



Professor John Rachlin

FooCycle

Final Report Paper

Group 5

Hiren Patel

Vivian Chen

Ryan Liang

Vladislav Satchek

Spring 2020

4/14/2020

Class Section: T 6:00-9:15 PM

129 Forsyth Building, Northeastern University, Boston, MA 02115

FooCycle

DS4300 NoSQL Final Project

TABLE OF CONTENTS

I. OVERVIEW	Page 1
II. SYSTEM ARCHITECTURE	Page 2
III. VISUALIZATIONS	Pages 3 - 5
IV. ANALYSIS	Page 6

I. OVERVIEW

Mission Statement

To promote food sustainability by encouraging food waste reduction and facilitating food exchange.

Inspiration

Have you ever bought too many groceries or had food that you didn't want to eat anymore? Many Americans throw away perfectly edible food. According to the U.S. Food and Drug Administration, in the United States alone, Americans waste \$160B of food, which is nearly 30 to 40 percent of the entire U.S. food supply. At the same time, one in eight Americans struggle to put enough food on the table. Especially during today's circumstances, with the closing of restaurants, hotels and schools reducing the demand for farmers' food supply, and university housing evacuations leaving many people to abandon full refrigerators, there is an increasing need for a tool that distributes the supply to the demand.

What is FooCycle?

FooCycle provides the platform to allow consumers to exchange food items or leftovers within communities. This food that is not necessarily expired or bad, just not wanted by that particular consumer anymore. Community members can see the food postings within any community, browse through, and will be able to contact other users for food they might be interested in. For businesses that have extra food to sell before closing, FooCycle connects individuals to tasty meals that otherwise would not have been sold, creating an extra revenue stream and eco-friendly branding for your restaurant. FooCycle helps enhance supply chains in the food industry, connecting donors with consumers.

Great food should be tasted, not wasted.

II. SYSTEM ARCHITECTURE

Stack

FooCycle is split up between back-end and front-end components. The back-end architecture we chose was **MongoDB**, a NoSQL database we learned about in class. MongoDB is great for scalability, especially for the large amount of data that we are going to have, and omits using schemas. The front-end architecture we chose was **Flask**, which is a front-end framework based in JavaScript that allows users to build their user interface and design their web applications.

The project's main database system is built in NoSQL's MongoDB. The front-end of the database uses Flask to create an interactive web app for our project. Users will be able to query our MongoDB database through the interactive web app, and also be able to perform all the CRUD operations through MongoDB.

Functionality

In MongoDB, we are able to perform the basic CRUD operations for FooCycle. This includes create, for creating a user account, posting a new food item and adding a new community. Additionally, we are able to read data -- viewing different users by any of its attributes i.e., user ID, name, zip code, and viewing food posting by attributes. We can also update data, including updating basic profile attributes (i.e., age of user, name, password, etc.) and updating whether an item has been donated or not. Finally, we can delete data, such as deleting user profiles and deleting a food posting.

Repository

The following is the link to our GitHub repository where all of our code, presentation powerpoint, and final report lies. Enjoy!

https://github.com/ryanl35/DS4300_homework/tree/master/Project/fooCycleApp

III. VISUALIZATIONS

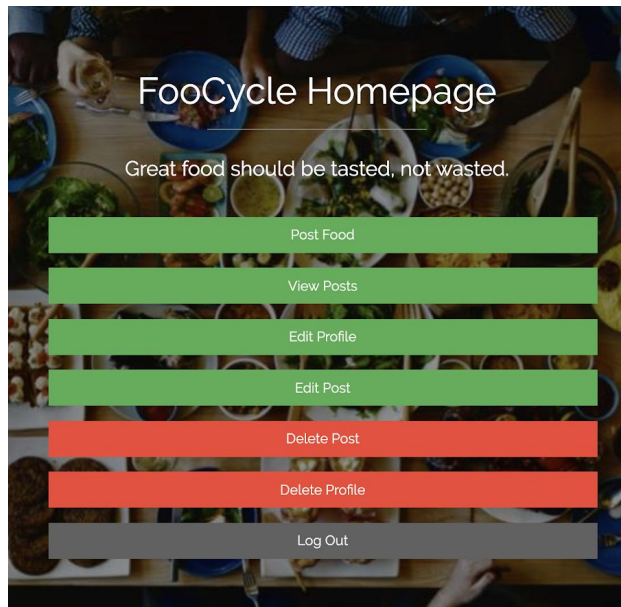
1. This is FooCycle's webpage. Users are brought here when they go to <http://localhost:5000/>



2. Users can register for an account at this webpage.

A screenshot of the FooCycle registration form. The background is a top-down view of a wooden table filled with various dishes, including bread, vegetables, and meat. The text "Register" is centered in a large, white, sans-serif font. Below it, there are five input fields: "Name" (with the text "Professor" entered), "Username" (with the text "jrachlin" entered), "Zipcode" (with the text "02115" entered), "Password" (with a single dot entered), and a "Back" button. At the bottom, there is a green "Register" button.

