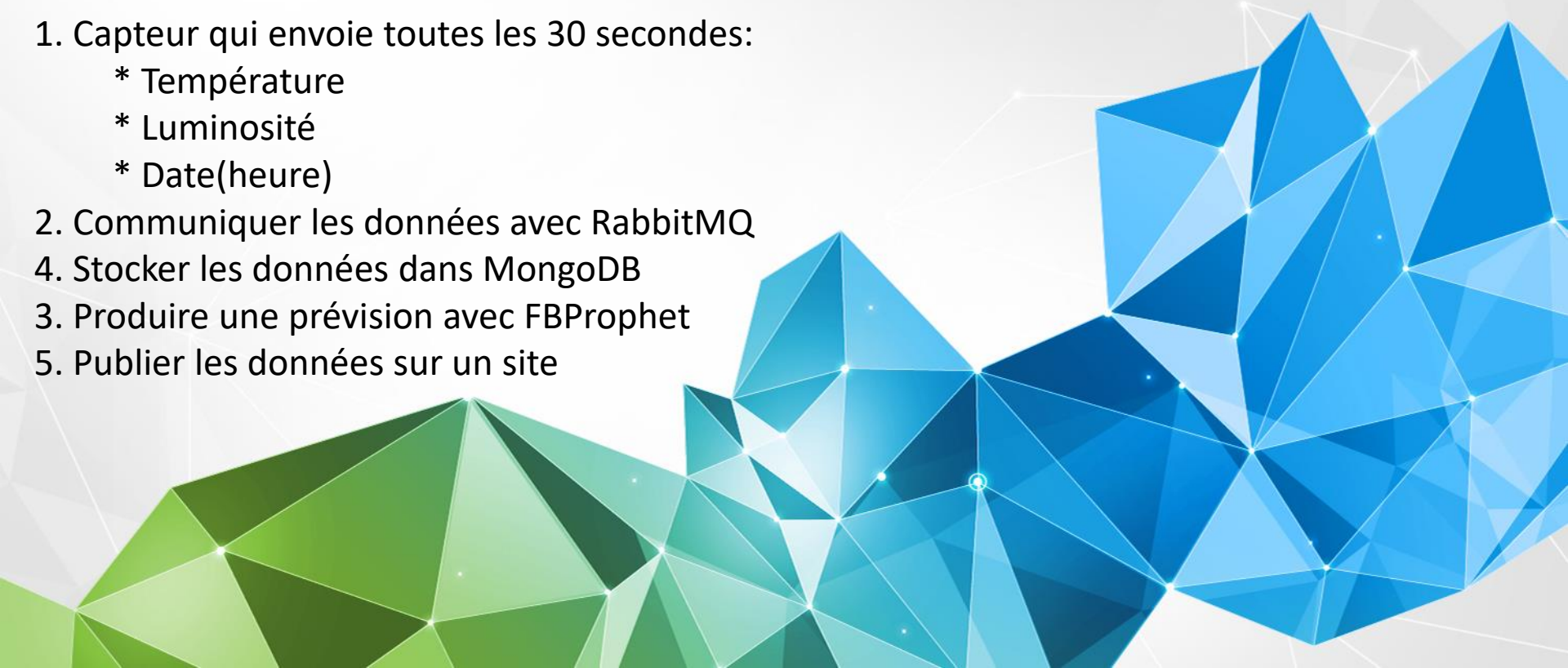
Données & Objet connectés

Projet de fin de module

1. Capteur qui envoie toutes les 30 secondes:
    * Température
    * Luminosité
    * Date(heure)
2. Communiquer les données avec RabbitMQ
4. Stocker les données dans MongoDB
3. Produire une prévision avec FBProphet
5. Publier les données sur un site



