Graze

Making food, easier.

Michael Barlow, Greg Benton, Rasheeq Jahan, Sofia Mehrotra, Erin Ruby

University of Colorado

Table of contents

Introduction

Ideology and Goal

Often the hardest part of meal time is asking ”what should I make?”, to eliminate this familiar culinary woe we wanted,

- A clean, simple, efficient user interface
- To take in ingredients or main ideas
- To output a straightforward, one by one, list of possible recipes, eliminating the need to decide from a massive list of options

Requirements

We made a list of requirements to make sure we met our goals of user-focused simplicity, including,

- No user accounts or sign up necessary
- Easy to restart or refresh user progress
- Output user results in a simple and digestible format (i.e. one at a time)

Methodology

We used the Agile methodology, slightly adapted to fit into the time constraints of the course.

This meant that (starting at around the third week) we had a scrum nearly every day that class met.

Sprints were formed on a shortened cycle, with a typical sprint lasting between 3-7 days depending on the scope of our goals

Sprints

1. Initial Setup
   - Choose tools (Docker, Django, etc)
   - Get everyone a development environment
   - Familiarize ourselves with chosen tools
2. Database & User Input
   - Use our API to build a starting database
   - Build a skeleton front end that accepted user input
3. Basic Search Functionality & Initial Site Layout
   - Start building control flow to create final layout of site
   - Get basic searches working (input one ingredient get back one recipe) to prove we are fully connected
   - Begin restructuring welcome page to be more user-friendly

Sprints (cont.)

4. Full Search Function & UI redesign
    - Get the full (basic) search function working (multiple inputs, multiple results)
    - Continue on UI polishing to make a clean and enticing user interface
    - Research more robust methods of user input (tags/autocomplete)
5. Backend Layout Updates, Results Page Design, & Dynamic Database Loading
    - Restructure the Django side layout of the project to increase containerization and ensure we were following best practices
    - Create a results page with UI beyond raw HTML
    - Create a method so that a user search with no matches in our database made an API call and stored the results.

Sprints (cont.)

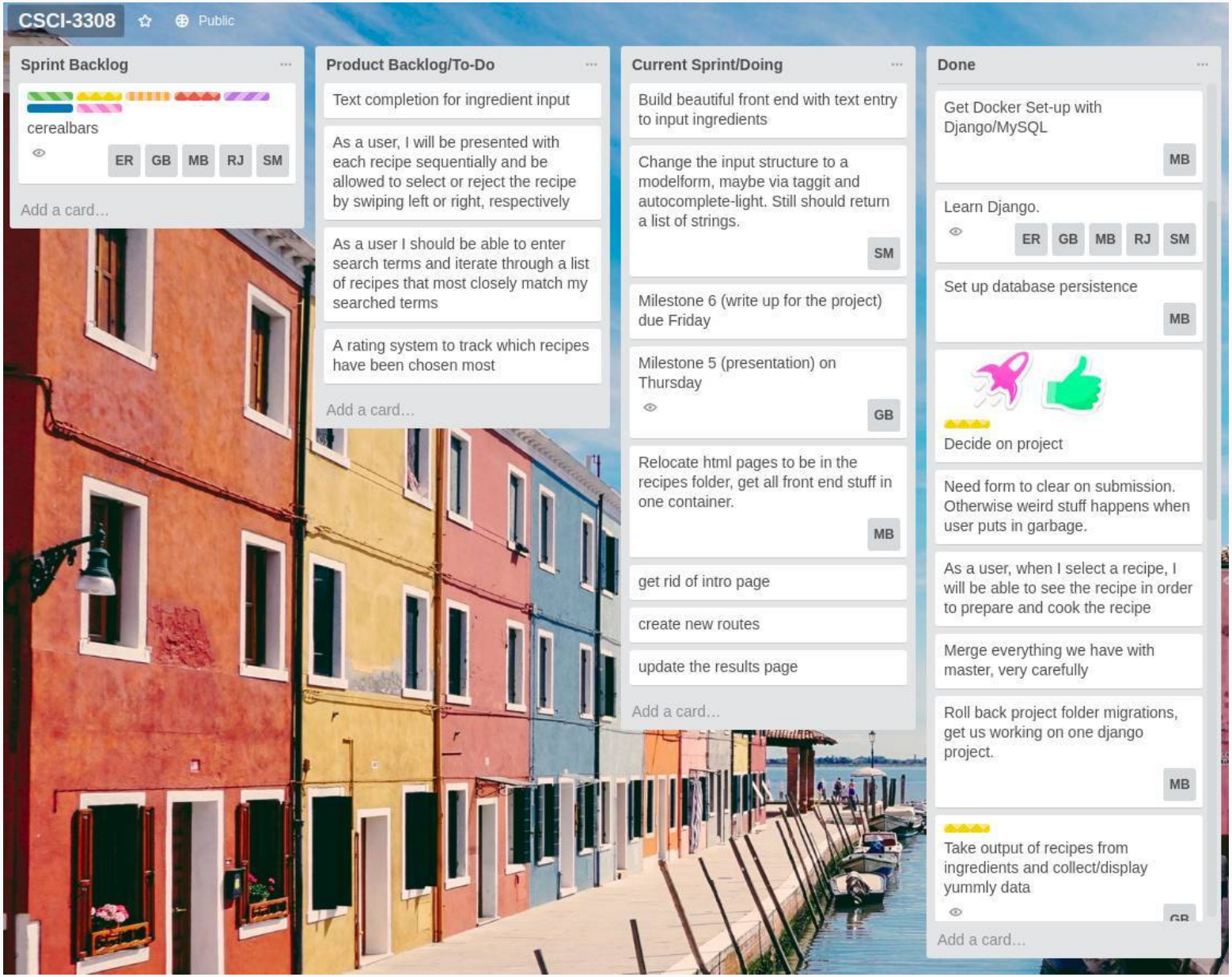
6. Finalize the input and output pages, search algorithm, & test cases and final deployment
    - Make a cohesive appearance for the website across pages
    - Ensure that the core functionality holds up to all test cases
    - Deploy the product to a public Heroku site

