Urban Kissan : Python MQTT

Please Read below before making further changes to code

Latest Note : Please Follow MQTTLogger_RECM.py format for future code changes. The layered structure is important

- ▼ Sever Details
  - Host : 103.248.60.212
  - Username : UKIot
  - Pswd: log!c2021
  - Port 8288

Instructions to get access to root server

- ▼ Sever Details
  - Host : 103.248.60.212
  - Username : UKIot
  - Pswd: log!c2021
  - Port 8288
- ▼ SQL DB details
  - server = '185.136.157.12'
  - user = 'mqttuser'
  - paswd = '@5TdXk-XzN?PRd$M'
  - database = 'mqtt'

sudo su

UKIot

log!c2021

log!c2021

cd /home/Amos/UK_MQTT_Json_Rep

source MQTT-PythonVirtEnv/bin/activate

python MQTTLogger.py

Open another terminal and

python basicFlaskHTML.py

Github details :

UserName : YellowMinato

Token Authenticaltion : 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

Open web broswer and http://103.248.60.212:5000/summary

You should see something like this

PH	EC	Water Temp
5.019305	1.087247	21.875
5.007722	1.093718	21.875
5.111969	1.093718	21.875
5.100386	1.093718	21.875
5.158301	1.093718	21.875
5.227799	1.093718	21.875
5.250965	1.093718	21.875
5.297297	1.093718	21.875
5.250965	1.093718	21.875
5.158301	1.080775	21.875

Command to check if data is even coming or not

mosquitto_sub -h 103.248.60.212 -p 1883 -u UKIot -P 'log!c2021' -t 117/ukndu/deviceTosys/809146F23A08

Issue while connecting to FileZilla

First connect with port 22

Then change the details to 8288

Current code Repository

MQTT Logger Link : https://github.com/YellowMinato/UK_MQTT_Logger/blob/main/MQTTLogger.py

Loading data FROM Json file

```
# Python program to read
# json file


import json

# Opening JSON file
f = open('data.json',)

# returns JSON object as
# a dictionary
data = json.load(f)

# Iterating through the json
# list
for i in data['emp_details']:
  print(i)

# Closing file
f.close()
```

Link for reference : https://www.geeksforgeeks.org/read-json-file-using-python/

Dumping data into a json file

▼ Code

```
# Python program to write JSON
# to a file


import json
def storeDictInJsonFile(UK_JsonObj,filename):
    # example for filename : rec_m_data.json
    jsonMessage=json.dumps(UK_JsonObj)
    with open(filename, "a+") as outfile:
            print('Writing to ',filename)
            outfile.write(jsonMessage+'\n')
# Data to be written
dictionary ={
    "name" : "sathiyajith",
    "rollno" : 56,
    "cgpa" : 8.6,
    "phonenumber" : "9976770500"
}

storeDictInJsonFile(dictionary ,'store.json')
```

Simple Subsribe for MQTT

▼ Code

```
#Hello world
import paho.mqtt.client as mqtt #import the client1
import time
############
def on_message(client, userdata, message):
    print("message received " ,str(message.payload.decode("utf-8")))
    print("message topic=",message.topic)
    print("message qos=",message.qos)
    print("message retain flag=",message.retain)
########################################
broker_address="103.248.60.212"
#broker_address="iot.eclipse.org"
print("creating new instance")
client = mqtt.Client("P1") #create new instance
client.username_pw_set(username="UKIot",password="log!c2021")
client.on_message=on_message #attach function to callback
print("connecting to broker")
client.connect(broker_address, port=1883) #connect to broker
print("Subscribing to topic","house/bulbs/bulb1")
client.subscribe("117/ukndu/deviceTosys/809146F23A08")
while(True):
    client.loop()
```

