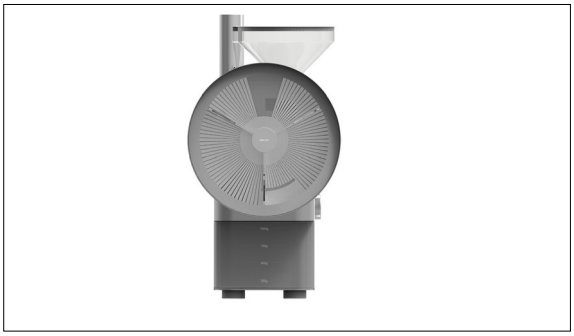

Lukas
Bersinger

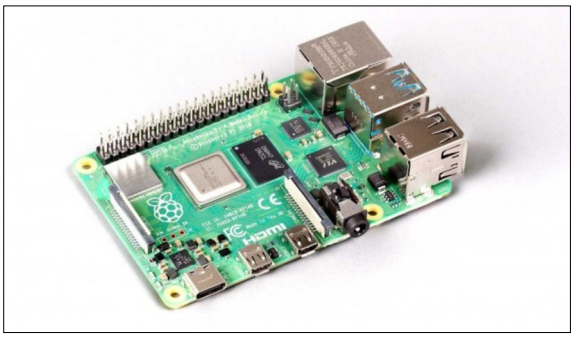

Pascal
Pichler


Fabian
Zanetti

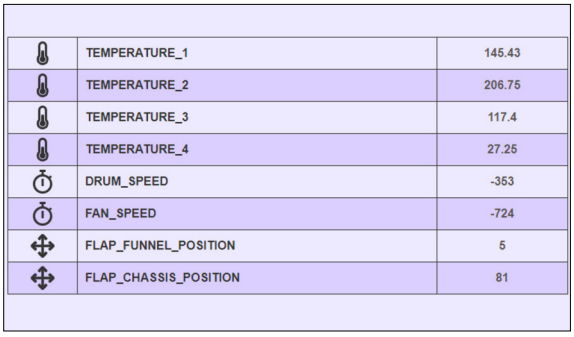
IoT Coffee Roaster



Röster
Mikafi GmbH



Alternativlösung mit Raspberry Pi 4
https://www.pi-shop.ch/raspberry-pi-4-model-b-8gb



	TEMPERATURE_1	145.43
	TEMPERATURE_2	206.75
	TEMPERATURE_3	117.4
	TEMPERATURE_4	27.25
	DRUM_SPEED	-353
	FAN_SPEED	-724
	FLAP_FUNNEL_POSITION	5
	FLAP_CHASSIS_POSITION	81

Admin Interface mit simulierten Sensor Daten
Eigene Darstellung

Ziel der Arbeit: Das Start-Up mikafi stellt Kaffeeröster her und möchte diese in eine IoT-Umgebung einbinden. Der Röster soll über eine App gesteuert werden und ein Admin Interface als Web Applikation soll einen Überblick über alle Röster geben. Die App ist bereits vorhanden, arbeitet derzeit aber mit einer USB-Verbindung. Über diese kann eine Röstung gestartet und gestoppt werden. Während der Röstung können die verschiedenen Sensoren im Röster überwacht werden. Das Admin Interface gibt mikafi eine Übersicht über alle registrierten Röster. In einer Detail Ansicht sollen die Sensor Daten der einzelnen Röster angezeigt werden. Der Röster empfängt Kommandos, welche er bearbeiten wird und sendet Sensor Daten an die App bzw. dem Interface zurück. Diese Kommunikation soll mit einem Kommunikationsprotokoll definiert werden, welches leicht erweiterbar sein soll, falls in Zukunft neue Befehle oder Sensoren hinzustossen.

Vorgehen / Technologien: Mikafi äusserten ihre Wünsche während des Kickoffs des Projektes. Diese mussten eingegrenzt werden und in Ziele ausformuliert werden. Da sich das Team nicht mit der Materie auskannte, hat sich jedes Teammitglied ins Thema eingelesen. Nach dem Verfassen des Projektplans und der Anforderungen, wurde mit der Umsetzung begonnen. Als erstes wurde das Kommunikationsprotokoll erstellt. Es gab Probleme mit dem Eval Board, welches in den Röster installiert werden sollte, das für das Senden für Sensor Daten und Empfangen von Kommandos zuständig ist. Die vom Hersteller bereitgestellten Libraries waren teilweise inkompatibel, spärlich dokumentiert und das Team verfügte über zu wenige Kentnisse in der Elektrotechnik. Deswegen hat das Team eine Alternativlösung vorgeschlagen, damit die Arbeit dennoch fortgesetzt werden konnte. Diese Alternative bestand darin, einen Röster auf einem Raspberry Pi 4 als State Machine zu simulieren, sodass die Kommunikation dennoch erstellt werden konnte. Zum Schluss wurde noch das Admin Interface und eine Library für die IoT Funktionen implementiert.
Das Kommunikationsprotokoll basiert auf MQTT. Dabei handelt es sich um den de-facto Standard in der IoT Entwicklung. Weil mit dem AWS IoT Core von Amazon gearbeitet wurde, konnten nicht alle MQTT Features verwendet werden. Die App wurde mit React Native entwickelt, entsprechend wurden das Admin Interface und die Library mit React und Typescript programmiert.

Ergebnis: Das Ergebnis dieser Arbeit ist eine funktionierende Kommunikation zwischen App, Admin Interface und Röster. Es ist möglich, einen Befehl über die implementierte Library an den Raspberry Pi (simulierter Röster) zu senden. Erhält dieser einen Befehl zum Rösten, wechselt er in den "Roasting" State. In diesem simuliert er das Erhalten von verschiedenen Sensordaten. Diese Werte werden veröffentlicht und können auf dem Admin Interface betrachtet werden. Soll eine Röstung abgebrochen werden, kann dies mit einem "Abbruch" Befehl getan werden. Das Kommunikationsprotokoll kann einfach erweitert werden, falls neue Befehle oder Sensoren hinzugefügt werden. Auch das Admin Interface und die Library können mit nur wenig Aufwand ergänzt werden. Als nächster Schritt sollte das Eval Board konfiguriert werden, damit die Sensor Daten nicht mehr simuliert werden müssen, sondern direkt echte Daten verwendet werden können.