CS4001NI Programming

30% Individual Coursework - 2

2023-24 Spring

Student Name: Sahil Bista

London Met ID: 22085598

College ID: NP01CP4S230127

Group: C20

Assignment Due Date: Friday, August 11, 2023

Assignment Submission Date: Thursday, August 10, 2023

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.

Table of Contents

Table Of Figures

Table of Figures:

List of Tables:

Acknowledgement

I would like to express my profound gratitude to the programming module teachers and Module leader Of Islington College for their contributions to the completion of my project. I would like to express my special thanks to our mentors, Mr. Ujjwal Subedi Sir ,Mr. Mohit Sharma Sir and Mr Bishal GC sir for your time and efforts that you have provided throughout the semester. Your useful advice and suggestions were really helpful to me during the project's completion. In this aspect, I'm eternally grateful to you. I would also like to acknowledge that this project was completed entirely by me and not by someone else.

-Sahil Bista

1.Introduction

1.1 Introduction to the project

Java Programming Language:

Java is an object oriented programming language which was first developed by James Gosling at Sun Microsystems,which is now a part of Oracle Corporation. It was released in 1995 As a part of Sun Microsystems Java Platform. Java has many features that makes it one of the most used and simplest programming languages,some of which are the program being object oriented, simple ,secured,platform independent,robust, portable,architecture neutral, dynamic, interpreted, high Performance, multi-threaded and distributed. Java programmes are not run directly by the operating system unlike Windows executable or Macintosh applications.Java programmes, on the other hand, are interpreted by the Java Virtual Machine(JVM), which runs on a variety of platforms.This means that all the Java programmes are multiplatform.Meaning they can operate on multiple platforms like mac, Windows and Unix machines.In order for Java to run,JRE must be installed.

The objective of this project is to apply the concepts of GUI(Graphical User Interface) in Java to provide a real-time platform/ interface for the users to run a program that references a student registration system. The project allows students like me to be enriched in knowledge regarding the user interface creation in Java programming. Here, the program provides an interface for different classes like Student, Regular and Dropout class which allows the users to inherit and use various methods that are used for registering student related datas. The Graphical User Interface makes it easier for the users to use those functions and keep the information about the students. So, the main objective/ gist of the program can be concluded to be an interface for all (both with and without the knowledge of programming).

Class:

In OOP, A class is a building block which can be defined as a template that describes the data and behaviour associated with the class. Creating a class is a preliminary step in writing a JAVA programme. In this project , a StudentGUI class has been created.

In the StudentGUI class, a code has been written to create a GUI that stores an arraylist of the type Student class to hold Regular and Dropout classes. Text fields have been created for entering fields such as: Student Name, Enrollment ID ,Course Name ,Course Duration , Tuition Fee , Number of Modules ,Number of Credit Hours ,Number of Days Present , Number of Remaining Modules , Number of Months Attended xi. Remaining Amount. Similarly , buttons like Add a Regular Student, Add a Dropout Student, Calculate Present Percentage , Grant certificate, Pay Bills , Remove Dropout , Display, Clear have been created to perform their corresponding actions.

1.2 Tools Used

1.2.1 Blue J

FIG 1: BLUEJ LOGO

Blue Jay is a Windows based platform Java Development Kit. It provides an easy to use environment that helps in learning the Java programming language. This tool was used to write the programming language.

1.2.2 MS Word

FIG 2: MS WORD

Microsoft Word is a word processor developed by Microsoft. It was first released on October 25, 1983, under the name Multi-Tool Word for Xenix systems. Word for Windows is available stand-alone or as part of the Microsoft Office suite. Word contains rudimentary desktop publishing capabilities and is the most widely used word processing program on the market. Word files are commonly used as the format for sending text documents. This tool is being used to write the report.

Reference: https://en.wikipedia.org/wiki/Microsoft_Word

1.2.3 Moqups

FIG 3 : MOQUPS LOGO

Moqups is a visual collaboration tool that combines whiteboard, diagram and design features in a single, online app. Moqups is used by over 2 million product managers, business analysts UX professionals, executives and cross-domain teams doing foundational work on complex projects. Here, moqups was used to make it easier for finding the setBounds axes and also for class diagrams making.

2.Class Diagram

A class diagram in java is a way to visually depict the components of a Java program,including classes,interfaces,objects as well as their attributes.They can be used to understand the layout of a java program and to automatically generate the corresponding code.There are various tools such as Eclipse,IntelliJ , Visual paradigm etc that can be used to create class diagrams in Java.

