Abiel’s Meta U intern project

Overview:

What do you want to build?
Recipe sharing and discovery website, where users can create / share their own recipes.

Key Features:

- Recipe browser with simple search and category/type filtering.
- Recipe editor, with image upload capabilities and table data entry system for the ingredient list.
- API integration with a messaging platform in order to send the ingredient list and/or recipe instructions to a mobile phone.
- Recipe rating system, where users can rate recipes.
- Recipe search that shows you which recipes you can cook based on the ingredients that you have.
- Basic user creation / authentication system

Stretch Goal:

- Simple comment system, where users can comment on recipes with text or images.
- Recommendation system that can recommend recipes to users based on how they rated other recipes
- User action logging
- Unit/integration testing for the backend
- Video upload support for recipes

Timeline:

- Week 4
    - Add mock / real data to DB for backend testing
    - Create basic backend, which supports getting recipes from DB
    - Scrape recipe data from an already existing DB for better demos
    - Figma website wire-framing
- Week 5
    - Start work on basic components for frontend (Recipe card, Ingredients List card, etc.)
    - Create basic recipe grid view with data from backend
    - Add user support
- Week 6
    - Start work on the recipe editor
    - Implement recipe creation on the backend
    - Add basic query functionality
    - Start implementing messaging API functionality
- Week 7
    - Finish messaging API functionality

    - Add recipe rating system
- Week 8
    - Implement visual polish / transition animations
- Week 9
- Week 10: demo

add favorite recipes
post a recipe to socials thing
cooksnaps
    people who made recipes
react with different emojis

MVP Feature Table

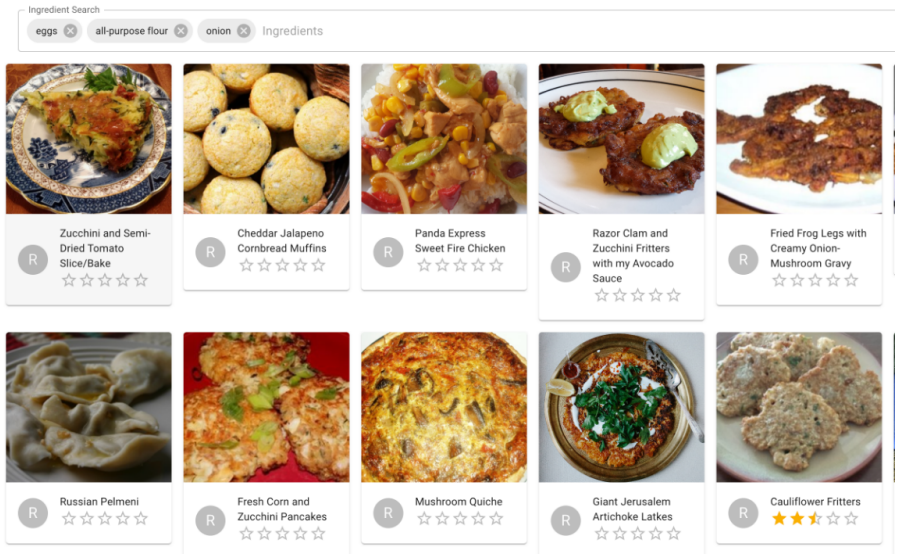
Feature	Status	PR	Workplace Link	**Comments**
MVP Features				
Search Bar	Finished	https://github.co		
Add search bar with $text query	☑			
Add typeahead support	☑			
Switch to elasticsearch (?)	☐			Optional
Rating System	Finished	https://github.co		
Implement rate feature in front end	☑			
Messaging API	Finished			
Add send message button to front end	☑			
Add user prompt for phone number during signup	☑			
Improve messaging button looks	☑			
Profile Page	In Progress			??? Is this needed?
Implement profile user profile page	☐			
Show created recipes	☑			
Show posted comments (?)	☐	Are these neede		
Allow setting a new password	☐			
Allow setting a new phone number	☐			
Comment system	Finished	https://github.co		
Add comment POST route	☑			
Add comment GET route	☑			
Create comment viewer for frontend	☑			
Add comment posting functionality	☑			
Navbar Re-design	Finished			
Move search bar to navbar	☑			
Add relevant routes to user icon (Log Out, Profile View)	☑			TODO Actual routes needs to be added into backend
Refactor editor button into create new recipe button	☑			
Recipe Grid re-design	Finished	https://github.co		
Fix layout on large screens	☑			
Improve grid functionality (make it look more even)	☑			
Add sort by (rating, date, views) functionality	☑			
Add load `n` recipes drop down (?)	☐			
Website Polish	Finished			
Do not store recipes without valid images	☑			
Add website footer	☐			Is this even necessary
Add icon or name to navbar	☑			
Add Hero banner (?)	☑			
Fix favicon and website name	☑			
Fix user input forms in order to show errors when an invalid input is given	☑			
Implement loading states when waiting on the server	☑			
Favorite Recipies	Finished	https://github.co		
Add favorite recipies system where a user can like recipies to save them	☑			

