

Database Final Project Report

By: Agastya Das, Kristin Lin

Description

ClassTrack is a student review site for Northeastern courses. A student can search for reviews for any course and write reviews for any course they are taking.

When writing a review for a course, students can provide:

- Overall rating [required]
- General comment
- One strength of the course/professor
- Another strength of the course/professor
- One weakness of the course/professor
- Another weakness of the course/professor

When searching for reviews, students can search by course subject, course number, or professor last name. On a course by professor review page, students can toggle whether to see content for *All Semesters* or for just one specific semester. Content that is provided is:

- Average rating from reviews
- Graph of rating & frequency
- Graph of top strengths & weaknesses
- Table of reviews
- Ability to like a review & see number of likes

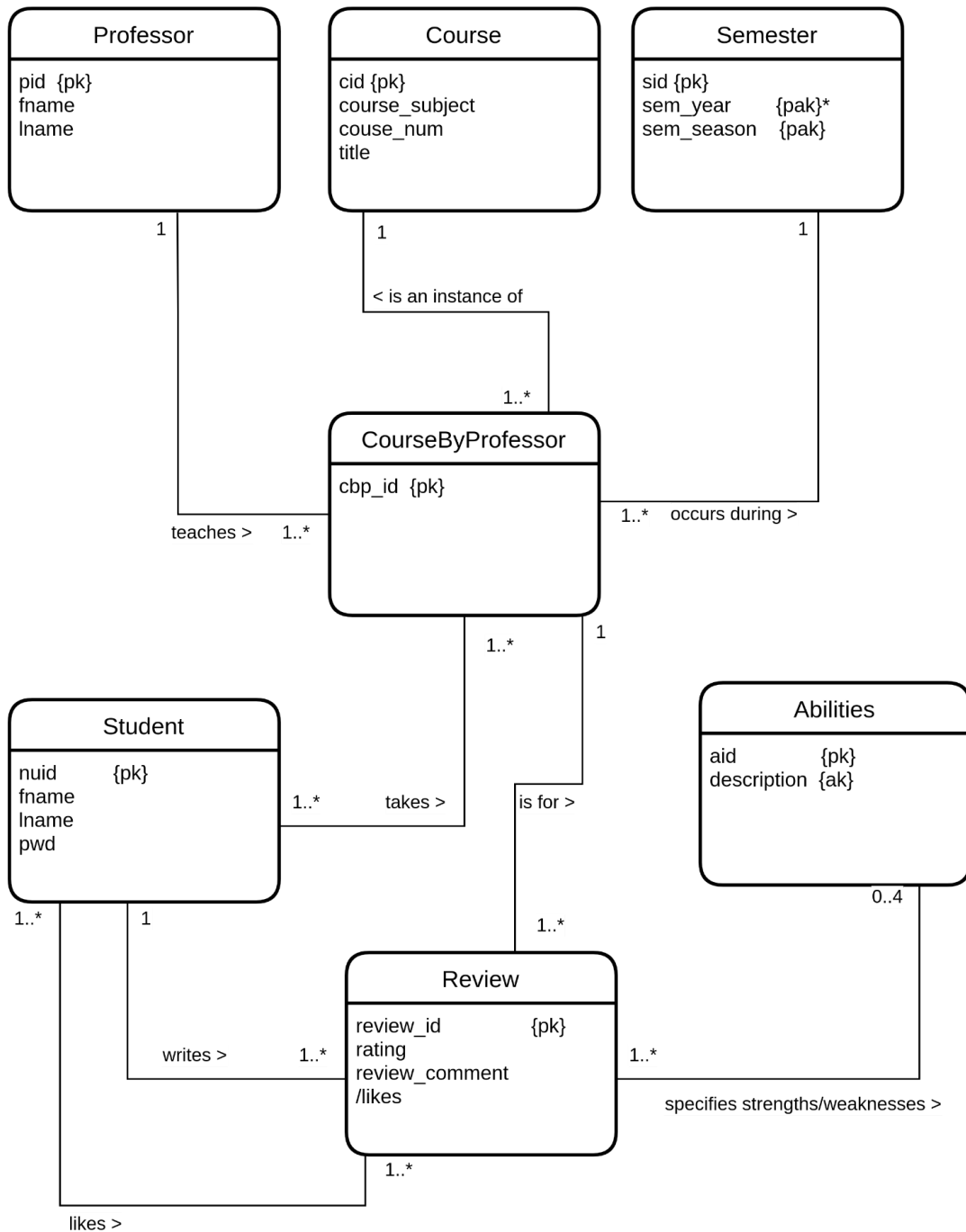
Technical Specifications

The backend for this application was built using **Python** and the **Flask** framework. To connect to our MySQL database, we used the **PyMySQL** package.

