This document present the construction of Python script doing the treatment of the data from the MQTT server.

I - Accessing the MQTT

We need to access the server, for this purpose we will use the mqqt library of python called paho-mqtt, we install it using pip in a terminal,

```
sudo pip3 install paho-mqtt
```

To use the library in the script we use :

```
import paho.mqtt.client as mqtt
```

Then we have to define two functions, the one called by the library when it’s connected to the server. In this case it just print the result of the connexion.

```
def on_connect(client, userdata, flags, rc):
    print("Connected with result code "+str(rc))
```

The second one is called each time a message is received.

```
def on_message(client, userdata, msg):
    print(msg.topic+" "+info)
```

To be able to connect to the server we use :

```
1. client = mqtt.Client(client_id="Pylight")
2. client.on_connect = on_connect
3. client.on_message = on_message
4. client.username_pw_set("system", password="modif")
5. client.connect("localhost", 1883, 60)
```

Line 1 create an object Client from the library, we can specify an id that work like a MAC adress, it represent the script to the server. Line 2 and 3 are necessary for the library to work. Line 4 specify the username of the client, here it is “system” and the password, here “modif”. This line must be before client.connect() in all cases. Line 5 is the order of connection of the client, in first parameter is the IP of the broker, the second one is the port to use, by default 1883. The third argument is the timeout parameter in second, it define the period between each ping of the client to the server.

We are now connected to the server, to access information we must subscribe to a subject with :

```
client.subscribe("value/light")
```

We now get the information published in the subject “value/light” with the current configuration we will only print the message in the console.

To send information we use :

```
client.publish("light/order",a)
```

We will send the value of ‘a’ in the subject “light/order” on the server.
To keep the connection between the client and the server alive we need to call a loop, the library offer two different loop that we can use. The first one is :

```
1. client.loop_forever()
```

The only problem is that it lock the main tread of the program in the loop, rending any other operation impossible to execute, any instructions beyond this line will never be reached. Hopefully the library as as second possibility :

```
1. client.loop_start()
2. client.loop_stop()
```

These instruction start the loop as a thread, letting the main thread free to continue to execute your script.

II – Watching a file

We decided to stock our reference value to issue order in a txt file. This way the file can be modified each time it’s needed without modifying the script, at the staring up we read the file and stock the value into the live memory. Then we decided to make a web interface to modify the value, so the question arose how to monitor the change in the file. We will use “inotify_simple”, it is easy to put in place and do exactly what we need to. Howewer it only work on linux system.
First we need to install it using pip in command line :

```
1. sudo pip3 inotify_simple
```

Then we import it in the script :

```
1. from inotify_simple import INotify, flags
```

We configure what we want to see, by instantiating an object from the library (line 1). The we specify witch event we want to observe (line 2). In the end we specify the path to the directory or file you want to monitor (line 3).

```
1. inotify = INotify()
2. watch_flags = flags.CREATE | flags.DELETE | flags.MODIFY | flags.DELETE_SELF
3. wd = inotify.add_watch('/home/user/SSDinvite/PythonScript/ref.txt', watch_flags)
```

To see the events we just add the following lines.

```
1. while(1):
2.     for event in inotify.read():
3.         print(event)
4.         for flag in flags.from_mask(event.mask):
5.             print('    ' + str(flag))
```

Obviously line 1 is used to observe in continue the file, line 2 and 3 print the events registered while line 4 and 5 print the flag that triggered the event.

We can now use this to implement reaction in our script for any modification happening on the watched folder or file.