StudentFirst:

A Student Engagement Site

Group 45: Hunter Cobb

Link to website Hosted on EC2:

18.116.5.0

Hunter Cobb

4/12/2021

Final Project Report

Link to Site:18.116.5.0

Introduction:

In short, my project idea is to create an online learning application that connects Professors to their students through assignment distribution, and the ability for students to take assignments and submit their answers for the professor to review.

There are two main types of users being Professors and students.

Each professor will be able to create classes to post materials such as those listed above mainly being assignments and homeworks for their classes. They will also be in control of who is in the class with the ability to invite and kick people from their class at their own discretion.

Each student will be able to view all of their different classes and view the material within them posted by the professor of each class. Students join classes when invited by professors and can choose to either accept or decline the invite. Students will be able to access all materials posted by the professor as well as take quizzes and exams posted on the site and the results will be kept for the professor to look over.

Classes are another important element, controlled by the professor that created it, that act as the holding element for all information related to the class as well as assignments, exams, and other materials.

For testing purposes i used:

Example Student:

Username: hunter@ku.edu

Password: password

Example Professor:

Username: henryjacob@ku.edu

Password: password

Table Structures:

Professors

ProfessorID	ProfessorPassword	ProfessorName	Professor Email	Professor Bio

Students

StudentID	StudentPassword	StudentName	StudentEmail	Student Bio

Classes

ClassID	ClassName	ProfessorID	ClassDescription

Assignments

AssignmentID	AssignmentName	ClassID	Assignment Desc	AssignmentQuestions

Memberships

Membership ID	ClassID	StudentID

Submissions

SubmissionID	AssignmentID	StudentID	StudentAnswers

Sample queries using these tables:

Access all classes of a student:

```
SELECT DISTINCT CLASSNAME, CLASSES.DESCRIPTION
FROM CLASSES,MEMBERSHIPS
WHERE CLASSES.CLASSID = MEMBERSHIPS.CLASSID
 AND MEMBERSHIPS.STUDENTID = 1
```

Access Roster of a class:

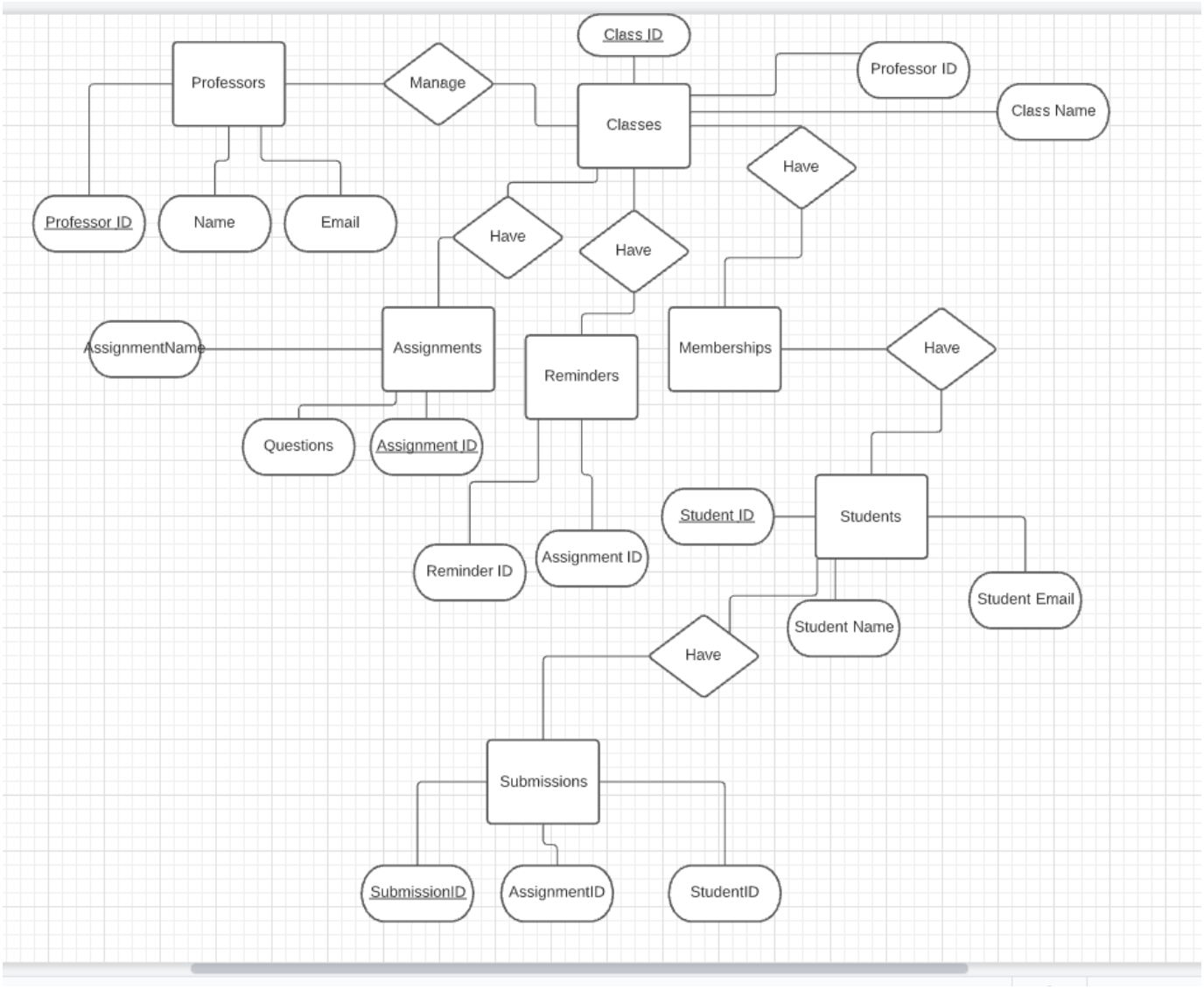
```
SELECT STUDENTNAME FROM
STUDENTS,MEMBERSHIPS
WHERE STUDENTS.STUDENTID = MEMBERSHIPS.STUDENTID
AND MEMBERSHIPS.CLASSID = 2
```

