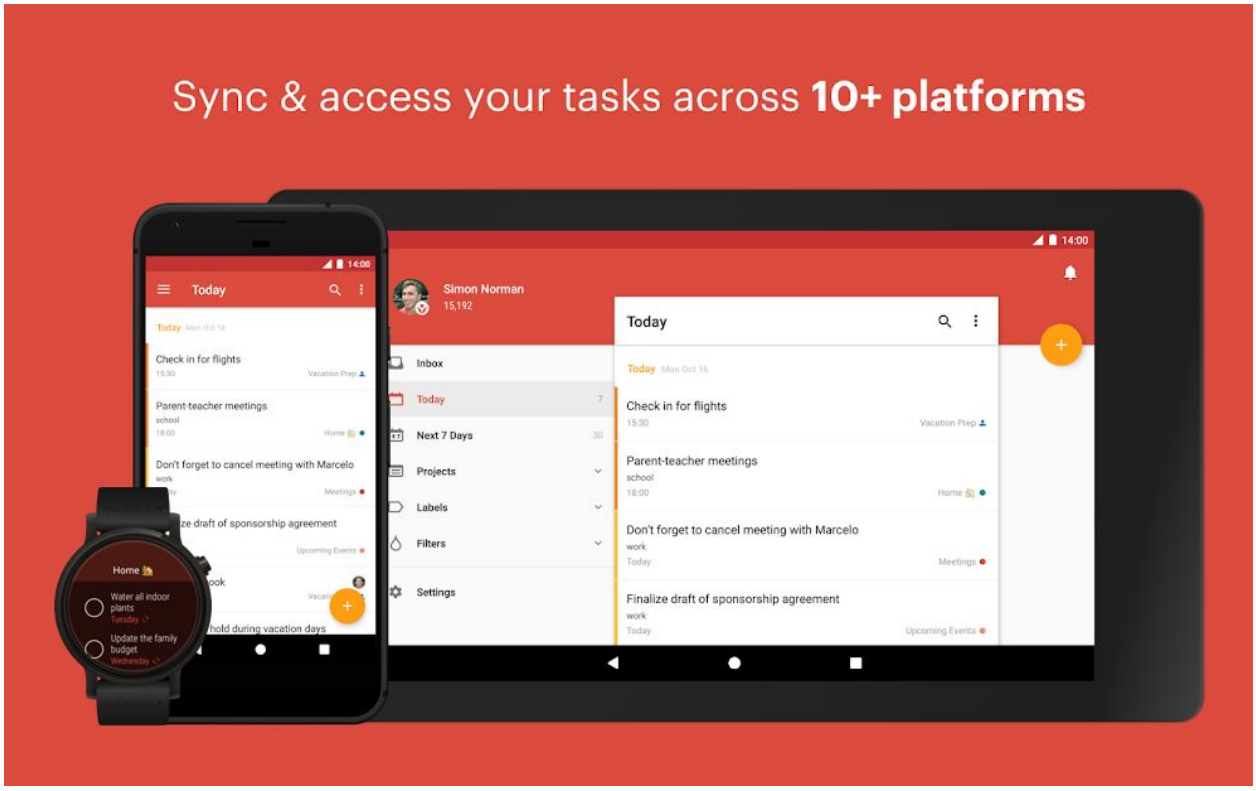
Improving Digital Task Management

You're probably familiar with the myriad of to-do list apps available like Wunderlist, Things, and, my personal favorite, Todoist. All of these apps essentially serve the same purpose: to make managing tasks easier.



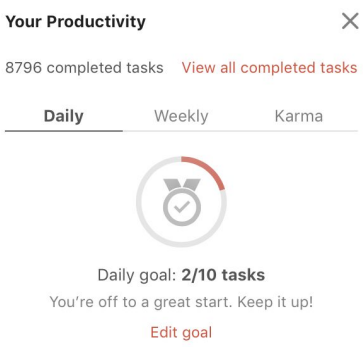
I can hear the cries already.. "But why not just use a pencil and paper?!?" Well, as it turns out, making the to-do list digital allows for some pretty darn useful features. For example, you can organize your tasks into projects and subprojects, set up recurring tasks, create task templates, and get some basic analytics. Plus, most to-do list apps will automatically sync your list between devices, so you don't have to worry about leaving your physical to-do list behind. All things considered, there's just more flexibility.

