

# Werking Grafana Server

**IoT project**

# Inhoud

- Opzet Server
- Werking MQTT
- Visualisatie Grafana
- Uitdagingen en Oplossingen

# Opzet Server

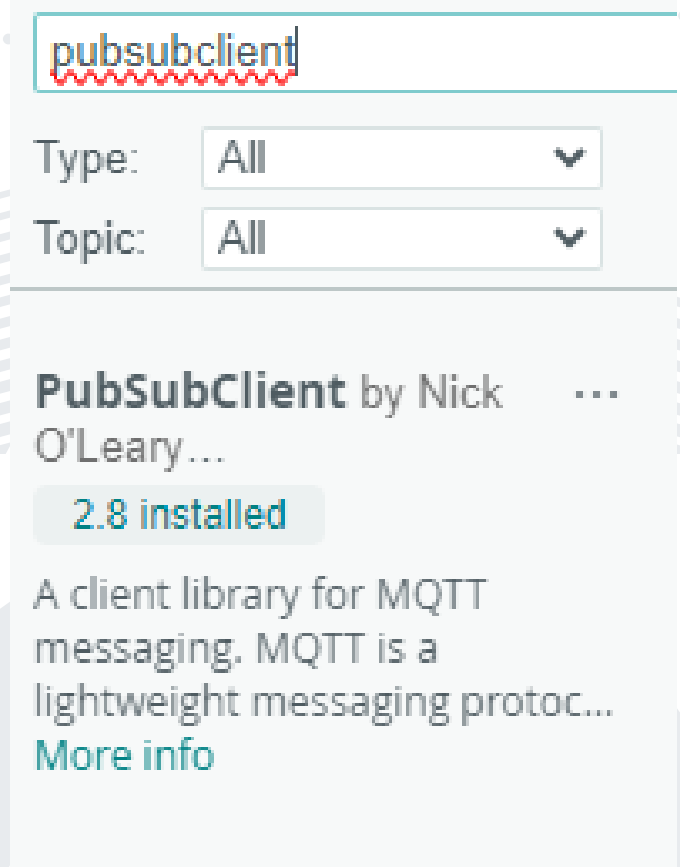
- Duidelijkheid over situatie serre
- Kijken of de planten goed verzorgd geweest zijn
- Industrieel -> kijk om kosten te besparen waar mogelijk
- Van op afstand -> toegankelijker



# Werking MQTT

- MQTT installeren op RaspberryPi en MQTT library installeren op ESP32

```
sudo apt install -y mosquitto mosquitto-clients
```



# Werking MQTT

- Configuratie bestanden aanpassen

```
GNU nano 5.4 /etc/mosquitto/mosquitto.conf *
# Place your local configuration in /etc/mosquitto/conf.d/
#
# A full description of the configuration file is at
# /usr/share/doc/mosquitto/examples/mosquitto.conf.example

pid_file /run/mosquitto/mosquitto.pid

persistence true
persistence_location /var/lib/mosquitto/

log_dest file /var/log/mosquitto/mosquitto.log

include_dir /etc/mosquitto/conf.d

listener 1883
allow_anonymous true
```

# Werking MQTT

- MQTT topics aanmaken in ESP32 code, zeggen wie de broker is en MQTT wachtwoord en username meegeven

```
const char* mqtt_server = "192.168[REDACTED]";  
const int mqtt_port = 1883;  
const char* MQTT_USER = "[REDACTED]";  
const char* MQTT_PASSWORD = "[REDACTED]";  
const char* MQTT_CLIENT_ID = "MQTTclient";
```

```
//Waarden  
const char* MQTT_TOPICtemp = "serre/waarden/temperature";  
const char* MQTT_TOPICHum = "serre/waarden/humidity";  
const char* MQTT_TOPIClight = "serre/waarden/light";  
const char* MQTT_TOPICbv = "serre/waarden/bodemvocht";  
const char* MQTT_TOPICbt = "serre/waarden/bodemtemp";  
const char* MQTT_TOPICWater = "serre/waarden/waterres";  
//Actuatoren  
const char* MQTT_TOPICVentilTempTijd = "serre/actuatoren/VentilTempDraaitijd";  
const char* MQTT_TOPICVentilHumTijd = "serre/actuatoren/VentilHumDraaitijd";  
const char* MQTT_TOPICPompTijd = "serre/actuatoren/PompDraaitijd";  
const char* MQTT_TOPICLichtTijd = "serre/actuatoren/LampDraaitijd";  
const char* MQTT_TOPICVerwTijd = "serre/actuatoren/VerwDraaitijd";  
//Parameters  
const char* MQTT_TOPICParameterMaxTemp = "serre/parameters/MaxTemp";  
const char* MQTT_TOPICParameterMinTemp = "serre/parameters/MinTemp";  
const char* MQTT_TOPICParameterMaxHum = "serre/parameters/MaxHum";  
const char* MQTT_TOPICParameterMinLight = "serre/parameters/MinLight";  
const char* MQTT_TOPICParameterMinBv = "serre/parameters/MinBv";  
const char* MQTT_REGEX = "serre/([^\s]+)/([^\s]+)";
```

