Opdracht 2 Manual

Kasper Engelen

August 5, 2019

Contents

1 Algemeen gebruik

De docker applicatie kan gestart worden door `start.sh` uit te voeren. Dit script start de docker containers en reset the SQL-databases van de verschillende services. De docker applicatie kan gestopt worden door `end.sh` uit te voeren. De applicatie is toegankelijk op het adres `localhost` poort `5000`.

Opmerking: Bij het starten van de docker applicatie komt het heel zelden voor dat er zich een fout voordoet, de applicatie nogmaals proberen te starten verhelpt dit probleem.

Men kan de applicatie bezoeken door naar `localhost:5000` te surfen. Men wordt dan gepresenteerd met een index pagina. In de navigation bar vind men drie links:

Users Overzicht van alle gebruikers en tegelijk de toegang tot het aanmaken en bekijken van gebruikers,

Vehicles Overzicht van alle voertuigen en tegelijk de toegang tot het aanmaken en bekijken van voertuigen.

Stops Overzicht van haltes op basis van een te selecteren filter en tegelijk toegang tot het bekijken van haltes.

Bij het openen van de *Users* pagina ziet men een overzicht van alle geregistreerde users. Er is ook een *Add User* knop die de mogelijkheid biedt om een gebruiker aan te maken. Elke gebruiker die aanwezig is in de lijst bevat een *details* knop die verwijst naar een pagina met informatie over de gebruiker.

Indien men klikt op de details knop ziet men een pagina met daarop alle voertuigen die toegevoegd zijn door de gebruiker, alsook een lijst van alle gerecenseerde voertuigen en haltes.

Wanneer men de *Vehicles* pagina opent, krijgt men een overzicht van alle geregistreerde voertuigen, alsook de eigenaar van het voertuig en de gemiddelde score. Elk voertuig is voorzien van een details knop die leidt naar extra informatie over het voertuig. Op deze pagina is er ook een *Add Vehicle* knop die geregistreerde gebruikers toelaat om een voertuig toe te voegen.

Op de details pagina vindt men de gemiddelde score, alsook elke recensie die van het voertuig gemaakt is.

Om een voertuig te beoordelen klikt men op de *Add Review* knop op de voertuig informatie pagina. Men moet dan de login informatie invullen en een beoordeling uitkiezen.

Elk voertuig op de Vehicles pagina is voorzien van een delete knop. Deze laat geregistreerde gebruikers toe om een voertuig te verwijderen indien zijzelf eigenaar zijn van het te verwijderen voertuig en er nog geen andere gebruikers zijn die het voertuig beoordeeld hebben.

Om een overzicht van alle haltes te bekomen kan men de *Stops* pagina openen. Men krijgt dan drie knoppen te zien waarmee men kan filteren op provincie, lijn en gemeente. Er is echter geen optie om alle haltes in Vlaanderen in één keer te zien, dit komt omdat zo een request te lang duurde. Ter vervanging is het mogelijk om alle haltes binnen één provincie te zien.

Indien men een filter heeft gekozen komt er een lijst van haltes tevoorschijn, elk voorzien van een details knop. De details pagina toont de gemiddelde score en een lijst van recensies die over de halte gemaakt zijn. Er is ook een *Add Review* knop waarmee geregistreerde gebruikers een halte kunnen beoordelen

2 API

De API's zijn zowel intern als extern toegankelijk. De reden waarom deze API's ook extern beschikbaar zijn, is omdat er op die manier rechtstreeks met de service gecommuniceerd kan worden via javascript. Een alternatief zou zijn om de javascript client requests te laten sturen naar de webapp service waarna de requests geforwarded worden naar de interne services, maar dit vond ik te omslachtig. Daarnaast is het door de externe API ook mogelijk om de applicatie te porten naar andere platformen (bijv een android applicatie, etc). Tenslotte vermindert dit de load op de webapp service, het is namelijk mogelijk om naar

de users pagina te surfen (wat één request naar de webapp service vereist) en vervolgens meerdere gebruikers toe te voegen (wat dan requests naar de users service stuurt zonder de webapp service te belasten.)

