CS4001NI Programming

30% Individual Coursework

2022-23 Autumn

Student Name: Sumit Shrestha

London Met ID: 22085637

College ID: NP01CP4S230046

Group: C18

Assignment Due Date: Friday, August 11, 2023

Assignment Submission Date: Friday, August 11, 2023

Table of Content

Table of Figures

Table of Tables

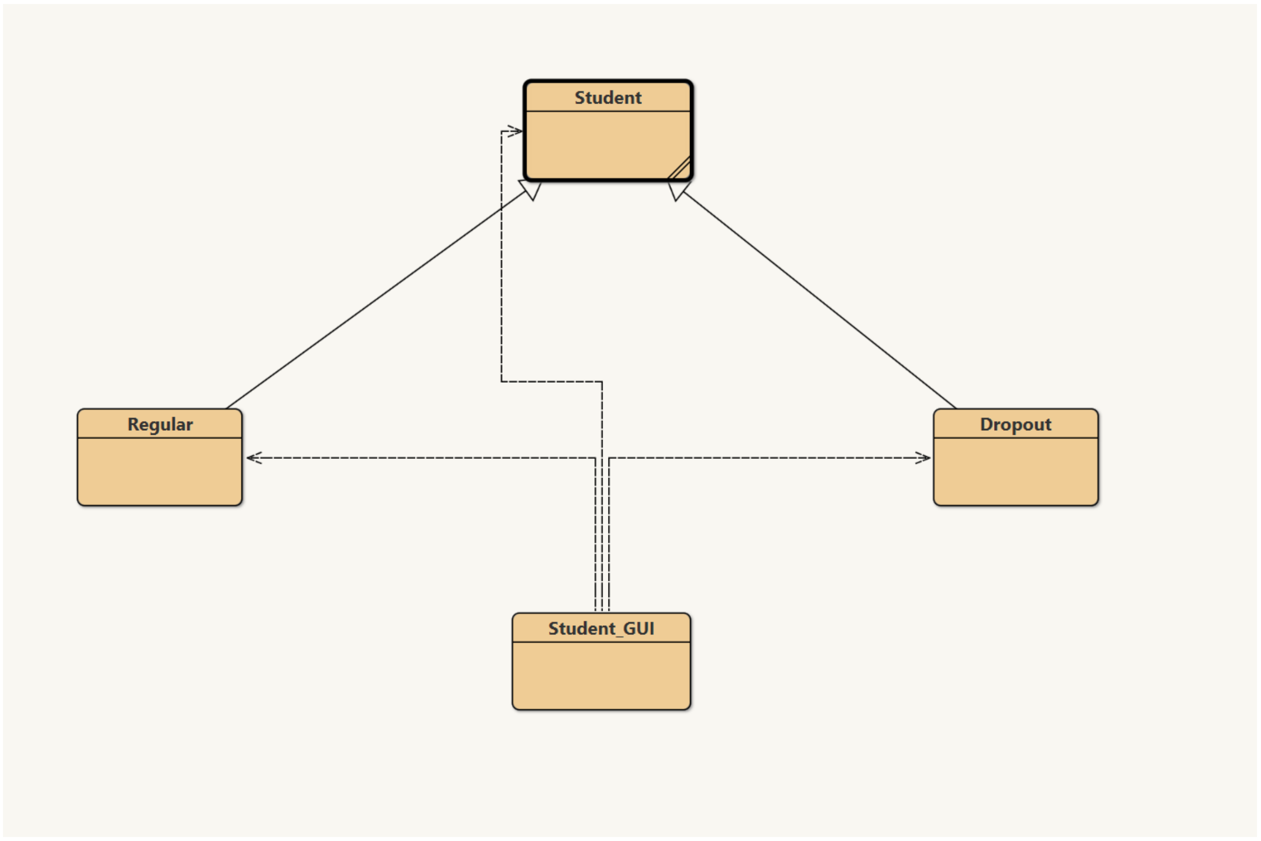
1. Introduction

Programming is the language used to instruct computers to do task. Java is a high-level programming language developed by James Gosling. It is known for its platform independence, robustness, and wide range of applications, from building desktop applications to web-based services and mobile application.

In this coursework, we are engaged to improve the functionality and user experience of our first semester project. We are assigned to develop a graphical user interfaces (GUI) using Java's flexible Swing library, the system that stores details of Students of Regular and Dropout and the store the data in ArrayList.

BlueJ was used during the development process. BlueJ is an integrated development environment (DIE) specially designed for teaching and learning Java programming, It provides an interactive and user-friendly environment that aims to make it easier for beginners, such as students, to understand and use Java programming concepts. BlueJ was developed by a team led by Michal Kolling at the University of Kent in the UK.

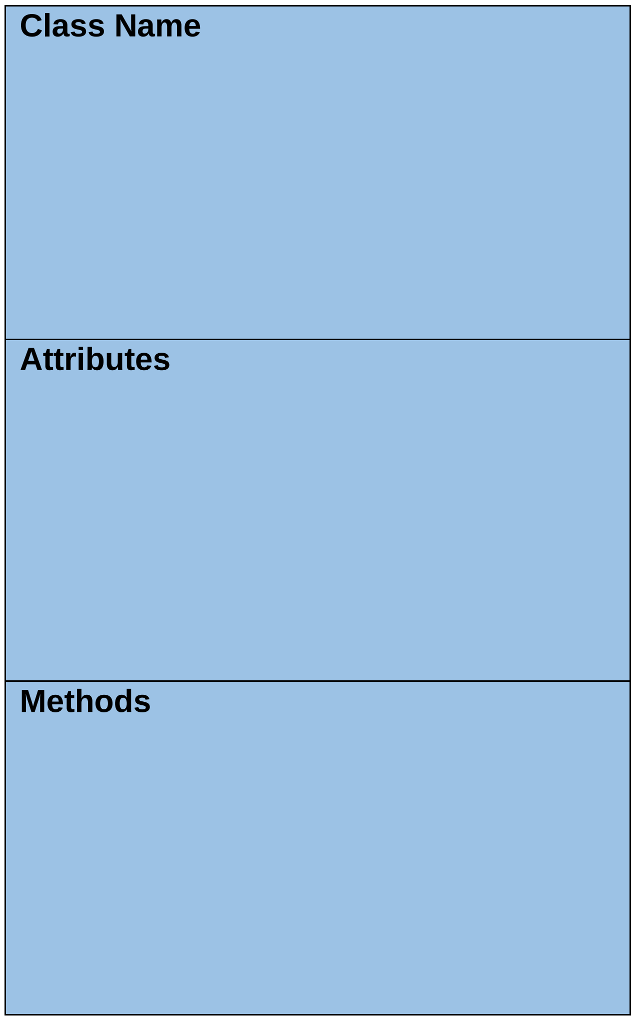
Showing Inheritance of classes:



2. Class diagram

Class diagram is the blueprints of system or the program. It is also called the building block of Object-Oriented Modelling. Class diagram provides a high-level view of the system. Class diagrams provides a visual aid for software architects, designers and developers to plan the structure of a software system. Class Diagram is made of Class Name, Attributes and Methods.

Example of Class Diagram:



a. Class diagram of Student_GUI Class

Student_GUI
- studentList : ArrayList -frame_StudentGUI: JFrame -panel_StudentGUI: JPanel -label_StudentGUI: JLabel -logolabel_StudentGUI: JLabel -btnRegular: JButton -btnDropout: JButton -image: ImageIcon -brgImage: ImageIcon -imageRegular: ImageIcon -imageDropout: ImageIcon - frame_RegularGUI: JFrame - panel_RegularGUI: JPanel - label_RegularGUI: JLabel - lblEnrollmentID: JLabel - lblStudentName: JLabel - lblTuitionFee: JLabel - lblDateofEnrollment: JLabel - lblCourseDuration: JLabel - lblDateofBirth: JLabel - lblCourseName: JLabel - lblNumOfModules: JLabel - lblNumOfCreditHours: JLabel - lblDaysPresent: JLabel - txtEnrollmentID: JTextField - txtStudentName: JTextField - txtTuitionFee: JTextField - txtNumOfModules: JTextField - txtNumOfCreditHours: JTextField - txtDaysPresent: JTextField - txtCourseDuration: JTextField - txtDateofEnrollmentDay: JComboBox - txtDateofEnrollmentMonth: JComboBox - txtDateofEnrollmentYear: JComboBox - txtDateofBirthDay: JComboBox - txtDateofBirthMonth: JComboBox - txtDateofBirthYear: JComboBox - txtCourseName JComboBox - regSubmit: JButton - reset: JButton - backRegular: JButton - btnPresentPercentage: JButton - btnGrantCertificate: JButton - frame_DropoutGUI: JFrame

- panel_DropoutGUI: JPanel
- label_DropoutGUI:
- lbldateOfBirth: JLabel
- lblcourseName: JLabel
- lblstudentName: JLabel
- lbldateOfEnrollement: JLabel
- lblenrollmentID: JLabel
- lblcourseDuration: JLabel
- lbltuitionFee: JLabel
- lblNumOfRemainingModules: JLabel
- lblNumOfMonthsAttended: JLabel
- lbldateofDropout: JLabel
- lblremainingAmount: JLabel
- txtstudentName: JTextField
- txtenrollmentId: JTextField
- txtcourseDuration: JTextField
- txttuitionFee: JTextField
- txtNumOfRemainingModules: JTextField
- txtNumOfMonthsAttended: JTextField
- txtremainingAmount: JTextField
- txtdateOfBirthDay: JComboBox
- txtdateOfBirthMonth: JComboBox
- txtdateOfBirthYear: JComboBox
- txtdateOfEnrollmentDay: JComboBox
- txtdateOfEnrollmentMonth: JComboBox
- txtdateOfEnrollmentYear: JComboBox
- txtdateOfDropoutDay: JComboBox
- txtdateOfDropoutMonth: JComboBox
- txtdateOfDropoutYear: JComboBox
- txtcourseName: JComboBox
- resetDropout: JButton
- submitDropout: JButton
- backDropout: JButton
- btnPayBills: JButton
- removeStudent: JButton
- table: JTable
- btnDisplayRegular: JButton
- btnDisplayDropout: JButton
- btnClear: JButton

+<<constructor>>Student_GUI()
+ regular_UIUX(): void
+ dropout_UIUX(): void
+ addRegular(): void
+ submitDropout(): void
+ clearRegular(): void

+ clearDropout(): void
+ presentPercentage(): void
+ btnGrandCertificate(): void
+ display(): void
+ removeDropout(): void
+ payBills(): void
+ actionPerformed(ActionEvent e): void
+ Courses(): String
+ Years(): String
+ Months(): String
+ Days(): String

Table 1 Class Diagram of Student_GUI

3. Pseudocode

Pseudocode is a text based informal language that helps programmers to develop algorithms. It allows programmers to plan algorithm structure using simple commands. Pseudocode acts as a bridge between human understanding and machine logic, providing a high-level, human-readable representation of programming algorithms and procedures. It is a visual and informal language that allows developers to define and conceptualize their code before translating it into a particular programming language. Pseudocode bypasses the strict syntax of real code, allowing programmers to focus on the logic and flow of their algorithms, making it a valuable tool for both design and communication. It's like a blueprint for a building, guiding programmers to build efficient, error-free code while promoting collaboration and clear problem-solving strategies during development.

Pseudocode of Student_GUI Class:

```
CREATE  class Student_GUI which implements ActionListener

Do

Declare studentList as arraylist

Declare JFrame frame_StudentGUI, frame_RegularGUI, frame_DropoutGUI

Declare JPanel panel_StudentGUI, panel_RegularGUI, panel_DropoutGUI

Declare JLabel label_StudentGUI, logolabel_StudentGUI, label_RegularGUI, lblEnrollmentID, lblStudentName, lblTuitionFee, lblDateofEnrollment, lblCourseDuration, lblDateofBirth, lblCourseName, lblNumOfModules, lblNumOfCreditHours, lblDaysPresent, label_DropoutGUI, lbldateOfBirth, lblcourseName, lblstudentName, lbldateOfEnrollement, lblenrollmentID, lblcourseDuration, lbltuitionFee, lblNumOfRemainingModules, lblNumOfMonthsAttended, lbldateofDropout
Declare ImageIcon image, brgImage,imageRegular,imageDropout

Declare JTextField txtEnrollmentID, txtStudentName, txtTuitionFee, txtNumOfModules, txtNumOfCreditHours, txtDaysPresent, txtCourseDuration,txtstudentName, txtenrollmentId, txtcourseDuration, txttuitionFee,txtNumOfRemainingModules, txtNumOfMonthsAttended

Declare JButton btnRegular, btnDropour, regSubmit, reset, backRegular, btnPresentPercentage, btnGrantCertificate, resetDropout, submitDropout, backDropout, btnPaybills, removeStudent, btnDisplayRegular, btnDisplayDropout,btnClear

Declare JComboBox txtDateofEnrollmentDay, txtDateofEnrollmentMonth, txtDateofEnrollmentYear, txtDateofBirthDay, txtDateofBirthMonth, txtDateofBirthYear, txtCourseName, txtdateOfBirthDay,txtdateOfBirthMonth,txtdateOfBirthYear, txtdateOfEnrollmentDay, txtdateOfEnrollmentMonth,txtdateOfEnrollmentYear, txtdateOfDropoutDay,txtdateOfDropoutMonth,txtdateOfDropoutYear,txtcourseName

Declare JTable table

Create constructor Student_GUI

DO

Initialize frame_StudentGUI as new JFrame


Initialize image with "Images/c.jpg"

Initialize brgImage with "Images/a.jpg"


Initialize panel_StudentGUI as new JPanel

Set layout of panel_StudentGUI as null

Add panel_StudentGUI to frame_StudentGUI


Initialize label_StudentGUI as new JLabel

Initialize logolabel_StudentGUI as new JLabel


Set icon of label_StudentGUI to brgImage

Set icon of logolabel_StudentGUI to image
Set bounds of label_StudentGUI as (0,0,1080,720)
Set bounds of logolabel_StudentGUI as (350,150,100,100)

Initialize btnRegular as new JButton
Initialize btnDropout as new JButton

Set bounds of btnRegular as (430,390,186,66)
Set bounds of btnDropout as (430,505,186,66)

Set font of btnRegular to "Helvetica", Font.PLAIN, 20
Set font of btnDropout to "Helvetica", Font.PLAIN, 20

Set focusable of btnRegular to false
Set focusable of btnDropout to false

Add ActionListener (this) to btnRegular
Add ActionListener (this) to btnDropout

Add btnRegular, btnDropout, label_StudentGUI, logolabel_StudentGUI to panel_StudentGUI

Set icon image of frame_StudentGUI
Set title of frame_StudentGUI to "Student's Form"
Set size of frame_StudentGUI to (1080, 720)
Set resizable of frame_StudentGUI to false
Set default close operation of frame_StudentGUI to JFrame.EXIT_ON_CLOSE
Set visibility of frame_StudentGUI to true
Set location of frame_StudentGUI to center of screen
```

END CONSTRUCTOR

Create method regular_UIUX()

Do

Initialize frame_RegularGUi as new JFrame

Initialize imageRegular as new ImageIcon

Initialize panel_RegularGUi as new JPanel

Set layout of panel_RegularGUI as null

Add panel_RegularGUI to frame_RegularGUI

Initialize label_RegularGui, lblEnrollmentID, lblStudentName, lblCourseName, lblCourseDuration, lblDateofEnrollment, lblDateofEnrollment, lblDateofBirth, lblTuitionFee, lblNumOfModules, lblNumOfCreditHours, lblDaysPresent as new JLabel

Initialize txtEnrollmentID, txtStudentName, txtTuitionFee, txtNumOfModiules, txtNumOfCreditHours, txtCourseDuration, txtDaysPresent as new JTextField

Initialize txtDateofEnrollmentDay, txtDateofEnrollmentMonth, txtDateofEnrollmentYear, txtDateofBirthDay, txtDateofBirthMonth, txtDateofBirthYear, txtCourseName as new JComboBox

Initialize regSubmit, reset, bacKRegualr, btnPresentPercentage, btnGrantCertifiacate, btnDisplayRegualr as new JButtons

Set bounds of label_RegularGUI as (400, 40, 513, 75)

Set bounds of lblEnrollmentID as (47, 150, 222, 76)

Set bounds of lblStudentName as (47, 220, 222, 75)

Set bounds of lblDateofBirth as (47, 290, 222, 75)

Set bounds of lblDaysPresent as (47, 360, 222, 75)

Set bounds of lblTuitionFee as (47, 430, 222, 75)

Set bounds of lblDateofEnrollmentas (540, 150, 289, 52)

Set bounds of lblCourseName as (540, 220, 248, 73)

Set bounds of lblCourseDuration as (540, 290, 248, 60)

Set bounds of lblNumOfModules as (540, 360, 305, 68)

Set bounds of lblNumOfCreditHours as (540, 430, 379,80)

Set bounds of txtEnrollmentID, txtStudentName, txtTuitionFee, txtNumOfModiules, txtNumOfCreditHours, txtCourseDuration, txtDaysPresent

Set bounds of txtDateofEnrollmentDay, txtDateofEnrollmentMonth, txtDateofEnrollmentYear, txtDateofBirthDay, txtDateofBirthMonth, txtDateofBirthYear, txtCourseName

Set bounds of regSubmit, reset, bacKRegualr, btnPresentPercentage, btnGrantCertifiacate, btnDisplayRegualr

Set Fonts and size of label_RegularGui, lblEnrollmentID, lblStudentName, lblCourseName, lblCourseDuration, lblDateofEnrollment, lblDateofEnrollment, lblDateofBirth, lblTuitionFee, lblNumOfModules, lblNumOfCreditHours, lblDaysPresent

Set Fonts and size of txtDateofEnrollmentDay, txtDateofEnrollmentMonth, txtDateofEnrollmentYear, txtDateofBirthDay, txtDateofBirthMonth, txtDateofBirthYear, txtCourseName

Set Fonts and size of regSubmit, reset, bacKRegualr, btnPresentPercentage, btnGrantCertifiacate, btnDisplayRegualr

Set Focusable as false of regSubmit, reset, bacKRegualr, btnPresentPercentage, btnGrantCertifiacate, btnDisplayRegualr

Add regSubmit, reset, bacKRegualr, btnPresentPercentage, btnGrantCertifiacate, btnDisplayRegualr to Action Listener as this

Add label_RegularGui, lblEnrollmentID, lblStudentName, lblCourseName, lblCourseDuration, lblDateofEnrollment, lblDateofEnrollment, lblDateofBirth, lblTuitionFee, lblNumOfModules, lblNumOfCreditHours, lblDaysPresent to panel_RegularGUI

```
Add txtDateofEnrollmentDay, txtDateofEnrollmentMonth, txtDateofEnrollmentYear, txtDateofBirthDay, txtDateofBirthMonth, txtDateofBirthYear, txtCourseName to panel_RegularGUI

Set icon imageRegular of frame_RegularGUI
Set title of frame_RegularGUI  to "Regular Form"
Set size of frame_RegularGUI to (1080, 720)
Set resizable of frame_RegularGUI to false
Set location relative to null
Set default close operation of frame_RegularGUI to JFrame.EXIT_ON_CLOSE
Set visibility of frame_RegularGUI to true
Set location of frame_RegularGUI to center of screen
END METHOD

CREATE  method dropout_UIUX()
DO
Initialize frame_DropoutGUI as new JFrame

Initialize imageDropout as new ImageIcon

Initialize panel_DropoutGUI as new JPanel
Set Layout of panel_DropoutGUI as null
Add panel_Dropout to frame_Dropout

Initialize label_DropoutGUI, lblenrollmentID, lblstudentName, lnlcourseName, lblcourseDuration, lbldateOfEnrollment, lbldateOfBirth, lbltuitionFee, lblNUmOfRemainingModules, lbldateofDropout, lblNumOfMonthsAttended as new JLabel
```

Initialize txtenrollmentId, txtstudentName, txttuitionfee, txtNumOfRemainingModules, txtcourseDuration, txtNumOfMonthsAttended as new JTextField

Initialize txtdateOfDropoutDay, txtdateOfDropoutMonth, txtdateOfDropoutYear, txtdateOfEnrollemtnDay , txtdateOfEnrollemtnMonth, txtdateOfEnrollemtnYear ,txtdateOfBirthDay, txtdateOfBirthMonth, txtdateOfBirthYear as new JComboBox

Initialize submitDropout, resetDropout, backDropout, btnPayBills, removeStudent, btnDisplayDropout as new JButtons

Set bounds of label_DropoutGUI, lblenrollmentID, lblstudentName, lnlcourseName, lblcourseDuration, lbldateOfEnrollment, lbldateOfBirth, lbltuitionFee, lblNUmOfRemainingModules, lbldateofDropout, lblNumOfMonthsAttended

Set bounds of txtenrollmentId, txtstudentName, txttuitionfee, txtNumOfRemainingModules, txtcourseDuration, txtNumOfMonthsAttended, txtdateOfDropoutDay, txtdateOfDropoutMonth, txtdateOfDropoutYear, txtdateOfEnrollemtnDay , txtdateOfEnrollemtnMonth, txtdateOfEnrollemtnYear ,txtdateOfBirthDay, txtdateOfBirthMonth, txtdateOfBirthYear

Set bounds of submitDropout, resetDropout, backDropout, btnPayBills, removeStudent, btnDisplayDropout

Set font and size of label_DropoutGUI, lblenrollmentID, lblstudentName, lnlcourseName, lblcourseDuration, lbldateOfEnrollment, lbldateOfBirth, lbltuitionFee, lblNUmOfRemainingModules, lbldateofDropout, lblNumOfMonthsAttended

Set font and size of txtenrollmentId, txtstudentName, txttuitionfee, txtNumOfRemainingModules, txtcourseDuration, txtNumOfMonthsAttended, txtdateOfDropoutDay, txtdateOfDropoutMonth, txtdateOfDropoutYear, txtdateOfEnrollemtnDay , txtdateOfEnrollemtnMonth, txtdateOfEnrollemtnYear ,txtdateOfBirthDay, txtdateOfBirthMonth, txtdateOfBirthYear

Set font and size of submitDropout, resetDropout, backDropout, btnPayBills, removeStudent, btnDisplayDropout

Set focusable as false of submitDropout, resetDropout, backDropout, btnPayBills, removeStudent, btnDisplayDropout

Add submitDropout, resetDropout, backDropout, btnPayBills, removeStudent, btnDisplayDropout to action listener as this

Add label_DropoutGUI, lblenrollmentID, lblstudentName, lnlcourseName, lblcourseDuration, lbldateOfEnrollment, lbldateOfBirth, lbltuitionFee, lblNUmOfRemainingModules, lbldateofDropout, lblNumOfMonthsAttended to panel_DropoutGUI

Add txtenrollmentId, txtstudentName, txttuitionfee, txtNumOfRemainingModules, txtcourseDuration, txtNumOfMonthsAttended, txtdateOfDropoutDay, txtdateOfDropoutMonth, txtdateOfDropoutYear, txtdateOfEnrollemtnDay , txtdateOfEnrollemtnMonth, txtdateOfEnrollemtnYear ,txtdateOfBirthDay, txtdateOfBirthMonth, txtdateOfBirthYear to panel_DropoutGUI

Add submitDropout, resetDropout, backDropout, btnPayBills, removeStudent, btnDisplayDropout to panel_DropoutGUI

