

# **Web-Based Application for Dietary Focused Consumption:**

*LooseChange.co*

Master Project

Department of Computer Science

Montclair State University

Benjamin A. Colsey

Advisor: Dr. Stefan Robila

May 2016

## Table of Contents

List of Figures .....	3
Abstract.....	4
Acknowledgment .....	5
1.0 Introduction .....	6
2.0 Background .....	8
2.1 Programming Languages and Libraries .....	8
2.2 Domain and Hosting.....	12
2.3 Programming Environment.....	13
2.4 Motivation.....	16
2.5 Scope .....	17
2.6 Similar Applications.....	17
3.0 Application Requirements and Architecture .....	21
3.1 Functional Requirements.....	21
3.2 System Requirements .....	21
3.3 Application Set-Up .....	21
4.0 Application Design .....	22
4.1 Interface Design .....	24
4.2 Data .....	25
5.0 Implementation .....	27
5.1 Final Interface Design.....	28
6.0 User Guide .....	30
7.0 Conclusion.....	32
8.0 Future Work .....	33
9.0 References .....	34
10.0 Appendix .....	37
10.1 Code .....	37
10.1.1 home.html.....	37
10.1.2 Main.html.....	38
10.1.3 javascript.js .....	45

## List of Figures

Figure 1. Mobile vs Desktop Users	Page 6
Figure 2. import.io Home Screen	Page 11
Figure 3. cPanel Home Screen	Page 12
Figure 4. cPanel Code Editor	Page 14
Figure 5. Sublime Code Editor	Page 15
Figure 6. JS Bin Home Screen	Page 16
Figure 7. Restaurant Finder Home Screen	Page 18
Figure 8. Restaurant Finder Location Based Search Results	Page 18
Figure 9. Restaurant Finder Restaurant Screen	Page 19
Figure 10. LooseChange.co Design Phase Outline	Page 22
Figure 11. LooseChange.co UI Design	Page 24
Figure 12. LooseChange.co Implementation Phase Outline	Page 27
Figure 13. LooseChange.co UI	Page 29

## Abstract

The goal of this project is to develop and implement a web-based application that provides the user with a list of food options from a restaurant of their choice based upon a set of inputs. These inputs include proximity, budget, a set of primary nutrients (protein, carbohydrates, fats, and calories) and secondary nutrients (protein, carbohydrates, fats, and calories). The user's location is found automatically and the location of the desired restaurants is displayed on a Google Maps window. The web-based application was developed using a variety of programming languages and interfaces. Simplicity and ease of use drive a comfortable user experience with expected outcomes. This web-based, mobile-centric design allows users across all platforms to access and utilize the application effectively. The application is hosted online at [LooseChange.co](http://LooseChange.co).

## Acknowledgment

The success of this project would not have been possible without the help and support of my significant other, my family, my teachers, and my friends.

I would like to extend my deepest thanks to Dr. Stefan Robila for providing me with the opportunity to not only work under his tutelage, but for his support in taking an abstract concept and bringing it to life.

I would like to thank Dr. Herman M. Dolezal for his continued support through my four years at Montclair and sitting on the committee.

I would like to thank Dr. John Jenq for sitting on the committee and for his support in my final semester.

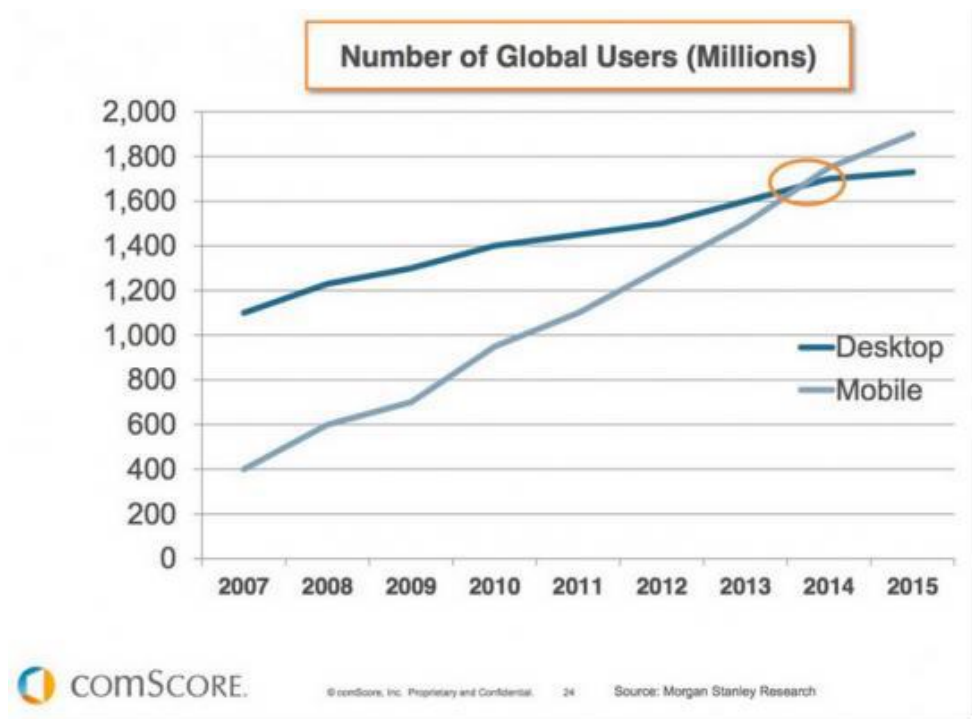
I would like to thank Mr. Mathew Faustini for providing me with his beautiful graphic designs

Lastly, I would like to thank all of the faculty and staff, past and present, during my tenure at the Department of Computer science for their guidance and support during my time in perusing my Master Degree at Montclair State University.

## 1.0 Introduction

Since the advent of the first computer, scientists have continued to make them smaller and more powerful. The computing power that previously required a very large room to house with countless numbers of subsystems in other rooms, now fits into the palm of your hand. Mobile devices are now utilizing 64-bit processors, giving them similar computational power to entry-level netbooks and laptops [1]. Migration away from the personal computer towards handheld devices is a trend exemplified in social interactions and purchasing habits [2]. In an effort to maintain relevance in the current market conditions it benefits programmers to design and implement services catered to the majority, not the minority. According to research done by Morgan Stanley, the number of mobile users eclipsed the number of desktop numbers in Q1 2014 and continues to rise. [2]

Figure 1. Trends in Mobile vs Desktop Users [2]



In addition to the increased ownership of the mobile platform, users are also spending a larger percentage of their time utilizing these devices. With overall usage exceeding fifty percent of Internet usage, it is critical to cater to this trend as it is no longer the future, it's the here and now [2]. As such, an HTML, web-based approach is the epitome of cross-platform development. Both mobile users and desktop users can effectively take advantage of this technological platform as well as enjoy a unified user experience across the multitude of platforms.

An alternative to a web-based approach would be an application designed specifically for a mobile device, such as iOS or Android. Later on, one such application is discussed in detail called Restaurant Finder. Native applications have a distinct advantage of being able to make use of all onboard sensors and can more closely dictate a user experience tailored to that device. However as in the case of Apple, there are typically multiple devices that hold a large portion of their market share. For Android there are dozens of manufacturers that bring multiple phones to market. Designing applications for each model is a large, time-consuming operation and can be plagued with problems. Managing version control across multiple platforms is a daunting task. Future updates of operating systems can cause unintended breaks and bugs in what was previously a stable platform. A web-based approach can function independently of these hardware and software cycles and provide a more flexible, stable platform with a single iteration.

## 2.0 Background

Under the broad spectrum of web-based technologies, there is a multitude of programming languages and libraries that can be utilized and tailored to provide the precise type of user experience that an application designer is attempting to emulate. Extracting the most out of these powerful tools and putting them together in a symbiotic fashion is critical for a successful, high functioning application.

### 2.1 Programming Languages and Libraries

- **HTML (Hypertext Markup Language):** is the foundation for programming websites and web-based applications. It describes the structure of web pages and allows for the design and structure of headings, text, tables, lists, and a wide variety of other content. The current version, and the one utilized in this application is HTML 5 [3].
- **CSS (Cascading Style Sheets):** is the primary means to design the graphical user interface and provide the user with an aesthetically pleasing experience. It describes the way that colors, layouts and fonts are interpreted by the web browser. A developer can customize web experiences based on screen size, and devices to ensure readability and consistent feel [3][4].
- **JavaScript :** is a cross-platform object-oriented scripting language that provides a foundation to acquire user input and perform background actions to provide the appropriate output. JavaScript has a library of objects as well as Java-Like operators, control structures, and statements. “Client-side JavaScript extends the

