Group #2
Sayema Islam, Martin Tang, Kaitlyn Zhou
CSCI233 - Programming Projects Seminar
Professor Gennady Maryash
Capstone Project Paper
05.09.2018

The Final Product: The Application Explained.

For our capstone project, we have created an interactive, computer-based application that utilizes SQLite and TkInter of Python in order to allow a user to consistently keep track of his or her calorie intake. Essentially, every time a user consumes a food item, he or she should enter that food into the app by selecting it from a provided dropdown menu. Each listing in this menu is connected to an average calorie value stored in our CalorieTracker database, and upon choosing a specific food, the user is able to view said item's calorie value. This same value is entered into a repository of all foods consumed by the user, each with timestamps of when the food was consumed. And using this table, all the foods consumed on the current day can be summed, thus also providing the user with a view of their total daily calorie intake.

The Journey: A Snapshot of the Semester.

Originally, we aspired to create an app aimed to provide a means for users to obtain estimated blood sugar levels in their body. The user would input a blood sugar reading obtained from a blood glucose meter at the start of the day, and for the rest of the day, he or she would simply input any foods items consumed. Using a database of nutritional information of various foods (i.e. sugars, carbohydrates, etc.) as well as an algorithm for converting consumed sugar amounts into metabolized sugar level changes, the app would allow users to roughly keep track of their blood sugar levels without having to prick their fingers several times a day, a concept

that we felt would appeal especially to diabetic patients. However, as we started to populate our database with nutritional information, we came to realize that coming up with a static algorithm for the app would be quite difficult given the idea that different users would have different metabolic rates for sugar breakdowns, and that the app could not account for sudden sugar spikes or drops. It remained an imperfect idea. Thus, using the repositories of food information we had already begun to input into our database, we decided to create a calorie-tracking app instead. It could work off of the same tables already created for our original app--a user information table, a food intake table, and a stored nutritional information table--with some minor adjustments.

Having firmly decided what our app should be capable of doing, the biggest issue we now faced was finding a way to connect the calorie values stored in the database to each food listed in the dropdown menu, so that picking an item would return a value on the GUI. Given that the SQLite values were being taken as strings in order to be executed as Python commands, the values were not being returned as integers (or floats) as we needed them to be, yielding parameter errors every time the program was run. But with some assistance from peers, we eventually learned that this issue could be fixed by committing *each* menu selection to the database separately (as opposed to committing all the menu selections as a whole), and then saving each of these values in the terminal as integers as they were being chosen, before being transferred to the appropriate table in our database.

Another big issue we encountered was in making the labels in our GUI dynamic so that the field for “total calories consumed” would be able to update the total number of calories without the user’s having to close the app (originally, this value was only updated when the application was closed and reopened). Interestingly enough, we found that this issue could be

resolved by similarly committing *each* menu selection (i.e. calorie value) to the database individually, i.e. placing the label's execution within the same, recursive, saving function. In other words, the function would be executing constantly, each time a new food item was chosen and inputted.

What We Learned.

We all learned a lot during the course of our creation of this application. This was one of the first larger, independently-developed projects that any of us had ever created, and we found that even the creation of a "simple" app is no easy task. In terms of SQLite, we discerned the importance of considering each tuple of data individually, as well as of being *extremely* attentive to datatypes. As for the use of Python, we discovered the importance of first mapping out one's ideas using words before proceeding to implement the necessary code. And more than anything, we were faced with the importance of organization in our code, especially in terms of packing the components of our GUI--each function call must be considered carefully as part of a larger whole that brings any GUI together.

Going Forward.

While our app has finally become a fully-functioning program, there are still some potential improvements it can see, moving forward. For one thing, while our database holds the capacity to store the information of multiple users (i.e. using foreign keys linking unique inputted food ID's to specific users and their respective information), as of now the app is designed for use by one user only. Additionally, our database also takes care to record the date and time associated with each entry, opening up the potential for an automated reset of user calorie totals each day. However, as of now, instead of having the intake table reset daily, it includes all

calories ever consumed and logged on the app within the CalorieTracker database, but returns only a total for the calories consumed on the *current day*. Interestingly enough, we had not considered this idea when planning the features of our app, but this may actually work better than implementing a daily reset, as it would potentially allow the user to input foods that he or she may have forgotten on a previous day.

Thus, we plan to work to polish our app moving forward, improving these aspects by employing the use of user login prompts, and more datetime-based functions, respectively. We are even considering an additional dropdown menu for choosing a previous date for which one wishes to input and/or view calorie totals, outside of those for the current day. And of course, we will continue to update our own food-calorie repositories as well; there can never be too much information after all!

So, where will our code lead us next? Truth be told, only time can tell, but we are quite excited to hop along for the ride.