Chau Lucky Nguyen | A10742893 • Kenneth Vu | A08680828 • Ryan Weng | A10713613 // CSE 134B

HW 4: Prototyping the Core CRUD and Authentication Features

GITHUB REPO & WEBSITE URL

Github: https://github.com/ryanweng24/cse134b
Website: https://moviedex-99d1a.firebaseapp.com/

USER IMPACT

***We used a Nexus 5, with a throttling of regular 3G.**

File Name	HW4 load time	HW4 byte count	HW3 load time	HW3 byte count
index.html	1.77 s	250 KB	4.7 s	395 KB
explore.html	1.86 s	250 KB	31.56 s	2.8 MB
profile.html	5.13 s	779 KB	18.44 s	1.6 MB
movie.html	1.31 s	116 KB	775 ms	1.6 KB
watchlist.html	1.80 s	225 KB	10.6 s	837 KB
team.html	1.41 s	164 KB	6.3 s	535 KB

PERFORMANCE CHALLENGES

There are several strategies our team used to refine and optimize the load time of our web application. The most overt method that assured a decrease in load time was simply reducing the sizes of our images. Prior to assignment 4, we did not realize how significantly image sizes were slowing the app down. During the image resizing, several images were scaled down by nearly a factor of 10, and this rescaling assisted very much in reducing our load time. This strategy follows the example Professor Powell gave in class demonstrating how people can barely tell the difference between a high quality image and a passable image, but they can immediately recognize a slightly slower page.

Another improvement was moving the script tags and its contents from its original location in the head tag down to the very last lines of the body tag. Doing so increases throughput in the beginning of web page loads. This modification is made based off the specification that most browsers are capable of loading two components simultaneously, but they can only load one script component at a time. Moving the faster loading components up top will give the illusion that the page is loaded faster as much of the page's content gets loaded before the slow script components.

Other adjustments to the code include removing JavaScript and CSS from HTML and organizing them into their own respective directories and files. This improves the runtime as JavaScript and CSS files will be cached for later and repetitive reuse.

.

DEVELOPMENT CHALLENGES

We have implemented a robust foundation for our web application including bare-bone CRUD features of adding and removing movie-watching-list objects. Some challenges we faced centered around how we want to organize our images as each option has its strength and weaknesses. We have not decided whether we are using a system similar to pointers or the image organization system that Firebase gives us. Another challenge with Firebase is that the documentation advises against making nested JSON structures. When Firebase retrieves data, it retrieves the data's subtree of child nodes along with the specified data, and this can be disastrous if the desired JSON data has a large nested subtree. The lack of nested structuring is also opposed to object-oriented design and takes away from the organizational integrity of the data structure. Other development challenges mostly centered around learning how to utilize VueJS correctly and efficiently.