**Dokumentation Wetterstation**

|  |  |
| --- | --- |
| **Projektbezeichnung** | Projektarbeit Mikrocomputertechnik |
| **Projektleiter** | Noah Canadea / Milan Bursac |
| **Erstellt am** | 20.12.2021 |
| **Letzte Änderung am** | 10.01.2022 |
| **Status** | abgeschlossen |
| **Aktuelle Version** | 1.0 |

**Änderungsverlauf**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Nr.** | **Datum** | **Version** | **Geänderte Kapitel** | **Art der Änderung** | **Autor** | **Status** |
| 1 | 20.12.2021 | 0.5 | Alle | Erstellung | Noah Canadea | ok |
| 2 | 03.01.2022 | 0.6 | 6, 7, 8 | Ergänzung | Noah Canadea | Ok |
| 3 | 08.01.2022 | 0.7 | Alle | Anpassungen | Noah Canadea | Ok |

Inhalt

[1 Schemas 3](#_Toc92549046)

[1.1 Elektronik Schema ESP (Wetterstation) 3](#_Toc92549047)

[1.2 Aktivitätsdiagramme 3](#_Toc92549048)

[1.2.1 ESP Code (Wetterstation) 3](#_Toc92549049)

[1.2.2 NodeRed (Backend) 3](#_Toc92549050)

[1.3 MySQL Schema 4](#_Toc92549051)

[1.4 Prinzipschema Coud 4](#_Toc92549052)

[2 Soll – Ist Abgleich 6](#_Toc92549053)

[3 Verwendete Produkte und Librarys 8](#_Toc92549054)

[3.1 Backend / Frontend 8](#_Toc92549055)

[3.2 ESP8266 Wetterstation Librarys 8](#_Toc92549056)

[3.3 ESP8266 Wetterstation Hardware 8](#_Toc92549057)

[4 Anhang 8](#_Toc92549058)

# Schemas

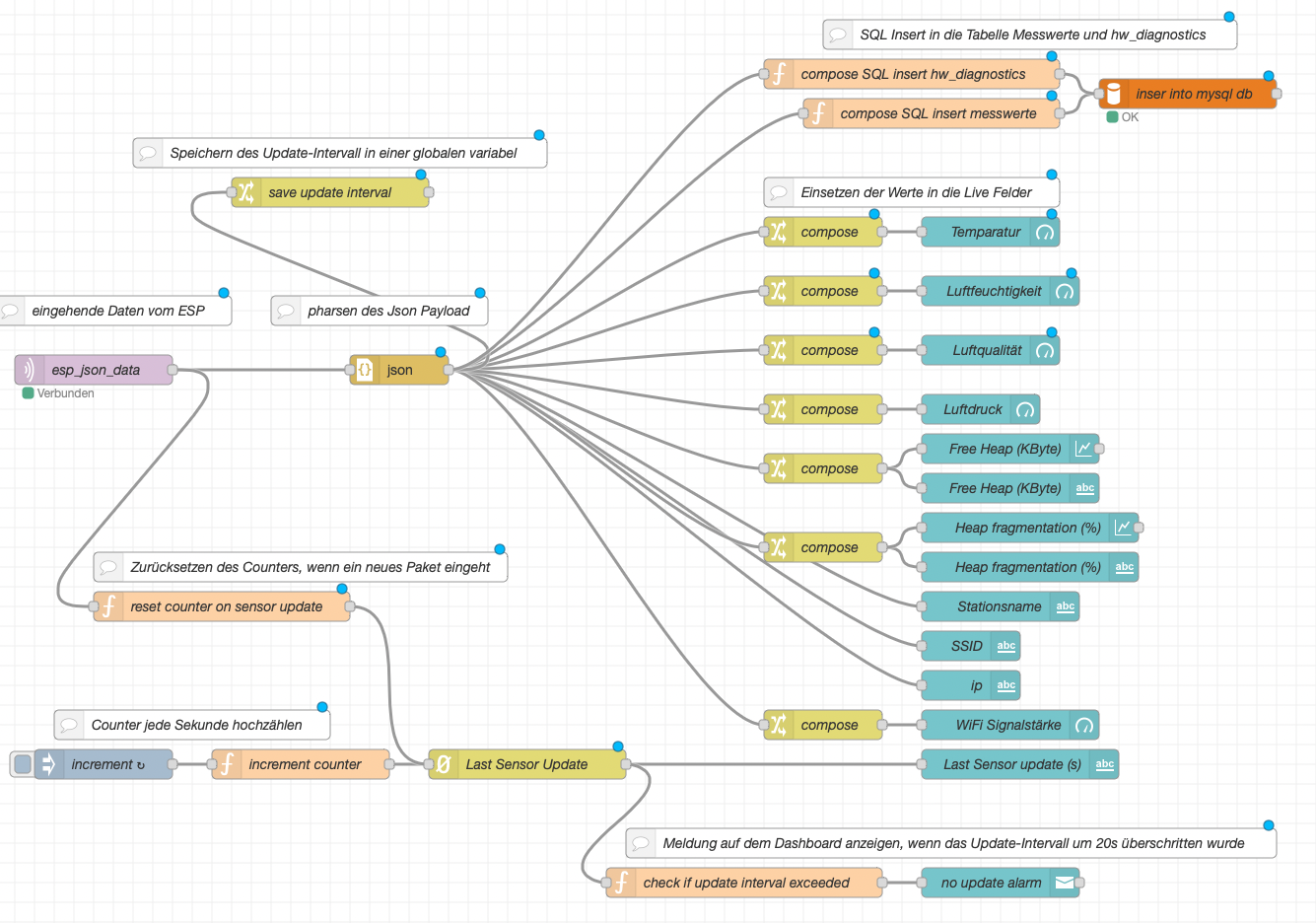
## Elektronik Schema ESP (Wetterstation)

