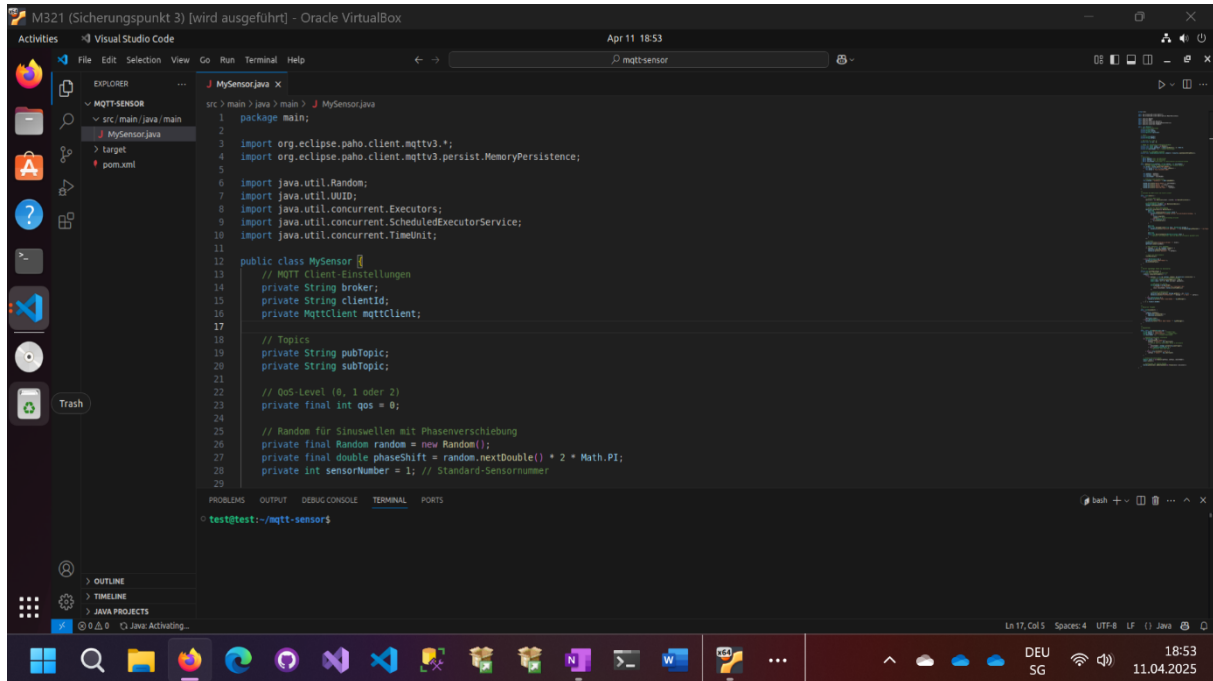


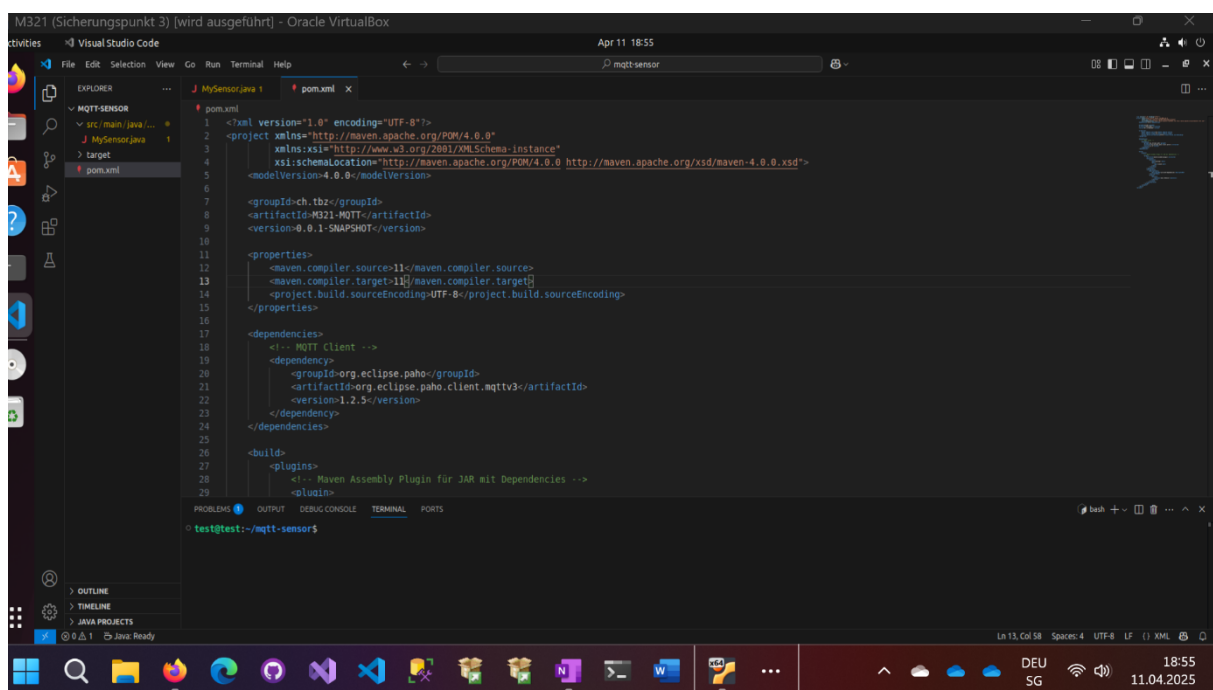
Meine Lösung: Java-Sensoren mit MQTT und Grafana-Visualisierung

