Performance Analysis Report

Bovenaan het rapport geeft PSI een score die de prestaties van de pagina samenvat. Deze score wordt bepaald door Lighthouse uit te voeren om lab gegevens over de pagina te verzamelen en te analyseren. Een score van 90 of hoger wordt als goed beschouwd. 50 tot 90 is een score die moet worden verbeterd en onder de 50 wordt als slecht beschouwd.

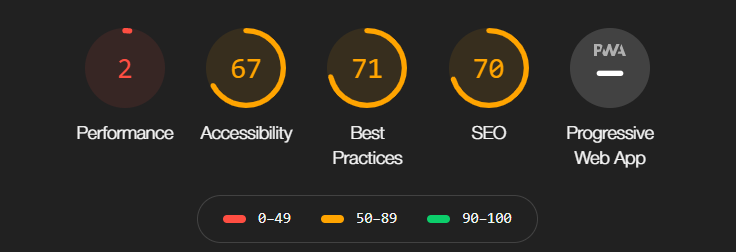
## Classifying Good, Needs Improvement, Poor

PSI classificeert ook data in 3 buckets en beschrijft ervaringen die als goed worden beschouwd, verbetering behoeven of slecht zijn. PSI stelt de volgende drempels vast voor verbetering / behoefte / verbetering / slecht, op basis van onze analyse van de CrUX-dataset:

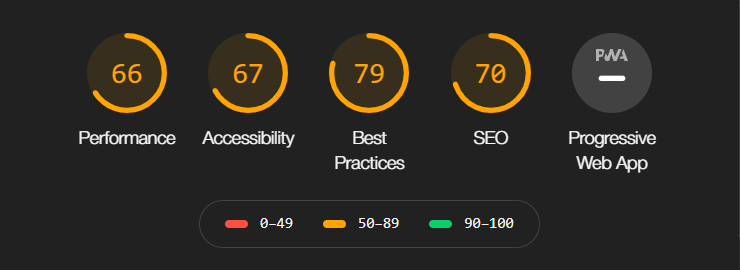
|  |  |  |  |
| --- | --- | --- | --- |
|  | Good | Needs Improvement | Poor |
| FCP | [0, 1000ms] | (1000ms, 3000ms] | over 3000ms |
| FID | [0, 100ms] | (100ms, 300ms] | over 300ms |
| LCP | [0, 2500ms] | (2500ms, 4000ms] | over 4000ms |
| CLS | [0, 0.1] | (0.1, 0.25] | over 0.25 |

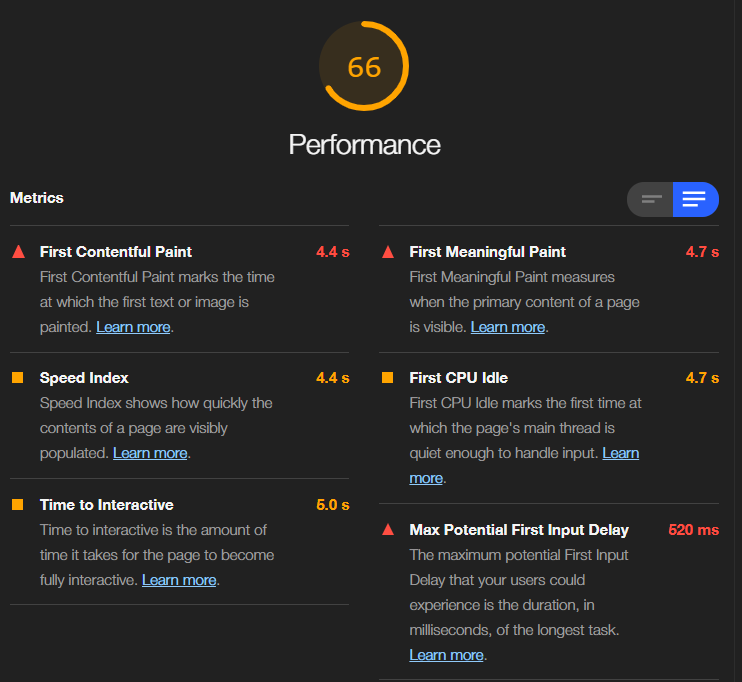
Wanneer PSI een URL krijgt, zal deze deze opzoeken in de Chrome User Experience Report (CrUX) -gegevensset. Indien beschikbaar, rapporteert PSI de metrische gegevens First Contentful Paint (FCP), First Input Delay (FID), Largest Contentful Paint (LCP) en Cumulative Layout Shift (CLS) voor de oorsprong en mogelijk de specifieke pagina-URL.

