# Arduino Wetterstation

Sara Billing

M242 MIKROPROZESSORANWENDUNG REALISIEREN

M242
Arduino Wetterstation
Sara Billing

# Funktionsbeschrieb

In diesem Projekt setze ich eine Arduino Wetterstation um, die Temperatur, Luftfeuchtigkeit und Luftdruck misst und auf einem LCD anzeigt. Zusätzlich werden das Datum und die Zeit angezeigt mithilfe einer RTC. Die Daten werden als HTTP Request an meinen PHP Endpoint geschickt, welcher sie in einer SQL-Datenbank abspeichert.

# Use Cases

## Use Case Wetterdatenanzeige

| **Name** | **Wetterdaten-Anzeige** | |
|---|---|---|
| Akteur | BME 280 Sensor (Luftdruck-, Luftfeuchtigkeits- und Temperatursensor), LCD | |
| Trigger | Keine | |
| Kurzbeschreibung | Es werden der momentane Luftdruck, Luftfeuchtigkeit und die Temperatur gemessen und anschliessend auf der linken Seite des LCD angezeigt. | |
| Vorbedingungen | Keine | |
| | | |
| Komponenten | GUI | |
| Essenzielle Schritte | ***Intention der Systemumgebung*** | ***Reaktion des Systems*** |
| | Werte werden gemessen | |
| | Werte werden im Code gelesen und auf LCD ausgegeben | LCD zeigt Wetterdaten an |
| | | |
| | | |
| | | |
| | | |
| | | |
| Ausnahmefälle | | |
| Nachbedingung | | |
| Zeitverhalten | Die LCD Reaktionszeit | |
| Verfügbarkeit | | |

M242
Arduino Wetterstation
Sara Billing

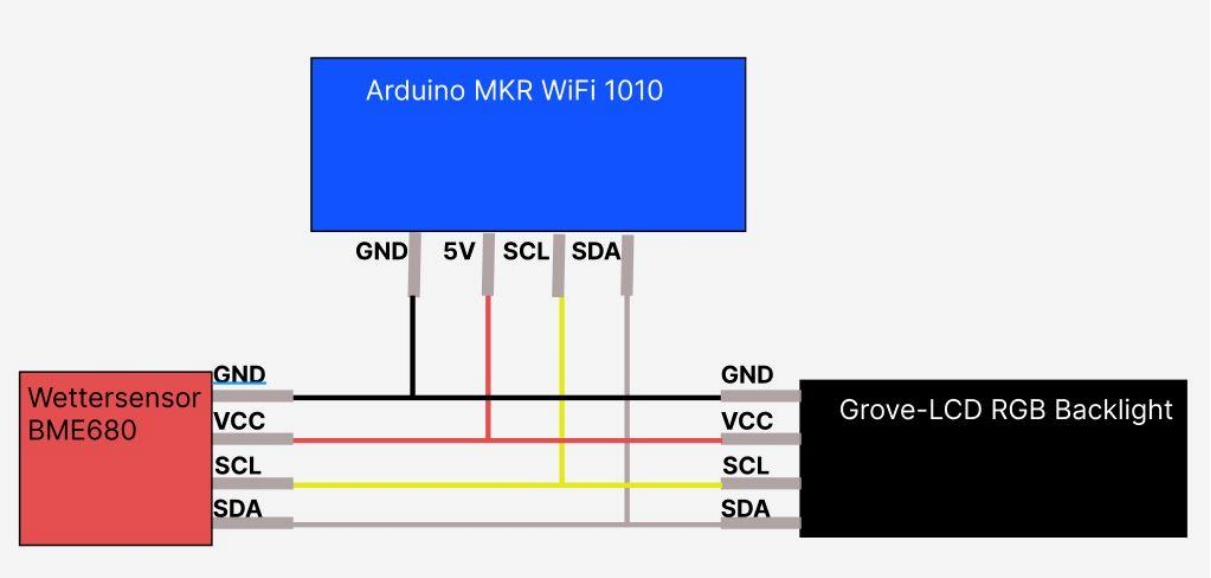
## Use Case Uhrzeitanzeige

| **Name** | **Uhrzeit-Anzeige** | |
|---|---|---|
| Akteur | Real Time Clock, LCD | |
| Trigger | keine | |
| Kurzbeschreibung | Rechts oben auf dem LCD wird die Uhrzeit angezeigt. Die Uhrzeit-Anzeige wird korrekt aktualisiert. | |
| Vorbedingungen | Keine | |
| | | |
| Komponenten | GUI | |
| Essenzielle Schritte | ***Intention der Systemumgebung*** | ***Reaktion des Systems*** |
| | Uhrzeit vom Sensor wird eingelesen | |
| | Uhrzeit wird auf LCD ausgegeben | LCD zeigt Uhrzeit an |
| Ausnahmefälle | | |
| Nachbedingung | | |
| Zeitverhalten | Die LCD Reaktionszeit | |
| Verfügbarkeit | | |
| Folge Use Case | | |