Set icon imageDropout to frame_DropoutGUI

Set size of frame_DropoutGUI

Set resizeable to false of frame_DropoutGUI

Set loacal relative to null of frame_DropoutGUI

Set default close operation JFrame.EXIT_ON_CLOSE to frame_DropoutGUI

Set visible of frame_DropoutGUI to true

END METHOD

CREATE method assRegular()

Do

Assign value of Enrollment ID

```
Assign value of Student Name
Assign value of Date OF Birth
Assign value of Days Present
Assign value of TuitionFee
Assign value of Date Of Enrollment
Assign Value of Course Name
Assign value of Course Duration
Assign value of Number Of Modules
Assign value of Number Of Credit Hours
try DO
if (all the text fields are empty)
DO
show message dialog box
END DO
else
DO
Change numerical strings to integer value
If (text fields of integers are filled with other values except number)
DO
Show message dialog
END DO
else
Assign Boolean isAlreadyRegistered as false
For (Student student : studenList)
DO
If (condition checks if the current student object is an instance of the Regular class)
DO
If ( assigned enrollment id and get enrollment id is equal)
```

```
DO
Show message dialog box the student is already registered
Assign isAlreadyRegister equal to true
Break
END
DO
END
DO
END DO
if (isAlreadyRegistered is equal to false)
DO
Create object of regular class and pass the parameters.
Add the parameters to array list
Show message dialog box the student added successfully.
END DO
END DO
END DO
END DO
Catch numeric exception
DO
Show message dialogbox please enter valid numberic value.
END DO
Catch other exception
DO
Show message dialog box an error occurred.
END DO
END DO
CREATE method submitDropout()
Do
```

```
Assign value of Enrollment ID
Assign value of Student Name
Assign value of Course Duration
Assign value of Date of Enrollment
Assign Value of Course Name
Assign value of Date of Birth
Assign value of Date of Dropout
Assign value of Tuition Fee
Assign value of Number Of Remaining Modules
Assign value of Number Of Months Attended
try DO
if (all the text fields are empty)
DO
show message dialog box
END DO
else
DO
Change numerical strings to integer value
If (text fields of integers are filled with other values except number)
DO
Show message dialog
END DO
else
Assign Boolean isAlreadyRegistered as false
For (Student student : studenList)
DO
If (condition checks if the current student object is an instance of the Regular class)
DO
```

```
If ( assigned enrollment id and get enrollment id is equal)
DO
Show message dialog box the student is already registered
Assign isAlreadyRegister equal to true
Break
END DO
END DO
END DO
if (isAlreadyRegistered is equal to false)
DO
Create object of dropout class and pass the parameters.
Add the parameters to array list
Show message dialog box the student added successfully.
END DO
END DO
END DO
END DO
Catch numeric exception
DO
Show message dialogbox please enter valid numberic value.
END DO
Catch other exception
DO
Show message dialog box an error occurred.
END DO
END DO
CREATE method clearRegular()
DO
Set all the Text Field of regular_UIUX method as empty
Set all the combo box of regular_UIUX method to index 0
END METHOD
CREATE method clearDropout ()
DO
Set all the Text Field of dropout_UIUX method as empty
Set all the combo box of dropout_UIUX method to index 0
END METHOD
CREATE method presentPecentage()
DO
try DO
Assign value to daysPresent
Assign value to enrollmentID
Assign value to dateOfBirth
Assign value to courseName
Assign value to studentName
Assign value to dateOfEnrollment
Assign value to courseDuration
Assign value to tuitionFee
Assign value to numOfModules
Assign value to numOfCreditHours

Create object of regular class and pass all the parameters.
Call the presentPErcentage of regular class

 Show the presentPErcentage in message dialog box
END DO
Catch numeric error
Do
```

```
Show error message in dialog box
END DO
Catch other exception
DO
Show error message in dialog box
END DO
END METHOD
CREATE method btnGrandCertificate()
DO
try
DO
Assign enrollment ID in String from the dialog box
If ( enrollment ID is assigned)
DO
Change string enrollment id to integers
Find the regular student with the given enrollment ID
Using For (Student student as studentList)
DO
If (Student instanceof Regular)
Do
If (student.getEnrollmentID() equals to enrollmentID)
DO
regualrStudent is equal to regular
break
END DO
END DO
END DO
if (regularStudent is not null)
DO
```

```
Get value of course name
Get value of date of enrollment
Get days present
Get present percentage
If present percentage is equals to A or more than 80%
DO
Show the message box that the student is granted with scholarship.
END DO
else
DO
Show message dialog box that scholarship is not granted
END DO
END DO
else
DO
Show message dialog box that no regular student found
END DO
else DO
show message dialog box that to show valid enrollment id
END DO
END DO
Catch numeric exception
DO
Show message dialog box enter valid numeric number
END DO
Catch other exception
DO
Show message dialog box to enter valid values
END METHOD
```

```
**CREATE** method display()
**DO**
Initialize model as new Default Model
Set column identifiers for model
For ( Student student asstudentList)
**DO**
Get all the values
**END DO**
Initialize table as new JTable
Set table to auto resizeable mode
Set fills viewport height to true

Initialize panelDisplay  as new new JPanel
add scroll and border layout

Initialize btnClear as new JButton
Add action listener to btnClear

Initialize panelButtons as new JPanel
Set layout for panelButtons
Add button to panelButtons

Add action listener to btnClear
**DO**
Action performed **DO**
Make the array list clear
Show message dialog box list cleared.
```

```
END DO
END METHOD
CREATE removeDropout()
DO
Input enrollment id from text field
Try
Do
If(enrollment id is not null)
DO
Changing the enrollment id to integers
Find the dropout student with the given enrollmentID
END DO
If (dropoutStudent is not null)
DO
Remove dropout from student list
Show message dialog
END DO
Else
DO
Show message dialog no dropout student found
END DO
END DO
else DO
show message dialog box enter the valid numeric value
END DO
END DO catch numeric exception
DO
Show message dialog box please enter valid numeric value
```

```
END DO
Catch other exception
DO
Show message dialog box an error occurred.
END METHOD

CREATE payBills()
DO
Try
DO
Input enrollment ID from textfield
If ( enrollment id is null or is empty)
DO
Show message dialog box that enrollment id cannot be empty
END DO
Change enrollment id to integers
Check the dropout student with the given enrollment id
If (dropoutStudent is not equal to null)
DO
Input tuition fee from the dialog question box
If ( tuition fee is equal to null or is empty)
DO
Show message dialog box tuition fee cannot be empty
END DO
Ask for amountPaid
if(amount paid is equal to ;null or is empty)
DO
Show message dialog box amount paid cannot be empty.
```

```
END DO
Calling amount paid from dropout student
If(isPaid)
DO
Show message dialog box the bills paid successfully
END DO
Else
DO
Show message dialog box bills are not paid fully
END DO
Catch numeric exception
Catch other exception
END METHOD
CREATE method actionPerformed(ActionEvent e)
DO
If (get source btn regular)
DO
Call regular_UIUX()
END DO
else if (get source btnDropout)
Do
Dropout_UIUX()
END DO
else if (get source regStudent)
Do
addRegular()
END DO
else if (get source reset)
```

```
Do
clearRegular()
END DO
else if (get source backRegular)
Do
Set Frame_RegularGUI visible
END DO
else if (get source submitDropout)
Do
submitDropout()
END DO
else if (get source resetDropout)
Do
clearDropout()
END DO
else if (get source backDropout)
Do
Frame_StudentGUI set visible
END DO
else if (get source btnDisplayRegular)
Do
Display()
END DO
else if (get source btnDisplayDropout)
Do
Display()
END DO
else if (get sourceremoveStudent)
```

```
Do
removeDropout()
END DO
else if (get source btnPayBills)
Do
payBills()
END DO
else if (get source btnPresentPercentage)
Do
presentPercentage()
END DO
else if (get source btnGrantCertificate)
Do
btnGrantCertificate()
END DO
END METHOD
CREATE string courses
DO
Add names of courses in string
END DO
CREATE string years
DO
add years in string
END DO

CREATE string months
DO
Add months in string
END DO
```

```
CREATE string days
DO
Add days in string
END DO

CREATE main method
DO
New Student_GUI()
END DO
END CLASS
```

4. Method description

4.1. Method description Student_GUI Class

- regular_UIUX(): This method is used for the frame or GUI of Regular Student form.
- dropout_UIUX(): This method is used for the frame or GUI of Dropout Student form.
- addRegular (): This method is used for getting the values entered from Regular Student form. This method also checks if all the text fields are used and there are not any other exceptional cases. After that it checks if the student with same enrollment Id is already registered. If not then the values are passed to Regular Class and added to the array List (studentList).
- submitDropout (): This method is used for getting the values entered from Dropout Student form. This method also checks if all the text fields are used and there are not any other exceptional cases. After that it checks if the student with same enrollment Id is already registered. If not then the values are passed to Dropout Class and added to the array List (studentList).
- clearRegular(): This method is used to set text fields of regular_UIUX empty or to clear all the text field in the GUI of Regular Student Form.
- clearDropout():This method is used to set text fields of dropout_UIUX empty or to clear all the text field in the GUI of Dropout Student Form.
- presentPercentage(): The values are processed by this method from a text field, and will then be returned to the Regular Class parameter. The present percentage method of the Regular Class will be called and results displayed in the dialog box. The values are processed by this method from a text field, and will then be returned

to the Regular Class parameter. The present percentage method of the Regular Class will be called and results displayed in the dialog box.

- btnGrandCertificate(): The enrollment ID are processed by this method from a Question dialog box and if the enrollment Id is found, it calculates the present percentage. At last, if the given student of given enrollment Id has 80% or above 80% the Scholarship granted Dialog Box is shown.
- removeDropout():The enrollment ID are processed by this method from a Question dialog box and if the enrollment Id is found, it clears the values of that enrollment Id from the array list. At last, if the given enrollment Id is not found then Error Dialog Box is shown.
- payBills(): This method processes the value of enrollment Id if the enrollment ID is found then Tuition Fee is asked then Amount To pay. The parameter is passed to Dropout Class and billsPayable from Dropout Class is called.
- display() : This method prints the values of the regular or dropout student from the array list in the form of table.
- actionPerformed(ActionEvent e): When the buttons are pressed or clicked this method processes the buttons.

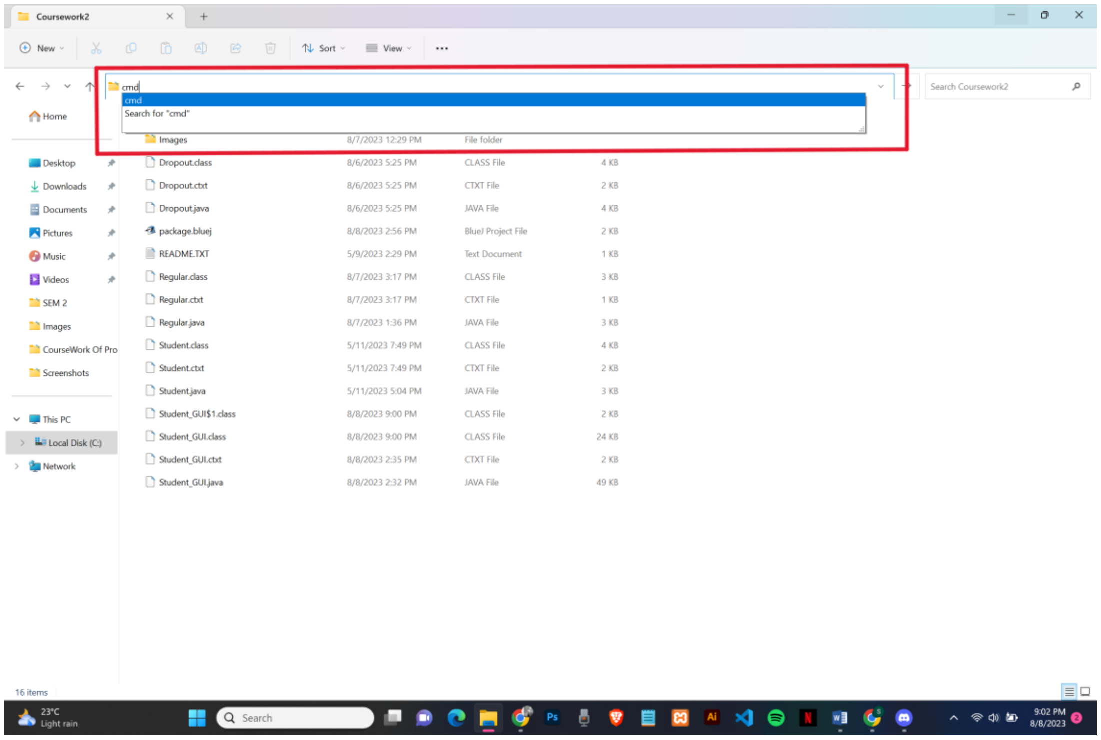
5. Testing

5.1. Test Case 1

Objective:	Test that the program can be compiled and run using the command prompt, including a screenshot like Figure 1 from the command prompt learning aid.
Action:	Command prompt was opened with the java file location. Then checked if the cmd prompt was in the right location using command:dir , after that compiled the java file using command: javac filename. After compiling the file run the code using command : java filename.
Expected Result:	The java code file should compile and run perfectly from command prompt.
Actual Result:	he java code file was compile and run perfectly from command prompt.
Conclusion:	The test was successful.

Table 2 Test 1 Running from command prompt

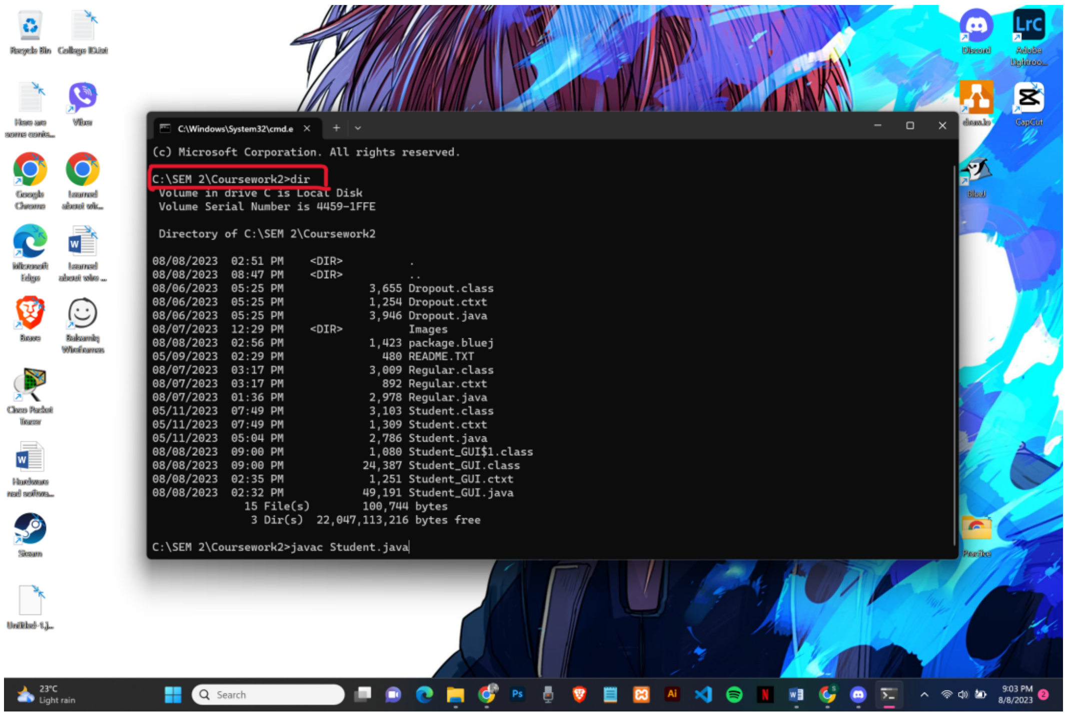
Opening the location in command prompt



```
cmd
cmd
Search for "cmd"

Images                8/7/2023 12:29 PM   File folder
Dropout.class         8/6/2023 5:25 PM    CLASS File          4 KB
Dropout.ctxt          8/6/2023 5:25 PM    CTXT File           2 KB
Dropout.java          8/6/2023 5:25 PM    JAVA File           4 KB
package.bluej         8/8/2023 2:56 PM    BlueJ Project File  2 KB
README.TXT            5/9/2023 2:29 PM    Text Document       1 KB
Regular.class         8/7/2023 3:17 PM    CLASS File          3 KB
Regular.ctxt          8/7/2023 3:17 PM    CTXT File           1 KB
Regular.java          8/7/2023 1:36 PM    JAVA File           3 KB
Student.class         5/11/2023 7:49 PM   CLASS File          4 KB
Student.ctxt          5/11/2023 7:49 PM   CTXT File           2 KB
Student.java          5/11/2023 5:04 PM   JAVA File           3 KB
Student_GUI$1.class   8/8/2023 9:00 PM    CLASS File          2 KB
Student_GUI.class     8/8/2023 9:00 PM    CLASS File         24 KB
Student_GUI.ctxt      8/8/2023 2:35 PM    CTXT File           2 KB
Student_GUI.java      8/8/2023 2:32 PM    JAVA File          49 KB

16 items
```

Figure 1Opening the location in command prompt

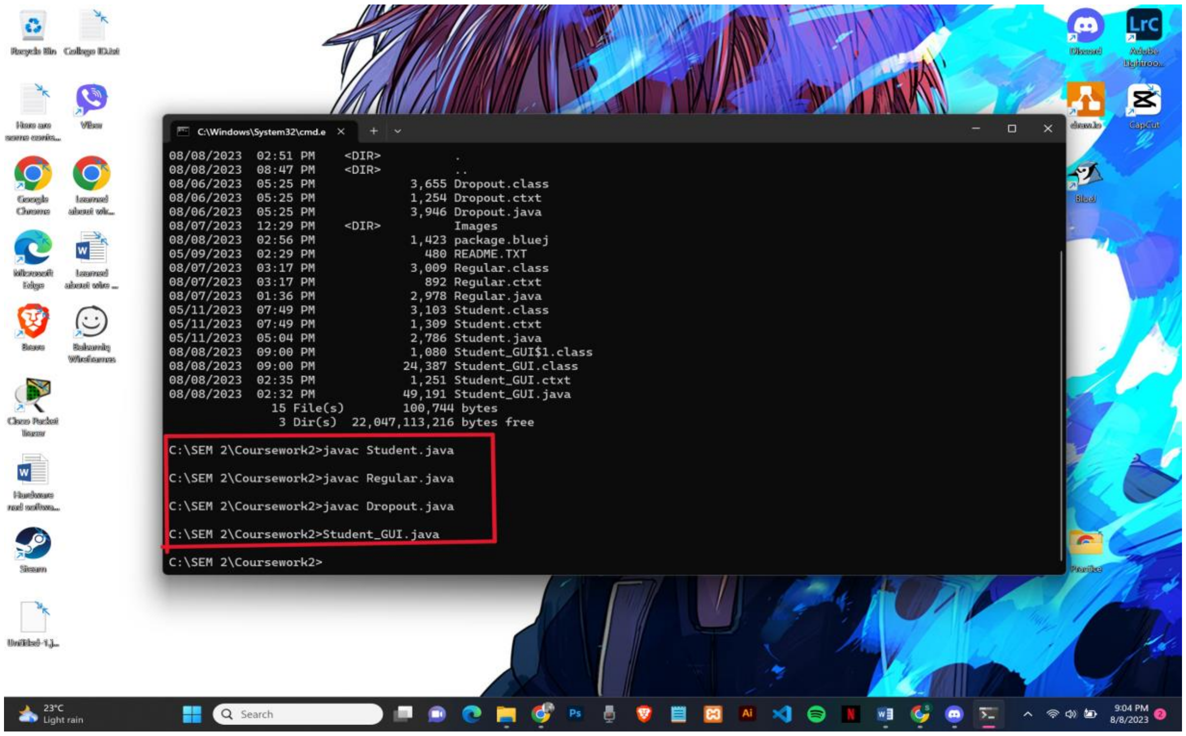
Checking if command prompt was in right location using dir command



```
(c) Microsoft Corporation. All rights reserved.

C:\SEM 2\Coursework2>dir
 Volume in drive C is Local Disk
 Volume Serial Number is 4459-1FFE

 Directory of C:\SEM 2\Coursework2

08/08/2023  02:51 PM    <DIR>          .
08/08/2023  08:47 PM    <DIR>          ..
08/06/2023  05:25 PM             3,655 Dropout.class
08/06/2023  05:25 PM             1,254 Dropout.ctxt
08/06/2023  05:25 PM             3,946 Dropout.java
08/07/2023  12:29 PM    <DIR>          Images
08/08/2023  02:56 PM             1,423 package.bluej
05/09/2023  02:29 PM               480 README.TXT
08/07/2023  03:17 PM             3,009 Regular.class
08/07/2023  03:17 PM               892 Regular.ctxt
08/07/2023  01:36 PM             2,978 Regular.java
05/11/2023  07:49 PM             3,103 Student.class
05/11/2023  07:49 PM             1,309 Student.ctxt
05/11/2023  05:04 PM             2,786 Student.java
08/08/2023  09:00 PM             1,080 Student_GUI$1.class
08/08/2023  09:00 PM            24,387 Student_GUI.class
08/08/2023  02:35 PM             1,251 Student_GUI.ctxt
08/08/2023  02:32 PM            49,191 Student_GUI.java
              15 File(s)        100,744 bytes
               3 Dir(s)  22,047,113,216 bytes free

C:\SEM 2\Coursework2>javac Student.java
```

