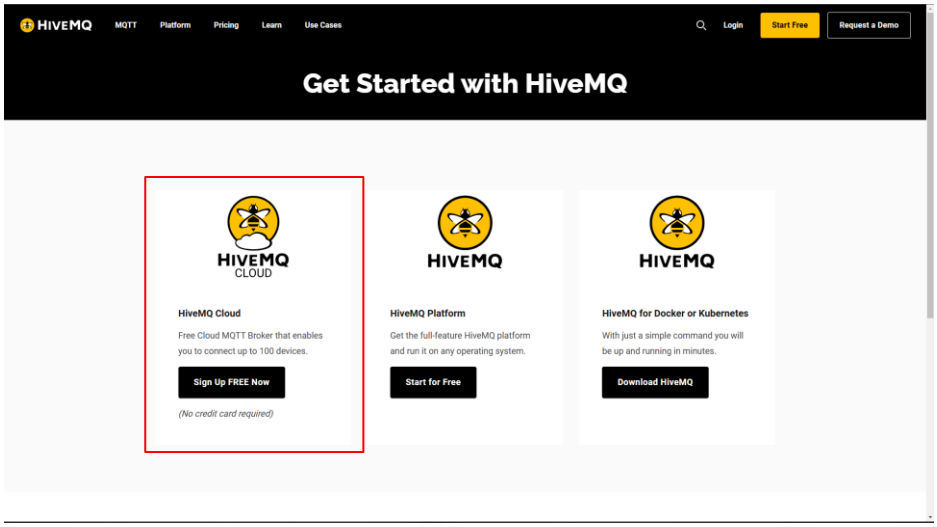
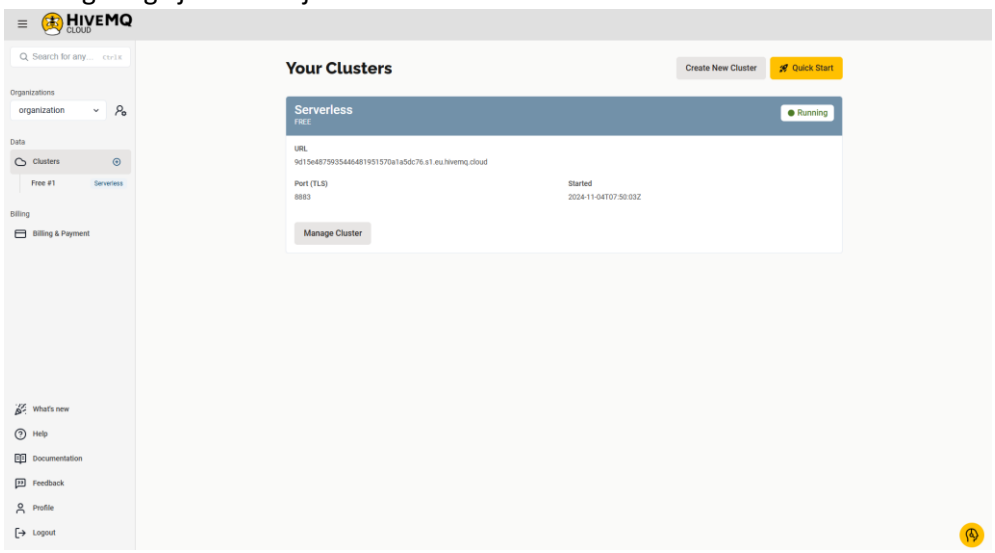
UNS (Zonder Sparkplug B)