## Aktivitätsdiagramme

### ESP Code (Wetterstation)

* + 1. NodeRed (Backend)

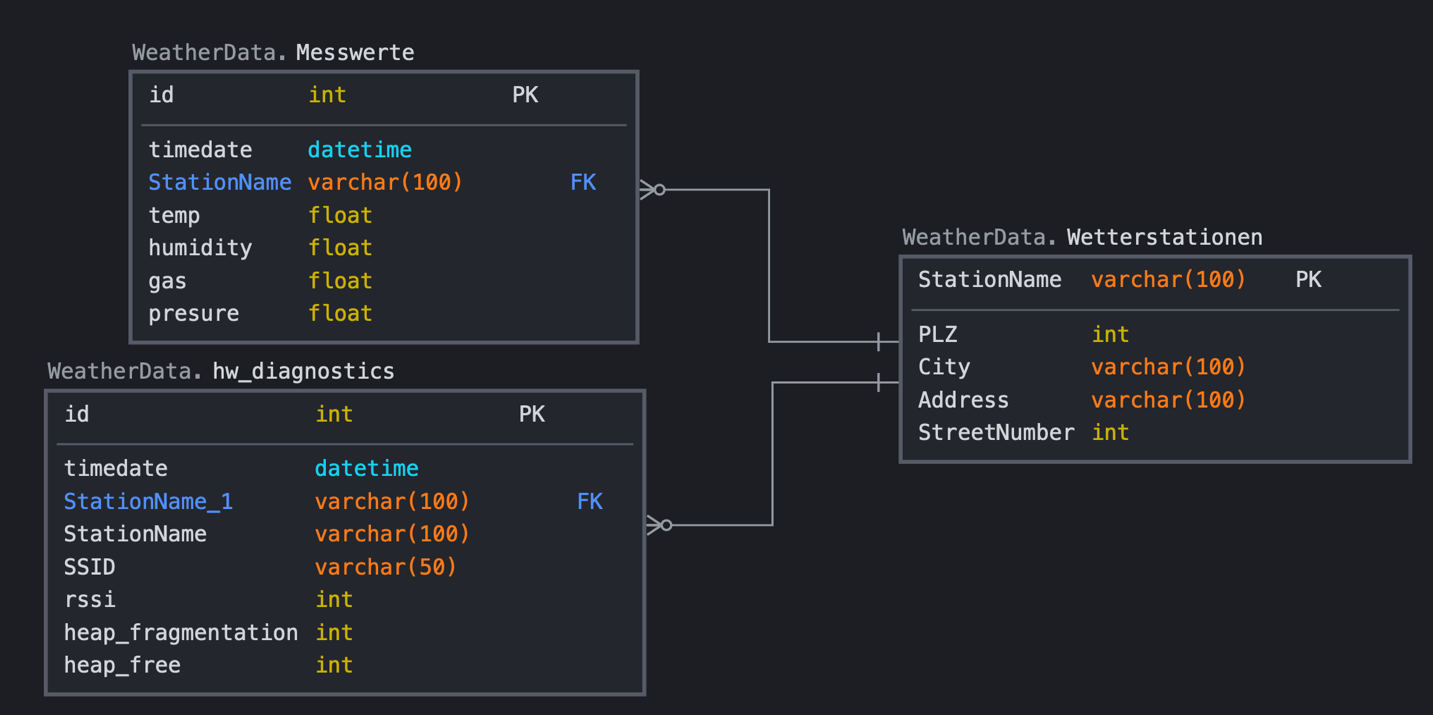
Live Data Handler



Hystory Data HandlerEin Bild, das Text enthält.

Automatisch generierte Beschreibung

## MySQL Schema



## Prinzipschema Coud

# Soll – Ist Abgleich

|  |  |  |
| --- | --- | --- |
| **Soll** | **Ist** | **Erfüllt** |
| **4.1 Erheben folgender Messwerte:**  - Temperatur  - Luftfeuchtigkeit  - Luftqualität  - Luftdruck | Temperatur, Luftfeuchtigkeit, Luftqualität und Luftdruck werden aufgezeichnet. | JA |
| **4.2 Erheben diverser Metriken der Wetterstation:**   * iP Adresse * Hostname * Systemzustand * Letztes Update der Station usw. | Die Station sendet diverse Metadaten an den Node-Red Server:   * Hostname * Verbundene SSID * WiFi Stärke in RSSI * iP Adresse * Heap Fragmentierung * Freier Heap Speicher | JA |
| **4.3 Persistente Speicherung der Messwerte in der Cloud:**  Sämtliche, aufgezeichneten Messwerte sollen auf einem zentralen Server in der Cloud für einen definierten Zeitraum gespeichert werden. Hierzu soll eine DB wie Bspw. MySQL verwendet werden, auf welche zu einem späteren Zeitpunkt weitere Systeme für die Auswertung angebunden werden können. | Die Messwerte werden von der Wetterstation als Json Payload per MQTT an den Node-Red gesendet. Dieser speichert die Werte anschließend in einem MySQL Server. | JA |
| **4.4 Fehlererkennung:**  Die Wetterstation soll die Möglichkeit haben, im Falle eines Fehlers (Server Verbindung getrennt, keine Netzwerkverbindung usw.), dies mithilfe einer LED oder eines Displays zu signalisieren. | Verliert die Station die MQTT oder WLAN-Verbindung, wird dies mittels einer RGB LED signalisiert. Je nach Fehler, wird eine andere Farbe verwendet:  Rot blinkend: Kein WLAN.  Orange blinkend: MQTT Fehler | JA |
