



---

# RAPPORT SAE 23 – Partie Collecte

---

DESCAMPS Nathan, WAGNER Nicolas et LOSSER Julien

## Table des Matières

Table des Matières .....	1
1) MQTTBOX .....	1
a) Installation .....	1
b) Mise en place du browser & du topic.....	1
1er. MQTT LOAD.....	2
2e. MQTT CLIENT.....	3
2) Python .....	4
a) Csv .....	5

### 1) MQTTBOX

#### a) Installation

Installation depuis Windows Store pour être sûr de la bonne compatibilité

[MQTTBox – Applications du Microsoft Store](#)

#### b) Mise en place du browser & du topic

- Broker : broker.hivemq.com
- Topic : IUT/Colmar/SAE24/Maison2



## 1er. MQTT LOAD

MQTTBox Edit Help

Menu MQTT LOAD TEST SETTINGS Load Settings Help

Load Test Name: GR20

Protocol: mqtt

Host: broker.hivemq.com

Load Test Type: Publish Load Test

# of messages to publish: 50

Run time (seconds): 5

Time Out (seconds): 30

# of instances to run: 2

Topic: IUT/Colmar/SAE24/Maison2

QoS: 1 - Atleast Once

Add payload: Each payload is sent sequentially

Save Delete

Vous pouvez constater que la connections au broker a bien été effectuer et qu'il y a eu aucune erreur lors de la connections.

MQTTBox

MQTTBox Edit Help

Menu Start Load Test View Graph View Data

GR20 - mqtt://broker.hivemq.com ,Test Type=publishing, Msg Count=50, Instances=2, Topic=IUT/Colmar/SAE24/Maison2

Name: Instance 1	Status: Done	Published Time: 5.3540s
Published Messages: 50	QoS Response: 50	QoS Time: 5.3980s
Load test completed successfully	Jun-23-2022 10:57:13:304 AM	✓
Waiting for QoS responses...	Jun-23-2022 10:57:13:260 AM	✓
Publishing completed	Jun-23-2022 10:57:13:260 AM	✓
Publishing messages to topic...	Jun-23-2022 10:57:07:902 AM	✓
Connected to broker	Jun-23-2022 10:57:07:902 AM	✓
Connecting to Broker...	Jun-23-2022 10:56:52:704 AM	✓
Starting MQTT load test	Jun-23-2022 10:56:51:798 AM	✓

Name: Instance 2	Status: Done	Published Time: 5.3710s
Published Messages: 50	QoS Response: 50	QoS Time: 5.4110s
Load test completed successfully	Jun-23-2022 10:57:13:426 AM	✓
Waiting for QoS responses...	Jun-23-2022 10:57:13:382 AM	✓
Publishing completed	Jun-23-2022 10:57:13:382 AM	✓
Publishing messages to topic...	Jun-23-2022 10:57:08:011 AM	✓
Connected to broker	Jun-23-2022 10:57:08:011 AM	✓
Connecting to Broker...	Jun-23-2022 10:56:52:805 AM	✓
Starting MQTT load test	Jun-23-2022 10:56:51:798 AM	✓

Lorsqu'on sort du menu précédent, nous avons un lien rapide vers notre configuration.

MQTTBox Edit Help

Menu MQTT LOAD Create MQTT Load

GR20

mqtt broker.hivemq.com

publishing 2

Nous pouvons voir, quelques informations comme le Nom, que nous avons donné. Le protocol utilisé et le broker.



## 2e. MQTT CLIENT

Comme précédemment, nous devons entrer le protocole mqtt/tcp, le broker dans l'onglet Host et le topic.

N'oubliez pas de « Save » avant de fermer la page.

Vous pouvez voir le nom que nous avons donné, le protocole mqtt avec le broker utiliser.

On est correctement connecté au topic « IUT/Colmar/SAE24/Maison2 »

Les informations publier sont afficher sous la forme id= #,piece=#,date=#,time=#,temp=#

**Id** correspond à l'adresse MAC du capteur, cela est important vu qu'on dispose de 2 capteurs pour pouvoir les identifier comme l'adresse MAC est unique.

**Piece** correspond à la location des capteurs, car chaque capteur est dans un lieu différent, chambre1 ou séjour dans notre cas.

**Date** et **time** correspond au moment de la prise de la capture de la température.

**Temp** correspond à la température prise par le capteur.

## 2) Python

Ci-dessus vous pouvez voir notre script python qui nous permet de récupérer les données du MQTT et de créer un fichier csv avec ces données.

