NEXT.JS SERVERLESS ARCHITECTURE RESEARCH

Sjoerd Burhenne & Joost Raemakers



# INHOUDSOPGAVE

[INLEIDING 3](#_Toc153981632)

[HOOFDVRAAG 4](#_Toc153981633)

[DOT FRAMEWORK EN DE TOEPASSING HIERVAN 5](#_Toc153981634)

[DEELVRAGEN 7](#_Toc153981635)

[HOOFDVRAAG BEANTWOORD & CONCLUSIE 10](#_Toc153981636)

# INLEIDING

Het Volunteer project wordt ontwikkeld met behulp van Next.js. Aangezien dit de eerste ervaring is met dit framework, is er besloten om onderzoek te doen naar een specifieke functionaliteit van Next.js. Het onderzoek concentreert zich op serverless architectuur, een essentieel aspect van het Next.js framework. Het is van groot belang voor het project om een grondig begrip te krijgen van hoe dit precies functioneert.  
  
Dit onderzoek wordt uitgevoerd als 1 van de 2 verplichte onderzoeken tijdens het 3e semester van de HBO ICT-opleiding (software specialisatie) met behulp van het DOT Framework.

# HOOFDVRAAG

Dit onderzoek staat in het teken van serverless architecture waarvan gebruik wordt gemaakt binnen het Next.Js framework. Daarbij wordt deze vraag als hoofdvraag gesteld:

* **Hoe maakt Next.js gebruik van serverless architecturen, en welke voordelen bieden deze benaderingen voor het ontwikkelen van schaalbare webapplicaties?**

Om deze vraag goed te kunnen beantwoorden, moeten er meer vragen gesteld worden. Deze deelvragen zorgen voor extra informatie en verduidelijking voor het beantwoorden van de hoofdvraag:

* **Hoe implementeert Next.js serverless architecturen?**  
  Wat zijn de fundamentele principes van serverless architecturen en hoe past Next.js deze toe?
* **Wat zijn de voordelen en nadelen van serverless architecturen in Next.js?**  
  Welke specifieke voordelen/nadelen biedt het gebruik van serverless architecturen voor het ontwikkelen van webapplicaties met Next.js?
* **In welke situaties is het gebruik van serverless architecturen in Next.js het meest geschikt?**  
  Zijn er bepaalde use-cases waarin serverless architecturen in Next.js bijzonder effectief zijn, en zo ja, welke zijn dat dan?

# DOT FRAMEWORK EN DE TOEPASSING HIERVAN

Als hulp bij dit onderzoek is gebruikgemaakt van het Development Oriented Triangulation of DOT framework. Het DOT framework kan helpen bij het structuur geven aan de manier van onderzoek doen en hoe je hier over communiceert. Het DOT framework bestaat uit 3 levels: What, Why en How.  
  
**What**:

Voor het "What" level van het onderzoek wordt er gedoken in drie onderling verbonden domeinen. Ten eerste is er het "toepassingsdomein", dat de specifieke context van je ICT-project verkent. Het tweede domein omvat "beschikbaar werk", met alle bestaande theorieën en modellen die relevant zijn voor je project. Ten slotte is er het "innovatiedomein", waar de unieke bijdrage vorm krijgt door grondig onderzoek.

**Why**:

Het "Why" van het onderzoek richt zich op het verduidelijken van de doelstellingen. Of er nu de pasvorm van het product binnen de toepassingscontext wordt optimaliseert of hedendaagse kwaliteitsnormen worden gewaarborgd, deze fase houdt in dat je afwegingen maakt, zoals het balanceren tussen pasvorm en expertise of overzicht en zekerheid. De onderzoeksredenering leidt vaak de keuze van methoden, waarbij de nadruk ligt op een op data gebaseerde of op inspiratie gebaseerde aanpak.