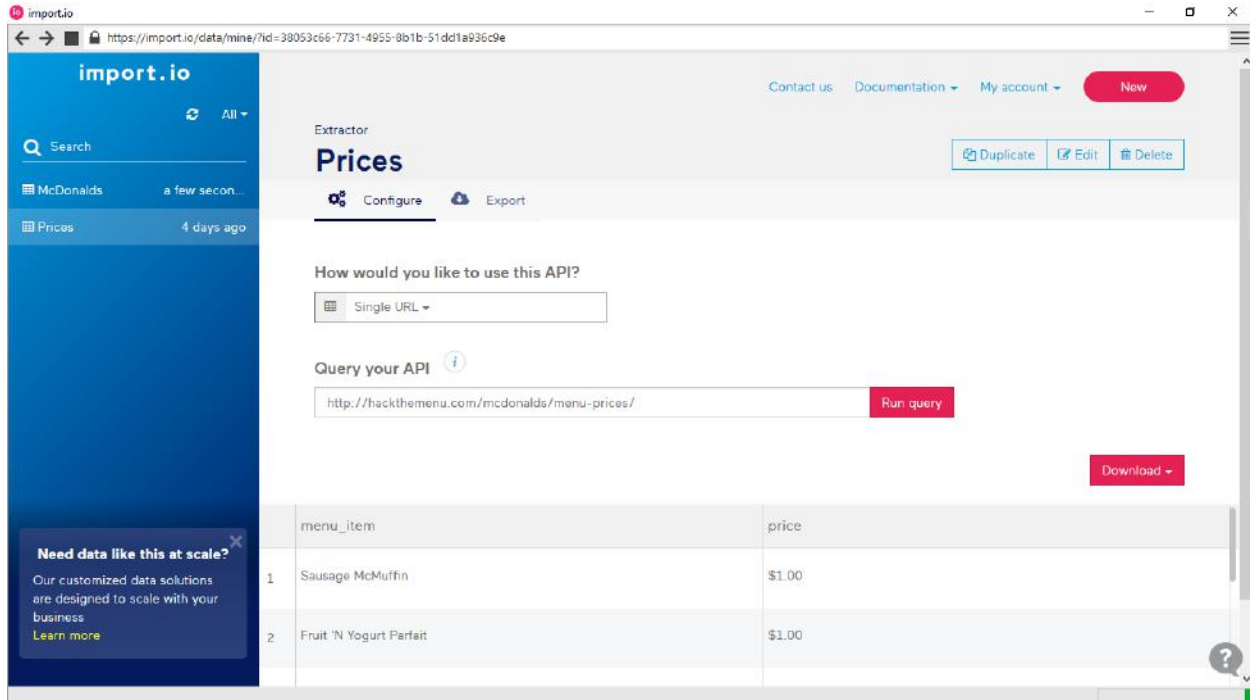


core language by supplying objects to control a browser and its Document Object Model (DOM)” [5]. This allows a developer to listen for mouse clicks or button presses and perform an associative action.

- **JQuery:** is a library of JavaScript functions that streamline commonly performed actions and assist in AJAX calls. It is a fast and feature-rich library that makes manipulation, event handling and animation significantly easier [6]. JQuery has a detailed API that makes implanting it into any project simple. It can be added to the project either through downloading the library directly and putting it into the web page folder or by calling from a hosted library such as Google’s [7].
- **JQuery Mobile:** is a library of JQuery based interface styles that are designed for mobile web-based applications. It provides a framework for highly responsive interface objects that are designed for use primarily on smart devices, but also seamlessly on desktop applications [8]. JQuery Mobile also features the “ThemeRoller” that allows developers to customize the visual user experience by changing colors, shadows, and many other visual aspects [8].
- **JSON (JavaScript Object Notation):** is a light-weight way to facilitate the exchange of data. “JSON is a text format that is completely language independent but uses conventions that are familiar” from the family of C-languages Java, and JavaScript amongst the most popular [9]. JSON is the method to format data called by other applications primarily from the server to client and can be utilized in virtually all modern programming languages [9]. JSON formatted data is what import.io returns when a query is made to its API.

- **AJAX(Asynchronous JavaScript and XML):** allows for objects within a web page to communicate with a back-end server and make updates to the data displayed to the user. The developer can “push” data to the back-end server or “pull” data based on application requirements and AJAX is the methodology that these handshakes are performed through [10].
- **Import.IO:** is an API that allows a developer to create a WebCrawler and database. Once the crawler is set up, Import.io provides a restful-based API that the developer can access via AJAX and delivers data formatted in JSON that can then be manipulated and served to a client [11]. The WebCrawler is initialized through what is referred to as an “extractor”. This extractor is set up by the developer choosing column headers for the data and then indicating to import.io what datasets from a given website the developer wants to extract. Below is a screenshot of the import.io home screen. This shows an extractor that has gathered menu items and pricing data from a website. From this screen, you can either create a new extractor, re-run the query, download the data or run the same extractor on another website to gather more data.

Figure 2. Displays theimport.io Home Screen [11]

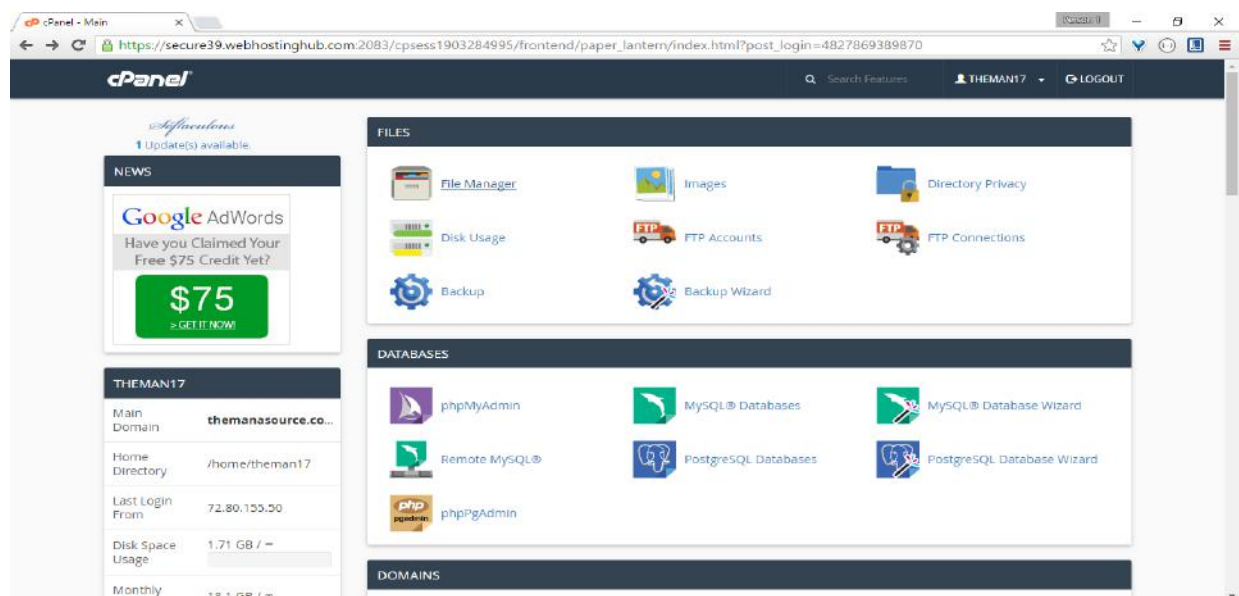


- **Google Maps API:** allows for the utilization of an interactive map feature to be implemented on the site and to be manipulated and queried on demand to provide location and proximity information to the user. Developers can change visual appearances and create a customized experience for their users [12]. For LooseChange.co, the Google Maps API is used to display the user's location and the location of restaurants within a seven-kilometer vicinity.
- **Google+ Sign-In API:** is an authentication and user data set that provides OAuth2 level authentication and gives the programmer access to profile information such as name, and home location [13]. The API is used to allow user's access to LooseChange.co.

## 2.2 Domain and Hosting

For users to access web-based applications, these web pages must be stored on a publicly accessible web server. Developers purchase domain names that redirect the user to the web server through a Domain Name Server (DNS). For the purposes of this project, the web server host that is being utilized is Web Hosting Hub. All of the files required to serve the web pages to the end user are stored on Web Hosting Hubs' servers. Web Hosting Hub utilizes a Linux-based web hosting control system and user interface. It allows the end user to have full control over all necessary aspects of managing a website [14]. cPanel turns a complicated series of actions into a simplified user experience with clean site lines and simple instructions. The two most utilized features of CPanel, for the purposes of this project, were the file transfer protocol (FTP) and the code editor. The FTP segment of CPanel is where the developer uploads web-page files. Information on the code editor will be covered in section 2.3. Figure 3 displays the cPanel home screen. From here, the developer can manipulate domains and files stored on the cPanel web hosting server.

