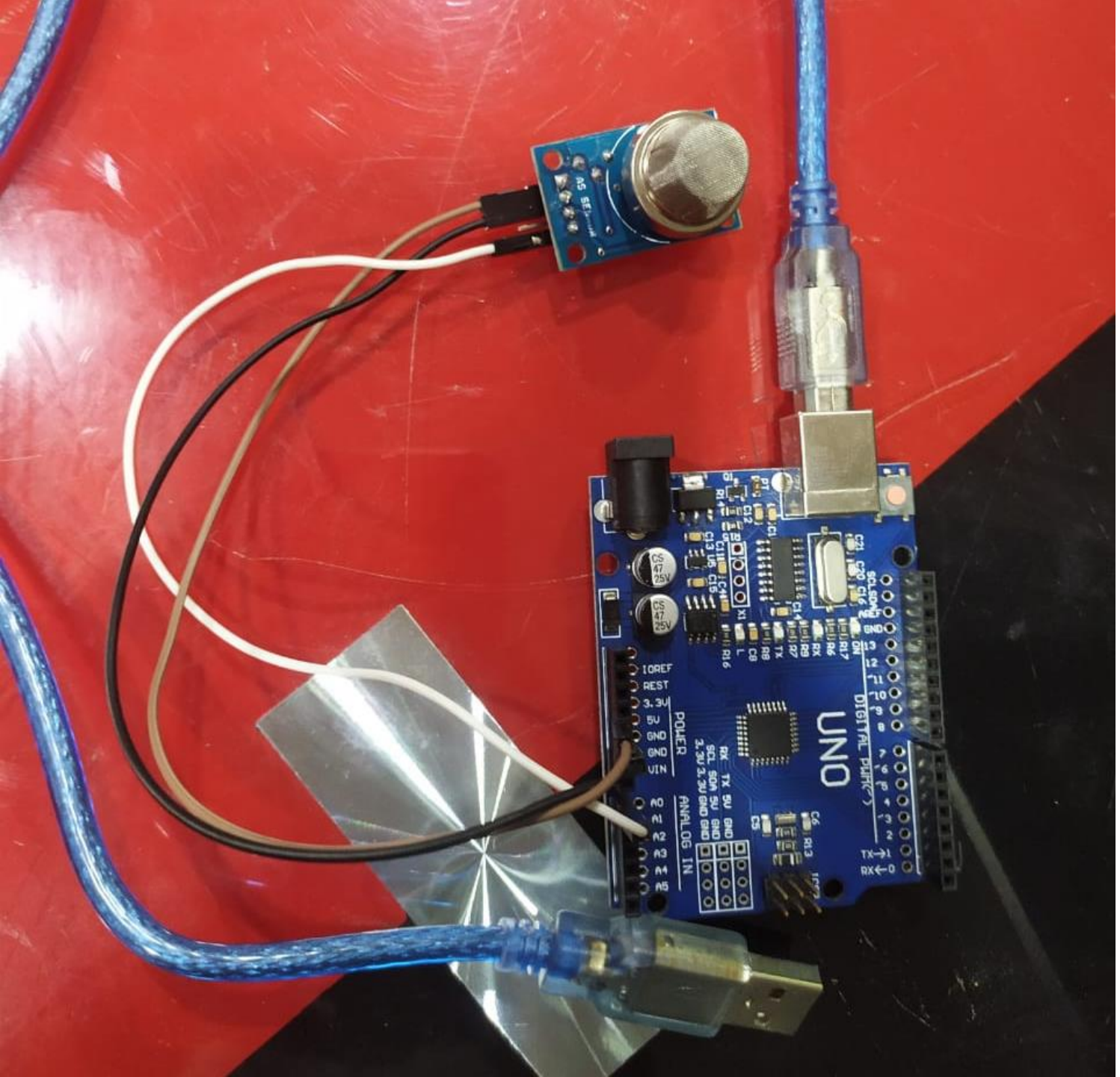
ARDUINO

INTRODUCTION

- **Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online**
- **INSTALLATION**
- **Download Arduino IDE Software.**
- **Power up your board.**
- **Launch Arduino IDE.**
- **Open your first project.**
- **To open an existing project example, select File → Example → Basics → Blink.**
- **Select your Arduino board.**
- **Go to Tools → Board and select your board.**
- **Select your serial port.**
- **Upload the program to your board.**
- **Reference link https://www.tutorialspoint.com/arduino/arduino_installation.htm**

