Attractiepark Lake Side Mania

Technisch ontwerp

M1 - Software

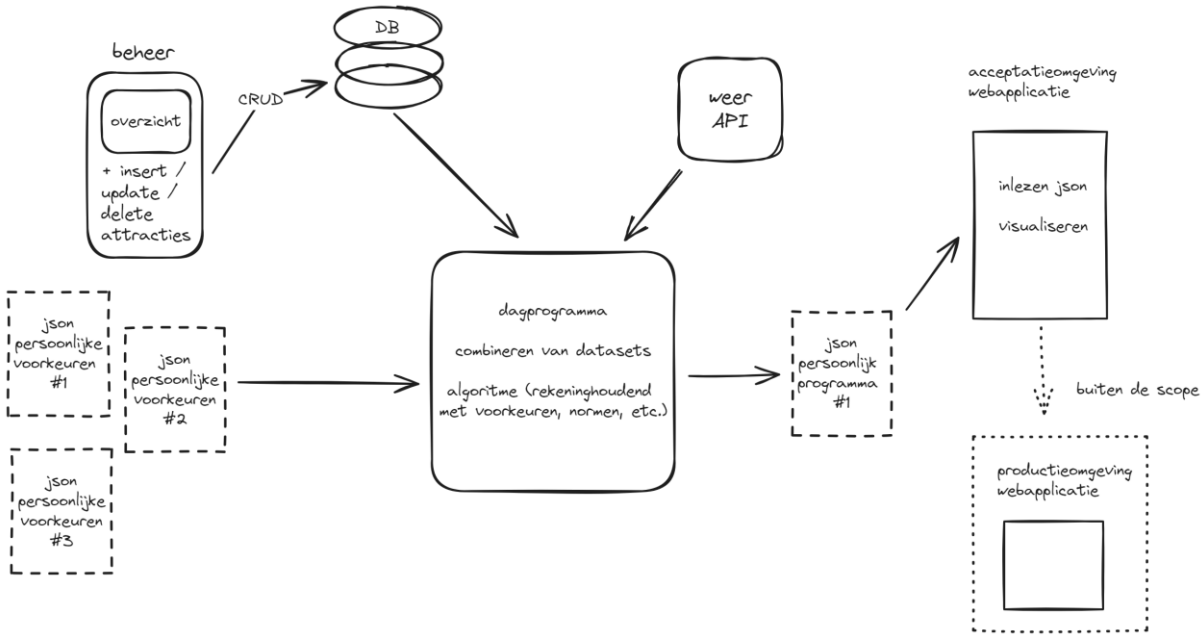
Daan Maat, S1213186
11-10-2024

Inhoud

Inleiding

Dit document beschrijft het technisch ontwerp voor de software die wordt ontwikkeld voor attractiepark Lake Side Mania. Dit ontwerp omvat specificaties voor de architectuur, het databaseontwerp en de functionaliteiten van het systeem, zoals de beheeromgeving en de generatie van gepersonaliseerde dagprogramma's voor bezoekers. De software integreert diverse componenten waaronder een beheeromgeving voor het beheren van attracties, winkels en horecagelegenheden, een algoritme dat gebruik maakt van persoonlijke voorkeuren van bezoekers en real-time weergegevens om op maat gemaakte dagprogramma's te genereren.

Systeemoverzicht / architectuur



Beheerder / beheeromgeving: Heeft de mogelijkheid om een overzicht van attracties / winkels / horecagelegenheden te bekijken en te beheren door middel van CRUD-operaties (Create, Read, Update, Delete) in de database (DB).

Database (DB): Een centrale opslagplaats waar gegevens over attracties / winkels / horecagelegenheden worden bewaard.

Weer API: Een externe interface waarmee de applicatie toegang krijgt tot real-time weergegevens, die invloed kunnen hebben op de dagprogramma's.

Dagprogramma-algoritme: Het centrale mechanisme dat de datasets combineert - de voorkeuren van de bezoeker (waarschijnlijk in JSON-formaat aangeleverd), de attractiegegevens uit de DB en de weersvoorspellingen van de Weer API - om een gepersonaliseerd dagprogramma te genereren.

Input-JSON [Persoonlijk Voorkeuren]: De gegevensstructuur waarin persoonlijke voorkeuren van bezoekers worden ontvangen, o.a. hun leeftijd, lengte, gewicht en voorkeuren voor attractietypes en eten.

