CS364 Project Proposal

Group:

Brock Bena

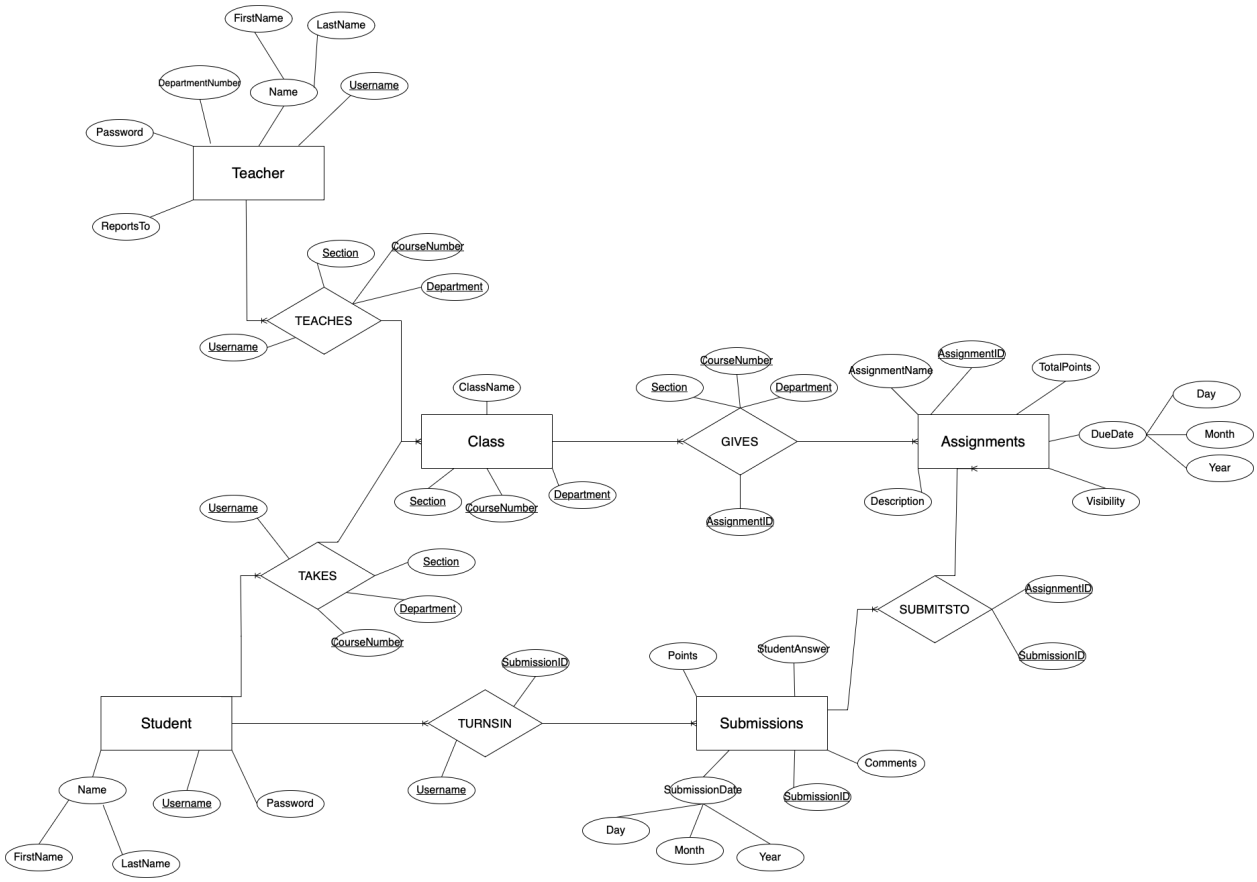
Travis Wiesner

Synopsis

We are planning on making a school grades tracker for students and teachers, which will be a recreation of Canvas. We will be making our program run from a class focused database where all data is routed through classes. All user entities will be routed through relationships specific to classes. Teachers teach classes, students take classes, and classes have assignments. This allows for easy parsing and editing of data. Assignments will also have an entity for Submissions where students will submit work through a Submissions entity that is related to Assignments entity. Teachers can view an Assignment's related Submissions for grading. Teachers can also add a comment to specific Submissions for the Student related to the Submission to read. The student can also see their grades for all assignments related to the class. Both Teachers and Students will be able to see a representation of their assignments on a dashboard. The dashboard will be sorted by the due date of the assignments. Teachers will have a link to the