Tools

We used the following resources to build our website,

- Github - VCS
- Trello - Virtual project board
- Django - Main development platform and service
- MySQL - Database software
- Docker - Containerization service to allow everyone to develop with the same services
- Heroku - Platform for final web hosting
- Selenium - Python testing platform

Trello


CSCI-3308
Public
Sprint Backlog
cerealbars
ER GB MB RJ SM
Add a card…
Product Backlog/To-Do
Text completion for ingredient input
As a user, I will be presented with each recipe sequentially and be allowed to select or reject the recipe by swiping left or right, respectively
As a user I should be able to enter search terms and iterate through a list of recipes that most closely match my searched terms
A rating system to track which recipes have been chosen most
Add a card…
Current Sprint/Doing
Build beautiful front end with text entry to input ingredients
Change the input structure to a modelform, maybe via taggit and autocomplete-light. Still should return a list of strings.
SM
Milestone 6 (write up for the project) due Friday
Milestone 5 (presentation) on Thursday
GB
Relocate html pages to be in the recipes folder, get all front end stuff in one container.
MB
get rid of intro page
create new routes
update the results page
Add a card…
Done
Get Docker Set-up with Django/MySQL
MB
Learn Django.
ER GB MB RJ SM
Set up database persistence
MB
Decide on project
Need form to clear on submission. Otherwise weird stuff happens when user puts in garbage.
As a user, when I select a recipe, I will be able to see the recipe in order to prepare and cook the recipe
Merge everything we have with master, very carefully
Roll back project folder migrations, get us working on one django project.
MB
Take output of recipes from ingredients and collect/display yummly data
Add a card…

