Team members

Martin Hill, Connor Baldes, Group 30

Project Title

Dope Eats

Peer Review Feedback and actions

“Yes. However I would recommend you choose either snake case or camel case, as of right now you have (Date_added) and IngredientID.”
“Everything is capitalized except for things that end in ID, which are not capitalized. It is consistent this way, and if it works it works, but it is unclear why the things ending in ID are the only ones not capitalized.”
“The names are all consistent so far but there still seems to be lots of possible error in the future so stay consistent with your naming!”
“There is some inconsistency between attribute names using camelCase versus snake_case particularly in the outline.”
Updated all attribute names to camel case.

“While none of the entities have a summary, their titles make them fairly easy to tell what they are aiming to store. However, the distinction between what is different about the ingredients in recipes database and the ingredients database is unclear.”
“Adding more descriptions of the purpose of each database would help define why you need them and what they do.”
“The entity names are fairly descriptive though it would be nice to describe the purpose in words. This can also help you pinpoint other possible attributes like unit measurements for quantity.”
Added table descriptions.

“There could be more details about how the dietary restrictions could be apart of a database.”
“Yes though I am missing some data that contextualizes the user context, such as how many users visit the site and how many recipes are there or their length.”
Updated overview to incorporate dietary restrictions and specific sizes on data stored in the database.

“The recipe entity is not complete and does have any connection to ingredients that is shown yet. This may be fixed in the future of the class though and i'm not sure how you'd store all the ingredients besides just a large string listing them all.”
“There are currently many to many relationships but I think there is potential for them here if you merge the ingredients with the ingredients in recipes and say that ingredients can have many recipes and recipes can have many ingredients.”
Ingredients are stored in the Ingredients in Recipes table.

Upgrades to Draft

Added FKs to Ingredients in Recipes, Recipes, User Recipes, and Users.
Added additional attributes that will give the tables more context.
Removed “not NULL” designation from restrictionID in users and recipes.

Updated Overview

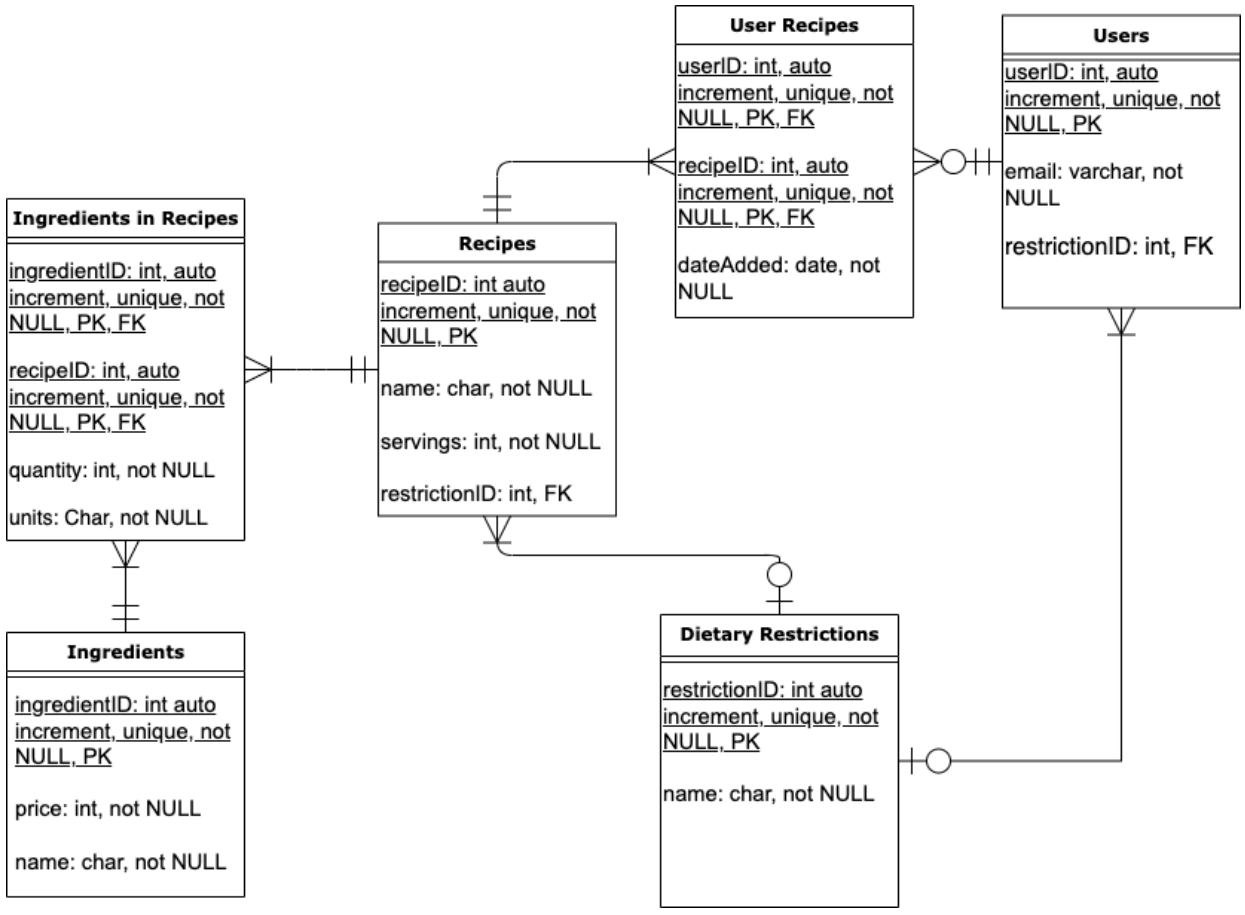
Our client is a company named “Dope Eats” which provides an online platform that connects thousands of food enthusiasts with a database of over 10,000 diverse recipes designed to accommodate a range of over 50 dietary restrictions. For this platform Dope Eats needs a relational database driven website which is effective at managing user profiles, and contains a library of recipes, ingredients, and dietary restrictions. Specifically, the platform should allow users to browse different recipes, as well as make specific searches for recipes containing desired ingredients or catering to their dietary restriction. Users should also be able to save recipes they like in a way that they will be easily accessible in the future. Dope Eats has a stable and steadily growing user base, and their desire is that the platform puts an emphasis on simplicity and usability while also requiring minimal system maintenance.

