**Documentatie**

Inhoudsopgave

[Hardware | sensoren & PCB 3](#_Toc149206740)

[Sensoren | SEN55 & SPG 41 3](#_Toc149206741)

[SEN55: 3](#_Toc149206742)

[SPG 41: 4](#_Toc149206743)

[PCB 5](#_Toc149206744)

[1ste versie 5](#_Toc149206745)

[TheThingsNetwork | LoRaWAN 8](#_Toc149206746)

[TheThingsNetwork 8](#_Toc149206747)

[Wat is de TheThingsNetwork: 8](#_Toc149206748)

[Hoe werkt TTN: 9](#_Toc149206749)

[Stappenplan: 10](#_Toc149206750)

[LoRaWAN 12](#_Toc149206751)

[Wat is LoRaWAN: 12](#_Toc149206752)

[Specificaties: 12](#_Toc149206753)

[Opzetten van de server (back-end) 13](#_Toc149206754)

[Wat hebben we nodig: 13](#_Toc149206755)

[Docker: 13](#_Toc149206756)

[Docker netwerk 13](#_Toc149206757)

[Opzetten 13](#_Toc149206758)

[Node-Red 16](#_Toc149206759)

[Wat is Node-Red: 16](#_Toc149206760)

[Waarom gebruiken wij Node-Red: 16](#_Toc149206761)

[Schema: 17](#_Toc149206762)

[Blokken: 17](#_Toc149206763)

[Database | Influxdb & docker 21](#_Toc149206764)

[Frontend | Grafana 22](#_Toc149206765)

[Wat is Grafana: 22](#_Toc149206766)

[Bronnen 23](#_Toc149206767)

# Hardware | sensoren & PCB

## Sensoren | SEN55 & SPG 41

### SEN55:

Afbeelding met tekst, schermopname, Lettertype, nummer

Automatisch gegenereerde beschrijvingAfbeelding met tekst, diagram, Plan, Technische tekening

Automatisch gegenereerde beschrijvingDe SEN55 is een eenvoudig, alles-in-één platform voor sensoroplossingen voor het nauwkeurig meten van verschillende omgevingsparameters, zoals fijnstof, vluchtige organische stoffen (VOC's), oxiderende gassen, zoals stikstofoxiden (NOx), en als vochtigheid en temperatuur.

Afbeelding met tekst, diagram, schermopname, Lettertype

Automatisch gegenereerde beschrijving

Over het algemeen kunnen de meeteenheden als volgt zijn:

* Fijnstof: Gemeten in microgram per kubieke meter (µg/m³) voor de concentratie van fijnstofdeeltjes in de lucht.
* VOC's: Gemeten in ppb (parts per billion) of ppm (parts per million) voor de concentratie van vluchtige organische stoffen in de lucht.
* NOx: Gemeten in ppb of ppm voor de concentratie van stikstofoxiden (NOx) in de lucht.
* Vochtigheid: Gemeten in procent (%), wat de relatieve luchtvochtigheid weergeeft.
* Temperatuur: Gemeten in graden Celsius (°C) voor de temperatuur in de omgeving.

De exacte eenheden en meetbereiken kunnen variëren afhankelijk van de specifieke sensoren en configuraties die worden gebruikt binnen het SEN55-platform van Sensirion.

### SPG 41:

De SGP41 is een VOC- en NOx-sensor van Sensirion. Deze sensor is zeer geschikt voor het voortdurend monitoren van de VOC- en NOx-situatie, inclusief potentieel schadelijke gebeurtenissen die niet door mensen worden waargenomen.

De meeteenheid voor VOC kan in het algemeen worden uitgedrukt in parts per billion (ppb) of parts per million (ppm), afhankelijk van de concentratie van vluchtige organische stoffen. NOx-concentraties worden meestal uitgedrukt in ppb of ppm voor gassen zoals stikstofdioxide (NO2) en stikstofmonoxide (NO). De precieze meeteenheden kunnen variëren afhankelijk van de specifieke configuratie en toepassing van de sensor.

