Nathan McMillan
A11700628

Homework 4 Write-up

https://ember-dex.firebaseapp.com/

When I first started this homework assignment, I found the firebase doc pages to be too much for me to easily find what I needed to accomplish. Because of this I explored using Vue.js in order to add all the functionality I needed. After playing with this library for a day I could not find the necessary information needed to create a grid of images. Since this was such an important feature, I ended up working on authentication first, which ended up being very straight forward using vanilla JS and firebase. By this point I began understanding where all the important information was located in the firebase docs, and I realized Vue.js was more than necessary for the functionality of this assignment and I didn't want to spend time learning a bunch of special syntax to work with it. The next thing I tackled was reading and writing from the firebase real time database. This was definitely the easiest part of the assignment and very straightforward when using the provided code from their examples. One of the most enjoyable parts of working on this assignment was converting my multi-page website into a single-page application. When I first saw on Slack that the professor recommended making it single-page, I decided it wasn't worth my time and wouldn't have too much of an effect. But after implementing signing in and out, the redirects needed to load the correct page was way too slow to be acceptable. Switching from this to a single-page format ended up being much more simple than I expected, and gave a huge performance boost switching between pages. Nonetheless, having everything in one big HTML file has made the single page load slower initially, and I will need to clean up my code to improve byte counts before it gets too much worse. Another challenge I found difficult to find a satisfactory answer to was giving the user properly informative feedback in response to their input. I had not thought about this before this assignment, so my page layout had already been in place and I was reluctant to change it at all. What I ended up doing was making the buttons themselves give the input response for a second before switching back to the regular text. This worked for a few buttons, but some responses were too large and needed to be placed in their own location near but not inside the button. Overall I was not too happy with my solution, and I think better planning ahead before designing the CSS would have been helpful. The last thing I implemented for this homework was the firebase storage feature for performing CRUD on images. I had not been planning on having any feature like this for my app, so I decided to simply give the user a profile picture that they can customize with their own image. Because firebase storage works almost the same as firebase real time database, it wasn't too hard to implement this feature. Most of my time spent on this problem was understanding the JavaScript File API for uploading files, and the firebase storage rules to make sure the user was only uploading images. This was an interesting challenge because JavaScript is in charge of asking for and getting the file the user wishes to use, but we cannot trust the user at all to upload an actual image in comparison with something else like a malicious exe. For a little while I didn't know what to do about this until I found out how many great rules firebase has for dealing with things like this. The only bad part about using firebase rules was that it simply gave a permission denied error in response, even when the user simply uploaded something like an image that was too big. The lack of specific feedback was disappointing, but because it is handled by firebase I don't see any way of fixing it.

Network results for single page

- Loading sign in page takes ~330 milliseconds, most of which is from app.html, firebase.js, and fonts

- DOM content loads in ~86 milliseconds
- NPC images load in parallel at about ~3.5 seconds total
- Fully loading takes ~7.79 seconds

As we can see from testing, having a single-page app takes considerably more time to load. However, once it completes loading the app feels very smooth and responsive. Because of this, it is definitely worth making the user wait longer for the initial page to load so that they can more easily use the actual applications functionality without any wait times. Notably, loading all the images of the characters can take quite a long time. Because this requires downloading a list from my app's firebase database, the time it takes varies greatly and there is not much I can do about it. At the moment, my image for each character is a 200x200 pixel PNG. These are definitely larger than they have to be, and I could change them to smaller JPG to improve loading times. However I prefer to have them large with good quality, so I won't change them unless it becomes too problematic with more characters to load later on. Unfortunately after trying to research a better way to load the list, I could not find how to download and display each character one by one so that the user does not have to look at the loading screen for such a long time. Currently this does not seem possible. One thing I could improve is separating the list of image URLs from the list of all the other character's information. Since this is only a small amount of text, and it has the benefit of making the view character page load faster, it doesn't seem necessary to change it.

Overall, this was by far my favorite assignment for the course. I have used firebase in the past for an Android application, but this was my first time using firebase for a website. The straightforward JavaScript code needed to make firebase work was a fun challenge without being too difficult. The biggest challenges for me were finding the important bits of information out of the firebase documentation, and I still believe they could do a better job at organizing it in order to attract more customers to their product. Working with it was definitely easier than learning a whole new syntax like Vue.js, and understanding authentication and CRUD were relatively painless. The biggest challenge I expect to face going into the next assignment is performance optimization, and making sure my app has better initial load times.