2.1 Class Diagram of Student class:

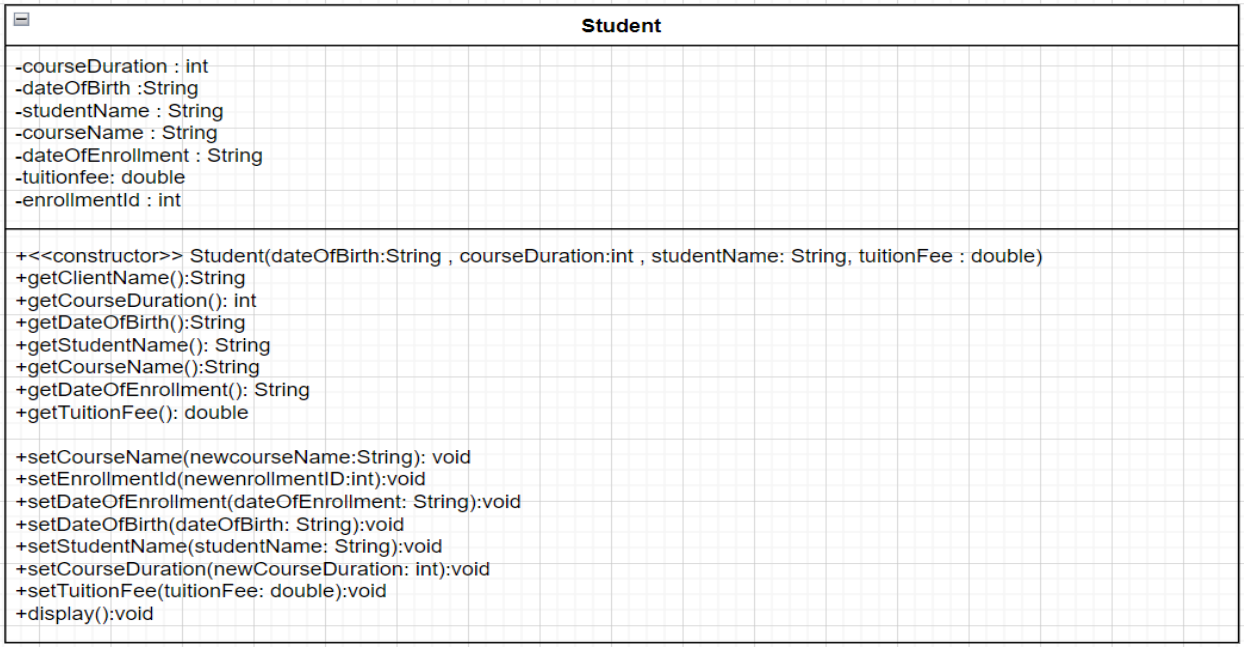

FIG4: CLASS DIAGRAM FOR STUDENT CLASS

2.2Class Diagram of Regular class

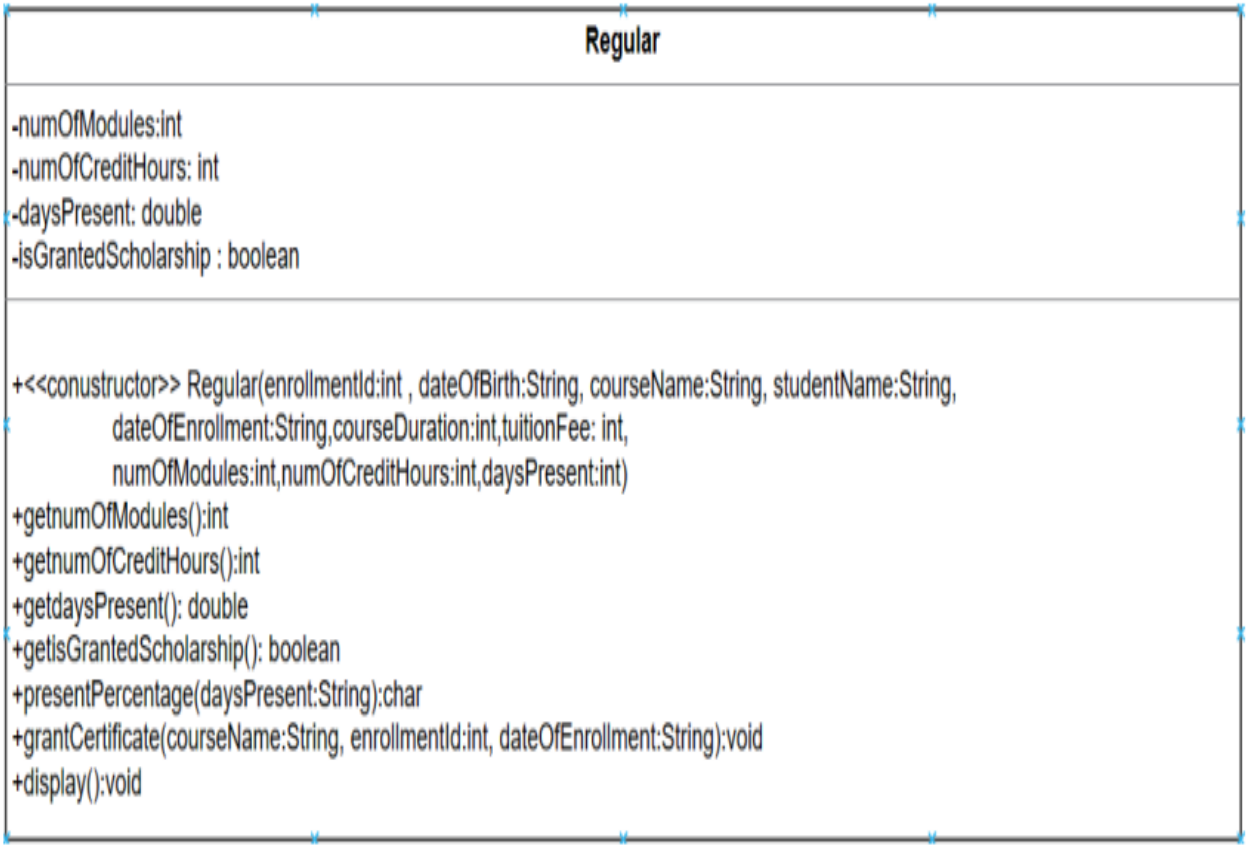

FIG5: CLASS DIAGRAM FOR REGULAR CLASS

2.3 Class Diagram of Dropout Class:

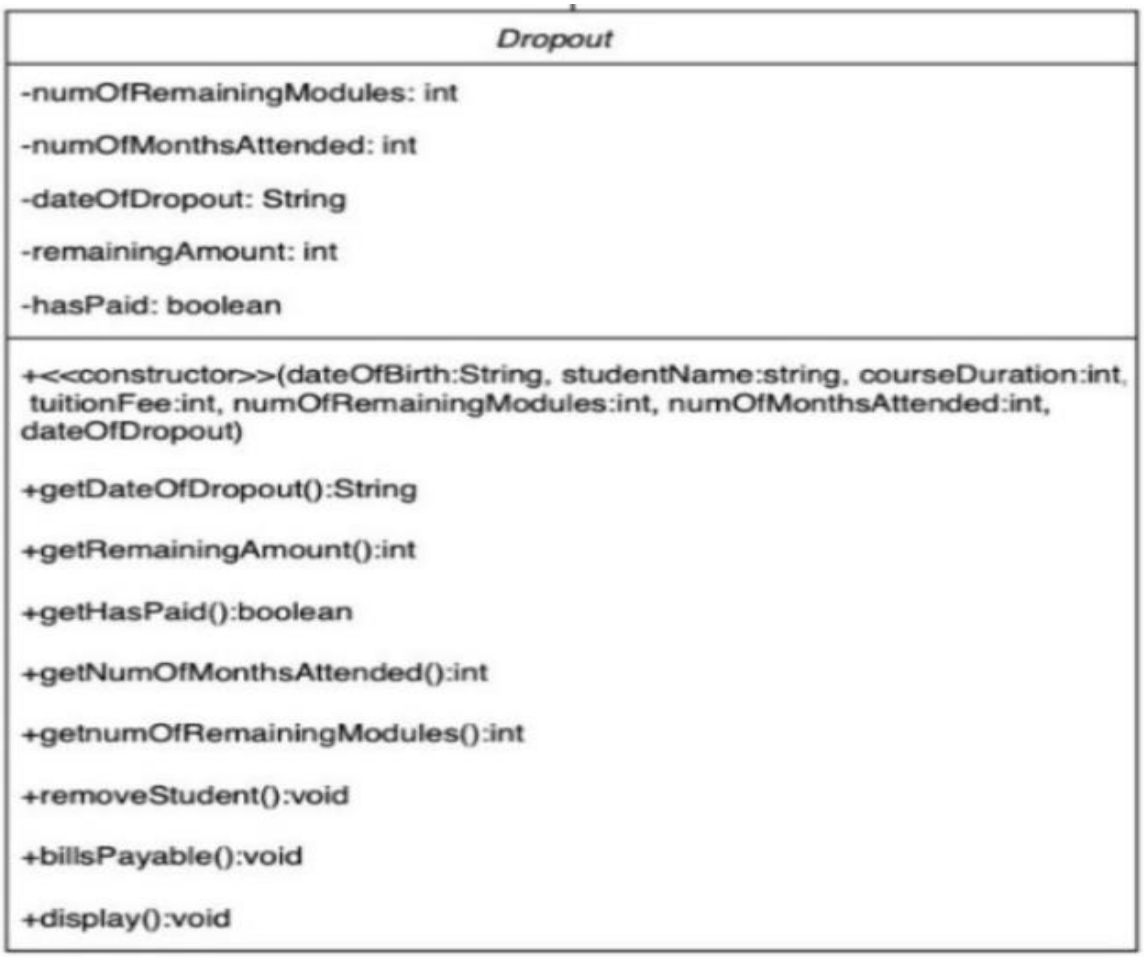

FIG 6: CLASS DIAGRAM OF DROPOUT CLASS

2.4 Inheritance Diagram

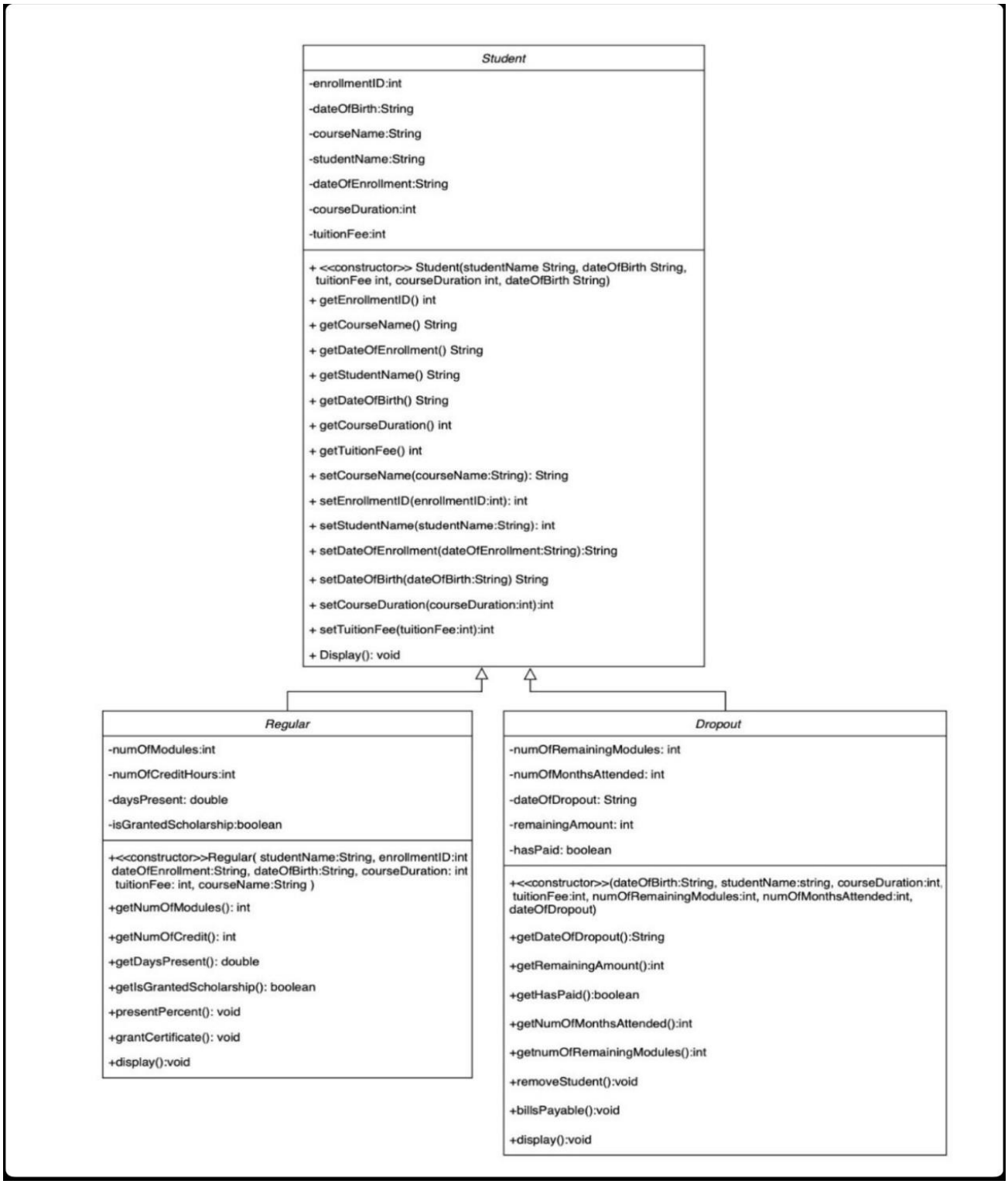

FIG7: INHERITANCE DIAGRAM

2.5 Class Diagram of StudentGUI class:

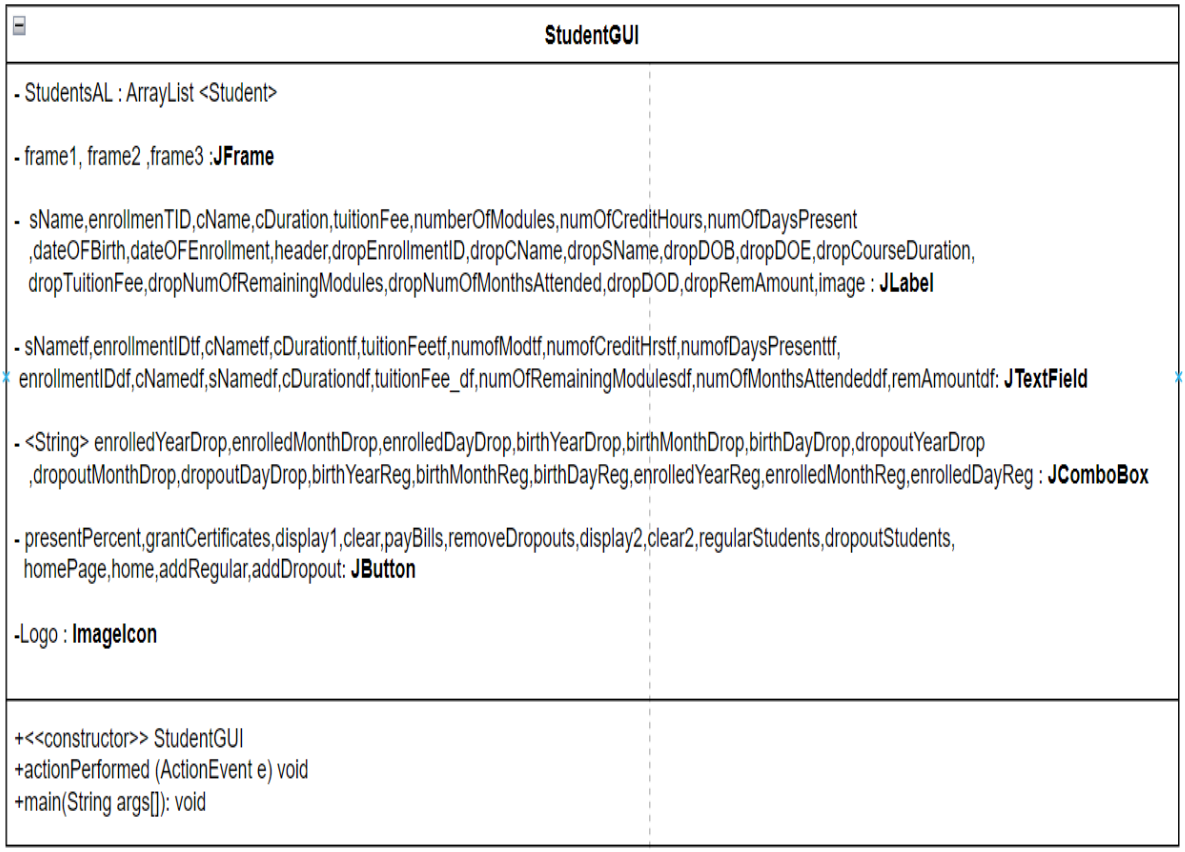

FIG8: CLASS DIAGRAM FOR STUDENTGUI CLASS

2.6 Final Class Diagram

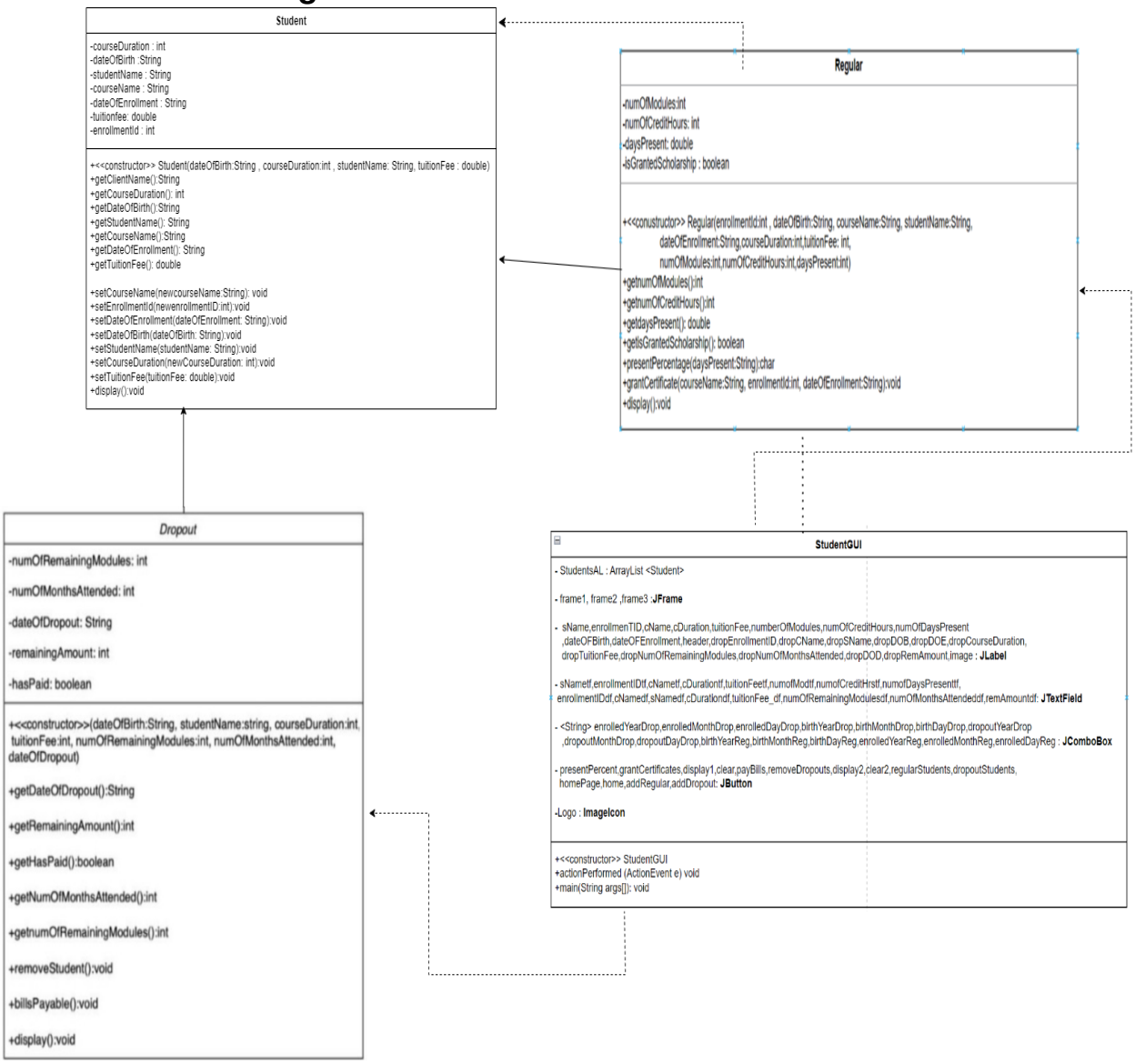

FIG9: FINAL CLASS DIAGRAM

3.Pseudocode

CREATE a class StudentGUI as public implements ActionListener

DO

DECLARE instance variables frame1,frame2 and frame3 of JFrame as private

DECLAREinstancevariable,sName,enrollmenTID,cName,cDuration,tuitionFee ,numberOfModules,numOfCreditHours,numOfDaysPresent,

dateOFBirth,dateOFEnrollment,header,dropEnrollmentID

,dropCName,dropSName,dropDOB,dropDOE,

dropCourseDuration,dropTuitionFee,

dropNumOfRemainingModules, dropNumOfMonthsAttended,

dropDOD,dropRemAmount,image of JLabel as private

DECLARE instance variables sNametf,enrollmentIDtf,cNametf,cDurationtf,

tuitionFeetf,numofModtf,numofCreditHrstf, sNamedf,

numofDaysPresenttf,enrollmentIDdf,cNamedf, cDurationdf

,tuitionFee_df,numOfRemainingModulesdf,numOfMonthsAttendeddf,

remAmountdf of JTextField as private

DECLARE instance variables enrolledYearDrop,enrolledMonthDrop,

enrolledDayDrop,birthYearDrop,birthMonthDrop,birthDayDrop,

dropoutYearDrop ,dropoutMonthDrop,dropoutDayDrop,

birthYearReg,birthMonthReg,birthDayReg,enrolledYearReg,

enrolledMonthReg,enrolledDayReg of JComboBox as private

DECLARE instance variables presentPercent,grantCertificates,display1,clear,

payBills,removeDropouts,display2,clear2,regularStudents,

dropoutStudents,homePage,home,addRegular,addDropout

of JButton as private

DECLARE instance variable logo of ImageIcon as private

CREATE a constructor as public for StudentGUI

DO

INITIALIZE the variables frame1,frame2 and frame3 to new JFrame()

INITIALIZE variables sName,enrollmenTID,cName,cDuration,tuitionFee,

numberOfModules,numOfCreditHours,numOfDaysPresent,

dateOFBirth,dateOFEnrollment,header,dropEnrollmentID

,dropCName,dropSName,dropDOB,dropDOE

,dropCourseDuration, dropTuitionFee,

dropNumOfRemainingModules,dropNumOfMonthsAttended,

dropDOD,dropRemAmount,image to new JLabel()

INITIALIZE variables sNametf,enrollmentIDtf,cNametf,cDurationtf,

tuitionFeetf,numofModtf,numofCreditHrstf, sNamedf,

numofDaysPresenttf,enrollmentIDdf,cNamedf, cDurationdf

,tuitionFee_df,numOfRemainingModulesdf,

numOfMonthsAttendeddf,remAmountdf to new JTextField()

INITIALIZE variables enrolledYearDrop,enrolledMonthDrop,

enrolledDayDrop,birthYearDrop,birthMonthDrop,birthDayDrop,
dropoutYearDrop ,dropoutMonthDrop,dropoutDayDrop,

birthYearReg,birthMonthReg,birthDayReg,enrolledYearReg,
enrolledMonthReg,enrolledDayReg of JComboBox as private

INITIALIZE variables presentPercent,grantCertificates,display1, clear,
payBills,removeDropouts,display2,clear2,regularStudents,
dropoutStudents,homePage,home,addRegular,addDropout
to new JButton().

INITIALIZE the variable logo of ImageIcon as private

SET seBounds to JButton components for frame1

SET setBounds to JLabel components for frame2

SET bounds to JTextField components for frame2

SET bounds to JComboBox components for frame2

SET bounds to JButton components for frame2

SET setBounds to JLabel components for frame3

SET bounds to JTextField components for frame3

SET bounds to JComboBox components for frame3

SET bounds to JButton components for frame3

ADD JLabel, JButton Components to frame1

ADD JLabel, JTextField,JComboBox,JButton Components to frame2

ADD JLabel, JTextField,JComboBox,JButton Components to frame3

ADD ActionListeners for all the JButton components

SET Default close operation to frame1

SET frame1 Layout to null

SET frame1 size

SET frame1 visible to true

CREATE an object of the Color class (for cyan color)

ADD the color(object) to all the buttons components of JButton

CREATE object of the Color class(for pink color)

ADD the color(object) to the background of the JFrame components

CREATE object of the Color class(for creamy color)

ADD the color(object) to the background of all JTextFields

CREATE object of the Font class

ADD the font to all the buttons

CREATE object of the Font class

ADD the font to all the JLabel components

END DO

CREATE a method actionPerformed() with ActionEvent e as parameter

DO

IF e.getSource() is equal to regularStudents

DO

SET frame2 Layout to null

SET frame2 size

SET frame2 visible to true

SET frame1 to disposal

END DO

ELSE IF e.getSource() is equal to dropoutStudents

DO

SET frame3 Layout to null

SET frame3 size

SET frame3 visible to true

SET frame1 to disposal

END DO

ELSE IF e.getSource() is equal to clear

DO

SET sNametf to empty string

SET cNametf to empty string

SET enrollmentIDtf to empty string

SET cDurartiontf to empty string

SET tuitionFeetf to empty string

SET numOfModtf to empty string

SET numOfCreditHrstf to empty string

SET numOfDaysPresenttf to empty string

SHOW information message All fields set to empty

END DO

ELSE IF e.getSource() is equal to clear2

DO

SET enrollmentIDdf to empty string

SET cNamedf to empty string

SET sNamedf to empty string

SET cDurationdf to empty string

SET tuitionFee_df to empty string

SET numOfRemainaingModulesdf to empty string

SET numOfMonthsAttendeddf to empty string

SET remAmountdf to empty string

SHOW information message All fields set to empty

END DO

ELSE IF e.getSource() is equal to homepage

DO

SET frame1 Layout to null

SET frame1 size

SET frame1 visible to true

SET frame2 to disposal

SET frame3 to disposal

END DO

ELSE IF e.getSource() is equal to home

DO

SET frame1 Layout to null

SET frame1 size

SET frame1 visible to true

SET frame2 to disposal

SET frame3 to disposal

END DO

ELSE IF e.getSource() is equal to grantCertificates

DO

IF enrollmentIDtf text field is empty OR cNametf textfield is empty

DO

SHOW warning message The fields Enrollment ID,Date of Enrollment and Course Name must all be filled to use this button

END DO

ELSE

DO

TRY

DO

SET Integer enrollmentIDtf textfield value to int enrollmentID

SET cNametf textfield value to String courseName

SET enrolledYearReg combobox value to String

REnrolledYear

SET enrolledMonthReg combobox value to String REnrolledMonth

SET enrolledDayReg combobox value to String REnrolledDay

SET REnrolledYear concade REnrolledMonth concade REnrolledDay to String DateOfEnrollment

INITIALIZE variable certificate to true

FOR object j in the StudentsAL arraylist

DO

IF j is an instance of Regular class

DO

Typecast the Regular class object certification

IF enrollmentID is equal to the object of Regular

Class getEnrollmentID()

DO

ASSIGN value true to certificate

CALL method grantCertificate with courseName,enrollmentID and dateOfEnrollment as parameters

SHOW information message Certificate has been granted to the student

END DO

END DO

END DO

IF Boolean Certificate is equal to false

DO

SHOW error message The enrollment ID doesnot match with the ID of a regular student

END DO

END DO

CATCH numberFormatException s

DO

SHOW error message Invalid Input

END DO

END DO

END DO

ELSE IF e.getSource() is equal to presentPercent

DO

IF enrollmentID OR numOfDaysPresent textfields are empty

DO

SHOW warning messageThe fields Enrollment ID and Number of Days present must be filled to use this button

END DO

ELSE

DO

TRY

DO

SET enrollmentIDtf textfield value to int enrollmentID

SET daysPresenttf textField value to double daysPresent

ASSIGN variable pass to Boolean value true

FOR object c in the StudentAL arraylist

DO

IF object c is an instance of Regular class

DO

Object ppCaculation of Regular class is typecasted

IF enrollmentID is equal to the getEnrollmentID object of Regular Class

DO

ASSIGN Boolean pass is equal to true

CALL method presentPercentage with daysPresent as parameter

SHOW information message Present percentage has been calculated

END DO

END DO

END DO

IF Boolean pass is equal to false

DO

SHOW error message The enrollment ID doesnot match with the Id of a regular student

END DO

END DO

CATCH numberFormatException

DO

SHOW error message invalid input

END DO

END DO

END DO

ELSE IF e.getSource is equal to removeDropouts

DO

IF enrollmentIDdf textfield is empty

DO

SHOW warning message The field Enrollment ID must be filled to use this button

END DO

ELSE

DO

TRY

DO

SET enrollmentID df textfield to int enrollmentID

ASSIGN Boolean drop to true

FOR object a in StudentsAL arraylist

DO

IF a is an instance of Dropout class

DO

Typecasting of the remove object of Dropout

IF enrollmentID is equal to remove.getEnrollmentID()

DO

SET Boolean Drop is equal to true

CALL the removeStudent() method from Dropout classs

SHOW information message The student has been removed

END DO

ELSE

DO

SET Boolean Drop is equal to false

END DO

END DO

END DO

IF Boolean Drop is equal to false

DO

SHOW error message The enrollment ID is not valid for dropout student

END DO

END DO

CATCH numberFormatException n

DO

SHOW error message Invalid input

END DO

END DO

END DO

ELSE IF e.getSource() is equal to addRegular

DO

IF the Regular class textfields are empty

DO

SHOW warning message All fields must be entered to use this button

END DO

ELSE

DO

TRY

DO

SET Integer enrollmentIDtf textfield value to int enrollmentID

SET cNametf textfield value to String courseName

SET sNametf textfield value to String studentName

SET cDuraiontf textfiled value to int courseDuration

SET tuitionFeetf textfiled value to int tuitionFee

SET numOfModtf textfiled value to int numOfModules

SET numOfCreditHrstf textField value to int numOfCreditHours

SET birthYearReg combobox value to String

RBirthYear

SET birthMonthReg combobox value to String

RBirthMonth

SET birthDayReg combobox value to String RBirthDay

SET RBirthYear concade RBirthMonth concade

RBirthDay to String DateOfEnrollment

SET enrolledYearReg combobox value to String

REnrolledYear

SET enrolledMonthReg combobox value to String

REnrolledMonth

SET enrolledDayReg combobox value to String REnrolledDay

SET REnrolledYear concade REnrolledMonth concade

REnrolledDay to String DateOfEnrollment

ASSIGN boolean regulars to true

IF StudentsAL arraylist is empty

DO

CREATE a new object reg of the Regular class with

enrollmentID,dateOfBirth,courseName,
studentName,
dateOfEnrollment,courseDuration,tuitionFee,

numOfModules, numOfCreditHours, daysPresent as parameters

ADD the object reg to the StudentsAL arraylist

SHOW information message Student Added

END DO

ELSE

DO

FOR object x in arraylist StudentsAL

DO

IF x is an instance of Regular class

DO

Typecast the always object of Regular class

IF enrollmentID is equal to always.getEnrollmentID()

DO

SET Boolean regulars to false

END DO

END DO

END DO

IF Boolean regulars is equal to true

DO

CREATE a new object regs of the Regular class with enrollmentID,dateOfBirth,courseName,

studentName,dateOfEnrollment,courseDuration,tuitionFee,numOfModules,numOfCreditHours, daysPresent as parameters

ADD the object regs to the StudentsAL arraylist

SHOW information message Student added

END DO

ELSE

DO

SHOW error message Student already exists

END DO

END DO

END DO

CATCH numberFormatException n

DO

SHOW errpr message Incorrect input

END DO

END DO

END DO

ELSE IF e.getSource() is equal to addDropout

DO

IF the dropout class textfields are empty

DO

SHOW warning message all fields must be entered to use this button

END DO

TRY

DO

SET Integer enrollmentIDdf textfield value to int enrollmentID

SET cNamedf textfield value to String courseName

SET sNamedf textfield value to String studentName

SET cDuraiondf textfiled value to int courseDuration

SET tuitionFee_df textfiled value to int tuitionFee

SET numOfRemainingModulesdf textfiled value to int numOfRemainingModules

SET numOfMonthsAttendeddf textField value to int numOfMonthsAttended

SET enrolledYearDrop combobox value to String DEnrolledYear

SET enrolledMonthDrop combobox value to String DEnrolledMonth

SET enrolledDayDrop combobox value to String DEnrolledDay

SET DEnrolledYear concade DEnrolledMonth concade

DEnrolledDay to String dateOfEnrollment

SET birthYearDrop combobox value to String DBirthYear

SET birthMonthDrop combobox value to String DBirthMonth

SET birthDayDrop combobox value to String DBirthDay

SET DBirthYear concade DBirthMonth concade

DBirthDay to String dateOfBirth

SET Boolean drops is equal to true

IF StudentsAL arraylist is Empty

DO

IF courseDuration is not equal to numOfMonthsAttended

DO

SHOW message Course duration must be equal to the number of months attended in order to pay the bills later

END DO

ELSE

DO

CREATE new dropout object drop with its parameters enrollmentID,courseName,dateOfEnrollment,dateOfBirth,studentName,courseDuration,tuitionFee,numOfRemainingModules,numOfMonthsAttended,dateOfDropout

ADD the drop object to the StudentsAL arraylist

SHOW information message student added.