Figure 2Checking if command prompt was in right location using dir command

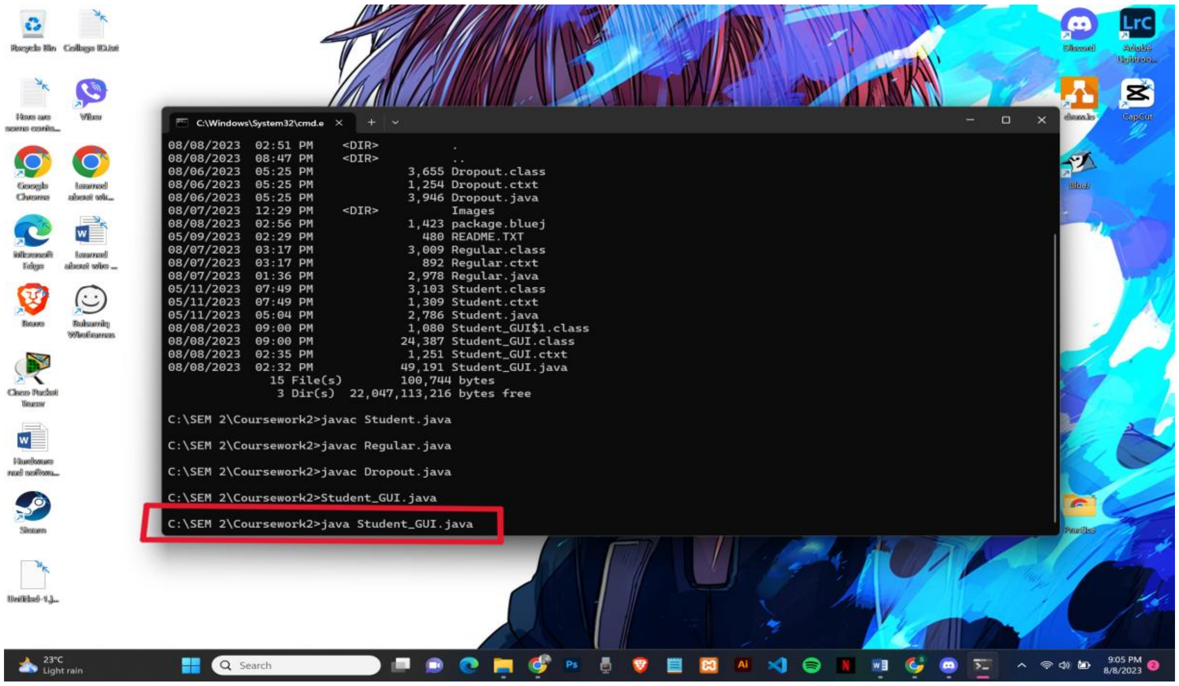
Compiling the java file using javac command



```
08/08/2023  02:51 PM    <DIR>          .
08/08/2023  08:47 PM    <DIR>          ..
08/06/2023  05:25 PM             3,655 Dropout.class
08/06/2023  05:25 PM             1,254 Dropout.ctxt
08/06/2023  05:25 PM             3,946 Dropout.java
08/07/2023  12:29 PM    <DIR>          Images
08/08/2023  02:56 PM             1,423 package.bluej
05/09/2023  02:29 PM               480 README.TXT
08/07/2023  03:17 PM             3,009 Regular.class
08/07/2023  03:17 PM               892 Regular.ctxt
08/07/2023  01:36 PM             2,978 Regular.java
05/11/2023  07:49 PM             3,103 Student.class
05/11/2023  07:49 PM             1,309 Student.ctxt
05/11/2023  05:04 PM             2,786 Student.java
08/08/2023  09:00 PM             1,080 Student_GUI$1.class
08/08/2023  09:00 PM            24,387 Student_GUI.class
08/08/2023  02:35 PM             1,251 Student_GUI.ctxt
08/08/2023  02:32 PM            49,191 Student_GUI.java
              15 File(s)        100,744 bytes
               3 Dir(s)  22,047,113,216 bytes free

C:\SEM 2\Coursework2>javac Student.java

C:\SEM 2\Coursework2>javac Regular.java

C:\SEM 2\Coursework2>javac Dropout.java

C:\SEM 2\Coursework2>Student_GUI.java

C:\SEM 2\Coursework2>
```

Figure 3 Compiling the java file using javac command

Running the code



```
08/08/2023  02:51 PM    <DIR>          .
08/08/2023  08:47 PM    <DIR>          ..
08/06/2023  05:25 PM             3,655 Dropout.class
08/06/2023  05:25 PM             1,254 Dropout.ctxt
08/06/2023  05:25 PM             3,946 Dropout.java
08/07/2023  12:29 PM    <DIR>          Images
08/08/2023  02:56 PM             1,423 package.bluej
05/09/2023  02:29 PM               480 README.TXT
08/07/2023  03:17 PM             3,009 Regular.class
08/07/2023  03:17 PM               892 Regular.ctxt
08/07/2023  01:36 PM             2,978 Regular.java
05/11/2023  07:49 PM             3,103 Student.class
05/11/2023  07:49 PM             1,309 Student.ctxt
05/11/2023  05:04 PM             2,786 Student.java
08/08/2023  09:00 PM             1,080 Student_GUI$1.class
08/08/2023  09:00 PM            24,387 Student_GUI.class
08/08/2023  02:35 PM             1,251 Student_GUI.ctxt
08/08/2023  02:32 PM            49,191 Student_GUI.java
              15 File(s)        100,744 bytes
               3 Dir(s)  22,047,113,216 bytes free

C:\SEM 2\Coursework2>javac Student.java

C:\SEM 2\Coursework2>javac Regular.java

C:\SEM 2\Coursework2>javac Dropout.java

C:\SEM 2\Coursework2>Student_GUI.java

C:\SEM 2\Coursework2>java Student_GUI.java
```

Figure 4 Running the code

Code was successfully run and compiled

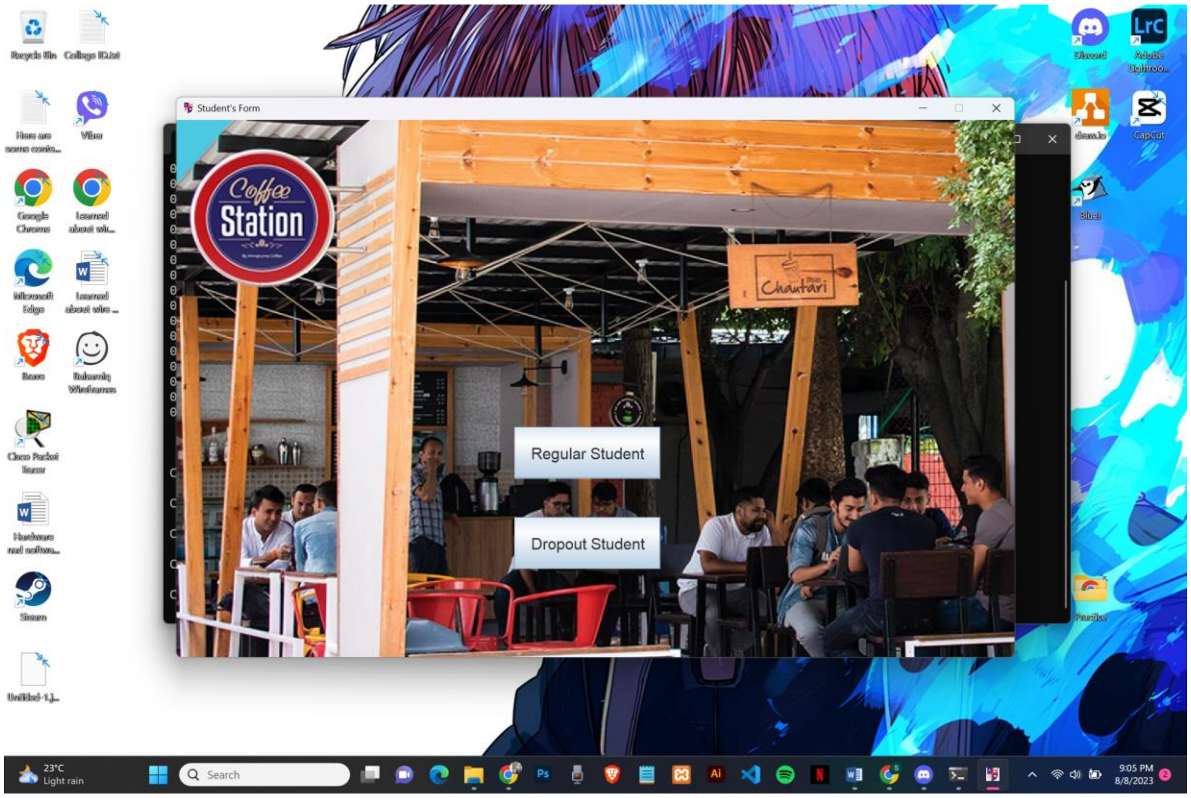


Figure 5 Code was successfully run and compiled

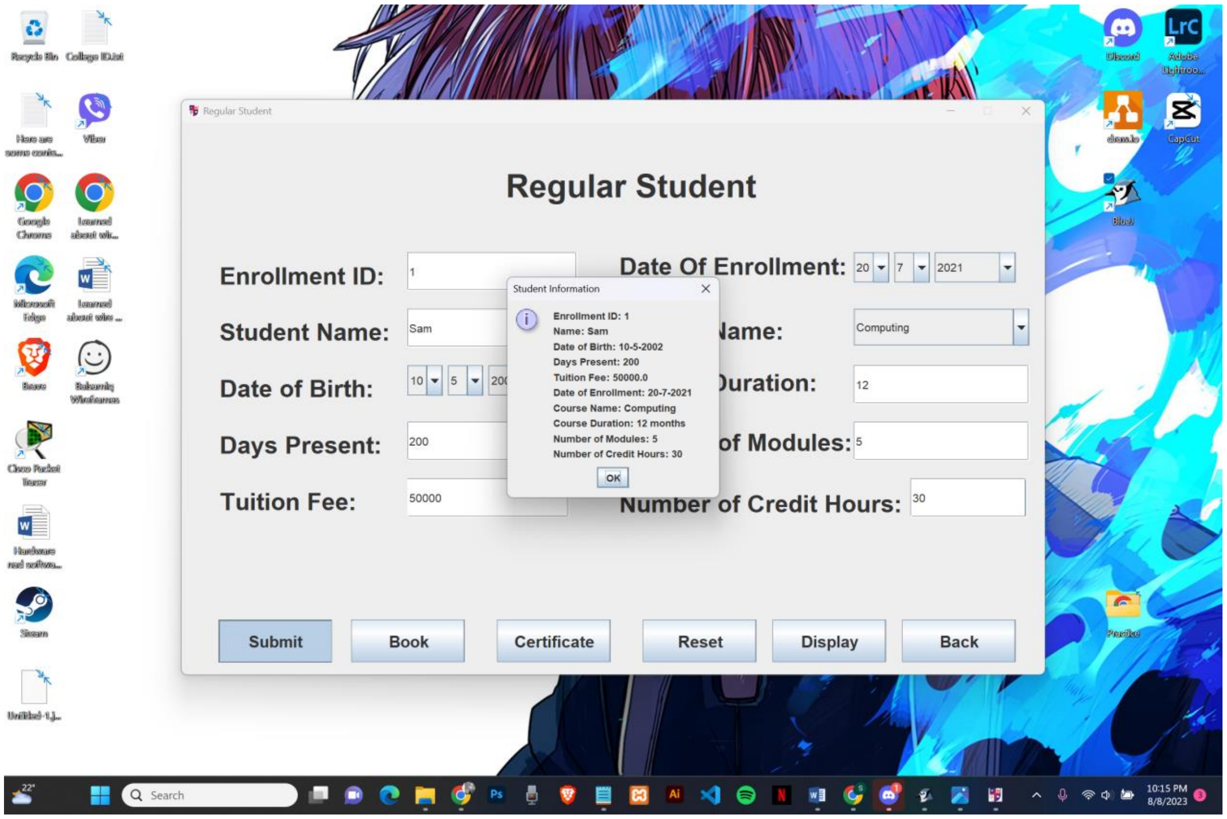
5.2. Test Case 2

a. Add a Regular Student

Objective:	a. Add a Regular Student
Action:	All the text field for Regular Student was filled, after that Submit Button was clicked and data was stored in array successfully.
Expected Result:	Regular Student should be added in Regular class and array list when add button clicked.
Actual Result:	Regular Student was be added in Regular class and array list when add button clicked.
Conclusion:	The test was successful.

Table 3 Test 2 Add a Regular Student

Screenshot:



Regular Student

Enrollment ID: 1
Student Name: Sam
Date of Birth: 10 5 200
Days Present: 200
Tuition Fee: 50000
Date Of Enrollment: 20 7 2021
Name: Computing
Duration: 12
of Modules: 5
Number of Credit Hours: 30

Student Information

Enrollment ID: 1
Name: Sam
Date of Birth: 10-5-2002
Days Present: 200
Tuition Fee: 50000.0
Date of Enrollment: 20-7-2021
Course Name: Computing
Course Duration: 12 months
Number of Modules: 5
Number of Credit Hours: 30

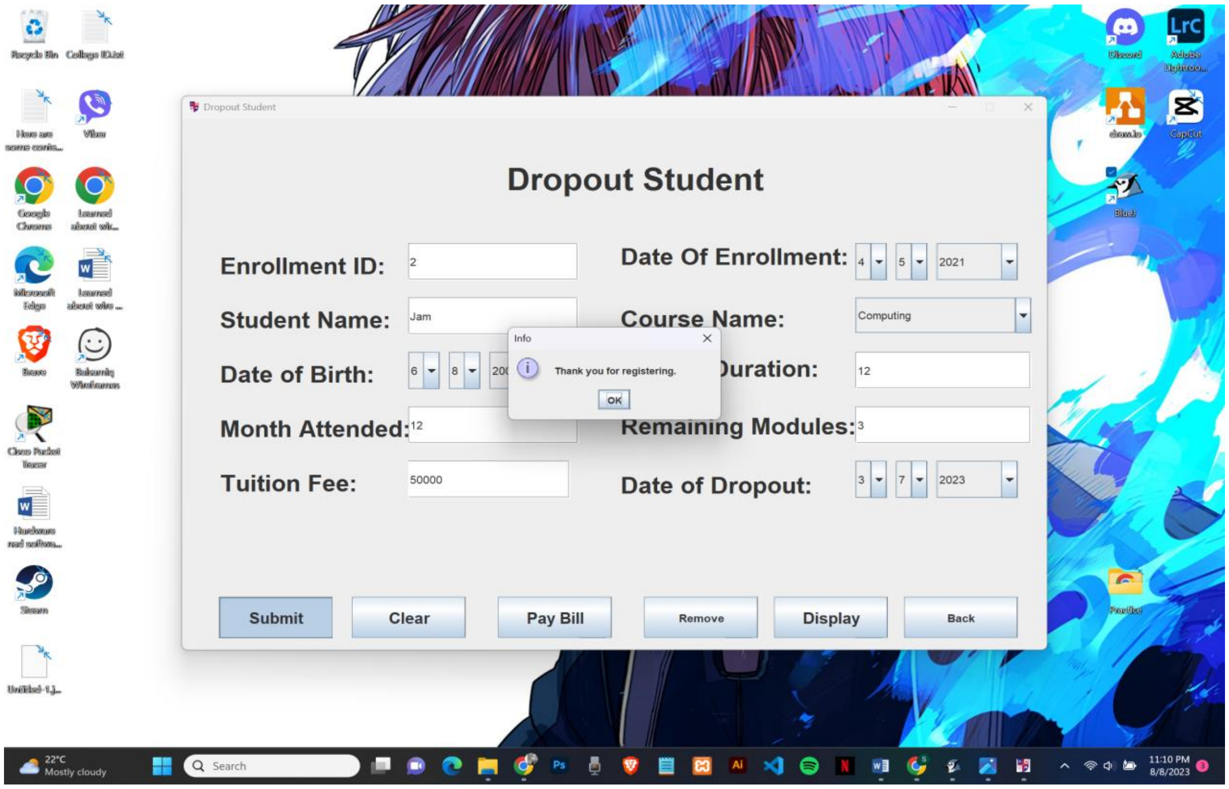
Figure 6 Add a Regular Student

b. Add Dropout Student

Objective:	Add a Dropout Student
Action:	All the text field for Dropout Student was filled, after that Submit Button was clicked and data was stored in array successfully.
Expected Result:	Dropout Student should be added in Dropout Class and array list when add button clicked.
Actual Result:	Dropout Student was added in array list when add button clicked.
Conclusion:	The test was successful.

Table 4 Test 2 Add Dropout Student

Screenshot:


Dropout Student

Enrollment ID: 2

Student Name: Jam

Date of Birth: 6 8 200

Month Attended: 12

Tuition Fee: 50000

Date Of Enrollment: 4 5 2021

Course Name: Computing

Duration: 12

Remaining Modules: 3

Date of Dropout: 3 7 2023

Info

Thank you for registering.

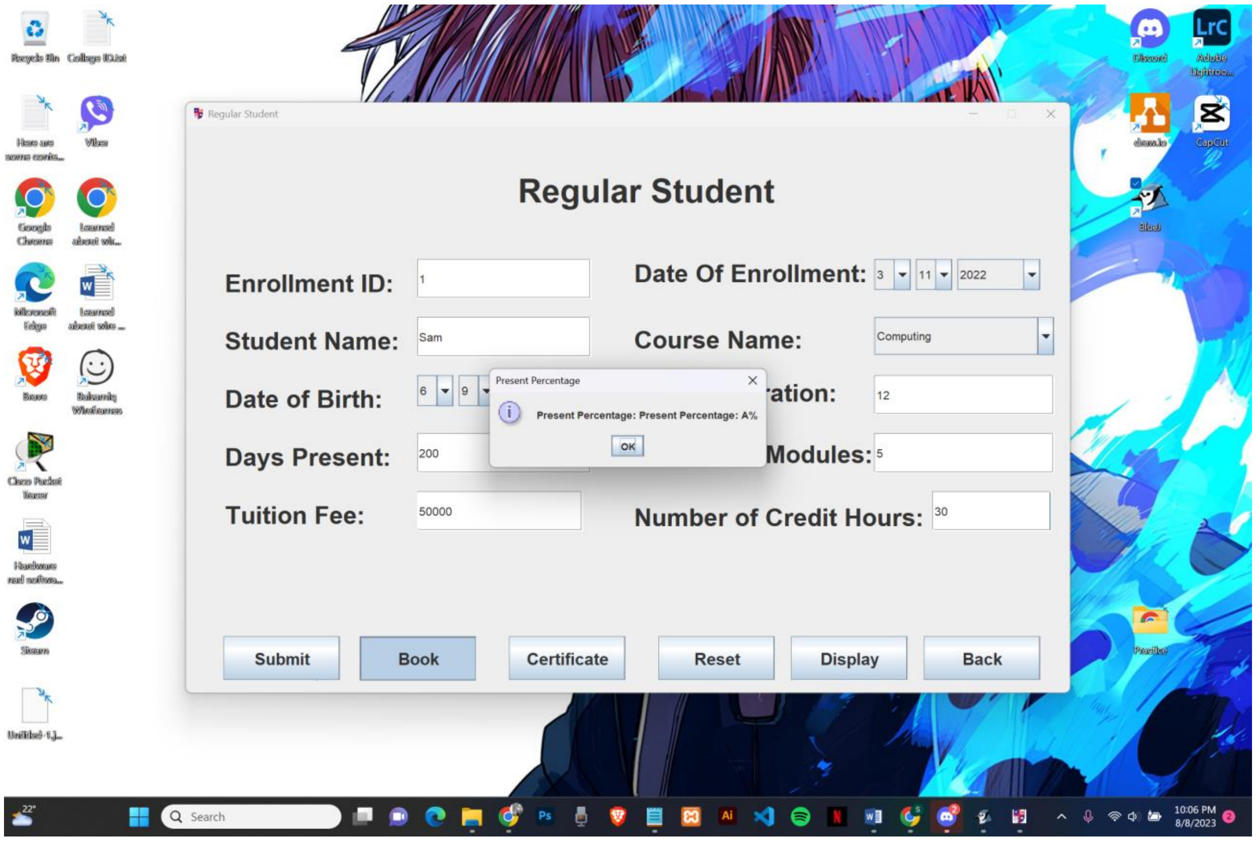
Figure 7 Add Dropout Student

c. Calculate Present Percentage of Regular Class

Objective:	Calculate Present Percentage of Regular Student
Action:	When grant certificate button is clicked information box should be appeared in dialog box.
Expected Result:	Student's result should display in dialog box.
Actual Result:	Student's result was display in dialog box.
Conclusion:	The test was successful.

Table 5 Test 2 Calculate Present Percentage of Regular Class

Screenshot:



Regular Student

Enrollment ID: 1

Date Of Enrollment: 3 11 2022

Student Name: Sam

Course Name: Computing

Date of Birth: 6 9

Duration: 12

Days Present: 200

Number of Modules: 5

Tuition Fee: 50000

Number of Credit Hours: 30

Present Percentage

Present Percentage: Present Percentage: A%

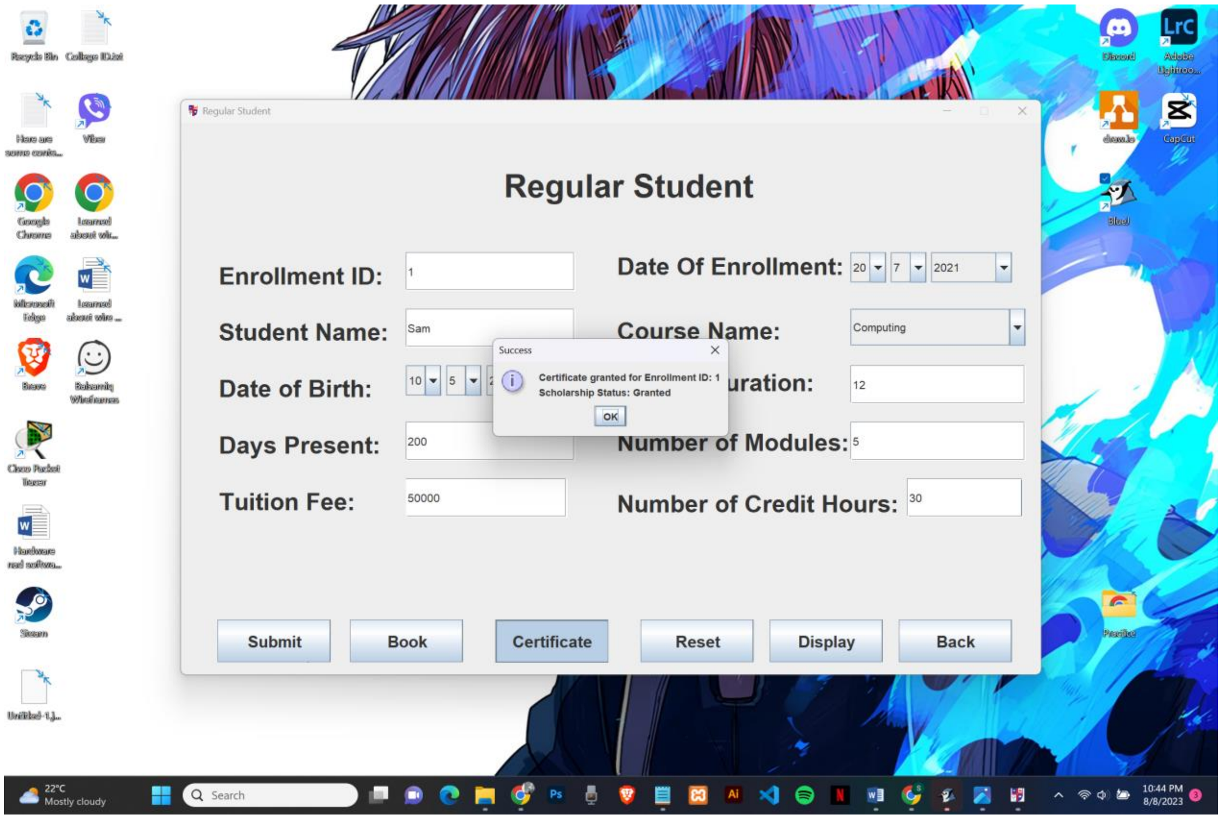
Figure 8 Calculate Present Percentage of Regular Class

d. Grant Certificate of Regular Student

Objective:	Grant Certificate of Regular Student
Action:	When grant certificate button is clicked information box should be appeared in dialog box.
Expected Result:	Student with Present Percentage 80% or A should be granted with Certificate.
Actual Result:	Student with Present Percentage 80% or A was be granted with Certificate.
Conclusion:	The test was successful.

