SafeLab Project IoT

Cristian Castiglione 0000895807

May 2021

1 Introduction

In this project we wanted to create a system that manage people access and get environmental parameters through Internet of Things technology. The proposal system allow you to show people number, temperature, humidity and pressure inside room in which things are. In case the values of the things exceed the preset values, the system generate an alert to notify the administrator, through Telegram, that one specific sensor exceeded preset value in order to fix promptly. Furthermore, the system is able to forecast next values of temperature, humidity, pressure and number of people inside the room.

2 Design

To do this project was used these hardware:

- 3 ESP-WROOM-32 (ESP32)

- 2 HC-SR04 (also test with PIR (Passive Infrared Sensor))
- 1 DHT11
- 1 BMP180

To one of the three ESP32 are connected two HC-SR04 sensors: one HC-SR04 is placed at the entrance and one at the exit of the room to count people. To another one ESP32 are connected one DHT11 sensor to measure temperature and humidity and one BMP180 to measure pressure. The last one ESP32 takes care of receive data from other two things and store it on InfluxDB time series database. Finally, all values produced by things are displayed on Grafana, because it shows data in a better way unlike InfluxDB. In this system all things communicate through WebSocket. InfluxDB and Grafana are running on a local Linux machine through docker compose, in order to allow the replication of the two tools on another system. In figure 1 you can see the scheme of the system.

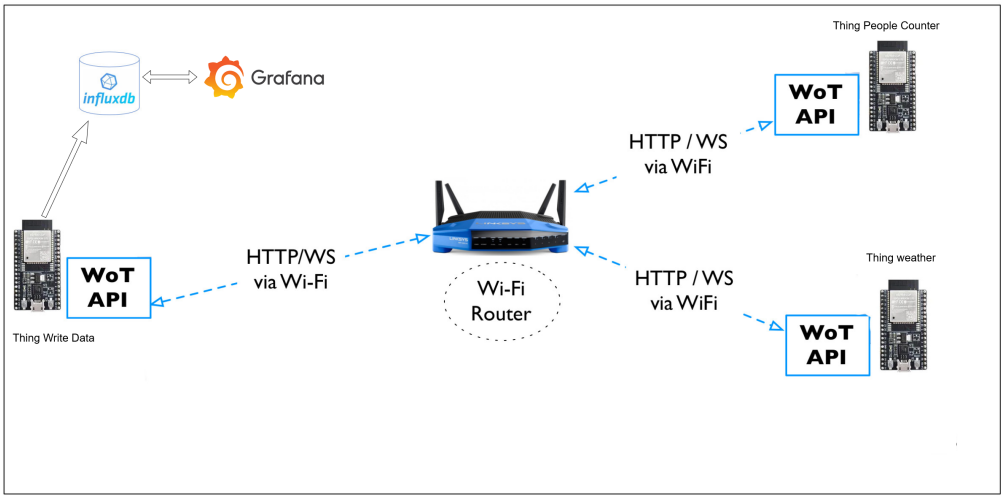

Figure 1: Safe Lab

3 Implementation

The system works through three things:

- Thing Weather: measures environmental parameters like temperature, humidity and pressure;
- Thing People Counter: counts people inside the room;
- Thing Write Data: receives data from other things and stores it on InfluxDB.

All things follow the standard W3C Web Of Things, are developed in Arduino languages (C/C++) and communicate through WebSocket.

3.1 Thing Weather

Thing Weather was developed using DHT11 and BMP180 hardware sensors and to retrieve data from the sensors was used the standard library. Once data are retrieved, Thing Weather sends them to Thing Write Data through WebSocket, this thing also provides a web server that is exposed in the root of port 80, this server shows the Thing Description, you can see in figure 3. In figure 2 you can see the thing.

