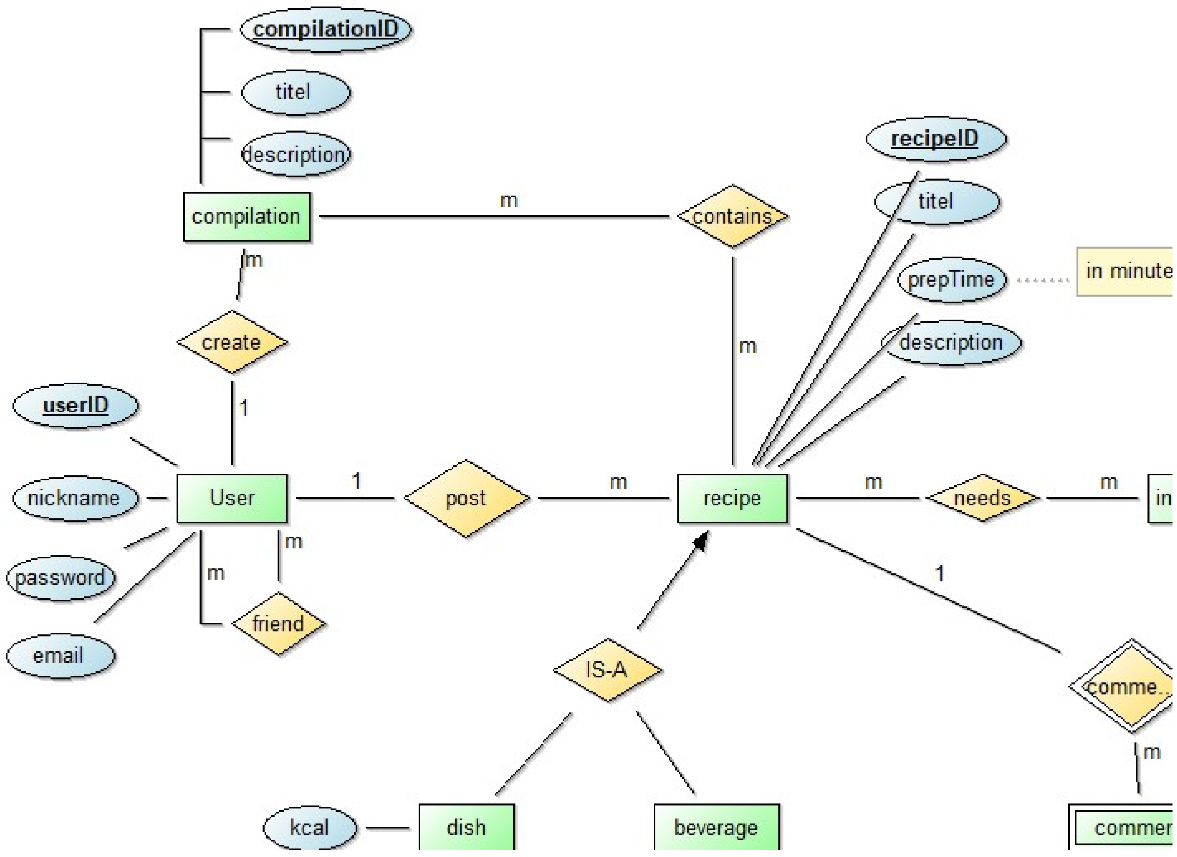
MY COOKING DAIRY

Milestone 1: Requirements Analysis

Each user has a nickname, a password, a email address, and a unique userID that is assigned to them upon registration. Once successfully registered, a user can befriend other users, post recipes, and create recipe compilations.
Compilations need a title, a description, and a compilationID. A title does not have to be unique because the different compilations are distinguished by the compilationID.
A recipe has a recipeID, a title, a time, a prepTime, and description. Recipes can be further categorized as a dish or beverage.
Dishes have kcal and a category as additional attributes. Whereas beverages have an alcohol percentage and an HOTorCOLD-attribute as additional attributes.
Recipes need ingredients which consist of an ingridientID, a name, and an allergen code.
People can anonymously share their opinions about the recipes in a comment section below. A comment is made up of a commentID, a star rating, and the actual comment.

Milestone 1: Conceptual Design



Milestone 2: Logical Desgin

users(userID, nickname, password, *email*)
PK={userID}

friends(*user1, user2*)
PK={user1,user2}
FK: friend.user1◊ users.userID
FK: friend.user2◊ users.userID

recipe(recipeID, titel, prepTime, description, *writer*)
PK = {recipeID}
FK: recipe.writer◊ users.userID

dish(kcal, category, *recipeID*)
PK = {recipeID}
FK: dish.recipeID ◊ recipe. recipeID

beverage(alcoholPercentage, HOTorCOLD,recipeID)
PK = {recipeID}
FK: beverage.recipeID ◊ recipe. recipeID

ingredients(ingredientID, ingredientName, allergenCode)
PK = {ingredientID}

needs(recipe, ingredient)
PK = {recipe, ingredient}
FK: needs.recipe ◊ recipe. recipeID
FK: needs.ingredient ◊ ingredients. ingredientsID

comments(commentID, rating, commented, recipe)
PK = {commentID, recipe}
FK: comment.recipe ◊ recipe.recipeID

compilations(compilationID, titel, description, userID)
PK = {compilationD}
FK: compilation.userID ◊ users.userID

contains(recipe, compilation)
PK = {recipe, compilation}
FK: add.recipe ◊ recipe.recipeID
FK: add.compilation ◊ compilation.compilationID

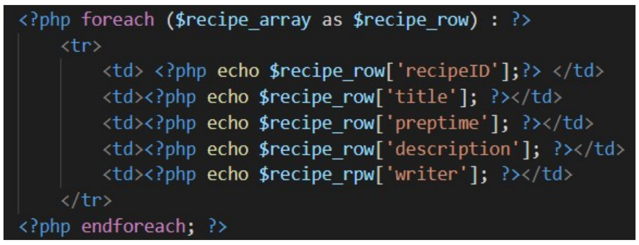
Milestone 4: Implementation

Java

At first, I generated a .txt file for each table and read the data from it. However, I later decided to write a "DataGenerator" class that generated data on runtime. The main components of this class were arrays which were filled with data and the Random.nextInt function.
One problem I encountered was that the insertion took a very long time (~10min).
The solution to this problem was to execute my statements in batches of 100 which tremendously reduced the execution time to around 1min.

PHP

I got an error message that the indizes of the $recipe_row array do not exist. I assumed that the index names had to match the names in my sql script but I was proven wrong and found out the they all had to be in uppercase.



```
<?php foreach ($recipe_array as $recipe_row) : ?>
    <tr>
        <td> <?php echo $recipe_row['recipeID'];?> </td>
        <td><?php echo $recipe_row['title']; ?></td>
        <td><?php echo $recipe_row['preptime']; ?></td>
        <td><?php echo $recipe_row['description']; ?></td>
        <td><?php echo $recipe_rpw['writer']; ?></td>
    </tr>
<?php endforeach; ?>
```

Another problem I encountered was that the data which I inserted using the SQL Developer did not show up on my website. This happened because the statements were not committed right after insertion. To solve this problem I either had to commit or close the SQL Developer.