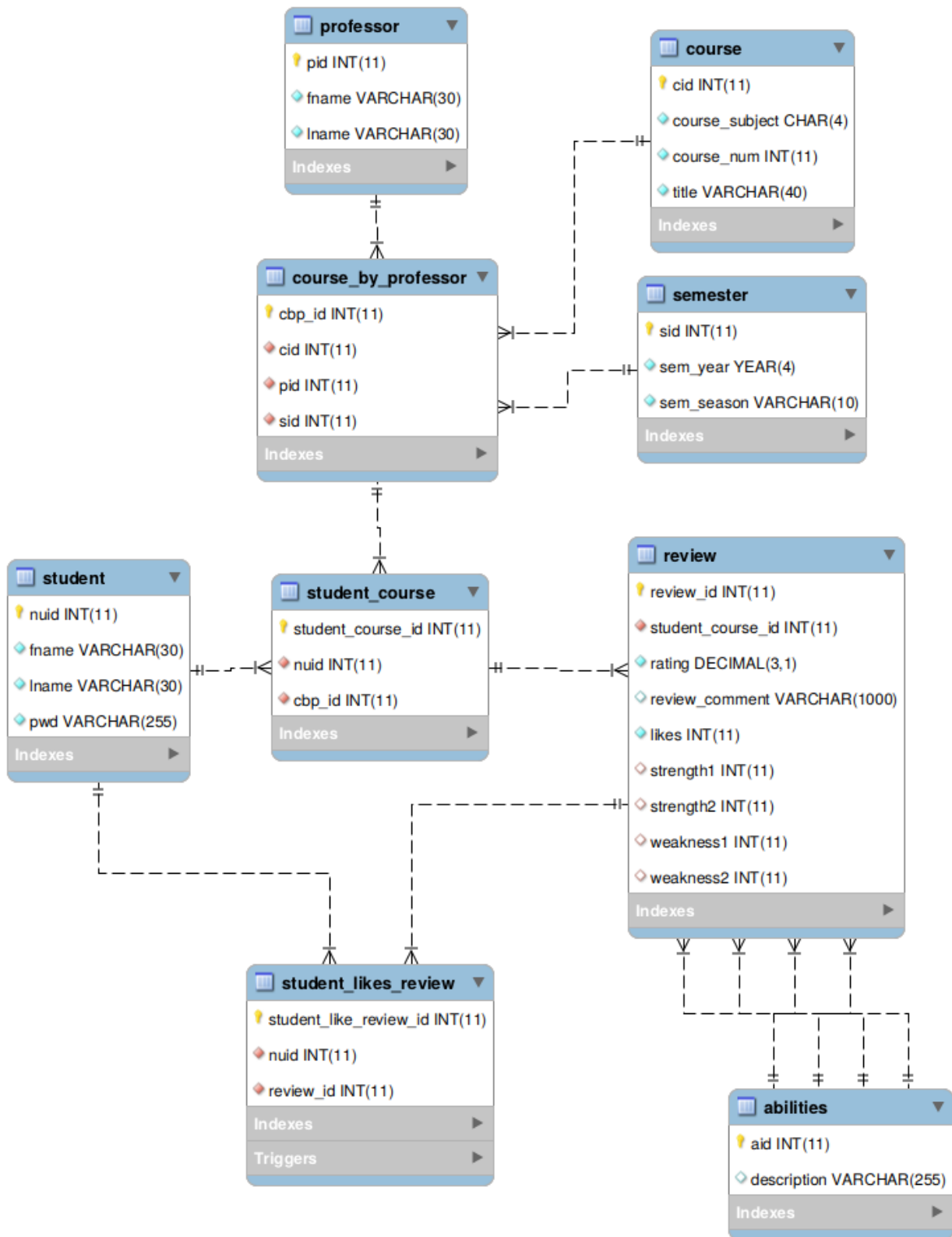
The frontend for our application was created using plain **HTML** and **CSS**. We were able to utilize **Bootstrap/JQuery** frontend library for their component classes, **FontAwesome** for their icons, **Chart.js** for creating nice graphs, and **GoogleFonts** for our site name.

UML Conceptual Design

* partial alternate key



Logical Design for Database Schema



User Flow

Log In/Out & Account Management:

1. Login by navigating to <http://127.0.0.1:5000/>
2. Login using NUID: 8, PASSWORD: password8. *Note that NUIDs go up to 20, and passwords are just password[NUID].*
3. Click the person icon on the far right of the navigation bar to navigate to <http://127.0.0.1:5000/account>
4. Update password using form.
5. Log out of the account using the “Log out” button.