Afbeelding met tekst, diagram, Plan, schermopname

Automatisch gegenereerde beschrijving Afbeelding met tekst, diagram, Technische tekening, Plan

Automatisch gegenereerde beschrijving

## PCB

### 1ste versie

|  |  |
| --- | --- |
| Front | A green circuit board with many small white and gray objects  Description automatically generated |
| Back | A green circuit board with many small holes and circles  Description automatically generated |
| Isometrische projectie | A close-up of a computer chip  Description automatically generated  A close-up of a computer chip  Description automatically generated |

A diagram of a computer

Description automatically generated

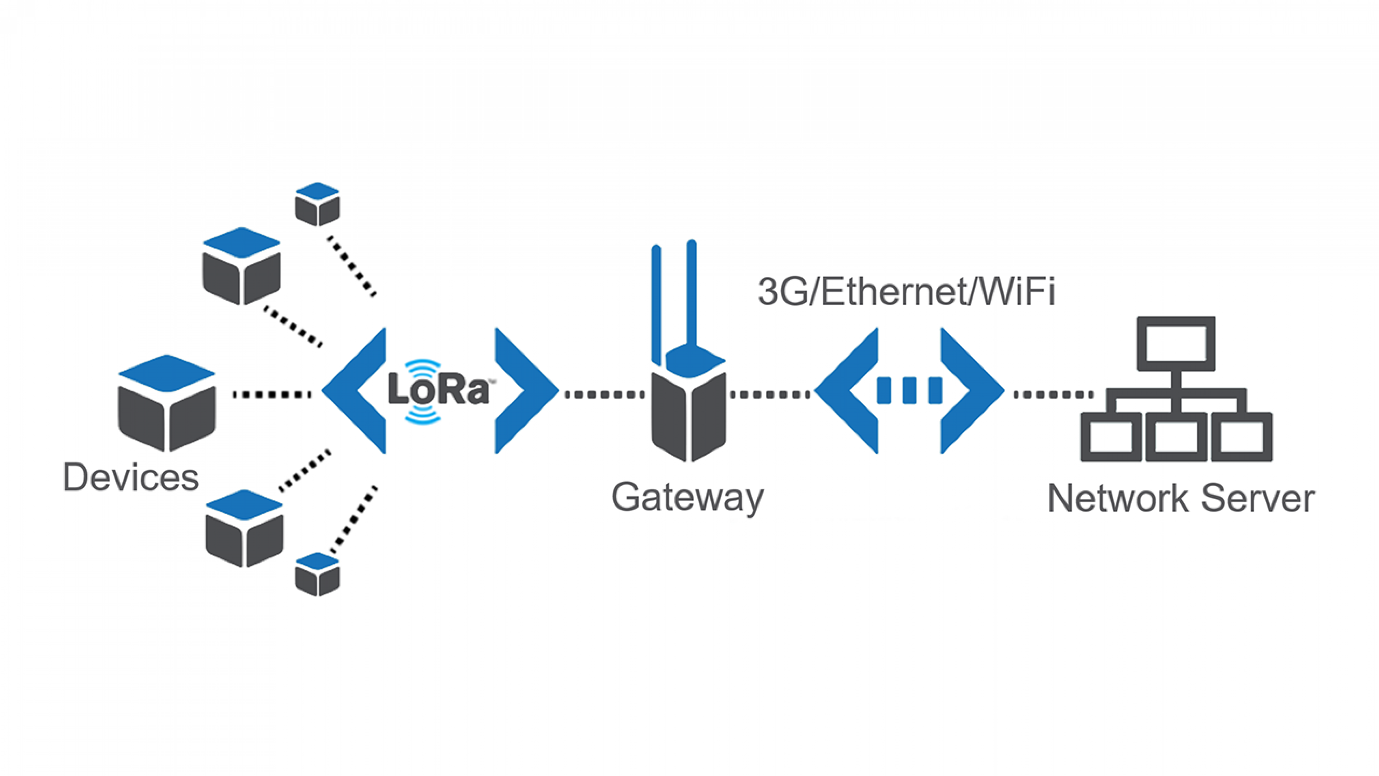


# TheThingsNetwork | LoRaWAN

## TheThingsNetwork

### Wat is de TheThingsNetwork:

TheThingsNetwork (TTN) is een open source en door de gemeenschap gedreven project dat tot doel heeft een wereldwijd, crowd-sourced en gedecentraliseerd Internet of Things (IoT)-netwerk te creëren. Het biedt de infrastructuur en tools die nodig zijn om een LoRaWAN (Long Range Wide Area Network) voor IoT-apparaten te bouwen en te exploiteren. TTN is gebaseerd op LoRaWAN-technologie en wordt onderhouden door vrijwilligers en gemeenschappen. Het stelt bedrijven en ontwikkelaars in staat om IoT-oplossingen te bouwen boven op dit netwerk.



Figuur Data overdracht sensor naar TTN

### Hoe werkt TTN:

Na het creëren van een end device en het genereren van de unieke sleutels en ID, volgt het volgende proces:

**End Device Communicatie**: Het end device, zoals een IoT-sensor of -apparaat, verzendt gegevens die het wil overdragen via het LoRaWAN-protocol.

**Data Verpakt**: De gegevens worden verpakt in LoRaWAN-pakketten, die de ID van het end device bevatten, evenals de beveiligingssleutels die nodig zijn voor de codering en beveiliging van de gegevens.