Figure 3. cPanel Home Screen [15]



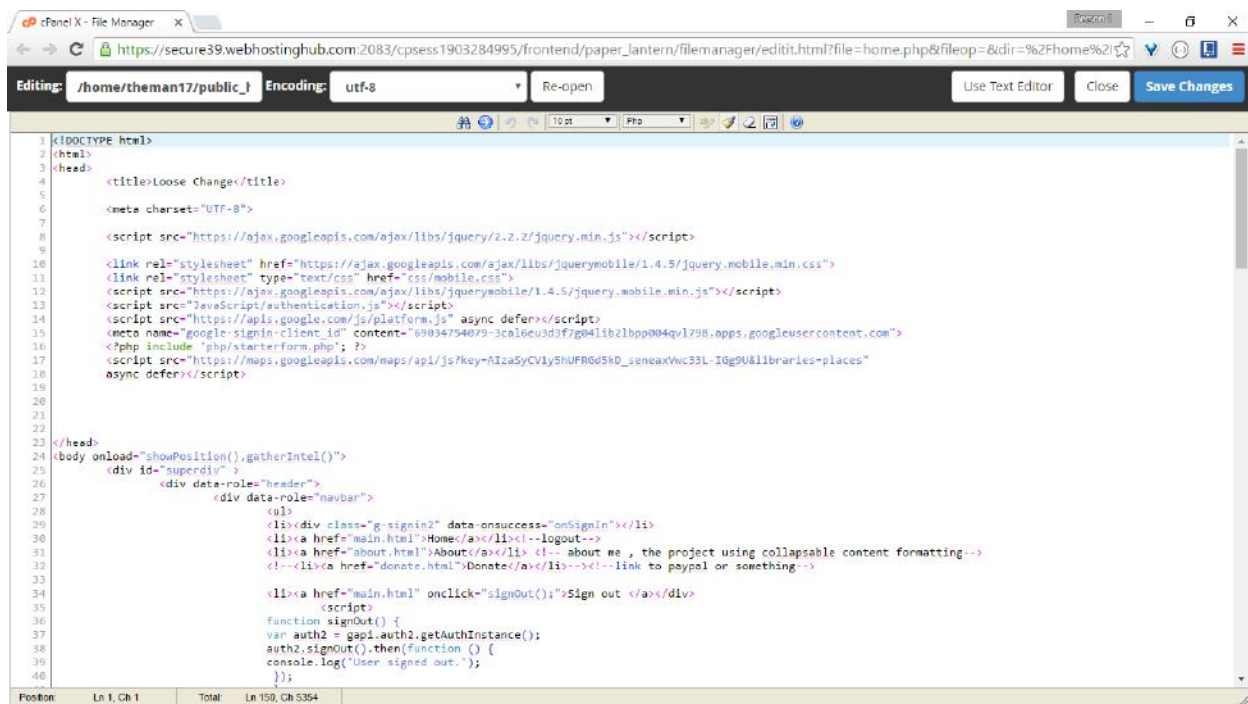
The domain name for this project was purchased from GoDaddy.com, which is a domain name register company that allows developers to purchase domain names with any existing extension and includes a variety of web hosting services. To facilitate communication between the back-end web hosting server with Web Hosting Hub and the Domain Name Register GoDaddy, it was necessary to change the primary DNS records for LooseChange.co domain name on GoDaddys server to point to the Web Hosting Hub servers. This provides direct access for all traffic attempting to access the LooseChange.co domain to be directed to the appropriate information.

## **2.3 Programming Environment**

All of the aforementioned languages and libraries can be written in any text editor, even the readily available Notepad. However, to create this project, cPanel Code Editor, Sublime 3, and JS Bin were utilized to their fullest extent.

The cPanel code editor is a simple code editor used to make changes to program files stored in the cPanel FTP server. Figure 4 provides a screenshot of the editor. The editor is provided to developers at no additional charge as long as they have a hosting account through the Web Hosting Hub and has little overhead with minimal features to assist developers. All changes made to files must be saved via the Save button in the top right part of the code page. Once saved, changes will appear immediately on the web page that you made the changes to.

Figure 4. cPanel Code Editor [15]



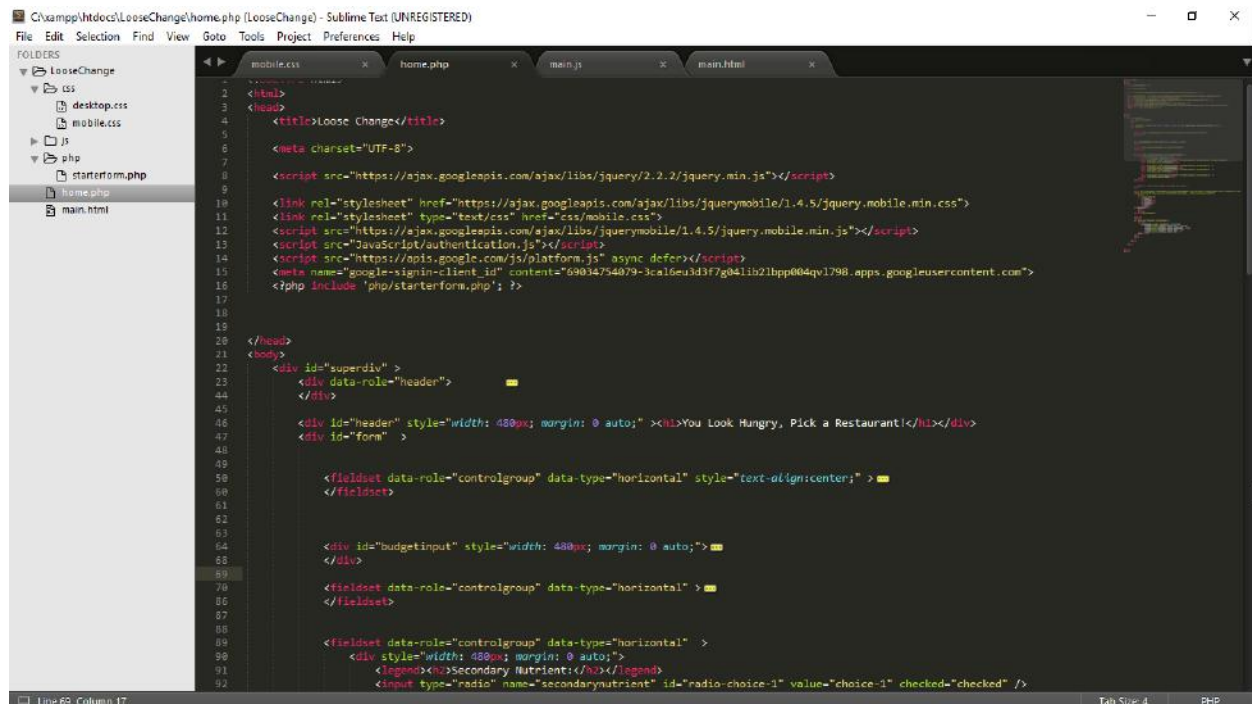
The screenshot displays the cPanel File Manager's code editor. The browser address bar shows the URL: `https://secure39.webhostinghub.com:2083/cpsess1903284995/frontend/paper_lantern/filemanager/editit.html?file=home.php&fileop=&dir=%2Fhome%2F`. The editor's status bar indicates the file path is `/home/theman17/public_` and the encoding is `utf-8`. The code being edited is an HTML document with the following structure:

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>loose Change</title>
5
6   <meta charset="UTF-8">
7
8   <script src="https://ajax.googleapis.com/ajax/libs/jquery/2.2.2/jquery.min.js"></script>
9
10  <link rel="stylesheet" href="https://ajax.googleapis.com/ajax/libs/jquerymobile/1.4.5/jquery.mobile.min.css">
11  <link rel="stylesheet" type="text/css" href="css/mobile.css">
12  <script src="https://ajax.googleapis.com/ajax/libs/jquerymobile/1.4.5/jquery.mobile.min.js"></script>
13  <script src="JavaScript/authentication.js"></script>
14  <script src="https://apis.google.com/js/platform.js" async defer></script>
15  <meta name="google-signin-client_id" content="69034794079-3ca16eu3d3f7g041b2lbp004qvl798.apps.googleusercontent.com">
16  <?php include 'php/starterform.php'; ?>
17  <script src="https://maps.googleapis.com/maps/api/js?key=AIzaSyCV1y5hUFR6d5k0_seneaxVnc33L-IQg9U&libraries=places"
18    async defer></script>
19
20
21
22
23 </head>
24 <body onload="showPosition(),gatherIntel()">
25   <div id="superdiv">
26     <div data-role="header">
27       <div data-role="navbar">
28         <ul>
29           <li><div class="g-signin2" data-onsuccess="onSignIn"></li>
30           <li><a href="main.html">Home</a></li><!--logout-->
31           <li><a href="about.html">About</a></li> <!-- about me , the project using collapsable content formatting-->
32           <!--<li><a href="donate.html">Donate</a></li>--><!--link to paypal or something-->
33
34           <li><a href="main.html" onclick="signOut()">Sign out </a></div>
35         </ul>
36         <script>
37           function signOut() {
38             var auth2 = gapi.auth2.getAuthInstance();
39             auth2.signOut().then(function () {
40               console.log("User signed out.");
41             });
42           }
43         </script>
44       </div>
45     </div>
46   </div>
47 </body>
48 </html>
```

The status bar at the bottom shows the cursor position as `Ln 1, Ch 1` and the total file size as `Ln 150, Ch 5354`.

Sublime 3 code editor is an advanced text editor that is a direct download from the publisher's site [16]. Sublime is extremely user-friendly and provides an auto-complete feature that makes utilizing this development tool refreshing to use over the basic code editor of CPanel. As apparent in the screen capture below (Fig. 5), the user interface provides a strong contrast between the dark background color and the light colored text to reduce eye strain even after hours of use.

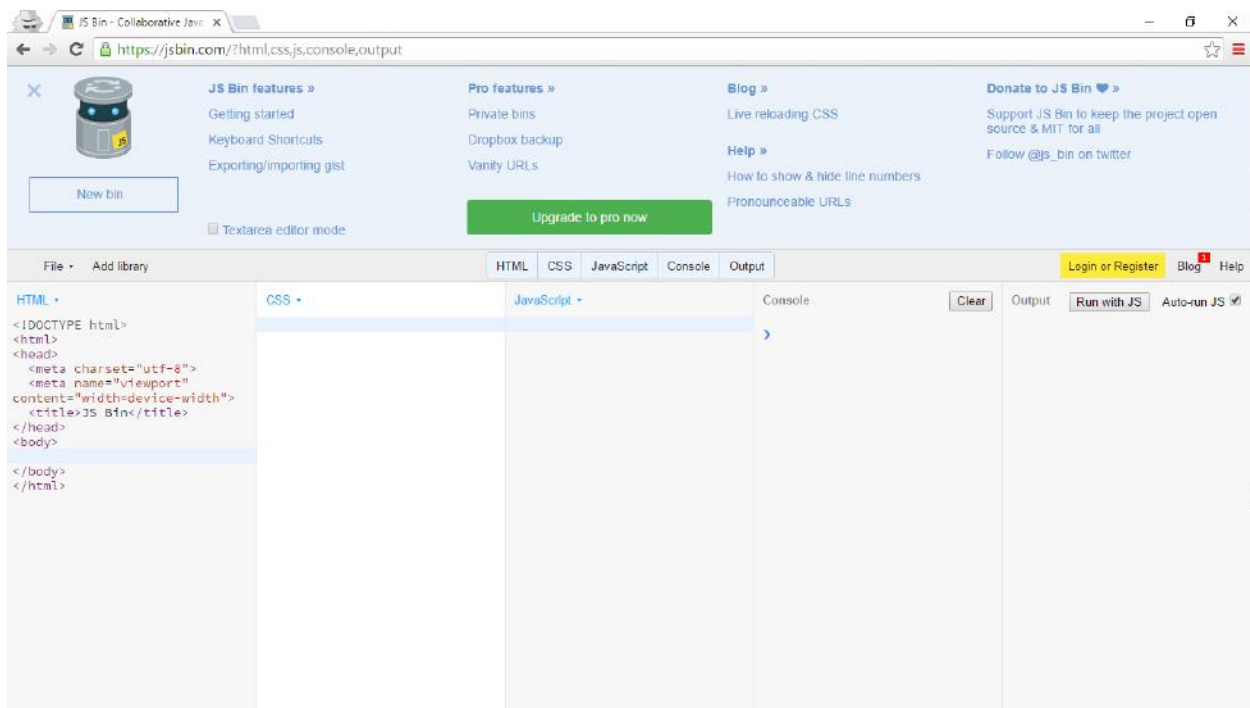
Figure 5. Sublime Code Editor [16]



On the right-hand side of the user interface is a condensed panel that shows all of the code in the current document that the user is working on. The gray square indicates where the section in view is highlighted. This feature is extremely useful when shifting between large sections of code and eliminates a lot of random scrolling to search for the next section. Sublime 3 will be the go-to text editor of choice on all future projects, it is that effective.

Prior to the start of this project, JS Bin was a developer tool unknown to me that became invaluable as the project began to take shape. Access to JS Bin is done through the JS Bin website ([www.JSBin.com](http://www.JSBin.com)) [17]. JS Bin assists in debugging JavaScript (JS) code by providing areas to insert the developing code into a manipulative environment where output can be changed on the fly and allow for real-time, instantaneous debugging of active JavaScript.

Figure 6. JSBin Home Screen [17]



## 2.4 Motivation

Concerns over nutrition and exercise are increasingly prevalent and at the forefront of socioeconomic trends. A recent change in dietary needs and budgetary concerns led to the idea of creating an application that handles all the guess work of ordering a meal with the focus on specific nutrients or calories. The ability to accomplish this goal with a simple and easy to use application became a passion. The application, named Loose Change, became synonymous with taking what money was available in order to acquire a meal that met a specific set of criteria. Whether that criterion is calories, carbohydrates, proteins, or fats, it was essential to be able to distinguish between them as they each play a key role in a number of different specialized diets or taken collectively.



## 2.5 Scope

Establishing a concrete set of objectives and guidelines prior to the start of this project was key in cementing a functioning application that met all of the objectives. In order to ensure the proper depth, for the purposes of this project, two common restaurant chains were chosen: McDonalds and Subway. These two particular establishments were chosen because of their popularity, enormous breadth of locations, and availability of their menu pricing and nutritional information on websites that could be crawled for that data. In the future, additional restaurant chains will be added.

Menu items contain many nutrients that are usually posted in the nutrition facts on the back label of the packaging or on the restaurant's website. The use of protein, carbohydrates, calories, and fats encompass the primary focus of diets. If the user is trying to track overall calorie consumption, hit certain protein goals for muscle growth, stay under a certain amount of carbohydrates or keep unhealthy fats in their diet to a minimum, locating menu items that match those criteria can be done in a simple manner using the application.

## 2.6 Similar Applications

When researching similar apps to LooseChange.co, only one was on par with similar functionality and that was the Restaurant Finder mobile app. The first thing that was apparent were the extremely intrusive advertisements. Before having the opportunity to utilize the application, it prompted the user to purchase the \$4.00 “pro version” that would remove all the advertisements. This was extremely frustrating and made using the application a poor experience. Once those Google Play Store purchase windows were closed out, another full-

screen advertisement appeared. Also, every time the application was closed and opened back up, another full-screen advertisement would appear.

Figure 7. Restaurant Finder Home Screen [18]

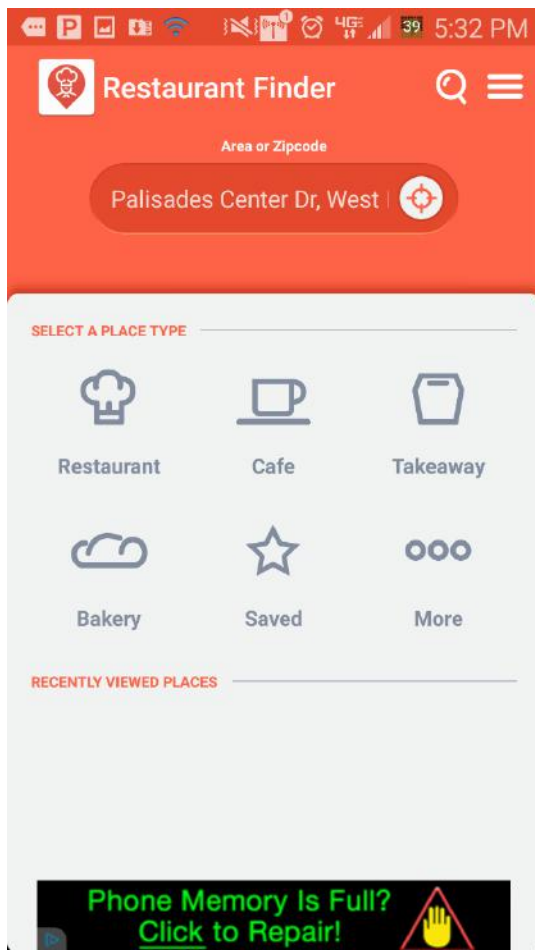
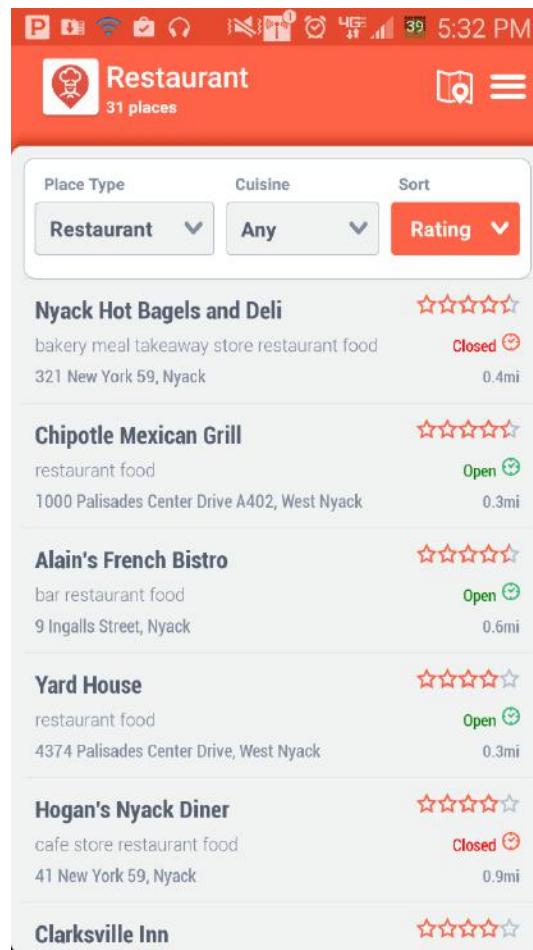


Figure 8. Restaurant Finder Location Based Search Results [18]

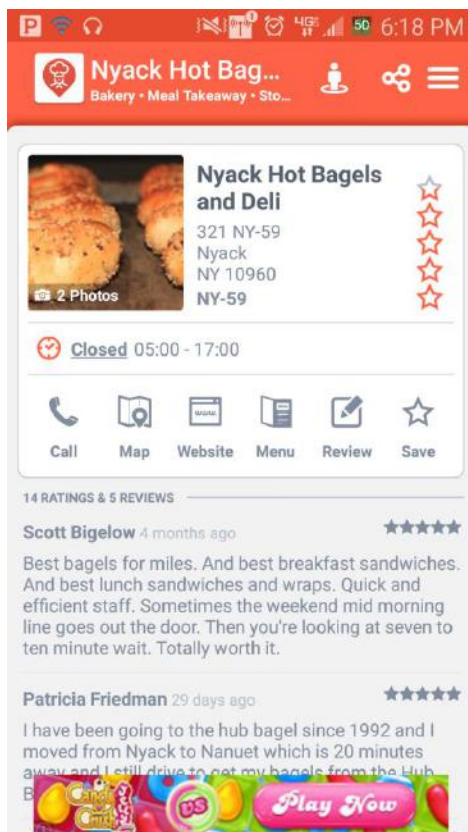


The screen capture above on the left (Fig. 7) is the landing screen when the app is first opened (after navigating past the advertisements). Similarly to LooseChange.co, the Restaurant Finder mobile application asks the user for location data either by pressing the “target” icon on the upper right-hand portion of the screen or by entering a desired location in the text input field to the left of the icon. Once the location is selected the application then prompts the user to select

the type of restaurant that they would like to find. The choices are Restaurant, Café, Takeaway (take-out), Bakery, Saved, or More.

After the location and the type of restaurant are collected from the user, the application then generates a list of businesses that match that criterion. This data is organized by customer rating and by proximity to the user's location. From this screen, it is also possible to change the type of restaurant and the location. Once a restaurant selection is made, the application takes the user to a screen that shows relevant information about it including address, hours of operation, and quick links to call them directly. It also includes their location on a map, the restaurants business website, reviews and the option to add the restaurant to the users lists of favorites. Underneath that is a list of reviews of various ratings that provide the user with a general feel of the restaurant and its overall popularity, whether that's good or bad.

Figure 9. Restaurant Finder Screen [18]



The general purpose of the Restaurant Finder application is to provide users with information and reviews on restaurants in their vicinity versus the LooseChange.co approach of providing meal choices based on a set of user nutrition requirements. The target audience of both applications is similar and the use of maps and location-based proximity to restaurant locations are the same, but the final purpose of the applications vary and ultimately are different enough to not be directly competing with one another.

## 3.0 Application Requirements and Architecture

### 3.1 Functional Requirements

In order to access the application, the user needs to have access to the internet. In order to receive the maximum benefit that the application can provide, the user must have location based services enabled on their device and/or browser.

### 3.2 System Requirements

In order to access a website, the user is required to have a web browser that facilitates their access to the World Wide Web. It is always recommended that for security and functionality purposes users have installed the most up-to-date versions of their respective browsers. LooseChange.co relies on Google Authentication to allow users to the site and because of this, it is recommended that Google Chrome be used for primary access. The application works just as well on other major browsers such as Safari and Mozilla FireFox. Refer to the appendix for browser compatibility testing.

### 3.3 Application Set-Up

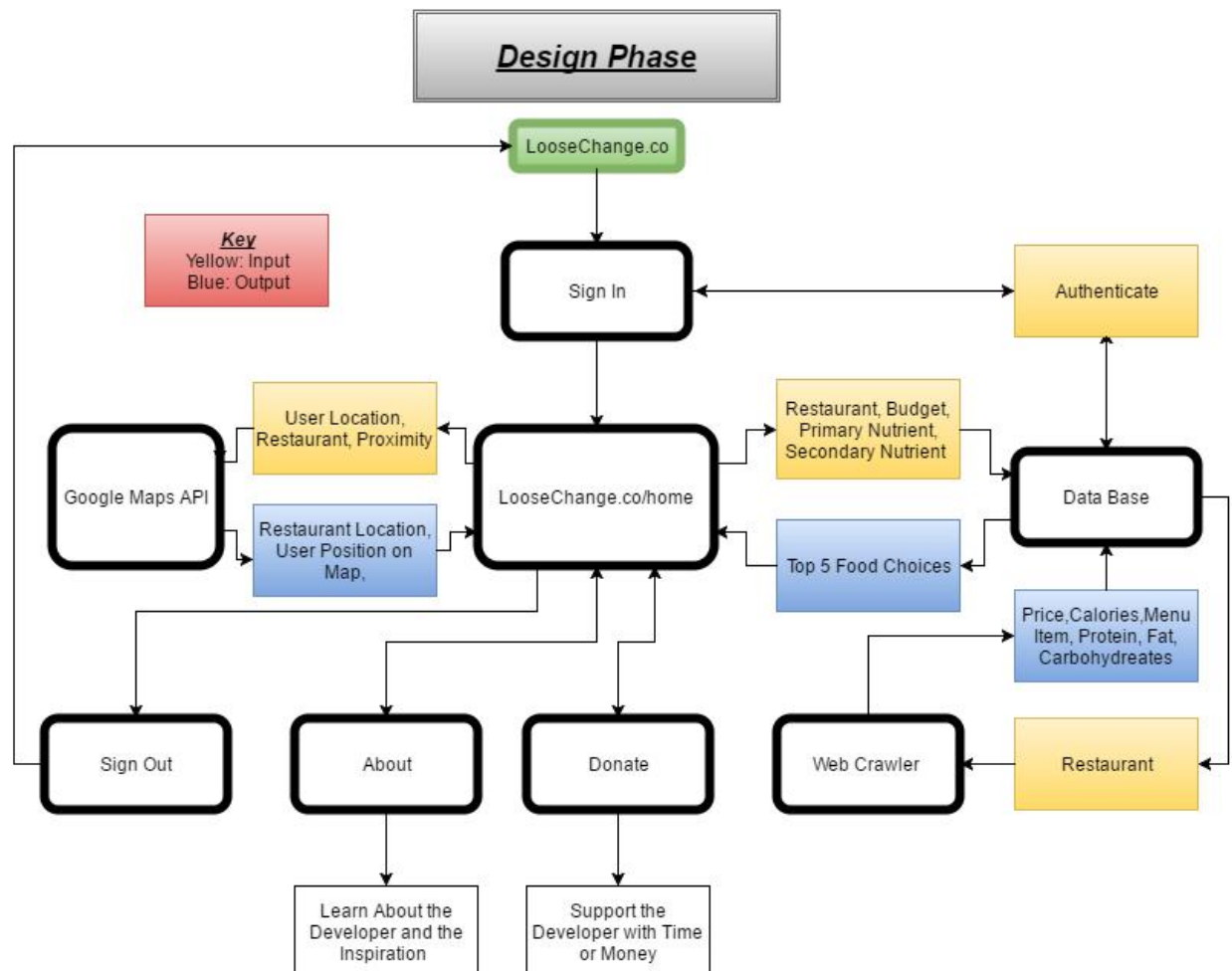