END DO

END DO

ELSE

DO

FOR object b in StudentsAL arraylist

DO

IF b is an instance of Dropout class

DO

Typecast never object of Dropout class

IF enrollmentID is equal to never.getEnrollmentID()

DO

SET the Boolean drops to false

END DO

END DO

END DO

IF boolean drops is equal to true

DO

CREATE new dropout object dropz with its parameters enrollmentID,courseName,dateOfEnrollment,dateOfBirth,studentName,courseDuration,tuitionFee,numOfRemainingModules,numOfMonthsAttended,dateOfDropout

SHOW information message student added

END DO

ELSE

DO

SHOW message Student already exists

END DO

END DO

END DO

CATCH numberFormatException v

DO

SHOW error message Incorrect input

END DO

END DO

ELSE IF e.getSource() is equal to display1

DO

IF any Regular class textfields are empty

DO

SHOW warning message all fields must be entered

END DO

ELSE

DO

IF StudentsAL arraylist is empty

DO

SHOW message no items to display

END DO

ELSE

DO

FOR object z in StudentsAL arraylist

DO

IF z is an instance of Regular class

DO

Typecast the Rdisplay object of Regular class

CALL the display() method

Show message The message is being displayed in the terminal/ command prompt

END DO

ELSE

DO

SHOW warning message The student belongs to dropout class

END DO

END DO

END DO

END DO

END DO

ELSE IF e.getSource() is equal to display2

DO

IF any Dropout class textfields are empty

DO

SHOW warning message all fields must be entered

END DO

ELSE

DO

IF StudentsAL arraylist is empty

DO

SHOW message no items to display

END DO

ELSE

DO

FOR object d in StudentsAL arraylist

DO

IF d is an instance of Dropout class

DO

Typecast the Odisplay object of Dropout class

CALL the display() method of Dropout class

Show message The message is being displayed in the terminal/ command prompt

END DO

ELSE

DO

SHOW warning message The student belongs to regular class

END DO

END DO

END DO

END DO

END DO

END DO

CREATE a main method main() with return type void as public

DO

INITIALIZE StudentGUI

END DO

END DO

4. Method description

Method	Description
Action performed()	The common method actionPerformed() is part of the ActionListener interface. It is automatically called whenever a button is clicked or other action even occurs.When a user interacts with the component, the corresponding action listener will automatically run the actionPerformed() method. All the buttons use this method to perform their respective actions

Table 1.Method description

Various methods have been used in different buttons of the StudentGUI class which are described below:

Method	Description
regularStudents	When this button is clicked, the program opens frame 2 in order to open the GUI for Regular Students.
dropoutStudents	When this button is clicked, the program opens frame 3 in order to open the GUI for Dropout Students.
clear	This button clears all the text fields of Regular students interface.
clear2	This button clears all the text fields of Dropout students interface.
homePage	When this button is clicked, the program opens frame 1/home page.
home	When this button is clicked, the program opens frame 1/home page.
grantCetificates	Firstly,the button checks for empty text fields.If the fields are empty, it shows a warning message.If the required fields are not empty, it enters the else block. Then, it attempts to extract values from textfields to prepare variables.A boolean variable Certificate is initialized as true.Then,it iterates through the StudentAL ArrayList.If a student is a Regular student and the provided enrollment ID matches, it calls the grantCertificate() method from the Regular class and displays a success message. If no match is found, it sets Certificate to false. After the loop, it checks if Certificate is false and displays an error message. If parsing fails, it displays an invalid input error message.

presentPercent	It checks if both the enrollmentIDtf and numofDaysPresenttf text fields are empty. If so, it displays a warning message.If not, it parses string values into integers,It iterates through a list of students (StudentsAL).If a student is a Regular student and the enrollment ID matches the textfield data, it calculates the present percentage using the presentPercentage() method from Regular class.If no match is found, it an error message is displayed.If parsing fails, it displays an invalid input error.
removeDropouts	If StudentsAL ArrayList is empty, it shows an info message.Otherwise, it attempts to parse the enrollment ID into an integer.Then,it iterates through StudentsAL looking for Dropout objects.If a matching enrollment ID is found, it removes the student from the ArrayList and diplays a success message.If no match is found, it displays an error message.If parsing fails, it shows an invalid input error message.
addRegular	It checks if all the textfields are empty using. If any are empty, it shows a warning message.If all required fields are filled, it attempts to parse various input values into appropriate data types.It initializes a boolean variable regulars as true.If StudentsAL is empty, it creates a new Regular object and adds it to the list. A success message is

	displayed.If the list isn't empty, it checks if the enrollment ID already exists among regular students. If it does, it sets regulars to false.If regulars is still true, it creates a new Regular instance and adds it to the list. A success message is displayed.If regulars was set to false, it displays an error message indicating the student already exists.If parsing fails, it displays an invalid input error message.
addDropout	It checks if all the textfields are empty u. If any are empty, it shows a warning message.If all required fields are filled, it attempts to parse various input values into appropriate data types.It initializes a boolean variable drops as true.If StudentsAL is empty, it creates a new Dropout object and adds it to the list if course duration matches months attended. Otherwise, it shows an info message.If the list isn't empty, it checks if the enrollment ID already exists among dropout students. If it does, it sets drops to false.If drops is still true, it creates a new Dropout instance and adds it to the list. A success message is displayed.If drops was set to false, it displays an error message indicating the student already exists.If parsing fails, it displays an invalid input error message.
display1	It is used to display all the information listed out in the Regular class by calling the display method from regular class

display2	It is used to display all the information listed out in the Regular class by calling the display method from regular class

Table 2: Method description of various methods used in the buttons

5.Testing

5.1 Test1

Test No.	1
Objective:	To Test that the program can be compiled and run using the command prompt
Action	Now entering the following command in the command prompt: **javac StudentGUI.java** **java StudentGUI**
Expected result	The program should be compiled and run after using the code in cmd prompt
Result obtained:	The program was compiled and run.
Conclusion	The program can be run through the command prompt

Table 3 : Testing whether the program runs with the command prompt

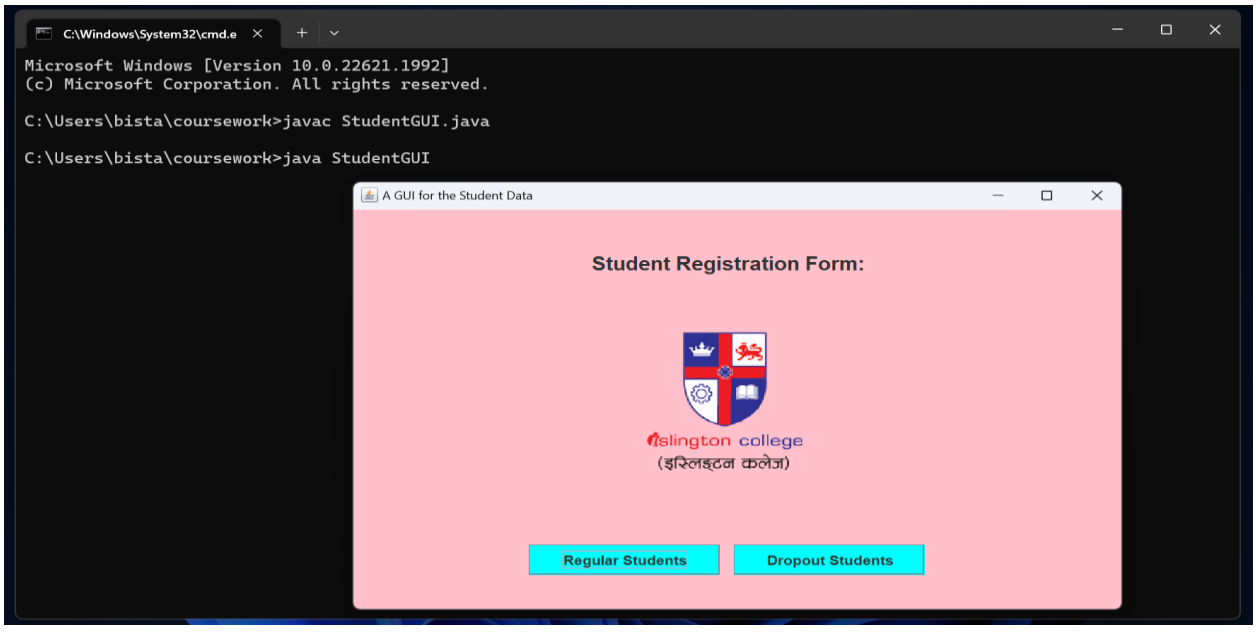

FIG 10:TEST 1 OPENING THE PROGRAM IN COMMAND PROMPT

5.2 Test 2

a.Add a Regular Student

Test No.	2
Objective	To add a regular student
Action	The datas enterd in the Regular class text fields are: Enrollment ID :01 Course Name : Computing Date Of Enrollment : 2023/02/31 Number Of Days Present : 170 Student Name: Sahil Bista Course Duration: 6 Number Of Modules : 5 Date Of Birth : 2004/06/10 Number Of Credit Hours: 30 Tuition Fee: 10000 Then , the Add Student button is clicked
Expected Result	Dialog Box should appear with the message “Student added”
Obtained result	Dialog Box appeared with the message “Student added”
Conclusion	Successful

Table 4:testing add regular student

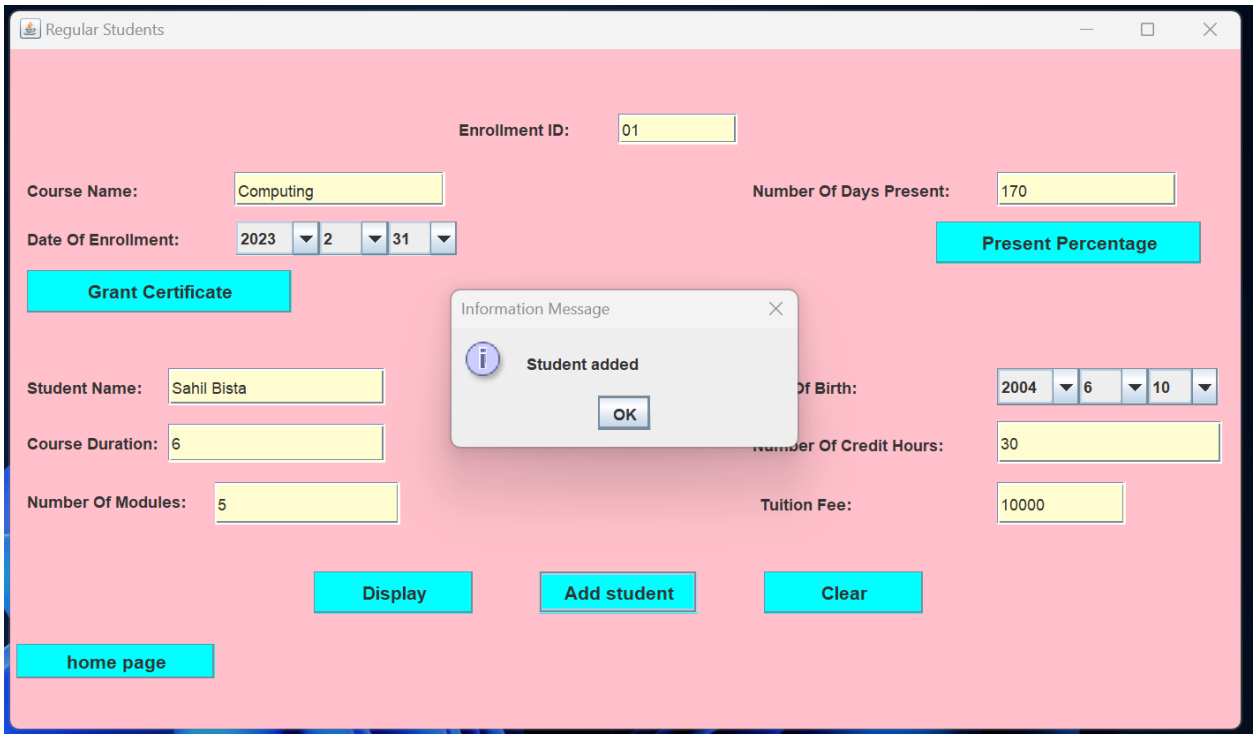

FIG 11 : TESTING ADD REGULAR STUDENT

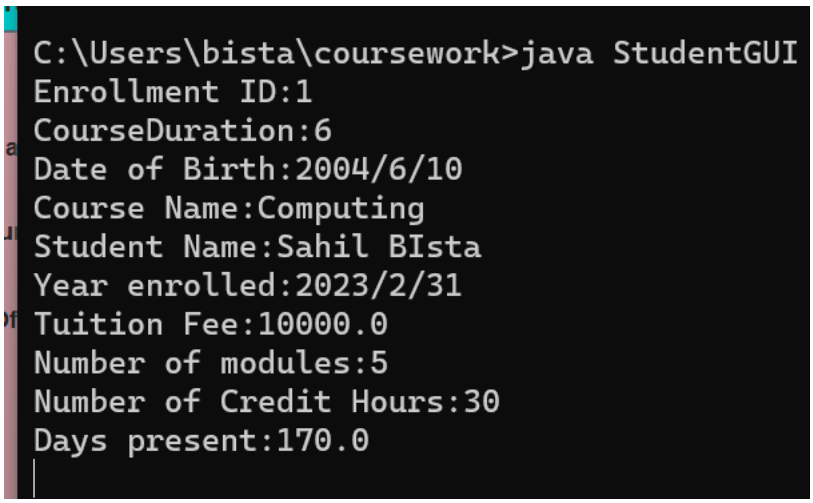

FIG 12: DISPLAY OF THE REGULAR STUDENT INFORMATION IN COMMAND PROMPT

b.Add a Dropout Student

Test No.	3
Objective	To add a dropout student
Action	The datas enterd in the Regular class text fields are: Enrollment ID :07 Course Name : Footballing Date Of Enrollment : 2023/07/07 TuitionFee : 10000 Remaining Amount: 0 Student Name: Cristiano Course Duration: 6 Number Of Remaining Modules : 2 Date Of Birth : 2000/02/03 Number Of Months Attended: 6 Date Of Dropout: 2023/12/07 Then , the Add Student button is clicked
Expected Result	Dialog Box should appear with the message “Student added”
Obtained result	Dialog Box appeared with the message “Student added”
Conclusion	Successful

Table 5: Testing of Add student button for Dropout

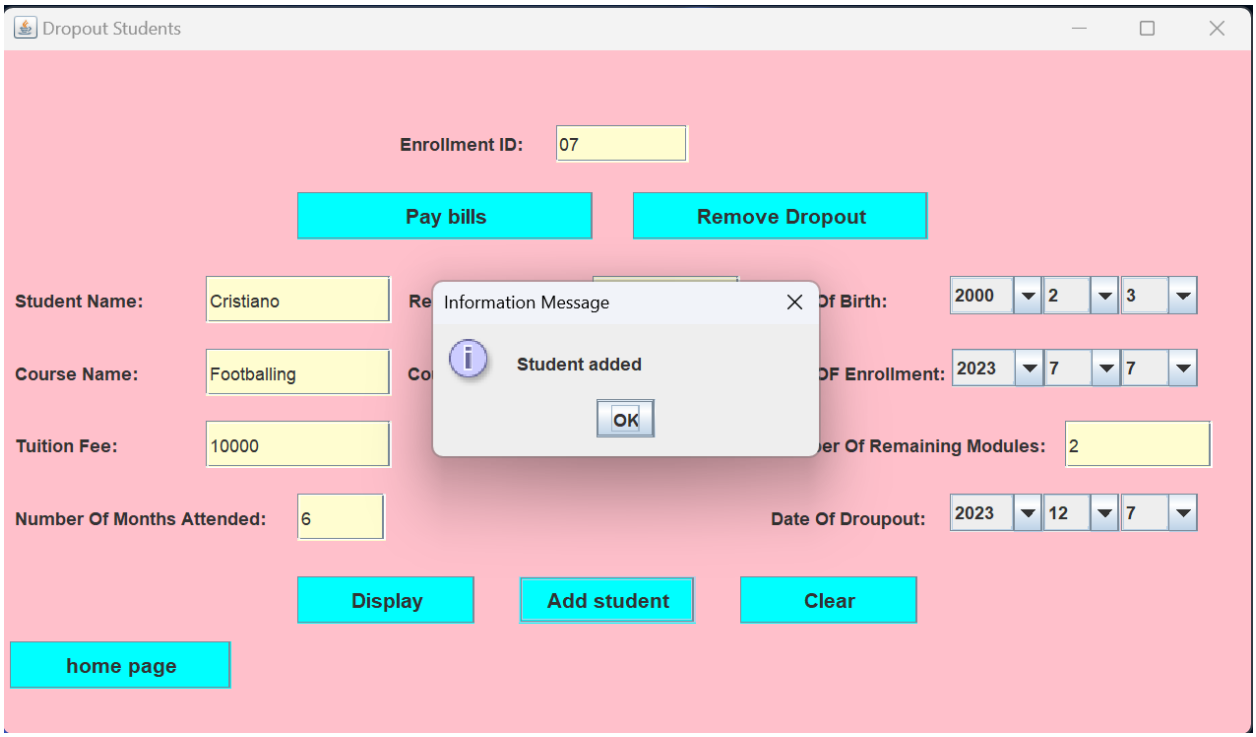

FIG 13: TESTING ADD DROPOUT STUDENT

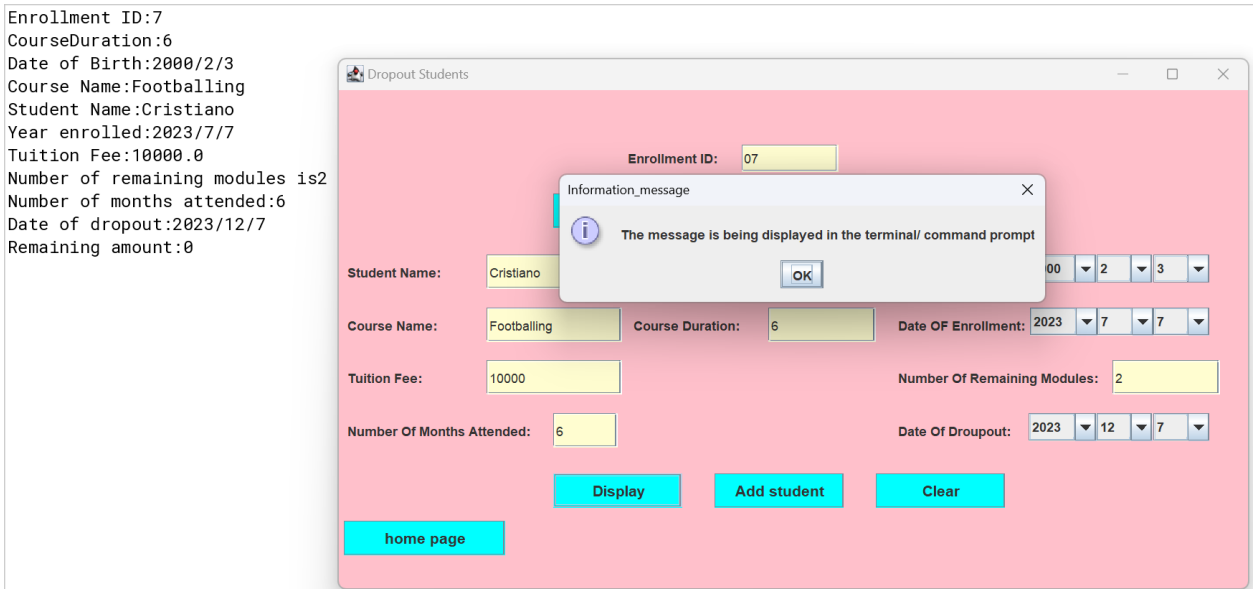

FIG 14: DISPLAY OF THE DROPOUT STUDENT INFO

c.Calculate Present Percentage of Regular Student

Test No.	3
Objective	To calculate the Present Percentage Of Regular Student
Action	After adding the Regular Student using Add Student Button , the present Percentage button is clicked
Expected result	The dialog box should pop out with the information “Present percentage has been calculated” and the present Percentage should be displayed in the terminal.
Obtained Result	The dialog box popped out with the information “Present percentage has been calculated” and the present Percentage was calculated and displayed in the terminal.
Conclusion	Successful

Table 6: Testing of the present Percentage calculation button

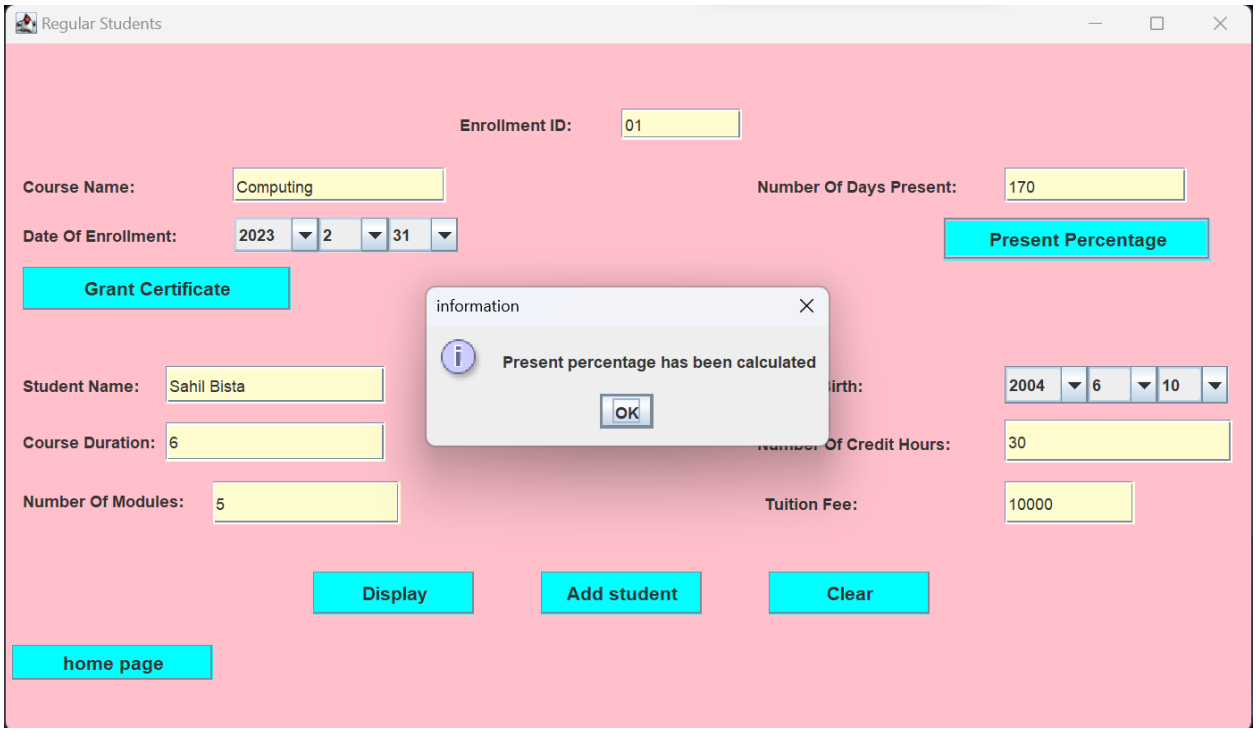

FIG 15: TESTING OF THE PRESENT PERCENTAGE BUTTON

```
Your total present days in percentage is 94.44444444444444
```

