

# Education Project Report

## Description:

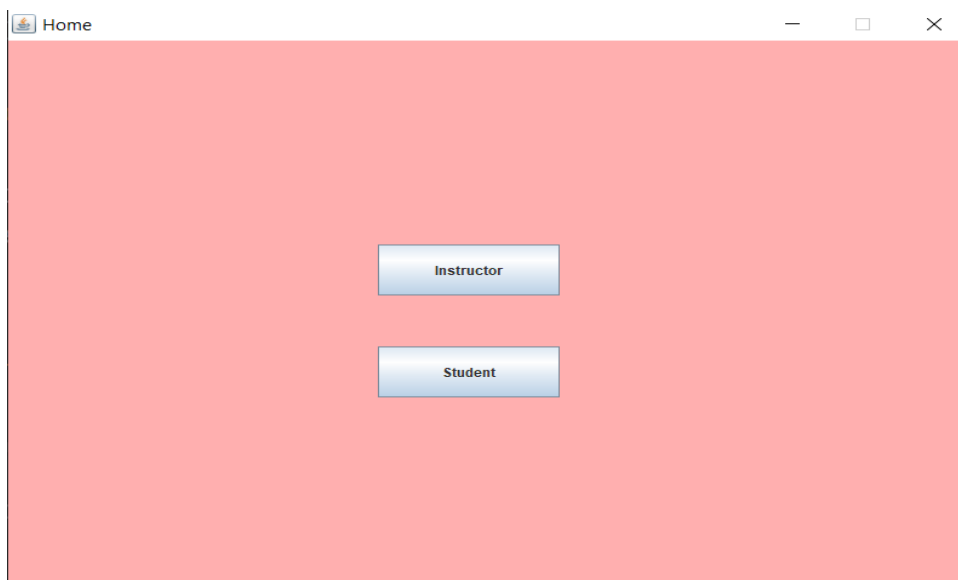
GUI Application using Java that connect Instructors and Students together such as LMS. Each Instructor manage specific course, instructor can add content to the course, add exams to the courses and search for student by his ID to show all his information (Name, Registered Courses, Grades in each course, GPA). Secondly student can register in the courses then show its content and finally take exams and system will compute his grade in the exam and Total Grade (GPA).

## Project Component:

### Development

#### 1-Main Frame:

Only screen have 2 buttons **student** to access student frame and **instructor** to access instructor frame.



## Code:

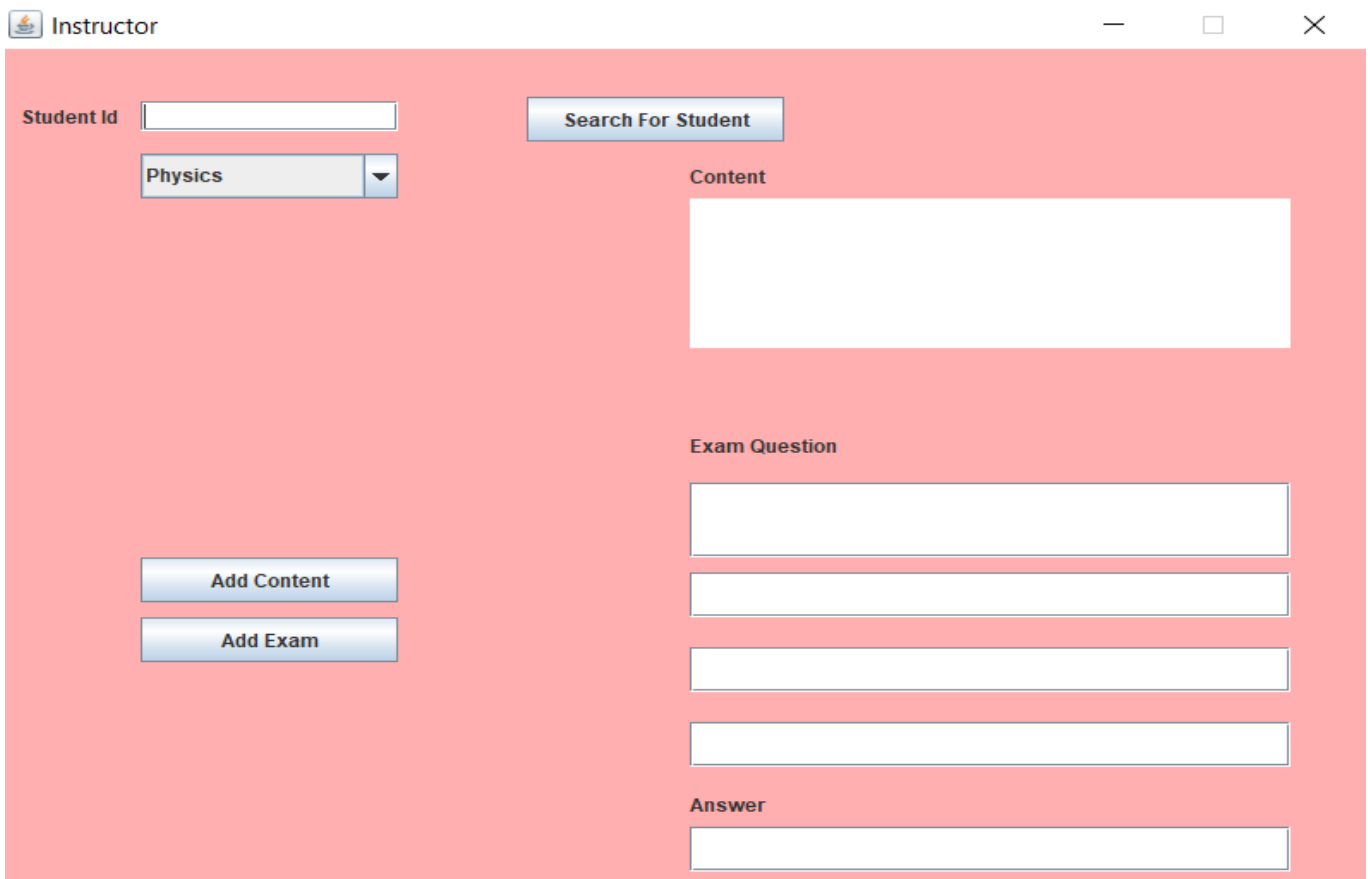
```
public void ShowHomeFrame() {
    Student = new JButton( text: "Student");
    Instructor = new JButton( text: "Instructor");
    HomePanel = new JPanel();
    ///////////////////////////////////////////////////
    this.setTitle("Home");
    this.setSize( width: 800, height: 600);
    HomePanel.setSize( width: 800, height: 600);
    this.setVisible(true);
    HomePanel.setVisible(true);
    this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    this.setLayout(null);
    HomePanel.setLayout(null);
    HomePanel.setBackground(Color.PINK);
    this.setResizable(false);
    this.setLocation( x: 700, y: 200);
    this.add(HomePanel);
    ///////////////////////////////////////////////////

    Student.setBounds( x: 305, y: 300, width: 150, height: 50);
    HomePanel.add(Student);
    Instructor.setBounds( x: 305, y: 200, width: 150, height: 50);
    HomePanel.add(Instructor);
    Student.addActionListener( l: this);
    Instructor.addActionListener( l: this);
}
```

## 2-Instructor Frame:

Screen contain all functions of instructor. Text box for writing id of student and click search.

Text area for adding content to the course. Five text boxes for adding question and 4 choices of it and the last text box is for the answer.



The screenshot shows a Java Swing window titled "Instructor" with a pink background. The window contains the following elements:

- Student Id:** A text input field.
- Search For Student:** A button.
- Physics:** A dropdown menu.
- Content:** A large text area.
- Exam Question:** Five text input fields.
- Answer:** A text input field.
- Add Content:** A button.
- Add Exam:** A button.

