Fitness and Nutrition Buddy Project

Report

Prepared by

Andy Hansana, Kevin Elliott, Vincent Weaver, Ryan Crowley
***For use in* CS 440**
at the
University of Illinois Chicago

September 2021

Table of Contents

List of Figures

List of Tables

I Project Description

1 Project Overview

Our application, Fitness and Nutrition Buddy, will be a great application utilized by those interested in fitness, as well as keeping track of their daily nutrition. This application will use the users current location and will be able to locate all of the nearest restaurants around them as well as using a filter for certain restrictions/cravings that they desire (for example low carbs, low fat, etc). This will benefit the user by allowing them to intuitively live a healthy lifestyle, while not having the hassle of searching for nearby restaurants. They also will not have to search through all of the results/each individual menu. It will also be able to keep track of the user's steps, and calories burned (similar to the Fitness app on the Apple Watch).

2 The Purpose of the Project

This project is being done for convenience for those who aspire or currently have a healthy fit life. With this application being developed, it will be much easier for someone to figure out what they would want to eat on a diet, or to accommodate for any dietary restrictions. This would target an audience that would be interested in having a healthier lifestyle, or even if any users are trying to lose weight or have restrictions on their diet.

2a The User Business or Background of the Project Effort

The business that would benefit from this application would be certain fitness companies such as Gymshark or Alphalete, however it could branch out to many other companies to benefit them such as Nike or any specific company interested in branching out and working with a fitness application. The business being doing would be users needing an application for their needs and conveniences and then them applying it to their life.

2b Goals of the Project

We want to provide a convenient accessible application to customers so that they can access foods around them that accommodate their diet and tracking their fitness all in the palm of their hands.

2c Measurement

We would like to provide a premium subscription of our application if the user refers 5 people that download and use our application, as well as providing those with a free trial of how it is to upgrade to premium.

3 The Scope of the Work

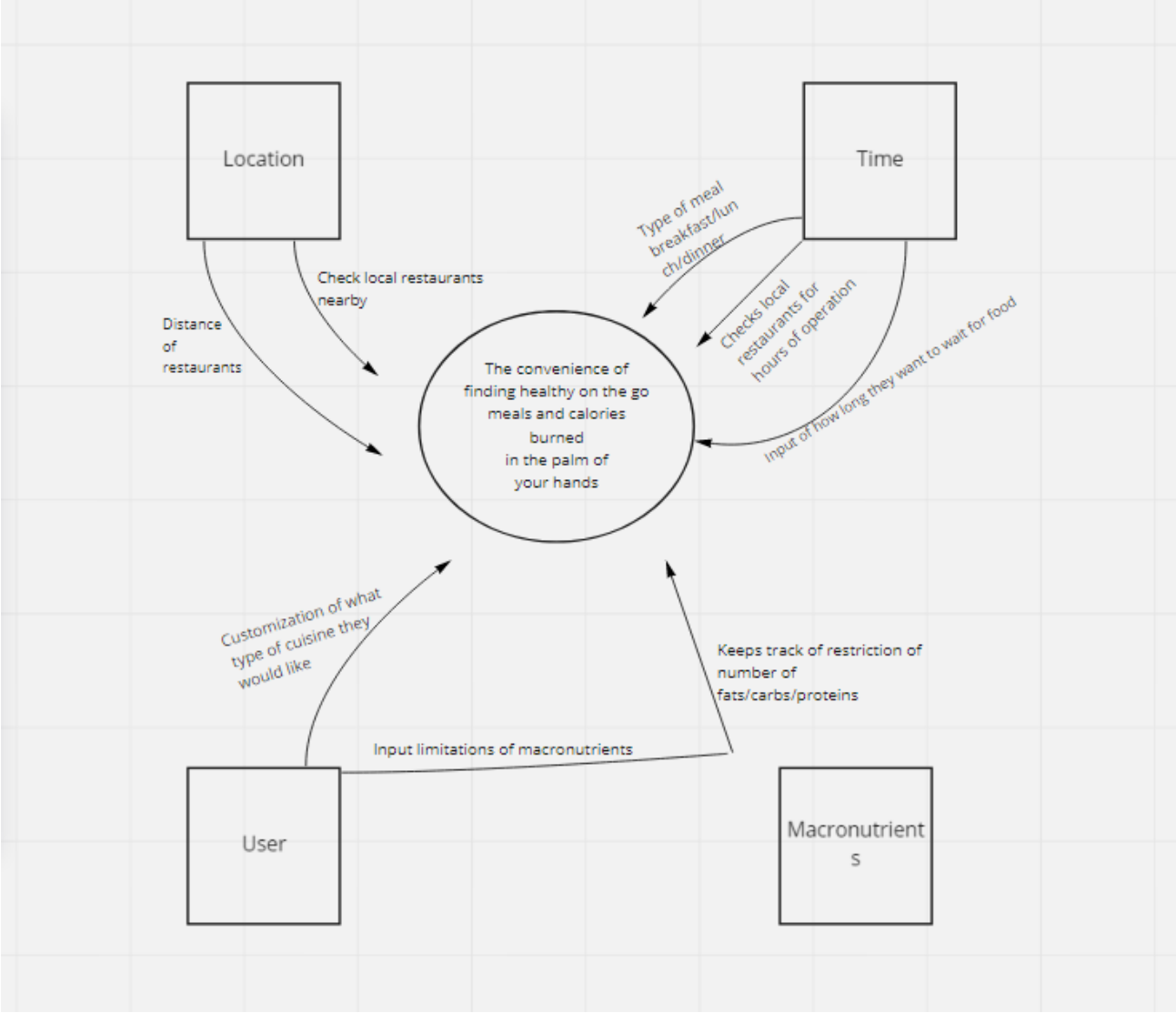
The work would be described as dietary nutrition needed for the users and the work would be providing the different dining options for the user available around them.

3a The Current Situation

The client conducting the work is the application being built in order for the users to be able to access it once it launches on the app store. Launching this business will incorporate updating the application daily and incorporating new restaurants in the area.

3b The Context of the Work

Page left blank



What is included within this application is the location, macronutrients, user inputs and location for the app to pull data and display it onto the interface which will consist of meals from different restaurants. The application has the capabilities of getting user customizations of what meals are liked and what aren't. What wouldn't be provided for this application would be certain restaurants that aren't specific on their macronutrient information for customers online.

3c Work Partitioning

Page left blank

Event Name	**Input Summary**	**Output Summary**
User burns calories	Application tracks number of calories burned from heartrate and movement	Calories burned output
User searches for food near them	Location and restaurants near them	Meals that they are able to buy on the go near them
Ketogenic foods near UIC	Location of UIC with	Ketogenic carbless foods

	restrictions of carbohydrates	such as (grilled chicken and vegetables) at Panda Express
Low Fat Snacks around the area	Takes the low fat restriction into consideration with x amount of fat and under	Snacks that are the same or less fat than requested fat such as greek yogurt
Input of a certain location where a restaurant is closed	Hours of operation for that certain restaurant	Gives message saying restaurant is closed and recommends somewhere else
User locates food near them at 5 AM with low calories	Location as well as late time	Only would output the lowest calorie meals at a late night restaurant/food area.
User updates their macros on their profile	User changes their macro goals or calorie goals for their daily counter	Notification is sent that the update was successful and the values are properly updated in the goals
User goes over calorie limit	User inputs a food that puts them over one of their goals for the day	Notification that gives tips on how to stay under and target the foods that contributed the most
User logs in for a streak	User logs in for the first time of the day	Every day the user logs in it is recorded on a calendar or counter
User breaks streak	User fails to login at any point in the day	Notification informing then on their streak and if they want to input what they ate yesterday

