STUDENTEN-CONSUMPTIE THOMAS MORE

Data Distribution

Abstract

Onderzoek naar de consumptie van studenten, ter verbetering van de huidige middelen.

Casteels Shila -

r0257725 – 3IMS

Inhoudsopgave

Inleiding

Aan de hand van de gegevens die we verkregen hebben via de Drones, gaan we 2 cruciale business vragen verder uitdiepen: Wanneer gaan de meeste studenten eten? en hoe vaak per dag gaan ze naar de cafetaria?
Op basis van deze business vragen en de verworven gegevens kunnen we bepalen hoeveel personeel, op welk tijdstip aanwezig moet zijn in de cafetaria. Alsook gaan we bepalen tussen welke tijdstippen de cafetaria best open kan blijven, maar ook of het noodzakelijk is om extra automaten op de tussenverdiepingen te plaatsen. Dit alles zullen we bepalen aan de hand van de verzamelde gegevens die we verder gaan analyseren en uiteindelijk gaan integreren in een dashboard dat dit makkelijk vertaalt naar de verantwoordelijke voor de cafetaria. Hierbij zorgen we steeds voor gegevens die REST-full genoteerd zijn, met CRUD (Create, Read, Update en Delete) in het achterhoofd.

Use-Case

Allereerst gaan we de 2 business vragen verder uitwerken en bepalen welke attributen en resources we hiervoor nodig zullen hebben.
Als eerste businessvraag gaan we bepalen wanneer de meeste studenten hun middagmaal consumeren.
Hiervoor zijn we voornamelijk afhankelijk van de drone, die zich in de cafetaria bevindt.
Hierbij gaan we als resource de aantal Device_id's die migreren van 1 plaats naar de andere dus het aantal migraties gebruiken, alsook de attributen locatie, locatie_id, , tijd en bezetting.
Op basis van deze gegevens gaan we proberen te achterhalen tussen welke 2 tijdstippen we het hoogste aantal migraties kunnen detecteren. Afhankelijk van deze aantallen kunnen we dan later bepalen of we genoodzaakt zijn een extra werkneemster aan te stellen in de cafetaria voor deze periode.

Als tweede businessvraag gaan we controleren hoe vaak een student zich naar de cafetaria begeeft voor een snack of drankje. Hierbij zullen we de gegevens van zowel de drones in de lokalen als de drones in de cafetaria gebruiken.
Op basis van de gegevens van de drones in de lokalen gaan we bepalen hoe vaak de student zijn lokaal verlaat voor een kleine snack. Deze gegevens gaan we controleren met die van de cafetariadrone alsook met de geopende uren van de cafetaria. Dit als zekerheid dat de student een kleine snack wil consumeren en niet gewoon een plaspauze neemt.
Afhankelijk van deze gegevens kunnen we eventueel ook de omzet van de cafetaria opvragen, alsook gegevens verzamelen over de verkooptransacties per tijdstip. Dit doen we opdat we kunnen bepalen of de student zijn snack in de cafetaria aankoopt of gebruik maakt van de automaten.
Op basis van deze gegevens kunnen we dan zien of de student eerder de automaat verkiest of de cafetaria. Deze resultaten kunnen dan leiden tot een extra automaat op de tussenverdiepingen, nabij de lokalen, of extra openingsuren van de cafetaria.

De gegevens van de cafetaria kassa zullen we implementeren via een POST /Kassa voor de verkoopcijfers van de cafetaria, alsook een POST /Rapport voor de tijdsbepaling van deze aankopen. Deze voegen we samen met de aantal migraties en de locatie, locatie_id, tijd en bezetting van de desbetreffende drones.

Voor deze taak, hebben we besloten om verder in te gaan op de al dan niet noodzakelijke aankoop van automaten op de tussenverdiepingen.
Als toelichting geven we een voorbeeld:

> Vb.: De eerste voorziene pauze is voorzien tussen 10u30 en 10u45. Om te kunnen bepalen hoeveel studenten tijdens deze tijdspanne zich naar de cafetaria gaan begeven, zullen we de bezetting bepalen in een specifiek lokaal van 10u15 tot 10u30. Tijdens de pauze, zullen we de bezetting in de cafetarie bepalen tussen 10u30 en 10u45.
> Een eerste onderdeel is controleren of de devices waargenomen in het lokaal, ook waargenomen worden in de cafetaria. Indien dit zo is gaan we deze gegevens controleren met de rapporeten op dat moment, om zo te proberen bepalen of deze studenten al dan niet een consumptie hebben aangekocht in de cafetaria.

Beschrijving API

Zoals eerder gezegd zullen we de use-case rond het al dan niet aanschaffen van automaten op de tussenverdiepingen verder gaan analyseren op basis van de verzamelde gegevens.

In dit onderdeel zullen we de benodigde gegevens verder uitdiepen om exact te bepalen wat we nodig zullen hebben om onze analyse te kunnen voltrekken.

