Affamato: Phase One Report

Team:
Canvas Group: **Falcon**

Phase One Team Lead: Alex Issa

Members:
- Cameron Clark: cameron.clark0821@utexas.edu \\ github.com/cameronclark0821
- Justin Henry: justinhenry@utexas.edu \\ github.com/justinhenry
- Alex Issa: alex.issa32@utexas.edu \\ github.com/alexissa32
- Julia Rebello: julialrebello@utexas.edu \\ github.com/JLRebello
- Samir Riad: sriad123@utexas.edu \\ github.com/sriad123
- Rooshi Patidar: rooshipatidar@utexas.edu \\ github.com/rooshimadethis

URL to Github/Gitlab repo and shared Google docs:
- Github: https://github.com/alexissa32/Affamato
- Google Drive: https://drive.google.com/drive/folders/0ANUp-cLx6lnZUk9PVA
- Slack: https://team-falcon-group.slack.com

Website URL: https://www.affamato.xyz/

Phase I Report Contents:

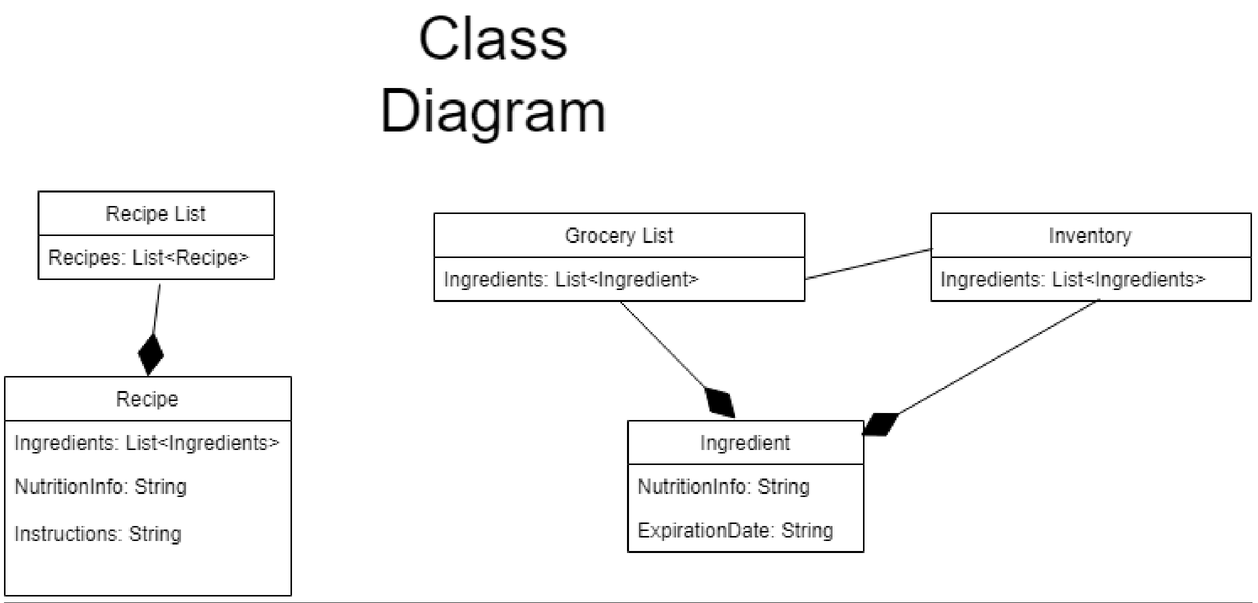
<u>1. Goals and Accomplishments</u>

- At least 5 user stories on issue boards (Done)
- **Beginnings of scraping and database: (In Progress)**
    - Start scraping (Wrote own python scripts) (Done)
    - Food item data scraping functional - over 150 ingredients scraped and saved as JSON files (spoonacular/yummly) (Done)
    - Recipe scraping functional - over 20 recipes scraped and saved as JSON files (spoonacular/yummly) (Done)
    - **Basic database running on GCP Datastore (NoSQL) with the information above stored. JSON files saved as Recipe/Ingredient Entities defined in Java (In Progress, very close to working)**
- Barebones UI built from DHTML, reactjs, and bootstrap: (Done)
    - Static site with at least 5 pages hosted on GCP (Done)
    - Basic pages for: Welcome/login page, grocery list page, my recipes page, my inventory page, about page (Done)
        - NOTE: Will not be fully functional: Users won't be able to interact with food items, recipes, or grocery lists yet, nor will accounts be functioning
    - Stats derived on the About page, dynamically, from GitHub (Done)
    - Progress made on dynamic versions of the page with reactjs and bootstrap (Done)
- Other things:
    - Team needs to learn about basics of APIs, GCP, SQL, NoSQL, Postman, DHTML, reactjs, bootstrap, Selenium, Mocha, Slack and Github (Done)
    - URL from a hostname provider (Namecheap) (Done)
    - Report, updating as we go (Done)

Comments:

- To see our user stories on issue boards, check our Github. To see them in the report, check the "Design and Requirements" section below.
- To see the other major bullet points and their progress, check the relevant sections titles below.
- Note: Getting the data into the datastore has been a bigger challenge than we thought, due to conversion and organization issues. It will be done by phase 2.

