

www.Info Support.com



FAQ afstuderen

Een collectie van veel gestelde vragen en keuzes

Merik Westerveld - 2737450

27 mei 2019

Definitief

**Hoofdkantoor**

Kruisboog 42

3905 TG Veenendaal

Nederland

Tel. +31(0)318 - 55 20 20

Fax +31(0)318 - 55 23 55

**Kenniscentrum**

De Smalle Zijde 39

3903 LM Veenendaal

Tel. +31(0)318 - 50 11 19

Fax +31(0)318 - 51 83 59

info.nl@Info Support.com

www.Info Support.com

K.v.K 3013 5370

BTW NL8062.30.277B01

IBAN NL92 RABO 0305 9528 89

BIC RABONL2U

IBAN NL74 INGB 0004 7385 93

BIC INGBNL2A

FAQ afstuderen

Een collectie van veel gestelde vragen en keuzes

|  |  |
| --- | --- |
| Gegevens |  |
| Titel | FAQ afstuderen |
| Project | Een collectie van veel gestelde vragen en keuzes |
| Versie | 1.0 |
| Status | Definitief |
| Datum | 27 mei 2019 |
| Bestand | FAQ veel gestelde vragen en keuzes – Merik Westerveld – Info Support.docx |
| Bedrijf | Info Support |

Historie

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Versie | Status | Datum | Auteur | Wijziging |
| 0.1 | Concept | 13 mei 2019 | Merik Westerveld | Beschrijving eerste vragen/keuzes |
| 1.0 | Definitief | 27 mei 2019 | Merik Westerveld | Definitieve versie maken |

Distributielijst

|  |  |  |  |
| --- | --- | --- | --- |
| Versie | Status | Datum | Aan |
|  |  |  |  |

**© Info Support B.V., Veenendaal 2019**

Niets uit deze uitgave mag worden verveelvoudigd en/of openbaar gemaakt door middel van druk, fotokopie, microfilm of op welke andere wijze ook, zonder voorafgaande toestemming van **Info Support B.V.**

No part of this publication may be reproduced in any form by print, photo print, microfilm or any other means without written permission by **Info Support B.V.**

Prijsopgaven en leveringen geschieden volgens de Algemene Voorwaarden van **Info Support B.V.** gedeponeerd bij de K.v.K. te Utrecht onder nr. 30135370. Een exemplaar zenden wij u op uw verzoek per omgaande kosteloos toe.

# Vragen, keuzes en verantwoording

*Waarom is er gekozen voor een monolitische architectuur in de back-end?*

Vóór het ontwikkelen van het POC (proof of concept) is er een keuze gemaakt om de back-end op te zetten met een monolitische architectuur. Een andere optie was een microservice architectuur met een orchestra/HPA (Horizontal Pod Autoscaler) en een toevoeging van de monolitische architectuur zou een load balancer zijn geweest. Waarom is er niet gekozen voor een microservice architectuur en is er geen load balancer toegevoegd aan het POC? Er is niet voor een microservice architectuur gekozen omdat er een strakke deadline was met een doorlooptijd van ‘maar’ 10 weken voor het POC waardoor er te weinig tijd was om een gehele goed werkende microservice architectuur op te zetten. Daarnaast past een microservice architectuur niet goed bij een POC omdat er te veel overhead/tijd in gaat zitten waar het uiteindelijk niet voor nodig is. Er werd namelijk verwacht dat het maximumaantal gebruikers rond de 1000 zou zijn waardoor een microservice architectuur maar daarnaast ook een load balancer niet nodig zijn.