assignment's submission box where all student's submissions will have a link to grade each submission. Whereas, Students will have a link to their assignment's submission page where they can review submissions or make a new submission for the assignment. The dashboard will also offer a class filter so the student can look at individual class assignment lists.

ER Diagram



(See a higher resolution of the diagram on the attached .png)

ER Diagram Attribute Description

Teacher Table:

Name: Comprised of FirstName and LastName, the teacher's name

Username: The teacher's username and the key.

Password: The teacher's password

DepartmentNumber: What department the teacher belongs to

ReportsTo: The teacher's supervisor.

Class Table:

ClassName: The name of the course

CourseNumber: The number of the course and the key

Department: The department that the course belongs to and the key

Section: The section number of the course and the key

TEACHES Table (Relationship between Teacher and Class):

Username: The teacher's username and the foreign key from the Teacher Table

CourseNumber: The number of the course and the foreign key from the Class Table

Department: The department that the course belongs to and the foreign key from the Class Table

Section: The section number of the course and the foreign key from the Class Table

Student Table:

Username: The student's username and the key

Name: The student's name, comprised of FirstName and LastName

Password: The student's password

TAKES Table (Relationship between Student and Class):

Username: The student's username and the foreign key from the Student table

CourseNumber: The number of the course and the foreign key from the Class Table

Department: The department that the course belongs to and the foreign key from the Class Table

Section: The section number of the course and the foreign key from the Class Table

Assignments Table:

AssignmentID: The ID of the specific assignment and the key.

AssignmentName: The assignment's name

Description: The description of the assignment

TotalPoints: The total amount of points that can be earned for this assignment

DueDate: The date that the assignment is due, is comprised of day, month, and year

Visibility: If the assignment is visible to the students or not

GIVES Table (Relationship between Class and Assignments):

AssignmentID: The ID of the specific assignment and the foreign key from the Assignments Table.

CourseNumber: The number of the course and the foreign key from the Class Table

Department: The department that the course belongs to and the foreign key from the Class Table

Section: The section number of the course and the foreign key from the Class Table

Submissions Table:

SubmissionID: The ID of the submission and the key

SubmissionDate: The date the assignment was submitted, comprised of day, month, and year

Comments: Comments left by a teacher on the submission

Points: How many points the student received from the submission

StudentAnswer: The student's submission, their answer

SUBMITSTO Table (Relationship between Assignments and Submissions):

SubmissionID: The ID of the submission and the foreign key from the Submissions Table

AssignmentID: The ID of the specific assignment and the foreign key from the Assignments Table.

TURNSIN Table (Relationship between Student and Submissions):

SubmissionID: The ID of the submission and the foreign key from the Submissions Table

Username: The student's username and the foreign key from the Student table

Functionality

- Ability to add students, teachers, classes, and assignments
- Teachers can see which students have or have not submitted an assignment, as well as give them a grade on that assignment. This query satisfies advanced query one, as it gets a count of the students who have and have not submitted their assignment. This query is located in assignment.js. It is as follows:

```
SELECT *,
    (SELECT count(*)
        FROM Student JOIN TAKES JOIN Class
            ON Class.Department = TAKES.Department
            AND Class.CourseNumber = TAKES.CourseNumber
            AND Class.Section = TAKES.Section) AS maxCount,
    (SELECT count(*)
        FROM TURNSIN JOIN Submissions JOIN SUBMITSTO JOIN Assignments
            ON Submissions.SubmissionID = TURNSIN.SubmissionID
            AND Submissions.SubmissionID = SUBMITSTO.SubmissionID
            AND SUBMITSTO.AssignmentID = Assignments.AssignmentID
        GROUP BY TURNSIN.Username) AS actualCount
    FROM Assignments NATURAL JOIN GIVES NATURAL JOIN Class
```

```
WHERE Assignments.AssignmentID = '${id}'
```