Output-JSON [Dagprogramma]: Het resultaat van het dagprogramma-algoritme, een uitvoer in JSON-formaat met een lijst van attracties en horecagelegenheden aangepast aan de persoonlijke voorkeuren van de bezoeker en het huidige weer.

Acceptatietoetsing/Webapplicatie: Hier vindt de validatie plaats of het gegenereerde dagprogramma voldoet aan de gestelde criteria.

Buiten de scope: Er zijn twee activiteiten gemarkeerd als buiten de scope van dit technische ontwerp: het inlezen van de JSON en het visualiseren van het dagprogramma en de productieomgeving/webapplicatie zelf.

Databaseontwerp / datamodel

Rekeninghoudend met de vereisten FR4 en FR5 is het volgende databaseontwerp opgesteld. In deze database is het mogelijk om gegevens op te slaan van attracties, winkels en horecagelegenheden.

Voorziening			*Opmerking*
PK	Id	int(11) – Auto Increment	
	Naam	Varchar(255), Not Null	
	Type	Varchar(255), Not Null	*Achtbaan, Water, Draaien, Familie, Simulator, Winkel, Horeca*
	Overdekt	Tinyint(4), Null	*True/false*
	Geschatte_wachttijd	Int(11), Not Null	*In minuten*
	Doorlooptijd	Int(11), Not Null	*In minuten*
	Actief	Tinyint(4), Not Null	*True/false*
	Attractie_min_lengte	Int(11), Null	*In centimeters*
	Attractie_max_lengte	Int(11), Null	*In centimeters*
	Attractie_min_leeftijd	Int(11), Null	*In jaren*
	Attractie_max_gewicht	Int(11), Null	*In kilo's*
	Productaanbod	Varchar(255), Null	***Winkel**: Souvenirs, Zomerartikelen, Regenaccessoires* ***Horeca**: Pizza, Patat, Pannenkoeken, Pasta, Snoep, IJs*

Beheeromgeving

Dit heeft betrekking op use cases 1, 2, 3 en 4.

Vanuit de beheeromgeving kan de gebruiker attractiegegevens, winkels en horecagelegenheden toevoegen, wijzigen, verwijderen en opvragen. De gegevens zijn opgeslagen in een MySQL-database. De technische specificaties van deze database staat beschreven onder het kopje *Databaseontwerp / datamodel*.

De beheeromgeving wordt aangestuurd middels een grafische user interface. Voor de GUI wordt er gebruik gemaakt van de (aangeleverde) GUI module.

De aansturing van de database vanuit de applicatie (lees: Python programma) wordt gedaan met behulp van de (aangeleverde) Database module.

Uiteindelijk dient de gegevens, die in de database staan, als input (dataset) voor de applicatie die dagprogramma's genereert voor de bezoeker.

Genereren van dagprogramma's

Dit heeft betrekking op use case 6.

Het genereren van een dagprogramma (algoritme) voor een bezoeker omvat een aantal stappen.

