Michael McDermott

CS441 Software Engineering

Midterm Personal Project Write (I)

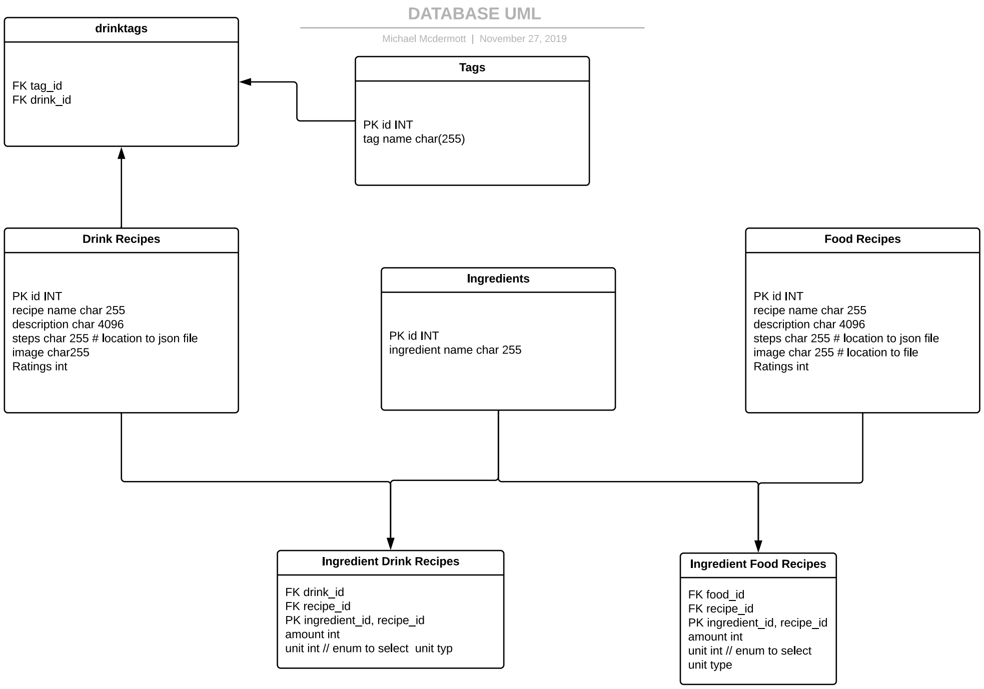
At the beginning of this project, I had no knowledge of deployment of a server using cloud services, configuration of a database in a cloud, nor flask as a framework for a server. Our front end was built on HTML, CSS, Javascript, and Apache Cordova. I did not end up doing any software development on that side of the application. For the cloud storage, I learned Google Cloud Platform through their documentation on the website and through very specific google searches on how to set things up like create cloud storage to store static files and how to change permissions on stored data. I also had to search things like how to add IP addresses that would be allowed to access the database to view or edit so that our team members could work from home and manage the database. The database itself is a SQL database that is part of Google Cloud Platform. The Pycharm IDE has a nice database feature built into it which made it easy to view and update the database on the fly. It even has a generate SQL feature which made it easy to duplicate the rules of tables. This came in handy when we finished functionality of the drinks table, but wanted to add the same to the food table. Python was the choice of language for the server and Flask was the framework. To learn Flask, I used a combination of my peers who have used it before, online documentation and tutorials, and YouTube videos.

Most of the requirements and use cases were decided collectively as a group or refined as a group. An example of this was deciding on what the starting page would say when the app is launched. Originally there was going to be an account creation page, but we decided to skip that and go straight to where the user chooses whether they want to search drink or food recipes. One requirement I wanted was there to be an admin feature for an admin to log in and be able to edit the recipes or upload new recipes, but this was a low priority and has not been implemented yet.

