Smart Home Control System mit MQTT, Sensorik und GUI

Name and Student IDs:

Florent Salihi (k12223924) - Teamleader

Stefan Djukic (k12203945)

Harun Imamovic (k12213168)

Arian Kycyku (k12220337)

Philipp Riener (k12206255)

Projektidee:
Im Rahmen dieses Projekts wird ein vernetztes eingebettetes System entwickelt, das verschiedene Sensoren zur Überwachung der Umgebungsbedingungen verwendet und in Abhängigkeit davon automatische Reaktionen durch Aktoren auslöst. Zusätzlich können alle relevanten Informationen und Steuerfunktionen über eine zentrale grafische Benutzeroberfläche (GUI) beobachtet und kontrolliert werden.

Das System besteht aus fünf Kernfunktionen:

1. **Lichtautomatisierung:**
   Ein Lichtsensor erkennt den Helligkeitswert im Raum. Bei Unterschreiten eines definierten Schwellwerts wird automatisch eine LED eingeschaltet, um eine Lichtquelle zu simulieren. Zusätzlich kann die Lichtsteuerung manuell über die GUI ein- oder ausgeschaltet werden.

2. **Temperaturgeregelte Klimasteuerung:**
   Ein Temperatursensor misst die Umgebungstemperatur. Bei Temperaturen unter zum Beispiel 20 °C wird eine LED aktiviert, die eine Heizung simuliert. Bei Temperaturen über 25 °C wird ein kleiner Motor eingeschaltet, der einen Ventilator darstellt.

3. **Sicherheitsabschaltung des Ventilators:**
   Um eine Gefährdung durch den laufenden Ventilator zu vermeiden, wird ein Abstandssensor verwendet, der erkennt, ob sich eine Person oder ein Objekt zu nahe nähert. In diesem Fall wird der Ventilator automatisch abgeschaltet, unabhängig von der Temperatursteuerung. Sobald der Abstand wieder ausreichend groß ist, kann der Ventilator erneut aktiviert werden.

4. **Abstandsbasiertes Alarmsystem:**
   Ein weiterer Abstandssensor überwacht einen Eingangsbereich. Wenn das Alarmsystem über die GUI aktiviert wurde und sich jemand dem Sensorbereich nähert, wird ein Alarm ausgelöst (z. B. durch blinkende LED, Lautsprecher oder eine Benachrichtigung in der GUI).

5. **Zentrale grafische Benutzeroberfläche:**
   Die GUI dient als zentrale Steuer- und Visualisierungseinheit. Sie zeigt alle Sensorwerte in Echtzeit an, informiert über den Status der Aktoren (z. B. Licht AN/AUS, Ventilator EIN/AUS) und erlaubt manuelle Steuerung aller Funktionen. Die Kommunikation zwischen den Komponenten erfolgt mittels MQTT über definierte Topics.

Ziel des Projekts ist es, ein modulares, interaktives und benutzerfreundliches Smart-Home-System zu entwickeln, das sowohl autonom (durch Regeln) als auch manuell (über die GUI) betrieben werden kann. Dabei werden die typischen Herausforderungen eingebetteter Systeme wie Datenkommunikation, Entscheidungslogik, Sicherheit und Benutzerinteraktion praxisnah umgesetzt.

Systemarchitektur – Beschreibung:

Die Systemarchitektur besteht aus drei Hauptkomponenten:
Sensor-RPi, **Aktor-RPi** und der **grafischen Benutzeroberfläche (GUI)**. Die Kommunikation zwischen den Komponenten erfolgt über das MQTT-Protokoll nach dem Publish/Subscribe-Prinzip.