Logging JSON data from MQTT into SQL Database

▼ Code

```
#!/usr/bin/python -u

import mysql.connector as mariadb
import paho.mqtt.client as mqtt
import ssl

mariadb_connection = mariadb.connect(user='USER', password='PW', database='MYDB')
cursor = mariadb_connection.cursor()

# MQTT Settings
MQTT_Broker = "192.XXX.XXX.XXX"
MQTT_Port = 8883
Keep_Alive_Interval = 60
MQTT_Topic = "/weather/pywws/#"

# Subscribe
def on_connect(client, userdata, flags, rc):
  mqttc.subscribe(MQTT_Topic, 0)

def on_message(mosq, obj, msg):
```

```
  # Prepare Data, separate columns and values
  msg_clear = msg.payload.translate(None, '{}""').split(", ")
  msg_dict =    {}
  for i in range(0, len(msg_clear)):
    msg_dict[msg_clear[i].split(": ")[0]] = msg_clear[i].split(": ")[1]

  # Prepare dynamic sql-statement
  placeholders = ', '.join(['%s'] * len(msg_dict))
  columns = ', '.join(msg_dict.keys())
  sql = "INSERT INTO pws ( %s ) VALUES ( %s )" % (columns, placeholders)

  # Save Data into DB Table
  try:
      cursor.execute(sql, msg_dict.values())
  except mariadb.Error as error:
      print("Error: {}".format(error))
  mariadb_connection.commit()

def on_subscribe(mosq, obj, mid, granted_qos):
  pass

mqttc = mqtt.Client()

# Assign event callbacks
mqttc.on_message = on_message
mqttc.on_connect = on_connect
mqttc.on_subscribe = on_subscribe

# Connect
mqttc.tls_set(ca_certs="ca.crt", tls_version=ssl.PROTOCOL_TLSv1_2)
mqttc.connect(MQTT_Broker, int(MQTT_Port), int(Keep_Alive_Interval))

# Continue the network loop & close db-connection
mqttc.loop_forever()
mariadb_connection.close()
```

Flask for loading json File - INCLUDING webpage template

▼ Code

Python code

```
from flask import json
from flask import Flask
from flask import render_template
app = Flask(__name__)
@app.route('/summary')
def summary():
    contents = open('sample.json', "r").read()
    data = [json.loads(str(item)) for item in contents.strip().split('\n')]
#    print(data[:20])
    #response = app.response_class(
     #   response=json.dumps(data),
     #   mimetype='application/json'
    #)
    return render_template('recordsJson.html',len=len(data) , records=data)
#    return response
if __name__ == '__main__':
        app.run(host='0.0.0.0',port = 5020)
```

Inside the same folder create a folder 'templates' (Very important)

Inside the templates folder , create recordsJson.html

```
<html>
<head>
    <title>Urban Kisaan</title>
</head>
<body>

<ol>
<!-- JSON Data -->
{%for i in range(0, len)%}
```

```
    <li>{{records[i]}}</li>
{%endfor%}

</ol>

</body>
</html>
```

For further reference : https://www.geeksforgeeks.org/python-using-for-loop-in-flask/

Adding new fields to dictionary

▼ Code

```
def addSerialId(jsonDict,topicStr):
    serialId = topicStr.split('/')[3]
    jsonDict['serialId']=serialId
    return jsonDict
```

MQTT Installation on Server

https://www.vultr.com/docs/how-to-install-mosquitto-mqtt-broker-server-on-ubuntu-16-04

apt-get update

apt-get install mosquitto

sudo apt-get install mosquitto-clients

Virtual Environments

▼ What libraries need to get installed , what are the commands invovled ?

Link for more details : https://www.geeksforgeeks.org/python-virtual-environment/

```
pip3 install virtualenv
```

1)installing required libraries first

```
virtualenv my_name
```

2)Creating local environment, it doesent nessescrily have to be "my_name" it can be anything else

```
sudo apt-get install python3-venv
```

3)You may also need to run the above command to make sure that proper libraries are installed

```
virtualenv -p my_name/bin/python3 virtualenv_name
```

4)creating virtual env using python3 ,since Im using python3 in this particular case

```
source virtualenv_name/bin/activate
```

This will create my preferred work environment.

▼ How do we enter an environment?

```
source virtualenv_name/bin/activate
```

▼ How do we exit an envoronment?/

```
deactivate
```

▼ How to create an environment?

```
virtualenv my_name
```

1)Creating local environment, it doesent nessescrily have to be "my_name" it can be anything else

```
virtualenv -p my_name/bin/python3 virtualenv_name
```

2)creating virtual env using python3 ,since Im using python3 in this particular case

Commands for executing the project

Commands for Testing the code

cd /home/Amos/UK_MQTT_Json_Rep/summaryFiles

wget http://103.248.60.212:5000/summary

Creating a daemon service

First create a service in systemd file

sudo vi /lib/systemd/system/MQTTLogger.service

Then add following details within

```
[Unit]
Description=MQTT Logger
After=multi-user.target
Conflicts=getty@tty1.service

[Service]
Type=simple
WorkingDirectory=/home/Amos/UK_MQTT_Json_Rep
ExecStart= /usr/bin/nohup /home/Amos/UK_MQTT_Json_Rep/MQTT-PythonVirtEnv/bin/python -u /home/Amos/UK_MQTT_Json_Rep/basicFlaskHTML.py > /
StandardInput=tty-force

[Install]
WantedBy=multi-user.target
```

