Zuyd Hogeschool
BD01
Fysiek ontwerp

Groepsleden:
Dylan Giesberts
Rik Heijmann
Jordy Clignet

Opdrachtgever:
Maarten Vaessen

Begeleidende docent:
Marcel Schmitz

Document informatie

Versiebeheer

Vastleggen wanneer iemand aan het document heeft gewerkt en wat veranderd is.

Versie	Gewijzigd	Auteur	Datum
0.1	Initiële template aangeleverd	Allen	15-09-2021
1.0	Eerste uitgave	Rik Heijmann	27-01-2021

Distributielijst

Wanneer is welke versie uitgegeven/ingeleverd en door wie.

Versie	Omschrijving	Distributie	Datum
0.1	Delen templatedocument aan casusgroep	Samen	15-9-2021

1. Informatiepagina

Studenten: Dylan Giesberts, Rik Heijmann, Jordy Clignet.
Opleiding: Zuyd hogeschool, Bachelor ICT
Schoolbegeleider/procesbegeleider:
Periode: Blok 2 van studiejaar 2021-2022
Opdrachtgever: Zuyd Hogeschool en Politie Nederland
Datum en plaats: 15-9-2021, Heerlen

Inhoudsopgave

2. Afkortingen

Afkorting	Uitleg
MQTT	Protocol om op een stabiele, snelle en beveiligde manier databerichten naar een centraal opslagsysteem te brengen.
MQTT-broker	Het centrale opslagsysteem van MQTT. Functioneert als het ware als een internet forum. Waarop berichten geplaatst kunnen worden en uitgelezen kunnen worden.
MQTT-subscriber	Leest berichten uit de MQTT-broker en verwerkt deze naar behoren. Bijvoorbeeld het decoderen van de berichten of het uitvoeren van bepaalde acties op basis van de inhoud van de berichten.
MQTT-publisher	Plaatst berichten op de MQTT-broker.
MQTT-bericht	Bestaat uit de inhoud en een onderwerp. Dit onderwerp kan door de MQTT-subscriber gebruikt worden om aan te geven wat voor soort data wordt doorgestuurd en hoe deze moet worden behandeld.
Sensorkastje	Een eigen term voor het prototype. Deze is bestaande uit een behuizing, microcontrollers en meerdere sensoren.

Tabel 1: Lijst met afkortingen

3. Inleiding

Predictive maintenance is het idee om de toestand van in gebruik zijnde apparatuur te helpen bepalen (bv door sensoren) om in te schatten en te voorspellen wanneer onderhoud moet worden uitgevoerd. Vanuit het lectoraat DI is er de behoefte om met dit concept aan de slag te gaan. Predictive maintenance heeft de meeste toegevoegde waarde in een live setting, echter is een productie omgeving (bv een echte fabriekshal) mogelijk geen goede optie om onderzoek en ontwikkeling te doen. Vandaar dat zowel een fysieke en virtuele omgeving onderzocht moeten worden

3.1. Leeswijzer

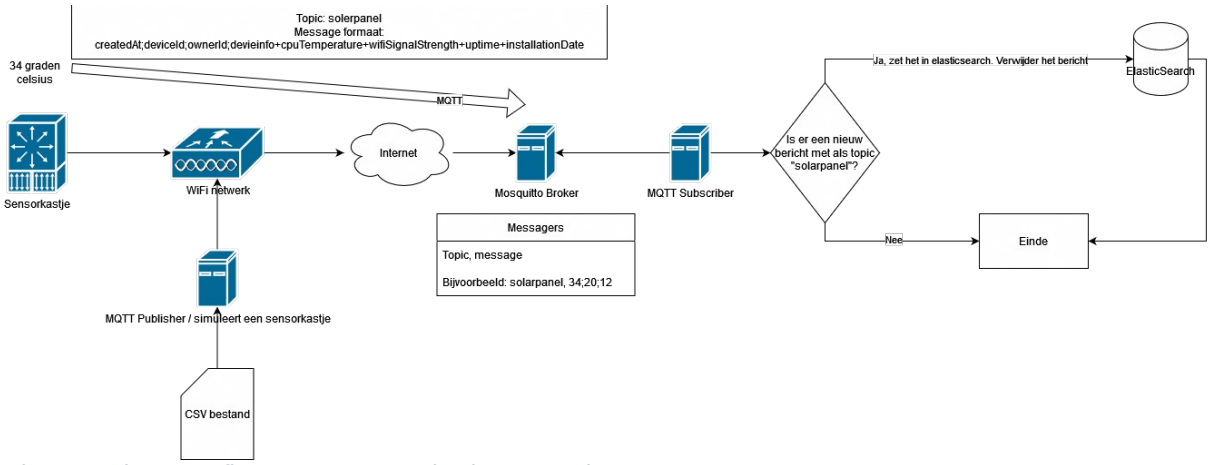
- **Functioneel ontwerp**
  Beschrijft in algemene termen hoe dat het systeem er uit gaat zien en zal functioneren.

