

Group 14

Team: Kenton Ma, Charn Rai, Benny Lo, Alex Macdonald, Adil Kydyrbayev
Developer Section: <https://github.com/Alexmac22347/DontEatThat>

Don't Eat That!

November 27, 2016

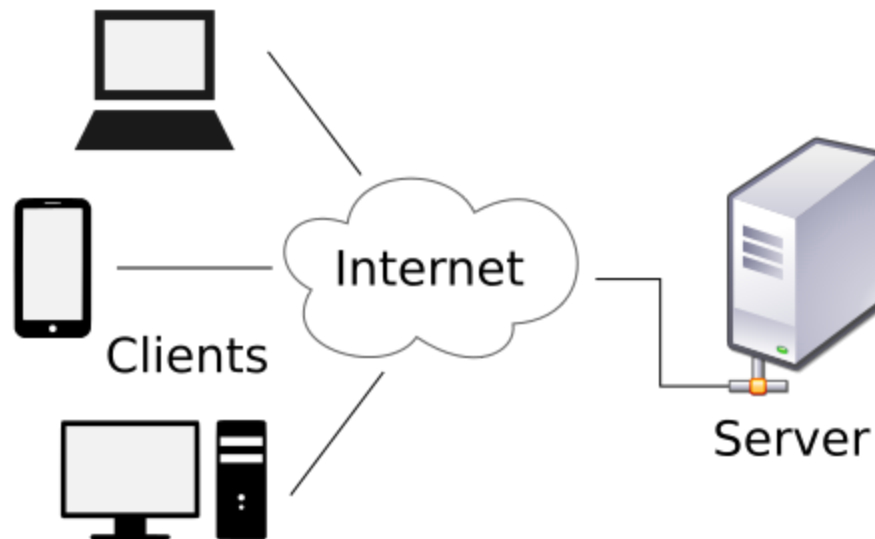
Design Document

INTRODUCTION

We plan to build Don't Eat That! as a web application that can be used by both general and registered users to compare two food products to determine the healthier option. The web application will retrieve nutritional data from the FatSecret Platform REST API, which uses a database of nutritional information. After retrieving the nutritional facts, Don't Eat That! will use an algorithm to determine the healthier option. The algorithm can be altered to meet the specific needs of a user.

This document will attempt to explain the architecture and design of the system using the 4+1 architecture model.

SYSTEM ARCHITECTURE AND RATIONALE



Client-Server Architecture

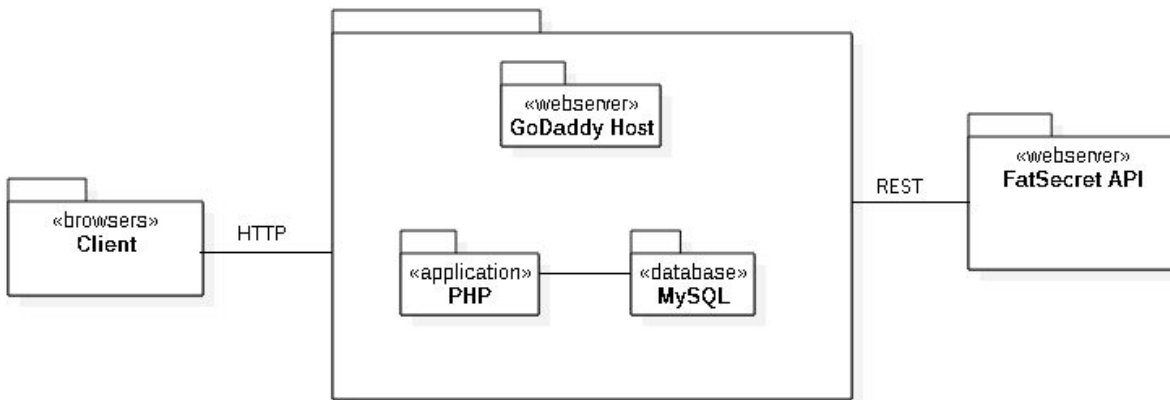
Source: <https://en.wikipedia.org/wiki/File:Client-server-model.svg>

This is a web application, so various languages and software platforms will be used. The application will follow a “Client-Server” architecture. We chose this architectural pattern because it resembles the type of interactions seen in our use cases. The user (*client*) is curious about the nutritional values of two food items and wants to learn more. The user requests information regarding their two food items of choice by using our web application (*server*). The front-end is written in HTML, CSS, JavaScript and uses the Bootstrap framework. The back-end consists of a REST API and a MySQL database. To connect the front and back ends, we are using PHP as our server-side programming language.

