

MINI-KUSIS

Group Members:

Bariş KAPLAN (**KU ID NUMBER:** 0069054, **KU LOGIN:** bkaplan18)

Ege SEÇİLMİŞ (**KU ID NUMBER:** 0069312, **KU LOGIN:** esecilmis18)

İsmail Ozan KAYACAN (**KU ID NUMBER:** 0069103, **KU LOGIN:** ikayacan18)

Lütfü Mustafa Kemal ATO (**KU ID NUMBER:** 0069579, **KU LOGIN:** lato18)

Moayed Haji ALI (**KU ID NUMBER:** 0066888, **KU LOGIN:** mali18)

Project Description:

In our project, we have created a university management application. The name of our application is “MINI-KUSIS”. We have developed our application in the integrated development environment (IDE) called “Eclipse IDE”. In our application, we have used the relational database management system called “MySQL”. The programming language that we have used in our project is “Java Programming Language” (For coding parts). For implementing the graphical user interface (GUI) part of our system, we have used the “Java Swing” tool. By using our system; instructors will be able to enter the letter grades of the students, view the assigned lectures, and view the lecture hours. By using our system; the students will be able to enroll to the lectures, drop a lecture, view their letter grades for the lessons they took, update his/her login password, and view their GPA. Moreover; by using our system, the administrators will be able to assign lectures to instructors, add lectures, and delete lectures. We think that the system we have created can be used to improve the current KUSIS system of our university. Moreover, we think that this system (the system we have created) can be used to develop more complex and functional university management applications in the future.

Tools and/or Technologies that we have used in our project:

IDE that we used: Eclipse Integrated Development Environment (Eclipse IDE)

Dependency Management Tool that we used: Apache Maven

Database Language that we used: MySQL

Programming Language that we used: Java Programming Language (For Coding Parts)

For the Graphical User Interface (GUI) Implementation: Java Swing

- **Note:** We have used MYSQL terminal for observing the created databases, tables, records, and so on.

The libraries that we have used in our project and the purposes of these libraries:

- **Snakeyaml:** For reading yaml files which contain the database schema.
- **Mysql-connector-java:** for connecting with the mysql database.

import java.awt.FlowLayout; //for adjusting layout of the page to flowlayout

import java.awt.event.ActionEvent; //for handling action events in GUI

import java.awt.event.ActionListener; //for listening to the GUI actions. The actions inside the action listener statements will be executed after the GUI actions

import javax.swing.JButton; //for buttons

import javax.swing.JCheckBox; //for having some decisions such as allowing the enrollment, allowing the course dropping, and so on.

import javax.swing.JDialog; //for dialog boxes

import javax.swing.JPanel; //for panels such as opening panel, and student grades panel

import javax.swing.JTextField; //for keeping the entered texts (such as username) in GUI

import java.awt.Color; //for the color of the background

import java.awt.GridLayout; //for adjusting layout of the page to gridlayout

import javax.swing.JComboBox; //for keeping track of the list of possible letter grades for the instructors

import javax.swing.JFrame; //for the frame containing the panels

import java.awt.Font; //for adjusting the font type, and font size of the title in opening panel

import java.awt.Image; //for inserting images

import java.sql.DriverManager; //for managing the database connection drivers which exist in Java

import java.sql.ResultSet; // for executing the queries and assigning the result of these queries to ResultSet type of variable

import java.sql.SQLException; //for handling the exceptions of sql connection

import java.util.Random; //for randomly assigning the background color of the pages