Github

t

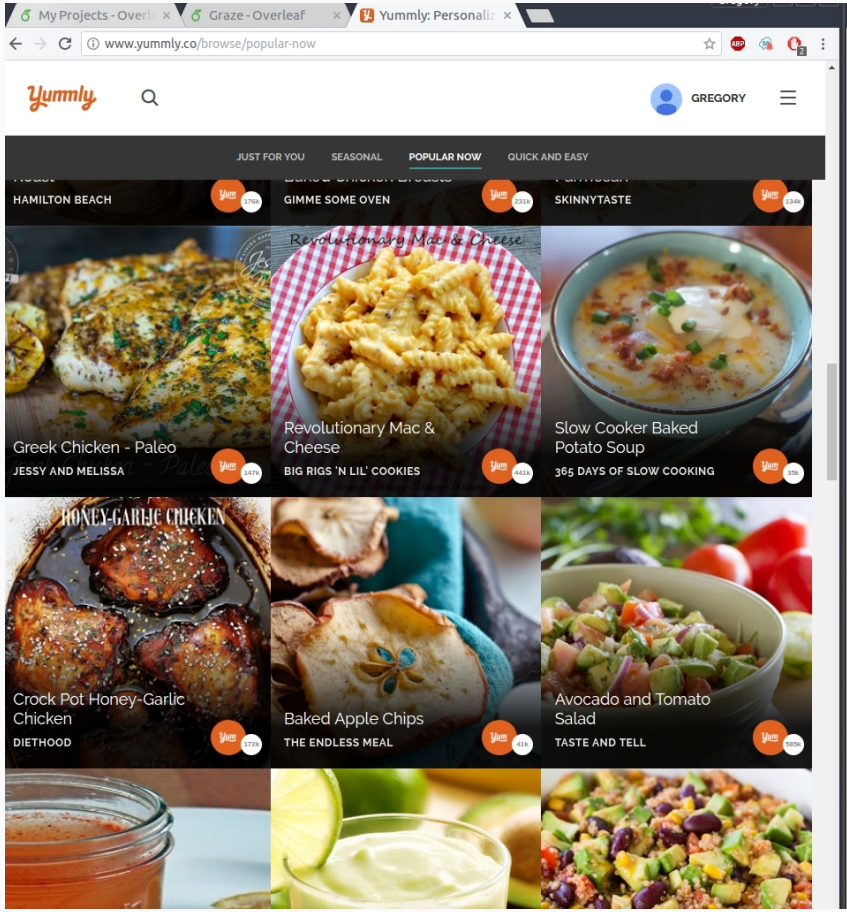
73 commits | **5** branches | **0** releases | **5** contributors

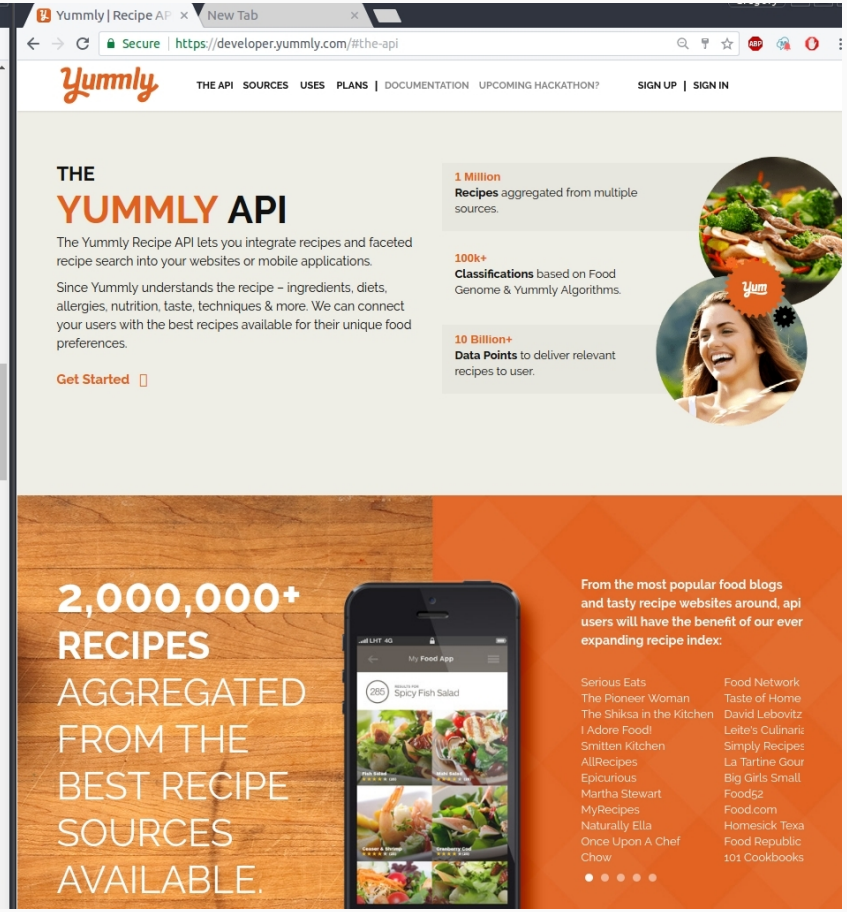
Branch: **master** | **New pull request** | **Create new file** | **Upload files** | **Find file** | **Clone or download**