PLAN

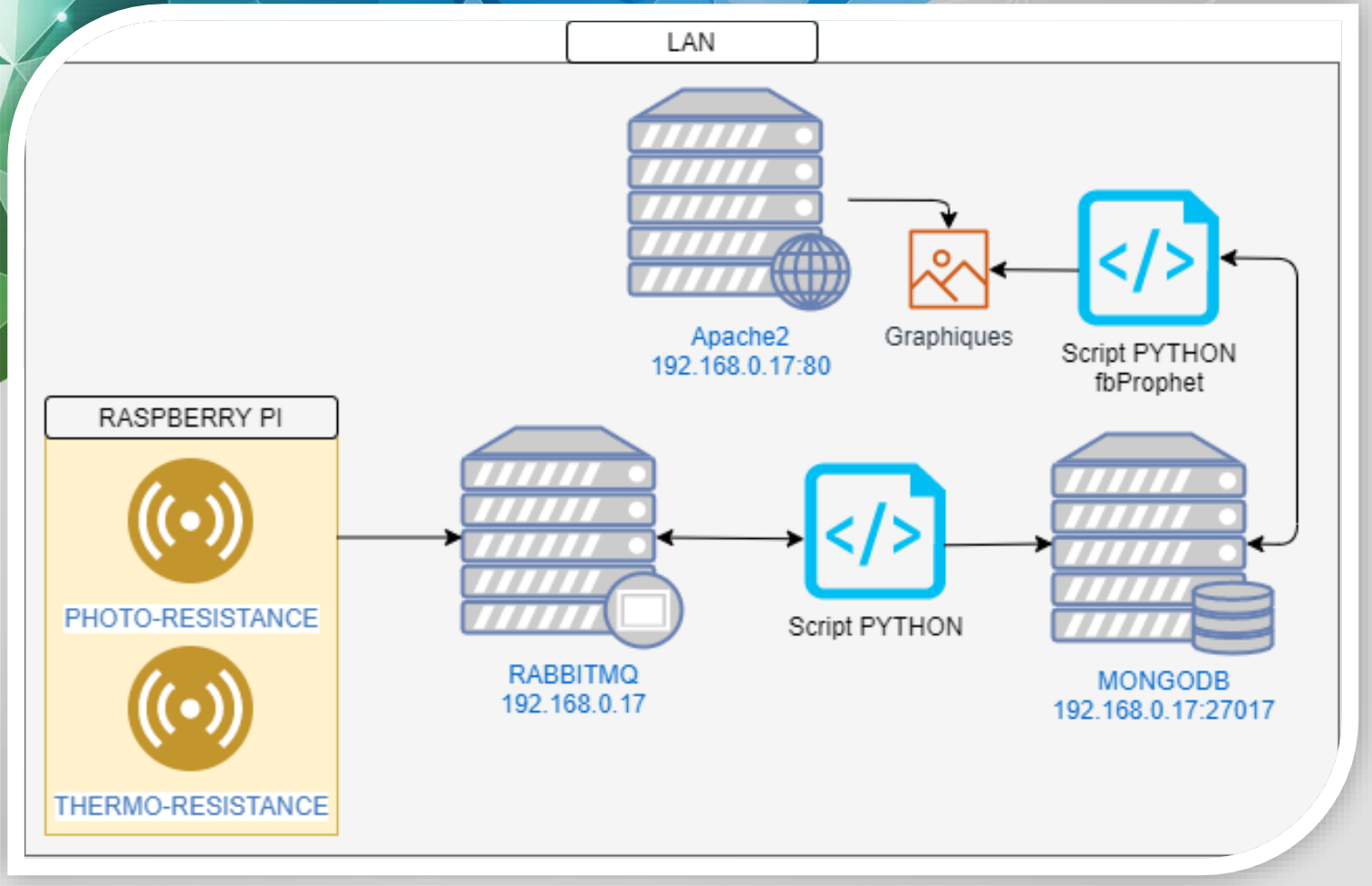
1 – Récupérer

2 – Communiquer

3 – Stocker

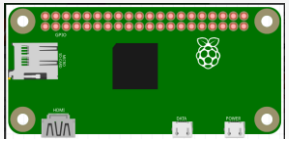
4 – Analyser

SCHEMA


LAN
Apache2
192.168.0.17:80
Graphiques
Script PYTHON
fbProphet
RASPBERRY PI
PHOTO-RESISTANCE
THERMO-RESISTANCE
RABBITMQ
192.168.0.17
Script PYTHON
MONGODB
192.168.0.17:27017

