Project Reflection Report

Our team built a web application where students can submit reviews for courses that they have taken at UIUC as well as professors that taught those courses. These reviews provide other students with information about different courses, professors, and departments to help inform their decisions about what courses to take in the future. Reviews include multiple aspects including a difficulty and usefulness rating, and the average and maximum hours the reviewer spent on the course each week. Users can then search for a particular course or professor and filter reviews based on these attributes. They can also search for a department which shows aggregate information about the department as a whole, and all the courses in the department.

This reflection provides a short overview of how this project developed over the course of the semester. Each of the required questions are answered separately below.

1. *Please list out changes in directions of your project if the final project is different from your original proposal (based on your stage 1 proposal submission).*

The direction of our project has not changed since our original proposal. The core concept of providing students with a centralized source of information has remained our primary goal. What has changed is additional functionality which helps regulate the website. Initially we had discussed some possible administrator functionality but had no specific requirements in mind. Furthermore, this functionality was not planned to be part of the core website since we figured administrators would have access to the databases directly. Now, we have started to add such functionality to the website itself through our advanced database programs.

2. *Discuss what you think your application achieved or failed to achieve regarding its usefulness.*

This application achieved its mission: provide useful information regarding courses, time spent, professors, etc. We wanted students to have a better understanding of many factors that are involved in the decision making process for choosing courses. Our application would likely be beneficial to many students at the university, not just students in the computer science department. In addition, it may be useful to prospective students who are considering attending UIUC or are uncertain what they wish to major in. Previously, there was no central source of information and students had to manually search the internet for any useful information. Usually this information is outdated, biased, or just unhelpful. Having our structured review system helps provide easy access to helpful information.

3. *Discuss if you changed the schema or source of the data for your application.*

For our administrative functionality, we updated our schema by adding three new tables: one to keep track of all words which are deemed inappropriate by the administrator, one which stores professors who have received offensive reviews and the number of offensive reviews they've received, and the last one keeps track of students who posted inappropriate comments (comments containing inappropriate words). The data source for our application did not change, data for the new tables are populated solely based on the inappropriate words which can be added directly through the administrator's unique URL.

The only change to the original tables is that we added an attribute to the Student relation which specifies whether the student is considered 'banned' or not. If students submit too many reviews with inappropriate words, this attribute is set to true based on our stored procedure which prevents that student from inserting new reviews.

4. *Discuss what you change to your ER diagram and/or your table implementations. What are some differences between the original design and the final design? Why? What do you think is a more suitable design?*

Except for the minor change discussed above, our original relation implementations have not changed. We added the new tables however these have no relationships with any other tables. During our initial design process, we spent substantial time on forming our relations and relationships so we thankfully did not have to alter them later. One improvement could be to account for more administrative attributes – such as the 'banned' attribute in the Student relation – instead of having new tables to store aggregate information as we currently do for our administrative functionality. This would reduce the total number of relations though it may make certain queries run longer.

5. *Discuss what functionalities you added or removed. Why?*

Core functionalities have remained the same since our initial proposal. However, we built on basic functionality between our midterm demo and final demo to improve the user experience. These improvements include adding filtering to review results to narrow the search results based on certain criteria like course usefulness or professor name. Previously, we just returned all reviews for the course or professor with no option to narrow or order results.

We also added autocomplete functionality to our search bar. When the user starts to enter text into the search bar, similar entries are shown in a drop-down below the search bar. The user can then click on one of these items to auto populate the search bar to that value.

Lastly, we started adding administrator functionality. This provides more flexibility with the application where admins can more easily interact with our system rather than having to manually edit or query tables through the DBMS. This also allows for more flexibility going forward since there is now a simple page where functionality can be added.

No functionality has been removed.