g-benton committed on **GitHub** Update README.md — Latest commit `1592efa` just now

app	changed a few lines to work with updated models	2 hours ago
database-models	reformatted the database to play well with django and included the pr…	13 days ago
documents	added js plugin file to directory for scrolling effect:	15 hours ago
populate-db-yummly	added search function to get recipes based on user inputs	8 days ago
recipe_db_hard_backups	Add files via upload	8 days ago
.gitignore	fixing .pyc git issues	9 days ago
Dockerfile	added data persistence	16 days ago
README.md	Update README.md	just now
docker-compose.yml	added data persistence	16 days ago
env_web	added data persistence	16 days ago

Yummly


My Projects - Overl
Graze - Overleaf
Yummly: Personaliz
www.yummly.co/browse/popular-now
Yummly
GREGORY
JUST FOR YOU
SEASONAL
POPULAR NOW
QUICK AND EASY
HAMILTON BEACH
GIMME SOME OVEN
SKINNYTASTE
Greek Chicken - Paleo
JESSY AND MELISSA
Revolutionary Mac & Cheese
BIG RIGS 'N LIL' COOKIES
Slow Cooker Baked Potato Soup
365 DAYS OF SLOW COOKING
HONEY-GARLIC CHICKEN
Crock Pot Honey-Garlic Chicken
DIETHOOD
Baked Apple Chips
THE ENDLESS MEAL
Avocado and Tomato Salad
TASTE AND TELL


Yummly | Recipe AP
New Tab
Secure
https://developer.yummly.com/#the-api
Yummly
THE API
SOURCES
USES
PLANS
DOCUMENTATION
UPCOMING HACKATHON?
SIGN UP | SIGN IN
THE
YUMMLY API
The Yummly Recipe API lets you integrate recipes and faceted recipe search into your websites or mobile applications.
Since Yummly understands the recipe – ingredients, diets, allergies, nutrition, taste, techniques & more. We can connect your users with the best recipes available for their unique food preferences.
Get Started
1 Million
Recipes aggregated from multiple sources.
100k+
Classifications based on Food Genome & Yummly Algorithms.
10 Billion+
Data Points to deliver relevant recipes to user.
2,000,000+ RECIPES AGGREGATED FROM THE BEST RECIPE SOURCES AVAILABLE.
From the most popular food blogs and tasty recipe websites around, api users will have the benefit of our ever expanding recipe index:
Serious Eats
The Pioneer Woman
The Shiksa in the Kitchen
I Adore Food!
Smitten Kitchen
AllRecipes
Epicurious
Martha Stewart
MyRecipes
Naturally Ella
Once Upon A Chef
Chow
Food Network
Taste of Home
David Lebovitz
Food52
Food.com
101 Cookbooks

Testing

Have three classes of inputs

- ingredients in the database
- valid ingredients *not* in the database but could be found
- garbage that will never be found (random inputs)

Wrote python scripts using Selenium and ChromeDriver to insert many of these combinations in attempts to raise errors to squash any bugs that may have been in our code

Errors Found:

- Some ingredients contain the "™" symbol, which cannot be entered as a search term - fixed by matching substrings
- Some recipes don't have images associated with them, caused a reference error later on - fixed by changing the search parameters