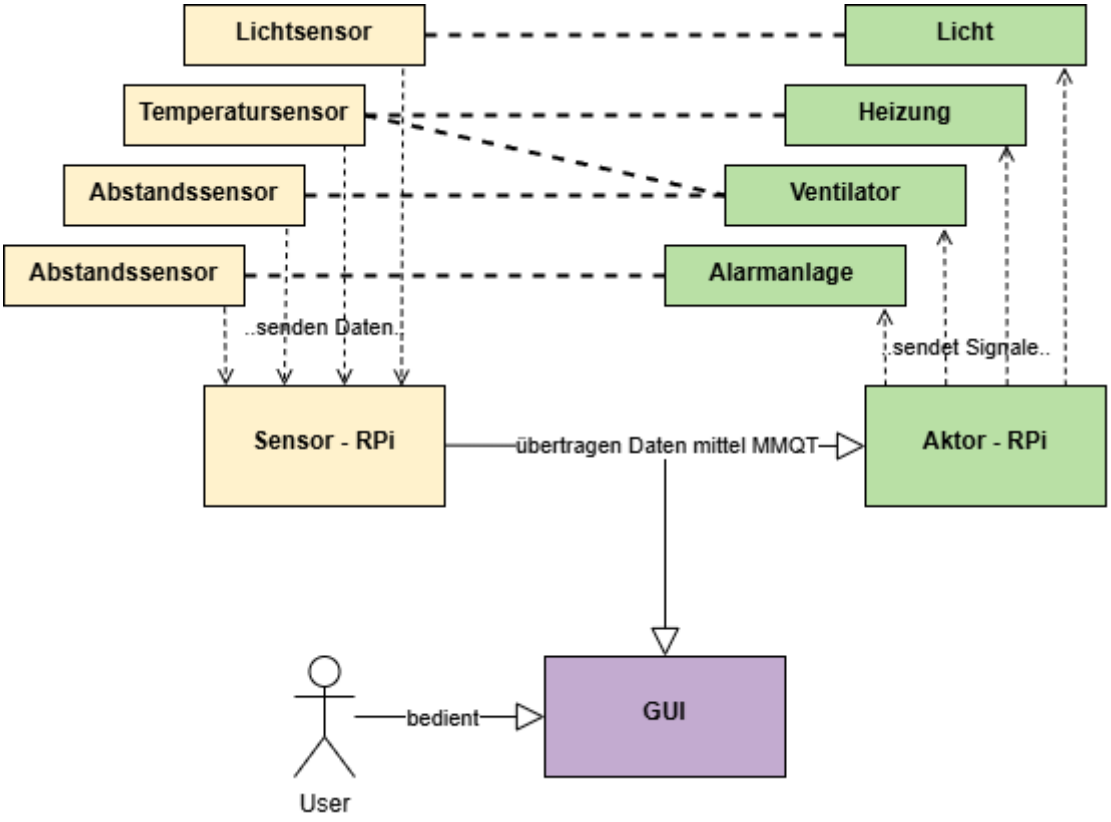

Abbildung 1: Systemarchitektur

Sensor-Raspberry Pi (Sensor-RPi):

Der Sensor-RPi ist für die Erfassung der Umgebungsdaten zuständig. Es sind folgende Sensoren angeschlossen:

- **Lichtsensor** – misst die Helligkeit
- **Temperatursensor** – misst die Raumtemperatur
- **Abstandssensor 1** – prüft, ob sich jemand dem Ventilator nähert
- **Abstandssensor 2** – überwacht einen Eingangsbereich für die Alarmfunktion

Die erfassten Daten werden über definierte MQTT-Topics an den Aktor-RPi übermittelt, der die Auswertung und Steuerung der Aktoren übernimmt.

Aktor-Raspberry Pi (Aktor-RPi):

Der Aktor-RPi empfängt sowohl:

- **Sensorwerte** vom Sensor-RPi
- **Benutzereingaben** von der GUI

Anhand dieser Informationen steuert der Aktor-RPi folgende Aktoren:

- **Licht (LED):** wird bei Dunkelheit oder per GUI eingeschaltet
- **Heizung (simuliert als LED):** wird bei niedriger Temperatur eingeschaltet
- **Ventilator (Motor):** wird bei hoher Temperatur aktiviert, jedoch bei Nähe automatisch abgeschaltet
- **Alarmanlage (simuliert als Lautsprecher falls vorhanden):** wird bei erkannter Bewegung aktiviert, sofern die Anlage über die GUI scharf geschaltet wurde

Zusätzlich sendet der Aktor-RPi Statusinformationen über den aktuellen Zustand aller Aktoren (z. B. Licht AN/AUS, Alarm AKTIV, Ventilator AUS wegen Sicherheitsabschaltung) an die GUI zurück. Diese Kommunikation erfolgt ebenfalls über MQTT.

