Arduino RoomControlling

Inhaltsverzeichnis

[1.0 Auftrag 3](#_Toc2435785)

[1.1 Projektdaten und Versionierung 3](#_Toc2435786)

[2.0 Planung 3](#_Toc2435787)

[2.1 Zielsetzung 3](#_Toc2435788)

[2.1.1 Gesamtziel 3](#_Toc2435789)

[2.1.2 Teilziele 3](#_Toc2435790)

[2.2 Arbeitsergebnisse 4](#_Toc2435791)

[2.3 Arbeitsplan 4](#_Toc2435792)

[2.4 Verwendete Tools und Instrumente 4](#_Toc2435793)

[2.5 Struktur 5](#_Toc2435794)

[3.0 Hardware 5](#_Toc2435795)

[3.1 Microcontroller 5](#_Toc2435796)

[3.1.1 Pinout 6](#_Toc2435797)

[3.2 PCB 6](#_Toc2435798)

[3.3 HYT 939 Temperatur und Feuchtesensor 6](#_Toc2435799)

[3.3.1 Technische Daten 7](#_Toc2435800)

[4.0 Software 7](#_Toc2435801)

[4.1 Entwicklungsumgebung 7](#_Toc2435802)

[5.0 Web Development 8](#_Toc2435803)

[5.1 API 8](#_Toc2435804)

[5.1.2 Anbieter 9](#_Toc2435805)

[5.2 Datenbank 9](#_Toc2435806)

[5.2.1 Einführung 9](#_Toc2435807)

[5.2.3 Anbieter 9](#_Toc2435808)

[5.2.4 Struktur 9](#_Toc2435809)

[5.3 Grafana 10](#_Toc2435810)

[5.3.1 Einführung 10](#_Toc2435811)

[5.3.2 Anbieter 10](#_Toc2435812)

[5.3.3 Dashboard 11](#_Toc2435813)

[6.0 Arduino/Node MCU Code 11](#_Toc2435814)

[6.1 Funktionen 11](#_Toc2435815)

[6.1.1 void readData(void) 12](#_Toc2435816)

[6.1.2 writeDatabase 12](#_Toc2435817)

[6.2 Stromverbrauch 12](#_Toc2435818)

# 1.0 Auftrag

Im Fach Informatik muss ein Arduino Project erstellt werden. Zweck dieses Projektes ist es das organisierte durchführen von Projekten und das anwenden vom erlernten Stoff über Arduino Entwicklung.

Da wir in der Firma, in der ich arbeite, unter anderem Temperatur und Feuchtigkeit Sensoren entwickeln, habe ich mich entschieden ein System aufzubauen, welches Die Temperatur und Feuchtigkeit misst.

Das System basiert auf einem NodeMCU. Dieser liest ein HYT939 Feuchtesensor alle 60 Sekunden aus und lädt die daten auf einen Webserver hoch. Auf dem Server läuft eine MySQL Datenbank, auf der die Daten abgespeichert werden.

Damit der Benutzer jeder Zeit die Daten auslesen kann verwende ich Grafana. Auf Grafana kann man von überall zugreifen.

Das Projekt sollte ca. 50 Arbeitsstunden umfassen.

## Projektdaten und Versionierung

Für das verwalten der Daten habe ich das Versionierungs Tool «Github» verwendet. Dadurch ist es mir möglich den Überblick über die angepassten Dateien zu behalten. Zusätzlich habe ich ae da

Alle für das Projekt relevante Daten wie Code und Datenblätter können unter folgendem Link eingesehen werden.

<https://github.com/LoXeras/Arduino-RoomControlling>

# Planung

## 2.1 Zielsetzung

### 2.1.1 Gesamtziel

Das Ziel beinhaltet ein funktionierendes System zur Aufzeichnung, Übertragung und Visualisierung der gemessenen Temperatur- und Feuchtigkeitswerten. Die Daten sollen aus dem Internet überall verfügbar sein. Das Projekt muss mit einem Arduino oder ähnlichem Microcontroller basiertem System realisiert werden.