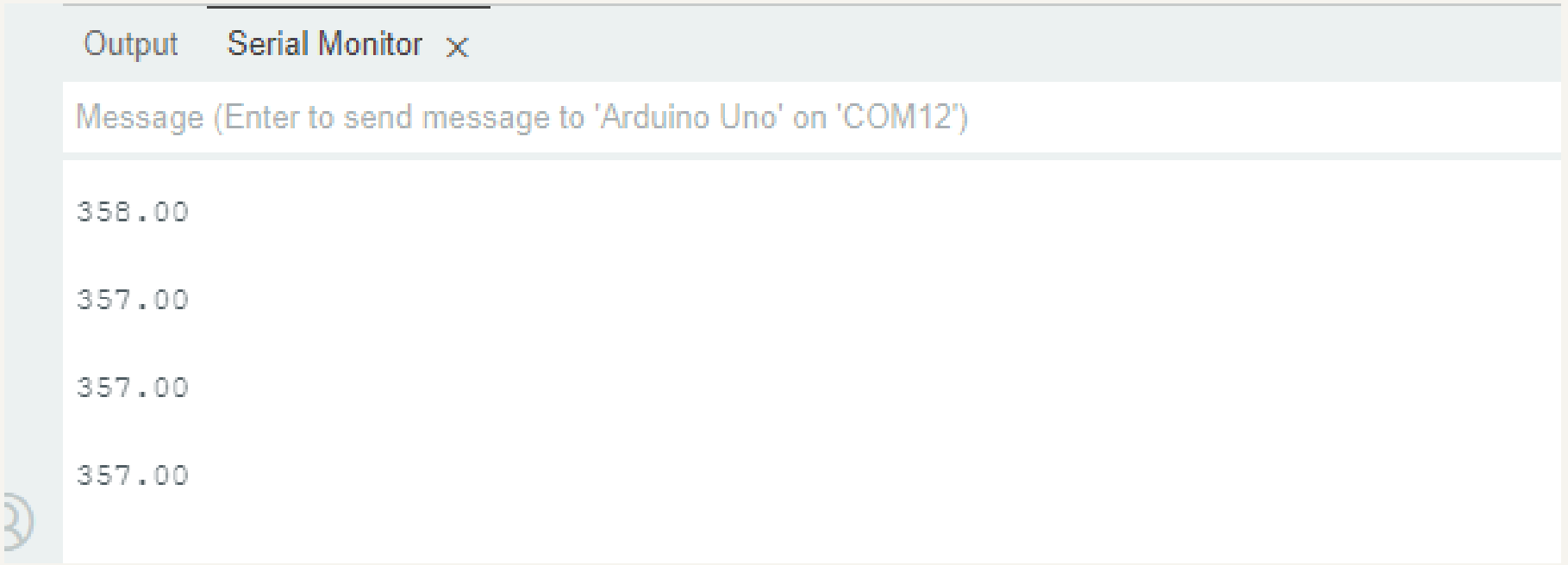


MQ2 sensor

Arduino program

```
#define MQ2pin (0)
float sensorValue;  //variable to store sensor value
String nk = "";
void setup()
{
  Serial.begin(9600); // sets the serial port to 9600
  //Serial.println("Gas sensor warming up!");
  delay(20000); // allow the MQ-2 to warm up
}
void loop()
{
sensorValue = analogRead(MQ2pin); // read analog input pin 0
```

```
 //Serial.print("Sensor  Value: ");
  //sensorValue  = String(sensorValue)
  //
  nk = nk + sensorValue;
  Serial.print(nk);
  nk = "";
 if(sensorValue > 130)
  {
    //Serial.print(" |Smoke detected!");
  }

  Serial.println("\n");
  delay(2000); // wait 2s for next reading
}
```