Grafische Benutzeroberfläche (GUI):

Die GUI dient zur:

- Anzeige aller aktuellen Sensorwerte (z. B. Temperatur, Licht, Abstände)
- Visualisierung der aktuellen Zustände der Aktoren (z. B. Ventilator EIN/AUS, Alarm aktiv)
- Manuellen Steuerung der Aktoren (z. B. Licht an, Alarm scharf/unscharf)

Die GUI kommuniziert dabei bidirektional über MQTT:

- Sie abonniert Status- und Sensordaten vom Sensor-RPi und Aktor-RPi
- Sie sendet Steuerbefehle an den Aktor-RPi

Erfüllung der Projektanforderungen:

Das Projekt erfüllt alle geforderten Mindestanforderungen. Es besteht aus fünf Modulen, die jeweils einer Person zugewiesen sind. Vier Gruppenmitglieder setzen jeweils eine Sensor-Aktor-Kombination um: ein Lichtsensor steuert eine LED, ein Temperatursensor regelt eine Heizung (LED) und einen Ventilator (Motor), ein Abstandssensor dient der Sicherheitsabschaltung des Ventilators und ein zweiter Abstandssensor löst bei aktivem Alarmsystem eine Warnung aus. Die fünfte Person ist für die zentrale grafische Benutzeroberfläche verantwortlich, über die Sensorwerte visualisiert und alle Aktoren manuell gesteuert werden können.

Die Kommunikation zwischen den Modulen erfolgt über das MQTT-Protokoll. Sensorwerte werden vom Sensor-Raspberry Pi an den Aktor-Raspberry Pi übertragen, der die Aktoren steuert und den aktuellen Systemstatus an die GUI zurückmeldet. Damit ist die Kommunikation zwischen mindestens zwei eigenständigen Einheiten gegeben.

Alle Steuerungen basieren auf einfachen datengetriebenen Regeln, etwa zur automatischen Licht- oder Temperaturregelung sowie zur sicherheitskritischen Abschaltung und Alarmauslösung. Damit sind auch die Anforderungen an datenbasierte Entscheidungslogik erfüllt.

Materialliste:

Recheneinheiten:

- 2× Raspberry Pis

Sensoren:

- 1× Sense HAT (Lichtsensor und Temperatursensor)
- 1× VL6180X Time-of-Flight Distanzsensor
- 1× HC-SR04 Ultraschallsensor (für Alarmsystem, falls nicht vorhanden → wird zusätzlich beschafft)

Aktoren:

- 2× LEDs (für Licht und Heizungssimulation)
- 1× DC-Motor oder Rotation-Servo (Ventilator)
- 1× Lautsprecher, LED oder Buzzer (für Alarmanzeige)

Zubehör:

- Steckbrett, Jumper-Kabel, Widerstände (aus dem Adafruit Parts Pal)
- Spannungsversorgung (USB-Netzteile / Powerbank)

Phase 1 (Woche 1–2): Planung & Setup

Ausarbeitung der Projektidee, Aufgabenverteilung, Einrichtung der Entwicklungsumgebung sowie Installation und Test des MQTT-Brokers. Erste Tests mit Sensoren und Aktoren.

Phase 2 (Woche 2–3): Modul-Implementierung

Umsetzung der einzelnen Sensor-Aktor-Kombinationen durch die Gruppenmitglieder. Erste Integration von MQTT-Kommunikation zwischen Sensoren und Aktoren. Grundgerüst der GUI entsteht.

Phase 3 (Woche 3–4): Systemintegration & GUI

Zusammenführung aller Module in ein Gesamtsystem. Die GUI wird vollständig angebunden und um manuelle Steuerfunktionen erweitert.

Phase 4 (Woche 4–5): Finalisierung & Präsentation

Fehlerbehebungen, Feinschliff am System, Erstellung des Demo-Videos sowie Fertigstellung der Projektdokumentation und GitHub-Abgabe.