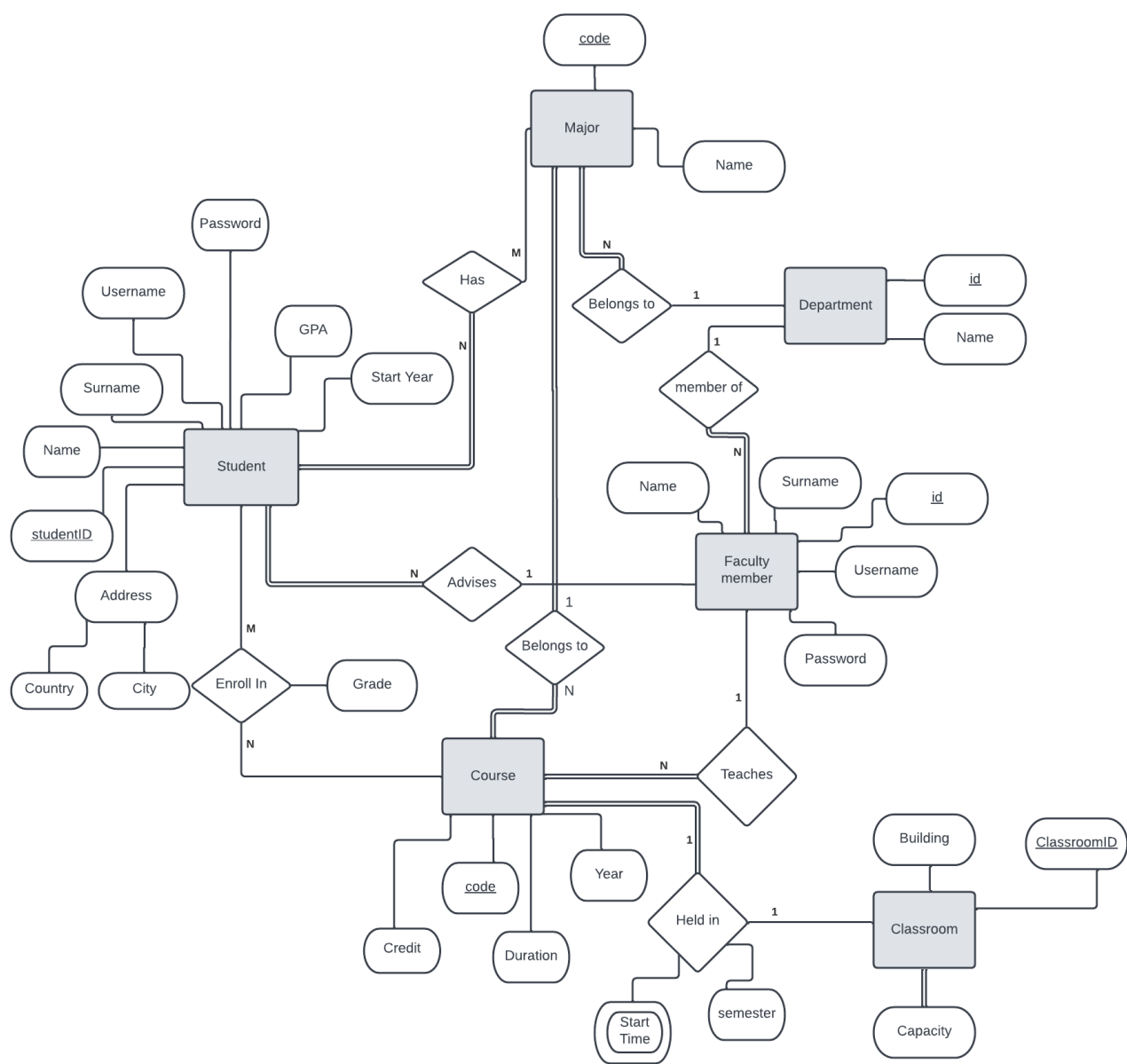
import javax.swing.ImageIcon; //for Koç University Logos in the opening panel

import javax.swing.JLabel; //for displaying the indicator texts

import javax.swing.JOptionPane; //for showing popup messages often for the invalid and/or wrong cases

import javax.swing.JPasswordField; //for keeping entrance passwords in GUI

Entity-Relationship Diagram



System Design

1- Student panel:

- Check the courses enrolled in
- Check personal information
- Enroll in a course
- Drop a course
- Log out from system
- Change password

2- Admin panel:

- Check information
- Change password
- Add a new course
- Arrange a new course with a teacher at a specific classroom
- Check available class rooms
- check the course taken by the students from the maximum number of different countries
- checks most successful students by year
- checks the faculty members with the least workload
- checks the potential advisors for a student
- Log out from the system
- checks most visited classrooms
- checks the best advisors