Opmerking, alle API adressen moeten op het einde een `"/"` hebben. Indien niet, zal de nginx reverse proxy deze adressen niet correct rerouten! Het oplossen van dit probleem vereist geavanceerdere nginx opties die ik niet heb aangeraakt.

2.1 Users service API

Hello World

intern: `http://users:5000/hello_world/`

extern: `http://localhost:5000/api/users/hello_world/`

Deze API call kan gebruikt worden om te checken of de service actief is en is toegankelijk met een GET request.

```
{
    'responseCode': 200,
    'message': 'Hello world!'
}
```

Create user

intern: `http://users:5000/create/`

extern: `http://localhost:5000/api/users/create/`

Deze API call kan gebruikt worden om een gebruiker aan te maken. Het vereist onderstaande parameters in de form body en is toegankelijk met een POST request.

`email` Het emailadres van de nieuwe gebruiker. Deze moet uniek zijn.

`username` De gebruikersnaam van de nieuwe gebruiker. Deze moet uniek zijn.

`password` Het wachtwoord van de nieuwe gebruiker.

Indien alles in orde is, wordt onderstaande response gereturned.

```
{
    'responseCode': 200
}
```

Authenticate user

intern: `http://users:5000/auth/`

extern: `http://localhost:5000/api/users/auth/`

Deze API call kan gebruikt worden om een gebruikersnaam en wachtwoord te verifieren. Het vereist twee parameters `username` en `password` in de HTTP header en is toegankelijk met een GET request.

De response is een JSON object met onderstaande parameters:

`userId` Het gebruikersnummer indien die authenticatie gelukt is, anders -1.

`result` `True` indien de authenticatie gelukt is, anders `False`.

`reason` De reden waarom de authenticatie niet gelukt is, anders is deze `N/A`.

`responseCode` Altijd gelijk aan 200.

Retrieve list of users

intern: `http://users:5000/list/`

extern: `http://localhost:5000/api/users/list/`

Deze API call kan gebruikt worden om een lijst van gebruikers te bekomen. De call vereist geen parameters en is toegankelijk met een GET request. Er wordt onderstaande reponse gereturned.

```
{
    "users": [
        {
            "id": 1,
            "username": peter.selie,
            "email": peter.selie@hotmail.com
        },
        {
            "id": 53,
            "username": een.gebruiker,
            "email": gebruiker.email@mail.abc
        },
    ],
    "responseCode": 200
}
```

Retrieve user info

intern: `http://users:5000/get/<user_id>/`

extern: `http://localhost:5000/api/users/get/<user_id>/`

Deze API call laat toe om informatie over een gebruiker op te halen. Onderstaande response wordt teruggegeven.

```
{
    "id": 1,
    "username": peter.selie,
    "email": peter.selie@hotmail.com,
    "responseCode": 200
}
```

2.2 Vehicles service API

Hello World

intern: `http://vehicles:5000/hello_world/`

extern: `http://localhost:5000/api/vehicles/hello_world/`

Deze API call kan gebruikt worden om te checken of de service actief is en is toegankelijk met een GET request.

```
{
    'responseCode': 200,
    'message': 'Hello world!'
}
```

Create vehicle

intern: `http://vehicles:5000/create/`

extern: `http://localhost:5000/api/vehicles/create/`

Deze API call laat toe om een voertuig toe te voegen aan het systeem. Het vereist twee parameters `username` en `password` in de HTTP header en is toegankelijk met een POST request. Het vereist ook onderstaande parameters in de form body:

`vehicleNr` Het unieke nummer van dit voertuig.

`vehicleType` `TRAM` of `BUS`.

