AirGuardX

Benedikt Scholz  
Faculty of Electrical Engineering and Informatics  
University of Applied Sciences and ArtsCoburg, Germany  
Benedikt.Scholz@stud.hs-coburg.deYannik Wolf  
Faculty of Electrical Engineering and Informatics  
University of Applied Sciences and ArtsCoburg, Germany  
Yannik.Wolf@stud.hs-coburg.de

*Zusammenfassung* — Dieses Dokument beschreibt den Aufbau und die Durchführung eines Projekts im Rahmen des Moduls Hardware cyber-physischer Systeme im Masterstudiengang Informatik an der Hochschule für angewandte Wissenschaften in Coburg. Ziel des Projekts ist die Erstellung eines Projektaufbaus, der Sensordaten erfassen, sammeln und auswerten kann. Dieses Projekt beschäftigt sich konkret mit der Messung von Raumdaten wie Temperatur und Luftfeuchtigkeit. Die Daten werden mit einem Sensor, der an einem Raspberry Pi angeschlossen ist, erfasst. Per HTTP-Schnittstelle werden die Daten an einen Webserver übertragen. Der Webserver sammelt die Daten in einer Datenbank und liefert eine Webanwendung aus, über die der aktuelle Zustand des Raumes jederzeit ersichtlich ist. Bei Bedarf wird ein Nutzer per Telegram-Nachricht kontaktiert, um Maßnahmen wie das Belüften des Raumes einzuleiten, um Schimmelbildung sowie sonstige Schäden am Gebäude zu verhindern. Besonders die Gesundheit der Bewohner kann durch gezieltes Lüften und eine gute Luftqualität in Innenräumen gesichert werden.

Schlagworte—airguardx, raspberry pi, scd-30, oracle cloud, spring boot application, i2c, telegram (bot), weather api

# Einleitung

Eine Idee zur Schimmelprävention und Erhaltung der Luftqualität ist die Verwendung von IoT-Sensoren und Cloudservices. Diese messen die Luftfeuchtigkeit, Temperatur, CO, CO2, Feinstaub und VOCs in einer Wohnung beziehungsweise einem Haus. Primär stehen dabei die Gesundheit der Menschen im Haushalt und die Erhaltung der Gebäudequalität im Fokus.

Schimmelbildung tritt häufig auf, wenn die Luftfeuchtigkeit zu hoch ist. CO entsteht bei unvollständiger Verbrennung, beispielsweise bei einem Gaskocher in der Küche, und wirkt in größeren Mengen als starkes Atemgift. CO2 wird von Menschen und Tieren ausgeatmet und kann sich bei einer zu hohen Menge auf die Konzentrationsfähigkeit auswirken. Ähnlich ist es bei VOCs. Diese beeinträchtigen ebenfalls die Konzentrationsfähigkeit und verursachen Kopfschmerzen, Müdigkeit und Schlafstörungen. Zuletzt steht noch der Feinstaub. Dieser wird

durch das Abbrennen von Kerzen, Staubsaugen, Kochen und Drucken im Gebäude verteilt und wirkt sich negativ auf die Atemwege aus. Eine regelmäßige Überwachung und Kontrolle dieser Parameter können dazu beitragen, gesundheitlichen Problemen des Menschen vorzubeugen und Schäden in den Gebäuden zu verhindern.

Die IoT-Sensoren werden in mehreren Räumen installiert und sammeln Daten zu den oben genannten Parametern. Diese Daten werden über eine Cloud-Plattform an eine zentrale Anwendung gesendet, die Warnungen und Handlungsempfehlungen per Messenger (Telegram) ausgibt, wenn ein Parameter ein bestimmtes Niveau erreicht oder überschreitet. Eine Übersicht über die aktuelle Luftsituation der einzelnen Räume, kann zudem über eine Web-Oberfläche

abgefragt werden, die den zeitlichen Verlauf der Sensor-Daten berücksichtigt und dadurch Rückschlüsse ziehen kann.

Zusammenfassend lässt sich sagen, dass eine IoT-Lösung zur Schimmelprävention und Erhaltung der Luftqualität die Verwendung von Sensoren, Cloudservices und intelligenten Geräten zur Überwachung und Kontrolle von Luftfeuchtigkeit, Temperatur und Luftqualität umfassen würde, um sicherzustellen, dass ein Raum und die Menschen darin sicher und gesund sind.

# Strategien

## Berechnungen und Logik auf dem Raspberry Pi

Warum sollte man das machen?

## Berechnungen und Logik auf dem Webserver

Warum sollte man das machen?

# Sensoren

## Sensorübersicht

### MOX = Metalloxid

* Mithilfe eines MOX-Sensors können VOCs ermittelt werden. Oft gibt der Sensor aber nur einen Wert zurück und kann die Gase nicht einzeln be-stimmen.
* Sensoren benötigen meist „Warmlaufzeit“ von bis zu 48h und vor jeder weiteren Nutzung 30min im Vorlauf

### VOC = Volatile organic compounds [

* Kohlenmonoxid (CO), Wasserstoff (H2)
* Stickstoffmonoxid (NO), Stickstoffdioxid (NO2), Ozon (O3), Ammoniak (NH3), Schwefeldioxid (SO2)
* Kohlenwasserstoffe (CXHY von C1-C8), z.B. Methan (CH4), Propan (C3H8)
* Alkohole (CXHYOH), z.B. Ethanol (C2H5OH) etc., weitere ausgewählte Volatile Organic Compounds (VOCs), Kältemittel (R134a etc.)

