

# Smart & Secure Home

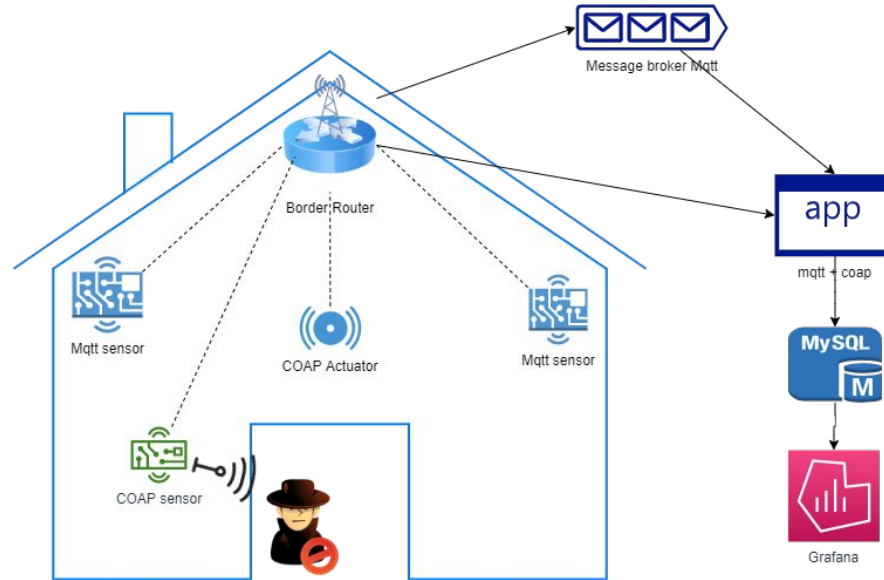
IOT project

# Project goal:

Our project aims to create a home monitoring system , environmental factors and an intrusion detection system. All the data are then saved in a database.

"**mqtt**" sensors take care of the monitoring part of gas , lighting , temperature and humidity.

"**Coap**" sensors deal with the intrusion system, as soon as the motion sensors sense the intruder , the alarm is triggered and gradually increases in intensity. To manually stop the alarm, just press the button of the sensor emitting the alarm.



01

MQTT

how we operated

...

02

CoAP

How we operated

...

03

Grafana

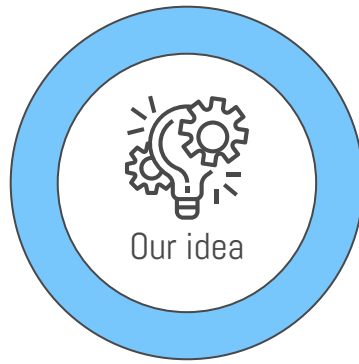
Dashboard

...



# 01

## Mqtt



## Process:

The mqtt sensors take care of monitoring some parameters including gas, light , temperature and humidity in the room.

Once the connection is made, the data is sent in json format , on the topic "info". Not having the actual sensors , the data are randomly generated with the rand() function.

```
temp = rand() % 40;  
humidity = rand() % 101;  
light = rand() % 3;  
gas = rand() % 100;
```

This is the message sent to the broker:

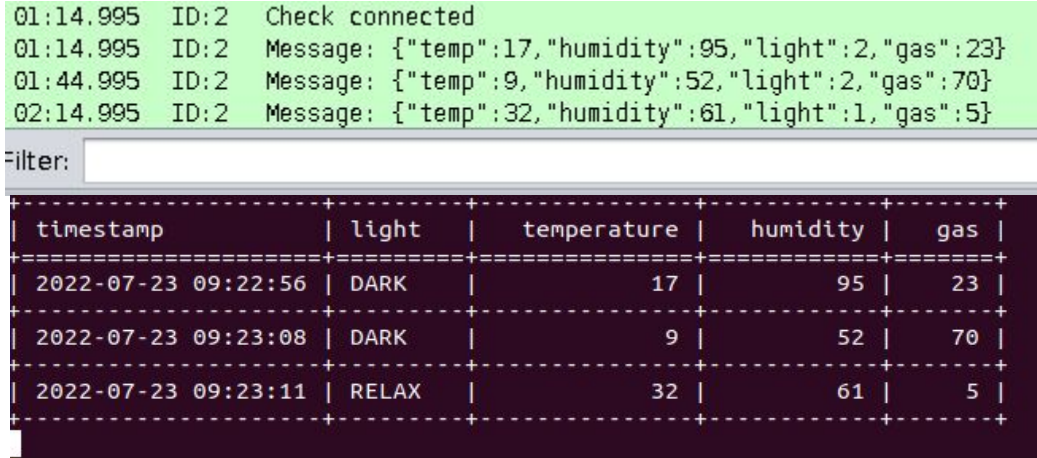
```
sprintf(app_buffer, "{\"temp\":%d,\"humidity\":%d,\"light\":%d,\"gas\":%d}\", temp, humidity, light, gas);
```

As soon as the app receives the data via json (onMessage()) from the mqtt sensors , it extracts the data and makes a query to the SQL Database to insert it into the table "mqttensors" , with an associated timestamp , useful for graph visualization.

```
sql = "INSERT INTO `mqttensors` (`temperature`, `humidity`, `light`, `gas`) VALUES (%s, %s, %s, %s)"
```


A thread is created for the mqtt client.

Testing on cooja , you can see that messages are sent and received correctly:



The image shows a screenshot of the Cooja network simulator. At the top, a log window displays four MQTT messages sent from ID:2. Below the log, there is a 'Filter:' input field. At the bottom, a table represents the 'mqttensors' database table, showing the data received from the sensors.

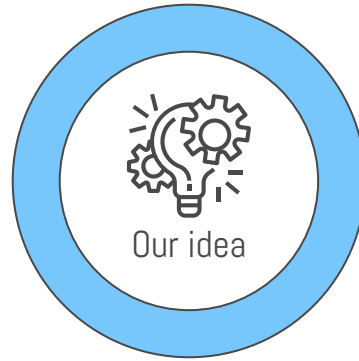
timestamp	light	temperature	humidity	gas
2022-07-23 09:22:56	DARK	17	95	23
2022-07-23 09:23:08	DARK	9	52	70
2022-07-23 09:23:11	RELAX	32	61	5



# 02

## CoAP





## Process:

"Coap" sensors take care of the alarm system; we have the motion sensors placed in various parts of the house, which detect movement and start an alarm when movement is detected.

As soon as the state of the resource changes , a query is made to the MySQL database of the sensor , and a POST is made to the alarm to turn the alarm on or off.

```
response = self.client.post(self.actuator_resource,"OFF")
```

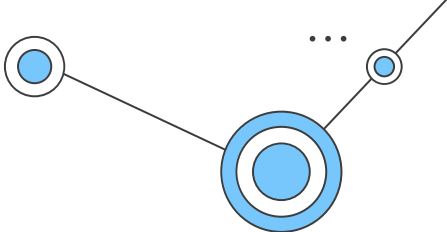


When an intruder is detected, the alarm is activated. It emits a variable intensity sound , in fact the intensity is increased according to the time of the intruder detection.

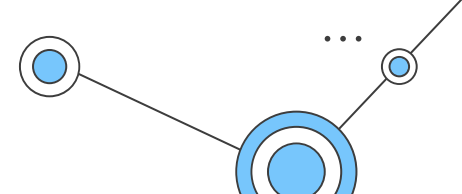
Communication via json was also used for these sensor types , here is an example of creating the GET message of the alarm.