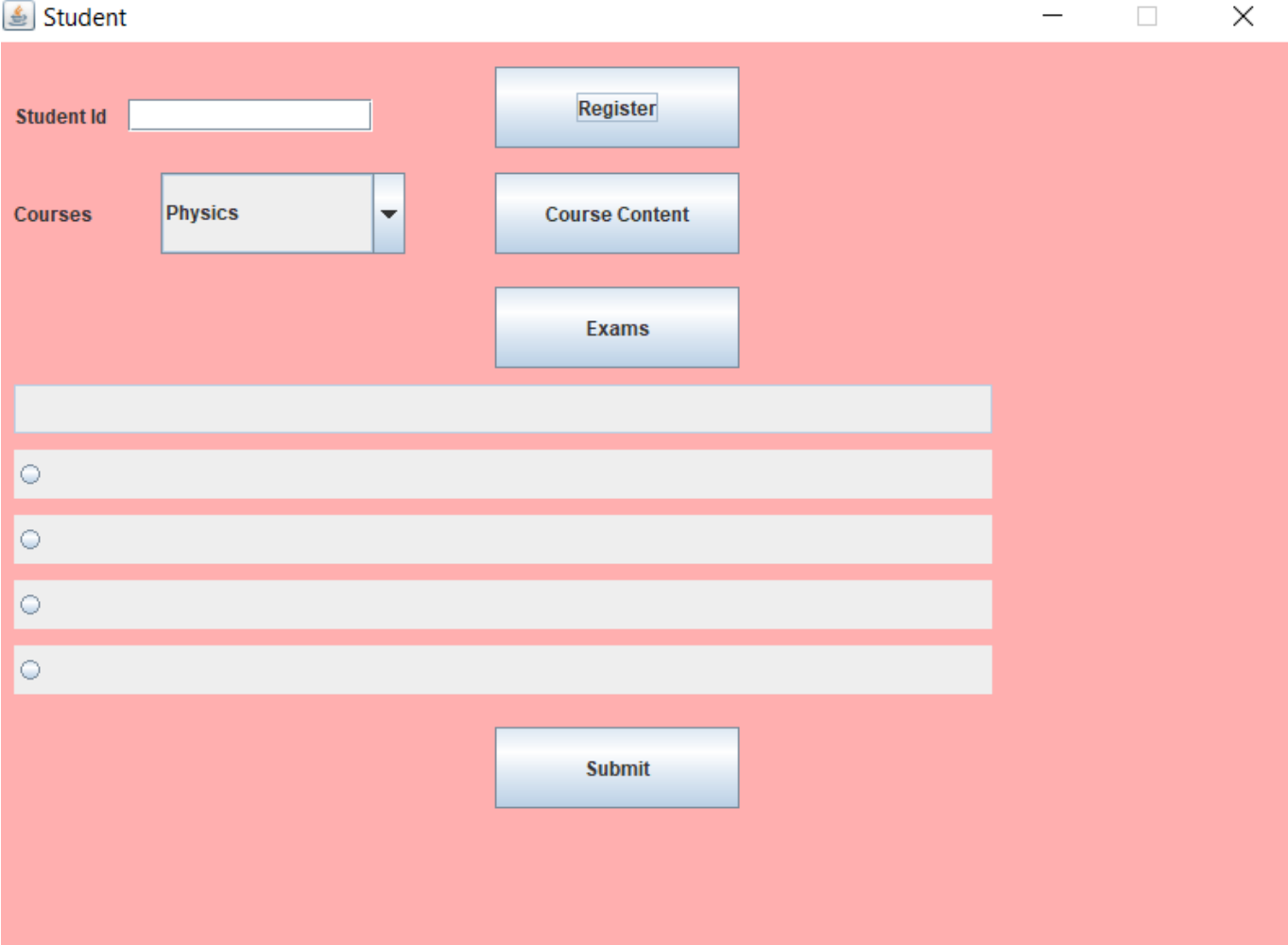
## Code:

```
public void ShowInstructorFrame(String course1 ,String course2 , String course3 , String course4) {

    ContentB3 = new JButton("Add Content");
    ExamB3= new JButton("Add Exam");
    SearchForStudent = new JButton("Search For Student");
    StudentIdText = new JTextField();
    Text = new JTextArea();
    Exam = new JTextField();
    c1 = new JTextField();
    c2 = new JTextField();
    c3 = new JTextField();
    Answer= new JTextField();
    StudentIdLabel = new JLabel("Student Id");
    Content = new JLabel("Content");
    Ex = new JLabel("Exam Question");
    Ans = new JLabel("Answer");
    InstructorPanel = new JPanel();
    combo = new JComboBox();
    //////////////////////////////////////
    this.setTitle("Instructor");
    this.setSize(800, 600);
    InstructorPanel.setSize(800,600);
    this.setVisible(true);
    InstructorPanel.setVisible(true);
    this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    this.setLayout(null);
    InstructorPanel.setLayout(null);
    InstructorPanel.setBackground(Color.PINK);
    this.setResizable(false);
    this.setLocation(700, 200);
    this.add(InstructorPanel);
    //////////////////////////////////////
    StudentIdLabel.setBounds(10, 40, 100, 10);
    InstructorPanel.add(StudentIdLabel);
    StudentIdText.setBounds(80, 35, 150, 20);
    InstructorPanel.add(StudentIdText);
    SearchForStudent.setBounds(305, 32, 150, 30);
    InstructorPanel.add(SearchForStudent);
    combo.setBounds(80, 70, 150, 30);
    combo.addItem(course1);
    combo.addItem(course2);
    combo.addItem(course3);
    combo.addItem(course4);
    InstructorPanel.add(combo);
    ContentB3.setBounds(80, 340, 150, 30);
    InstructorPanel.add(ContentB3);
    ExamB3.setBounds(80, 380, 150, 30);
    InstructorPanel.add(ExamB3);
    Content.setBounds(400, 70, 350, 30);
    InstructorPanel.add(Content);
    Text.setBounds(400, 100, 350, 100);
    InstructorPanel.add(Text);
    Ex.setBounds(400, 250, 350, 30);
    InstructorPanel.add(Ex);
    Exam.setBounds(400, 290, 350, 50);
    InstructorPanel.add(Exam);
    c1.setBounds(400, 350, 350, 30);
    InstructorPanel.add(c1);
    c2.setBounds(400, 400, 350, 30);
    InstructorPanel.add(c2);
    c3.setBounds(400, 450, 350, 30);
    InstructorPanel.add(c3);
    Ans.setBounds(400, 490, 350, 30);
    InstructorPanel.add(Ans);
    Answer.setBounds(400, 520, 350, 30);
    InstructorPanel.add(Answer);
    SearchForStudent.addActionListener(this);
    ContentB3.addActionListener(this);
    ExamB3.addActionListener(this);
}
```

### 3-Student Frame:

Firstly, student write his id and choose courses which he needs to register and click register. Then he can show the course content and click on exam and take the exam of the chosen course.



The image shows a software window titled "Student" with standard window controls (minimize, maximize, close). The window has a light red background. It contains the following elements:

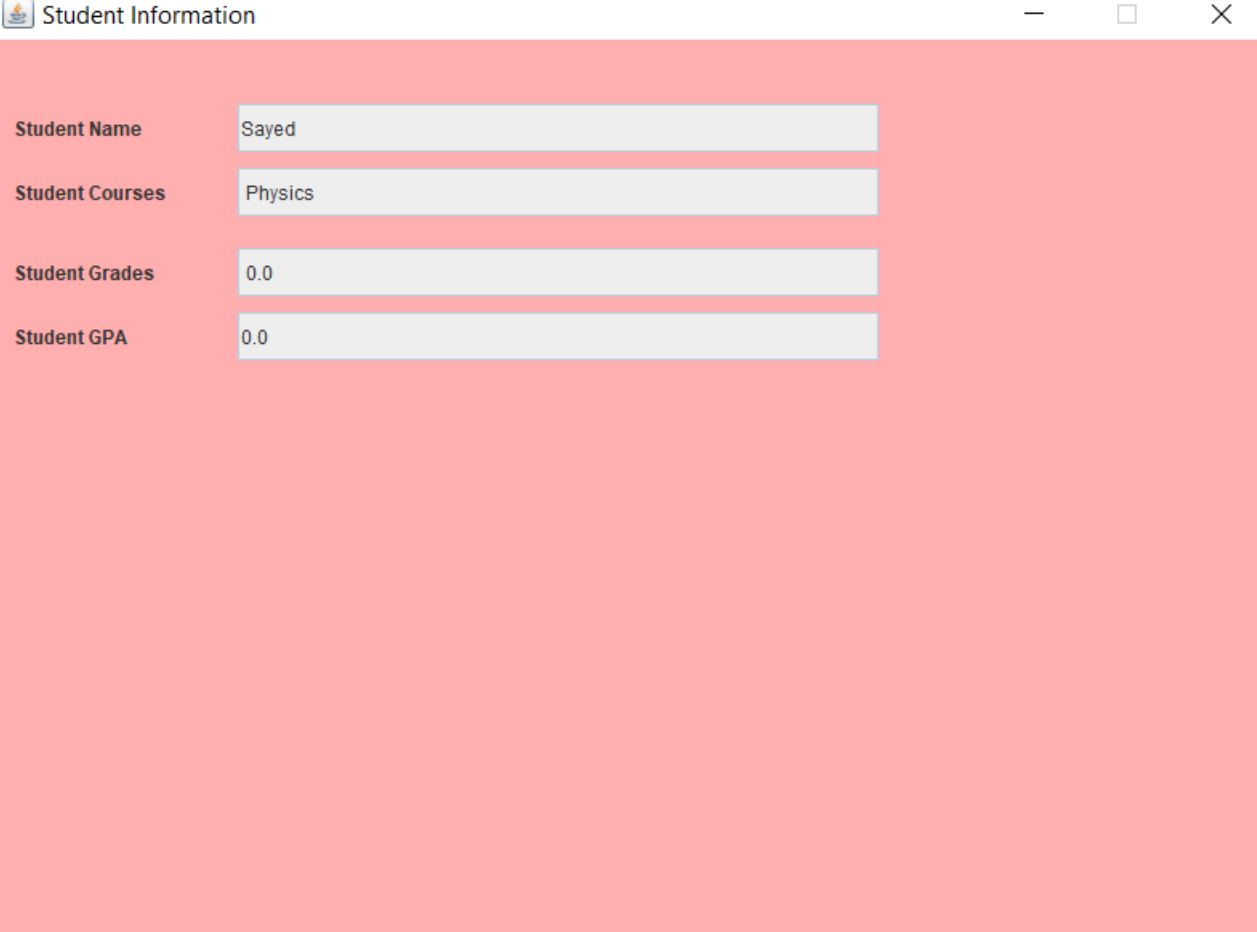
- Student Id:** A text input field.
- Register:** A blue button with white text.
- Courses:** A dropdown menu currently showing "Physics".
- Course Content:** A blue button with white text.
- Exams:** A blue button with white text.
- Form Fields:** A series of five horizontal light gray bars. The first bar is empty. The following four bars each have a radio button on the left.
- Submit:** A blue button with white text, located at the bottom center.

