Laravel Horizon Insights

Wat zijn de huidige mogelijkheden die de Laravel Horizon package biedt?

Inhoudsopgave

Versiebeheer

Datum	Aanpassingen	Auteur
07-02-23	Inleiding, context en methodieken toegevoegd	Servi Huijbregts
08-02-23	Literatuurstudie en Conclusie toegevoegd	Servi Huijbregts

Inleiding & Context

Onderzoeksmethode

De onderzoeksvraag is een evaluerende vraag en vraagt om een inventarisatieonderzoek, met een inventarisatieonderzoek wordt de stand van zaken op een bepaald gebied geïnventariseerd. Je kunt hierbij denken aan het in kaart brengen van bezoekersaantallen of verkoopcijfers. In dit geval is het doel dus het ontdekken van de mogelijkheden van Laravel Horizon en dit ga ik doen door het trianguleren van de onderzoeksmethoden in de library en field onderzoeksgebieden.

Specifiek gezien ga ik gebruik maken van de methodes: Document Analysis, Literature study en Community research

De literatuurstudie gaat voornamelijk plaatsvinden in de source code van Laravel Horizon die je hier kunt vinden: https://github.com/laravel/horizon

De document analyse zal plaatsvinden op de ReadMe van de Laravel Horizon repository (https://github.com/laravel/horizon) en de Laravel documentatie die je kunt vinden op: https://laravel.com/docs/9.x

De triangulatie ziet er dus als volgt uit: Library <-> Field

Onderzoeksresultaten

Literatuurstudie

Het doel van de literatuurstudie is om te achterhalen wat de eventuele mogelijkheden zijn binnen de broncode van Laravel Horizon.

Hiervoor ga ik inventariseren welke endpoints de Laravel Horizon package zelf heeft en wat deze doen.

Method	API Endpoint	Functionaliteit
/stats	GET	Returns: Key performance statistieken voor op het dashboard
/workload	GET	Returns: Array van huidige queue workloads.
/masters	GET	Returns: Array van alle master en hun onderliggende supervisors.
/monitoring	GET	Returns: Array van tags en het aantal jobs
/monitoring	POST	Zorgt ervoor dat een tag vanaf dat moment gemonitord wordt.
/monitoring/{tag}	GET	Returns: Paginate the jobs for a given tag.
/monitoring/{tag}	DELETE	Zorgt ervoor dat een tag vanaf dat moment niet meer gemonitord wordt.
/metrics/jobs	GET	Returns: Alle gemeten jobs
/metrics/jobs/{id}	GET	Returns: Meetwaarden voor gegeven job
/metrics/queues	GET	Returns: Alle gemeten queues.
/metrics/queues/{id}	GET	Returns: Meetwaarden voor gegeven queue.
/batches	GET	Returns: Alle batches
/batches/{id}	GET	Returns: Details van een gegeven batch
/batches/retry/{id}	POST	Probeert de gegeven batch opnieuw te runnen
/jobs/pending	GET	Returns: Alle jobs met de pending status
/jobs/completed	GET	Returns: Alle jobs met de completed status
/jobs/silenced	GET	Returns: Alle jobs met de silenced status

/jobs/failed	GET	Returns: Alle jobs met de failed status
/jobs/failed/{id}	GET	Returns: De data van de gegeven gefaalde job
/jobs/retry/{id}	POST	Voert de gegeven gefaalde job opnieuw uit
/jobs/{id}	GET	Returns: de details van de gegeven recente job

Documentanalyse

Bij de documentanalyse ga ik inzoomen op de documentatie van Laravel Horizon. Wat voor giveaways staan er in de documentatie die mij eventueel vooruit zouden kunnen helpen met het schetsen van een correct project plan?

Aan het begin van de Laravel Horizon documentatie wordt aangeraden om eerst de Laravel documentatie met betrekking tot “queues” door te nemen. Dit is een heel belangrijk onderwerp voor mijn project. Laravel Queues verstrekt namelijk een queueing API die verschillende queue backends kan gebruiken zoals Redis of Amazon SQS. Laravel biedt momenteel een oplossing om de queues inzichtelijk te maken en dit is Laravel Horizon. Laravel Horizon kan echter alleen de queues van het huidige project visualiseren, en dat is niet wat ik uiteindelijk moet gaan realiseren.

Het lijkt mij handig om een onderzoek naar de mogelijkheid met de Laravel Queues uit te voeren omdat ik daarin wel degelijk de mogelijkheid zie om alle requirements van het project te realiseren.

De documentatie van Laravel Horizon laat eigenlijk vooral zien hoe je de visualisatie van de queues door middel van Laravel Horizon op moet zetten. En hoe je hierbij bijvoorbeeld “tags” of “notifications” kan instellen.

Conclusie

De conclusie is dat er veel mogelijkheden binnen Laravel Horizon zijn, maar dat Laravel Horizon enkel fungeert als visualisatie van bijvoorbeeld Redis queues. Om queues van verschillende applicaties samen te voegen in één dashboard moet er onderzocht worden of ik zelf een queue stack kan inrichten die dit kan realiseren.