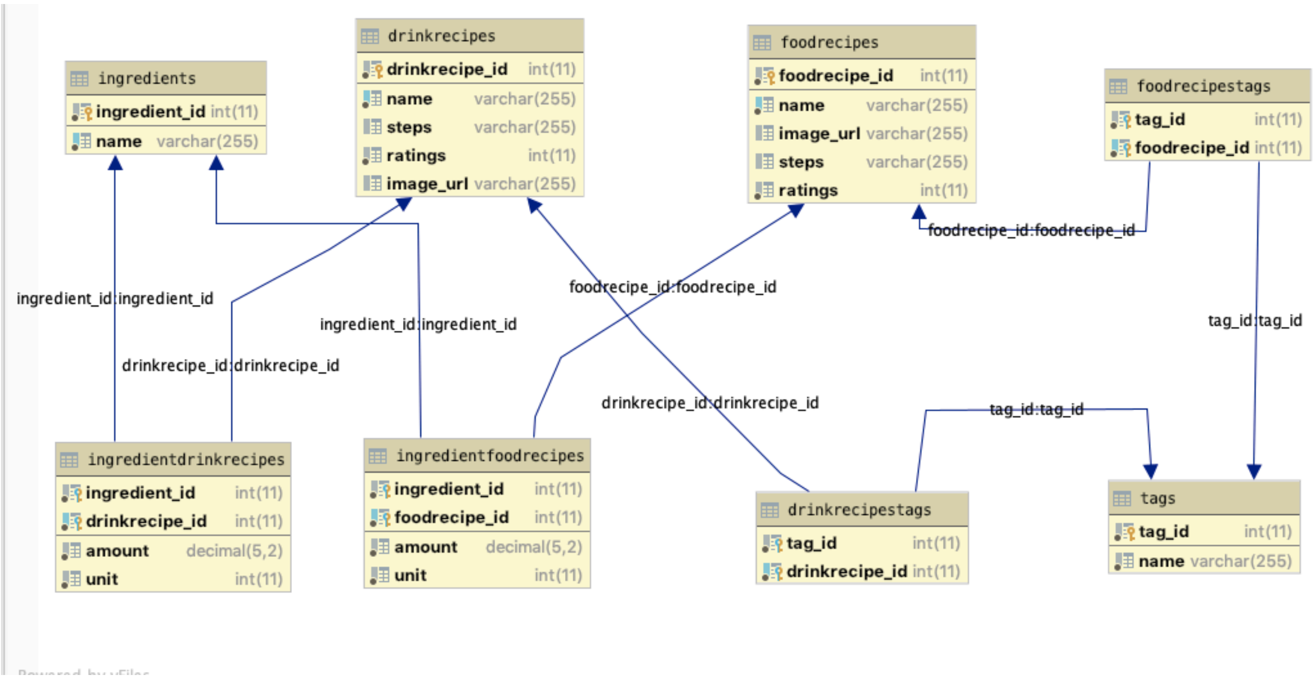
The main use case that I developed is searching for a food recipe by name, which is still a work in progress. In the beginning, I was working coding the database queries for food recipes, but found that it was going to be the same as the drink recipes which was already in progress, so it would have just been duplicated work.

In the beginning, I worked with the backend team to design the entity relationship diagram (ERD) for the database, which evolved over time into the figure below. The main thing that changed was we created a table for all the tags and then made a many to many relationship table between tags and drinks and between tags and food.

Initial Database ERD:



Current ERD:



ingredients	
ingredient_id	int(11)
name	varchar(255)

drinkrecipes	
drinkrecipe_id	int(11)
name	varchar(255)
steps	varchar(255)
ratings	int(11)
image_url	varchar(255)

foodrecipes	
foodrecipe_id	int(11)
name	varchar(255)
image_url	varchar(255)
steps	varchar(255)
ratings	int(11)

foodrecipestags	
tag_id	int(11)
foodrecipe_id	int(11)

ingredientdrinkrecipes	
ingredient_id	int(11)
drinkrecipe_id	int(11)
amount	decimal(5,2)
unit	int(11)

ingredientfoodrecipes	
ingredient_id	int(11)
foodrecipe_id	int(11)
amount	decimal(5,2)
unit	int(11)

drinkrecipestags	
tag_id	int(11)
drinkrecipe_id	int(11)

tags	
tag_id	int(11)
name	varchar(255)

ingredient_id:ingredient_id
drinkrecipe_id:drinkrecipe_id
ingredient_id:ingredient_id
foodrecipe_id:foodrecipe_id
drinkrecipe_id:drinkrecipe_id
foodrecipe_id:foodrecipe_id
tag_id:tag_id
tag_id:tag_id
Powered by yFiles

I created a database handler class that should act as a singleton and handle all of the query sanitization and getting data from database and putting it in json, this would be passed to the server which can then relay that to the user. This dbHandler class has not been integrated into the code yet and will when we get to refactoring. I also created all the json files for the recipe steps and stored them in Google Cloud Storage as well as organizing and storing all the images for the recipes in Cloud Storage.

Next steps would be modularizing the database queries and making the API more restful. Finishing up the food queries and creating a way for an admin to add full recipes to the database rather than having to do it manually would be next tasks for me to accomplish.