### NDIR = Nondispersive infrared sensor (Gas-Sensor)

* oft genauer als MOX-Sensor

## Sensorauswahl

In der engeren Auswahl der potentiell möglichen Raumluftsensoren stehen der Adafruit SCD-30 und der Adafruit BME680. Da in der Hochschule Coburg bereits der Adafruit BME680 vorhanden und verwendbar ist, fällt die Entscheidung auf diesen. Im Laufe des Projektes wurde er allerdings durch den SCD30 ersetzt.

Zusätzlich soll der Adafruit PMSA003I als Feinstaubsensor zum Einsatz kommen. Da dessen Beschaffung allerdings mit Zeit- und Geldaufwand verbunden ist, wird zunächst auf dessen Inbetriebnahme verzichtet.

# Architekturübersicht

Die nachfolgende Abbildung zeigt den groben Aufbau der Gesamtarchitektur des Projektes AirGuardX. Zu Beginn sei angemerkt, dass AirGuardX mit dem Projekt SmartStall Überschneidungspunkte aufweist. Beide Projekte nutzen dieselbe Cloud, um ihre Daten abzulegen und in Form einer Webanwendung zu repräsentieren. Die Webanwendungen werden aller-dings getrennt voneinander entwickelt. Die gemeinsame Nutzung der Cloud resultiert in der Ähnlichkeit der Projekte. Beide versuchen anhand von Sensordaten die Luftqualität zu messen und Handlungsanweisungen auszusprechen. AirdGuardX tut dies für den privaten Haus-halt, während SmartStall auf landwirtschaftlich genutzte

Tierställe abzielt. Das Ziel soll es sein, beide Projekte am Ende zusammenzuführen, um ein gemeinschaftliches, vollumfängliches All-In-One-Produkt für private als auch gewerbliche Zwecke anbieten zu können.

Beim Projekt AirGuardX werden die in der Tabelle aus III. zu sehenden Messdaten des Adafruit BME680 erhoben und über I2C an den Raspberry Pi, mit welchem der Sensor verbunden ist, übertragen. Der Raspberry Pi nutzt dann WLAN als Gateway, um die Daten alle 5 Minuten über das HTTP-Protokoll an die IP-Adresse des Webservers zu senden. Dort werden die Daten in einer Datenbank abgelegt. Sollten bestimmte Messwerte überschritten werden, wird vom Webserver automatisch eine Push-Benachrichtigung über Telegram an den Nutzer gesendet und eine rote LED leuchtet an der Messtation auf. Solange die Messwerte in einem normalen Bereich liegen, leuchtet ein grünes Licht. Des Weiteren steht auf dem Webserver noch eine Webanwendung zur Verfügung. Über diese kann der Zustand der Räume monitort und nachvollzogen werden.

Die Beschreibung der Arbeitsweise des SmartStall ist in deren gesonderter Dokumentation zu finden.

# Grundlagen verwendeter Technologien

## Raspberry Pi

Ein Raspberry Pi ist ein kleiner, kostengünstiger Einplatinencomputer, der von der Raspberry Pi Foundation entwickelt wurde. Mittlerweile hat sich der Raspberry Pi jedoch zu einem beliebten Gerät für viele Anwendungen entwickelt, darunter Heimautomatisierung, Mediacenter, Robotik, IoT-Projekte und vieles mehr.

Der Raspberry Pi besteht aus einer einzigen Platine, die alle Hauptkomponenten eines Computers enthält, darunter Prozessor, Arbeitsspeicher, Speicher, Anschlüsse und Erweiterungsschnittstellen. Es gibt verschiedene Modelle des Raspberry Pi, die sich in Bezug auf Leistung, Speicher, Anschlüsse und Funktionen unterscheiden. Die meisten Modelle verwenden eine ARM-Architektur und laufen mit einem Linux-basierten Betriebssystem, wie zum Beispiel Raspbian, das auf Debian basiert.

Der Raspberry Pi verfügt über eine Vielzahl von Anschlüssen, darunter USB-Ports, HDMI- und Audioausgänge, Ethernet-Anschluss, GPIO-Pins (General Purpose Input/Output) und Kameraanschlüsse. Dadurch kann er mit verschiedenen Geräten und Sensoren verbunden werden, um komplexe Projekte umzusetzen.

## I2C

I2C steht für Inter-Integrated Circuit und ist ein serieller Kommunikationsbus, der entwickelt wurde, um die Kommunikation zwischen integrierten Schaltkreisen (ICs) zu ermöglichen. Er wird häufig zur Verbindung von Komponenten in elektronischen Geräten verwendet, insbesondere in Mikrocontrollern und eingebetteten Systemen.

I2C verwendet zwei Signalleitungen: eine Datenleitung (SDA - Serial Data Line) und eine Taktleitung (SCL - Serial Clock Line). Diese Signalleitungen ermöglichen die bidirektionale Kommunikation zwischen einem Master-Gerät und einem oder mehreren Slave-Geräten. Der Master-Gerät initiiert die Kommunikation und kontrolliert den Ablauf, während die Slave-Geräte auf Anfragen des Masters reagieren.