3d Competing Products

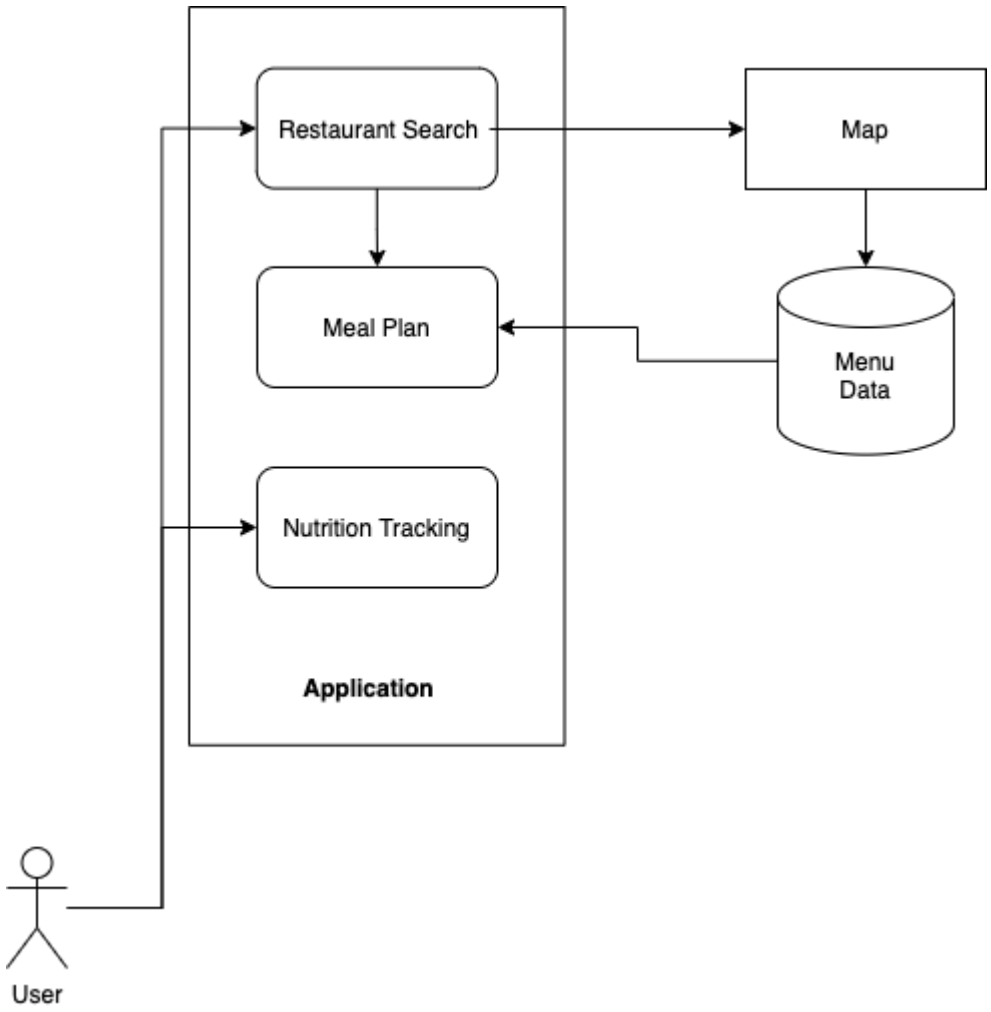
Other competitors that would compete in this market for this application would be MyFitnessPal, which is a fitness/nutrition mobile app that is capable of tracking your calories and fitness. One benefit that we would implement that our competitor doesn't provide is the ability to output what meals you would exactly be able to buy near you that fit your health goals. On top of that MyFitnessPal isn't always accurate as users are able to input calories of a certain food on the database that other users are able to access and read

which leads to data inaccuracy.

4 The Scope of the Product

The scope of our product would be the consistency and convenience of being able to provide healthy/convenient on the go meals that fit in our users dietary restrictions with them not having to stress over researching restaurants in their area and calculating all of the macronutrient information required for their diets. What is included in this application is tracking meals from restaurants around you that you're able to pick up and grab on the go and a tracker to see the amount of calories you burned. What isn't included on this application is tracking calories of meals that are from restaurants that do not provide nutritional information of their menus online.

4a Scenario Diagram(s)



4b Product Scenario List

Scenario Name	*Participating Actors*
1. *Nutrition Tracking*	*Fitness and Nutrition Buddy users*
2. *Meal Plan*	*Fitness and Nutrition Buddy users, Developers*
3. *Restaurant Search*	*Fitness and Nutrition Buddy users*
4. *Map*	*Fitness and Nutrition Buddy users, Google*
5. *Menu Data*	*Developers, Restaurants*

4c Individual Product Scenarios

- Nutrition Tracking: The user is able to track their calories and nutrition data on a day-to-day basis using the app's interface. This will be one of the main screens that the user interacts with when they load into the Fitness and Nutrition Buddy.
- Meal Plan: The user can create meal plans based on the restaurants they search for. This

will consist of pulling menu data from different restaurants and calculating calories, as well as sorting foods by their nutrition groups. This portion of the app will also filter any specifications the user may input based off of their current diet.

- Restaurant Search: A search feature is available for the user to find specific restaurants in their general vicinity, which will be connected to Google Maps. The user is able to input preferences like the type of food or the general location.
- Map: The app utilizes Google Maps to display a map to the user, which will show various restaurants around their location. The app will display the user's search results with this map by indicating the restaurants' locations, as well as a summary of some relevant data based on their search specifications.
- Menu Data: Data is pulled from Google pertaining to menu information, which is sent to the meal plan portion of the app in order to do the calculations for optimizing a meal for the user.

5 Stakeholders

5a The Client

The development organization would be the initial clients, working towards the goal of eventually selling the product to a large company in the fitness industry, similar to under armor owning myfitnesspal.

5b The Customer

The customer would be a big fitness brand that is looking to expand into a mobile market, some brands that could possibly be interested would be gym shark, nike, weight watchers, gnc. A brand that has an established presence in the fitness industry with or without a mobile app presence.

5c Hands-On Users of the Product

The user could be anyone on the app stores of an iphone or android, with the widespread availability of the apps in the app store the potential customers are very large. It would be up to the client if they want to charge the customers up front or via premium services for a subscription. The target demographic of the customers is those who workout and incorporate meal plans, people who want to lose weight, or even anyone that just wants to find new meals from restaurants in the area. The users are responsible for inputting their goals and calories eaten, and requesting meals from the app. The user's knowledge can be novice since the app is designed to do the harder parts of macro management for them, but being an expert does not harm the user's experience. The app will not have any particularly hard to navigate portions so anyone who can competently use their smartphone will be able to use

the app. The users can be fit or unfit, and anywhere in between. Disabilities are not relevant as long as the user is able to go out to eat. Users would typically be anyone that works out or cuts calories so teenagers and up. Education and linguistic skills are not needed besides basic english. Users' attitude towards technology helps with the app as being more connected to your phone means more opportunities to interact with the app based on the food consumed.

5d Maintenance Users and Service Technicians

No physical installation or maintenance is required besides server upkeep which would be handled by the updates team. The team that maintains the product would handle any of the servers needs along with and addition features the client wants.

5e Other Stakeholders

Other stakeholders that this application would need to include would be:

- Testers (for debugging purposes)
    - Knowledge needed for test cases and debugging code
    - Testers would be semi-involved as a bug could potentially prohibit users from using the application until fixed
    - Working together with Original developer team for combining knowledge for promotion of growth for the app.
- Marketing
    - Sponsorships towards influencers (better advertising for the product)
    - Important involvement as marketing is a huge factor for sales and promotion
    - Huge influence on success rates if advertised well
- Legal Team
    - Knowledgeable of lawsuits as well as dealing with any terms of the law if needed
    - Will be important as if any other companies try to copy/rebrand our own unique product
    - Significant impact and reliability on the company as if there is no legal team, company can get sued and shut down.

5f User Participation

Users can expect to participate in the development of the app by using a beta version of it enough features have been added. Whether the beta will be free or paid for is up to the client but there would be temporary features involved to poll users experience in the app so far to provide feedback.

5g Priorities Assigned to Users

The key users are those that are fit and looking to maintain a diet with exact macros and also those who are trying to lose weight through a diet. These are the two groups that have the most to gain by consistent use and the features should be prioritizing them. Secondary users would be those who use the app on a semi consistent basis and are generally into fitness but not fully committed. These users are the most likely to grow into the key user group as time goes on so their needs should also be considered so the biggest part of the userbase is cared for. The unimportant users are the ones that use the app for new food recommendations, the app could certainly be used in this way but it is not the intended use, therefore if users are unhappy with the focus on fitness and not just focusing on new meals from restaurants their concerns don't have much weight to the team.

6 Mandated Constraints

6a Solution Constraints

Description: The final product should be a mobile application, or at least have its main platform be a mobile application.

Rationale: This must be done to provide the user with on-the-go capability. The user should be able to obtain healthy meal options wherever they are.

Fit criterion: The app must be available on the Apple App Store and Google Play for users to download, and should provide the user with a view of their current location on startup.

Description: The final product should provide accurate nutritional information and calculations for the user's personal dietary specifications.

Rationale: The app needs to provide an effective means of improving the fitness and nutrition of the user.

Fit criterion: The app must utilize Google Maps data, or menu data directly from the restaurant only in order to form accurate nutritional calculations.