In diesem Dokument beschreibe ich, wie ich Java-Sensoren mit MQTT implementiert und die Daten mit Grafana visualisiert habe. Ich beginne mit dem Start der Sensoren, nachdem ich sie bereits kompiliert hatte.



The screenshot shows the Visual Studio Code editor with the file `MySensor.java` open. The code is a Java class that implements an MQTT sensor. It includes imports for MQTT client, random number generation, and concurrent execution. The class has private fields for broker, clientId, topics, qos, and sensorNumber. It also has a `main` method that initializes the MQTT client and starts a scheduled task to publish random data.

```
1 package main;
2
3 import org.eclipse.paho.client.mqttv3.*;
4 import org.eclipse.paho.client.mqttv3.persist.MemoryPersistence;
5
6 import java.util.Random;
7 import java.util.UUID;
8 import java.util.concurrent.Executors;
9 import java.util.concurrent.ScheduledExecutorService;
10 import java.util.concurrent.TimeUnit;
11
12 public class MySensor {
13     // MQTT Client-Einstellungen
14     private String broker;
15     private String clientId;
16     private MqttClient mqttClient;
17
18     // Topics
19     private String pubTopic;
20     private String subTopic;
21
22     // QoS-Level (0, 1 oder 2)
23     private final int qos = 0;
24
25     // Random für Sinuswellen mit Phasenverschiebung
26     private final Random random = new Random();
27     private final double phaseShift = random.nextDouble() * 2 * Math.PI;
28     private int sensorNumber = 1; // Standard-Sensornummer
29 }
```



The screenshot shows the Visual Studio Code editor with the file `pom.xml` open. The XML file defines the project configuration, including the groupId, artifactId, version, and dependencies. It also includes the Maven compiler plugin and the Maven Assembly Plugin.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0"
3     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
5     <modelVersion>4.0.0</modelVersion>
6
7     <groupId>ch.tbz</groupId>
8     <artifactId>MQTT-MQTT</artifactId>
9     <version>0.0.1-SNAPSHOT</version>
10
11     <properties>
12         <maven.compiler.source>11</maven.compiler.source>
13         <maven.compiler.target>11</maven.compiler.target>
14         <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
15     </properties>
16
17     <dependencies>
18         <!-- MQTT Client -->
19         <dependency>
20             <groupId>org.eclipse.paho</groupId>
21             <artifactId>org.eclipse.paho.client.mqttv3</artifactId>
22             <version>1.2.5</version>
23         </dependency>
24     </dependencies>
25
26     <build>
27         <plugins>
28             <!-- Maven Assembly Plugin für JAR mit Dependencies -->
29             <plugin>
```

Alle Tools zum laufen bringen