It is saved in the Database

To disable the alarm and motion detection in manual mode, you can press the button of any sensor.



```
char val2 = isActive == 1 ? 'T' : 'N';
strcpy( Dest: msg, Source: "{\"info\": \"\"");
strncat( Dest: msg, Source: &val2, Count: 1);
//strcat(msg, "\" \");
strcat( Dest: msg, Source: "\", \"intensity\": \"");
char intensity_str[400];
sprintf(intensity_str, \"%d\", intensity);
//printf(\"intensity: %s\\n\", intensity_str);
strcat( Dest: msg, Source: intensity_str);
strcat( Dest: msg, Source: \"}\"");
printf(\"MSG: %s\\n\", msg);
```



# Testing on cooja:

1. we see the motion sensor changing state:
2. The server receives the status change and activates the alarm sensors:
3. The server makes a query to the database with "isDetected = 1" and the timestamp:
4. The alarm is activated with an intensity that will vary over time:

```
02:58.167 ID:2 Trigger Motion
02:58.167 ID:2 MSG res.c invio : {"isDetected":"T"}
```

```
Callback called, resource arrived
{"isDetected":"T"}
{"isDetected":"T"}
Detection value .py ricevo :
['T']
Attivo allarme .py
```

```
01:46.890 ID:4 Trigger Motion
01:46.890 ID:4 MSG res_motion send : {"isDetected":"N", "info":"N", "intensity":"20"}
01:46.950 ID:4 entered in POST function!
01:46.950 ID:4 [DBG : motion sensor] POST/PUT Request Sent
01:46.950 ID:4 Post handler called
01:47.258 ID:5 Trigger Motion
01:47.258 ID:5 MSG res_motion send : {"isDetected":"N", "info":"N", "intensity":"20"}
01:47.441 ID:3 Message: {"temp":5,"humidity":71,"light":0,"gas":17}
01:49.443 ID:5 entered in POST function!
01:49.443 ID:5 [DBG : motion sensor] POST/PUT Request Sent
01:49.443 ID:5 Post handler called
```

value	alarm	intensity	timestamp
0	N	10	2022-09-08 11:37:09
1	T	10	2022-09-08 11:37:18
1	T	10	2022-09-08 11:37:19
0	N	20	2022-09-08 11:38:18
0	N	20	2022-09-08 11:38:24

# 03

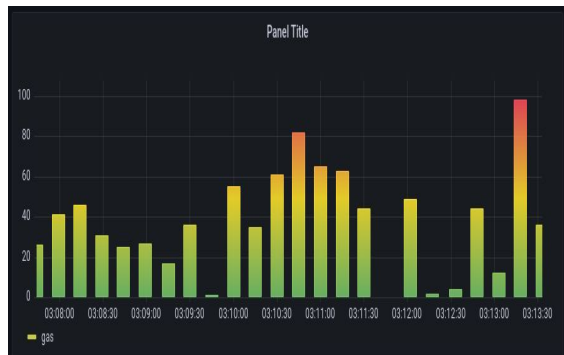
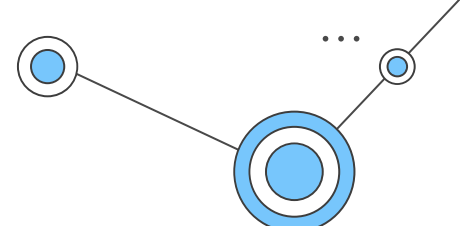
## Grafana

We used Grafana for visualization of the data collected from the sensors and saved in a SQL database , coupling them with timestamp.

Recommended use of Chrome.

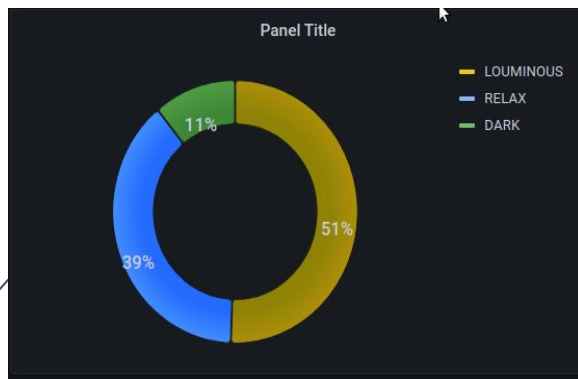
We update the graphs every 30 seconds.