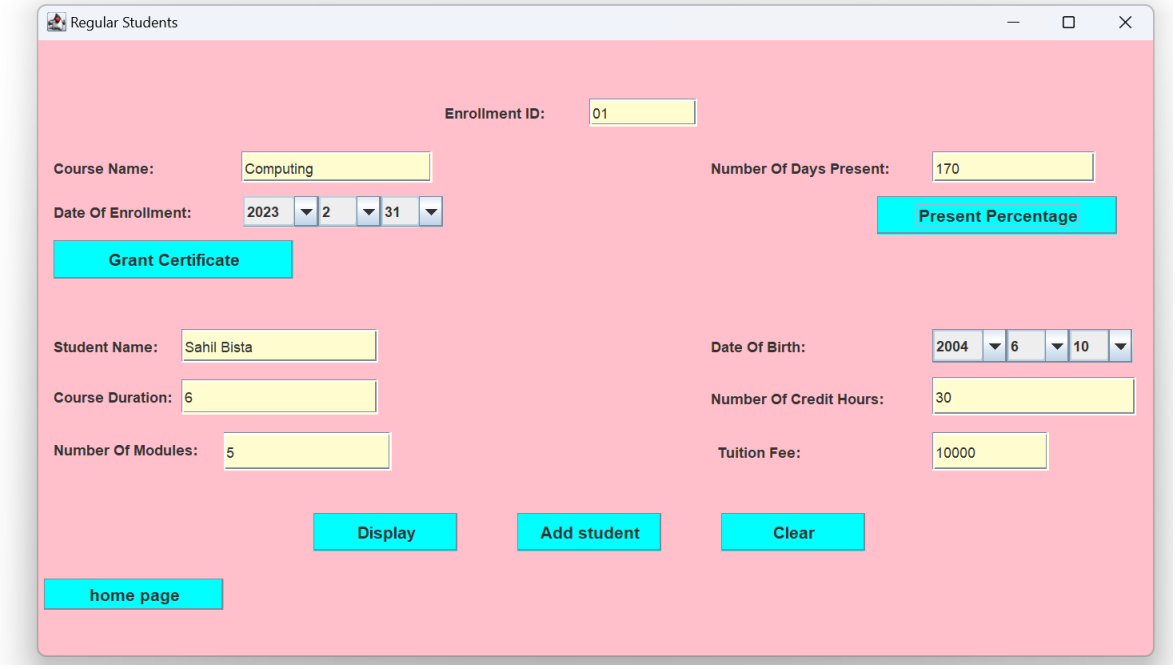

FIG 16 : THE CALCULATION OF PRESENT PERCENTAGE IN THE BLUEJ TERMINAL

d.Grant the certificate to regular Students

Test No.	4
Objective	To grant the certificate to Regular Students
Action	The object of Regular class was created and added to the arraylist and now the Grant Certificate button is to be clicked
Expected Result	The dialog box should pop out with the information “Certificate has been granted to the student” and the certificate granting message be displayed in the terminal.
Obtained Result	The dialog box should pop out with the information “Certificate has been granted to the student” and the certificate granting message should be displayed in the terminal.
Conclusion	Successful

Table 7:Testing of the Grant Certificate button

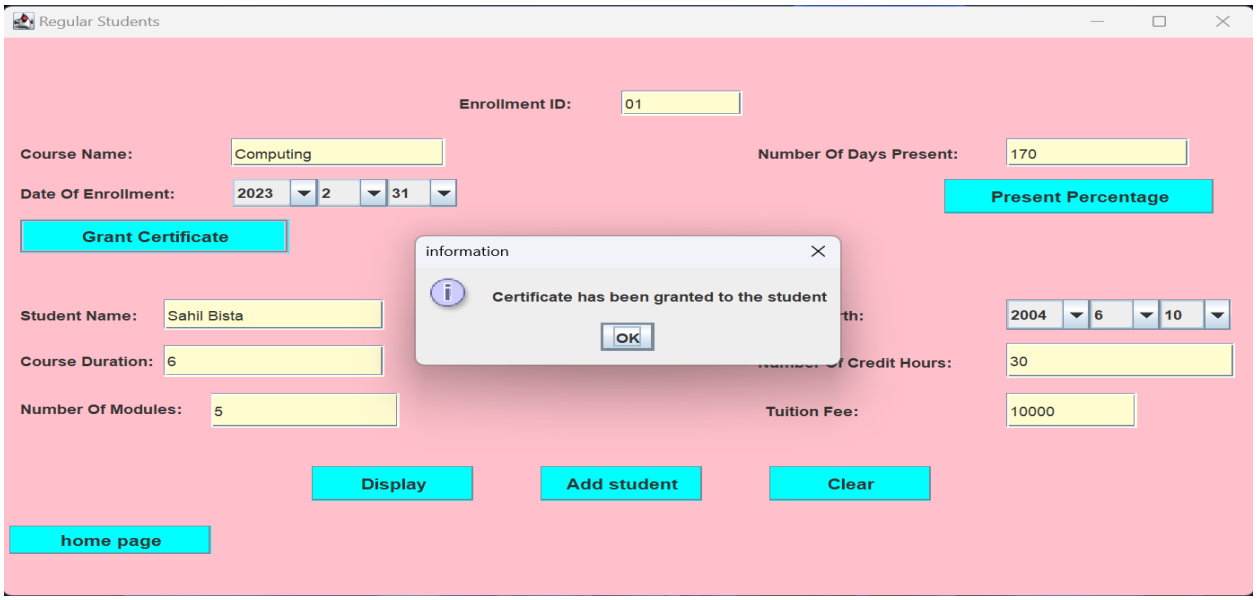

FIG 17:TESTING OF THE GRANT CERTIFICATE BUTTON FOR POP UP MESSAGE

```
Your total present days in percentage is 94.44444444444444
Heartfelt congratulations on your graduation: Computing, enrollmentID: 1, dateOfEnrollment: 2023/2/31
The scholarship has been granted
```

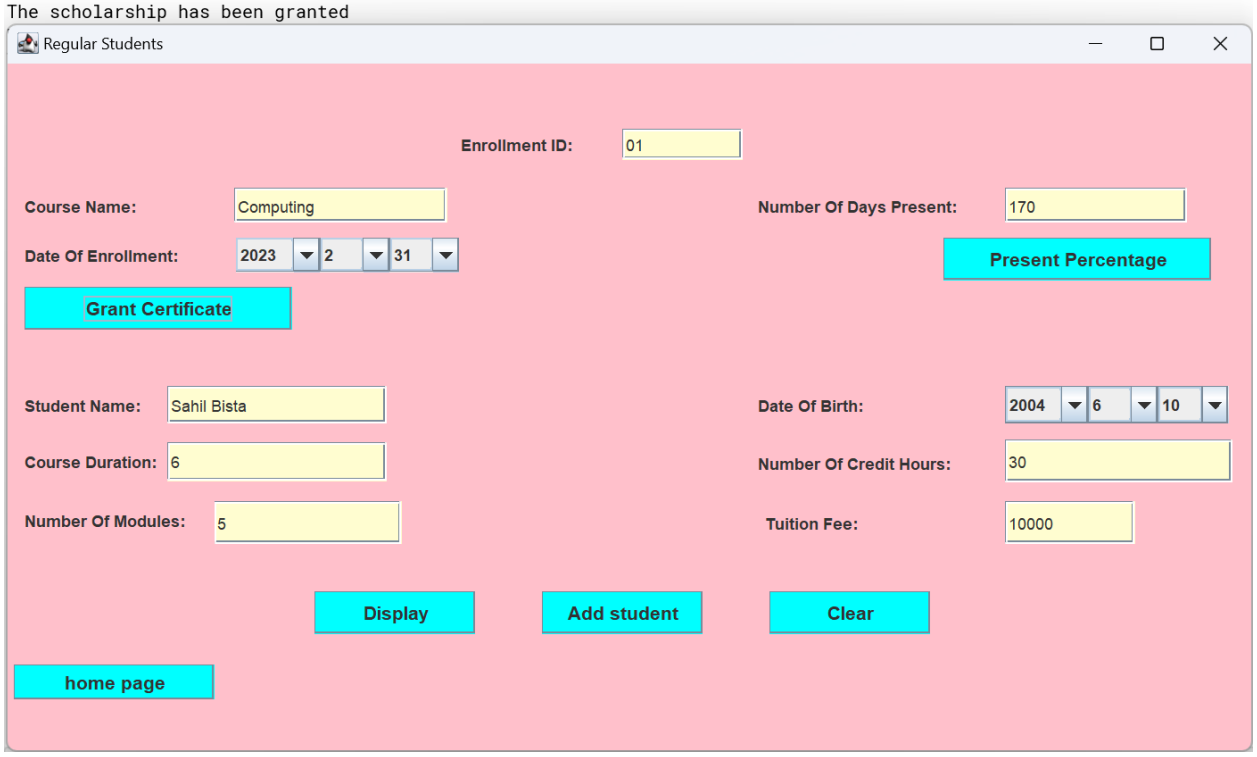

FIG 18: TESTING OF THE GRANT CERTIFICATE BUTTON WITH ITS DISPLAY IN THE BLUEJ TERMINAL

e. Pay the bills of Dropout Student

Test no.	5
Objective	To pay the bills of Dropout Student
Action	To press the Pay Bills button after adding the student of Dropout class by filling all the empty textfields
Expected result	The bills or the remaining Amount of the Dropout student should be cleared out
Obtained result	The bills or the remaining Amount of the Dropout student was cleared out
Conclusion	Successul

Table 8: Test of the Pay bills button of Dropout Students

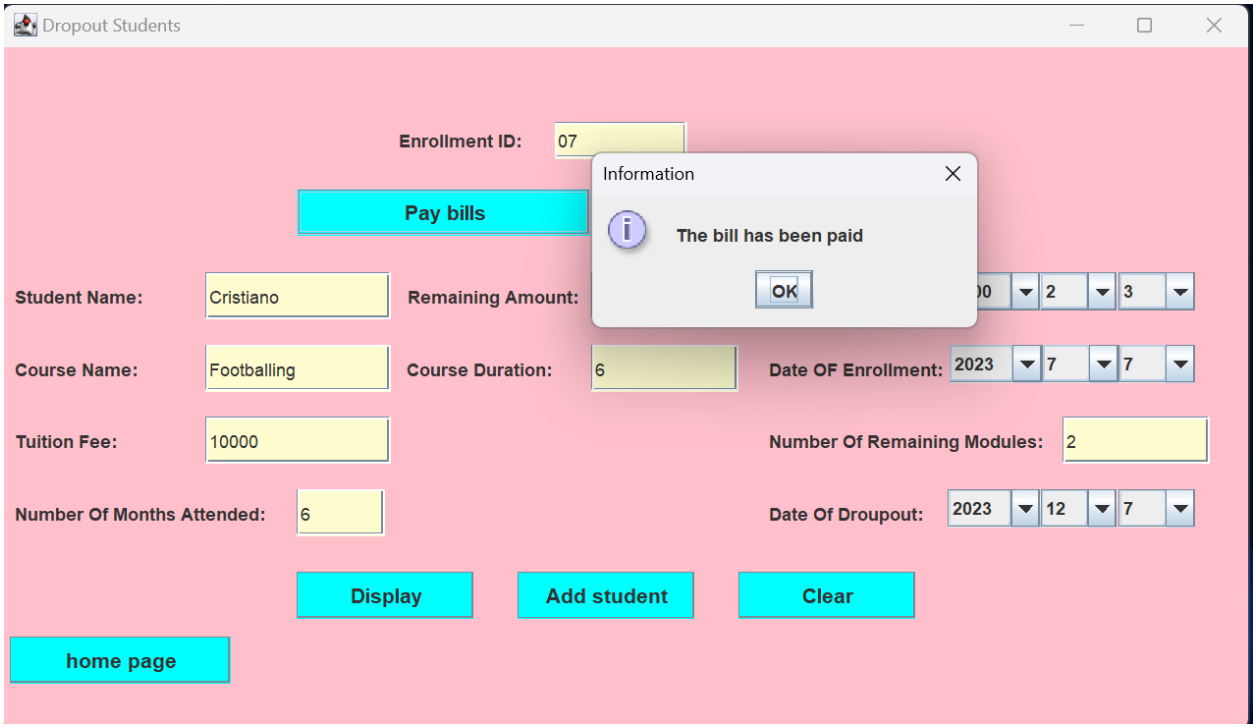

FIG 19: THE POP OUT MESSAGE UPON PRESSING THE PAY BILLS BUTTON

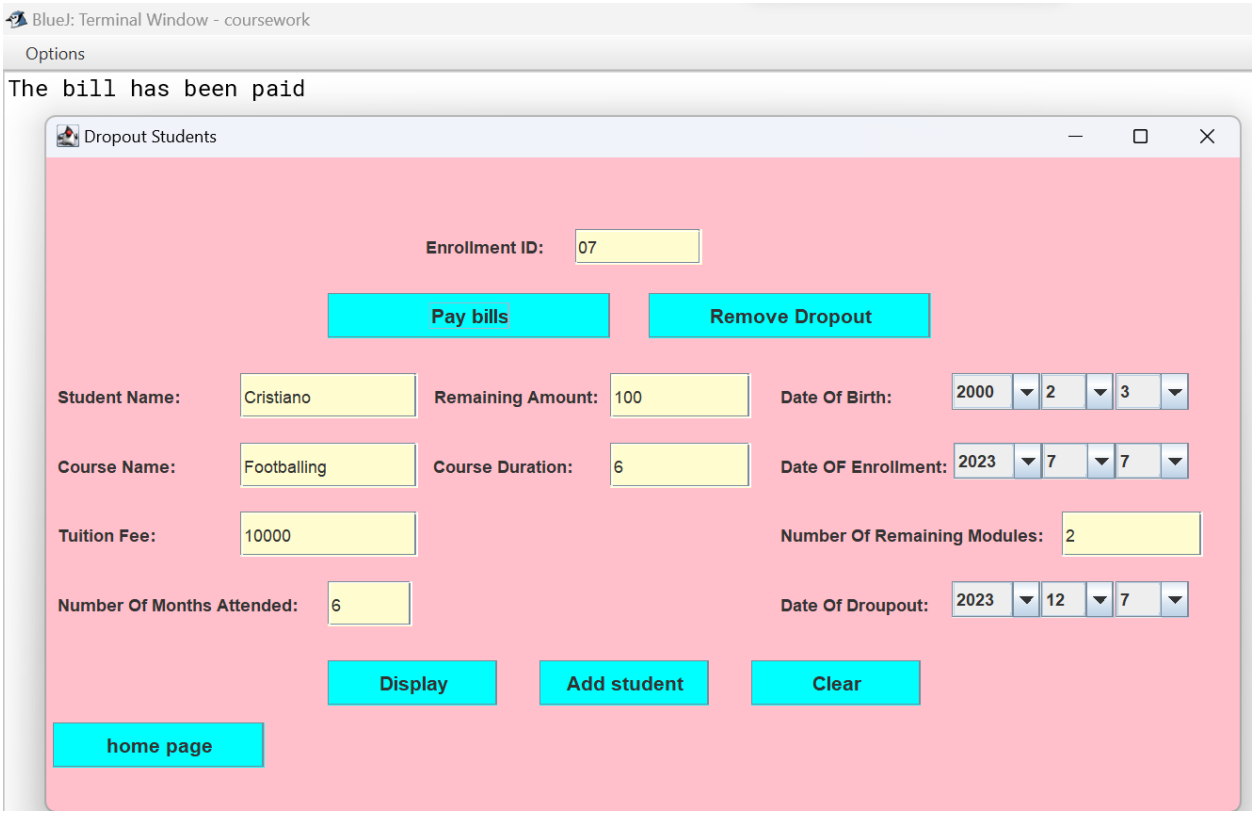

FIG 20:THE TERMINAL DISPLAY AFTER PRESSING THE PAY BILLS BUTTON.

f.Remove Dropout Student

Test No.	6
Objective	To remove the Dropout student
Action	To press the Remove Dropout button after adding the student and paying the bills of Dropout class by filling all the empty textfields and pressing the pay bills button
Expected Result	
Obtained Result	
Conclusion	Successful

Table 9:Test the removeDropout Button

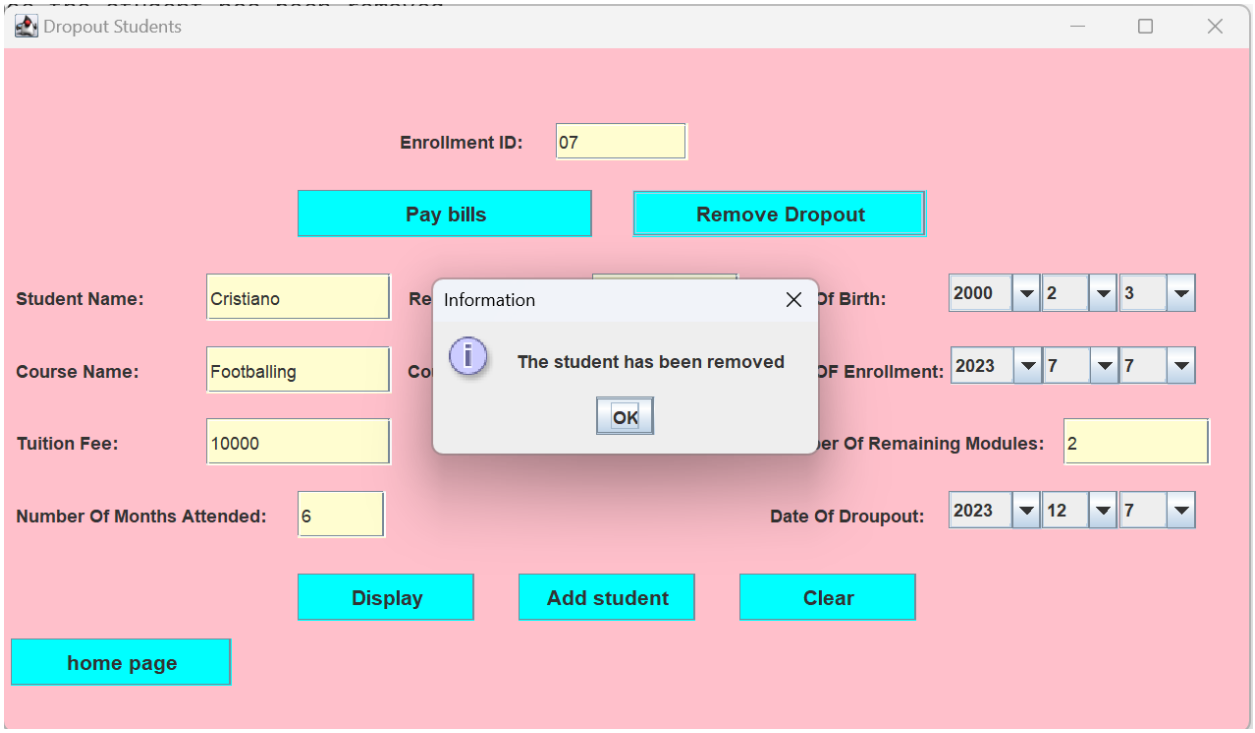

FIG 21 : THE POP UP MESSAGE DISPLAYED AFTER PRESSING THE REMOVE DROPOUT BUTTON.

```
Options
The bill has been paid
Hence,the student has been removed
```