Reload Deamon services

sudo systemctl daemon-reload

Enable and start Daemon service

```
sudo systemctl enable MQTTLogger.service
sudo systemctl start MQTTLogger.service
```

To check the status of service

sudo systemctl status MQTTLogger.service

To stop a service

sudo systemctl stop MQTTLogger.service

Possible Error 1 : Unable to access despite succesfully running daemon service

Sometimes even after running the service , we will not be able to get the webpage

```
● MQTTLogger.service - MQTT Logger
   Loaded: loaded (/lib/systemd/system/MQTTLogger.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2021-08-13 17:38:04 IST; 5s ago
 Main PID: 54682 (python3)
    Tasks: 1 (limit: 8601)
   CGroup: /system.slice/MQTTLogger.service
           └─54682 /home/Amos/UK_MQTT_Json_Rep/MQTT-PythonVirtEnv/bin/python3 /home/Amos/UK_MQTT_Json_Rep/basicFlaskHTML.py
```

It will show us the process is running properly , but the website will remain inaccessible. How can we fix this?

Answer : We need to mention the WorkingDirectory .If we dont mention that , then it will take by defaul "/lib/systemd/system/" as the directory of execution .Where you run the command from is important , otherwise it will check for files in the same

```
[Unit]
Description=MQTT Logger
After=multi-user.target
Conflicts=getty@tty1.service

[Service]
Type=simple
WorkingDirectory=/home/Amos/UK_MQTT_Json_Rep
ExecStart=/home/Amos/UK_MQTT_Json_Rep/MQTT-PythonVirtEnv/bin/python3 /home/Amos/UK_MQTT_Json_Rep/basicFlaskHTML.py
StandardInput=tty-force

[Install]
WantedBy=multi-user.target
```

All current files that are **NOT** included in the git repo

sudo vi /lib/systemd/system/MQTTLogger.service

sudo vi /lib/systemd/system/testDBInsert.service

sudo vi /lib/systemd/system/MQTTLogNotHTML.service [This is related to the actual logger file, we are not deploying as of yet , because its still in developement phase]

```
sudo vi /lib/systemd/system/testDBInsert.service
```

```
sudo systemctl stop testDBInsert.service
```

Running service for testInsertDB.py

```
sudo systemctl daemon-reload
sudo systemctl enable testDBInsert.service
sudo systemctl start testDBInsert.service
sudo systemctl status testDBInsert.service
```

Service for basicFlaskHTML_v2.py

```
sudo systemctl daemon-reload
sudo systemctl enable MQTTLogger.service
sudo systemctl start MQTTLogger.service
sudo systemctl status MQTTLogger.service
```

Service for MQTTLogger.py

sudo vi /lib/systemd/system/MQTTLogNotHTML.service

```
sudo systemctl stop MQTTLogNotHTML.service
sudo systemctl daemon-reload
sudo systemctl enable MQTTLogNotHTML.service
sudo systemctl start MQTTLogNotHTML.service
sudo systemctl status MQTTLogNotHTML.service
```

Service for MQTTLogger_RECM

sudo vi /lib/systemd/system/MQTTLogNotHTMLRECM.service

```
sudo systemctl stop MQTTLogNotHTMLRECM.service
sudo systemctl daemon-reload
sudo systemctl enable MQTTLogNotHTMLRECM.service
sudo systemctl start MQTTLogNotHTMLRECM.service
sudo systemctl status MQTTLogNotHTMLRECM.service
```

For editing files and making changes , please execute the fllowing commands .It is key to kill those process before deploying newly made chages

```
sudo systemctl stop MQTTLogger.service
sudo systemctl stop testDBInsert.service
```

For running nohup process

nohup python -u basicFlaskHTML.py > output.log &

nohup /home/Amos/UK_MQTT_Json_Rep/MQTT-PythonVirtEnv/bin/python -u /home/Amos/UK_MQTT_Json_Rep/basicFlaskHTML.py > /home/Amos/UK_MQTT_Json_Rep/output.log &

Device ID

Transactin Date

Form ID and

Json data

Unable to install pyodbc

▼ We are getting a gcc error while trying to install pyodbc ,How can we fix this error?