**How**:

Afbeelding met tekst, cirkel, compactdisk, Graphics

Automatisch gegenereerde beschrijvingDe "How"-fase omvat vijf onderzoeksstrategieën binnen het DOT Framework. "Library"onderzoek verkent bestaande kennis, "Field"onderzoek begrijpt de context van eindgebruikers, "Lab"onderzoek test productelementen, "Showroom"onderzoek beoordeelt bijdragen in relatie tot bestaand werk, en "Workshop"onderzoek verkent mogelijkheden via prototyping en co-creatie. Deze strategieën dragen gezamenlijk bij aan het verkrijgen van uitgebreide inzichten in de context van het project, beschikbaar werk en het innovatiedomein.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Deelvraag | Werkzaamheden | Aanpak | Methode | Resultaat |
| Hoe implementeert Next.js serverless architecturen? | * Onderzoek naar hoe Next.Js gebruikt maakt van serverless achitecturen en hoe deze zijn geimplementeerd | * Onderzoek | * Online Onderzoek | * Duidelijk begrip over serverless architecturen binnen Next.Js |
| Wat zijn de voordelen en nadelen van serverless architecturen in Next.js? | * Onderzoek wat voor en nadelen zijn van het gebruik van serverless architecturen. | * Onderzoek | * Online Onderzoek | * De voor en nadelen in kaart gebracht over serverless architecturen |
| In welke situaties is het gebruik van serverless architecturen in Next.js het meest geschikt? | * Onderzoek doen naar in welke situaties het gebruik van serverless architecturen het meest van toepassing zijn. * Eigen toepassing maken en analyseren * Testen of het daadwerkelijk werkt | * Onderzoek * Eigen toepassing maken * testen | * Online Onderzoek * Implementeren in eigen project * Testen | * Duidelijk begrip over welke situaties het meest geschikt zijn om serverless architecturen toe te passen |

# DEELVRAGEN

**Hoe implementeert Next.js serverless architecturen?**

Een serverless architectuur is een soort van cloud computing dat het mogelijk maakt voor ontwikkelaars om applicaties te maken zonder eigen server beheer. In plaats hiervan zorgt een cloud service provider, of CSP, voor al het server beheer wat er voor zorgt dat ontwikkelaars zich volledig kunnen focussen op het ontwikkelen van applicaties.

Ook maakt de serverless architectuur het mogelijk om snel een applicatie te deployen. Het is gemakkelijk om de benodigde resources voor de applicatie tijdens het ontwikkelen aan te passen naar wens.

BRON: [Getting started with the Next.js serverless framework - Bejamas](https://bejamas.io/blog/getting-started-with-the-next-js-serverless-framework/)

Sinds versie 8 van Next.Js zit er een mode in die het mogelijk serverless architectuur te gebruiken binnen het framework: Serverless Mode. Hierdoor kunnen ontwikkelaars verder met de developer-friendly en geïntegreerde omgeving van Next.Js voor het maken van applicaties en tegelijkertijd gebruik maken van de voordelen van de serverless architectuur.  
  
Serverless Mode heeft enkele ingebouwde configuraties, deze zijn echter complex. Echter zijn er ook services die het deployen in Serverless Mode makkelijker maken zoals Vercel en Netlify. Veel startups en kleinere bedrijven maken gebruik van deze services. Wat handig is aan deze services is dat de server benodigdheden gemakkelijk mee kunnen groeien met de benodigdheden van de applicatie  
  
Het Serverless Framework is een tool die het implementeren op cloudplatforms aanzienlijk vereenvoudigt door automatisering mogelijk te maken via het definiëren van de infrastructuur van je project in YAML.  
BRON: [Deploying Next.js to AWS using Serverless Next.js - LogRocket Blog](https://blog.logrocket.com/deploying-nextjs-aws-serverless-next-js/#serverless-mode)

**Wat zijn de voordelen en nadelen van serverless architecturen in Next.js?**

Bij het werken met een Next.js-functie moet er een event worden getriggerd voordat de lambda-functie kan worden uitgevoerd. Op dat moment worden verschillende middelen toegewezen aan de serverless architectuur om aan de verwachtingen van deze event te voldoen. In zo'n situatie wordt de verdeling van middelen voornamelijk bepaald door het aantal binnenkomende of gelijktijdige gebeurtenissen of verzoeken.

Dit wordt bepaald en afgehandeld door de cloudproviders die deze diensten aanbieden, waaronder AWS, GCP en Azure.  
  
Enkele voordelen van serverloze toepassingen zijn:

* Organisaties kunnen eenvoudig hun apps schalen met serverless architectuur zonder zich zorgen te maken over de infrastructuur.
* Ontwikkelaars kunnen zich concentreren op het maken van businesslogic terwijl de infrastructuur wordt afgehandeld door de serverless architectuur.
* Door Serverless Next.js te implementeren, kunnen organisaties hun infrastructuurkosten aanzienlijk verminderen door de noodzaak voor een dedicated server weg te nemen.
* In vergelijking met traditionele server-side rendering biedt Serverless Next.js verbeterde prestaties.
* Beveiligingsupdates of patches worden voor je afgehandeld.
* De cloudproviders beheren alle andere technologische details, waardoor ingenieurs zich kunnen concentreren op functie-implementatie en onderhoud van de kernapplicaties.
* Ontwikkelaars kunnen zich richten op het leveren van bedrijfswaarde dankzij de serverless architectuur omdat routinematige beheertaken niet zelf uitgevoerd hoeven te worden.

