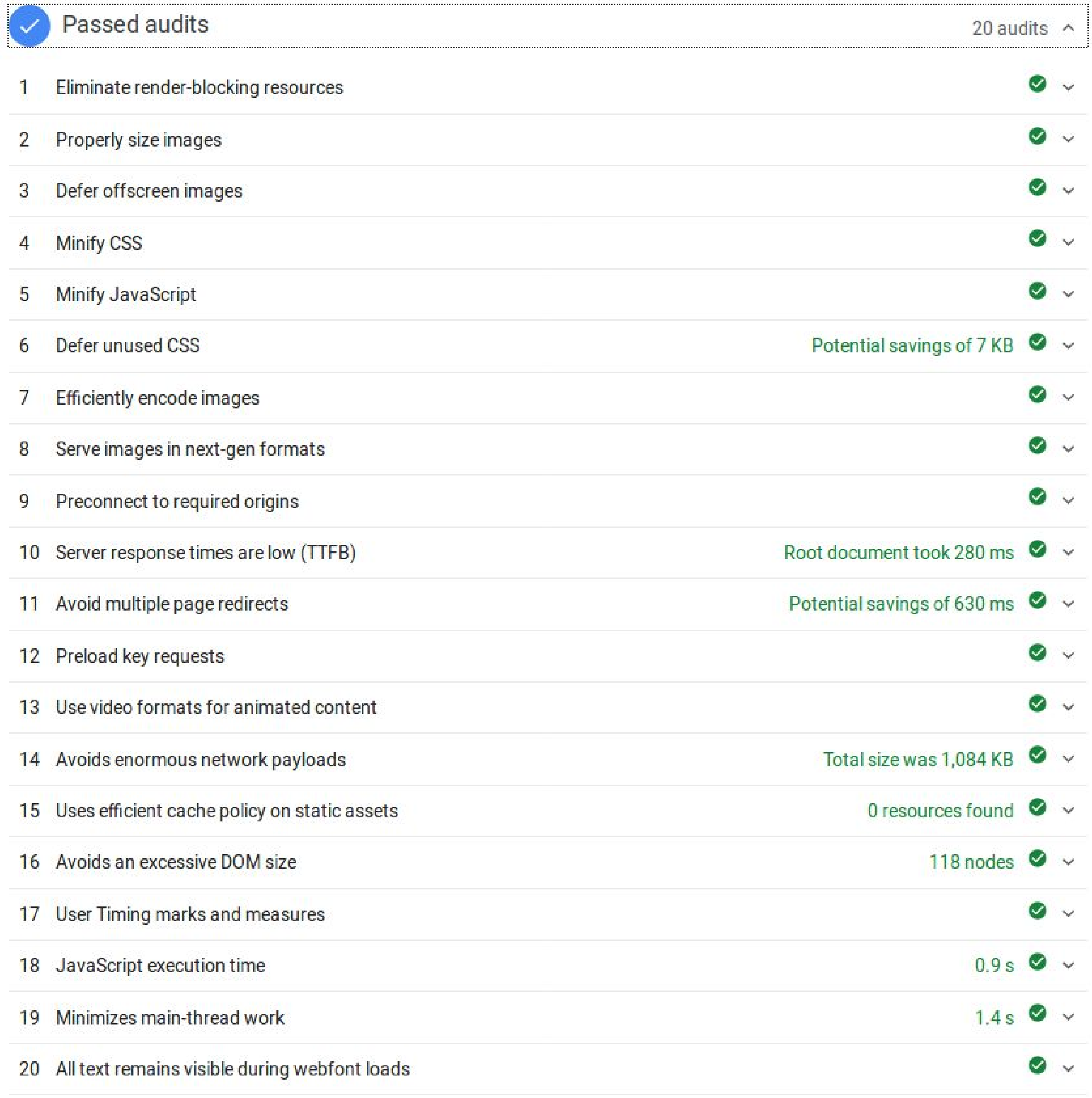
Team Golf

Dr. Nuxoll

CS 341

April 7th, 2019

Performance Analysis



Passed audits — 20 audits

1	Eliminate render-blocking resources		✓
2	Properly size images		✓
3	Defer offscreen images		✓
4	Minify CSS		✓
5	Minify JavaScript		✓
6	Defer unused CSS	Potential savings of 7 KB	✓
7	Efficiently encode images		✓
8	Serve images in next-gen formats		✓
9	Preconnect to required origins		✓
10	Server response times are low (TTFB)	Root document took 280 ms	✓
11	Avoid multiple page redirects	Potential savings of 630 ms	✓
12	Preload key requests		✓
13	Use video formats for animated content		✓
14	Avoids enormous network payloads	Total size was 1,084 KB	✓
15	Uses efficient cache policy on static assets	0 resources found	✓
16	Avoids an excessive DOM size	118 nodes	✓
17	User Timing marks and measures		✓
18	JavaScript execution time	0.9 s	✓
19	Minimizes main-thread work	1.4 s	✓
20	All text remains visible during webfont loads		✓

As demonstrated by Google PageSpeed insights, this page is optimized in a variety of ways to ensure quick load times. A lot of this is accomplished by js-uglification and minification occuring thanks to webpack, along with lazy-loading of several less critical modules. Cloudflare provides some extra help in this regard, primarily offering a content distribution network for static assets as well as interesting services like RocketLoader (which boils down to smart asynchronous packet ordering).

Concerning the assignment questions, we were able to load Google.com after sending our assets from our server-side index.js in an average of 0.625 seconds.

Regarding the client-side page load/render, we are currently having issues with CORS (likely either due to GCloud permissions or CloudFlare). This should be fixed later today with this assignment resubmitted.