All work required to set up this project for use by an end user has been completed by the developer. If the end user would like to duplicate this application on another server they would need to have access to a web server or host one themselves. In order to use the application, the end user simply needs to go to the site domain LooseChange.co and proceed to use the application from there.

## 4.0 Application Design

We, as a society, are extremely data conscious. This approach utilizes minimal data on the part of the user. They are not required to download an app that would need constant updates with the latest information necessary for a fulfilling user experience. In addition, any queries to information are done by the server and then only the relevant information is delivered to the user.

Below is a web diagram of how the website's initial design phase intended operations to occur:

Figure 10. LooseChange.co Design Phase Outline



It is intended that the user access the site via the primary domain name of [www.LooseChange.co](http://www.LooseChange.co). In the first step, the user will be redirected to the landing page of

/main.html and the website will prompt the user to authenticate to the application after they have created an account that is stored on the database.

Once they have authenticated against the database and/or created an account, the website allows them to then access the primary web page of /home.html. At the top of this page will be a horizontal menu bar with the following options:

- **Home:** On this web page they will be shown a list of radio buttons where they can select the restaurant where they would like to eat. Once they have made a selection, the Google Map will show them all of the restaurants of that name that are within their immediate vicinity. Underneath the map, the user will be prompted to enter in a budget through a series of plus and minus buttons. The minus will be on the left with the total in the middle and a plus button on the right to allow for precise user input through a straight forward mechanism. Underneath the budget input, there will be a set of duplicate radio buttons that are differentiated by the headers of primary and secondary. These will be the “focus” of the query and will instruct the application what foods will be listed in the table that will appear just below the secondary nutrient radio button. The home option will reset the query and start the Loose Change process over.
- **About:** the “About” page that will give a brief summary of the developer of the application and a link to the final submission of the project and its proposed future.
- **Donate:** the “Donate” option that will bring the user to the donate page. When the user navigates to the donation menu option in the horizontal menu bar at the top of the Loose Change home page they will be given the following options: The first is through a donate button that allows users to donate via PayPal to the developer. The second is the option to watch a short advertisement delivered to the user via Google AdSense that would then

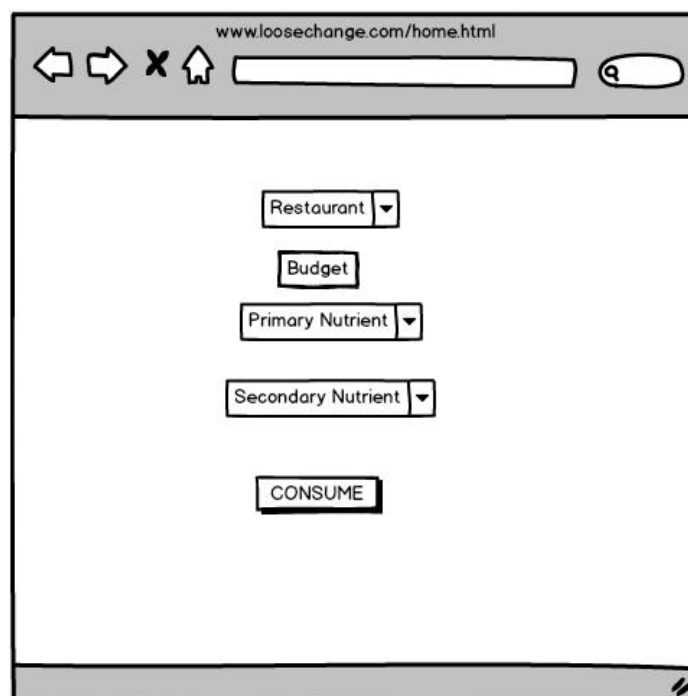
generate a small amount of revenue for the developer. This gives the user a choice, they can either donate their time 15-30 seconds or their money, whichever they decide is more valuable to them.

- **Signout:** allows the user to sign out of the website and return the user to the Loose Change landing page.

## 4.1 Interface Design

The goal of the interface of LooseChange.co was to create a simple, straight forward and easy-to-use graphical interface allowing the user to create a clear path to completion. All of the screens are center-aligned with automatic margins to provide a clear and consistent visual appearance across a wide variety of screen sizes, shapes, and pixel densities. The minimalistic buttons and options provide a clear path to completion for the user. Buttons and text are aligned top to bottom and left to right to allow for a comforting and familiar user experience.

Figure 11. LooseChange.co UI Design





In the initial design of the LooseChange.co called for a very simplistic user interface with no implementation of google maps. The top of figure 11 shows a drop down menu of the restaurant choices where the user would be able to choose from a list of predefined choices. Below that is a text input box for the end user to enter in their budget. Next, two drop down menus of nutrients, both containing calories, fats, proteins, and carbohydrates, allow the user to make a selection. At the bottom is a button indicated as “consume” that would then take the user input, and run that information as a query against the database.

## 4.2 Data

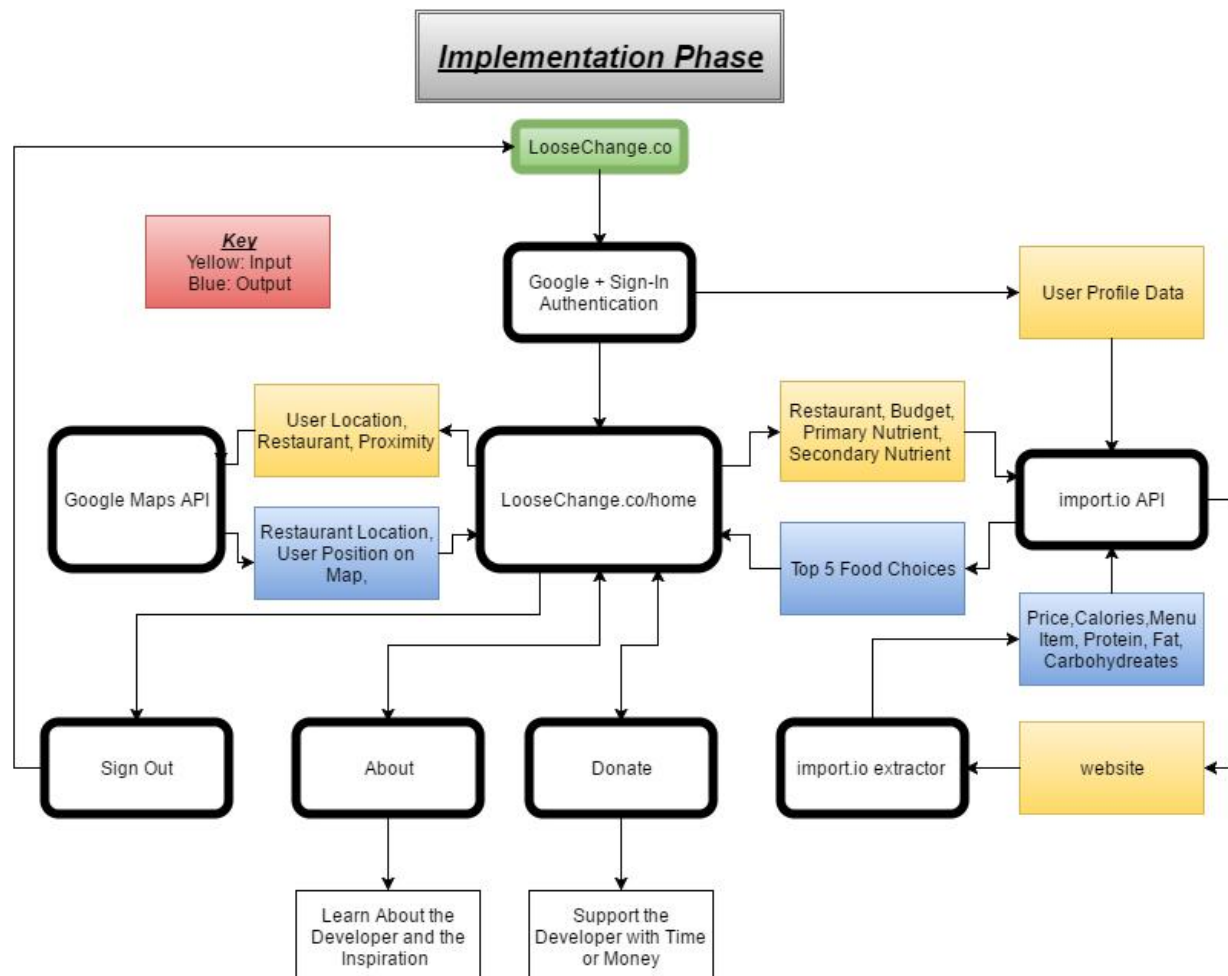
The acquisition of accurate and up to date menu information is important for the functionality of LooseChange.co. Most chain restaurants have their menu information online but as in the case with McDonalds and Subway, it is in a format that doesn’t lend itself well to web crawling. The use of secondary sources was necessary to gather all of the required information. The import.io extractor is configured to access three websites for each restaurant in order to access all of the required data [19][20][21]. This data is combined on the server side of LooseChange.co by calling each of the import.io extractor APIs and combining the data sets into an array. Once the data is combined, each menu item has a ratio derived for each of the nutrients by dividing the nutrient amount by the cost of the menu item. This ratio is used to rank the menu item choices and is the determining factor for what menu item choices appear when the user enters in their parameters. To add additional restaurants to LooseChange.co, additional information sources would need to be added and extractors set up for each of the additional

options and linked with the user interfaces. Each of the additional restaurants data would need to be configured and aligned to allow the application to function properly.

## 5.0 Implementation

One of the major changes from the design phase to the implementation phase of the project was the discovery of a tool known as import.io [11]. This tool is a web-based and standalone application available on Mac OS, Windows and Linux that allows the user to enter in to a website, select the data from that website that they want, and then import.io generates an API that allows you to call against it in any application using the RESTful API.

Figure 12. LooseChange.co Implementation Phase Outline



The initial planning stage called for the use of a custom created back-end web crawler and subsequent database in order to collect the information from component websites and

combine it to serve up to the LooseChange.co web page upon user request. This changed after an investigation on the subject matter of crawlers and when the discovery of import.io was made.

The integration of Google Maps into the project was a necessary feature for the continuity of the project, but one that was met with a steep learning curve. Once mastered, however, it was easy to manipulate the Google Map API to provide a shockingly customizable map experience. Every time the user presses the radio button at the top of the home.html, a query is submitted to the Google Map API to search for the restaurant of their choosing within a preset radius of seven thousand meters. This distance was set because it equated to roughly four miles and in a suburban environment with a point-to-point distance it is likely that one of the predefined restaurants would fall within that radius.

## **5.1 Final Interface Design**

The final user interface is very similar to the initial design with a few changes that allow for easier user input. Instead of having a slide for an end user to input the amount of money they have in their budget, there is an input box that when used on a mobile device, prompts the user with a number only keyboard to make entering a number more concise. The user interface takes full advantage of the JQuery Mobile API for styling effects by creating large, rounded stylized buttons and text input fields for an aesthetically pleasing and simple experience.

Figure 13 shows the completed user information for the LooseChange.co platform. At the top is the option to sign into the users Google Account (if they have not done so on the initial landing page) as well as the home, about and signout options. Below that is a welcome banner that draws the user into and attempts to set a positive mood. The drop down menu choices that where in the initial design are now radio buttons that provide the user with a clearer choice of what they need to decide upon. The map shows the location of the user and then the nearest

restaurant on the map with an arrow indicating its approximate location. A number of different design iterations of how to enter in the budget where considered but the ultimate decision was the input button that prompts mobile users with a number only keyboard when they click on it. Both the primary and secondary nutrient dropdown menus where replaced with radio buttons for a visual change and a clearer path to completion for the user. Below the nutrients where the submit button existed in the initial design was exchanged for a dynamically modified table of menu options that change whenever the user updates a field or button.

Figure 13. LooseChange.co UI

[Sign in](#)
[Home](#)
[About](#)
[Sign out](#)

## You Look Hungry!

Choose a Restaurant:

Mcdonalds
Subway

Budget:

Primary Nutrient:

Calories
Carbohydrates
Protein
Fat

Secondary Nutrient:

Calories
Carbohydrates
Protein
Fat

Columns to display...

Item	Serving	Protein	Carbs	Fat	Calories
Hamburger	3.7 oz	13	33	9	260
Cheeseburger	4.2 oz	15	35	12	310
Double Cheeseburger	6.1 oz	25	37	23	460

## 6.0 User Guide

