UFC-382: Capacitor Spike

Started: August 3, 2021

I would like to spike out the mobile/desktop apps by testing Capacitor JS and seeing how well it performs.

Preliminary Thoughts

General

- Installing and getting Capacitor set up is... basically trivial. Like, it was literally just "install capacitor, pick platforms, run" and done.

  Heck, it took me longer to get the stupid Android emulator working.

  - Seems like the API 30 emulators aren't great right now, but 29 and below seemed to work.

Android

- Once I realized we needed an <30 API emulator, the app ran first time.
- However, it couldn't connect to my local Backend instance.
  - Again, seems to be an emulator thing. Seems like we need to access localhost using 10.0.2.2 instead of actual `localhost`.
    - Didn't actually get this working yet.
  - Figured it out: https://stackoverflow.com/a/43277765
    - Basically, we can access it over 10.0.2.2 or by doing a port forward using adb:
      - `adb reverse tcp:5000 tcp:5000`
  - The reason it wasn't initially working when just changing the host to 10.0.2.2 or when doing the port forward is because of CORS.
    - We need to put http://localhost in the CORS list, rather than `http://localhost:3000`, since the emulator doesn't have a port.
  - However, this does mean that we need to remember to do this port forwarding every time we deal Android emulators.
- Realized that the Service Worker fails to register (this was noticed under Android, but it's likely also true for iOS)

    - Basically, it's probably because Capacitor runs under `http://localhost`, but Service Workers require https to register.
        - For reference: https://stackoverflow.com/questions/60872617/ionic-angular-capacitor-service-worker-registration-failed
    - I guess this doesn't really matter? Our Service Worker is only needed to make the app work offline, but the app 'natively' works offline, so...
        - I think we'll be fine without it.
- Figured out why `adb` was being such a pain on my desktop: turns out I had a separate `adb` executable in `/usr/bin` that was severely outdated, while Android Studio was using the version packaged with the SDK tools.
    - Removing the one from `/usr/bin` fixed all of the problems (so far...)
- Noticed some "couldn't load encryption keys from storage" error messages after (re)deploying the app to an Android emulator.
    - This was while logged in.
    - Can't seem to reproduce reliably.

iOS

- Well, at least getting the simulators working for iOS was... much more trivial.
- App ran first time.
- Here's the problems I ran into (a lot of which were just Safari things all over again):
    - Our JavaScript function that checked for Safari wasn't covering these native web views, so we had to add the Capacitor platform check for iOS to it.
    - One of the most obvious visual bugs was that the layout wasn't accounting for the device notches properly (even though they seemed to natively in a web browser?).
        - Figured out how to fix this using the `env(safe-area-inset-SIDE)` variables, along with setting `viewport-fit=cover` on the `viewport` meta tag in `index.html`.
    - Just like with Android, a different URL had to be used for local testing.
        - However, this time, rather than the host changing, it was the *protocol* that changed.
        - For some reason, `http` gets changed to `capacitor`. Like, our URLs become `capacitor://localhost`.

        - That messed with the Backend's CORS a good bit.
    - But after adding it to the CORS list, requests seemed to work fine.
- The app seems to get into a strange state where it won't change pages.
    - Like, logging in will show the loading screen but not get anywhere.
    - And clicking on the "Use without account" link doesn't do anything.
    - Yet restarting the simulator seems to fix it.
- For some awful reason that I've yet to determine, Transactions just don't seem to work properly.
    - Like, transactions don't get stored during the initial app fetch.
    - And when creating transactions, they never actually get created on the Backend.
    - At first I thought it was because the encryption middleware wasn't working, but the accounts are showing up just fine (i.e. they are being correctly fetched and displayed, plus they can be successfully created), so I have no idea what is causing this (yet).
    - Since I don't have the Redux DevTools, debugging this is gonna be a pain.
        - I added a console logging middleware for actions, but all I see is that the action for creating the transaction is dispatched and that the attempts for the request are incremented.
        - Specifically, I see the action for encrypting encrypting data being dispatched.
    - OK, after putting a bunch of console.logs everywhere, it seems like things *are* getting hung up at the encryption middleware.
        - It hits the call for invoking the crypto worker and then just times out.
        - So something in the workers ain't working right... but only for transactions??
        - God, I figured it out... it's the same problem we ran into initially with Safari: empty strings don't decrypt properly.
        - So the real problem was that I forgot that the crypto middleware had its own implementation of the `isSafariBrowser` function.
            - Just had to update that and we're good to go.