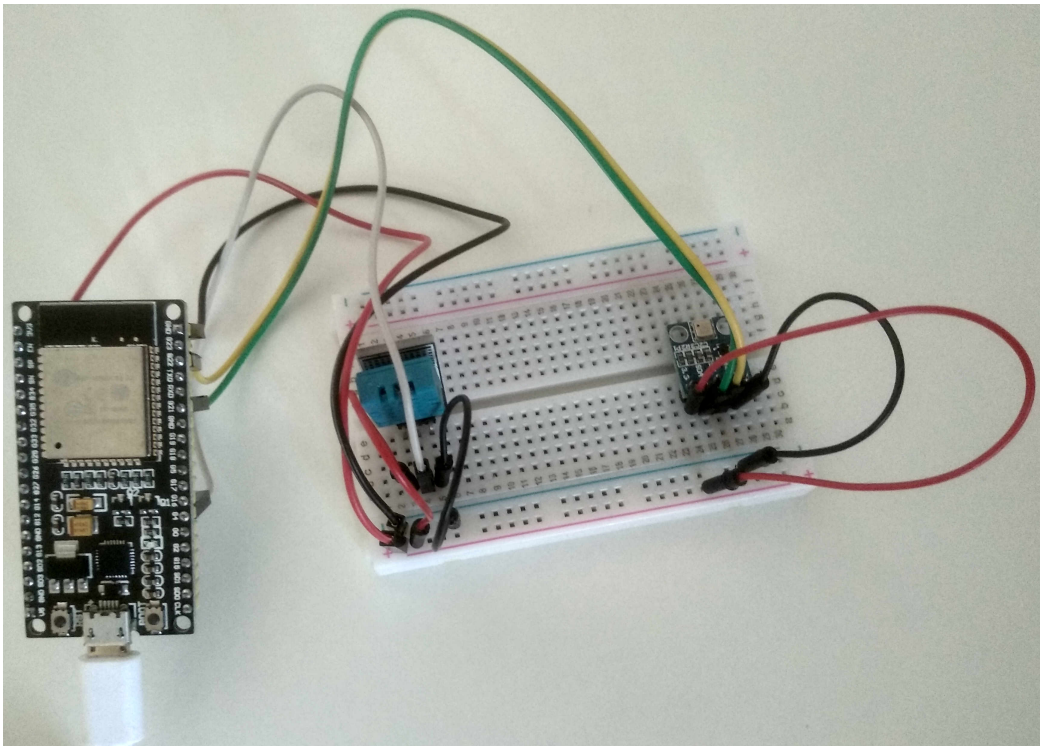

Figure 2: Thing Weather

Figure 3: Thing Description Weather

3.2 Thing People Counter

The first implementation of the Thing People Counter was developed with two PIR sensors and after various test the thing didn't work well because after detecting one motion (HIGH state), must pass about 5 seconds to return to the LOW state, this delay makes the system really limiting.
The solution adopted is to replace PIR sensors with two HC-SR04 sensors. In general HC-SR04 sensor is not used for this purpose but for its characteristics it works discreetly well in this context. Therefore, this thing has one HC-SR04 sensor in entrance location and one in exit location, if one sensor detects reducing distance it means that someone has passed and if the other sensor activates before two seconds counts the increase or decrease of the number of people in the room, based on which sensor activates first, otherwise does no action, even if the both sensors detects someone at the same moment. After detecting a person, the system sends data to Thing Write Data through WebSocket. Also this thing provides a web server exposed in the root of port 80, this server shows the Thing Description as you can see in figure 5. In figure 4 you can see the thing.

