abstract

There is an abundance of recipes and recipe blogs out there on the internet. For those who want to cook or bake at home, it can be difficult to find a recipe that suits their needs. They might even have to go through many websites to find what they need, which can waste time. Our system solves this problem by allowing the user to filter through an accumulation of all different kinds of recipes. A user can filter according to allergies, diet type, the country a recipe comes from, and ingredients.

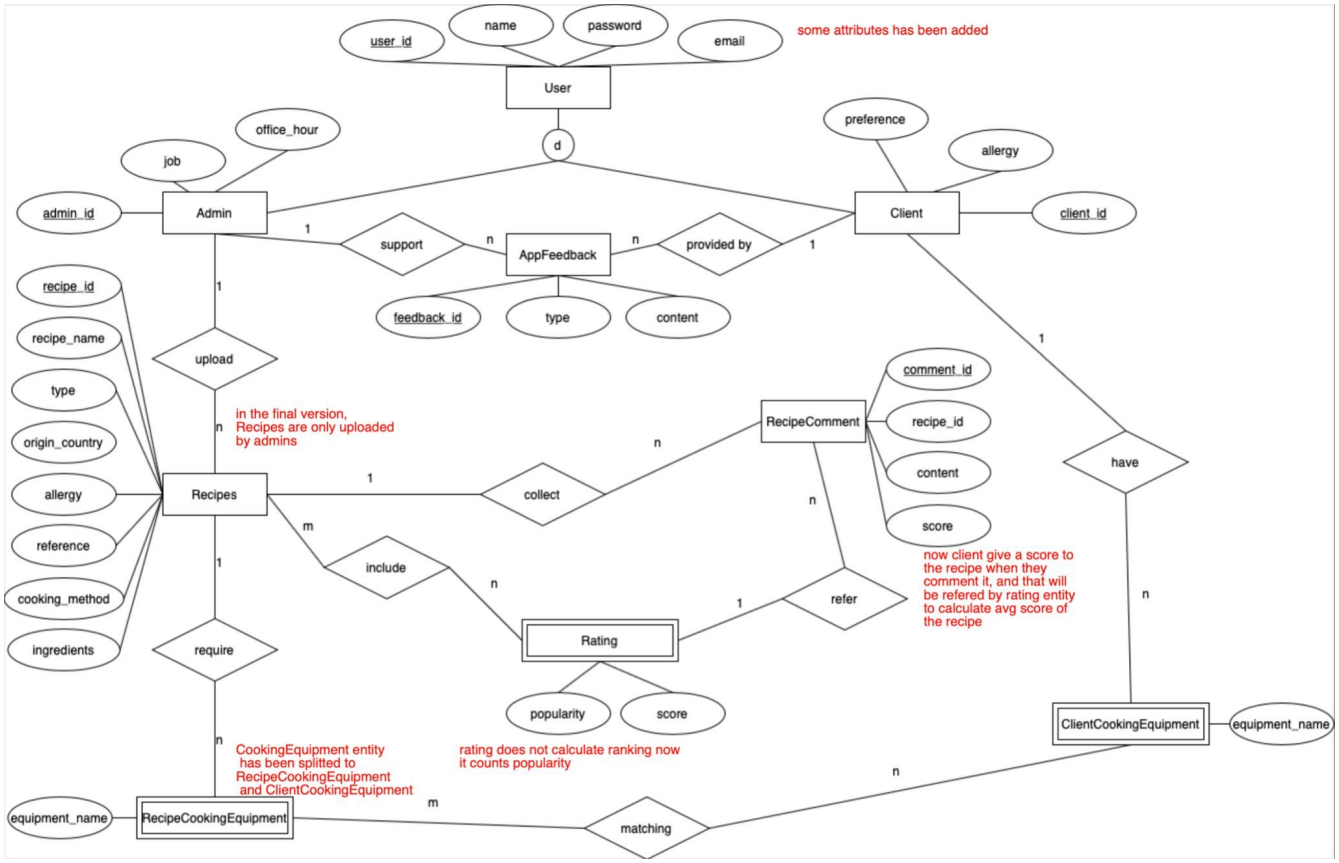
introduction

The database was designed to deal with finding recipes easily and based on a user's needs, so that they could find something to make without having to search through so many other sites. It also holds recipes of all kinds to accommodate people's different tastes.
An admin can add recipes, deal with feedback, and manage the website and information.
Clients have preferences and allergies, so when they are logged in, the site can automatically filter to show the preferred foods and exclude the ones that contain the allergy.
The client has cooking equipment and the recipes have equipment associated with them too so the site can show recipes that will be easier for the client to make.
The ratings allow us to find out which recipe is doing well, so as to recommend it.
The database stores reviews for the users that are to be displayed whenever.
There is also user information to allow for sign in. IDs determine who is an admin and who is a client.