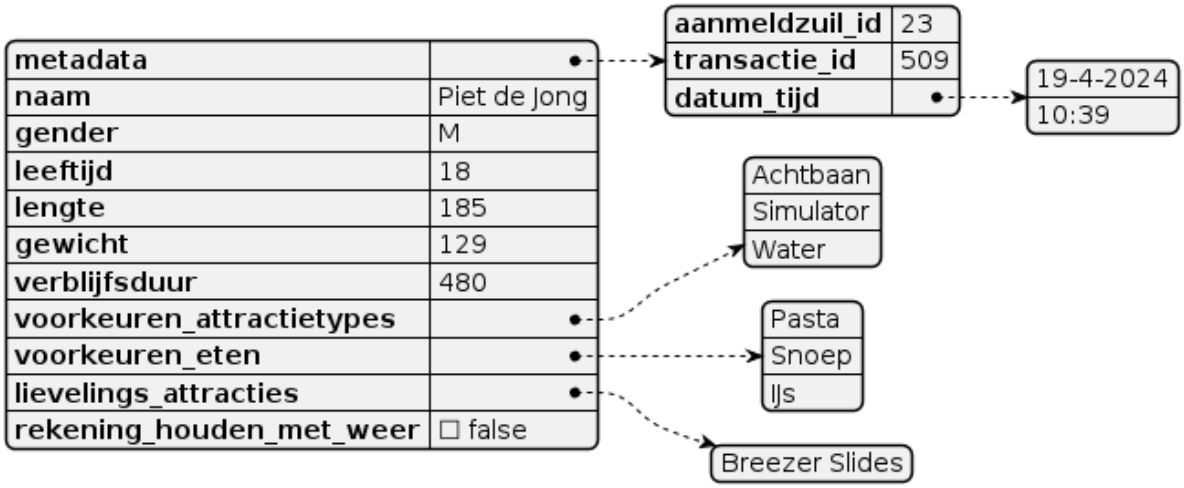
Invoer en gereed maken van data

Zoals beschreven in het systeemoverzicht, krijgt het algoritme input vanuit de database, maar ook input vanuit de persoonlijke voorkeuren van de bezoeker, in de vorm van een .JSON bestand. De technische specificaties van dit .JSON bestand is als volgt:

persoonlijke_voorkeuren	
naam	string
gender	string
leeftijd	number
lengte	number
gewicht	number
verblijfsduur	number
rekening_houden_met_weer	boolean
▾ metadata	object
· aanmeldzuil_id	number
· transactie_id	number
· ▸ datum_tijd	array
▸ voorkeuren_attractietypes	array
▸ voorkeuren_eten	array
▸ lievelings_attracties	array

Ingevuld met (dummy) data ziet er het zo uit:

```
{
    "metadata": {
        "aanmeldzuil_id": 23,
        "transactie_id": 509,
        "datum_tijd": ["19-4-2024", "10:39"]
    },
    "naam": "Piet de Jong",
    "gender": "M",
    "leeftijd": 18,
    "lengte": 185,
    "gewicht": 129,
    "verblijfsduur": 300,
    "voorkeuren_attractietypes": ["Achtbaan", "Simulator", "Water"],
    "voorkeuren_eten": ["Pasta", "Snoep", "IJs"],
    "lievelings_attracties": ["Breezer Slides"],
    "rekening_houden_met_weer": false
}
```



De .JSON in combinatie met de gegevens uit de database (zie schema specificatie onder het kopje *datamodel / databaseontwerp*) dient als invoer voor het algoritme / genereren van de dagprogramma's.

