PROJECT FINAL REPORT

STUDENT RECORDS

Berkcan Altungöz - 20170808014

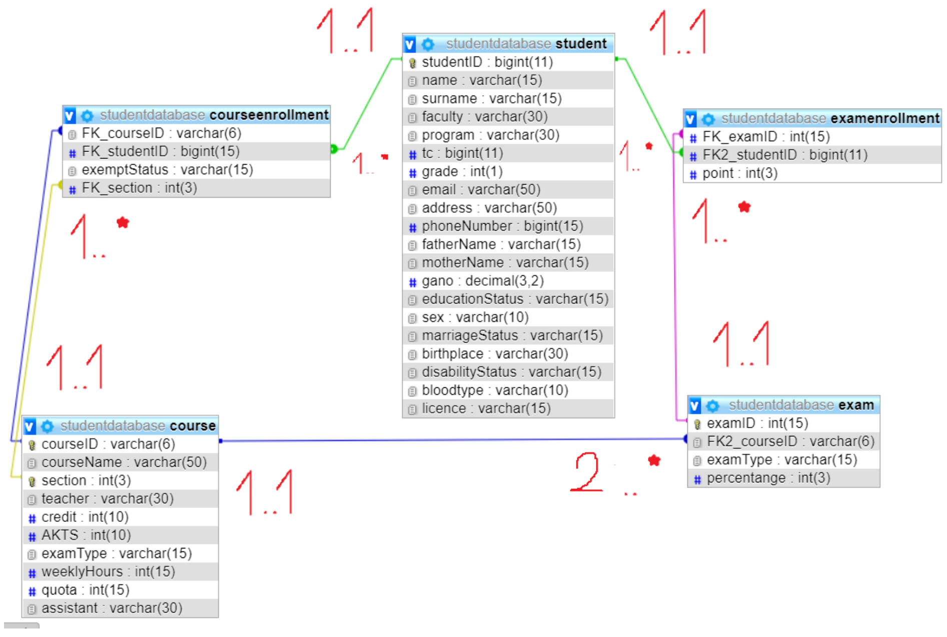
Mirvan Sadıglı – 20160807004

Tunahan Burak Dirlik – 20170808076

Nasip Efe Tığlı - 20170808020

After our first assignment making the transition from UNF table to 3NF table and inserting the data into them with our ER diagram ready we fixed some issues that were present in our ER diagram logic. And went on to implementing the UI and the queries requested from us. Our now fixed and fully working creating the tables and inserting data queries are included in the ZIP file.

This Is Our Final ER Diagram



First we made the student table. It includes personal and student information of the individual. It has the studentID primary key. It connects to the enrollment tables by this key.

