# Report Answers

Link to original Stage 1 proposal:
https://github.com/cs411-alawini/sp24-cs411-team035-dass/blob/main/doc/Project_Proposal.md

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

The most important changes in the direction of our project are listed below:

1. Changes in functionality: Did not implement the “sorting”, “Thumbs up”, and “Journal” functionality as mentioned in stage 1. However, we added a “User functionality” (where users can change password), filtering based on dropdowns where the user can select ingredients, category, nutrition, and nutrition ranges for their recipes, as well as a “Favorite Ranking” feature that displays the recipes that are the most popular among users. We also expanded on the “MyRecipes” inserting functionality, that allows users to add in their own, customized recipes by inserting the RecipeName, Directions, and ingredients.

A more detailed description of these changes are in the question (Discuss what functionalities you added or removed. Why?) below.

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

What the application achieved regarding its usefulness:

- Recipe Ranking feature: users have the ability to rank recipes based on ingredients, category of recipe, a nutrition feature of their choice, and limits of that nutrition (where only recipes belonging to that category, which have certain ingredients, and a nutrition value that is less than or greater than a chosen value. We believe this is very useful for helping users find which recipes they want!
- User MyRecipe Feature: Users can have the ability to upload their own recipes, which is very helpful! If they have a recipe they want to remember for the future, they can insert that recipe into our application so that they always have it for later. They can also delete recipes.
- User Change Password Feature: Users can change their password, much like a real-world application.

What the application failed to achieve regarding its usefulness:

- We could have incorporated a login feature, which would allow users to log in and log out using their login ID and password like a real website.
- Users could have an option to “Like” recipes. Although there was a favorite count ranking feature, we could have implemented a way for users to rank recipes or heart them, and have all “hearted” recipes stored in another database specifically for that user.

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

- We did not change the source of data for our application.
- There were a few unnecessary columns in our original source of data (such as the source of the recipe, the measurements, and the exact measurements – unnecessary columns such as these were not used in our final application).

**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?**

- Relative to our Stage 2 doc, we only changed the MyRecipes table - originally, we had a ThumbsUp (INT) attribute, but we did not end up adding this in our application. The rest of the table attributes remained the same.
- We originally called the table with all the common food ingredients “Nutrition Facts”, but we changed this to “Food” in our final implementation.
- The rest of our schema from Stage 2 remained the same and was fully reflected in our final application.
- We believe that a more suitable design from the original and final design would be the final design, as it was very concise and easy to understand, without including unnecessary table attributes. We believe that the Recipes had enough information (such as ingredients, directions, and recipe title), and it was a good design choice to connect the Recipe table to the Food table for ease of access to recipes that contained certain ingredients that the user wanted.

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

The functionalities that were changed are described below:
The changes from our original proposal are:

1. We mentioned that when the user “types” the ingredients, the app will recommend possible recipes that can be made with those ingredients. We ended up changing the typing option for ingredients, category, and nutrition to drop-downs with possible choices that the user can select from.
2. We mentioned that the recommendation would be “sorted and filtered by the user’s preference and also the number of common ingredients with the recipe and the user’s input,” however, we did not include this sorting feature, and instead just did a query to filter out the recipes that aligned with the user’s preferences: an ingredient, category to choose recipes from, and a chosen nutrition value less than or greater than a specified amount. We mentioned sorting multiple times in our proposal. We also mentioned how users can choose how the list should be sorted (i.e. more common ingredients with input, less complicated, less difficult; however, we did not get to implement these specific sorting features when the user selected their choice of ingredients/category/nutrition. We instead implemented other sorting features, described below.

3. We mentioned in our “Usefulness” that we can have a journal feature that will allow users to create journal entries, and also have a save/favorite/bookmark functionality within this journal feature. However, we were not able to implement this journal feature, and instead creating a “MyRecipes” feature where users can upload their own recipes. This is described more in detail below.
4. We added several features. They include a “MyRecipes” feature along with a “User” component, where users can create a login with their user.name and user.password. Users have the ability to change their password. Users can upload Recipes that include the list of ingredients, directions, and the recipe title. Users can also delete their MyRecipes. This recipe will be associated with the user and will be added to the MyRecipes database. A user can have many recipes, but a recipe only belongs to one user.
5. We also made a Fav_Ranking feature, which shows the recipes that were selected as “favorite” by users, and the number of users who selected that recipe as their favorite.
6. We created a Users_Ranking feature also, that shows the users and their number of favorite recipes.

We added functionalities 3, 4, 5, 6 because we thought this could give users a more accessible and fun recipe-finding experience that would aid their personal health goals. For instance, we added a nutrition range filter because we thought that finding recipes that have a certain amount of Protein, Fat, or Carbs is very realistic and important for any individual searching for healthy recipes to make, if that is their goal. Furthermore, functionalities like the “Password change” functionality for users and the favorite ranking feature make the website application a lot more realistic and customizable.