**Verzending naar de Gateway**: De LoRaWAN-pakketten worden via radiofrequentie verzonden naar een nabijgelegen gateway, die fungeert als een ontvangststation. Gateways kunnen overal in de omgeving zijn geïnstalleerd en kunnen de gegevens van meerdere end devices ontvangen.

**Gateway naar Netwerk Server**: De gateway ontvangt de pakketten, decodeert de gegevens en stuurt deze door naar een TTN-netwerkserver via een internetverbinding.

**Netwerkserver**: De TTN-netwerkserver ontvangt de gegevens van de gateway en is verantwoordelijk voor het routeren van de gegevens naar de juiste bestemming. Het verifieert ook de beveiligingssleutels om de integriteit en beveiliging van de gegevens te waarborgen.

**Applicatie Server**: De gegevens worden doorgestuurd naar een applicatieserver die specifiek is opgezet voor de toepassing of service waarmee de gegevens worden gebruikt. Deze server verwerkt en analyseert de gegevens en voert de nodige acties uit, zoals opslag, visualisatie of het triggeren van meldingen.

**Terugkerende Communicatie**: Als bidirectionele communicatie is vereist, kan de applicatieserver gegevens naar het end device sturen via dezelfde route, waarbij de server de gegevens naar de juiste gateway en uiteindelijk naar het end device routeert.

**Beheer en Monitoring**: TTN biedt beheer- en monitoringstools waarmee gebruikers de prestaties van hun netwerk kunnen volgen, apparaat registraties kunnen beheren en beveiligingsinstellingen kunnen configureren.

### Stappenplan:

1. Maak een applicatie aan.
2. Voeg een end device toe met volgende instellingen.
3. Genereer alle sleutels (keys).

Afbeelding met tekst, schermopname, document, nummer

Automatisch gegenereerde beschrijvingAfbeelding met tekst, schermopname, Lettertype, lijn

Automatisch gegenereerde beschrijving

Afbeelding met tekst, schermopname, Lettertype, nummer

Automatisch gegenereerde beschrijving

EXTRA! Bij ABP ga naar *general setting* < *Network layer* < *advanced MAC settings* < *Reset frame counters*

Dit zorgt ervoor dat dezelfde sensor herkend wordt nadat deze uit is getrokken en later terug gebruikt moet worden.