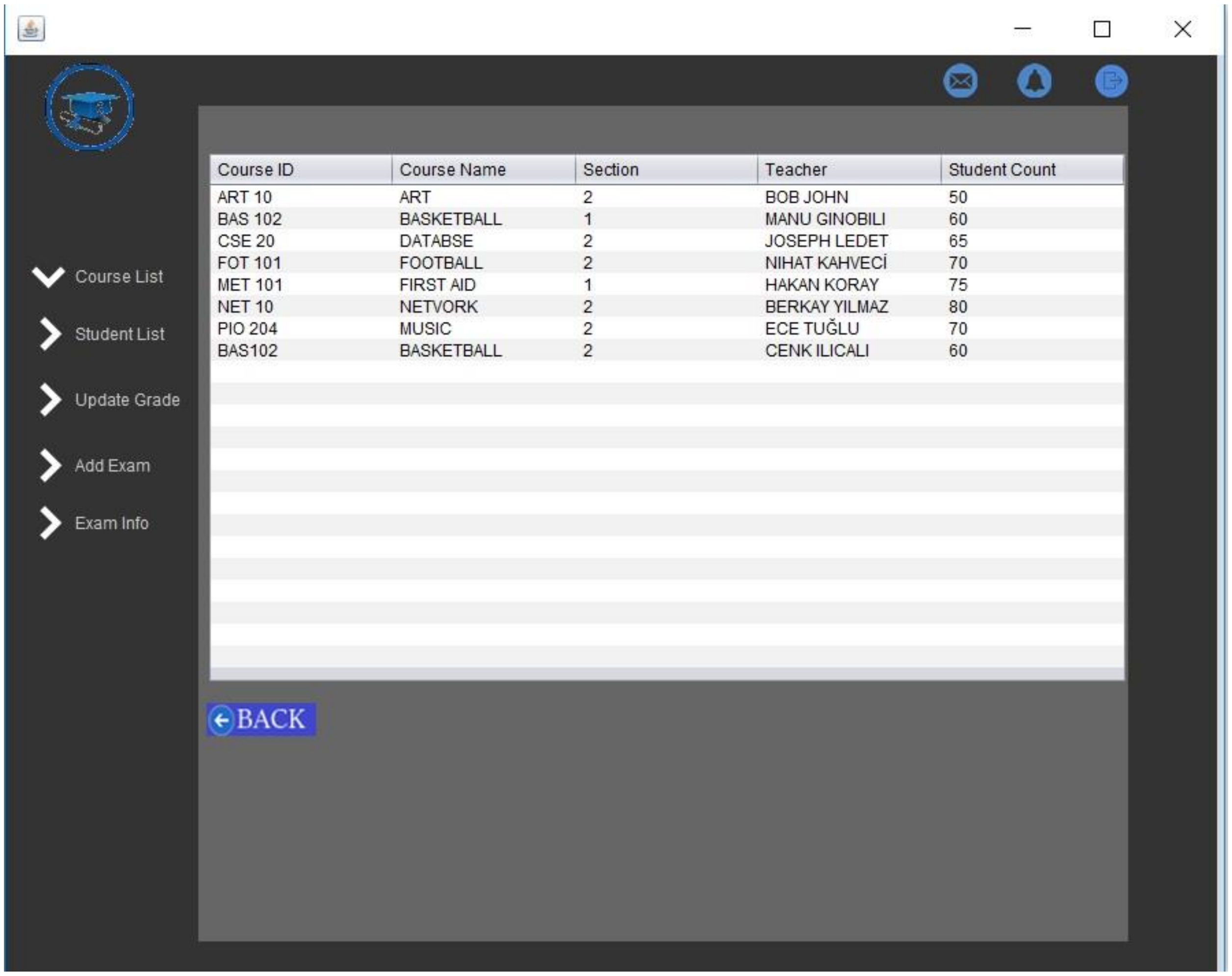
Secondly since students take courses we created the course table and connected it to the course enrollment table that keeps track of which course the student is enrolled in. Course table has the information regarding the course which has attributes like course name , credit , it's teacher etc. It has a composite primary key by using the courseID and the section attributes, since different sections of the same course has the same courseID. It connects to the enrollment table by using the courseID attribute. Enrollment table has the student's exempt status attribute as well.

Thirdly we created the exam table since courses had to have exams to pass them. It has the examID primary key. This table is connected to the exam enrollment table that keeps track of which exams the student participated in and which course it belongs to. Exam table has attributes like it's percentage of the total grade and type be it midterm,final or quiz etc. Exam enrollment table has the student's grade on that exam.

On the relation side of the diagram logically a student can have one or more courses, a course must have at least two or more exams. Similarly a course can have one or more students and an exam must have one or more participating student.