| **5.1 Technologien**  Für die Entwicklung sollen Produkte und Technologien nach aktuellem Stand der Technik eingesetzt werden. Gibt es für eine spezifische Anforderung bereits eine Library, kann diese sofern ausgewiesen verwendet werden. | Sämtlicher Code wurde nach bestem Wissen und Stand der Technik entwickelt. Die verwendeten Module können im Punkt 3 dieses Dokuments entnommen werden. | JA |
| **5.2 Sicherheit**  Sämtliche Systeme sollen nach aktuellem Stand der Technik abgesichert werden. Verbindungen, welche über das Internet aufgebaut werden, müssen mit Hilfe von SSL/TLS und gültigen Zertifikaten verschlüsselt sein, um ein Mitlesen oder Manipulieren der Daten seitens Dritter zu erschweren. | Das Node-Red UI und Grafana sind über HTTPs mit einem gültiges SSL Zertifikat gesichert.  Sämtliche MQTT Kommunikation zwischen Wetterstation und Node-Red (Mosquitto) ist per TLS gesichert. Lediglich mit der Zertifikatsprüfung gibt es noch Probleme, weshalb diese zum aktuellen Zeitpunkt deaktiviert ist. | Teilweise |
| **5.3 Hosting**  Weil der Kunde die Server nicht lokal betreiben möchte, sollen diese bei einem großen Cloud-Provider gehostet werden. Die Auswahl des Anbieters ist hierbei dem Entwickler überlassen. | Der MySQL Server, Grafana Server und Node-Red Server werden als separate Ubuntu VMs bei DigitalOcean gehostet. | JA |
| **5.4 Erweiterbarkeit**  Um Erweiterungen zu einem späteren Zeitpunkt zu ermöglichen, müssen sämtliche Software-Module möglichst modular und erweiterbar konzipiert werden. | Der Source-Code der Wetterstation wurde möglichst modular aufgebaut. Für verschiedene Aufgaben wurden eigene Klassen erstellt (Bspw. für das MQTT Handling).  Die DB wurde so konzipiert, dass mehrere Wetterstationen gleichzeitig Daten aufzeichnen können. | JA |