- **Technische ontwerp**
  Gaat ten opzichte van het functioneel ontwerp dieper in op de materie. Beschrijft letterlijk hoe dat het systeem is ontworpen en met welke achterliggende gedachten.

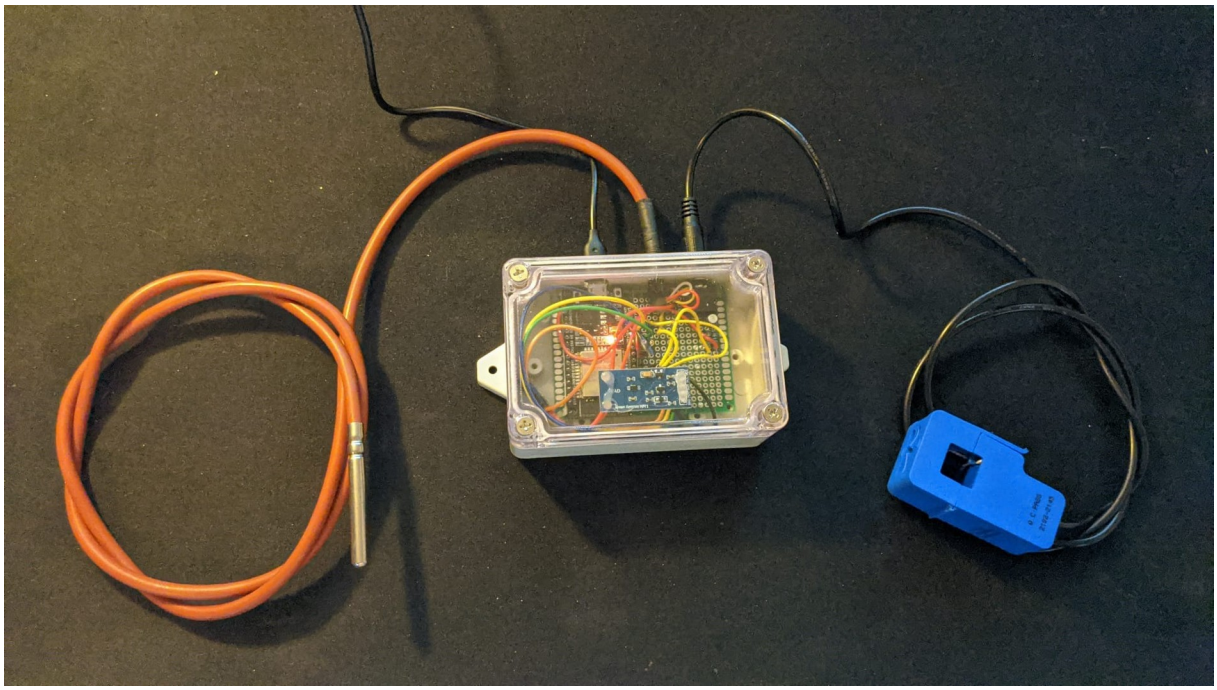
3.2. Context

Het onderzoeken van de mogelijkheden voor een simulatie omgeving voor predictive maintenance, die door het lectoraat Data Intelligence gebruikt kan worden voor onderzoek en ontwikkeling.

4. Functioneel ontwerp



Figuur 1: Algemene flow van meten tot databaserecord.



Figuur 2: Temperatuursensor links (bruine kabel met metale punt), sensorkastje met lichtsensor in het midden (lichtsensor zit op de blauweprintplaat) en de stroomsensor aan de rechterkant.

4.1. Het sensorkastje

Het doel van dit systeem is om de stroom die uitmondt uit een zonnepaneel (als ampere), de temperatuur rondom een zonnepaneel (in graden celsius), de dag-nachtcyclus te meten en de mate van licht door de zon (als lux). Deze waarden worden gemeten middels een stroomsensor (YHDC SCT013-050), een lichtsensor (BH1750) en een temperatuursensor (DS18B20, waterdichte variant).