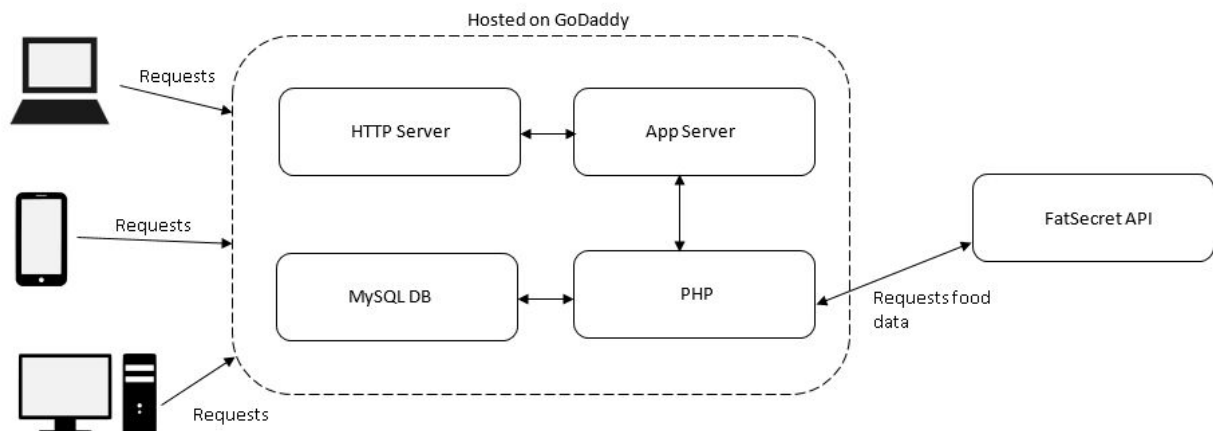
The web application consists of the following major components:

- 1) Web server on which the application is hosted and where client requests are processed
- 2) User database used to create and manage user accounts
- 3) FatSecret Platform REST API used to access nutritional information of food products

Deployment Diagram - Static View



Dynamic View and Description



HTML (HyperText Markup Language) & CSS (Cascading Style Sheets)

The user interfaces are written in HTML, the standard markup language for creating web pages and applications. HTML describes the structure and layout of a web page by using tags and attributes. CSS describes how HTML elements are displayed on the screen. CSS can be used to separate document content from presentation. This separation can improve content accessibility and provide flexibility and control in presentation characteristics. We chose to use HTML and CSS because these languages are common to make web pages, making it easy to find online reference/support.

Bootstrap

The front end (user interfaces) are developed using the Bootstrap framework. The Bootstrap framework is commonly used in web applications for responsiveness. Bootstrap automatically adjusts the layout of web pages according to the user's screen size. Therefore, our web application will have a user-friendly front end regardless if the user is on an Android phone/tablet, iPhone/iPad, or a commonly used browser such as Google Chrome or Mozilla Firefox. This responsiveness is the main reason we decided to use the Bootstrap framework. Another reason the Bootstrap framework was selected is because of its popularity, which made it easy for us to find tutorials and support.

jQuery

jQuery is a lightweight JavaScript library that is commonly used in web application development to make coding in JavaScript easier. jQuery can take tasks that require many lines of JavaScript code to run and turn the tasks into methods that can be easily called in a line of code. That was the reason we chose to use jQuery in our web application. By using jQuery, we can take complicated tasks, such as updating a form, autocomplete, and validating a form, and easily perform these tasks by calling a method. jQuery is also very popular amongst software developers, making it easy to find online resources/support for problems that arise during development.

MySQL

The web application is hosted by GoDaddy.com Inc., an Internet domain registrar, and web hosting company. GoDaddy provides web domains with database services using the MySQL database system. MySQL is a database system that runs on a server and uses standard SQL. We chose to use a domain host with MySQL because MySQL is free to use, compiles on multiple platforms, works for applications of all sizes, and has earned a reputation for reliability. These attributes allow our web application to work on multiple operating systems and browsers.

