Opdracht 1 Manual

Kasper Engelen

August 11, 2019

Contents

1 Algemeen gebruik

De server van de applicatie kan gestart worden door `app.py` te runnen met python 3.6. De enige gebruikte packages zijn `Flask` 1.1.1 en `requests` 2.22.0. Er is een `requirements.txt` meegeleverd waarmee deze geïnstalleerd kunnen worden d.m.v. `pip3 install -r requirements.txt`.

De applicatie kan gebruikt worden door naar `localhost:5000` te surfen. Bij het betreden van de webpagina krijgt men een lege wereldkaart te zien, deze is gecentreerd rond Vlaanderen.

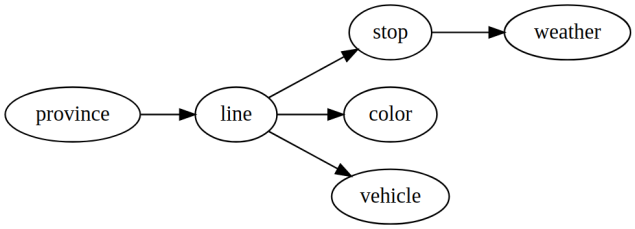
Om lijnen, haltes en voertuigen toe te voegen aan deze kaart moet men op *Add Route* klikken. Men krijgt vervolgens een dialoogvenster te zien met twee inputs: één voor de provincie te selecteren en een andere om de lijn te kiezen. Als men een keuze gemaakt heeft moet er op "Add" geklikt worden. Na enige tijd wachten verschijnt er een traject op de kaart.

Door in te zoemen op de de route worden de haltes zichtbaar in de vorm van blauwe stippen, welke verbonden zijn door gekleurde lijnen. Door op de lijnen de klikken wordt er informatie over de lijn beschikbaar, namelijk het lijnnummer en de bestemming. Daarnaast kan men ook op de haltes klikken, waardoor er na enige tijd informatie over de halte en de weersituatie tevoorschijn komt.

Op de lijn zijn er ook grotere markers zichtbaar, deze stellen de verschillende voertuigen voor die op dat moment actief zijn op de lijn. Door er op te klikken kan men het ritnummer zien.

2 API

Bij het opstellen van de API heb ik geprobeerd om de API zo modulair mogelijk te maken. Dit heeft als gevolg dat er intern soms redundante calls gemaakt worden wanneer bepaalde data op meerdere plaatsen nodig is. Ik heb ook geprobeerd om het hierarchisch te maken. Hierdoor kan men navigeren doorheen de API door steeds gebruik te maken van de vorige call. De onderstaande figuur geeft deze hierarchie aan.



De API voorziet de volgende calls:

Provincies `<domeinnaam>/api/provinces/`

Lijnen `<domeinnaam>/api/provinces/<provincie nr>/lines/`

Haltes `<domeinnaam>/api/provinces/<provincie nr>/lines/<lijn nr>/stops/`

Voertuigen `<domeinnaam>/api/provinces/<provincie nr>/lines/<lijn nr>/vehicles/`

Kleuren `<domeinnaam>/api/provinces/<provincie nr>/lines/<lijn nr>/color/`

Weer `<domeinnaam>/api/provinces/<provincie nr>/lines/<lijn nr>/stops/<halte nr>/weather/`

Elk van deze calls is toegankelijk d.m.v. een GET request. De responses staan hieronder uitgelicht. Een parameter die alle responses gemeenschappelijk hebben is `responseCode`. Deze bevat de HTTP statuscode (404, 500, 200, etc.) waardoor er gemakkelijk een fout gedetecteerd kan worden nadat de respons is ingelezen in JSON formaat.

Provincies

```
{
    "provinces": [
        {
            "id": 1,
            "name": "Antwerpen"
        },
        {
            "id": 4,
            "name": "Limburg"
        }
    ],
```

```
    "responseCode": 200
}
```

Lijnen

```
{
    "lines": [
        {
            "id": 15,
            "name": "P+R Boechout - P+R Linkeroever",
        }
        {
            "id": 32,
            "name": "Edegem - Rooseveltplaats",
        }
    ],
    "responseCode": 200
}
```

Haltes

```
{
    "dirs": [
        {
            "name": "Centraal Station - Berchem
               Station - Edegem",
            "type": "HEEN",
            "stops": [
                {
                    "id":104811,
                    "name":"Quellin perron 1",
                    "city":"Antwerpen",

                    "coord": {
                        "lat":51.2176240266208,
                        "long":4.416617990452736
                    }
                }
            ]
        },
        {
            "name": "Edegem - Berchem Station -
               Centraal Station",
            "type": "TERUG",
            "stops": [
                {
                    "id":105170,
```

```
                    "name":"Sint-Goriksplein",
                    "city":"Edegem",

                    "coord":{
                        "lat":51.14889388944266,
                        "long":4.454832695280607
                    }
                }
            ]
        }
    ],

    "responseCode": 200
}
```