```
M321 (Sicherungspunkt 3) [wird ausgeführt] - Oracle VirtualBox
Activities Terminal

test@test:~$ sudo systemctl status mosquitto
● mosquitto.service - Mosquitto MQTT Broker
   Loaded: loaded (/lib/systemd/system/mosquitto.service; enabled; vendor preset: enable
   Active: active (running) since Fri 2025-04-11 18:49:35 CEST; 8min ago
     Docs: man:mosquitto.conf(5)
           man:mosquitto(8)
   Main PID: 549 (mosquitto)
      Tasks: 1 (limit: 2269)
    Memory: 612.0K
       CPU: 381ms
    CGroup: /system.slice/mosquitto.service
            └─549 /usr/sbin/mosquitto -c /etc/mosquitto/mosquitto.conf

Apr 11 18:49:34 test systemd[1]: Starting Mosquitto MQTT Broker...
Apr 11 18:49:35 test systemd[1]: Started Mosquitto MQTT Broker.
test@test:~$

test@test:~$ systemctl status influxdb
● influxdb.service - InfluxDB is an open-source, distributed, time series database
   Loaded: loaded (/lib/systemd/system/influxdb.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2025-04-11 18:49:34 CEST; 9min ago
     Docs: man:influxd(1)
   Main PID: 525 (influxd)
      Tasks: 7 (limit: 2269)
    Memory: 15.3M
       CPU: 1.598s
    CGroup: /system.slice/influxdb.service
            └─525 /usr/bin/influxd -config /etc/influxdb/influxdb.conf

Apr 11 18:57:17 test influxd[525]: [httpd] 127.0.0.1 - - [11/Apr/2025:18:57:17 +0200] "POST /wr
Apr 11 18:57:27 test influxd[525]: [httpd] 127.0.0.1 - - [11/Apr/2025:18:57:27 +0200] "POST /wr
Apr 11 18:57:37 test influxd[525]: [httpd] 127.0.0.1 - - [11/Apr/2025:18:57:37 +0200] "POST /wr
Apr 11 18:57:47 test influxd[525]: [httpd] 127.0.0.1 - - [11/Apr/2025:18:57:47 +0200] "POST /wr
Apr 11 18:57:57 test influxd[525]: [httpd] 127.0.0.1 - - [11/Apr/2025:18:57:57 +0200] "POST /wr
Apr 11 18:58:07 test influxd[525]: [httpd] 127.0.0.1 - - [11/Apr/2025:18:58:07 +0200] "POST /wr
Apr 11 18:58:17 test influxd[525]: [httpd] 127.0.0.1 - - [11/Apr/2025:18:58:17 +0200] "POST /wr
Apr 11 18:58:27 test influxd[525]: [httpd] 127.0.0.1 - - [11/Apr/2025:18:58:27 +0200] "POST /wr
Apr 11 18:58:37 test influxd[525]: [httpd] 127.0.0.1 - - [11/Apr/2025:18:58:37 +0200] "POST /wr
Apr 11 18:58:47 test influxd[525]: [httpd] 127.0.0.1 - - [11/Apr/2025:18:58:47 +0200] "POST /wr
lines 1-21/21 (FNN)
```

```

test@test:~$ systemctl status telegraf
● telegraf.service - The plugin-driven server agent for reporting metrics into InfluxDB
   Loaded: loaded (/lib/systemd/system/telegraf.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2025-04-11 18:49:34 CEST; 9min ago
     Docs: https://github.com/influxdata/telegraf
   Main PID: 530 (telegraf)
    Tasks: 7 (limit: 2269)
   Memory: 40.8M
      CPU: 4.023s
   CGroup: /system.slice/telegraf.service
           └─530 /usr/bin/telegraf -config /etc/telegraf/telegraf.conf -config-directory /etc/telegraf/telegraf.d