# Werking MQTT

- RaspberryPi laten subscriben op die MQTT topics, zeggen dat hijzelf de broker is en MQTT wachtwoord en username meegeven

```
# MQTT instellingen
MQTT_ADDRESS = 'localhost'
MQTT_USER = '██████'
MQTT_PASSWORD = '██████'
MQTT_TOPICWaarden = "serre/waarden/+"
MQTT_TOPICActuatoren = "serre/actuatoren/+"
MQTT_TOPICParameters = "serre/parameters/+"
MQTT_REGEX = "serre/([^/]+)/([^/]+)"
```

# Werking MQTT

- Gegevens doorsturen via de topic's naar de broker

```
void sendParameters() {  
    String maxTemps = String(maxTemp);  
    String minTemps = String(minTemp);  
    String maxHums = String(maxHum);  
    String minLights = String(minLight);  
    String minBodemvochts = String(minBodemvocht);  
    mqttClient.publish(MQTT_TOPICParameterMaxTemp, maxTemps.c_str());  
    mqttClient.publish(MQTT_TOPICParameterMinTemp, minTemps.c_str());  
    mqttClient.publish(MQTT_TOPICParameterMaxHum, maxHums.c_str());  
    mqttClient.publish(MQTT_TOPICParameterMinLight, minLights.c_str());  
    mqttClient.publish(MQTT_TOPICParameterMinBv, minBodemvochts.c_str());  
    Serial.println("parameters verzonden");  
}
```



# Werking MQTT

- Gegevens ontvangen op de geabonneerde topic's

```
serre/actuatoren/LampDraaitijd b'0'  
serre/actuatoren/PompDraaitijd b'1'  
serre/actuatoren/VentilTempDraaitijd b'0'  
serre/actuatoren/VerwDraaitijd b'0'  
serre/actuatoren/VentilHumDraaitijd b'0'  
serre/actuatoren/LampDraaitijd b'0'  
serre/actuatoren/PompDraaitijd b'1'  
serre/actuatoren/VentilTempDraaitijd b'0'  
serre/actuatoren/VerwDraaitijd b'0'  
serre/actuatoren/VentilHumDraaitijd b'0'  
serre/actuatoren/LampDraaitijd b'0'  
serre/actuatoren/PompDraaitijd b'1'  
serre/waarden/waterres b'26.39'  
serre/actuatoren/VentilTempDraaitijd b'0'  
serre/actuatoren/VerwDraaitijd b'0'  
serre/actuatoren/VentilHumDraaitijd b'0'  
serre/actuatoren/LampDraaitijd b'0'  
serre/actuatoren/PompDraaitijd b'1'  
serre/actuatoren/VentilTempDraaitijd b'0'  
serre/actuatoren/VerwDraaitijd b'0'  
serre/actuatoren/VentilHumDraaitijd b'0'  
serre/actuatoren/LampDraaitijd b'0'  
serre/actuatoren/PompDraaitijd b'1'
```

