

# Final Project

CSE 640 - 50

Barrett Gamber

4/24/2022

# Assignment

This assignment tasked me with developing a web app for managing a database of multiple tables and using the pooling connection method. I decided to make an app for organizing information about the students and professors of an engineering university. It is based off of my proposal that I put forward in Assignment 3.

# Overview

My app consists of 3 tables. The first of these is a table to keep track of the students. The columns are an auto-incremented id (serves as the primary key), a first name, a last name, the major being studied, the student's GPA, and whether they are a graduate student or not. The other main table is for professors. The columns in this table include an auto-incremented id (serves as the primary key), a first name, a last name, the department they are a part of, the class/classes they teach, and the semesters that they teach. The major and department of the two tables are used as foreign keys in reference to the third table. The third table is a list of all the majors/departments in the engineering school, each associated with an auto-incremented id. My app allows for the viewing of all three of these tables as lists. The student and professor tables are the only ones that can be edited. My app allows for the adding, deleting, and editing of both the student and professor tables.

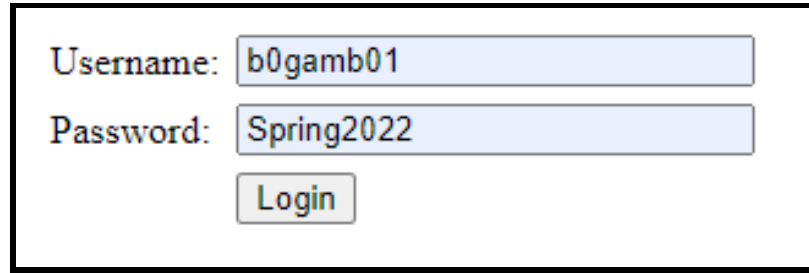
# Operation

## **Welcome to the Database of Engineering Students and Professors!**

Use this app to manage the students and professors on the engineering university's database.

[Login](#)

When the user begins my app they are greeted with a welcome page with a link to login. This can be seen above. By clicking this link the user can log in. The user credentials for this are "b0gamb01" as the username, and "Spring2022" as the password. If the user does not log in, they will be unable to access the database. The user is then greeted with the welcome screen again, but this time, with all of the available functions of the app displayed. Screenshots pertaining to this can be seen on the next page.



A login form with a black border. It contains two text input fields: the first is labeled 'Username:' and contains the text 'b0gamb01'; the second is labeled 'Password:' and contains the text 'Spring2022'. Below the password field is a 'Login' button.

Username:	<input type="text" value="b0gamb01"/>
Password:	<input type="password" value="Spring2022"/>
	<input type="button" value="Login"/>

## Welcome to the Database of Engineering Students and Professors!

Use this app to manage the students and professors on the engineering university's database.

You have been logged in successfully!

[Student List](#)

[Professor List](#)

[Department List](#)

[Add Student](#)

[Delete Student](#)

[Edit Student](#)

[Add Professor](#)

[Delete Professor](#)

[Edit Professor](#)

[Login](#)

One may also notice the successful login message. A message is also displayed any time the database is modified to provide confirmation that the operation was successful. This will be demonstrated later as well.

By clicking on any one of the list links the respective table will be displayed. Screenshots of this can be seen below. Also notable is the main menu link. This link is present on every JSP that isn't the main menu. By clicking it, the user can be returned to the main menu with ease.

## List of Students

ID	First Name	Last Name	Major	GPA	Graduate?
6	Barrett	Gamber	CSE	3.38	Yes
21	Shaun	Doe	ME	3.0	No

[Return to Main Menu](#)

## List of Professors

ID	First Name	Last Name	Department	Class Taught	Active Semesters
23	Ibrahim	Imam	CSE	CSE-640	Fall

[Return to Main Menu](#)

## List of Departments

ID	Department Name
1	CSE
2	ECE
3	ME
4	IE
5	CHE

[Return to Main Menu](#)

These list links are followed by the student and professor database editing links. When a user clicks on the student/professor add, they are greeted by a respective JSP, as displayed in the screenshots below. I have gone ahead and added data to the boxes of both the student and professor JSPs for demonstration purposes. The first screenshot is the student add JSP and the second is the professor counterpart.

