CS 102 - Project

ÖZYEĞİN ÜNİVERSİTESİ

Project Due Date: Sunday, December 29, 2017, 23:55

(Late submissions will not be accepted)

Part 2

In the first part of the project, you have implemented a model part of a LMS like application. In this second part, you will extend your project by implementing the view and the controller parts.
This application will be used only by the instructors for simplicity.

GuiMain.java

You are provided with this starter code. It contains the main function. In the main function a department object is created by using the createDepartment() function. Use this department object to create a login screen object.

Login Screen

This is the first screen of your application. User needs to provide an email of an instructor in order to successfully login to the system.

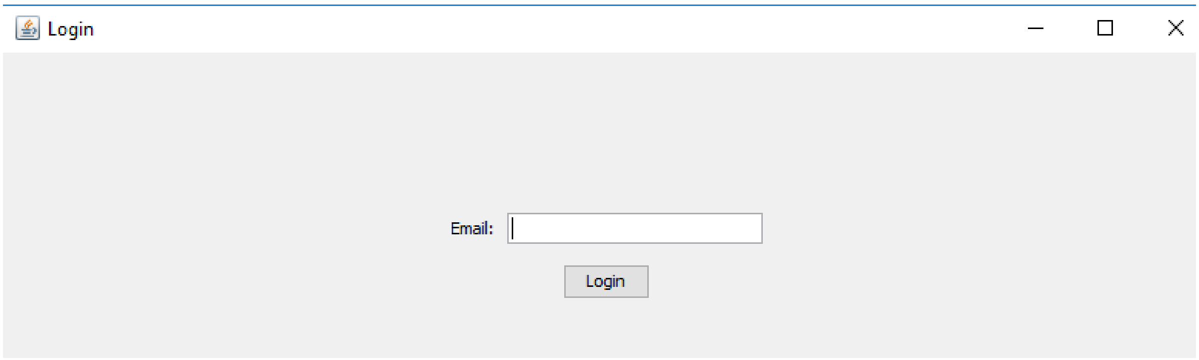


Figure 1

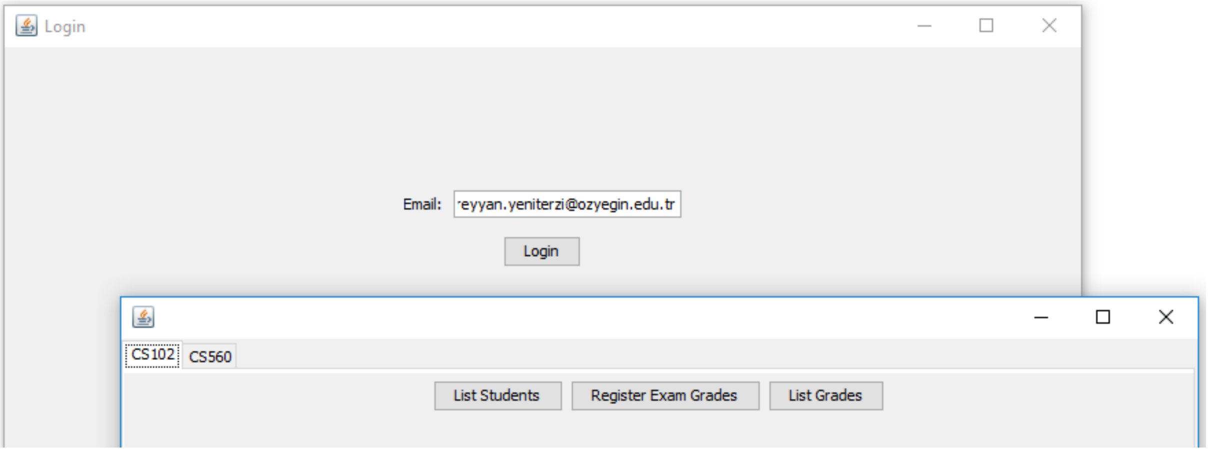
After a successful login, a new frame is created for the instructor with inputted email. Let's call it the Instructor Frame. The login frame continues running as seen in Figure 2.

Instructor Frame

If user inputs *reyyan.yeniterzi@ozyegin.edu.tr*, an instructor frame with Professor Yeniterzi's courses will be created. Remember that Professor Yeniterzi is giving two courses.

```
department.assignInstructorToCourse("Reyyan Yeniterzi", "CS102");
department.assignInstructorToCourse("Reyyan Yeniterzi", "CS560");
```

A frame with two tabbed panes will be created as seen in Figure 2. One for CS102 and another for CS560. A starter code for instructor frame is provided to you. The code has the necessary part for you to implement the tabbed panes.