Tout d'abord vérifier que vous avez python 3.6 et au-dessus. Pouvez utiliser la commande :

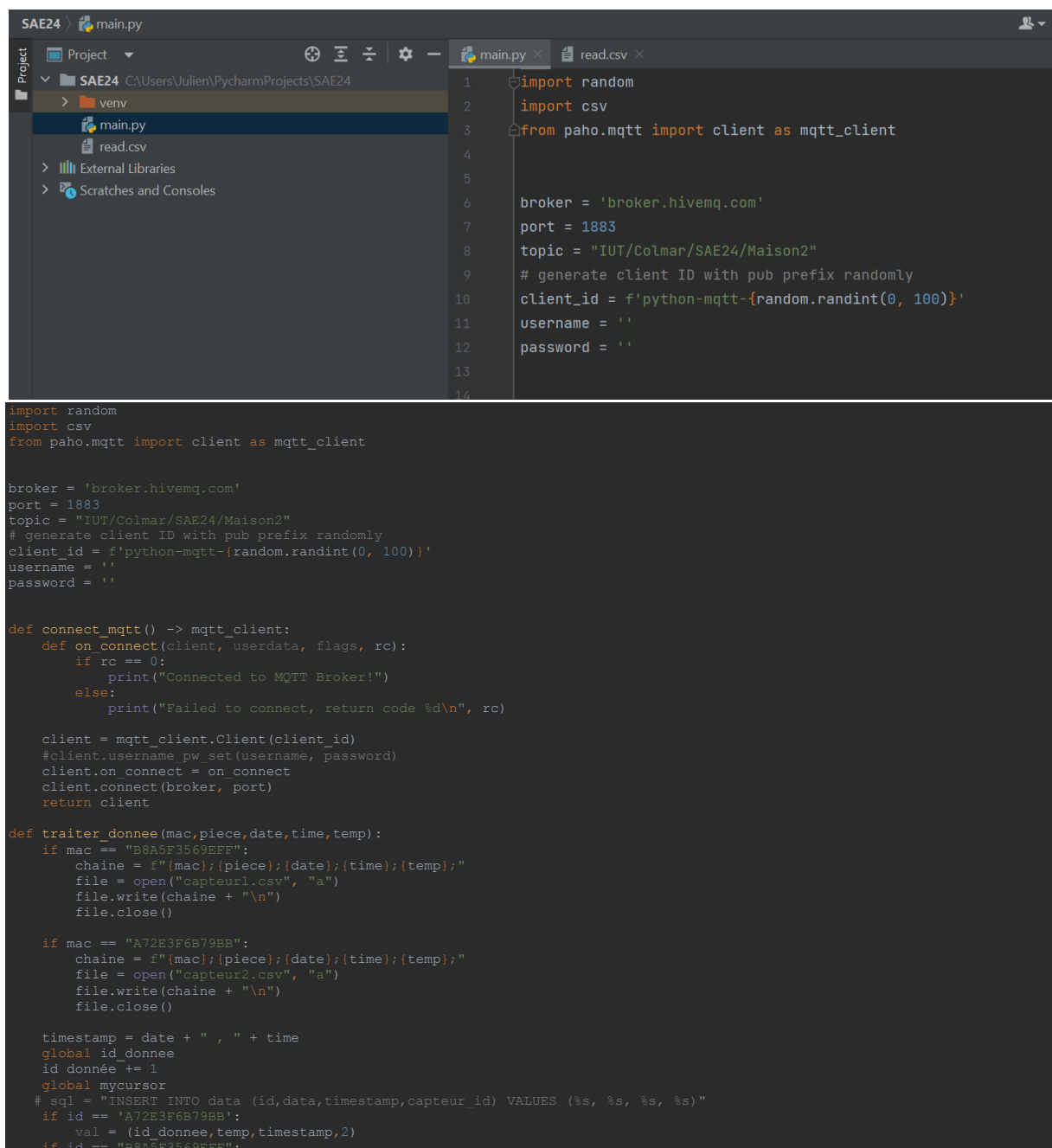
```
python3 -version
```

il faut installer dans le terminal la librairie « paho.mqtt » avec la commande

```
pip3 install paho-mqtt
```

Tout d'abord on identifie notre broker, puis le port 1883 et le topic.