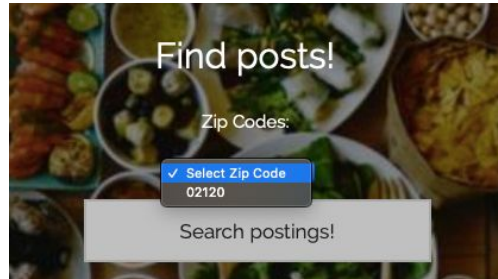
- After registering for an account, users are brought to the FooCycle Homepage, where they have options to make a food post for their donation, view it, edit any details after posting, or remove it when necessary (CRUD), as well as edit or delete their profile.



- Users can make a food post onto FooCycle at this screen.

The screenshot shows the 'Add Food' form. It includes a title 'Add Food' and a label 'Name of Food' above a text input field containing 'Carrots'. Below this is a label 'Describe your food!' above a text area containing 'A bag of carrots purchased at Star Market on Tuesday.' followed by a checkmark icon. The next section has a label 'Price' above a text input field containing '2'. Below that is a label 'Enter the zipcode of your food' above a text input field containing '02115'. At the bottom are two buttons: 'Post' (green) and 'Back' (grey). The background is a collage of various food dishes.

5. Before viewing posts, users must enter the zipcode of the posts they wish to view. FooCycle will then display all the food posts made in that community, and whether or not it has been donated yet.



Find posts!

Zip Codes:

☒ Select Zip Code
02120

Search postings!

Post ID:	Name of food:	Description:	Price:	Has this item been donated yet?
1	Carrots	A bag of carrots purchased at Star Market last Tuesday.	2	False
2	Banana	5 bananas, perfect for banana bread recipe	1	False
3	Pizza	Pizza left over from club meeting	1	False
4	Spinach	Bag of spinach, leaving because of NEU housing evacuation in 2 days	4	False
5	Bread	Bread, I have too much bread and too little PBJ	3	False

Back

6. Once users log out of their account, they are brought back to the FooCycle Webpage.



IV. ANALYSIS

We began this project armed with a MySQL database schema from a previous project and an ambitious aim to reduce food waste. Since the program's inception, we realized the need for the support of a large FooCycle community to support our cause to make a tangible impact on curbing food waste. Implementing our program in the MongoDB NoSQL database allows us to query food posts and large amounts of users much faster than MySQL. FooCycle is successful due to the collaboration of our active users and the amount of food posted on our webpage. MongoDB enables our users to achieve these aims.

When implementing our program, we fine tuned some features to make the webpage most efficient and user-friendly. First, we implemented a function that randomly generates a unique User ID of 6 alphanumeric characters. When users select their own username, the webpage provides feedback on whether or not the username has been taken yet. Additionally, we created flash messages like "Account created successfully!" so users receive feedback from MongoDB once they add an account. When users create food posts, they can post foods to zip codes as well as look up all posts within a drop-down list of zip codes. We also implemented the functionality for users to only be able to edit posts that they created, verifying their User ID stored as a session variable with the User ID of the user who created the post. Each webpage has a back button so the user can easily navigate to our various html templates. We also implemented a table layout of all food posts on FooCycle. As the database expands, this will allow for user-friendly navigation of all the food items available for exchange.

The Future of FooCycle

As we continue to improve our project, we would like to distinguish between FooCycle users and administrators. Admins would have the ability to delete posts that they did not make. This would also enable the authority to "create a community" within FooCycle. Given that FooCycle's purpose is achieved through the in-person exchange of food, users should be able to filter any of their local communities, even outside their zip code, to arrange pickups. Coupled with a graph database, admins would be able to query for zip codes within a specified radius to create a community. The admin will be able to create and delete communities, as well as update community names. Users will be able to view different communities by any of its attributes (zip code or name), and add themselves to these communities, and view the food postings within these communities. Admins will also be able to identify which users are trustworthy, either through a background check or frequent successful food exchanges, and display this authentication with a badge on their profile.

There is much more functionality and scalability for our project in its future. In the future, we would like to implement in-app communication to enable users to coordinate how to arrange pickups or inquire about the food product within the app, rather than using external forms of contact.