Adolf-Reichwein-Schule Marburg
Lernfeld 07 - Herr Michael Köppl

Dokumentation

„Erstellung eines cyberphysischen Systems mit Raspberry Pi und ESP8266“

Vorgelegt von:
Philipp Stein, Enes Koc & Luca Elia Drescher

Inhaltsverzeichnis

1. Abstract

Ziel dieses Projekts ist die Entwicklung eines einfachen cyber-physischen Systems, das aus zwei zentralen Komponenten besteht: einem Sensor-Client auf Basis eines ESP-Moduls und einem Dashboard-Server auf einem Raspberry Pi. Der Sensor-Client erfasst Umweltdaten über einen angeschlossenen Sensor und überträgt diese drahtlos an den Dashboard-Server. Dort werden die empfangenen Daten verarbeitet, gespeichert und in Echtzeit auf einem übersichtlichen Dashboard visualisiert.

2. Verzeichnisse und technische Definitionen

2.1 Abkürzungs- und Symbolverzeichnis

- SSH: Secure Shell
- VNC: Virtual Network Computing
- ESP: Espressif Systems Processor
- SD-Karte: Secure Digital Memory Card
- LAN: Local Area Network
- WLAN: Wireless Local area Network
- IP-Adresse: Internet Protocol Address
- MQTT: Message Queuing Telemetry Transport
- DHT11 Sensor: Digital Temperature and Humidity sensor
- LED: Light emitting Diode
- USB-C: Universal Serial Bus Type C
- GB: Gigabyte
- TCP: Transmission Control Protocol
- npm: Node-Package-Manager

2.2 Hardware-Verzeichnis

- Raspberry Pi 4 Model B 8GB Arbeitsspeicher
- SD-Karte
- Netzteil Raspberry Pi
- Ethernet Kabel
- DHT11 Sensor
- LoLin V3 ESP 8266
- USB-C auf USB-A Kabel
- Breadboard
- LED
- Widerstand 200 Ohm
- Weiblich auf weiblich Steckkabel
- Weiblich auf männlich Steckkabel
- Notebook zum Konfigurieren

2.3 Software-Verzeichnis

- Arduino DIE
- Raspberry Pi OS (& Raspberry Pi Imager)
- PuTTY
- RealVNC Viewer
- Eclipse Mosquitto
- Node-RED

3. Projektteam

Das Projekt wurde in einer dreier Gruppe durchgeführt die Mitglieder waren: Enes Koc, Luca Elia Drescher und Philipp Stein. Es wurde im Rahmen des Berufsschulunterrichtes in Lernfeld 7 (Cyberphysische Systeme) durchgeführt.

Die Projektaufgaben wurden alle gemeinsam bearbeitet jedes Mitglied hatte dazu noch seinen gewissen Schwerpunkt.

- Enes Koc: Zusammenstellung der Hardwarekomponenten
- Luca Elia Drescher: Programmierung
- Philipp Stein: Konfiguration der Linux Server Landschaft

4. Projektplanung

4.1 In-Scope & Out-Scope

4.1.1 In-scope

- Aufbau eines funktionsfähigen Sensor-Clients auf Basis eines ESP-Moduls
- Anbindung eines Sensors zur Datenerfassung (Temperatur, Luftfeuchtigkeit)
- Drahtlose Übertragung der Sensordaten per MQTT
- Einrichtung eines Raspberry Pi als Dashboard-Server
- Entwicklung eines Dashboards zur Echtzeit-Anzeige der Messwerte mit Node-RED
- Grundlegende Netzwerkkonfiguration (WLAN, IP-Zuweisung)
- Dokumentation und Präsentation des Projekts

4.1.2 Out-scope

- Sicherheitsmechanismen (z. B. TLS-Verschlüsselung, Benutzer-Authentifizierung)
- Nutzung von Cloud-Plattformen oder externer Datenspeicherung (z. B. AWS, Azure)
- Fernzugriff über das Internet
- Energieoptimierung für Langzeitbetrieb (z. B. Akkubetrieb, Sleep-Modes)
- Skalierung auf mehrere Sensoren oder Geräte (Multi-Client-System)

