Project Report

PROJECT DIRECTION CHANGES

We originally named our project Cookey and changed it to Fridgey.

We added more tables during the ER diagram phase. For example, a ratings and follows table.

There was not enough data on the website where we said we'd get data from in our project proposal so we randomly generated most of our data.

We also had to change the contents of some tables to ensure we were able to link and query data correctly. This involved splitting tables or removing certain fields and adding fields to improve data handling.

APPLICATION ACHIEVEMENTS

Our application fulfilled its usefulness in terms of being able to contain recipes from many different cuisines as a lot of current recipe websites cater to a certain cuisine/diet. I think it fell short in terms of listing the price for the recipes. Originally in our project proposal we wanted this feature because it'd allow the user to pick recipes based on price but as we didn't implement ingredients contained in a recipe until later, we didn't end up using price as a sorting option. Our application used calories and ratings instead to sort the recipes for the User so our project is useful to a different group of people than originally intended.

SCHEMA AND SOURCE DATA CHANGES

In the original proposal we were planning to find relevant data on the internet (dishes, recipes, ingredients and their measurements), but unfortunately we could not do that. Thus, we have to create all data randomly, which resulted in some names being weird and some dishes having unusual sets of ingredients.

The schema for our project stayed almost the same, we just had to correct a few things that we made mistakes in during our proposal stage.

DESIGN AND ER DIAGRAM CHANGES

Our table implementations and ER diagrams remained the same throughout the project. We used our original design for all the stages following the stage where we had submitted the original ER diagram.

FUNCTIONALITY CHANGES

In our project proposal we wanted to sort the recipes based on the price of ingredients. We ended up sorting based on calories and ratings. We changed this because we changed the target demographic for our app to be people who cared about the quality and nutrition of their food rather than people trying to budget. Sorting this way also allowed us to create more advanced SQL queries. We were originally going to use the name of an ingredient or dish to delete them but we ended up using the DishId because we wanted the process to be more secure as many dishes could have the same name and using DishId makes it possible to only delete one of those dishes.

ADVANCED DATABASE PROGRAMS

The stored procedure uses the average calories of dishes made by a user and the rating they receive. This information is used to display information to the user about the calorie profiles of a particular user and how they rank. They can make an informed choice about what they like to eat and which user they want to follow based on their dishes.

The trigger also creates a discount for bulk items. This is keeping in mind our vision for future expansions where we could have a tie-in with wholesale retailers and get discounted prices for our users.

The website we have involves a lot of user interaction with a lot of data manipulation and these advanced database programs help maintain the database and condense data while managing a huge intake of inputs from the users.

TECHNICAL CHALLENGES

Indexing was a really hard part of this project. We did not use mySQL and used MariaDB instead on which the metrics were displayed differently. At first, we didn't really understand what was being displayed. After a bit of research, we generated JSON files for the indexing reports and went through a lot of time related data to figure out the query performance and what index was really helpful.

Useful commands for indexing: ANALYZE FORMAT=JSON

What to look for while testing the index: run time of main queries in addition to the run times of the subquery or the tables being queried.

Displaying our tables for search queries because we weren't able to display post requests. The way we got around this struggle is by using a get request and req.query to access user inputs as get requests don’t have a req.body.

We also had difficulties accessing the GCP VM. Because some of our group members were outside the US, GCP didn’t work even with a VPN, and caused a lot of disconnections. This made working on the VM impossible. We worked around this by dividing labor and integrating everything together on a call. Our credits also ran out and it took us a while to figure out what was happening because we originally believed GCP was down before asking on campuswire.

OTHER CHANGES

The biggest change is the frontend since our website looks completely different from the design that was originally proposed. We had to make it much simpler, because there were only three people in our team and our main priority was functionality, not the web-page appearance.

FUTURE IMPROVEMENTS

We want to make a login system instead of having only UserId to make it more secure. As the database would expand we would also like to find better values to index on to improve query efficiency. Also more ways to store user related data and more stored procedures to execute this rather than maintain various tables and join them every time would benefit our application a lot. One more thing to do is to make the web-page more user friendly: it should be easy to navigate through the web-site, but also it shouldn’t look messy. Thus, making the application look more aesthetically pleasing and organized would also be among our next steps.In addition to this, a link to another

retailer that sells the ingredients used in the dishes could help generate more accurate and consistent prices and will also give more meaning to our trigger.

TEAMWORK AND LABOR DIVISION

We had group meetings at 9am on most days post Stage 3. Everyone would take an active part in the calls and help debug and ideate for the project.

Rohan and Irene had issues connecting to GCP and were in a different timezone, so they would think of ideas and come up with pseudo codes for different applications involved in the project. This was presented in the call.

Aishani integrated all the code and expanded what was ideated to help make the project work. This was done during the calls for the most part. She also expanded display tables and added some code in her free time.

Everyone also put in time to think about how we could improve the project in their free time and presented these ideas on the call.

It was a highly collaborative project and all members were involved in development at each step of the project.