PHP (Hypertext Preprocessor)

PHP is an open-source, general-purpose scripting language that can be embedded into HTML. PHP code is executed by the server (back end), which generates HTML and sends it to the client (front end). Within the HTML code of the user interfaces, PHP code can be embedded, allowing the user interface pages to interact with the FatSecret API and user database. For example, a user can enter two food products and PHP code can be used to retrieve information about the food products from the API. We chose to use PHP because PHP supports a wide range of databases, can be used on multiple operating systems and supports most browsers. PHP is also a commonly used server side language, so online help and support are easily found.

When a user registers, sensitive information such as the user's name and password could potentially be retrieved by malicious users. To prevent these malicious actions, account security and authentication is achieved by using PHP's built-in encryption functions. As of version 5.5, PHP provides password encryption by using a bcrypt hash function `password_hash()`. Login authentication is also supported by PHP. Passwords are verified via a built-in function `verify_password()`.

FatSecret Platform REST API

The FatSecret Platform REST API allows us to integrate the comprehensive FatSecret database of nutritional information into our web application. We chose to use this API for multiple reasons. First, the FatSecret database has a reputation for accuracy and efficiency, which is a functional requirement of our application. Second, FatSecret offers a great deal of support for their API, making it easy to find guides, manuals, and libraries to make the most effective use of the API. Third, the API returns the correct result even if the user input is slightly incorrect, which enhances the user experience. Finally, the FatSecret REST API is free and allows unlimited calls a day with no throttling limits, which is more than enough for the planned usage of Don't Eat That!.

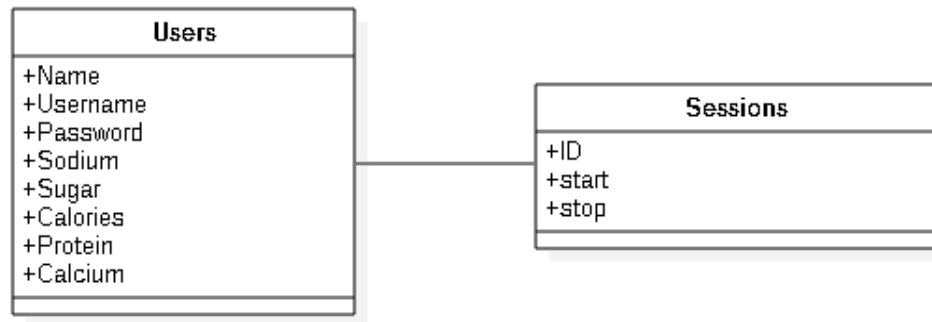
Originally, we had planned to use our own MySQL database of food items, but encountered a great deal of difficulty in finding the correct items. The original database was unable to deal with slightly incorrectly spelled inputs from the user. Furthermore, the database often returned variations of an item instead of the actual item. Attempts to fix these problems proved fruitless. It was apparent that the FatSecret REST API was a much more effective alternative.

Secure Sockets Layer

The application uses Secure Sockets Layer (SSL) security technology to establish an encrypted link between the server (backend) and the browser (frontend). We chose to use SSL technology for multiple reasons. First, GoDaddy offers an SSL Manager feature that makes it very easy to use SSL to secure our system. Second, SSL is widely used amongst web developers as it has a reputation for reliability and safety. Finally, SSL offers a visual cue to users that our application is secured. On Google Chrome, the visual cue is a green lock in the URL bar. The visual cue lets users know that their data is safe - providing them a peace of mind and enhancing their experience.

DATA

The web application uses two sources of data: user information and nutritional facts. The user information will be stored in a MySQL database provided by our domain and server host, GoDaddy. The nutritional facts will be stored in a database that is accessed by the FatSecret REST API. We will be responsible for the addition, removal, and maintenance of the data in the user database. However, we have no control over the FatSecret database. We can only access nutritional data from the FatSecret database through their API.



DETAILED DESIGN

Static Behaviour

The browser is connected to the application via Hypertext Transfer Protocol (HTTP). The application is hosted on a web server by GoDaddy. The application uses a MySQL database, which is accessed by using the server side language PHP. The application also uses a REST API from FatSecret.

