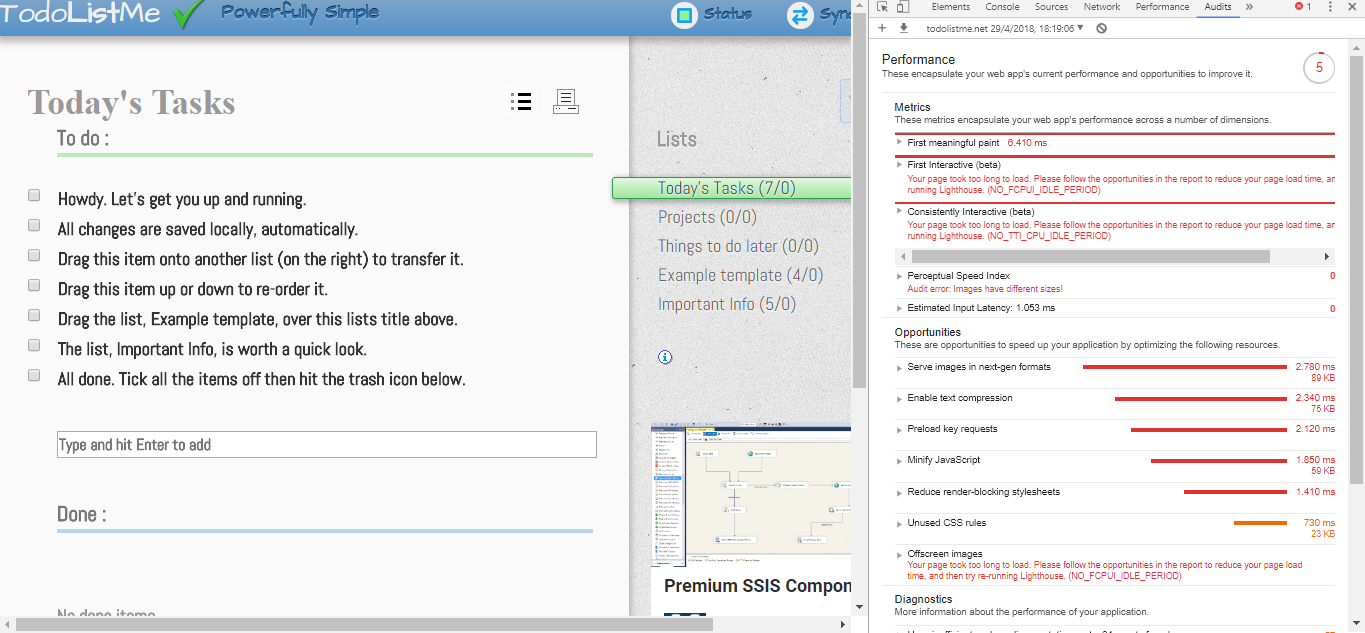
**Audit Performance Document**

1. **Todolistme.net**

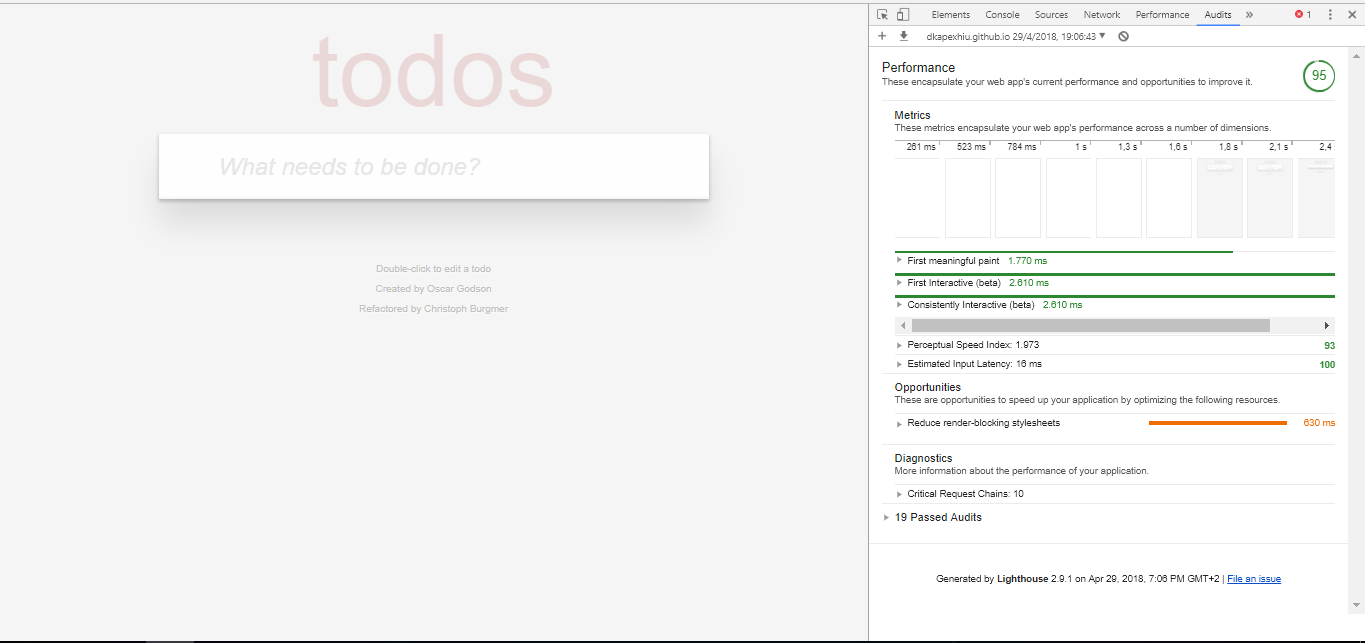
We firstly gonna analyze the performance of the website todolistme.net with Chrome Developer Tools and find issues in performance and a way how to fix them:



* The big issue for this app is the loading time: it has a First meaningful paint and First interactive time too high and there are different opportunities shown by Google to fix it:
* We need to serve the images in the next-gen format (in the webp format) for improving the performance of the app
* We need also to enable text compression on server-side (text-based responses should be served with compression (gzip, deflate or brotli) to minimize total network bytes).
* We need to preload key requests using <link rel=preload> and instead offer the most basic resources for the page to load
* Minify JavaScript (minifying javascript files can reduce payload sizes and script parse time)
* Reduce render-blocking stylesheets (external stylesheets are blocking the first paint of the page; we need to deliver critical css via style tags and defer the non-critical styles)
* Remove unused CSS rules from stylesheets to reduce unnecessary bytes consumed by network activity
* Resize the images in different sizes to serve the right image in the right time to avoid offscreen images

1. <https://dkapexhiu.github.io/to-do-app/>

Now it’s time to analyze the performance of our app.



When analyzing the performance of our app in Chrome Developer Tools, we can see that the performance score is way better!

* We can start analyzing the First meaningful paint and First interactive which are way lower than the other app and the website is loading much faster.
* The app doesn’t load images and the graphics are all loaded in css.
* The text compression is activated in the web server and that offers much more speed when loading the app
* The most basic resources for the app to load are served immediately and the bigger ones are served later (when required)
* JavaScript and CSS are minified reducing payload sizes and parse time
* Unused CSS rules are removed from stylesheets to reduce unnecessary bytes consumed by network activity
* We need to reduce render-blocking stylesheets (external stylesheets are blocking the first paint of the page; we need to deliver critical css via style tags and defer the non-critical styles and so the page can load even faster)