9/30/24 - Week 2 Updates

- Tobias managed to get the download to work for the Open Recipe Database, so now we have a very very long JSON of recipe data - name, ingredients, URL, image, and some other data
    - We agreed that we are going to crop it to 1000 recipe entries and create a new database table for the recipes
    - We're going to put each individual recipe JSON into the table that way we don't have to parse through the JSON every time the recipe generator runs - it can just fetch the recipes from the database
    - Recipe ranksort + generator pseudocode at the bottom of the file
- We need to determine our ranking criteria for searching/generating recipes in the recipe generator based on pantry ingredients
- Brand scheming finalized:
    - Logo: Chef Hat
    - Logo Font: Philosopher
    - Text Font: Albert Sans
    - Color Scheme: Yellow (highlight/accent), Blue (text, sections, default-type stuff), White (background)
- Alanah created the MySQL database and started some PHP.
    - Will continue to work on it tonight and finish up some loose ends with the user id.
    - Will continue to work on the login using the Bcrypt API
- Discussing database architecture options
    - Import recipe JSON database into php database (JS)
    - Search JSON manually and then interact with php database(memory heavy, slow)
- Discussing search/ranking algorithm
    - How will we search for recipes based on certain criteria assuming our entire recipe database exists on the server (php)?
        - "Score ranking"
            - Ingredient exists = +1, lacking ingredient = -1, recipes lacking all ingredients are disregarded
            - Set intersection

Recipe ranksort + generator pseudocode:

function b pseudocode:

```
SELECT * FROM ALL_RECIPES (get every row from the recipe table)

idk what format that is returned in but im assuming its iterable (json or array) and im gonna call the returned data recipelist

create a hashmap

for recipe in recipelist:
  save the recipe id in a variable called recipeID
  get recipe ingredient list
  for ingredient in ingredient list:
    check to see if there's a hashmap key for the ingredient yet
```

9/30/24 - Week 2 Updates

```
    if (there is){
       add the recipeID to the hashmap key's value list
    } else {
       create a new key for the hashmap for the new ingredient and create a recipe
       list to serve as the value of that ingredient key and make the recipeID the first
       id in that list
    }
```

so now you have a hashmap that maps every ingredient to all of the recipeIDs that are recipes associated with that ingredient

CREATE TABLE ALL_RECIPE_INGREDIENTS

for each key in the hashmap, create a new row in the ALL_RECIPE_INGREDIENTS table where there is an id (int), ingredient (varchar), and recipeID list (JSON)