Add page where users can view their favorited recipies	☑			
Stretch Features				
Show recipes similar to...	Finished	https://github.co		
Setup elasticsearch with mongodb	☑			
Implement elasticsearch similarity searching based on recipe title / body	☑			Needs to be tested, might work well might not.
User action logging	Finished			
Log user actions to google analytics				

Stretch feature documentation

SEARCH BY INGREDIENT

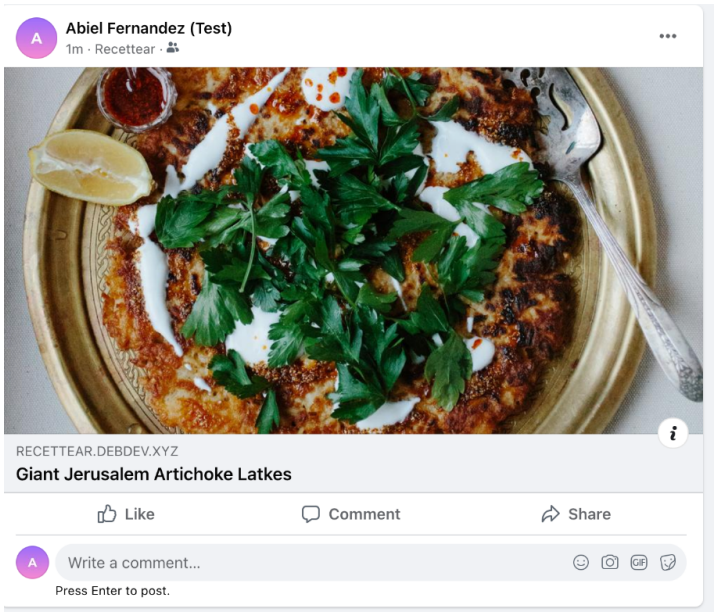
To be able to search for recipes with certain ingredients, first the ingredient data had to be converted into cleaner ingredient "keywords" which could be used for searching. The recipe dataset contains un-formatted ingredients with units and quantity in the same string, for example: "1 cup potatoes, peeled and grated". To normalize the data into ingredients that could be easily searchable, I found a database which listed 3000 of the most common ingredients used in recipes. With that a search can be done for each ingredient, obtaining the ingredient with the longest character length inside the original ingredient string (code). This makes it so that the previously unorganized ingredient strings can be turned into ingredient keywords. Running this algorithm on the previous example, the detected keyword would be potato. By running this algorithm on the recipe before posting it to the DB, and then posting the ingredient keywords to the DB this expensive operation only needs to be done once. Afterwards, a variety of searches can be done with this information, in this case a MongoDB aggregation obtains the matching ingredients between an array of target ingredient keywords and each recipe. Then, the data is sorted based on the number of matches, giving the user a list of recipes that contain the given ingredients (code).



SHARE WITH FACEBOOK

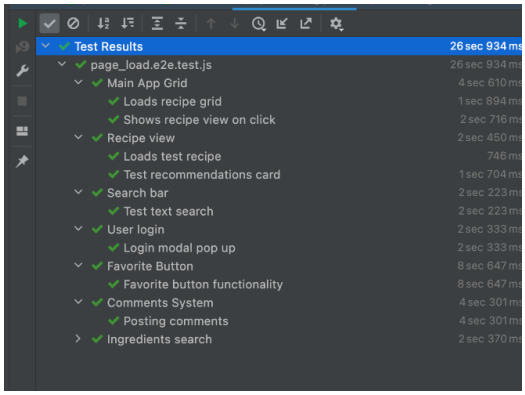
In order to be compatible with the Facebook scraper, my website had to have OpenGraph tags with the recipe title and image so that the preview could correctly show up. For this, I had to make my website available under a domain name, this way it could be accessed from the internet. Since the facebook scraper does not run javascript while getting the data, an external api was used, prerender.io. This is used so that all scraping calls made to the website go through the api first, which fetches data from my website but it "prerenders" and runs javascript before returning the website to the scraper. That way I can use react

helmet, a library which handles setting `meta` tags with react to dynamically set the OpenGraph meta tags when the recipe information was fetched (code). Afterwards, I added the `Share with Facebook` button utilizing the `Facebook JS SDK` (code) . This feature also makes it so that my website can be scraped by other social media networks or search engines.



