Team members

Martin Hill, Connor Baldes, Group 30

Project Title

Dope Eats

Website

http://flip1.engr.oregonstate.edu:56305/

Updates from Step 4:

- Added User CRUD functionality and implemented it on website
- Added User Recipes CRUD functionality and implemented it on website
- Added Recipes CRUD functionality and implemented it on website
- Added Ingredients CRUD functionality in code, not implemented on website
- Added Dietary Restrictions CRUD functionality in code, not implemented on website
- Added Recipe Ingredients CRUD functionality in code, not implemented on website

Peer Review Feedback Step 4:

Do the implemented CRUD steps function as the team expects? If the team stated that a CRUD step worked but you found an error, please let them know.

- The user table is working for adding, deleting, and updating.

Would a user easily be able to use the UI to complete the step? If not, or you have suggestions for how the UI can be improved, then please detail the suggestions.

- I think the UI is good but I would suggest displaying the actual restriction for the user table instead of the ID.

What suggestions do you have for the team in any areas where they are blocked or are having difficulty? Detailed and helpful feedback will receive full credit.

- remove the duplicate CRUD on the ingredients, dietary restrictions, user recipe, and recipe ingredients.

Do the implemented CRUD steps function as the team expects? If the team stated that a CRUD step worked but you found an error, please let them know.

The update doesn't work as mentioned in the post description. CRD works as expected.

Would a user easily be able to use the UI to complete the step? If not, or you have suggestions for how the UI can be improved, then please detail the suggestions.

The UI does make sense, although the edit button in the table is slightly confusing, as it doesn't seem to do anything. The edit menu is always present, so the button doesn't appear to have any functionality?

The same goes for the add user button at the top of the page. The add user menu is always present below the table, so the button at the top doesn't seem to do anything, and even links to a page that doesn't exist and errors out.

What suggestions do you have for the team in any areas where they are blocked or are having difficulty? Detailed and helpful feedback will receive full credit.

Other than implementing the update portion, I'd recommend cleaning up the redundant UI features

as explained above (add user button, edit button), and while not necessary, making the UI nicer looking could help. Given that that wasn't a deliverable for this draft, it is not a big concern, though

Do the implemented CRUD steps function as the team expects? If the team stated that a CRUD step worked but you found an error, please let them know.

- Create, Read, and Delete work for the Users tab, but not Update, although the group already knows this.

Would a user easily be able to use the UI to complete the step? If not, or you have suggestions for how the UI can be improved, then please detail the suggestions.

- There is an Add New link above the table, a New link in the table, and an Add user area under the table. I would reccomend only having one area that allows for New, similar to how you have delete set up. The same goes for the Edit section.

What suggestions do you have for the team in any areas where they are blocked or are having difficulty? Detailed and helpful feedback will receive full credit.

- I don't know how using python as the app differs from using javascript, which I used. However, what you have in your app.py file looks like it should work. Maybe to fix it, instead of using a passed ID from the function edit_user, you take id with something like userID = request.form["userID"].

Updates from Step 3

Updated UI spacing and labelling errors
Added cascading delete
Added dynamic search

Peer Review Feedback and Actions Taken from Step 2

“There is no schema present”
Added Schema

“I would however recommend removing spaces for the entity names like User Recipes to UserRecipes and perhaps renaming “Ingredients in Recipe” to something like RecipeIngredients” Changed name to recipeIngredients

“create a userRecipeID as the PK and then have userID and recipeID as the FK” We made the primary key the combination of them both so we don;t have 2 PKs, made this more clear on ERD and Outline

“Inconsistent use of quotation marks in the INSERT INTO statements:

- The values for integer columns (e.g., `servings`, `price`, `quantity`) should not be enclosed in quotation marks.
- The values for date columns (e.g., `dateAdded`) should be enclosed in single quotation marks.

- Inconsistent use of quotation marks and NULL value in the INSERT INTO statements for the `recipes` table:
  - The `restrictionID` column in the `recipes` table is nullable, so the value should be specified as NULL instead of `(SELECT id from dietaryRestrictions where name = 'Gluten-free')`. “

Updated use of quotation marks, NULL was already inserted correctly.

