<Student Name>

Senior Project Proposal

Tasty Bytes: Nutrition Tracker iPhone Application

<Date>

Table of Contents

Summary………………………………………… Page 3

Significance……………………………………… Page 3

Required Tools and Availability………………… Page 4

Demonstration Plans…………………………….. Page 4

Qualifications……………………………………. Page 4

Project Specification…………………………….. Page 4

Technical Details………………………………... Page 5

Timeline…………………………………………. Page 6

Future Enhancements……………………………. Page 7

**Summary**

Tasty Bytes: Nutrition Tracker is an iPhone application that allows users to keep a log of the calories and macronutrients that they consume throughout the day. Users are able to track calories and macronutrients through two methods. The first method allows for manual entry of the nutrients directly into the user interface. The second method allows users to upload a screenshot of a nutrition label, which is then “scanned” with optical character recognition technology. The application then identifies specific keywords from the nutrition label (calories, carbohydrates, fat, and protein) and logs the numerical value of the respective nutrients automatically.

**Significance**

The scope of this project fits well with the Object Oriented Software Development course and the Human Computer Interaction course. The programming language that will be used to create this application is Swift. Swift, like java, is an object oriented program language. Many of the skills that were learned in the Object Oriented Software Development course, such as the creation of classes/methods, instance of the classes (objects), and the active manipulation of data held in the objects through user-interface interaction will be used to input, update, and store nutrition data. This project will build upon skills learned in Object Oriented Software Development, but will demand a much higher level of functionality as well as a more robust user interface in order to have an enjoyable user experience. To create a well designed user interface, skills learned in Human Computer Interaction such as, avoiding cognitive overload and innate user error prevention, will make users feel comfortable, and will avoid confusion while entering nutritional information and navigating the application.

**Required Tools and Availability**

This project will be developed in Xcode 8.2.1 with the Swift 3.0 programming language. Both of these are available on my personal computer. The optical character recognition that I will be implementing is OCR Tesseract. OCR Tesseract is open-sourced and available online for free.

**Demonstration Plans**

Demonstration will take place in the Julian building room 260. The code can be presented in Xcode from my computer. The application can be built and then presented on an Xcode iPhone simulator on my computer.

**Qualifications**

I have basic knowledge of iPhone application development with Swift and Xcode from an iPhone programming independent study that I completed during Winter Term 2016. During this course, to become familiar with the tools in Xcode, I built a variety of mini, sample applications. Additionally, the completion of courses such as, Object Oriented Software Development, Data Structures, and Human Computer Interaction has equipped me with the skills and knowledge needed to develop a fully functioning iPhone application.

**Project Specification**

The goal of this application is to make a quick, easy to use single-view nutrient logger. The user interface will present users with intuitive, editable fields that will allow for the entry, edit, and update of nutritional information. Fields will be labeled and separated based on category (Calories, Carbohydrates, Fat, Protein). At the bottom of the application view, users will be presented with the option to upload a photo from their personal library to use OCR. Users will be shown the information gathered from the photo and prompted for confirmation to use the information. Nutrition information will then be automatically updated and presented in the user interface. Users will set specific numerical goals to reach for each nutrient and will be reminded to drink water/fluids every two hours.

**Technical Details**

Manual nutrition data entry will be input by users and stored in UI labels. The OCR Tesseract will be installed into the Xcode project via Cocoa pods. A pod file will be created in the Xcode project and the OCR Tesseract data will be downloaded into the pod file. The OCR Tesseract framework will then be imported into the View Controller of the Xcode project. The English language file dictionary will then be downloaded from Tesseract iOS github and added to the Xcode project. The file dictionary may be altered later to search for words that are specific to nutrition information.

Words that are recognized by the OCR Tesseract will be stored in a data structure for later access. I currently plan on indexing the recognized words as strings in an array, but this data structure may change if it does not effectively index/access words that are needed. Once data is organized in the data structure, an accessor method will be used to locate the data and send it to its corresponding UI label for the user to view.

Reminders/Notifications will be implemented using the Swift API.

**Timeline**

Checkpoint 1:

* I will demonstrate the use of OCR Tesseract on a simple text image (.png file). This image will be used to show that text from an image can be converted into editable text. The image will be a screen shot of text that I have typed in simple text editor.
* I will display the recognized text in an editable UI textview in the user interface of my application.
* I will demonstrate the organization and storage of recognized text in an array.
* I will demonstrate the access and the display of specific elements in the array.

Checkpoint 2:

* I will demonstrate a single view user interface for manual entry of user nutrition data. User interface will include labels, textfields, and buttons for the display and update of nutrition information.
* I will demonstrate the use of a photo upload button. The button will access the users photo library and allow the user to select a photo to be uploaded for character recognition.
* I will update the user interface with nutritional information that was gained through the OCR Tesseract. This will be demonstrated to show the integration between user interface and the OCR Tesseract.

Checkpoint 3:

* I will demonstrate the use of the OCR Tesseract on an uploaded nutrition label. Nutrition labels will be found online and then saved in the iPhone simulator photo library. I will then access and the photo library from within the application and perform character recognition on it.
* I will optimize the OCR Tesseract for nutrition labels. I will show improvements by displaying the recognized text before and after optimization. Various methods to improve OCR Tesseract will be trialed prior to the demonstration. These attempted methods include, but are not limited to: modifying Tesseract control parameters, in-app image processing (rescaling, noise removal, rotation, border removal, etc.), page segmentation settings, custom dictionary/word list/ or patterns.
* I will document the optimization methods attempted and how I reached conclusions for the settings that I implemented.

Checkpoint 4:

* I will implement daily nutritional goals. I will demonstrate how the user sets goals as well as the notification that is displayed when goals are reached.
* I will implement and demonstrate a local notification that will inform the user that they should remember to drink water.

**Future Enhancements**

* Implement more detailed, in depth tracking. For example, instead of just tracking fat, the user may also track unsaturated, saturated, and trans fat. Nutrients such as vitamins, minerals, or specific ingredients may also be valuable to track.
* Implement a food database. Users can choose the food and nutritional information will be logged automatically.
* Upgrade OCR Tesseract to use a neural-network based recognition engine to improve accuracy.

**Bibliography**

<https://github.com/gali8/Tesseract-OCR-iOS>

<https://github.com/tesseract-ocr/tesseract/wiki>

<https://github.com/tesseract-ocr/tesseract/blob/master/doc/tesseract.1.asc#config-files-and-augmenting-with-user-data>

<https://github.com/tesseract-ocr/tesseract/wiki/ImproveQuality>