Team Members: Thomas Evans, Vanessa Dowd
Group 45: Team Online Fake University
Project Title: The Online Fake University (TOFU)

Project Step 6: Portfolio

Website URL: http://flip1.engr.oregonstate.edu:6453/

Executive Summary

We began our database plan as a web-based database for an online university. As the project progressed, our entities changed based on a variety of factors. These factors included assessing whether an entity was necessary, time constraints and how large we wanted to make our project. The tables we kept were determined through normalization and discussions on how to create a simple but effective database. We made multiple changes to the database based on feedback in the reviews and based on our own opinion as the project went on, which we will highlight in further detail below.

- We began with Students, Enrollments, Classes, Professors and Departments. Through the different iterative steps of the project these later became Students, Professors, Enrollments, EnrollmentDetails, Classes, Departments, Majors, Terms and TermClassDetails. The Terms entity was added during the normalization process to eliminate a partial dependency. As Terms and Classes have an M:M relationship, an additional intersection table was created. A 'User' entity table was added and later removed.
- We made term class details attributes NOT NULL. We were finding that when the values were able to be NULL, an empty value for both attributes could potentially be added, which would not be of any value. So the attribute was updated in the outline, the DDL and the code to be NOT NULL.
- Based on feedback, we added MySQL CASCADE functionality to the professor and department foreign keys in the Classes entity of the DDL file.
    - The ON DELETE SET NULL accounts for setting the values to Null if the attribute was deleted in another entity so the class is not deleted.
    - We added a nullable Between Classes and its foreign keys, Departments and Professors using an UPDATE CASCADE.
- Based on feedback, we connected the professor and department entity after it was suggested in a review so more information about the professors could be gathered for the university to make decisions (as stated in our project outline).
- We've updated the CSS theme to improve readability and overall look.
- Based on feedback, we changed the Enrollments intersection table naming to EnrollmentDetails to match the other intersection table naming convention (TermClassDetails).
- Based on feedback, we changed how we displayed foreign keys from referencing ID numbers to a more user friendly way as well as implemented FK user friendly drop down menus

Project Outline

The Online Fake University (TOFU) is an online university that serves 10,000 students across the world. TOFU has a 5:1 students to professor ratio, 2,000 professors in 10 departments teaching 4000 classes offering 100 different majors. Our web-based database management system tracks the number of students enrolled in each class by storing this information in an enrollment table. In addition, student's majors are tracked within the student table and the professor table tracks information about the department they are associated with. This allows TOFU to have the most up-to-date information on classes, students, professors, departments and enrollments. With this information, TOFU can make informed decisions surrounding classes, which majors to offer and which professors work for which departments. This gives TOFU the required business intelligence to expand on popular classes, remove classes that are least popular, which majors have the most students and which departments need to hire professors.

Database Outline

1. **Students:** Records the details of a student that attends "Online Fake University". The student ID will be used as the primary key. It will record the information related to each student that is relevant to enrolling at the university, such as name, and major. It has relationships with the majors, Users, classes, and Enrollments entities.
    - **studentID:** bigint, auto_increment, NOT NULL, **PK**
    - **firstName:** varchar(255), NOT NULL
    - **lastName:** varchar(255), NOT NULL
    - **userName:** varchar(255), NOT NULL
    - **email:** varchar(320), Unique, NOT NULL
    - **majorID**: int, **FK**
    - **Relationship:**
        - 1:M relationship between Students and Enrollments (1 student can be enrolled in many classes) - intersection table
        - M:1 relationship between Students and Majors(many students can be in 1 major)
        - M:M relationship between Students and Classes (many students can be in many classes)

2. **Professors:** Records the details of the professors that teach at the university and what department they teach in. Its primary key is the professor ID. Has relationships with the Enrollments, Departments and Users entities.
    - **professorID**: bigint, auto_increment, NOT NULL, **PK**
    - **firstName:** varchar(255), NOT NULL
    - **lastName:** varchar(255), NOT NULL

- **userName:** varchar(255), NOT NULL
- **email:** varchar(320), Unique, NOT NULL
- **deptID:** int, **FK**
- **Relationship:**
  - 1:M relationship between Professors and Classes(one professor can teach many Classes)
  - M:1 relationship between Professors and Departments (many professors in 1 department)

3. **EnrollmentDetails:** Records students enrolled in a class and the class ID. The enrollment ID will be used as a primary key. It has a relationship with the Students and Classes entities. **Intersection Table between Students and Classes**
   - **enrollmentID:** bigint, auto_increment, NOT NULL, **PK**
   - **enrollmentDate:** date, NOT NULL
   - **studentID:** bigint, NOT NULL, **FK**
   - **classID:** bigint, NOT NULL, **FK**
   - **Relationship:**
     - M:1 relationship with Enrollments and Students (many enrollments to 1 student),
     - M:1 relationship with Enrollments and Classes (many enrollments to 1 class)