Das I2C-Protokoll ermöglicht es, eine große Anzahl von Geräten über eine einzige Busleitung zu verbinden. Jedes Gerät auf dem Bus hat eine eindeutige Adresse, die verwendet wird, um es zu identifizieren und anzusprechen. Dadurch können mehrere ICs miteinander kommunizieren, ohne dass jedes Gerät eine separate Verbindung zum Master benötigt.

Die Kommunikation auf dem I2C-Bus erfolgt in Form von Datenpaketen, die in Start- und Stop-Bedingungen eingebettet sind. Der Master initiiert die Kommunikation mit einer Start-Bedingung und gibt die Adresse des Slave-Geräts an, mit dem er kommunizieren möchte. Dann erfolgt der Datenaustausch, bei dem der Master Daten sendet oder vom Slave empfängt. Am Ende der Kommunikation wird eine Stop-Bedingung gesendet.

I2C bietet eine einfache und effiziente Möglichkeit für die Kommunikation zwischen ICs und ermöglicht es Geräten, Informationen auszutauschen, Befehle zu senden oder Sensordaten zu übertragen. Es ist ein weit verbreitetes Kommunikationsprotokoll in der Elektronik und findet Anwendung in verschiedenen Bereichen wie der Steuerung von Peripheriegeräten, Sensoranbindungen, Displayansteuerung und vielen anderen Anwendungen. Im Falle dieses Projektes ist es die Schnittstelle, über welche die Daten vom Sensor an den Raspberry Pi übertragen werden.

## WLAN

WLAN steht für Wireless Local Area Network. Es handelt sich um eine drahtlose Netzwerktechnologie, die es Geräten ermöglicht, miteinander zu kommunizieren und auf das Internet zuzugreifen, ohne dass physische Verbindungskabel erforderlich sind.

Ein WLAN basiert auf dem IEEE 802.11-Standard und nutzt Funkwellen, um Daten zwischen Geräten zu übertragen. Die Geräte, die WLAN nutzen möchten, müssen mit einem drahtlosen Netzwerkadapter ausgestattet sein, der in der Lage ist, die Signale zu empfangen und zu senden. Typische Geräte mit WLAN-Funktion sind Laptops, Smartphones, Tablets, Smart-TVs und viele andere internetfähige Geräte.

WLAN bietet eine bequeme und flexible Möglichkeit, Geräte drahtlos zu vernetzen und auf das Internet zuzugreifen. Es ermöglicht die Mobilität der Geräte innerhalb der Reichweite des Netzwerks und hat sich zu einer weit verbreiteten Technologie entwickelt, die in privaten Haushalten, Büros, öffentlichen Bereichen und vielen anderen Umgebungen genutzt wird. Im Falle dieses Projektes wird es als Gateway benutzt, um die Sensordaten vom Raspberry Pi über das HTTP-Protokoll an den Webserver und somit ins Internet zu senden.

## Gateway

Ein Gateway ist eine Netzwerkgerät, das als Schnittstelle zwischen verschiedenen Netzwerken dient. Es ermöglicht die Kommunikation zwischen unterschiedlichen Netzwerkprotokollen, Netzwerktypen oder Netzwerksegmenten, die ansonsten nicht direkt miteinander kommunizieren könnten.

Ein Gateway fungiert als Übersetzer, der Datenpakete von einem Netzwerkformat in ein anderes umwandelt, um die reibungslose Kommunikation zwischen den Netzwerken zu ermöglichen. Es übersetzt die Protokolle, Adressierungen und andere Netzwerkparameter, sodass die Daten korrekt von einem Netzwerk zum anderen übertragen werden können.

Es gibt verschiedene Arten von Gateways, die je nach den spezifischen Anforderungen und Konfigurationen des Netzwerks unterschiedlich sein können.

WLAN, welches in diesem Projekt als Gateway genutzt wird, ist als Netzwerk-Gateway einzustufen. Netzwerk-Gateways verbinden Netzwerke unterschiedlicher Typen miteinander, beispielsweise ein lokales Ethernet-Netzwerk mit einem drahtlosen WLAN-Netzwerk oder einem Mobilfunknetzwerk.

## HTTP-Protokoll

HTTP steht für Hypertext Transfer Protocol. Es handelt sich um ein Protokoll, das zur Über-tragung von Daten über das Internet verwendet wird. HTTP ermöglicht die Kommunikation zwischen einem Webbrowser/Client (in diesem Fall der Raspberry Pi) und einem Webserver.

HTTP funktioniert nach dem Client-Server-Modell. Der Webbrowser sendet eine Anfrage an den Webserver, der daraufhin eine Antwort zurückgibt. Diese Anfrage- und Antwortnachrichten bestehen aus einem Header und einem optionalen Nachrichteninhalt.

Der Header enthält Informationen über die Art der Anfrage oder Antwort, wie z.B. den verwendeten HTTP-Verb (GET, POST, etc.), den Hostnamen des Servers und andere Metadaten. Der Nachrichteninhalt kann optional sein und Daten wie HTML-Seiten, Bilder, Videos oder andere Ressourcen enthalten.

## IP-Adresse

Eine IP-Adresse (Internet Protocol-Adresse) ist eine eindeutige numerische Kennung, die einem Gerät zugewiesen wird, das mit einem Computernetzwerk verbunden ist. Sie dient dazu, Geräte innerhalb eines Netzwerks zu identifizieren und ihnen die Kommunikation miteinander zu ermöglichen.

