CONTIKI-NG and NODE RED

# Introduction

IoT devices are used to measure temperature and humidity every T=10 sec. A sliding window is applied that computes the average of the last six readings. Should the value of the average exceed a certain threshold K, the raw readings are reported instead of the average obtained from the sliding window.

The average or the readings are reported to the backend. At the backend, information on the hottest, coolest, and most/least humid day of the month is kept in a log that is periodically communicated via email to a specic address.

This log is persistently stored so rebooting the backend will not make the system lose the data gathered until that time.

# Approach and Assumptions

To tackle this problem to technologies were used, one for the IoT devices and one for the backend, these are Contiki-ng and Node Red.

### Contiki-ng

---contiki---

### Node Red

The technology used for the backend is Node Red, this was an easy choice thanks to its ease of use and the integrated nodes to receive the data, send them via email and create a persistent storage.

--parlare di mqtt—

The persistency was achieved saving the log in a json file every time it is updated, but since the system is continuously computing the max and min temperature and humidity for the current month this should not be lost neither. Another json file is saved containing this information, the cumulative temperature and humidity of the current day, the number of readings and the day.

# Design

# Algorithm

# Test

### Installation

Once Node Red is running the only thing that has to be done before the system is operational is launching the prestart flow, this will create the temp.json file that is used to load the initial values in the main function. However this should be done just once, if the system fail and is rebooted the prestart should not be launched else the stored information will be overwritten.

### Performances

### Challenges

# Conclusion