Dynamic Behaviour

When a user uses the application, whether on a laptop, phone, or desktop computer, the application sends requests to the GoDaddy server. On the server, PHP scripts process the user's request and attempt to gather the appropriate nutritional data from the FatSecret REST API.

GUI

The user interface will be built using HTML, CSS, JavaScript, and the Bootstrap framework. With the use of PHP, we are able to create dynamic web pages. The theme of the user interface is to be learnable, efficient, and visually appealing. These design goals influence the choice of logo, layout, buttons, and colours. At the bottom of every page is a disclaimer that informs users

about the limitations of this web application (preventing users from confusing this web application as a source of medical advice) and attribution to FatSecret.

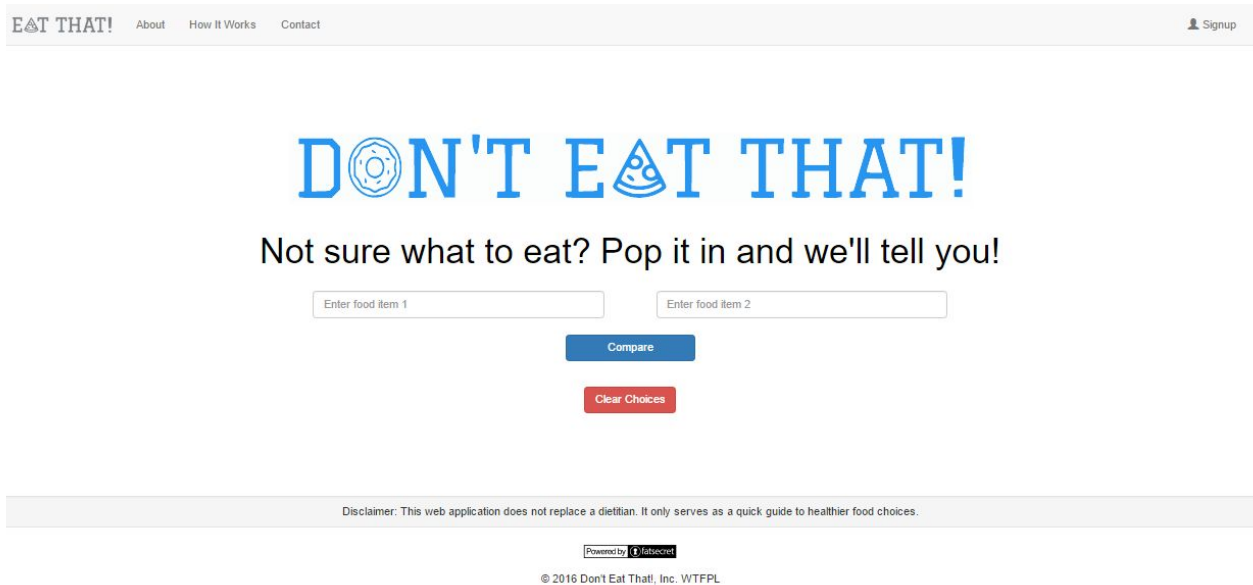
Validation

The user interface is reviewed with the client on a weekly basis. These reviews yielded some changes to our initial proposal of the user interface. The user recommended that minor changes be made to the fonts for an easier reading experience. Furthermore, the user interface was not consistent across different platforms and devices, so we implemented the Bootstrap framework. These changes resulted in a more user-friendly interface that worked across multiple platforms and devices.

Likewise, the functional requirements are reviewed with the client on a weekly basis as the developers gain experience developing the system. The changes in requirements resulting from these reviews are scheduled to be performed in the following iterations.