Table 6 Test 2 Grant Certificate of Regular Student

Screenshot:


Regular Student

Enrollment ID: 1

Student Name: Sam

Date of Birth: 10 5

Days Present: 200

Tuition Fee: 50000

Date Of Enrollment: 20 7 2021

Course Name: Computing

uration: 12

Number of Modules: 5

Number of Credit Hours: 30

Success

Certificate granted for Enrollment ID: 1
Scholarship Status: Granted

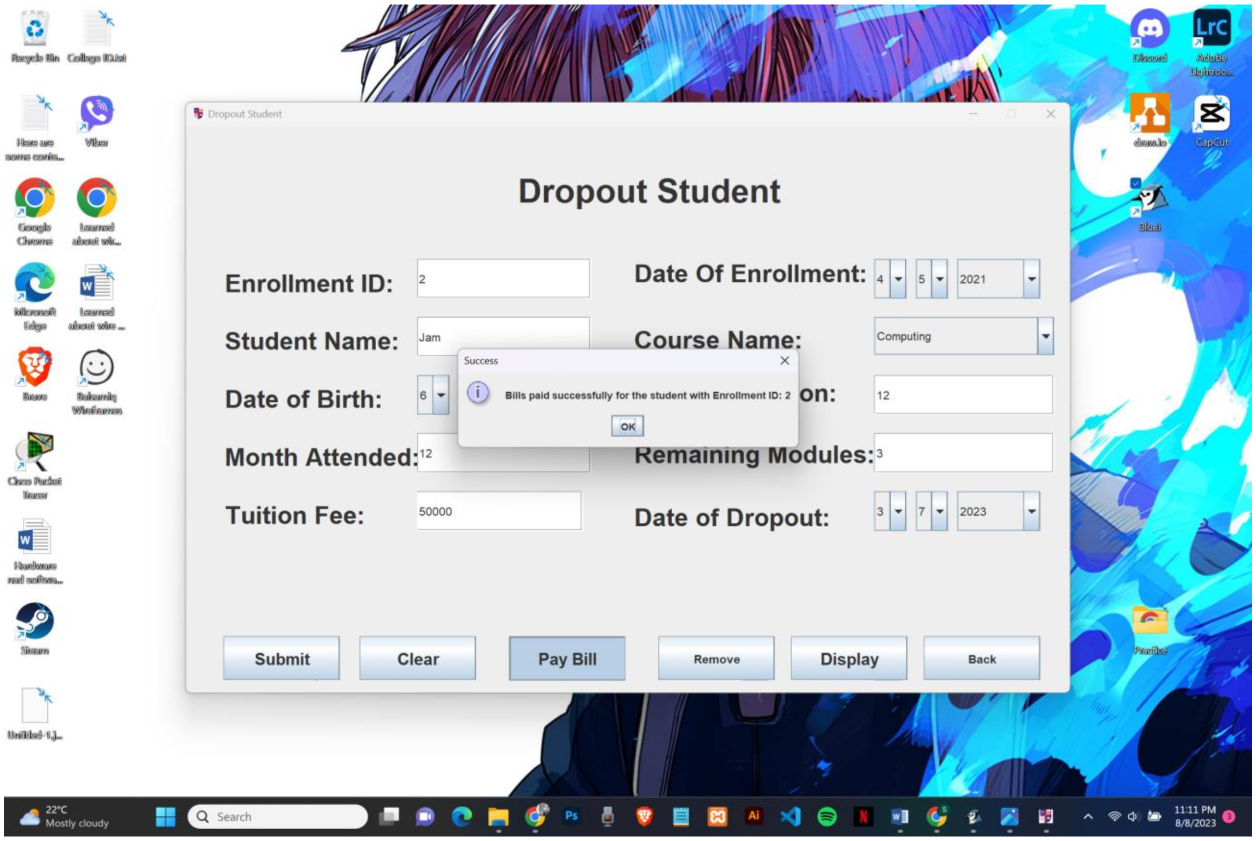
Figure 9 Grant Certificate of Regular Student

e. Pay Bills of Dropout Student

Objective:	Pay the bills of Dropout Student
Action:	When pay bills button is clicked Fee was asked after that amount to pay.
Expected Result:	When Fee and amount paid equal, then has paid message should show.
Actual Result:	When Fee and amount paid equal the has paid message was show.
Conclusion:	The test was successful.

Table 7 test 2 Pay Bills of Dropout Student

Screenshot:



Enrollment ID: 2
Student Name: Jam
Date of Birth: 6
Month Attended: 12
Tuition Fee: 50000
Date Of Enrollment: 4 5 2021
Course Name: Computing
12
Remaining Modules: 3
Date of Dropout: 3 7 2023

Bills paid successfully for the student with Enrollment ID: 2

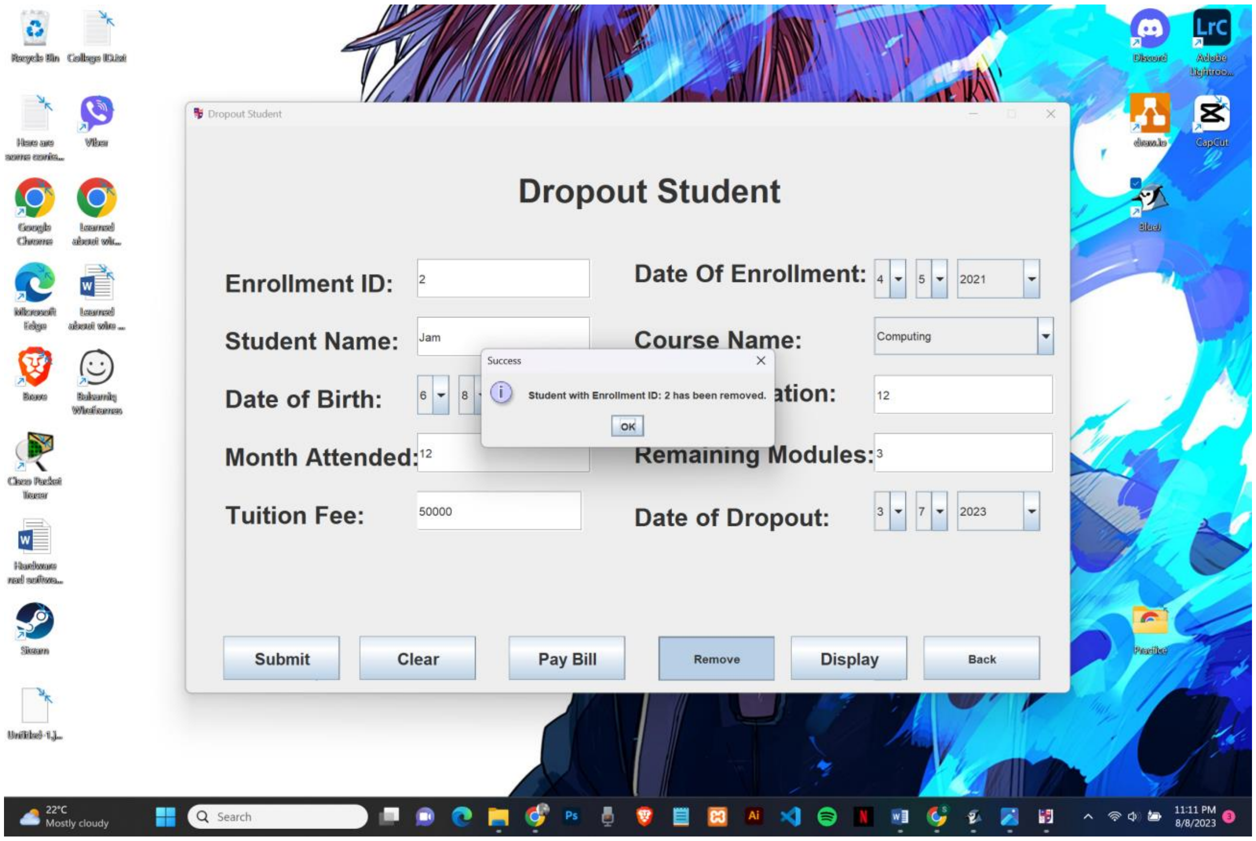
Figure 10 Pay Bills of Dropout Student

f. Remove Dropout Student

Objective:	Remove Dropout Student
Action:	When remove button is clicked the information of that student is removed from the array list, information box should be appeared in dialog box.
Expected Result:	The student's information should be removed from the array list.
Actual Result:	The student's information was removed from the array list.
Conclusion:	The test was successful.

Table 8 Test 2 Remove Dropout Student

Screenshots:



Dropout Student

Enrollment ID: 2

Student Name: Jam

Date of Birth: 6 8

Month Attended: 12

Tuition Fee: 50000

Date Of Enrollment: 4 5 2021

Course Name: Computing

ation: 12

Remaining Modules: 3

Date of Dropout: 3 7 2023

Success

Student with Enrollment ID: 2 has been removed.

Figure 11 Remove Dropout Student

5.3. Test Case 3

Testing the errors detection when unsuitable values are entered.

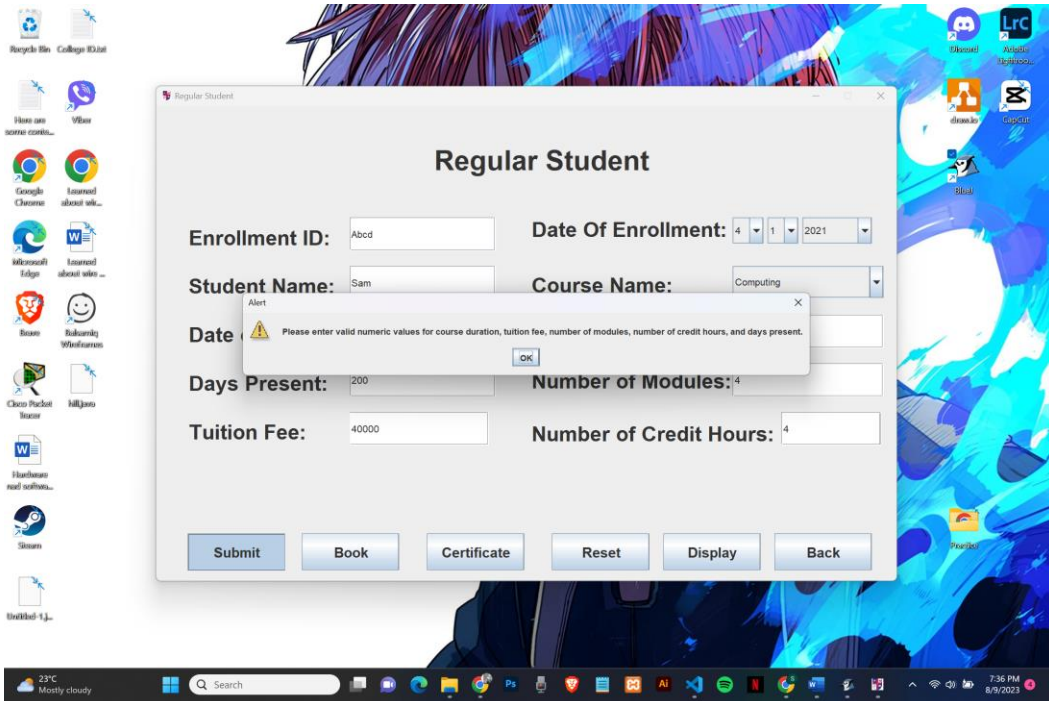
a) Regular Student Form

Objective:	Error detection when unsuitable values are entered.
Action:	a. Alphabet was entered in the text field of Enrollment ID. b. Numeric values were entered in alphabet text field. c. All the text field was kept empty.
Expected Result:	Dialog message box should appear with error message.
Actual Result:	Dialog message box was appeared with error message.
Conclusion:	The test was successful.

Table 9 Regular Form Error Detection

Screenshots:

When Enrollment Id was filled with String.



Regular Student

Enrollment ID: Abcd

Date Of Enrollment: 4 1 2021

Student Name: Sam

Course Name: Computing

Date

Days Present: 200

Number of Modules: 4

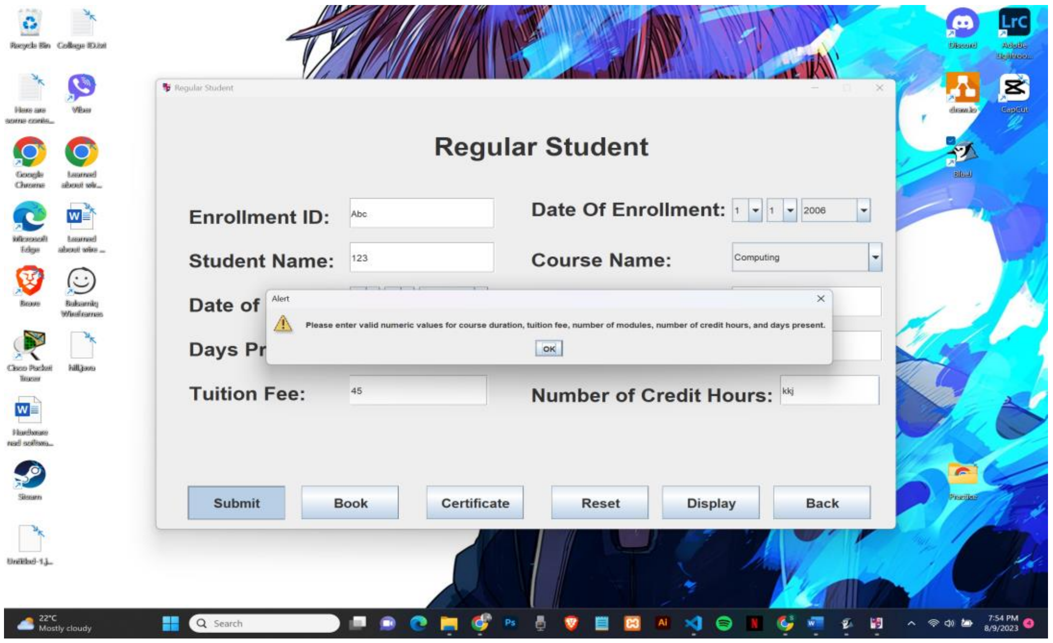
Tuition Fee: 40000

Number of Credit Hours: 4

Please enter valid numeric values for course duration, tuition fee, number of modules, number of credit hours, and days present.

Figure 12 When Enrollment Id was filled with String.

When Student Name was filled with integer.



Regular Student

Enrollment ID: Abc

Date Of Enrollment: 1 1 2006

Student Name: 123

Course Name: Computing

Date of

Days Pr

Tuition Fee: 45

Number of Credit Hours: kkj

Please enter valid numeric values for course duration, tuition fee, number of modules, number of credit hours, and days present.

Figure 13 When Student Name was filled with integer.

When all the fields were kept empty.

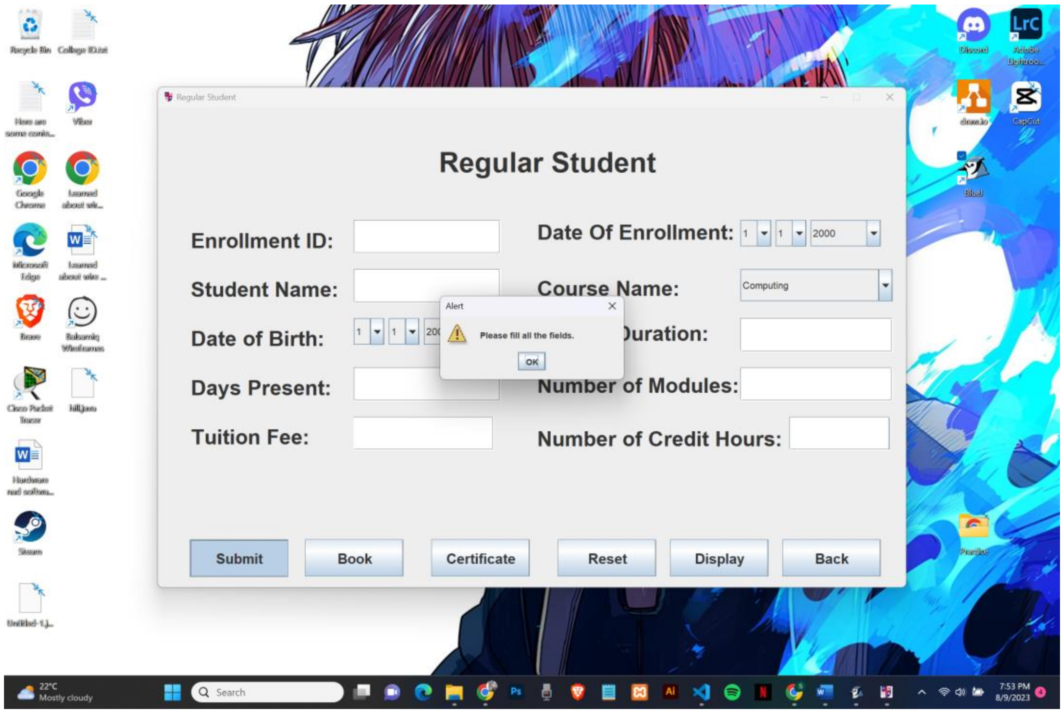


Figure 14 When all the fields were kept empty.

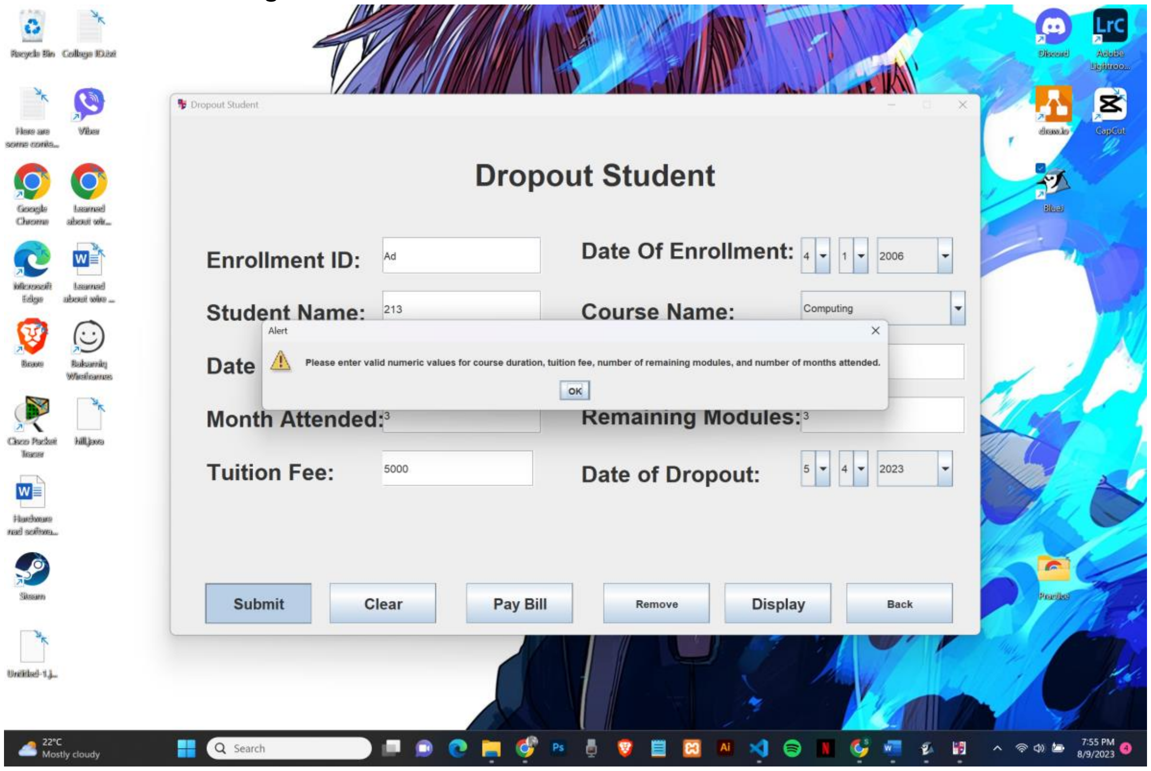
b) Dropout Student Form

Objective:	Error detection when unsuitable enrolment Id are entered.
Action:	Alphabet was entered in the text field of Enrollment ID.
Expected Result:	Dialog message box should appear with error message.
Actual Result:	Dialog message box was appear with error message.
Conclusion:	The test was successful.

Figure 15 Error Detection Of Dropout Student Form

Screenshot:

When Integer field were field with String.



Dropout Student

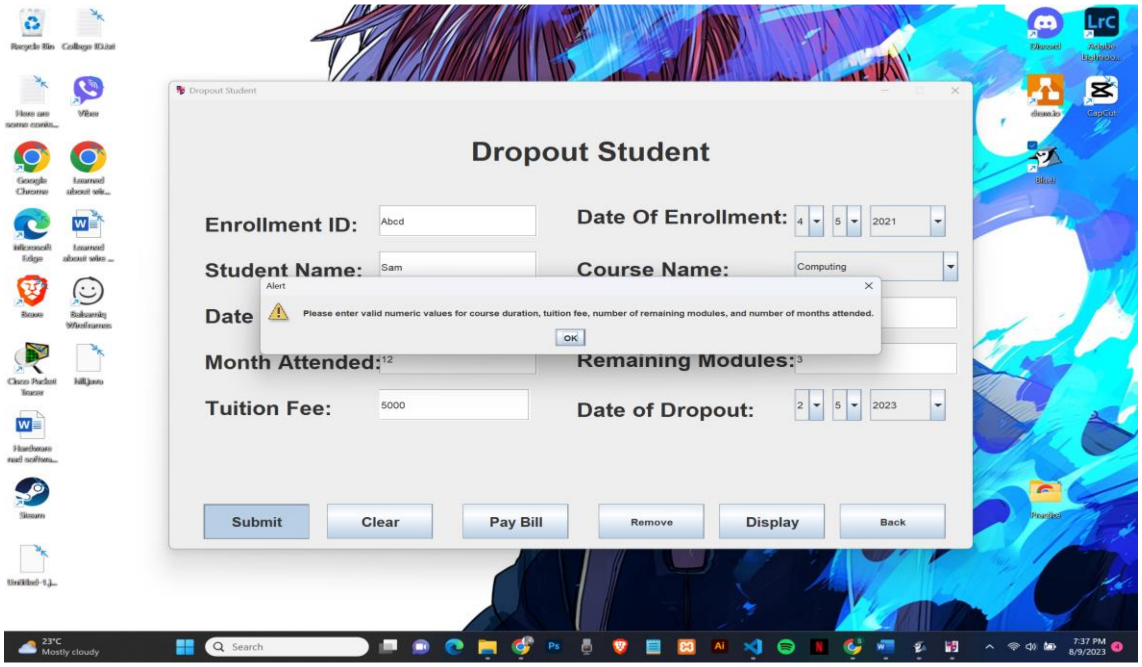
Enrollment ID:	Ad
Student Name:	213
Month Attended:	3
Tuition Fee:	5000
Date Of Enrollment:	4 1 2006
Course Name:	Computing
Remaining Modules:	3
Date of Dropout:	5 4 2023

Alert

Please enter valid numeric values for course duration, tuition fee, number of remaining modules, and number of months attended.

Figure 16 When Integer field were field with String.

When String field is filled with Integer.



Dropout Student

Enrollment ID:	Abcd
Student Name:	Sam
Month Attended:	12
Tuition Fee:	5000
Date Of Enrollment:	4 5 2021
Course Name:	Computing
Remaining Modules:	3
Date of Dropout:	2 5 2023

Alert

Please enter valid numeric values for course duration, tuition fee, number of remaining modules, and number of months attended.