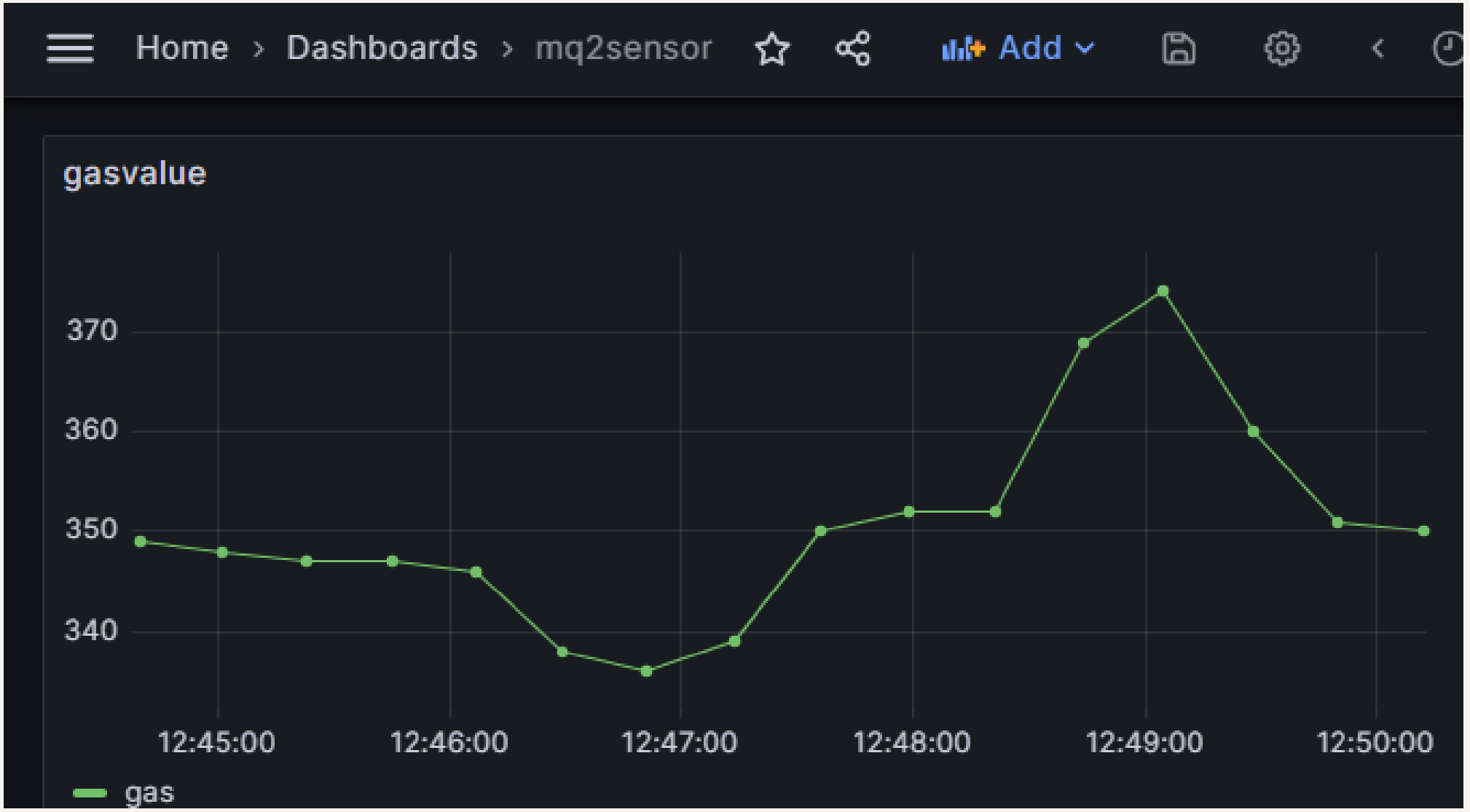

Output
Serial Monitor
Message (Enter to send message to 'Arduino Uno' on 'COM12')
358.00
357.00
357.00
357.00

PYTHON PROGRAM

```
import serial
import mysql.connector

def insert_data(mydata):
    mydb = mysql.connector.connect(
        host="localhost",
        user="root",
        password="",
        database="db"
        )
    my_arr = mydata.split(',')
    mycursor = mydb.cursor()
    sql='INSERT INTO gasdata(gas) VALUES ('+my_arr[0]+')'
    mycursor.execute(sql)
    mydb.commit()
    print("insertion success")
```