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

We think that advanced database programs complement the application because of the sheer size and volume of our recipes and foods database. Specifically, our original Recipes dataset had over 2M recipes! We had to shorten this down to 3500 recipes along with 1200 Food items. We believe that without using an advanced database system to implement queries in the backend, it would have been almost impossible to navigate through our database which included thousands of observations. We believe that using SQL not only increased the efficiency of our algorithms (as seen through the Indexing done in Stage 3) but also increased the replicability of our application - which would allow users interested in it to replicate and also add onto it since using a database is something that is understood by most individuals in tech.

**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.**

1. One technical challenge that our team faced was displaying the Top User Rankings after clicking the button. We had built a clickable button to click and display the Top User Rankings, however upon clicking it, no entries would actually display in the UI. No errors were present in the console either, so there were no API issues. When the button clicked, we would see the headers pop up, but no data actually displayed. We checked the front end and back end to make sure we had our connections properly set up, and our button was calling them properly. Upon further inspection, we realized that we were using the same rankBody component for the User Rankings button and another Rankings button, causing conflicting queries and resulting in no output. We changed to create a different component for the Users button, and our data was able to display.

2. Another technical challenge that we encountered was connecting the VM with our IP address. We followed every step from the 411 course's tutorial for setting up for VM and GCP, but when we worked on SSH in our VM and followed the starting tutorial for index.ejs file for our own IP address, we couldn't connect to our website with our IP address. We solved this problem by adding our IP address in SQL->Connections and added our IP address in the Authorized Networks section to access our instance.

3. One technical challenge that the team encountered was GitHub issues, specifically in working with the code simultaneously while avoiding merge conflicts and branch issues in our code repository. We felt that it would be difficult to resolve merge conflicts if each team member worked on the code individually, and would be difficult to test without other components. We were able to partly resolve this challenge by having coding sessions where most team members were present via Zoom, and we all worked on a common set of files using VSCode Liveshare. This made it much easier to work on the code simultaneously and made it easier to test and push a common, singular file to GitHub without any issues.

4. There were a lot of technical issues with fetching the data into the backend and displaying it on the front end. Firstly, getting the user's input from the front end and using the value of that input in the query in the backend was difficult at first. But we figured out how to pass the value by using data: {} in index.ejs and get that value in the backend with req.query. Also sometimes it was confusing to know req.query is right or req.params is right but we figured out it now. We spent a lot of time why we could not fetch the values in Put and Delete, but the reason was we needed to use req.body instead of query or params.

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

For the dataset, the recipe dataset we planned to use was too big to open, so we trimmed it and used it. Also, we modified the datasets before we inserted them into SQL such as making the words in lowercase or removing the special characters districts the reading of data in SQL.

For the functionality, there were many changes after we realized some functionalities were not efficient for using advanced queries or the purpose of the app we made. We planned to make the user sort the recipes (output of filtering) according to their preferences, but we made the filtering and sorting a separate functionality. We made users filter the recipes based on ingredients, category, and amount of some specific nutrition. And we made a new tap that shows the recipes in order of popularity and how many users clicked that recipe as a favorite.

Lastly, we added some user authentication. When user type in their user, they can change their password, can see the recipes they uploaded and delete, and upload a new recipe. According to how many recipes the user uploaded, we made a tap that shows the ranking of users.
This user authentication was not planned with details in the beginning but after we had several meetings with the TA and got feedback, we developed it like this.

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

One thing our application could improve on is adding more relevant features. Currently, we have features to allow for basic CRUD operations, however, we could enhance functionality by adding some more robust features. Some ideas for features could be more advanced search filters, to filter for various other categories that we include in our database. Furthermore, we could add a more personalized feature by allowing users to save “food and nutrition” preferences to their account. This way, the recipe finder could find recipes that are also relevant to the users' preferences stored in their accounts. One cool feature could be adding an allergies section to the user preferences. If a user has an allergy, any recipes with that ingredient could be instantly filtered out and not shown at all. Furthermore, we could try and build a “similar recipe” feature. When the user filters for certain recipes, we could output exact matches but also show some close matches that are only off by a few values, in case the user wants to try for an alternative recipe. I also think we could make our application faster because there is some delay in response after clicking a button and actually seeing results, so improving time efficiency would be helpful to produce quicker results. Furthermore, we could improve the readability of our application by writing more concise and efficient code, so it is easier to debug and add additional functionality later on.

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

All of our members worked on making databases and connecting the databases to GCP. Dawn and Sunwoo worked on writing a draft of our web page, working on the frontend and backend and connecting. Anagha worked on triggers, constraints, stored procedures, and some sections of the frontend and backend for MyRecipe ranking and update operation for the user’s password. Shalini worked on transactions and a section of frontend and backend for

UserRanking. Sunwoo and Dawn worked on connecting the keyword search and crud operations (reads, insertions, updates, and deletions) in both the frontend and backend. Most of the time, we managed to meet through Zoom meeting and share each person's progress, and resolve any struggles that each teammate had.