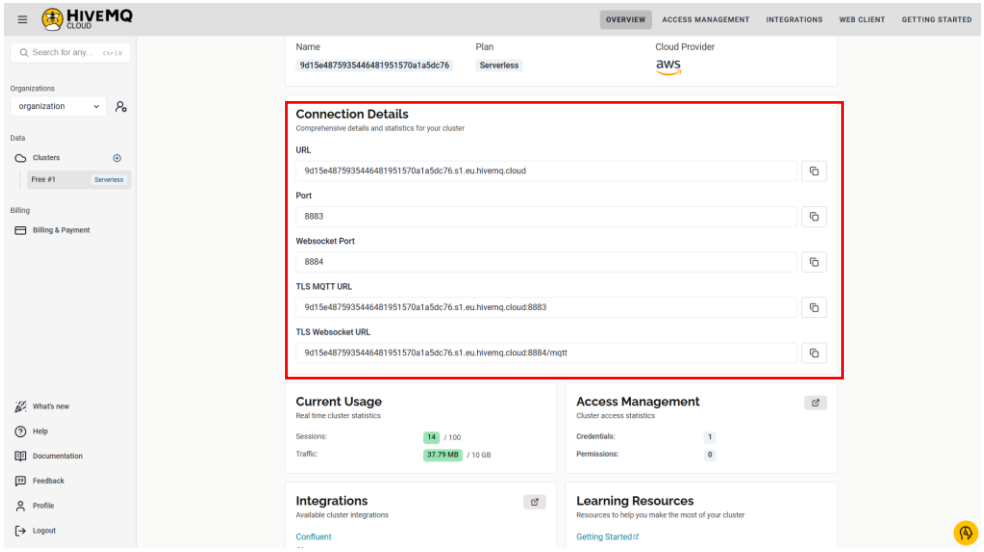
- **Hivemq Cloud**



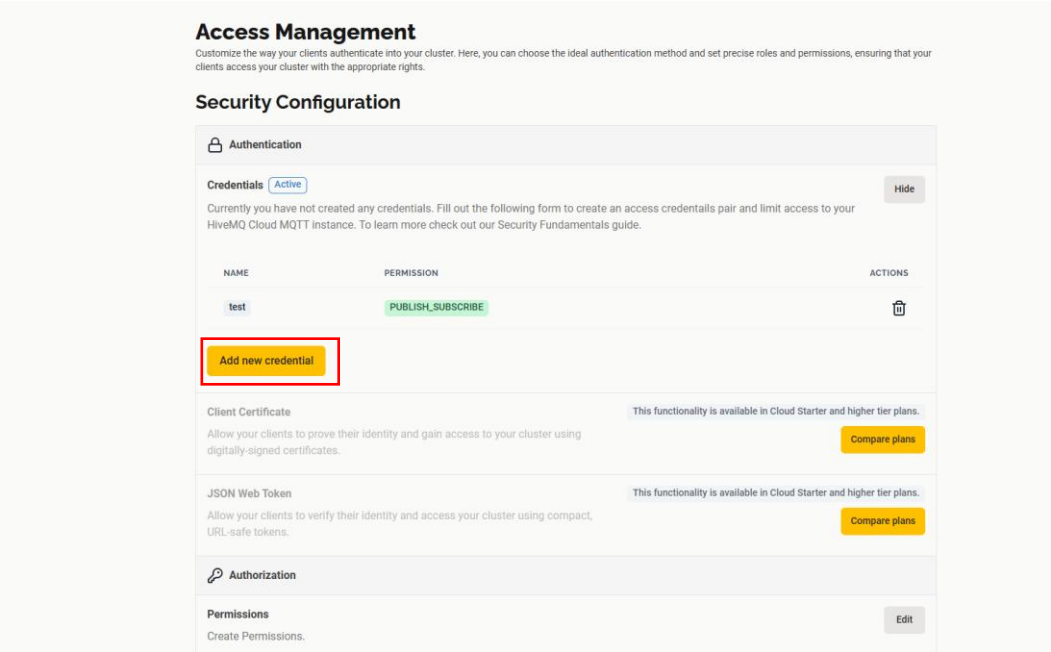
  - Vervolgens ga je naar de juiste cluster