4. **Classes:** Records the classes offered at the university with the class ID being the primary key. Also records the name of the class and information relating to the class, such as the name and credits it offers. Its primary key is the class ID. It has relationships with the Enrollments and Departments entities.
   - **classID**: bigint, auto_increment, NOT NULL, **PK**
   - **className:** varchar(145), NOT NULL
   - **classCredits**: int(10), NOT NULL
   - **professorID:** bigint, Default NULL, **FK**
   - **deptID:** int, Default NULL, **FK**
   - **Relationship:**
     - M:M relationship with Students (many classes to many students).
     - M:1 relationship between Classes and Departments (many classes in one department)
     - 1:M relationship between classes and Enrollments (1 class to many enrollments)
     - M:M relationship between Terms and Classes (one term can have many classes

5. **TermClassDetails**: Records the terms that classes are offered in. The primary key is termsClassesID. **Intersection Table between Terms and Classes**
   - **termclassdetailsID:** int, auto_increment, NOT NULL, **PK**

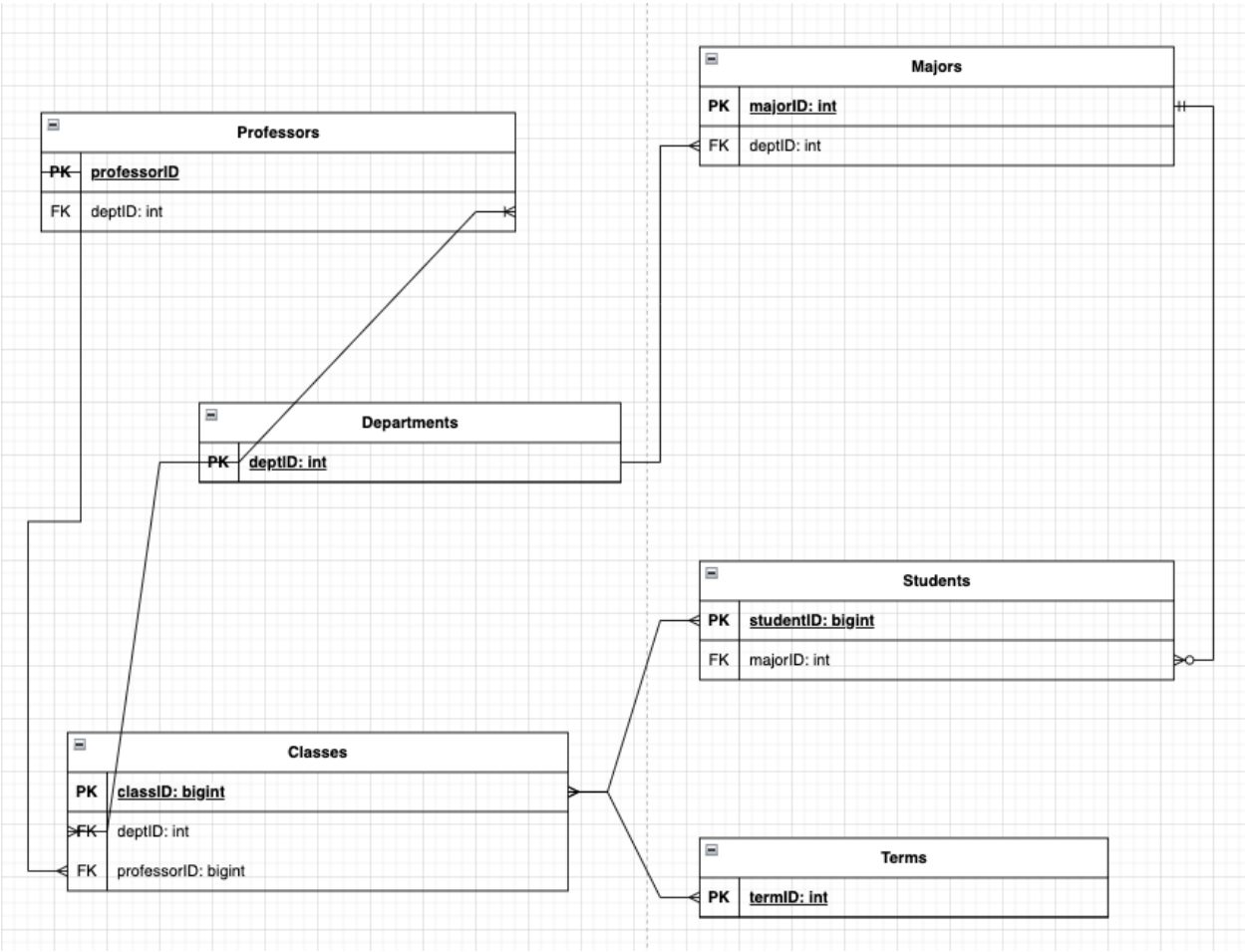
- **termID:** int, **FK,** NOT NULL
- **classID:** bigint, **FK,** NOT NULL
- **Relationships:**
  - M:1 relationship with Classes and termClassDetails (many classes to 1 term)
  - M:1 relationship with Terms and termClassDetails (many terms to 1 class)