Writing a Review:

1. Navigate to <http://127.0.0.1:5000/courses> using the button on homepage or third button on navigation bar (pencil icon).
2. Click “Write Review” or “Edit Review” button next to a course you are taking.
3. Fill out or edit existing responses in the form: Rating, Comment, Strength1, Strength2, Weakness1, Weakness2
4. Click “Submit” or “Save”
5. To delete a review, click “Delete” instead.

Review Search and Viewing Reviews:

Review Search:

In the search page there are 3 search fields. There are 2 drop down fields: “Course Subject”, “Course Number” and one text field: “Professor Last Name”. Any combination of these search fields can be used to identify courses and professors. Click the magnifying glass icon to conduct a search.

Course Subject	Course Number	Professor Last Name	
<input type="text"/>	<input type="text"/>	<input type="text" value="e.g. Hescat"/>	<input type="button" value="🔍"/>

Examples of searching for reviews:


- If no fields are filled out no results are returned:

Course Subject

Course Number

Professor Last Name

e.g. Hescat



No search results found.

- Searching for CS 1800 and leaving “Professor Last Name” blank returns all CS 1800 classes taught by different professors so as to easily check the performance of each professor teaching the class:

Course Subject


Course Number

Professor Last Name

CS

1800

e.g. Hescat



Course	Course Title	Professor
CS 1800	Discrete Structures	Nate Tucker
CS 1800	Discrete Structures	Bean Hescat
CS 1800	Discrete Structures	Videro Mihajlovikj

- Searching for only the course subject CS returns all CS courses and all the professors that teach it:


Course Subject

Course Number

Professor Last Name

CS

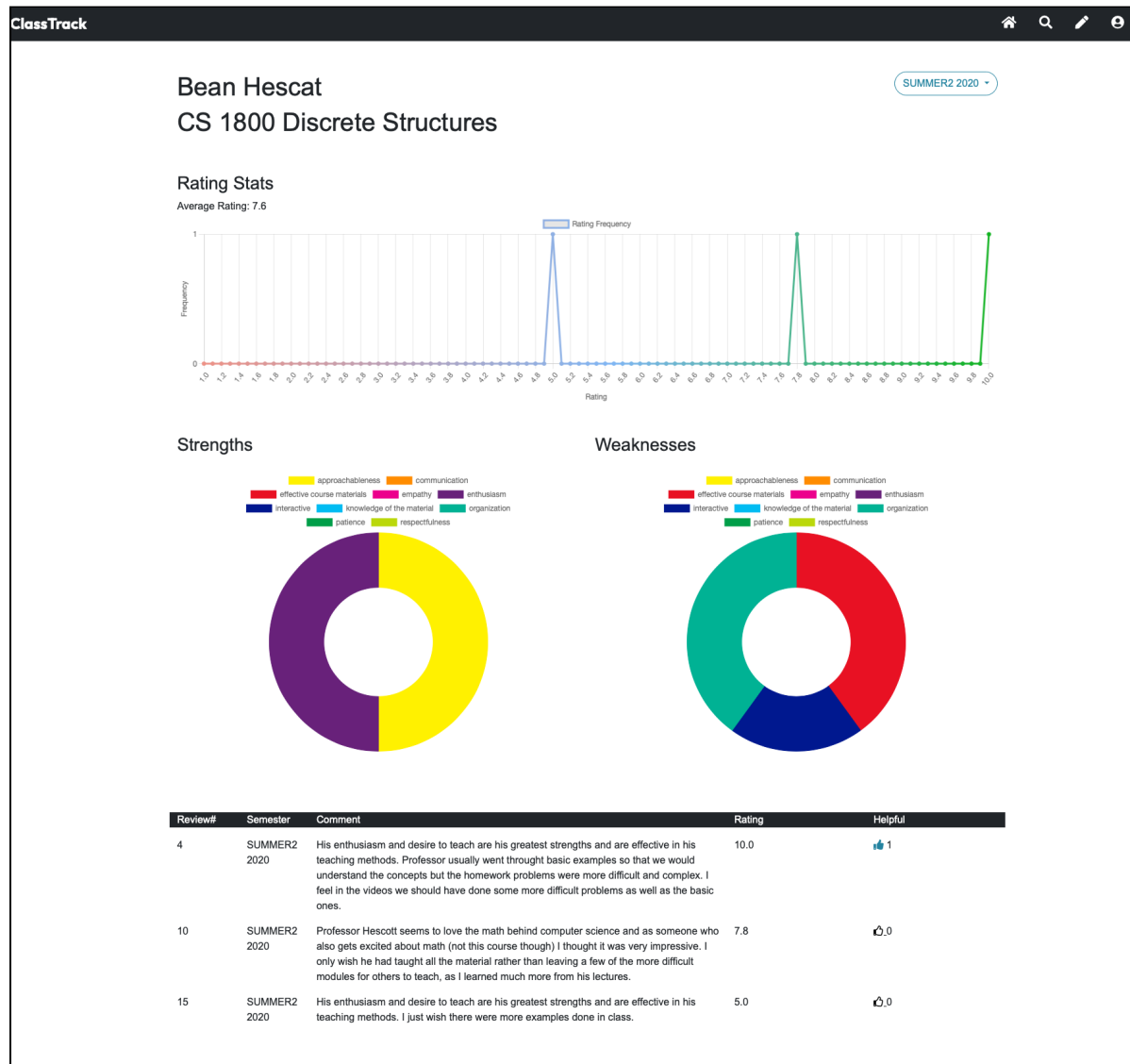
e.g. Hescat



Course	Course Title	Professor
CS 2800	Logic and Computation	Olin Shudders
CS 1800	Discrete Structures	Nate Tucker
CS 1800	Discrete Structures	Bean Hescat
CS 1800	Discrete Structures	Videro Mihajlovikj
CS 3800	Theory of Computation	Walker Schneider
CS 3200	Database Design	Katherine Durane

Viewing Reviews:

After [searching for a review](#), click one of the results to view reviews for a class taught by a professor:



Note the average rating at the top under “Rating Stats”. You can use the drop down menu in the top right corner to select a specific semester to view information for. The line graph represents the frequency of ratings and the donut charts represent the distribution of strengths and weaknesses respectively.

Below the chart, all requested reviews are displayed in a table. If a review is helpful you can click the thumbs up logo to mark it as such for other students. If you no longer find a review helpful you can click it again to unmark it.

Lessons Learned

Technical expertise gained

Kristin: I got to do a lot of practice with writing MySQL procedures and functions and recognizing when a trigger could come in handy. I also got to work with Chart.js for the first time to visualize our data on a Course by Professor Reviews page, which turned out to be really straightforward.

Agastya: While I had done a little web-dev work before for a co-op I was largely working on the back end side. I got to spend a lot of time on the front end for this project and learned a lot about prototyping webpages using Figma, using flask to create HTML templates and incorporating javascript to make a more visually appealing site (i.e. chart.js). While I am quite experienced with Python it was a whole different ball game thinking about how to pass data between all these layers, and was a great learning experience. We also created a lot of stored procedures and functions, and as we had only used a trigger in one assignment prior I got to become more familiar with them by implementing one for this project.

Insights, time management insights, data domain insights etc.

Kristin: It was really effective to get on the same page with my partner by making a checklist of things we had to do and doing a quick lo-fi prototype of our website. That kept us on the same page and also made it very obvious what kinds of queries we needed to write. We also pair-programmed for most of the project, which helped us catch each others' bugs and communicate what direction we wanted to go in.

Agastya: This was not a project that could be completed in one night and took a lot of organization and coordination to determine necessary tasks. It took creating a suitable data model, building out necessary stored procedures and functions and making returned data more visually digestible through good design. Figma was extremely helpful for making a conceptual idea into a specific user flow, and getting my partner and I on the same page. Effective communication was essential to completion of this project as we worked together a lot to get it done.

Realized or contemplated alternative design / approaches to the project

- We considered using a noSQL database such as MongoDB for this project. However, our data is highly structured, and for internal school use has little need to accommodate horizontal expansion due to a relatively fixed number of students. We favored the reliability and concrete nature of MySQL
- We considered doing this project as a console program to simplify our job. However, one of the biggest issues with TRACE is its unwieldy user interface and its approach of throwing a lot of information at you to the point where it can be difficult to make any sense of it. We believe that a simplified yet intuitive UI can be used to help students make concrete decisions about their choice of professor for a course.
- We also considered following Trace's approach of listing each professor's course per semester. Currently that means that you have to click through reviews for a course for every semester. We think that a cumulative approach allows a student to see the cumulative opinions of other students, and added a semester tab to also allow them to narrow long lists of reviews.

Document any code not working in this section

- Rating is "required" in the form for writing a review, but there is nothing enforcing that other than the fact that the default value is 5.0.
- The comment box in the review has a limit of 1000 characters, but the site 404s if someone goes over the limit.

Planned uses of the database

- Store entities such as students, professors and courses.
- Inserting and deleting likes/reviews.
 - Also editing reviews.
- Getting reviews for a specific course and professor.
- Storing entity relationships that can make for flexible searches for ease of use.
- Storing review data in a format to ease data visualization.

Potential areas for added functionality.

- It would be nice to have more sorting and filtering capabilities for viewing reviews. It may be convenient for users to filter or sort by review ratings, sort by age, sort by likes, etc.
- We also considered whether professors could have accounts too and respond to feedback.

- It would be useful to have head-to-head professor comparison pages to compare stats between them in one view.
- Implementing an “unhelpful button” to help students weigh the value of reviews when making important decisions.