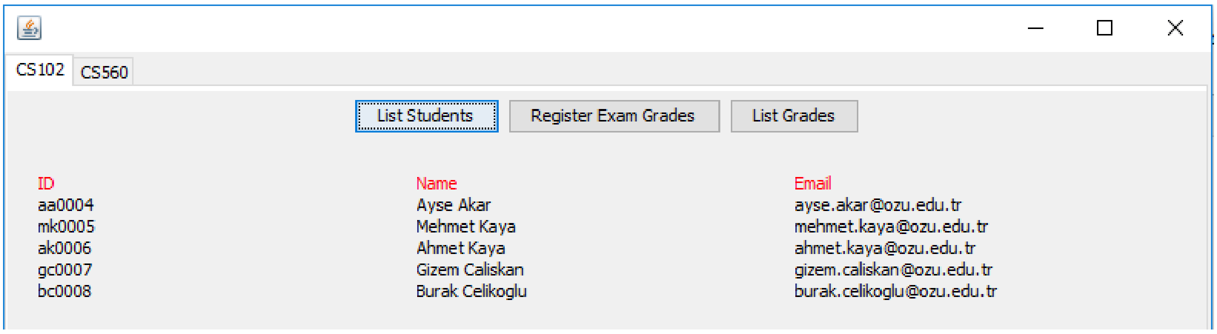


Email: ˈeyyan.yeniterzi@ozyegin.edu.tr

Figure 2

An instructor can do three things with a course.

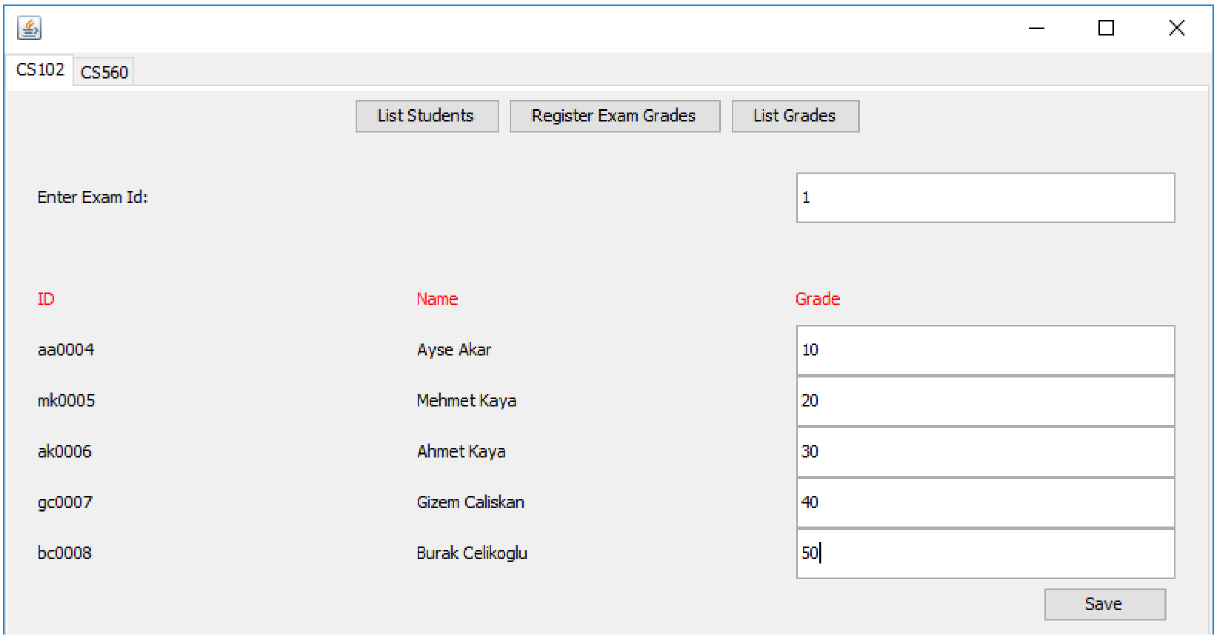
1. **List Students:** The enrolled students are listed as shown in Figure 3.



ID	Name	Email
aa0004	Ayse Akar	ayse.akar@ozu.edu.tr
mk0005	Mehmet Kaya	mehmet.kaya@ozu.edu.tr
ak0006	Ahmet Kaya	ahmet.kaya@ozu.edu.tr
gc0007	Gizem Caliskan	gizem.caliskan@ozu.edu.tr
bc0008	Burak Celikoglu	burak.celikoglu@ozu.edu.tr

Figure 3

2. **Register Exam Grades:** Exam grades are entered for an exam.