6. **Terms:** Records the term and year that a class is offered. Its primary key is termID, it has a relationship with classes.
   - **termID:** int, auto_increment, NOT NULL, PK
   - **term:** varchar(50), NOT NULL
   - **yr**: year(3000), NOT NULL
   - **Relationships:**
     - M:M relationship between Terms and Classes (one term can have many classes

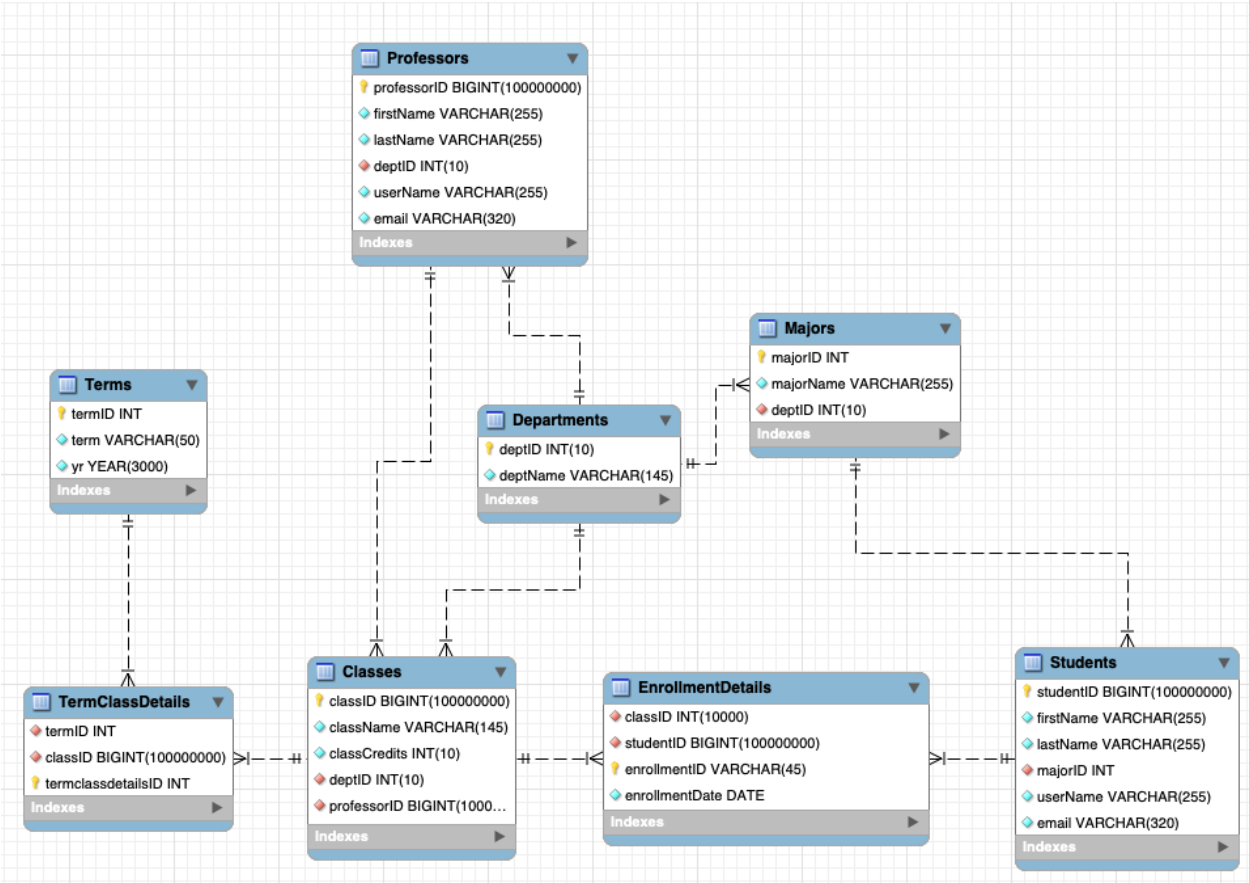
7. **Departments:** Records the different departments within the university and the names of the departments. Its primary key is the department ID. It has relationships with the Classes, Students and Professors entities.
   - **deptID:** int(10), auto_increment, NOT NULL, **PK**
   - **deptName:** varchar(145), NOT NULL
   - **Relationship:**
     - 1:M relationship between Departments and Classes (one department with many classes)
     - 1:M relationship between Departments and Majors (one department with many students)
     - 1:M relationship between Departments and Professors (one department with many professors)

8. **Majors:** Records the current Majors offered at TOFU. Its primary key is the majorID, it has relationships with students and departments.
   - **majorID:** int, auto_increment, **PK**
   - **majorName:** varchar(255), NOT NULL
   - **deptID:** int, **FK**
   - **Relationships:**
     - M:1 relationship between departments and majors(one department can have many majors
     - 1:M relationship between major and students (one major can have many students)

