLabelsalary);

this.salarySalary = new JTextField();

setsalaryJPanel.add(salarySalary);

JLabel performanceIndexJLabelSalary = new JLabel("Perfoemance index");

setsalaryJPanel.add(performanceIndexJLabelSalary);

this.performanceIndexSalary = new JTextField();

setsalaryJPanel.add(performanceIndexSalary);

// button for setting slary

this.setsalary = new JButton("SET SALARY");

frame.add(setsalary);

setsalary.setBounds(530, 580, 150, 30);

setsalary.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

setSalary();

}

});

// layout for remove tutor

GridLayout removeGridLayout = new GridLayout();

removeGridLayout.setColumns(2);

removeGridLayout.setRows(1);

// for the remove tutor

JPanel removeTutJPanel = new JPanel();

frame.add(removeTutJPanel);

removeTutJPanel.setBounds(710, 410, 200, 100);

removeTutJPanel.setBorder(removetutBorder);

removeTutJPanel.setLayout(removeGridLayout);

removeTutJPanel.setBackground(Color.getHSBColor(36, 65, 76));

JLabel teacheridJPanelTutor = new JLabel("Teacher ID");

removeTutJPanel.add(teacheridJPanelTutor);

this.teacehrIDTutor = new JTextField();

removeTutJPanel.add(teacehrIDTutor);

// button for removing tutor

this.removeTutor = new JButton("Remove tutor");

frame.add(removeTutor);

removeTutor.setBounds(730, 510, 150, 50);

removeTutor.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

removeTutor();

}

});

// Overall display button

this.display = new JButton("Disply");

frame.add(display);

display.setBounds(600, 650, 150, 30);

display.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

display();

}

});

frame.setVisible(true);

// clear button

this.clear = new JButton("Clear");

frame.add(clear);

clear.setBounds(750, 650, 150, 30);

clear.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

clearAllTextFields();

}

});

}

public void addLectureInfo() {

try {

int teacherId = convetToInt(getTextproper(teacherIdTf));

String teacherName = getTextproper(teacherNameTf);

String address = getTextproper(addressTf);

String workingType = getTextproper(worktypeTf);

String employmentStatus = getTextproper(employeeStatusTf);

String department = getTextproper(dempartmentTF);

int yearsOfExp = convetToInt(getTextproper(yearsOfExperinceTf));

int workingHr = convetToInt(getTextproper(workingHrTf));

if (!isIdUnique(teacherId)) {

JOptionPane.showMessageDialog(frame, "Id already exist", "Error", JOptionPane.ERROR\_MESSAGE);

return;

}

// Create a new Lecturer object

Lecturer lecturer = new Lecturer(teacherId, teacherName, address, workingType, employmentStatus, department,

yearsOfExp, workingHr);

// Add the lecturer to the ArrayList of teachers

teacherListArray.add(lecturer);

// Optionally, display a message or perform other actions

JOptionPane.showMessageDialog(frame, "Lecturer added successfully.");

} catch (NumberFormatException e) {

JOptionPane.showMessageDialog(frame,

"Invalid entry 'teacherid,years of experience and working hours must be in numbers only'", "ERROR",

JOptionPane.ERROR\_MESSAGE);

} catch (IllegalArgumentException e) {

JOptionPane.showMessageDialog(frame, "Fill all the fields", "ERROR", JOptionPane.ERROR\_MESSAGE);

}

}

// this is for adding the tutor to the list

public void addTutor() {

try {

int teacherID = convetToInt(getTextproper(teacheridTU));

String teacherName = getTextproper(teacheridTU);

String address = getTextproper(addressTU);

String workType = getTextproper(workTypeTU);

String employeeStatus = getTextproper(employementStatusTU);

int workingHr = convetToInt(getTextproper(workingHrTU));

int salary = convetToInt(getTextproper(salaryTU));

String specialization = getTextproper(specializationTU);

String academicQualifications = getTextproper(acedemicQualificationTU);

int performanceIndex = convetToInt(getTextproper(performanceIndexTU));

if (!isIdUnique(teacherID)) {

JOptionPane.showMessageDialog(frame, "Id already exist", "Error", JOptionPane.ERROR\_MESSAGE);

return;