3- Faculty member panel:

- Grades a student
- See course schedule
- Log out from the system

Relational Database Design

```
CREATE TABLE Department (  
ID VARCHAR(20) NOT NULL,  
name VARCHAR(40) NOT NULL,  
PRIMARY KEY (ID)  
)
```

```
CREATE TABLE FacultyMember (  
ID INT NOT NULL,  
Name VARCHAR(20) NOT NULL,  
Surname VARCHAR(20) NOT NULL,  
DepID VARCHAR(20) NOT NULL,  
Username VARCHAR(20) NOT NULL,  
Password VARCHAR(40) NOT NULL,  
FOREIGN KEY (DepID) REFERENCES Department (ID),  
PRIMARY KEY (ID)  
)
```

```
CREATE TABLE student (  
StudentID INT NOT NULL,  
Name VARCHAR(30) NOT NULL,  
Surname VARCHAR(30) NOT NULL,  
GPA DOUBLE NOT NULL,  
Start_year INT NOT NULL,  
country VARCHAR(30) NOT NULL,  
city VARCHAR(30) NOT NULL,  
AdvisorID INT NOT NULL,
```

```
Username VARCHAR(20) NOT NULL,  
Password VARCHAR(40) NOT NULL,  
FOREIGN KEY (AdvisorID) REFERENCES FacultyMember (ID),  
PRIMARY KEY (StudentID)  
)
```

```
CREATE TABLE major (  
Code CHAR(4) NOT NULL,  
Name VARCHAR(40) NOT NULL,  
DepID VARCHAR(20) NOT NULL,  
FOREIGN KEY (DepID) REFERENCES Department (ID),  
PRIMARY KEY (Code)  
)
```

```
CREATE TABLE admin (  
id INT NOT NULL,  
username VARCHAR(40) NOT NULL,  
password VARCHAR(40) NOT NULL,  
PRIMARY KEY (id)  
)
```

```
CREATE TABLE Classroom (  
ClassroomID INT NOT NULL,  
Building VARCHAR(30) NOT NULL,  
Capacity INT NOT NULL,  
PRIMARY KEY (ClassroomID)  
)
```

```
CREATE TABLE Course (  
    Duration INT NOT NULL,  
    Credit INT NOT NULL,  
    Code VARCHAR(10) NOT NULL,  
    ClassroomID INT NOT NULL,  
    Semester VARCHAR(20) NOT NULL,  
    Year INT NOT NULL,  
    majorCode CHAR(4) NOT NULL,  
    teacherID INT NOT NULL,  
    FOREIGN KEY (teacherID) REFERENCES FacultyMember (ID),  
    FOREIGN KEY (majorCode) REFERENCES Major (Code),  
    FOREIGN KEY (ClassroomID) REFERENCES Classroom (ClassroomID),  
    PRIMARY KEY (code)  
)
```

```
CREATE TABLE EnrollIn (  
    StudentID INT NOT NULL,  
    CourseCode VARCHAR(10) NOT NULL,  
    Grade VARCHAR(20),  
    FOREIGN KEY (CourseCode) REFERENCES Course (Code),  
    FOREIGN KEY (StudentID) REFERENCES Student (StudentID),  
    PRIMARY KEY (StudentID, CourseCode)  
)
```

```
CREATE TABLE Has (  
    StudentID INT NOT NULL,  
    MajorCode VARCHAR(4) NOT NULL,
```

```
FOREIGN KEY (MajorCode) REFERENCES Major (Code),  
FOREIGN KEY (StudentID) REFERENCES Student (StudentID),  
PRIMARY KEY (StudentID, MajorCode)  
)
```

```
CREATE TABLE StartTime (  
StartTime VARCHAR(20) NOT NULL,  
CourseCode VARCHAR(10) NOT NULL,  
FOREIGN KEY (CourseCode) REFERENCES Course (Code),  
PRIMARY KEY (CourseCode, StartTime)  
)
```

Data Sources

Explain how you populate your database, i.e., where did you get your data?