1. Voeg nu de nodige keys toe aan de credentials in de code voor de sensor.

|  |
| --- |
| /\* Credentials file \*/  #pragma once  // Only one of these settings must be defined  #define USE\_ABP  //#define USE\_OTAA  #ifdef USE\_ABP  // LoRaWAN NwkSKey, network session key  static const u1\_t PROGMEM NWKSKEY[16] = { *Voeg hier de NWSKEY toe* };  // LoRaWAN AppSKey, application session key  static const u1\_t PROGMEM APPSKEY[16] = { *Voeg hier de APPSKEY toe* };  // LoRaWAN end-device address (DevAddr)  // This has to be unique for every node  static const u4\_t DEVADDR = 0x*Voeg hier de DEVADDR toe*;  #endif |

1. Upload de code met de gewenste sensor code.
2. *Payload Formatter* < *Uplink* voor het decoderen van bytes. (Voorbeeld voor SPG41)

|  |
| --- |
| *SPG41*  function Decoder(bytes, port) {  var decoded = {};  decoded.VOC = (bytes[0] << 8) + bytes[1];  decoded.NOX= (bytes[2] << 8) + bytes[3];  return decoded;  } |

|  |
| --- |
| *SEN55*  function Decoder(bytes, port) {  var decoded = {};  decoded.pm1p0 = (bytes[0] << 8) + bytes[1] /100;  decoded.pm2p5= (bytes[2] << 8) + bytes[3] /100;  decoded.pm4p0= (bytes[4] << 8) + bytes[5] /100;  decoded.pm10p0= (bytes[6] << 8) + bytes[7] /100;  decoded.hum= (bytes[8] << 8) + bytes[9] /100;  decoded.temp= (((bytes[10] << 8) + bytes[11]) /100) -10;  decoded.voc= (bytes[12] << 8) + bytes[13] /100;  decoded.nox= (bytes[14] << 8) + bytes[15] /100;  return decoded;  } |

## LoRaWAN

### Wat is LoRaWAN:

LoRaWAN is een draadloos communicatieprotocol dat is ontworpen voor lange afstandsconnectiviteit van batterij gevoede IoT-apparaten. Het biedt een groot bereik, laag energieverbruik en wordt veel gebruikt in toepassingen zoals smart cities en industriële automatisering. Het is een open standaard en kan zowel publieke als private netwerken omvatten.

### Specificaties:

De specificaties van LoRaWAN (Long Range Wide Area Network) omvatten de volgende belangrijke kenmerken:

* **Frequentiebanden**: LoRaWAN werkt op de 868 MHz-band in Europa.
* **Lange Afstand**: Het biedt connectiviteit over lange afstanden, variërend van enkele kilometers tot tientallen kilometers, afhankelijk van de omgeving en de frequentieband.
* **Laag Energieverbruik**: LoRaWAN is ontworpen voor energiezuinigheid, waardoor batterij gevoede IoT-apparaten maanden tot jaren op een enkele batterij kunnen werken.
* **Bij schaalbaarheid**: LoRaWAN-netwerken hebben meestal een gelaagde structuur die schaalbaarheid mogelijk maakt. Dit omvat eindapparaten (devices), gateways en een netwerkserver.
* **Adaptieve Modulatie**: LoRaWAN past de modulatie aan op basis van de afstand en omgevingsomstandigheden, waardoor het betrouwbaarheid en prestaties levert over variabele afstanden.
* **Bidirectionele Communicatie**: Het ondersteunt tweerichtingscommunicatie, waarbij apparaten gegevens kunnen verzenden en ontvangen via LoRaWAN-netwerken.
* **Beveiliging**: LoRaWAN bevat beveiligingsmaatregelen om gegevens te beschermen, waaronder encryptie van gegevens en apparaat authenticatie.
* **Open Standaard**: LoRaWAN is een open standaard, wat betekent dat het niet afhankelijk is van een specifieke fabrikant en interoperabiliteit tussen verschillende apparaten mogelijk maakt.
* **Join Procedure**: Apparaten moeten zich aanmelden bij een LoRaWAN-netwerk voordat ze gegevens kunnen verzenden of ontvangen. Dit wordt doorgaans gedaan via een join procedure.
* **Publieke en Private Netwerken**: LoRaWAN-netwerken kunnen zowel publiek als privaat zijn, afhankelijk van de toepassing en de beheerder.