}

Tutor tutor = new Tutor(teacherID, teacherName, address, workType, employeeStatus, workingHr, salary,

specialization, academicQualifications, performanceIndex);

teacherListArray.add(tutor);// Add the tutor to the ArrayList of teachers

// Optionally, display a message or perform other actions

JOptionPane.showMessageDialog(frame, "tutor added successfully.");

} catch (NumberFormatException e) {

JOptionPane.showMessageDialog(frame,

"hi please enter The teacherid,years of experience and working hours in numbers only", "ERROR",

JOptionPane.ERROR\_MESSAGE);

} catch (IllegalArgumentException e) {

JOptionPane.showMessageDialog(frame, "Fill all the fields", "ERROR", JOptionPane.ERROR\_MESSAGE);

}

}

// for converting integer from string in th field

public int convetToInt(String text) {

int intValue = Integer.parseInt(text);

return intValue;

}

// this is for getting proprer trimed text from the textfield

public String getTextproper(JTextField tf) {

String text = tf.getText().trim();

if (text.isEmpty()) {

throw new IllegalArgumentException();

}

return text;

}

/// this method is for the checking if id is unque or not in a list

private boolean isIdUnique(int id) {

for (Teacher teacher : teacherListArray) {

if (teacher.getTeacherId() == id) {

return false;

}

}

return true;

}

private void gradeAssignment() {

try {

int teacherId = convetToInt(getTextproper(teacherIdGATextField));

int gradedScore = convetToInt(getTextproper(gradedScoreGATextField));

String department = getTextproper(departmentGATextField);

int yearsOfExp = convetToInt(getTextproper(yearsOfexpGATextField));

// to find check given teacher ID

Teacher teacher = getTeacherById(teacherId);

if (teacher == null) {

JOptionPane.showMessageDialog(frame, "Teacher with ID " + teacherId + " not found.", "Error",

JOptionPane.ERROR\_MESSAGE);

return;

}

if (yearsOfExp < 5 ) {

JOptionPane.showMessageDialog(frame, "need to have more than 5 yrs of experience", "Error",

JOptionPane.ERROR\_MESSAGE);

return;

}

// Check if the teacher is a lecturer

if (!(teacher instanceof Lecturer)) {

JOptionPane.showMessageDialog(frame, "Only lecturers can grade assignments.", "Error",

JOptionPane.ERROR\_MESSAGE);

// after grading

return;

}

Lecturer lecturer = (Lecturer) teacher;

// Grade the assignment

int choice = JOptionPane.showConfirmDialog(frame, "Are you sure? \n ");

if (choice == JOptionPane.OK\_OPTION) {

JOptionPane.showMessageDialog(frame, "Assignment graded successfully.");

lecturer.gradeAssignment(gradedScore, department, yearsOfExp);

return;

}

} catch (NumberFormatException e) {

JOptionPane.showMessageDialog(frame, "Only enter valid numeric values", "Error", JOptionPane.ERROR\_MESSAGE);

} catch (IllegalArgumentException e) {

JOptionPane.showMessageDialog(frame, "Fill all the fields", "ERROR", JOptionPane.ERROR\_MESSAGE);

}

}

private Teacher getTeacherById(int id) {

for (Teacher teacher : teacherListArray) {

if (teacher.getTeacherId() == id) {

return teacher;

}

}

return null; // Return null if teacher with the given ID is not found

}

// ((Tutor) teacher).getPerformanceIndex()

public void setSalary() {

try {

int teacherId = convetToInt(getTextproper(teacherIdSalary));

int salary = convetToInt(getTextproper(salarySalary));

int performanceIndex = convetToInt(getTextproper(performanceIndexSalary));

// Find the tutor in the teacherArrayList based on teacherId

Tutor tutor = null;

for (Teacher teacher : teacherListArray) {

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

tutor = (Tutor) teacher;

break;