Figure 17When String field is filled with Integer.

When all the fields are kept empty.

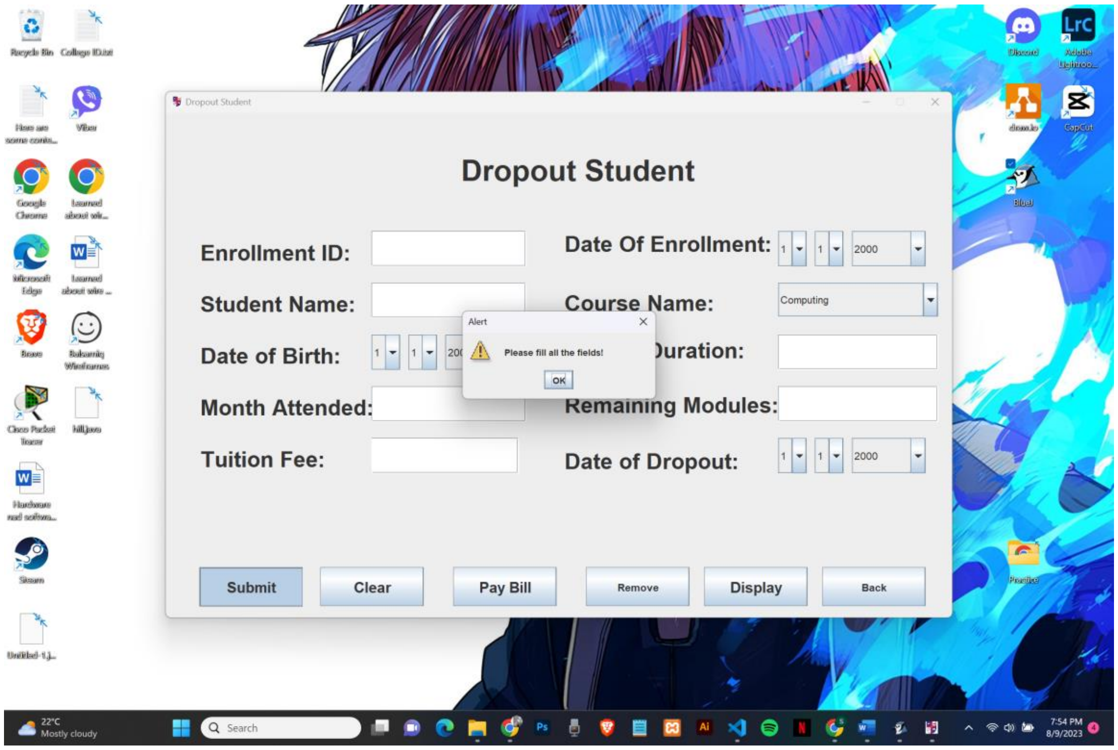


Figure 18 When all the fields are kept empty.

6. Error detection and correction

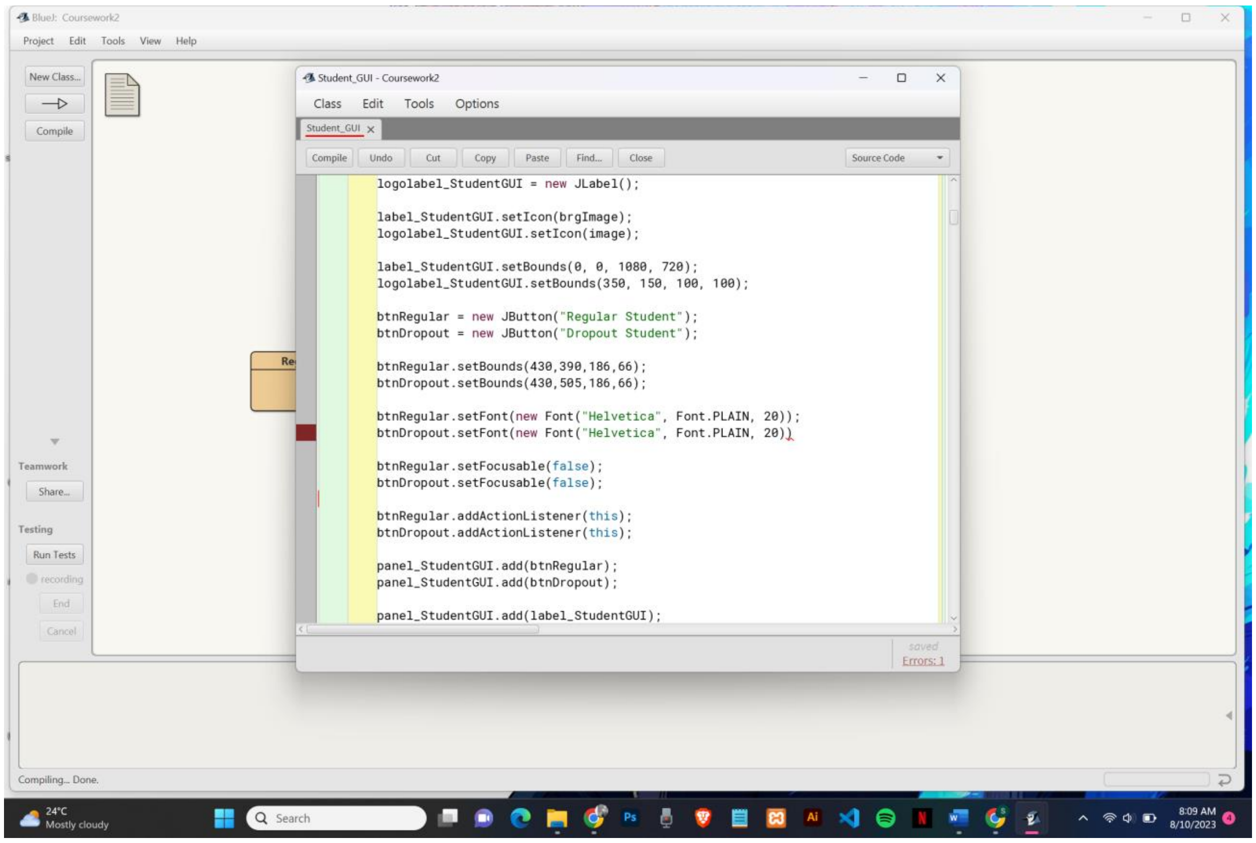
Error 1: Syntax error

Syntax is the set of rules that govern a language. In written and spoken language, rules can be bent or broken to accommodate the speaker or writer. However, in a programming language the rules are completely rigid. A syntax error occurs when the programmer writes an instruction using incorrect syntax. For example, 1 = x is not legal in the MATLAB programming language because numbers cannot be assigned as variables. If the programmer tries to execute one of these instructions or any other syntactically incorrect statement. (Alexandre M. Bayen, Timmy Siauw, 2015)

Error	1
Problem	There was syntax error while setting the Font of button because semi-colon was missing.
Solution	The problem was solved immediately since BlurJ notified the semi-colon was missing.
Encounter	While compiling and running code BlueJ detected error in code where semi-colon (;) was missing.

Table 10: Syntax error

Detection:

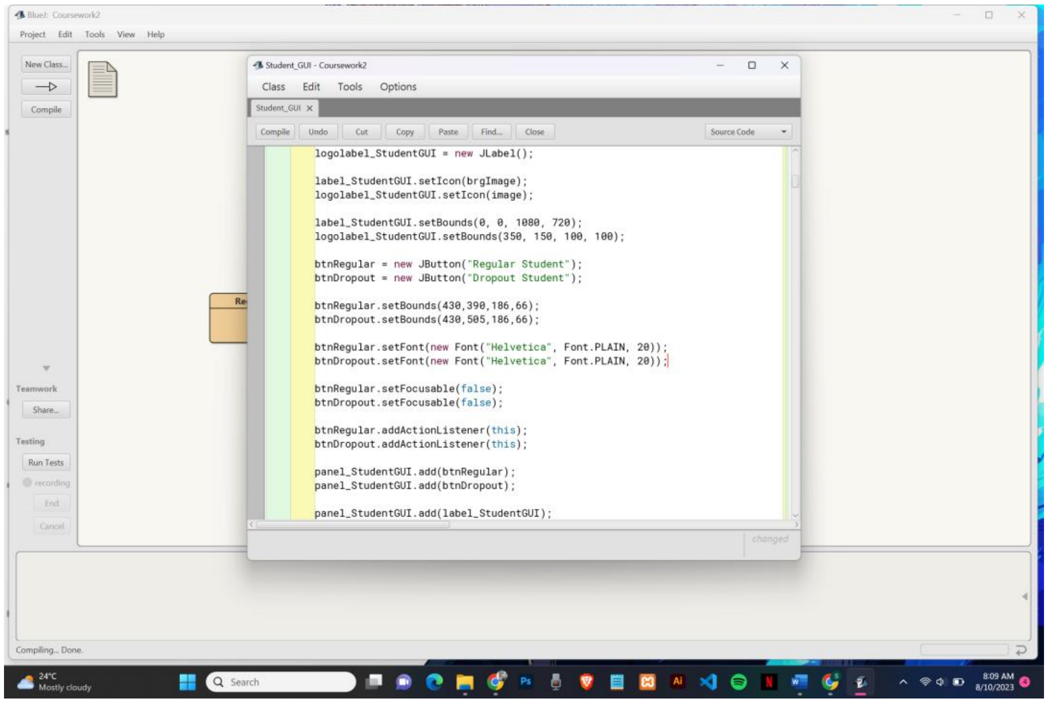


```
logolabel_StudentGUI = new JLabel();

label_StudentGUI.setIcon(brgImage);
logolabel_StudentGUI.setIcon(image);

label_StudentGUI.setBounds(0, 0, 1080, 720);
logolabel_StudentGUI.setBounds(350, 150, 100, 100);

btnRegular = new JButton("Regular Student");
btnDropout = new JButton("Dropout Student");

btnRegular.setBounds(430,390,186,66);
btnDropout.setBounds(430,505,186,66);

btnRegular.setFont(new Font("Helvetica", Font.PLAIN, 20));
btnDropout.setFont(new Font("Helvetica", Font.PLAIN, 20))

btnRegular.setFocusable(false);
btnDropout.setFocusable(false);

btnRegular.addActionListener(this);
btnDropout.addActionListener(this);

panel_StudentGUI.add(btnRegular);
panel_StudentGUI.add(btnDropout);

panel_StudentGUI.add(label_StudentGUI);
```

saved
Errors: 1

Compiling... Done.

Solution:



```
logolabel_StudentGUI = new JLabel();

label_StudentGUI.setIcon(brgImage);
logolabel_StudentGUI.setIcon(image);

label_StudentGUI.setBounds(0, 0, 1080, 720);
logolabel_StudentGUI.setBounds(350, 150, 100, 100);

btnRegular = new JButton("Regular Student");
btnDropout = new JButton("Dropout Student");

btnRegular.setBounds(430,390,186,66);
btnDropout.setBounds(430,505,186,66);

btnRegular.setFont(new Font("Helvetica", Font.PLAIN, 20));
btnDropout.setFont(new Font("Helvetica", Font.PLAIN, 20));

btnRegular.setFocusable(false);
btnDropout.setFocusable(false);

btnRegular.addActionListener(this);
btnDropout.addActionListener(this);

panel_StudentGUI.add(btnRegular);
panel_StudentGUI.add(btnDropout);

panel_StudentGUI.add(label_StudentGUI);
```

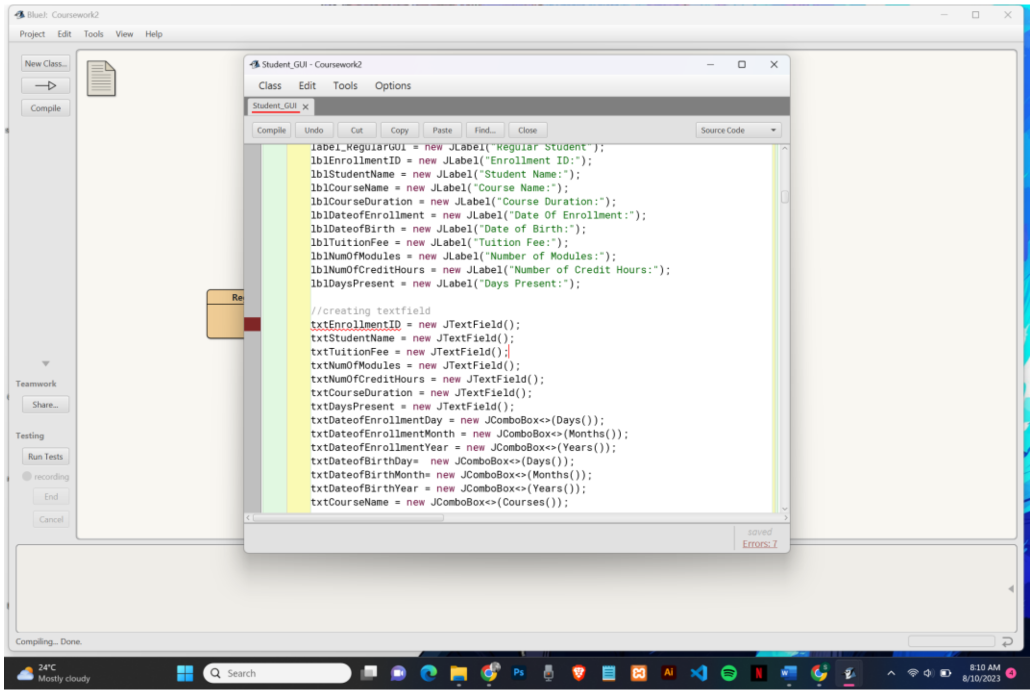
Error 2: Semantic error

Semantic errors are those errors which is detected during the compile time. Most of the compile time errors are scope and declaration error. Semantic errors can arises during use of wrong variables or using wrong operator or doing operation in wrong order.

Error	2
Problem	Semantic error was detected in creating object of JTextField() because obj name txtEnrollmentID was not decleared.
Solution	txtEnrollmentID variable was decleared which solved the problem.

Encounter	While compiling the code,BlueJ notified semantic error was detected in creating object of JTextField().

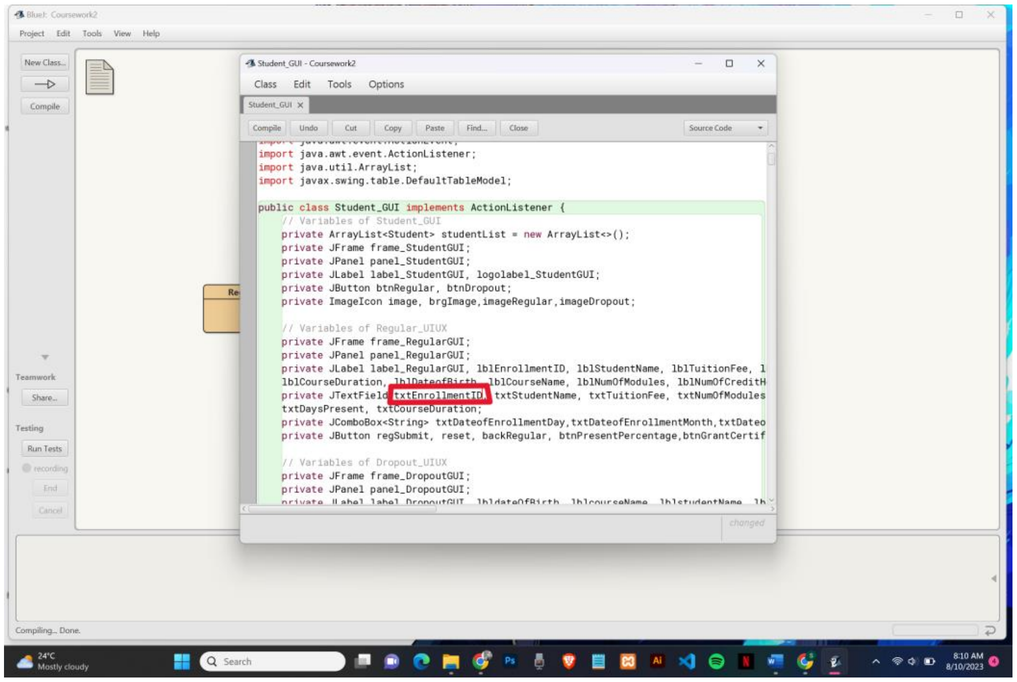
Table 11 Semantic error



```
label_RegularGUI = new JLabel("Regular Student");
lblEnrollmentID = new JLabel("Enrollment ID:");
lblStudentName = new JLabel("Student Name:");
lblCourseName = new JLabel("Course Name:");
lblCourseDuration = new JLabel("Course Duration:");
lblDateofEnrollment = new JLabel("Date Of Enrollment:");
lblDateofBirth = new JLabel("Date of Birth:");
lblTuitionFee = new JLabel("Tuition Fee:");
lblNumOfModules = new JLabel("Number of Modules:");
lblNumOfCreditHours = new JLabel("Number of Credit Hours:");
lblDaysPresent = new JLabel("Days Present:");

//creating textfield
txtEnrollmentID = new JTextField();
txtStudentName = new JTextField();
txtTuitionFee = new JTextField();
txtNumOfModules = new JTextField();
txtNumOfCreditHours = new JTextField();
txtCourseDuration = new JTextField();
txtDaysPresent = new JTextField();
txtDateofEnrollmentDay = new JComboBox<>(Days());
txtDateofEnrollmentMonth = new JComboBox<>(Months());
txtDateofEnrollmentYear = new JComboBox<>(Years());
txtDateofBirthDay=  new JComboBox<>(Days());
txtDateofBirthMonth= new JComboBox<>(Months());
txtDateofBirthYear = new JComboBox<>(Years());
txtCourseName = new JComboBox<>(Courses());
```

Errors: 7

Figure 19Semantic Error Problem



```
import java.awt.event.ActionListener;
import java.util.ArrayList;
import javax.swing.table.DefaultTableModel;

public class Student_GUI implements ActionListener {
    // Variables of Student_GUI
    private ArrayList<Student> studentList = new ArrayList<>();
    private JFrame frame_StudentGUI;
    private JPanel panel_StudentGUI;
    private JLabel label_StudentGUI, logolabel_StudentGUI;
    private JButton btnRegular, btnDropout;
    private ImageIcon image, brgImage,imageRegular,imageDropout;

    // Variables of Regular_UIUX
    private JFrame frame_RegularGUI;
    private JPanel panel_RegularGUI;
    private JLabel label_RegularGUI, lblEnrollmentID, lblStudentName, lblTuitionFee, l
    lblCourseDuration, lblDateofBirth, lblCourseName, lblNumOfModules, lblNumOfCreditH
    private JTextField txtEnrollmentID, txtStudentName, txtTuitionFee, txtNumOfModules
    txtDaysPresent, txtCourseDuration;
    private JComboBox<String> txtDateofEnrollmentDay,txtDateofEnrollmentMonth,txtDateo
    private JButton regSubmit, reset, backRegular, btnPresentPercentage,btnGrantCertif

    // Variables of Dropout_UIUX
    private JFrame frame_DropoutGUI;
    private JPanel panel_DropoutGUI;
```

Figure 20 Semantic Error Solution

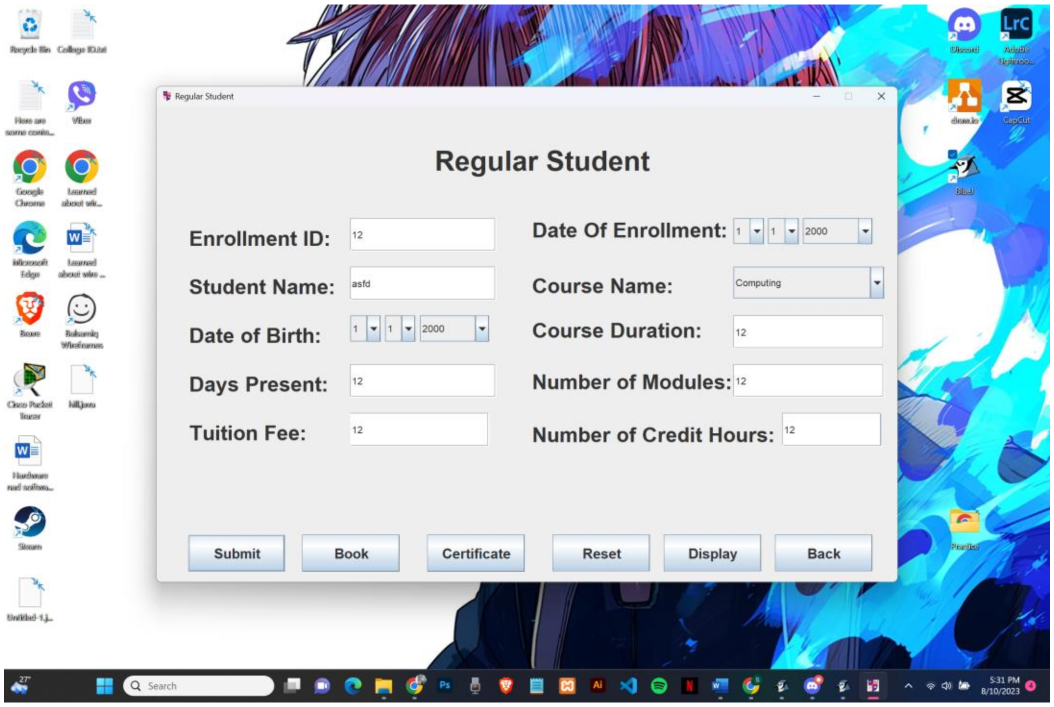
Error 3: Logical error

A logic error (or logical error) is a 'bug' or mistake in a program's source code that results in incorrect or unexpected behaviour. It is a type of runtime error that may simply produce the wrong output or may cause a program to crash while running. (Teacher;s Note, n.d.)

Error	3
Problem	In regular form when submit button was clicked it did not register or showed dialog message box.
Solution	The problem was solved after reviewed the logical code of submit button of regular form, after reviewing code I found the error in if else condition where if(isAlreadyRegistered == true) and fixed this as if(isAlreadyRegistered == false).
Encounter	Reviewed the logical code of submit button of regular form, after reviewing code I found the error in if else condition where if(isAlreadyRegistered == true).