Indien alles in orde is, wordt onderstaande response gereturned.

```
{
    'responseCode': 200
}
```

List vehicles

intern: `http://vehicles:5000/list/`

extern: `http://localhost:5000/api/vehicles/list/`

Deze API call kan gebruikt worden om een lijst van alle voertuigen op te vragen. Deze is toegankelijk met een GET request en vereist geen extra parameters.

```
{
    "vehicles": [
        {
            "vehicleNr": 12334,
            "vehicleType": "BUS",
            "ownerId": 33
        },
        {
            "vehicleNr": 574187,
            "vehicleType": "TRAM",
            "ownerId": 45
        },
    ],
    "responseCode": 200
}
```

Delete vehicle

intern: `http://vehicles:5000/delete/<vehicle_nr>/`

extern: `http://localhost:5000/api/vehicles/delete/<vehicle_nr>/`

Deze API call laat de eigenaar van een voertuig toe om het voertuig te verwijderen indien andere gebruikers het voertuig nog niet beoordeeld hebben. De call is toegankelijk met een DELETE request en vereist twee parameters `username` en `password` in de HTTP header. Indien de eigenaar van het voertuig het voertuig beoordeeld heeft, worden deze beoordelingen ook verwijderd.

Indien alles in orde is, wordt onderstaande response gereturned.

```
{
    'responseCode': 200
}
```

Get Vehicle

intern: `http://vehicles:5000/get/<vehicle_nr>/`

extern: `http://localhost:5000/api/vehicles/get/<vehicle_nr>/`

Deze API call laat toe om informatie op te vragen over een specifiek voertuig. Het is toegankelijk met een GET request en vereist geen parameters.

```
{
    'vehicleNr': 420,
    "vehicleType": "BUS",
```

```
    "ownerId": 69,
    "responseCode": 200
}
```

Get Vehicles by owner

intern: `http://vehicles:5000/owner/<owner_id>/`

extern: `http://localhost:5000/api/vehicles/owner/<owner_id>/`

Deze API call laat toe om alle voertuigen op te vragen die zijn toegevoegd door de gespecifieerde gebruiker. Het is toegankelijk met een GET request en vereist geen verdere parameters.

```
{
    "vehicles": [
        {
            'vehicleNr': 83214,
            "vehicleType": "BUS",
        },
        {
            'vehicleNr': 24156,
            "vehicleType": "BUS",
        },
        {
            'vehicleNr': 87863,
            "vehicleType": "TRAM",
        },
    ],
    "responseCode": 200
}
```

2.3 Vehicle reviews service API

Hello World

intern: `http://vehiclereviews:5000/hello_world/`

extern: `http://localhost:5000/api/vehiclereviews/hello_world/`

Deze API call kan gebruikt worden om te checken of de service actief is en is toegankelijk met een GET request.

```
{
    'responseCode': 200,
    'message': 'Hello world!'
}
```

Create review

intern: `http://vehiclereviews:5000/create/`

extern: `http://localhost:5000/api/vehiclereviews/create/`

Deze API call kan gebruikt worden om een voertuig te beoordelen. Het is toegankelijk met een POST request en vereist twee parameters `username` en `password` in de HTTP header. Daarnaast vereist het een parameter `vehicleNr` en een parameter `score` die een float bevat in de form body.

Indien alles in orde is, wordt onderstaande response gereturned.

```
{
    'responseCode': 200
}
```

Get average score for vehicle

intern: `http://vehiclereviews:5000/vehicle/<vehicle_nr>/score/`

extern: `http://localhost:5000/api/vehiclereviews/vehicle/<vehicle_nr>/score/`

Deze call kan gebruikt worden om de gemiddelde score van een voertuig op te vragen. Het is toegankelijk met een GET request en vereist geen parameters.

```
{
    "score": 9.31,
    "responseCode": 200
}
```

Get reviews for user

intern: `http://vehiclereviews:5000/reviewer/<reviewer_id>/reviews/`

extern: `http://localhost:5000/api/vehiclereviews/reviewer/<reviewer_id>/reviews/`

Deze API call returned een lijst van alle reviews gemaakt door de gespecifieerde gebruiker. De call is toegankelijk met een GET request en vereist geen parameters.

```
{
    "reviews": [
        {
            "vehicleNr": 254215,
            "reviewerId": 20,
            "score": 2.5
        },
        {
            "vehicleNr": 87474,
            "reviewerId": 20,
```

```
            "score": 3.6
        },
        {
            "vehicleNr": 521541,
            "reviewerId": 20,
            "score": 3.0
        }
    ],
    "responseCode": 200
}
```