system

The system that was designed to address the problem or task includes the website, api and database. The website has a homepage where the current highest rated recipe is shown. There is another page that shows all the recipes, and on that page the user can use the filtering function to pick options to include or exclude in the recipes. Clicking on a recipe sends the user to another page where it shows information about the recipe. For a certain recipe, the user can rate it and/or comment on the recipe. Of course, the api handles getting information of the recipes, and when the user rates or comments, the api inserts that information into the database.

project design



The different users of the system are those who are using the website to look for recipes and the admins who manage the website.
Changes from the api endpoints file:

- we have not added an endpoint for getting user preference
- we have not added an endpoint for setting user information
- we removed the endpoint for getting user information
- when we get ingredients, cooking method, origin country, and type, we also grab the recipe id
- when we get a rating we get all the columns of that table
- when we get review content we changed it from getting it from the comment id to recipe id, we also grab a score instead of just content
- we removed the endpoint for getting a review's time
- when we add a review, the input changed from only content and time to needing a comment id, recipe id, content and score
- when we grab the highest ranked recipe, it gives us an array of scores and recipe from highest score to lowest
- we removed the endpoint for getting the highest ranked recipe by preference, ingredient, and nutrition value, type, and keyword
- we removed the endpoint for getting a review by time
- when a user gives a score for a recipe, the input has changed from only using a score to using a recipe id, popularity score and the average rating score

User transactions:

- they can see the highest ranked recipe
- they see recipe names (this is a new endpoint)
- they see recipe ingredients
- they see cooking method information
- they see if a recipe will affect their allergies
- they see the country a recipe comes from
- they can see what type of category the recipe falls in
- they see the cooking equipment that will be helpful for a recipe
- they see recipe reviews
- they see the rating of recipe
- they can post a review
- they can make a rating on a recipe

Admin transactions

- they can see app feedback according to its ID and type (this is a new endpoint)
- they can register a new recipe
- after registering a new recipe, they can associate cooking equipment with it (this is a new endpoint)

implementation

The way we set the ingredients and cooking method content in the database is unusual, so the way it shows up on the website is not formatted properly. The way the cooking method is described is also not “proper” English and might be confusing. Also, some of the entities that we have created in the database are not used in the website.
We used Navicat and Mysql workbench for creating the database.
User transaction SQL statements :

- they can see the highest ranked recipe
  - ```
    SELECT MAX(score) as max_score, recipe_id FROM Rating GROUP BY recipe_id
    ORDER BY max_score DESC
    ```
- they see recipe names
  - all
    - ```
      SELECT recipe_id, recipe_name FROM Recipes
      ```
  - one
    - ```
      SELECT recipe_name FROM Recipes WHERE recipe_id=?
      ```
- they see recipe ingredients
  - all
    - ```
      SELECT recipe_id, content FROM Ingredients
      ```
  - one
    - ```
      SELECT recipe_id, content FROM Ingredients WHERE recipe_id = ?
      ```
- they see cooking method information
  - ```
    SELECT recipe_id, content FROM CookingMethod WHERE recipe_id=?
    ```
- they see if a recipe will affect their allergies
  - all