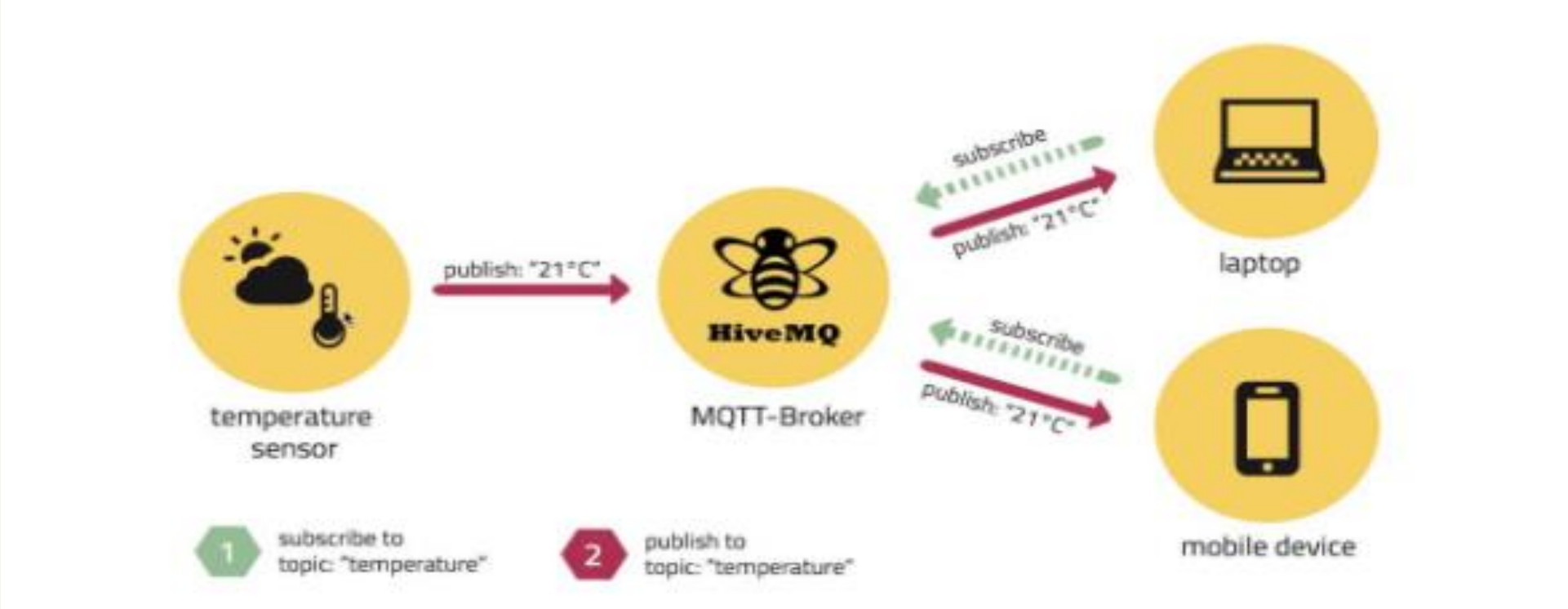
```
while True:

    myser = serial.Serial("COM10"  ,9600,
                          parity=serial.PARITY_NONE,
                          stopbits=serial.STOPBITS_ONE,
                          bytesize=serial.EIGHTBITS)

    line  = (myser.readline())

    data = line.decode('utf-8')
    print(data)



    insert_data(data)
    myser.close()
```