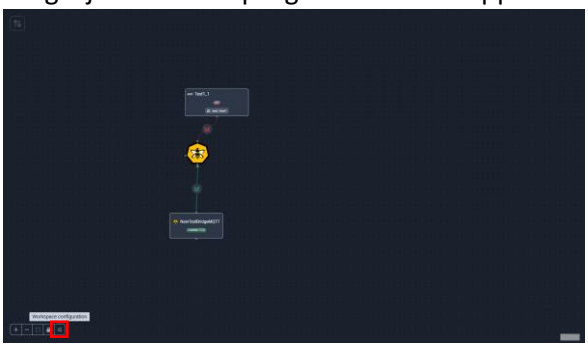


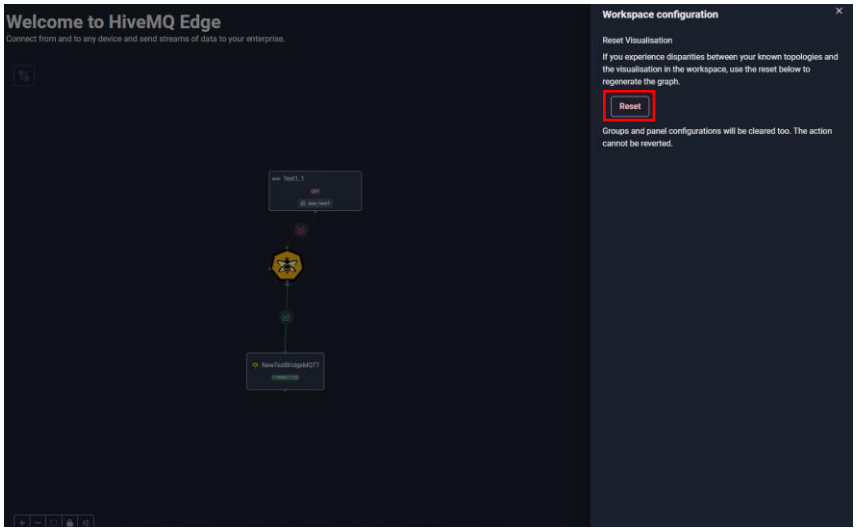
  - Onder overview vind je alle credentials die je nodig hebt



  - Om hivemq edge te gebruiken moeten we nog een webclient aanmaken

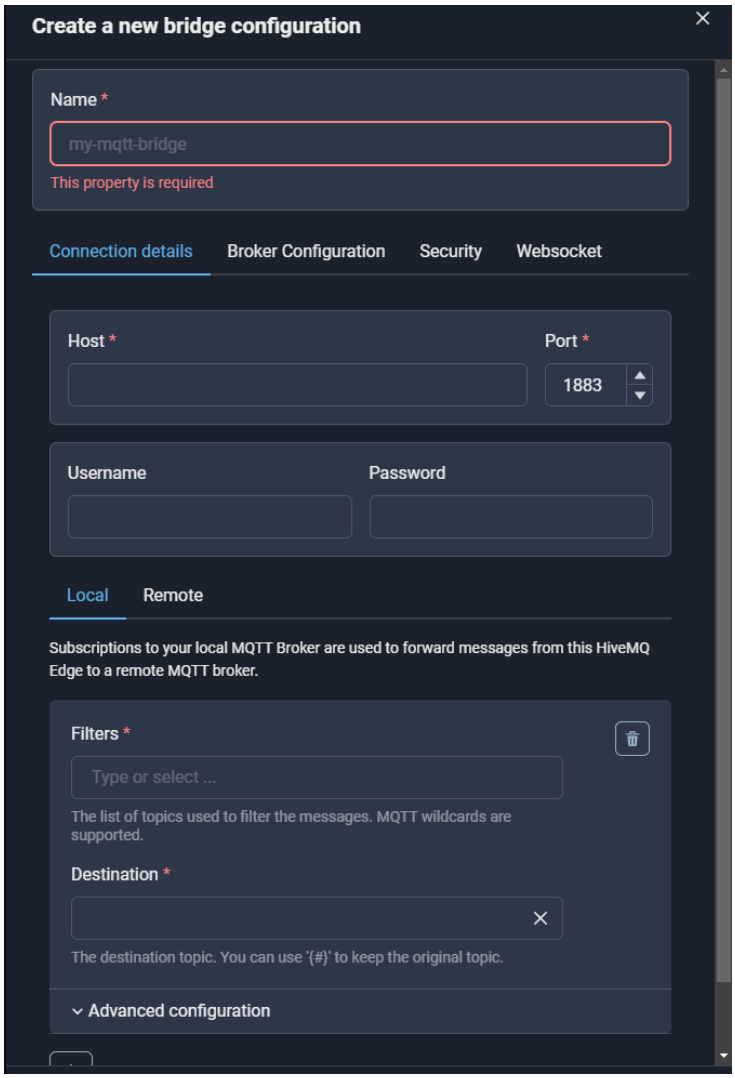
- Onder ACCESS MANAGEMENT kan je nieuwe credentials toevoegen
  - 