Table 12 Logical error

Screenshot of Logical error problem:


Regular Student

Enrollment ID: 12

Date Of Enrollment: 1 1 2000

Student Name: asfd

Course Name: Computing

Date of Birth: 1 1 2000

Course Duration: 12

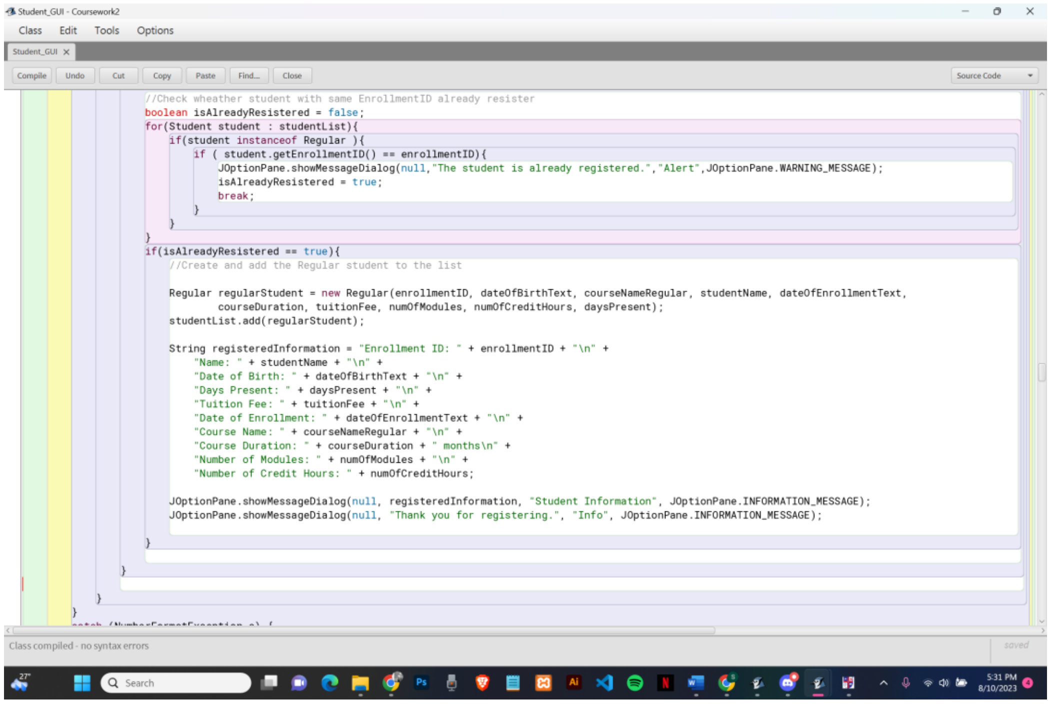
Days Present: 12

Number of Modules: 12

Tuition Fee: 12

Number of Credit Hours: 12

Figure 21Logical Error Problem

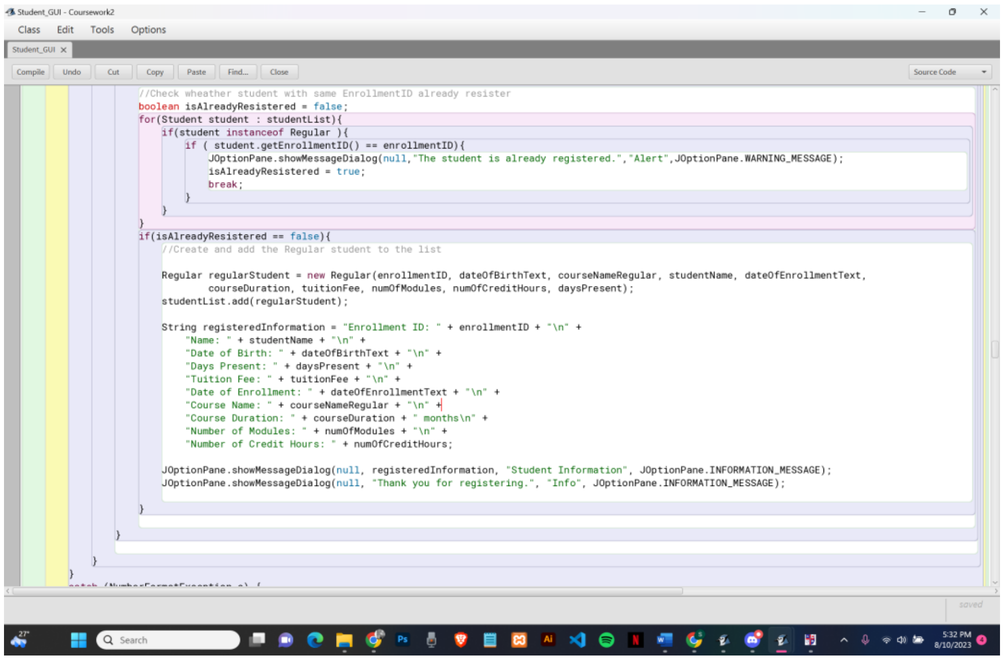


```java
//Check wheather student with same EnrollmentID already resister
boolean isAlreadyResistered = false;
for(Student student : studentList){
    if(student instanceof Regular ){
        if ( student.getEnrollmentID() == enrollmentID){
            JOptionPane.showMessageDialog(null,"The student is already registered.","Alert",JOptionPane.WARNING_MESSAGE);
            isAlreadyResistered = true;
            break;
        }
    }
}
if(isAlreadyResistered == true){
    //Create and add the Regular student to the list

    Regular regularStudent = new Regular(enrollmentID, dateOfBirthText, courseNameRegular, studentName, dateOfEnrollmentText,
            courseDuration, tuitionFee, numOfModules, numOfCreditHours, daysPresent);
    studentList.add(regularStudent);

    String registeredInformation = "Enrollment ID: " + enrollmentID + "\n" +
        "Name: " + studentName + "\n" +
        "Date of Birth: " + dateOfBirthText + "\n" +
        "Days Present: " + daysPresent + "\n" +
        "Tuition Fee: " + tuitionFee + "\n" +
        "Date of Enrollment: " + dateOfEnrollmentText + "\n" +
        "Course Name: " + courseNameRegular + "\n" +
        "Course Duration: " + courseDuration + " months\n" +
        "Number of Modules: " + numOfModules + "\n" +
        "Number of Credit Hours: " + numOfCreditHours;

    JOptionPane.showMessageDialog(null, registeredInformation, "Student Information", JOptionPane.INFORMATION_MESSAGE);
    JOptionPane.showMessageDialog(null, "Thank you for registering.", "Info", JOptionPane.INFORMATION_MESSAGE);
}
```

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

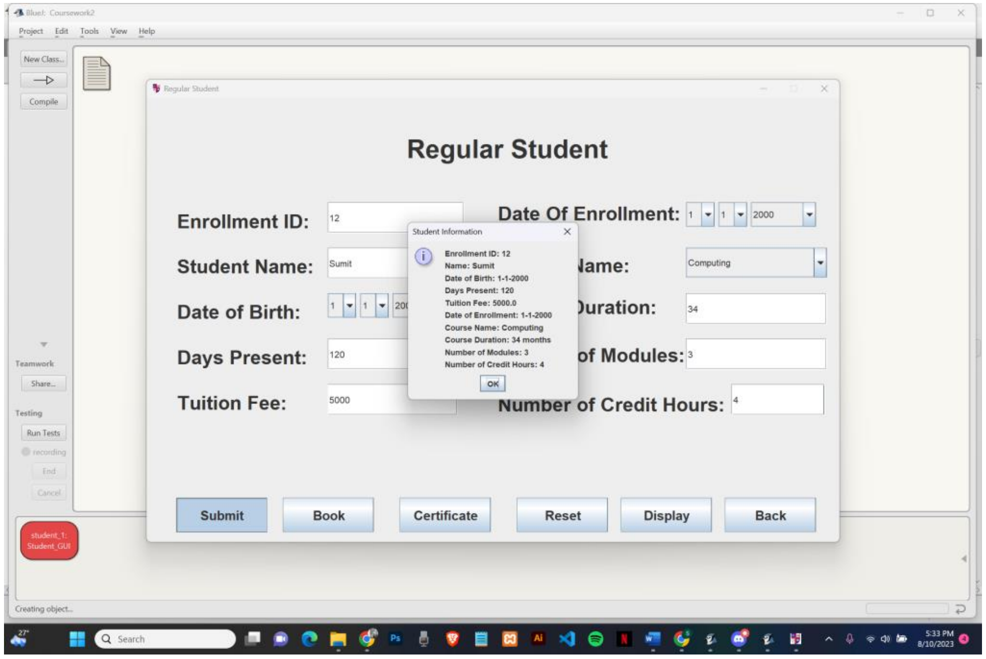
Figure 22 Logical Error Problem in code

Screenshot of Logical error solution:



```
//Check wheather student with same EnrollmentID already resister
boolean isAlreadyResistered = false;
for(Student student : studentList){
    if(student instanceof Regular ){
        if ( student.getEnrollmentID() == enrollmentID){
            JOptionPane.showMessageDialog(null,"The student is already registered.","Alert",JOptionPane.WARNING_MESSAGE);
            isAlreadyResistered = true;
            break;
        }
    }
}
if(isAlreadyResistered == false){
    //Create and add the Regular student to the list

    Regular regularStudent = new Regular(enrollmentID, dateOfBirthText, courseNameRegular, studentName, dateOfEnrollmentText,
            courseDuration, tuitionFee, numOfModules, numOfCreditHours, daysPresent);
    studentList.add(regularStudent);

    String registeredInformation = "Enrollment ID: " + enrollmentID + "\n" +
        "Name: " + studentName + "\n" +
        "Date of Birth: " + dateOfBirthText + "\n" +
        "Days Present: " + daysPresent + "\n" +
        "Tuition Fee: " + tuitionFee + "\n" +
        "Date of Enrollment: " + dateOfEnrollmentText + "\n" +
        "Course Name: " + courseNameRegular + "\n" +
        "Course Duration: " + courseDuration + " months\n" +
        "Number of Modules: " + numOfModules + "\n" +
        "Number of Credit Hours: " + numOfCreditHours;

    JOptionPane.showMessageDialog(null, registeredInformation, "Student Information", JOptionPane.INFORMATION_MESSAGE);
    JOptionPane.showMessageDialog(null, "Thank you for registering.", "Info", JOptionPane.INFORMATION_MESSAGE);

}
```

Figure 23 Logical error Solution in code



Regular Student

Enrollment ID: 12
Student Name: Sumit
Date of Birth: 1 1 200
Days Present: 120
Tuition Fee: 5000
Date Of Enrollment: 1 1 2000
Name: Computing
Duration: 34
of Modules: 3
Number of Credit Hours: 4

Student Information

```
Enrollment ID: 12
Name: Sumit
Date of Birth: 1-1-2000
Days Present: 120
Tuition Fee: 5000.0
Date of Enrollment: 1-1-2000
Course Name: Computing
Course Duration: 34 months
Number of Modules: 3
Number of Credit Hours: 4
```

Figure 24 Logical error solution

7. Conclusion

This Java coursework was assigned to the students of year one second semester, where we had created GUI or Graphical User Interface for the Registering students data. Java being a object oriented programming language, use of objects have made the coursework more efficient and easy. In this coursework we used and learned about java swings, java event handling and array list.

Since, GUI development was my first time it was difficult to understand all the processes and couldn’t be able to create the good GUI as I wanted. Learning the GUI in java was enjoyable and helped to boost my confidence in programming.

Taking or getting the value from the GUI was really hard task but after searching in different platforms and googles i was able to make the GUI workable. There were different buttons in the GUI such as submit, clear, back. Remove, etc. Making all buttons to work was difficult task, it took a lot of time for me to make them work. I tried to create user friendly GUI but I was not able to make the GUI as good as I thought.

I learned that making multiple methods and calling the methods can make the coding process and development more efficient and easier.

Doing this coursework I learned about different java packets, event handling and event handling. My coding skill has been improved after doing this coursework.

8. References

Alexandre M. Bayen, Timmy Siauw, 2015. *Errors, Good Programming Practices, and Debugging.* [Online]
Available at: https://www.sciencedirect.com/book/9780081008775/essential-matlab-for-engineers-and-scientists

Google, n.d. [Online]
Available at: https://www.google.com/

Teacher;s Note, n.d. *Cambridge.* [Online]
Available at:
https://cambridgegcsecomputing.org/sites/94/upload/userfiles/217p_q_teachersnotes_editedct_3proof.pdf