Get reviews for vehicle

intern: `http://vehiclereviews:5000/vehicle/<vehicle_nr>/reviews/`

extern: `http://localhost:5000/api/vehiclereviews/vehicle/<vehicle_nr>/reviews/`

Deze API call returned een lijst van alle reviews gemaakt voor het gespecifieerde voertuig. De call is toegankelijk met een GET request en vereist geen parameters.

```
{
    "reviews": [
        {
            "vehicleNr": 3265,
            "reviewerId": 21,
            "score": 2.5
        },
        {
            "vehicleNr": 3265,
            "reviewerId": 46,
            "score": 3.6
        },
        {
            "vehicleNr": 3265,
            "reviewerId": 3,
            "score": 0.3
        }
    ],
    "responseCode": 200
}
```

Delete reviews for vehicle

intern: `http://vehiclereviews:5000/vehicle/<vehicle_nr>/reviews/delete/`

extern: `http://localhost:5000/api/vehiclereviews/vehicle/<vehicle_nr>/reviews/delete/`

Deze API call voorziet een manier om alle beoordelingen voor het gespecifieerde voertuig te verwijderen. Het is toegankelijk met een DELETE request en vereist twee parameters `username` en `password` in de HTTP header.

Indien alles in orde is, wordt onderstaande response gereturned.

```
{
    'responseCode': 200
}
```

2.4 Stops service API

Hello World

intern: `http://stops:5000/hello_world/`

extern: `http://localhost:5000/api/stops/hello_world/`

Deze API call kan gebruikt worden om te checken of de service actief is en is toegankelijk met een GET request.

```
{
    'responseCode': 200,
    'message': 'Hello world!'
}
```

Get provinces

intern: `http://stops:5000/provinces/`

extern: `http://localhost:5000/api/stops/provinces/`

Deze call laat toe om een lijst van provincies op te vragen door middel van een GET request.

```
{
    'provinces': [
        {
            'id': 1,
            'name': "Antwerpen",
        },
        {
            'id': 2,
            'name': "Oost-Vlaanderen",
        },
    ],
    'responseCode': 200,
}
```

Get stops for province

intern: `http://stops:5000/provinces/<province_id>/stops/`

extern: `http://localhost:5000/api/stops/provinces/<province_id>/stops/`

Deze API call laat toe om alle haltes op te vragen die zich bevinden binnen een provincie door middel van een GET request.

```
{
    'stops': [
        {
            "stopId": 101000,
            "stopName": "A. Chantrainestraat",
            "cityName": "Wilrijk"
        },
        {
            "stopId": 101001,
            "stopName": "Zurenborg",
            "cityName": "Antwerpen"
        },
        {
            "stopId": 101002,
            "stopName": "Verenigde Natieslaan",
            "cityName": "Hoboken"
        }
    ],
    'responseCode': 200
}
```

Get lines for province

intern: `http://stops:5000/provinces/<province_id>/lines/`

extern: `http://localhost:5000/api/stops/provinces/<province_id>/lines/`

Deze API call laat toe om alle lijnen binen een provincie op te vragen d.m.v. een GET request.

```
{
    "lines": [
        {
            "id": 2,
            "name": "Hoboken - P+R Merksem"
        },
        {
            "id": 3,
            "name": "P+R Merksem - P+R Melsele"
        },
        {
            "id": 4,
```

```
            "name": "Hoboken - Groenplaats - Silsburg"
        },
        {
            "id": 298,
            "name": "Boom - Niel - Berchem"
        }
    ],
    'responseCode': 200
}
```

Get stops for line

intern: `http://stops:5000/provinces/<province_id>/lines/<line_id>/stops/`

extern: `http://localhost:5000/api/stops/provinces/<province_id>/lines/<line_id>/stops/`

