# Case Dockbite

Daniël Kappelle
daniel.kappelle@dockbite.nl

2 april 2019

## 1 Doel van de case

Het doel van deze case is te kijken of gewenste competenties aanwezig zijn. Het is dan ook niet erg als kennis van bepaalde zaken ontbreekt, maar juist belangrijk hoe deze nodige kennis dan wordt aangevuld en wordt opgezocht.

Voor deze case is het de bedoeling om een back-end te maken voor een heel simpel blogsysteem. Een beschrijving van het systeem en een aantal eisen staan in de volgende paragraaf. Het systeem kan op allerlei verschillende manieren geïmplenteerd worden. Het is dus juist belangrijk dat de keuzes onderbouwd kunnen worden.

Het back-end dient als webserver te draaien op Node.js, eventueel door gebruik te maken van bestaande frameworks of packages. Een front-end hoeft niet geïmplementeerd te worden (een simpele mag natuurlijk wel), maar er kan gebruik gemaakt worden van een tool als *Postman*[1]. Een paar voorbeeld api endpoints staan ook in de volgende paragraaf. De data (artikelen en reacties) dient opgeslagen te worden door middel van bijvoorbeeld (waarschijnlijk) een database.

## 2 Case: mini blogsysteem

Het blogsysteem bestaat uit artikelen en reacties. Een artikel bevat een titel, een tekst en een categorie. Verder moet het mogelijk zijn het artikel te 'liken'. Het aantal likes moet bijgehouden worden. Er moeten ook reacties geplaatst kunnen worden bij een artikel. Deze reacties bestaan ten minste uit een naam en de reactie.

### 2.1 Eisen

#### 2.1.1 Artikel

Het moet mogelijk zijn ...

- een artikel aan te maken
- een artikel op te vragen (inclusief aantal likes en reacties)
- een artikel te updaten
- een lijst van alle artikelen te krijgen
- alle artikelen binnen een bepaalde categorie op te vragen
- een artikel te 'liken'
- een reactie te plaatsen voor een specifiek artikel

[1] Kan hier gedownload worden https://www.getpostman.com/downloads/

### 2.1.2 Reactie

Het moet mogelijk zijn ...

- alle reacties op te vragen voor een specifiek artikel
- een reactie te plaatsen bij een specifiek artikel
- een reactie te verwijderen

### 2.1.3 API endpoints

De API endpoints dienen volgens het REST principe te werken [1], een paar voorbeelden:

**Artikel met id 1234 ophalen** GET /artikel/1234

**Artikel plaatsen** POST /artikel
In de body van de post een JSON object in de vorm:

```
{
        "title": "Titel van het artikel",
        "body": "de tekst",
        ...
}
```

De precieze invulling van de endpoints, zoals url's en attribuutnamen mogen zelf ingevuld worden.

### 2.1.4 Overig

Het systeem dient lokaal als webserver te draaien op Node.js [2]. Het moet dus mogelijk zijn calls te maken (d.m.v. bijvoorbeeld Postman) naar `http://localhost:3000` of `http://127.0.0.1:3000` (niet per se poort 3000, dat is slechts een voorbeeld).

De artikelen en reacties moeten opgeslagen worden, dus er moet iets van een database geïmplenteerd worden om de data op te slaan. De keuze hierin is vrij.

# Referenties


[1] Representational state transfer, `https://en.wikipedia.org/wiki/Representational_state_transfer#Architectural_constraints`

[2] Node.js, `https://nodejs.org/en/`