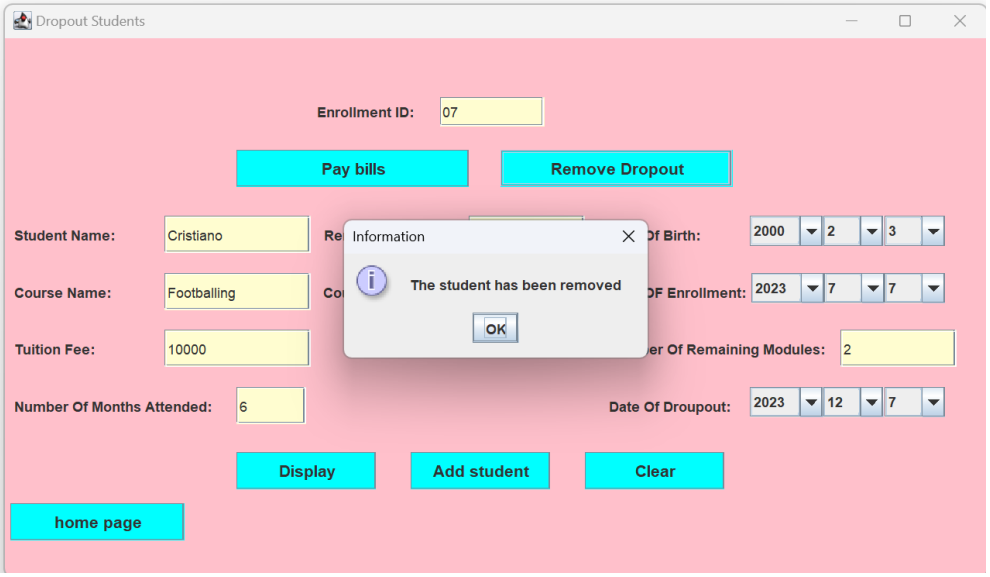

FIG 22: USE OF REMOVE STUDENT BUTTON AND THE DISPLAYED MESSAGE ON THE BLUEJ TERMINAL

5.3Test 3:

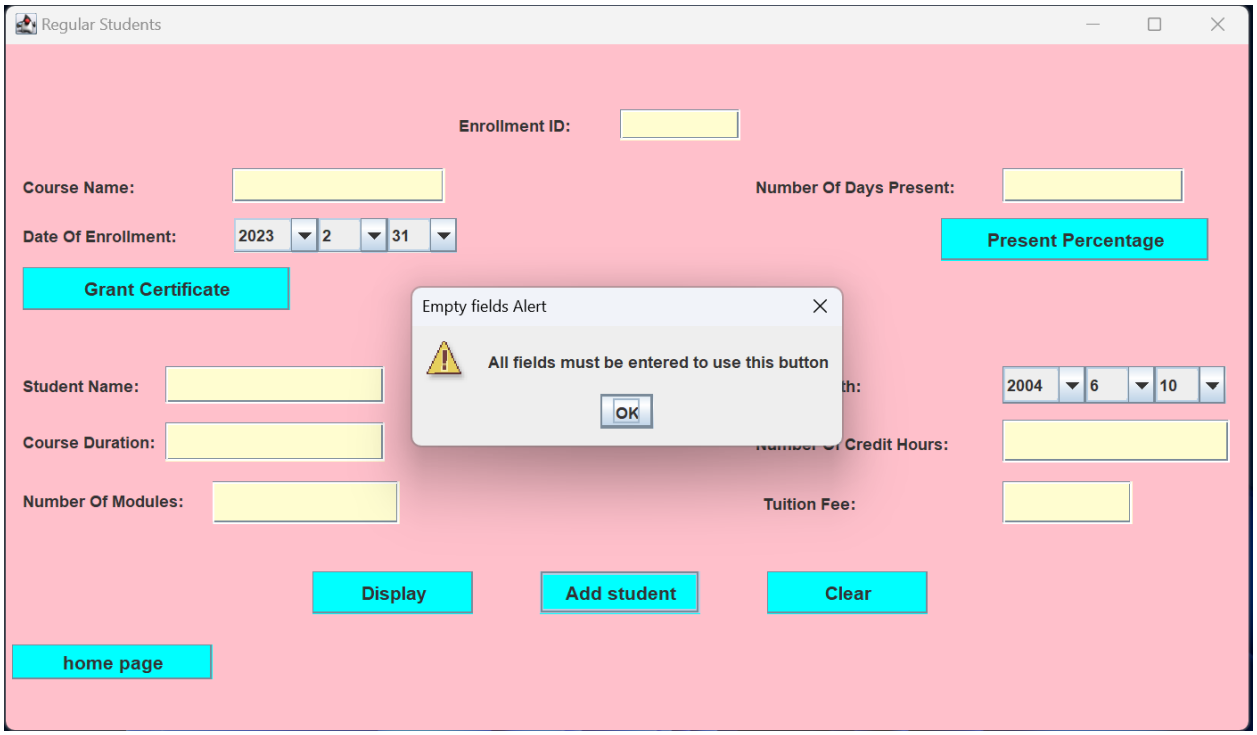

FIG 23:ADDING STUDENT WHEN NONE OF THE FIELDS ARE ENTERED

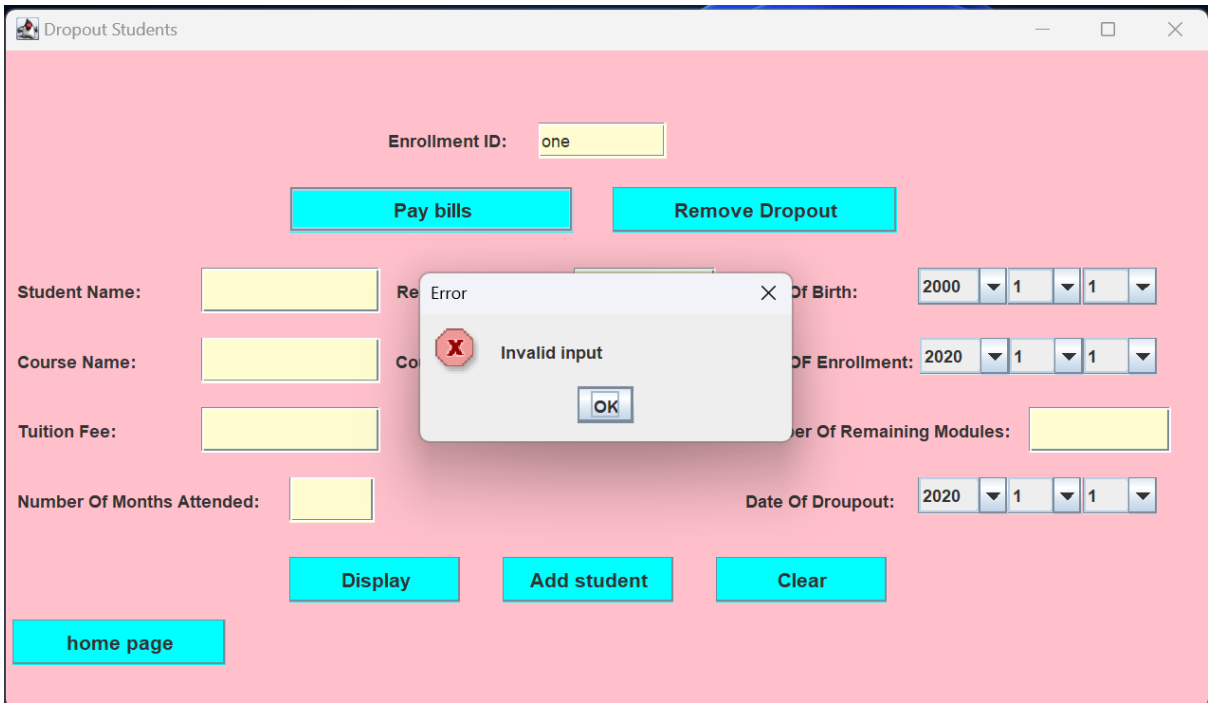

FIG 24:ADDING STUDENT WITH ENROLMENT ID ENTERED IN AN INCORRECT INPUT

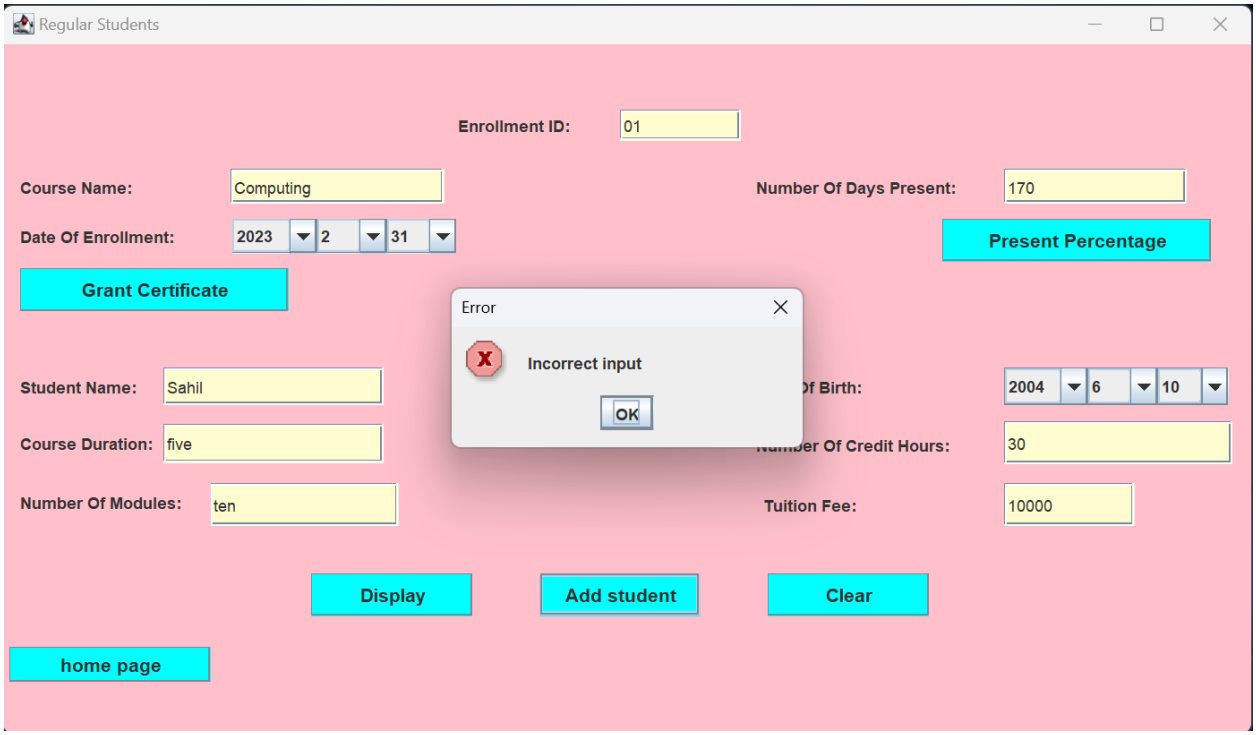

FIG 25:ADDING STUDENT WITH STRING VALUES ENTERED AT PLAXES WHERE INT SHOULD BE PROVIDED

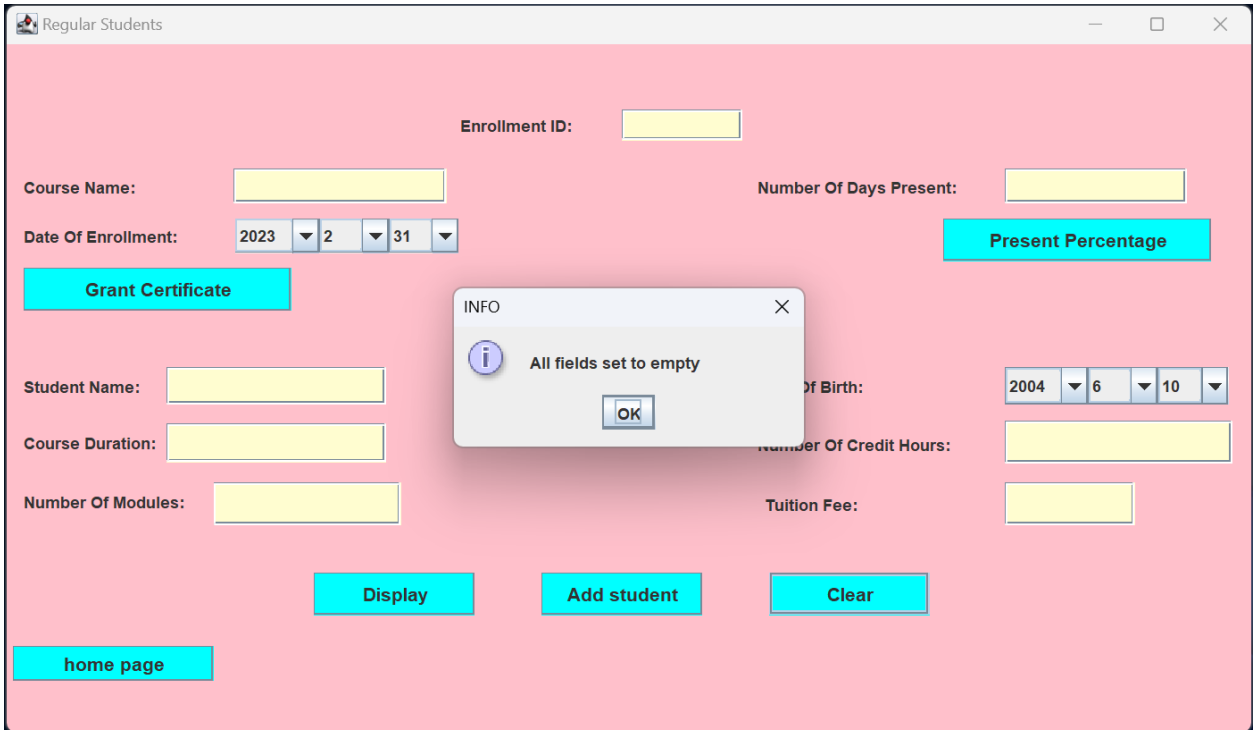

FIG 26:USING THE CLEAR BUTTON

Using the present Percentage Button without filling the Number Of Days Present Textfield