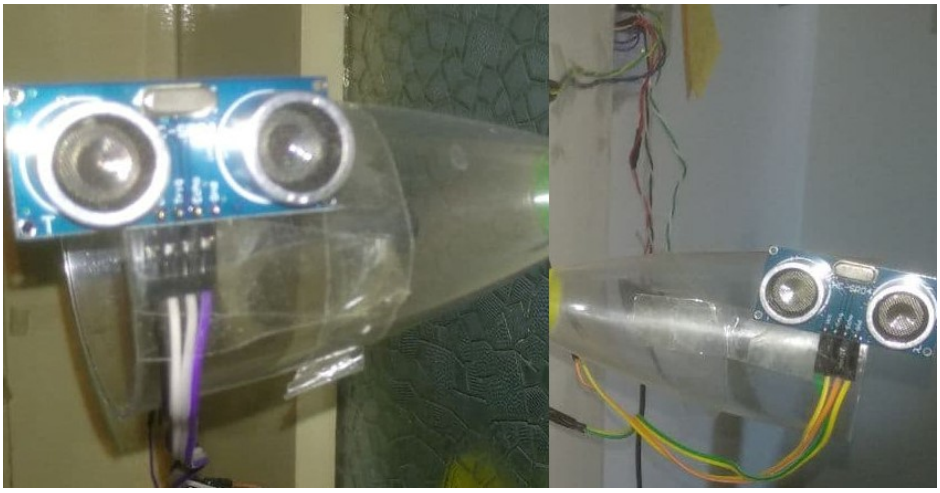

Figure 4: Thing People Counter


```
@context:                "https://www.w3.org/2019/wot/td/v1"
title:                   "ThingPeopleCounter"
description:             "A Thing to control entry and exit of the room"
securityDefinitions:
  nosec_sc:
    scheme:              "nosec"
security:
  0:                     "nosec_sc"
properties:
  numberOfPeople:
    description:         "Shows the current number of people in the room"
    type:                "string"
    readOnly:            true
    forms:
      0:
        op:              "readproperty"
        href:            "http://192.168.1.4/properties/numberofpeople/"
        contentType:     "application/json"
```


Figure 5: Thing Description People Counter

3.3 Thing Write Data

Last thing, Thing Write Data, was developed to receive data from other two things through WebSocket and to store it on InfluxDB. The thing stores data on InfluxDB through standard library provided by InfluxDB. Moreover, the thing can also receive data from other things not present in the system as long as they respect the required input described in Thing Description, in figure 7 you can see the input type. Also this thing provides a web server exposed in the root of port 80, this server shows the Thing Description as you can see in figure 6. In this thing there are no sensors connected.

Figure 6: Thing Description Write Data

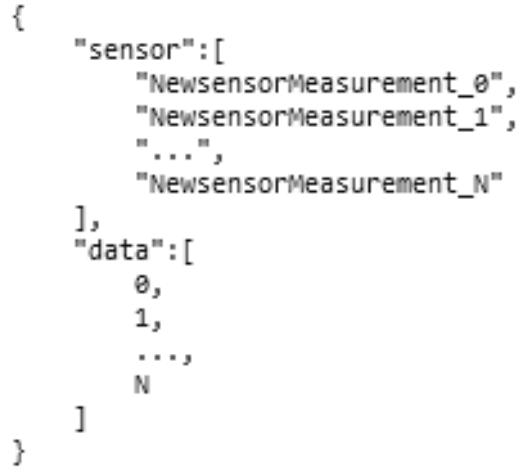

Figure 7: Example Json to be sent to the thing Write Data

This system communicates with the Thing Write Data with WebSocket server while Thing Weather and Thing People Counter with WebSocket client.

3.4 InfluxDB and Grafana

InfluxDB and Grafana are running on a local Linux machine, Thing Write Data writes data to InfluxDB and Grafana retrieves these data to show them in a better way. To avoid saturating all the physical memory of the Linux machine, a retention policy was configured through the tasks available on InfluxDB, specifically, every 24 hours the task aggregates data produced from the things in a hourly interval and stores it in new bucket, after this operation it is possible to delete the data from the main bucket by setting

it from the control panel. For now, the data is not deleted from the main bucket because there are few data that are useful for analyzing forecasts. This feature may be activated in the future.

3.5 Alert

With Grafana it was possible to create an alert system that notifies the room administrator through a telegram group that one of the sensors has exceeded a certain predetermined threshold or the system no longer sends data.

3.6 Forecasting

The time series stored in InfluxDB are also used to make predictions of temperature, humidity, pressure and number of people in the room. To make predictions was used ARIMA (Auto Regressive Integrated Moving Average) and Holt Winters. All software for predictions and predictions analysis are developed in Python language. First of all, the series was analyzed to check if it was stationary or non-stationary, through the use of the Dickey-Fuller test, the series analyzed contain 10 days of data measurements in the room. Dickey-Fuller test results are: temperature is non-stationary, humidity is non-stationary, pressure is non-stationary and people is stationary.
Another analysis checks the components of the time series, the figures of analysis are illustrated below.