The error can be fixed by installing an extra sudo apt **SPECIFICALLY on linux systems**

```
sudo apt-get install unixodbc-dev
```

You can now rerun pip install pyodbc ,It should work .

Alternatively if this doesent work we might have to reinstall python and pip .

Unable to find drivers for

```
es/sqlalchemy/engine/default.py", line 584, in connect
    return self.dbapi.connect(*cargs, **cparams)
sqlalchemy.exc.InterfaceError: (pyodbc.InterfaceError) ('IM002', '[IM002] [unixO
DBC][Driver Manager]Data source name not found, and no default driver specified
(0) (SQLDriverConnect)')
(Background on this error at: https://sqlalche.me/e/14/rvf5)
```

In order fix this issue , we directly need to install pymssql . But not **ANY** version .There is a specific version that we need to install

```
pip install sqlalchemy==1.4.12
```

Reference(Optional)

▼ Expand

> Very late, but experienced the same such problem myself recently. Turned out it was a problem with the latest SQLAlchemy version. Had to rollback my version from 1.4.17 to 1.4.12 (unsure of in-between versions, just went with a version I knew worked).
>
> ```
> pip install sqlalchemy==1.4.12
> ```

Link : https://stackoverflow.com/questions/41344054/python-sqlalchemy-data-source-name-not-found-and-no-default-driver-specified

Python code to test connection to mssql Server

```
import sqlalchemy as sa
engine = sa.create_engine('mssql+pymssql://mqttuser:@5TdXk-XzN?PRd$M@185.136.157.12/mqtt')
conn = engine.connect()
#Host : 185.136.157.12
#User : mqttuser
#Password : @5TdXk-XzN?PRd$M
#Database : mqtt
```

Column details

sno - int

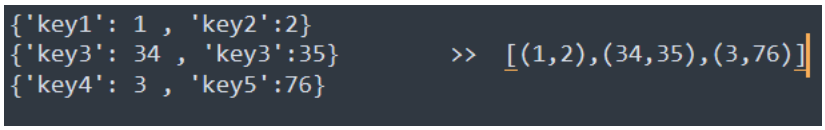
Transdate - datetime

deviceid - varchar

farmid - int

jsondata - text

Python code to sperate json strings in file with specific delimitter



Python code to convert list of dictionary to tuple

```
import json
from datetime import datetime
contents = open('deviceData.json', "r").read()
data = [json.loads(str(item)) for item in contents.strip().split('\n')]
insertionQueryArr = [(datetime.strptime(rec['Time'],'%m/%d/%Y, %H:%M:%S'),rec['serialId'],rec['storeId'],json.dumps(rec)) for rec in data]
#Note: contents consists of dictionaries seperated by '\n'
```

Python code to convert string date time to date time object

```
from datetime import datetime
(datetime.strptime(rec['Time'],'%m/%d/%Y, %H:%M:%S')
```

Python code to Empty file

```
open('deviceData.json', 'w').close()
```

[Github]Moving forward: Password authentication has been revoked

Password Authentication has been revoked since august 13 .

In order to work around this we need to create authorization tokens

For complete link on how to setup your own Tokens , visit https://docs.github.com/en/github/authenticating-to-github/keeping-your-account-and-data-secure/creating-a-personal-access-token

How to deal with following error?

If you are facing this error , then that most probably means that we didnt set permissions properly .In order to do that go to Setting > Developer settings

Under personal access tokens .Select generate new personal tokens .Save the token somehwhere.

Repeating Task Python Class

Basic class for task repeaters

```
from threading import Timer, Lock


class Periodic(object):
    """
    A periodic task running in threading.Timers
    """

    def __init__(self, interval, function, *args, **kwargs):
        self._lock = Lock()
        self._timer = None
        self.function = function
        self.interval = interval
        self.args = args
        self.kwargs = kwargs
        self._stopped = True
        if kwargs.pop('autostart', True):
            self.start()

    def start(self, from_run=False):
        self._lock.acquire()
        if from_run or self._stopped:
            self._stopped = False
            self._timer = Timer(self.interval, self._run)
            self._timer.start()
            self._lock.release()

    def _run(self):
        self.start(from_run=True)
        self.function(*self.args, **self.kwargs)

    def stop(self):
        self._lock.acquire()
        self._stopped = True
        self._timer.cancel()
        self._lock.release()
```