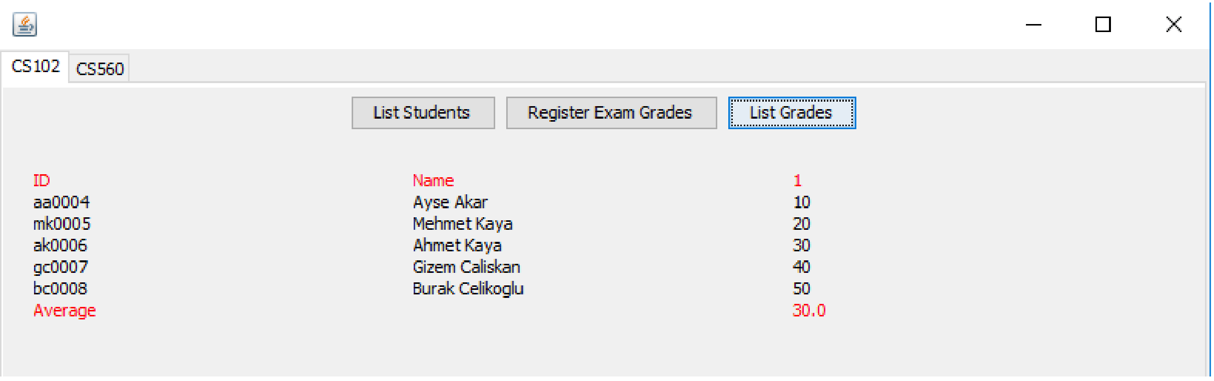


Enter Exam Id: 1

ID	Name	Grade
aa0004	Ayse Akar	10
mk0005	Mehmet Kaya	20
ak0006	Ahmet Kaya	30
gc0007	Gizem Caliskan	40
bc0008	Burak Celikoglu	50

Figure 4

3. **List Grades:** Grades of the enrolled students are displayed for the registered exams, together with the exam average.



ID	Name	1
aa0004	Ayse Akar	10
mk0005	Mehmet Kaya	20
ak0006	Ahmet Kaya	30
gc0007	Gizem Caliskan	40
bc0008	Burak Celikoglu	50
Average		30.0

Figure 5

In order to show different output for these three functionalities, you need to clear the previous components in the panel. You can do this with *removeAll()* function. After adding the necessary components to the panel you can call the *revalidate()* and *repaint()* functions to refresh the panel.

You have already implemented the necessary functions to get the enrolled student lists, register grades, get student exam grades and calculating exam average. You just need to call these functions when necessary.

An executable will be provided to you to play around with. For all other necessary details please refer to the provided executable. Your program needs to have all the same functionalities that this executable has. If you have any other questions, please ask through Piazza.

On the other hand, you are free regarding the design part of the GUI part. Feel free to choose a component which you think will work. As long as you display the same content, you are free to display it any other format you want.

While implementing the view and the controller, you are free to make necessary changes in your model. Overall make sure that your model is separate then the view and the controller.

You learned about exception handling. During the project, you are responsible from handling the any necessary exceptional cases. One way to handle those exceptions is to use pop-up windows.

Use GuiMain.java as an example main. Your project will be tested will different department objects. Department created in this class is only a sample. Try to test your program with different test cases.

Submission

You will submit this homework via the LMS system. You should follow the file and method naming conventions for the model part. Point will be deducted if you don't do so.

Submit the following java files related to your model:

- Department.java
- Person.java
- Student.java

- Instructor.java
- Course.java
- GradeItem.java

Put all these files into a package named “Model”.

Rest of your classes for GUI View and Controller implementation, create another package named as “GUI”. Overall you will have two packages. Then zip these two packages.

Name your classes "exactly" the same as stated above. Capitalization of letters are important, too. (Do the same for the methods if stated in the pdf).

You should submit your source files as a **ZIP** archive file (**NOT** RAR or other formats). The name of the file should be in format “**<USER-ID>_project.zip**”. For example, if your username is un1234, then the name of the submitted file should be “un1234_project.zip”. Pay attention that all the letters are in lower-case.

You can add extra classes, extra methods or fields to your classes if you need any.

Late submissions, missing submissions and source files that do not compile are **not** accepted and will receive 0 as the grade. Even if you plan to make a partial submission, you still have to put the method declaration with empty bodies, which would prevent compile-time errors. Do not upload *.class files. These files can't be graded and you will get 0. Do not zip the whole Eclipse project. Do not zip src folder. Select the java files and zip them directly.

After submission, download your files to a location different than where your original project codes are. Make sure that the downloaded files contain everything to compile/build your project. Projects that do not compile will receive 0.

Do all the implementations yourself. In case of a plagiarism, you will receive -100. Do not share any part of your code for any reason.