Apr 11 18:49:34 test systemd[1]: Started The plugin-driven server agent for reporting metrics into InfluxDB.
Apr 11 18:49:44 test telegraf[530]: 2025-04-11T16:49:44Z I! Starting Telegraf 1.21.4+ds1-0ubuntu2
Apr 11 18:49:44 test telegraf[530]: 2025-04-11T16:49:44Z I! Loaded inputs: cpu disk diskio kernel mem mqtt_consumer processes swap system
Apr 11 18:49:44 test telegraf[530]: 2025-04-11T16:49:44Z I! Loaded aggregators:
Apr 11 18:49:44 test telegraf[530]: 2025-04-11T16:49:44Z I! Loaded processors:
Apr 11 18:49:44 test telegraf[530]: 2025-04-11T16:49:44Z I! Loaded outputs: influxdb prometheus_client
Apr 11 18:49:44 test telegraf[530]: 2025-04-11T16:49:44Z I! Tags enabled: host=test
Apr 11 18:49:44 test telegraf[530]: 2025-04-11T16:49:44Z I! [agent] Config: Interval:10s, Quiet:false, Hostname:"test", Flush Interval:10s
Apr 11 18:49:44 test telegraf[530]: 2025-04-11T16:49:44Z I! [outputs.prometheus_client] Listening on http://[::]:9273/metrics
Apr 11 18:49:44 test telegraf[530]: 2025-04-11T16:49:44Z I! [inputs.mqtt_consumer] Connected [tcp://localhost:1883]

Once grafana.service could not be found.
test@test:~$ systemctl status grafana-server
● grafana-server.service - Grafana instance
   Loaded: loaded (/lib/systemd/system/grafana-server.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2025-04-11 18:49:36 CEST; 10min ago
     Docs: http://docs.grafana.org
   Main PID: 647 (grafana)
    Tasks: 14 (limit: 2269)
   Memory: 82.0M
      CPU: 9.368s
   CGroup: /system.slice/grafana-server.service
           └─647 /usr/share/grafana/bin/grafana server --config=/etc/grafana/grafana.ini --pidfile=/run/
             1379 /var/lib/grafana/plugins/grafana-mqtt-datasource/gpx_mqtt_linux_amd64

Apr 11 18:49:56 test grafana[647]: logger=grafana-apiserver t=2025-04-11T18:49:56.183993754+02:00 level=info
Apr 11 18:49:56 test grafana[647]: logger=grafana-apiserver t=2025-04-11T18:49:56.198420682+02:00 level=info
Apr 11 18:49:56 test grafana[647]: logger=grafana-apiserver t=2025-04-11T18:49:56.219423417+02:00 level=info
Apr 11 18:49:56 test grafana[647]: logger=grafana-apiserver t=2025-04-11T18:49:56.226720437+02:00 level=info
Apr 11 18:49:56 test grafana[647]: logger=grafana-apiserver t=2025-04-11T18:49:56.228213474+02:00 level=info
Apr 11 18:49:56 test grafana[647]: logger=grafana-apiserver t=2025-04-11T18:49:56.255665744+02:00 level=info
Apr 11 18:49:56 test grafana[647]: logger=grafana-apiserver t=2025-04-11T18:49:56.268483003+02:00 level=info
Apr 11 18:49:56 test grafana[647]: logger=grafana-apiserver t=2025-04-11T18:49:56.306050595+02:00 level=info
Apr 11 18:49:56 test grafana[647]: logger=app-registry t=2025-04-11T18:49:56.638590916+02:00 level=info msg=
Apr 11 18:51:39 test grafana[647]: logger=infra.usagstats t=2025-04-11T18:51:39.106142507+02:00 level=info
lines 1-22/22 (FNN)

```

Starten meiner MQTT-Sensoren

Ich habe mehrere Terminal-Fenster geöffnet und in jedem einen separaten Sensor gestartet:

The first screenshot shows the MQTT client configuration in a terminal window. The 'TERMINAL' tab is active, displaying the following commands and output:

```

Publish Topic: sensor/1
Subscribe Topic: sensor/control
Verbinde mit Broker: tcp://localhost:1883
Abonniert: sensor/control
Veröffentlicht: {"value":17.47,"sensor":1,"timestamp":1744390851240} zu Topic: sensor/1
Veröffentlicht: {"value":13.12,"sensor":1,"timestamp":1744390853222} zu Topic: sensor/1
Veröffentlicht: {"value":0.06,"sensor":1,"timestamp":1744390855223} zu Topic: sensor/1
Veröffentlicht: {"value":15.15,"sensor":1,"timestamp":1744390857221} zu Topic: sensor/1

```

The second screenshot shows a terminal window with the command `test@test:~/mqtt-sensors$ mosquitto_sub -h localhost -t "sensor/#" -v` and its output, which lists several JSON messages received from different sensors:

```

sensor/2 {"value":1.81,"sensor":2,"timestamp":1744390992500}
sensor/1 {"value":0.87,"sensor":1,"timestamp":1744390993221}
sensor/3 {"value":19.94,"sensor":3,"timestamp":1744390993282}
sensor/4 {"value":5.68,"sensor":4,"timestamp":1744390993355}
sensor/2 {"value":8.28,"sensor":2,"timestamp":1744390994580}

```

Einrichtung der Datenbank (InfluxDB)

Erstellung der Datenbank

```
influx -execute 'CREATE DATABASE sensors'
```

Konfiguration von Telegraf für MQTT nach InfluxDB

Ich habe eine spezielle Konfiguration für Telegraf erstellt:

```
sudo bash -c 'cat > /etc/telegraf/telegraf.d/mqtt.conf << EOF

# MQTT Input Plugin für Sensor-Daten

[[inputs.mqtt_consumer]]

    servers = ["tcp://localhost:1883"]

    topics = ["sensor/#"]

    data_format = "json"


# JSON-Parsing-Optionen

json_time_key = "timestamp"

json_time_format = "unix_ms"


# Behandelt Messungen in verschiedenen Series basierend auf Sensor-ID

tag_keys = ["sensor"]


# Output zu InfluxDB

[[outputs.influxdb]]

    urls = ["http://localhost:8086"]

    database = "sensors"