4.2 Gantt-Diagram

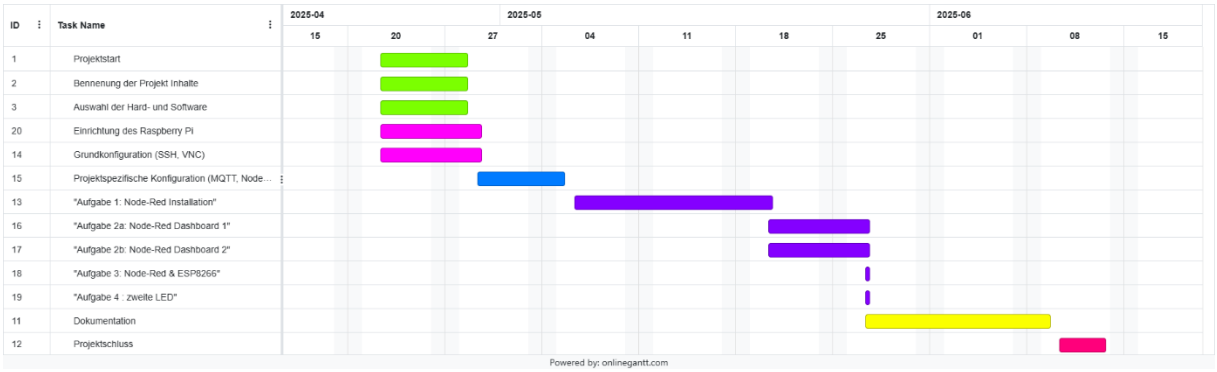

Abbildung 1: Planung des Projektes

5. Umsetzung

5.1 Einrichtung des Raspberry Pi

Zuerst wurde auf eine SD-Karte das Raspberry PI OS geschrieben und vor konfiguriert.

Vergeben wurden ein Hostname, ein Username und ein Passwort:

- Hostname: PEL
- Username: ITM
- Passwort: ITM2024

Dazu wurde dem Pi eine Feste IP-Adresse vergeben hierzu benötigte es eine Eingabe der IP –Adresse in der cmdline.txt Datei auf der SD-Karte

```
console=serial0,115200 console=tty1 root=PARTUUID=d2428907-02 root-
fstype=ext4 fsck.repair=yes rootwait quiet splash plymouth.ignore-
serial-consoles ip=192.168.69.69 cfg80211.ieee80211_regdom=DE
```

5.1.1 Grundkonfiguration (SSH, VNC)

Durch die Feste IP-Adresse kann nun eine SSH-Verbindung aufgebaut werden dazu wurde das Programm PuTTY verwendet. Für die Verbindung wurde die Angabe der IP-Adresse sowie des verwendeten Ports benötigt in diesem Fall war dies der Port 22. Dies ist der Standard-Port für sichere TCP-Verbindungen.

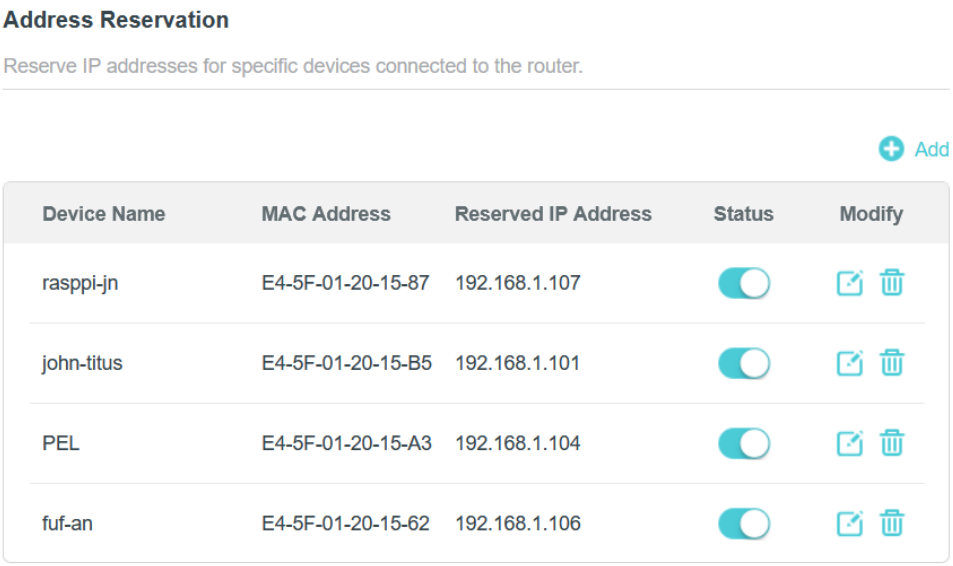

Abbildung 2: IP-Reservierungen

Über die SSH-Verbindung konnte nun auf dem PI VNC Aktiviert werden und mit demReal-VNC Viewer eine remote Verbindung auf den Desktop des Pis gestartet werden. Über diese Verbindung wurde die WLAN-Einstellungen konfiguriert und der Pi in das Netzwerk mit der SSID: "IT-Berufe" eingebunden.

