Final Report - Submission 5, Adding a Web Server

There is body validation in the search function, but none of the other pages of the site - none of the other pages are dealing with the body of the request. The search function just checks to see if the request is formatted for a string, so we never use JSON schema validation to check the body of the request.

User authentication is required for some parts of the website, but not all. Anyone is free to look at recipes and browse, as well as look at instamode. Favoriting and adding to the calendar (from the recipe page), looking at the profile, calendar, and favorites pages all require user authentication.

HTTP Requests / Routes

Browse Feed:
PUT /feed { restrictions }
Replacement of the getFeedData function. Does not require any user authentication and does not modify the database. Filters the recipes shown by the restrictions inputted.

Recipe Page:
Retrieves a recipe:
GET /recipe/:recipeid
Replacement of the getRecipe function. Does not require any user authentication and does not modify the database. It retrieves a recipe based on :recipeid.

Favoriting from a recipe page:
PUT /recipe/:recipeid/favorites/user/:userid
Replacement of the addFavorite function. Requires user authentication - you must be user 1 to complete this request. Modifies the user/:userid by adding to the user's list of favorites.

Unfavoriting from the recipe page:
DELETE /recipe/:recipeid/favorites/user/:userid
Replacement of the removeFavorite function. Requires user authentication - you must be user 1 to complete this request. Modifies the user/:userid by removing a recipe from the user's list of favorites.

Checking to see if a recipe is in the user's favorites:
GET /recipe/:recipeid/favorites/check/user/:userid
Replacement of the checkUserFavorites function. Requires user authentication - you must be user 1 to complete this request. Does not modify anything in the database; just returns a boolean for the current recipe (if it is in the user's list of favorites, return true, else, return false).

Adding something to calendar from recipe page:
PUT /recipe/:recipeid/user/:userid/calendar/:dayid
Replacement of the addRecipeToCalendar function. Requires user authentication - you must be user 1 to complete this request. Replaces whatever the dinner was in week 2 with the recipe and the day the user has specified. Modifies calendar/:dayid (at week 2.)

Search Page:
Searches for Recipe Items with the given search text:
POST /results [searchText]
Replaces the mock server method findRecipe, and does not require user authorization. Does not modify the database; returns the search results in a new “search results” item.

Favorites Page:
Retrieving recipes using their ids from a favorites page:
GET /user/:userid/favorites/
This method replaces the findRecipesFromId method in the mock server.
It retrieves the favorites’ list based on the :userid and does not modify anything in the database.
A user with the specified :userid can issue this request (hardcoded user id for Kuk is 1.)

Instamode Page:
Retrieve recipes from the search by ingredients:
POST /instaresults
Replacement of the findRecipeByIngredients function from the mock server.

Profile Page:
Retrieve a user’s profile data:
GET /user/:userid

Retrieve a user’s restriction ids:
GET /user/:userid/restrictions

Add a restriction to the user’s restrictions:
PUT /user/:userid/restriction/:restrictionid

Remove a restriction from the user’s restrictions:
DELETE /user/:userid/restriction/:restrictionid

Calendar:
Display the calendar for the respective week:
GET user/:userid/calendar/:week
Requires user authentication; does not modify the database.

Remove a recipe from a week:

DELETE user/:userid/calendar/:week/:day/:meal
Requires user authentication; removes a recipe from the user's calendar (in the database. Does not remove the overall recipe from the database, just from that user's calendar object.)

Reset Database:
Resets the database to its original state:
POST /resetdb
Does not require user authentication.

Special Server Setup Procedure

Our server does not require additional setup besides npm install and node src/server.js.

Individual Contributions

Recipe Page (Emma):

- Last workshop fixes:
    - Favoriting from a recipe page now works perfectly, with the recipe displaying the proper status on first loading of the recipe page (if the user has the recipe in their favorites list and is logged in, the "x" option appears in the recipe page - you can remove the recipe. If the recipe is not in the user's list of favorites the button appears as a heart.)

- Product Features:
    - Implemented all the HTTP routes used for the recipe page (listed above)
    - Favoriting works with the real server and database
    - Adding a recipe to the calendar works, uses the real server and real database
    - Loading the general recipe information works with the real server and database (name, ingredients, instruction list, etc)
    - Authentication is required to add something to favorites or the calendar, but not to view the recipe in general. This is because the recipe overall is public, but you must be logged in to add to your favorites or calendar.