# Opzetten van de server (back-end)

## Wat hebben we nodig:

Voor de back-end van ons project maken we gebruik van drie essentiële services:

**Node-Red**: Deze service wordt ingezet om gegevens op te halen van TheThingsNetwork, deze te verwerken en vervolgens in de 'InfluxDB' database te integreren.

**InfluxDB**: We gebruiken InfluxDB als onze database-oplossing. Hierin worden de gegevens opgeslagen die we verzamelen en verwerken met Node-Red.

**Grafana**: Om de verzamelde gegevens op een visueel aantrekkelijke manier te presenteren, vertrouwen we op Grafana. Met Grafana kunnen we de gegevens in onze Influx-database grafisch weergeven.

Deze combinatie van services stelt ons in staat om gegevens efficiënt te verzamelen, verwerken en presenteren in ons project

## Docker:

We hebben besloten om deze drie services binnen een Linux-omgeving op te zetten met behulp van containers. Het gebruik van containers biedt tal van voordelen, waaronder aanzienlijke flexibiliteit en een hoge mate van veiligheid. Containers zijn namelijk geïsoleerd van elkaar, wat betekent dat elke service in zijn eigen afzonderlijke omgeving opereert. Dit zorgt ervoor dat eventuele problemen of wijzigingen in één container de andere containers niet beïnvloeden. Bovendien maakt het opzetten van services in containers het proces van ontwikkelen, testen en implementeren aanzienlijk efficiënter en schaalbaarder.

## Docker netwerk

Om effectieve communicatie tussen de Docker-containers mogelijk te maken, is het essentieel dat ze zich binnen hetzelfde Docker-netwerk bevinden. Dit zorgt ervoor dat de containers naadloos met elkaar kunnen communiceren en gegevens kunnen uitwisselen. Door ze in hetzelfde netwerk te groeperen, kunnen we ervoor zorgen dat de services en applicaties die deze containers hosten, moeiteloos en efficiënt met elkaar kunnen samenwerken, waardoor de algehele systeemprestaties worden geoptimaliseerd.

Opzetten:

1. Zet een linux server op (bij voorkeur --> debian)
2. Installeer zowel docker als docker-compose
3. Maak een folder aan (met naam naar keuze) en cd in deze folder
4. Zet deze ‘docker-compose.yml’ op de linux server in de folder die je in vorige stap hebt gemaakt:

|  |
| --- |
| version: "3.8"  services:  node-red:  container\_name: node-red-air-quality-hoboken  image: nodered/node-red:latest  restart: unless-stopped  environment:  - TZ=Europe/Brussels  ports:  - '1880:1880'  networks:  - air-quality-hoboken  volumes:  -./node-red-data:/data  grafana:  container\_name: grafana-air-quality-hoboken  image: grafana/grafana:latest  restart: unless-stopped  environment:  - TZ=Europe/Brussels  ports:  - '3000:3000'  networks:  - air-quality-hoboken  volumes:  - ./grafana-data:/var/lib/grafana  influxdb:  container\_name: influx-db-air-quality-hoboken  image: influxdb:latest  networks:  - air-quality-hoboken  security\_opt:  - no-new-privileges:true  restart: unless-stopped  ports:  - '8086:8086'  volumes:  - ./influxdb2-data/config:/etc/influxdb2  - ./influxdb2-data/appdata:/var/lib/influxdb2  networks:  air-quality-hoboken: |

1. Bouw de docker-containers met behulp van docker-compose: zorgt dat je in de folder bent waar de docker-compose file zich bevindt en voer het volgende commando uit:

|  |
| --- |
| docker-compose up -d |

1. Bekijk het docker ip-address van de influxdb aangezien je deze nodig gaat hebben voor de connectie met Node-Red en Grafana:

|  |
| --- |
| docker inspect influx-db-air-quality-hoboken |





# Node-Red

### Wat is Node-Red:

Node-RED is een visuele programmeeromgeving waarmee je Internet of Things (IoT) en automatiseringstaken kunt maken en beheren. Het is gebaseerd op knooppunten die je kunt slepen en neerzetten om workflows te bouwen. Deze knooppunten vertegenwoordigen verschillende functies en apparaten en kunnen worden verbonden om gegevens te sturen, verwerken en integreren.