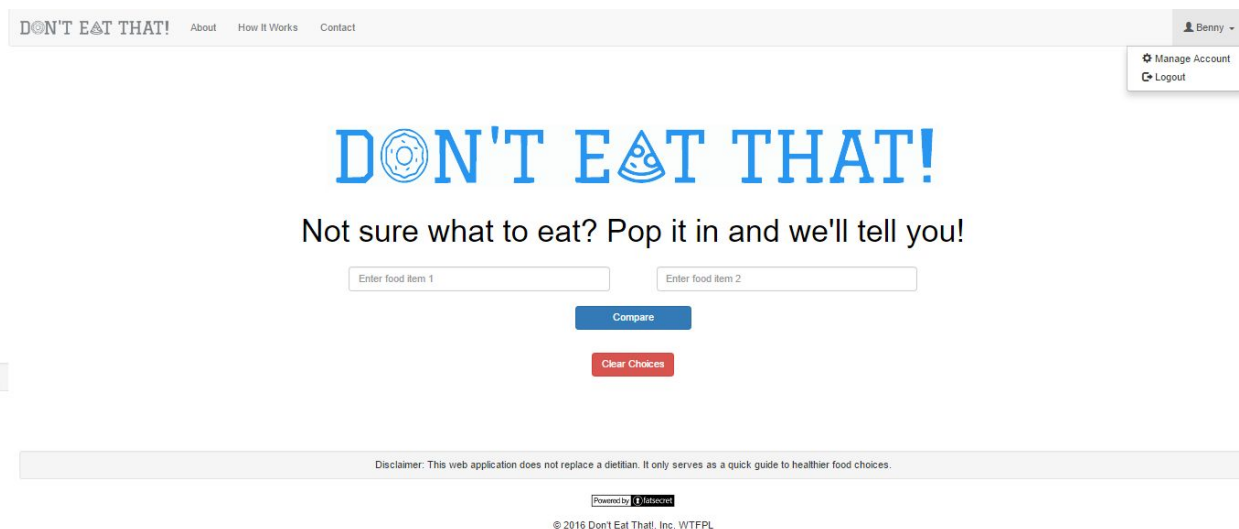
Home Page

The home page will be used by both registered and general users. Our logo and a catchy phrase telling the user to compare two food items are located at the top middle of the home page. Below, in the centre of the home page, the user will be able to enter in two food products for comparison. The user can clear the text boxes by using the “Clear Choices” button. The placement of the text boxes and the buttons is designed to be easy to understand and use. At the top of the home page, the user can provide feedback about the web application, log into their account, or sign up for an account.



Above: The home page when the user is not signed in (on Microsoft Edge)

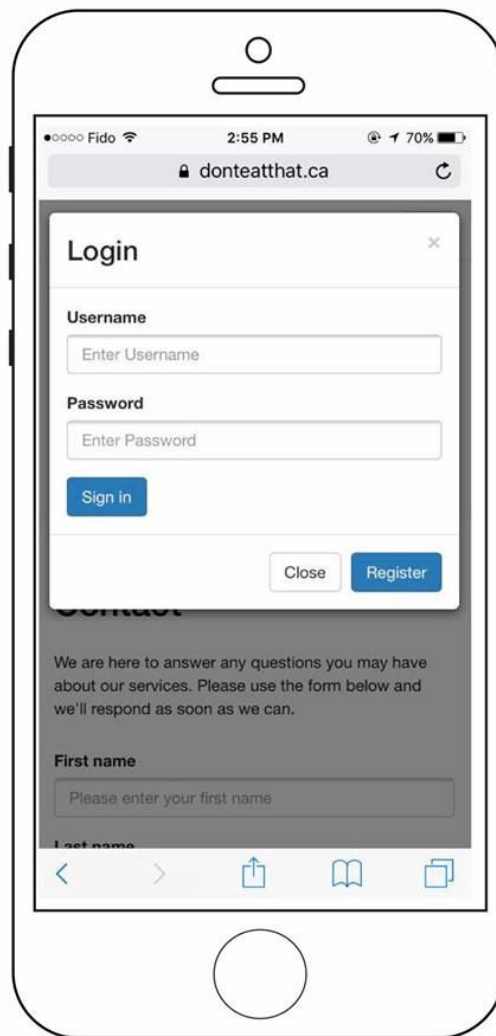
If the user has signed in, then the upper right corner of the home page will show the user's registered name and allow the user to access account settings or sign out. The home page displays the user's registered name to ensure the user that they are logged in and to make the user feel more comfortable.



