John O'Neill

COMP 4620: GUI II

Wenjin Zhou, Ph.D.

5/3/2022

Final Individual Report

Preface: Throughout all of mankind's history, people have had to get things done. Maslow's hierarchy of needs describes the path humans take on their journey towards self-actualization. For our ancestors, these needs were as simple as procuring food, shelter, and clothing. The next objective is securing safety and stability in one's own life, such that resources like food and shelter are no longer constant worries. After that people can begin to cultivate their sense of group belonging and love, whether that be among friends, family, or lovers. Only then can one begin to establish esteem in oneself, edging closer and closer to that goal of self-actualization which, regrettably, is never achieved by far too many.

Why might this be the case? What asymptotic barrier deflects and prevents people from achieving their goals and being the best person that they can be? There exists no one definitive answer to this question, but a consistent pattern among these failures is that some may lose sight of their goals. Once the ever-gnawing pangs that come from being hungry, cold, or anxious subside, there isn't much to keep a human being from doing just about whatever they want, even if they know that it is counter intuitive to their own actualization. Even having established a long-term plan for one's own future, it is all too easy for one to get lost in the moment and become distracted by short-term diversions.

The conventional solution to this repeatedly demonstrated human shortcoming was the advent of the "to-do list". Whether that pertains to a collection of sticky notes, a checklist of action items, or a series of alarms on one's phone, there is a purpose shared across all modes of list keeping, that being to keep track of goals or objectives in one's own future. Instruments of this kind have served mankind well over several millennia, allowing them to stay on top of their own goals, duties, and responsibilities. By using a list, even if someone is distracted by a temporary diversion, they can simply refer to their records and know what remains to be done.

Conventional lists are not without their shortcomings, however. First and foremost, a list is typically only as effective as its maintainer allows it to be. For example, if the list's maintainer consistently marks their tasks down, checks the list, and acts on that list, it serves the user well and allows them to stay on top of their tasks very easily. Regrettably, not everyone is this vigilant. Some individuals will write a list to stay on top of their tasks, but ultimately forget to check it when the time comes. Some people aren't so readily on top of their duties and may forget to add items to the list which might otherwise be important. In the worst-case, some users are apathetic to the idea of their list and will intentionally shirk duties on the list for just about any reason that doesn't jeopardize the lowest rungs of the hierarchy of needs. "Wash the car? Fix the microwave? I can get along just fine without them," is a common rationalization for ignoring one's own list.

In this information age that we live in, distractions are more readily available and prevalent then they ever have been in the history of our species. It is not all that uncommon for someone to pick up their phone, and when they finally set it down realize that they just sunk 5 hours into mindlessly scrolling through TikTok, YouTube, Twitter, et cetera ad infinitum. It is more difficult to maintain a list than it was 40 or even 20 years ago, as the sheer amount of information vying for our attention each and every day has multiplied upon itself exponentially thereby obscuring the "to-do list" from our sights.

What is the point of this diatribe, then? If list-keeping has seemingly become a hopeless endeavor in our current era, what more is there to be said about it? The answer is that there is in fact much more growth the to-do list as a concept is capable of. If the defining feature of our information age is said info constantly vying for our attention, why can't the list be that info and ask for our attention in its stead? What if there was an incentive to engage with the list which simultaneously disincentivized apathy? What if people could finally start to get things done once more instead of losing their day to commonplace time-sinks?

Project Overview: This line of questioning implies a solution, and our team's answer was "Get it Done". Get it Done is a web-based application that centers around the incentivization of task completion by gamifying it. Like any good list-keeping application or system, Get it Done allows users to dynamically set tasks that they must complete in the future. On a normal task manager, completing a task amounts to simply checking it off as complete without any payoff. Get it Done rewards the user for persistence and gives them an incentive to return. This solves the issue of apathy, as even if the user could live without doing a task, leaving it incomplete has them lose out on the possibility of a reward. This results in more consistent use by the user, more tasks added, and more tasks completed. This consistency in terms of userbase is a feature that would make the application highly valuable to users and more marketable to advertisers, resulting in a positive outcome for both user and developer.

Related Work: Many user persistence features implemented throughout Get it Done are inspired by applications whose purpose is different but application is the same. First and foremost among these is Duolingo, a language learning application that encourages the user to engage with it consistently and rewards engagement. Different components of the language are broken into categories represented by that selfsame "donut" system mentioned previously; the more the user engages with a category and attains mastery, the more the donut fills up, and the user is rewarded with virtual currency they can use to redeem cosmetics for their profile. There is also a "streak" system, made popular by Snapchat but also seen here, which encourages users to work on learning their language every day so as to avoid "losing their streak." This encourages engagement with the application, feeding back into its gamified loop.

Other sources of inspiration were games developed by Valve, which have shown a successful model of encouraging user engagement through the proffering of cosmetic rewards. Even if the price-tag of the game is free, if the user is given the option of changing cosmetics they have been shown to engage more with the game.

A final source of inspiration was MMOs of all stripes, like WOW, FFXIV, or otherwise. It has often been said since the advent of MMOs that these games oftentimes have job-like qualities to them. The core gameplay loop is engaging on its own, but there are also daily and weekly events as well as real-world timers players must keep up with to get their "full value" out of the game. These sorts of "limited time offer" style events have been shown to dramatically increase user engagement and are a big factor in why MMOs are so popular today.

System Specification: Our final project has a welcome page, sign in with Google OAuth integration, a "how to use" page, habit creation page, to-do list in both sticky note and traditional format, and user achievements page.

The welcome page shows the user what can be done with "Get it Done" including its convenient trackers, sticky note and traditional task views, and rewards like cosmetics and achievements. The user can sign in by either traditional email and password or by using OAuth to connect their Google account to the website. The signup page has form validity checks to make sure a proper email and password are provided by the user.