We populated our database by creating data inside the following website (for only the student table and the faculty member table inside the database called “minikusi”):

<https://www.mockaroo.com/>

Since we do not need much data for the tables other than tables called “student” and “facultymember”, we have manually populated the tables which are other than the tables called “student” and “facultymember”.

NOTE: Except for the above link, we have researched famous data sources such as “Kaggle”, “Datahub.io”, “Data.gov”, and “Google Dataset Search”. However, from these data sources, we could not find data which is suitable for our project.

Complex SQL Queries

Your 5 complex SQL queries: copy-paste the queries here, explain briefly what they do, why they are useful in your project, where have they been integrated in your prototype.

First Query:

Meaning of the query: For each year, find the student name, student start year, maximum student gpa where the student start years are ordered in ascending order.

Query:

```
SELECT s.start_year, s.name, s.gpa
FROM Student s
WHERE s.gpa in (SELECT MAX(s1.gpa)
                FROM student s1
                GROUP BY (s1.start_year))
ORDER BY (s.start_year) ASC;
```

Where we have integrated the first query: We have integrated the first query inside the java file called "AdminPanel.java".

Second Query:

Meaning of the query: We want to find the most multinational course enrolled by students (i.e. The class taken by students who have the largest number of different countries).

Query:

```
SELECT count(distinct(s1.country)), C.code
FROM Student s1, Course C, Enrollin E
WHERE E.CourseCode = C.Code AND E.StudentID = s1.StudentID
GROUP BY C.Code
ORDER BY count(distinct(s1.country)) desc
LIMIT 1;
```

Where we have integrated the second query: We have integrated the second query inside the java file called "AdminPanel.java".

Admin uses this query to see which class has the students from the most number of different nationalities.

Third Query:

Meaning of the query: Find the classroom building, classroom id, and the number of visits of the students to the classroom where the number of the visits to the classroom is the maximum.

Query:

```
SELECT building, classroom.classroomid, count(*) as total_visit_to_classroom
FROM classroom, course, enrollin, student
WHERE course.classroomid=classroom.classroomid AND enrollin.studentid=student.studentID
AND enrollin.coursecode = course.code
GROUP BY (classroom.classroomid)
ORDER BY count(*) desc
LIMIT 1;
```

Where we have integrated the third query: We have integrated the third query inside the java file called "AdminPanel.java".

Fourth Query:

Meaning of the query: Find the count of advisees of the faculty members who teaches less than 3 courses. Admin uses this query to check which faculty members have the least workload.

Query:

```
SELECT f.id, f.name, count(*) as advisees
FROM facultymember f, student s
WHERE f.id = s.advisorid AND id in (SELECT id
                                   FROM facultymember f2, course c
                                   WHERE f2.id=c.teacherid
                                   GROUP BY (f2.id)
                                   HAVING count(*) < 3)
GROUP BY (id);
```

Where we have integrated the fourth query: We have integrated the fourth query inside the java file called "AdminPanel.java".

Fifth Query:

Meaning of the query: Find the faculty member name, the faculty member surname, and the average gpa of the students advised by the faculty members where the number of the advised

students is greater than 3 and where the average GPAs of the advised students are ordered in the descending order.

The aim is to find the most successful advisors.

Query:

```
SELECT f.name, f.surname, avg(gpa)
FROM student s, facultymember f
WHERE s.advisorid = f.id
GROUP BY (advisorID)
HAVING count(*) > 3
ORDER BY (avg(gpa)) desc;
```

Where we have integrated the fifth query: We have integrated the fifth query inside the java file called "AdminPanel.java".

Sixth Query

Meaning of the Query: Find potential advisors for a given student ID. The advisors should belong to the same department as one of the students' majors. Order all potential advisors by the number of advisees that they have. In the sixth query written below, "SOMEID" refers to the student id which is inputted by an admin.