        - ```
          SELECT recipe_id, allergy FROM Recipes
          ```
    - one
        - ```
          SELECT recipe_id, allergy FROM Recipes WHERE recipe_id = ?
          ```
- they see the country a recipe comes from
    - all
        - ```
          SELECT recipe_id, origin_country FROM Recipes
          ```
    - one
        - ```
          SELECT recipe_id, origin_country FROM Recipes WHERE recipe_id = ?
          ```
- they can see what type of category the recipe falls in
    - all
        - ```
          SELECT recipe_id, type FROM Recipes
          ```
    - one
        - ```
          SELECT recipe_id, type FROM Recipes WHERE recipe_id=?
          ```
- they see the cooking equipment that will be helpful for a recipe
    - ```
      SELECT equipment_name FROM RecipeCookingEquipment WHERE recipe_id= ?
      ```
- they see recipe reviews
    - ```
      SELECT content, score FROM RecipeComment WHERE recipe_id=?
      ```
- they see the rating of recipe
    - ```
      SELECT * FROM Rating WHERE recipe_id=?
      ```
- they can post a review
    - ```
      INSERT INTO RecipeComment values(?, ?, ?, ?)
      ```
- they can make a rating on a recipe
    - ```
      INSERT INTO Rating values(?, ?, ?)
      ```

Admin transactions

- they can see app feedback according to its ID and type
    - all
        - ```
          SELECT feedback_id, content, type FROM AppFeedback
          ```
    - by ID
        - ```
          SELECT feedback_id, content, type FROM AppFeedback WHERE
          feedback_id = ?
          ```
    - by type
        - ```
          SELECT feedback_id, content, type FROM AppFeedback WHERE type = ?
          ```
- they can register a new recipe
    - recipe
        - ```
          INSERT INTO Recipes values(?, ?, ?, ?, ?)
          ```
    - cooking method
        - ```
          INSERT INTO CookingMethod values(?, ?)
          ```
    - ingredient
        - ```
          INSERT INTO Ingredients values(?, ?)
          ```
- after registering a new recipe, they can associate cooking equipment with it
    - ```
      INSERT INTO RecipeCookingEquipment values(?, ?)
      ```

-

api documentation

https://documenter.getpostman.com/view/18848626/UVRAJ7H7

quick user guide

To start the backend:
With the terminal open, go to the root folder, cd into the server folder, type “node index.js” and hit enter.
To start the front end:
With the terminal open go to the root folder, cd into the client folder, type “npm start” and hit enter

Starting on the homepage, users will see a navigation bar and the highest rating recipe. Clicking on the recipe will lead users to the recipe page, where they will see how to cook or bake the food.

On the navigation bar, there are links that lead to the home page, and the page where all the recipes are shown.

The recipes page has a bar where users can pick some options. Picking an allergy means to exclude it, and the rest of the options means to include them. When users are done picking options, they can press the “Find recipes” button to filter. Here, filtering will be correct when choosing in the order of allergies first, then type, country and then ingredients.

On a specific recipe, rating is done by hitting one of the five numbers. Commenting is accomplished by filling in the input box and pressing the “Comment” button. Ratings are set to be in integer form. Currently, only after refreshing the page will the updated content be shown.

references

- The information from here https://www.tutorialspoint.com/reactjs/reactjs_pagination.htm was used to create the Pager file (client/component/Pager.js) to accomplish pagination.
- https://www.w3schools.com/Css/css_navbar.asp was used to design the navigation bar
- There was some inspiration taken from https://stackoverflow.com/questions/3998780/general-is-there-a-way-to-calculate-an-average-based-off-of-an-existing-averag to calculate the new average rating after a user rated.
- To design a nice looking button we used https://www.w3schools.com/csS/css3_buttons.asp.
- The header image was borrowed from here https://dumplingschool.com/wp-content/uploads/2020/03/shutterstock_1501560017-2.jpg
- The websites we used for the recipes
    - https://www.maangchi.com/recipe/haemuljeon
    - https://www.spendwithpennies.com/the-best-chili-recipe/
    - https://www.justonecookbook.com/curry-bread/

- https://www.tasteofhome.com/recipes/best-lasagna/
- https://pokethedough.com/2019/11/03/vietnamese-fried-rice-cakes-banh-bot-chien/
- https://thewoksoflife.com/turnip-cake-lo-bak-go/
- https://www.recipetineats.com/chicken-pad-thai/
- https://www.thehongkongcookery.com/2020/05/chinese-hot-and-sour-soup.html
- https://www.chinasichuanfood.com/winter-melon-soup/
- https://www.vickypham.com/blog/vietnamese-kabocha-squash-soup-chicken-canh-bi-do-nau-ga