### 2.1.2 Teilziele

Konzept ausarbeiten  
PCB Designen

Arduino Code:

* Auslesen vom Sensor
* Kommunikation mit der Datenbank

Datenbankstruktur  
Visualisierung  
Präsentation  
Dokumentation

## 2.2 Arbeitsergebnisse

* Dokumentation
* Funktionierendes Arduino
* Datenbank
* Visualisierung im Internet
* Präsentation

## 2.3 Arbeitsplan

|  |  |  |
| --- | --- | --- |
| Arbeitsschritt | Ziel/ Arbeitsergebnis | Termin |
| Konzept Ausarbeiten | Grobe Struktur über das ganze Project | 20.12.18 |
| Prototyp aufbauen | Test Prototyp für die Entwicklung des Codes etc. | 24.12.18 |
| Arduino Code entwickeln | Arduino Code | 28.1.19 |
| Datenbank erstellen |  | 28.1.19 |
| Schnittstelle zu Datenbank |  | 18.2.19 |
| PCB Layouten | Layout | 22.2.19 |
| Web Visualisierung | Web Visualisierung | 15.3.19 |
| Präsentation |  | 4.3.19 |

## 2.4 Verwendete Tools und Instrumente

|  |  |  |
| --- | --- | --- |
| Arbeitsschritt | Methodik | Instrument |
| Bericht erstellen | Text | Word |
| Tabellen | Excel |
| Arbeit speichern und sichern | Elektronisch | PC/NAS/Github |
| Webentwicklung | Web code | Atom |
| Entwicklungsumgebung | Arduino | Visual Studios |
| PCB Layout | Layout | Circuit Studio |
| Präsentation | Präsentation | Power Point |

## 2.5 Struktur

NodeMCU + Sensor

Webserver

MySQL Database



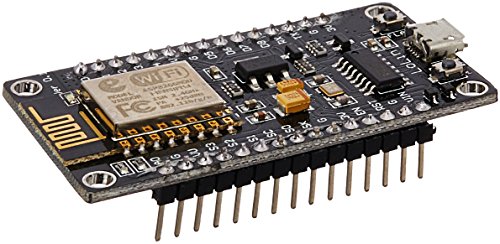
Das ganze System besteht aus 4 Elementen. Als liest der NodeMCO den Feuchtigkeits- und Temperatur Sensor aus. Diese Daten werden aufbereitet. Über das Aufrufen der API werden per GET die Werte am Web Server übergeben. Der Webserver macht anschliessend einen INSERT über die Datenbank.

Der Benutzer kann sich die Daten über Grafana anschauen unter dash.loxeras.com

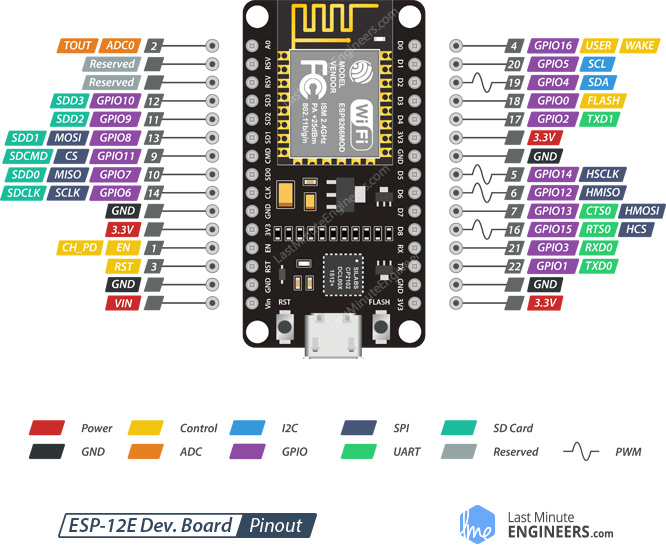
# 3.0 Hardware

## 3.1 Microcontroller

Als Microcontroller verwende ich einen NodeMCU. Dieser hat direkt ein WLAN Modul integriert und ist sehr kostengünstig und kompakt. Der Chip ist derselbe wie man auf dem Arduino Micro findet, er wird wie ein normales Arduino programmiert.



