**Ideenskizze für das IoT-Projekt**

**1. Allgemeine Informationen**

* **Projektname:** *RFID-basierte Benutzererkennung mit MQTT Übertragung und LED-Rückmeldung*
* **Datum:** *04.05.2025*
* **Name:** Tom Stoelken

**2. Projektbeschreibung**

**Kurzbeschreibung:**  
*Dieses System liest die UID einer RFID-Karte aus und vergleicht sie mit zuvor hinterlegten, bekannten UIDs. Wird eine gültige UID erkannt, leuchtet eine* ***RGB-LED grün****, und der zugehörige Benutzername wird zusammen mit einem* ***Zeitstempel über MQTT an Node-RED*** *gesendet und in einer Datenbank gespeichert.   
Ist die UID nicht im System hinterlegt, leuchtet die* ***RGB-LED******rot*** *und es erscheint eine Fehlermeldung.  
Das Projekt dient zur einfachen Identifikation von Personen oder zum Zugriff auf weitere Systeme (z. B. Türsteuerung, Zeiterfassung etc.).*

**3. Anforderungen und Funktionalitäten**

✅ **Sensorik 🡪 RFID-Erkennung über RC522**

✅ **Aktoren-Steuerung 🡪 RGB-LED zur optischen Rückmeldung**

✅ **Webinterface & Benutzerinteraktion 🡪 Node-Red über MQTT**

**4. Benötigte Komponenten**

**4.1 Hardware**

| **Komponente** | **Modell/Typ** | **Funktion** |
| --- | --- | --- |
| **Mikrocontroller** | ESP 32 Wroom |  |
| **Sensor 1** | RFID-RC522 | Erkennung des RFID-Chips |
| **Sensor 2** | BHT1750 | Messung der aktuellen Helligkeit in LUX |
| **Aktor 1** | RGB LED | Grün bei positiver Abfrage Rot bei negativer Abfrage |
| **Aktor 2** |  |  |
| **Stromversorgung** | Powerbank/Notebook | Versorgung für ESP |
| **Weitere Bauteile** |  |  |

**4.2 Software & Datenbank**

| **Komponente** | **Technologie** | **Funktion** |
| --- | --- | --- |
| **Microcontroller-Code** | Micropython | Erstellen des Codes |
| **Webinterface** | Node-Red Dashboard | Anzeigen der Stempeldaten |
| **Datenbank** | MariaDB | Speichern der Stempeldaten |

**5. Systemarchitektur**

1. **ESP32 (Client/Publisher & Subscriber)**
   * Misst Sensordaten (z. B. entschlüsselten Code des RFID Scanners nach Chip auflegen, Ausgabe der Lichtstärke in Lux durch BHT1750)
   * Sendet Daten als JSON per MQTT an einen Broker
2. **MQTT-Broker**
   * Vermittelt Nachrichten zwischen ESP32 und anderen Clients
3. **Node-RED (Visualisierung & Automatisierung)**
   * Verarbeitet MQTT-Daten als JSON Format und zeigt sie in einem Dashboard an
   * Setzt Automatisierungsregeln um
4. **LTE-/WLAN-Verbindung**
   * ESP Verbindung über mobilen entsprechende WLAN Verbindung

**6. Zeitplanung (Meilensteine)**

| **Datum** | **Aufgabe** |
| --- | --- |
| KW 11 | Bestellung der Einzelteile für Projektumsetzung Definieren der Projektidee Erste Planungen für die Umsetzung |
| KW 12 | Grobes Software Setup und Node-Red Anwendung ohne Hardware |
| KW 13 | UID-Vergleich programmieren, LED Reaktion testen |
| KW 14 | MQTT Anbindung und Note-Red-Konfiguration |
| KW 15 | Zeitstempel und Logging über Node-Red |
| KW 16 | Aufbau der Hardware, RFID-Test |
| KW 17 | Testphase, Fehlerbehebung |
| KW 18 | Feinschliff und zusätzliche Features BHT1750 |
| KW 19 | Dokumentation und Fehlerbehebung |

**7. Offene Fragen & Herausforderungen**

*Wie robust ist die UID-Erkennung bei schneller Kartenbewegung?*

*Ist der Speicherplatz für bekannte UIDs auf dem ESP32 ausreichend?*

*Reagiert das System zuverlässig auch bei instabiler WLAN-Verbindung?*

*Wie hoch darf der Schaltabstand des RFID-Lesers maximal sein?*

**8. Fazit & Zielsetzung**

Am Ende soll das System zuverlässig RFID-Karten erkennen, bekannte Nutzer optisch mit grünem Licht bestätigen und deren Anmeldezeit per MQTT an Node-RED übermitteln.  
Zusätzlich sollen die Zeiten mit dem dazugehörigen Benutzernamen in der Datenbank gespeichert werden.  
Unbekannte Karten sollen deutlich mit rotem Licht abgewiesen werden. Ein Tonsignal ist optional.  
Zudem kann das System später um weitere Aktionen (z. B. Türöffnung, Webinterface, Display) erweitert werden.