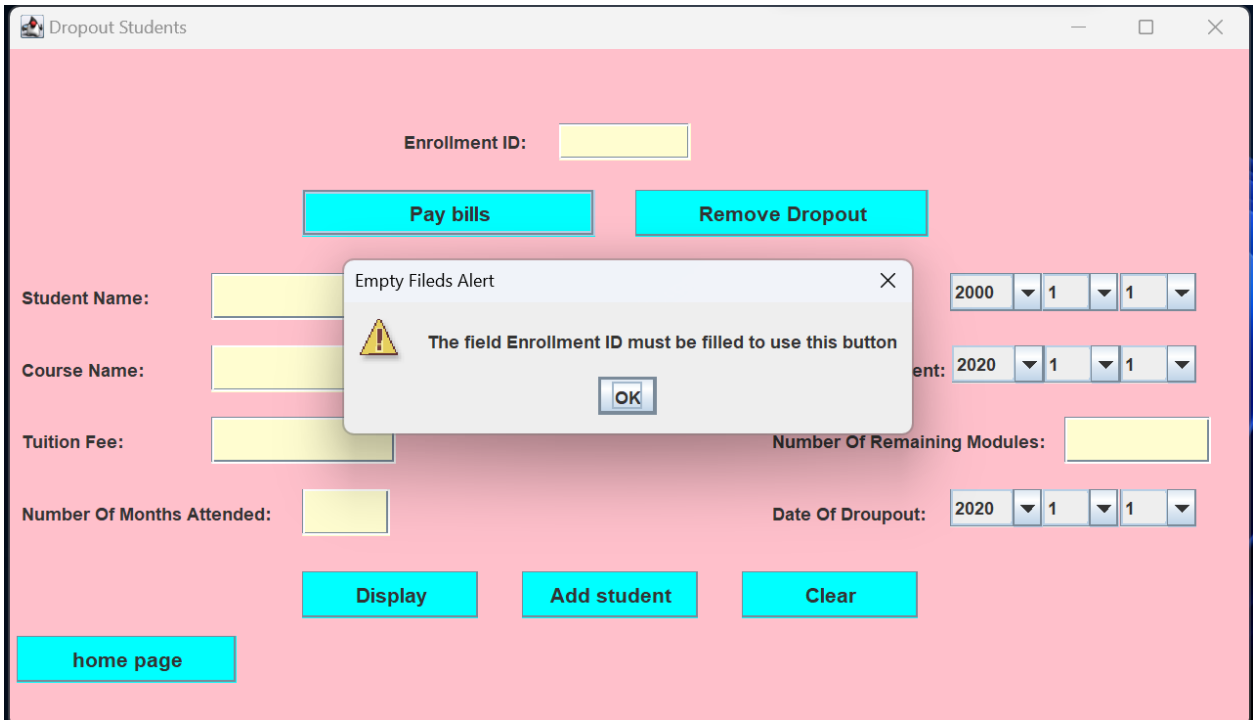

FIG 27:USING THE PAY BILLS WITHOUT ENTERING THE ENROLMENT ID

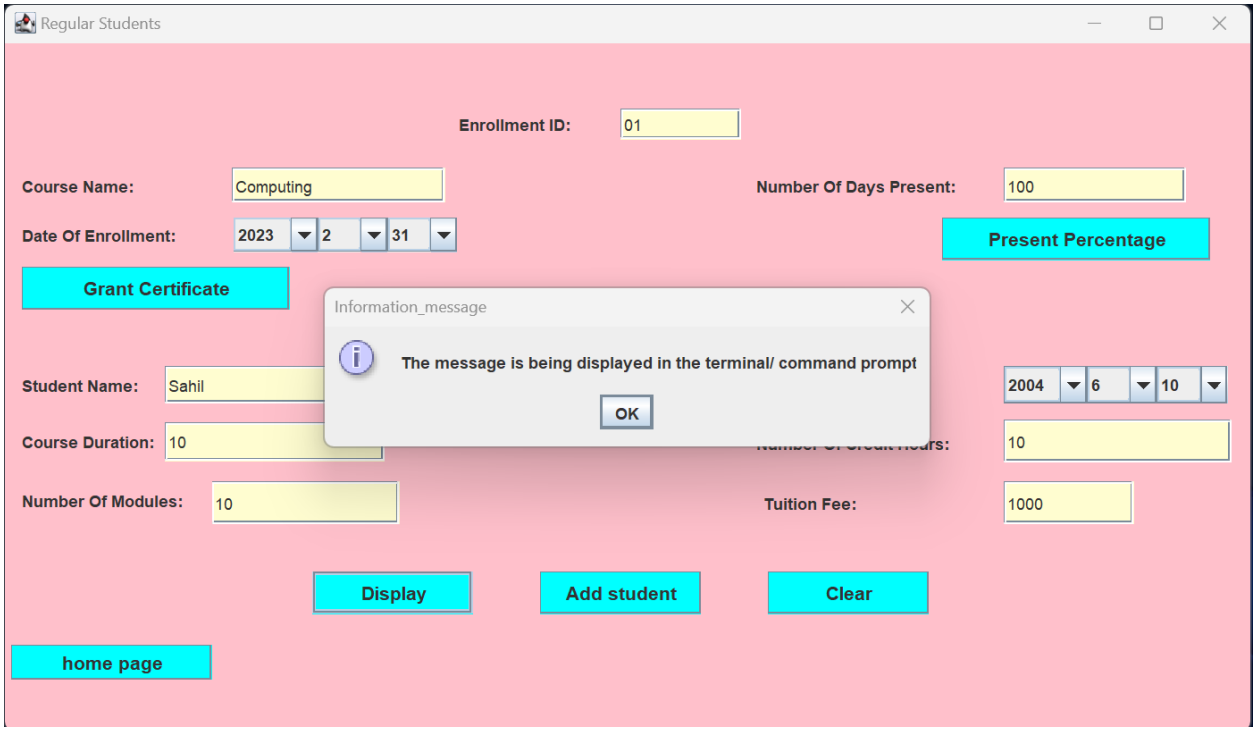

FIG 28:USING THE DISPLAY BUTTON

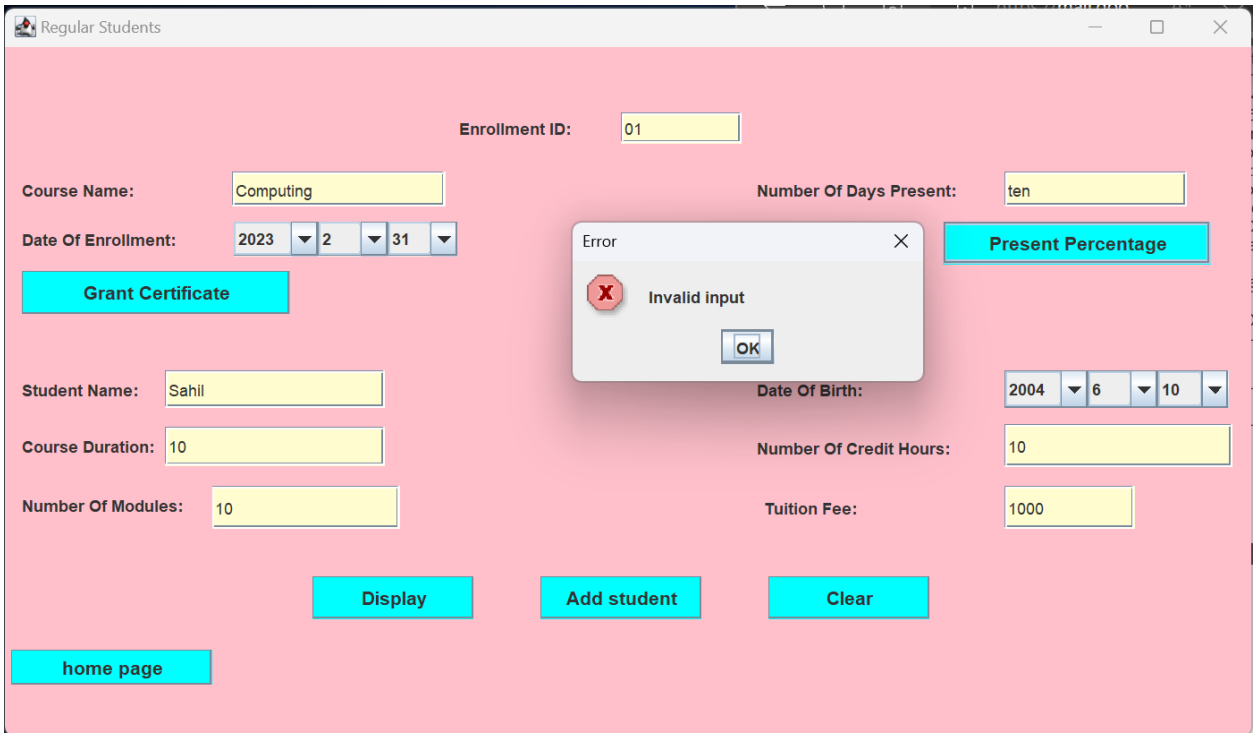

FIG 29:CALCULATING PRESENT PERCENTAGE WITH STRING INPUT IN NUMBER OF DAYS PRESENT TEXTFIELD

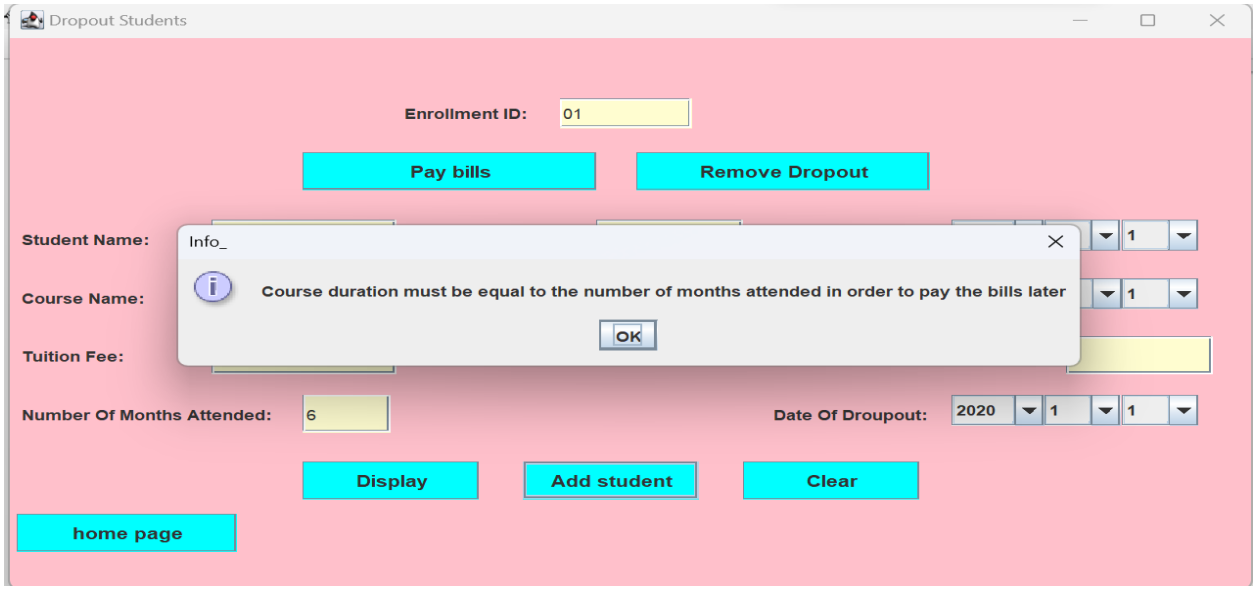

FIG 30: ADDING STUDENT WITH ENTERING INPUTS WHERE NUMBER OF MONTHS ATTENDED IS NOT EQUAL TO COURSEDURATION WHICH DOES NOT ALLOW THE BILL TO BE PAID

6.Error Detection and correction

6.1 Syntax Error

A syntax error is an error in the syntax of a coding or programming language,entered by a programmer. Syntax erros are caught by a software program called a compiler, and the programmer must fix them before the program is compiled and run.

Error:

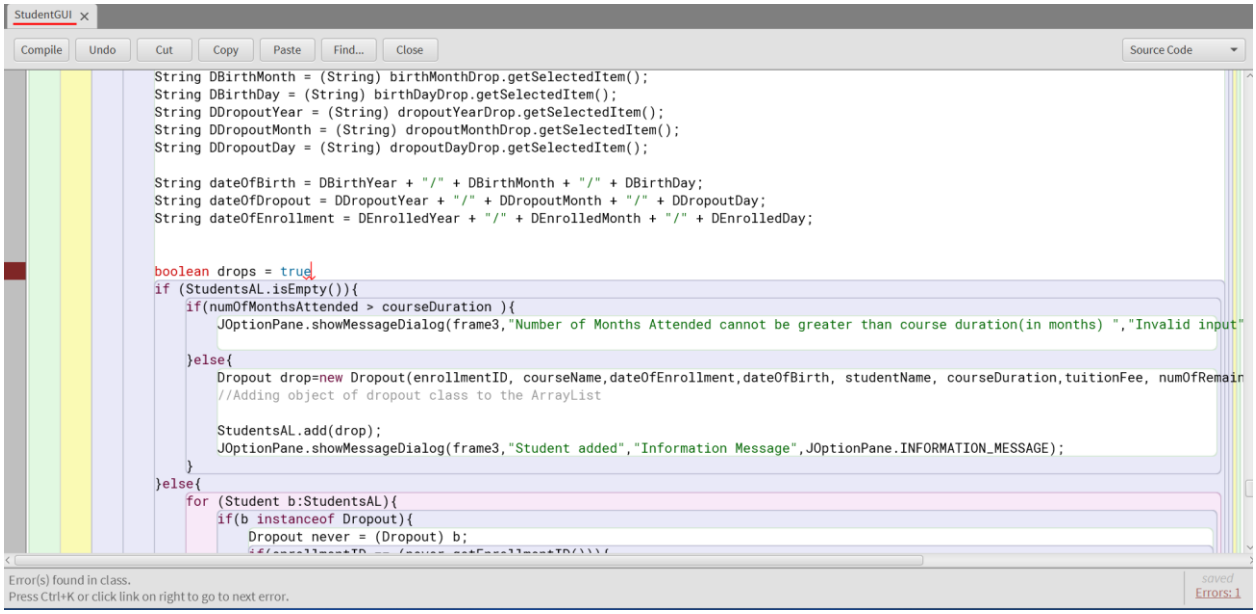

FIG 31: SYNTAX ERROR

The above figure shows a semantic error where the semi- colon is missing hence preventing the program from being compiled.

Error correction

Adding of the semi-colon

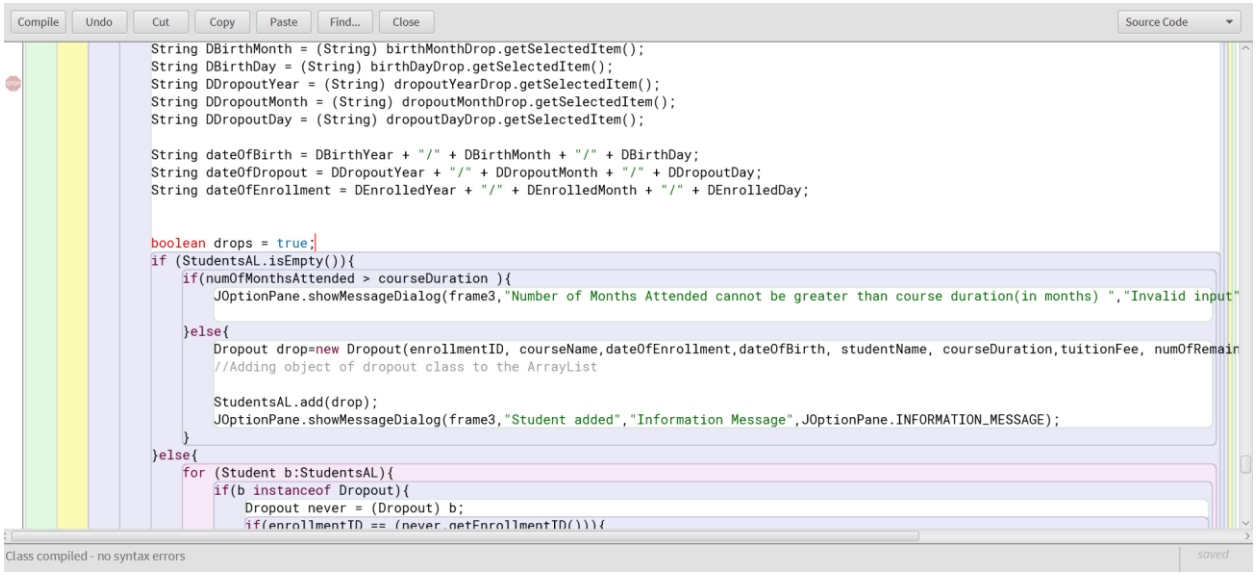

FIG 32: SYNTAX ERROR CORRECTION

6.2 Semantic Error

Semantic erros are a type of compile errors which are grammatically correct unlike syntax errors. A semantic error is a violation of the rules of meaning of a programming language

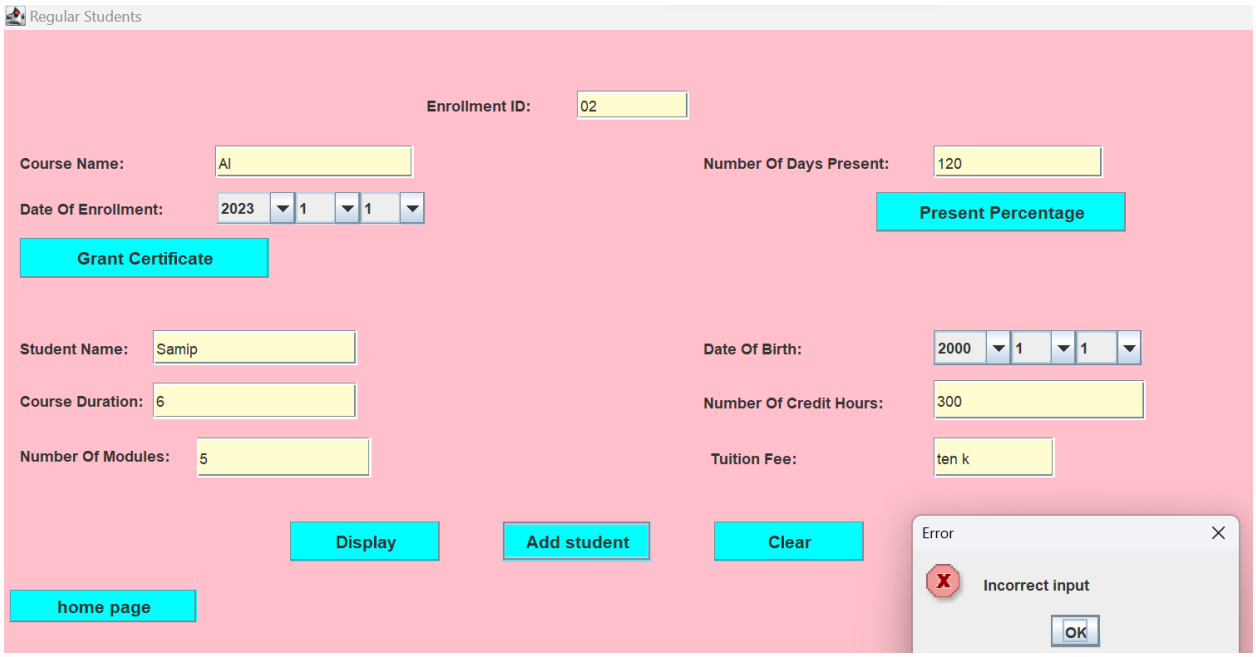

FIG 33 : SEMANTIC ERROR DETECTION

Here the tuition fee field is not filled with integer data type

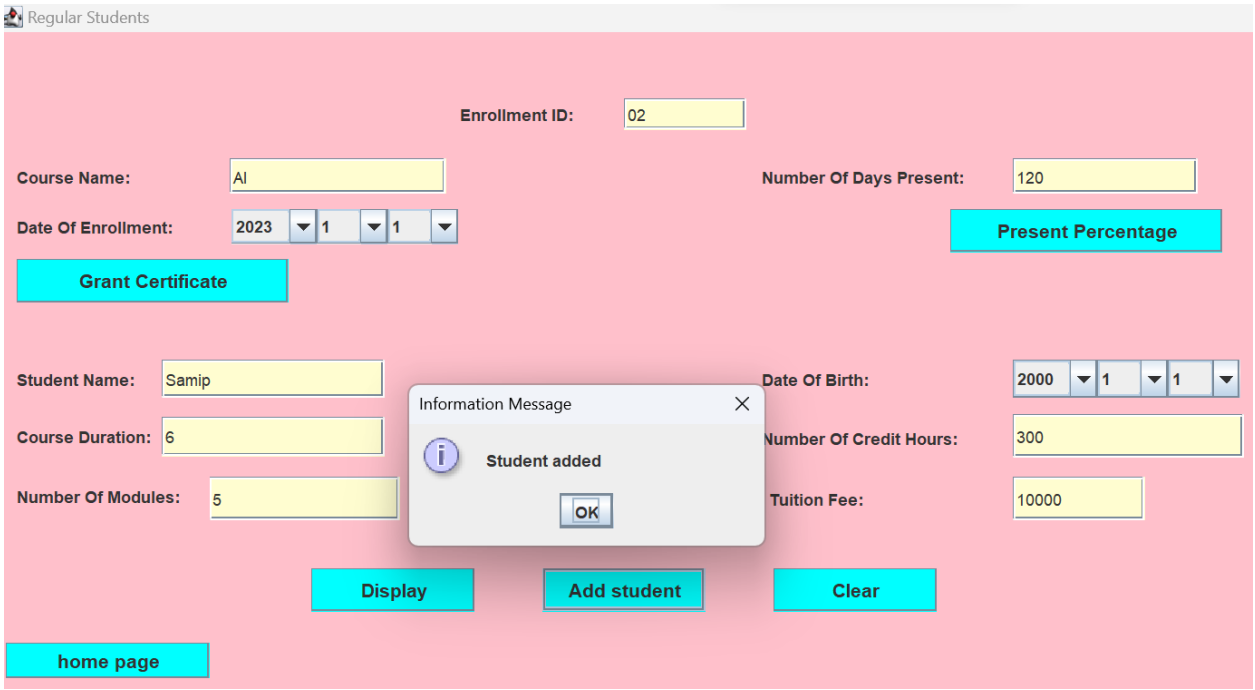

FIG 34: SEMANTIC ERROR CORRECTION

Here, the semantic error has been corrected by putting integer value for the tuition fee

6.3 Logical Error

A logical error is a bug in a program that causes it to operate incorrectly, but not to terminate abnormally. A logic error produces unintended or undesired output or other behaviour, although it may not immediately be recognized as such

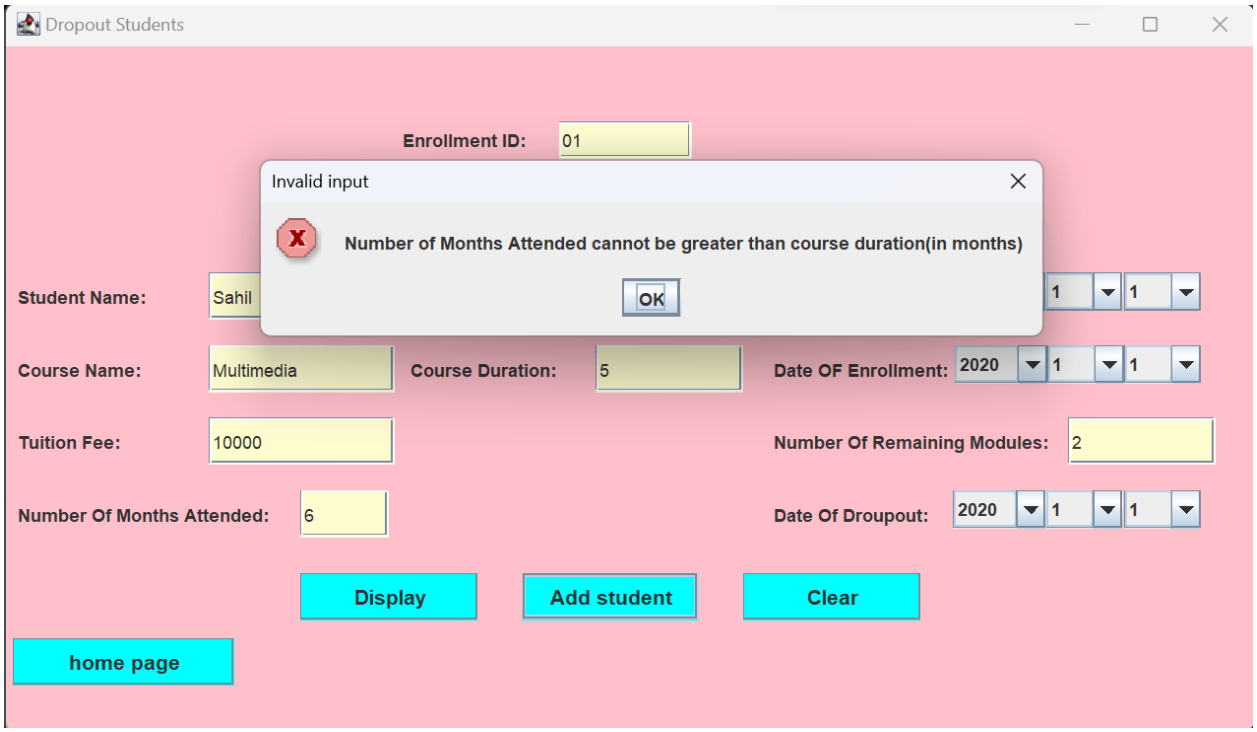

FIG 35: LOGICAL ERROR

Here, the error is caused because the number of months attended is not equal to course duration which disallows the bills to be paid later

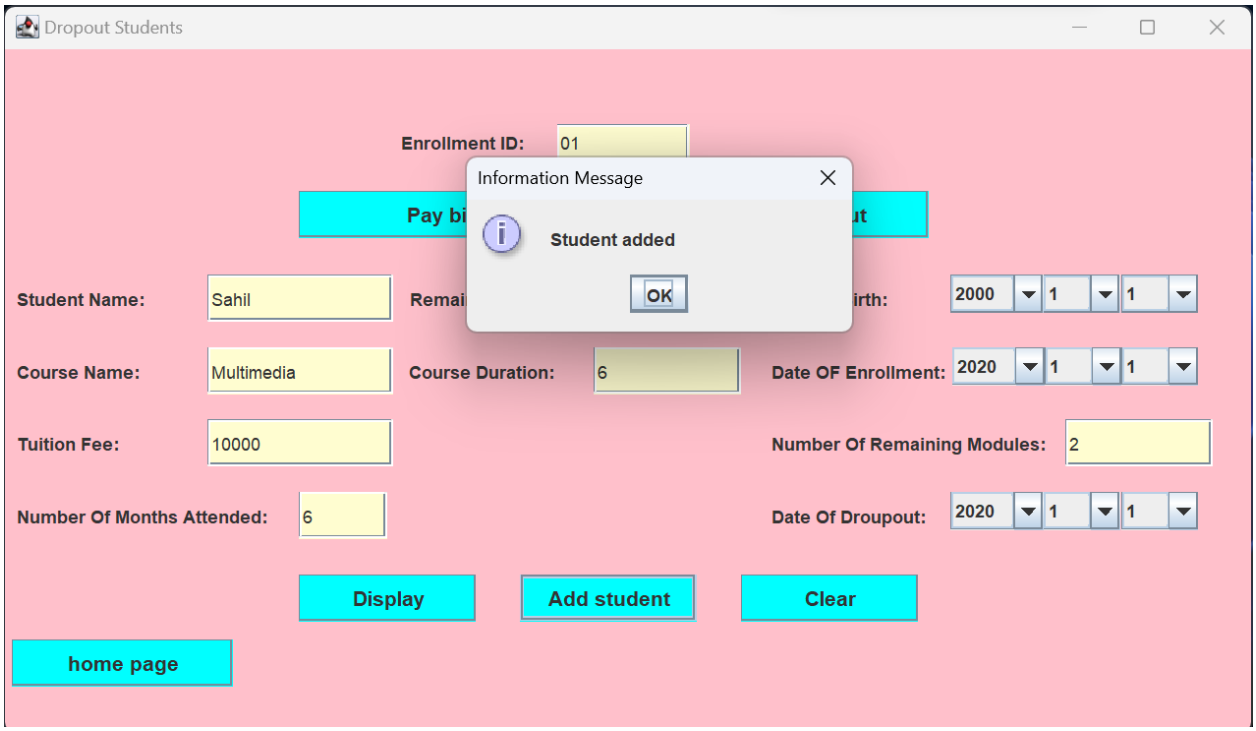

FIG 36:LOGICAL ERROR CORRECTION

FIG 1

Here, the courseDuration is made equal to the number of months attended. While the courseDuration is equal to the number of months attended the program runs smoothly and the student is added.