Voertuigen


```
{
    "dirs": [
        {
            "name": "Centraal Station - Berchem
               Station - Edegem",
            "type": "HEEN",
            "vehicles": [
                {
                    "seqNr": 85,

                    "coord":{
                        "lat": 51.14943627902871,
                        "long": 4.452467397398141 5
                    }
                }
            ]
        },
        {
            "name": "Edegem - Berchem Station -
               Centraal Station",
            "type": "TERUG",
            "vehicles": [
                {
                    "seqNr": 84,

                    "coord":{
                        "lat": 51.21741921366849,
                        "long": 4.417401485404908
                    }

                }
            ]
```

```
        }
    ],

    "responseCode": 200
}
```

Kleuren

```
{
    "color": "#FFCC11",
    "responseCode": 200
}
```

Weer

```
{
    "clouds":20.0,
    "humidity":63.0,
    "temp":17.82,
    "windSpeed":12.96,
    "OWM_icon_url":"http://openweathermap.org/img/wn/02d@2x.png",

    "responseCode": 200
}
```

De parameter `OWM_icon_url` stelt hier de URL van een illustratie voor. Deze illustratie beeldt de locale weersituatie uit. `clouds` en `humidity` zijn in procenten, `temp` is de temperatuur in graden Celsius, en `windSpeed` is de windsnelheid in kilometer per uur.

Error

```
{
    "causeErrorMessage": "N/A",
    "causeErrorStatus": -1,
    "errorMessage": "Internal error.",
    "responseCode": 500,
}
```

Indien er een "externe" fout aan de oorzaak ligt, bevat de parameter `causeErrorMessage` de oorspronkelijke foutmelding en `causeErrorStatus` de oorspronkelijke foutcode (500,404, etc). Indien deze achterliggende oorzaak er niet is, of als deze niet relevant is bevatten deze parameters respectievelijk `N/A` en -1. De parameter `errorMessage` bevat een zelfgemaakte foutmelding en `responseCode` bevat de uiteindelijke foutcode (normaal gezien 500).

3 Gebruikte tools

Om de kaart weer te geven heb ik gebruikgemaakt van de leaflet Javascript library. De data die de kaart weergeeft is afkomstig van OSM. Daarnaast heb ik voor de vormgeving gebruikgemaakt van de bootstrap library. Tenslotte gebruik ik de `requests` Python package om HTTP requests te versturen.

4 Veelvoorkomende problemen

Tijdens het gebruik van de applicatie kunnen enkele problemen optreden:

- **DeLijn API is traag** De API van DeLijn is vaak traag, en soms kan een request wel rond de 6 seconden duren. Om te voorkomen dat de applicatie blijft vasthangen wordt de API asynchroon aangesproken zodat de applicatie responsief blijft. Daarnaast is er ook een loading screen aanwezig. Om aan te tonen dat de traagheid van de applicatie te wijten is aan de externe APIs, wordt er voor en na elke request een print gedaan naar de console (zowel server-side als client-side).

- **DeLijn API is soms onbetrouwbaar** Soms geeft de API van DeLijn foute resultaten terug zoals het ontbreken van dienstregelingen, het ontbreken van haltes in lijnen, het niet overeenkomen van dienstregelingen met lijnen, etc. Om dit op te vangen voorziet de API foutcodes die worden opgevangen door de client. Hierdoor wordt de gebruiker op de hoogte gesteld van fouten. De enige mogelijkheid om dit op te lossen, is door het op een andere moment opnieuw te proberen.

- **Slecht geformatteerde informatie** De API van DeLijn geeft de data soms weer in een raar formaat. Een voorbeeld is dat sommige lijnen meerdere trajecten hebben, maar deze worden toch samengevoegd en zo weergegeven als één traject (bijv Lijn 45 in Limburg). Daarnaast staan de haltes van een lijn niet in volgorde. Binnen de applicatie wordt dit opgelost door deze haltes te sorteren a.d.h.v. een dienstregeling. Het nadeel is dat wanneer er geen dienstregeling is, er ook geen haltes kunnen worden weergegeven op de kaart.

- **Infrequente lijnen** Sommige lijnen worden infrequent bereden (bijv een bus die slechts om het uur komt) waardoor ze soms weergegeven worden zonder voertuigen. Enkele minuten later opnieuw proberen verhelpt dit probleem.

- **Geen routing API** Het incorporeren van een routing API is niet gelukt. Het is namelijk zo dat HERE.com rare resultaten gaf, Google's API vereiste billing information en OSM ondersteunt maximaal 25 waypoints. Daarnaast waren er ook meerdere problemen met het ondersteunen van tramtunnels waardoor deze functionaliteit niet werkte.

Om de applicatie te testen volstaat het om enkele lijnen binnen Antwerpen Stad te proberen zoals Lijn 2, 6, 9, 11, 15, 32, etc. Deze zijn gebruikt om te testen doorheen het ontwikkelen van de applicatie en werken doorgaans zonder problemen.