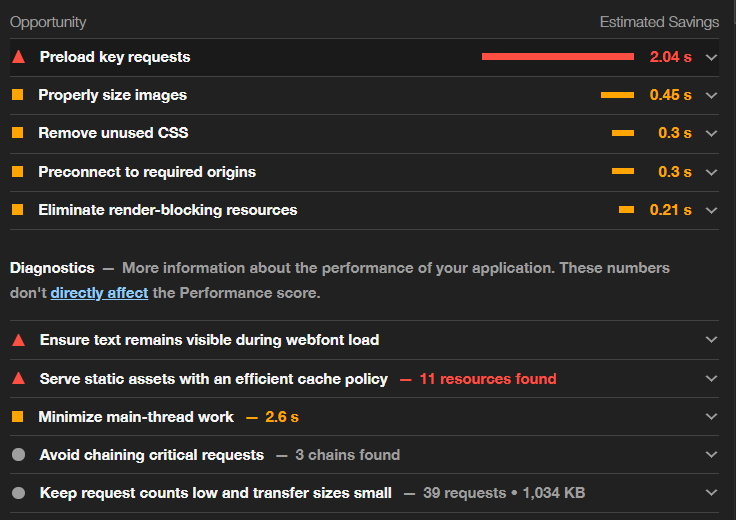
De performance test is uitgevoerd in een development omgeving hierbij worden compontenten opgestuurd met extra veel informatie voor het debuggen en makkelijk uitlezen van data binnen de site. Omdat vue nog vrij nieuw is kunnen de meeste tools nog niet goed overweg met dit framework.



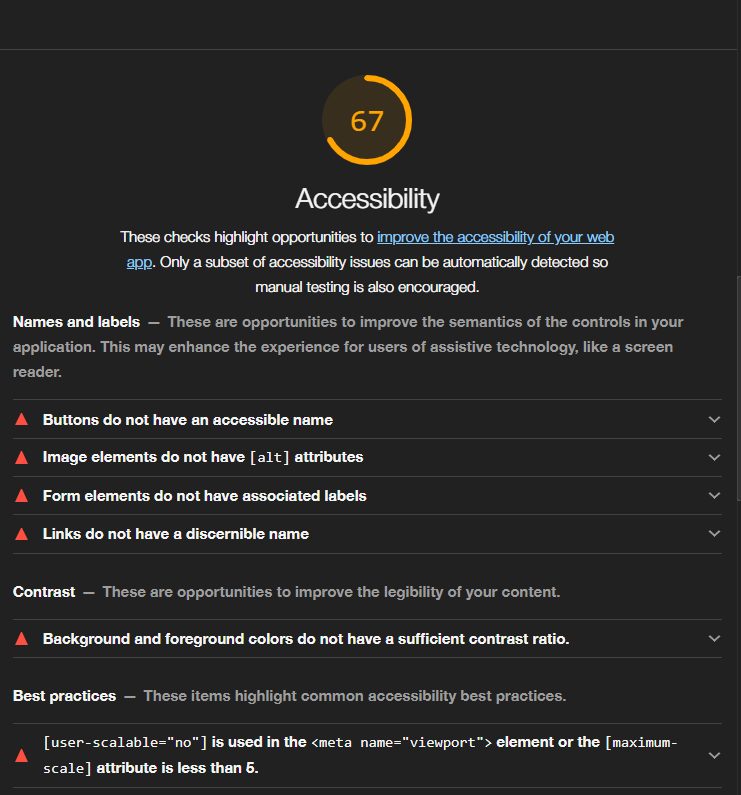
Wanneer deze applicatie in deployment gaat worden de resultaten pas echt zichtbaar, hieronder is het resultaat van het eindproduct in production. Alle scores vallen binnen de norm, wel is er mogelijkheid tot verbetering.