7. Conclusion

In conclusion, the coursework for the programming module was both insightful and difficult. I was able to acquire a better understanding of the Java programing in GUI and overall in Java programming.The coursework and the curriculum as a whole assisted me in honing my logical thinking and problem solving abilities, which are considered to be very important virtues of a programmer.Additionally, the coursework allowed me with the opportunity to put my knowledge to use in the real world problems solving and increased my self-assurance while utilizing java. I am highly satisfied with the amount of efforts that I have put in this project. Despite the difficulties, with the aid of my module teacher, books and online platforms , I was able to complete my coursework on time. Overalll, the coursework for the programming module was a worthwhile educational experience that has well-equipped me with knowledge and skills for my future endeavours in the field if computer science

Overall , it was a wholesome learning experience that allowed me the practical use of my knowledge increasing both my knowledge and experience in the field of Java programming.

8.Appendix:

```
/**
* This class represents a Student and contains information about the student.
*
* Author:Sahil Bista
* Version 1.0
*/

import javax.swing.*;
import java.awt.event.*;
import java.awt.Font;
import java.awt.Color;
import java.util.ArrayList;

public class StudentGUI implements ActionListener
{
    private JFrame frame1,frame2,frame3;
    private JLabel sName,enrollmenTID,cName,cDuration,tuitionFee,numberOfModules,numOfCreditHours,numOfDaysPresent/*sName2*/

,dateOFBirth,dateOFEnrollment,header,dropEnrollmentID,dropCName,dropSName,dropDOB,dropDOE,dropCourseDuration,

dropTuitionFee,dropNumOfRemainingModules,dropNumOfMonthsAttended,dropDOD,dropRemAmount,image;
```

```
    private                                                    JTextField
sNametf,enrollmentIDtf,cNametf,cDurationtf,tuitionFeetf,numofModtf,numofCreditHrs
tf,numofDaysPresenttf,


enrollmentIDdf,cNamedf,sNamedf,cDurationdf,tuitionFee_df,numOfRemainingModul
esdf,numOfMonthsAttendeddf,remAmountdf;

    private                                                    JComboBox<String>
enrolledYearDrop,enrolledMonthDrop,enrolledDayDrop,birthYearDrop,birthMonthDro
p,birthDayDrop,dropoutYearDrop


,dropoutMonthDrop,dropoutDayDrop,birthYearReg,birthMonthReg,birthDayReg,enrol
ledYearReg,enrolledMonthReg,enrolledDayReg;

    private                                                    JButton
presentPercent,grantCertificates,display1,clear,payBills,removeDropouts,display2,cl
ear2,regularStudents,dropoutStudents,

        homePage,home,addRegular,addDropout;

    private ImageIcon logo;


    //Arraylist creation of the student class

    ArrayList<Student> StudentsAL= new ArrayList<Student>();




    /**

    * Initializes the graphical user interface (GUI) for managing student data.

    * Creates frames, labels, buttons, text fields, and combo boxes for regular and
dropout students.

    * Sets up event listeners, layouts, colors, and fonts for the components.
```

```
 *
 * @version 1.0
 */
 public StudentGUI()
 {
     frame1 = new JFrame("A GUI for the Student Data");
     frame2 = new JFrame("Regular Students");
     frame3 = new JFrame("Dropout Students");



     //Islington college logo addition in first frame
     logo = new ImageIcon(getClass().getResource("islington.png"));
     image = new JLabel(logo);
     image.setBounds(0,0,700,400);



     //labels for regular students:
     sName = new JLabel("Student Name:");
     enrollmenTID = new JLabel("Enrollment ID:");
     cName = new JLabel("Course Name:");
     cDuration = new JLabel("Course Duration:");
     tuitionFee = new JLabel("Tuition Fee:");
     numberOfModules = new JLabel("Number Of Modules:");
     numOfCreditHours = new JLabel("Number Of Credit Hours:");
```

```
        numOfDaysPresent = new JLabel("Number Of Days Present:");
        dateOFBirth = new JLabel("Date Of Birth:");
        dateOFEnrollment = new JLabel("Date Of Enrollment:");



        //label for the main window
        header = new JLabel("Student Registration Form:");
        // Setting the new Font for the JLabel
        Font Heading = new Font("Montserrat", Font.BOLD, 20);
        header.setFont(Heading);



        //labels for dropout students:
        dropEnrollmentID = new JLabel("Enrollment ID:");
        dropCName = new JLabel("Course Name:");
        dropSName = new JLabel("Student Name:");
        dropDOB = new JLabel("Date Of Birth:");
        dropDOE = new JLabel("Date OF Enrollment:");
        dropCourseDuration = new JLabel("Course Duration:");
        dropTuitionFee = new JLabel("Tuition Fee:");
        dropNumOfRemainingModules = new JLabel("Number Of Remaining
Modules:");
        dropNumOfMonthsAttended = new JLabel("Number Of Months Attended:");
        dropDOD = new JLabel("Date Of Droupout:");
        dropRemAmount = new JLabel("Remaining Amount:");
```

```
// Text fields for regular students:
sNametf = new JTextField();
enrollmentIDtf = new JTextField();
cNametf = new JTextField();
cDurationtf = new JTextField();
tuitionFeetf = new JTextField();
numofModtf = new JTextField();
numofCreditHrstf = new JTextField();
numofDaysPresenttf = new JTextField();


//text fields for dropout students:
enrollmentIDdf = new JTextField();
cNamedf = new JTextField();
sNamedf = new JTextField();
cDurationdf = new JTextField();
tuitionFee_df = new JTextField();
numOfRemainingModulesdf = new JTextField();
numOfMonthsAttendeddf = new JTextField();
remAmountdf = new JTextField();
```

```
//buttons for the first frame
regularStudents = new JButton("Regular Students");
dropoutStudents = new JButton("Dropout Students");



//buttons for regular students:
presentPercent = new JButton("Present Percentage");
grantCertificates = new JButton("Grant Certificate");
display1 = new JButton("Display");
clear = new JButton("Clear");
homePage = new JButton("home page");
addRegular = new JButton("Add student");



//buttons for dropout students
payBills = new JButton("Pay bills");
removeDropouts = new JButton("Remove Dropout");
display2 = new JButton("Display");
clear2 = new JButton("Clear");
home = new JButton("home page");
addDropout = new JButton("Add student");



//comboboxes for dropout students interface:
```

```
String[]EnrolledYear = {"2020","2021","2022","2023","2024"};

enrolledYearDrop = new JComboBox <String> (EnrolledYear);


String[]EnrolledMonth = {"1","2","3","4","5","6","7","8","9","10","11","12"};

enrolledMonthDrop = new JComboBox <String> (EnrolledMonth);


String[]EnrolledDay = {"1","2","3","4","5","6","7","8","9","10","11","12","13","14","15","16","17","18","19","20","21","22","23","24","25","26","27","28","29","30","31"};

enrolledDayDrop = new JComboBox <String> (EnrolledDay);


String[]BirthYear = {"2000","2001","2002","2003","2004","2005"};

birthYearDrop = new JComboBox <String> (BirthYear);


String[]BirthMonth = {"1","2","3","4","5","6","7","8","9","10","11","12"};

birthMonthDrop = new JComboBox <String> (BirthMonth);


String[]BirthDay = {"1","2","3","4","5","6","7","8","9","10","11","12","13","14","15","16","17","18","19","20","21","22","23","24","25","26","27","28","29","30","31"};

birthDayDrop = new JComboBox <String> (BirthDay);


String[]DropoutYear = {"2020","2021","2022","2023","2024"};

dropoutYearDrop = new JComboBox <String> (DropoutYear);
```

```
String[]DropoutMonth = {"1","2","3","4","5","6","7","8","9","10","11","12"};

dropoutMonthDrop = new JComboBox <String> (DropoutMonth);


String[]DropoutDay = {"1","2","3","4","5","6","7","8","9","10","11","12","13","14","15","16","17","18","19","20","21","22","23","24","25","26","27","28","29","30","31"};

dropoutDayDrop = new JComboBox <String> (DropoutDay);




// Comboboxes for regular students interface

String[]birthYear = {"2000","2001","2002","2003","2004","2005"};

birthYearReg = new JComboBox <String>(BirthYear);


String[]birthMonth = {"1","2","3","4","5","6","7","8","9","10","11","12"};

birthMonthReg = new JComboBox <String> (BirthMonth);


String[]birthDay = {"1","2","3","4","5","6","7","8","9","10","11","12","13","14","15","16","17","18","19","20","21","22","23","24","25","26","27","28","29","30","31"};

birthDayReg = new JComboBox <String>(BirthDay);


String[]enrolledYear = {"2020","2021","2022","2023","2024"};

enrolledYearReg = new JComboBox<String>(EnrolledYear);
```

```
String[]enrolledMonth = {"1","2","3","4","5","6","7","8","9","10","11","12"};
enrolledMonthReg = new JComboBox <String>(EnrolledMonth);

String[]enrolledDay = {"1","2","3","4","5","6","7","8","9","10","11","12","13","14","15","16","17","18","19","20","21","22","23","24","25","26","27","28","29","30","31"};
enrolledDayReg = new JComboBox<String> (EnrolledDay);

//Set bounds for the first frame:
regularStudents.setBounds(166,350,180,32);
dropoutStudents.setBounds(359,350,180,32);

// set Bounds for regular students:
sName.setBounds(13,241,100,28);
sNametf.setBounds(120,241,164,28);
enrollmenTID.setBounds(339, 49, 120, 23);
enrollmentIDtf.setBounds(460, 49, 90, 23);
cName.setBounds(13,93,153,26);
cNametf.setBounds(170,93,159,26);
cDuration.setBounds(13,283,100,29);//
```

```
cDurationtf.setBounds(120,283,164,29);
tuitionFee.setBounds(566,327,232,32);
tuitionFeetf.setBounds(746,327,97,32);
numberOfModules.setBounds(13,326,150,31);
numofModtf.setBounds(155,327,140,32);
numOfCreditHours.setBounds(561,282,232,32);
numofCreditHrstf.setBounds(746,281,170,32);
numOfDaysPresent.setBounds(561,93,205,26);
numofDaysPresenttf.setBounds(746,93,136,26);
dateOFBirth.setBounds(561,241,180,28);
enrolledYearReg.setBounds(171,130,63,26);
enrolledMonthReg.setBounds(234,130,52,26);
enrolledDayReg.setBounds(286,130,52,26);
dateOFEnrollment.setBounds(13,130,153,26);
birthYearReg.setBounds(746,241,62,28);
birthMonthReg.setBounds(808,241,52,28);
birthDayReg.setBounds(860,241,53,28);
header.setBounds(225,40,400,30);
presentPercent.setBounds(700,130,200,32);
grantCertificates.setBounds(13,167,200,32);
display1.setBounds(230,394,120,32);
clear.setBounds(570,394,120,32);
homePage.setBounds(5,449,150,26);
addRegular.setBounds(400,394,120,32);
```

```
//set Bounds for dropouts
dropEnrollmentID.setBounds(269,51,100,25);
enrollmentIDdf.setBounds(375,51,90,25);
dropCName.setBounds(8,202,121,32);
cNamedf.setBounds(138,202,126,32);
dropSName.setBounds(8,153,121,32);
sNamedf.setBounds(138,153,126,32);
dropDOB.setBounds(520,153,153,32);
birthYearDrop.setBounds(642,153,63,26);
birthMonthDrop.setBounds(705,153,52,26);
birthDayDrop.setBounds(758,153,52,26);
dropDOE.setBounds(520,202,149,32);
enrolledYearDrop.setBounds(643,202,63,26);
enrolledMonthDrop.setBounds(706,202,52,26);
enrolledDayDrop.setBounds(758,202,52,26);
dropCourseDuration.setBounds(274,202,130,32);
cDurationdf.setBounds(400,202,100,32);
dropTuitionFee.setBounds(8,251,121,32);
tuitionFee_df.setBounds(138,251,126,32);
dropNumOfRemainingModules.setBounds(520,251,187,32);
numOfRemainingModulesdf.setBounds(720,251,100,32);
dropNumOfMonthsAttended.setBounds(8,300,190,32);
```

```
numOfMonthsAttendeddf.setBounds(200,300,60,32);
dropDOD.setBounds(520,300,146,32);
dropoutYearDrop.setBounds(642,300,63,26);
dropoutMonthDrop.setBounds(705,300,52,26);
dropoutDayDrop.setBounds(758,300,52,26);
payBills.setBounds(200,96,200,32);
removeDropouts.setBounds(427,96,200,32);
display2.setBounds(200,356,120,32);
addDropout.setBounds(350,356,120,32);
clear2.setBounds(500,356,120,32);
home.setBounds(5,400,150,32);
dropRemAmount.setBounds(275,153,153,32);
remAmountdf.setBounds(400,153,100,32);


//adding comopinents to frame 1
frame1.add(regularStudents);
frame1.add(dropoutStudents);
frame1.add(header);
frame1.add(image);


//adding components to frame 2
frame2.add(sName);
```

```
frame2.add(enrollmenTID);
frame2.add(cName);
frame2.add(cDuration);
frame2.add(tuitionFee);
frame2.add(numberOfModules);
frame2.add(numOfCreditHours);
frame2.add(numOfDaysPresent);
frame2.add(dateOFBirth);
frame2.add(dateOFEnrollment);
frame2.add(sNametf);
frame2.add(enrollmentIDtf);
frame2.add(cNametf);
frame2.add(cDurationtf);
frame2.add(tuitionFeetf);
frame2.add(numofModtf);
frame2.add(numofCreditHrstf);
frame2.add(numofDaysPresenttf);
frame2.add(birthYearReg);
frame2.add(birthMonthReg);
frame2.add(birthDayReg);
frame2.add(enrolledYearReg);
frame2.add(enrolledMonthReg);
frame2.add(enrolledDayReg);
frame2.add(presentPercent);
```

```
frame2.add(grantCertificates);
frame2.add(display1);
frame2.add(clear);
frame2.add(homePage);
frame2.add(addRegular);

//adding components to frame3
frame3.add(dropEnrollmentID);
frame3.add(dropCName);
frame3.add(dropSName);
frame3.add(dropDOB);
frame3.add(dropDOE);
frame3.add(dropCourseDuration);
frame3.add(dropTuitionFee);
frame3.add(dropNumOfRemainingModules);
frame3.add(dropNumOfMonthsAttended);
frame3.add(dropDOD);
frame3.add(enrollmentIDdf);
frame3.add(cNamedf);
frame3.add(sNamedf);
frame3.add(cDurationdf);
frame3.add(tuitionFee_df);
frame3.add(numOfRemainingModulesdf);
frame3.add(numOfMonthsAttendeddf);
```

```
frame3.add(enrolledYearDrop);
frame3.add(enrolledMonthDrop);
frame3.add(enrolledDayDrop);
frame3.add(birthYearDrop);
frame3.add(birthMonthDrop);
frame3.add(birthDayDrop);
frame3.add(dropoutYearDrop);
frame3.add(dropoutMonthDrop);
frame3.add(dropoutDayDrop);
frame3.add(payBills);
frame3.add(removeDropouts);
frame3.add(display2);
frame3.add(clear2);
frame3.add(home);
frame3.add(addDropout);
frame3.add(remAmountdf);
frame3.add(dropRemAmount);

//adding action listeners for buttons
regularStudents.addActionListener(this);
dropoutStudents.addActionListener(this);
presentPercent.addActionListener(this);
grantCertificates.addActionListener(this);
display1.addActionListener(this);
```

```
clear.addActionListener(this);
payBills.addActionListener(this);
removeDropouts.addActionListener(this);
display2.addActionListener(this);
clear2.addActionListener(this);
grantCertificates.addActionListener(this);
homePage.addActionListener(this);
home.addActionListener(this);
addRegular.addActionListener(this);
addDropout.addActionListener(this);

//Making frame1 visible
frame1.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
frame1.setLayout(null);
frame1.setSize(739,455);
frame1.setVisible(true);



Color ButtonColor = new Color(0, 255, 255);
 //Using color class to create a new color:cyan using RGB values
presentPercent.setBackground(ButtonColor);
grantCertificates.setBackground(ButtonColor);
display1.setBackground(ButtonColor);
clear.setBackground(ButtonColor);
```

```
payBills.setBackground(ButtonColor);
removeDropouts.setBackground(ButtonColor);
display2.setBackground(ButtonColor);
clear2.setBackground(ButtonColor);
addRegular.setBackground(ButtonColor);
addDropout.setBackground(ButtonColor);
home.setBackground(ButtonColor);
homePage.setBackground(ButtonColor);
regularStudents.setBackground(ButtonColor);
dropoutStudents.setBackground(ButtonColor);



Color frameBackgroundColor = new Color(255, 192, 203);
// Pink color for the frames' background
frame2.getContentPane().setBackground(frameBackgroundColor);
frame3.getContentPane().setBackground(frameBackgroundColor);
frame1.getContentPane().setBackground(frameBackgroundColor);



Color TextFieldColor = new Color(255, 253, 208);
// Creamy  color for textfields
sNametf.setBackground(TextFieldColor);
enrollmentIDtf.setBackground(TextFieldColor);
cNametf.setBackground(TextFieldColor);
```

```
cDurationtf.setBackground(TextFieldColor);
tuitionFeetf.setBackground(TextFieldColor);
numofModtf.setBackground(TextFieldColor);
numofCreditHrstf.setBackground(TextFieldColor);
numofDaysPresenttf.setBackground(TextFieldColor);
enrollmentIDdf.setBackground(TextFieldColor);
cNamedf.setBackground(TextFieldColor);
sNamedf.setBackground(TextFieldColor);
cDurationdf.setBackground(TextFieldColor);
tuitionFee_df.setBackground(TextFieldColor);
numOfRemainingModulesdf.setBackground(TextFieldColor);
numOfMonthsAttendeddf.setBackground(TextFieldColor);
remAmountdf.setBackground(TextFieldColor);



Font Button_Font = new Font("Arial", Font.BOLD, 14);
//Increasing the font size by a bit and changing the font using Font class from AWT package for buttons
presentPercent.setFont(Button_Font);
grantCertificates.setFont(Button_Font);
display1.setFont(Button_Font);
clear.setFont(Button_Font);
payBills.setFont(Button_Font);
removeDropouts.setFont(Button_Font);
display2.setFont(Button_Font);
```

```
clear2.setFont(Button_Font);

addRegular.setFont(Button_Font);

addDropout.setFont(Button_Font);

home.setFont(Button_Font);

homePage.setFont(Button_Font);

regularStudents.setFont(Button_Font);

dropoutStudents.setFont(Button_Font);



Font Label_Font = new Font("Montserrat", Font.BOLD, 12);

//Increasing the font size by a bit and changing the font using Font class from AWT package for JLabels

sName.setFont(Label_Font );

enrollmenTID.setFont(Label_Font );

cName.setFont(Label_Font );

cDuration.setFont(Label_Font );

tuitionFee.setFont(Label_Font );

numberOfModules.setFont(Label_Font );

numOfCreditHours.setFont(Label_Font );

numOfDaysPresent.setFont(Label_Font );

dateOFBirth.setFont(Label_Font );

dateOFEnrollment.setFont(Label_Font );

dropEnrollmentID.setFont(Label_Font );

dropCName.setFont(Label_Font );

dropSName.setFont(Label_Font );
```

```
        dropDOB.setFont(Label_Font );
        dropDOE.setFont(Label_Font );
        dropCourseDuration.setFont(Label_Font );
        dropTuitionFee.setFont(Label_Font );
        dropNumOfRemainingModules.setFont(Label_Font );
        dropNumOfMonthsAttended.setFont(Label_Font );
        dropDOD.setFont(Label_Font );
    }

    /**
    * Responds to user actions on various buttons by performing corresponding actions.
    * Handles button clicks for opening frames, clearing fields, calculating percentages,
    * granting certificates, paying bills, removing students, and adding students.
    *
    * @param e The ActionEvent object representing the user's action.
    */

    @Override
    public void actionPerformed(ActionEvent e)
    {
        if(e.getSource() == regularStudents){
            //This button is used to open the frame2 of regular students
            frame2.setLayout(null);
            frame2.setSize(944,550);
```

```
            frame2.setVisible(true);
            frame1.dispose();


        }
        else if(e.getSource() == dropoutStudents){
            //This button is used to open the frame3 of dropout students
            frame3.setLayout(null);
            frame3.setSize(860,500);
            frame3.setVisible(true);
            frame1.dispose();


        }
        else if(e.getSource() == clear){
            //This button is used to clear all text fields of regular students
            sNametf.setText("");
            enrollmentIDtf.setText("");
            cNametf.setText("");