## Webserver

Ein Webserver ist ein spezieller Server, der HTTP (Hypertext Transfer Protocol) verwendet, um Webseiten und andere Ressourcen über das World Wide Web bereitzustellen. Er nimmt Anfragen von Clients (meist Webbrowsern) entgegen und liefert die angeforderten Inhalte zu-rück.

Ein Webserver ist in der Regel eine Softwareanwendung, die auf einem physischen Server oder einer virtuellen Maschine läuft. Diese Software ermöglicht es dem Server, Webseiten, Bilder, Videos, Dateien und andere Inhalte über das Internet zu hosten und bereitzustellen.

Bekannte Webserver-Softwarelösungen sind Apache HTTP Server, Nginx, Microsoft Internet Information Services (IIS) und Lighttpd. Diese Server-Softwaren haben unterschiedliche Funktionen, Leistungsfähigkeit und Konfigurationsmöglichkeiten, aber sie dienen alle dem Zweck, Webinhalte über das Internet bereitzustellen. In diesem Projekt wird Apache genutzt.

Ein Webserver bildet die Grundlage für die Bereitstellung von Websites und Webanwendungen im Internet. Er ermöglicht es Benutzern, auf Webseiten zuzugreifen, Informationen abzurufen und mit den Inhalten zu interagieren.

## Cloud

Eine Cloud bezieht sich in der Informationstechnologie auf eine Infrastruktur, die es Unter-nehmen, Organisationen und Privatpersonen ermöglicht, verschiedene Dienste, Ressourcen und Anwendungen über das Internet bereitzustellen, zu nutzen und zu verwalten.

Im Wesentlichen handelt es sich bei der Cloud um eine virtuelle Umgebung, in der Daten, Software, Speicherplatz und Rechenleistung auf Servern gehostet werden, die über das Internet zugänglich sind. Diese Server können sich an verschiedenen Standorten weltweit befinden und sind über ein Netzwerk miteinander verbunden.

Die Cloud bietet verschiedene Dienste und Modelle an, darunter Infrastructure as a Service (IaaS). Hierbei werden virtuelle Maschinen, Speicher und Netzwerkinfrastruktur bereitgestellt. Benutzer können ihre eigenen Anwendungen und Betriebssysteme in der Cloud ausführen, ohne physische Hardware erwerben oder verwalten zu müssen. In diesem Projekt wurde das IaaS Angebot von Oracle genutzt um eine Linux-VM zu betreiben.

## Datenbank

Eine Datenbank ist eine organisierte Sammlung von strukturierten Daten, die elektronisch gespeichert, verwaltet und abgerufen werden können. Sie dient dazu, große Mengen von Informationen effizient zu speichern, zu organisieren und abzufragen.

Eine Datenbank besteht aus einer oder mehreren Tabellen, die Informationen in Zeilen und Spalten strukturieren. Jede Zeile repräsentiert einen Datensatz, während jede Spalte ein Attribut oder eine Eigenschaft darstellt. Durch die Verwendung von Schlüsseln und Beziehungen zwischen den Tabellen können Datenbanken komplexe Zusammenhänge und Abhängigkeiten zwischen den Datensätzen darstellen.

Datenbanken spielen eine entscheidende Rolle bei der Speicherung und Verwaltung von Daten in verschiedenen Anwendungen und Systemen. Sie ermöglichen es, große Mengen an In-formationen effizient zu organisieren, abzurufen und zu analysieren.

## Webanwendung

Eine Webanwendung, auch als Web-App abgekürzt, ist eine Softwareanwendung, die über einen Webbrowser oder eine webbasierte Benutzeroberfläche aufgerufen und genutzt wird. Im Gegensatz zu herkömmlichen Desktop-Anwendungen, die auf dem lokalen Computer installiert werden, läuft eine Webanwendung auf einem Webserver und wird über das Internet bereitgestellt.

Webanwendungen werden mithilfe von Webtechnologien wie HTML (Hypertext Markup Language), CSS (Cascading Style Sheets) und JavaScript entwickelt. Sie können auf verschiedenen Plattformen und Betriebssystemen genutzt werden, da der Zugriff über den Webbrowser erfolgt, der auf den meisten Geräten verfügbar ist.

# Inbetriebnahme Adafruit BME680 und Raspberry Pi

Im Rahmen des Projekts "Hardware cyber-physischer Systeme" erhielt das Projektteam von Prof. Dr.-Ing. Matthias Mörz einen Raspberry Pi sowie den Adafruit BME680-Sensor. Nach Erhalt der Hardware wurde diese sorgfältig überprüft und in Betrieb genommen.

Zur Inbetriebnahme wurde das Betriebssystem Raspberry Pi OS (64-bit) with desktop mithilfe des Raspberry Pi Imagers auf die mitgelieferte microSD-Karte geschrieben und in den Raspberry Pi eingesteckt. Nach dem Anschließen an eine Stromquelle startete und bootete das Ge-rät innerhalb kurzer Zeit und war einsatzbereit.

Die Verbindung zum Raspberry Pi erfolgt über SSH, um eine sichere und ferngesteuerte Ver-bindung herzustellen. Durch Ausführen des Befehls "sudo apt update && sudo apt upgrade" wurde die installierte Software auf den neuesten Stand aktualisiert.