### 3.1.1 Pinout



## 3.2 PCB

Das PCB habe ich im Circuit Studios erstellt. Mit dem PCB wir der Feuchte Sensor mit dem NodeMCU verbinden. Dadurch ist alles sauber miteinander verbunden. Zusätzlich habe ich alle Pins Mit einem Header Verbunden damit man es noch erweitern kann.

Gefertigt habe ich das PCB bei [www.jlcpcb.com](http://www.jlcpcb.com). Für 2$ habe ich 10 PCBs bekommen, dadurch kann ich das System beliebig erweitern.

## 3.3 HYT 939 Temperatur und Feuchtesensor

Die wichtigste Aufgabe bei der Messung ist der stabile Nachweis eines korrekten Feuchtewerts und Temperaturwerts innerhalb einer angemessenen Ansprechzeit. Für diesen zweck verwende ich ein HYT 939 von der Firma IST AG da dieser sehr genau ist und direkt auch die Temperatur misst.

Der Sensor befindet sich in einem TO-36 Gehäuse und wir über I2C ausgelesen.

### 3.3.1 Technische Daten



# 4.0 Software

## 4.1 Entwicklungsumgebung

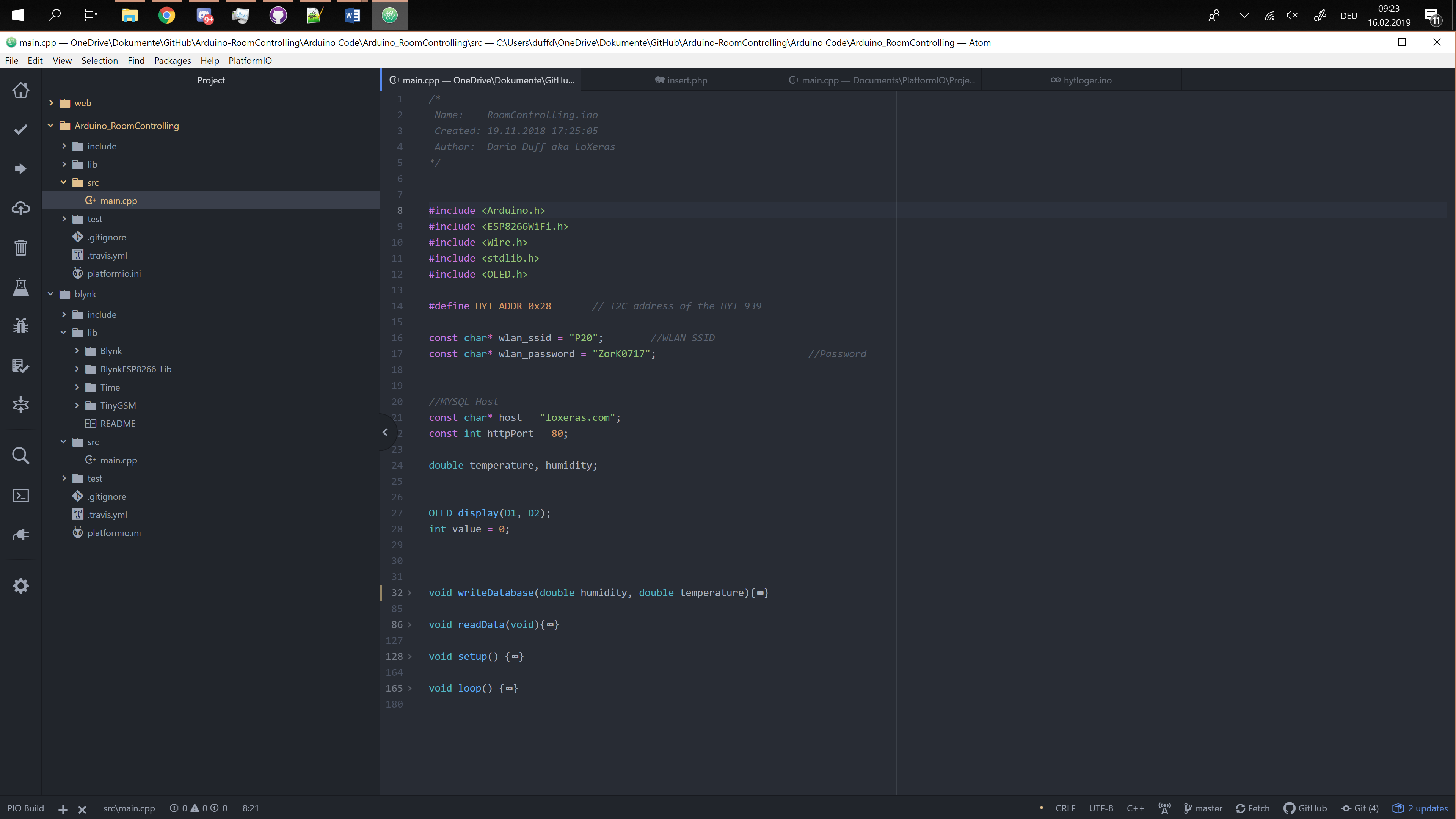
Für die Programmierung des Arduino Codes habe ich nicht die normale Arduino Entwicklungsumgebung verwendet, sondern Atom.io.