Node-RED is populair vanwege zijn gebruiksvriendelijke interface en flexibiliteit, waardoor het ideaal is voor taken zoals gegevensverwerking, het aansturen van hardware, en het automatiseren van taken in huis of op het werk. Het wordt vaak gebruikt in combinatie met Raspberry Pi en andere IoT-apparaten. Het biedt ondersteuning voor tal van protocollen en services, waardoor je gegevens van verschillende bronnen kunt verzamelen en ermee kunt interageren.

In essentie is Node-RED een tool waarmee je eenvoudig IoT- en automatiserings workflows kunt ontwerpen zonder uitgebreide programmeervaardigheden te hebben.

### Waarom gebruiken wij Node-Red:

Node-RED is een uitstekende keuze voor het stroomlijnen van het proces om gegevens van TheThingsNetwork via MQTT binnen te halen en specifieke waarden uit de JSON-gegevens te filteren voordat deze worden opgeslagen in onze database, INFLUXDB. Node-RED biedt een intuïtieve visuele interface waarmee we gemakkelijk de MQTT-gegevensstroom kunnen vastleggen, de JSON-berichten kunnen analyseren en alleen de relevante informatie kunnen selecteren die we willen bewaren. Het bespaart ons tijd en vereenvoudigt het gegevensverwerkingsproces, waardoor we gegevens efficiënt kunnen opslaan en analyseren in INFLUXDB voor verdere inzichten en acties.

### Schema:

A diagram of a set of text

Description automatically generated

### Blokken:

Instellingen MQTT blok à Bij het aanmaken van een MQTT integration een API key genereren en deze opslaan (deze ga je later nodig hebben als wachtwoord in Node-Red).

Afbeelding met tekst, schermopname, software, Computerpictogram

Automatisch gegenereerde beschrijving

MQTT node:

* Server: Hier vul je de naam of het IP-adres van de MQTT-broker in, samen met de poort (meestal 1883).
* Action:
  + Subscribe to Single Topic (Abonneren op enkelvoudig topic): Hiermee abonneer je de MQTT-node op een specifiek MQTT-topic dat je vooraf instelt.
  + Dynamic Subscription (Dynamisch abonnement): Hiermee kun je het MQTT-topic dynamisch instellen op basis van informatie in het inkomende bericht. Dit maakt je abonnement flexibel en afhankelijk van de gegevens die je ontvangt.
* Topic: Dit is een tekstlabel waarmee je berichten kunt categoriseren. Abonneer je op het gewenste MQTT-topic voor de juiste data te krijgen.
* QoS: Kies de gewenste kwaliteit van service voor berichten (0, 1 of 2).

A screenshot of a computer

Description automatically generated

Security: Vul de juiste username en wachtwoord in. Dit is de username en wachtwoord (api key) die je in het begin hebt aangemaakt in TTN.

A screenshot of a computer

Description automatically generated

JSON node:

We plaatsen de JSON node voor de inkomende berichten naar een JSON formaat te brengen.

A screenshot of a computer

Description automatically generated

Change node:

Bij deze node zullen we de ontvangen payload omzetten naar een zeer specifiek onderdeel (alleen het gene wat wij willen) van de payload en deze vervolgens doorsturen.

A screenshot of a computer

Description automatically generated

Debug node:

In deze node gaan we de data weergeven in de console.