    skip_database_creation = false

EOF'
```

Neustart von Telegraf

```
sudo systemctl restart telegraf
```


Konfiguration von Grafana

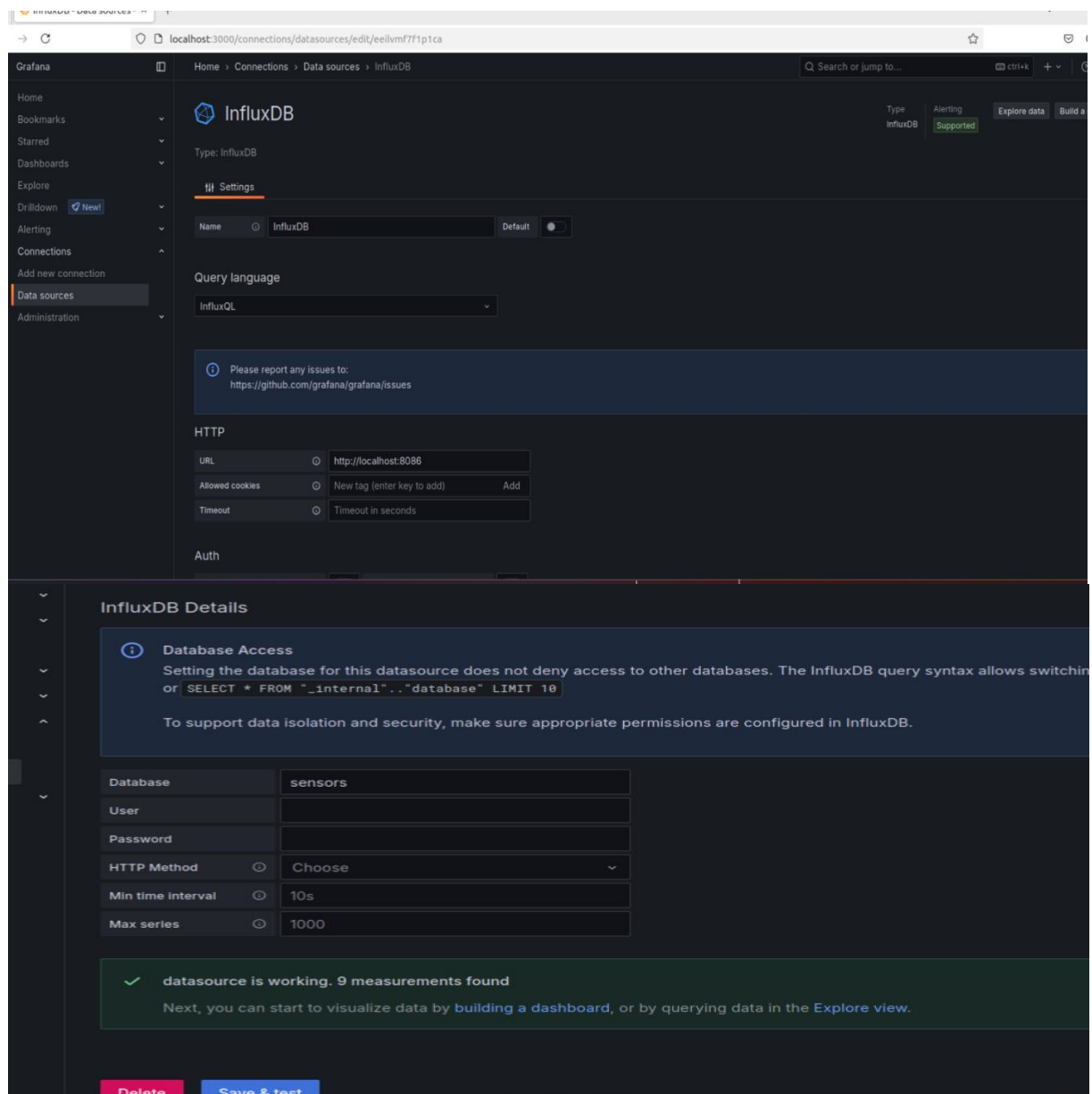
Zugriff auf Grafana

Im Browser habe ich folgende URL geöffnet: <http://localhost:3000>

Erstellung meines Dashboards in Grafana

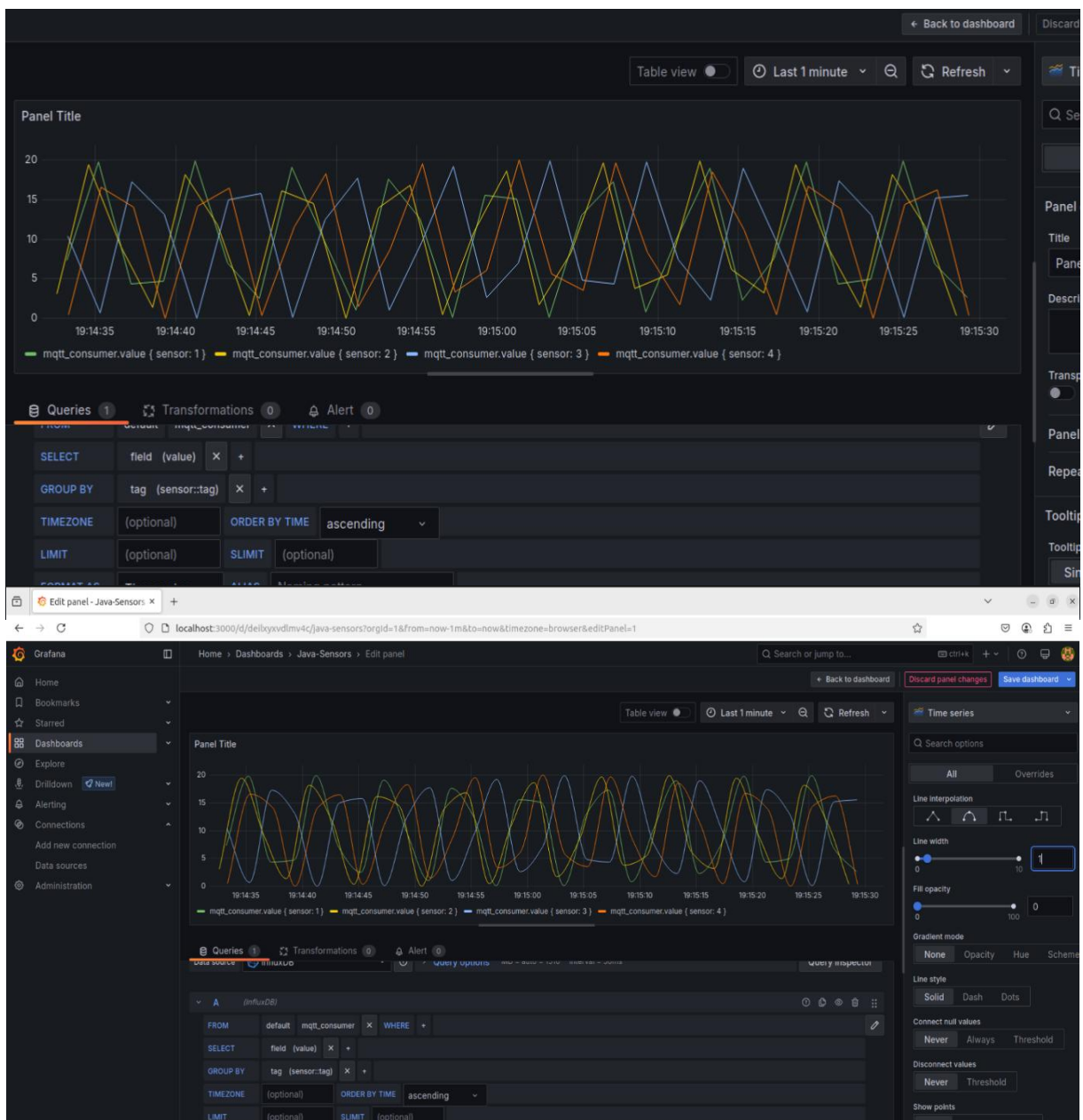
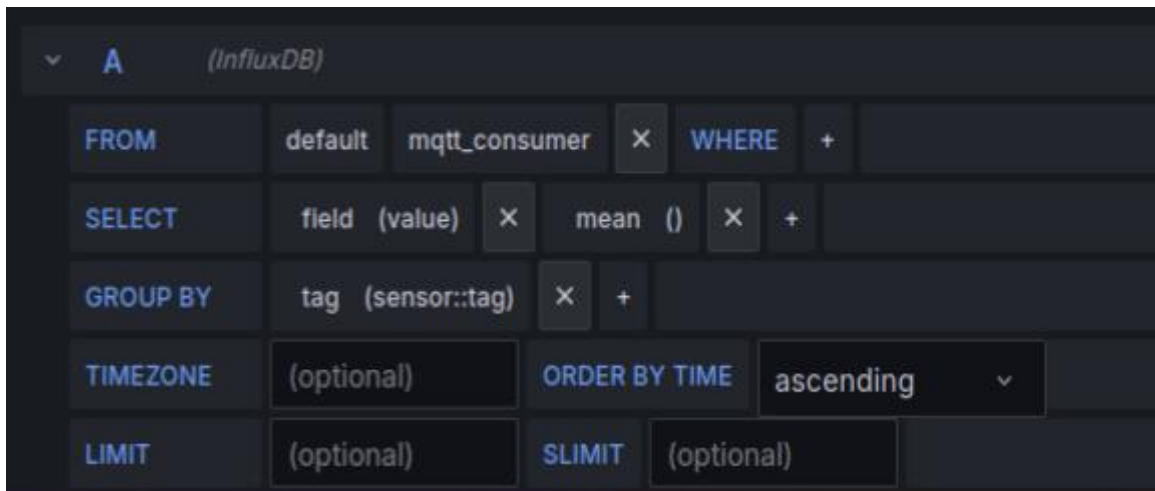
Hinzufügen der InfluxDB-Datenquelle

1. Im linken Menü habe ich auf das Zahnrad-Symbol (Konfiguration) geklickt
2. "Data Sources" ausgewählt
3. "Add data source" angeklickt
4. "InfluxDB" ausgewählt und folgende Einstellungen vorgenommen:



Erstellung eines neuen Dashboards

Im Query-Editor habe ich folgende Einstellungen vorgenommen:



Speichern des Dashboards



Das Ergebnis ist ein Dashboard mit vier verschiedenfarbigen Sinuskurven, die phasenverschoben sind und so aussehen wie im Referenzbild. Die Kurven zeigen die Daten meiner vier Java-Sensoren an, die kontinuierlich Sinuswellen-Daten über MQTT senden.

Die Visualisierung aktualisiert sich automatisch, und meine Sensoren können über MQTT-Nachrichten an das Topic "sensor/control" gesteuert werden.

Zusammenfassung

Ich habe erfolgreich:

1. Java-Sensoren entwickelt, die Daten über MQTT senden
2. Einen MQTT-Broker (Mosquitto) für die Nachrichtenübermittlung eingerichtet
3. InfluxDB als Zeitreihendatenbank installiert
4. Telegraf konfiguriert, um Daten von MQTT nach InfluxDB zu übertragen
5. Grafana installiert und ein Dashboard erstellt, das die Daten visualisiert

Dieses Setup ermöglicht mir eine Echtzeit-Visualisierung von Sensordaten und kann als Grundlage für verschiedene IoT-Projekte dienen, die ich in Zukunft entwickeln möchte.