Don't get me wrong—I know that certain methods work better for others. If you find that the pencil and paper works best for you, I'm not going to insist that you switch to a to-do

list app if it's going to make you less productive or more anxious. So, if you've already tried the digital task management life and *hated* it, there might be better ways to spend your time than reading this article.

During my Fall 2018 semester at the University of Michigan, I took the two-credit ENTR 390 digital product design class. There were two major projects and, since I cringe a little whenever looking at my first project, I've decided to share [my second project].

For this project, I decided to tackle some of my biggest qualms with digital task management. While my years of extensive use of Todoist has helped me be more productive and organized, I've found the app doesn't really extend beyond being a great to-do list app. While it does employ a



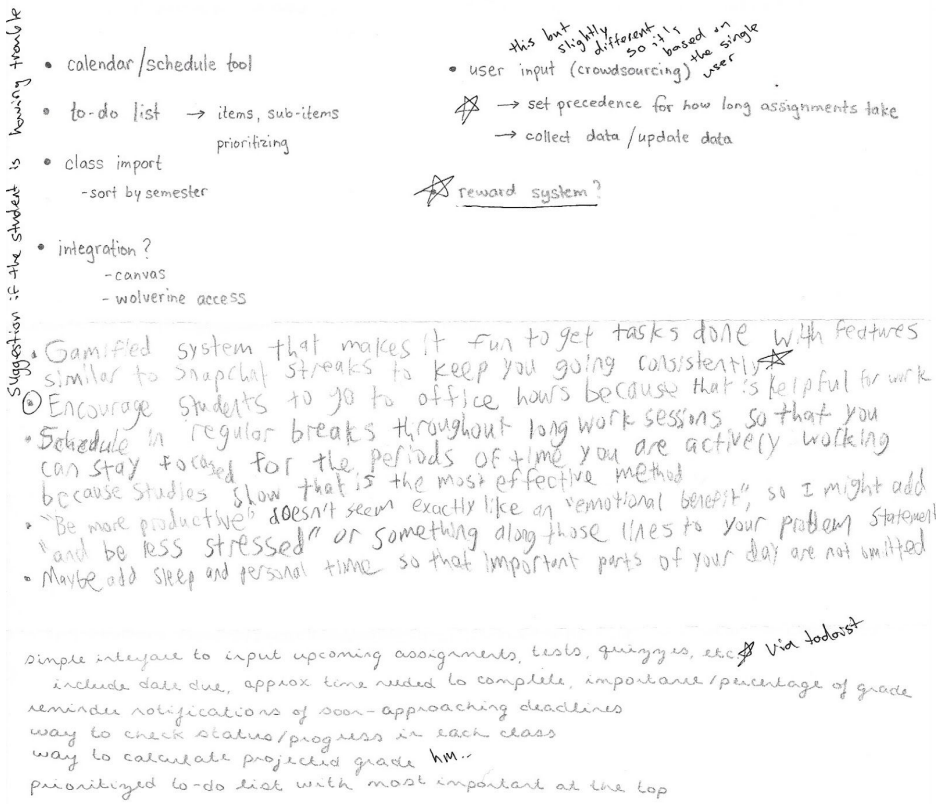
karma system to encourage you to complete more tasks, it doesn't take an active role in helping you manage whatever is on your list. I've yet to find an app that does, though, so I decided to design one.

Before drawing some sketches or writing out a feature list, ENTR 390 taught me to start with a problem statement. It's supposed to be short and sweet, but ultimately should guide all of the design work to ensure that the problem is best addressed. Here's mine...

Students
want something to
help them track how much time they spend on their assignments
so they can
feel less anxious when deciding what to work on.