“There seems to be no normalization issues. However, to make it more clear, perhaps have repeated users in the "User Recipes" table, unless users can only have one recipe.”
Updated sample data

“Instead of using NULL to say no dietary restrictions consider using a number”
We decided to leave this as NULL just because we thought having an absence of a dietary restriction made sense vs making users pick something (even if it was just a number that meant no restriction)

“Some of the entities’ ID do not have a specific name such as recipeID, ingredientID”
Fixed entity names

“One thing that I noticed was that the entity names in the ERD and the outline are spaced and do not use camelCase as seen in the schema, which can be fixed quickly. “
Updated entity names in ERD

Peer Review Feedback and actions from Step 1

“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 for Proj 1

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. **Overview**

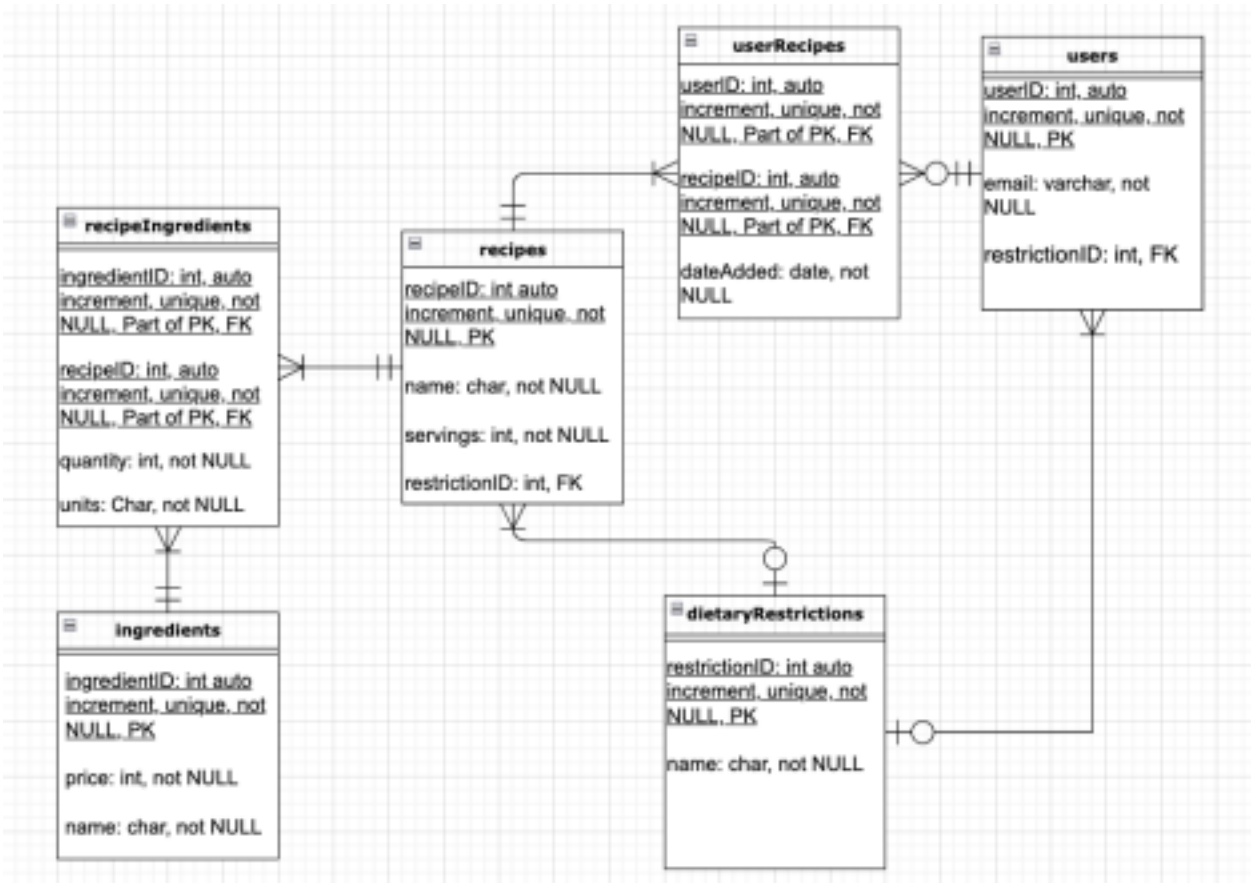
Our client is a company named "Dope Eats" which provides an online platform that connects thousands of food enthusiasts with a database of over 10,000 diverse recipes designed to accommodate a range of over 50 dietary restrictions. For this platform Dope Eats needs a relational database driven website 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.