6. *Explain how you think your advanced database programs complement your application.*

The advanced database programs complement our application very well. For example, our trigger works to find students who make inappropriate remarks in their comment sections. Once a student is found to be violating our terms, their violating comment is immediately omitted. If this same student is found to be doing this three different times, they are temporarily banned from posting on our website. To become unbanned, they must edit these violating comments such that they no longer contain inappropriate words. The stored procedure part of our project assists the administrator of the website in running it logistically, and getting reports for things that they deem necessary. They can see if particular professors receive a lot of inappropriate reviews, or if certain students make too many inappropriate reviews. This information can then be relayed to the university if deemed necessary.

7. *Each team member should describe one technical challenge that the team encountered. This should be sufficiently detailed such that another future team could use this as helpful advice if they were to start a similar project or where to maintain your project.*

Jainil: Learning how to communicate a frontend application with the backend was a challenging part of the project, in my opinion. Additionally, learning how to host something on GCP, then create an application using NodeJS was a new concept for me. Creating indexes on the tables was difficult: knowing whether or not an index would be clustered or unclustered is useful, yet tricky. Not all of the indexes we created were useful.

Sushma: hosting on GCP was quite challenging. There were a lot of security issues, alongside application permissions, that I had to figure out. Additionally, the autocomplete functionality was quite confusing when we tried to implement it. It took a lot of debugging and fixing to get it working.

Viktor: Although we did not focus heavily on styling most of our templates, even the minimal amount of styling proved challenging. I personally had minimal past experience with front-end development and had forgotten most of what I had learned previously. Even integrating our advanced search form for the course and professor templates was tedious.

Michael: Web scraping was very difficult. I had to teach myself how HTML parsers work, as well as scrape through every page we thought was necessary. Generating random data was also quite difficult, as we had a table or two that had over 26,000 entries. Lastly, once all that was done, making sure that the data was entered correctly, and into the correct tables was a bit challenging.

8. *Are there other things that changed comparing the final application with the original proposal?*

Because our project remained very sturdy and fluent throughout the course of the semester, there was not anything that really changed with the final application, as compared to the original proposal.

9. *Describe future work that you think, other than the interface, that the application can improve on.*

We could potentially create more infrastructure and queries that allow sorting/aggregating data in different ways. We could also just have more filtering options for people to find things that may be more relevant to them: sort by the time courses were administered. In addition, the admin functionality can be expanded to allow admins to quickly run certain queries through our interface rather than having to do it manually through the DBMS. This functionality could include updating student standings at the start of a new year or adding new entries for new students at the start of each semester. This depends on whether or not the application could be integrated with actual UIUC data. If the application were connected to an existing database then a lot of this admin functionality is outsourced directly to the university. However, this is unlikely. So, having a more robust admin page can help improve the backend of the application that the average user does not see.

10. *Describe the final division of labor and how well you managed teamwork.*

Name	Division of Labor
Jainil	Creation of tables, relational schemas originally used, ideas for triggers/stored procedures, advanced queries, initial instance of database on MySQL. Initial designs of UML diagrams.
Viktor	Front-end development, hosting the application on GCP when doing demos, creating the initial application template and writing some of the queries to be translated.
Michael	Scraping/Parsing data from RateMyProfessors, creating .csv files with populated data (with randomly generated data for info we did not have), frontend/backend development, triggers,

	Stored Procedure, Schema Draft. Main page styling.
Sushma	Designing the UML diagram, Developing the database creation schema, Indexing to improve the query performance, developing the backend and front end for add review and edit review functions, deploying and hosting the application on gcp, developing the frontend and backend for autocomplete functionality.

In order to ensure successful teamwork, constant communication was of the utmost importance. When discussing upcoming milestones in the project, requirements, and preparing for demos, we met either in person (during/after class), or via group calls on Discord. Additionally, tasks were roughly divided up in the initial stages of the project, so we would spend less time figuring out who was responsible for their respective tasks, and more time debugging and configuring pieces together. We would complete pieces, let the rest of the team know what was complete (and where), and then let others test the functionality to ensure its reliability. This manner of communication and collaboration was really effective, and we believe this helped us put forth the best product possible.