## Use Case Datenspeicherung

| **Name** | **Daten-Speicherung** | |
|---|---|---|
| Akteur | BME 280 Sensor, Real Time Clock, WiFi Modul | |
| Trigger | Wetterdatenmessung | |
| Kurzbeschreibung | Bei jeder Messung der Wetterdaten werden die Daten an einen SQL-Server geschickt und in einer Datenbank gespeichert. Wenn dies Erfolgreich ist, zeigt das LCD für 5 Sekunden «Daten gespeichert», sonst zeigt es «Fehler beim Speichern». | |
| Vorbedingungen | Internetverbindung | |
| | | |
| Komponenten | SQL-Datenbank-Verbindung, GUI | |
| Essenzielle Schritte | ***Intention der Systemumgebung*** | ***Reaktion des Systems*** |
| | Wetterdaten werden gemessen | Daten werden eingelesen |
| | Sendet Daten an SQL-Server | Daten werden in Datenbank gespeichert und LCD zeigt «Daten gespeichert» |
| | | Daten konnten nicht gespeichert werden und LCD zeigt «Fehler beim Speichern» |
| Ausnahmefälle | Keine Internetverbindung → zeigt «Fehler beim Speichern» | |
| Nachbedingung | | |
| Zeitverhalten | Zeit, in welcher die Daten an den Server gesendet werden und eine Antwort zurückkommt | |
| Verfügbarkeit | | |

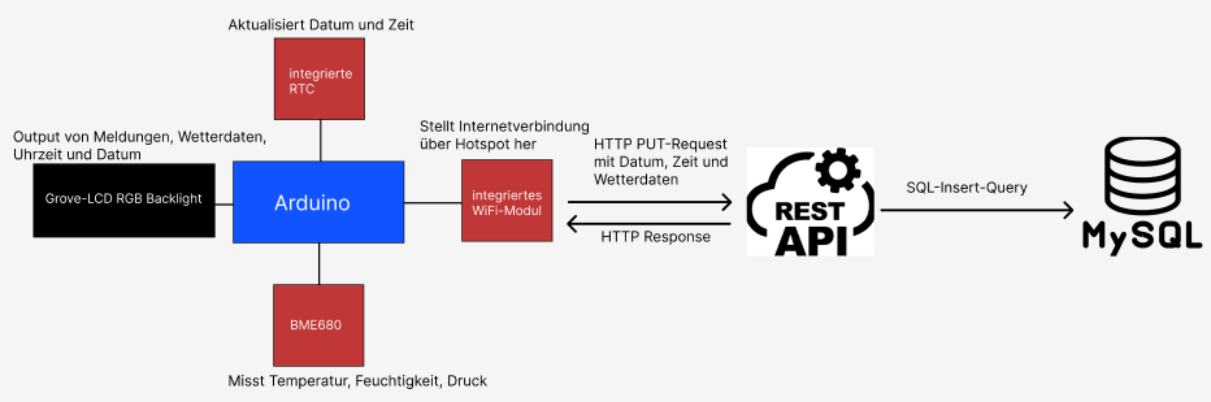
## Blockschaltbild




## Architektur

Die Architektur sieht folgendermassen aus:

- Integriertes WiFi-Modul: Stellt beim Aufstarten die Internetverbindung über meinen Mobile Hotspot auf. Fragt ein einziges Mal das aktuelle Datum und die aktuelle Uhrzeit über das Internet ab.
- Integrierte RTC: Aktualisiert das Datum und die Uhrzeit.
- BME680: Misst alle 30 Sekunden Temperatur, Feuchtigkeit und Druck.
- Integriertes WiFi-Modul: Nach jeder Messung wird ein HTTP PUT-Request mit den Daten an meine PHP REST-API geschickt.
- PHP REST-API: Hier wird eine SQL-Insert-Query mit den Wetterdaten, dem Datum und der Uhrzeit an den MySQL-Server geschickt.
- MySQL-Server: Speichert die Daten in meiner Datenbank
- PHP REST-API: Nach dem Speichern wird eine HTTP-Response zurückgeschickt an das Wifi-Modul.
- Arduino: Analysiert die HTTP-Response, bereitet Daten auf.
- LCD: Zeigt das Datum, die Uhrzeit, die Wetterdaten, sowie verschiedene Meldungen