Query:

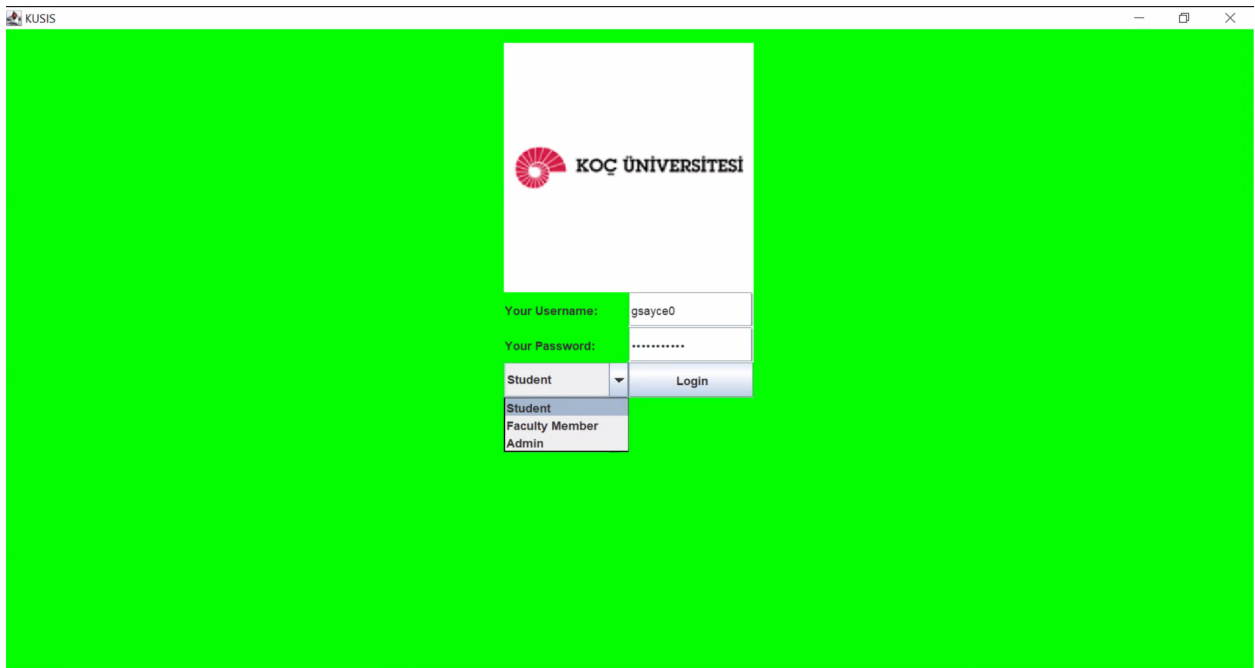
```
SELECT F.id, F.name, count(AD.studentID) as advisee_cnt from Facultymember as F
    JOIN Department as D on F.DepID = D.Id
    JOIN Student as AD on AD.AdvisorID = F.ID
WHERE D.id IN (SELECT M.deplD
    FROM Major as M JOIN Has as H on M.code = H.MajorCode
    JOIN Student as S on H.StudentID = S.studentID
    WHERE S.ID = SOMEID)
GROUP BY (F.id)
ORDER BY advisee_cnt ASC
```

Where we have integrated the sixth query: We have integrated the sixth query inside the java file called "AdminPanel.java".