Implementation of python class

```
from time import sleep

def hello(name):
    print "Hello %s!" % name

print "starting..."
rt = Periodic(1, hello, "World") # it auto-starts, no need of rt.start()
try:
    sleep(5) # your long-running job goes here...
finally:
    rt.stop()
```

More interesting answers can be found here : https://stackoverflow.com/questions/2398661/schedule-a-repeating-event-in-python-3

Checking if file is empty

```
>>> import os
>>> os.stat("file").st_size == 0
```

MSSQL Query

mssql checking top 10 entries based on serial id

```
select top 10 * from datareadings order by sno desc;
```

Scalablilty with Energy Meter

Originally we only had one topic in mind . ukndu , but now we are adding(scaling up) it with a different topic .

[**Updated**] For running nohup with logs

First kill all processes related to these three using kill -9

then

rm outputInsertDB.log

rm outputMQTTLogger.log

outputbasicFLaskHTML.log

python -u testInsertDB.py > logs/outputInsertDB.log &

python -u MQTTLogger.py > logs/outputMQTTLogger.log &

python -u basicFlaskHTML.py > logs/outputbasicFLaskHTML.log &

REC-M unit for Urban Kissan Latest webpage developement

Checking the data using mosquitto command

```
mosquitto_sub -h 103.248.60.212 -p 1883 -u UKIot -P 'log!c2021' -t 430/ukrecm/deviceTosys/B8BEF0ACCD98
```

MQTTLogger_RECM

```
#Hello world
import paho.mqtt.client as mqtt #import the client1
import time
import json
from datetime import datetime
import re
############
def addStoreId(jsonDict,topicStr):
    storeId = topicStr.split('/')[0]
    jsonDict['storeId']=storeId
    return jsonDict
def addSerialId(jsonDict,topicStr):
    serialId = topicStr.split('/')[3]
    jsonDict['serialId']=serialId
    return jsonDict
def addDateTime(jsonDict):
#Note : Dont foget to import datetime or else it wont raise any error in try catch
    now = datetime.now()
    jsonDict['Time']=now.strftime("%m/%d/%Y, %H:%M:%S")
    print("date and time:",jsonDict['Time'])
    return jsonDict
def addType(jsonDict,topicStr):
    jsonDict['type']=topicStr.split('/')[1]
    return jsonDict
def checkRegex(topicStr):
    regExp = re.search('^[0-9]+/[a-zA-Z0-9+_.-]+/[a-zA-Z0-9+_.-]+/[a-zA-Z0-9+_.-]+$',topicStr)
    if(regExp):
        print('Valid Topic')
        return True
    return False
def convert_ld_to_dict(jsonObject):
    UK_JsonObj ={}
    for param in jsonObject:
```

```
       for key in param.keys():
            UK_JsonObj[key]=param[key]
    return UK_JsonObj
def storeDictInJsonFile(UK_JsonObj,filename):
    # example for filename : rec_m_data.json
    jsonMessage=json.dumps(UK_JsonObj)
    with open(filename, "a+") as outfile:
            print('Writing to ',filename)
            outfile.write(jsonMessage+'\n')
def on_message(client, userdata, message):
    print("message received " ,str(message.payload.decode("utf-8")))
    print("message topic=",message.topic)
    print("message qos=",message.qos)
    print("message retain flag=",message.retain)
    jsonMessage = str(message.payload.decode("utf-8"))
    topicStr = message.topic
    try:
        if(checkRegex(topicStr)):
            jsonObject = json.loads(str(message.payload.decode("utf-8")))
            dataType = topicStr.split('/')[1]
            if(dataType=='ukrecm'):
                UK_JsonObj = convert_ld_to_dict(jsonObject)
                storeDictInJsonFile(UK_JsonObj,'ukrecm_data.json')
            # elif(dataType==something else ):
                # UK_JsonObj = convert_ld_to_dict(jsonObject)
                # storeDictInJsonFile(UK_JsonObj,'that_particular_data.json')
    except:
        print("Invalid Json")

########################################
broker_address="103.248.60.212"
#broker_address="iot.eclipse.org"
print("creating new instance")
client = mqtt.Client("P1") #create new instance
client.username_pw_set(username="UKIot",password="log!c2021")
client.on_message=on_message #attach function to callback
print("connecting to broker")
client.connect(broker_address, port=1883) #connect to broker
print("Subscribing to topic","house/bulbs/bulb1")
client.subscribe("+/+/deviceTosys/+")
while(True):
    client.loop()
```

What files should you focus on ?