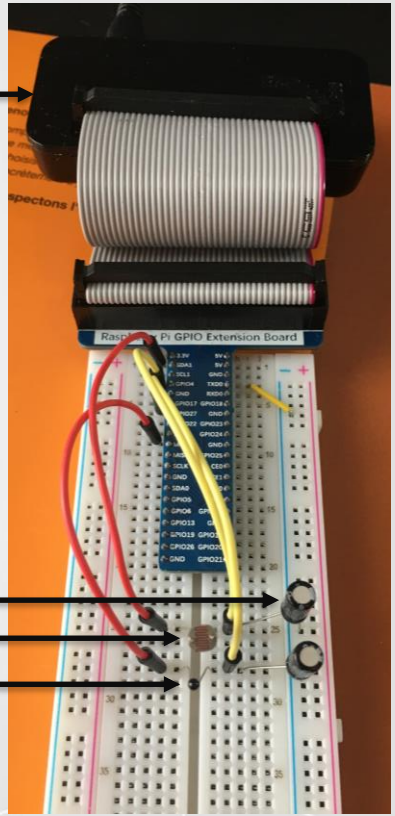
→ Mise en place des capteurs
- Partie : HARDWARE

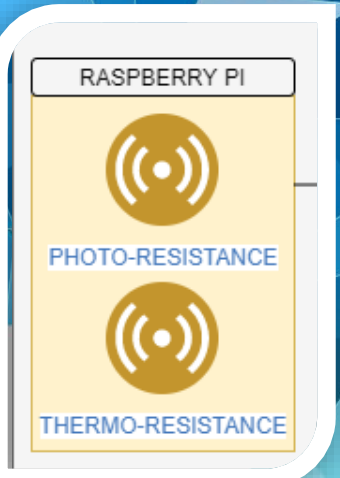
Récupérer



- 1Ghz, Single-core CPU
- 512MB RAM
- Mini HDMI and USB On-The-Go ports
- Micro USB power
- HAT-compatible 40-pin header