5.1.2 Projektspezifische Konfiguration (MQTT, Node-RED)

Mosquitto Konfiguration

Als nächstes wurde mit den Befehlen "`sudo apt install mosquitto`" und "`sudo apt install mosquitto-client`" der verwendete Brooker und der dazugehörige Client installiert. Als nächstes wurde mit "`sudo nano /etc/mosquitto/mosquitto.conf`" der Broker konfiguriert.

Dazu wurde in die Datei folgende Zeilen hinzugefügt:

```
listener 1883
Allow_anonymous true
```

Danach wurde mit dem Befehl "sudo systemctl enable mosquitto" der Broker in den Autostart des Pi's gebracht.

Node-RED Konfiguration

Zuerst wurde mit dem Befehl "`bash <(curl -sL https://raw.githubusercontent.com/node-red/linux-installers/master/deb/update-nodejs-and-nodered)`" Node-Red installiert und mit dem befehl "`Node-RED`" gestartet. Nun kann über einen Browser unter der eingabe der IP mit Port auf das Dashboard zugegriffen werden, in unserem Falle war dies 192.168.1.14:1880. Dort wurde eine Vorkonfiguration vorgenommen und als .json file exportiert wurde und im Github unter /code/node-red/flows.json hinterlegt.

6. Ergebnisse und Aufgabenbearbeitung

6.1 „Aufgabe 1: Node-Red Installation“

Aufgabenstellung: Installation von Node-RED und Einrichtung eines einfachen Test-Flows zur Überprüfung der Funktionalität.

Umsetzung: Node-RED wurde – wie bereits in 5.1.2 Projektspezifische Konfiguration (MQTT, Node-RED) dokumentiert – installiert und ein simpler Flow wurde zu Testzwecken erstellt, welcher auf Knopfdruck einen string in der Debug-Konsole wiedergibt.

6.2 „Aufgabe 2a: Node-Red Dashboard 1“

Aufgabenstellung: Installation von „node-red-dashboard v.3.6.5“ und Testen der Erreichbarkeit über die IP-Adresse des Dashboard-Servers (Raspberry Pi).

Umsetzung: Die Node-RED Erweiterung „node-red-dashboard“ wurde durch npm heruntergeladen. Daraufhin wurde ein neues (leeres) Dashboard erstellt, um die Erreichbarkeit zu testen.

6.3 „Aufgabe 2b: Node-Red Dashboard 2“

Aufgabenstellung: Einrichten des für das Projekt benötigten Dashboard auf dem Dashboard-Server (Raspberry Pi), Erstellung des Layouts sowie die Einrichtung der individuellen Nodes.

Umsetzung: Zur Visualisierung und Steuerung wurde ein Dashboard erstellt und konfiguriert. Im Bereich „Layout“ wurde ein Tab mit dem Namen „Raum“ angelegt. Innerhalb dieses Tabs wurden zwei Gruppen eingerichtet: „LED“ und „Sensor“.

Folgende Nodes wurden erstellt und anhand der vorgegebenen Dokumentation konfiguriert (für die Konfiguration siehe: /code/node-red/flows.json):