Updated Outline

- Users - database of users, storing their email, ID connects to recipes
    - userID: int, auto increment, unique, not NULL, PK
    - email: varchar, not NULL
    - restrictionID: int, FK
    - M:N relationship with Recipes
    - M:N relationship with Dietary Restrictions
- Recipes - database of recipes, storing name and serving, ID connects to ingredients, users, and dietary restrictions
    - recipeID: int auto increment, unique, not NULL, PK
    - name: char, not NULL
    - servings: int, not NULL
    - restrictionID: int, FK
    - M:N relationship with Users
    - 1:N relationship with Dietary Restrictions
    - M:N relationship with Ingredients
- Ingredients - database of ingredients, storing their price and name, ID connects to recipes
    - ingredientID: int auto increment, unique, not NULL, PK
    - price: int, not NULL
    - name: char, not NULL
    - M:N relationship with Recipes
- Dietary Restrictions - database of dietary restrictions storing name, ID connects to users and recipes

    - restrictionID: int auto increment, unique, not NULL, PK
    - name: char, not NULL
    - M:1 relationship with Users
    - M:1 relationship with Recipes
- User Recipes - database to join Users and Recipes. Stores both IDs and date added
    - userID: int auto increment, unique, not NULL, PK, FK
    - recipeID: int auto increment, unique, not NULL, PK, FK
    - dateAdded: date, not NULL
    - 1:N relationship with Users
    - 1:N relationship with Recipes
- Ingredients in Recipes - database to connect recipes and ingredients, stores ID and quantity
    - recipeID: int auto increment, unique, not NULL, PK, FK
    - ingredientID: int auto increment, unique, not NULL, PK, FK
    - quantity: int, not NULL
    - units: char, not NULL
    - 1:N relationship with ingredients
    - 1:N relationship with Recipes

Updated ERD



Normalization Spreadsheet with Example Data

3NF

Users

userID	email	restrictionID
1	john.doe@example.com	1
2	jane.smith@example.com	2
3	mark.jones@example.com	3
4	emma.taylor@example.com	1
5	will.brown@example.com	2

Recipes

recipeID	name	servings	restrictionID
1	Tomato Soup	4	1
2	Grilled Chicken	2	NULL
3	Vegan Tacos	4	3
4	Gluten-free Pasta	4	2
5	Garden Salad	2	1

User Recipes

userID	recipeID	dateAdded
1	1	2023-04-01
2	2	2023-04-05
3	3	2023-04-10
4	4	2023-04-15
5	5	2023-04-20

Ingredients

ingredientID	price	name
1	1	Tomatoes
2	2	Chicken
3	1	Lettuce
4	3	Vegan Cheese
5	1	Gluten-free Noodles

Dietyary Restrictions

restrictionID	name
1	Vegetarian
2	Gluten-free
3	Vegan

Ingredients in Recipes

recipeID	ingredientID	quantity	units
1	1	4	cups
1	3	2	cups
2	2	1	pounds
3	4	1	packages
4	5	1	packages
5	3	4	cups
5	1	2	cups