Deze API call laat toe om alle haltes die bediend worden door een lijn op te vragen.

```
{
    "responseCode": 200,
    "stops": [
        {
            "stopId": 107285,
            "stopName": "Kattenbroek",
            "cityName": "Edegem"
        },
        {
            "stopId": 103999,
            "stopName": "Groeningenlei",
            "cityName": "Kontich"
        },
        {
            "stopId": 103998,
            "stopName": "Groeningenlei",
            "cityName": "Kontich"
        }
    ]
}
```

Get cities

intern: `http://stops:5000/cities/`

extern: `http://localhost:5000/api/stops/cities/`

Deze API call laat toe om alle gemeentes op te vragen.

```
{
    "cities": [
```

```
        {
            "cityId": 1530,
            "cityName": "'S-GRAVENVOEREN"
        },
        {
            "cityId": 129,
            "cityName": "'S-GRAVENWEZEL"
        },
        {
            "cityId": 2321,
            "cityName": "'S-HERENELDEREN"
        },
        {
            "cityId": 892,
            "cityName": "'T LUIK"
        }
    ],
    "responseCode": 200
}
```


Get stops for city

intern: `http://stops:5000/cities/<city_id>/stops/`

extern: `http://localhost:5000/api/stops/cities/<city_id>/stops/`

Deze API call laat toe om alle haltes binnen een gemeente op te vragen.


```
{
    "responseCode": 200,
    "stops": [
        {
            "stopId": 209044,
            "stopName": "Collegestraat",
            "cityName": "Zottegem"
        },
        {
            "stopId": 209878,
            "stopName": "Station perron 6",
            "cityName": "Zottegem"
        },
        {
            "stopId": 208076,
            "stopName": "Meileveld",
            "cityName": "Strijpen"
        },
        {
            "stopId": 219017,
            "stopName": "Vandendriesschestraat",
            "cityName": "Zottegem"
```

```
        },
        {
            "stopId": 209699,
            "stopName": "Bevegemse Vijvers",
            "cityName": "Zottegem"
        },
        {
            "stopId": 208143,
            "stopName": "Meerlaan",
            "cityName": "Strijpen"
        }
    ]
}
```

2.5 Stop reviews service API

Hello World

intern: `http://stopreviews:5000/hello_world/`

extern: `http://localhost:5000/api/stopreviews/hello_world/`

Deze API call kan gebruikt worden om te checken of de service actief is en is toegankelijk met een GET request.

```
{
    'responseCode': 200,
    'message': 'Hello world!'
}
```

Create review

intern: `http://stopreviews:5000/create/`

extern: `http://localhost:5000/api/stopreviews/create/`

Deze API call kan gebruikt worden om een halte te beoordelen. Het is toegankelijk met een POST request en vereist twee parameters `username` en `password` in de HTTP header. Daarnaast vereist het een parameter `stopId` en een parameter `score` die een float bevat in de form body.

Indien alles in orde is, wordt onderstaande response gereturned.

```
{
    'responseCode': 200
}
```

Get average score for stop

intern: `http://stopreviews:5000/stop/<stop_id>/score/`

extern: `http://localhost:5000/api/stopreviews/stop/<stop_id>/score/`

Deze API call kan gebruikt worden om de gemiddelde score van een halte op te vragen met behulp van een GET request.

```
{
    'score': 6.39,
    'responseCode': 200
}
```

Get reviews by user

intern: `http://stopreviews:5000/reviewer/<user_id>/reviews/`

extern: `http://localhost:5000/api/stopreviews/reviewer/<user_id>/reviews/`

Deze API call maakt het mogelijk om alle reviews op te vragen voor de gespecifieerde gebruiker.

```
{
    "reviews": [
        {
            "stopId": 100671,
            "reviewerId": 32,
            "score": 3.67
        },
        {
            "stopId": 100673,
            "reviewerId": 32,
            "score": 0.2
        }
    ],
    "responseCode": 200
}
```