Entity-Relationship Diagram



Schema Diagram



Sample Data

Students

	studentID	firstName	lastName	userName	email	majorID
Edit Copy Delete	1	Lucy	Sulton	sultonl	lucy.sutton@TOFU.edu	1
Edit Copy Delete	2	Maelya	Meyer	meyerma	maelya.meyer@TOFU.edu	3
Edit Copy Delete	3	Jamie	Cooper	cooperj	jamie.cooper@TOFU.edu	5
Edit Copy Delete	4	Christine	Smits	smitsch	christine.smits@TOFU.edu	5
Edit Copy Delete	5	Katerine	Thomas	thomask	katerine.thomas@TOFU.edu	1
Edit Copy Delete	6	Lucas	Rubio	rubioluc	lucas.rubio@TOFU.edu	1

EnrollmentDetails

	enrollmentID	enrollmentDate	studentID	classID
Edit Copy Delete	1	2024-05-06	6	1
Edit Copy Delete	2	2024-05-07	5	1
Edit Copy Delete	3	2024-05-08	4	5
Edit Copy Delete	4	2024-05-09	3	5
Edit Copy Delete	5	2024-05-10	2	4
Edit Copy Delete	6	2024-05-11	1	2
Edit Copy Delete	7	2024-05-12	1	1

Professors

	professorID	firstName	lastName	userName	email	deptID
Edit Copy Delete	1	Andrew	Dennis	dennisa	andrew.denis@TOFU.edu	2
Edit Copy Delete	2	Hannah	Wright	wrighth	hannah.wright@TOFU.edu	3
Edit Copy Delete	3	Thomas	Evans	evanstho	thomas.evans@TOFU.edu	5
Edit Copy Delete	4	Beth	Davis	davisbe	beth.davis@TOFU.edu	6
Edit Copy Delete	5	Jordan	Jenkins	jenkinsj	jordan.jenkins@TOFU.edu	1
Edit Copy Delete	6	Hugo	Green	greenhu	hugo.green@TOFU.edu	4

Classes

	classID	className	classCredits	professorID	deptID
Edit Copy Delete	1	Organic Chemistry I	4	5	1
Edit Copy Delete	2	Quantum Mechanics	4	1	2
Edit Copy Delete	3	Poets: Poetry and Everything	3	2	3
Edit Copy Delete	4	Contemporary Art History	3	6	4
Edit Copy Delete	5	Intro to Computer Science	4	3	5
Edit Copy Delete	6	History of Jazz	2	4	6

Departments

	deptID	deptName
Edit Copy Delete	1	Chemistry
Edit Copy Delete	2	Physics
Edit Copy Delete	3	English
Edit Copy Delete	4	Art History
Edit Copy Delete	5	Electrical Engineering and Computer Science
Edit Copy Delete	6	Music

Majors

	majorID	majorName	deptID
Edit Copy Delete	1	Chemical Engineering	1
Edit Copy Delete	2	Chemistry	1
Edit Copy Delete	3	English Literature	3
Edit Copy Delete	4	Creative Writing	3
Edit Copy Delete	5	Computer Science	5
Edit Copy Delete	6	Electrical Engineering	5

Terms

	termID	term	yr
Edit Copy Delete	1	Winter	2023
Edit Copy Delete	2	Spring	2023
Edit Copy Delete	3	Summer	2023
Edit Copy Delete	4	Fall	2023
Edit Copy Delete	5	Winter	2024
Edit Copy Delete	6	Spring	2024