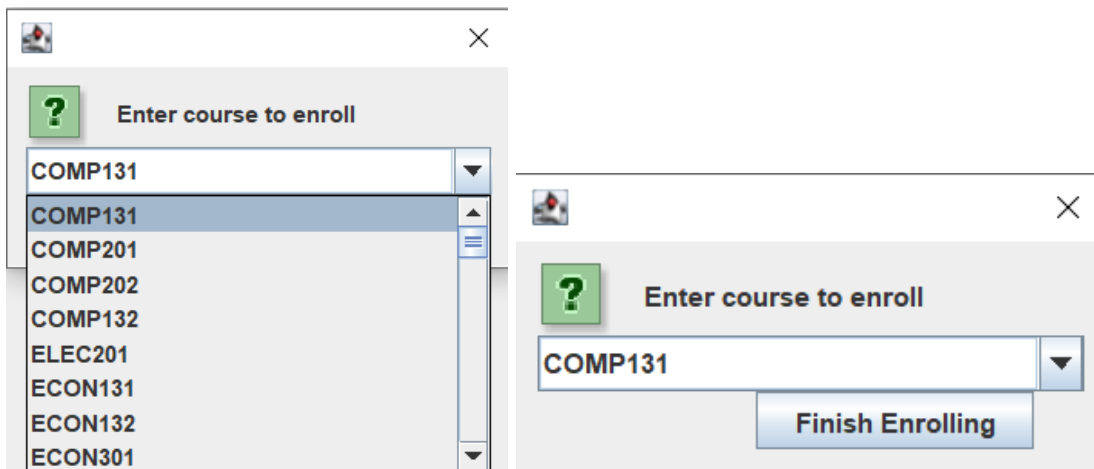
Screenshots

Add a few screenshots from the finalized version of your prototype. A video demo is also fine.

Opening Panel




While a student is enrolling in a course.




While a student is viewing his/her courses.



Message ×

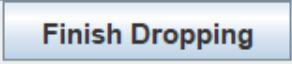




Code	ClassroomID	Semester	Year	Start Time	Duration	Credit	Grade
COMP131	2	Spring	2022	Wednesday - 12:00	2	3	
COMP202	5	Spring	2022	Wednesday - 18:00	2	4	



While a student is dropping a course.

 Enter course to drop

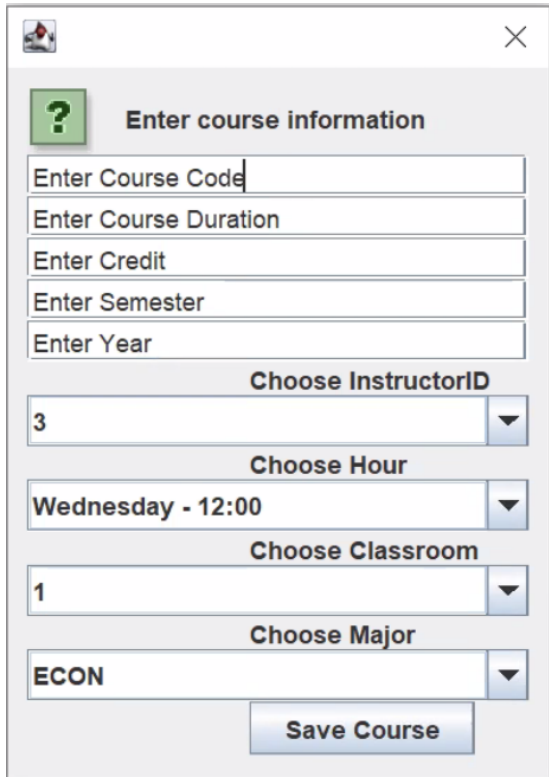


 Enter course to drop

COMP131

COMP202

While an admin is adding a course to the system.



A screenshot of a software dialog box titled "Enter course information". The dialog box has a close button (X) in the top right corner. It contains several input fields and dropdown menus. The fields are: "Enter Course Code", "Enter Course Duration", "Enter Credit", "Enter Semester", and "Enter Year". Below these are three dropdown menus: "Choose InstructorID" (showing "3"), "Choose Hour" (showing "Wednesday - 12:00"), and "Choose Classroom" (showing "1"). Below these is another dropdown menu: "Choose Major" (showing "ECON"). At the bottom right is a "Save Course" button.

Enter course information

Enter Course Code

Enter Course Duration

Enter Credit

Enter Semester

Enter Year

Choose InstructorID

3

Choose Hour

Wednesday - 12:00

Choose Classroom

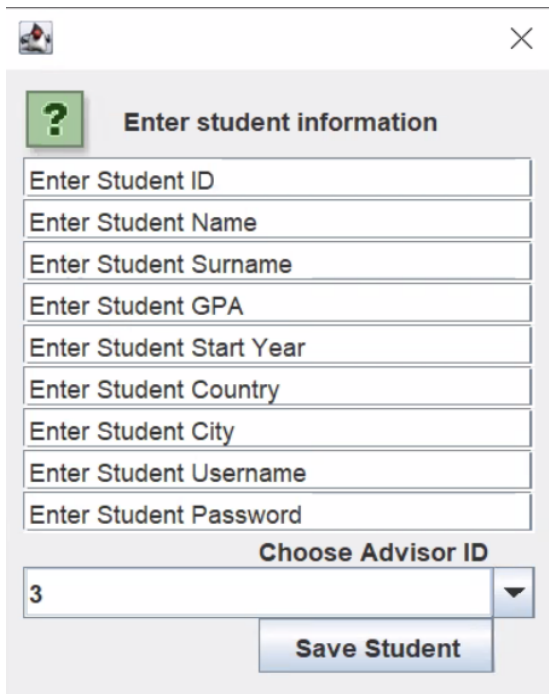
1

Choose Major

ECON

Save Course

While an admin is adding a student to the system.