De stroomsensor klemt zich om de output-kabel van het zonnepaneel. De temperatuursensor dient op een schaduwvrije plek onder het zonnepaneel te worden geplaatst. De lichtsensor is geïntegreerd in het centrale sensorkastje, samen met de microcrontroller (ESP32) die alle gegevens verzameld en doorstuurt.

De microcontroller is gebasseerd op de Espressif ESP32. Dit microcontroller platform zal middels WiFi verbonden zijn aan het internet. De ESP32 is voor dit domein geen kant-en-klare-oplossing, in principe voor geen enkel domein. Deze dient te worden geprogrammeerd. De ESP32 biedt simpelweg een microcontroller, een ingebouwde WiFi-modem en een hoop GPIO-poorten (General Purpose Input Output). Deze poorten zijn smalle pinnen aan de zijkanten van de ESP32. Deze GPIO-poorten kunnen stroom doorzetten (als output) of opnemen (als input). Met het opnemen bedoel ik de spanning (volt) en de stroom (ampere) meten. Verwar dit niet met de stroomsensor. De gemeten waarden via de GPIO-poorten zijn zeer laag. De stroomsensor doet als het ware niets anders dan waarden meten en die waarde zodanig in verhouding omzetten zodat deze door de GPIO-poorten gelezen kunnen worden. In de programmering van de microcontroller wordt deze verhouding weer omgezet.

4.2. Van sensorkastje tot database

Het sensorkastje is verbonden met het WiFi-netwerk van de eigenaar van de zonnepanelen. In gevallen waarbij er geen beschikbaar WiFi-netwerk aanwezig is, kan er gebruik worden gemaakt van een GPRS-modem of een LoRa-modem. Dit valt buiten de scope van dit project.

Zodra het sensorkastje is verbonden worden er verbinding gemaakt met de MQTT-broker. Deze MQTT-broker werkt als het ware als een internet forum. Gebruikers kunnen er berichtjes opzetten en andere gebruikers kunnen deze lezen. Deze MQTT-broker ontvangt de metingen van de sensorkastjes als bericht, met daarbij een standaard onderwerp “solarpanelanalytics”. Dit onderwerp is zeer belangrijk. Op basis van dit onderwerp kan de MQTT subscriber berichten aflezen.

De MQTT subscriber is de schakel tussen de MQTT-broker en de database. Zodra er een bericht met de titel “solarpanelanalytics” binnenkomt leest de subscriber dit uit, converteerd het bericht naar een formaat dat de database gebruikt (dit past de data niet aan) en stuurt de data naar de database.

4.1. Van database tot data scientist

De gebruikte databaseserversoftware heet OpenSearch (gebasseerd op ElasticSearch). Deze database is erg goed geïntegreerd met een van de meeste belangrijke data science tools: Python. De speciale python library genaamd “eland”[1] laat haar gebruikers de data uit de OpenSearch-database virtueel openen. Virtueel als in enkel de data die op dat moment nodig is wordt uit de database gehaald. Dit vermindert de druk op de database en geeft een snellere response uit. Zodra de datascientist de data heeft opgehaald, kan deze direct worden doorgezet naar diverse machine learning en statistische analyse Python libraries.

[1] https://www.elastic.co/guide/en/elasticsearch/client/eland/current/overview.html