Select all assignments from classes where the student is a member

```
SELECT ASSIGNMENTNAME, ASSIGNMENTID FROM
ASSIGNMENTS,MEMBERSHIPS,CLASSES
WHERE MEMBERSHIPS.STUDENTID = '$stuID'
AND MEMBERSHIPS.CLASSID = CLASSES.CLASSID
AND ASSIGNMENTS.CLASSID = CLASSES.CLASSID
```

ER Diagram:

This diagram should also include StudentPassword under students and ProfessorPassword under professors. I used lucidchart.com for this chart and I ran out of free shapes two short



System Architecture:

For the architecture of my system, I used four main components being:

- MYSQL
- PHP
- HTML/CSS/BOOTSTRAP
- AWS EC2/XAMPP

MYSQL(Password and login for mysql are in each PHP code segment included):

MYSQL was used for the database backend of this project and represents the actual storage of data for this site including all the tables mentioned above. Some notable tables needed to access information pertaining to the site are the student and professor tables as this is where signin information is stored. The example students/teachers I used for logging into the site were:

Example Student:

Username: hunter@ku.edu

Password: password

Example Professor:

Username: henryjacob@ku.edu

Password: password

These can be used on my site to verify that the data is correctly being transmitted although you can also create a new student or professor using their respective register pages under their respective login pages.

PHP:

PHP was used as my query language and was the only way that my code can directly interact with the MYSQL database in this project. I grabbed this data using PHP and echoed it into javascript variables that were then used to display the data to the user. I used PHP as my backend query language because its the most commonly used and I had prior experience using it from my time at KU.

HTML/BOOTSTRAP CSS/JAVASCRIPT:

I used HTML as my markup language as it's the most common and I had prior experience with it at my time at KU and prior to college. I also used Bootstrap css library for the first time in this project to give my projects elements a more polished look and to get some experience using the library. I also used some Javascript to communicate between the PHP backend and the HTML frontend and to display data whose size wasn't static such as number of people in a class or number of classes a student has using the javascript map function. This project actually gave me a lot of insight in how to use javascript to communicate non-static data on a webpage without using a javascript framework such as react as i've done in the past.

EC2/XAMPP:

I used EC2 as my hosting service to make this webpage publicly available and I'm currently running XAMPP on the server to host my webpage on the EC2 server. This was also quite new to me and I learned quite a bit both about local hosting and hosting

on the AWS servers. I used an EC2 instance loaded with a xampp install and used EC2s elastic ip service to give the site a home that can be searched directly through the search bar after figuring out how to set up an EC2 server to accept HTTP traffic.

Site Structure:

The structure of my site resembles that of most online learning tools such as Blackboard and Canvas. The landing page of my site is a login page for students. From here we can:

- Login as an existing student
    - PHP does in fact check to make sure that such a student exists.
- Click on the register button and be taken to the student register page
- Click on the teacher sign in button and sign in as a professor
    - From here, there is another register button where a professor can register for the site.
    - The professor login also verifies the user using PHP to make sure the professor indeed exists and has the correct username,password tuple.

If the user logs in as a **student** they will be taken to their **dashboard** where they can:

- See all their assignments for all classes where they are a student.
    - Assignments, when clicked, will redirect the student to the appropriate assignment where they can then
- See all their classes where they are a student.
    - Classes, when clicked, will redirect the student to the appropriate class home page.
    - For students, on the class page, all that is visible is the name of the class, class description, and a dropdown with all assignments posted for this class where they can click and be redirected to the assignment page.

If the user clicks on an **assignment**,either on the dashboard or their classes page, they will be redirected to the respective assignments page:

- Students fill out answers to the assignment and submit it for review for the professor
    - After submission, they are taken to a page saying either error or submission received with a button to redirect them to their dashboard.

Now, If we choose to sign on as a professor, we are taken to the **PROFESSOR DASHBOARD,** From here we can:

- Create a new class
    - Uses HTML form to deliver information to the php backend to then be inserted into the MYSQL table.
- Click on, and redirect to a class being taught by this professor
    - Professor class view differs from the students.

Finally, out of the interesting pages, we have the **PROFESSOR CLASS VIEW** where we can:

- Create new assignments
    - Uses HTML forms and PHP to insert the assignment into the assignments table so it can be seen both by students and professors.
- Look at class roster
    - Lists all students in the class using a MYSQL query that is communicated to the HTML using PHP.

- Look at all student submissions for an assignment
    - Separates students' submissions by name and displays all answers that they gave for all the questions on the assignment using PHP to query the MYSQL submissions table.