*Waarom is er gekozen voor een clean architectuur in de back-end?*

Om de verschillende elementen in de geschreven applicatie los te kunnen koppelen of te kunnen vervangen is er gekozen voor het gebruik van clean architectuur waarbij elke laag steeds iets specifieker is van een UI/controller laag tot een domein laag. Hierdoor is de API schaalbaar, onderhoudbaar, uitwisselbaar en testbaar. Andere mogelijke alternatieven waren bijvoorbeeld een microservices architectuur, service oriented architectuur met een message bus of het gebruik van een MVC-patroon. Zoals vermeld in de vraag hierboven is er niet gekozen voor een microservices architectuur door het gebrek aan tijd en het feit dat het niet nodig is en ditzelfde geldt voor een message bus implementatie. Een MVC of soortgelijk patroon was ook een mogelijkheid, echter is er gekozen voor clean architectuur omdat het een richtlijn is voor het ontkoppelen van verschillende lagen en niet een patroon met strakke regels zoals MVC waar er per se gebruikgemaakt moet worden van een Model, View en Controller laag.

*Waarom is er gekozen voor CQS en verschillende third party libraries in de back-end?*

Er is voor het CQS-patroon gekozen om de schrijf en lees modellen te scheiden van elkaar waardoor wederom de API beter schaalbaar, onderhoudbaar en uitwisselbaar is. DaMt kan omdat er een duidelijke scheiding is waar wat geïmplementeerd moet worden en de lees en schrijf use cases een verschillende koppeling kan hebben met een CQS-patroon. Hiervoor zou ook CQRS gebruikt kunnen worden, maar hiervoor zijn er twee verschillende databases nodig, één voor alle queries en één voor alle commands.

Naast dit patroon is er ook gebruikgemaakt van Mediatr, AutoMapper en FluentValidation. Hiervoor zou ook gekozen kunnen zijn om een eigen implementatie te maken zodat de applicatie niet vastzit aan het gebruik van bepaalde frameworks, maar hiervoor is niet gekozen omdat dat voor deze libraries naar inzien niet noodzakelijk is omdat er nog steeds een loskoppeling is tussen de componenten binnen de API en het wederom terugkomt op tijd en dus prioriteit van het POC.

*Er is gekozen voor .Net voor de back-end, React-Native voor de front-end en MongoDB als Database, waarom deze keuzes?*

De argumentatie voor de back-end taal en de database kunnen in dit document gevonden worden: [keuze .Net en MongoDb](https://github.com/Merikw/afstuderen_documenten/blob/master/notes/2019-02-12_08-37_Onderzoek-beste-backend-en-database.md)

De argumentatie voor het gebruik van React-Native kan in dit document gevonden worden: [keuze React-Native](https://github.com/Merikw/afstuderen_documenten/blob/master/notes/2019-02-14_09-40_Onderzoek-cross-platform-mobile-development.md)

*Waarom is er gekozen voor een VM in combinatie met Docker in plaats van verschillende resources in Azure?*

Hierbij is er gekozen voor een VM in combinatie met Docker omdat dat goedkoper is en er meer vrijheid is voor de beheerder/ontwikkelaar aangezien alles gehost kan worden in Docker containers. Daarnaast kan alles wat geconfigureerd staat in de VM ook draaien op elk willekeurig ander apparaat dat Docker ondersteunt waardoor het uitwisselbaar is.

*Hoe is de applicatie getest?*

De applicatie is op code niveau getest met unit testen en integratie testen welke afgetrapt worden in de bouwstraat voordat er een nieuwe versie gedeployed wordt. Wanneer deze testen niet slagen, zal er geen nieuwe versie op de server gezet worden. Daarnaast is er ook op statische code kwaliteit getest en dat is gedaan met SonarQube. Na het bouwen in de bouwstraat wordt de code geanalyseerd door een SonarQube server en worden de resultaten hiervan in een portal op de SonarQube server weergegeven zodat dat verbeterd kan worden in de volgende iteratie.

Naast de applicatie te testen op code niveau is de applicatie ook getest op gebruikersniveau. Weliswaar zijn er geen UI/UX testen gedaan, maar is de app wel in productie genomen tijdens het Paaspop festival waardoor er live testen gedaan konden worden en de bezoekers daadwerkelijk gebruik konden maken van de app. Uit deze live test is gebleken dat de app werkte zoals het moest werken en zijn de problemen die tijdens deze test ondervonden zijn gelijk dat weekend nog verbeterd zodat er een nieuwe versie getest kon worden.