# Mqtt



→ Chart for gas in the room

Chart for Temperature ←

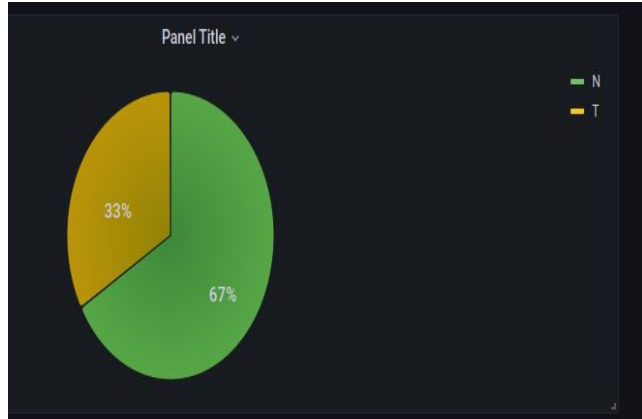
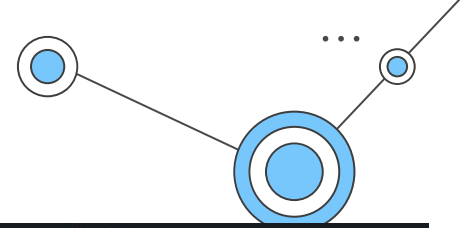


→ Chart for the luminosity of the room

Chart for the humidity ←

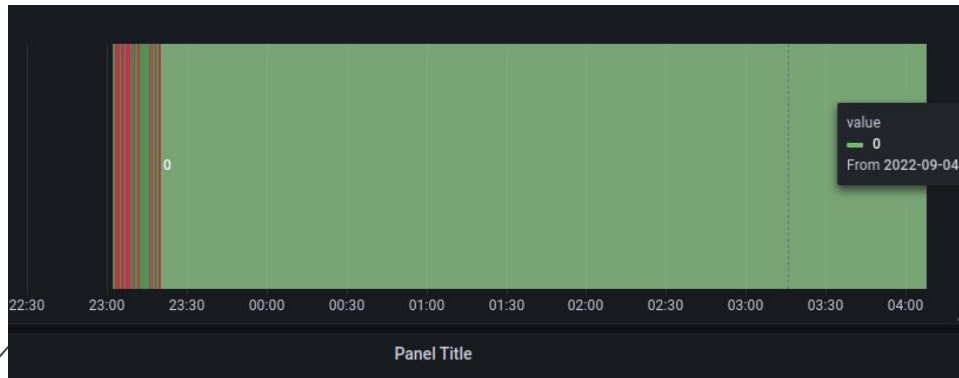
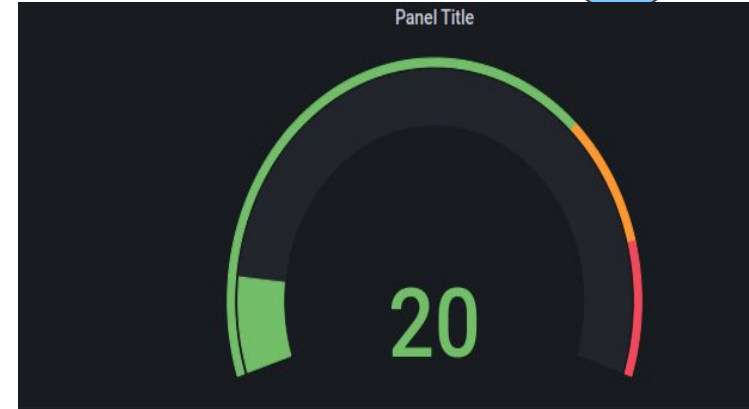


# CoAp



→ Chart for alarm %  
activation over the  
time

Dashboard for the  
intensity ←



→ Chart for the detection of the intrusion