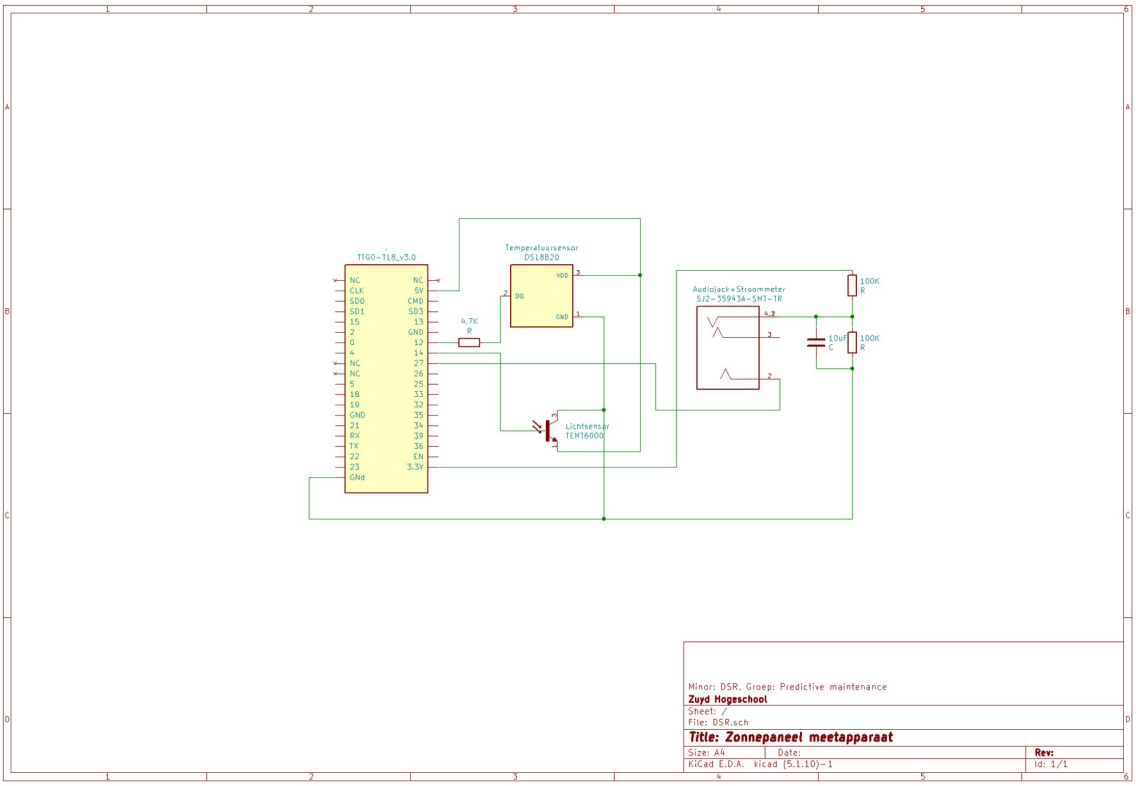
```
>>> import eland as ed
>>> from elasticsearch import Elasticsearch

# First instantiate an 'Elasticsearch' instance connected to Elastic Cloud
>>> es = Elasticsearch(cloud_id="...", api_key=("...", "..."))

# then wrap the client in an Eland DataFrame:
>>> df = ed.DataFrame(es, es_index_pattern="flights")
>>> df.head(5)
       AvgTicketPrice  Cancelled  ... dayOfWeek           timestamp
0          841.265642      False  ...         0 2018-01-01 00:00:00
1          882.982662      False  ...         0 2018-01-01 18:27:00
2          190.636904      False  ...         0 2018-01-01 17:11:14
3          181.694216       True  ...         0 2018-01-01 10:33:28
4          730.041778      False  ...         0 2018-01-01 05:13:00
[5 rows x 27 columns]
```

Figuur 3: Het ophalen van voorbeelddata uit ElasticSearch middels Eland.

5. Technische ontwerp

5.1. Sensorkastje

5.1.1. Elektrischschema



Figuur 4: Dit schema toont hoe dat de individuele componenten met elkaar zijn verbonden.

5.1.2. Microcontroller firmware

Voor het programmeren van de microcontroller wordt gebruik gemaakt van de Arduino IDE suite (versie 1.8.16). Deze set aan standaard libraries zal grotendeels de firmware van de ESP32 bepalen. De firmware is gebasseerd op Super-ESP [2]. Super-ESP is een door Rik Heijmann ontworpen firmware voor de ESP32. Deze zorgt standaard voor een betrouwbare WiFi- en MQTT-verbinding en is erg modulair van aard. Gebruikers van de sensor zouden zelf eigen configuraties kunnen wegschrijven zonder dat hiervoor de source code bekend hoeft te worden gemaakt. Dit bespaart een hoop programmeer werk.

5.1.2.1. Arduino Libraries

Om de diverse componenten aan te sluiten en over het algemeen een stabiele microcontroller te krijgen is er gebruik gemaakt van diverse Arduino Libraries. Deze libraries bieden voor geprogrammeerde oplossingen voor diverse situaties. Bijvoorbeeld wanneer er een temperatuur sensor gekoppeld dient te worden. Alle berekeningen en controles worden dan voor de programmeur al gedaan.