```
/*                Welcome to LooseChange.co!                */
/*                User Guide                                */
/*                version: 0.33 Beta                          */
```

This is the functional user guide that will get you on your way to a meal in no time!

- 1) Enable Location Based Services (if using a smart device);
- 2) Open your preferred browser;
- 3) Type [www.LooseChange.co] into the address bar;
- 4) Log in using Google authentication;
- 5) Select the restaurant you would like to eat in;
- 6) Enter your budget;
- 7) Select the primary nutrient you would like;
- 8) Select a secondary nutrient if desired;
- 9) Tell the clerk which of the choices you would like;
- 10) Enjoy!;

### Q&A

Q: Why is there a gray square in the middle of my screen?

A: In order for the application to show your location on the map and the restaurants in your vicinity, you must enable location based services.

Q: Do I need a Google account to access LooseChange.co's magical ways?

A: Yes, in the current version a Google account is required but other forms of authentication will be available soon.

Q: Do I need to donate?

A: Nope! LooseChange.co is free to use. If you would like to support the developer through watching advertisements or a direct donation that option is available, but not at all necessary!

## 7.0 Conclusion

The development of this project was a long tedious endeavor that ultimately yielded extremely promising results with a bright outlook on its continued development outside the current scope and timeframe. All of the goals initially put forth at the beginning of this project have been accomplished. One of the more challenging issues faced when dealing with a project that has such large possibilities is managing the scope and staying within a series of predefined boundaries in order to stay on task and not get lost in features but instead, ensure a functioning product. Through the research required to accomplish LooseChange.co, a number of useful tools were discovered; import.io and JS Bin add to my knowledge base for any future projects.



## 8.0 Future Work

The future of this project relies primarily on its ability to grow in the number of restaurants that the user has access to, and in doing so, greatly expanding the amount of restaurant data that is utilized and accessed. This can be paired with utilizing the Google+ signin and authentication to accrue anonymous user data about location-based eating habits, matched with budgetary and availability trends, which would be useful to corporations in plotting future expansion of locations to ensure maximum profitability. This will be accomplished by leveraging existing cloud technologies and infrastructure while maintaining a minimalist user experience. The site can be self-sustaining by allowing users to choose if and how they would like to support the site. Either through a direct, anonymous donation or by watching a short advertisement on their time if they choose to do so.

Fiscal responsibility for maintaining the domain and hosting the website will be mine and the website will remain active for a minimum of a year after the completion of this project.

## 9.0 References

- [1] Anonymous ABI research; Mobile Device Platforms Continue Trend Towards More Integration Despite Apple and Samsung Skewing the Market, says ABI research. *Telecommunications Weekly* pp. 105. 2013. Available: <http://search.proquest.com/docview/1449111790?accountid=12536>
- [2] D. Chaffey, "Mobile Marketing Statistics 2016", *Smart Insights*, 2015. [Online]. Available: <http://www.smartinsights.com/mobile-marketing/mobile-marketing-analytics/mobile-marketing-statistics/>. [Accessed: Apr-2016]
- [3] "HTML & CSS - W3C", *W3.org*, 2014. [Online]. Available: <https://www.w3.org/standards/webdesign/htmlcss>. [Accessed: 06- May- 2016].
- [3] "Hosted Libraries", *Google Developers*, 2016. [Online]. Available: <https://developers.google.com/speed/libraries/#libraries>. [Accessed: 02- May- 2016].
- [4] "CSS", *Mozilla Developer Network*, 2016. [Online]. Available: <https://developer.mozilla.org/en-US/docs/Web/CSS>. [Accessed: 06- May- 2016].
- [5] "Introduction to JavaScript", *Mozilla Developer Network*, 2016. [Online]. Available: <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Introduction>. [Accessed: 06- May- 2016].
- [6] jquery.org, "jQuery API Documentation", *Api.jquery.com*. 2016. [Online]. Available: <https://api.jquery.com/>. [Accessed: Apr-2016].
- [7] "Hosted Libraries", *Google Developers*, 2016. [Online]. Available: <https://developers.google.com/speed/libraries/#libraries>. [Accessed: Apr- 2016].
- [8] jquerymobile.org, "jQuery Mobile API Documentation", *Api.jquerymobile.com*, 2016. [Online]. Available: <https://api.jquerymobile.com/>. [Accessed: Apr-2016].

- [9]"JSON", *Json.org*, 2016. [Online]. Available: <http://json.org/>. [Accessed: 07- May- 2016].
- [10]"Ajax", *Mozilla Developer Network*, 2015. [Online]. Available:  
<https://developer.mozilla.org/en-US/docs/AJAX>. [Accessed: 09- May- 2016].
- [11]"Import.io | Web Data Platform & Free Web Scraping Tool", *Import.io*, 2016. [Online].  
Available: <http://import.io>. [Accessed: 05- May- 2016].
- [12] "Google Maps JavaScript API \_\_\_\_ Google Developers", *Google Developers*, 2016.  
[Online]. Available: <https://developers.google.com/maps/documentation/javascript/>.  
[Accessed: Apr-2016].
- [13] "Google+ API", *Google Developers*, 2016. [Online]. Available:  
[https://developers.google.com/+/web/api/rest?hl=en\\_US](https://developers.google.com/+/web/api/rest?hl=en_US). [Accessed: Apr-2016].
- [14]"cPanel Inc. | The Hosting Platform of Choice", *Cpanel.com*, 2016. [Online]. Available:  
<http://cpanel.com/>. [Accessed: 09- May- 2016].
- [15]"Best Website Hosting Services | Web Hosting Hub", *Webhostinghub.com*, 2016. [Online].  
Available: <http://webhostinghub.com>. [Accessed: 05- May- 2016].
- [16]"Sublime Text - Download", *Sublimetext.com*, 2016. [Online]. Available:  
<http://www.sublimetext.com/3>. [Accessed: 02- May- 2016].
- [17]"JS Bin", *Jsbin.com*, 2016. [Online]. Available: <http://jsbin.com>. [Accessed: 02- May- 2016].
- [18]"Restaurant.com | Mobile", *Restaurant.com*, 2016. [Online]. Available:  
<http://www.restaurant.com/mobile>. [Accessed: 02- May- 2016].
- [19]"Calorie Counter (CalorieLab)", *Calorielab.com*, 2016. [Online]. Available:  
<http://calorielab.com>. [Accessed: May- 2016].
- [20]"#HackTheMenu - The Ultimate List of Secret Menu Items", *#HackTheMenu*, 2016.  
[Online]. Available: <http://hackthemenu.com>. [Accessed: May- 2016].

- [21]"Fast Food News, Deals, and Reviews | FastFoodWatch.com", *Fast Food Watch*, 2016.  
[Online]. Available: <http://www.fastfoodwatch.com>. [Accessed: May- 2016].
- [22] Api.docs.import.io. 2016 [Online]. Available: <http://api.docs.import.io/>. [Accessed: Apr- 2016].
- [23]"OverAPI.com | Collecting all the cheat sheets", *Overapi.com*, 2016. [Online]. Available: <http://overapi.com>. [Accessed: 02- May- 2016].

## 10.0 Appendix

### 10.1 Code

#### 10.1.1 home.html

Subject to change

