<http://sulley.cah.ucf.edu/~ni927795/SimilarDish/>

suggested photo for testing food though any should work.

If no food is found or less than 85% probability it won’t show any food.

These images were used for testing purposes.

<http://sulley.cah.ucf.edu/~ni927795/SimilarDish/images/PhoFactory.jpg>

<http://sulley.cah.ucf.edu/~ni927795/SimilarDish/images/food.jpg>

Walkthrough –

1. Click red button that is at the bottom right.
2. Click the button above the black line to start the photo-upload process.
3. Click the submit button beneath it to the right.
4. Site will slide down to the newly shown options section.
5. The user can scroll through the top 20 options available from the api mashup.
6. After hovering over an item, it will turn light green and upon clicking a new modal will popup showing the full details of the meal selected.
7. Once done click anywhere else to return to the list
8. Hitting the bottom right button again the user can try a new image and restart the process over again.

1) Two Apis used:

https://developer.clarifai.com/welcome/

Used Clarifi for image recognition

Lines 51-69: dish.js

https://market.mashape.com/spoonacular/recipe-food-nutrition

Spoonacular for taking the images returned as a string (from Clarifi) to find similar meals.

Lines 111-122: dish.js

Lines 1-17: spoonacular.php

2) The main goal of the site is to upload an image and find all ingredients that Clarifi thinks has an 85% possibility of being in the uploaded image. These ingredients are passed to Spoonacular and returns up to 20 meals that have similar ingredients. In each item, it can be expanded to a modal with a larger image, all ingredients, and steps to cook.

3) The user interface was built with googles materialize and hand coded css. The site is responsive and has a thoughtful color pallete to represent food (light green and salmon). The desktop view differes from the mobile view where the table of items goes from two in a row two a single 100% width.

All the content is shown or hidden when the user needs it and can get access within a few steps of work.

The table for suggested items is slid into the viewers screen when the meals are loaded in

Lines 107: dish.js

4) The content is dynamically created based upon the user’s individual photo-upload. Each image uploaded will provide different results.

Lines 163-251 dish.js

5) My original element is the ability to upload photos to Sully and then parse back the location to which now sully holds a copy of the image uploaded. This image url is what is given to Clarifi to read the image.

phpMyAdmin db called SimilarDish

Lines 1-27: upload.php