- **<u>Hivemq edge</u>**
  - <u>Docker installeren op orange pi</u>
    - *Curl -sSL https://get.docker.com | sh*
    - *Sudo usermod -aG docker $(whoami)*
    - *Docker –version*
  - <u>Pull hivemq edge image</u>
    - *Docker pull hivemq/hivemq-edge*
  - <u>Aanmaken van docker-container</u>
    - *docker run -d --name hivemq-edge \ -v ~/hivemq-edge-config:/opt/hivemq/config \ -p 8080:8080 \ -p 1883:1883 \ hivemq/hivemq-edge*
  - <u>Surfen naar hivemq edge container</u>
    - *http://<Orange-Pi-IP>:8080*
  - **<u>Opmerking</u>**
    - Mogelijk dat hivemq edge container al apparaten bevat op de Workspace
    - 
    - Onderaan klik op WorkSpace Configuration

- 

- *Klik op reset om de workspace te legen en de niet bestaande aparaten te verwijderen*

- **<u>Connectie leggen tussen hivemq cloud en hivemq edge</u>**
    - Op de sidepanel klik op MQTT Bridge, en klik op “Add Bridge Connection” in de rechter bovenhoek



    -
    - <u>Name</u>: *geef hier de unieke naam aan je bridge*
    - <u>Poort</u>: *pas deze aan naar 8883*
    - <u>Host</u>: *de url die je kan halen van hivemq cloud onder “Connection Details”*
    - <u>Username</u>: *de username van de webclient die je net hebt aangemaakt*
    - <u>Password</u>: *het wachtwoord van de webclient die je net hebt aangemaakt*
    - <u>Filters</u>: *geef hier # op om alle topics te gebruiken*
    - <u>Destination</u>: *geef # op om ook alle topics te selecteren*
    - Onder security klik schakel “Enable Transport Layer Security (TLS)” in
    - Klik op “create the bridge” en de connectie is gemaakt

- **Sparkplug b installatie Orange pi**
  - Uitvoeren volgende commando's:
    - *wget https://raw.githubusercontent.com/eclipse/tahu/master/sparkplug_b/sparkplug_b.proto*
    - *protoc --python_out=. sparkplug_b.proto*
  - Nu zou er in de directory waar je de commando's hebt uitgevoerd een bestand met de naam "sparkplug_b_pb2.py" moeten aanwezig zijn.
  - Importeer dan de library in je python project
    - *from sparkplug_b_pb2 import Payload, DataType*
  - Voorbeeld van payload
    -
```python
def create_payload(device):
    payload = Payload()
    metric = payload.metrics.add()
    metric.name = device["sensor"]
    metric.alias = 1
    metric.timestamp = int(time.time() * 1000)

    # Randomly choose data type based on sensor
    if device["sensor"] == "temperatuur":
        metric.datatype = DataType.String  # Use String data type for temperature
        metric.string_value = str(bmp280.get_temperature())  # Convert float to string
    elif device["sensor"] == "luchtdruk":
        metric.datatype = DataType.String  # Use String data type for pressure
        metric.string_value = str(bmp280.get_pressure())  # Convert float to string

    return payload
```