Als nächstes muss der Adafruit BME680-Sensor mit dem Raspberry Pi verbunden werden. Hierfür werden Jumper-Kabel benötigt, die von Herrn Mörz zur Verfügung gestellt wurden. Die genaue Pinbelegung für den Anschluss des Sensors kann mithilfe des Befehls "pinout" abgerufen werden. Zur Ausführung dieses Befehls muss das Programm "python3-gpiozero" mit dem Befehl "apt install python3-gpiozero" installiert werden.

Der Adafruit BME680 bietet sieben Anschlüsse VIN, 3VO, GND, SCK, SDO, SDI und CS. Der Sensor und der Raspberry Pi werden nun mittels Jumper Kabel verkabelt.

Tabelle 1: Pinbelegung BME680

|  |  |
| --- | --- |
| **Raspberry Pi GPIO Pin** | **Adafruit BME680 Pin** |
| 1 (3v3 power) | VIN |
| 6 (Ground) | GND |
| 5 (GPIO3 / SCL) | SCK |
| 3 (GPIO2 / SDA) | SDI |

Nun sieht der Versuchsaufbau folgendermaßen aus.

# Konfiguration der I2C-Schnittstelle

Um den Sensor verwenden zu können, muss zunächst das I2C Interface aktiviert werden. Dies erfolgt mit dem Befehl "raspi-config nonint do\_i2c 0". Anschließend muss das Softwarepaket für das I2C Interface mittels "apt install -y python3-smbus i2c-tools" installiert werden.

Nun sollte getestet werden, ob das Vorgehen erfolgreich war und der Sensor erreichbar ist. Mittels "lsmod | grep i2c\_" erhalten wir alle geladenen Module. Das Ergebnis sollte wie folgt aussehen.

Mit "i2cdetect -y 1" können wir uns den Sensor anzeigen (77) lassen.

Nachdem der Sensor korrekt angeschlossen und die Module geladen sind, kann mit dem aus-lesen der Werte des Sensors begonnen werden.

# Auslesen der Sensordaten

Um Daten vom BME680-Sensor zu erhalten, wird die Adafruit-Bibliothek benötigt. Zunächst müssen wir das Paketverwaltungstool pip installieren, falls es noch nicht vorhanden ist. Dies kann mit dem Befehl apt install python3-pip -y im Terminal durchgeführt werden.

Sobald pip installiert ist, können wir das Python-Paket adafruit-circuitpython-bme680 mithilfe des Befehls pip3 install --user adafruit-circuitpython-bme680 installieren. Das --user-Flag ermöglicht die Installation des Pakets für den aktuellen Benutzer. Dadurch werden keine Administratorrechte benötigt.

Nachdem die Adafruit-Bibliothek erfolgreich installiert wurde, können wir die Sensorwerte auslesen. Hierzu verwenden wir das vorliegende Python-Programm. Der Code ermöglicht die Kommunikation mit dem BME680-Sensor über den I2C-Bus.

Der erste Teil des Codes importiert die erforderlichen Module und erstellt ein I2C-Objekt, das die Kommunikation über den I2C-Bus ermöglicht. Anschließend wird ein BME680-Objekt erstellt, das den Zugriff auf die Sensorfunktionen bereitstellt. Durch die Zuweisung eines Wertes zur sea\_level\_pressure-Eigenschaft kann der Druck auf Meereshöhe für den aktuellen Standort festgelegt werden.

Um genaue Temperaturwerte zu erhalten, kann ein Temperatur-Offset hinzugefügt werden. Dieser Offset wird üblicherweise verwendet, um die Abweichungen des Sensors zu korrigieren. Es wird empfohlen, einen separaten Temperatursensor zur Kalibrierung zu verwenden. In diesem Beispiel wird der temperature\_offset auf 0 gesetzt.

Schließlich werden die gemessenen Werte des Sensors ausgelesen und mit Hilfe von print() auf der Konsole ausgegeben. Die Formatierung erfolgt mithilfe von Platzhaltern (%-Operator), um die entsprechenden Werte einzufügen.

Durch Ausführen des Python-Programms erhalten wir die aktuellen Temperatur-, Gas-, Feuchtigkeits-, Druck- und Höhenwerte des BME680-Sensors.

Der Output sieht nun folgendermaßen aus.

# Konfiguration Respository und Webserver

## Github

Zur Erstellung und Verwaltung des Codes wird Github als Versionsverwaltungsprogramm genutzt. Hierfür wurde ein Repository unter dem Pfad https://github.com/der-benni/airguardx angelegt. Dies ermöglicht es den Entwicklern ihren Code zu teilen und auf einer Codebasis zu arbeiten.

## Webserver

Zur Bereitstellung der Webanwendung wird auf der Oracle Cloud Plattform (https://www.oracle.com/cloud) ein Account erstellt. Innerhalt des Oracle Free Tiers kann eine Linux-Instanz installiert werden, die kostenlos ist und auf die per SSH zugegriffen werden kann. Die Konfiguration des Webservers konnte mit dem Tutorial unter https://docs.oracle.com/en-us/iaas/developer-tutorials/tutorials/spring-on-ol/01oci-ol-spring-summary.htm und weiterer Recherche erfolgreich abgeschlossen werden. Dabei wurden die Pakete GIT, Java, Maven und MySQL installiert.