Als eerste gaan we kijken naar de verschillende locaties van waaruit studenten kunnen komen of gaan. Hiervoor hebben we per locatie het id van de locatie en het tijdstip waarop we de locatie gaan analyseren alvast nodig.
Verder is het ook van belang te weten, hoeveel studenten er zich op deze locatie kunnen bevinden en hoeveel er zich op dat moment effectief bevinden. Dit doen we door naar de bezettingsgraad te kijken, alsook te controleren of de locatie al dan niet bezet is en zo ja door welke klas. Afhankelijk van de klas die op dat moment in het lokaal hoort te zitten hebben we al enig idee van het aantal studenten dat er op dat moment hoort te zitten.
Dit kunnen we dan in eerste instantie al vergelijken met het werkelijk aantal geregistreerde studenten op de locatie. Dit door een peiling te doen per 15 minuten van de geregistreerde apparaten met een rssi van minimum -70.

Dit alles doen we zodat we op een bepaald tijdstip de migratie van de studenten kunnen bepalen en dus ook of deze migratie overeenkomt met deze van de cafetaria. Zo weten we ongeveer hoeveel van de studenten tijdens de pauze al dan niet naar de cafetaria gaan. Wij gaan hierbij van de veronderstelling dat alle studeten ook effectief hun GSM meenemen indien zij een bepaalde locatie verlaten.

Vervolgens gaan we kijken naar de kassa van de cafetaria op een bepaald tijdstip, alsook naar het rapport van de verkopen op het tijdstip overeenkomend met de pauze waarop we de migratie waarnemen.
Op basis van de verkopen vastgelegd in elk rapport gaan we kijken of er al dan niet een verkoop plaatsgevonden heeft op het bepaalde tijdstip.
Zo kunnen we een veronderstelling maken, dat wanneer 15 studenten tijdens de pauze naar de cafetaria gaan maar we op dat moment slechts 5 verkopen waarnemen, dat slechts 5 van deze studenten een effectieve aankoop in de cafetaria voltrokken hebben. Wij gaan dan van de veronderstelling dat de andere studenten hun snack dan aankopen met behulp van de automaten.

Indien gewenst, kunnen wij ook specifieker kijken naar het aangekochte product op dat moment, de aantallen ervan en de kostprijs van deze producten. Dit alles om een idee te hebben wat men best in de automaten aanbiedt, indien men deze op de tussenverdiepingen zou plaatsen.
Verder kunnen we ook de betalingswijze van de student raadplegen, om te bepalen of de eventuele automaten best ook over een bankcontact applicatie beschikken.

In verdere delen van deze paper, zullen we onze resources schematiseren en gaan we ook de oorsprong van onze data verder achterhalen.

- Resource: locaties
    - Voor het opvragen van de data voor de resource locatie maken we gebruik van volgende GET-codes:
        - GET /locatie
        - GET /locatie/{id} (id=Cafetaria)
        - GET /locatie/{id}/migratie
    - Indien men een locatie wil toevoegen maken we gebruik van volgende PUSH-code. Er wordt verwacht dat men zelf de locatie van een id voorziet. De aantal geregistreerde machines wordt automatisch uit de data van de drone gehaald. De migratie wordt per pauze automatisch berekend op basis van de geregistreerde studenten voor en na de pauze.
        - De gekozen pauze_tijd moet echter wel overeenkomen met de op voorhand vastgesteld pauze. Indien deze niet overeenkomt zal er een 404-foutmelding verschijnen waarbij meegedeeld wordt dat deze pauze-tijd onbestaand is en eerst dient aangemaakt te worden in de resource “pauze”. De pauze_id wordt automatisch toegevoegd aan de resource pauze.

            - PUSH /locatie
            - V.b.: Body: {
                “id”: “Cafetaria”;
                “pauze_id”: “20161114”;
                “pauze_tijd”: “10h45”;
                “klas”: “C2.25”;
                “aantal_studenten_in_klas”: 21;
                }

    - Schema:

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "locatie": {
      "type": "object",
      "properties": {
        "id": {
          "type": "string"
        },
        "migratie": {
          "type": "integer"
        },
        "pauze_id": {
          "type": "integer"
        },

        "pauze_tijd": {
          "type": "integer"
        },
        "bezetting": {
          "type": "integer"
        },
        "bezet": {
```

```
          "type": "boolean"
        },
        "klas": {
          "type": "string"
        },
        "aantal_studenten_in_klas": {
          "type": "integer"
        },
         “aantal_geregistreerde_studenten”: {
            “type”: “integer”
      },
      "required": [
        "id",
        "migratie",
        "tijd",
        "bezetting",
        "bezet",
        "klas",
        "aantal_studenten_klas”,
         “aantal_geregistreerde_studenten”
      ]
    }
  },
  "required": [
    "locatie"
  ]
}
```

- Resource: pauze
    - Voor het opvragen van de data voor de resource pauze maken we gebruik van volgende GET-codes:
        - GET /pauze
        - GET /pauze/{pauze_id} (V.b. id = 20161029)
        - GET /pauze/{pauze_id}/pauze_tijd (V.b. pauze-tijd: 1045)
    - Indien men een pauze wil toevoegen maken we gebruik van volgende POST-code. Het systeem creëert zelf een id op basis van de huidige datum. Indien men een locatie toevoegt wordt er ook automatisch een nieuwe pauze_id en andere properties aangemaakt binnen de resource pauze.
        - POST /pauze
        - V.b. Body: { “pauze_tijd”= “1045”;}
    - Schema:

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "pauze": {
      "type": "object",
      "properties": {
        "pauze_id": {
          "type": "integer"
        },
        "pauze_tijd": {
          "type": "integer"
        }
      },
      "required": [
        "id",
        "pauze"
      ]
    }
  },
  "required": [
    "pauze"
  ]
}
```