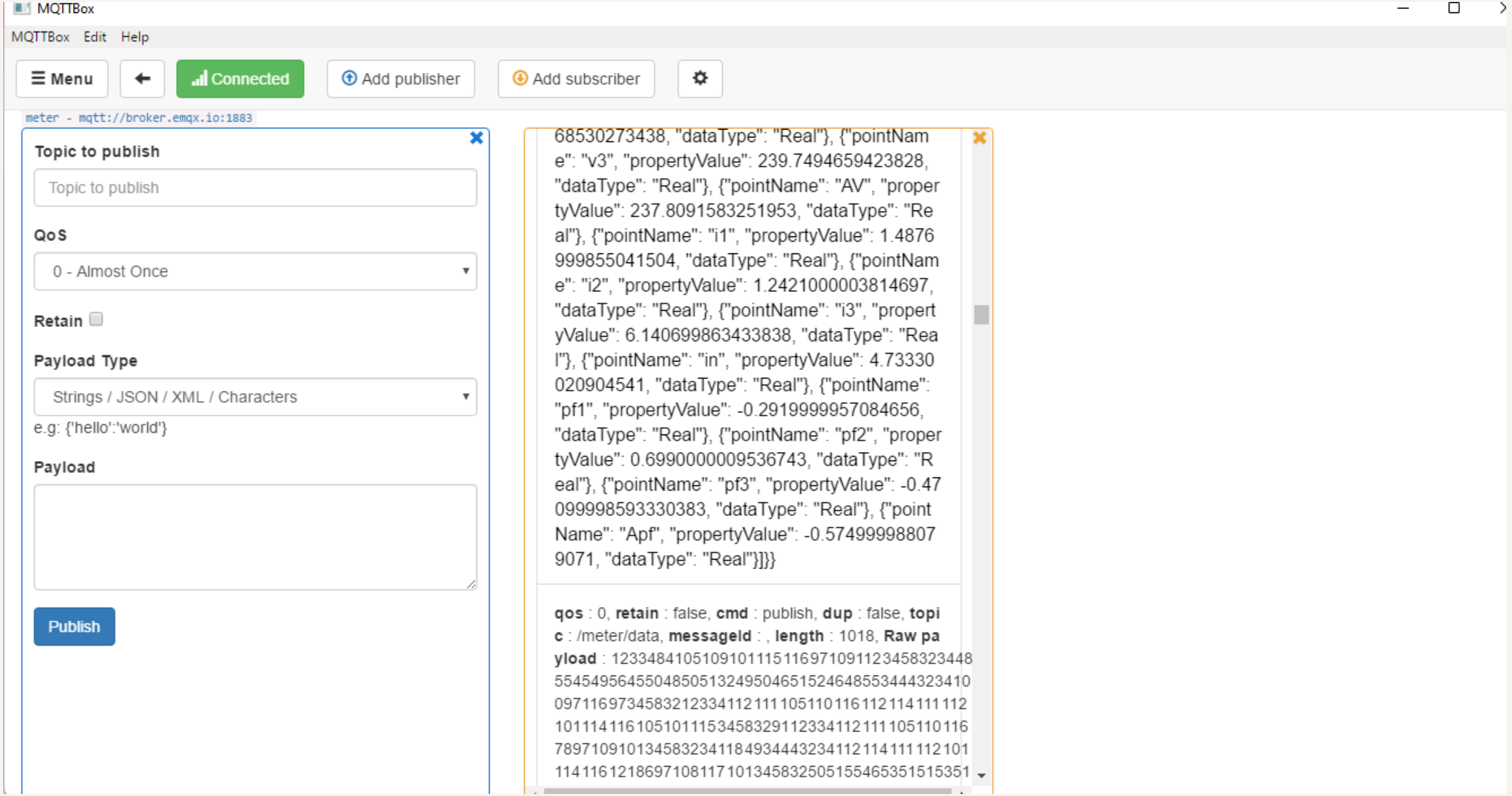
- **GRAFANA OUTPUT**



MQTT BOX

MQTTbox lets you publish messages to an MQTT broker, subscribe to MQTT topics,receive messages and do load testing.
Reference link http://www.steves-internet-guide.com/using-mqttbox/




MQTTBox
MQTTBox Edit Help
Menu
Connected
Add publisher
Add subscriber
meter - mqtt://broker.emqx.io:1883
Topic to publish
Topic to publish
QoS
0 - Almost Once
Retain
Payload Type
Strings / JSON / XML / Characters
e.g: {'hello':'world'}
Payload
Publish
68530273438, "dataType": "Real"}, {"pointNam e": "v3", "propertyValue": 239.7494659423828, "dataType": "Real"}, {"pointName": "AV", "proper tyValue": 237.8091583251953, "dataType": "Re al"}, {"pointName": "i1", "propertyValue": 1.4876 999855041504, "dataType": "Real"}, {"pointNam e": "i2", "propertyValue": 1.2421000003814697, "dataType": "Real"}, {"pointName": "i3", "propert yValue": 6.140699863433838, "dataType": "Rea l"}, {"pointName": "in", "propertyValue": 4.73330 020904541, "dataType": "Real"}, {"pointName": "pf1", "propertyValue": -0.2919999957084656, "dataType": "Real"}, {"pointName": "pf2", "proper tyValue": 0.6990000009536743, "dataType": "R eal"}, {"pointName": "pf3", "propertyValue": -0.47 099998593330383, "dataType": "Real"}, {"point Name": "Apf", "propertyValue": -0.57499998807 9071, "dataType": "Real"}]}}
qos : 0, retain : false, cmd : publish, dup : false, topi c : /meter/data, messageId : , length : 1018, Raw pa yload : 123348410510910111511697109112345832344855454956455048505132495046515246485534443234100971169734583212334112111051101161121141111121011141161051011153458329112334112111051101167897109101345832341184934443234112114111121011411612186971081171013458325051554653515153511