# Verwendete Produkte und Librarys

## Backend / Frontend

* [Node-Red](https://nodered.org/): Backend und Frontend UI.
  + [Node-red-dashboard](https://flows.nodered.org/node/node-red-dashboard): Diverse UI Elemente.
  + [Node-red-node-mysql](https://flows.nodered.org/node/node-red-node-mysql): Verbindungen zum MySQL Server.
  + [Node-red-contrib-counter](https://flows.nodered.org/node/node-red-contrib-counter): Baustein welcher einen Counter ausgibt.
* [MySQL](https://www.mysql.com/de/): DB Server zur Speicherung der Messwerte.
* [Grafana](https://grafana.com/): Visualisierung der MySQL Daten.
  + [DynamicText](https://grafana.com/grafana/plugins/marcusolsson-dynamictext-panel/): Modul zur darstellung von dynamischem Text basierend auf SQL Querys.

## ESP8266 Wetterstation Librarys

* [Adafruit BME680](https://github.com/adafruit/Adafruit_BME680): Library zum Auslesen der Sensorwerte.
* [ESP8266WiFi](https://github.com/esp8266/Arduino/tree/master/libraries/ESP8266WiFi): Library für WLAN und TLS Verbindungen.
* [PubSubCLient](https://github.com/knolleary/pubsubclient): Library für MQTT Verbindungen zwischen Node-Red und Wetterstation.
* [NTPClient](https://github.com/arduino-libraries/NTPClient): Library für Zeitsync per NTP.
* [ArduinoJson](https://github.com/bblanchon/ArduinoJson): Library zum parsen von Daten zu Json Objekten.

## ESP8266 Wetterstation Hardware

* [NodeMCU ESP8266](https://www.make-it.ca/nodemcu-details-specifications/): Hauptmodul
* [RGB LED](https://components101.com/diodes/rgb-led-pinout-configuration-circuit-datasheet): Status LED
* [BME 680](https://www.bosch-sensortec.com/media/boschsensortec/downloads/datasheets/bst-bme680-ds001.pdf): Wettersensor

# Anhang

* Plattform IO Projekt der Wetterstation
* NodeRed Konfiguration
* Grafana Konfiguration
* SQL Statements zur Erstellung der DB
* Pflichtenheft