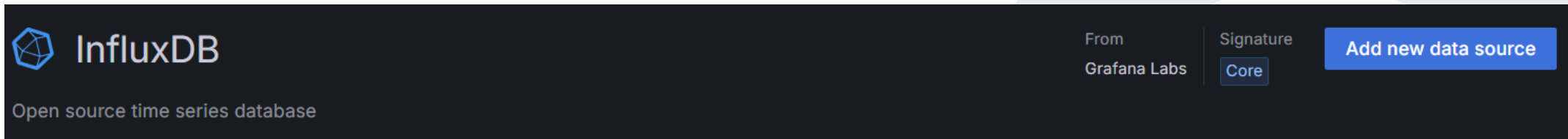
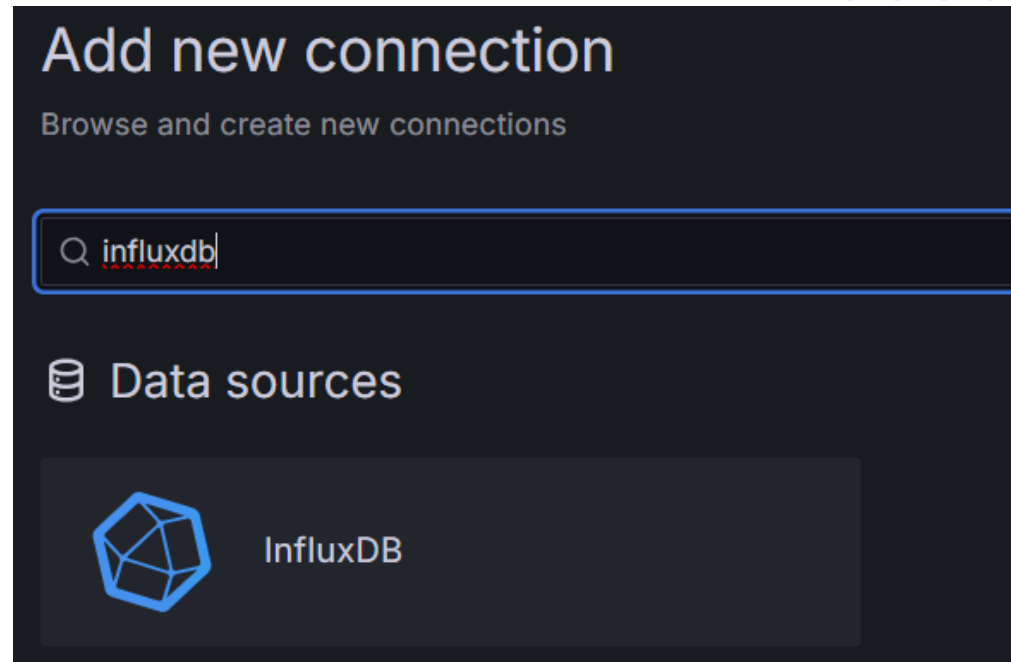
# Grafana Visualisatie

- Data sturen naar database (InfluxDB)
- Via webserver naar Grafana server gaan
  - IP adres van broker opzoeken via poort 3000 (192.168....:3000)
- Inloggen standaard “admin” “admin”



