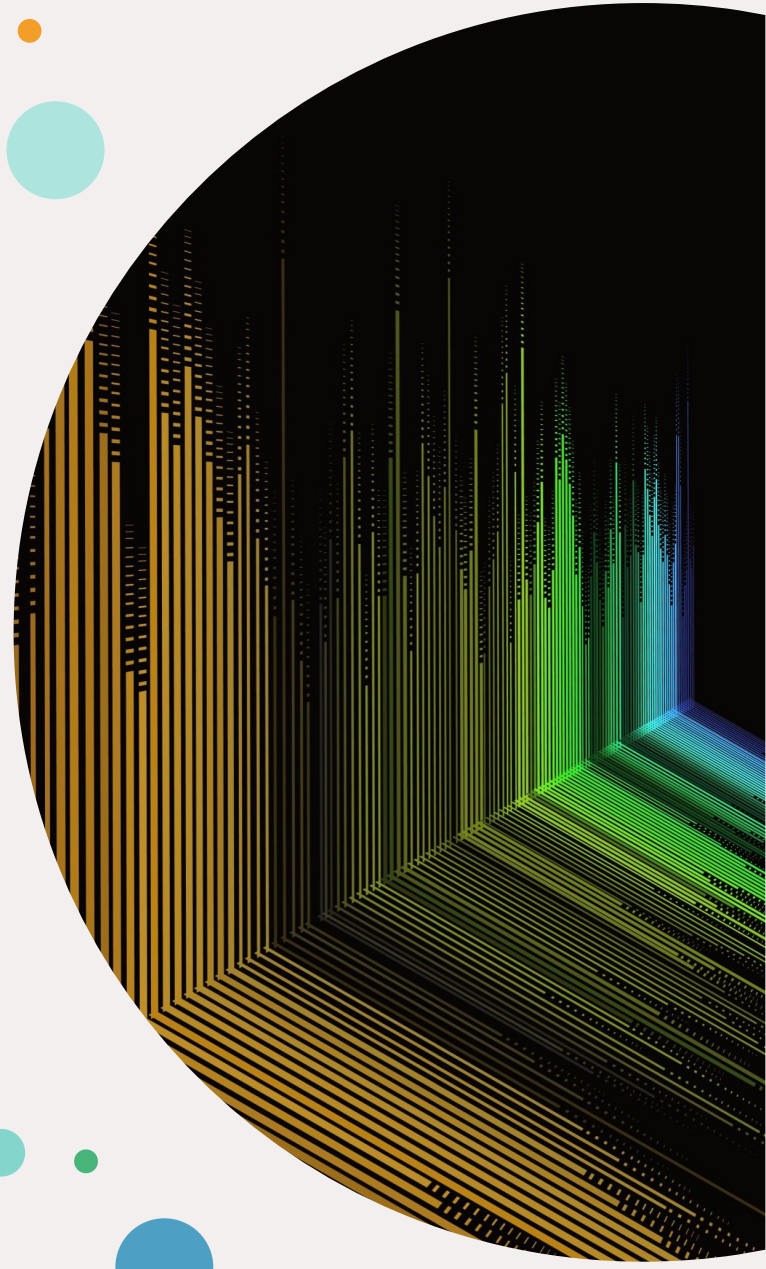
Home Automation made cheap and simple



Objectives

Create a Rest API that lets a user create, read and update the status of certain parts

What is a part?

Analog Part: Read only, Sensors like temperature, light humidity etc.	Digital Part: Read and Write: Lights, switches etc.

Objectives

Allow for remote parts to be controllable as well

Technologies used

- Go as base programming language
- Gin Gonic Library for REST API
- Go-rpio for GPIO Pins used on the Raspberry Pi
- MQTT with the help of adafruit.io
- C and lots of Arduino Libraries for the remote part

Hardware used

- Raspberry Pi 4 as the brain of the system
- ESP8266 for the remote part
- Resistors and LEDs for the binary parts
- Potentiometers to simulate analog parts

Development: The Raspberry Pi

- The REST API is developed in Go using the Gin Gonic Framework. This is deployed to a raspberry pi and handles all the existing parts, their status and can toggle the digital parts and the read from the analog parts. This Go program either directly toggles an LED on the Pi, or sends the request off to the adafruit.io MQTT host for the ESP8266 to handle. It then returns data in JSON format to the requester.

Development: Adafruit.io

- Very Simple and Free
- Acts as a MQ-Broker (specifically for MQTT)
- Ensures communication from Pi to ESP8266 (waaaaaay easier than implementing it yourself)

Development: ESP8266

- Handles remote parts (i.e. parts in another room)
- Only needs access to the internet, can be in a different network than the Pi
- Programmable using the Arduino IDE and Arduino-flavored C
- Very good development boards for low price available

Demo!