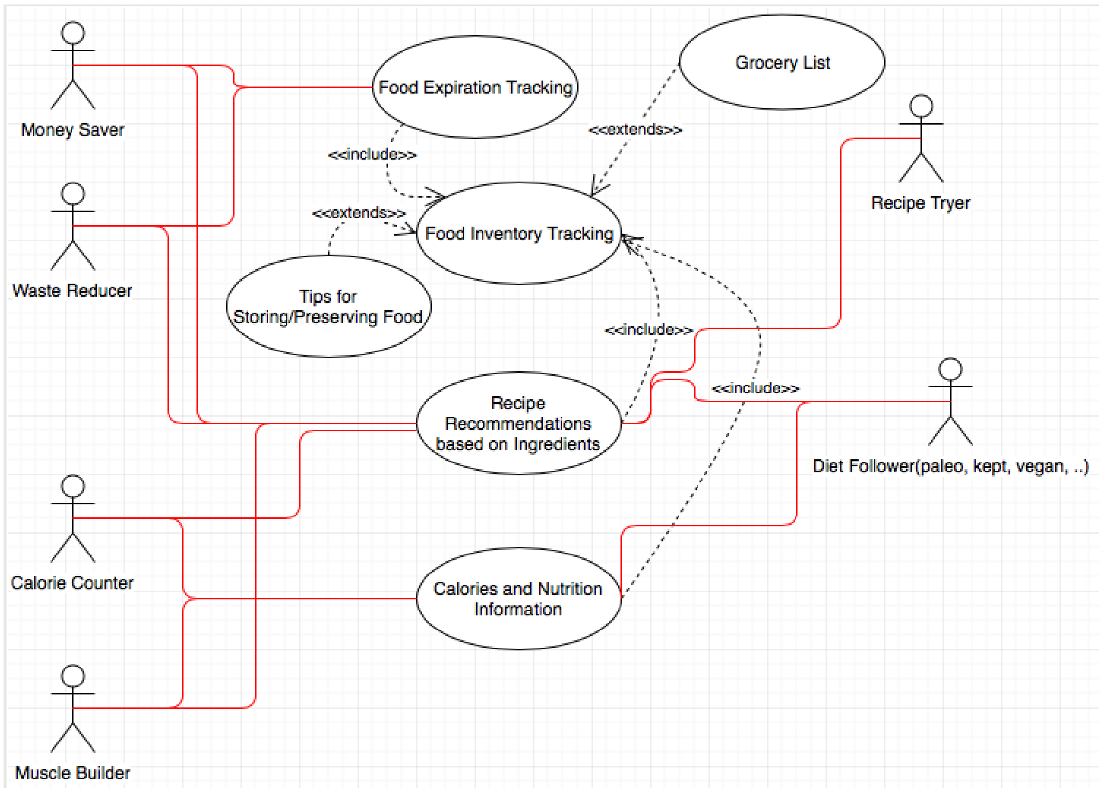
2. Design and Requirements



Phase I User Stories:

1. Save user's money.
    a. As a thrifty consumer, I would like to cut down on grocery shopping by efficiently using my groceries, so I am not re-purchasing items unnecessarily.
2. Reduce food waste.
    a. As a green consumer, I would like to reduce the amount of food I waste by tracking when items expire so that I may use all of the perishables that I purchase.
3. Calorie counter: lose weight, focusing on calories.
    a. As someone who would like to lead a healthier lifestyle, I would like to have quick access to the nutritional information of the food I buy.
4. Tries new recipes (For example, wanting cuisine type X).
    a. As an adventurous foodie, I would like to try new recipes.
5. Bulking: gain weight, focusing on maximizing volume of protein and healthy fats, getting more with less so one can fit it in the fridge/only need to get groceries once a week.
    a. As a bodybuilder, I would like to increase my muscle mass by consuming the right foods at a great enough volume.
6. Trying to follow a specific diet - keto, vegan, kosher, etc.
    a. As a someone who has dietary and health restrictions, I would like to have easy access to what foods I can eat, and what recipes I can make with them.

Phase I User Diagrams:



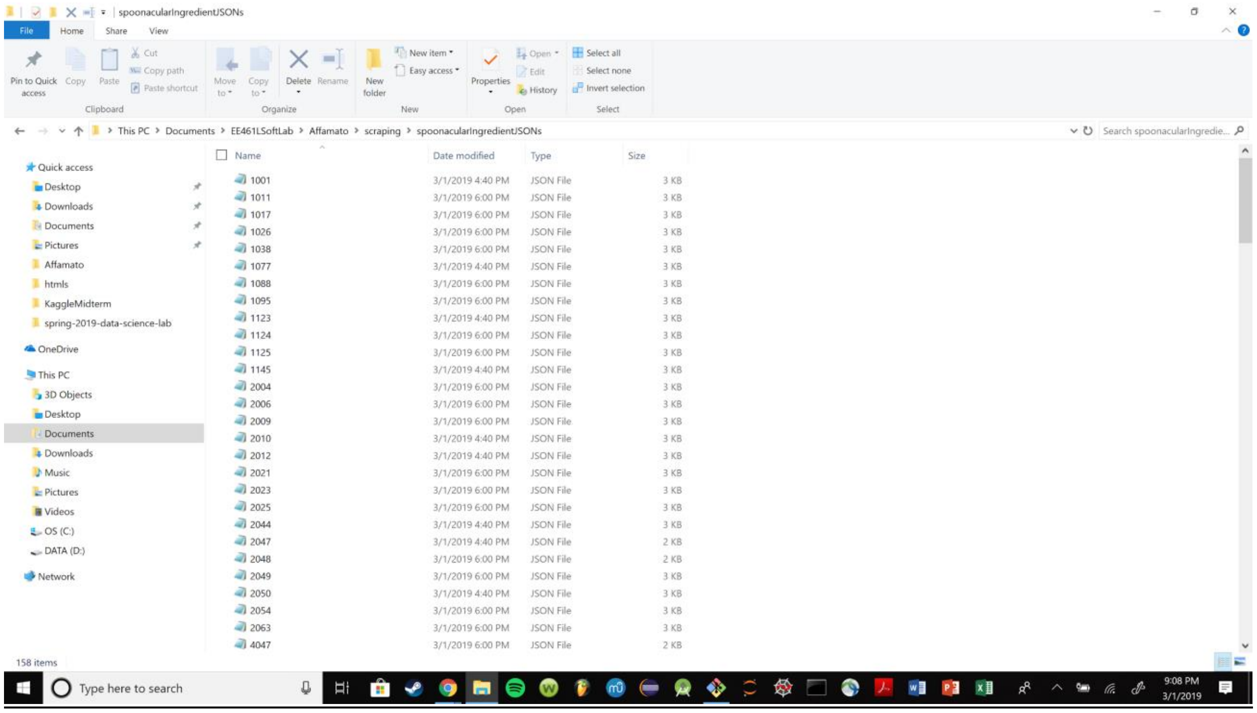
3. Scraping and Database (Description)

For phase one, we have scraped over 150 ingredients and over 20 recipes from Spoonacular and Yummly. Writing the Python scripts to do all this took quite a bit of time, and we stored this JSON information in individual .json files, a big .txt file, and a big .xlsx file. We did this so that based on different formats/database options, we could easily move the information to the respective database. It wasn't until recently that we decided to use a NoSQL GCP Datastore over a MySQL database. We made this decision because there are many places where we will have sparse information, and overall we felt it would be more efficient to use NoSQL as a result. We have used Java with Google's objectify library to get towards storing our scraped .json files into the datastore, and will complete and expand on that in phase 2. The major block there has been figuring out how to implement the Java code such that it can upload the json information to the database one time, and not re-run everytime we update the AppEngine or create multiple copies of the same information.

All of our code can be seen on the Github.

4. Scraping and Database (Screenshots)

-Extensive List of Ingredients TOP:

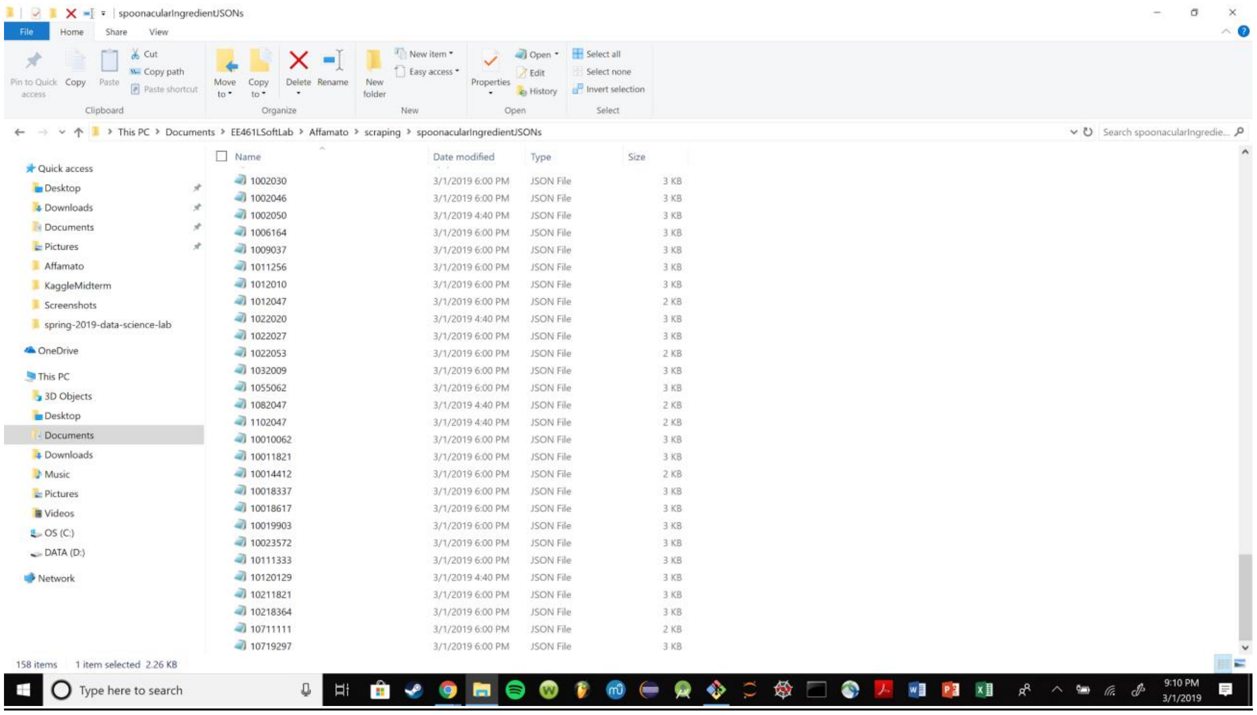


This PC > Documents > EE461LSoftLab > Affamato > scraping > spoonacularIngredientJSONs

Name	Date modified	Type	Size
1001	3/1/2019 4:40 PM	JSON File	3 KB
1011	3/1/2019 6:00 PM	JSON File	3 KB
1017	3/1/2019 6:00 PM	JSON File	3 KB
1026	3/1/2019 6:00 PM	JSON File	3 KB
1038	3/1/2019 6:00 PM	JSON File	3 KB
1077	3/1/2019 4:40 PM	JSON File	3 KB
1088	3/1/2019 6:00 PM	JSON File	3 KB
1095	3/1/2019 6:00 PM	JSON File	3 KB
1123	3/1/2019 4:40 PM	JSON File	3 KB
1124	3/1/2019 6:00 PM	JSON File	3 KB
1125	3/1/2019 6:00 PM	JSON File	3 KB
1145	3/1/2019 4:40 PM	JSON File	3 KB
2004	3/1/2019 6:00 PM	JSON File	3 KB
2006	3/1/2019 6:00 PM	JSON File	3 KB
2009	3/1/2019 6:00 PM	JSON File	3 KB
2010	3/1/2019 4:40 PM	JSON File	3 KB
2012	3/1/2019 4:40 PM	JSON File	3 KB
2021	3/1/2019 6:00 PM	JSON File	3 KB
2023	3/1/2019 6:00 PM	JSON File	3 KB
2025	3/1/2019 6:00 PM	JSON File	3 KB
2044	3/1/2019 4:40 PM	JSON File	3 KB
2047	3/1/2019 4:40 PM	JSON File	2 KB
2048	3/1/2019 6:00 PM	JSON File	2 KB
2049	3/1/2019 6:00 PM	JSON File	3 KB
2050	3/1/2019 4:40 PM	JSON File	3 KB
2054	3/1/2019 6:00 PM	JSON File	3 KB
2063	3/1/2019 6:00 PM	JSON File	3 KB
4047	3/1/2019 6:00 PM	JSON File	2 KB

158 items

-Extensive List of Ingredients BOTTOM:

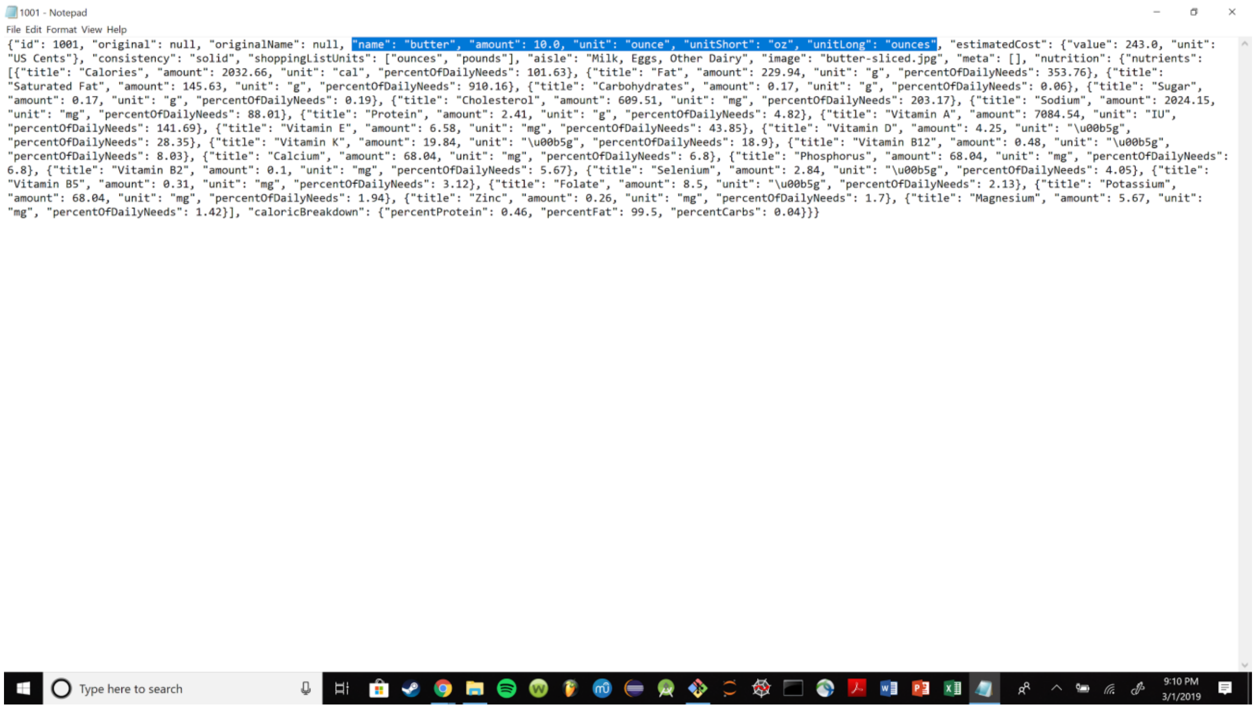


This PC > Documents > EE461LSoftLab > Affamato > scraping > spoonacularIngredientJSONs

Name	Date modified	Type	Size
1002030	3/1/2019 6:00 PM	JSON File	3 KB
1002046	3/1/2019 6:00 PM	JSON File	3 KB
1002050	3/1/2019 4:40 PM	JSON File	3 KB
1006164	3/1/2019 6:00 PM	JSON File	3 KB
1009037	3/1/2019 6:00 PM	JSON File	3 KB
1011256	3/1/2019 6:00 PM	JSON File	3 KB
1012010	3/1/2019 6:00 PM	JSON File	3 KB
1012047	3/1/2019 6:00 PM	JSON File	2 KB
1022020	3/1/2019 4:40 PM	JSON File	3 KB
1022027	3/1/2019 6:00 PM	JSON File	3 KB
1022053	3/1/2019 6:00 PM	JSON File	2 KB
1032009	3/1/2019 6:00 PM	JSON File	3 KB
1055062	3/1/2019 6:00 PM	JSON File	3 KB
1082047	3/1/2019 4:40 PM	JSON File	2 KB
1102047	3/1/2019 4:40 PM	JSON File	2 KB
10010062	3/1/2019 6:00 PM	JSON File	3 KB
10011821	3/1/2019 6:00 PM	JSON File	3 KB
10014412	3/1/2019 6:00 PM	JSON File	2 KB
10018337	3/1/2019 6:00 PM	JSON File	3 KB
10018617	3/1/2019 6:00 PM	JSON File	3 KB
10019903	3/1/2019 6:00 PM	JSON File	3 KB
10023572	3/1/2019 6:00 PM	JSON File	3 KB
10111333	3/1/2019 6:00 PM	JSON File	3 KB
10120129	3/1/2019 4:40 PM	JSON File	3 KB
10211821	3/1/2019 6:00 PM	JSON File	3 KB
10218364	3/1/2019 6:00 PM	JSON File	3 KB
10711111	3/1/2019 6:00 PM	JSON File	2 KB
10719297	3/1/2019 6:00 PM	JSON File	3 KB