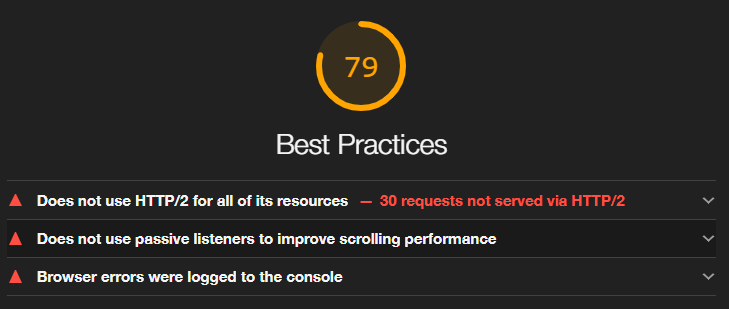
Optionele mogelijkheden en sugesties om de applicatie performance te verbeteren: 



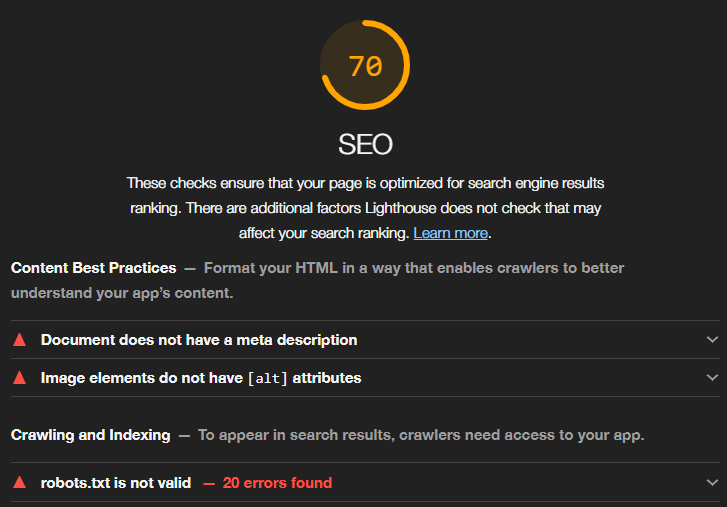
De sugesties die worden gegeven in de performance analyse zijn niet van crusiaal belang en de scores die zijn behaald en vallen gewoon binnen de norm. Wanneer er voldoende tijd was zou ik deze punten kunnen verwerken. Hierbij zou ik dan eerst naar de punten kijken die het langst nodig hebben bij het laden. Hierbij is te zien dat de preload het langst duurt. Een oplossing hierop zou het een preload rel key te gebruiken. Dit zorgt ervoor dat je browser weet dat hij bijv. De font zo snel mogelijk moet inladen zonder te wachten op overige zaken.



Ook hier is te zien dat ik voor bepaalde tags in mijn html dingen ben vergeten toe te voegen. Dit komt mede ook omdat ik een vue&css framework gebruik. Een oplossing hierop zou zijn het toevoegen van de ontbrekende elementen.



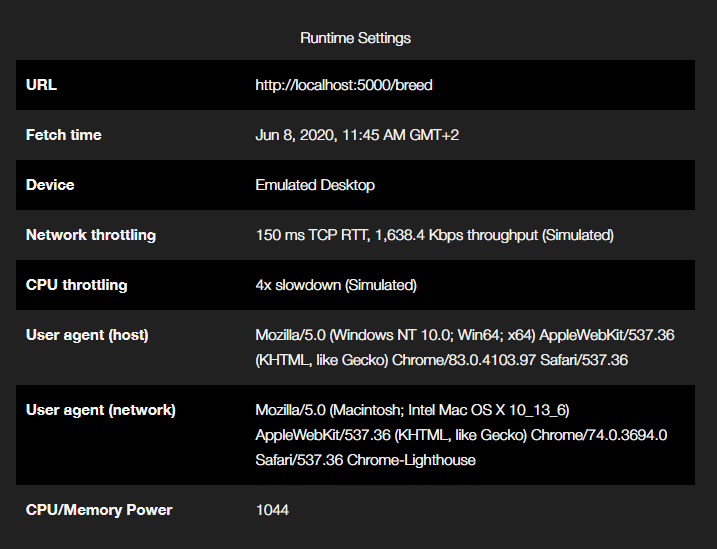
Ook hier kunnen we zijn dat er dingen te verbeteren zijn omdat ik nu nog in de development stage zit is het van belang om bepaalde resultaten in de console te laten zien.



Ook voor de search enigne zijn er optimalisaties te maken zo kan ik de ontbrekende elementen in de tags toevoegen zodat google een beter overzicht kan generen van mijn applicatie. En hier bijpassende zoektermen aan kan toevoegen.

## Simulatie runtime settings

Wanneer u deze test resulaten zelf wilt nabootsen kunt u de volgende runtime settings gebruiken.



Conclusie concreet wat zijn de dingetjes op een rij die je gaat veranderen