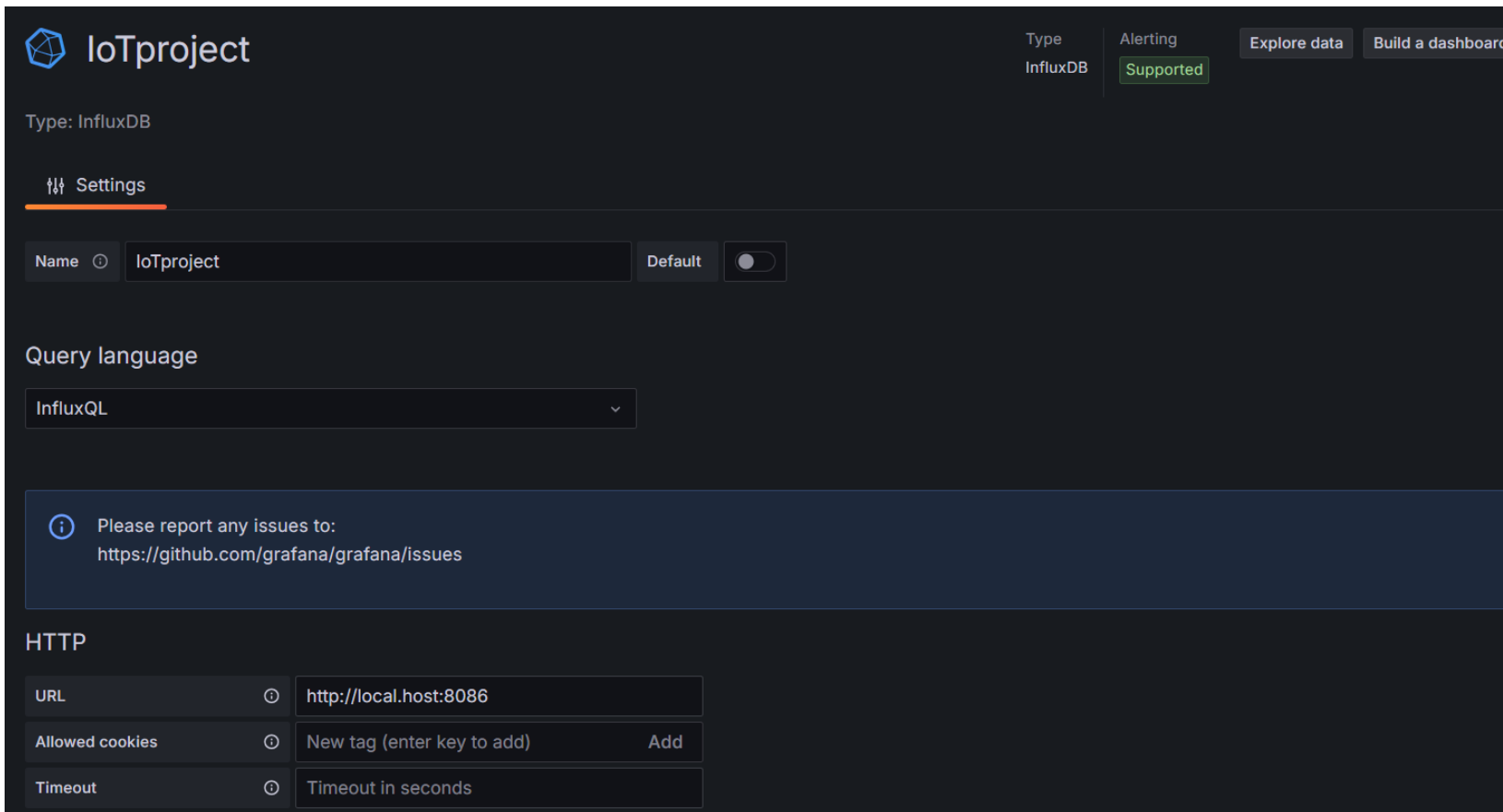
# Grafana Visualisatie

- Database connecteren met Grafana



# Grafana Visualisatie

- Database connecteren met Grafana

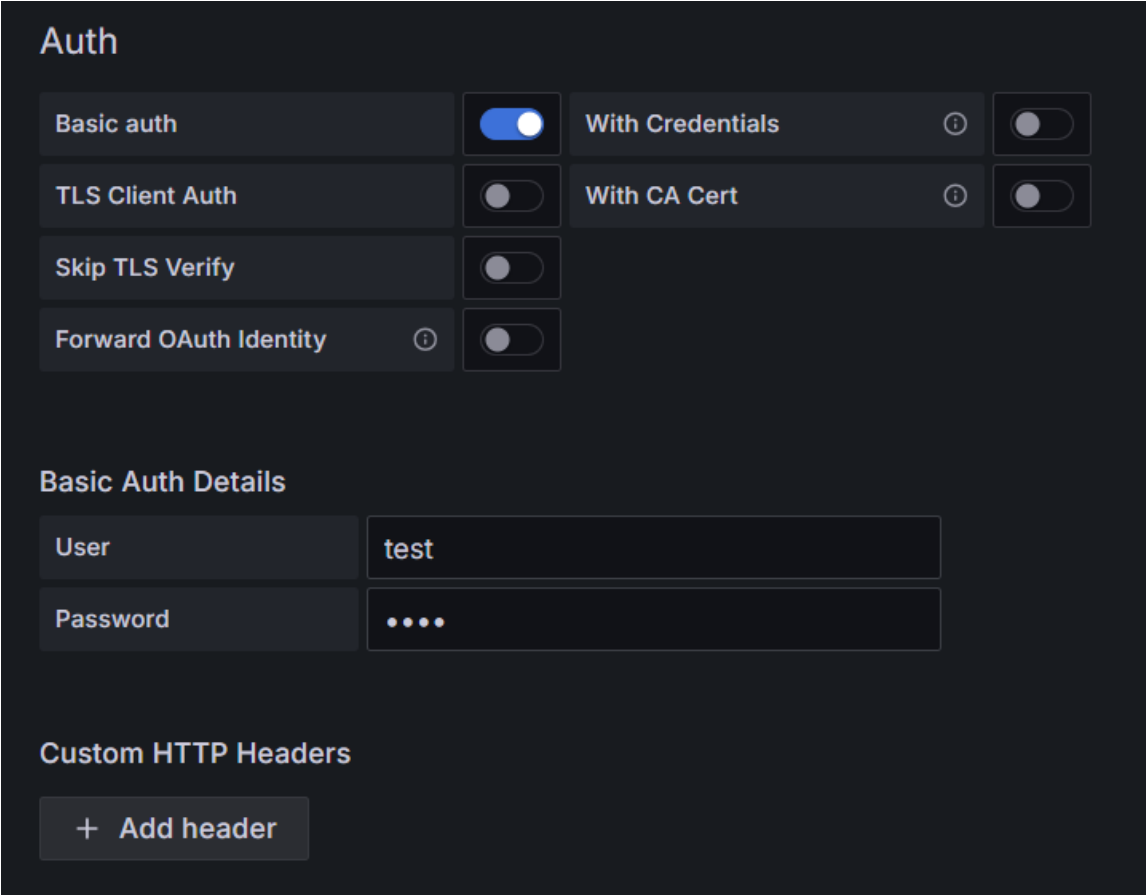


The screenshot shows the Grafana IoTproject settings page for an InfluxDB database. The page has a dark theme. At the top, the 'IoTproject' logo is on the left, and navigation links for 'Type InfluxDB', 'Alerting Supported', 'Explore data', and 'Build a dashboard' are on the right. Below the logo, the text 'Type: InfluxDB' is displayed. A 'Settings' tab is selected and underlined. The settings are organized into sections: 'Name' with a text input 'IoTproject' and a 'Default' toggle switch; 'Query language' with a dropdown menu set to 'InfluxQL'; an information box with a link to report issues; and an 'HTTP' section with a table of settings.

HTTP	
URL	<input type="text" value="http://local.host:8086"/>
Allowed cookies	<input type="text" value="New tag (enter key to add)"/> <button>Add</button>
Timeout	<input type="text" value="Timeout in seconds"/>

# Grafana Visualisatie

- Database connecteren met Grafana



The image shows the 'Auth' configuration panel in Grafana. It is a dark-themed interface with several sections. The 'Auth' section at the top contains four rows of settings, each with a label, a toggle switch, and an information icon. 