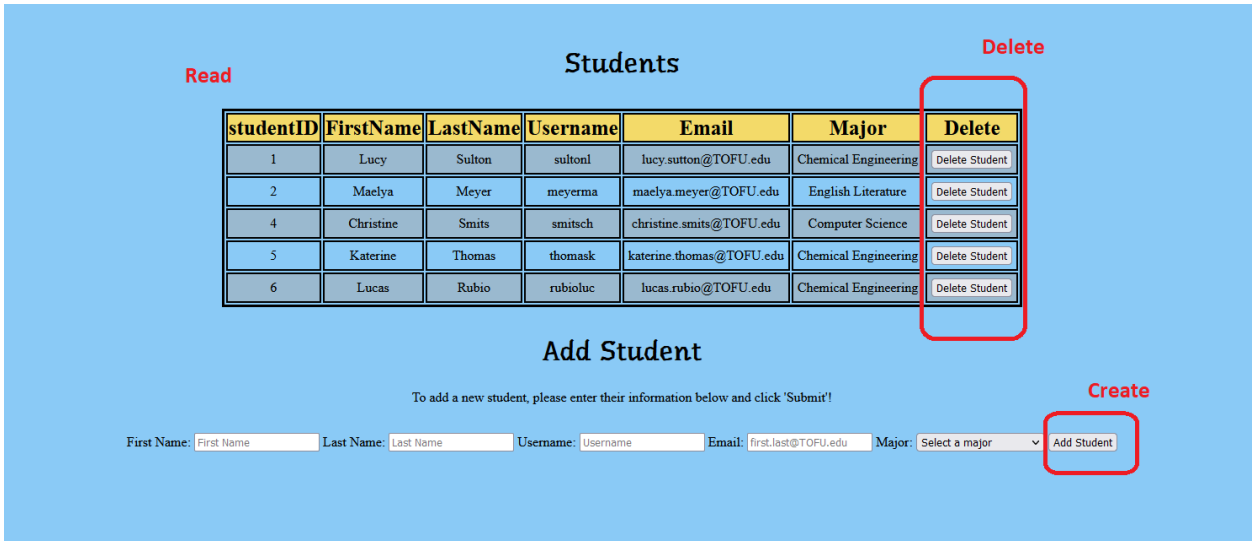
TermClassDetails

	termclassdetailsID	termID	classID
Edit Copy Delete	1	1	1
Edit Copy Delete	2	1	2
Edit Copy Delete	3	2	3
Edit Copy Delete	4	2	4
Edit Copy Delete	5	3	5
Edit Copy Delete	6	3	6

Screen Captures

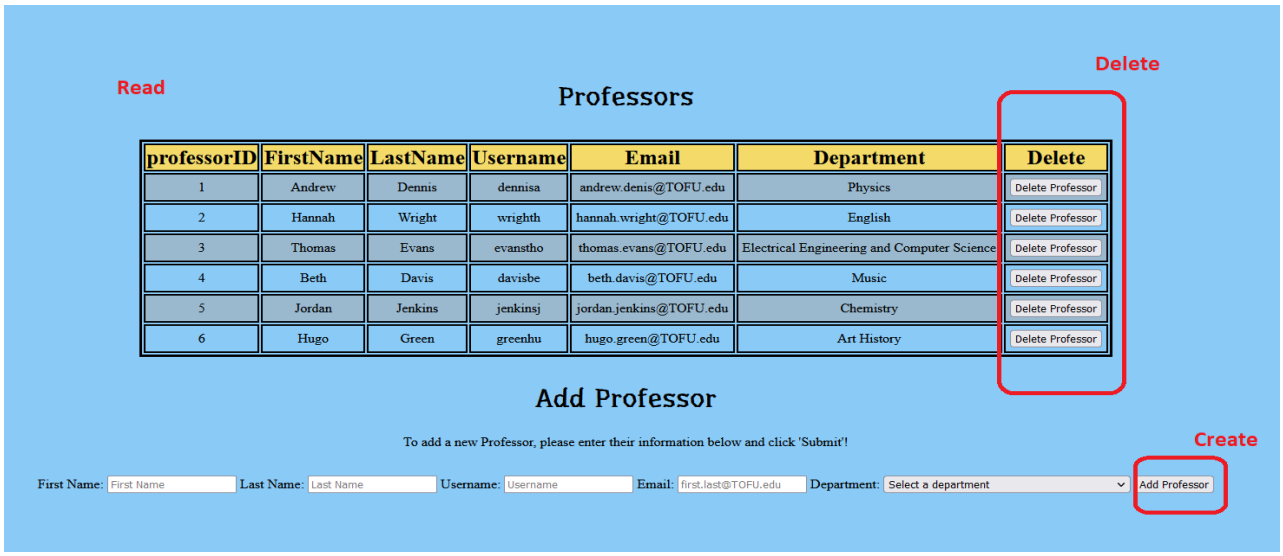
Students(Create, Read, Delete)

- Create
    - Fk user friendly
- Read
- Delete



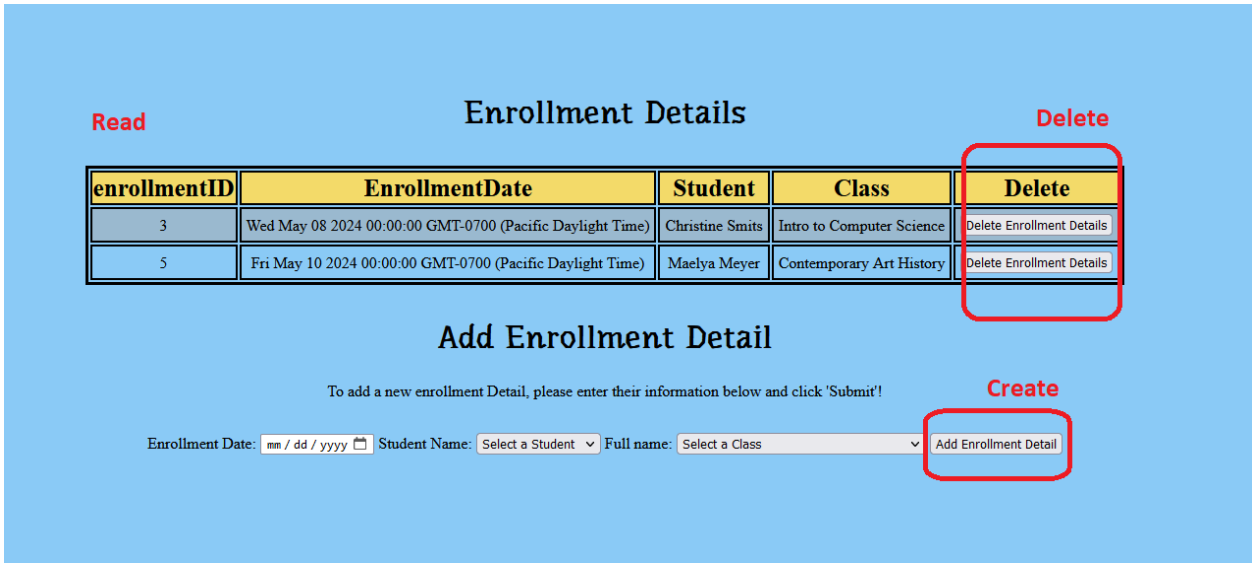
Professors(Create, Read, Delete)