```
<!DOCTYPE html>

<html>

<head>

  <title>Loose Change</title>

  <link rel="stylesheet" type="text/css" href="css/mobile.css">

  <script src="https://apis.google.com/js/platform.js" async defer></script>

  <meta name="google-signin-client_id" content="69034754079-
3cal6eu3d3f7g04lib2lbpp004qv1798.apps.googleusercontent.com">


</head>

<body>

  <div id="superdiv">

    <div id="header"><h1>Landing</h1></div>


    <div id="login">

      <div class="g-signin2" data-onsuccess="onSignIn"></div>

      <a href="home.php">Login</a>

    </div>

  </div>
```

</div>

</body>

</html>

### 10.1.2 Main.html

<!DOCTYPE html>

<html>

<head>

<title>Loose Change</title>

<meta charset="UTF-8">

<script src="https://ajax.googleapis.com/ajax/libs/jquery/2.2.2/jquery.min.js"></script>

<link rel="stylesheet"  
href="https://ajax.googleapis.com/ajax/libs/jquerymobile/1.4.5/jquery.mobile.min.css">

<link rel="stylesheet" type="text/css" href="css/mobile.css">

<script  
src="https://ajax.googleapis.com/ajax/libs/jquerymobile/1.4.5/jquery.mobile.min.js"></scrip  
t>

<script src="JavaScript/authentication.js"></script>

<script src="https://apis.google.com/js/platform.js" async defer></script>

<meta name="google-signin-client\_id" content="69034754079-  
3cal6eu3d3f7g04lib2lbp004qvl798.apps.googleusercontent.com">