'Basic auth' is enabled (blue toggle), while 'TLS Client Auth', 'Skip TLS Verify', and 'Forward OAuth Identity' are disabled (grey toggles). To the right of the 'Basic auth' and 'TLS Client Auth' rows are additional settings: 'With Credentials' and 'With CA Cert', both with their own toggle switches and information icons. Below this is the 'Basic Auth Details' section, which has two input fields: 'User' with the value 'test' and 'Password' with masked characters '....'. At the bottom is the 'Custom HTTP Headers' section, which contains a single button labeled '+ Add header'.

Auth

Basic auth ☒ With Credentials ⓘ ☐

TLS Client Auth ☐ With CA Cert ⓘ ☐

Skip TLS Verify ☐

Forward OAuth Identity ⓘ ☐

Basic Auth Details

User test

Password .....

Custom HTTP Headers

+ Add header

# Werking MQTT

- Gegevens ontvangen op de geabonneerde topic's

Database	IoTproject
User	thibau
Password	••••
HTTP Method ⓘ	Choose ▼
Min time interval ⓘ	10s
Max series ⓘ	1000

- Als alles goed is komt het aantal topics van je database tevoorschijn en kan je beginnen aan je visualisaties

# Visualisaties Grafana

- Vrij makkelijk en duidelijk (<https://www.youtube.com/watch?v=yNRnLyVntUw>)



# Uitdagingen en Oplossingen

- Alles moet exact hetzelfde zijn
  - Kleine typ fout en de communicatie kan volledig vastlopen





THOMAS  
**MORE**

**Bedankt!**