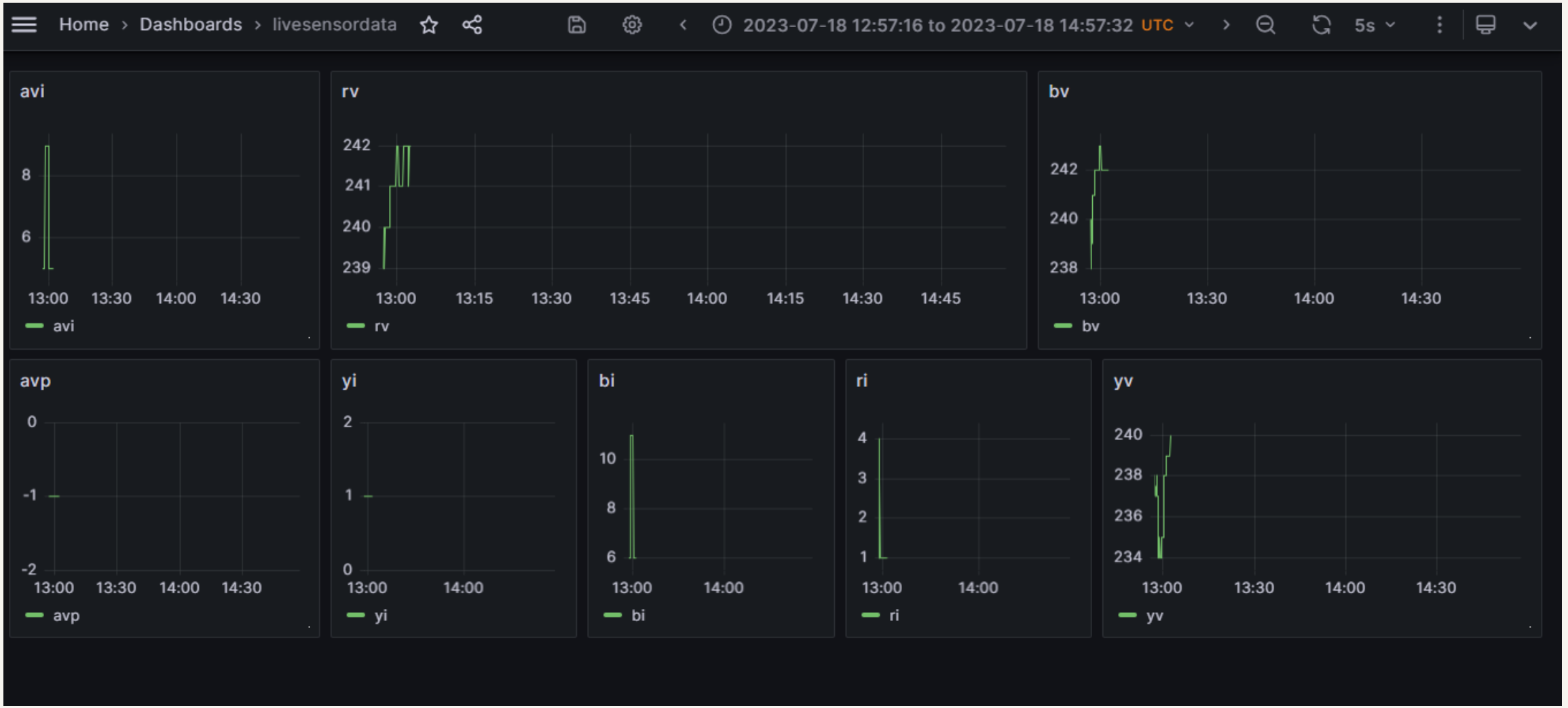
```
import mysql.connector
import paho.mqtt.client as mqttClient
from threading import Thread
import json


class Mqtt:
 def __init__(self):
      self.json_data = {}
      self.db = mysql.connector.connect(
         host="localhost",
         user="root",
         password="",
         db="sensordatas")
      mqttclient = mqttClient.Client("52244535475668454")
      mqttclient.on_connect = self.on_connect
      mqttclient.on_message = self.on_message
      #print(mqttclient.on_message)
      mqttclient.username_pw_set(username="",password="")
      mqttstatus = mqttclient.connect("broker.emqx.io", 1883,60)
      mqttclient.subscribe("/meter/data",2)
      mqttclient.loop_forever()
```

```
def upload(self,msg):
        mqtt_msg = str(msg.payload).replace("b'", "").replace("'", "").replace("  ", "").replace("\\n", "").replace("\n", '')
        print(msg.payload)
        mqtt_msg = str(mqtt_msg).replace("\\","")
        mqtt_msg = str(mqtt_msg).replace('}"','}')
        mqtt_msg = str(mqtt_msg).replace('"{','{')
        mqtt_msg = str(mqtt_msg).replace('{','')
        mqtt_msg = str(mqtt_msg).replace('}','')
        mqtt_msg = str(mqtt_msg).replace('"','')
        mqtt_msg = mqtt_msg.split(",")
        #print("====",mqtt_msg)
        #print("=----",mqtt_msg[2])
```

```
rv = mqtt_msg[2].split(":")[1]
        print("rv:"+rv+"end")
        #ya = mqtt_msg[4].split(":")[1]
        #ba = mqtt_msg[5].split(":")[1]
        #rv = mqtt_msg[1].split(":")[1]
        #print("rv:"+rv+"end")
        yv = mqtt_msg[5].split(":")[1]
        print("yv:"+yv+"end")
        bv = mqtt_msg[8].split(":")[1]
        print("bv:"+bv+"end")
        Av = mqtt_msg[11].split(":")[1]