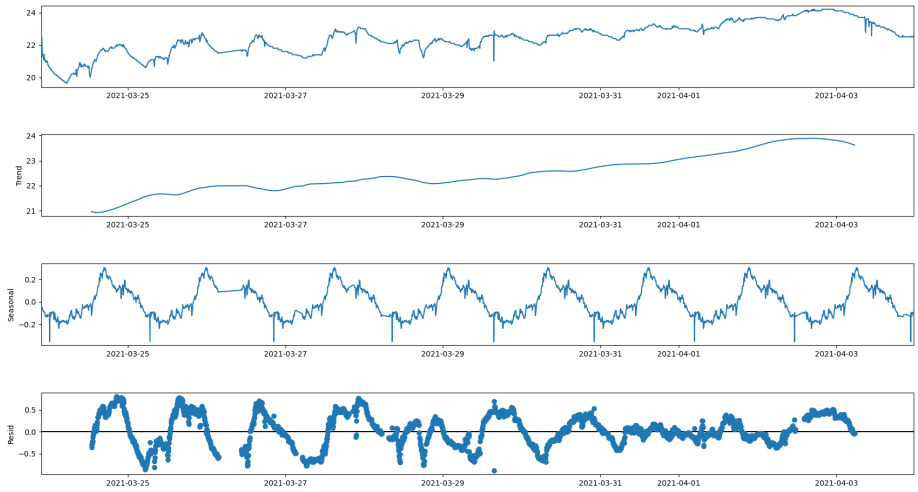

Figure 8: Temperature analysis

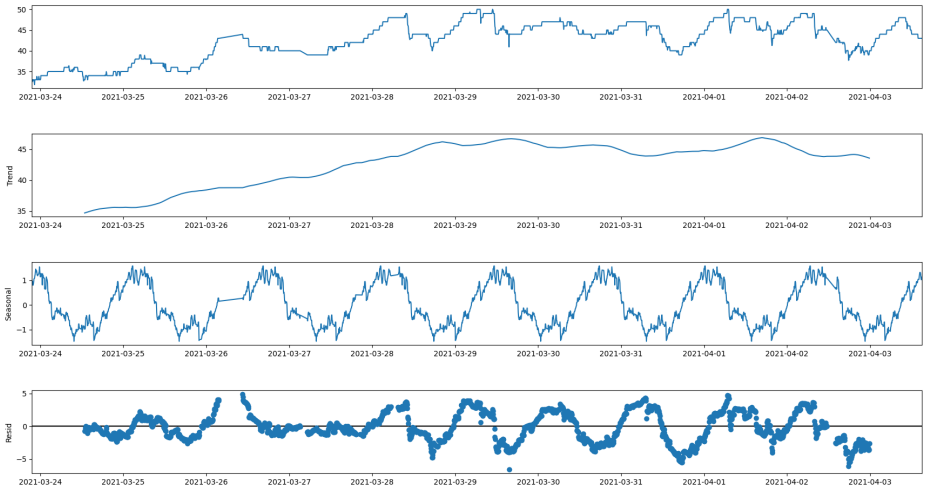

Figure 9: Humidity analysis

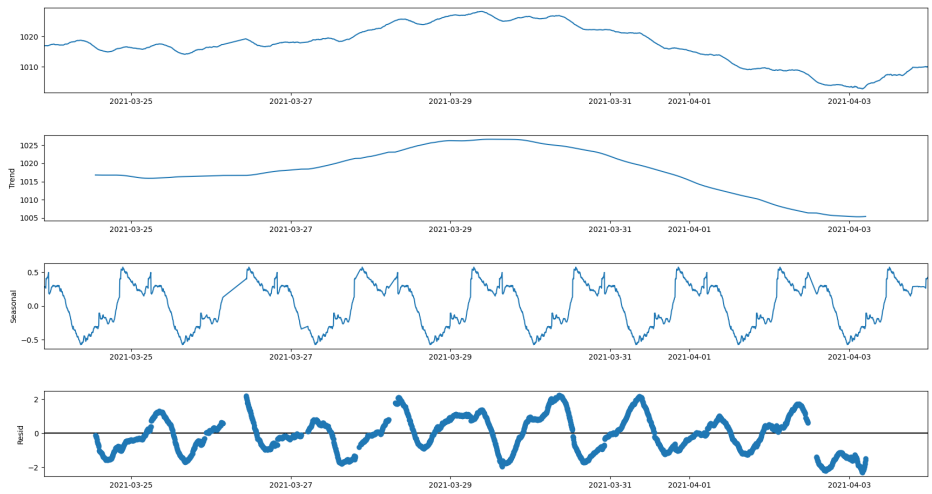

Figure 10: Pressure analysis

Figure 11: People analysis

After checking these informations, the series were split into train and test,

about 80% and 20% respectively, to make predictions on the test and finally to find the Mean squared error (MSE) and Mean absolute error (MAE) on the predictions of these results.
To find good ARIMA order was used auto_arima function provided by pmdarima library that automatically discovers the optimal order for an ARIMA model. For these analysis the ARIMA order and the values of the errors are showed below:

- Temperature:
  - ARIMA: (2,1,2)
  - MSE: 0.2906079194524305
  - MAE: 0.47521125433524575
- Humidity:
  - ARIMA: (1,1,1)
  - MSE: 13.633365584508304
  - MAE: 2.935123618864821
- Pressure:
  - ARIMA: (0,2,1)
  - MSE: 36.96538895032698
  - MAE: 4.675381083713221
- People:
  - ARIMA: (1,0,1)

– MSE: 0.2797485319907423

– MAE: 0.4601522146607416

The pressure has a high MSE in ARIMA and even using the Holt Winters method it is high, by varying the order of the ARIMA model the error remains high. Probably having more data available this value could decrease or we could get good results using other methods. Humidity also has a very high MSE which, however, is lower using the Holt Winters method, but still high. Finally, the temperature and the number of people have a low error. In the figure below you can see the graphs produced, the blue line is the expected data and the orange line is the forecasting data

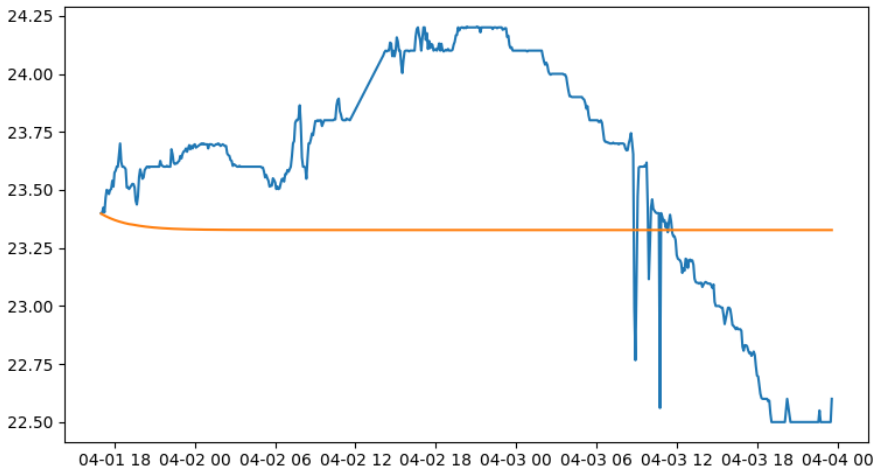

Figure 12: Temperature ARIMA predictions analysis

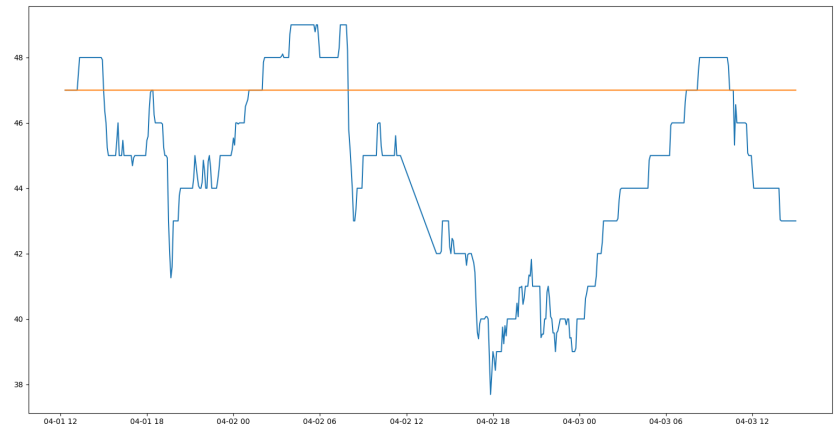

Figure 13: Humidity ARIMA predictions analysis

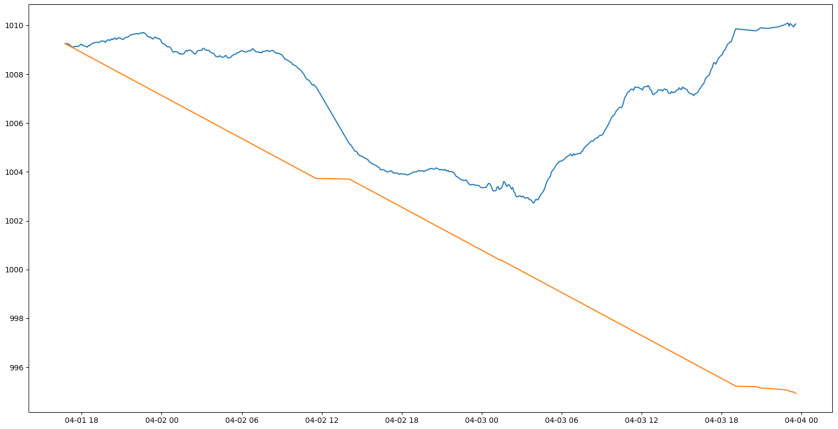

Figure 14: Pressure ARIMA predictions analysis

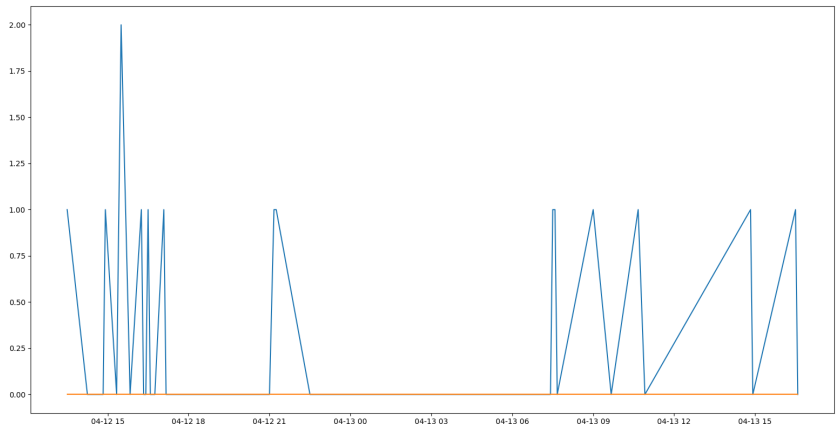

Figure 15: People ARIMA predictions analysis

Also Holt Winters predictions was used to make predictions of temperature, humidity, pressure and number of people in the room. Unlike ARIMA, Holt Winters was used through native function of InfluxDB (holtWinters()). Also in this case was calculated the MSE and MAE. Results and relative graphs are showed below:

- Temperature:
  - MSE: 2.433965934792185
  - MAE: 1.3193971374486675
- Humidity:
  - MSE: 9.809783155494024
  - MAE: 2.4902505502916417

- Pressure:
  - MSE: 193.56735526466124
  - MAE: 12.32965485715925
- People:
  - MSE: 0.6554934823091247
  - MAE: 0.6517690875232774