Raspberry pi zéro w





Condensateurs 50v 100uF

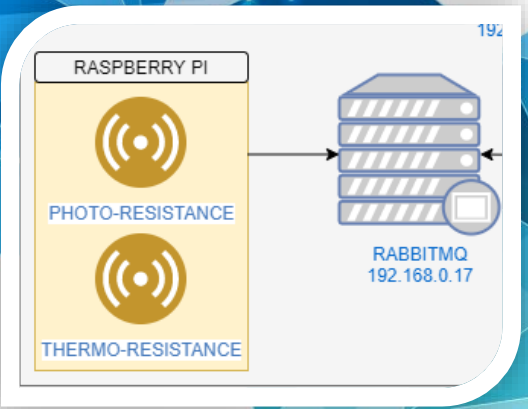
Photo-résistance

Thermo-résistance

→ Mise en place des capteurs
- Partie : SOFTWARE

Répartition des PIN sur le RPI

```
ct@rbpizero:~ $ gpio readall
 +-----+-----+---------+------+---+-Pi ZeroW-+---+------+---------+-----+-----+
 | BCM | wPi |   Name  | Mode | V | Physical | V | Mode | Name    | wPi | BCM |
 +-----+-----+---------+------+---+----++----+---+------+---------+-----+-----+
 |     |     |    3.3v |      |   |  1 || 2  |   |      | 5v      |     |     |
 |   2 |   8 |   SDA.1 |   IN | 1 |  3 || 4  |   |      | 5v      |     |     |
 |   3 |   9 |   SCL.1 |   IN | 1 |  5 || 6  |   |      | 0v      |     |     |
 |   4 |   7 | GPIO. 7 |   IN | 1 |  7 || 8  | 0 | IN   | TxD     |  15 |  14 |
 |     |     |      0v |      |   |  9 || 10 | 1 | IN   | RxD     |  16 |  15 |
 |  17 |   0 | GPIO. 0 |   IN | 1 | 11 || 12 | 0 | IN   | GPIO. 1 |   1 |  18 |
 |  27 |   2 | GPIO. 2 |   IN | 0 | 13 || 14 |   |      | 0v      |     |     |
 |  22 |   3 | GPIO. 3 |   IN | 0 | 15 || 16 | 0 | IN   | GPIO. 4 |   4 |  23 |
 |     |     |    3.3v |      |   | 17 || 18 | 0 | IN   | GPIO. 5 |   5 |  24 |
 |  10 |  12 |    MOSI |   IN | 0 | 19 || 20 |   |      | 0v      |     |     |
 |   9 |  13 |    MISO |   IN | 0 | 21 || 22 | 0 | IN   | GPIO. 6 |   6 |  25 |
 |  11 |  14 |    SCLK |   IN | 0 | 23 || 24 | 1 | IN   | CE0     |  10 |   8 |
 |     |     |      0v |      |   | 25 || 26 | 1 | IN   | CE1     |  11 |   7 |
 |   0 |  30 |   SDA.0 |   IN | 1 | 27 || 28 | 1 | IN   | SCL.0   |  31 |   1 |
 |   5 |  21 | GPIO.21 |   IN | 1 | 29 || 30 |   |      | 0v      |     |     |
 |   6 |  22 | GPIO.22 |   IN | 1 | 31 || 32 | 0 | IN   | GPIO.26 |  26 |  12 |
 |  13 |  23 | GPIO.23 |   IN | 0 | 33 || 34 |   |      | 0v      |     |     |
 |  19 |  24 | GPIO.24 |   IN | 0 | 35 || 36 | 0 | IN   | GPIO.27 |  27 |  16 |
 |  26 |  25 | GPIO.25 |   IN | 0 | 37 || 38 | 0 | IN   | GPIO.28 |  28 |  20 |
 |     |     |      0v |      |   | 39 || 40 | 0 | IN   | GPIO.29 |  29 |  21 |
 +-----+-----+---------+------+---+----++----+---+------+---------+-----+-----+
 | BCM | wPi |   Name  | Mode | V | Physical | V | Mode | Name    | wPi | BCM |
 +-----+-----+---------+------+---+-Pi ZeroW-+---+------+---------+-----+-----+
```



Récupérer

1

```python
import RPi.GPIO as GPIO
import time
from datetime import datetime
import pika

GPIO.setmode(GPIO.BOARD)
```

2

```python
#define the pin that goes to the circuit
temp_pin = 7
light_pin = 11

def rc_time (temp_pin):
    count = 0

    #Output on the pin for
    GPIO.setup(temp_pin, GPIO.OUT)
    GPIO.output(temp_pin, GPIO.LOW)


    #Change the pin back to input
    GPIO.setup(temp_pin, GPIO.IN)
```

3

```python
    #Count until the pin goes high
    while (GPIO.input(temp_pin) == GPIO.LOW):
        count += 1
    return count

#Catch when script is interupted, cleanup correctly
try:
    # Main loop
```

4

```python
    while True:
        credentials = pika.PlainCredentials(username='rbmqUser', password='rbmqPassword')
        connection = pika.BlockingConnection(pika.ConnectionParameters(host='192.168.0.17', credentials=credentials))
        channel = connection.channel()
        channel.queue_declare(queue='firstQueue')
        now = datetime.now()
        dt_string = now.strftime("%d/%m/%Y %H:%M:%S")
        msg = dt_string+"|"+str(rc_time(temp_pin))+"|"+str(rc_time(light_pin))
        channel.basic_publish(exchange='', routing_key='firstQueue', body=msg)
        connection.close()

        print({'date': msg.split('|')[0],'temp': msg.split('|')[1],'lum': msg.split('|')[2]})
        # Wait 5 Minutes
        time.sleep(300)
```