Description: The app must provide a quick way to find meals for the user.

Rationale: The user may be in certain social situations where decisions need to be made quickly (i.e. at a sit-down restaurant). Also, this is for the user's convenience when trying to find healthy options.

Fit criterion: The time frame for searching for restaurants and meals must be under 30 seconds.

6b Implementation Environment of the Current System

The product will be designed as a mobile application that will be catered towards the user installing on their personal device. The application will run on all major mobile platforms (i.e. IOS, Android, etc.).

6c Partner or Collaborative Applications

There are an extreme number of restaurants around the world with different data and information regarding their menu items. Collaborating with applications such as Google Maps, Apple Maps, and Waze, as well as restaurant databases and APIs such as Nutritionix and Chomp would be useful. Their location services would increase the speed and efficiency of the app, and the API data would allow for accurate and informative data to provide to the user.

6d Off-the-Shelf Software

Content

Some software that must be included in the final product includes a 24 hour server that updates real time restaurant data regularly. The server software must also store data from the users and build specific diets based on prior food results. Having this 24 hour server would also include physical hardware that continuously runs the server.

Motivation

Ideally, other products that are being looked at to be incorporated into the final product include some potential calorie tracking software in other devices.

Examples

Utilizing a smart watch from companies like Fitbit or Apple's watch product is in consideration when building this product. The application would really benefit from adapting to a separate mobile device such as smart watches since it can be more accessible and convenient for users than a phone.

Considerations

The immediate conflict with attempting to get the OTS software working with the Fitness and Nutrition Buddy is compatibility with the smart watch device and communicating with the companies who own these devices.

6e Anticipated Workplace Environment

Content

The product is designed to be used while the user is both stationary or on the move. Ideally, users have the flexibility to be either at home or away while using the application. Especially when users are out shopping or getting out of the gym, being able to use the app anywhere is core to its functionality.

Motivation

The application uses data from satellite maps and restaurants and combines both data sets to feed the user relevant information about where and what to eat based on the users choice of food. Allowing the application to be compatible with smart watches as well as mobile devices allows for flexibility in user workspace.

Examples

Some examples of applications for satellite map data/APIs are from Google Maps, Waze, or

Apple's map data. Some examples of programs that utilize food chains are Uber Eats, GrubHub, etc. These applications are examples that allow for both mobile use or stationary use.

Considerations

The developers would be expected to work in person for the majority of the week. Whether or not a hybrid system like many other companies would be up to the team based on how they have been working post start of development. If they choose to go hybrid, employees are expected to supply their own devices.

6f Schedule Constraints

There are not any hard deadlines for the app in the development phase, and the best timing for it is a mix of waiting to release as the budget allows it. Given the pandemic is not gone away yet some fast food locations are operating differently and the longer the app takes to release the more normal things will become. Obviously the team doesn't have the budget to wait forever but if there were to be a large set back the app does not fail. Having to miss the release for this calendar year and take Q1 of the next is not ideal but still doable.

6g Budget Constraints

Since the project aims to get bought out by a large company eventually, it is in the team's best interest to create a quality product as fast as possible to maximize the use of the budget. There wouldn't be any budget cuts exactly since no company will buy a cheaply made app, but the emphasis for the budget would instead be on avoiding setbacks if possible to make the time of development smaller. The resources for the product would be the developers and the team.

7 Naming Conventions and Definitions

7a Definitions of Key Terms

Meal: a combination of various menu items provided to the user after requesting a food recommendation from the app.

Calories: how many calories a food item has

MealCalories: how many calories an entire meal has

Carbs: one of the 3 macros the app will track, carbohydrates, refers to anything that falls

under the carbohydrates section of a nutrition label

Fat: one of the 3 macros the app will track, fat, refers to anything that falls under the fat section of a nutrition label

Protein: one of the 3 macros the app will track, protein, refers to anything that falls under the protein section of a nutrition label

Macros: all three of carbs fat and protein together make up a users macros that they set

Restaurant: a place that sells fast food, dine in, take out etc. does not refer to grocery stores

Streak: user logging in for a x amount of days

Location: Where the user currently is, particularly when they request for a meal

Profile: where the users attributes are stored such as macros, streak, etc

User: current instance of the app

7b UML and Other Notation Used in This Document

This document generally follows the Version 2.0 OMG UML standard, as described in, M. Fowler, UML Distilled, Third Edition, Boston: Pearson Education, 2004. Exceptions are noted in their specific cases.

7c Data Dictionary for Any Included Models

DailyTotal = totalCalories and totalCarbs and totalFat and totalProtein

Meal = 1 or more menu items

MealCalories = sum of calories from every item in a meal

8 Relevant Facts and Assumptions

8a Facts

67% of americans are overweight or obese
84.8 million people eat fast food a day in the US
Application will be made in the appropriate language for the platform(swift for iOS)
19.3% of americans are participating in some form of exercise each day
The average american consumes 3600 calories a day
The recommended daily caloric intake is 2000 for women and 2500 for men

8b Assumptions

Page left blank

Assuming fast food chains will either continue to operate at current capacity or improve

Assuming all developers will be able to code in all mobile environments

Assuming servers will either be hosted via a reliable database or in house

Assuming google maps api access is maintained

Assuming once a client purchases the product the team is staying with the product