Tabelle 2: Installierte Pakete auf dem Webserver

| Paket (Version) | Zweck |
| --- | --- |
| GIT (2.36.5) | Mittels GIT kann der aktuelle Code von Github (remote) auf die Linux-Instanz (lokal) geklont werden. |
| Java (17.0.7) | Das für die Erstellung der Webanwendung verwendete Framework basiert auf Java. Damit die Anwendung produktiv auf dem System laufen kann, muss Java installiert sein. |
| Maven (3.6.3) | Mittels Maven kann aus dem Java-Code, der geklont wurde, ein ausführbares JAR gebaut und auf dem Server abgelegt werden. |
| MySQL (8.0.33) | Die Daten, die der Raspberry sendet, werden über die Webanwendung in die Datenbank geschrieben. Diese ist unter localhost:3306/db\_airguardx erreichbar. Für die Datenbank musste ein entsprechender Nutzer mit Passwort erstellt werden. |

Damit der Webserver vom Internet aus erreichbar ist, muss ebenfalls eine Portfreigabe (8080) eingerichtet werden. Dies kann über die Weboberfläche der Oracle Cloud konfiguriert werden.

# Senden der Daten an den Webserver

Nach Erfassung der Sensordaten sollen diese an eine Webserverschnittstelle gesendet, um sie in einer Webanwendung zu visualisieren. Die Kommunikation erfolgt über das HTTP-Protokoll.

Zunächst wird die URL des Webservers definiert, an die die Daten gesendet werden sollen. In diesem Fall ist die URL 'http://141.147.6.122:8080/parameters'.

Dann werden die Sensordaten behandelt. Dabei werden die Temperatur, das Gas, die Luft-feuchtigkeit, die relative Luftfeuchtigkeit, der Druck und die Höhe direkt aus den entsprechenden Variablen des Sensors (z.B. bme680.temperature) übernommen.

Anschließend wird ein JSON-Datenobjekt erstellt, das die erfassten Daten enthält. Zusätzlich wird ein Zeitstempel generiert, der den Zeitpunkt der Erfassung angibt. Das JSON-Datenobjekt sieht wie folgt aus:

Schließlich werden die Daten mit Hilfe eines HTTP-POST-Requests an den Webserver gesendet. Dabei wird das JSON-Datenobjekt als Payload des Requests übermittelt.

Nach dem Senden der Daten wird die Antwort des Servers überprüft. Wenn der Statuscode der Antwort 200 ist, war der Request erfolgreich und die Sensordaten wurden erfolgreich an den Webserver gesendet. Andernfalls wird eine Fehlermeldung ausgegeben, zusammen mit dem erhaltenen Statuscode.

# Automatisierung der Datenabfrage und -sendung

Um die automatisierte Ausführung der Skripte zur Datenabfrage und dem Senden zu steuern, wird der Cron-Dienst verwendet. Cron überprüft regelmäßig im Hintergrund, ob Aufträge vorhanden sind, die ausgeführt werden können, und führt diese dann aus. In diesem Fall soll der Nutzer die Skripte alle 5 Minuten laufen lassen, und um dies dem Cron mitzuteilen, wer-den sogenannte Crontabs verwendet.

Eine Crontab ist eine Tabelle, die alle erforderlichen Informationen für den Cron enthält. In diesem Fall würde die Crontab beispielsweise den Eintrag "\*/5 \* \* \* \* /usr/local/sbin/bme680.py" enthalten. Diese Angabe legt fest, dass das Skript "/usr/local/sbin/bme680.py" alle 5 Minuten ausgeführt wird.

Die Crontab besteht insgesamt aus sechs Feldern, die jeweils durch Leerzeichen getrennt sind. Die ersten fünf Felder bestimmen den Zeitpunkt der Ausführung, während das sechste Feld den auszuführenden Befehl enthält. Somit bedeutet der oben genannte Eintrag, dass das Skript "bme680.py" alle 5 Minuten an jedem Tag ausgeführt wird.

# Erweiterung des Versuchsaufbau um eine LED

Nachdem die Sensordaten erfolgreich erfasst wurden, soll nun eine automatisierte Reaktion (blinken einer LED) auf bestimmte Ereignisse implementiert werden. Im Rahmen des Projekts wurde der Code so entwickelt, dass auf eine Schnittstelle zugegriffen wird, um die letzten beiden Messungen zu überprüfen. Die Daten werden von der Schnittstelle http://141.147.6.122:8080/roomStatus abgerufen. Diese gibt entweder den Wert 0 oder 1 zurück. Wenn der Wert 0 zurückgegeben wird, bedeutet dies, dass die letzten beiden Messungen innerhalb akzeptabler Werte lagen. Wenn jedoch der Wert 1 zurückgegeben wird, weist dies darauf hin, dass die letzten beiden Messungen erhöht waren und eine Aktion erforderlich ist. In diesem Fall lässt der angepasste Code die LED blinken, um auf die erhöhten Werte hinzuweisen.

Zu Beginn des Codes wird der GPIO für den Raspberry Pi vorbereitet, indem der Modus auf "BCM" gesetzt wird und der gewünschte Pin als Ausgang definiert wird. Anschließend wird über die requests-Bibliothek eine GET-Anfrage an die Schnittstelle unter der URL 'http://141.147.6.122:8080/roomStatus' gesendet, um den aktuellen Status abzurufen. Der zurückgegebene Wert wird als Integer gespeichert. Der Code überprüft den Statuswert. Wenn der Wert 1 ist, wird eine Schleife gestartet, in der die LED für 2 Sekunden eingeschaltet und dann für 2 Sekunden ausgeschaltet wird. Dies signalisiert dem Benutzer, dass die letzten beiden Messungen erhöht waren und Maßnahmen ergriffen werden sollten. Am Ende des Codes wird der GPIO-Bereich aufgeräumt und die Pins werden in ihren ursprünglichen Zustand zurückgesetzt, um sicherzustellen, dass keine unerwünschten Signale oder Zustände aktiv bleiben.