*Wat zijn belangrijke punten geweest tijdens en vóór het ontwikkelen van het POC?*

Vóor en tijdens het ontwikkelen van het POC zijn er een aantal elementen geweest die extra belangrijk waren en invloed hebben gehad op het maken van sommige keuzes. Dit zijn de volgende punten:

* De deadline van het project ligt op 19 april, dat betekent dat de doorlooptijd 10 weken is en in deze tijd alles af moet zijn en de app klaar moet zijn voor productie. Dit punt heeft veel invloed gehad op bepaalde keuzes, aangezien er met meer tijd waarschijnlijk andere keuzes zouden worden gemaakt. Bijna alle keuzes die zijn gemaakt tijdens het project hebben enigszins te maken gehad met de strakke deadline.
* De app is een POC wat betekend dat het geen af product is en er niet op voortgebouwd hoeft te worden, het dient enkel als (ondersteuning van) een advies en onderzoek.
* Voordat het project begon waren er nog geen eisen van de functionaliteiten beschikbaar. Hier is onderzoek naar gedaan en de uitkomst van dit onderzoek is de leidraad geweest gedurende de hele ontwikkelfase.

*Hoe is er onderzoek gedaan tijdens het project?*

Tijdens het project zijn er verschillende vormen van onderzoek gedaan:

* Interviews met experts in hun vakgebied zoals mobile development, directie van paaspop en de privacywetgeving.
* Een enquête naar wat potentiële festivalbezoekers in een festival app willen zien.
* Technisch onderzoek naar bijvoorbeeld locatiebepaling van festivalbezoekers en schaalbaarheid van een API.
* Onderzoek voor keuze naar de taal van de back-end, soort database en front-end mobile cross platform framework.
* Onderzoek naar architectuur en patronen voor de implementatie van functionaliteit in de app.
* Onderzoek naar verschillende libraries door het project heen.

*Waarom is er een bouwstraat opgezet voor de applicatie?*

Er is een bouwstraat opgezet voor zowel de back-end als de front-end. Bij de back-end wordt de nieuwe versie in een Docker container op de VM gezet en bij de front-end wordt de nieuwe versie op de PlayStore gezet. Dat is opgezet om gemakkelijk veranderingen aan te kunnen passen en online te kunnen zetten zonder dat daar handmatige acties voor uitgevoerd moeten worden. Hierdoor heb ik tijdens het festival zo snel mogelijk fouten in de code aan kunnen passen en door kunnen voeren terwijl de gebruikers de app konden blijven gebruiken en hier geen ongemakken door ondervonden werden.

*Waarom is er gebruikt gemaakt van Git voor versiebeheer?*

Tijdens het ontwikkelen van de applicatie is er gebruikgemaakt van Git voor versiebeheer. Ondanks dat er in een eensmansteam werd gewerkt was het nog steeds erg wenselijk om gebruik te maken van een versiebeheersysteem voor bij het gebruik van een bouwstraat. Daarnaast is Git een erg handig systeem om de versies van code te beheren mocht er iets fout zijn gegaan wat lastig terug te vinden is , dan kan de code teruggezet worden naar een vorige versie. Er zijn naast Git ook nog andere systemen voor versiebeheer zoals Mercurial SCM en Apache SVN, echter is hier niet naar gekeken omdat het hele bedrijf met Git werkt en er bij de Fontys 4 jaar lang met Git is gewerkt en er voor dit project geen reden was om dat te veranderen.