## Code:

```
public void ShowStudentFrame(String course1 ,String course2 , String course3 , String course4) {
    Submit =new JButton("Submit");
    CourseContent = new JButton("Course Content");
    Register =new JButton("Register");
    StudentIdText2 = new JTextField();
    Question = new JTextField();
    Courses = new JLabel("Courses");
    Exams = new JButton("Exams");
    StudentIdLabel2 = new JLabel("Student Id");
    combostd = new JComboBox();
    StudentPanel = new JPanel();
    C1 = new JRadioButton("");
    C2 = new JRadioButton("");
    C3 = new JRadioButton("");
    C4 = new JRadioButton("");
    choices = new ButtonGroup();
    //////////////////////////////////////
    this.setTitle("Student");
    this.setSize(800, 600);
    StudentPanel.setSize(800,600);
    this.setVisible(true);
    StudentPanel.setVisible(true);
    this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    this.setLayout(null);
    StudentPanel.setLayout(null);
    StudentPanel.setBackground(Color.PINK);
    this.setResizable(false);
    this.setLocation(700, 200);
    this.add(StudentPanel);
    //////////////////////////////////////
    StudentIdLabel2.setBounds(10, 40, 100, 10);
    StudentPanel.add(StudentIdLabel2);
    StudentIdText2.setBounds(80, 35, 150, 20);
    StudentPanel.add(StudentIdText2);
    Register.setBounds(305, 15, 150, 50);
    StudentPanel.add(Register);
    CourseContent.setBounds(305, 80, 150, 50);
    StudentPanel.add(CourseContent);
    Courses.setBounds(10, 80, 150, 50);
    StudentPanel.add(Courses);
    combostd.setBounds(100, 80, 150, 50);
    combostd.addItem(course1);
    combostd.addItem(course2);
    combostd.addItem(course3);
    combostd.addItem(course4);
    StudentPanel.add(combostd);
    Exams.setBounds(305, 150, 150, 50);
    StudentPanel.add(Exams);
    Question.setBounds(10, 210, 600, 30);
    StudentPanel.add(Question);
    Question.setEditable(false);
    C1.setBounds(10, 250, 600, 30);
    choices.add(C1);
    StudentPanel.add(C1);
    C2.setBounds(10, 290, 600, 30);
    choices.add(C2);
    StudentPanel.add(C2);
    C3.setBounds(10, 330, 600, 30);
    choices.add(C3);
    StudentPanel.add(C3);
    C4.setBounds(10, 370, 600, 30);
    choices.add(C4);
    StudentPanel.add(C4);
    Submit.setBounds(305, 420, 150, 50);
    StudentPanel.add(Submit);
    Register.addActionListener(this);
    Submit.addActionListener(this);
    Exams.addActionListener(this);
    CourseContent.addActionListener(this);
}
```

## **4-Student Information Frame:**

When instructor type student id and then click show all information the frame will appear that contain all information about student.



A screenshot of a web application window titled "Student Information". The window has a light red background and a white title bar with standard window controls (minimize, maximize, close). Inside the window, there is a form with four rows, each with a label on the left and a text input field on the right. The labels are "Student Name", "Student Courses", "Student Grades", and "Student GPA". The input fields contain the values "Sayed", "Physics", "0.0", and "0.0" respectively.

Student Name	Sayed
Student Courses	Physics
Student Grades	0.0
Student GPA	0.0

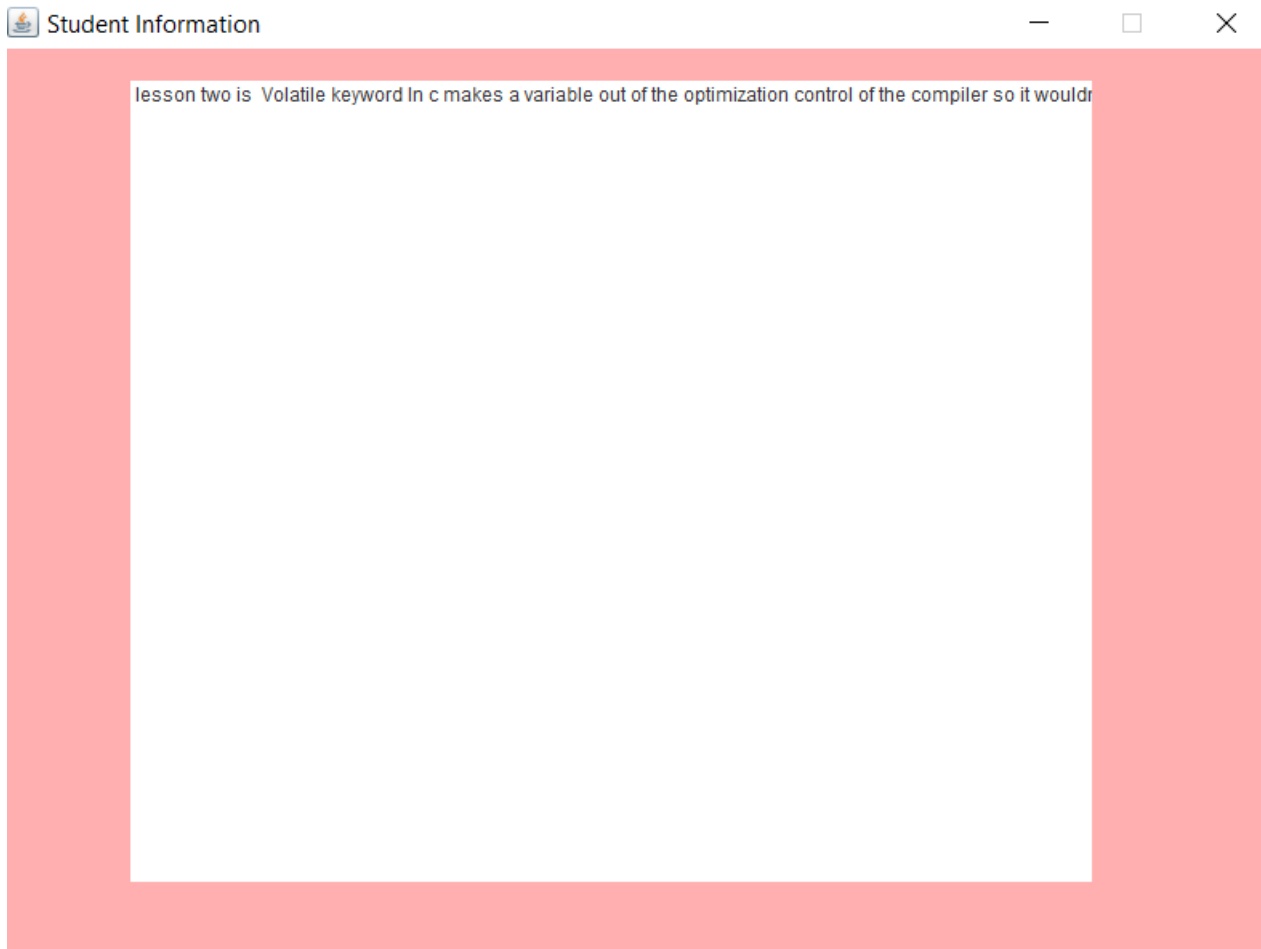
## Code:

```
public void ShowStudentInformation() {
    StudentName = new JLabel("Student Name");
    StudentCourse = new JLabel("Student Courses");
    Studentgrade = new JLabel("Student Grades");
    StudentGPA = new JLabel("Student GPA");
    StudentNameT= new JTextField();
    StudentCourseT= new JTextField();
    StudentgradeT= new JTextField();
    StudentGPAT= new JTextField();
    StuInfoPanel = new JPanel();
    StudentNameT.setEditable(false);
    StudentCourseT.setEditable(false);
    StudentgradeT.setEditable(false);
    StudentGPAT.setEditable(false);
    //////////////////////////////////////
    this.setTitle("Student Information");
    this.setSize(800, 600);
    StuInfoPanel.setSize(800,600);
    this.setVisible(true);
    StuInfoPanel.setVisible(true);
    this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    this.setLayout(null);
    StuInfoPanel.setLayout(null);
    StuInfoPanel.setBackground(Color.PINK);
    this.setResizable(false);
    this.setLocation(700, 200);
    this.add(StuInfoPanel);
    //////////////////////////////////////
    StudentName.setBounds(10, 40, 100, 30);
    StuInfoPanel.add(StudentName);
    StudentCourse.setBounds(10, 80, 100, 30);
    StuInfoPanel.add(StudentCourse);
    Studentgrade.setBounds(10, 130, 100, 30);
    StuInfoPanel.add(Studentgrade);
    StudentGPA.setBounds(10, 170, 100, 30);
    StuInfoPanel.add(StudentGPA);
    StudentNameT.setBounds(150, 40, 400, 30);
    StuInfoPanel.add(StudentNameT);
    StudentCourseT.setBounds(150, 80, 400, 30);
    StuInfoPanel.add(StudentCourseT);
    StudentgradeT.setBounds(150, 130, 400, 30);
    StuInfoPanel.add(StudentgradeT);
    StudentGPAT.setBounds(150, 170, 400, 30);
    StuInfoPanel.add(StudentGPAT);

    Ahmed.GetStudentInfo(flag,Integer.parseInt(stuId),StudentNameT,StudentCourseT,Stu
udentgradeT,StudentGPAT);
}
```