M242
Arduino Wetterstation
Sara Billing

# Test Cases

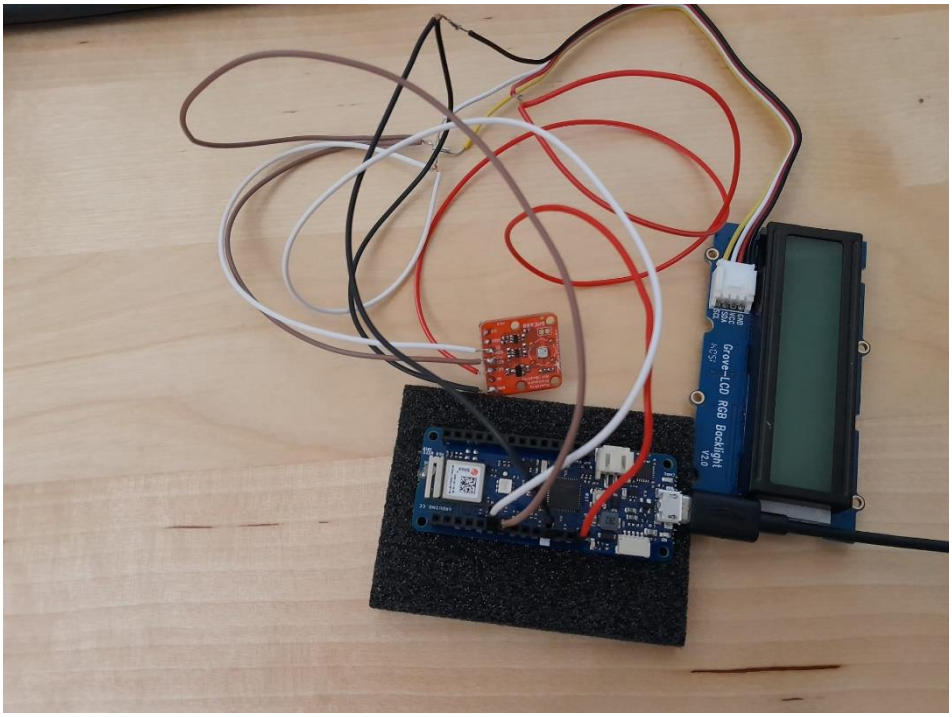
## Vorbedingungen

Mobile Hotspot, Webserver und SQL Server laufen

| Nr. | Was? | Wie? | Erwartet | Resultat |
|---|---|---|---|---|
| 1 | Zeitanzeige | - LCD Uhrzeit vergleichen mit Computer Uhrzeit<br>- 5 Minuten warten<br>- Nochmal vergleichen | Zeit wird angezeigt mit nicht mehr als 3 Sekunden Verzögerung.<br>Eine geringe Abweichung von 1 – 3 Sekunden ist zu erwarten, weil es noch einen Moment geht bis der Code ausgeführt ist. | Es hat eine Verzögerung von ca. 2 Sekunden und nach 5 Minuten ist die Verzögerung immer noch gleich. |
| 2 | Wetterdaten | - Temperatur und Feuchtigkeit<br>- Warten bis zu einer halben Minute<br>- Schauen ob neue Daten angezeigt werden | Jede halbe Minute wird eine neue Messung gemacht und die Daten werden angezeigt | Es werden alle 30 Sekunden die neuen Daten angezeigt. |
| 3 | Wetterdaten | - Bis kurz vor der Messung warten<br>- Sensor anhauchen | Temperatur und Feuchtigkeit sollten ein bisschen ansteigen | Beides ist ein bisschen angestiegen. |
| 4 | Daten-Speicherung | - Auf Messung warten<br>- In Datenbank nachschauen | Neuer Eintrag in der Datenbank, Erfolg-Meldung auf LCD | Die Daten werden gespeichert und es steht «Erfolgreich gespeichert». |
| 5 | Fehler-Meldung | - Im Code falsche URI angeben für Webserver | Fehlermeldung auf LCD | Es kommt die Meldung «Fehler beim Speichern» |

## Umsetzung

Als Erstes habe ich die einzelnen Komponenten über ein Breadboard mit dem Arduino verbunden. Die RTC, den Wettersensor und das LCD habe ich mit kleinen Testprogrammen ausprobiert. Als ich mir sicher war, dass alles funktioniert, habe ich die Kabel angelötet:




Danach habe ich Schritt für Schritt meine Software geschrieben.

1. Zuerst habe ich nur die Uhrzeit und das Datum angezeigt
2. Dann habe ich zusätzlich die Wetterdaten angezeigt
3. Als nächstes musste ich für mein LCD Scrolling implementieren, um all meine Daten anzeigen zu können
4. Danach habe ich eine Datenbank erstellt und PHP einen Endpoint aufgesetzt, der Daten in die Datenbank schreiben kann
5. Das Schwierigste war, den HTTP Request im Arduino Code korrekt aufzubauen, hat dann aber mit einem PUT Request funktioniert
6. Dann habe ich LCD-Meldungen implementiert

## Meldungen

### Bei Datenmessung

Diese Meldung wird angezeigt, wenn gerade eine Messung stattfindet.