Above: Home page when user is signed in (on Google Chrome)

Login Page

The login page is a modal used by users to log into their account. The login modal will temporarily deactivate all other features on the web page until it is closed. The login modal will have two textboxes: one for the username and one for the password. After entering the information, the user will log in by pressing the “Sign in” button. If the user does not have an account but wishes to make one, the user can do so by pressing the “Register” button, which will redirect the user to the Registration page. A modal was used instead of a separate page to create a more interactive and interesting user experience. Instead of simply going to another page, the user has something pop up at them.



Above: The Login modal (on Apple iPhone 6)

Registration Page

The registration page will be used by general users to create new accounts. The registration form will consist of three textboxes: one for the user's name, one for the desired username, and one for the password. The form will also have five drop down menus where the user can indicate their meal preferences. The user will be able to adjust their preference for the amount of calories, sugar, sodium, protein, and calcium in their foods. The drop down menus are used to create consistency among the user preferences. If users were allowed to freely enter their preferences, answers could vary widely, making it difficult for the application to store and use the user preferences. However, with drop down menus, the preferences must be one of: Normal, High or Low. Limiting users down to these three choices allows users to express their preferences while allowing the web application to easily store and use the preferences.

After entering all of the information and registering (using the "Register" button), the user will be redirected to the main page where the user will be logged in with their name showing in the top right corner. If the user does not fill out all of the desired information, the system will prompt the user to fill in the missing information. The user will be unable to register until all information is complete. This check is to prevent users from making incomplete accounts, which would make it difficult for the system to store and use their information effectively.

The screenshot shows the registration page for a website called "DON'T EAT THAT!". The page has a header with the site name and navigation links: "About", "How It Works", and "Contact". On the right side of the header are links for "Signup" and "Login". Below the header, the title "Registration Form" is displayed. The form itself contains several input fields and dropdown menus. There are three text input fields labeled "Name", "Username", and "Password", each with a placeholder text "Enter Name", "Enter Username", and "Enter Password" respectively. Below these are five dropdown menus under the heading "Meal Preference". The dropdowns are labeled "Calories", "Sugar", "Sodium", "Protein", and "Calcium", and each currently shows "Normal" as the selected option. A blue "Register" button is positioned below the dropdowns. At the bottom of the page, there is a disclaimer: "Disclaimer: This web application does not replace a dietitian. It only serves as a quick guide to healthier food choices." Below the disclaimer is a small logo for "Powered by" followed by a logo for "WTFPL". At the very bottom, the copyright notice "© 2016 Don't Eat That!, Inc. WTFPL" is visible.

DON'T EAT THAT! About How It Works Contact Signup Login

Registration Form

Name Enter Name

Username Enter Username

Password Enter Password

Meal Preference

Calories Normal

Sugar Normal

Sodium Normal

Protein Normal

Calcium Normal

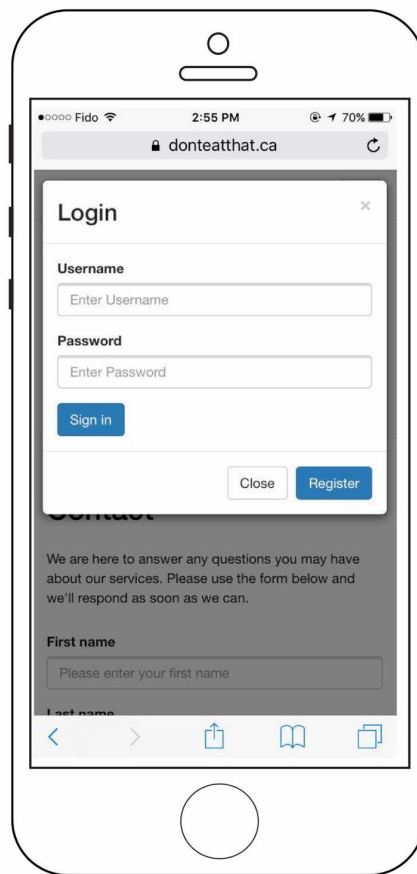
Register

Disclaimer: This web application does not replace a dietitian. It only serves as a quick guide to healthier food choices.