        print("Av:"+Av+"end")
        ri = mqtt_msg[14].split(":")[1]
        print("ri:"+ri+"end")
        yi = mqtt_msg[17].split(":")[1]
        print("yi:"+yi+"end")
        bi = mqtt_msg[20].split(":")[1]
        print("bi:"+bi+"end")
        avi = mqtt_msg[23].split(":")[1]
        print("avi:"+avi+"end")
        rpf = mqtt_msg[26].split(":")[1]
        print("rpf:"+rpf+"end")
```

```
ypf = mqtt_msg[29].split(":")[1]
        print("ypf:"+ypf+"end")
        bpf = mqtt_msg[32].split(":")[1]
        print("bpf:"+bpf+"end")
        avp = mqtt_msg[35].split(":")[1]
        print("avp:"+avp+"end")
        energy = mqtt_msg[3].split(":")[1]

        mycursor = self.db.cursor()
        sql = 'INSERT INTO db (rv,yv,bv,Av,ri,yi,bi,avi,avp) VALUES
('+rv+','+yv+','+bv+','+Av+','+ri+','+yi+','+bi+','+avi+','+avp+')'
        mycursor.execute(sql)
        self.db.commit()
        #print(ra)
        print(rv)
        print(yv)
        print(bv)
        print("Data Inserted!")
```

```
• def on_connect(self,mqttclient, userdata, flags,rc):
        if rc == 0:
            print("connected!")
        else:
            print("Connection failed")


    def on_message(self, mqttclient, userdata, msg):
        Thread(target=self.upload, args=(msg,)).start()


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


Home › Dashboards › livesensordata
2023-07-18 12:57:16 to 2023-07-18 14:57:32 UTC
5s
avi
8
6
13:00 13:30 14:00 14:30
avi
rv
242
241
240
239
13:00 13:15 13:30 13:45 14:00 14:15 14:30 14:45
rv
bv
242
240
238
13:00 13:30 14:00 14:30
bv
avp
0
-1
-2
13:00 13:30 14:00 14:30
avp
yi
2
1
0
13:00 14:00
yi
bi
10
8
6
13:00 14:00
bi
ri
4
3
2
1
13:00 14:00
ri
yv
240
238
236
234
13:00 13:30 14:00 14:30
yv