Assuming covid does not worsen and create another lockdown

II Requirements

9 Product Use Cases

9a Use Case Diagrams

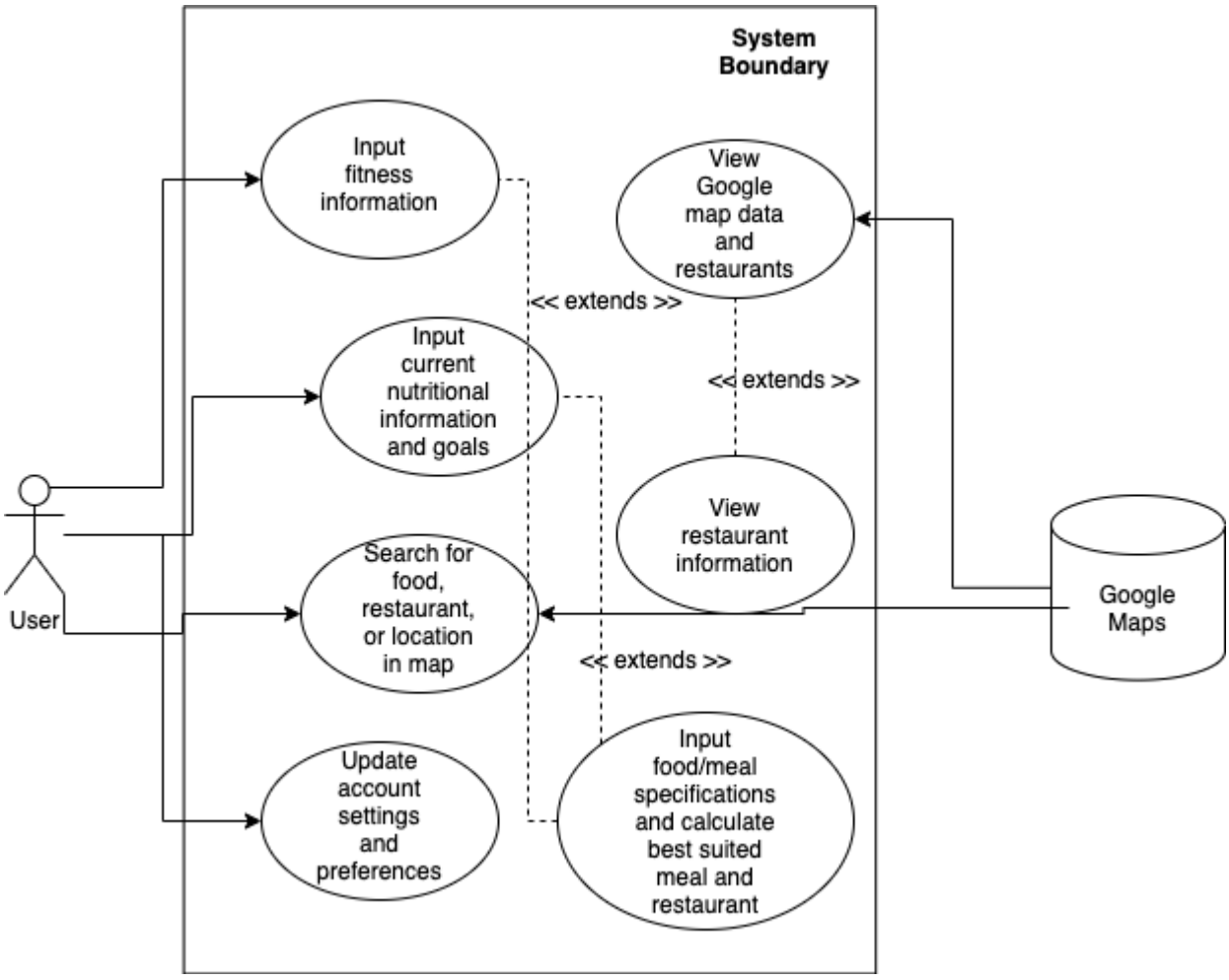


Figure 1 - Sample Use Case Diagram for Fitness and Nutrition Buddy

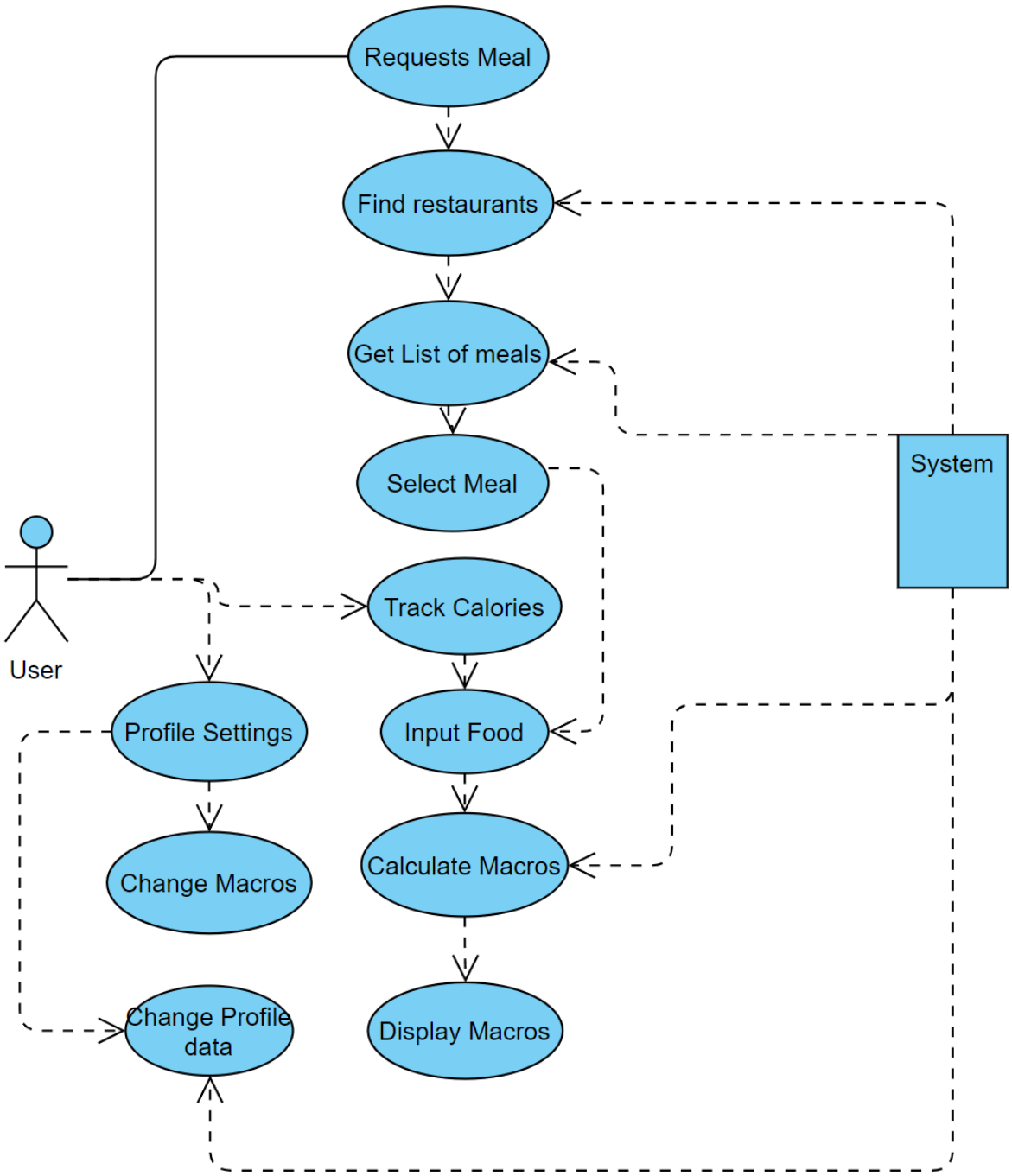

Requests Meal
Find restaurants
Get List of meals
Select Meal
System
User
Track Calories
Profile Settings
Input Food
Change Macros
Calculate Macros
Change Profile data
Display Macros

Figure 2 - Sample Use Case Diagram for Fitness and Nutrition Buddy

9b Product Use Case List

9c Individual Product Use Cases

- Input fitness information
- Input current nutritional information and goals
- View Google map data and restaurants
- View restaurant information
- Search for food, restaurant, or location in map
- Input food/meal specifications and calculate best suited meal and restaurant
- Update account settings and preferences

Use case ID: 1	Name: Input fitness information
pre-conditions:	
post-conditions:	
Initiated by:	
Triggering Event:	
Additional Actors:	

Sequence of Events:

1. Initiating event or action should be step 1, taken by initiating

actor. 2. System response follows, indented right.

3. All external action steps are aligned with step 1. ("stimulus" style)

4. All system responses are indented right, aligned with step 2. ("response" style)

5. All steps should be expressed in the active voice, clearly indicating **who** performs each action

6. The sequence of events should show a back-and-forth stimulus response relationship.

Alternatives: These would be normal and expected variations from the base case.

Exceptions: These would be unusual variations from the base case, often caused by problems.

10 Functional Requirements

SV: The user must be able to have a functional application to their needs. All functional requirements must be working for the user to be satisfied with the application.

001 - Diet Style

Description: This is where the user will input their style of calorie and macro restrictions so that the app can proceed to make their customized menus
Rationale: The app cannot make any custom meals without the restraints on it

Fit Criterion: Does the user's profile have its custom settings prompted on start up for the first time and is it saved upon relaunch, along with being able to change at any time.

Acceptance Tests: Save user data, input diet data

002 - Location Services

Description: When the user requests a meal from the app, it needs to know where it is and what's around, pulling in data from google maps will solve both
Rationale: The apps primary function is based on location

Fit Criterion: When the user requests food does location services draw where they are and also the restaurants around them

Acceptance Tests: Location Loads, restaurants load

003 - Meal Creation

Description: Drawing from the menus of restaurants around the user, the app will create a meal for them that is restrained by the users calorie limit along with macro limits. After the meals are created they are added to a table along with every other meal combo from the other restaurants
Rationale: The app does not need any advanced filtering for these meals since its purpose is to show the users all of their options even if one is not nearly as satisfying as the others

Fit Criterion: When the user requests meals, a table populates with meal combos based of user limits from the restaurants around them

Acceptance Tests: Meal Creation, Meals under set limits, Meals from all restaurants

ID# - Meal Creation

Description: The system must have a requirement towards dietary restrictions and inputs towards the users macronutrient intakes.

Rationale: This requirement is necessary in order to provide towards the users inputs in order to provide towards their fits in healthy lifestyle.

Fit Criterion: This will be met if the user is able to keep track of their macronutrient intake and have it be accurate towards their lifestyle .

Acceptance Tests: Macronutrient tests, nutritional information unit test.

11 Data Requirements

SV: Data requirements: User profile data containing nutritional information, location services, restaurant menu information

Content: User class, Restaurant class, Map object, Meal object

Motivation: User should be able to login and based on their macros get a meal from surrounding restaurants

Considerations: Google maps is the most likely way to get location services, and many times they use Yelp for menus since restaurant owners put their own on there very commonly.

ID# - Data requirements

Description: The system must have data input towards showing the user the restaurants around their area. -

Rationale: This requirement is required towards the user inputs in order to provide the custom meals towards the users macronutrient requirements.

Fit Criterion: This will be met if the user is able to track their macronutrient

requirements in the application as well as the application inputting accurate data of custom meals of restaurants around their area for their needs.

Acceptance Tests: Accurate data unit tests, macronutrient tests.

12 Performance Requirements

12a Speed and Latency Requirements

SV: Any time a user presses a button there needs to be some kind of feedback that input was received. If it is a process that takes longer than a couple seconds then a loading indicator of some kind should be in place. On long tasks that take more than 10 seconds, a loading bar with updates should be in place.