Search (Stephanie):

- Last workshop fixes:
    - Search now works perfectly, without the pesky infinite loops from setting the state within componentDidUpdate in ResultsFeed
    - Search text now stays in the search bar
    - The average rating displayed is now accurate

- Product features:
    - HTTP Route: POST /results [searchText]
    - Searches for Recipe Items whose names match the given search text:
    - The client side search server method uses a specialized XMLHttpRequest instead of the sendXHR method, since our search does not need user authentication - the results of the search are not limited to individual users.
    - However, need first to verify that the body of the request (i.e. the search text) is a string before it can find the recipe items

Browse & Filter (Farida):

- Last workshop fixes:
    - Filtering now works smoothly! I refactored my getFeedData method so it takes in a set of restrictions every time the feed refreshes and populates the feed with the appropriate recipes.
    - Tied the value of checked on the input box to a state variable, so it will properly work with React and would not be overwritten

- Product Features:
    - HTTP Route: PUT /feed/ { restrictions }, where restrictions is an array of checked dietary restrictions
    - Displays the feed of recipes, and filters out recipes when a dietary restriction is checked
    - I actually tried to use a GET request in the beginning, since all I do is retrieve recipes from the database, but because I needed to send the restrictions in the request body it wasn't working. So I moved over to a PUT request instead, so I could continually update the feed removing the recipes that are filtered out.

Calendar (Syeda Zainab):

- Last workshop fixes:
    - The buttons now work perfectly and change to the right week in the state. This was done by re-writing the refresh method so that it takes in a week and loading the subsequent data.
    - Improved the database design so that each user has a calendarId and one calendar basically represents a month with four weeks in it.

- Product Features:
    - HTTP Route: GET /user/:userid/calendar/:week
    - Displays the recipes for the current week
    - Using the buttons Previous Week and Next Week, you can also toggle between different weeks
    - HTTP Route: DELETE user/:userid/calendar/:week/:day/:meal
    - When the deletion glyphicon in the top right corner is clicked, it deletes the recipe from the particular week.

Favorites (Karen):

- Last workshop fixes:
    - Two sorts have been implemented on the Favorites page. The default is the order in which the items are added to the favorites' list, and the additional sort is the reverse order in which the recipes were added.

- Product Features:
    - HTTP Route: GET /user/:userid/
    - Retrieving all the favorited recipes of a user works perfectly
    - Favoriting and unfavoriting from the favorites page works perfectly
    - Authentication is added to view the Favorites page as well as to favorite/unfavorite recipes from the page because the favorites' list is unique to the user

Profile Page (Eleanor):

- Last workshop fixes:
    - Adjusted the profile restrictions so that the checkboxes refer to an array of boolean objects in the state rather than being checked in the server method callback function.

- Product Features:
    - HTTP Routes:
        - GET /user/:userid (body: undefined) → replaces getProfileData() functionality from the client. It returns an object that holds resolved references to all the user's data, excepting their list of dietary restrictions, which is left unresolved. Uses user authentication: if the user is not requesting their own profile data, they are unauthorized.
        - GET /user/:userid/restrictions (body: undefined) → replaces getUserRestrictions() client functionality. It returns an array holding the string tags of the user's dietary restrictions. Uses user authentication: if the user is not requesting their own dietary restrictions, they are unauthorized.
        - PUT /user/:userid/restriction/:restrictionid (body: undefined) → replaces the addUserRestriction() client functionality. It adds a restriction to the user's restrictions in the db and returns the updated restrictions list (unresolved). Uses user authentication: if the user is not adding a restriction to their own dietary restrictions, they are unauthorized.
        - DELETE /user/:userid/restriction/:restrictionid (body: undefined) → replaces removeUserRestriction() client functionality. It removes a restriction from the user's restrictions in the db and returns the updated restrictions list (unresolved). Uses user authentication: if the user is not

removing the restriction from their own dietary restrictions, they are unauthorized.

Instamode Page (Jennifer):

- Last workshop Fixes:
    - Clear the search properly deletes all ingredients
    - Added an ingredients-only checkbox for when users want to search for recipes containing only certain ingredients
- Product features:
    - HTTP Routes:
        - POST /instaresults replaces findRecipeByIngredient in the client server. It will return a list of recipes that contain at least one of the ingredients the user lists.
        - POST /instaresults/ingredientsONLY is a new server method that will collect a list of recipes that contain only the listed ingredients the user has entered.
    - Since authentication is not required for users to enter in ingredients to find recipes, I'm simply checking to see that the list of ingredients sent over was a string
    - Clear Search will refresh the page, erasing previous data stored in the react components regarding ingredients - was experience some issues with communication by coding the ingredients list to be empty, so I decided to do a refresh of the page instead.

Lingering Bugs / Issues / Dropped Features

Favorites (Karen):

- The Favorites page does not have the filter bar anymore after analysis of user's use of the Favorites page. It should contain a collection of recipes so a sorting mechanism is preferred. The sort button was implemented instead.

Recipe Page (Emma):

- Currently, the calendar button in the recipe only allows users to choose the day of the week, rather than the week, day, and meal. I will update the UI and server methods later to support more user choice. (Right now, in the server recipes are automatically added to week 2 dinner, with users choosing only the day of the week.)

Calendar Page (Syeda Zainab):

- The only problem that remains with the Calendar is that even though the right recipe is deleted from the calendar, once the page refreshes, the remaining recipes shift to the

left. This will be fixed by using a placeholder when a recipe is deleted so that when React renders again after deletion, it ignores the placeholder.