- **Logger.js met sparkplug b**
  - Npm install protobufjs
  - Npm install mqtt
  - Node <bestandsnaam.js> → voor het bestand te kunnen runnen
  - Hou er rekening mee dat door gebruik te maken van sparkplug b de verzonden payload geëncodeerd wordt in binair formaat en dus terug gedecodeerd moet worden voordat deze gebruikt kan worden voor visualisatie
    - Voorbeeld van hoe dit gedaan kan worden is onderstaande afbeelding
    -
```javascript
// Function to process a topic and handle messages
function processTopic(topic, message) {
  let decodedMessage;
  try {
    if (Payload) {
      decodedMessage = Payload.decode(message);
    } else {
      throw new Error('Protobuf Payload not loaded yet.');
    }
  } catch (e) {
    log(`Failed to decode message: ${e}`);
    decodedMessage = { raw: message.toString('hex') };
  }

  if (decodedMessage && decodedMessage.metrics && decodedMessage.metrics[0]) {
    decodedMessage.value = decodedMessage.metrics[0].stringValue;
    const timestamp = decodedMessage.metrics[0].timestamp;
    decodedMessage.timestamp = parseInt(timestamp, 10) + 3600000;
  }

  addTopicToHierarchy(topics, topic, decodedMessage);
}
```

- **Modbus python**
    - pip install pymodbus
    - modbus apparaat (Siemens PAC3200) verbinden met de orange pi
        - Zorg ervoor dat beide apparaten een ip address hebben in eenzelfde range met eenzelfde subnetmasker
            - In het huidige voorbeeld
                - Ethernet poort van orange pi: 192.168.137.2 (255.255.255.0)
                - Modbus apparaat: 192.168.137.3 (255.255.255.0)
            - Instellen ip adres modbus apparaat
                - F4 (Menu) → naar onder gaan tot je op "Settings" komt → F4 (Enter) → ga vervolgens naar "communication" → F4 (enter) → hier vind u alle communicatie instellingen van het apparaat
        - Controleer of de orange pi kan pingen naar het ip adres van het modbus apparaat
        - Indien er een poort gevraagd word of opgegeven moet worden gebruik dan poort 502 (dit is de standaard modbus poort)
            - Voor te weten welke registers er nodig zijn kan je volgende url raadplegen
            - https://www.aggsoft.com/serial-data-logger/tutorials/modbus-data-logging/siemens-sentron-pac-3200.htm
        - Vervolgens ga je de registers uitlezen en in een sparkplugb payload zetten die dan verstuurd worden naar de container

Errors

- **Geen WIFI gevonden**
    - Gebruiker het commando *ip route*
        -
```
(IoTProjectPycharm) orangepi@PIJaroWagener:~$ ip route
default via 10.19.60.1 dev wlan0 proto dhcp metric 600
10.19.60.0/24 dev wlan0 proto kernel scope link src 10.19.60.65 metric 600
169.254.0.0/16 dev eth0 scope link metric 1000
172.17.0.0/16 dev docker0 proto kernel scope link src 172.17.0.1 linkdown
192.168.137.0/24 dev eth0 proto kernel scope link src 192.168.137.2
192.168.137.0/24 dev br-0893c04488fa proto kernel scope link src 192.168.137.1 linkdown
```
    - Indien er 2 default routes zijn, verwijderd diegene die gebruikt word voor te communiceren met het modbus apparaat
        - *sudo ip route del default via 192.168.137.1 dev eth0*
    - Probeer het commando *ping 8.8.8.8*, normaal zou je nu een response moeten terugkrijgen
-