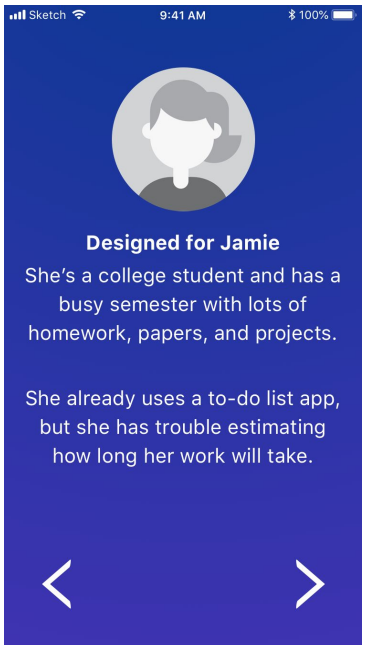
While I set up my problem statement to target busy students like myself, I also wanted my app design to be helpful for anyone with a lot of different tasks to get done.

Once I had my problem statement down, though, it was time to get some feedback from some of my classmates. Here's what they came up with...



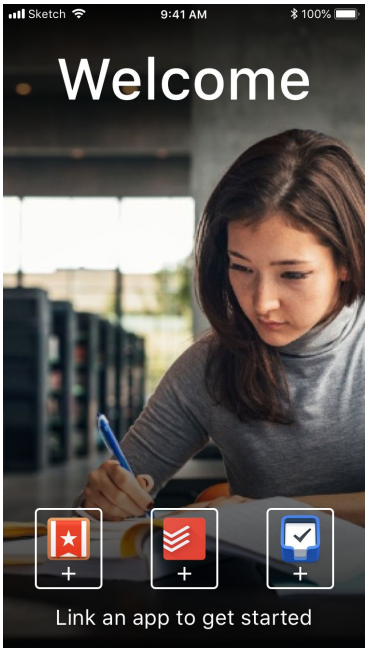
Initially, I was unsure of what else my app might entail in addition to basic time tracking, but the feedback above really helped me determine what else I wanted: some positive reinforcement, breaks, etc.

When I finally dove into Sketch, I immediately knew that it simply wouldn't make sense to attempt to design another to-do list app. There are plenty of great apps out there already and most people using them wouldn't want to switch to a new app anyway, even if it offered some extra features. Since most to-do list apps share their data via APIs, this would be feasible through simple integrations.



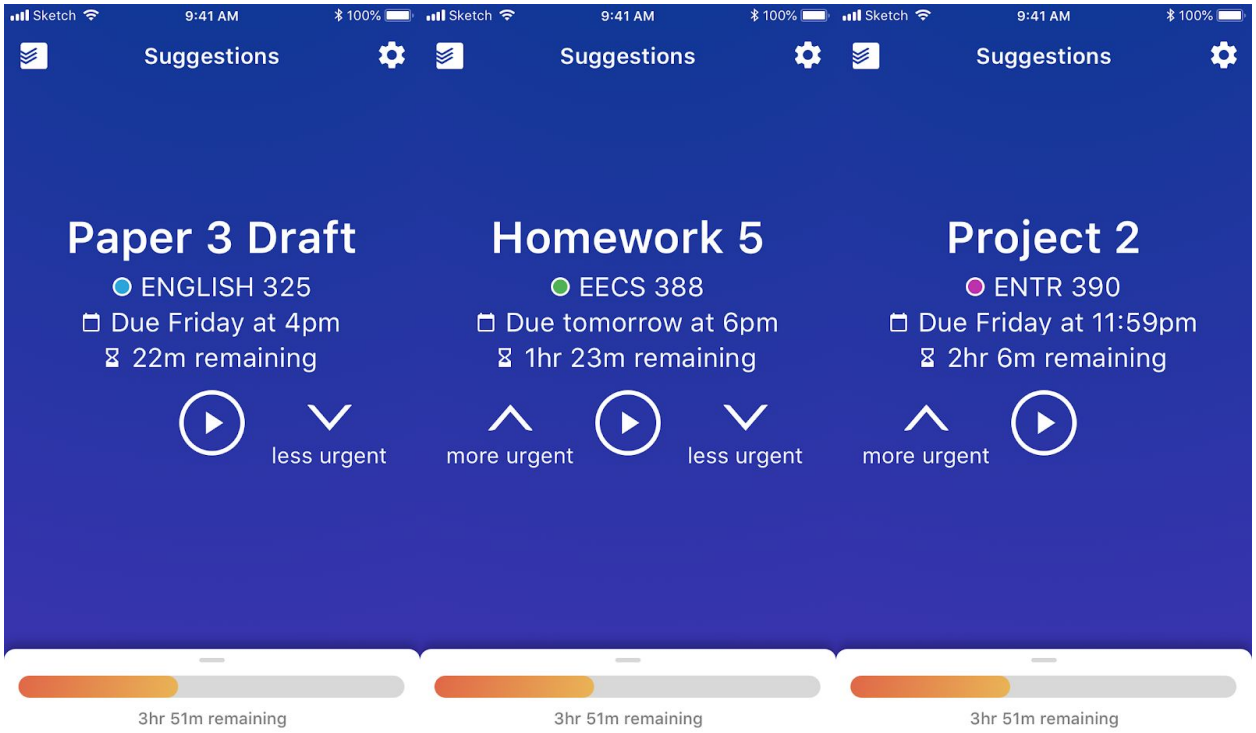
While I limited my design to supporting three different to-do list apps, the overall app design would stay pretty much the