Powered by WTFPL

© 2016 Don't Eat That!, Inc. WTFPL

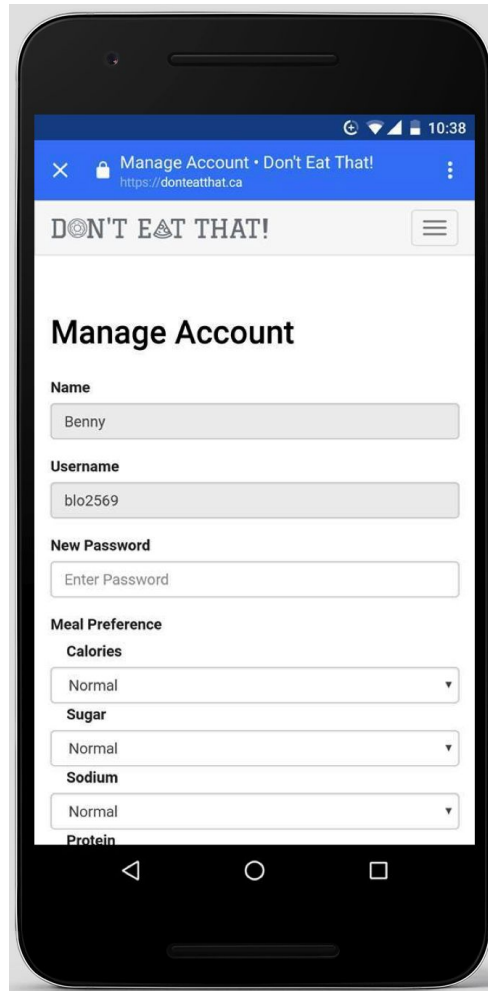
Above: Registration Page (on Google Chrome)



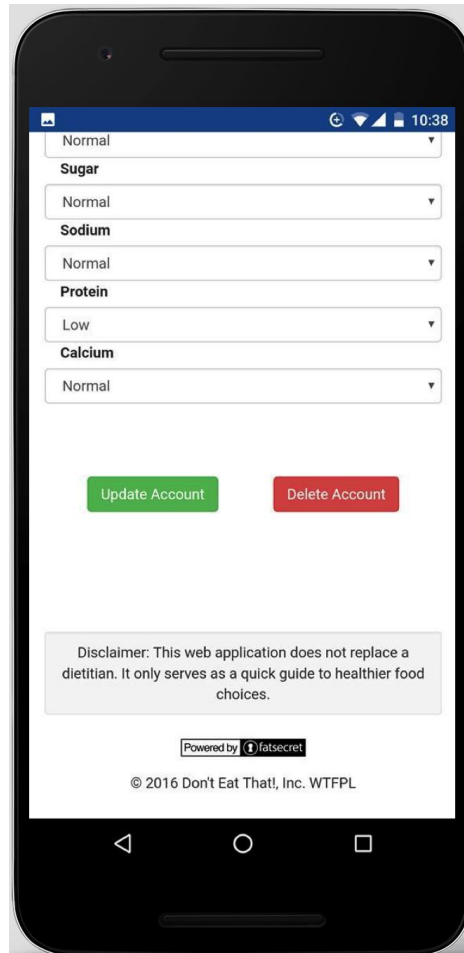
Above: Top half of Registration page (on Google Chrome - iPhone 6)

Manage Account Page

The manage account page is based off of the registration page. When a signed in user accesses the manage account page, they will see a page very similar to the registration page. This is to make editing account information easy for the user, as the user will have already seen the page and knows where to put their information. However, the input boxes for the name and username will be grayed out and inaccessible, as the user cannot change their name or username. The drop down menus will display the user's most recently saved preferences. The user can change their password and/or meal preferences by using the input box and the drop down menus. Changes will be recorded after the user presses the "Update Account" button. Alternatively, if the user wishes to delete their account, it will be deleted from the database after the user presses the "Delete Account" button and confirms by clicking "Yes" on the prompt that shows up. The "Delete Account" button is red to capture the user's attention, so they do not accidentally delete their account.