SV: accepting user input should be instant. Getting users location should be under 5 seconds from when they allow it. Loading restaurants menus should be no longer than 1 minute and creation and loading of meals should be no longer than 2 minutes

Motivation: The app has an emphasis on speed but ultimately is not the highest priority, when someone is out and looking for something to eat they do not want to wait more than a few minutes to figure out what they want. Therefore the app should be able to complete its function from requesting a meal to displaying it in around 2-3 minutes at most, the faster the better. Users taking time to decide what they want is not factored into the time.

Examples: User pressing a button means the button moves or highlights on press and the function is carried out within 2 seconds. Response needs to be fast enough to not be distraction or interrupting. The product should update location services every 10 seconds so that if they user in a vehicle it stays accurate.

Fit criterion: the creation of meals should take no longer than 3 minutes

Considerations: The users will be hungry so making them wait is not ideal, but not the end of the world, the process shouldnt make users feel like they are sitting there waiting around when they could just go get a meal before the app even shows anything.

ID# - Performance

Description: The system must have feedback required when pressing a button.

Rationale: Necessary for feedback given back.

Fit Criterion: Needed for putback when user presses a button.

Acceptance Tests: Performance unit test.

12b Precision or Accuracy Requirements

SV: The location services do not have to be very accurate, so anywhere within 100 feet of the user is fine as an anchor point. The meal creation needs to be hard limited to the users set restrictions.

Content: People are counting ever calorie and would not have any use for a meal thats “only” 50 cals over their limit.

Motivation: People are using the app to lose weight or gain muscle and both need strict rules and attention to detail to get right

. **ID# - Precision**

Description: The system must have the user input to be instant and output the custom meals within the next 3 minutes .

Rationale: This requirement is necessary in order to have the user to be able to have custom meals towards their needs in a timely manner.

Fit Criterion: This requirement is required in order to have the users input their meals in a timely manner in order to have meals outputted in their application for their needs. Example: A user is on lunch break and uses the application to have a meal created for their macronutrient needs and needs to be created on a timely manner. Since this user is on lunch break, they are limited on time and need to get their meal created ASAP in order to enjoy their lunch break. If the application is slow and isn’t able to create their meal on a timely manner, their lunch break time will be winding down, and they wont be able to enjoy the time on their break.

Acceptance Tests: Speed test towards accuracy.

12c Capacity Requirements

SV: Largest thing the app will handle is location and restaurant menus, with the smaller things being user profile and meal table

Content: App will have to download map data and restaurants around it which is an estimated 50MB every other part of the app is estimated to be 100MB

Motivation: This is a mobile app so every MB counts towards more users being able to download the app

ID# - Capacity

Description: The system must have the application be at least 50 MB in order for the user to successfully work with the application and the features efficiently.

Rationale: Requirement is necessary in order for the user to download the application.

Fit Criterion: This is needed for the user to operate the application as well as download it for the user to successfully use it.

Acceptance Tests: Memory space test.

13 Dependability Requirements

13a Reliability Requirements

SV: Under no circumstances should the app fail to load user data, location services can fail if the users connection isn't strong enough or disconnected, same with restaurants. Given the past two load, the creation and delivering of meals should not ever fail to get to the user

Content: if the users connection is fine then there should never be a failure to laid, the only reason it should happen is if the connection is poor or offline

Motivation: If the app fails too often then users will leave, but its not a major emergency if it does fail so occasionally if it does it is alright if the users connection is not consistent. The product should not fail more than once a day, in the event of a failure the users data needs to be preserved above all else.

Considerations: The product needs to be reliable so that at any given time if someone is hungry they can use it, but does not have many critical functions that cant fail no matter what, therefore the functionality should be what a user expects for any app.

ID# - Data Dependability

Description: The system must not fail to load user data and location services.

Rationale: Main reasoning of function and purpose of this application is for location service tracking and data loading within food menus of nearby restaurants.

Fit Criterion: Determination of whether failing of loading user data and location

services may be acquired by constant updates/checking on application functionality as well as having specific bug tests to make sure users are constantly able to use this feature and this application is able to manage to have this working at all times.

Acceptance Tests: Location services unit tests, pulling data and loading it from specific location test, menu display test.

13b Availability Requirements

The product should be available for the vast majority of the day, with any downtime for maintenance planned for when there are the least users active. Since most people will be eating from anywhere between 6am to 11pm, the app must not have any maintenance during these hours, since maintenance can take place from 12am to 6am it should never take longer than 6 hours. Outside of maintenance downtime, the apps services should not be down for more than an hour a day, 23/24 hours should have the system running and when it goes down it should not take longer than 30 minutes to fix. Even when the system is down users should still be able to track their calories since that is locally stored and can be synced with the server later.

ID# - Application Runtime

Description: The system must be able to run for the vast majority of the day.

Rationale: Necessary operation time is required in order for users to be able to access this application throughout the day for their uses.

Fit Criterion: The uptime of this application is mandatory from the times to at least 6AM-11PM minimum. Uptime is needed in order for users to check around them of places to eat, and track their nutrition information. This will be acquired through checking the uptime periodically and having it maintain a steady rate of availability.

Acceptance Tests: Uptime Unit tests, Availability to users and functionality working correctly.

13c Robustness or Fault-Tolerance Requirements

The app will retain some functionality in offline mode, the majority of which will be related to the users profile and features surrounding it. Users will be able to track calories they have consumed in their daily tracker, make edits to their macro goals, and make changes to some of their profile data. Obviously users can't change their password if the system is down but they could change their text based profile customization. Any changes made to the users profile and macros will be saved in a temporary file until connection to the server is regained, then it can sync with the server and save changes.

ID# - Offline Mode

Description: The system will have an offline mode similar to airplane mode where user could still access the app but be limited with resources and functionality.

Rationale: An offline mode is necessary due to the fact of in the case of an application miswork or failure. Applications must always have a back up in the case of failure for reliable reasons as well as being sufficient to work through any mishappenings.

Fit Criterion: In the modern day era, all software applications and programs are not always reliable and there is always room for improvement as well as it being prone to errors. Software always will have bugs and times of failure and issues, however with offline mode there will be at least some leisure in the case of happenings such as that while errors are being resolved at that time.

Acceptance Tests: Unit tests include functionality of offline mode while app being down, and offline mode application tests on different devices such as android/apple/etc.

13d Safety-Critical Requirements

There is not any property or OSHA compliance for the app, but there is health and safety. There needs to be a notification system that notifies users of new health information when relevant. Scenarios such as a mask mandate for restaurants or a recent salmonella outbreak found in lettuce are things that users need to be aware of as soon as possible and since the app is guiding them to take out locations, it is its responsibility to provide them with the information relevant to their selections when applicable.

ID# - Safety Conditions

Description: The system must update users of safety conditions around the local area in the case of any safety/health issues.

Rationale: Updates for users are mandatory in the case of any issues going on with particular health issues in restaurants around that they would be traveling to.

Fit Criterion: Safety is a requirement needed to be meant to suffice the well being of all users of this application. Conditions that lead to this can be met when application notifies the user of any mishappenings around their local area.

Acceptance Tests: Pandemic test, mask mandate test, salmonella outbreaks in any restaurants notification.

14 Maintainability and Supportability Requirements

14a Maintenance Requirements

System maintenance can be divided into two sections, server and application. Application maintenance such as bug fixes and product releases can only be performed by the apps current dev team, whether that is the original team or a new team after

purchase does not matter. Server maintenance can only be done by someone that is either on the team and qualified to conduct maintenance or someone who is solely responsible for network maintenance. In the case of servers being hosted in house then it would be the responsibility of the dev team's/owners appointed maintenance person, and in an outsourced server company it would have to be performed by whoever the outsource company elects.

ID# -Maintenance

Description: The system must have regular maintenance in order for application to run smoothly at all times and minimize bug errors.

Rationale: Maintenance must be made during outside busy hours of the application for stable application purposes.

Fit Criterion: In the case of no maintenance being performed, would lead to application issues that can generate from bugs, and also failures and crashes of this application.

Acceptance Tests: Bug unit tests, system maintenance tests, run time tests.

14b Supportability Requirements

Ongoing support can be in the form of updates to the food information data along with news diet plans being implemented to users profiles. When new products get released there will need to be new data on their macros, and whenever a new diet plans becomes popular people will certainly want to try it out in the app. Updates to the app can come in the form of monthly updates featuring bug fixes and new food data. Besides monthly updates, there will be big content updates that are not on a set schedule and instead release based on when the dev team can complete it. For example a monthly update for the month of june could include nutrition data on some new dairy free ice cream that exploded in popularity and also fixes a few bugs. A content update would look like in january the dev team announcing a new feature such as a friends list system where you can see each others achievements and that its expected release is Q3 of that year. Then by the end of Q2 when the team has a clearer picture on how much progress is left they can give an accurate release date of something like late August.