        - Workers seem to be working just fine.
            - ... OR NOT??
            - I had removed the workers pool (instead calling the worker directly without invoking it as a web worker) and the Safari patch worked.
            - However, after reverting back to the web worker pool, it doesn't work again — with the same bug.
            - Is like something cached wrong??
            - I think my best guess is that the Capacitor module isn't being accessed properly within the web worker, but works fine outside of it.
                - Definitely seems to be this.
            - Yep, Capacitor wasn't working properly in the web worker, so it wouldn't properly detect the platform, which means the empty string errors would be silently thrown.
                - As such, we've changed to just checking that the browser isn't Chrome or Firefox. Error handling for other browsers suffers, but who cares about them.
    - For some reason, the Search input on the Transactions page has the old Safari "inset shadow" styles, but none of the other inputs do.
        - ???

Todo

- Add Electron platform.
- Add script/command for doing the port forwarding for Android emulators.
- Abstract the calls to Capacitor in case we ever need to switch it out.
- Modify the service worker sagas to not register when running under Capacitor.
    - Registering the service worker seems to just fail on iOS/Android/Electron, and since there's no real reason to get them working (native apps work offline by default, duh), we might as well just disable them so that our intention is clear and there's less cruft in the logs.
- Remove everything about signing up from iOS/Android.

    - Cause the app stores don't like it when we provide external payment options...
- Fix the search inputs on iOS that have the default "inset shadow" styling.
- Add app icon.
    - References:
        - Capacitor: Cross-platform native runtime for web apps
        - GitHub - ionic-team/cordova-res: Local Cordova icon/splash...
- Increase the lifetime of our auth JWTs to like... a year.
    - Cause who wants to get logged out of an app they installed?
    - Meh, it's at a month already; should be fine enough.

Future Todo

- Sign up for App store/Play store accounts.
- Create scripts/processes for building each/every platform.
- Figure out how we're gonna handle Electron distribution.
- Contemplate how we're gonna handle updates/auto-updates.
- Minor issue... how do we refresh data? UFC-411
    - Like if create a transaction on another device, then want to view it on the mobile app, how do we refresh the mobile app? We don't have a 'pull to refresh' or 'refresh page' button like a browser does, so...
    - I guess we need both a way to manually sync plus a background process that periodically syncs.
    - Cause otherwise, the only way would be to logout and log back in. Not the greatest UX.
- External links need to be fixed. UFC-412
    - Specifically, the link to the Marketing Changelog.
        - Since it's external, it doesn't load within the app. Instead, it shows the 404 page.
    - Also, the link to the Stripe Customer Portal.
        - Might end up having to disable this on mobile cause of app stores...

- Here's some solutions:
  - Electron: How can I force external links from browser-window to open...
  - Capacitor: How do I open all links in Capacitor's In-App Browser?

Spike Conclusion

Ended: August 9, 2021

I think that Capacitor is indeed a valid solution for providing us 'native' mobile/desktop apps.

Just from the work done in this spike, I'd say that we've done 80-90% of the work needed, all in a single week. It would have likely taken *months* to get to this point if we had gone with React Native.

We've documented what else needs to be done above. Though, most of it has to do with distribution rather than the app itself. And that's really just because the app itself pretty much *just works*.

I think this spike was *very* successful. I mean, I'd basically proven the core idea within the first couple hours, so doing the other work was just gravy to get it done now.

I don't know how mobile/desktop apps will actually fit into our schedule. I suspect that just dealing with distribution (and incoporating distribution into our development workflow) will take 2-3x as long as this spike. Which means that it *probably* doesn't factor into our current plans for another marketing push come September.

Of course, we could just push back the marketing push until we have the mobile/desktop apps ready. I'll have to think on it more. Really depends on how quickly we finish up the Ancillary Todo items that I'd planned for August.

It's looking promising!

Missing Electron Plugins

Date: February 12, 2022

Just chucking this note in here. If you run into `Cannot find module '...uFincs/services/frontend/electron/node_modules/@capacitor-community/electron/dist/runtime/electron-plugins.js', then follow` capacitor-community/electron#110 and run:

```
npx cap update @capacitor-community/electron
```

from `services/frontend`.