## 5-Show Content of course

when student click on show content of course the frame will appear that contain all content of course.



## Code:

```
public void ShowCourseContent() {
    CourseCon = new JTextArea();
    CourseCon.setEditable(false);
    CourseContentPanel = new JPanel();
    //////////////////////////////////////
    this.setTitle("Student Information");
    this.setSize( width: 800, height: 600);
    CourseContentPanel.setSize( width: 800, height: 600);
    this.setVisible(true);
    CourseContentPanel.setVisible(true);
    this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    this.setLayout(null);
    CourseContentPanel.setLayout(null);
    CourseContentPanel.setBackground(Color.PINK);
    this.setResizable(false);
    this.setLocation( x: 700, y: 200);
    this.add(CourseContentPanel);
    //////////////////////////////////////
    CourseCon.setBounds( x: 80, y: 20, width: 600, height: 500);
    CourseContentPanel.add(CourseCon);
}
```



## **6-Instucor: Add Content:**

Function that makes instructor can add content of the course. It take 2 parameter course and contetn string.

```
public static boolean AddContent (Course course , String content)
{
    course.setContent(content);
    return !course.getContent().isEmpty();
}
```

## **7-Instucor: Add Exam:**

Function that makes instructor can add exam of the course. It take 2 parameter course and exam string.

```
public static boolean AddExam (Course course , String exam)
{
    course.setExam(exam);
    return !course.getExam().isEmpty();
}
```

## **8-Instucor: Show Student Info:**

Function that makes instructor can add show all information of student. It take 5 parameter Student and text fields will show them id , name, course, grade and GPA.

```
public void GetStudentInfo (Student student, int id, JTextField name,JTextField course,JTextField grade,JTextField gpa)
{
    name.setText(student.getStudentName());
    String courses = " ",garde = " ";

    for (int i =0 ; i < student.StudentCourses.size(); i++)
    {
        courses += student.StudentCourses.get(i).getCourseName()+ " ";
        garde+= student.StudentCourses.get(i).getGrade()+ " ";
    }
    course.setText(courses);
    grade.setText(String.valueOf(garde));
    gpa.setText(String.valueOf(student.GenerateGPA()));
}
```

## **9-Student: Registering Course:**

Function that takes course as parameter and add it to array list of student registered courses. Return true if added and false if there any problem.

```
public boolean RegisteringCourse (Course course)
{
    if(this.StudentCourses.contains(course))
        JOptionPane.showMessageDialog( parentComponent: null, message: "This course is registered", title: "Student", JOptionPane.INFORMATION_MESSAGE);
    else
        this.StudentCourses.add(course);

    return this.StudentCourses.contains(course);
}
```

## **10-Student: Taking Exam:**

Function that takes course as parameter and grade of student in this course add it to array list of student registered courses. Return true if added and false if there any problem.

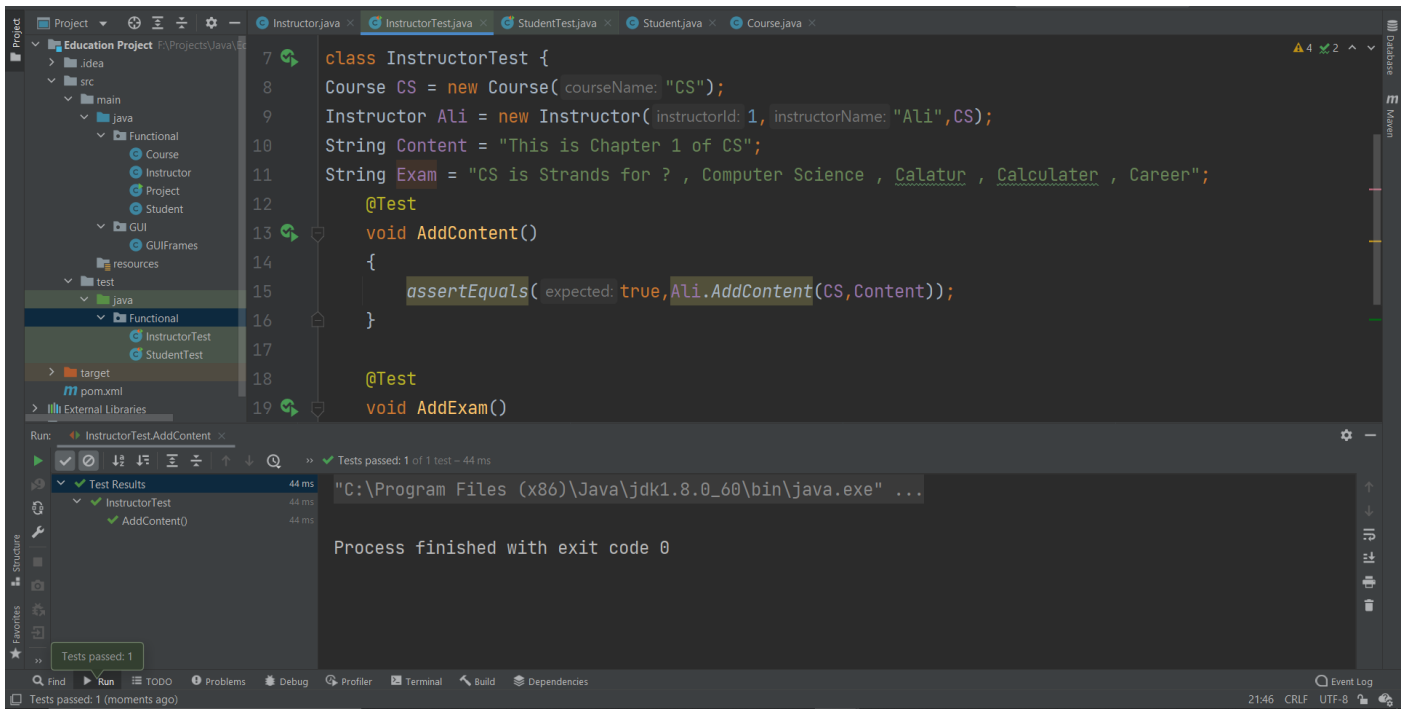
```
public boolean TakingExam (Course course,double Grade)
{
    CourseCount = StudentCourses.indexOf(course);
    StudentCourses.set(CourseCount ,course).setGrade(Grade);
    StudentGPA += Grade ;
    if(StudentCourses.contains(course))
        return true;
    else
        return false;
}
```