The latest HTML addition

RECM_data.html

Latest new JSON file

ukrecm_data.json

Changes being made on the new python file

basicFlaskHTML_v2.py

Running webpage file

nohup python -u basicFlaskHTML_v2.py>outputRecm.log &

Running MQTT logger

nohup python -u MQTTLogger_RECM.py > RECM_Logger.log &

output Logs for Webpage

tail -f outputRecm.log

output logs for MQTT Logger RECM

tail -f RECM_Logger.log

For checking if all services are active or not

```
ps -ef|grep MQTTLogger
ps -ef|grep basicFlaskHTML_v2.py
ps -ef|grep testInsertDB.py
```

For checking if data is coming over MQTT for RECM

```
mosquitto_sub -h 103.248.60.212 -p 1883 -u UKIot -P 'log!c2021' -t +/ukrecm/deviceTosys/#
```

For checking if data is coming across all channels

```
mosquitto_sub -h 103.248.60.212 -p 1883 -u UKIot -P 'log!c2021' -t +/+/deviceTosys/#
```

For Sending specific data to webpage

```
mosquitto_pub -h 103.248.60.212 -p 1883 -u UKIot -P 'log!c2021' -t sunnySpecific -m 0xc9
```

Listening to the data

```
mosquitto_pub -h 103.248.60.212 -p 1883 -u UKIot -P 'log!c2021' -t sunnySpecific
```

MQTT Logger with Dynamic parameter loading.

Location : D:\MQTT Logger

Version 3:

Location : /home/Amos/UK_MQTT_Json_Rep_v2

MQTT Logger with daemon service : Mqtt_logger_dynamic.service

To edit : sudo vi /lib/systemd/system/Mqtt_logger_dynamic.service

```
[Unit]
Description=MQTT Logger
```

```
After=multi-user.target
Conflicts=getty@tty1.service

[Service]
Type=simple
WorkingDirectory=/home/Amos/UK_MQTT_Json_Rep
ExecStart= /usr/bin/nohup /home/Amos/UK_MQTT_Json_Rep_v2/MQTT-Env/bin/python -u /home/Amos/UK_MQTT_Json_Rep_v2/basicFlaskHTML.py > /home/Am
StandardInput=tty-force

[Install]
WantedBy=multi-user.target
```

Execution Steps :

```
sudo systemctl stop Mqtt_logger_dynamic.service
sudo systemctl daemon-reload
sudo systemctl enable Mqtt_logger_dynamic.service
sudo systemctl start Mqtt_logger_dynamic.service
sudo systemctl status Mqtt_logger_dynamic.service
```

MQTT Logger with daemon service : Mqtt_Flask.service

To edit : sudo vi /lib/systemd/system/Mqtt_Flask.service

```
[Unit]
Description=MQTT Flask
After=multi-user.target
Conflicts=getty@tty1.service

[Service]
Type=simple
WorkingDirectory=/home/Amos/UK_MQTT_Json_Rep
ExecStart= /usr/bin/nohup /home/Amos/UK_MQTT_Json_Rep_v2/MQTT-Env/bin/python -u /home/Amos/UK_MQTT_Json_Rep_v2/basicFlaskHTML.py > /home/Am
StandardInput=tty-force

[Install]
WantedBy=multi-user.target
```

Execution Steps :

```
sudo systemctl stop Mqtt_Flask.service
sudo systemctl daemon-reload
sudo systemctl enable Mqtt_Flask.service
sudo systemctl start Mqtt_Flask.service
sudo systemctl status Mqtt_Flask.service
```

Latest Steps for Execution :

Instructions for running Python MQTT logger

```
/home/Amos/UK_MQTT_Json_Rep_v3/runScheduler.py
cd /home/Amos/UK_MQTT_Json_Rep_v3
source
python MQTT_Logger.py (or add a nohup to make it run)
```

Getting descending order of time date

```
=>Selecting entries from database with descending order of datetime
    SELECT TOP (1000) [Id]
      ,[FK_ProductID]
      ,[Description]
      ,[UpdatedDate]
      ,[isActive]
```

```
  ,[Price]
FROM [mqtt].[dbo].[sampleTable1] order by UpdatedDate desc;
```