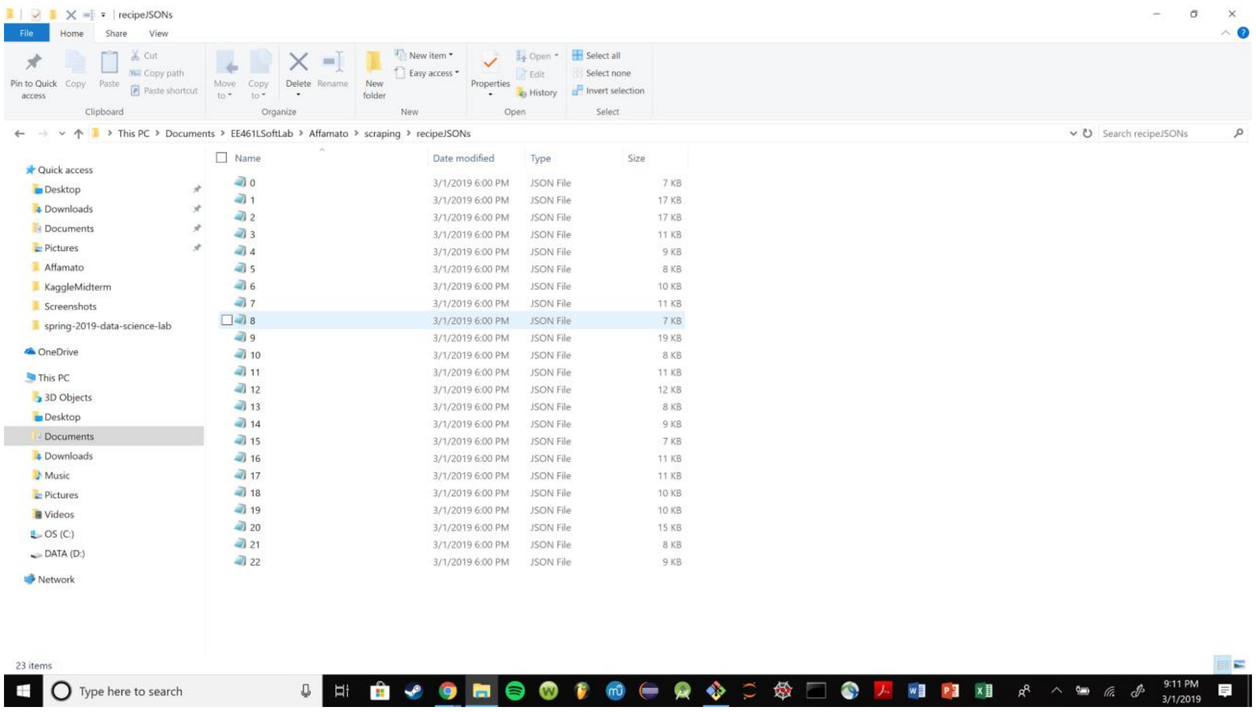
158 items 1 item selected 2.26 KB

-Example Ingredient JSON:



```
{"id": 1001, "original": null, "originalName": null, "name": "butter", "amount": 10.0, "unit": "ounce", "unitShort": "oz", "unitLong": "ounces", "estimatedCost": {"value": 243.0, "unit":
"US Cents"}, "consistency": "solid", "shoppingListUnits": ["ounces", "pounds"], "aisle": "Milk, Eggs, Other Dairy", "image": "butter-sliced.jpg", "meta": [], "nutrition": {"nutrients":
[{"title": "Calories", "amount": 2032.66, "unit": "cal", "percentOfDailyNeeds": 101.63}, {"title": "Fat", "amount": 229.94, "unit": "g", "percentOfDailyNeeds": 353.76}, {"title":
"Saturated Fat", "amount": 145.63, "unit": "g", "percentOfDailyNeeds": 910.16}, {"title": "Carbohydrates", "amount": 0.17, "unit": "g", "percentOfDailyNeeds": 0.06}, {"title": "Sugar",
"amount": 0.17, "unit": "g", "percentOfDailyNeeds": 0.19}, {"title": "Cholesterol", "amount": 609.51, "unit": "mg", "percentOfDailyNeeds": 203.17}, {"title": "Sodium", "amount": 2024.15,
"unit": "mg", "percentOfDailyNeeds": 88.01}, {"title": "Protein", "amount": 2.41, "unit": "g", "percentOfDailyNeeds": 4.82}, {"title": "Vitamin A", "amount": 7084.54, "unit": "IU",
"percentOfDailyNeeds": 141.69}, {"title": "Vitamin E", "amount": 6.58, "unit": "mg", "percentOfDailyNeeds": 43.85}, {"title": "Vitamin D", "amount": 4.25, "unit": "\u00b5g",
"percentOfDailyNeeds": 28.35}, {"title": "Vitamin K", "amount": 19.84, "unit": "\u00b5g", "percentOfDailyNeeds": 18.9}, {"title": "Vitamin B12", "amount": 0.48, "unit": "\u00b5g",
"percentOfDailyNeeds": 8.03}, {"title": "Calcium", "amount": 68.04, "unit": "mg", "percentOfDailyNeeds": 6.8}, {"title": "Phosphorus", "amount": 68.04, "unit": "mg", "percentOfDailyNeeds":
6.8}, {"title": "Vitamin B2", "amount": 0.1, "unit": "mg", "percentOfDailyNeeds": 5.67}, {"title": "Selenium", "amount": 2.84, "unit": "\u00b5g", "percentOfDailyNeeds": 4.05}, {"title":
"Vitamin B5", "amount": 0.31, "unit": "mg", "percentOfDailyNeeds": 3.12}, {"title": "Folate", "amount": 8.5, "unit": "\u00b5g", "percentOfDailyNeeds": 2.13}, {"title": "Potassium",
"amount": 68.04, "unit": "mg", "percentOfDailyNeeds": 1.94}, {"title": "Zinc", "amount": 0.26, "unit": "mg", "percentOfDailyNeeds": 1.7}, {"title": "Magnesium", "amount": 5.67, "unit":
"mg", "percentOfDailyNeeds": 1.42}], "caloricBreakdown": {"percentProtein": 0.46, "percentFat": 99.5, "percentCarbs": 0.04}}}
```

-List of Recipes:



Name	Date modified	Type	Size
0	3/1/2019 6:00 PM	JSON File	7 KB
1	3/1/2019 6:00 PM	JSON File	17 KB
2	3/1/2019 6:00 PM	JSON File	17 KB
3	3/1/2019 6:00 PM	JSON File	11 KB
4	3/1/2019 6:00 PM	JSON File	9 KB
5	3/1/2019 6:00 PM	JSON File	8 KB
6	3/1/2019 6:00 PM	JSON File	10 KB
7	3/1/2019 6:00 PM	JSON File	11 KB
8	3/1/2019 6:00 PM	JSON File	7 KB
9	3/1/2019 6:00 PM	JSON File	19 KB
10	3/1/2019 6:00 PM	JSON File	8 KB
11	3/1/2019 6:00 PM	JSON File	11 KB
12	3/1/2019 6:00 PM	JSON File	12 KB
13	3/1/2019 6:00 PM	JSON File	8 KB
14	3/1/2019 6:00 PM	JSON File	9 KB
15	3/1/2019 6:00 PM	JSON File	7 KB
16	3/1/2019 6:00 PM	JSON File	11 KB
17	3/1/2019 6:00 PM	JSON File	11 KB
18	3/1/2019 6:00 PM	JSON File	10 KB
19	3/1/2019 6:00 PM	JSON File	10 KB
20	3/1/2019 6:00 PM	JSON File	15 KB
21	3/1/2019 6:00 PM	JSON File	8 KB
22	3/1/2019 6:00 PM	JSON File	9 KB

5. Screens, Features and Functionality (Description)

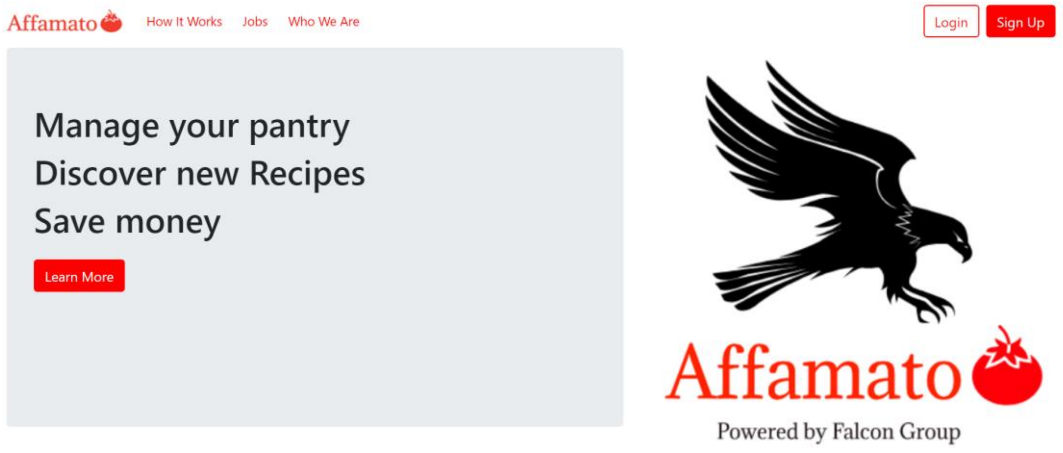
For phase one, we have implemented a static website (In DHTML), which will have the login/welcome page, grocery list page, my recipes page, my inventory page, and about page. The pages are not fully functional; the accounts are not implemented, and interaction with food/recipes is not functional. Users won't be able to interact with food items or recipes until next phase, as most of our work was on setting up the framework for the website and the database. Please go to the link to see the static website.

We also have made steps toward developing a dynamic user interface (With reactjs and bootstrap) which is demonstrated below.

All of our code can be seen on the Github.

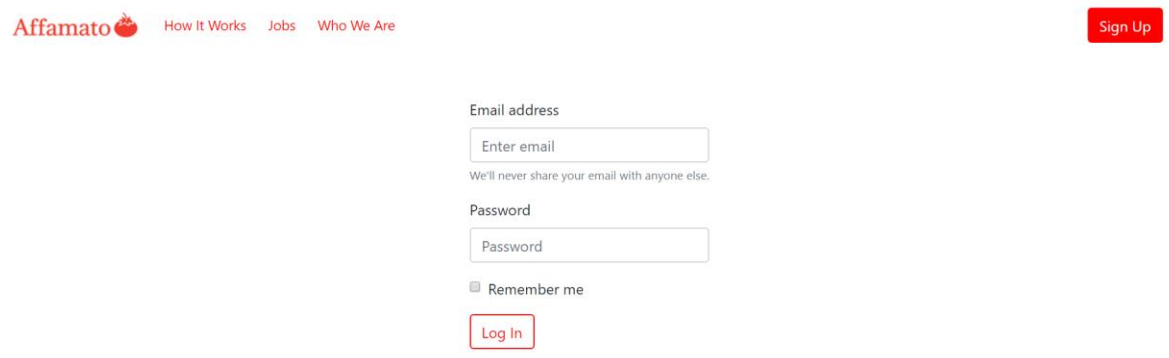
6. Screens, Features and Functionality (Screenshots)

-Affamato Welcome Page:



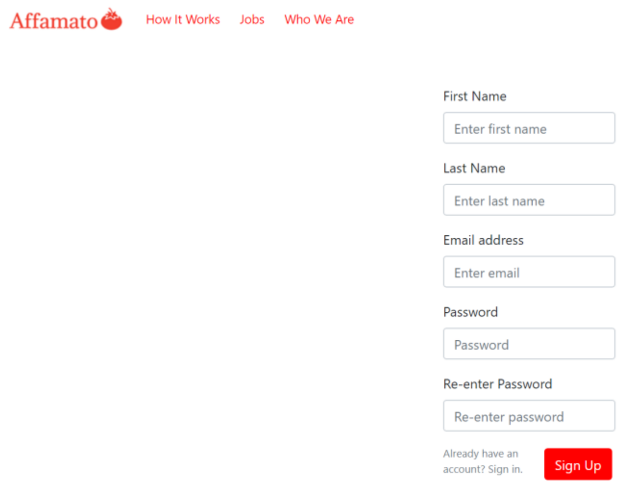
Manage your pantry
Discover new Recipes
Save money