- switch
- mqtt (3x)
- chart
- gauge

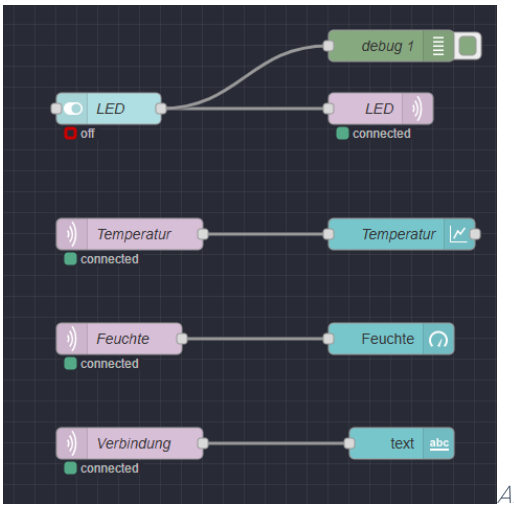

Abbildung 3: Node-RED Flow-chart

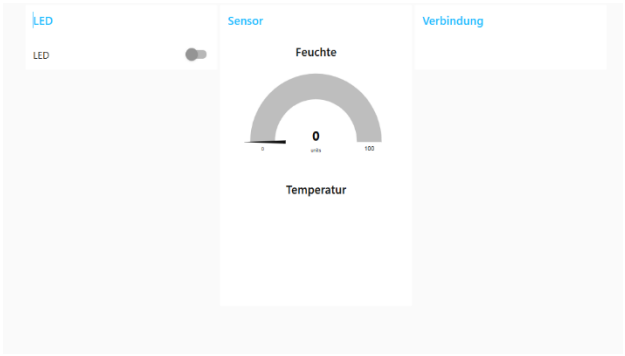

Abbildung 4: Node-RED Dashboard

6.4 „Aufgabe 3: Node-Red & ESP8266“

Aufgabenstellung: Installation der DHT-Library für Temperatur & Feuchtesensor und Datenübertragung von Sensor-Client (ESP-Modul) auf den Dashboard-Server (Raspberry Pi) via MQTT.

Umsetzung: Die DHT-Library wurde über den Library-Manager der Arduino IDE installiert. Daraufhin wurde der von der Lehrkraft gestellte Code (siehe /code/esp8266/MQTT-Node-Red-Vorlage.ino) entsprechend an die Netzwerkeinstellungen angepasst und der MQTT Broker verbunden. Anschließend wurde der Code auf das ESP-Modul überspielt.

Die Funktionalität wurde durch die Ergebnisse aus dem Serial Monitor und der erfolgreichen Darstellung im Node-RED Dashboard erfolgreich überprüft.

Abbildung 5: Ergebnis "Aufgabe 4"

6.5 „Aufgabe 4: zweite LED“

Aufgabenstellung: „Füge einen Schalter für eine zweite LED hinzu. Passe den Code in Node-Red und in Arduino so an, dass sich diese ebenfalls schalten lässt.“

Umsetzung: Der zugrunde liegende Code basiert auf der Vorlage aus Aufgabe 3 und wurde um eine zusätzliche MQTT-Topic („LED2“) erweitert. Über diese wird eine zweite LED angesteuert, die am Pin 2 angeschlossen ist (const int lamp = 2). Der angepasste Code ist unter /code/esp8266/mqtt.ino abgelegt. Eine externe LED wurde auf das Bread-board gesteckt und entsprechend der Abbildung mit einem Widerstand an das ESP-Modul angeschlossen.

Die Funktionalität wurde entsprechend Aufgabe 3 getestet und die externe LED konnte ebenfalls über das Node-RED Dashboard gesteuert werden.