Verwerken en het algoritme

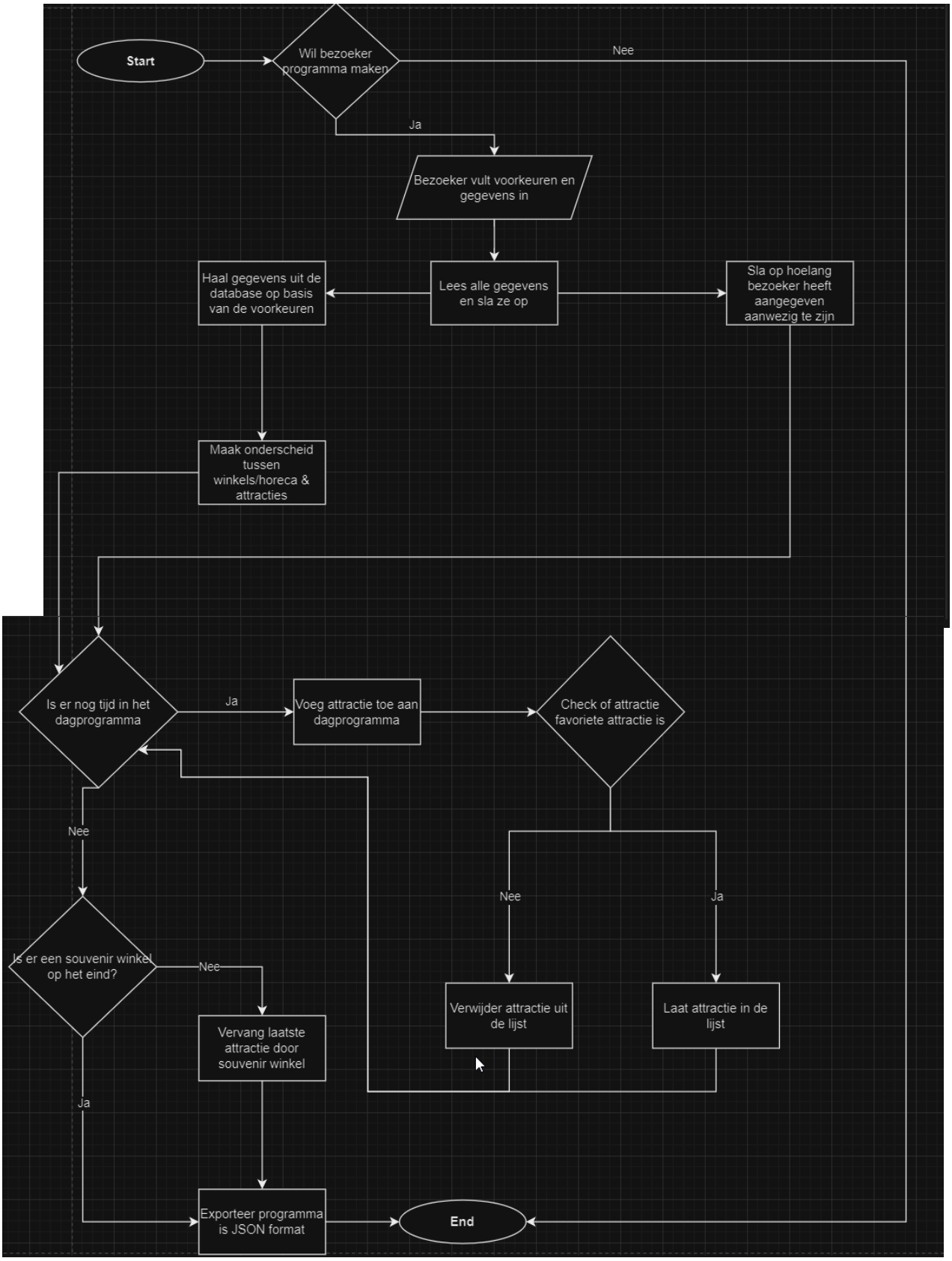
De data dient gereed te worden gemaakt ter verwerking voor het algoritme. Met andere woorden: de twee datasets moeten vooraf het daadwerkelijk genereren van een dagprogramma zodanig gestructureerd worden dat het algoritme er mee uit de voeten kan.

In de requirements zijn een aantal criteria beschreven waaraan het algoritme moet voldoen. Hieronder staan ze nog een keer beschreven:

Het systeem moet op basis van de verblijfsduur van de bezoeker en de geschatte wachttijd en doorlooptijd van de attracties, winkels en horeca bepalen hoeveel attracties in het programma worden opgenomen.	A-TC1	Must
Het systeem mag alleen attracties toevoegen aan het programma waar een bezoeker toegang tot mag hebben, gebaseerd op leeftijd, lengte en gewicht	A-TC2	Must
Het systeem moet het programma eerst vullen met attractietypes waar de bezoeker de voorkeur aan heeft gegeven. Attracties mogen maximaal één keer voorkomen, behalve lievelingsattracties die mogen twee keer in het programma voorkomen. Alleen wanneer er ruimte in het programma is, worden andere attracties toegevoegd.	A-TC3	Must
Het systeem moet ervoor zorgen dat de attractietypes gemixt voorkomen in het programma.	A-TC4	Should
Het systeem moet voor iedere bezoeker minimaal één horecagelegenheid toevoegen aan het programma, die aansluit bij de opgegeven voorkeuren voor eten.	A-TC5	Must
Het systeem moet voor bezoekers die langer dan 4 uur in het park verblijven ongeveer iedere 2 uur een horecagelegenheid toevoegen aan het programma.	A-TC6	Should
Het systeem moet voor iedere bezoeker aan het einde van het programma een souvenirwinkel toevoegen.	A-TC7	Must
Het systeem moet via een weer-API kunnen uitlezen wat de temperatuur van de dag is en of het gaat regenen.	A-TC8	Could
Het systeem moet voor iedere bezoeker aan de start van het programma een winkel met zomerartikelen toevoegen, als de temperatuur boven de 20 graden komt.	A-TC9	Could
Het systeem moet minimaal één ijswinkel aan het programma toevoegen, als de temperatuur boven de 20 graden komt.	A-TC10	Could
Het systeem moet voor iedere bezoeker aan de start van het programma een winkel met regenaccessoires toevoegen als het gaat regenen.	A-TC11	Could

Om het proces / de flow inzichtelijk te maken is er een stroomdiagram en pseudo-code gemaakt die het algoritme weergeeft dan wel toelicht.