<https://atom.io/>

Atom.io nennt sich “A hackable text editor for the 21st Century”. Der Editor ist für sehr viele Programmiersprachen geeignet und kann mit diversen Erweiterungen erweitert werden.

Für das Programmieren vom NodeMCU wird die Erweiterung PlattforIO verwendet. Diese stellt alles Wichtige für das Programmieren des Arduino zur Verfügung. Unter anderem einen besseren Compiler, auto complete, Serial monitor und eine Abdeckung von 588 Bards.

<https://platformio.org/>



# 5.0 Web Development

## 5.1 API

Aus Sicherheitsgründen habe ich den genauen Pfad der API zensiert. Später werde ich mir noch eine Authentifizierung für die einzelnen NodeMCUs programmieren.

Die API erlaubt die Kommunikation zwischen der Datenbank und dem NodeMCU. Hierzu dient ein PHP Skript, welches mit der Datenbank verbunden ist. Die Temperatur und Feuchtigkeit wir mit der Room ID per PHP GET übergeben. Das Script macht dann anschliessend den Insert in die Datenbank.

Die API lässt sich unter loxeras.com/xxx/insert.php mit den geforderten Parametern aufrufen

Aufstellung der GET Parameter:

t: Temperature  
h: humidity  
r: room  
  
Der Parameter «room» wird für die Identifizierung des Raumes verwendet. Jedes Gerät kriegt seine eigene Room ID. Falls mehrere Geräte in einem Raum sind werden die so gekennzeichnet.

Im File sieht due Zuweisung so aus:

$temperature=$\_REQUEST['t'];

$humidity=$\_REQUEST['h'];

$room=$\_REQUEST['r'];

Anschliessend macht das PHP script den Insert in die Datenbank.

INSERT INTO environment(room,temperature,humidity) VALUES ('$room','$temperature','$humidity')

### 5.1.2 Anbieter

Die API läuft mit dem PHP Scripts auf meinem Webspace, der bei Nitrado gemietet wurde. Diese bieten diverse Grössen von Webspaces an. Da sie Externen zugriff auf die Datenbank zulassen waren sie meine Wahl.