# Austausch des BME680 gegen den SCD30 Sensor

Wie zuvor bereits erwähnt, wurde der BME680 im Laufe des Projekts durch den SCD30 ausgetauscht, da dieser in der Lage ist, genaue CO2-Werte zu messen. Die Einrichtung des Sensors erfolgte ähnlich wie zuvor beschrieben, jedoch wurden anstelle der BME680-Bibliotheken nun die entsprechenden Bibliotheken für den SCD30 verwendet.

Die Verkabelung des SCD30-Sensors mit dem Raspberry Pi erfolgte wie folgt:

Tabelle 3: Pinbelegung SCD30

|  |  |
| --- | --- |
| **Raspberry Pi GPIO Pin** | **Adafruit SCD30 Pin** |
| Pi 3V | VIN |
| Pi GND (Ground) | GND |
| Pi SCL | SCL |
| Pi SDA | SDA |

Der Code zur Datenabfrage wurde angepasst.

# Erstellung Webanwendung

## Backend Framework

Das Backend wird mit dem Java-Framework **Spring Boot** (https://spring.io/projects/spring-boot) entwickelt. Es vereinfacht den Aufbau von Java-Webanwendungen und -Microservices, indem es eine vorkonfigurierte Umgebung bereitstellt. Spring Boot basiert auf dem Spring-Framework und bietet eine große Bandbreite an Funktionen, die die Entwicklung erleichtern, wie zum Beispiel automatische Konfiguration und einen integrierten Webserver. Der Webserver wird standardmäßig mit dem Port 8080 initialisiert unter dem dieser erreichbar ist.

## Externe Schnittstellen

Um die geplante Funktionalität der Webanwendung zu ermöglichen, werden die zwei kostenlosen externen APIs von Telegram (https://core.telegram.org/bots/api) und Open-Meteo (https://open-meteo.com/en/docs) angebunden. Die Telegram-API ermöglicht es mittels eines erstellten Bots Daten mit Nutzern und Gruppen der Plattform auszutauschen. Dadurch kann der Webserver Warnungen an Nutzer versenden, falls Schwellwerte überschritten werden, und die Nutzer können mit Befehlen den aktuellen Status in Räumen und das aktuelle Wetter abfragen. Die Wetterdaten liefert die Open-Meteo-API, die die aktuellen Daten für einen Standort abfragt.

## Schnittstellen der Webanwendung

**HTTP-Controller** in Spring Boot sind Komponenten, die den Eingangspunkt für HTTP-Anfragen in einer Webanwendung darstellen. Sie empfangen die Anfragen von Clients und verarbeiten sie entsprechend. Ein HTTP-Controller in Spring Boot wird durch eine Java-Klasse repräsentiert, die mit der Annotation @Controller oder @RestController versehen ist. Im Folgenden werden die im Projekt definierten Controller beschrieben. Controller sind die Schnittstellen (APIs) zur Webanwendung.

Tabelle 4: Controller der Webanwendung

| Controller | Methode | Zweck |
| --- | --- | --- |
| /parameters | POST | Endpunkt, der JSON-Objekte mit den Sensor-Daten vom Rasperry Pi entgegennimmt. |
| /latestRecord | GET | Liefert den zuletzt empfangenen Datensatz zurück. |
| /hourRecord | GET | Liefert die letzten 12 Datensätze zurück. |
| /dayRecords | GET | Liefert die letzten 24 stündlich empfangenen Datensätze zurück. |
| / | GET | Liefert die Startseite im Browser. |
| /info | GET | Liefert die Infoseite im Browser. |
| /settings | GET | Liefert die Einstellungsseite im Browser. |
| /login | GET | Liefert die Login-Seite im Browser. |
| /setProfile | POST | Methode zum Setzen des aktuell ausgewählten Raumprofils. |
| /getProfileValues | GET | Methode zum erhalten der Schwellwerte für die einzelnen Raumprofile. |
| /sensorStatus | GET | Liefert den aktuellen Status des Sensors zurück. Sendet der Sensor über 5min keine Daten an den Webserver, so erhalt man einen Fehlerstatus. |
| /roomStatus | GET | Aufgrund des aktuell ausgewählten Profils und des letzten empfangenen Sensordaten wird ermittelt, ob Handlungsbedarf besteht. Die Methode liefert einen Status zurück. |