}

}

if (tutor == null) {

JOptionPane.showMessageDialog(null, "Tutor not found with ID: " + teacherId, "Error",

JOptionPane.ERROR\_MESSAGE);

return;

}

if (performanceIndex < 5 || tutor.getWorkingHr() <= 20) {

JOptionPane.showMessageDialog(frame,

"cannot set salary performance index must be greater than 5 and working hours must be greater than 20",

"Error", JOptionPane.ERROR\_MESSAGE);

return;

}

int choice = JOptionPane.showConfirmDialog(frame, "Are you sure? \n ");

if (choice == JOptionPane.OK\_OPTION) {

JOptionPane.showMessageDialog(null, "Salary set successfully.", "Success",

JOptionPane.INFORMATION\_MESSAGE);

tutor.setSalary(salary, performanceIndex);

return;

}

tutor.setSalary(salary, performanceIndex);

// after setting salary

} catch (NumberFormatException ex) {

JOptionPane.showMessageDialog(null,

"Error: Invalid input. Please enter numeric values for salary and performance index.", "Error",

JOptionPane.ERROR\_MESSAGE);

} catch (IllegalArgumentException e) {

JOptionPane.showMessageDialog(null, "Please enter all fields", "Error", JOptionPane.ERROR\_MESSAGE);

}

}

public void removeTutor() {

try {

int teacherId = convetToInt(getTextproper(teacehrIDTutor));

Tutor tutor = null;

for (Teacher teacher : teacherListArray) {

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

tutor = (Tutor) teacher;

break;

}

}

if (tutor == null) {

JOptionPane.showMessageDialog(null, "Tutor not found with ID: " + teacherId, "Error",

JOptionPane.ERROR\_MESSAGE);

return;

}

if (tutor.getIsCertified() == true) {

JOptionPane.showMessageDialog(null, "Cannot remove a certified tutor", "Error",

JOptionPane.ERROR\_MESSAGE);

return;

}

// Remove the tutor

// teacherListArray.remove(tutor);

int choice = JOptionPane.showConfirmDialog(frame, "Are you sure? \n ");

if (choice == JOptionPane.OK\_OPTION) {

tutor.removeTutor();

teacherListArray.remove(tutor);

JOptionPane.showMessageDialog(null, "Tutor removed successfully.", "Success",

JOptionPane.INFORMATION\_MESSAGE);

} // choice

} catch (NumberFormatException ex) {

JOptionPane.showMessageDialog(null,

"Error: Invalid input. Please enter valid teacher id", "Error",

JOptionPane.ERROR\_MESSAGE);

} catch (IllegalArgumentException e) {

JOptionPane.showMessageDialog(null, "Please enter all fields", "Error", JOptionPane.ERROR\_MESSAGE);

}

}

// for clearing all the fields

public void clearAllTextFields() {

teacherIdTf.setText("");

workingHrTf.setText("");

teacherNameTf.setText("");

addressTf.setText("");

worktypeTf.setText("");

employeeStatusTf.setText("");

gradedScoreTf.setText("");

yearsOfExperinceTf.setText("");

dempartmentTF.setText("");

teacherIdGATextField.setText("");

gradedScoreGATextField.setText("");

departmentGATextField.setText("");

yearsOfexpGATextField.setText("");

teacheridTU.setText("");

teacherNameTU.setText("");

addressTU.setText("");

workTypeTU.setText("");

employementStatusTU.setText("");

workingHrTU.setText("");

salaryTU.setText("");

specializationTU.setText("");

acedemicQualificationTU.setText("");

performanceIndexTU.setText("");

teacherIdSalary.setText("");

salarySalary.setText("");

performanceIndexSalary.setText("");

teacehrIDTutor.setText("");

}

public void display() {

for (Teacher teacher : teacherListArray) {

teacher.displayInfo();

System.out.println();

JOptionPane.showMessageDialog(null, "Teacher Details in Terminal", "Success",

JOptionPane.INFORMATION\_MESSAGE);

}

}

public static void main(String[] args) {

new TeacherGUI();

}

}

# Plagiarism report