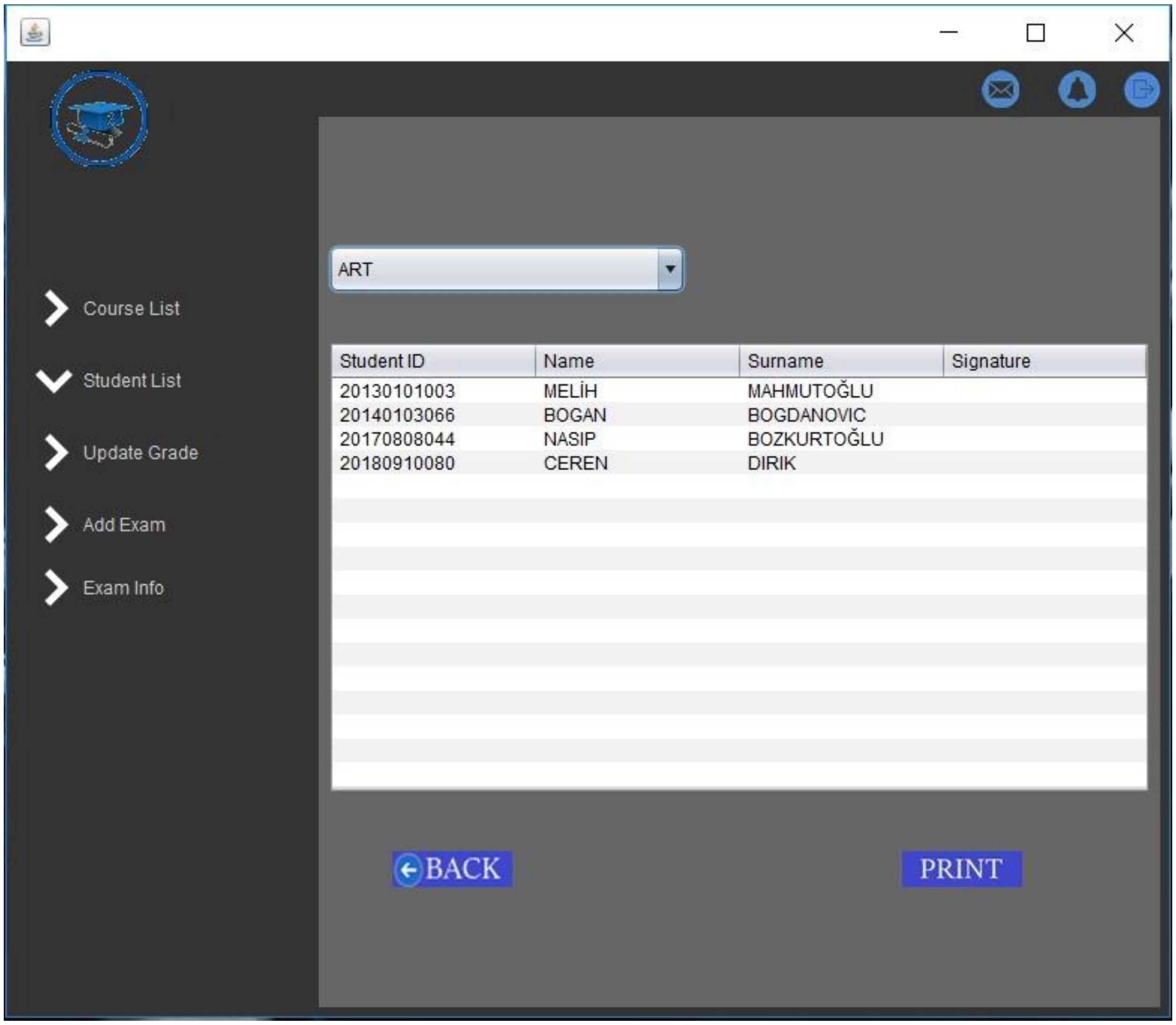
After we made our database ready we went on to make the interface and the requested queries from our customer simultaneously since we had to make extra queries based on our UI’s requested input and similarly we had to take account the queries to make the UI.