-Affamato Login Page:



Email address
Enter email
We'll never share your email with anyone else.
Password
Password
Remember me

-Affamato Sign Up Page:



First Name
Enter first name
Last Name
Enter last name
Email address
Enter email
Password
Password
Re-enter Password
Re-enter password
Already have an account? Sign in.

7. Tools, Software, Frameworks Used

In this phase, we obtained Affamato's URL from Namecheap and set up team communication on Slack, which has been integrated with Affamato's Github repository. Phase One user stories have all been added to our issue board. Our user interface has been developed with DHTML, reactjs and bootstrap. We've implemented scraping with Python scripts from Spoonacular and Yummly APIs. We have used Java with Google's objectify library to store the info in the GCP Datastore, and will complete and expand on that in phase 2. We also started tinkering with Java Spring for our backend framework, and will make more progress with that in phase 2. In addition, each team member was expected to familiarize themselves with the purpose and how to use the software we listed in the goals section above.

8. Testing

During Phase 1, our team visually inspected and manually tested all components of our web application. We will be moving towards unit testing during phase 2 with JUnit, Python tests, Mocha, and Selenium.

9. Updated Phase Goals

Phase I - due March 1, 10pm

- At least 5 user stories on issue boards (Done)
- **Beginnings of scraping and database: (In Progress)**
    - Start scraping (Wrote own python scripts) (Done)
    - Food item data scraping functional - over 150 ingredients scraped and saved as JSON files (spoonacular/yummly) (Done)
    - Recipe scraping functional - over 20 recipes scraped and saved as JSON files (spoonacular/yummly) (Done)
    - **Basic database running on GCP Datastore (NoSQL) with the information above stored. JSON files saved as Recipe/Ingredient Entities defined in Java (In Progress)**
- Barebones UI built from DHTML, reactjs, and bootstrap: (Done)
    - Static site with at least 5 pages hosted on GCP (Done)
    - Basic pages for: Welcome/login page, grocery list page, my recipes page, my inventory page, about page (Done)
        - NOTE: Will not be fully functional: Users won't be able to interact with food items, recipes, or grocery lists yet, nor will accounts be functioning
    - Stats derived on the About page, dynamically, from GitHub (Done)
    - Progress made on dynamic versions of the page with reactjs and bootstrap (Done)
- Other things:
    - Team needs to learn about basics of APIs, GCP, SQL, NoSQL, Postman, DHTML, reactjs, bootstrap, Selenium, Mocha, Slack and Github (Done)
    - URL from a hostname provider (Namecheap) (Done)
    - Report, updating as we go. (Done)

Phase II - due March 14

- At least 5 additional user stories on issue boards
- Collection of a lot of data from your sources and storing in Database:
    - Automated scraping to retrieve food items and associated info
    - Many recipes and food items scraped - 500+ each
    - Users have an account and login stored in the database
    - All objects have functional instances: user, food items, pantry, recipes, etc.
- Creating a dynamic website with many pages hosted on GCP:
    - Conversion from static to dynamic pages on GCP
    - Login page is fully functional but no new account creation
    - Inventory page with food information is fully functional
    - Recipe List is functional, making a recipe removes the items from your inventory
    - Grocery List is functional, can add grocery list to inventory
    - Results on search page (recipes or food items) are fully functional
    - Add pages for search results, discover page.
- Connection of frontend and database via a backend using Java Spring and Jackson:
    - Build internal APIs
    - Begin basic search functionality

    - Created Food, Recipe, and Grocery List objects stored in database?
- Testing:
    - Most unit tests written for the API with Postman
    - Most unit tests for JS written with Mocha
    - Most GUI tests using Selenium
    - Potential Python and Java tests for Spring and scraping?
- Report refinement, updating as we go

Phase III - due April 9
- At least 5 more user stories –put them on your issue boards
- Finish data collection- all recipes, food, and relevant info is scraped and stored in the database
- Fully functional search implementation
- Fully functional account creation (username+password) in the database
- Refine your dynamic site with many pages hosted on GCP
    - All pages are in their final form in both appearance and functionality
    - Search algorithm works for complex scenarios
- Refine and expand ALL tests
- Add to and refine the technical report, updating as we go

Phase IV - due April 30
- Development is IDEALLY completed in phase III
- Refactor, apply design patterns
- Finish the final project report
- Catch up with something if stuff gets behind
- Expand on calorie/exercise tracking in accordance with our long term vision for a lifestyle app if we have lots of extra time
- Create a presentation and put it on GitHub as a pdf (not required until you present)