same regardless of how many to-do list apps were supported, since they all supply the same basic data.



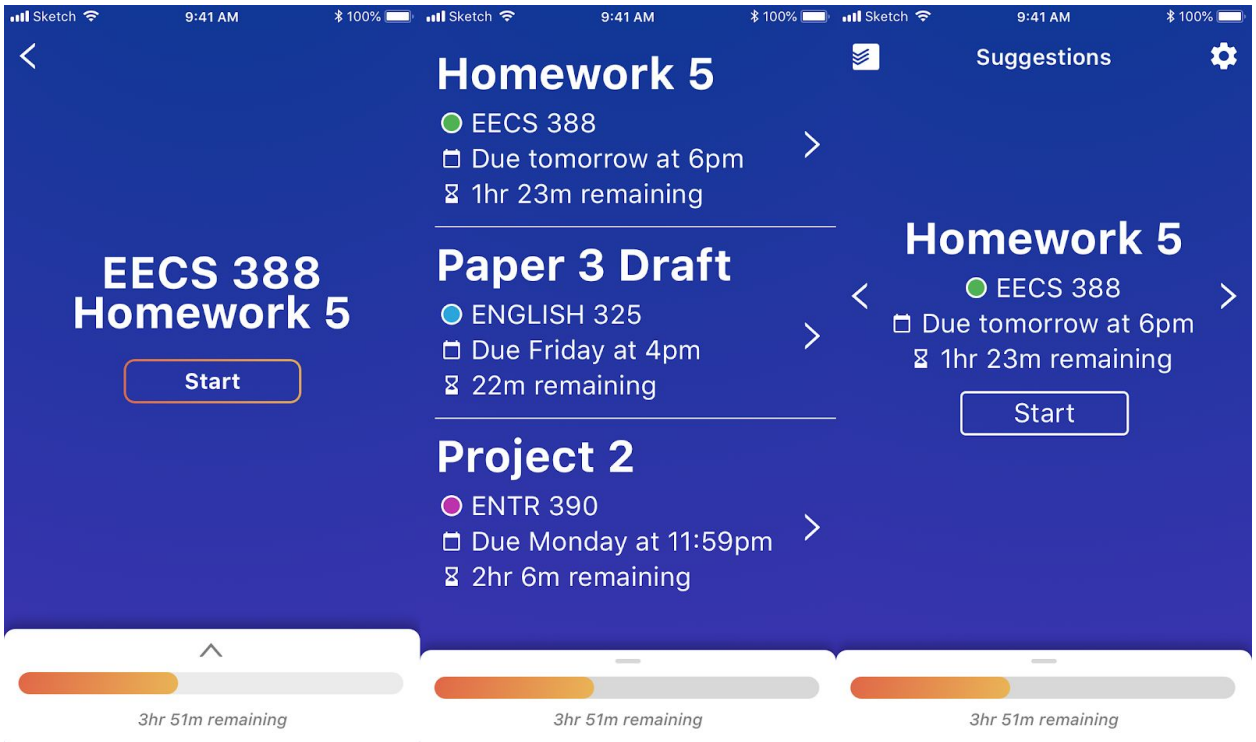
I wanted the primary focus of my app to suggest something for the user to work on. While it would take some time for the app to learn how long each task generally takes, a well-trained technical model would be extremely helpful in eliminating the overwhelming feeling that can come on when staring at a day's to-do list with 40 items on it.