## Sicherheit

**Spring Security** ist ein umfangreiches Framework für die Authentifizierung, Autorisierung und den Schutz von Webanwendungen in Spring Boot. Es bietet eine Vielzahl von Funktionen, um Sicherheitsmaßnahmen in einer Anwendung

zu implementieren. Basic Authentication ist eine Form der Authentifizierung, bei der der Client bei jeder Anfrage Benutzername und Passwort in den Header der HTTP-Anforderung einfügt. Spring Security ermöglicht die Implementierung von Basic Authentication in einer Spring Boot-Anwendung. Session Handling bezieht sich auf den Umgang mit Sitzungen in einer Webanwendung. Eine Sitzung wird verwendet, um den Zustand eines Benutzers über mehrere Anfragen hinweg zu verfolgen. Das Session Handling in Verbindung mit Basic Authentication ermöglicht es, den Zustand eines authentifizierten Benutzers über mehrere Anfragen hinweg zu speichern und zu verwalten. Dies ist besonders nützlich, um sicherzustellen, dass der Benutzer während seiner Sitzung angemeldet bleibt und auf geschützte Ressourcen zugreifen kann, ohne sich bei jeder Anfrage erneut authentifizieren zu müssen. Im Folgenden Code-Beispiel wird die Konfiguration für die Sicherheit der Webanwendung abgebildet. Basic Authentication wird aktiviert. Lediglich die Login-Seite ist für alle Nutzer ohne Authentifizierung zugänglich. Alle anderen Seiten und Controller werden ohne erfolgreichen Login blockiert.

## Frontend-Bibliotheken

Zur Erstellung der Webanwendung wurden verschiedene Bibliotheken in das Projekt eingebunden. **jQuery** ist eine JavaScript-Bibliothek, die es Entwicklern ermöglicht, das Schreiben von JavaScript-Code zu vereinfachen. Es bietet Funktionalität zur Manipulation des DOM (Document Object Model), zum Event-Handling, zur Animation und zur Durchführung von HTTP-Anfragen. **Bootstrap** ist ein Framework für das Frontend-Webdesign. Es bietet vorge- fertigte CSS- und JavaScript-Komponenten, die Entwicklern helfen, schnell und einfach ansprechende und responsive Webseiten zu erstellen. Bootstrap enthält beispielsweise Grid-Systeme, Navigationsleisten, Formulare und Modals. **Chart.js** ist eine JavaScript-Bibliothek zur Erstellung von interaktiven Diagrammen und Grafiken. Sie bietet Chart-Typen wie Linien-, und Balkendiagramme, die in Webseiten eingebunden werden können. Mit Chart.js können die Sensordaten auf anschauliche Weise visualisiert werden.

## Startseite mit Menü

Die folgende Abbildung zeigt die Übersicht über alle verfügbaren Seiten der Webanwendung. Zur Auswahl steht seine Startseite (Dashboard), eine Infoseite und eine Seite für Einstellungen (Zahnrad).

## Startseite (Dashboard)

Die Startseite (Dashboard) setzt sich aus zwei Bereichen zusammen. In der folgenden Abbildung (links) werden die Daten für den ausgewählten Raum dargestellt. Der Name des Raumes wird als Titel angezeigt. Daneben erscheint die Anzeige, ob der Sensor aktiv Daten sendet oder länger keine neuen Daten empfangen wurden. Es folgen die aktuellen (zuletzt empfangenen) Daten des Sensors. Die Luftfeuchtigkeit, Temperatur und der CO2-Wert werden angezeigt. Durch farbliche Indikatoren (grün = gute Werte, orange = mittel gute Werte, rot = schlechte Werte) und entsprechende Smileys wird dem Nutzer vermittelt, ob Handlungsbedarf besteht. Zuletzt wird ein Diagramm angezeigt, das die Werte der letzten Stunde oder wenn man den Button betätigt, der letzten 24 Stunden im Verlauf visualisiert. Der rechte Teil der folgenden Grafik stellt die aktuellen Wetterdaten für einen ausgewählten Standort dar. Dafür wird die Open-Meteo API abgefragt, die das Datum und Angaben zum Sonnenauf- und untergang liefert. Auch die aktuelle Luftfeuchtigkeit, Temperatur und Windgeschwindigkeit können über die API bezogen werden. Das Diagramm zeigt den Tagesverlauf für die Luftfeuchtigkeit und Temperatur an.

## Info- und Einstellungsseite

Neben der Startseite existieren die beiden Seiten Info (links) und Einstellungen (rechts), die in folgender Abbildung präsentiert werden. Die Infoseite liefert erweiterte Informationen für den Nutzer, wie zu welcher Jahreszeit richtig gelüftet werden kann, um Schäden und Schimmel in der Wohnung zu verhindern. Die Seite für die Einstellungen zeigt verfügbare, vorkonfigurierte Profile an, welche vom Nutzer ausgewählt werden können. Danach richten sich die Empfehlungen und Warnungen auf dem Dashboard aus, die von der Art des Raumes abhängen.

## Telegram-Bot

Der Telegram-Bot ermöglicht es, Nachrichten mit der Telegram-Plattform auszutauschen. Folgende Konfiguration zeigt einen Service im Webserver, der einen Text in der Methode sendToTelegram() entgegennimmt und zusammen mit dem Bot-Token zur Authentifizierung und einer ChatID an Telegram sendet.

Die Telegram-Kommunikation funktioniert bidirektional. Der Nutzer kann aus der Telegram-App heraus Anfragen per Bot an den Webserver senden und angefragte Daten zurückerhalten. Folgender Code beschreibt den Aufbau des /now Befehls, bei dem der Nutzer die zuletzt empfangenen Sensordaten als Antwort erhält.

Ein weiterer Befehl /outside kann aus der Telegram-App heraus abgesetzt werden. Dabei er mittelt der Webserver die aktuellen Wetterverhältnisse im Außenbereich.

##### Quellen

1. G. Eason, B. Noble, and I. N. Sneddon, “On certain integrals of Lipschitz-Hankel type involving products of Bessel functions,” Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955. *(references)*