A screenshot of a computer

Description automatically generated

Influxdb out node:

A screenshot of a computer

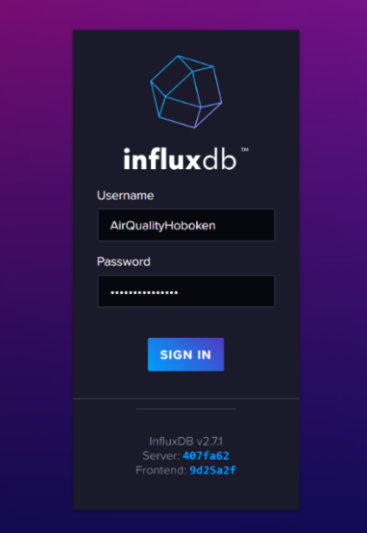
Description automatically generated

# Database | Influxdb & docker

Als database voor ons project gebruiken we InfluxDB. InfluxDB is een tijdreeksdatabase die we kunnen gebruiken om IoT sensordata op te slaan. We kunnen InfluxDB starten in een docker container met het commando: docker run -p 8086:8086 -v myInfluxVolume:/var/lib/influxdb2 influxdb:latest

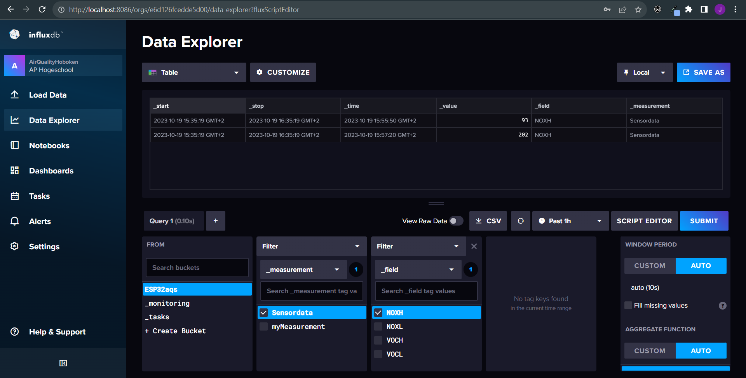
We kunnen daarna de database beheren en data bekijken via de ingebouwde web interface. We krijgen toegang via de browser op <http://localhost:8086/>.

In deze web interface moeten we vervolgens een nieuwe gebruiker en wachtwoord aanmaken. We kunnen dan een bucket aanmaken. Een bucket kan je vergelijken met de database waarin onze data zal worden opgeslagen. Wij maken de bucket “*ESP32aqs*” aan. We krijgen vervolgens ook een API-token te zien. Het is belangrijk dat je deze bewaart want je krijgt deze maar één keer te zien. Als we dit gedaan hebben kunnen we vervolgens inloggen in de database met de juist gemaakte gebruiker.



Als we ingelogd zijn kunnen we in de interface data toevoegen, buckets aanmaken en de data van onze buckets visualiseren.

Een "**bucket**" in InfluxDB is een container waarin tijdreeksgegevens worden opgeslagen. Buckets kunnen worden geconfigureerd met instellingen zoals hoe lang gegevens worden bewaard (tijdretentiebeleid) en welke verwerking erop wordt toegepast. Ze helpen bij het georganiseerd opslaan en beheren van gegevens.



# Frontend | Grafana

### Wat is Grafana:

Grafana is een open-source datavisualisatie- en monitoringplatform dat wordt gebruikt voor het analyseren en weergeven van gegevens uit verschillende bronnen. Het stelt gebruikers in staat om grafieken, diagrammen en dashboards te maken om inzicht te krijgen in gegevens, zoals systeemprestaties, netwerkstatistieken, sensordata en meer. Grafana kan worden geïntegreerd met verschillende gegevensbronnen, zoals databases, Cloud services en monitoringstools, waardoor het een krachtig instrument is voor het bewaken en analyseren van gegevens in real-time. Wij gaan het gebruiken om gegevens te visualiseren en besluitvorming te ondersteunen.

# Bronnen

Algemeen project: <https://github.com/JokerIsMyBae/AirQualityHoboken>

SEN55 Doc: <https://www.sensirion.com/media/documents/6791EFA0/62A1F68F/Sensirion_Datasheet_Environmental_Node_SEN5x.pdf>

SPG41 Doc: <https://sensirion.com/media/documents/5FE8673C/61E96F50/Sensirion_Gas_Sensors_Datasheet_SGP41.pdf>