## 5.2 Datenbank

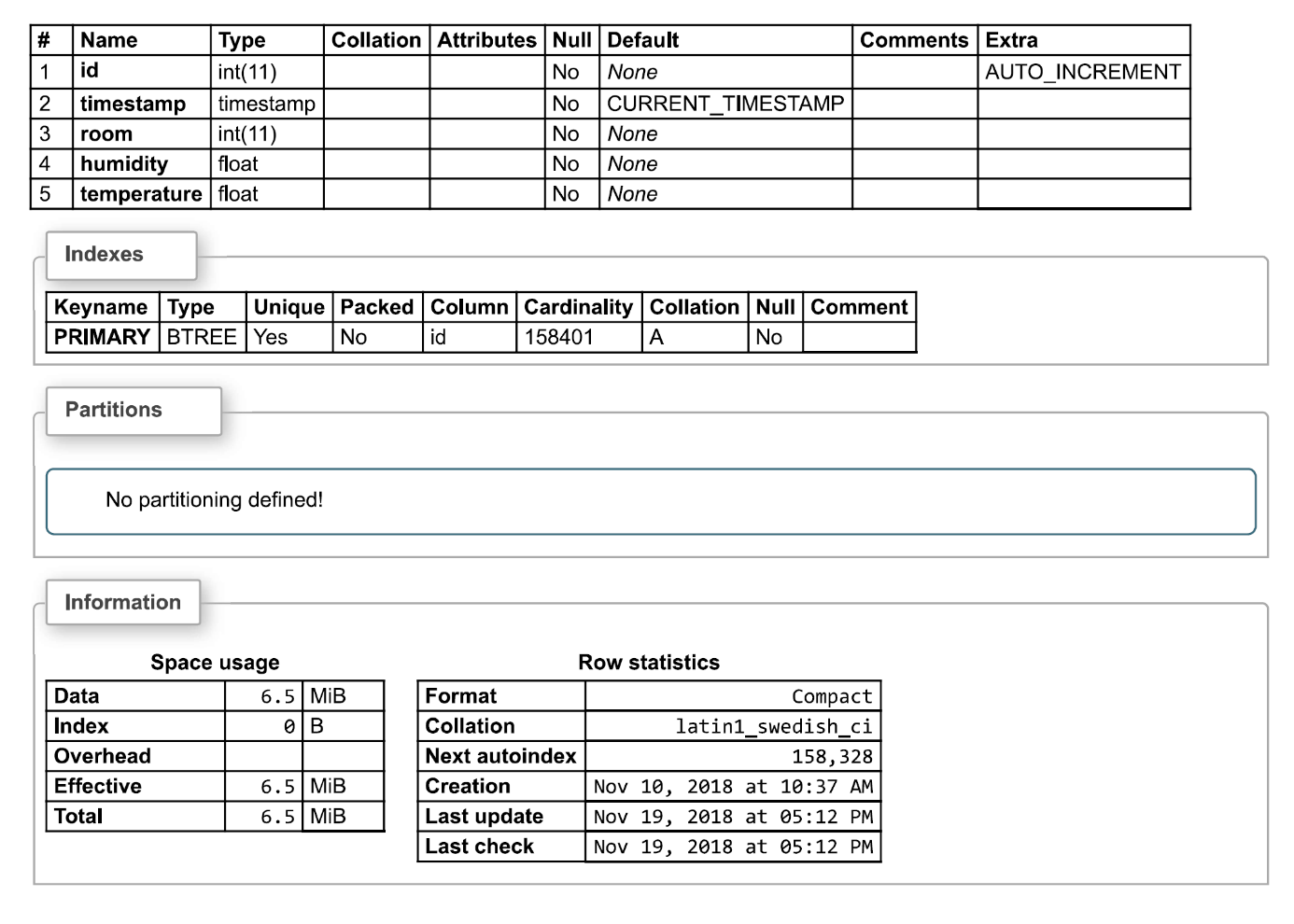
### 5.2.1 Einführung

Als Datenbank verwende ich MySQL, als web interface PhpMyAdmin. Die Datenbank ist dazu da um alle Werte abzuspeichern. Es fallen sehr schnell sehr viele werte an, dadurch ist eine Datenbank die Optimale Lösung für das abspeichern.

### 5.2.3 Anbieter

Die Datenbank läuft genau wie der Webspace bei Nitrado. Da die Datenbank externen zugriff erlaubt ist ein sehr langes Passwort wichtig, ausserdem haben alle Nutzer von aussen nur Lese Recht und kein Schreibrecht.

### 5.2.4 Struktur

Die Datenbank ist wie folgt strukturiert: 

## 5.3 Grafana

### 5.3.1 Einführung

Grafana ist ein Open source Monitoring und Analyse System. Es bietet diverse Darstellungsmöglichkeiten an. Ausserdem kann man die Daten von diversen Datenquellen auslesen und verarbeiten. In meinem Fall geschieht das alles über MY SQL.