ID# - Support

Description: The system must be able to provide support towards users within the application.

Rationale: Support means are needed in the case of users needing any type of assistance towards this application, whether it could be a manual to read off of or a FAQ page.

Fit Criterion: Being able to provide support to users is a great deal in the case of customer service so that the audience will be satisfied with how the business and name of this application are.

Acceptance Tests: Support Satisfaction from users.

14c Adaptability Requirements

The platforms that the application will run on are primarily iOS and Android. These two platforms are where the vast majority of mobile users are and thus they will be the focus of development. Both platforms will receive updates to keep them up to date with the latest OS versions. There are currently no plans to port the app to windows or mac since the app focuses on being out and about while using it. The product is designed to work on mobile devices. After a new major update is released for a users phones operating system such as iOS 15 the app should be compatible with it withing a week, and should have an optimized version within the month.

ID# - Adaptability

Description: The system must have adaptability requirements towards all systems such as IOS, and Android.

Rationale: This is needed to maximize all users on the platform.

Fit Criterion: This is able to satisfy the requirement in the case of all users to be satisfied with the application on their app stores on the platform their using such as apple/android.

Acceptance Tests: Platform acceptance test.

14d Scalability or Extensibility Requirements

On launch, the system should be able to provide users with 10,000 users creating custom meals per hour. This is expected to grow to at least 100,000 within 2 years. Assuming each user will need at least 5 meals to populate their choice table, the system should be able to provide 50,000 meals per hour on launch and 500,000 within 2 years..

ID# - Name

Description: The system must provide towards scalability towards users.

Rationale: In the case that the application blows up/

Fit Criterion: Application must be able to provide towards all users.

Acceptance Tests: Scalability test

14e Longevity Requirements

The app will be fully functional and operate at budget for a minimum of 3 years, with evaluations at the end of each year to adjust the path given performance metrics. 3 years is given so that there is enough time for a dedicated user base to take form as well as time for the restaurant industry to recover given the past year.

ID# - Longevity

Description: The system must be able to operate for at least 3 years.

Rationale: Requirement is necessary for the users to use the application throughout time.

Fit Criterion: Necessary to provide for all users.

Acceptance Tests: Scalability test.

15 Security Requirements

`

15a Access Requirements

The dev team as a whole will have access to anonymous user data but only managers/senior devs will have access to the encrypted database where sensitive user data is stored. This is because the encrypted database will not be needed for day to day programming and more for quarterly/yearly reports.

ID# - Security

Description: The system must be able to have the database of the user inputs stored.

Rationale: Necessary for users to have security purposes in the application.

Fit Criterion: If the application were to have a security breach, the data would at least

be stored in a secure spot.

Acceptance Tests: Security breach unit test.

15b Integrity Requirements

There are mandatory backups of the data related to the users on the app, occurring once a week. All user profile data, health data, and food data is included in this back up. Besides user data any data that the dev team collects must be backed up as well. The backup is to be stored in a different location than the main servers, whether that is through an external file hosting website or in another office site for the company does not matter. The aim of these backups is to minimize user progress loss in the case of a drive failure or loss of servers. Creating backups any later than a week could make users completely stop using the app in the case of data loss, people can make a lot of fitness progress in a month and 'losing' that could be very demoralizing. A week is a much more recoverable loss where many users would be inconvenienced but not angry about.

#14 - Back Ups

Description: The apps user data will be backed up once a week.

Rationale: This minimizes loss of user data in the case of data loss.

Fit Criterion: This ensures that the requirement of user data storage on servers is met and that users can trust that the data is saved correctly.

Acceptance Tests: Data Backup Test

15c Privacy Requirements

Users will not be able to access the data on any other user besides themselves, and the dev team can only access user data if they have agreed to anonymously share their data. Since the data revolving around someone's diet is considered health data it should only be able to be seen if the person agrees to it. These restrictions will be clear to the user as on account creation they will be asked if they would like to anonymously share their information to the devs for accurate data collection, users may also opt out or in at any time via their profile options. The app will also notify users of changes to the information and privacy policies. The app will also ask users for permission to share data with the 'health' app found on iOS and Android, along with users' options on how much they would like to share. All of this data will be stored in an encrypted database to prevent leaks of data.

55# - Data Clearance

Description: Users cannot access any other users data besides themselves, and the dev team can only access user data if they have allowed for anonymous sharing.

Rationale: Users have no need to see others data, and the dev team needs to have user

approval before seeing and using their data.

Fit Criterion: This meets the standard for security among user data and how it can be used by themselves and the developers of the app.

Acceptance Tests: Data Clearance Test

15d Audit Requirements

Support must be provided for monetary transactions related to the premium subscription program. There are no medical records stored in the app so the team will not have to worry about that. A system must be in place to record how many transactions are processed along with how many users are subscribed and for how long each user has been subscribed. The system must be compatible with the appropriate audit policies.

77# - Audit Test

Description: Records of all subscriptions and payments must be provided upon request.

Rationale: Since the premium payments are an option the app needs to be able to respond to an audit request.

Fit Criterion: This meets the requirements for a business that handles with transactions over in app purchases.

Acceptance Tests: Audit Request Test

15e Immunity Requirements .

The application should only use websites that are absolutely necessary for retrieving information to reduce the odds of a virus occurring. Google and Yelp are the main websites used for the apps data which are both safe websites, but occasionally the app may need to look for a restaurant's website by itself. This is where the greatest risk of virus happens, to avoid this, there must be a proxy system in place when going outside a whitelisted list of sites. In other terms there must be a system in place to protect from malicious sites and software. Since phone viruses are much more rare than computer/website viruses, the focus should be more on the latter. Mobile apps themselves are much harder to corrupt as well so standard security to prevent app tampering is needed.

98# - Firewall

Description: The app needs a firewall to stop any malicious code from websites infecting the app.

Rationale: Websites are a weak point in the security of the app since they cannot be

controlled by the app. Some websites could infect the app with a virus if they are not scanned beforehand so it is essential the app has a firewall.

Fit Criterion: This meets the requirement of the app to be secure for users

Acceptance Tests: Firewall Test

16 Usability and Humanity Requirements

16a Ease of Use Requirements

The software should be organized and intuitive enough so that a new user feels fairly comfortable with navigating the interface. Anyone with a smartphone or tablet should be able to use this product. It should limit the amount of manual input.

ID# - Usability

Description: The software should require the least amount of training possible for the user to undergo. Anyone with a smartphone or tablet should be able to comprehend how the system functions.

Rationale: Because of the amount of scientific information the app will store, providing an intuitive interface will ensure that the user does not get confused or discouraged while navigating the application.

Fit Criterion: No user should require more than 10 minutes to understand how to effectively navigate the system.

Acceptance Tests: Usability Time Test **REFER TO SECTION 21**

16b Personalization and Internationalization Requirements

ID# - Personalization

Description: The software is accessible to users around the world who have access to Apple's App Store and the Google Play Store. Users shall be able to customize their language settings within the profile settings menu.

Rationale: Implementing multiple languages within the app ensures that users from all over the world from different countries and cultures feel comfortable and in control while navigating the software.

Fit Criterion: No user should be forced into navigating the app in any language other than their first.

Acceptance Tests: Language Test **SEE SECTION 21**

16c Learning Requirements

The software should seamlessly train its users while they interact with the system.

ID# - Learnability

Description: The software should display helpful tips and guidelines for the user upon loading the app for the first time. The user should have the option to turn these guidelines back on at any time.

Rationale: Having a built-in tutorial mode allows the user to learn "on-the-go" while interacting with the actual application, rather than having a separate tutorial module.

Fit Criterion: Tips and guidelines should only appear once when the user first opens the application, and only display again when prompted by the user.

Acceptance Tests: Tips and Help Test **SEE SECTION 21**

16d Understandability and Politeness Requirements

Since many users all over the world have access to the Apple App Store and the Google Play Store, it is important that the application be accessible to all peoples who download it.

ID# - Understandability

Description: The product should refrain from displaying irrelevant scientific and calculation information.

Rationale: Since this app will be marketed for users of any fitness experience level, it is important for users to not feel overwhelmed by terms and scientific data that they do not understand.

Fit Criterion: The software should provide documentation as an optional feature for the user to select that explains the nutritional calculations and scientific reasoning behind these calculations and decisions made within the app regarding the user's fitness and nutrition.