            cDurationtf.setText("");
            tuitionFeetf.setText("");
            numofModtf.setText("");
            numofCreditHrstf.setText("");
            numofDaysPresenttf.setText("");

            JOptionPane.showMessageDialog(frame2,"All fields set to
empty","INFO",JOptionPane.INFORMATION_MESSAGE);
```

```
        }
        else if(e.getSource() == clear2){
            //This button is used to clear all text fields of droput students
            enrollmentIDdf.setText("");
            cNamedf.setText("");
            sNamedf.setText("");
            cDurationdf.setText("");
            tuitionFee_df.setText("");
            numOfRemainingModulesdf.setText("");
            numOfMonthsAttendeddf.setText("");
            remAmountdf.setText("");

            JOptionPane.showMessageDialog(frame3,"All fields set to empty","Information",JOptionPane.INFORMATION_MESSAGE);

        }
        else if(e.getSource() == homePage){
            //This button takes the user back to frame1 from regular students interface
            frame1.setLayout(null);
            frame1.setSize(739,455);
            frame1.setVisible(true);
            frame2.dispose();
            frame3.dispose();
```

```
        }
        else if(e.getSource() == home){
            //This button takes the user back to frame1 from droput students interface
            frame1.setLayout(null);
            frame1.setSize(739,455);
            frame1.setVisible(true);
            frame2.dispose();
            frame3.dispose();



        }
        else if(e.getSource() == grantCertificates){
            if(enrollmentIDtf.getText().isEmpty() || cNametf.getText().isEmpty()){
                JOptionPane.showMessageDialog(frame2,"The fields Enrollment ID, Date of Enrollment and Course Name must all be filled to use this button","Empty fields Alert",JOptionPane.WARNING_MESSAGE);
            }
            else{
                //try catch to catch number format exception
                try{
                    //extracting the values from text fields to store them in the parent class variables
                    int enrollmentID = Integer.parseInt(enrollmentIDtf.getText());
                    String courseName = cNametf.getText();
                    String REnrolledYear = (String) enrolledYearReg.getSelectedItem();
```

```
String REnrolledMonth = (String) enrolledMonthReg.getSelectedItem();
String REnrolledDay = (String) enrolledDayReg.getSelectedItem();
String dateOfEnrollment = REnrolledYear + "/" + REnrolledMonth + "/" + REnrolledDay;


boolean Certificate = true;


//Iterating throught the studenyt class' ArrayList
for(Student j:StudentsAL){
   //checking if the object belongs to Regular class
   if(j instanceof Regular){
      //Typecasting of the regular class
      Regular certification = (Regular) j;
      //checking if the student already exists
      if(enrollmentID == (certification.getEnrollmentID())){
         Certificate = true;
         //calling the grantCertificate() method from the Regular class
         certification. grantCertificate(courseName, enrollmentID,dateOfEnrollment);
         JOptionPane.showMessageDialog(frame2,"Certificate has been granted to the student","information",JOptionPane.INFORMATION_MESSAGE);
      }
   }
}
if(Certificate = false){
```

```
                JOptionPane.showMessageDialog(frame2,"The enrollment ID doesnot match with the Id of a regular student","Error",JOptionPane.ERROR_MESSAGE);
            }
        }
        catch(NumberFormatException s){
            JOptionPane.showMessageDialog(frame3,"Invalid input","Error",JOptionPane.ERROR_MESSAGE);
        }
    }
}
else if(e.getSource() ==  presentPercent){
    //button for calculating the presentPercentage of the regular students

if(enrollmentIDtf.getText().isEmpty()||numofDaysPresenttf.getText().isEmpty()){
        JOptionPane.showMessageDialog(frame2,"The fields Enrollment ID and Number of Days present must be filled to use this button","Empty Fileds Alert",JOptionPane.WARNING_MESSAGE);
    }else{
        try{
            int enrollmentID = Integer.parseInt(enrollmentIDtf.getText());
            int daysPresent = Integer.parseInt(numofDaysPresenttf.getText());

            boolean pass = true;
            for(Student c:StudentsAL){
                if(c instanceof Regular){
```

```
                    Regular ppCalculation = (Regular) c;
                    if(enrollmentID == (ppCalculation.getEnrollmentID())){
                        pass = true;
                        //calling of the presentPercentage(double daysPresent) method from the Regular class
                        ppCalculation.presentPercentage(daysPresent);
                        JOptionPane.showMessageDialog(frame2,"Present percentage has been calculated","information",JOptionPane.INFORMATION_MESSAGE);
                    }
                  }
                }
                if(pass = false){
                    JOptionPane.showMessageDialog(frame3,"The enrollment ID doesnot match with the Id of a regular student","Error",JOptionPane.ERROR_MESSAGE);
                }
              }
              catch(NumberFormatException x){
                  JOptionPane.showMessageDialog(frame3,"Invalid input","Error",JOptionPane.ERROR_MESSAGE);
              }
          }
      }
      else if(e.getSource() == payBills){
          //method to pay the vills for the droput students
          if(enrollmentIDdf.getText().isEmpty()){
```

```
            JOptionPane.showMessageDialog(frame3,"The field Enrollment ID must be filled to use this button","Empty Fileds Alert",JOptionPane.WARNING_MESSAGE);
        }
        else{
            try{
                int enrollmentID = Integer.parseInt(enrollmentIDdf.getText());
                boolean money = true;
                for(Student v:StudentsAL){
                    if(v instanceof Dropout){
                        Dropout bills = (Dropout) v;
                        //calling method billsPayable() from dropout class
                        bills.billsPayable();
                        JOptionPane.showMessageDialog(frame2,"The bill has been paid","Information",JOptionPane.INFORMATION_MESSAGE);
                    }
                }
                if(money == false){
                    JOptionPane.showMessageDialog(frame3,"The enrollment ID doesnot match with the Id of a dropout student","Error",JOptionPane.ERROR_MESSAGE);
                }
            }catch(NumberFormatException u){
                JOptionPane.showMessageDialog(frame3,"Invalid input","Error",JOptionPane.ERROR_MESSAGE);
            }
        }
```

```
        }
        else if(e.getSource() == removeDropouts){
            //method to remove The Dropout student
            if(enrollmentIDdf.getText().isEmpty()){
                JOptionPane.showMessageDialog(frame3,"The field Enrollment ID must be filled to use this button", "Empty field Alert",JOptionPane.WARNING_MESSAGE);
            }
            else{
                try{
                    int enrollmentID = Integer.parseInt(enrollmentIDdf.getText());
                    boolean Drop = true;
                    for (Student a:StudentsAL){
                        if(a instanceof Dropout){
                            Dropout remove = (Dropout) a;
                            if(enrollmentID == (remove.getEnrollmentID())){
                                Drop = true;
                                //Calling of the removeStudent() method from parent class
                                remove.removeStudent();
                                JOptionPane.showMessageDialog(frame3,"The student has been removed","Information",JOptionPane.INFORMATION_MESSAGE);
                            }
                            else{
                                Drop = false;
                            }
                        }
```

```
                }
                if(Drop = false){
                    JOptionPane.showMessageDialog(frame3,"The enrollment ID is not valid for dropout student","Error",JOptionPane.ERROR_MESSAGE);
                }
            }catch(NumberFormatException n){
                JOptionPane.showMessageDialog(frame3,"Incorrect input","Error",JOptionPane.ERROR_MESSAGE);
            }
        }
    }else if(e.getSource() == addRegular){
        //method to add a regular student to the arrayList
        if(enrollmentIDtf.getText().isEmpty() || sNametf.getText().isEmpty() || cNametf.getText().isEmpty() || cDurationtf.getText().isEmpty()
            || tuitionFeetf.getText().isEmpty() || numofModtf.getText().isEmpty() || numofCreditHrstf.getText().isEmpty() ||
            numofDaysPresenttf.getText().isEmpty()){
                JOptionPane.showMessageDialog(frame2,"All fields must be entered to use this button","Empty fields Alert",JOptionPane.WARNING_MESSAGE);
            }
        else{
            try{
                int enrollmentID = Integer.parseInt(enrollmentIDtf.getText());
                String studentName = sNametf.getText();
                String courseName = cNametf.getText();
                int courseDuration = Integer.parseInt(cDurationtf.getText());
```

```
int tuitionFee = Integer.parseInt(tuitionFeetf.getText());

int numOfModules = Integer.parseInt(numofModtf.getText());

int numOfCreditHours = Integer.parseInt(numofCreditHrstf.getText());

double daysPresent = Double.parseDouble(numofDaysPresenttf.getText());

String RBirthYear = (String) birthYearReg.getSelectedItem();

String RBirthMonth = (String) birthMonthReg.getSelectedItem();

String RBirthDay = (String) birthDayReg.getSelectedItem();

String REnrolledYear = (String) enrolledYearReg.getSelectedItem();

String REnrolledMonth = (String) enrolledMonthReg.getSelectedItem();

String REnrolledDay = (String) enrolledDayReg.getSelectedItem();


String dateOfBirth = RBirthYear + "/" + RBirthMonth + "/" + RBirthDay;

String dateOfEnrollment = REnrolledYear + "/" + REnrolledMonth + "/" + REnrolledDay;


boolean regulars=true;

if (StudentsAL.isEmpty()){

  Regular reg=new Regular(enrollmentID , dateOfBirth, courseName, studentName, dateOfEnrollment,

  courseDuration,tuitionFee, numOfModules, numOfCreditHours, daysPresent);

  //Adding object of Regular class to the ArrayList

  StudentsAL.add(reg);

  JOptionPane.showMessageDialog(frame2,"Student added","Information Message",JOptionPane.INFORMATION_MESSAGE);
```

```
            }else{
                for(Student x: StudentsAL)
                {
                    if(x instanceof Regular){
                        Regular always = (Regular) x;
                        if (enrollmentID == (always.getEnrollmentID()))
                        {
                            regulars = false;
                        }
                    }
                }
                if(regulars == true)
                {
                    Regular regs = new Regular(enrollmentID , dateOfBirth, courseName, studentName, dateOfEnrollment,
                    courseDuration,tuitionFee, numOfModules, numOfCreditHours, daysPresent);
                    StudentsAL.add(regs);

                    JOptionPane.showMessageDialog(frame2,"Student added","Information Message",JOptionPane.INFORMATION_MESSAGE);
                }
                else{
                    JOptionPane.showMessageDialog(frame2,"Student already exits","Error Message",JOptionPane.ERROR_MESSAGE);
```

```
                }
            }
        }catch(NumberFormatException n){
            JOptionPane.showMessageDialog(frame2,"Incorrect input","Error",JOptionPane.ERROR_MESSAGE);
        }
    }
}
else if(e.getSource() == addDropout){
    //method to add Dropout students to the arraylist
    if(enrollmentIDdf.getText().isEmpty() || sNamedf.getText().isEmpty() || cNamedf.getText().isEmpty() || cDurationdf.getText().isEmpty()
    || tuitionFee_df.getText().isEmpty() || numOfRemainingModulesdf.getText().isEmpty() || numOfMonthsAttendeddf.getText().isEmpty())
    {
        JOptionPane.showMessageDialog(frame2,"All fields must be entered to use this button","Empty Fields Alert",JOptionPane.WARNING_MESSAGE);
    }
    try{
        int enrollmentID = Integer.parseInt(enrollmentIDdf.getText());
        String studentName = sNamedf.getText();
        String courseName = cNamedf.getText();
        int courseDuration = Integer.parseInt(cDurationdf.getText());
        int tuitionFee = Integer.parseInt(tuitionFee_df.getText());
```

```
        int numOfRemainingModules = Integer.parseInt(numOfRemainingModulesdf.getText());
        int numOfMonthsAttended = Integer.parseInt(numOfMonthsAttendeddf.getText());
        String DEnrolledYear = (String) enrolledYearDrop.getSelectedItem();
        String DEnrolledMonth = (String) enrolledMonthDrop.getSelectedItem();
        String DEnrolledDay = (String) enrolledDayDrop.getSelectedItem();
        String DBirthYear = (String) birthYearDrop.getSelectedItem();
        String DBirthMonth = (String) birthMonthDrop.getSelectedItem();
        String DBirthDay = (String) birthDayDrop.getSelectedItem();
        String DDropoutYear = (String) dropoutYearDrop.getSelectedItem();
        String DDropoutMonth = (String) dropoutMonthDrop.getSelectedItem();
        String DDropoutDay = (String) dropoutDayDrop.getSelectedItem();

        String dateOfBirth = DBirthYear + "/" + DBirthMonth + "/" + DBirthDay;
        String dateOfDropout = DDropoutYear + "/" + DDropoutMonth + "/" + DDropoutDay;
        String dateOfEnrollment = DEnrolledYear + "/" + DEnrolledMonth + "/" + DEnrolledDay;


        boolean drops = true;
        if (StudentsAL.isEmpty()){
            if(courseDuration != numOfMonthsAttended){
```

```
                JOptionPane.showMessageDialog(frame3,"Course duration must be
equal to the number of months attended in order to pay the bills
later","Info_",JOptionPane.INFORMATION_MESSAGE);
            }
            else{
                Dropout drop=new Dropout(enrollmentID,
courseName,dateOfEnrollment,dateOfBirth, studentName,
courseDuration,tuitionFee, numOfRemainingModules,
numOfMonthsAttended,dateOfDropout);
                //Adding object of dropout class to the ArrayList

                StudentsAL.add(drop);
                JOptionPane.showMessageDialog(frame3,"Student
added","Information Message",JOptionPane.INFORMATION_MESSAGE);
            }           }else{
                Dropout drop=new Dropout(enrollmentID,
courseName,dateOfEnrollment,dateOfBirth, studentName,
courseDuration,tuitionFee, numOfRemainingModules,
numOfMonthsAttended,dateOfDropout);
                //Adding object of dropout class to the ArrayList

                StudentsAL.add(drop);
                JOptionPane.showMessageDialog(frame3,"Student
added","Information Message",JOptionPane.INFORMATION_MESSAGE);
            }
        }else{
            for (Student b:StudentsAL){
```

```
                if(b instanceof Dropout){
                    Dropout never = (Dropout) b;
                    if(enrollmentID == (never.getEnrollmentID())){
                        drops = false;
                    }
                }
            }
            if(drops == true){
                Dropout dropz= new Dropout(enrollmentID,courseName, dateOfEnrollment,dateOfBirth, studentName, courseDuration,tuitionFee, numOfRemainingModules, numOfMonthsAttended,dateOfDropout);
                StudentsAL.add(dropz);

                JOptionPane.showMessageDialog(frame3,"Student added","Information Message",JOptionPane.INFORMATION_MESSAGE);
            }else{
                JOptionPane.showMessageDialog(frame3,"Student already exits","Error Message",JOptionPane.ERROR_MESSAGE);
            }
        }
    }catch(NumberFormatException v){
        JOptionPane.showMessageDialog(frame3,"Incorrect input","Error",JOptionPane.ERROR_MESSAGE);
    }
}
else if(e.getSource() == display1){
```

```
        if(enrollmentIDtf.getText().isEmpty() || sNametf.getText().isEmpty() || cNametf.getText().isEmpty() || cDurationtf.getText().isEmpty()
          || tuitionFeetf.getText().isEmpty() || numofModtf.getText().isEmpty() || numofCreditHrstf.getText().isEmpty() ||
           numofDaysPresenttf.getText().isEmpty()){
              JOptionPane.showMessageDialog(frame2,"All fields must be entered to use this button","Empty fields Alert",JOptionPane.WARNING_MESSAGE);
        }
        else{
           //method to diaplay the informatoin of the regular students
           if(StudentsAL.isEmpty()){
              JOptionPane.showMessageDialog(frame2,"No items to display");
           }
           else{
              for(Student z: StudentsAL){
                 if (z instanceof Regular){
                    Regular Rdisplay = (Regular) z;
                    //Calling of the dipslay() method from the Regular class
                    Rdisplay.display();
                    JOptionPane.showMessageDialog(frame2,"The message is being displayed in the terminal/ command prompt","Information_message",JOptionPane.INFORMATION_MESSAGE);
                 }else{
                    JOptionPane.showMessageDialog(frame2,"The student belongs to dropout class","Information_message",JOptionPane.WARNING_MESSAGE);
                 }
```

```
            }
        }
    }
}
else if(e.getSource() == display2){
    if(enrollmentIDdf.getText().isEmpty() || sNamedf.getText().isEmpty() || cNamedf.getText().isEmpty() || cDurationdf.getText().isEmpty()
    || tuitionFee_df.getText().isEmpty() || numOfRemainingModulesdf.getText().isEmpty() || numOfMonthsAttendeddf.getText().isEmpty()){
        JOptionPane.showMessageDialog(frame3,"All fields must be entered to use this button","Empty Fields Alert",JOptionPane.WARNING_MESSAGE);
    }
    else{
        //method to display the information of the dropout students
        if(StudentsAL.isEmpty()){
            JOptionPane.showMessageDialog(frame3,"No items to display");
        }
        else{
            for(Student d: StudentsAL){
                if (d instanceof Dropout){
                    Dropout Odisplay = (Dropout) d;
                    //Calling of the dipslay() method from the dropout class
                    Odisplay.display();
```

```
                    JOptionPane.showMessageDialog(frame3,"The message is being displayed in the terminal/ command prompt","Information_message",JOptionPane.INFORMATION_MESSAGE);
                }
                else{
                    JOptionPane.showMessageDialog(frame3,"The student belongs to regular class","Information_message",JOptionPane.WARNING_MESSAGE);
                }
            }
        }
    }
}

/**
* Entry point of the application.
* Initializes the graphical user interface (GUI) for managing student information.
* Creates an instance of the StudentGUI class to display the application interface.
*
*/
public static void main(String[]args){
    new StudentGUI();
}
}
```