First Name:	<input type="text" value="John"/>
Last Name:	<input type="text" value="Smith"/>
Major:	<input type="text" value="ECE"/>
GPA:	<input type="text" value="3.78"/>
Graduate? (Yes/No):	<input type="text" value="Yes"/>
<input type="button" value="Add to List"/>	
<a href="#">Return to Main Menu</a>	

First Name:	<input type="text" value="Thomas"/>
Last Name:	<input type="text" value="Berfield"/>
Department:	<input type="text" value="ME"/>
Teaches:	<input type="text" value="ME-520"/>
Active Semesters:	<input type="text" value="Fall, Spring"/>
<input type="button" value="Add to List"/>	
<a href="#">Return to Main Menu</a>	

By clicking add to list, this student/professor is added to the respective student or professor list. Servlets are responsible for transferring this data to the database. The user is then returned to the main menu with a message indicating whether or not the database commit was successful. Screenshots of this can be seen below.

# Welcome to the Database of Engineering Students and Professors!

Use this app to manage the students and professors on the engineering university's database.

Student successfully added to database!

[Student List](#)

[Professor List](#)

[Department List](#)

[Add Student](#)

[Delete Student](#)

[Edit Student](#)

[Add Professor](#)

[Delete Professor](#)

[Edit Professor](#)

[Login](#)

# Welcome to the Database of Engineering Students and Professors!

Use this app to manage the students and professors on the engineering university's database.

Professor successfully added to database!

[Student List](#)

[Professor List](#)

[Department List](#)

[Add Student](#)

[Delete Student](#)

[Edit Student](#)

[Add Professor](#)

[Delete Professor](#)

[Edit Professor](#)

[Login](#)

Upon checking the student and professor lists, we can see that the commits to the database were indeed successful. The new student John Smith and new professor Thomas Berfield can now be seen on the list.

## List of Students

ID	First Name	Last Name	Major	GPA	Graduate?
6	Barrett	Gamber	CSE	3.38	Yes
21	Shaun	Doe	ME	3.0	No
22	John	Smith	ECE	3.78	Yes

[Return to Main Menu](#)

## List of Professors

ID	First Name	Last Name	Department	Class Taught	Active Semesters
23	Ibrahim	Imam	CSE	CSE-640	Fall
41	Thomas	Berfield	ME	ME-520	Fall, Spring

[Return to Main Menu](#)

The next of these editing functions that I implemented was deletion. This was the simplest of the database manipulation tasks to implement. To delete a student or professor, the user is prompted for the auto incremented id associated with that student or professor. We will delete the student and professor that we just previously created, John Smith and Thomas Berfield. Their respective ids are 22 and 41, so we will put this into the boxes of the deletion JSPs. This can be seen on the next page.



Enter ID of Student to Delete:

[Return to Main Menu](#)

Enter ID of Professor to Delete:

[Return to Main Menu](#)

We will then click “Delete from List”. This activates the student and professor deletion servlets, which will handle the deletion, and then re-direction of the user back to the main menu. We can see that this was successful by the messages displayed upon returning to the main menu.

**Welcome to the Database of Engineering Students and Professors!**

Use this app to manage the students and professors on the engineering university's database.

Student successfully deleted from the database!

[Student List](#)

[Professor List](#)

[Department List](#)

[Add Student](#)

[Delete Student](#)

[Edit Student](#)

[Add Professor](#)

[Delete Professor](#)

[Edit Professor](#)

[Login](#)

# Welcome to the Database of Engineering Students and Professors!

Use this app to manage the students and professors on the engineering university's database.

Professor successfully deleted from the database!

[Student List](#)

[Professor List](#)

[Department List](#)

[Add Student](#)

[Delete Student](#)

[Edit Student](#)

[Add Professor](#)

[Delete Professor](#)

[Edit Professor](#)

[Login](#)

Then, if we check the student and professor lists again, we can see that the student and professor have indeed been deleted.