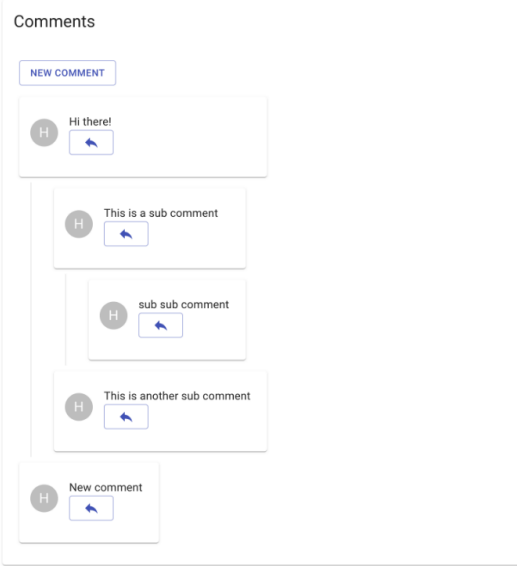
END TO END TESTING

End to end testing was implemented using Puppeteer and Jest (code). All of the basic functionality in my website is automatically tested. With this, I can make sure that one change does not break other previously implemented features.



COMMENT SYSTEM

Support for commenting on recipes was added, with support for nested comments. For this, a recursive algorithm was developed in order to retrieve the comment tree from the database so that nested comments could be displayed correctly. As comments had to be shown in the proper order. Comments are iterated over until no more comments need to be added to the tree (code). With this, the frontend is able to display comments in their proper hierarchical order.



Architecture

BACKEND ROUTES

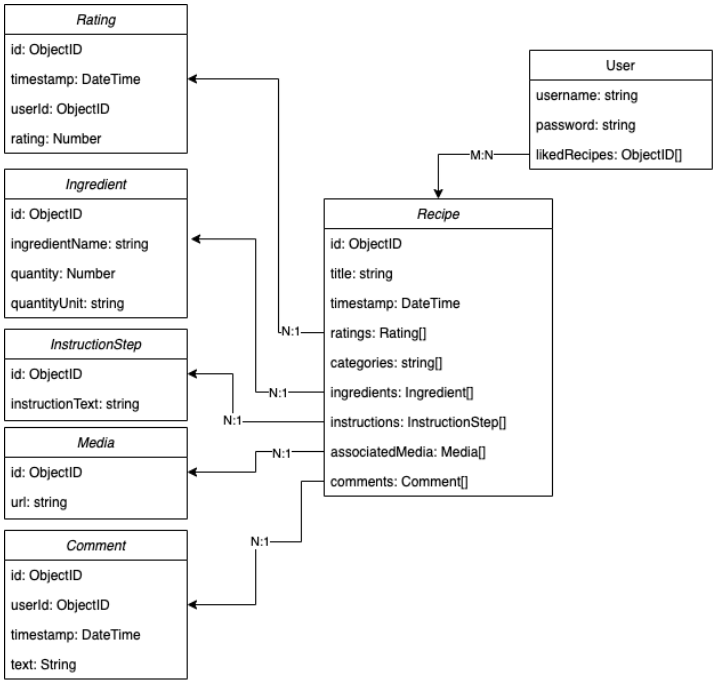
GET /recipes

Parameters:
offset: int → Offset of recipes to load, default 0
limit: int → Number of recipes to load, default 10
Query Filters:
title: string → Filter recipes by name
Returns:
Array of n Recipe objects

GET /recipes/:id

Returns:
Recipe object corresponding to ID

DB ARCHITECTURE


Rating
id: ObjectID
timestamp: DateTime
userId: ObjectID
rating: Number
User
username: string
password: string
likedRecipes: ObjectID[]
M:N
Ingredient
id: ObjectID
ingredientName: string
quantity: Number
quantityUnit: string
Recipe
id: ObjectID
title: string
timestamp: DateTime
ratings: Rating[]
categories: string[]
ingredients: Ingredient[]
instructions: InstructionStep[]
associatedMedia: Media[]
comments: Comment[]
N:1
InstructionStep
id: ObjectID
instructionText: string
N:1
N:1
Media
id: ObjectID
url: string
N:1
Comment
id: ObjectID
userId: ObjectID
timestamp: DateTime
text: String
N:1