### 5.3.2 Anbieter

Grafana wird auf einem VPS gehostet bei der Firma Hetzner. Der Server verfügt über folgende Eigenschaften:

1 vCPU  
2GB RAM  
20 GB HDD

Insgesamt ist 20TB Traffic im Monat inbegriffen, was für meine Zwecke niemals erreicht werden kann.

Auf dem Server ist Ubuntu 18.04 installiert.

### 5.3.3 Dashboard



Alle erfassten Daten werden in einem Dashboard angezeigt. Für die Temperatur und die Feuchtigkeit habe ich jeweils ein Temperatur/Zeit Diagramm erstellt. Auf der rechten seite des jeweiligen Diagramms können Minimum und maximum Temperatur in dem gewählten Zeitbereich abgelesen werden.

Zusätzlich werden die Mommentanwerte oben links angezeigt.

Im Feuchtigkeit Graph kann sehr gut erkannt werden, wann ich geduscht habe.

Oben rechts kann der Zeitraum gewählt werden, von dem man die werte sehen will. Standartmässig habe ich ihn so eingestellt, dass ich die Daten von Huetigen Tag habe. Alle 60 Sekunden werden die Daten aktualisiert.

Das Dashboard ist unter dash.loxeras.com verfügbar.

# 6.0 Arduino/Node MCU Code

## 6.1 Funktionen

### 6.1.1 void readData(void)

Diese Funktion liest die Daten von Feuchtigkeit und Temperatursensor aus. Es werden 4 Bytes von Sensor angefragt, falls der NodeMCU eine Antwort bekommt, werden die daten verarbeitet.

Die ersten 2 Bytes gehören zur Feuchtigkeit die letzten zwei zur Temperatur.

### 6.1.2 writeDatabase

Hier werden die Temperatur und die Feuchtigkeit indirekt der Datenbank übergeben. Das ganze pasiert über die API die im punkt 5.1 Beschrieben wird.

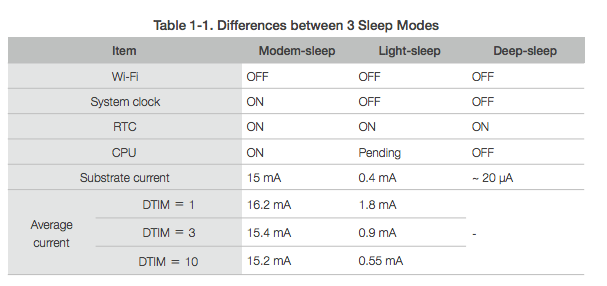
Für die Kommunikation wird die Verbindung mit dem Internet hergestellt. Danach wrrden die Parameter per GET an das File übergeben. Zur Veranschaulichung habe ich hier ein Beispiel.

loxeras.com/xxx/insert.php?r=99&h=16.9&t=+25.67

Es wird die Raum ID, die Feuchtigkeit und die Temperatur übergeben.

## 6.2 Stromverbrauch

Da man normalerweise mehrere Geräte pro Wohnung hat, die andauernd am Strom sind sollte der Stromverbrauch möglichst gering sein. Am meisten Strom verbraucht das WLAN Modul. Beim Senden von daten benötigt es ungefähr 150mA. Als Lösung gibt es 3 verschiedene Varianten von «Sleep Modes»



In meinem Projekt soll das Gerät nur aktiv sein, wenn auch gemessen bzw. gesendet werden soll.

Aus diesem Grund habe ich mich für den Deep-Sleep Mode entschieden. Sobald man den NodeMCU in den Deep-Sleep Mode sendet ist nur noch der RTC (Real time Clock) aktiv. Sobald die eingestellte zeit vergangen ist wird das Modul resetted und das Programm beginnt von vorne.

Im Deep-Sleep Mode braucht der NodeMCU ungefähr 20uA. Das ergibt einen Verbrauch von ungefähr 0.876 Watt pro Jahr. Natürlich wenn das Modul die ganze Zeit im Deep-Sleep Mode ist.