9. Appendix

```
public class Student
{
    //Attributes
    private  String dateOfBirth;
    private String courseName;
    private String studentName;
    private String dateOfEnrollment;
    private int enrollmentID;
    private int courseDuration;
    private double tuitionFee;
    //Constructor
    public Student(String dateOfBirth, String studentName, int courseDuration, double
tuitionFee)
    {
        this.courseName = "";
        this.dateOfEnrollment = "";
        this.enrollmentID = 0;
        this.dateOfBirth = dateOfBirth;
        this.studentName = studentName;
        this.courseDuration = courseDuration;
        this.tuitionFee = tuitionFee;
    }
    //Accessor Method
    public String getCourseName(){
        return courseName;
    }
```

```
public String getDateOfEnrollment(){
    return dateOfEnrollment;
}

public int getEnrollmentID(){
    return enrollmentID;
}

public String getDateOfBirth(){
    return dateOfBirth;
}

public String getStudentName(){
    return studentName;
}

public int getCourseDuration(){
    return courseDuration;
}

public double getTuitionFee(){
    return tuitionFee;
}
//set methods
public void setCourseName(String courseName){
    this.courseName=courseName;
}
```

```
public void setEnrollmentID(int enrollmentID){
    this.enrollmentID = enrollmentID;
}
// mutator method for the attribute dateOfEnrollment
public void setDateOfEnrollment(String dateOfEnrollment){
    this.dateOfEnrollment = dateOfEnrollment;
}
//setter method for date of birth, student name, course duration, tuition fee
public void setDateOfBirth(String dateOfBirth)
{
    this.dateOfBirth=dateOfBirth;
}
public void setStudentName(String studentName){
    this.studentName=studentName;
}
public void setCourseDuration(int courseDuration){
    this.courseDuration=courseDuration;
}
public void setTuitionFee(double tuitionFee){
    this.tuitionFee = tuitionFee;
}
//method for display
public void display(){
    if (enrollmentID == 0 || courseName == null || dateOfEnrollment == null )
    {
        System.out.println("Some of the parameters are missing.");
        }else{
            int yearsEnrolled = courseDuration/12;
            System.out.println("EnrollmentID:"+enrollmentID);
            System.out.println("DateOfBirth:"+dateOfBirth);
            System.out.println("CourseName:"+courseName);
            System.out.println("StudentName:"+studentName);
            System.out.println("YearsEnrolled:"+yearsEnrolled);
            System.out.println("CourseDuration:"+courseDuration);
            System.out.println("TuitionFee:"+tuitionFee);

        }
    }
}

public class Regular extends Student{
    private int numOfModules;
    private int numOfCreditHours;
    private double daysPresent;
    private boolean isGrantedScholarship;
    //Constructor
    public Regular(int enrollmentID, String dateOfBirth,String courseName, String
studentName, String dateOfEnrollment,int courseDuration,double tuitionFee,int
numOfModules,int numOfCreditHours,double daysPresent){
        //calling from parent class
        super(dateOfBirth,studentName,courseDuration,tuitionFee);
        super.setEnrollmentID(enrollmentID);
        super.setCourseName(courseName);
        super.setDateOfEnrollment(dateOfEnrollment);
        this.numOfModules = numOfModules;
        this.numOfCreditHours = numOfCreditHours;
        this.daysPresent = daysPresent;
        this.isGrantedScholarship = false;
    }
    //Accessor Method
    public int getNumOfModules(){
        return numOfModules;
    }

    public int getNumOfCreditHours(){
        return numOfCreditHours;
    }

    public double getDaysPresent(){
        return daysPresent;
    }

    public boolean getIsGrantedScholarship() {
        return isGrantedScholarship;
    }

    public char presentPercentage(double daysPresent){
        double presentPercentage = (this.daysPresent / super.getCourseDuration()) * 100;
        if (super.getCourseDuration() > daysPresent || daysPresent < 0){
            System.out.println("Invalid days present..");
            return 'N';
        }
        else if (presentPercentage >= 80)
        {
            this.isGrantedScholarship = true;
            return 'A';
        } else if (presentPercentage >= 60){
            this.isGrantedScholarship = false;
            return 'B';
        } else if (presentPercentage >= 40){
            this.isGrantedScholarship = false;
            return 'C';
        } else if (presentPercentage >= 20)
        {
            this.isGrantedScholarship = false;
            return 'D';
        } else {
            this.isGrantedScholarship = false;
            return 'F';
        }
    }
    //Method for grantCertificate
    public void grantCertificate(String courseName, int enrollmentID,String
dateOfEnrollment)
    {
        if (isGrantedScholarship == true)
        {
            System.out.println("The scholarship has been granted");
            System.out.println("The student having EnrollmentID"+enrollmentID+"has
graduated from the " +courseName+"course on " +dateOfEnrollment);
        }
        else
        {
            System.out.println("The student having EnrollmentID"+enrollmentID+"hasnot
graduated from the " +courseName+"course on " +dateOfEnrollment);
        }
    }
    //Method to Display
    public void displayRegular(){
        super.display();//calling from parent class
        System.out.println("NumOfModules :"+numOfModules);
        System.out.println("NumOfCreditHours:"+numOfCreditHours);
        System.out.println("DaysPresent:"+daysPresent);
        double presentPercentage = presentPercentage(daysPresent);
        if (presentPercentage >= 0) {
        System.out.println("Present Percentage: " + presentPercentage + "%");
    }
    }

}
//The Dropout class is also a subclass of the Student class and it has five attributes
public class Dropout extends Student

{
    //Attrributes
    private int numOfRemainingModules;
    private int numOfMonthsAttended;
    private String dateOfDropout;
    private int remainingAmount;
    private boolean hasPaid;
    //Creating Constructor
    public Dropout(String dateOfBirth, String studentName, int courseDuration, double
tuitionFee,
    int numOfRemainingModules, int numOfMonthsAttended, String dateOfDropout,int
enrollmentID,String courseName,String dateOfEnrollment) {

        super(dateOfBirth, studentName, courseDuration, tuitionFee);//A call is made to
the superclass constructor with four parameters
        super.setEnrollmentID(enrollmentID);
        super.setCourseName(courseName);
        super.setDateOfEnrollment(dateOfEnrollment);

        this.numOfRemainingModules = numOfRemainingModules;
        this.numOfMonthsAttended = numOfMonthsAttended;
        this.dateOfDropout = dateOfDropout;
        this.remainingAmount = 0;
        this.hasPaid = false;

    }
    //accessor method
    public int getNumOfRemainingModules() {
        return this.numOfRemainingModules;
    }

    public int getNumOfMonthsAttended() {
        return this.numOfMonthsAttended;
    }

    public String getDateOfDropout() {
        return this.dateOfDropout;
    }

    public int getRemainingAmount() {
        return this.remainingAmount;
    }

    public boolean getHasPaid() {
        return this.hasPaid;
    }
    //setter method
    public void setNumOfRemainingModules(int numOfRemainingModules) {
        this.numOfRemainingModules = numOfRemainingModules;
    }

    public void setDateOfDropout(String dateOfDropout) {
        this.dateOfDropout = dateOfDropout;
    }

    public void setRemainingAmount(int remainingAmount) {
        this.remainingAmount = remainingAmount;
    }

    public boolean hasPaid() {
        return hasPaid;
    }

    //method to calculate pending amount to be paid.
    public boolean billsPayable(double amountPaid) {
        double pendingAmount = super.getTuitionFee() - amountPaid;

        if (pendingAmount > 0) {
            this.remainingAmount = (int) pendingAmount;
            this.hasPaid = false;
            System.out.println("The pending amount to be paid: $" + pendingAmount);
            return false; // Bills are not fully paid
        } else if (pendingAmount == 0) {
            this.remainingAmount = 0;
            this.hasPaid = true;
            System.out.println("No pending amount. The student has paid the entire tuition
fee.");
            return true; // Bills are fully paid
        } else {
            System.out.println("Amount paid is more than the tuition fee. No pending
amount.");
            return true; // Bills are fully paid
        }
    }
    //Method to remove student
    public void removeStudent(int enrollmentID) {
        if (this.hasPaid == true) {
            setDateOfBirth("");
            setCourseName("");
            setStudentName("");
            setDateOfEnrollment("");
            setCourseDuration(0);
            setTuitionFee(0.0);
            setEnrollmentID(0);
            this.numOfRemainingModules = 0;
            this.numOfMonthsAttended = 0;
            this.dateOfDropout = "";
            this.remainingAmount = 0;
            System.out.println("Student removed.");
        } else {
            System.out.println("All bills not cleared.");
        }
    }
    //Method for Display
    public void display() {
        super.display();//calling from parent class
        System.out.println("Number of remaining modules: " +
this.numOfRemainingModules);
        System.out.println("Number of months attended: " + this.numOfMonthsAttended);
        System.out.println("Date of dropout: " + this.dateOfDropout);
        System.out.println("Remaining amount: " + this.remainingAmount);
    }


}
/**
 * Write a description of class Glkj here.
 *CourseWrork 2
 * Sumit Shrestha
 */
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.ArrayList;
import javax.swing.table.DefaultTableModel;

public class Student_GUI implements ActionListener {
    // Variables of Student_GUI
    private ArrayList<Student> studentList = new ArrayList<>();
    private JFrame frame_StudentGUI;
    private JPanel panel_StudentGUI;
    private JLabel label_StudentGUI, logolabel_StudentGUI;
    private JButton btnRegular, btnDropout;
    private ImageIcon image, brgImage,imageRegular,imageDropout;

    // Variables of Regular_UIUX
    private JFrame frame_RegularGUI;
    private JPanel panel_RegularGUI;
    private JLabel label_RegularGUI, lblEnrollmentID, lblStudentName, lblTuitionFee,
lblDateofEnrollment,
    lblCourseDuration, lblDateofBirth, lblCourseName, lblNumOfModules,
lblNumOfCreditHours, lblDaysPresent;
    private JTextField txtEnrollmentID, txtStudentName, txtTuitionFee, txtNumOfModules,
txtNumOfCreditHours,
    txtDaysPresent, txtCourseDuration;
    private JComboBox<String>
txtDateofEnrollmentDay,txtDateofEnrollmentMonth,txtDateofEnrollmentYear,
txtDateofBirthDay,txtDateofBirthMonth,txtDateofBirthYear, txtCourseName;
    private JButton regSubmit, reset, backRegular,
btnPresentPercentage,btnGrantCertificate;

    // Variables of Dropout_UIUX
    private JFrame frame_DropoutGUI;
    private JPanel panel_DropoutGUI;
    private JLabel label_DropoutGUI, lbldateOfBirth, lblcourseName, lblstudentName,
lbldateOfEnrollement, lblenrollmentID,
    lblcourseDuration, lbltuitionFee, lblNumOfRemainingModules,
lblNumOfMonthsAttended, lbldateofDropout;
    private JTextField  txtstudentName, txtenrollmentId, txtcourseDuration, txttuitionFee,
    txtNumOfRemainingModules, txtNumOfMonthsAttended;
    private JComboBox<String>
txtdateOfBirthDay,txtdateOfBirthMonth,txtdateOfBirthYear,
txtdateOfEnrollmentDay,txtdateOfEnrollmentMonth,txtdateOfEnrollmentYear,
txtdateOfDropoutDay,txtdateOfDropoutMonth,txtdateOfDropoutYear,txtcourseName;
    private JButton resetDropout, submitDropout, backDropout,
btnPayBills,removeStudent;


    //variables of display
    private JTable table;
    private JButton btnDisplayRegular,btnDisplayDropout, btnClear;


    public Student_GUI() {
        frame_StudentGUI = new JFrame();


        image = new ImageIcon("Images/c.jpg");
        brgImage = new ImageIcon("Images/a.jpg");


        panel_StudentGUI = new JPanel();
        panel_StudentGUI.setLayout(null);
        frame_StudentGUI.add(panel_StudentGUI);


        label_StudentGUI = new JLabel();
        logolabel_StudentGUI = new JLabel();

        label_StudentGUI.setIcon(brgImage);
        logolabel_StudentGUI.setIcon(image);

        label_StudentGUI.setBounds(0, 0, 1080, 720);
        logolabel_StudentGUI.setBounds(350, 150, 100, 100);

        btnRegular = new JButton("Regular Student");
        btnDropout = new JButton("Dropout Student");

        btnRegular.setBounds(430,390,186,66);
        btnDropout.setBounds(430,505,186,66);

        btnRegular.setFont(new Font("Helvetica", Font.PLAIN, 20));
        btnDropout.setFont(new Font("Helvetica", Font.PLAIN, 20));

        btnRegular.setFocusable(false);
        btnDropout.setFocusable(false);

        btnRegular.addActionListener(this);
        btnDropout.addActionListener(this);

        panel_StudentGUI.add(btnRegular);
        panel_StudentGUI.add(btnDropout);

        panel_StudentGUI.add(label_StudentGUI);
        panel_StudentGUI.add(logolabel_StudentGUI);
```

```
        frame_StudentGUI.setIconImage(image.getImage());
        frame_StudentGUI.setTitle("Student's Form");
        frame_StudentGUI.setSize(1080, 720);
        frame_StudentGUI.setResizable(false);
        frame_StudentGUI.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame_StudentGUI.setVisible(true);
        frame_StudentGUI.setLocationRelativeTo(null);
    }

    // Regular UIUX frame
    public void regular_UIUX() {
        frame_RegularGUI = new JFrame("Regular Student");

        imageRegular = new ImageIcon("Images/c.jpg");

        panel_RegularGUI = new JPanel();
        panel_RegularGUI.setLayout(null);
        frame_RegularGUI.add(panel_RegularGUI);

        //creating Label
        label_RegularGUI = new JLabel("Regular Student");
        lblEnrollmentID = new JLabel("Enrollment ID:");
        lblStudentName = new JLabel("Student Name:");
        lblCourseName = new JLabel("Course Name:");
        lblCourseDuration = new JLabel("Course Duration:");
        lblDateofEnrollment = new JLabel("Date Of Enrollment:");
        lblDateofBirth = new JLabel("Date of Birth:");
```

```
lblTuitionFee = new JLabel("Tuition Fee:");
lblNumOfModules = new JLabel("Number of Modules:");
lblNumOfCreditHours = new JLabel("Number of Credit Hours:");
lblDaysPresent = new JLabel("Days Present:");

//creating textfield
txtEnrollmentID = new JTextField();
txtStudentName = new JTextField();
txtTuitionFee = new JTextField();
txtNumOfModules = new JTextField();
txtNumOfCreditHours = new JTextField();
txtCourseDuration = new JTextField();
txtDaysPresent = new JTextField();
txtDateofEnrollmentDay = new JComboBox<>(Days());
txtDateofEnrollmentMonth = new JComboBox<>(Months());
txtDateofEnrollmentYear = new JComboBox<>(Years());
txtDateofBirthDay=  new JComboBox<>(Days());
txtDateofBirthMonth= new JComboBox<>(Months());
txtDateofBirthYear = new JComboBox<>(Years());
txtCourseName = new JComboBox<>(Courses());

//creating buttons
regSubmit = new JButton("Submit");
reset = new JButton("Reset");
backRegular = new JButton("Back");
btnPresentPercentage = new JButton("Book");
btnGrantCertificate = new JButton("Certificate");
btnDisplayRegular = new JButton("Display");
```

```
label_RegularGUI.setBounds(400, 40, 513, 75);
lblEnrollmentID.setBounds(47, 150, 222, 76);
lblStudentName.setBounds(47, 220, 222, 75);
lblDateofBirth.setBounds(47, 290, 222, 75);
lblDaysPresent.setBounds(47, 360, 222, 75);
lblTuitionFee.setBounds(47, 430, 222, 75);

lblDateofEnrollment.setBounds(540, 150, 289, 52);
lblCourseName.setBounds(540, 220, 248, 73);
lblCourseDuration.setBounds(540, 290, 248, 60);
lblNumOfModules.setBounds(540, 360, 305, 68);
lblNumOfCreditHours.setBounds(540, 430, 379,80);

txtEnrollmentID.setBounds(279, 160, 210, 48);
txtStudentName.setBounds(279, 230, 210, 48);
txtDateofBirthDay.setBounds(279, 300, 44, 38);
txtDateofBirthMonth.setBounds(329, 300, 44, 38);
txtDateofBirthYear.setBounds(379, 300, 100, 38);
txtDaysPresent.setBounds(279, 370, 210, 48);
txtTuitionFee.setBounds(279, 440, 200, 48);

txtDateofEnrollmentDay.setBounds(830, 160, 44, 38);
txtDateofEnrollmentMonth.setBounds(880, 160,44, 38);
txtDateofEnrollmentYear.setBounds(930, 160, 100, 38);

txtCourseName.setBounds(830, 230, 217, 46);
txtCourseDuration.setBounds(830, 300, 217, 48);
```

```
txtNumOfModules.setBounds(830,370, 217, 48);
txtNumOfCreditHours.setBounds(900,440, 144, 48);

regSubmit.setBounds(46, 615, 140, 53);
btnPresentPercentage.setBounds(210,615,140,53);
btnGrantCertificate.setBounds(390,615,140,53);
reset.setBounds(570, 615, 140, 53);
btnDisplayRegular.setBounds(730,615,140,53);
backRegular.setBounds(890, 615, 140, 53);

label_RegularGUI.setFont(new Font("Helvetica", Font.BOLD, 40));
lblEnrollmentID.setFont(new Font("Helvetica", Font.BOLD, 30));
lblStudentName.setFont(new Font("Helvetica", Font.BOLD, 30));
lblCourseName.setFont(new Font("Helvetica", Font.BOLD, 30));
lblCourseDuration.setFont(new Font("Helvetica", Font.BOLD, 30));
lblDateofEnrollment.setFont(new Font("Helvetica", Font.BOLD, 30));
lblDateofBirth.setFont(new Font("Helvetica", Font.BOLD, 30));
lblTuitionFee.setFont(new Font("Helvetica", Font.BOLD, 30));
lblNumOfModules.setFont(new Font("Helvetica", Font.BOLD, 30));
lblNumOfCreditHours.setFont(new Font("Helvetica", Font.BOLD, 30));
lblDaysPresent.setFont(new Font("Helvetica", Font.BOLD, 30));

txtEnrollmentID.setFont(new Font("Helvetica", Font.PLAIN, 14));
txtStudentName.setFont(new Font("Helvetica", Font.PLAIN, 14));
txtCourseName.setFont(new Font("Helvetica", Font.PLAIN, 14));
txtCourseDuration.setFont(new Font("Helvetica", Font.PLAIN, 14));
txtDateofEnrollmentDay.setFont(new Font("Helvetica", Font.PLAIN, 14));
txtDateofEnrollmentMonth.setFont(new Font("Helvetica", Font.PLAIN, 14));
```

```
txtDateofEnrollmentYear.setFont(new Font("Helvetica", Font.PLAIN, 14));
txtDateofBirthDay.setFont(new Font("Helvetica", Font.PLAIN, 14));
txtDateofBirthMonth.setFont(new Font("Helvetica", Font.PLAIN, 14));
txtDateofBirthYear.setFont(new Font("Helvetica", Font.PLAIN, 14));
txtTuitionFee.setFont(new Font("Helvetica", Font.PLAIN, 14));
txtNumOfModules.setFont(new Font("Helvetica", Font.PLAIN, 14));
txtNumOfCreditHours.setFont(new Font("Helvetica", Font.PLAIN, 14));
txtDaysPresent.setFont(new Font("Helvetica", Font.PLAIN, 14));

regSubmit.setFont(new Font("Helvetica", Font.BOLD, 20));
btnPresentPercentage.setFont(new Font("Helvetica", Font.BOLD, 20));
btnGrantCertificate.setFont(new Font("Helvetica", Font.BOLD, 20));
reset.setFont(new Font("Helvetica", Font.BOLD, 20));
btnDisplayRegular.setFont(new Font("Helvetica", Font.BOLD, 20));
backRegular.setFont(new Font("Helvetica", Font.BOLD, 20));

regSubmit.setFocusable(false);
reset.setFocusable(false);
backRegular.setFocusable(false);
btnPresentPercentage.setFocusable(false);
btnDisplayRegular.setFocusable(false);
btnGrantCertificate.setFocusable(false);

regSubmit.addActionListener(this);
reset.addActionListener(this);
backRegular.addActionListener(this);
btnPresentPercentage.addActionListener(this);
btnGrantCertificate.addActionListener(this);
```

```
btnDisplayRegular.addActionListener(this);

panel_RegularGUI.add(label_RegularGUI);
panel_RegularGUI.add(lblEnrollmentID);
panel_RegularGUI.add(lblStudentName);
panel_RegularGUI.add(lblCourseName);
panel_RegularGUI.add(lblCourseDuration);
panel_RegularGUI.add(lblDateofEnrollment);
panel_RegularGUI.add(lblDateofBirth);
panel_RegularGUI.add(lblTuitionFee);
panel_RegularGUI.add(lblNumOfModules);
panel_RegularGUI.add(lblNumOfCreditHours);
panel_RegularGUI.add(lblDaysPresent);

panel_RegularGUI.add(txtEnrollmentID);
panel_RegularGUI.add(txtStudentName);
panel_RegularGUI.add(txtCourseName);
panel_RegularGUI.add(txtCourseDuration);
panel_RegularGUI.add(txtDateofEnrollmentDay);
panel_RegularGUI.add(txtDateofEnrollmentMonth);
panel_RegularGUI.add(txtDateofEnrollmentYear);
panel_RegularGUI.add(txtDateofBirthDay);
panel_RegularGUI.add(txtDateofBirthMonth);
panel_RegularGUI.add(txtDateofBirthYear);
panel_RegularGUI.add(txtTuitionFee);
panel_RegularGUI.add(txtNumOfModules);
panel_RegularGUI.add(txtNumOfCreditHours);
panel_RegularGUI.add(txtDaysPresent);
```

```
        panel_RegularGUI.add(regSubmit);
        panel_RegularGUI.add(reset);
        panel_RegularGUI.add(backRegular);
        panel_RegularGUI.add(btnPresentPercentage);
        panel_RegularGUI.add(btnGrantCertificate);
        panel_RegularGUI.add(btnDisplayRegular);

        frame_RegularGUI.setIconImage(imageRegular.getImage());
        frame_RegularGUI.setTitle("Regular Form");
        frame_RegularGUI.setSize(1080, 720);
        frame_RegularGUI.setResizable(false);
        frame_RegularGUI.setLocationRelativeTo(null);
        frame_RegularGUI.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame_RegularGUI.setVisible(true);
        frame_RegularGUI.setLocationRelativeTo(null);
    }

    // Dropout UIUX frame
    public void dropout_UIUX() {
        frame_DropoutGUI = new JFrame("Dropout Student");

        imageDropout= new ImageIcon("Images/c.jpg");

        panel_DropoutGUI = new JPanel();
        panel_DropoutGUI.setLayout(null);
        frame_DropoutGUI.add(panel_DropoutGUI);
```

```
label_DropoutGUI = new JLabel("Dropout Student");
lblenrollmentID = new JLabel("Enrollment ID:");
lblstudentName = new JLabel("Student Name:");
lblcourseName = new JLabel("Course Name:");
lblcourseDuration = new JLabel("Course Duration:");
lbldateOfEnrollement = new JLabel("Date Of Enrollment:");
lbldateOfBirth = new JLabel("Date of Birth:");
lbltuitionFee = new JLabel("Tuition Fee:");
lblNumOfRemainingModules = new JLabel("Remaining Modules:");
lbldateofDropout = new JLabel("Date of Dropout:");
lblNumOfMonthsAttended= new JLabel("Month Attended:");

txtenrollmentId = new JTextField();
txtstudentName = new JTextField();
txttuitionFee = new JTextField();
txtNumOfRemainingModules = new JTextField();
txtcourseDuration = new JTextField();
txtNumOfMonthsAttended = new JTextField();

txtdateOfDropoutDay = new JComboBox<>(Days());
txtdateOfDropoutMonth = new JComboBox<>(Months());
txtdateOfDropoutYear = new JComboBox<>(Years());

txtdateOfEnrollmentDay = new JComboBox<>(Days());
txtdateOfEnrollmentMonth = new JComboBox<>(Months());
txtdateOfEnrollmentYear = new JComboBox<>(Years());

txtdateOfBirthDay=  new JComboBox<>(Days());
```

```
txtdateOfBirthMonth= new JComboBox<>(Months());
txtdateOfBirthYear = new JComboBox<>(Years());
txtcourseName = new JComboBox<>(Courses());

submitDropout = new JButton("Submit");
resetDropout = new JButton("Clear");
backDropout = new JButton("Back");
btnPayBills = new JButton("Pay Bill");
removeStudent = new JButton("Remove");
btnDisplayDropout = new JButton("Display");

label_DropoutGUI.setBounds(400, 40, 513, 75);
lblenrollmentID.setBounds(47, 150, 222, 76);
lblstudentName.setBounds(47, 220, 222, 75);
lbldateOfBirth.setBounds(47, 290, 222, 75);
lblNumOfMonthsAttended.setBounds(47, 360, 322, 75);
lbltuitionFee.setBounds(47, 430, 222, 75);

lbldateOfEnrollement.setBounds(540, 150, 289, 52);
lblcourseName.setBounds(540, 220, 248, 73);
lblcourseDuration.setBounds(540, 290, 248, 60);
lblNumOfRemainingModules.setBounds(540, 360, 305, 68);
lbldateofDropout.setBounds(540, 430, 379,80);

txtenrollmentId.setBounds(279, 160, 210, 48);
txtstudentName.setBounds(279, 230, 210, 48);
txtNumOfMonthsAttended.setBounds(279, 370, 210, 48);
txttuitionFee.setBounds(279, 440, 200, 48);
```

```
txtdateOfBirthDay.setBounds(279, 300, 39, 48);
txtdateOfBirthMonth.setBounds(329, 300, 39, 48);
txtdateOfBirthYear.setBounds(379, 300, 100, 48);

txtdateOfEnrollmentDay.setBounds(830, 160, 39, 48);
txtdateOfEnrollmentMonth.setBounds(880, 160, 39, 48);
txtdateOfEnrollmentYear.setBounds(930, 160, 100, 48);

txtdateOfDropoutDay.setBounds(830, 440, 39, 48);
txtdateOfDropoutMonth.setBounds(880, 440, 39, 48);
txtdateOfDropoutYear.setBounds(930, 440, 100, 48);

txtcourseName.setBounds(830, 230, 217, 46);
txtcourseDuration.setBounds(830, 300, 217, 48);
txtNumOfRemainingModules.setBounds(830,370, 217, 48);

submitDropout.setBounds(46, 615, 140, 53);
resetDropout.setBounds(210,615,140,53);
btnPayBills.setBounds(390,615,140,53);
removeStudent.setBounds(570, 615, 140, 53);
btnDisplayDropout.setBounds(730,615,140,53);
backDropout.setBounds(890, 615, 140, 53);

label_DropoutGUI.setFont(new Font("Helvetica", Font.BOLD, 40));
lblenrollmentID.setFont(new Font("Helvetica", Font.BOLD, 30));
lblstudentName.setFont(new Font("Helvetica", Font.BOLD, 30));
lblcourseName.setFont(new Font("Helvetica", Font.BOLD, 30));
```

```
lblcourseDuration.setFont(new Font("Helvetica", Font.BOLD, 30));
lbldateOfEnrollement.setFont(new Font("Helvetica", Font.BOLD, 30));
lbldateOfBirth.setFont(new Font("Helvetica", Font.BOLD, 30));
lbltuitionFee.setFont(new Font("Helvetica", Font.BOLD, 30));
lblNumOfRemainingModules.setFont(new Font("Helvetica", Font.BOLD, 30));
lbldateofDropout.setFont(new Font("Helvetica", Font.BOLD, 30));
lblNumOfMonthsAttended.setFont(new Font("Helvetica", Font.BOLD, 30));

txtenrollmentId.setFont(new Font("Helvetica", Font.PLAIN, 14));
txtstudentName.setFont(new Font("Helvetica", Font.PLAIN, 14));
txtcourseName.setFont(new Font("Helvetica", Font.PLAIN, 14));
txtcourseDuration.setFont(new Font("Helvetica", Font.PLAIN, 14));
txtdateOfEnrollmentDay.setFont(new Font("Helvetica", Font.PLAIN, 14));
txtdateOfEnrollmentMonth.setFont(new Font("Helvetica", Font.PLAIN, 14));
txtdateOfEnrollmentYear.setFont(new Font("Helvetica", Font.PLAIN, 14));
txtdateOfBirthDay.setFont(new Font("Helvetica", Font.PLAIN, 14));
txtdateOfBirthMonth.setFont(new Font("Helvetica", Font.PLAIN, 14));
txtdateOfBirthYear.setFont(new Font("Helvetica", Font.PLAIN, 14));
txttuitionFee.setFont(new Font("Helvetica", Font.PLAIN, 14));
txtNumOfRemainingModules.setFont(new Font("Helvetica", Font.PLAIN, 14));
txtNumOfMonthsAttended.setFont(new Font("Helvetica", Font.PLAIN, 14));
txtdateOfDropoutDay.setFont(new Font("Helvetica", Font.PLAIN, 14));
txtdateOfDropoutMonth.setFont(new Font("Helvetica", Font.PLAIN, 14));
txtdateOfDropoutYear.setFont(new Font("Helvetica", Font.PLAIN, 14));

submitDropout.setFont(new Font("Helvetica", Font.BOLD, 20));
btnPayBills.setFont(new Font("Helvetica", Font.BOLD, 20));
btnDisplayDropout.setFont(new Font("Helvetica",Font.BOLD,20));
```

```
resetDropout.setFont(new Font("Helvetica", Font.BOLD, 20));
backDropout.setFont(new Font("Helvetica", Font.BOLD, 20));
removeStudent.setFont(new Font("Helvetica", Font.BOLD, 20));

submitDropout.setFocusable(false);
resetDropout.setFocusable(false);
backDropout.setFocusable(false);
btnPayBills.setFocusable(false);
btnDisplayDropout.setFocusable(false);
removeStudent.setFocusable(false);

submitDropout.addActionListener(this);
resetDropout.addActionListener(this);
backDropout.addActionListener(this);
btnPayBills.addActionListener(this);
btnDisplayDropout.addActionListener(this);
removeStudent.addActionListener(this);

panel_DropoutGUI.add(label_DropoutGUI);
panel_DropoutGUI.add(lblenrollmentID);
panel_DropoutGUI.add(lblstudentName);
panel_DropoutGUI.add(lblcourseName);
panel_DropoutGUI.add(lblcourseDuration);
panel_DropoutGUI.add(lbldateOfEnrollement);
panel_DropoutGUI.add(lbldateOfBirth);
panel_DropoutGUI.add(lbltuitionFee);
panel_DropoutGUI.add(lblNumOfRemainingModules);
panel_DropoutGUI.add(lbldateofDropout);
```

```
    panel_DropoutGUI.add(lblNumOfMonthsAttended);

    panel_DropoutGUI.add(txtenrollmentId);
    panel_DropoutGUI.add(txtstudentName);
    panel_DropoutGUI.add(txtcourseName);
    panel_DropoutGUI.add(txtcourseDuration);
    panel_DropoutGUI.add(txttuitionFee);
    panel_DropoutGUI.add(txtNumOfRemainingModules);
    panel_DropoutGUI.add(txtNumOfMonthsAttended);

    panel_DropoutGUI.add(txtdateOfEnrollmentDay);
    panel_DropoutGUI.add(txtdateOfEnrollmentMonth);