This Is Our User Interface



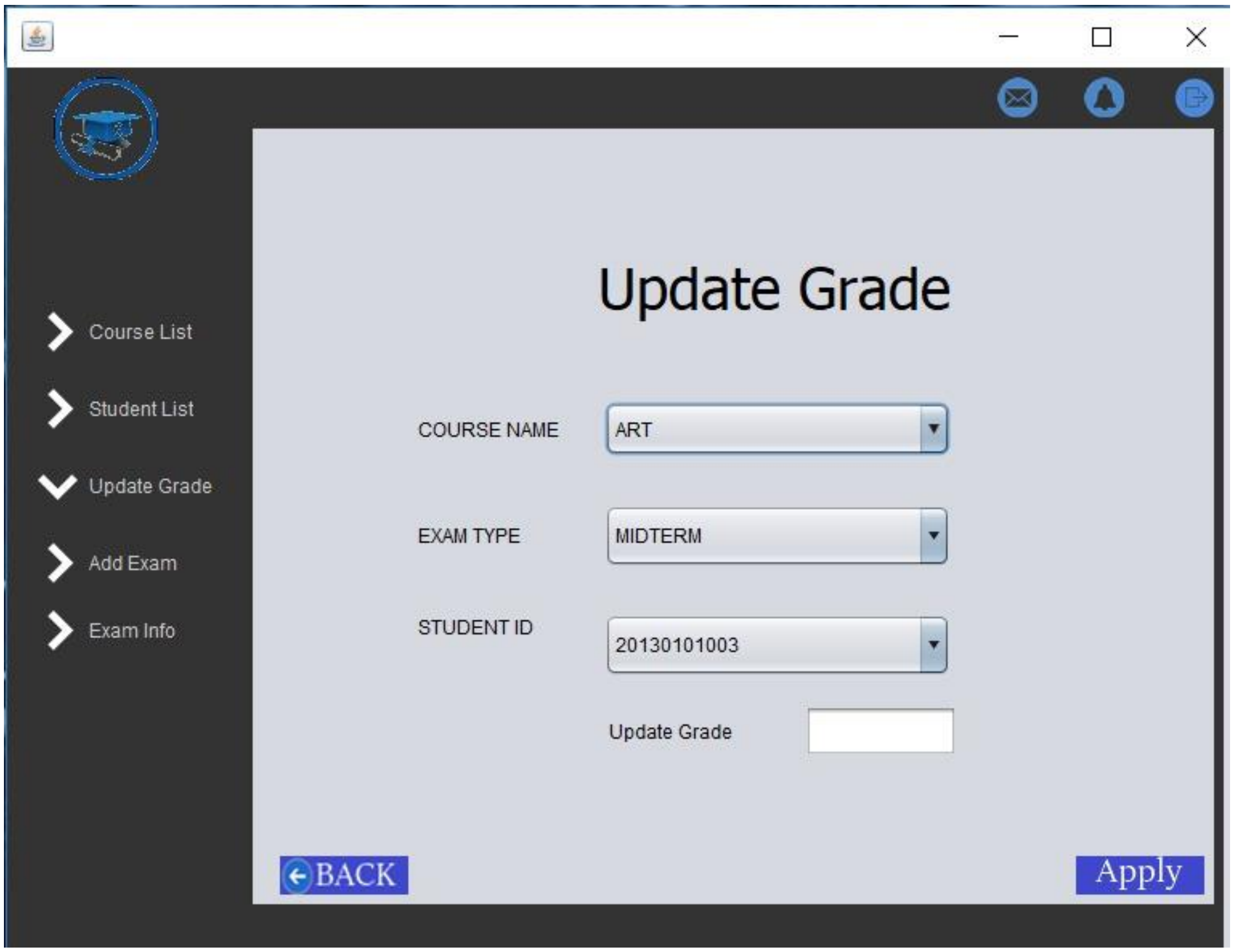
Course ID	Course Name	Section	Teacher	Student Count
ART 10	ART	2	BOB JOHN	50
BAS 102	BASKETBALL	1	MANU GINOBILI	60
CSE 20	DATABSE	2	JOSEPH LEDET	65
FOT 101	FOOTBALL	2	NIHAT KAHVECİ	70
MET 101	FIRST AID	1	HAKAN KORAY	75
NET 10	NETVORK	2	BERKAY YILMAZ	80
PIO 204	MUSIC	2	ECE TUĞLU	70
BAS102	BASKETBALL	2	CENK ILICALI	60

This is the first page of our UI. It lists courses that are currently being taught in that semester and the total count of students that are taking that course.