```
<?php include 'php/starterform.php'; ?>
```

```
<script
```

```
src="https://maps.googleapis.com/maps/api/js?key=AIzaSyCV1y5hUFRGd5kD_seneaxVw  
c33L-IGg9U&libraries=places"
```

```
async defer></script>
```

```
</head>
```

```
<body onload="showPosition(),gatherIntel()">
```

```
<div id="superdiv" >
```

```
<div data-role="header">
```

```
<div data-role="navbar">
```

```
<ul>
```

```
<li><div class="g-signin2" data-onsuccess="onSignIn"></li>
```

```
<li><a href="main.html">Home</a></li><!--logout-->
```

```
<li><a href="about.html">About</a></li> <!-- about me , the project  
using collapsable content formatting-->
```

```
<!--<li><a href="donate.html">Donate</a></li>--><!--link to paypal or  
something-->
```

```
<li><a href="main.html" onclick="signOut();">Sign out </a></div>
```

```
<script>
```

```
function signOut() {
```

```
var auth2 = gapi.auth2.getAuthInstance();

auth2.signOut().then(function () {

console.log('User signed out.');
```

});

}

</script></li>

</ul>

</div>

</div>

<div id="header" style="width: 480px; margin: 0 auto; text-align:center;" ><h1>You  
Look Hungry!</h1></div>

<div >

<fieldset data-role="controlgroup" data-type="horizontal" style="text-align:center;" >

<div id="restaurantselection" style="width: 480px; margin: 0 auto;">

<form id="myform">

<legend><h2>Choose a Restaurant:</h2></legend>

<input type="radio" name="restaurant" id="McdonaldsRadio"  
value="Mcdonalds" />



```

        <label for="McdonaldsRadio">Mcdonalds</label>

        <input type="radio" name="restaurant" id="SubwayRadio"
value="Subway" />

        <label for="SubwayRadio">Subway</label>

    </form>

</div>

</fieldset>

<div id="map_canvas" style="width:420px; height:350px; margin: 0
auto;"></div>

<div id="budgetinput" style="width: 480px; margin: 0 auto;">

    <label for="budget"><h2>Budget:</h2></label>

    <input type="number" name="budget" id="basic" value=""
step="0.01"style="text-align:center;" />

    <!--<h2>Budget :</h2> <input id="inputs" type="text"
name="budget" value="<?php echo $budget;?>"-->

</div>

<fieldset data-role="controlgroup" data-type="horizontal" style="text-align:
center" >

```

```

<div style="width: 480px; margin: 0 auto;">

    <legend><h2>Primary Nutrient:</h2></legend>

    <input type="radio" name="primarynutrient" id="Calories"
value="Calories" checked="checked" />

    <label for="Calories">Calories</label>

    <input type="radio" name="primarynutrient" id="Carbohydrates"
value="Carbohydrates" />

    <label for="Carbohydrates">Carbohydrates</label>

    <input type="radio" name="primarynutrient" id="Protien"
value="Protien" checked="checked" />

    <label for="Protien">Protien</label>

    <input type="radio" name="primarynutrient" id="Fat" value="Fat" />

    <label for="Fat">Fat</label>

</div>

</fieldset>

```

```

<fieldset data-role="controlgroup" data-type="horizontal" style="text-align:
center" >

```

```

<div style="width: 480px; margin: 0 auto;">

    <legend><h2>Secondary Nutrient:</h2></legend>

```

```
<input type="radio" name="secondarynutrient" id="radio-choice-1" value="choice-1" checked="checked" />
```

```
<label for="radio-choice-1">Calories</label>
```

```
<input type="radio" name="secondarynutrient" id="radio-choice-2" value="choice-2" />
```

```
<label for="radio-choice-2">Carbohydrates</label>
```

```
<input type="radio" name="secondarynutrient" id="radio-choice-5" value="choice-5" />
```

```
<label for="radio-choice-5">Protien</label>
```

```
<input type="radio" name="secondarynutrient" id="radio-choice-4" value="choice-4" />
```

```
<label for="radio-choice-4">Fat</label>
```

```
</div>
```

```
</fieldset>
```

```
<!--attempt at a page loading widget under pages and dialogs-->
```

```
<div >
```

```
<table data-role="table" id="menutable" data-mode="columnntoggle" class="ui-body-d ui-shadow table-stripe ui-responsive" data-column-btn-theme="b" data-column-btn-text="Columns to display..." data-column-popup-theme="a">
```

```
<thead>
```

```
<tr class="ui-bar-d">
```

```

        <th>Item</th>

        <th>Serving</th>

        <th>Protein</th>

        <th>Carbs</th>

        <th>Fat</th>

        <th>Calories</th>

    </tr>

</thead>

<tbody id="menuitembody">


</tbody>

</table>

<ul data-role="listview" id="menuitems" >

    <script>

        $( menuitems ).bind( 'mobileinit', function(){

            $.mobile.loader.prototype.options.text = "loading";

            $.mobile.loader.prototype.options.textVisible = false;

            $.mobile.loader.prototype.options.theme = "a";

            $.mobile.loader.prototype.options.html = "";

        });</script>

</ul>

</div>

</div>

```

```
</div>
```

```
</body>
```

```
</html>
```

### 10.1.3 javascript.js

```
/* Global Variables */
```

```
var globalmap;
```

```
var restselection;
```

```
var globalbudget;
```

```
/* ^ Global Variables ^ */
```

```
function gatherIntel(){
```

```
console.log("gather intel");
```

```
}
```

```
$(document).ready(function() {
```

```
    $('#McdonaldsRadio').click(function(){
```

```
restselection=document.getElementById('McdonaldsRadio').value;

showPosition();

});

$('#SubwayRadio').click(function(){

    restselection=document.getElementById('SubwayRadio').value;

    showPosition();

});

$('#basic').change(function(){

    globalbudget=document.getElementById('basic').value;

    console.log(globalbudget);

});

});
```

```
$(function(){

    var $menuitems=$('#menuitembody');

    $.ajax({

        type:"GET",

        dataType:'json',
```

```

url:"https://api.import.io/store/connector/f7b6d26c-3946-42c4-a8d5-
b3035c052e20/_query?input=webpage/url:http%3A%2F%2Fcalorielab.com%2Frestaurants
%2Fmcdonalds%2F1&&_apikey=d9cc296860fa459d83ee377c1f8fab91741f6294b17e4922
b2d0ca1120c389438a2fe57aabdf7acf7012ab8a903fb211dbe761f8e28fd997a7c935c7c6f621
83c08901b1a49c899933ca5fa6e8e6bdc4",

success:function(menuitems){

console.log("You have really done it this time batman");

$.each(menuitems.results, function(i,menuitem){

    $menuitems.append('<tr><th>'+ menuitem.menu_item + '</th><td>'+
menuitem.serving_size + '</td><td>'+ menuitem.protein + '</td><td>'+
menuitem.carbohydrates + '</td><td>'+ menuitem.fat + '</td><td>'+
menuitem.calories+'</td></tr>');

    });

    },

error: function(XMLHttpRequest, textStatus, errorThrown) {

console.log("Some Error:"+XMLHttpRequest+"Text Status:"+textStatus+"Error Thrown:"
+errorThrown);

console.log("I have failed my creator");

}

});

function showPosition() {

```

```

if(!navigator.geolocation) {

var map;

var mapOptions = {

    zoom: 12,

    mapTypeId: google.maps.MapTypeId.ROADMAP

};

map = new google.maps.Map(document.getElementById('map_canvas'), mapOptions);

globalmap=map;

if(restselection==null){

    console.log("User Has not selected a restaurant");

    //alert("Please select a restaurant");

}

navigator.geolocation.getCurrentPosition(function(position) {

    var geolocate = new
google.maps.LatLng(position.coords.latitude,position.coords.longitude);

    var userMarker = new google.maps.Marker({

        position: geolocate,

        map: map

    });

```



```

        infowindow = new google.maps.InfoWindow();

        var service = new google.maps.places.PlacesService(map);

        service.nearbySearch({

            location:geolocate ,

            radius: 5000,

            name:restselection

        },callback);

        map.setCenter(geolocate);

    });

} else {

    document.getElementById('map_canvas').innerHTML = 'No Geolocation Support.';

}

}

function callback(results, status) {

    if (status === google.maps.places.PlacesServiceStatus.OK) {

        for (var i = 0; i < results.length; i++) {

            createMarker(results[i]);

        }

    }

}

```

```
function createMarker(place) {  
    var placeLoc = place.geometry.location;  
    var marker = new google.maps.Marker({  
        map: globalmap,  
        position: place.geometry.location,  
        icon: "images/FoodHereMINI.png"  
    });  
  
    google.maps.event.addListener(marker, 'click', function() {  
        infowindow.setContent(place.name);  
        infowindow.open(globalmap, this);  
    });  
}
```