So, this is exactly what I did. When the user opens the app, they would see the "most urgent" task to be completed. If it's not something they'd like to work on at that very moment, they could use the arrows to the side of the start button to navigation between suggested tasks.



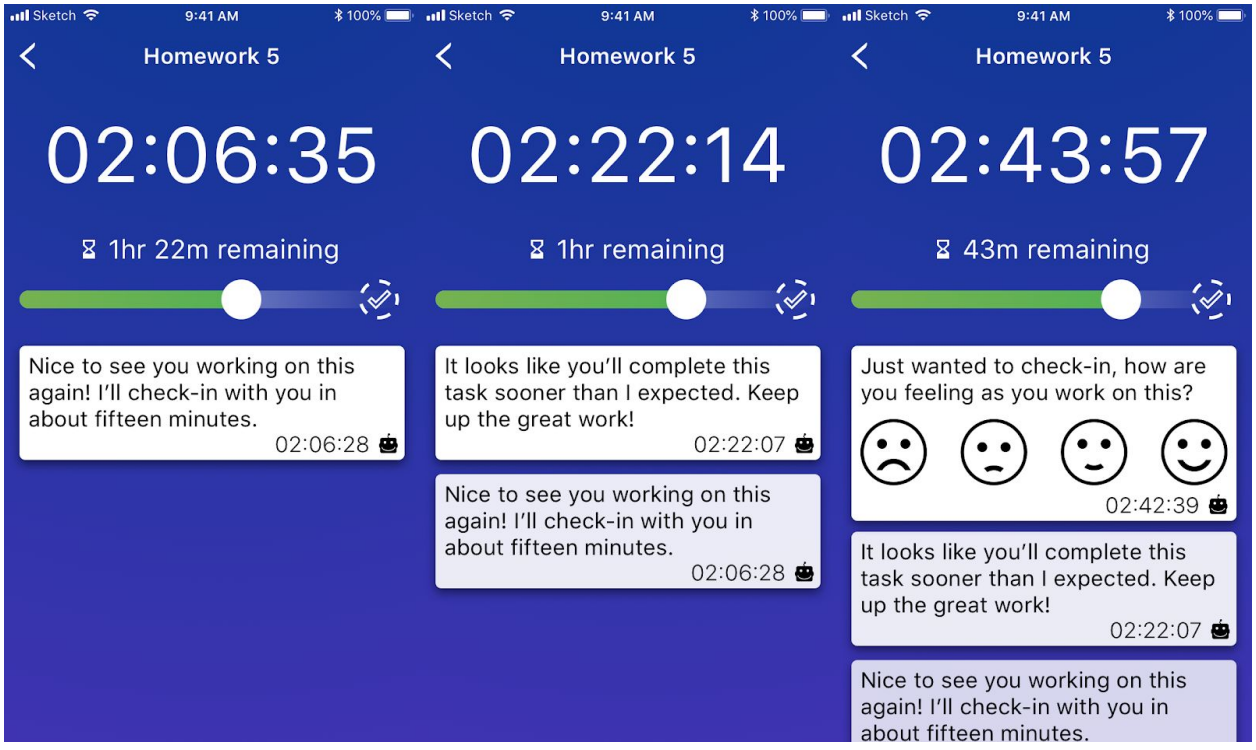
I ended up going through six different major iterations of this main page design as I was struggling to decide how exactly to present suggestions. I started out with a single task and a start button, but it was clear that the app would need to suggest more than one task at a time. So I decided to try a list of tasks instead; I also slimmed down the productivity pull-tab in that iteration.