### Bei Speicherung

Dies wird angezeigt, wenn die Daten gespeichert werden.


### Erfolg

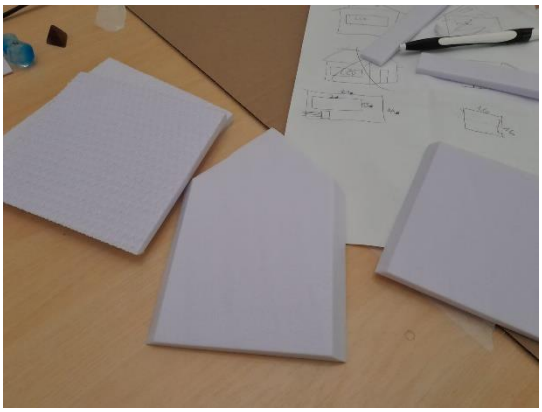
Wurden die Daten erfolgreich gespeichert, wird das angezeigt.


### Fehler

Gab es einen Fehler beim Speichern, wird das angezeigt.

## Hülle

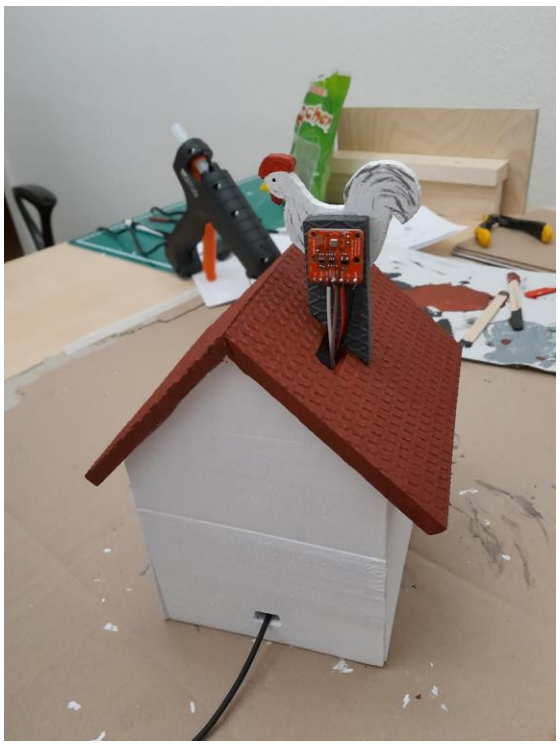
Für die Hülle habe ich aus Styropor ein Wetterhaus gemacht.



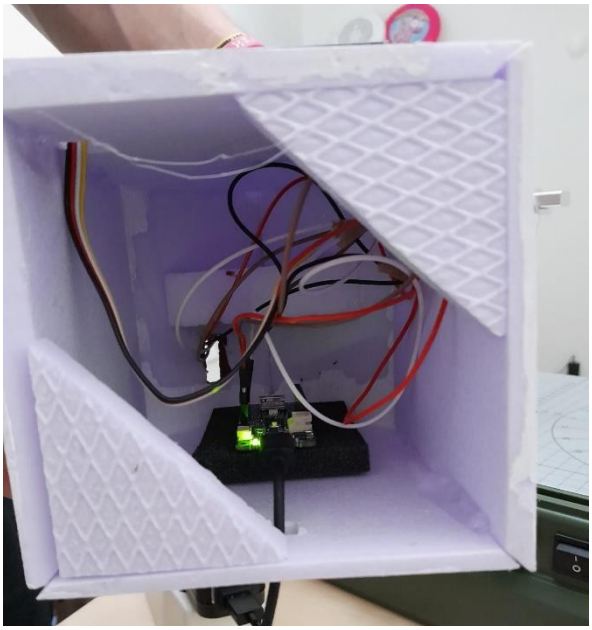
## Endresultat

Das Endresultat sieht so aus: Das LCD ist an der Vorderseite des Hauses, der Arduino ist im Haus versteckt und der Sensor befindet sich hinter dem Wetterhahn.

### Vorder- und Rückseite

### Innen



## Reflexion

Ich wusste von Anfang an, dass es ein paar Holpersteine geben wird, da ich es mir schwierig vorgestellt habe, einerseits die Uhrzeit korrekt zu aktualisieren und andererseits die Wetterdaten in einer Datenbank abzuspeichern. Vor allem beim letzten Punkt musste ich einige Zeit lang ausprobieren, bis ich es geschafft habe, meine Daten über einen HTTP Request an meinen PHP Endpoint zu schicken. Aber schlussendlich konnte ich doch noch all meine Anforderungen erfüllen. Das Projekt war somit eine spannende und erfolgreiche Herausforderung für mich.