- Create
    - FK USER friendly
- Read
- Delete



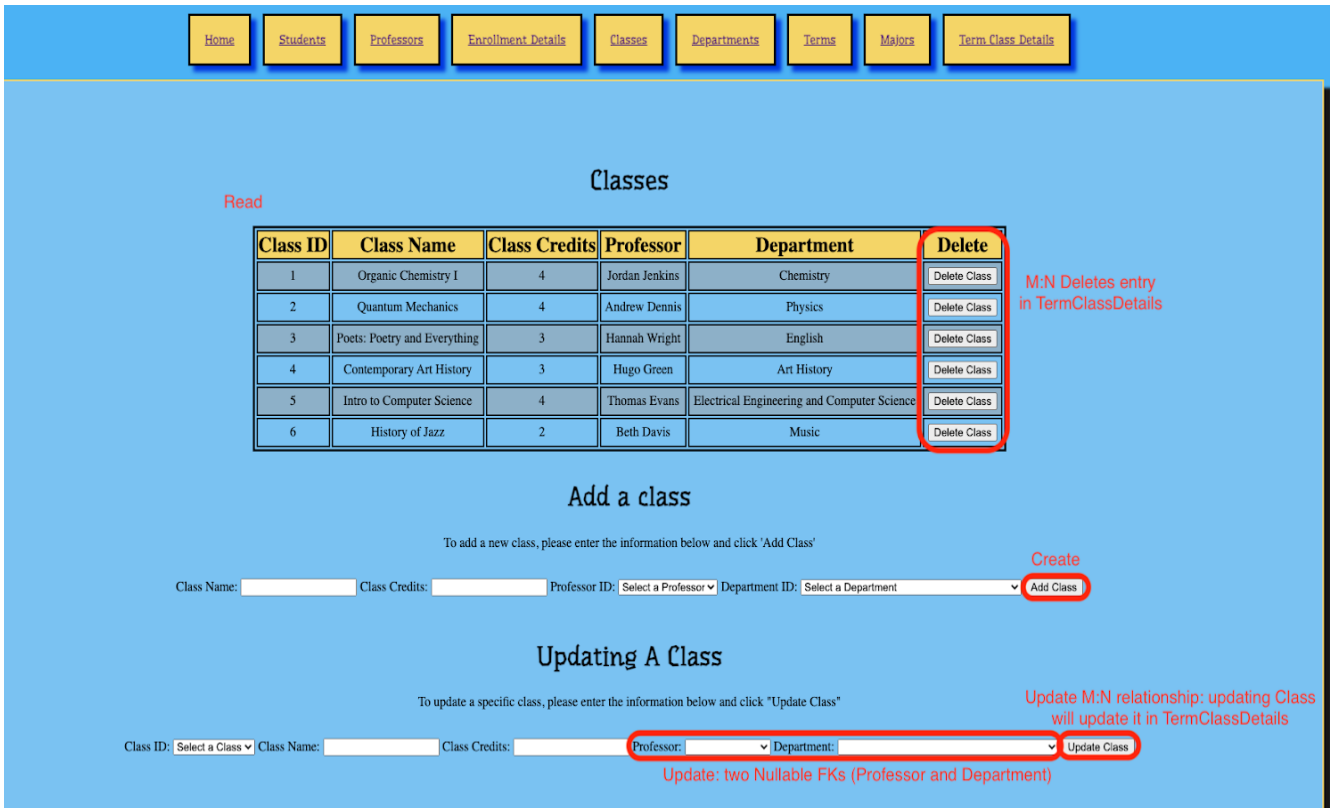
EnrollmentDetails(Create, Read, Delete)