Student ID	Name	Surname	Signature
20130101003	MELİH	MAHMUTOĞLU	
20140103066	BOGAN	BOGDANOVIC	
20170808044	NASIP	BOZKURTOĞLU	
20180910080	CEREN	DIRIK	

By clicking the second button on the left side of the interface we can see a sign sheet for an exam. Since a sign sheet consists of the students that are currently enrolled in that course the users select the course for that exam. The user can directly print the sign sheet for ease of use.



Update Grade

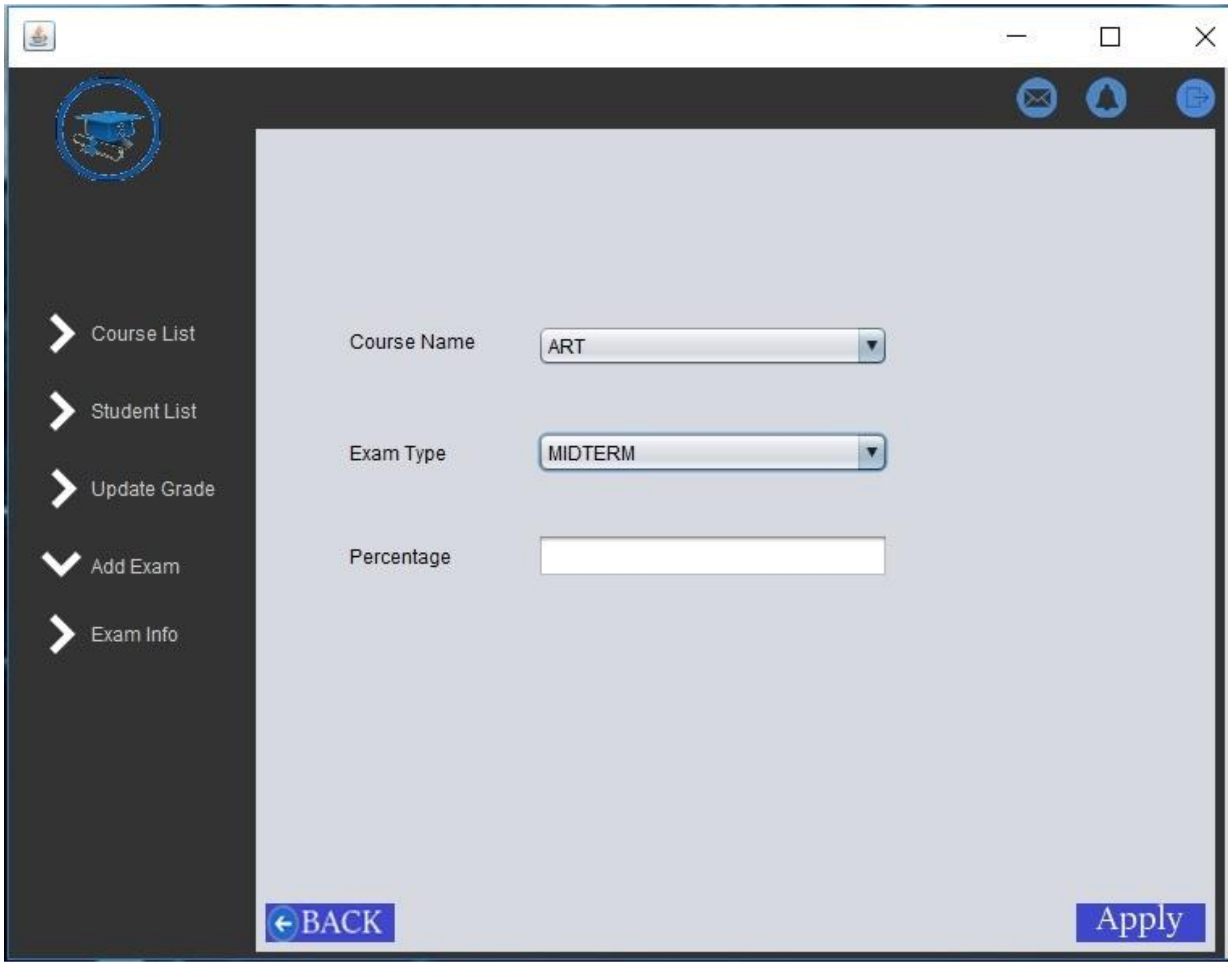
COURSE NAME: ART

EXAM TYPE: MIDTERM

STUDENT ID: 20130101003

Update Grade

Third feature we have on our interface is to update a student's grade that they received on a specific exam. It lists the courses in a combo box when the course is selected the second combo box lists the exams that are currently in the database for that course. When selected third combo box lists the students that participated in that exam. By selecting all three the user can input the new grade on the text field and click apply to finish the process.