→ Mise en place de RabbitMQ

RabbitMQ

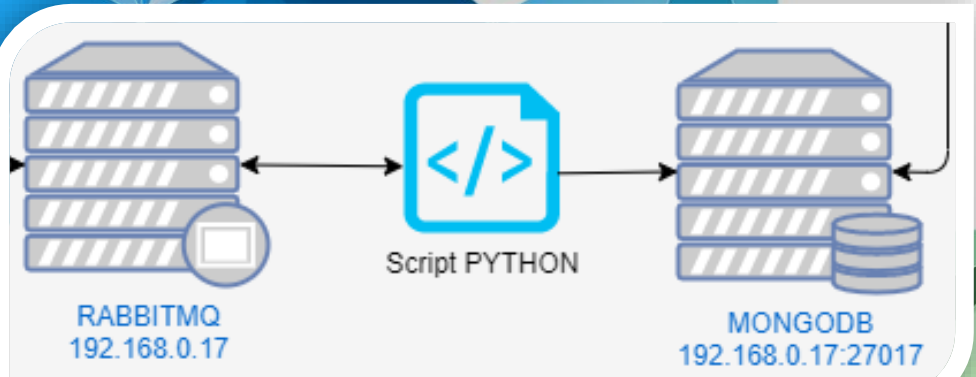
Communiquer

1

2

3

```python
# -*- coding: utf-8 -*-
#!/usr/bin/env python
import pika
from pymongo import MongoClient

credentials = pika.PlainCredentials(username='rbmqUser', password='rbmqPassword')
connection = pika.BlockingConnection(pika.ConnectionParameters(host='192.168.0.17', credentials=credentials))
channel = connection.channel()

channel.queue_declare(queue='firstQueue')

def callback(ch, method, properties, body):
    print(" [x] Received %r" % body)

    client = MongoClient('mongodb://192.168.0.17:27017')
    db = client['epsi_iot']
    post_data = {'ds': body.decode("utf-8").split('|')[0],'y': body.decode("utf-8").split('|')[2]}
    result = db.temp.insert_one(post_data)
    post_data = {'ds': body.decode("utf-8").split('|')[0],'y': body.decode("utf-8").split('|')[1]}
    result = db.lum.insert_one(post_data)

channel.basic_consume(queue='firstQueue', on_message_callback=callback, auto_ack=True)

print(' [*] Waiting for messages. To exit press CTRL+C')
channel.start_consuming()
```



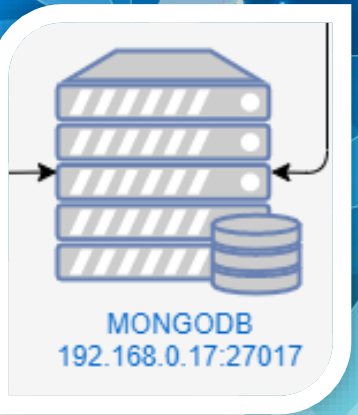
C:\WINDOWS\system32\cmd.exe - RabbitMQ_Sniffer.py

```
[x] Received b'30/03/2020 12:34:29|525|4557'
[x] Received b'30/03/2020 12:34:34|535|4600'
[x] Received b'30/03/2020 12:34:40|530|4531'
[x] Received b'30/03/2020 12:34:45|524|4538'
[x] Received b'30/03/2020 12:34:51|526|4536'
[x] Received b'30/03/2020 12:34:56|521|4594'
[x] Received b'30/03/2020 12:35:02|528|4567'
```

→ Mise en place de MongoDB

Stocker

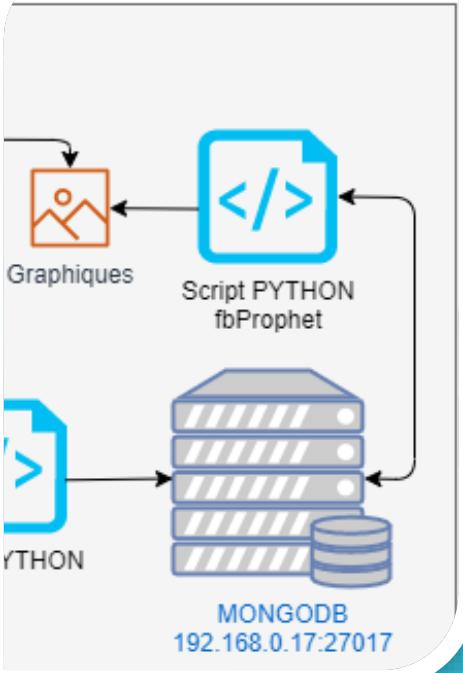
1
```
> use epsi_iot
switched to db epsi_iot
> show collections
```
2
```
lum
posts
temp
```
3
```
> db.temp.find().pretty()
{
        "_id" : ObjectId("5e81df1f024c2a23ec15bd48"),
        "ds" : "30/03/2020 11:59:25",
        "y" : "4338"
}
```