Natuurlijk zijn er naast voordelen ook nadelen, hieronder zijn er een paar vastgesteld:

* Bij het gebruik van serverloze architectuur zoals Serverless Next.js moet rekening worden gehouden met het idee van vendor lock-in.
* Het is uitdagend om een applicatie van de ene naar de andere cloudprovider te verplaatsen omdat het Next.js-framework afhankelijk is van de cloudprovider.
* Voor langlopende applicaties kan het bijvoorbeeld niet de ideale optie zijn omdat het duurder kan zijn dan het gebruik van een virtuele machine of een container die al is geconfigureerd.
* Gebrek aan controle is een ander nadeel van serverloos. Het Next.js-framework biedt geen totale controle over de omgeving en infrastructuur op zijn fundament.

BRON: [Getting started with the Next.js serverless framework - Bejamas](https://bejamas.io/blog/getting-started-with-the-next-js-serverless-framework/)

In welke situaties is het gebruik van serverless architecturen in Next.js het meest geschikt?

Ontwikkelaars die de tijd die het kost om een applicatie op de markt te brengen willen verkorten, lichte maar flexibele applicaties willen maken die uitgebreid kunnen worden, kunnen erg profiteren van serverless architecturen.

Serverloze architecturen zullen de kosten verminderen voor applicaties die onregelmatig worden gebruikt, met piekperiodes afgewisseld met tijden van weinig tot geen verkeer. Voor deze applicaties kan het kopen van een server of een blok servers die constant draaien en altijd beschikbaar zijn, zelfs wanneer ze niet worden gebruikt, verspilling van middelen zijn. Een serverless opstelling reageert direct wanneer dat nodig is en brengt geen kosten met zich mee wanneer deze in niet wordt gebruikt.  
  
Ook ontwikkelaars die bepaalde of alle functies van hun applicatie dichter bij eindgebruikers willen brengen voor verminderde latentie, zullen op zijn minst gedeeltelijk een serverloze architectuur nodig hebben, aangezien dit vereist dat sommige processen uit de oorspronkelijke server worden verwijderd.  
BRON: [Why use serverless computing? | Pros and cons of serverless | Cloudflare](https://www.cloudflare.com/en-gb/learning/serverless/why-use-serverless/)  
  
Om een beter beeld te krijgen over hoe je een Next.Js serverless applicatie nou opzet, is het Volunteer project voor dit onderzoek gedeployed op Vercel, een CSP. Hieronder de stappen die zijn genomen:  
  
Zoals eerder al aangegeven in dit onderzoek, zorgt de serverless architectuur onder andere voor het simpeler maken van het deployen van een schaalbare applicatie. Voor dit onderzoek is Volunteer gedeployd op de CSP Vercel. Dit is heel eenvoudig.

* Maak een account aan op Vercel
* Koppel een GitHub account
* Kies de repository met de NextJs applicatie die gedeployed moet worden
* Geef de benodigde environment variabelen aan Vervel
* Deploy

Afbeelding met tekst, schermopname, software, Multimediasoftware

Automatisch gegenereerde beschrijving

Wanneer dit is gedaan is de applicatie, in dit geval het Volunteer project, met behulp van serverless achritecturen op Vercel gedeployed.

Dit kan getest/bekeken worden via deze link: [Volunteer (volunteer-frontend-lemon.vercel.app)](https://volunteer-frontend-lemon.vercel.app/)

# HOOFDVRAAG BEANTWOORD & CONCLUSIE

Om de hoofdvraag “Hoe maakt Next.js gebruik van serverless architecturen, en welke voordelen bieden deze benaderingen voor het ontwikkelen van schaalbare webapplicaties?” te beantwoorden zijn eerst de hiervoor genoemde deelvragen beantwoord.  
  
Na het onderzoek naar deze vragen is er een duidelijk beeld van serverless architecturen en hoe deze werken.  
  
Next.Js maakt gebruik van een mode die in de 8e versie van het framework is toegevoegd, de Serverless Mode. Serverless architecturen zijn handig voor ontwikkelaars zodat zij zich volledig kunnen focussen op het ontwikkelen van goede applicaties en het server gedeelte naast zich neer kunnen leggen. Naast niet hoeven na te denken over server beheer, zijn infrastructuur kosten aanzienlijk minder doordat alleen de resources gebruikt worden die op een gegeven moment daadwerkelijk nodig zijn, biedt Serverless Next.js verbeterde prestaties en worden beveiligingupdates of patches automatisch afgehandeld.   
  
Al deze punten, maar ook punten die eerder genoemd zijn in dit onderzoek beantwoorden de vraag hoe Next.Js gebruik maakt van serverless architecturen, waarom dit handig kan zijn en welke voordelen het heeft voor het ontwikkelen van schaalbare webapplicaties.