But, I then realized that this type of list design could easily go against the emotional goal set out in the problem statement. When I showed some of my peers the list design, the response was generally that the text was too big. But the solution—slimming down the text—would require showing more suggestions on a single page, which is something I absolutely did not want to do. At that point, I thought the list design was starting to get too overwhelming as is. So, I scrapped that idea and returned to the mindset that presenting one suggestion at a time would be best.



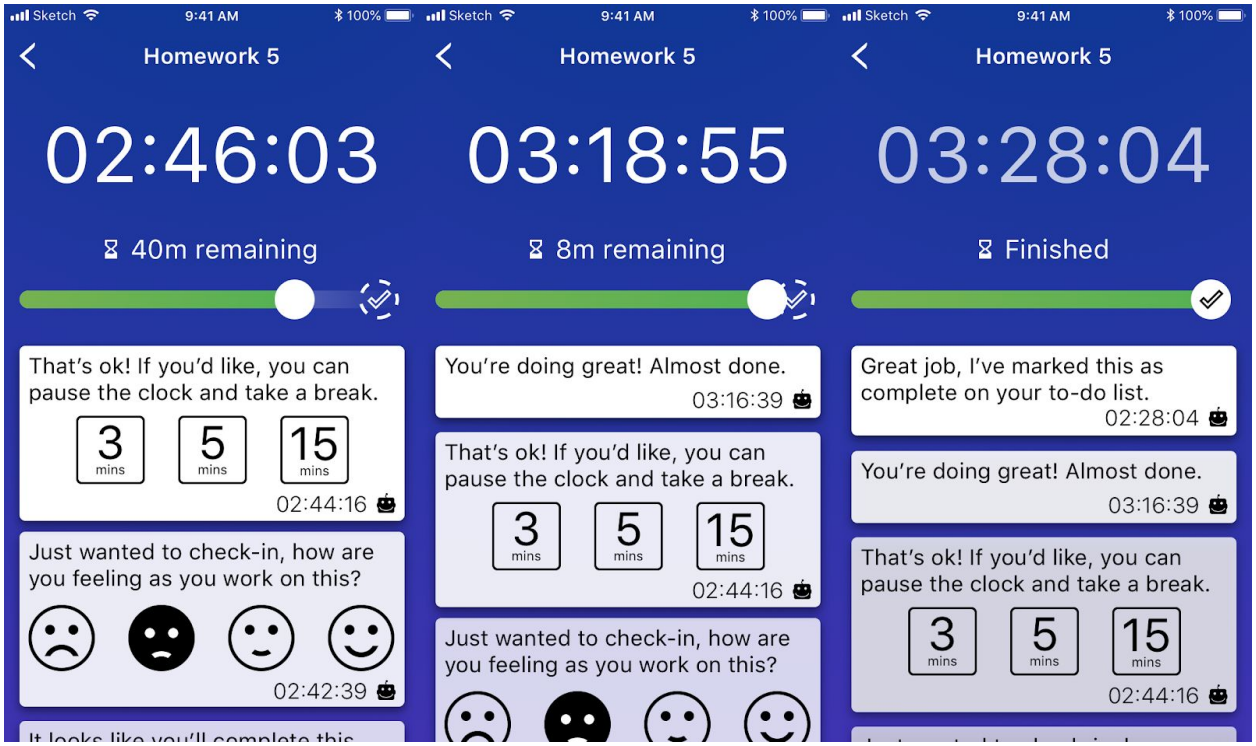
The next hurdle was those gosh darn arrows. I think I spent somewhere close to three hours figuring out how to deal with the arrows. In my defense, there was a lot to consider: how placing the arrows on the sides might affect the amount of content that could be displayed, how placing them above and below would make the “Start” action feel redundant as the user went through their suggestions, and how the arrows could be confusing without a bit of text to suggest their purpose.

Once I got the main screen down though, it was time to focus on the second most important part: the stopwatch view. This screen would be shown whenever the user tapped the start or “play” button for one of the task suggestions on the main screen.



