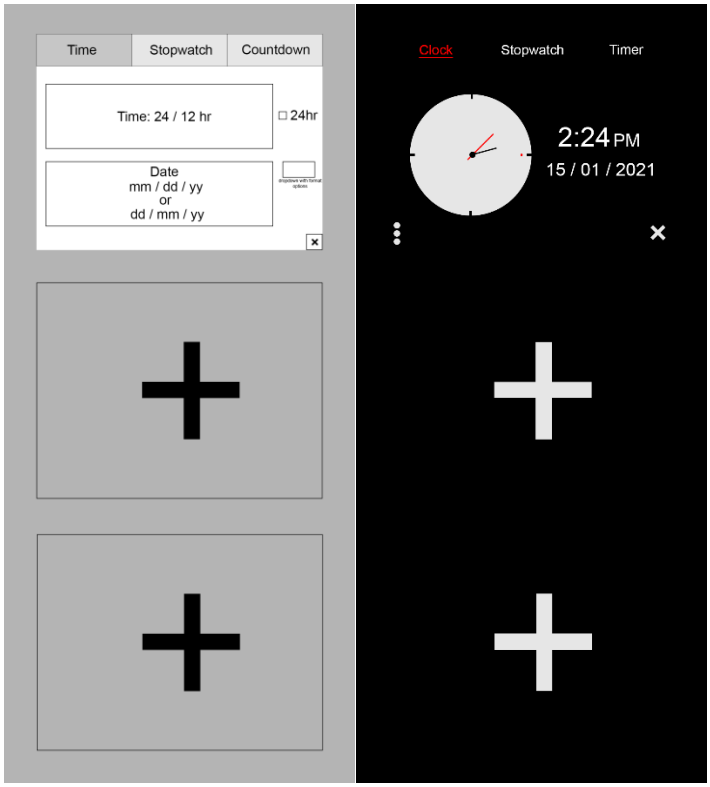
Visual Design

My first design of the UI was a pretty bog standard one – it wasn't very good in my opinion – with my second design I decided to take some inspiration from the alarm app on my phone to have a clearer idea of what would be clean/sleek. Going down this route of inspiration I was able to create a much nicer UX (in theory at least).



Comparison between original design mock-ups

Above is a comparison between my original design and the much sleeker design I ended up going with – I very much prefer the second one and am quite happy with it. This has the design of the clock, which has a settings menu for date and time formats. The pluses here are to allow the user to add new clocks (with a total of 3 on mobile and 6 on PC) and are accompanied by little x's at the bottom right of each clock.

The Pc design will be essentially be the same as mobile but with a max of 6 clocks instead of 3. I'm actually really happy with how clean this design looks, and I think I was able to add all the important elements – taking a look at what professional shipped similar products implemented. The next step from here will be to set it up in Unity and hook up all the mechanics (and honestly, I think just getting the UI to play nice will be the hardest part).

Next, I'll be working on getting each sub-section (Clock, Stopwatch and Timer) visually complete and then I'll move on to the mechanics side of things.

Section – Clock

I started working on the Clock mechanics first, this required me to have the clock hands rotate around the clock to sort out the time and date formatting. Getting the hands to rotate around an anchor (the centre of the clock) was easy enough – after two or three attempts – and I was able to get the formatting done really easily too, thankfully the .ToString() override of System.DateTime allows for easy formatting.

Once I got the hands working, I moved onto the date and time formatting, and like I mentioned that was a lot easier than I had originally expected. For each mechanic I added to this section I made a build and tested it on my phone to ensure that it worked and scaled correctly – and to make sure button sizes were big enough.

Section – Stopwatch

Moving onto the stopwatch presented a unique challenge. Originally, I thought it would be relatively easy to have a stop watch with pausing mechanics – which it was – but my original implementation was limited in this specific way.

To start with I was storing the time that the play button was pressed and just calculated the time that passed every frame in the update using Time.time, I quickly realised that this would cause some issues with pausing because time would change between pausing and starting again. Thinking back on it now I could have just set Time.timeScale to zero but decided to go in a different direction.

The implementation I went with was to basically accumulate the Time.deltaTime every frame and do a bit of "maths magic" to format it into minutes, seconds and milliseconds – what I did was checking if the value was over one minute, if it was then I would increment the minute counter and subtract 60 from the stored deltaTimes.

Once this issue was dealt with there wasn't anything else tricky about getting the stopwatch working, just more of the same; setting up the buttons and moving the circle second hand around the circle.

Section – Timer

Setting up the timer was more of the same as the other sections; setting up buttons, displaying the time as text and pausing etc. There was the addition of the user needing to input the time for the timer but that wasn't anything difficult either.

All-in-all this section proved to be little more than just the initial setup – there was also playing a sound when the timer ended but again nothing difficult, was achieved by feeding an audio clip to the PlayOneShot(…) of an audio source.

Building to Specification

- *Watch refers to the set of three functions (clock, stopwatch and timer)*

Spawnable Clocks: I took the meaning of "spawnable" a little bit liberally here where I played a bit of a bait and switch with some canvas elements, the watch prefab I use has a "main section" and an "add watch" section. So instead of instantiating watch prefabs I decided to just turn one section off and the other on.

Add/Remove Clocks: Like I mentioned I took the terms "add" and "remove" a bit liberally here too and I just switch out which object is on at any given time. This isn't to say that it would be hard to move over to an 'Instantiate based' methodology – it would be relatively easy – but is just to outline what I did.

The reason that I went with this design was because I thought it might come across 'cleaner' in some way, given that I use a vertical Layout Group to keep the watches grouped together it could be awkward to spawn and destroy them because it would reorder the layout group so that the newest ones were on the bottom. This isn't a problem but again I just felt that my method was a bit cleaner in this scenario – I could be wrong though.