# Testing

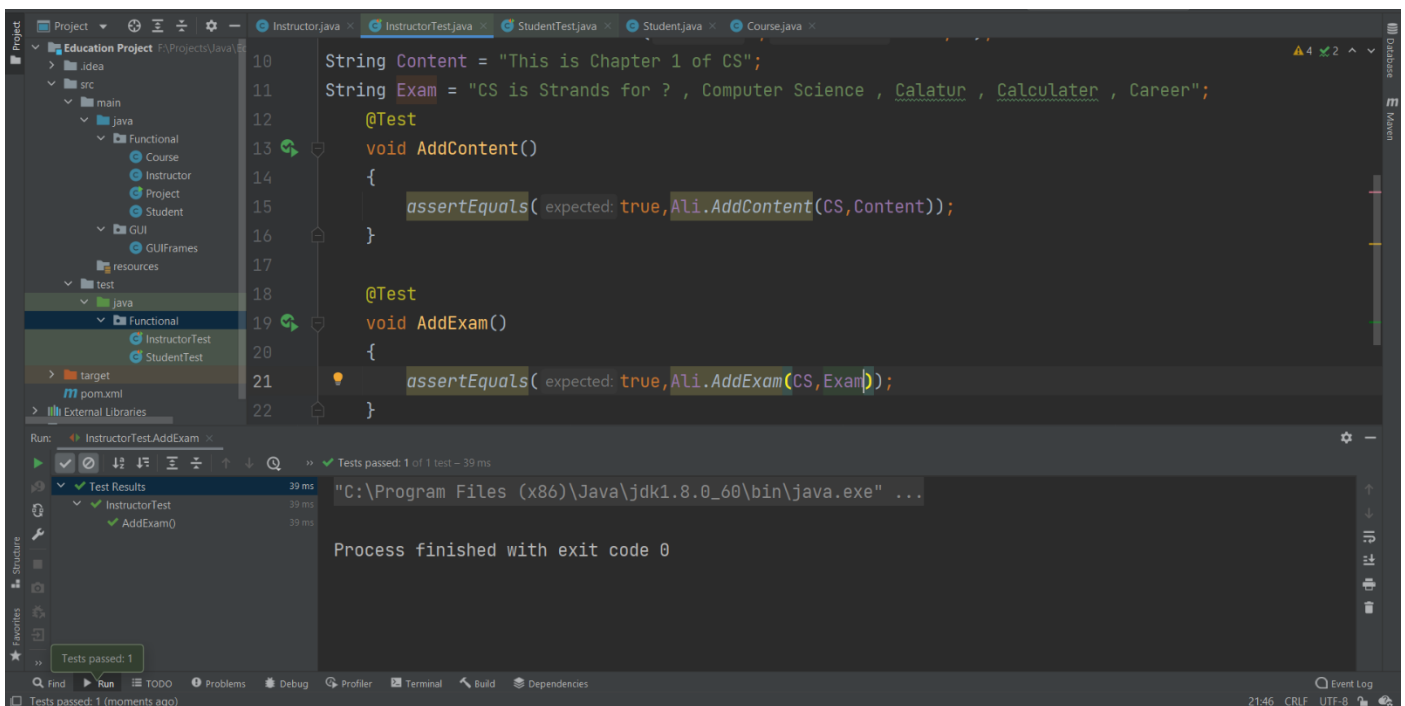
## 1-Unit Testin :

Using Junit, I tested 4 component of the system and it's the 4 major function. These function is:

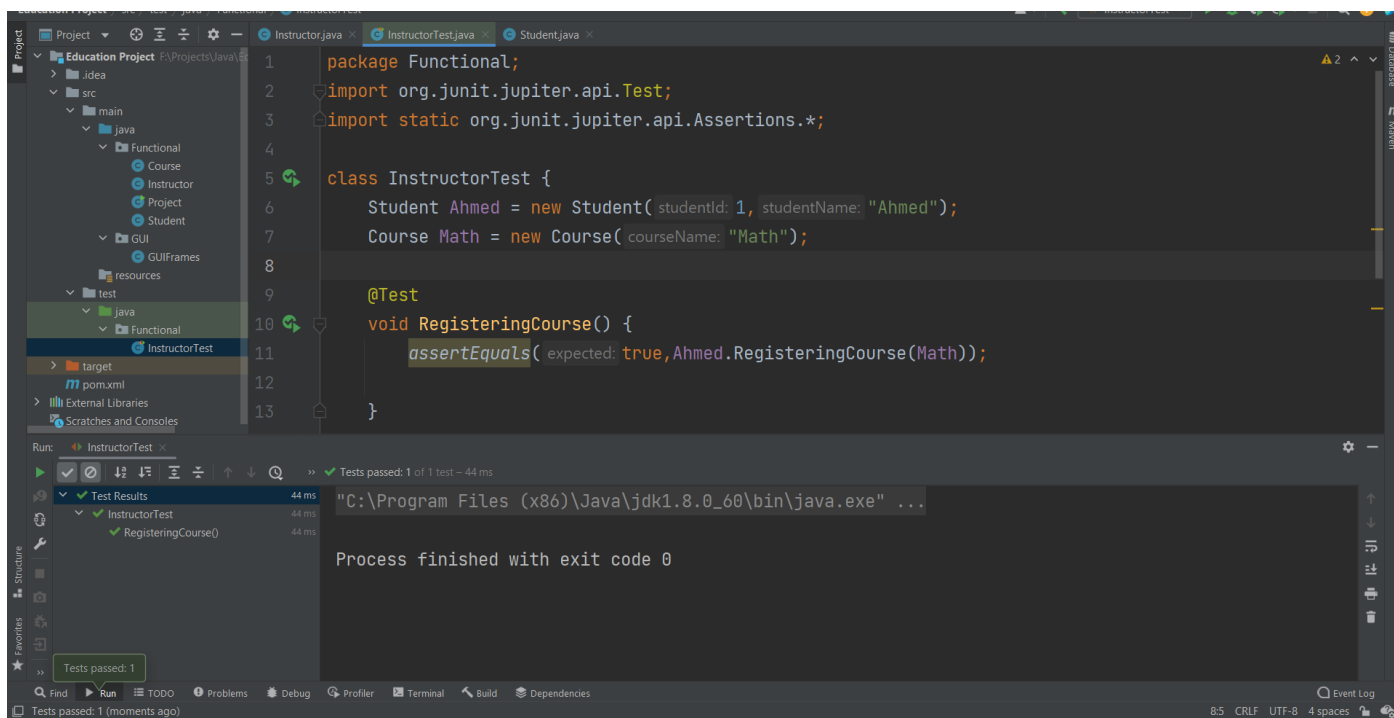
### 1-Add Content:



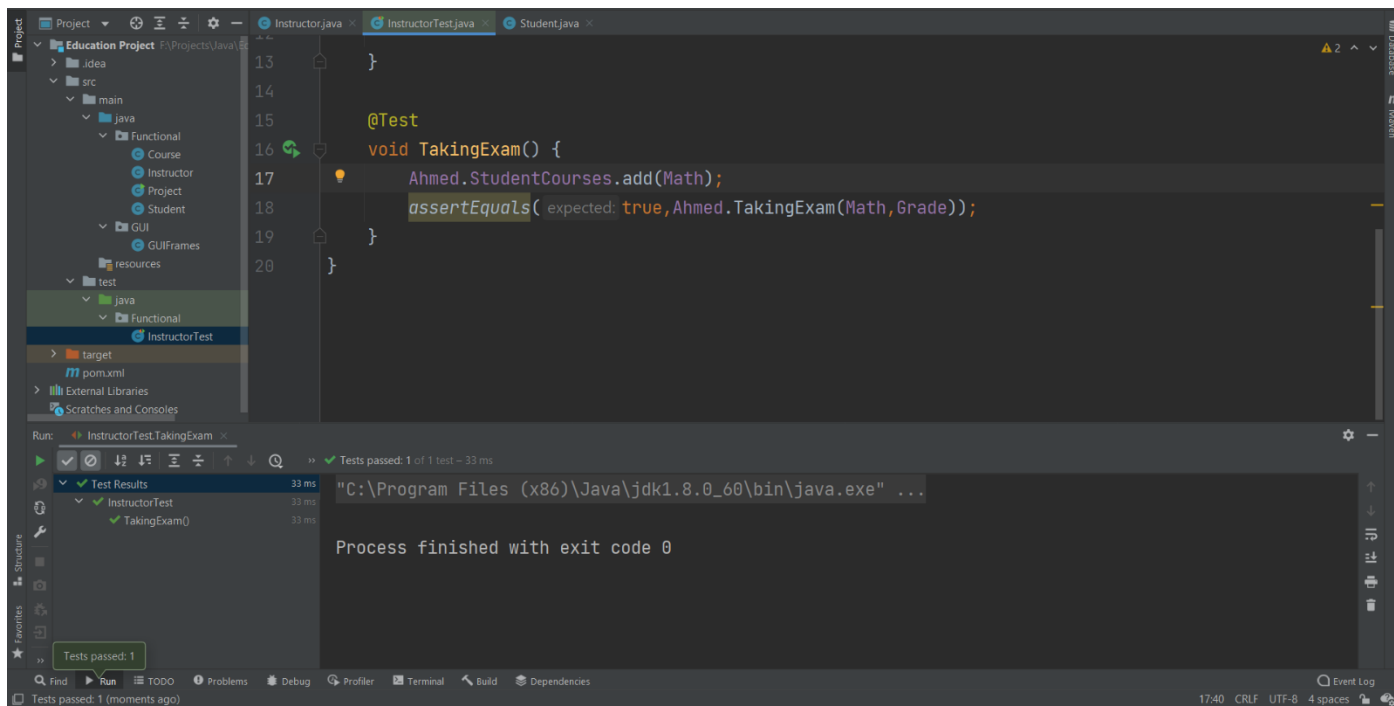
### 2-Add Exam:



### 3-Registering Course:

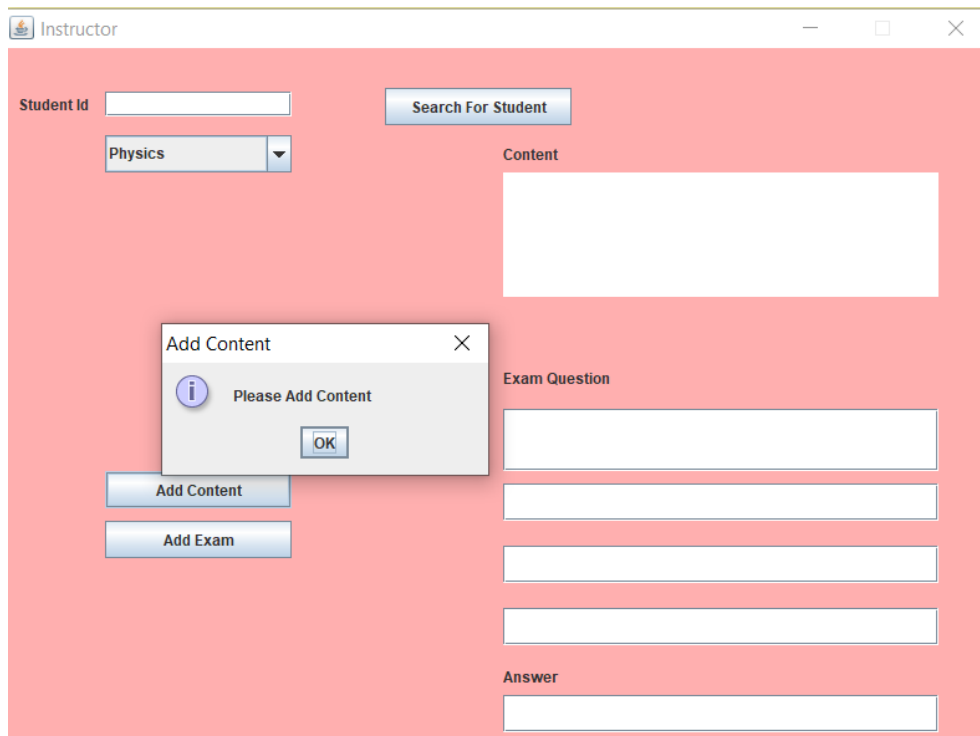


### 4-Taking Exam:

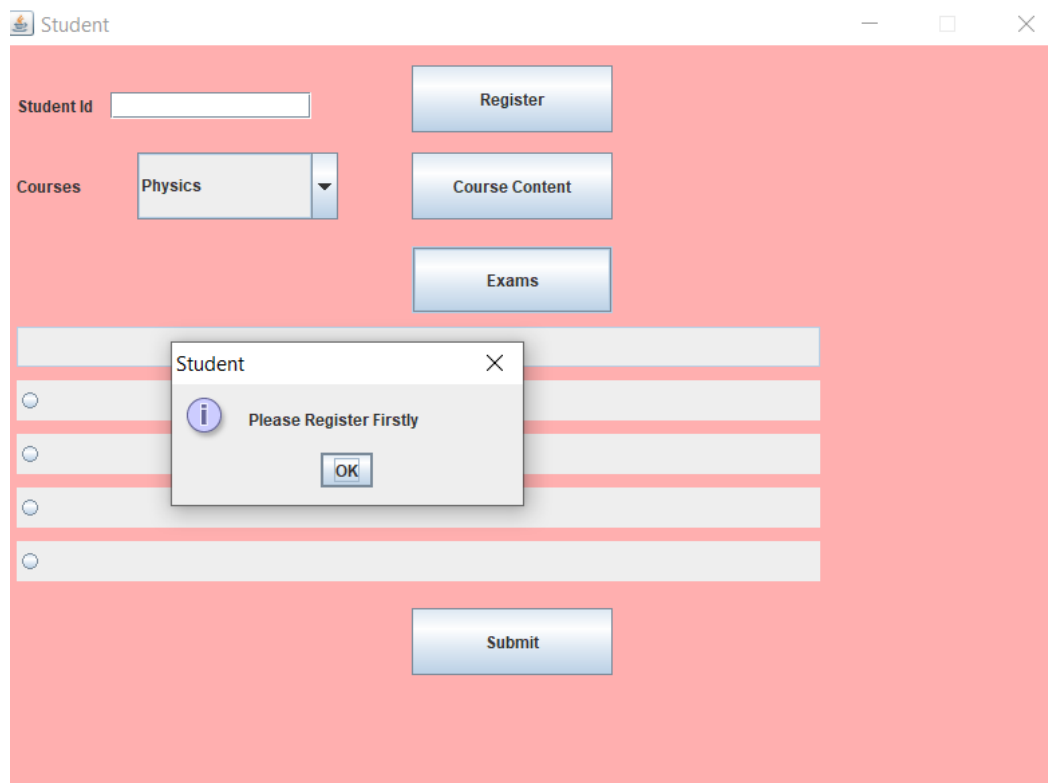


## 2-GUI Testing:

All fields are handled when its empty or contain wrong data. Follow the attachments in screenshots.



The screenshot shows the 'Instructor' window with a red background. It contains a 'Student Id' text field, a 'Search For Student' button, a 'Physics' dropdown menu, and a 'Content' text area. Below the 'Content' area is an 'Add Content' button. To the right, there is an 'Exam Question' section with four text fields and an 'Answer' text field. A modal dialog box titled 'Add Content' is open in the center, displaying an information icon, the text 'Please Add Content', and an 'OK' button.



The screenshot shows the 'Student' window with a red background. It contains a 'Student Id' text field, a 'Register' button, a 'Courses' dropdown menu set to 'Physics', a 'Course Content' button, and an 'Exams' button. Below these are four radio buttons. At the bottom is a 'Submit' button. A modal dialog box titled 'Student' is open in the center, displaying an information icon, the text 'Please Register Firstly', and an 'OK' button.

Instructor

Student Id

Search For Student

Physics

Content

Student Information

Please Enter Student Id

OK

Add Content

Add Exam

Exam Question

Answer

Instructor

Student Id

Search For Student

Physics

Content

Add Content

Add Exam

Add Content

Please Add Complete Exam

OK

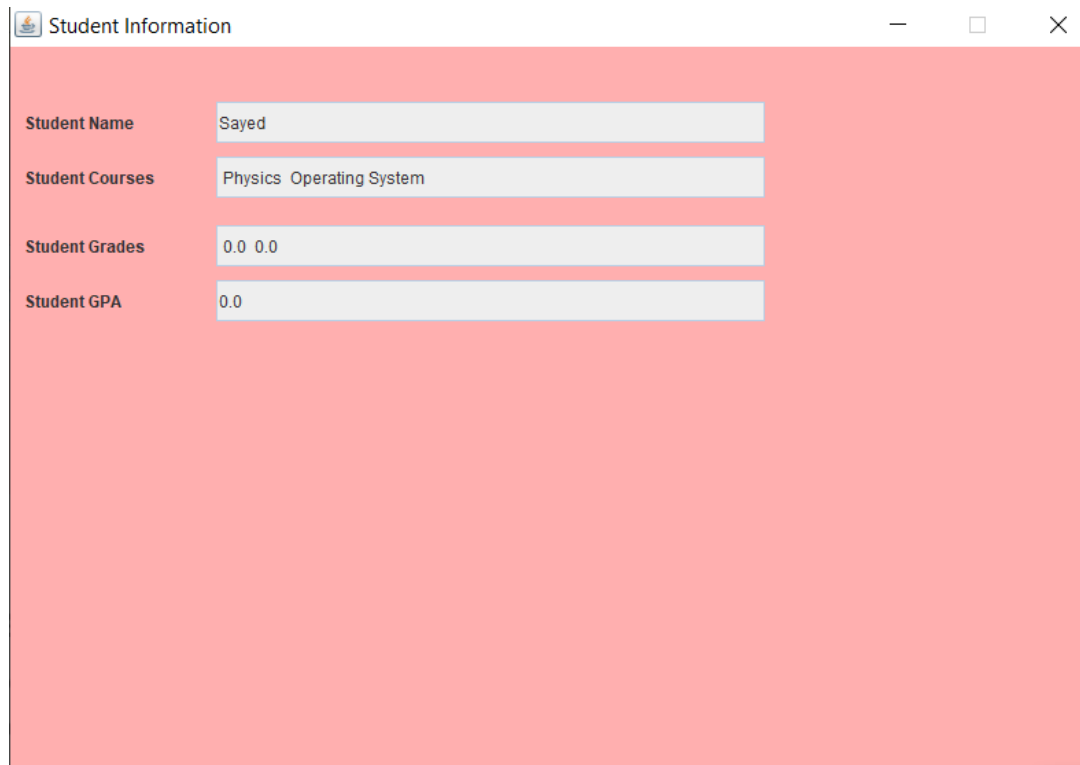
Exam Question

Answer

### 3-Integration Testing:

Integration is done manually using deferent scenarios. Follow the attachment for seeing some of them.