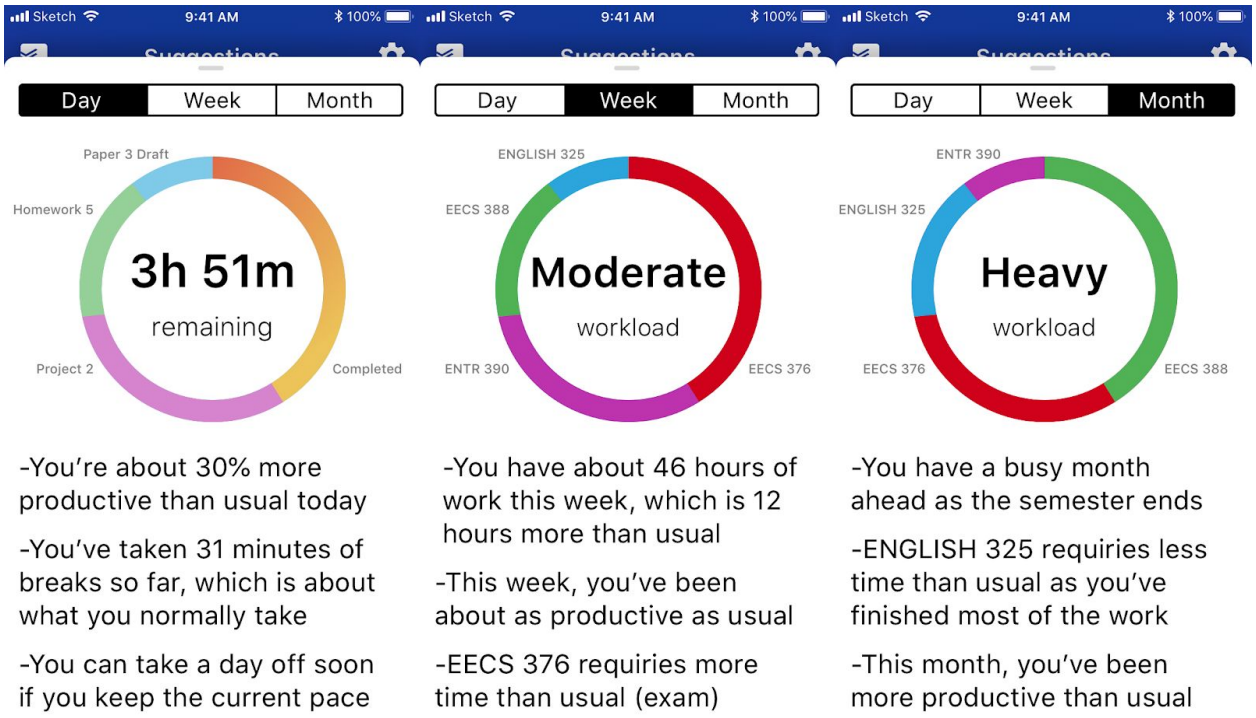
The idea for this screen was to show the amount of time spent on the task, offer an estimated time to completion, allow the user to manually update a progress bar, and, most importantly, find a way to guide the user through the task.

I decided on a notification-message hybrid style for anything that the app would need to communicate with the user. As new “messages” came in, the older messages would move down. Each shift would cause the background opacity of the message to drop by 10%, in order to better emphasize what the user should focus on. (And, yes, I’m responsible for designing the tiny robot icon in the bottom right corner of each message)



When working on different messages that the app might present, I inadvertently found that my break time selections looked a heck of a lot like a speed limit sign. But, I think that this actually works quite well. Taking a break, by definition, requires you to slow down. So why not keep a design similar to what we see as a signal to slow down when driving?

Aside from a few minor aspects, there was only one part left to design. And that was the productivity pull-tab I mentioned earlier. Admittedly, I was pretty close to the project deadline when I started working on this, so it could definitely still use some work.



