Phase 2: Introducing Service Workers

This phase pertains to the following acceptance criteria: 5, 11, 12, 15

The second step is the real transition to making the app work fully offline. Once service workers become enabled, users should be able to work fully offline, even refreshing the page and still having all data + assets.

Design Brainstorming

The following is a brainstorming of ideas for what would be needed to be implemented to achieve each of the acceptance criteria above.

OK, so this Phase should, in theory, be fairly simple. Based on the research I've done (see below), getting the service worker to work should be a fairly simple act of just enabling the service worker that comes with CRA. Then we just need to make use of the built-in `onSuccess` and `onUpgrade` handlers to handle showing toasts when the service worker is ready/there are updates available respectively.

However, there is one thing that I think throws a wrinkle into this plan. I think I want the service worker (henceforth, SW) to only be enabled while the user is logged in. As such, we'll need to be able to more tightly integrate the registration/deregistration of the SW into the Redux lifecycle. This is probably good to do regardless considering we'll need access to Redux to handle the toast dispatches.

As such, we'll probably have to modify the SW registration code that comes by default with CRA. Which should be fine.

So just to re-iterate what we need to do:

- Integrate the SW registration with login.
- Integrate the SW deregistration with logout.
- Configure/Make sure the SW itself is caching things as we need.
- Setup a toast for indicating when the app is ready to work online, after login.
- Setup a toast for indicating when the app has updates.
- Add a settings menu item for manually checking for updates.
- Setup manual update checks on non-login app boots.
- Increase the validity length of the JWTs.
- Add maskable icons to the manifest to get rounded icons on Android.

Overall, shouldn't be too difficult.

Now, as far as Story breakdowns go... this could technically all be one story? Like, we *could* break it up into more than one, but 'implementing the SW' is kind of just one concrete piece of work. So yeah, it'll be just 1 story.

- "I want the app to work completely offline."

Side note:

I had the below written here before officially planning this phase...

- The requests made while booting the app (to get all of the user's data) will need to be deferred to the queue so that the app loads from `redux-persist` first.

And I don't really know *why* I put it here? I mean, I guess, yes, the above should be true, but it's... already true? Like, the store will rehydrate from storage before any of the requests can be fired, so it's implicitly true? Whatever.

Research

- Get Started | Workbox | Google Developers
  - Workbox is the toolset that CRA uses for making working with Service Workers easier. Their docs are good at explaining things.
- PWA Update Notification with Create React App
- How to Make your React App a Progressive Web App (PWA)
  - These two guides seem to be quite good.
- Manual updates in service workers: Checking for service worker updates in a single page app
- The Service Worker Lifecycle | Web Fundamentals | Google Developers

Stories

- I want the app to work completely offline.