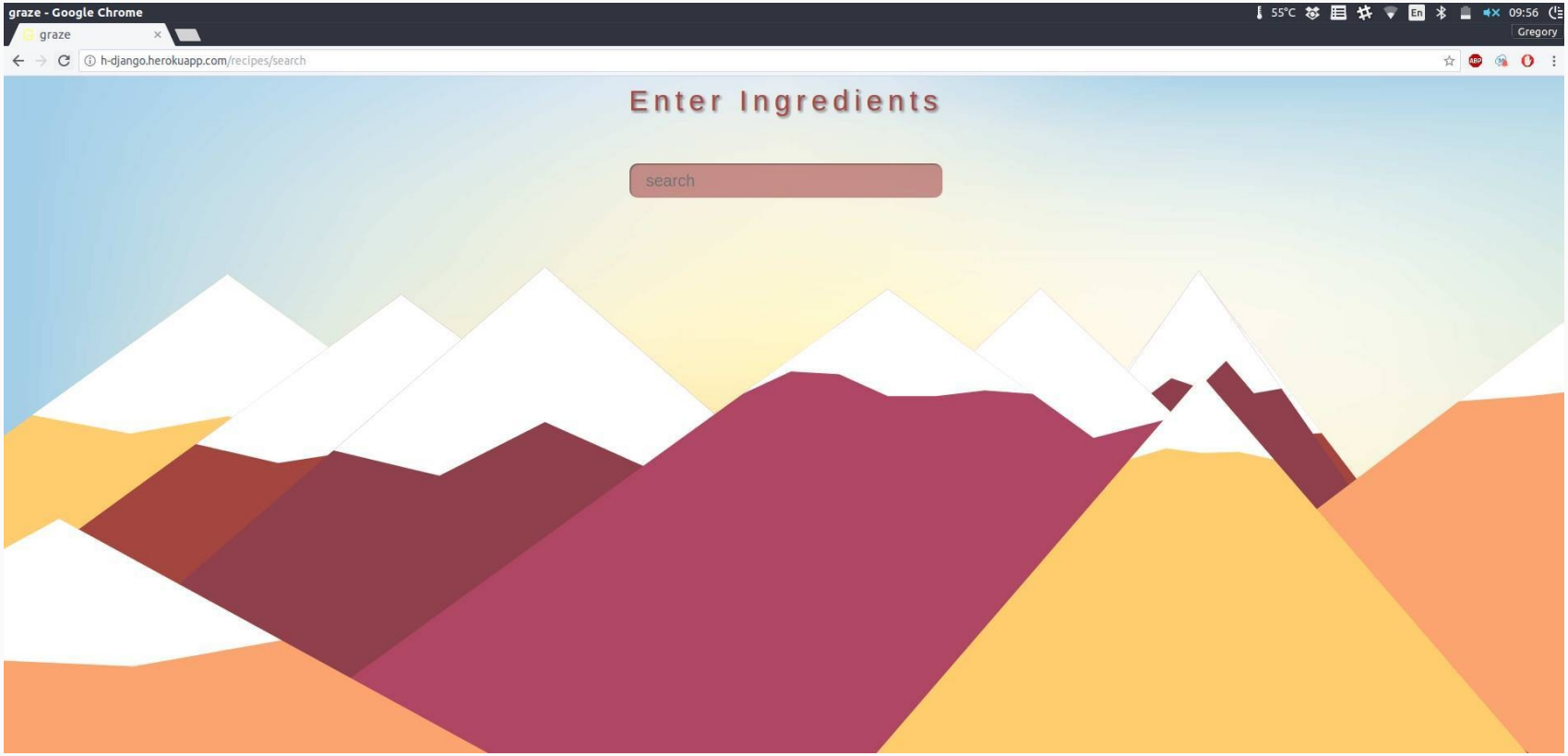
Features and Functionality

Feature: Ingredient Input

The main function of the website is the user's ability to enter whatever ingredients they have on hand.

The home page of the website is where the user accesses this feature, bypassing any need for logging in or registering, keeping in line with our ease-of-use beliefs.