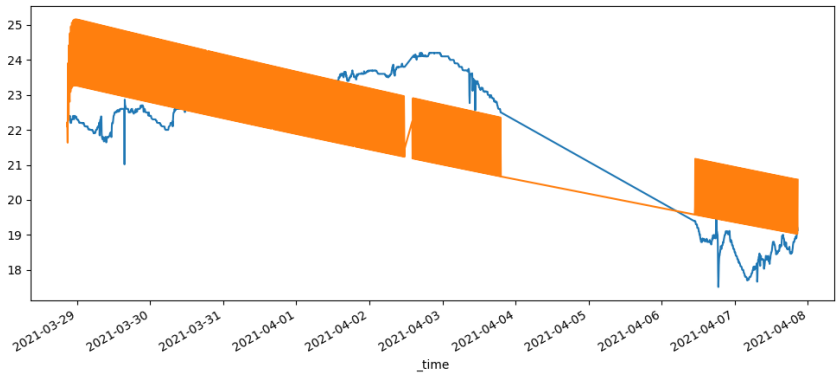

Figure 16: Temperature Holt Winters predictions analysis

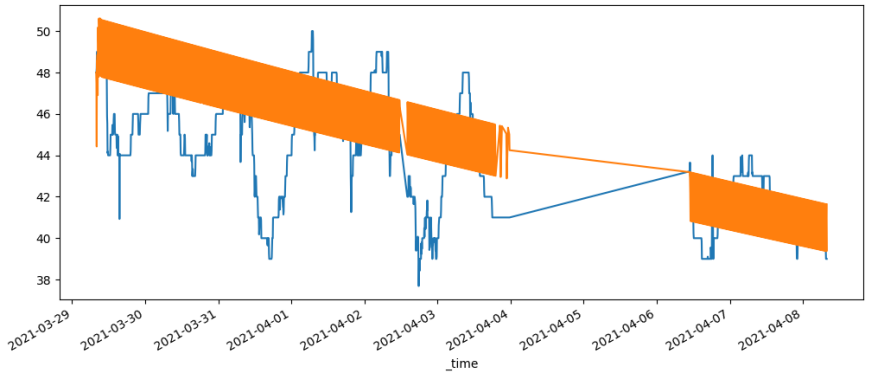

Figure 17: Humidity Holt Winters predictions analysis

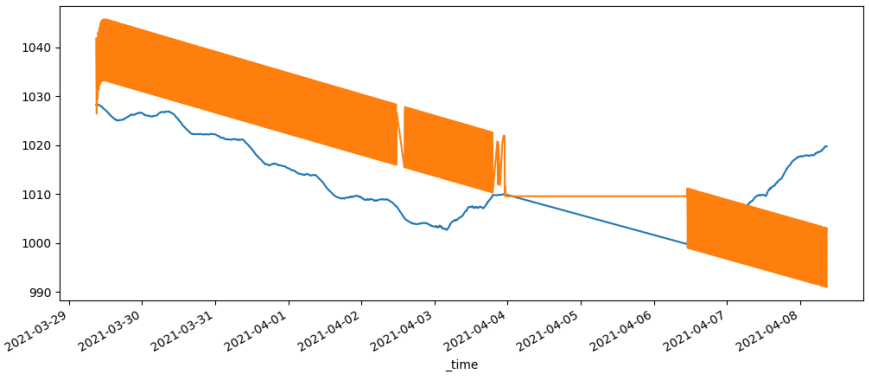

Figure 18: Pressure Holt Winters predictions analysis

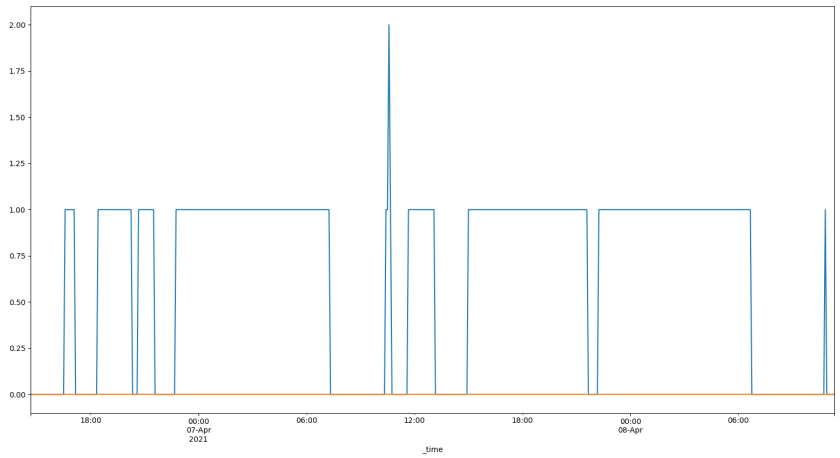

Figure 19: People Holt Winters predictions analysis

After these analysis was developed a forecasting system that runs, through cronjob, every hour the script to predict values for next one hour and stores them in InfluxDB. Forecasting system consists of four script, one for each sensor. Every script gets values from InfluxDB starting from previous 10 days and after clean it runs auto_arima to get the order of model and put it in ARIMA function to fit it and later runs prediction function to make predictions for next one hour with five minute frequency and finally stores new dataframe in InfluxDB.

4 Results

The results obtained from this system are all in all good. The biggest problem encountered was with the Thing People Counter developed with PIR sensors,

after identifying the limits of these sensors the Thing People Counter was developed with HC-SR04 sensors that resolved the main problem caused by PIR sensors. In any case, better hardware sensors should be used to control accesses. The limited number of measurements certainly does not guarantee the absolute effectiveness of forecasting methods but in this case and in some things it is obtained a low error.

5 Bibliography

1. https://github.com/gilmaimon/ArduinoWebsockets
2. https://arduinojson.org/
3. https://www.arduino.cc/
4. https://www.influxdata.com/
5. https://grafana.com/
6. https://www.w3.org/WoT/
7. http://alkaline-ml.com/pmdarima/
8. https://www.statsmodels.org/stable/index.html
9. https://otexts.com/fpp2/
10. https://pandas.pydata.org/
11. https://www.docker.com/
12. https://telegram.org/