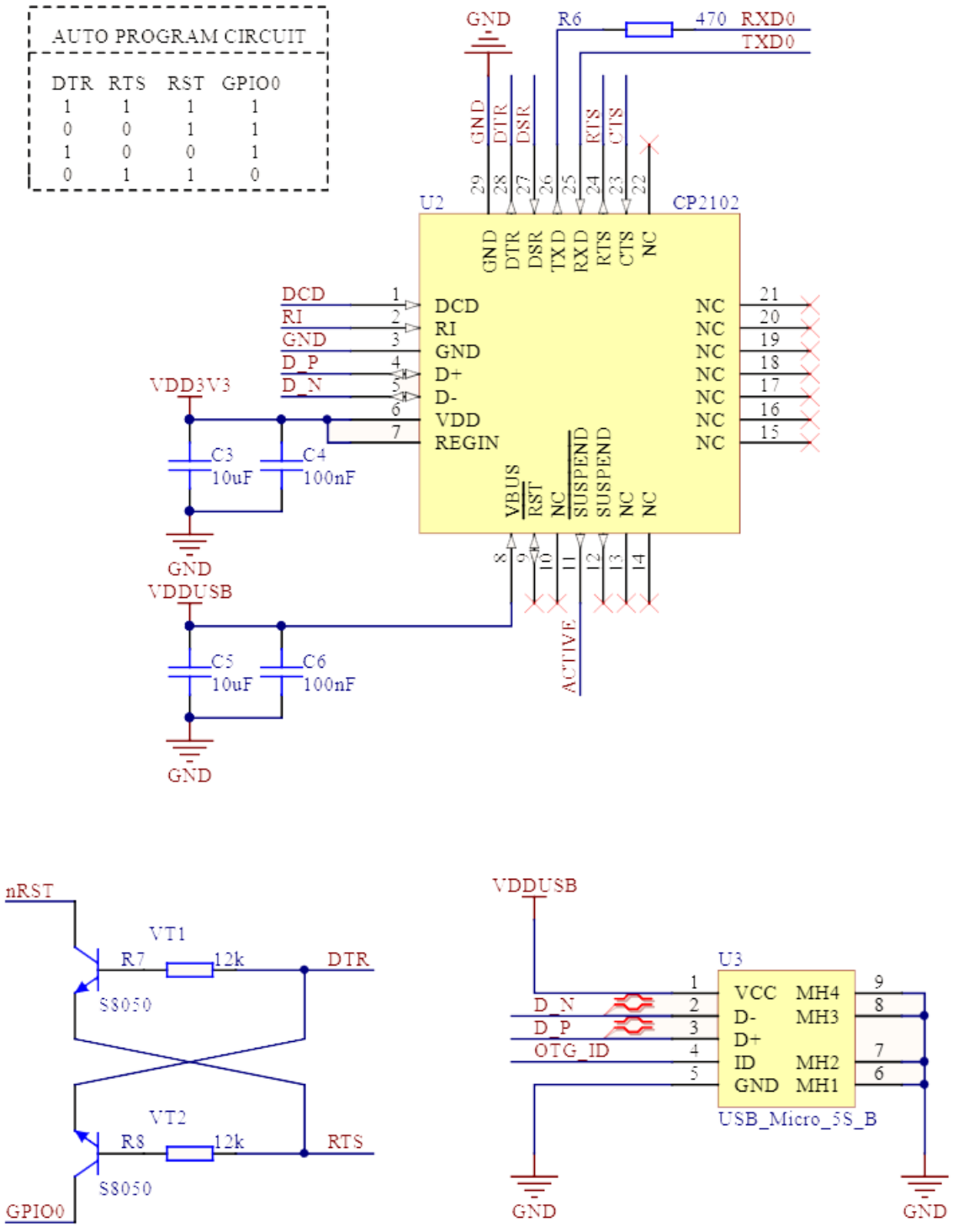

Abbildung 6: Schaltplan ESP und Sensoren

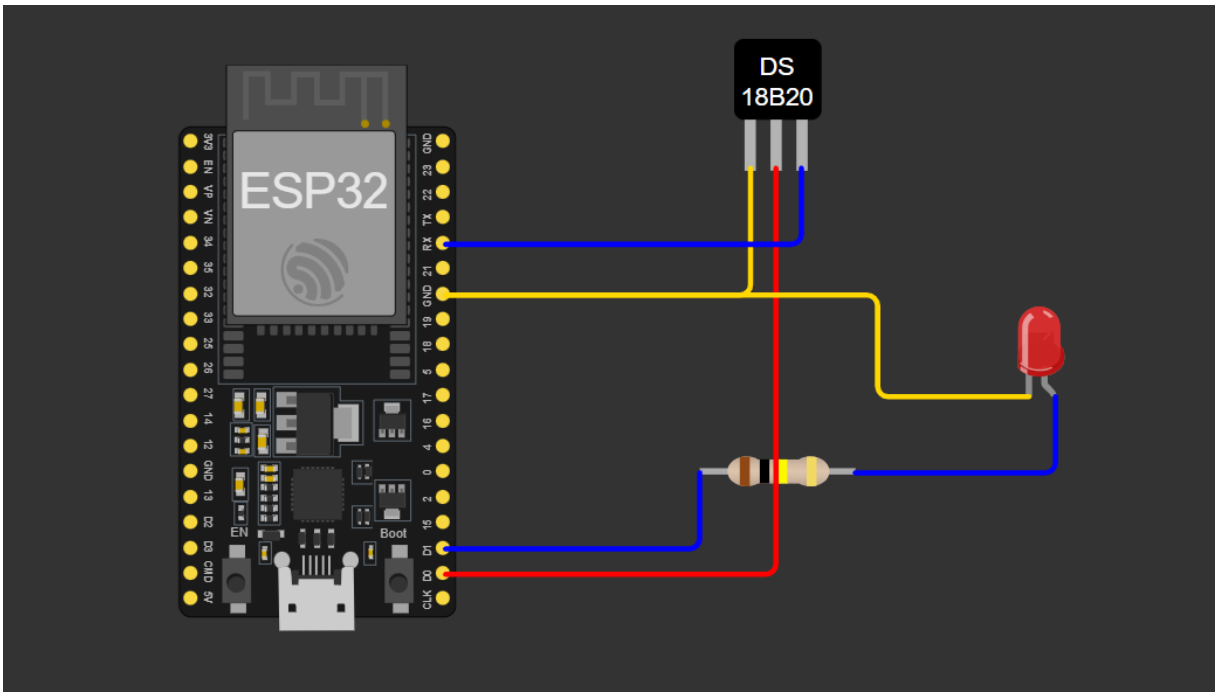

Abbildung 7: ESP-Verkabelung

7. Fazit

Alle im Projekt definierten Ziele wurden erfolgreich erreicht. Das entwickelte System besteht aus einem funktionierenden Sensor-Client auf Basis eines ESP-Moduls sowie einem Dashboard-Server auf einem Raspberry Pi, der die erfassten Umweltdaten zuverlässig empfängt, verarbeitet und in Echtzeit visualisiert.

Bis auf anfängliche Schwierigkeiten bei der Einrichtung des Raspberry Pi verlief die Umsetzung insgesamt reibungslos. Das Ergebnis entspricht vollständig den Erwartungen und demonstriert die grundlegenden Funktionen eines einfachen cyber-physischen Systems.

Ausblick: Für zukünftige Projekte wäre es hilfreich, wenn die Aufgabenstellung klarer und detaillierter formuliert wird. Dadurch ließe sich der Projektfokus besser erkennen und gezielter bearbeiten. Zudem würde ein praxisnahes Szenario, wie zum Beispiel die Entwicklung eines Systems zur Temperaturüberwachung in einem Serverraum - ähnlich wie bei früheren Arduino-Projekten - die Relevanz und Motivation weiter steigern.

8. Abbildungsverzeichnis

9. Literaturverzeichnis / Quellen