Flow diagram op volgende pagina


Start
Wil bezoeker programma maken
Nee
Ja
Bezoeker vult voorkeuren en gegevens in
Haal gegevens uit de database op basis van de voorkeuren
Lees alle gegevens en sla ze op
Sla op hoelang bezoeker heeft aangegeven aanwezig te zijn
Maak onderscheid tussen winkels/horeca & attracties
Is er nog tijd in het dagprogramma
Ja
Voeg attractie toe aan dagprogramma
Check of attractie favoriete attractie is
Nee
Nee
Ja
Is er een souvenir winkel op het eind?
Nee
Verwijder attractie uit de lijst
Laat attractie in de lijst
Vervang laatste attractie door souvenir winkel
Ja
Exporteer programma is JSON format
End

Pseudo-code

If bezoekers programma is ontvangen
Then Lees voorkeuren uit en stop deze in een list/dictionary
Loop door de lijst en filter alle voorzieningen waar de bezoeker voorkeuren en toegang tot heeft.
Then zet alle voorzieningen in een dagprogramma
If er nog tijd over is in het dagprogramma vul het programma met andere voorzieningen
If er geen souvernir winkel op het einde van de dag is **and** er is geen tijd meer
Then vervang de laatste attractie voor een souvenir winkel.

Uitvoer in .JSON

Na dat het algoritme is uitgevoerd rolt er een persoonlijk programma (dagprogramma) uit voor de bezoeker. Dit gebeurt in de vorm van een .JSON bestand. Het visualiseren van de data de in het .JSON bestand staat valt buiten de scope van dit project, echter is er wel een acceptatieomgeving waarin het .JSON bestand kan worden getest.

De technische specificaties van dit .JSON bestand is als volgt:

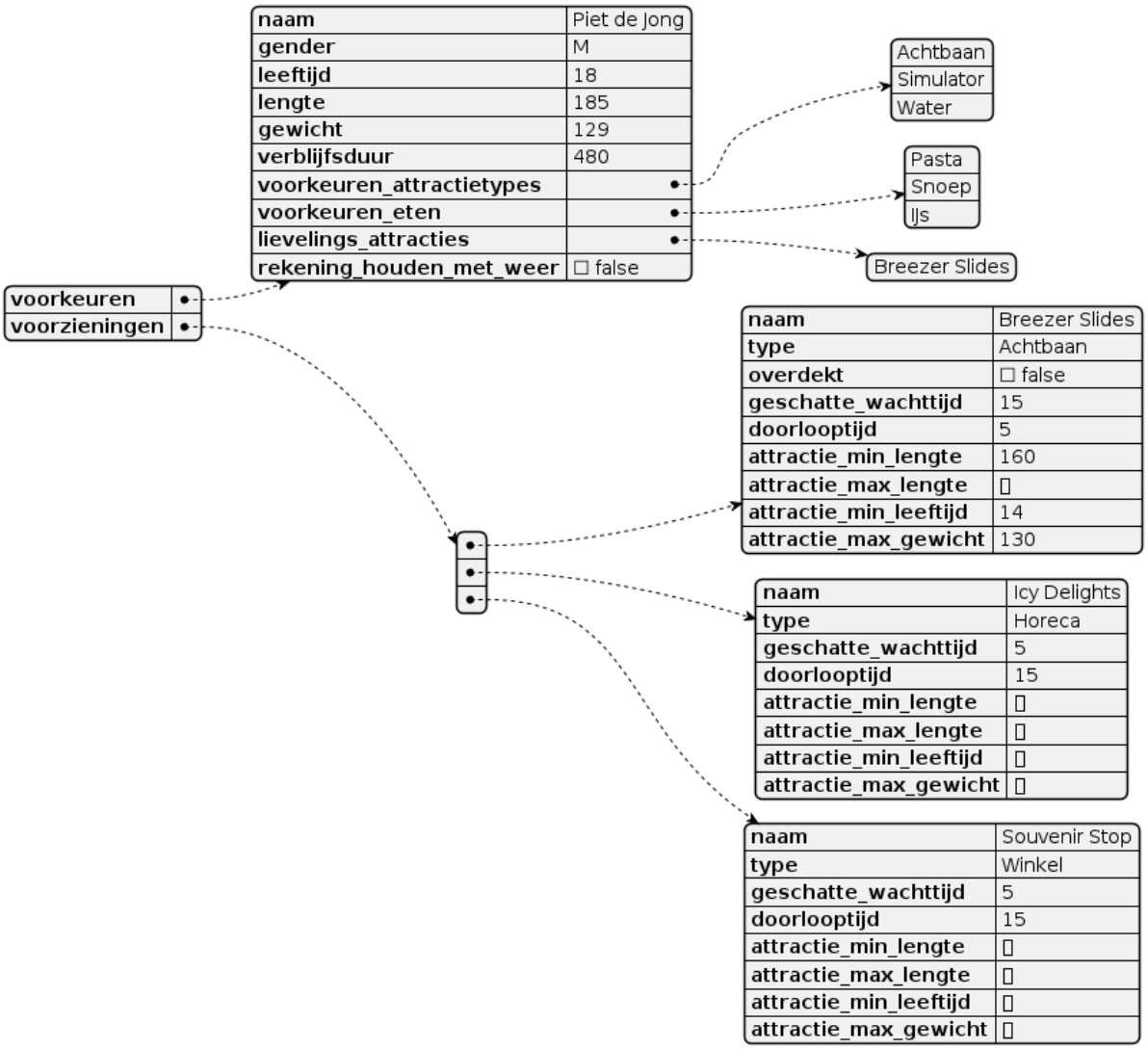
```
{
    "voorkeuren": {
        # alle gegevens van de persoonlijke voorkeuren .json
        # dus naam, gender, leeftijd etc.
        # exclusief metadata
    },
    "voorzieningen": [
        {
            "naam": "Breezer Slides",
            "type": "Achtbaan",
            "overdekt": false,
            "geschatte_wachttijd": 15,
            "doorlooptijd": 5,
            "attractie_min_lengte": 160,
            "attractie_max_lengte": null,
            "attractie_min_leeftijd": 14,
            "attractie_max_gewicht": 130
        },
        {
            "naam": "Icy Delights"
            # etc. etc.
        }
    ]
}
```