- Resource: kassa
    - Voor het opvragen van de data voor de resource kassa maken we gebruik van volgende GET-codes:
        - GET /kassa
        - GET /pauze/{id} (V.b. id = kassa_1)
        - GET /pauze/{id}/locatie
        - GET /pauze/{id}/locatie/kassa_rapport_id
    - Indien men een kassa wil toevoegen maken we gebruik van volgende POST-code. Indien men een nieuwe kassa toevoegt wordt er automatisch een nieuw kassa_rapport_id aangemaakt, dat automatisch wordt toegevoegd aan de resource kassa_rappot.
        - POST /kassa
        - V.b. body: {“id”: “kassa_1”;
            “locatie”: “Cafetaria”;}
    - Schema:

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "kassa": {
      "type": "object",
      "properties": {
        "id": {
          "type": "string"
        },
        "locatie": {
          "type": "string"
        },
        "kassa_rapport_id": {
          "type": "integer"
        }
      },
      "required": [
        "id",
        "locatie",
        "kassa_rapport"
      ]
    }
  },
  "required": [
    "kassa"
  ]
}
```

- Resource: kassa_rapport
    - Voor het opvragen van de data voor de resource kassa maken we gebruik van volgende GET-codes:
        - GET /kassa_rapport
        - GET /kassa_rapport/kassa_rapport_id
        - GET /kassa_rapport/kassa_rapport_id/datum
        - GET /kassa_rapport/kassa_rapport_id/datum/verkoop_totaal
    - Men kan geen data toevoegen aan de resource kassa_rapport. Deze data wordt automatisch opgevangen en gecapteerd bij de dagelijkse afsluiting van de kassa's. De resource kassa_rapport bevat de dagafsluiting van de kassa, met andere worden het totaal aantal verkopen en de datum van het rapport.
    - Schema:

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "kassa_rapport": {
      "type": "object",
      "properties": {
        "kassa_rapport_id": {
          "type": "integer"
        },
           "datum": {
          "type": "integer",

        "verkoop_totaal": {
          "type": "integer"
        }
      },
      "required": [
        "id",
        "tijd",
        "verkoop"
      ]
    }
  },
  "required": [
    "kassa_rapport"
  ]
}
```

- Resource: verkoop
    - Voor het opvragen van de data voor de resource verkoop maken we gebruik van volgende GET-codes:
        - GET /kassa
        - GET /kassa/kassa_id
        - GET /kassa /kassa_id/kassa_rapport
        - GET /kassa /kassa_id/kassa_rapport/kassa_rapport_id
        - GET /kassa /kassa_id/kassa_rapport/kassa_rapport_id/verkoop
        - GET /kassa /kassa_id/kassa_rapport/kassa_rapport_id/verkoop/verkoop_id
    - Ook voor de resource verkoop geldt de regel dat er hier geen data handmatig aan kan toegevoegd worden. Deze data wordt mee opgevangen en gecapteerd wanneer de kassa_rapport data wordt ingeladen. De resource verkoop bevat een gedetailleerde samenvatting van alle verkochte goederen die dag en het tijdstip waarop deze verkocht werden, alsook de betaalwijze van de klant.
    - Schema:

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "verkoop": {
      "type": "object",
      "properties": {
        "id": {
          "type": "integer"
        },
        "product": {
          "type": "string"
        },
        "aantal": {
          "type": "integer"
        },
        "kostprijs_product": {
          "type": "number"
        },
        "totaal_bedrag": {
          "type": "number"
        },
        "betaalwijze": {
          "type": "string"
        }
      },
      "required": [
        "id",
        "product",
        "aantal",
        "kostprijs_product",
        "totaal_bedrag",
        "betaalwijze"
      ]
```

```
    }
  },
  "required": [
    "verkoop"
  ]
}
```

Oorsprong data

Om te kunnen weten hoeveel studenten er op bepaalde momenten in de cafetaria zijn, alsook om te weten dat zij een snack aankopen in de cafetaria hebben we enkele resources en attributen nodig. Deze resources zullen hier verder uitgewerkt worden.

Als eerste moeten we kunnen bepalen waar onze studenten zich bevinden op bepaalde momenten, en of ze zich al dan niet in de cafetaria bevinden tijdens een pauze bv. Hiervoor gaan we gebruik maken van de resource “Bezetting”, deze resource doet een distinct_count van het aantal unieke device binnen een bepaalde tijdspanne. De getelde devices worden gefilterd op basis van het rssi, dat in de lokalen minder dan -70 moet zijn en in de cafetaria minder dan -110.
In dit geval zullen we specifiek kijken naar een tijdspanne van 15min.