→ Mise en place de fbProphet

PROPHET

Analyser



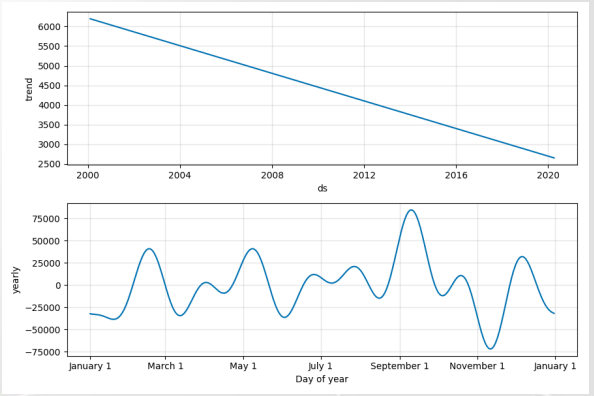
1
```
# -*- coding: utf-8 -*-
#!/usr/bin/env python

import pandas as pd
from fbprophet import Prophet
from pymongo import MongoClient
import matplotlib.pyplot as plt # first line
```

2
```
df = read_mongo(db='epsi_iot',collection='lum',query='',host='192.168.0.17')
df.head()
m = Prophet()
m.fit(df)
```

3
```
future = m.make_future_dataframe(periods=30)
future.tail()
forecast = m.predict(future)
```

4
```
m.plot_components(forecast).savefig('lum.png')
```

Résultat :