Outline

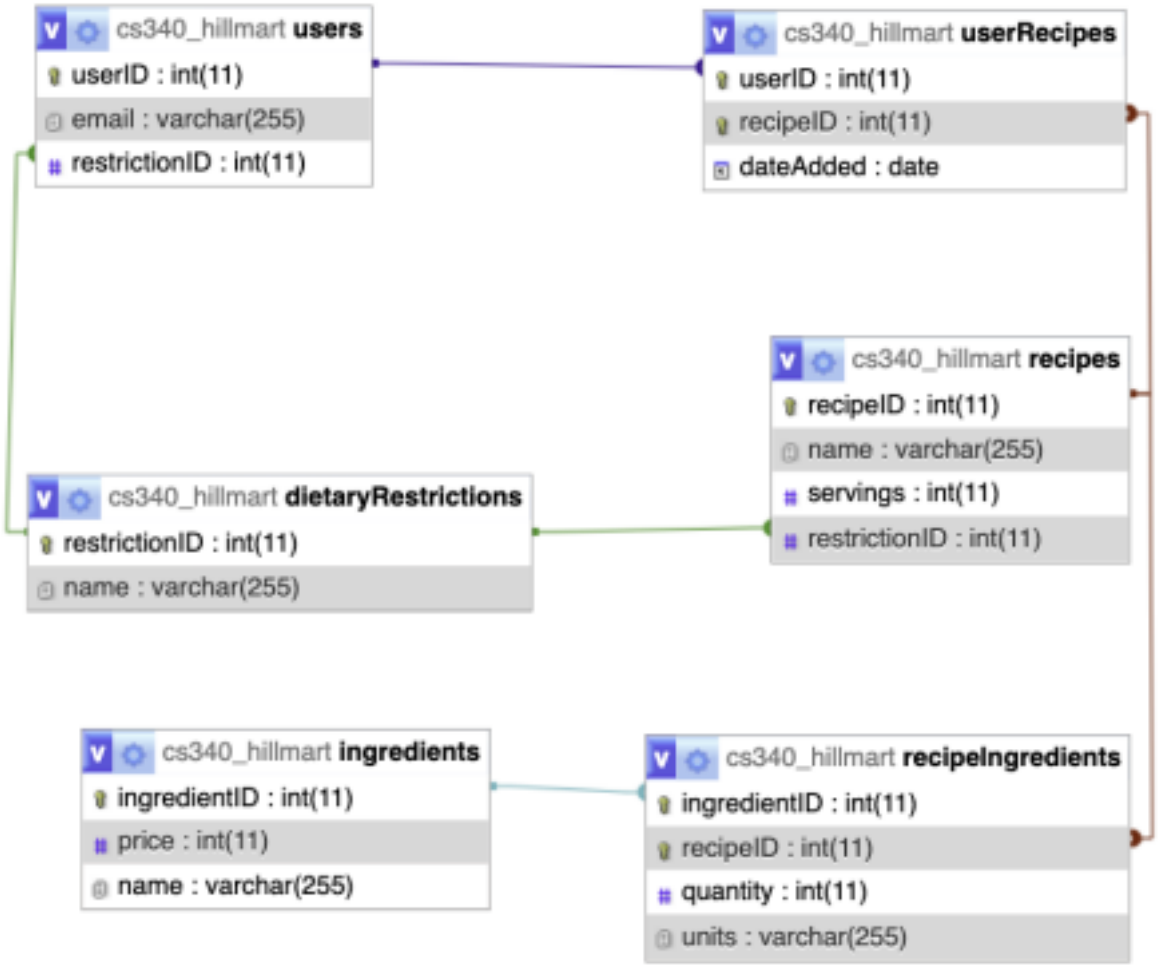
- users - database of users, storing their email, ID connects to recipes
    - userID: int, auto increment, unique, not NULL, PK
    - email: varchar, not NULL
    - restrictionID: int, FK
    - M:N relationship with Recipes
    - M:N relationship with Dietary Restrictions
- recipes - database of recipes, 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
- dietaryRestrictions - 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
- userRecipes - 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
- recipeIngredients - 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

ERD



Schema



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

userRecipes

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

dietaryRestrictions

restrictionID	name
1	Vegetarian
2	Gluten-free
3	Vegan

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

ingredients

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

recipeIngredients

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