#	Naam	Versie	Waarvoor wordt	Auteur	Bron

[2] https://github.com/RHeijmann/Super-ESP

			het gebruikt?		
1	*esp32*	*2.0.2*	*Be*	*Espressif Systems*	*Arduino IDE Library Manager*
2	*ArduinoJson*	*6.18.5*	*Het opslaan en openen van de vaste configuratie.*	*Benoit Blanchon*	*Arduino IDE Library Manager*
3	*BH1750*	*1.2.0*	*Het meten van licht, middels de BH1750 sensor.*	*Christoper Laws*	*Arduino IDE Library Manager*
4	*DS18B20*	*1.0*	*Het meten van temperatuur, middels de DS18B20 sensor.*	*Matthias Munk Hansen*	*Arduino IDE Library Manager*
5	*EmonLib*	*1.1.0*	*Het meten van stroom, middels de SCT013-050 sensor.*	*OpenEnergyMonitor*	*Arduino IDE Library Manager*
6	*OneWire*	*2.3.6*	*Ter ondersteuning van DallasTemperature library.*	*Paul Stoffregen*	*Arduino IDE Library Manager*
7	*PubSubClient*	*2.8.0*	*Voor het verzenden van MQTT-berichten*	*Nick O'Leary*	*Arduino IDE Library Manager*

5.1.2.2. Sensor waarden converteren.

5.1.2.2.1. Temperatuursensor

De waarden van de DS18B20-temperatuursensor worden eerst bekneld met een 4,7K ohm weestandje (een vereiste van de library) en daarna omgezet door de DS18B20-library. De DS18B20 zend zijn meetwaarden als analoog signaal. Dit signaal wordt door de ESP32 geconverteerd naar een digitaal signaal. Waarna de library deze omzet naar een temperatuur. In deze situatie weet de library de conversie van een digitaal naar graden celsius al. Hierbij wordt ervan uitgegaan dat de softwarematige de gemeten waarden al accuraat zijn. Het kan nog steeds zo zijn dat de temperatuur sensor zelve hierbij defect is.

5.1.2.2.1. Lichtsensor

De BH1750 lichtsensor is geïntegreerd in de GY-30 adapter. Deze adapter converteerd de meetwaarde naar accurate hoeveelheden lux. Zodat er door de BH1750-library haast geen verdere conversie nodig is.

5.1.2.2.1. Stroomsensor

De SCT013-050 stroomsensor is aangesloten op het sensorkastje middels een audiojack. Hierbij wordt de aarde en de stroom bekneld met een 100K ohm weerstand en vanuit de aarde een parallele capacitator van 10 uF (deze vult zich en zal wanneer het vol zit na enkele milliseconden de volledig stroom nog eens doorsturen).

De firmware van dit apparaat is te vinden op:
https://github.com/RHeijmann/Super-ESP/tree/main/Examples/Solarpanel.

5.2. MQTT Broker

De MQTT Broker is de tijdelijke opslag waartoe de sensorkastjes hun meetwaarden verzenden. De MQTT broker is gebasseerd op Eclipse Mosquitto (versie 20.14, docker image: eclipse-mosquitto:openssl). De configuratie van de broker is vrij barebones.
Er zijn niet veel aanpassingen uitgevoerd enkel het toevoegen van diverse extra gebruikers. Daarna wordt deze configuratie uitgevoerd door een Docker container.

5.2.1. Docker

Docker wordt geconfigureerd middels docker-compose versie 3. Ten opzichte van andere methodieken om docker te configureren maakt docker-compose enkel gebruik van voor geïnstalleerde images. Waarbij het configuratie werk beperkt is tot het leggen van netwerken tussen containers en het instellen van container environment variabelen.

De mosquitto container zal automatisch het meest recente image met openssl bevatten. Door de “expose”-optie zal Docker automatisch proberen om te port forwarden middels UPnP.

De “/mosquitto/”-map bevind zich in het image. Deze wordt gekoppeld aan een lokale-map. Zodat de inhoud van de “/mosquitto/”-map niet net als alle andere bestanden in het image verloren gaan bij afsluiten.

```
mosquitto:
  container_name: mosquitto
  image: eclipse-mosquitto:openssl
  ports:
    - 1883:1883
    - 9001:9001
  expose:
    - "1883"
    - "9001"
  volumes:
    - ./Volumes/Mosquitto:/mosquitto/:rw
  networks:
    - pm-net
```

5.2.1. Firewall regels

#	Afkomst	Bestemming	Afhandeling	
1	*:1883	DOCKER PC	Toestaan	MQTT over TCP
2	*:9001	DOCKER PC	Toestaan	MQTT over

				websockets

5.1. Database

De op OpenSearch 1.2.2 gebasseerde database wordt gevuld door de MQTT Subscriber. De data komt vervolgens terecht in de “solarpanelanalytics”-index. Deze index heeft de volgende structuur:

5.1.1. Docker

Tip! Het uitvoeren van OpenSearch onder Windows-Subsystem-for-Linux versie 2 kan vrij lastig zijn. Voordat de OpenSearch docker container wordt moet dit commando uitgevoerd worden: *wsl -d docker-desktop sysctl -w vm.max_map_count=262144*

Docker wordt geconfigureerd middels docker-compose versie 3. Ten opzichte van andere methodieken om docker te configureren maakt docker-compose enkel gebruik van voor geïnstalleerde images. Waarbij het configuratie werk beperkt is tot het leggen van netwerken tussen containers en het instellen van container environment variabelen.

De “/usr/share/opensearch/data”-map bevind zich in het OpenSearch image. Deze wordt gekoppeld aan een lokale-map. Zodat de inhoud van deze-map niet net als alle andere bestanden in het image verloren gaan bij afsluiten.

OpenSearch dashboards slaat haar configuratie op in OpenSearch. Zo heeft die image geen koppeling nodig tot het lokale bestandssysteem.

Als tweede visualisatie tool is er Grafana. Deze tool is meer gericht op het inzien van data in realtime. Tenopzichte van OpenSearch Dashboards dient Grafana eigen configuratie op te slaan in de “/var/lib/grafana”-map.

```
services:
  opensearch-node1:
    container_name: opensearch-node1
    image: opensearchproject/opensearch:latest
    restart: unless-stopped
    mem_limit: 1024m
    environment:
      - cluster.name=opensearch-cluster
      - node.name=opensearch-node1
      - discovery.seed_hosts=opensearch-node1
      - cluster.initial_master_nodes=opensearch-node1
      - bootstrap.memory_lock=true # along with the memlock settings
below, disables swapping
      - "OPENSEARCH_JAVA_OPTS=-Xms512m -Xmx512m" # minimum and maximum
Java heap size, recommend setting both to 50% of system RAM
      - network.host=0.0.0.0 # required if not using the demo security
configuration
    ulimits:
```

```
      memlock:
        soft: -1
        hard: -1
      nofile:
        soft: 65536 # maximum number of open files for the OpenSearch
user, set to at least 65536 on modern systems
        hard: 65536
    volumes:
      - ./Volumes/OpenSearch-Node1:/usr/share/opensearch/data:rw
    ports:
      - 9200:9200
      - 9600:9600 # required for Performance Analyzer
    expose:
      - "9200"
      - "9600"
    networks:
      - pm-net
  opensearch-dashboards:
    container_name: opensearch-dashboards
    image: opensearchproject/opensearch-dashboards:latest
    restart: unless-stopped
    mem_limit: 1024m
    ports:
      - 5601:5601
    expose:
      - "5601"
    environment:
      OPENSEARCH_HOSTS: https://opensearch-node1:9200
    networks:
      - pm-net
  grafana:
    container_name: grafana
    image: grafana/grafana:latest
    restart: unless-stopped
    mem_limit: 512m
    ports:
      - 3000:3000
    expose:
      - "3000"
    volumes:
      - ./Volumes/Grafana:/var/lib/grafana:rw
    networks:
      - pm-net
    environment:
      - GF_SECURITY_ADMIN_PASSWORD=0p3ns34rch
      - GF_USERS_ALLOW_SIGN_UP=false
networks:
```

```
  pm-net:
```

5.1.1. Firewall regels

#	Afkomst	Bestemming	Afhandeling	
1	*:9200	DOCKER PC	Alleen toestaan indien vertrouwde bron.	OpenSearch Web API
2	*:9600	DOCKER PC	Alleen toestaan indien vertrouwde bron.	OpenSearch Performance Monitor
3	*:6501	DOCKER PC	Toestaan	OpenSearch Dashboards web interface.
4	*:3000	DOCKER PC	Toestaan	Grafana web interface.

6. Bijlagen

6.1. Vergelijking microcontrollers:

Zie bijlage “Microcontroller vergelijking.xlsx”.

6.2. Vergelijking databases:

Zie bijlage “Database vergelijking.xlsx”.

6.3. Sensoren vergelijking:

Zie bijlage “Sensoren vergelijking.xlsx”

6.4. Eisen:

Zie bijlage “BD01_Predictive_Maintenance_Requirements.docx”

6.1. Stroomschema:

Zie bijlage “Fysiek > KiCad > DSR.pro”. Deze is te openen met KiCad versie 5.1.10.