Above: Top half of Manage Account page (on Android smartphone - Google Nexus 5x)

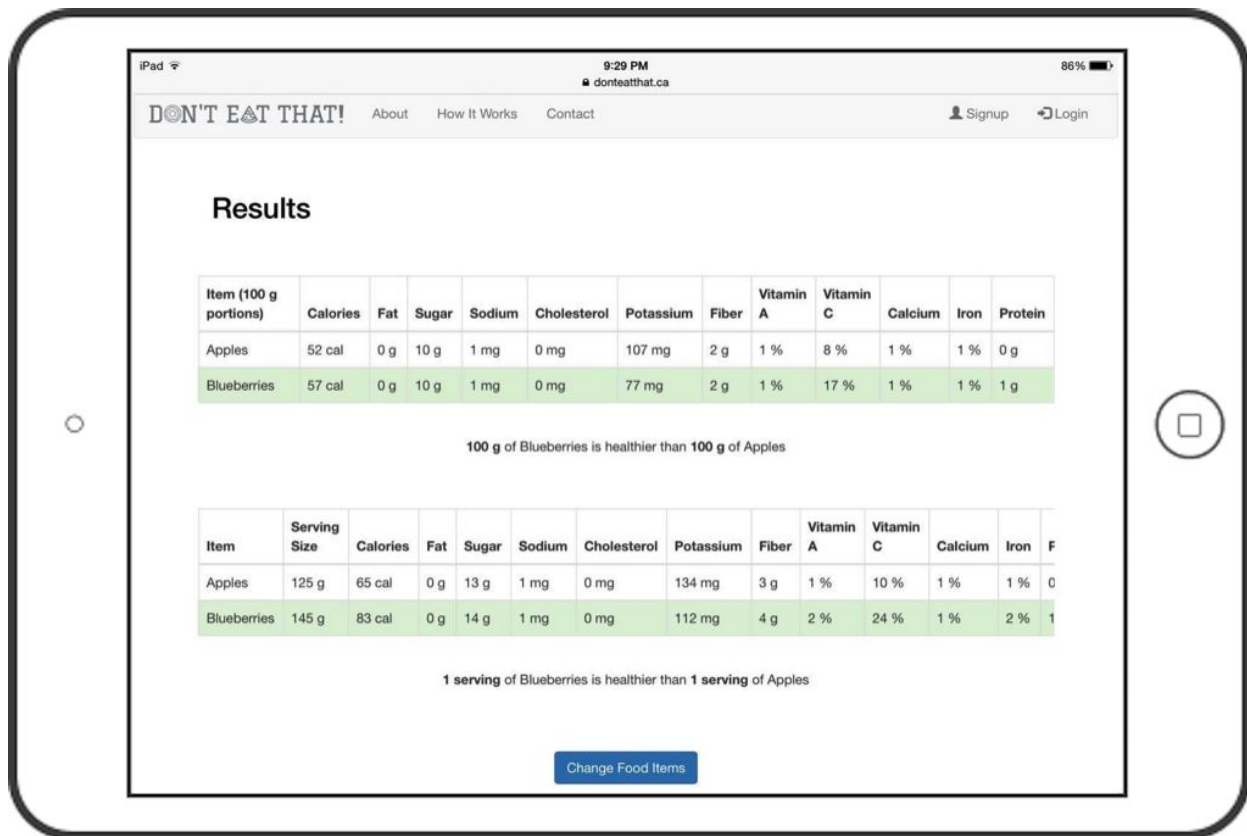


Above: Bottom half of Manage Account page (on Android smartphone - Google Nexus 5x)

Recommendation Page

The recommendation page displays the results of the user's comparison. The user will be able to see two tables; both display the amount of calories, fat, sugar, sodium, cholesterol, potassium, fiber, Vitamin A, Vitamin C, calcium, iron and protein in the user's submitted food choices. The top table displays the nutritional data of a 100 g portion. Below the top table is a recommendation that is made based on the nutritional data of a 100 g portion. The bottom table displays the nutritional data of a typical serving. Below the bottom table is a recommendation that is made based on the nutritional data of a typical serving. In both tables, the recommended choice is highlighted. Two sets of nutritional data and recommendations are used to give the user a complete understanding of their choices. We cannot use only the set of data using 100 g portions because food is not usually consumed in 100 g portions. We also cannot use only the set of data using typical portions because different portions sizes can affect the results. Thus, two sets of data are needed for a fair and complete view. Also, the tables have a scrollbar that user can use on smaller screen sizes to view the nutrition facts displayed on the table. The user

can go back to the home page to compare other food items by using the “Change Food Items” button.



Above: Results Page (on Apple iPad)