## List of Students

ID	First Name	Last Name	Major	GPA	Graduate?
6	Barrett	Gamber	CSE	3.38	Yes
21	Shaun	Doe	ME	3.0	No

[Return to Main Menu](#)

## List of Professors

ID	First Name	Last Name	Department	Class Taught	Active Semesters
23	Ibrahim	Imam	CSE	CSE-640	Fall

[Return to Main Menu](#)

I will now demonstrate the final database manipulation function that I incorporated into my app. This is, of course, the ability to edit entries in the database. Upon clicking either edit student or edit professor, the user is greeted with the corresponding JSP shown below. I have gone ahead and filled each out with data.

ID of Student to Edit:

Put newly updated student info for this ID below.

First Name:

Last Name:

Major:

GPA:

Graduate? (Yes/No):

[Return to Main Menu](#)

ID of Professor to Edit:	<input type="text" value="23"/>
Put newly updated professor info for this ID below.	
First Name:	<input type="text" value="Ibrahim"/>
Last Name:	<input type="text" value="Imam"/>
Department:	<input type="text" value="CSE"/>
Teaches:	<input type="text" value="CSE-640"/>
Active Semesters:	<input type="text" value="Spring, Summer"/>
	<input type="button" value="Update Professor Info"/>
<a href="#">Return to Main Menu</a>	

The editing operations work by first requesting the id of the student or professor that the user would like to edit. The user then has the opportunity to re-enter and modify the data for that student or professor. If one noticed, Imam had Fall as his active semester in the professor list. I am going to edit his file so that he is instead listed as teaching his classes in the Spring and Summer. After entering in my changes I click the update button to update Imam's info. It forwards this info to a servlet, which then records this change in the database. For the student version I edited the GPA of Barrett from a 3.38 to a 4.00. The changes we made can be seen below, as well as the messages indicating that we successfully updated this info. This concludes the display of my app in action.

# Welcome to the Database of Engineering Students and Professors!

Use this app to manage the students and professors on the engineering university's database.

Student info successfully updated!

[Student List](#)

[Professor List](#)

[Department List](#)

[Add Student](#)

[Delete Student](#)

[Edit Student](#)

[Add Professor](#)

[Delete Professor](#)

[Edit Professor](#)

[Login](#)

## List of Students

ID	First Name	Last Name	Major	GPA	Graduate?
6	Barrett	Gamber	CSE	4.0	Yes
21	Shaun	Doe	ME	3.0	No

[Return to Main Menu](#)

# Welcome to the Database of Engineering Students and Professors!

Use this app to manage the students and professors on the engineering university's database.

Professor info successfully updated!

[Student List](#)

[Professor List](#)

[Department List](#)

[Add Student](#)

[Delete Student](#)

[Edit Student](#)

[Add Professor](#)

[Delete Professor](#)

[Edit Professor](#)

[Login](#)

## List of Professors

ID	First Name	Last Name	Department	Class Taught	Active Semesters
23	Ibrahim	Imam	CSE	CSE-640	Spring, Summer

[Return to Main Menu](#)

## Future Improvements

With only finite time to complete this project, I could only incorporate so many operations and functionalities. With more time, I would have had time to also implement my other ideas for this project. For one, I would have liked to have been able to incorporate ways to gain even more insight into the operation of the university. For example, I would have liked to add sublists for each of the different engineering departments/majors. In this way a user could see all of the professors/students that are part of the same department/major at a glance. I would also have liked to implement a sort of class search system. As the professors have in their info the classes they teach and during which semesters, lists could be formulated to display these classes in useful ways. For example, by requesting the major of the student and the semester they are looking to enroll in, a list of classes could be displayed that are relevant to the student. These were just a few of my many ideas. I have become much more confident in my ability to develop web apps over the semester. I am hoping to continue to develop my knowledge of internet application design and development outside of this class.