The “how to use” page simply and concisely explains to the user the main features of the application and how to access them. Each feature is demonstrated using helpful visual guides and tooltips explaining what each field or button is used for.

The user can use the habit creation page to establish new habits for themselves. They can choose a name for the habit, describe the task, choose the frequency of the habit and establish the length of the habit. These habits will appear in their task list with the frequency and time specified. If the user decides they would like to end a habit prematurely, they have the option of deleting their habits on a separate page.

The to-do page functions like it would be expected to, but with two twists on the concept. The first is the option to view the tasks in a sticky note format instead of a list format. This is useful for users who prefer to think about things visually or get a big picture perspective of what they need to get done. The second iteration on the traditional to-do list is the donut completion feature. Presume that a user has 4 tasks to complete in a given day. If the user completes 3 tasks, the user can see that their daily completion is 3/4. This incentivizes them to complete the final task and close the donut, as completing donuts contributes to their chances to acquire a cosmetic. After completing enough donuts, users have the opportunity to redeem one out of three sticky note unlocks, each of which are obscured by a question mark. Once one is chosen, the cosmetic is revealed and added to their collection. The user can choose to set cosmetics as either their task or habit sticky note. This gamification of the to-do list model serves to encourage people with completionist tendencies to engage with the application, for doing so gives them the opportunity to unlock a wider array of cosmetics to use.

Another incentive to engage with the application is achievements. Users can see achievements that they have active progress towards on the achievements page. These achievements can be unlocked by a variety of means, whether that be setting a certain number of habits, completing a certain number of tasks, or unlocking a certain number of sticky notes. Achievements which have been unlocked can be seen separately in the user’s hall of fame page.

Project Design: Our program was initially designed with those system specifications already discussed in mind. Most differences that can be found between initial design and final implementation differ by the method in which these features were implemented. What differences were observed between the design and implementation versions of the applications will be discussed in the “future work” section.

In our initial implementation, we had planned to use React for our framework, MongoDB for our database, and Google Charts for specific GUI implementations. React persisted to the final implementation, but MongoDB and Google Charts were superseded by other solutions.

Discussing the lesser of the two differences from the outset, we didn’t end up implementing Google Charts because we had no need of it. Since we were using the React framework we had access to npm. By using npm we could install plenty of useful packages to work with react and give us the sort of GUI reactivity we were looking for.

The greater difference shown between design and implementation was the database solution. MongoDB, as aforementioned, was originally planned for implementation with our website. There proved to be complications surrounding its use in our project, some pertaining to ease of use and source control issues but most pertaining to cost of hosting. We are all students, meaning most of us are financially challenged. When this issue made itself evident, we knew a different solution must be pursued. For a spell I personally did some research and development with SQLite as a database solution. I had already written some in-team manuals and set up a basic data structure by the time I learned that there was an issue with this solution as well. GitHub Pages, the platform we were hosting on at the time, only hosts static content. This means that if we were to write JavaScript to interact with our database, the changes would not be

written and would ultimately be entirely ineffectual. Ultimately, we switched solutions once more and settled on Firebase as our database solution. Firebase is a cloud-based database service with a generous free plan. It had just enough data usage and allotment for a project of our size, and allowed us to write to the database while we still used GitHub Pages.

Team Work: Our team was comprised of five individuals. The project lead, Rebecca Alves, was the creative and inspirational lodestone of the group. She drew the original prototypes for the website and demonstrated strong conviction in her proposal by guiding development. Michael McGilvray was the lead developer of the project, making most of the implementation decisions and establishing most of the project's framework. Caleb Bergen was in charge of user progression, and likewise it was his responsibility to refine the user experience and ensure that user incentives like cosmetics and achievements were balanced and rewarding. Ankit Lalotra was a programmer who primarily worked on user sign-in and Google authentication with OAuth. John "Jack" O'Neill, that being myself, was a programmer who primarily worked on research, development, and implementation of the website's backend database, as well as some other quality of life changes on the frontend.

User Testing: Our user testing went relatively smoothly and produced fruitful results. Most feedback we received was positive, and what constructive criticism we did receive served to improve our project. One of the pieces of feedback we received is that, at the time, the user could create habits but could not delete them. The user now has the ability to delete habits on the "Edit Habits" page. Another recommendation was that the title header "lists" was too general and wasn't self-descriptive. We solved this by changing the header to say "to-do lists". Other than that, there were no glaring issues with our program and got an indication that we were on the right track in terms of development.

Future Work: Our current implementation lacks some features that would be good to expand upon if given another semester to work on it. A monthly to-do list like a calendar view would be a great way to get a "big picture" perspective of one's upcoming tasks, as our current implementation only allows for a daily and weekly view to be viewed. Additional sign-in methods through websites like Facebook or Twitter would be convenient as an alternative to Google sign-in or sign in using email. Our program also originally planned to have movable and collapsible sticky notes, which are not present in this current implementation. This would be helpful in organizing and categorizing tasks with similar qualities.

Closing Remarks: Ultimately, I am personally satisfied with what we have achieved here. The team faced many setbacks, both in terms of development and in our personal lives. Despite the adversity we all faced, we managed to deliver a product that is impressive in both scope and functionality given the four-month development period. "Get it Done" answers the question of engagement in today's information occupied era. Where one might otherwise become distracted or diverted and fail to meet their own goals as a consequence, our project allows its user to cross that mental gap and stick to their plan by making it fun and rewarding to engage with the application. As easy as it may be to lose sight of things in the information age, applications like "Get it Done" can help one be the best that they can be, bringing them one step closer to self-actualization.