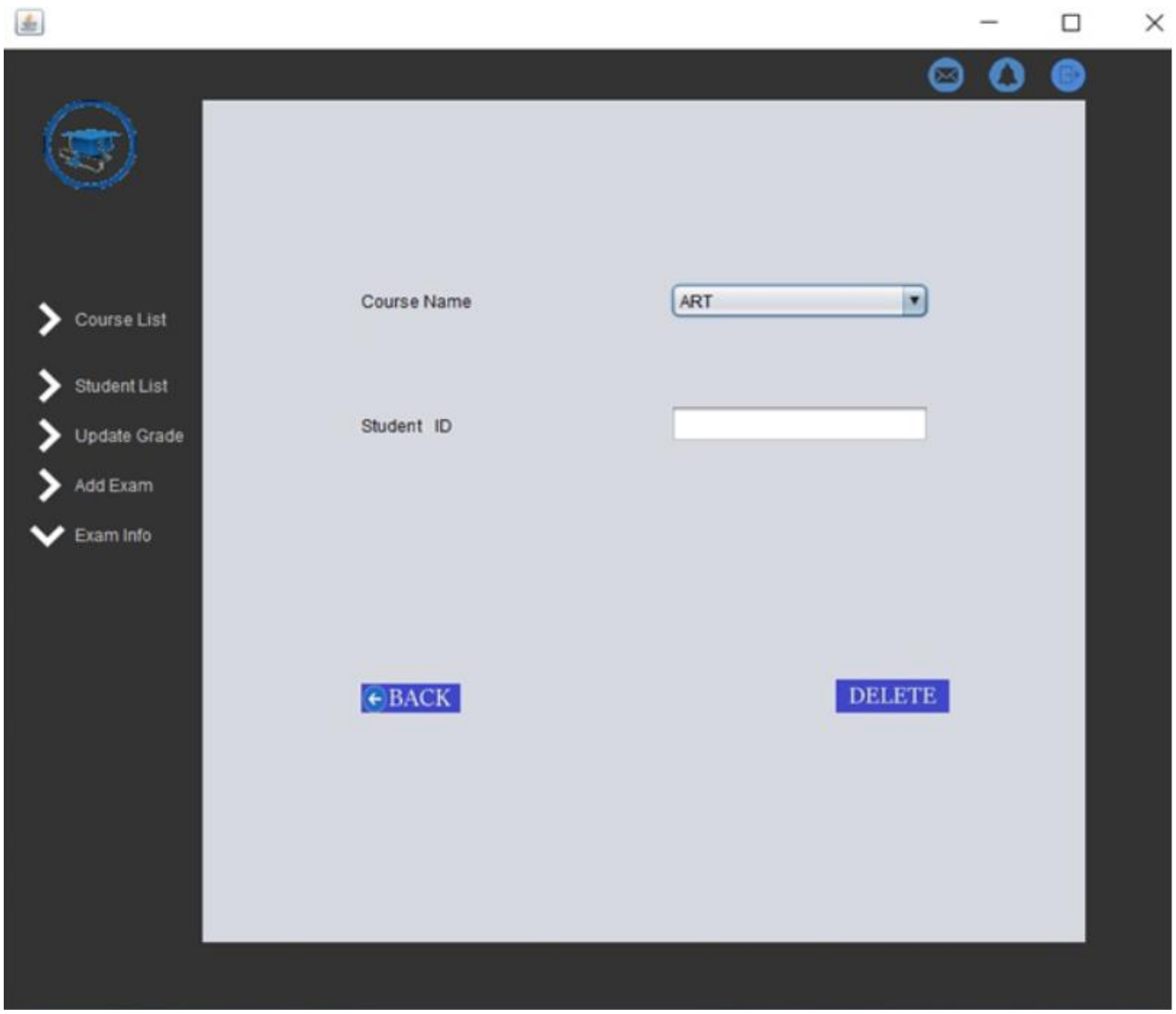


Course Name: ART

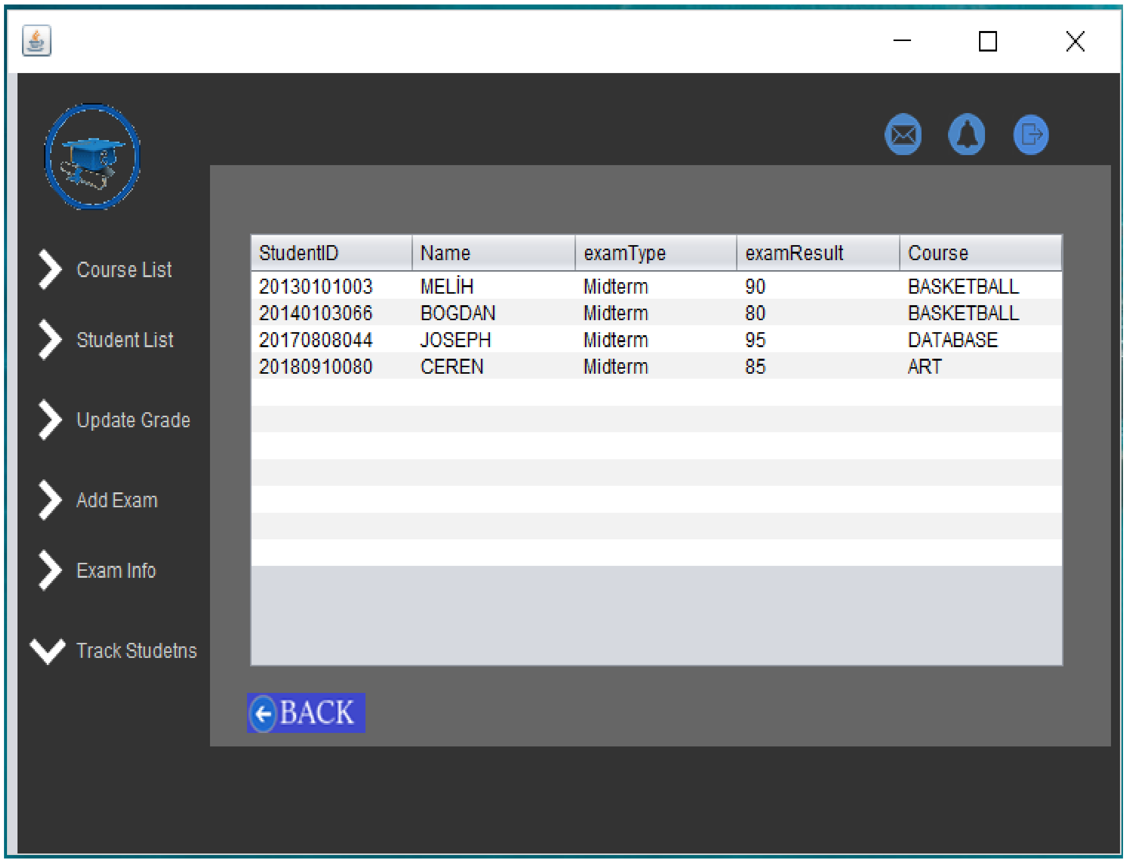
Exam Type: MIDTERM

Percentage:

Another feature is adding an exam to a course. The user is asked to select a course that they are supposed to add an exam to. Then they will be prompted to select an exam type by another combo box. After which they will input the percentage the exam will have on the final grade.