1. Node-RED. (o. J.). Node-RED Dashboard. Abgerufen am 10. Juni 2025, von https://flows.nodered.org/node/node-red-dashboard
2. Random Nerd Tutorials. (o. J.). Getting started with Node-RED on Raspberry Pi. Abgerufen am 10. Juni 2025, von https://randomnerdtutorials.com/getting-started-node-red-raspberry-pi/
3. Mikrocontroller-Elektronik. (o. J.). NodeMCU ESP8266 Tutorial WLAN Board Arduino IDE. Abgerufen am 10. Juni 2025, von https://www.mikrocontroller-elektronik.de/nodemcu-esp8266-tutorial-wlan-board-arduino-ide/
4. IBM Developer. (o. J.). MQTT-Protokoll allgemein. Abgerufen am 10. Juni 2025, von https://developer.ibm.com/articles/mqtt-protocol/
5. Espressif Systems. (o. J.). ESP8266 Dokumentation. Abgerufen am 10. Juni 2025, von https://docs.espressif.com/projects/esp8266-rtos-sdk/en/latest/
6. Node-RED. (o. J.). Node-RED Dokumentation. Abgerufen am 10. Juni 2025, von https://nodered.org/docs/
7. Arduino. (o. J.). Arduino IDE Dokumentation. Abgerufen am 10. Juni 2025, von https://docs.arduino.cc/software/ide-v2
8. Components101. (o. J.). DHT11 Temperature Sensor Datasheet. Abgerufen am 10. Juni 2025, von https://components101.com/sensors/dht11-temperature-sensor
9. Raspberry Pi Foundation. (o. J.). Raspberry Pi Projekte. Abgerufen am 10. Juni 2025, von https://projects.raspberrypi.org/en/