- Create
    - FK USER Friendly
- Read
- Delete

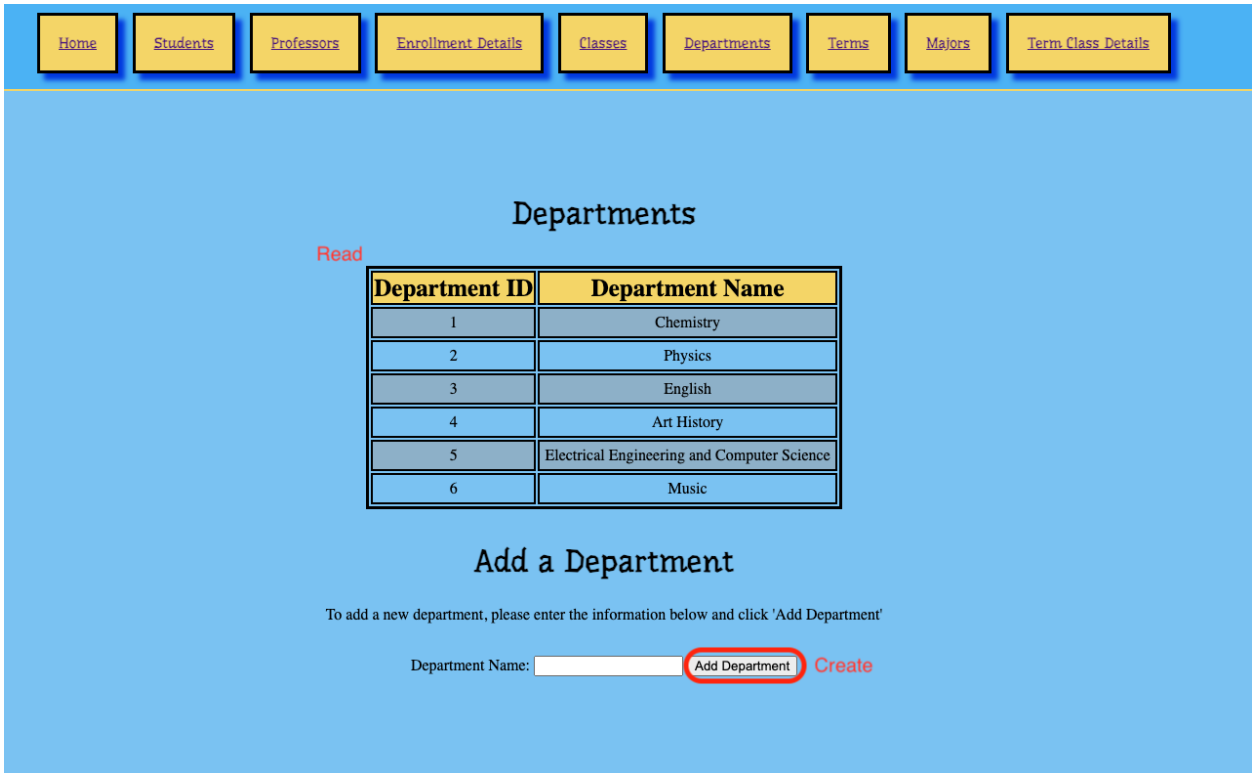


Classes(CREATE/READ/UPDATE/DELETE):

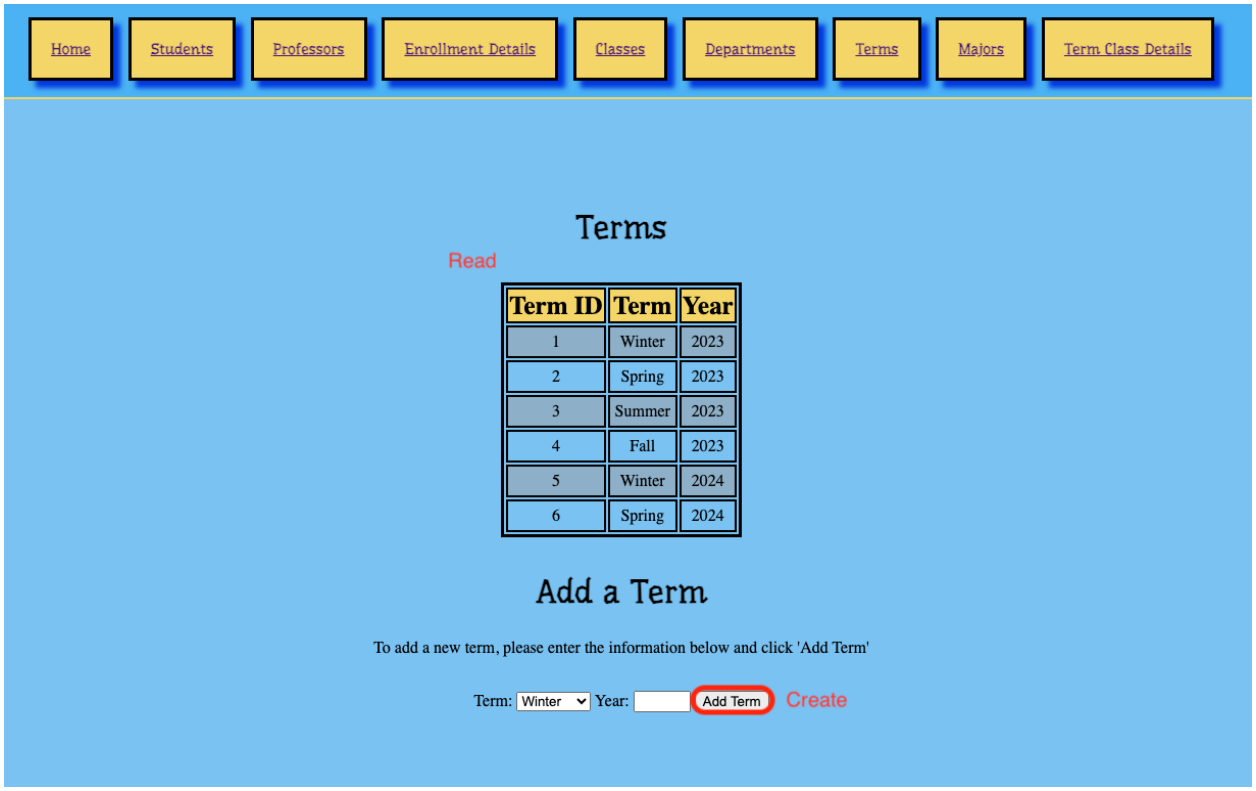
- Create
- Read
- Update:
    - **Updates M:N relationship: updates Class in TermClassDetails**
    - **Updates two nullable FKs (professor and department)**
- **Delete: M:N Delete: Deletes entry in TermClassDetails**