A screenshot of a software dialog box titled "Enter student information". The dialog box has a close button (X) in the top right corner. It contains several input fields and one dropdown menu. The fields are: "Enter Student ID", "Enter Student Name", "Enter Student Surname", "Enter Student GPA", "Enter Student Start Year", "Enter Student Country", "Enter Student City", "Enter Student Username", and "Enter Student Password". Below these is a dropdown menu: "Choose Advisor ID" (showing "3"). At the bottom right is a "Save Student" button.

Enter student information

Enter Student ID

Enter Student Name

Enter Student Surname

Enter Student GPA

Enter Student Start Year

Enter Student Country

Enter Student City

Enter Student Username

Enter Student Password


Choose Advisor ID

3


Save Student

While an instructor is viewing his/her schedule.

Message ×




Code	classroomid	Semester	Year	Start Time	Duration	Credit
COMP131	2	Spring	2022	Wednesday - 12:00	2	3
COMP201	4	Spring	2022	Thursday - 15:00	2	3
COMP202	5	Spring	2022	Wednesday - 18:00	2	4




While an instructor is entering the letter grades of the course codes.

Message ×



Code	studentid	name	grade
COMP131	1	Garland	
COMP131	4	Toby	
COMP131	6	Adrien	
COMP131	9	Louise	
COMP131	10	Lorrie	
COMP131	13	Charin	
COMP131	14	Creighton	
COMP131	15	Ree	
COMP131	19	Holt	
COMP131	24	Nicky	
COMP131	26	Allene	
COMP131	35	Gael	
COMP131	40	Kevyn	
COMP131	42	Tally	
COMP131	48	Sara	
COMP131	50	Gasparo	
COMP201	2	Randal	
COMP201	4	Toby	
COMP202	1	Garland	
COMP202	2	Randal	
COMP202	6	Adrien	
COMP202	9	Louise	
COMP202	10	Lorrie	
COMP202	13	Charin	
COMP202	14	Creighton	
COMP202	15	Ree	
COMP202	19	Holt	
COMP202	24	Nicky	
COMP202	26	Allene	



Admin checks the faculty members with the least workload.


[illegible]

Admin checks most successful students by year.

[illegible]

Admin checks the course taken by the students from the maximum number of different nations.

Message ×



Course	Number of Different Nations
COMP131	5

OK

Admin checks most visited classrooms.

[illegible]

Admin sees best advisors.

[illegible]

References

The logo used in source code is taken from:

<https://www.logovector.org/logo/koc-university/>

For downloading the mysql-connector jar for java:

<https://dev.mysql.com/downloads/connector/j/>

<https://www.geeksforgeeks.org/java-database-connectivity-with-mysql/>

For GUI Implementation of the Project:

<https://www.geeksforgeeks.org/java-swing-jtable/>

For Populating the Database:

<https://www.mockaroo.com>

For the Complex SQL Queries:

COMP-306 Lecture Slides

COMP-306 PS Slides

NOTES

- In order to run the project, you should run the "Kuis.java" file.
- For connecting to the mysql database, you should write down your username and your MYSQL password to the necessary part. (i.e. replace the mysql username and mysql password with yours).
- In our project, admins add the courses to the system. Since we did not use a relation between the admins and courses (such as list the courses added by the admins) inside the system and since we treated the admin entity as a separate entity (here, a separate entity means that an entity having no relation with any other entities) , we have not included it in our ER diagram. Moreover, we did not keep the "admin.id" attribute inside the "Course" entity in the ER diagram and in the relational database design.