    panel_DropoutGUI.add(txtdateOfEnrollmentYear);

    panel_DropoutGUI.add(txtdateOfBirthDay);
    panel_DropoutGUI.add(txtdateOfBirthMonth);
    panel_DropoutGUI.add(txtdateOfBirthYear);

    panel_DropoutGUI.add(txtdateOfDropoutDay);
    panel_DropoutGUI.add(txtdateOfDropoutMonth);
    panel_DropoutGUI.add(txtdateOfDropoutYear);

    panel_DropoutGUI.add(submitDropout);
    panel_DropoutGUI.add(resetDropout);
    panel_DropoutGUI.add(backDropout);
    panel_DropoutGUI.add(btnPayBills);
    panel_DropoutGUI.add(btnDisplayDropout);
    panel_DropoutGUI.add(removeStudent);
```

```
        frame_DropoutGUI.setIconImage(imageDropout.getImage());
        frame_DropoutGUI.setSize(1080, 720);
        frame_DropoutGUI.setResizable(false);
        frame_DropoutGUI.setLocationRelativeTo(null);
        frame_DropoutGUI.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame_DropoutGUI.setVisible(true);
        frame_DropoutGUI.setLocationRelativeTo(null);
    }
    //resistering regular student
    public void addRegular(){
        //Taking Vlaue In String
        String enrollmentIDText= txtEnrollmentID.getText();
        String studentName = txtStudentName.getText();
        String dateOfBirthText = txtDateofBirthDay.getSelectedItem()+"-"+(txtDateofBirthMonth.getSelectedIndex()+1)+ "-" + txtDateofBirthYear.getSelectedItem();
        String daysPresentText = txtDaysPresent.getText();
        String tuitionFeeText = txtTuitionFee.getText();

        String dateOfEnrollmentText = txtDateofEnrollmentDay.getSelectedItem()+"-"+(txtDateofEnrollmentMonth.getSelectedIndex() +1)+ "-" + txtDateofEnrollmentYear.getSelectedItem();
        String courseNameRegular = txtCourseName.getSelectedItem().toString();
        String courseDurationText = txtCourseDuration.getText();
        String numberOfModulesText = txtNumOfModules.getText();
        String numberOfCredithoursText = txtNumOfCreditHours.getText();
        try{
            //Checking if any of the required file is empty
            if(enrollmentIDText.isEmpty()||studentName.isEmpty()||
    dateOfBirthText.isEmpty()|| daysPresentText.isEmpty()|| tuitionFeeText.isEmpty()||

            dateOfEnrollmentText.isEmpty()||
    courseNameRegular.isEmpty()||courseDurationText.isEmpty()||
    numberOfModulesText.isEmpty()|| numberOfCredithoursText.isEmpty())

            {

               JOptionPane.showMessageDialog(null,"Please fill all the fields.","Alert",
    JOptionPane.WARNING_MESSAGE);

            }else

            {

               //Parse numeric values from the text fields

               int enrollmentID = Integer.parseInt(enrollmentIDText);

               int courseDuration = Integer.parseInt(courseDurationText);

               double tuitionFee = Double.parseDouble(tuitionFeeText);

               int numOfModules = Integer.parseInt(numberOfModulesText);

               int numOfCreditHours = Integer.parseInt(numberOfCredithoursText);

               int daysPresent = Integer.parseInt(daysPresentText);


               //Checking if the numberical textfield contains only numbers no alphabets

               if(!enrollmentIDText.matches("[0-9]+") || !courseDurationText.matches("[0-
    9]+") ||

               !tuitionFeeText.matches("[0-9.]+") || !numberOfModulesText.matches("[0-9]+")
    ||

               !numberOfCredithoursText.matches("[0-9]+") || !daysPresentText.matches("[0-
    9]+")){

                  JOptionPane.showMessageDialog(null,"Invalid inputs. Please enter valid
    numeric inputs.","Alert",JOptionPane.WARNING_MESSAGE);


               }else {

                  //Check wheather student with same EnrollmentID already resister

                  boolean isAlreadyResistered = false;
            for(Student student : studentList){
                if(student instanceof Regular ){
                    if ( student.getEnrollmentID() == enrollmentID){
                        JOptionPane.showMessageDialog(null,"The student is already registered.","Alert",JOptionPane.WARNING_MESSAGE);
                        isAlreadyResistered = true;
                        break;
                    }
                }
            }
            if(isAlreadyResistered == false){
                //Create and add the Regular student to the list

                Regular regularStudent = new Regular(enrollmentID, dateOfBirthText, courseNameRegular, studentName, dateOfEnrollmentText,
                        courseDuration, tuitionFee, numOfModules, numOfCreditHours, daysPresent);
                studentList.add(regularStudent);

                String registeredInformation = "Enrollment ID: " + enrollmentID + "\n" +
                    "Name: " + studentName + "\n" +
                    "Date of Birth: " + dateOfBirthText + "\n" +
                    "Days Present: " + daysPresent + "\n" +
                    "Tuition Fee: " + tuitionFee + "\n" +
                    "Date of Enrollment: " + dateOfEnrollmentText + "\n" +
                    "Course Name: " + courseNameRegular + "\n" +
                    "Course Duration: " + courseDuration + " months\n" +
                    "Number of Modules: " + numOfModules + "\n" +
                    "Number of Credit Hours: " + numOfCreditHours;
```

```
                        JOptionPane.showMessageDialog(null, registeredInformation, "Student Information", JOptionPane.INFORMATION_MESSAGE);
                        JOptionPane.showMessageDialog(null, "Thank you for registering.", "Info", JOptionPane.INFORMATION_MESSAGE);


                    }


                }


            }
        }
        catch (NumberFormatException e) {
            JOptionPane.showMessageDialog(null, "Please enter valid numeric values for course duration, tuition fee, number of modules, number of credit hours, and days present.", "Alert", JOptionPane.WARNING_MESSAGE);
        } catch (Exception ex) {
            // Handle any other exceptions that might occur during the parsing
            JOptionPane.showMessageDialog(null, "An error occurred. Please try again.");
        }
    }

    //add Dropout
    public void submitDropout()
    {
        String enrollmentIDText = txtenrollmentId.getText();
        String studentName = txtstudentName.getText();
        String courseDurationText = txtcourseDuration.getText();
        String dateOfEnrollment = txtdateOfEnrollmentDay.getSelectedItem() + "-" +
    (txtdateOfEnrollmentMonth.getSelectedIndex() + 1) + "-" +
    txtdateOfEnrollmentYear.getSelectedItem();

        String courseName = txtcourseName.getSelectedItem().toString();

        String dateOfBirth = txtdateOfBirthDay.getSelectedItem() + "-" +
    (txtdateOfBirthMonth.getSelectedIndex() + 1) + "-" +
    txtdateOfBirthYear.getSelectedItem();

        String dateOfDropout = txtdateOfDropoutDay.getSelectedItem() + "-" +
    (txtdateOfDropoutMonth.getSelectedIndex() + 1) + "-" +
    txtdateOfDropoutYear.getSelectedItem();

        String tuitionFeeText = txttuitionFee.getText();

        String numOfRemainingModulesText = txtNumOfRemainingModules.getText();

        String numOfMonthsAttendedText = txtNumOfMonthsAttended.getText();


        try {

            // Check if any of the required fields are empty

            if (enrollmentIDText.isEmpty() || studentName.isEmpty() || courseName.isEmpty()
    || courseDurationText.isEmpty() ||

            dateOfEnrollment.isEmpty() || dateOfBirth.isEmpty() || dateOfDropout.isEmpty() ||

            tuitionFeeText.isEmpty() || numOfRemainingModulesText.isEmpty() ||
    numOfMonthsAttendedText.isEmpty()) {


                JOptionPane.showMessageDialog(null, "Please fill all the fields!", "Alert",
    JOptionPane.WARNING_MESSAGE);

            } else {

                // Parse numeric values from the text fields

                int enrollmentID = Integer.parseInt(enrollmentIDText);

                int courseDuration = Integer.parseInt(courseDurationText);

                double tuitionFee = Double.parseDouble(tuitionFeeText);

                int numOfRemainingModules =
    Integer.parseInt(numOfRemainingModulesText);

                int numOfMonthsAttended = Integer.parseInt(numOfMonthsAttendedText);
                // Check if enrollmentID and courseDuration contain only numbers (no alphabet
    or special characters)
                if (!enrollmentIDText.matches("[0-9]+") || !courseDurationText.matches("[0-
    9]+") || !tuitionFeeText.matches("[0-9.]+") ||
                !numOfRemainingModulesText.matches("[0-9]+") ||
    !numOfMonthsAttendedText.matches("[0-9]+")) {

              JOptionPane.showMessageDialog(null, "Invalid inputs. Please enter valid
    numeric inputs.", "Alert", JOptionPane.WARNING_MESSAGE);
                } else {
              // Check if the student with the same enrollment ID is already registered as
    a dropout
              boolean isAlreadyRegistered = false;
              for (Student student : studentList) {
                if (student.getEnrollmentID() == enrollmentID) {
                  JOptionPane.showMessageDialog(null, "The student is already
    registered as a dropout.", "Alert", JOptionPane.WARNING_MESSAGE);
                  isAlreadyRegistered = true;
                  break;
                }
              }

              if (!isAlreadyRegistered) {
                // Create and add the Dropout student to the list
                Dropout dropoutStudent = new Dropout(dateOfBirth, studentName,
    courseDuration, tuitionFee,
                    numOfRemainingModules, numOfMonthsAttended, dateOfDropout,
    enrollmentID, courseName, dateOfEnrollment);
                studentList.add(dropoutStudent);
                String registeredInformation = "Enrollment ID: " + enrollmentID + "\n" +
                  "Name: " + studentName + "\n" +
                  "Date of Birth: " + dateOfBirth + "\n" +
                  "Tuition Fee: " + tuitionFee + "\n" +
                  "Date of Enrollment: " + dateOfEnrollment + "\n" +
                  "Course Name: " + courseName + "\n" +
                  "Course Duration: " + courseDuration + " months\n" +
                  "Number of Remaining Modules: " + numOfRemainingModules + "\n" +
                  "Number of Months Attended: " + numOfMonthsAttended + "\n" +
                  "Date of Dropout:" + dateOfDropout;

                JOptionPane.showMessageDialog(null, registeredInformation, "Student Information", JOptionPane.INFORMATION_MESSAGE);
                JOptionPane.showMessageDialog(null, "Thank you for registering.", "Info", JOptionPane.INFORMATION_MESSAGE);
              }
                }
              }
            } catch (NumberFormatException e) {
                JOptionPane.showMessageDialog(null, "Please enter valid numeric values for course duration, tuition fee, number of remaining modules, and number of months attended.", "Alert", JOptionPane.WARNING_MESSAGE);
            } catch (Exception ex) {
                // Handle any other exceptions that might occur during the parsing
                JOptionPane.showMessageDialog(null, "An error occurred. Please try again.");
            }
        }

        //Clear Regular Field
        public void clearRegular(){
            txtEnrollmentID.setText("");
            txtStudentName.setText("");
            txtCourseName.setSelectedIndex(0);
            txtCourseDuration.setText("");
            txtDateofEnrollmentDay.setSelectedIndex(0);
            txtDateofEnrollmentMonth.setSelectedIndex(0);
            txtDateofEnrollmentYear.setSelectedIndex(0);
            txtDateofBirthDay.setSelectedIndex(0);
            txtDateofBirthMonth.setSelectedIndex(0);
            txtDateofBirthYear.setSelectedIndex(0);
            txtTuitionFee.setText("");
            txtNumOfModules.setText("");
            txtNumOfCreditHours.setText("");
            txtDaysPresent.setText("");
        }
        // Clear Dropout Fields
        public void clearDropout() {
            txtenrollmentId.setText("");
            txtstudentName.setText("");
            txtcourseName.setSelectedIndex(0);
            txtcourseDuration.setText("");
            txtdateOfEnrollmentDay.setSelectedIndex(0);
            txtdateOfEnrollmentMonth.setSelectedIndex(0);
            txtdateOfEnrollmentYear.setSelectedIndex(0);
            txtdateOfBirthDay.setSelectedIndex(0);
            txtdateOfBirthMonth.setSelectedIndex(0);
            txtdateOfBirthYear.setSelectedIndex(0);
            txtdateOfDropoutDay.setSelectedIndex(0);
            txtdateOfDropoutMonth.setSelectedIndex(0);
            txtdateOfDropoutYear.setSelectedIndex(0);
            txttuitionFee.setText("");
            txtNumOfRemainingModules.setText("");
            txtNumOfMonthsAttended.setText("");


        }
        //calculate present percentage
        public void presentPercentage(){
            try {
                // Parse the numeric value for daysPresent from the text field
                double daysPresent = Double.parseDouble(txtDaysPresent.getText());


                // Parse other required values from text fields
                int enrollmentID = Integer.parseInt(txtEnrollmentID.getText());
                String dateOfBirth = txtDateofBirthDay.getSelectedItem()+"-
    "+(txtDateofBirthMonth.getSelectedIndex() + 1) + "-" +
    txtDateofBirthYear.getSelectedItem();
                String courseName = txtCourseName.getSelectedItem().toString();
                String studentName = txtStudentName.getText();
                String dateOfEnrollment =txtDateofEnrollmentDay.getSelectedItem()+"-
    "+(txtDateofEnrollmentMonth.getSelectedIndex() + 1) + "-" +
    txtDateofBirthYear.getSelectedItem();
                int courseDuration = Integer.parseInt(txtCourseDuration.getText());
                double tuitionFee = Double.parseDouble(txtTuitionFee.getText());
                int numOfModules = Integer.parseInt(txtNumOfModules.getText());
                int numOfCreditHours = Integer.parseInt(txtNumOfCreditHours.getText());


                // Regular class with the required parameters
                Regular regularStudent = new Regular(enrollmentID, dateOfBirth, courseName, studentName, dateOfEnrollment, courseDuration, tuitionFee, numOfModules, numOfCreditHours, daysPresent);


                // Call the presentPercentage method of Regular class and get the result
                char result = regularStudent.presentPercentage(daysPresent);


                // Show the result in a dialog box
                String message = ("Present Percentage: " + result + "%");
                JOptionPane.showMessageDialog(frame_StudentGUI, "Present Percentage: " + message , "Present Percentage", JOptionPane.INFORMATION_MESSAGE);
            } catch (NumberFormatException ex) {
                //Handle numerical exception
                JOptionPane.showMessageDialog(null, "Please enter a valid numeric value for daysPresent.", "Alert", JOptionPane.WARNING_MESSAGE);
            } catch (Exception ex) {
                // Handle any other exceptions that might occur during the process
                JOptionPane.showMessageDialog(null, "An error occurred. Please try again.", "Alert", JOptionPane.ERROR_MESSAGE);
            }


        }


        public void btnGrandCertificate(){
            try {
                String enrollmentIDText = JOptionPane.showInputDialog(null, "Enter the Enrollment ID of the Student:", "Grant Certificate", JOptionPane.QUESTION_MESSAGE);


                // Check if the user clicked "OK" and entered a value
                if (enrollmentIDText != null && !enrollmentIDText.isEmpty()) {
```

```
int enrollmentID = Integer.parseInt(enrollmentIDText);

// Find the Regular student with the given enrollmentID
Regular regularStudent = null;
for (Student student : studentList) {
    if (student instanceof Regular){
        if(student.getEnrollmentID() == enrollmentID) {
            regularStudent = (Regular) student;
            break;
        }
    }}

// If the Regular student with the given enrollmentID is found
if (regularStudent != null) {
    String courseName = regularStudent.getCourseName();
    String dateOfEnrollment = regularStudent.getDateOfEnrollment();

    // Calculate the present percentage
    double daysPresent = regularStudent.getDaysPresent();
    char presentPercentage = regularStudent.presentPercentage(daysPresent);

    // Check if the present percentage is "A" or 80%
    if (presentPercentage == 'A' || daysPresent >= 80) {
        regularStudent.grantCertificate(courseName, enrollmentID,
dateOfEnrollment);

        boolean isGrantedScholarship =
regularStudent.getIsGrantedScholarship();
```

```
                String scholarshipStatus = isGrantedScholarship ? "Granted" : "Not Granted";

                JOptionPane.showMessageDialog(null, "Certificate granted for Enrollment ID: " + enrollmentID + "\nScholarship Status: " + scholarshipStatus, "Success", JOptionPane.INFORMATION_MESSAGE);
              } else {
                JOptionPane.showMessageDialog(null, "Scholarship is not granted. Present percentage is below 80%.", "Not Granted", JOptionPane.WARNING_MESSAGE);
              }
            } else {
              JOptionPane.showMessageDialog(null, "No Regular student found with the Enrollment ID: " + enrollmentID, "Not Found", JOptionPane.WARNING_MESSAGE);
            }
          } else {
            // Show a message if the user canceled or entered an empty value
            JOptionPane.showMessageDialog(null, "Please enter a valid Enrollment ID.", "Alert", JOptionPane.WARNING_MESSAGE);
          }
        } catch (NumberFormatException e) {
          JOptionPane.showMessageDialog(null, "Please enter a valid numeric value for Enrollment ID.", "Alert", JOptionPane.WARNING_MESSAGE);
        } catch (Exception ex) {
          JOptionPane.showMessageDialog(null, "An error occurred. Please try again.", "Alert", JOptionPane.ERROR_MESSAGE);
        }
    }

    //display
    public void display(){
        // Create a new DefaultTableModel with column headers
        DefaultTableModel model = new DefaultTableModel();
        model.setColumnIdentifiers(new String[]{"Enrollment ID", "Student Name", "Course
Name", "Course Duration",
                "Date of Enrollment", "Date of Birth", "Tuition Fee", "Student Type"});

        // Add data to the model from the studentList
        for (Student student : studentList) {
            model.addRow(new Object[]{
                    student.getEnrollmentID(),
                    student.getStudentName(),
                    student.getCourseName(),
                    student.getCourseDuration(),
                    student.getDateOfEnrollment(),
                    student.getDateOfBirth(),
                    student.getTuitionFee(),
                    student instanceof Regular ? "Regular Student" : "Dropout Student"
            });
        }

        // Create a JTable with the model
        table = new JTable(model);
        table.setAutoResizeMode(JTable.AUTO_RESIZE_ALL_COLUMNS);
        table.setFillsViewportHeight(true);

        // Wrap the JTable in a JScrollPane to enable scrolling
        JScrollPane scroll = new JScrollPane(table);
```

```
scroll.setHorizontalScrollBarPolicy(JScrollPane.HORIZONTAL_SCROLLBAR_AS_NEE
DED);

scroll.setVerticalScrollBarPolicy(JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED);

        // Create a panel and add the JScrollPane to it
        JPanel panelDisplay = new JPanel(new BorderLayout());
        panelDisplay.add(scroll, BorderLayout.CENTER);

        // Create "Remove Dropout" and "Clear" buttons

        btnClear = new JButton("Clear");

        // Add action listeners to the buttons
        btnClear.addActionListener(this);

        // Create a panel for buttons
        JPanel panelButtons = new JPanel();
        panelButtons.setLayout(new FlowLayout());
        panelButtons.add(btnClear);

        // Add the buttons panel to the main panel
        panelDisplay.add(panelButtons, BorderLayout.SOUTH);

        // Create a new JFrame to display the panel with the table
        JFrame displayFrame = new JFrame("Student Records");
        displayFrame.setSize(800, 800);
        displayFrame.setLocationRelativeTo(null);
        displayFrame.add(panelDisplay);
        displayFrame.setVisible(true);

        btnClear.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                studentList.clear();
                DefaultTableModel model = (DefaultTableModel) table.getModel();
                model.setRowCount(0);
                JOptionPane.showMessageDialog(displayFrame, "List Cleared
Successfully");
            }
          });

    }
    // Remove Dropout Student
    public void removeDropout() {
        String enrollmentIDText = JOptionPane.showInputDialog(null, "Enter the Enrollment
ID of the student to remove:", "Remove Student", JOptionPane.QUESTION_MESSAGE);

        try {
          // Check if the enrollmentID field is not empty and the user clicked "OK"
          if (enrollmentIDText != null && !enrollmentIDText.isEmpty()) {
            // Parse the enrollmentID value from the user input
            int enrollmentID = Integer.parseInt(enrollmentIDText);

            // Find the Dropout student with the given enrollmentID
            Dropout dropoutStudent = null;
            for (Student student : studentList) {
                if (student instanceof Dropout && student.getEnrollmentID() == enrollmentID) {
                    dropoutStudent = (Dropout) student;
                    break;
                }
            }

            // If the student with the given enrollmentID is found, remove it from the list
            if (dropoutStudent != null) {
                studentList.remove(dropoutStudent);
                JOptionPane.showMessageDialog(null, "Student with Enrollment ID: " + enrollmentID + " has been removed.", "Success", JOptionPane.INFORMATION_MESSAGE);
            } else {
                JOptionPane.showMessageDialog(null, "No Dropout student found with the Enrollment ID: " + enrollmentID, "Not Found", JOptionPane.WARNING_MESSAGE);
            }
        } else {
            // Show a message if the user canceled or entered an empty value
            JOptionPane.showMessageDialog(null, "Please enter a valid Enrollment ID to remove the student.", "Alert", JOptionPane.WARNING_MESSAGE);
        }
    } catch (NumberFormatException e) {
        JOptionPane.showMessageDialog(null, "Please enter a valid numeric value for Enrollment ID.", "Alert", JOptionPane.WARNING_MESSAGE);
    } catch (Exception ex) {
        JOptionPane.showMessageDialog(null, "An error occurred. Please try again.", "Alert", JOptionPane.ERROR_MESSAGE);
    }
}
//Pay bills
```

```
  public void payBills(){
      try {
          // Ask for Enrollment ID
          String enrollmentIDText = JOptionPane.showInputDialog(null, "Enter Enrollment
ID:", "Pay Bills", JOptionPane.PLAIN_MESSAGE);
          if (enrollmentIDText == null || enrollmentIDText.isEmpty()) {
              JOptionPane.showMessageDialog(null, "Enrollment ID cannot be empty.",
"Alert", JOptionPane.WARNING_MESSAGE);
              return;
          }
          int enrollmentID = Integer.parseInt(enrollmentIDText);

          // Dropout student with the given enrollmentID
          Dropout dropoutStudent = null;
          for (Student student : studentList) {
              if (student instanceof Dropout && student.getEnrollmentID() == enrollmentID) {
                  dropoutStudent = (Dropout) student;
                  break;
              }
          }

          // If the student with the given enrollmentID is found, ask for Tuition Fee and
Amount Paid
          if (dropoutStudent != null) {
              // Ask for Tuition Fee
              String tuitionFeeText = JOptionPane.showInputDialog(null, "Enter Tuition
Fee:", "Pay Bills", JOptionPane.PLAIN_MESSAGE);
              if (tuitionFeeText == null || tuitionFeeText.isEmpty()) {
                  JOptionPane.showMessageDialog(null, "Tuition Fee cannot be empty.",
"Alert", JOptionPane.WARNING_MESSAGE);
```

```
            return;
        }

        // Ask for Amount Paid
        String amountPaidText = JOptionPane.showInputDialog(null, "Enter Amount Paid:", "Pay Bills", JOptionPane.PLAIN_MESSAGE);
        if (amountPaidText == null || amountPaidText.isEmpty()) {
            JOptionPane.showMessageDialog(null, "Amount Paid cannot be empty.", "Alert", JOptionPane.WARNING_MESSAGE);
            return;
        }
        double amountPaidByStudent = Double.parseDouble(amountPaidText);

        boolean isPaid = dropoutStudent.billsPayable(amountPaidByStudent); // Pass the amountPaid argument here

        if (isPaid) {
            JOptionPane.showMessageDialog(null, "Bills paid successfully for the student with Enrollment ID: " + enrollmentID, "Success", JOptionPane.INFORMATION_MESSAGE);

            // Check if the remaining amount is fully paid
            boolean isFullyPaid = dropoutStudent.getRemainingAmount() == 0;
            if (isFullyPaid) {
                JOptionPane.showMessageDialog(null, "The student has fully paid the remaining amount.", "Info", JOptionPane.INFORMATION_MESSAGE);
            }
        } else {
            JOptionPane.showMessageDialog(null, "Bills are not fully paid for the student with Enrollment ID: " + enrollmentID, "Info", JOptionPane.INFORMATION_MESSAGE);
            }
        } else {
            JOptionPane.showMessageDialog(null, "No Dropout student found with the Enrollment ID: " + enrollmentID, "Not Found", JOptionPane.WARNING_MESSAGE);
        }
    } catch (NumberFormatException e) {
        JOptionPane.showMessageDialog(null, "Please enter a valid numeric value for Enrollment ID, Tuition Fee, and Amount Paid.", "Alert", JOptionPane.WARNING_MESSAGE);
    } catch (Exception ex) {
        JOptionPane.showMessageDialog(null, "An error occurred. Please try again.", "Alert", JOptionPane.ERROR_MESSAGE);
    }
}

@Override
public void actionPerformed(ActionEvent e) {
    if (e.getSource() == btnRegular) {
        regular_UIUX();
        frame_StudentGUI.dispose();
    }else if (e.getSource() == btnDropout) {
        dropout_UIUX();
        frame_StudentGUI.dispose();
    } else if (e.getSource() == regSubmit) {
        addRegular();
    } else if (e.getSource() == reset) {
        clearRegular();;
    } else if (e.getSource() == backRegular) {
        frame_StudentGUI.setVisible(true);
        frame_RegularGUI.dispose();
    } else if (e.getSource() == submitDropout) {
        submitDropout();

    } else if (e.getSource() == resetDropout) {
        clearDropout();
    } else if (e.getSource() == backDropout) {
        frame_StudentGUI.setVisible(true);
        frame_DropoutGUI.dispose();

    } else if (e.getSource() == btnDisplayRegular) {
        display();

    } else if (e.getSource() == btnDisplayDropout){
        display();
    }else if (e.getSource() == removeStudent) {
        removeDropout();
    } else if (e.getSource() == btnPayBills) {
        payBills();

    }else if (e.getSource() == btnPresentPercentage){
        presentPercentage();

    }else if (e.getSource() == btnGrantCertificate){
        btnGrandCertificate();

    }
}
// String Courses
public String[] Courses() {
    return new String[] {"Computing","Networking","Multimedia"};
}
//String for Years
public String[] Years() {
    return new String[] { "2000", "2001", "2002", "2003", "2004", "2005", "2006", "2007",
"2008", "2009",
        "2010", "2011", "2012", "2013", "2014", "2015", "2016", "2017", "2018", "2019",
        "2020", "2021", "2022", "2023"};
}
//String for Months
public String[] Months() {
    return new String[] { "1","2","3","4","5","6","7","8","9","10","11","12"};
}
//String for Days
public String[] Days() {
    return new String[]
{"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"};
}
//Main Method
public static void main(String[] args) {
    new Student_GUI();
}}
```