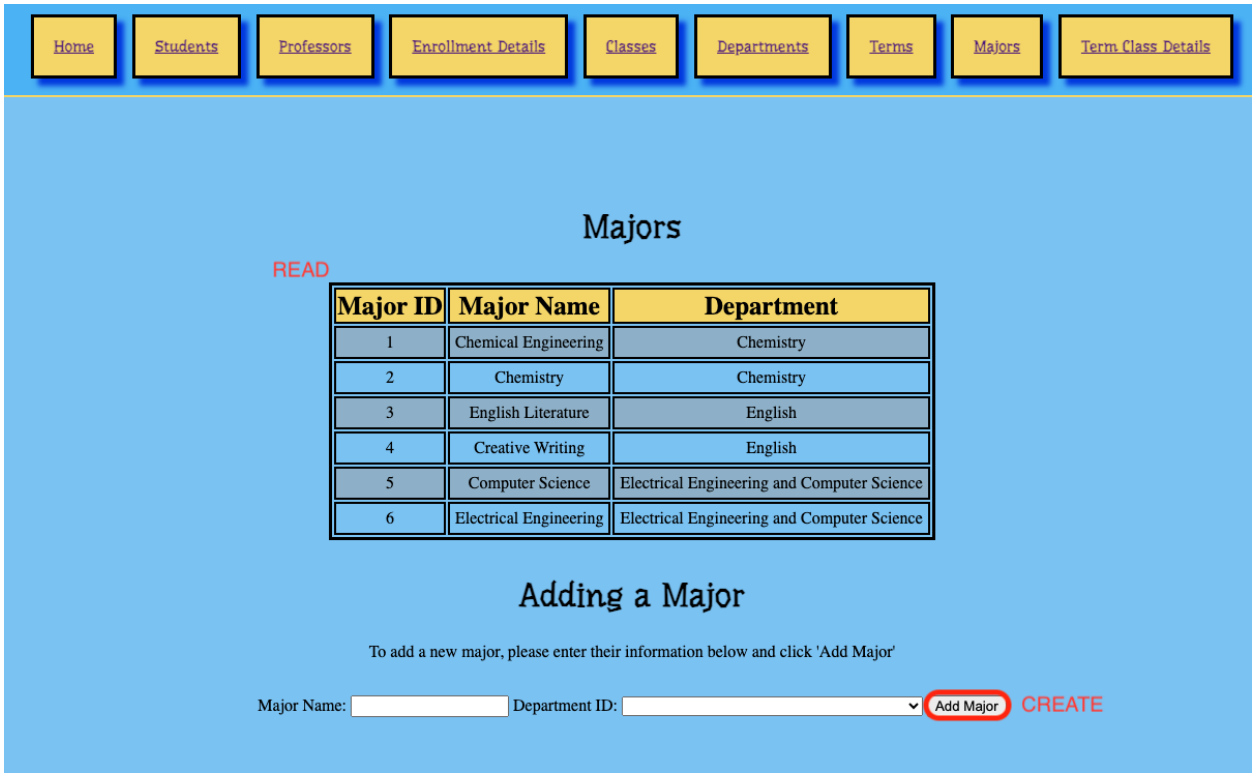
Departments (CREATE/READ)



Terms(CREATE/READ)



Majors (CREATE/READ)



TermClassDetails(CREATE/READ)