- Teachers can see the list of all assignments for a class, as well as if all students have submitted it or not. This query satisfies advanced query number 2, and it joins five tables in one query. This query is also located in assignment.js. It is as follows

```
SELECT *,
(SELECT count(*)
        FROM Student JOIN TAKES JOIN Class
                ON Class.Department = TAKES.Department
                AND Class.CourseNumber = TAKES.CourseNumber
                AND Class.Section = TAKES.Section
                AND Student.Username = TAKES.Username
        WHERE Class.Department = '${classId[0]}' AND Class.CourseNumber = ${classId[1]}
        AND Class.Section = ${classId[2]}) AS maxCount,
(SELECT count(*)
        FROM TURNSIN JOIN Submissions JOIN SUBMITSTO JOIN Assignments JOIN GIVES
                ON Submissions.SubmissionID = TURNSIN.SubmissionID
                AND Submissions.SubmissionID = SUBMITSTO.SubmissionID
                AND SUBMITSTO.AssignmentID = Assignments.AssignmentID
                AND Assignments.AssignmentID = GIVES.AssignmentID
        WHERE Class.Department = '${classId[0]}' AND Class.CourseNumber = ${classId[1]}
        AND Class.Section = ${classId[2]}
        GROUP BY TURNSIN.Username) AS actualCount
        FROM Assignments NATURAL JOIN GIVES NATURAL JOIN Class
        WHERE Class.Department = '${classId[0]}' AND Class.CourseNumber = ${classId[1]}
        AND Class.Section = ${classId[2]}
```

- Teacher can see a list of courses they teach
- Teacher can see all students in the courses they teach
- Students can see all assignments they have or not submitted, as well as the grade they received for those assignments

- Students can see all courses they are in, as well as the teacher that teaches that course. This satisfies advanced query 3, and is located in class.js. It is as follows:

```
SELECT DISTINCT ${userDef.type}.Username, Class.CourseNumber, Class.Department,
Class.Section, TeacherOfClass.TeacherUsername
        FROM ${userDef.type} JOIN ${userDef.verb} JOIN Class JOIN
        (SELECT TEACHES.Username as TeacherUsername
                FROM TEACHES JOIN Class
                        ON TEACHES.CourseNumber = Class.CourseNumber
                        AND TEACHES.Department = Class.Department
                        AND TEACHES.Section = Class.Section
        ) AS TeacherOfClass
                ON ${userDef.type}.Username = ${userDef.verb}.Username
                AND ${userDef.verb}.CourseNumber = Class.CourseNumber
                AND ${userDef.verb}.Department = Class.Department
                AND ${userDef.verb}.Section = Class.Section
        WHERE ${userDef.type}.Username = '${username}'
```

Stakeholders

We believe that those who will primarily be using this program will be students and teachers, and their use may differ based on the teacher's style of instruction. Students will be able to see what classes they are in, who teaches those classes, and complete and turn in assignments. Teachers will be able to see all students in the classes they teach, create assignments, and grade the students' assignments.

Technological Requirements

For the whole project, we will be making a web application. The frontend UI will be using NodeJS with the Angular framework to dynamically change the webpage based on the parsed data from the backend. The backend will be run using ExpressJS as a middleware to receive endpoint hits from the frontend, and query the database. For our relational database, we will be using SQL Workbench and the server provided by Dr. Foley. All of our code and documentation will be shared through a GitHub repo between the two of us. We also used discord for communications, and a Discord bot to notify use when the other pushes a new commit to the repo. Brock has experience in web development, and Travis will be running documentation and backend. We will both be communicating on what type of database queries we need to display data.