De gegevens uit de database worden voor de volledigheid ook opgenomen in het .JSON bestand.

Key	Datatype
Naam	String
Type	String
Overdekt	Boolean
Geschatte_wachttijd	Integer
Doorlooptijd	Integer
Attractie_min_lengte	Integer – nullable*
Attractie_max_lengte	Integer – nullable*
Attractie_min_leeftijd	Integer – nullable*
Attractie_max_gewicht	Integer – nullable*

* het kan of leeg zijn (null) of gevuld zijn (getal)

Zie afbeelding hieronder voor een voorbeeld met (dummy) data.


naam
Piet de Jong
gender
M
leeftijd
18
lengte
185
gewicht
129
verblijfsduur
480
voorkeuren_attractietypes
voorkeuren_eten
lievelings_attracties
rekening_houden_met_weer
false
Achtbaan
Simulator
Water
Pasta
Snoep
Ijs
Breezer Slides
voorkeuren
voorzieningen
naam
Breezer Slides
type
Achtbaan
overdekt
false
geschatte_wachttijd
15
doorlooptijd
5
attractie_min_lengte
160
attractie_max_lengte
attractie_min_leeftijd
14
attractie_max_gewicht
130
naam
Icy Delights
type
Horeca
geschatte_wachttijd
5
doorlooptijd
15
attractie_min_lengte
attractie_max_lengte
attractie_min_leeftijd
attractie_max_gewicht
naam
Souvenir Stop
type
Winkel
geschatte_wachttijd
5
doorlooptijd
15
attractie_min_lengte
attractie_max_lengte
attractie_min_leeftijd
attractie_max_gewicht

Weer API

Dit heeft betrekking op use case 7.

De wens is om het dagprogramma aan te passen op basis van de weersvoorspelling. Een derde dataset dus, maar niet cruciaal voor de werking van het systeem. Voor het ophalen van het weer wordt er gebruik gemaakt van: -

Tooling / technologie stack

Software

Visual Studio Code

MySQL Workbench

Draw.io

Technologie

Python 3

MySQL Community Edition