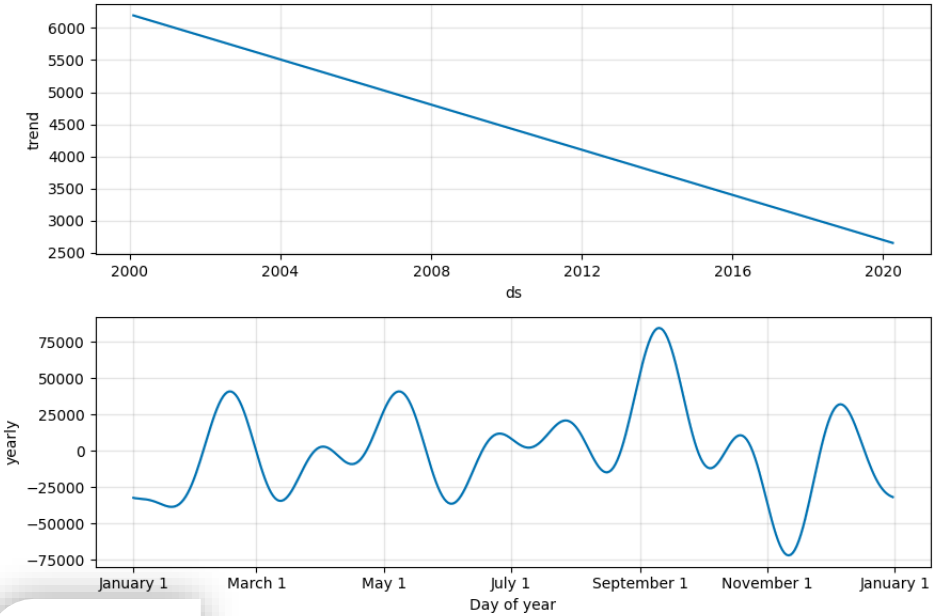
→ Mise en place de fbProphet

Analyser

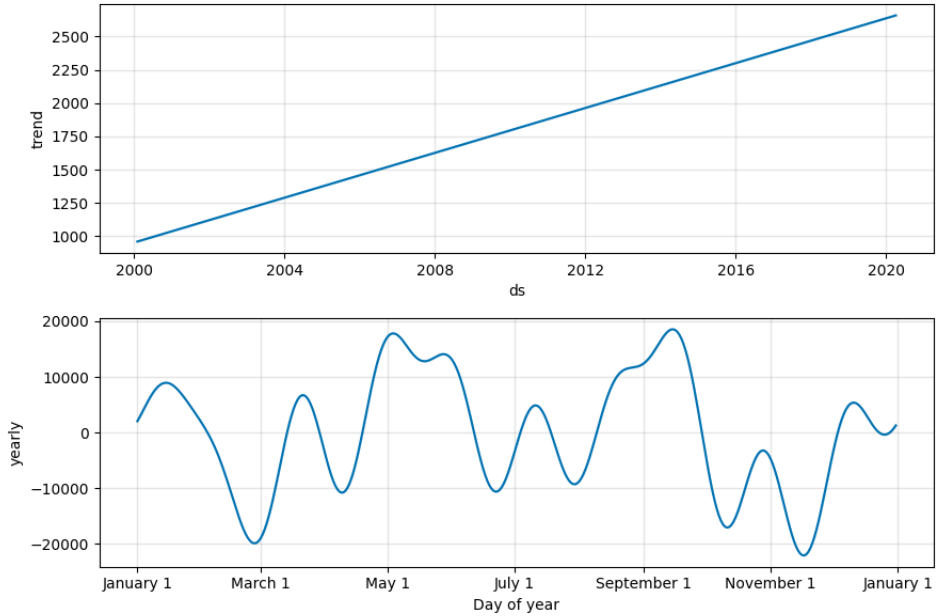
http://192.168.0.17/

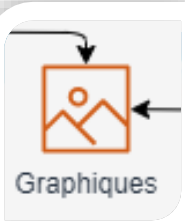
http://sonar.cbarange.ovh/

Graphique Température



Graphique Luminosité

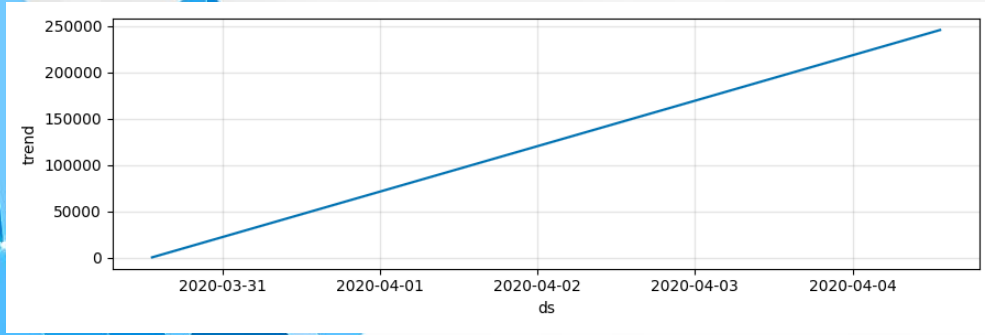




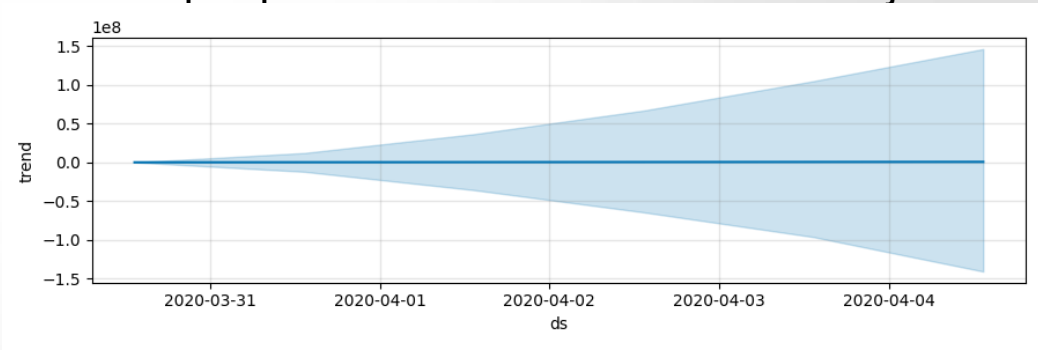
→ Mise en place de fbProphet

Analyser

Graphique Température – Prévision sur 5 jours



Graphique Luminosité – Prévision sur 5 jours



→ Convertir les données en C° ou en Lux

Analyser

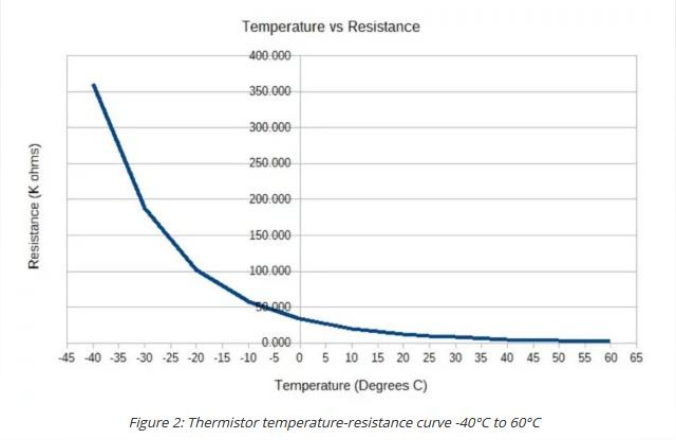
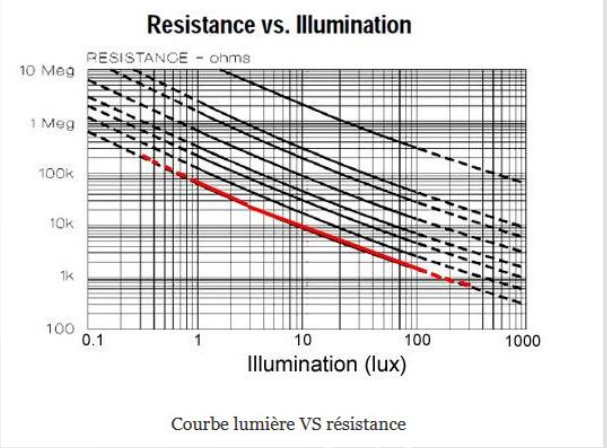


Figure 2: Thermistor temperature-resistance curve -40°C to 60°C



Courbe lumière VS résistance

Graphiques

→ Mise en place d'un job

Automatisation

CRON JOB

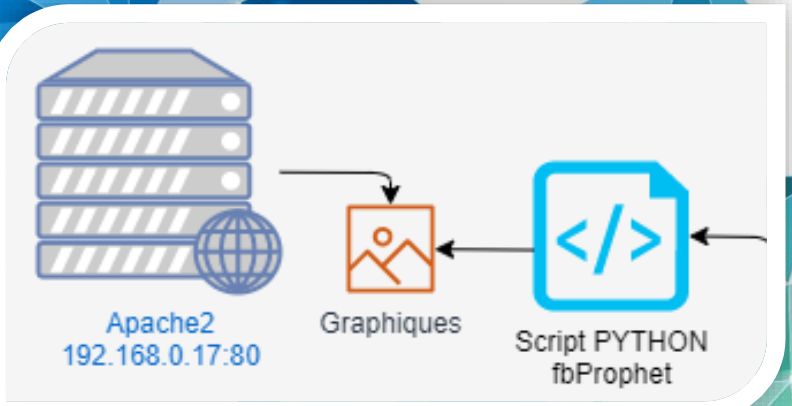
Cron - task:

→ Toutes les 59 minutes

→ Exporte les images sur le site

```
ct@sanctury:/var/www/html$ crontab -l
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single l
# indicating with different fields when the task will b
# and what command to run for the task
#
# To define the time you can provide concrete values fo
# minute (m), hour (h), day of month (dom), month (mon)
# and day of week (dow) or use '*' in these fields (for
# Notice that tasks will be started based on the cron's
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent
# email to the user the crontab file belongs to (unless
#
# For example, you can run a backup of all your user ac
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(
#
# m h  dom mon dow   command
*/59 * * * * python3 -m pytest /gostyle/sonarqube/testa
log
```

```
1 */59 * * * * python3 /home/ct/iot/fbprophet.py
```



ANNEXES

- https://github.com/cbarange/IOT_EPSI_B3
- http://sonar.cbarange.ovh/
- https://www.hackster.io/ahmartareen/iot-temperature-sensor-with-raspberry-pi-2-and-thermistor-7e12db