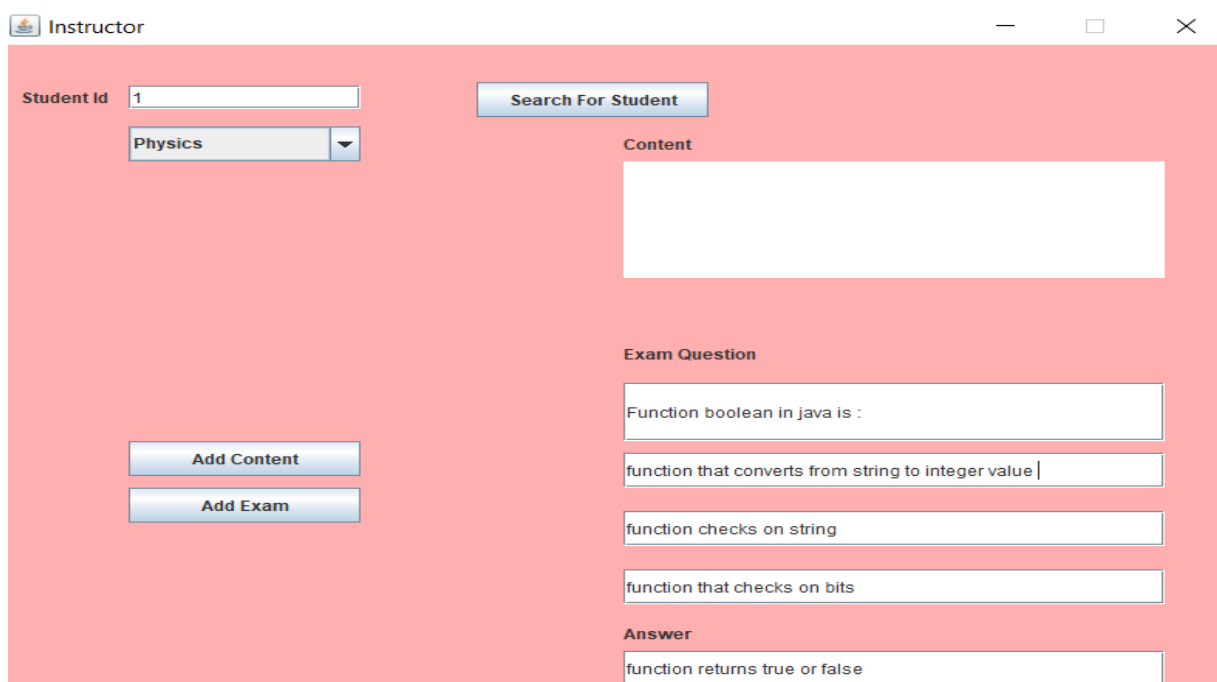
1-Register 2 courses by the student which his id is 1 and didn't take the exam then show his info by instructor.



A screenshot of a window titled "Student Information". The window has a pink background and contains four rows of data, each with a label on the left and a text input field on the right:

Label	Value
Student Name	Sayed
Student Courses	Physics Operating System
Student Grades	0.0 0.0
Student GPA	0.0

2-Add exam for physics course by instructor and take it by student and test one positive test case with the really answer and one negative test case with the wrong answer follow the screenshots.



A screenshot of a window titled "Instructor". The window has a pink background and contains several fields and buttons:

- Student Id:** A text input field containing the value "1".
- Search For Student:** A button.
- Physics:** A dropdown menu.
- Content:** A large text input field.
- Exam Question:** A section containing four text input fields with the following text: "Function boolean in java is :", "function that converts from string to integer value |", "function checks on string", and "function that checks on bits".
- Answer:** A text input field containing the text "function returns true or false".
- Add Content:** A button.
- Add Exam:** A button.

## \*Negative Test Case:

The screenshot shows a web application window titled "Student". It contains a "Student Id" field with the value "1" and a "Register" button. Below this is a "Courses" dropdown menu set to "Physics" and a "Course Content" button. Further down is an "Exams" button. The main content area displays a question: "Function boolean in java is :". Below the question are five radio button options:

- ☐ function returns true or false
- ☒ function checks on string
- ☐ function that converts from string to integer value
- ☐ function that checks on bits

A "Submit" button is located at the bottom. A modal dialog box is open in the center, titled "Student", with a close button (X). It contains an information icon (i), the text "Wrong Answer", and an "OK" button.

## \*Positive Test Case:

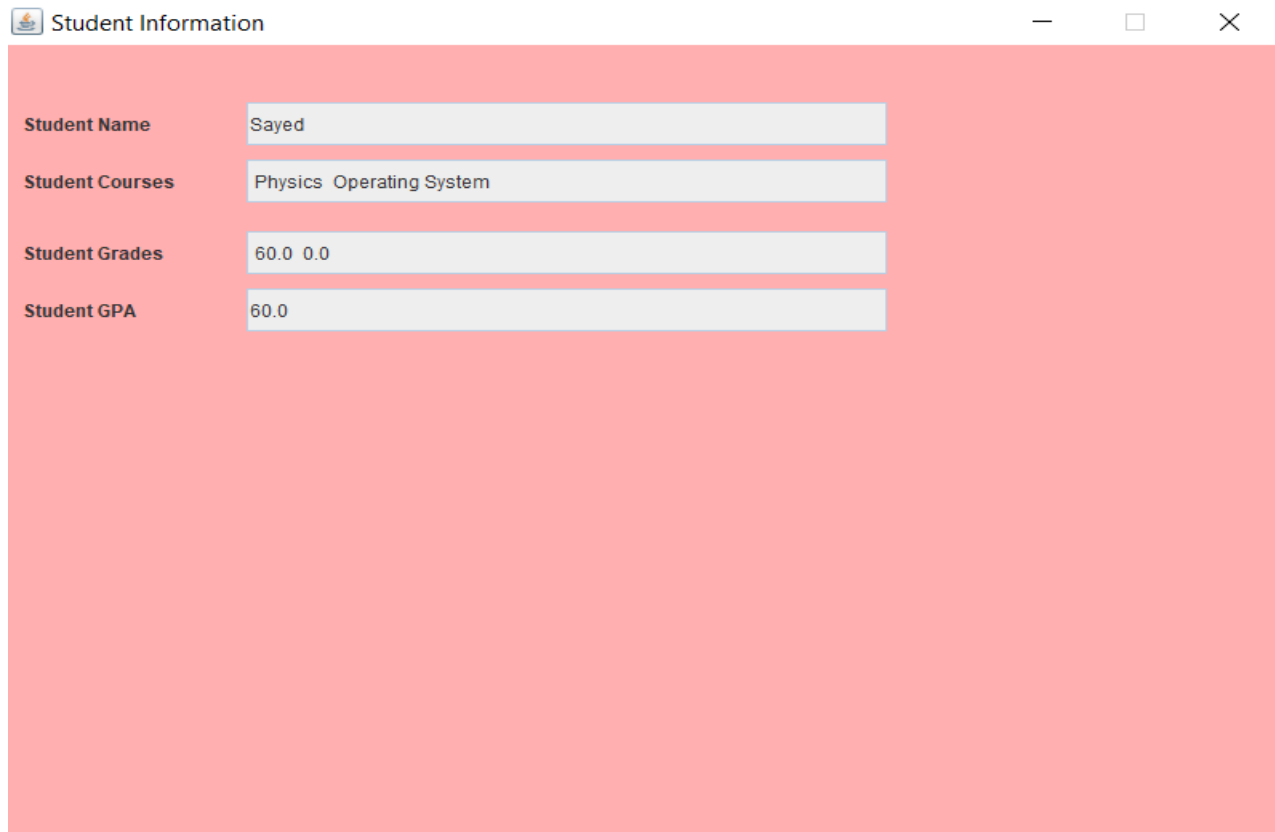
The screenshot shows the same web application window as above. The "Student Id" field still contains "1". The "Courses" dropdown is still set to "Physics". The question "Function boolean in java is :" is displayed. The radio button options are the same as in the negative test case:

- ☒ function returns true or false
- ☐ function checks on string
- ☐ function that converts from string to integer value
- ☐ function that checks on bits

The "Submit" button is at the bottom. A modal dialog box is open in the center, titled "Student", with a close button (X). It contains an information icon (i), the text "Good Answer", and an "OK" button.