*Wat is de keuze om gebruik te maken van de agile methode Kanban tijdens het gehele ontwikkelproces?*

De keuze voor het gebruik van Kanban kan hier worden gevonden: [keuze voor het gebruik van Kanban als ontwikkelmethode](https://github.com/Merikw/afstuderen_documenten/blob/master/notes/2019-02-07_10-23_Argumenten-kanban-vs-scrum.md)

*Hoe zijn keuzes gemaakt tijdens het project?*

De belangrijke keuzes zijn tijdens dit project altijd samen gemaakt met de interne opdrachtgever binnen InfoSupport, Hans Geurtsen. Minder belangrijke keuzes gebaseerd op functionaliteit zijn gemaakt op basis van het enquête onderzoek en keuzes betreffende tooling en technieken zijn gemaakt op basis van klein(e) (beschreven) onderzoeken.

*Hoe is nieuwe kennis vergaard?*

Nieuwe kennis is vooral vergaard door het gebruik van internet en betrouwbare websites/documentatie van de tooling of gebruikte techniek. Daarnaast zijn er een aantal interviews gehouden met experts binnen een bepaald veld en ISKA’s (Info Support Kennisdelings Avonden) bijgewoond. Een belangrijke vorm van nieuwe kennis opdoen was het volgen van een volledige cursus op Udemy betreffende React-Native.

*Waarom is er uiteindelijk voor gekozen om de app niet uit te brengen voor IOS tijdens het festival?*

Van tevoren had de directie van Paaspop het ideale beeld geschetst dat ze het liefst willen zien dat er geen bezoekers worden uitgesloten van een toevoeging aan het festival. Dit gold dus ook voor de festival app omdat er bezoekers zijn met Android, IOS en waarschijnlijk Windows (een hele kleine groep) telefoons. Echter is ervoor gekozen om het POC van de app uiteindelijk alleen in productie te brengen voor Android omdat het erg lang kan duren voordat een IOS app in de playstore komt en hier was geen tijd voor. Daarnaast kunnen er voor IOS niet zomaar updates uitgerold worden in zo’n korte tijd waardoor het live testen met nieuwe versies op het festival voor IOS onmogelijk zou zijn.

*Hoe zijn de eisen voor het POC opgesteld?*

De eisen voor het POC zijn opgesteld naar aanleiding van het enquête onderzoek dat is uitgestuurd en ingevuld door potentiële festivalbezoekers. Het resultaat van deze enquête heeft gezorgd voor een requirements analyse waaruit de eisen met de MoSCoW methode zijn opgesteld.

*Wat is er met het POC gedaan tijdens het festival en wat is er met het resultaat gedaan?*

Tijdens het festival is het POC in productie gegaan. Dit wil zeggen dat de bezoekers de app konden downloaden en gebruiken op het festival mits ze een Android telefoon met Android 4.0.1 of hoger hebben. De app is gedurende het hele weekend getest en gebruikt door mensen binnen Info Support en andere bezoekers (700+ in totaal). Uit de eerste testen zijn wat fouten gehaald en die zijn datzelfde weekend nog verbeterd en als update doorgevoerd. Met de uitkomst van de live test die gedaan is op het festival zelf en het geschreven onderzoek is een adviesrapport opgesteld voor de directie van Paaspop met daarin advies wat ze verder met een mobiele app zouden moeten doen de komende jaren.

*Waarom wordt er gebruikgemaakt van SonarQube en ESLint?*

SonarQube en ESLint zijn ingezet tijdens het ontwikkelproces om de statische code kwaliteit te kunnen waarborgen. Dit wil zeggen dat de code overal hetzelfde is en de regels worden gehandhaafd. Dit is belangrijk om eventuele fouten te voorkomen en de onderhoudbaarheid van de code te versterken.

*Waarom zijn er geen gebruikerstesten uitgevoerd tijdens het festival?*

Van tevoren was er het idee om gebruikerstesten/UX testen te doen tijdens het festival. Echter is dit niet meer gebeurt om de reden dat het internet op het festival erg slecht was waardoor niet de hele app getest kon worden en er nog te veel tijd nodig was om (kleine) bug fixes door te voeren.