Get reviews by stop

intern: `http://stopreviews:5000/stop/<stop_id>/reviews/`

extern: `http://localhost:5000/api/stopreviews/stop/<stop_id>/reviews/`

Deze API call maakt het mogelijk om alle reviews op te vragen voor de gespecifieerde halte.

```
{
    "reviews": [
        {
```

```
            "stopId": 100671,
            "reviewerId": 32,
            "score": 4.65
        },
        {
            "stopId": 100671,
            "reviewerId": 64,
            "score": 2.98
        }
    ],
    "responseCode": 200
}
```

2.6 Errors

Indien een request niet correct verwerkt kan worden door de API wordt een JSON object teruggeven met onderstaande parameters:

`errorMessage` Een tekstuele boodschap omtrent de fout.

`responseCode` De resulterende HTTP status (400, 500, etc).

`causeErrorMessage` Een tekstuele boodschap van een achterliggende fout. Indien deze niet van toepassing is is deze `"N/A"`.

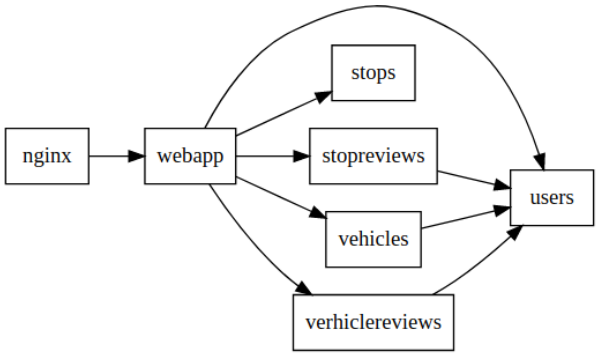
`causeErrorStatus` De HTTP status van de achterliggende fout. Indien deze niet van toepassing is, is deze -1.

3 Service structuur

De applicatie bestaat uit de volgende services:

- **Nginx** reverse proxy,
- **Webapp** voorziet de web UI,
- **Users** voorziet toegang tot de users,
- **Users-db** persistence service voor de users service,
- **Vehicles** toevoegen, opvragen en verwijderen van voertuigen,
- **Vehicles-db** persistence,
- **Vehiclereviews** toevoegen en opvragen van voertuig beoordelingen,
- **Vehiclereviews-db** persistence,
- **Stops** Opvragen van haltes,
- **Stopreviews** toevoegen en opvragen van halte beoordelingen,
- **Stopreviews-db** persistence.

Onderstaande figuur illustreert de verschillende services en hoe ze onderling communiceren. Merk op dat veel services een eigen persistence service hebben, deze persistence services zijn niet toegevoegd aan de illustratie om het duidelijker te maken.



De nginx reverse proxy biedt toegang tot de webapp. De webapp service biedt een HTML-gecodeerde user interface aan. De webapp haalt informatie uit alle andere services (stops, stopreviews, vehicles, vehiclereviews, users). De andere services communiceren op hun beurt ook met de users service om requests te verifiëren. Zo wordt er bijvoorbeeld bij het aanmaken van een voertuig, nagegaan of de gebruiker geregistreerd is.

Ik heb ervoor gekozen om niet één grote persistence service toe te voegen aangezien deze een bottleneck zou kunnen vormen. Door meerdere persistence services te hebben kunnen deze gemakkelijker herschaald worden (voor elke users service die je instantieert, instantieer je ook een users-db service bijvoorbeeld). Hierdoor blijft de load beter verspreid. Om diezelfde reden heb ik ook voor de voertuigen en haltes een aparte recensie service toegevoegd.

Voor het uitwerken van de services heb ik gebruikgemaakt van het tutorial op `testdriven.io`. De meeste services lijken dan ook op de users service in het tutorial. Ik heb ook gebruikgemaakt van de tools die in het tutorial aan bod komen. Daarnaast heb ik ook van de `passlib` Python package gebruikgemaakt om wachtwoorden in de database te versleutelen, en ik heb van de `requests` Python package gebruikgemaakt om GET, POST en DELETE requests te sturen. Alle services zijn voorzien van een `requirements.txt` file waarmee `pip` automatisch de packages installeert voor elke service.