Another feature our application has is the option to remove a student from a course and all associated exams. The user selects a course and inputs a studentID. After clicking the delete button the student will be deleted from that course.



StudentID	Name	examType	examResult	Course
20130101003	MELİH	Midterm	90	BASKETBALL
20140103066	BOGDAN	Midterm	80	BASKETBALL
20170808044	JOSEPH	Midterm	95	DATABASE
20180910080	CEREN	Midterm	85	ART

Lastly we created a view to see students' exam results and which course they are enrolled to make our user's experience smoother.

These Are Our Queries For The Interface

1. SELECT c.courseID , c.courseName , c.section , c.teacher , COUNT(c.courseID) AS StudentCount

FROM course c , courseenrollment ce

WHERE c.courseID = ce.FK_courseID AND c.section = ce.FK_section

GROUP BY c.courseID , c.courseName , c.section , c.teacher

ORDER BY c.courseID

First query we have is for the combo box selection for the courses. Second query takes that input and pulls the students that currently enrolled in that course using the selected courseID.

2. SELECT DISTINCT courseID

FROM course

SELECT DISTINCT s.studentID , s.name , s.surname

FROM course c, student s, courseenrollment ce

WHERE s.studentID = ce.FK_studentID AND ce.FK_courseID = courseIDInput

The query works by pulling the courses from the course table and the count of the students from the course enrollment table connected by the courseID primary key.

3. SELECT DISTINCT courseID

FROM course

SELECT e.examType

FROM exam e, course c

WHERE courseIDInput = c.courseID AND c.courseID = e.FK2_courseID

SELECT s.studentID

FROM course c , exam e , examenrollment ee , courseenrollment ce , student s

WHERE e.examID = examIDInput AND e.examID = ee.FK_examID AND ee.FK2_studentID = s.studentID AND c.courseID = ce.FK_courseID AND ce.FK_studentID = s.studentID

UPDATE examenrollment

SET point = pointinput

WHERE examIDInput = FK_examID AND studentIDInput = FK2_studentID

The first query pulls the distinct courses from the database.The second query pulls the exam types according to the inputted course using courseID primary key. Similary by using the two previous information it pulls the students'ID that took that exam using the connecting primary and foreign keys accordingly.After which the updated query is inputted then applied the database runs the update query that finishes the update application.

4.

```
    SELECT DISTINCT courseID
FROM  course
```

```
    SELECT e.examType
FROM exam e, course c
WHERE courseIDInput = c.courseID AND c.courseID=e.FK2_courseID
```

```
    INSERT INTO exam (FK2_courseID, examType, percentage)
VALUES ('courseIDInput','examTypeInput','percentageInput');
```

The first query is for listing the courses from the database. Second query takes the inputted courseID and pulls the exam types for that course using the courseID primary key.

After inputting the percentage on the text field the third query takes these values and inserts them to the exam table in the database.

5. SELECT DISTINCT courseID

FROM course

DELETE FROM courseenrollment

WHERE courseIDInput = FK_courseID AND studentIDInput = FK_studentID (BECAUSE CASCADE)

The first query lists the courses that are currently in the database. The second query takes the selected courseID and inputted studentID from the text field and puts them in a where clause in the delete query. Since keys are connected by CASCADE ON DELETE restriction. The exams the student is associated with will be deleted.

6. CREATE VIEW studentGrades

AS SELECT DISTINCT

s.studentID,s.name,s.surname,c.courseID , c.courseName , e.examType, ee.point

FROM examenrollment ee, exam e , course c , courseenrollment ce, student s

WHERE s.studentID = ee.FK2_studentID AND ee.FK_examID = e.examID AND e.FK2_courseID = c.courseID AND c.courseID = ce.FK_courseID AND ce.FK_studentID = s.studentID

This query pulls distinct information about the students' from the connected tables and separates them according to the studentID primary key with help from the other keys to connect the tables.

Thank You For Reading!

This concludes our project. We actually managed to connect the login page to the database on our own computers but it probably won’t work on another computer. Thanks for teaching us SQL to the best of your ability and making us improve by assigning this project. Have a good day.