Nous n'avons pas besoin d'identifier l'utilisateur et le mot de passe, car notre broker/topic est en public et nous n'avons pas besoin de s'identifier pour y accéder.



```
SAE24 > main.py
Project
  Project
    SAE24 C:\Users\Julien\PycharmProjects\SAE24
      > venv
        main.py
        read.csv
      > External Libraries
      > Scratches and Consoles

1 import random
2 import csv
3 from paho.mqtt import client as mqtt_client
4
5
6 broker = 'broker.hivemq.com'
7 port = 1883
8 topic = "IUT/Colmar/SAE24/Maison2"
9 # generate client ID with pub prefix randomly
10 client_id = f'python-mqtt-{random.randint(0, 100)}'
11 username = ''
12 password = ''
13
14
import random
import csv
from paho.mqtt import client as mqtt_client

broker = 'broker.hivemq.com'
port = 1883
topic = "IUT/Colmar/SAE24/Maison2"
# generate client ID with pub prefix randomly
client_id = f'python-mqtt-{random.randint(0, 100)}'
username = ''
password = ''

def connect_mqtt() -> mqtt_client:
    def on_connect(client, userdata, flags, rc):
        if rc == 0:
            print("Connected to MQTT Broker!")
        else:
            print("Failed to connect, return code %d\n", rc)

    client = mqtt_client.Client(client_id)
    #client.username_pw_set(username, password)
    client.on_connect = on_connect
    client.connect(broker, port)
    return client

def traiter_donnee(mac,piece,date,time,temp):
    if mac == "B8A5F3569EFF":
        chaine = f"{mac};{piece};{date};{time};{temp};"
        file = open("capteur1.csv", "a")
        file.write(chaine + "\n")
        file.close()

    if mac == "A72E3F6B79BB":
        chaine = f"{mac};{piece};{date};{time};{temp};"
        file = open("capteur2.csv", "a")
        file.write(chaine + "\n")
        file.close()

    timestamp = date + " , " + time
    global id_donnee
    id_donnee += 1
    global mycursor
    # sql = "INSERT INTO data (id,data,timestamp,capteur_id) VALUES (%s, %s, %s, %s)"
    if id == "A72E3F6B79BB":
        val = (id_donnee,temp,timestamp,2)
    if id == "B8A5F3569EFF":
```



```
val = (id_donnee, temp, timestamp,1)
# mycursor.execute(sql, val)

# mydb.commit()

def subscribe(client: mqtt_client):
    def on_message(client, userdata, msg):
        print(f"Received {msg.payload.decode()} from {msg.topic} topic")
        donne = msg.payload.decode()

        data = donne.split(',')
        print(data)
        mac = data[0][3:len(data[0])]
        piece = data[1][6:len(data[1])]
        date = data[2][5:len(data[2])]
        time = data[3][5:len(data[3])]
        temp = data[4][5:len(data[4])]

        datafinal = []
        datafinal.append(mac)
        datafinal.append(piece)
        datafinal.append(date)
        datafinal.append(time)
        datafinal.append(temp)

        with open('read.csv','a+', newline='') as f:
            write = csv.writer(f, delimiter=';', quotechar='"', quoting=csv.QUOTE_MINIMAL)
            write.writerow(datafinal)

    client.subscribe(topic)
    client.on_message = on_message

def run():
    client = connect_mqtt()
    subscribe(client)
    client.loop_forever()

if __name__ == '__main__':
    run()
```

#### a) Csv

On voit bien que le script python créer le fichier csv avec l'affichage des données demander.

Du type : **Id=12A6B8AF6CD3, piece=sejour,date=15/06/2022,heure=12:13:14,temp=26,35**

	Id	Piece	Date	Time	Temp
1	B8A5F3569EFF	sejour	23/06/2022	09:54:00	16.45
2	A72E3F6B79BB	chambre1	23/06/2022	09:54:00	1.95
3	B8A5F3569EFF	sejour	23/06/2022	09:54:05	6.83
4	A72E3F6B79BB	chambre1	23/06/2022	09:54:05	23.33
5	B8A5F3569EFF	sejour	23/06/2022	09:54:10	11.36
6	A72E3F6B79BB	chambre1	23/06/2022	09:54:10	23.91
7	B8A5F3569EFF	sejour	23/06/2022	09:54:15	15.01
8	A72E3F6B79BB	chambre1	23/06/2022	09:54:15	14.05
9	B8A5F3569EFF	sejour	23/06/2022	09:54:20	15.25
10	A72E3F6B79BB	chambre1	23/06/2022	09:54:20	5.6
11	B8A5F3569EFF	sejour	23/06/2022	09:54:25	15.25
12	A72E3F6B79BB	chambre1	23/06/2022	09:54:25	20.2
13	B8A5F3569EFF	sejour	23/06/2022	09:54:30	17.08
14	A72E3F6B79BB	chambre1	23/06/2022	09:54:30	3.94
15	B8A5F3569EFF	sejour	23/06/2022	09:54:35	25.05
16	A72E3F6B79BB	chambre1	23/06/2022	09:54:35	13.09
17	B8A5F3569EFF	sejour	23/06/2022	09:54:40	17.56