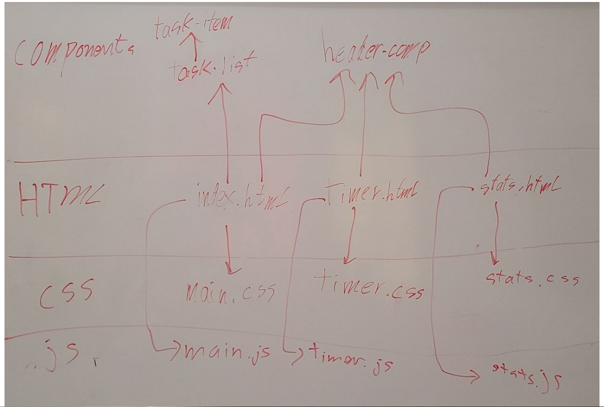
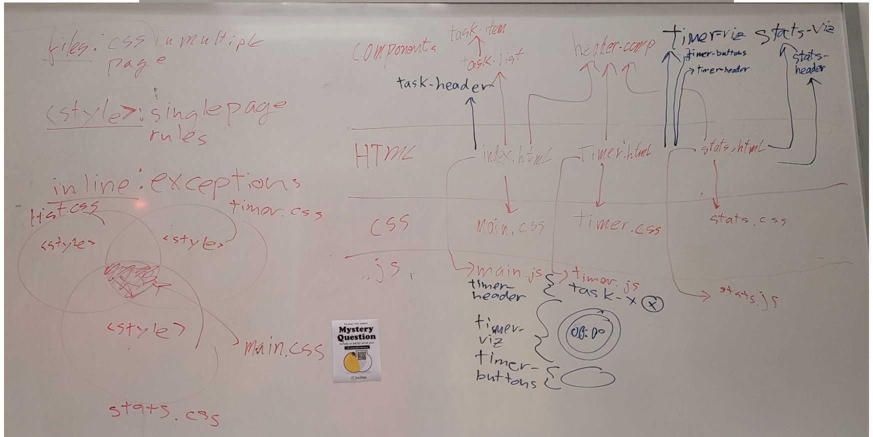
Codebase Structure Revisions

Some things sketched up during the meeting that I feel would important to talk about. The first image is a mockup of our current system (the one with only red marker). It is separated by HTML, CSS, JS, and components. The 2nd image talks about changes to be made. Notably there are two big picture ideas for these. The first, and simpler one to implement, is the notion that we should have a universal main.css file which houses all the rules which are shared by all three pages and then throw all the remaining styles into the HTML. This is so that the user only needs to download the main.css once to get all the shared info, the html file, the js, and the web components. The big reason is that instead of each html file having a separate css file which needs to be downloaded by the user, they only need 1 file (main.css) and any spare rules will be put the HTML.

The next big picture idea is how we are going to turn the items in other pages into web components. We are currently planning on having the timer page be composed of the primary parts, the timer-header, and timer-viz, and the timer-buttons. The timer page will have a stats-viz and stats-header. These implementations will have in two fields: It will keep our html simple and organized, and it will also allow for future mobility of items. Say we wanted to create a SPA version of the website in the future (not to say this is in the road map), instead of having to rebuild all the HTML we can just drag and drop the components. which will be a definite win in developer experience with little to no loss in User experience (the only real downfall of components is that it depends on JS but if you're on our website you need JS for it to work so this con doesn't affect us in the slightest).

TL;DR One, Universal CSS rules will be thrown into a main.css file and non-universal rules will be put into style elements in their respective html files. Two, web components will replace items on pages that have some sort of functionality.





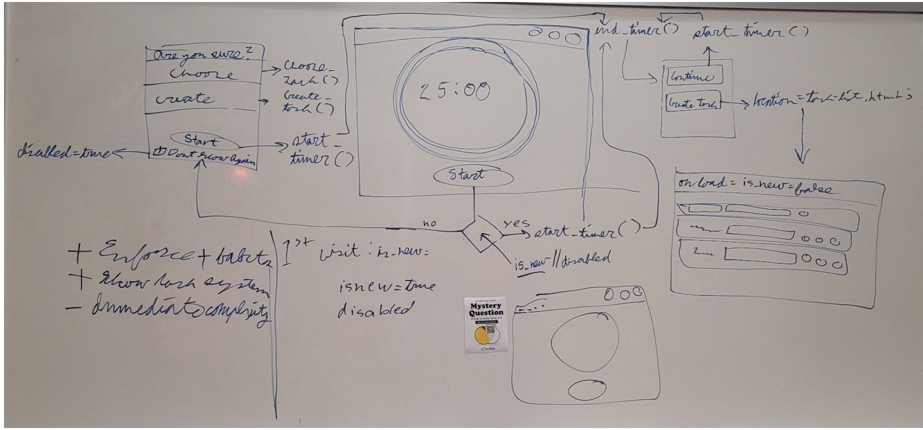
“Noob” System

We were planning on adding a new-user system. Essentially, now that you can start the timer without a task, we should be able to tell the user as a warning that they don't have a task enabled incase they forgot to choose/create one (and it also allows us to sneak our opinionated approach on the Pomodoro process onto the user). The user can either choose a created task, create a new one, or just continue onwards. The user can also have the option of disabling this option if they are not the type to make tasks. However, we are planning on having a logic system for "noobs". Essentially, when the user first enter the page, they will be marked as a noob. A noob will not receive this pop up and will be able to run the start task. In other words, a noob will not be prompted to choose/create a task. However, after every session, the noob will be asked if they wish to continue or create a task. They can either continue and stay a noob, or create a task. They will then be redirected to the task page and will no longer be marked a noob (they can also be unmarked by perusing the top bar and going to the task page on their own). Once they are no longer a noob, they will now receive the warning if they don't have a task (and they can also disable it).

The logic behind this is that there are two big things going on with the Pomodoro timer: the timer and the task management, both of which have the purpose of improving time management. The timer is meant to act as a micro time management where the user learns to stay focused on 25 minute intervals and adhere to some form of immediate regiment. The the task management is meant to act as a macro time management, where the user can begin to consider managing their time on a larger scale, that is they can plan out their day with tasks and manage not just their immediate time but their entire day. With this in mind, we must look back to our audience and see that many of the people coming in are people with poor management skills but are willing to change. If we gave both tools to the user, they might get easily burned out trying learn to manage both their immediate time and their long term time. So we first give them the ability to manage their "micro", and we don't bring up the idea of tasks to them so that we don't overload them immediately. The hope is that once they participate with the timer long enough, they will have a fair handling of the immediate time management (or at least a tolerance to it), and they would then begin to explore into other options. At this point now that they have the micro down, they can begin to learn the macro, which is the task system. But it hinges the idea that without learning how to control the micro, how can we expect them to have a handle on the macro?

There was some contention on this point because James though this would undermine our opinionated natured for the user and may frustrate users who would want tasks immediately, but we decided to go through with implementation with A/B testing as a compromise (have the noob system enabled for some and disabled for others to see which types of users have a better retention).

TL;DR : 1st time visitors won't be asked to set up tasks until they are comfortable with the timer system. After they enter the task page, they will be unmarked as "noobs" and will be prompted with warnings to have a task when starting the timer without one


Are you sure?
choose
create
Start
☐ Don't show Again
disabled=true
choose_task()
create_task()
start_timer()
25:00
Start
no
yes
start_timer()
is_new // disabled
end_timer()
start_timer()
Continue
Create task
location=task-list.html;
on load = is_new=false
+ Enforce + habits
+ Show task system
- Immediate complexity
1st visit : is_new:
isnew=true
disabled
Mystery Question