Acceptance Tests: Understandability Test **SEE SECTION 21**

16e Accessibility Requirements

The software is equipped with features and settings like Text To Speech and visual preferences, making it suitable for users with hearing and visual disabilities.

ID# - Accessibility

Description: This software is accessible by users with disabilities, as it is equipped with

settings and preferences that accommodate both visual and hearing disabilities.

Rationale: Anyone should be able to use this product regardless of their disability.

Fit Criterion: Those with hearing and visual disabilities should be able to use the product just as fluidly as those without them.

Acceptance Tests: Accessibility Test **SEE SECTION 21**

16f User Documentation Requirements

The software should provide proper documentation of its purpose and license to the user to ensure their understanding of the product's functionality.

ID# - Documentation

Description: The product documentation should disclose its license agreement to the users, as well as describe its intent.

Rationale: The user should have clarity about the functionality of the product they are using.

Fit Criterion: Every user should be aware of the product's purpose.

Acceptance Tests: License Agreement Test **SEE SECTION 21**

16g Training Requirements

The product is designed so that tips and guidelines for how to use the app are built into the user's first time experience, so no formal training should be required.

ID# - Training

Description: No formal training is necessary.

Rationale: The user is already provided with tips and guidelines upon loading the app for the first time, and can toggle these in settings whenever they please.

Fit Criterion: The user should be sufficiently trained through the tips and guidelines they are presented with upon first loading the app.

Acceptance Tests: Training Test **SEE SECTION 21**

17 Look and Feel Requirements

17a Appearance Requirements

ID# - Color

Description: The appearance of the application would require certain colors to play a role in the application such as greys and blacks to portray the color of weights, as well as vibrant colors that display colorful nutritious foods such as salads and fruits. It shall be attractive to young adults that are interested in the fitness lifestyle.

Rationale: Users would need to be drawn in from the modernized GUI and special features that appeal to the eye.

Fit Criterion: Within glancing through the application, the application should keep the user drawn into the different features implemented such as having the restaurant meal maker around your area.

Acceptance Tests: Color Test

17b Style Requirements

Description: The style of this product would take a modern approach to applications to such as having matte black colors, for young adults to get the "cool" look towards the application as well as certain fonts standing out to show that it is a modernized application rather than having it look more old fashioned.

Rationale: With a modern look this application will appeal to the younger generation as well as the audience.

Fit Criterion: Once the application is downloaded, users will be coming back to the application for their needs.

Acceptance Tests: Professional look with modern twists and takes.

18 Operational and Environmental Requirements

18a Expected Physical Environment

The product will be used by people on their phones while either in their homes or outside. Since people may be at the store or the gym it can be expected that the user will be in a noisy environment when using. This means that the communication between the app and the user should primarily be visual not auditory. Brightness of app can be adjusted via the users phone and weather should not have any impact on the use of the app itself. Users may not have the opportunity to hear any sound that the app is making. Unless it is something like a tutorial video where the user clearly would need to listen as well as watch, the app should prioritize text and visual feedback for actions taken.

#82- Feedback

67

Description: The user can be either at home or outside on the go. Ideally, the app is most effective when the user is nearby an abundant number of restaurants, allowing them to have the most options available.

Rationale: The difference between the user being inside or outside has no direct difference as long as the user has a stable internet connection in both environments.

Fit Criterion: The user should ideally be stationary when using the app, since driving or walking through a city could be dangerous.

Acceptance Tests: Visual Audio Feedback Test

18b Requirements for Interfacing with Adjacent Systems

The app will pair with the users health app on their smartphone, this is so that users can be sure that the app is only accessing what is needed from their health data and that its doing so in a safe manner. The app needs to be compatible with the 'Health' app on iOS and 'CommonHealth' for android systems. The app needs to be compatible with the previous version of these apps to in the event of an update. These being properly connected allows for easy access of user health metrics such as steps taken in a day, active hours etc… which all contribute to calorie counts.

#93 - Health App

68

Description: The product software will be compatible with the four most recent versions of both the Health app on IOS and Common Health app on Android.

Rationale: The product needs multiple versions to accommodate for users who prefer earlier versions or are unable to update due to phone capabilities.

Fit Criterion: Using the four most recent updates allows for some flexibility in allowing users a version that would still be compatible even during updates.

Acceptance Tests: Health app connectivity Test

18c Productization Requirements

The product will be distributed via the Apple and Google Play app stores, there will be no training necessary to download the app since it comes in the same way any other app does. If there is any addition to the supported platforms the application will be released on that respective app store, since it is the easiest way for users to discover and download the app.

#91 - App Store

Description: Using the apple's app store and google play store allows users to download the app simply by browsing their respective app store and selecting the product.

Rationale: The app stores are built into the core programs of iPhone and Android devices.

Fit Criterion: As of now, this is the simplest way of providing the product to potential clients and users.

Acceptance Tests: App Store Test

18d Release Requirements

Each month there will be a bug fix and small additions patch, this is where the small tweaks and balances will be rolled out. There will also be irregular large content updates with an approximate launch of at least 2 per year. These updates will drastically change the way the users use the app such as having new friends lists or social media style feed addition. Each release will improve the app and not break any old features.

#66 - Update

Description: Each new release of the app is required to only make add-on or additions to the updates and not make changes to core elements of the product that may lead to or cause previous versions of the app to break or fail.

Rationale: Making significant changes to the apps core and rendering previous versions inaccessible lessens the user count and does more harm than good for the overall company.

Fit Criterion: The plan to keep at most the 4 most recent versions of the product accessible to users still remains and by spacing out the updates, users will have an easier time adapting and transitioning to more recent versions.

Acceptance Tests: Update criteria Test

19 Cultural and Political Requirements

19a Cultural Requirements

#21 - Culture

Description: The cultural requirements for this product includes anyone with intentions of preserving good self health. The product is intended to include people of all ages and not discriminate or offend obese people or over weight people.

Rationale: The application in its current state appeals towards a younger audience of individuals who are more familiar with both restaurant choices and mobile applications.

Fit Criterion: The goal is to get users of all ages and backgrounds to refer back to the application whenever they choose meals from restaurants.

Acceptance Tests: Emphasis on cultural impact and audience.

19b Political Requirements

#123 - Political Aspects

Description: The political aspect of the application involves being as open to as many users as possible while still maintaining full control of the product and who has access to it. Ideally, the political considerations involve making the product accessible to both IOS and Android, allowing not only users in the United States but also in other countries where those devices are popular.

Rationale: The CEO will ultimately be in charge of which groups of people are getting access to this product.

Fit Criterion: The application will be accessible in the US and countries that support android/IOS.

Acceptance Tests: Access Test

20 Legal Requirements

20a Compliance Requirements

ID# - Legal Needs

Description: All user data including login information, application food data choices, preferences, and all personal information will all be protected under the Data Protection Act in order to preserve user privacy and user confidentiality.

Rationale: Users should feel safe and secure using the product and should not feel discouraged putting personal information in order to allow the application to work fully and properly.

Fit Criterion: Protecting all user data from unauthorized outside sources is mandated through the Data Protection Act and will be honored.

Acceptance Tests: Legal needs unit test.

20b Standards Requirements

#78 - Standards

Description: The standard that is being pushed for this project is to adhere to all Data Laws that may conflict with the mal-usage of personal information being leaked.

Rationale: There is little to low expectation of any data being leaked but extra precaution for setting standards is required.

Fit Criterion: The standard for gathering and using data in the application will be monitored and secured to prevent the data leakage.

Acceptance Tests: Standards unit test

21 Requirements Acceptance Tests

21a Requirements – Test Correspondence Summary

Requirements

Test 1 **:**Macronutrient test
Test 2 nutritional information unit test.
Test 3 Accurate data unit tests
Test 4 Performance unit test.
Test 5 Speed test towards accuracy.
Test 6 Memory space test.
Test 7 Location services unit tests
Test 8 pulling data and loading it from specific location test
Test 9 menu display test
Test 10 Uptime Unit tests
Test 11 Availability to users and functionality working correctly

Test 12 Pandemic test
Test 13 mask mandate test
Test 14 Bug unit tests
Test 15 run time tests.
Test 16 Platform acceptance test
Test 17 Scalability test
Test 18 Security breach unit test
Test 19 Data Backup Test
Test 20 Data Clearance Test
Test 21 Audit Request Test
Test 22 Firewall Test
Test 23 Usability Time Test
Test 24 Language Test
Test 25 Tips and Help Test
Test 26 Understandability Test
Test 27 Accessibility Test
Test 28 License Agreement Test
Test 29 Training Test
Test 30 Color Test
Test 31 Visual Audio Feedback Test
Test 32 Health app connectivity Test
Test 33 App Store Test
Test 34 Update criteria Test
Test 35 Emphasis on cultural impact and audience
Test 36 Access Test
Test 37 Legal needs unit test.
Test 38 Standards unit test

21b Acceptance Test Descriptions

ID # - Name

Description: Acceptance tests must be able to satisfy the requirements of all requirements.