Feature: Ingredient Input

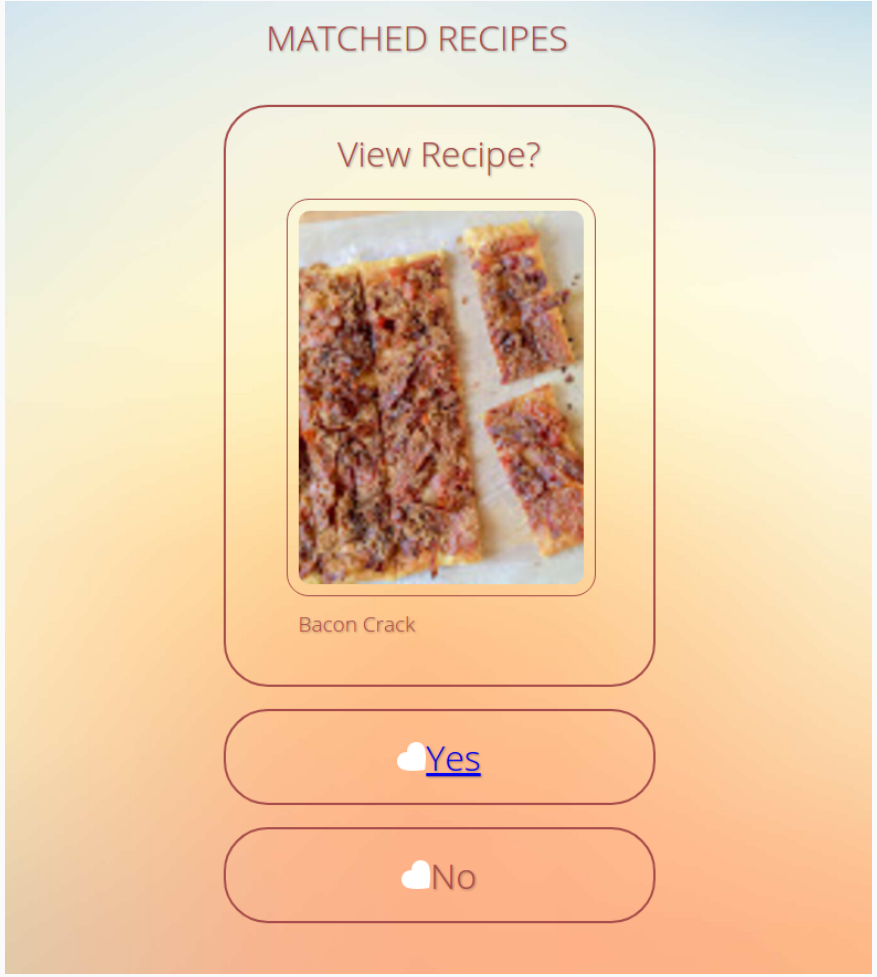


Feature: Recipe Retrieval

The hardest part of choosing recipes from an online service is the sheer number of options.

Taking cues from some popular dating apps we figured the user would be better served to get results one at a time with an accept or reject option attached to each recipe returned.

Feature: Recipe Retrieval



Growing Database

Yummly has over 2,000,000 recipes, far beyond the scope of our project in terms of database size.

- To start pulled in $\sim$ 100 recipes from Yummly to have a database that could be queried
- Built search function such that:
  - If ingredient is *not* found do a Yummly search via the API
  - Add results to database
- Now have growing database
- As site gets used functionality and efficiency are increased

Demo

Demo

Link to website

Challenges

Challenges: Initial Setup

One major challenge we faced was getting everyone on the same page with development. We are working on three different operating systems each with its own requirements. Docker was a major help with this, but we encountered an unforeseen delay in setting everyone up with a working development environment.

Challenges: Data

Getting meaningful data.

- Could enter a small amount of data by hand, but will be limited and time consuming
- Found *Yummly*, a large API that allowed us to build an initial database using python scripts and allows the database to grow as the website is used

Challenges: Language Processing

- How to differentiate between ”kumquat” and ”kumquats” (actual test case)
- Or ”cardamom” and ”green cardamom pods” (another actual test case)

Given time and resources a more robust language processing approach that allowed for things like close matches and spelling errors would have been employed.

- Final method: match on substrings (any ingredient containing the user’s input string
- ”beef” will match with ”ground beef”
- ”peppers” will get ”bell peppers” and ”jalapeno peppers”

It is not a perfect solution but it runs quickly and was a way to improve functionality given our time constraints.

Conclusion

Results

- Deployed a functional website from scratch using common software development tools and methodology
- Maintained a working Git repository hosted on Github with appropriate use of branching and merging
- Learned to employ the Agile methodology to keep our project on track and adapt to any difficulties we encountered
- Learned the priceless value of friendship

Next Steps

Given time and resources a future for this project could be to:

- Improve search functionality to optimize the search algorithm and use more string comparison techniques
- Implement a user based rating system so that a secondary sorting can be performed on feedback of each recipe
- Create *optional* accounts to allow users to save their favorite recipes or recommend to friends
- Extend to mobile platforms (either web or app based)

Questions?