Then if we go to see this student's information by instructor, we will see it:



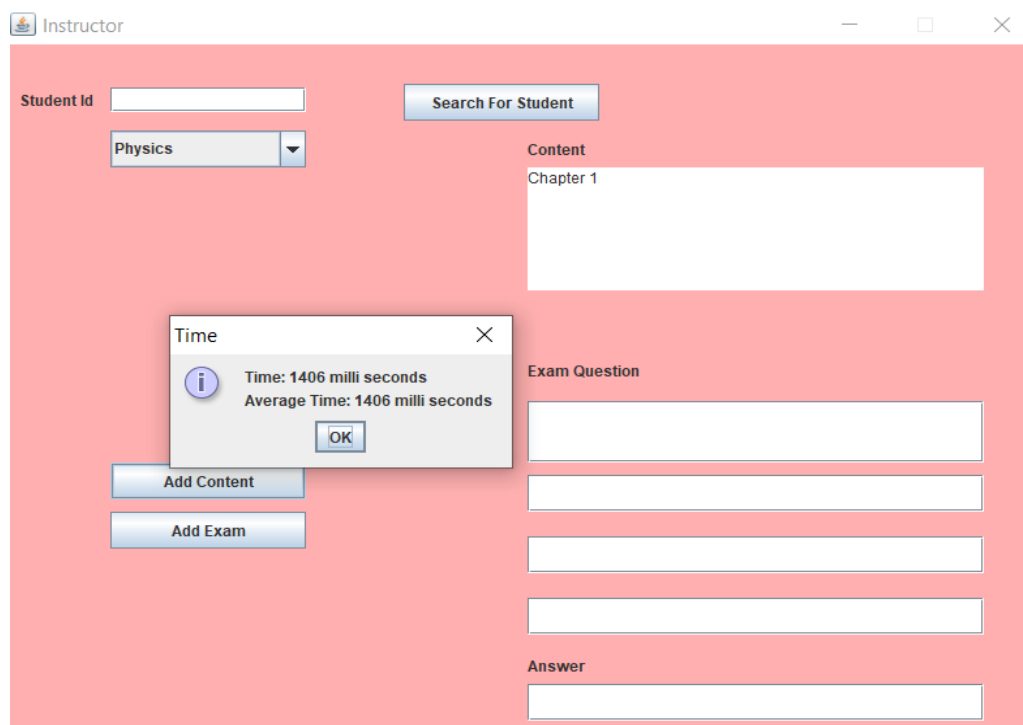
A screenshot of a web application window titled "Student Information". The window has a light blue header bar with a small icon on the left and standard window controls (minimize, maximize, close) on the right. The main content area is white and contains four rows of information, each with a label on the left and a text input field on the right:

Label	Value
Student Name	Sayed
Student Courses	Physics Operating System
Student Grades	60.0 0.0
Student GPA	60.0

## **Performance Testing:**

When you click to call any function there is alert will be visible show the time that function takes. Also, alert will show the average time when you call this function a lot of time.

- **Example: Instructor: Add Content**  
After one call only:



A screenshot of a web application window titled "Instructor". The window has a light blue header bar with a small icon on the left and standard window controls (minimize, maximize, close) on the right. The main content area is white and contains several form elements:

- Student Id:** A text input field.
- Search For Student:** A button.
- Physics:** A dropdown menu.
- Content:** A text input field with the value "Chapter 1".
- Exam Question:** A series of five text input fields.
- Answer:** A text input field.
- Add Content:** A button.
- Add Exam:** A button.

An alert dialog box titled "Time" is overlaid on the "Add Content" button. The dialog box contains the following text:

Time: 1406 milli seconds  
Average Time: 1406 milli seconds

There is an "OK" button at the bottom of the dialog box.

After second call with a big amount of data:

The screenshot shows the 'Instructor' application window. It features a 'Student Id' input field, a 'Search For Student' button, a 'Physics' dropdown menu, and a 'Content' text area containing 'Chapter 1 Chapter 2'. Below these are 'Add Content' and 'Add Exam' buttons. To the right, there is an 'Exam Question' section with four empty text input fields and an 'Answer' section with one empty text input field. A 'Time' dialog box is overlaid on the 'Add Content' button, displaying the following information:

Time	
Time:	1145 milli seconds
Average Time:	572 milli seconds
<input type="button" value="OK"/>	

After third call with a big amount of data:

The screenshot shows the 'Instructor' application window after a third call with a large amount of data. The layout is identical to the previous screenshot, but the 'Content' text area now contains 'Chapter 1 Chapter 2 Chapter 3'. The 'Time' dialog box is still overlaid on the 'Add Content' button, displaying the following information:

Time	
Time:	1079 milli seconds
Average Time:	359 milli seconds
<input type="button" value="OK"/>	

- **Example2: Instructor: Add Exam**

After one call only:

The screenshot shows the 'Instructor' application window. It has a red background. At the top left, there is a 'Student Id' text box and a 'Search For Student' button. Below the 'Student Id' is a dropdown menu currently showing 'Physics'. To the right of the dropdown are two buttons: 'Add Content' and 'Add Exam'. On the right side of the window, there is a 'Content' text box, an 'Exam Question' text box containing 'What ?', and three answer boxes labeled 'a', 'b', and 'c'. Below these is an 'Answer' text box containing 'd'. A small 'Time' dialog box is open in the center, displaying 'Time: 786 milli seconds' and 'Average Time: 786 milli seconds' with an 'OK' button.

After second call with a big amount of data:

This screenshot is identical to the one above, but the 'Exam Question' text box now contains the text 'What ? When ? Why ?'. The 'Time' dialog box remains open, showing the same timing information.

After third call with a big amount of data:

Instructor

Student Id

Physics

Content

Time

Time: 1138 milli seconds  
Average Time: 379 milli seconds

Exam Question

What ? When ? Why ? blablablablablablablablabla

aaaaaa

bbbbbbb

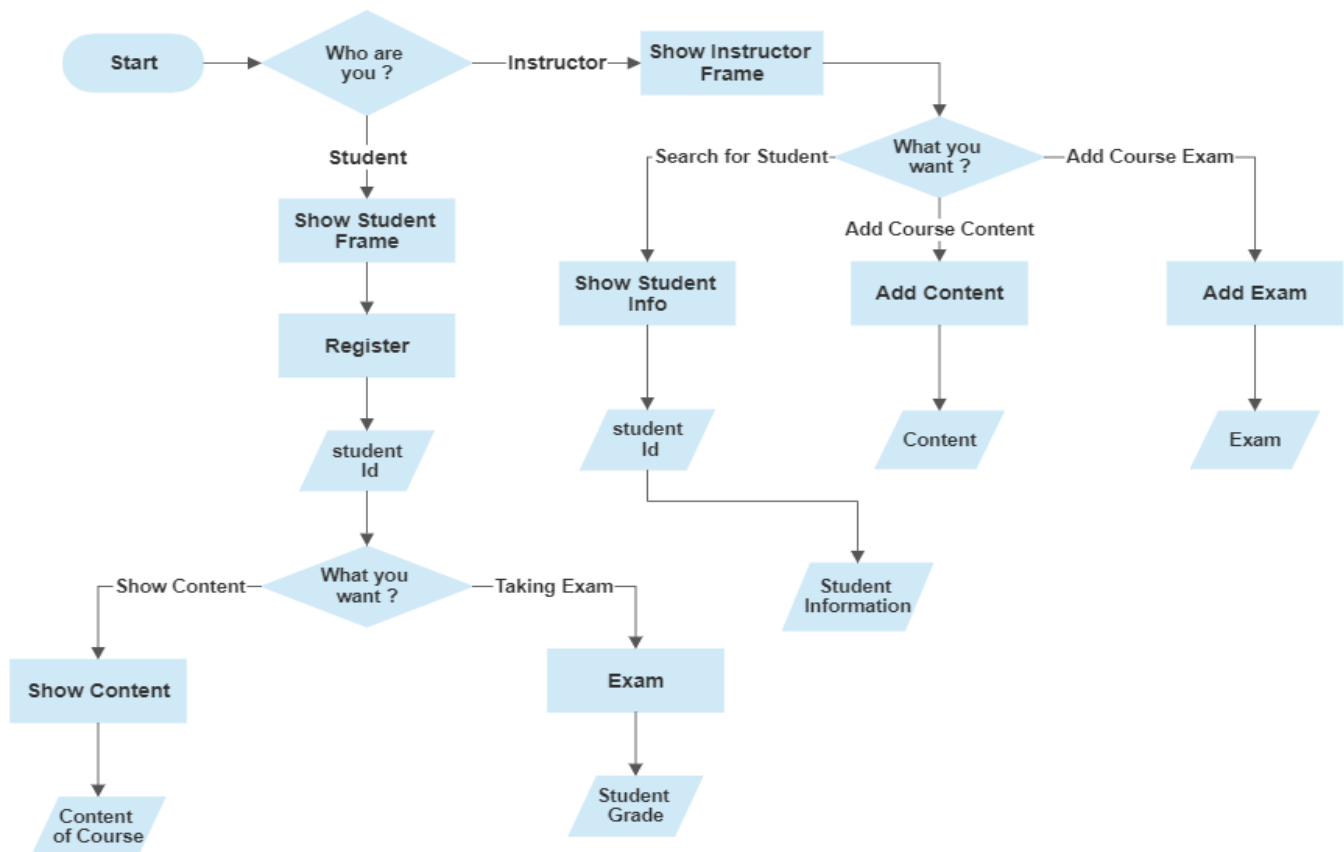
ccccc

Answer

dddddddd

## Finally Follow Charts :

- Development Process



- Integration Testing Process