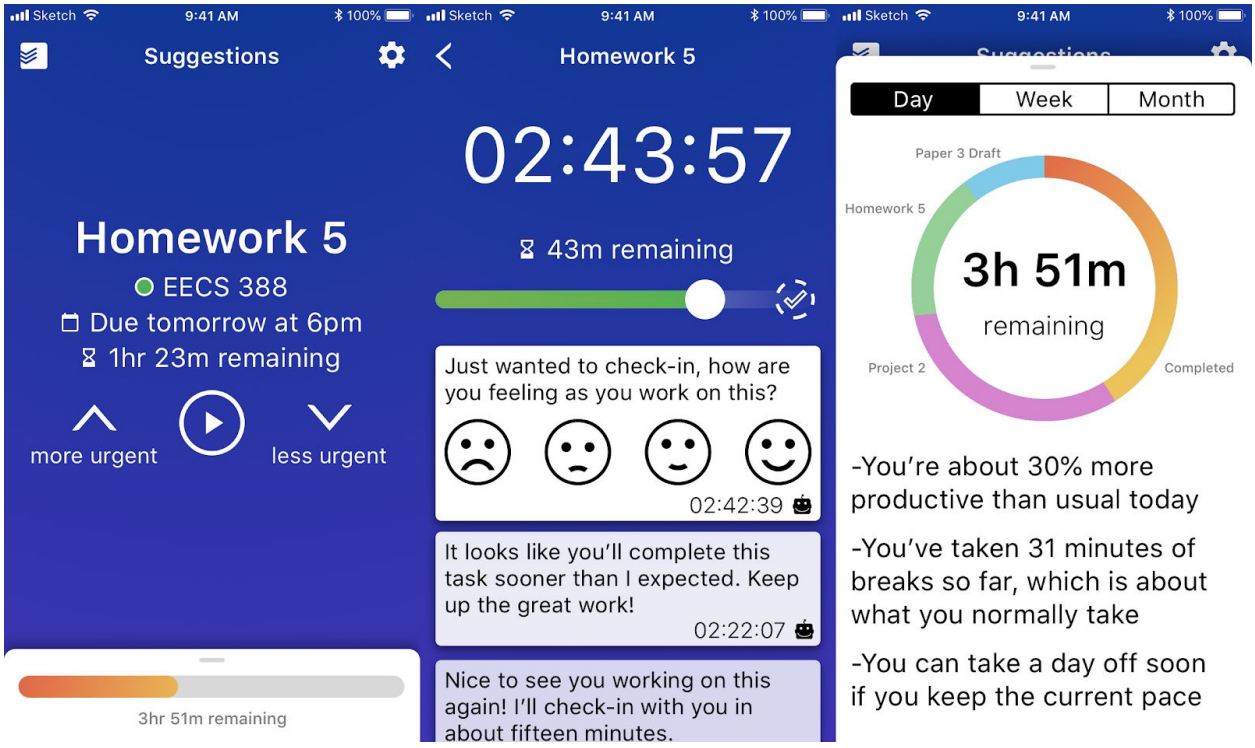
In general, I'm unhappy with how insights on each of the sections are currently presented. It's not only pretty text-heavy (at least, relative to the rest of the app), but there's currently no rhyme or reason. I suppose the main cause of this was not knowing exactly what a user would want to see when pulling up on this tab. If I had the time, I would've conducted some research to determine what should be shown for additional insights.

For the day view, I like the general idea of showing a circular progress bar. Although, if I had the opportunity to work on this more, I would change each section to represent a project (or, class, if those are what the user has as projects in their to-do list app) rather than a specific task since there could very easily be multiple task suggestions that fall under the same project.

I think the week and month circular progress bars are *ok*, but I'm not sure if I'd keep them in another iteration. They basically only tell the user which classes might take up most of their time moving forward, and how heavy their workload is. While these might be valuable insights, I think there might be better formats to present them in which would free up the space for something perhaps more useful.

Overall, I'm happy with this design. I think handing it off to a software developer would set some pretty clear expectations for what the product needed to focus on. And, based on the feedback I've received so far, I think most people would see the value in this design.

I know the idea of intentionally and heavily restricting how much content is displayed might seem crazy in a world filled with content-heavy apps, but I think doing so is refreshing in a way and is consistent with solving the problem statement.



My goal was to keep this idea in mind throughout the design process. It's why there's a barely-noticeable, dark blue-purple gradient as the background, white sans-serif text to call focus to certain items, simple but clear buttons, minimalistic icons, and a simple onboarding process.