III Design

22 Design Goals

SV: Identify the important design goals that are to be optimized in the proposed design. Content

Design goals are important properties of the system to be optimized, and which may affect the overall design of the system. For example computer games place a higher priority on speed than accuracy, and so the physics engine for a computer game may make some rough approximations and assumptions that allow it to run as fast as possible while sacrificing accuracy, whereas the physics calculations performed by NASA must be much more rigorously correct, even at the expense of speed.

Note an important difference between design goals and requirements: Requirements include specific values that must be met in order for the product to be acceptable to the client, whereas design goals are properties that the designers strive to make "as good as possible", without specific criteria for acceptability. (Note also that the same property may appear in both a requirement and a design goal, so a design goal may be to make the system run as fast as possible, with a requirement that says any speed below a certain specified threshold is unacceptable.)

Your text goes here . . .

23 Current System Design

*SV: **IF** the proposed new system is to replace an existing system, then the current system should be described here. Otherwise insert a brief statement that there is no pre-existing system.*

Your text goes here . . .

24 Proposed System Design

This section will make heavy use of class diagrams, and also sequence and deployment diagrams where noted. However don't overlook finite state, activity, communication, or other diagram types as needed for effective communication.

24a Initial System Analysis and Class Identification

SV: Perform grammatical and similar analyses to identify the most import and obviously needed classes, and to organize them into an initial class structure. An initial class diagram is appropriate, containing few if any internal details.

Your text goes here . . .

24b Dynamic Modelling of Use-Cases

SV: Insert sequence diagrams of (at least the most important) use-cases, as a means of identifying other needed classes.

Content

Include sequence diagrams of each important use-case here. This is a first step towards identifying preliminary objects. (If the sequence diagram would be too big to fit, then it can either be broken down into pieces or a communication diagram can be used in its place.)

Your text goes here . . .

24c Proposed System Architecture

SV: Identify the Software Architecture to be applied to this project, such as Client Server, Repository, MVC, etc., along with justification for the choice.

Your text goes here . . .

24d Initial Subsystem Decomposition

SV: A slightly more detailed class diagram, showing the classes identified in sections 24a, 24b, and 0 above, partitioned into subsystems. For each subsystem provide a brief description of the subsystem, including its key responsibilities. There should still be few if any internal details.

Your text goes here . . .

25 Additional Design Considerations

SV: The sections listed here do not need to be presented in the order given, and may not all be relevant for any particular project. Those that are relevant can help identify additional classes that are needed as a result.

25a Hardware / Software Mapping

SV: This is particularly important for distributed systems, such as those employing a client-server architecture. Use a deployment diagram to indicate which subsystems are mapped onto which piece(s) of hardware, and what communication subsystems need to be added to the system as a result.

Your text goes here . . .

SV: The communication subsystem required for this program is a data retrieval system that constantly grabs data from the DIVVY API since it gets updated in real time.

25b Persistent Data Management

SV: Document the classes and perhaps subsystems necessary to store persistent data when the system shuts down, and to restore that data when the system starts back up again.

Reiterate key data structures and information as necessary for the understanding of this design phase. Refer the reader back to the data dictionary in section I7c above to avoid undue repetition, while reviewing only the most relevant items here.

Your text goes here . . .

The classes and data structures for storing location data from the API will be in the form of a set or a graph. The data will be loaded once at startup and then will update manually as the application continues to run.

The graph, set, and map data structures are useful since data retrieval is faster and data relevance is stored accordingly.

25c Access Control and Security

SV: Identify the access control and security concerns for this system, and the new classes and/or subsystems that must be added to handle those concerns.

Your text goes here . . .

To access the application, each user must login with their registered credentials and their data will be stored and saved in the system.

25d Global Software Control

SV: Identify the global software control concerns for this system, and the new classes and/or subsystems that must be added to handle those concerns.

Your text goes here . . .

The global software control concerns for this system would be controlling traffic of multiple users. Classes or subsystems to be added could be to link client/users to a shared server and allow the server to be the centralized system for handling traffic and client requests.

25e Boundary Conditions

SV: Identify the boundary condition concerns for this system, and the new classes and/or subsystems that must be added to handle those concerns. In particular consider startup, shutdown (normal or abnormal), and the creation and/or maintenance of any configuration files, databases, or similar supporting data files.

Your text goes here . . .

Boundary condition concerns for this application would include creating a server to handle and manage multiple users. This would prevent issues regarding users attempting to search and create new menu items from scratch rather than saving their data. Storing the users prior data into a server will allow the application to suggest other items of similar interest to the based on data already collected.

25f User Interface

SV: Include a preliminary user interface design here, possibly as a rough sketch or other mockup, in order to identify additional classes needed to implement the interface.

The final user interface design will normally be developed by appropriate experts in that area. However it is appropriate to include an initial design here, including possibly a low- or high- fidelity sketch/mockup, in order to identify key classes necessary to implement the user interface, such as forms and dialog windows. It may also go towards addressing usability and/or look-and-feel requirements, and/or identifying other overlooked components.

Your text goes here . . .

25g Application of Design Patterns

SV: Any design patterns applied as a result of previous sections should have been addressed there, and identified as such at the time. Use this section to document only the additional design patterns that were not previously covered elsewhere. (If any.)

Your text goes here . . .

26 Final System Design

SV: Include here the final version of the overall system design, incorporating all the subsystems and classes added as a result of additional design considerations. Multiple diagrams may be needed, possibly starting with an overall package diagram showing all the different subsystems and the (important) classes contained within each one. Still not a lot of internal details.

Your text goes here . . .

27 Object Design

This section documents the internal details of each class, to the extent that they can be designed at this time. Included should be the class interfaces (public method signatures and responsibilities) and constraints. It is probably best to break this section up into subsections corresponding to subsystems as documented above, and/or

by (Java) packages if those are designed. It may also be appropriate to address additional design pattern considerations here, but not to the point of being redundant of previous documentation.

Certain methods, such as simple getters, setters, and constructors are not always documented, unless there is something special about them such as in the Singleton or Factory Method design patterns.

27a Packages

SV: If the design involves assigning classes to packages (.e.g Java packages), then the packages to be created should be documented here.

Your text goes here . . .

27b Subsystem I

Your text goes here . . .

27c Subsystem II

Your text goes here . . .

27d etc.

Your text goes here . . .

IV Project Issues

28 Open Issues

SV: Issues that have been raised and do not yet have a conclusion.

Content

A statement of factors that are uncertain and might make significant difference to the product.

Motivation

To bring uncertainty out in the open and provide objective input to risk

analysis. Examples

Our investigation into whether the new version of the processor will be suitable for our application is not yet complete.

The government is planning to change the rules about who is responsible for gritting the motorways, but we do not know what those changes might be.

Considerations

Are there any issues that have come up from the requirements gathering that have not yet been resolved? Have you heard of any changes that might occur in the other organizations or systems on your context diagram? Are there any legislative changes that might affect your system? Are there any rumors about your hardware or software suppliers that might have an impact?

Your text goes here . . .

The application is currently not developed for mobile devices.

Content

Some unknown factors include runtime of certain algorithms, and memory.

Motivation

To highlight some important issues to be the focus for continuing this project.

Examples

Mobile development is large in two areas, IOS and Android. Figuring out which one is most suitable for our specific application is something we are still considering.

A plan to develop in both IOS and Android is also a discussion, but would still raise problems with workload management.

Considerations

More advanced and updated IOS and Android versions may be available in the near future that may make it more complicated to transition into a mobile platform. Planning on which version is available to us currently and what updates future versions will have is another event we are considering.

29 Off-the-Shelf Solutions

SV: Discussion of products or components currently available that could either be incorporated into the new solution or simply used instead of developing (parts of) the new solution. The distinction between sections 35 a, b, and c is subtle, and not very important.

Your text goes here . . .

An alternative solution for making the application more accessible to mobile devices would involve building up the same application from scratch. This is due to the fact that the current GUI version of the application is not compatible with mobile devices and could not be

carried over as of now. However, a new GUI would not require as much development time and work as the backend.