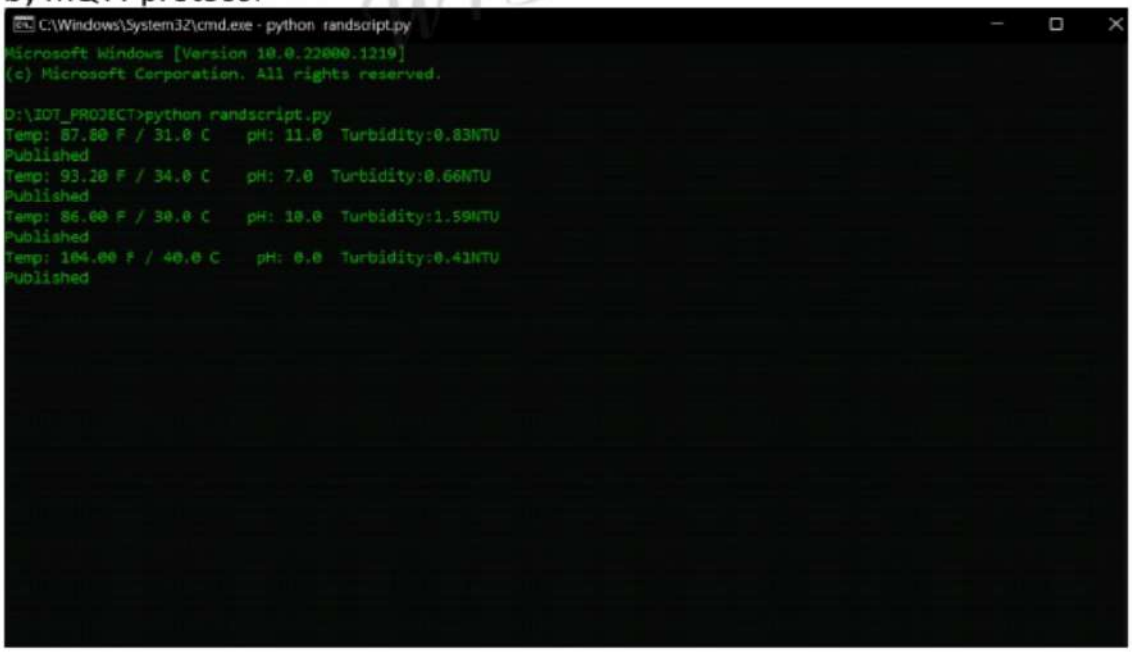
1.Development of python script

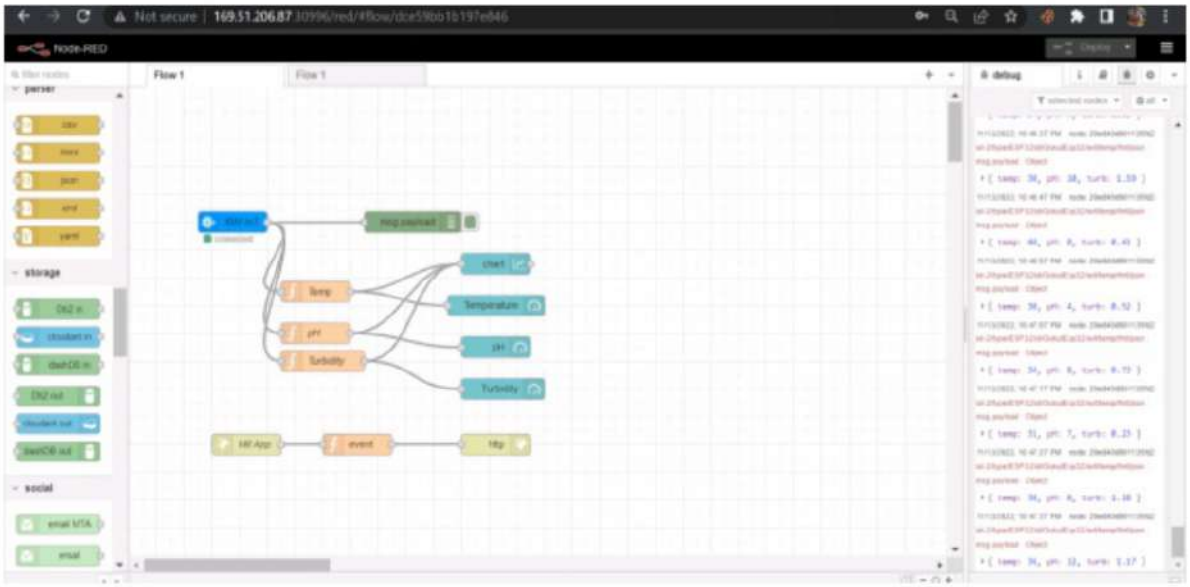
```
import paho.mqtt.client as mqtt
import time
import random
import json


def run():
    ORG ="q6sux6"
    DEVICE_TYPE ="ESP32"
    DEVICE_ID ="GokulEsp32"
    TOKEN ="gp5PA9!jfw7jf9cV-g"

    server = ORG + ".messaging.internetofthings.ibmcloud.com";
    pubTopic1 = "iot-2/evt/temp/fmt/json"
    pubTopic2 = "iot-2/evt/pH/fmt/json"
    pubTopic3 = "iot-2/evt/turb/fmt/json"
    #pubTopic3 = "iot-2/evt/wf/fmt/json"

    authMethod = "use-token-auth";
    token = TOKEN;
    clientId = "d:" + ORG + ":" + DEVICE_TYPE + ":" + DEVICE_ID;

    mqttc = mqtt.Client(client_id=clientId)
    mqttc.username_pw_set(authMethod, token)
    mqttc.connect(server, 1883, 60)

    while True:
        try:
            # Print the values to the serial port
            temperature_c = random.randint(30,40) * 1.0
            temperature_f = temperature_c * (9 / 5) + 32.0
            pH = random.randint(0,14)* 1.0
            turb=random.uniform(1,2)
            print(
```

```
                "Temp: {:.2f} F / {} C    pH: {}  Turbidity:{:.2f}NTU".format(
                    temperature_f, temperature_c, pH,turb
                )
            )
            payload={"temp":temperature_c,"pH":pH,"turb":round(turb,2)}

            mqttc.publish(pubTopic1,json.dumps(payload))
            #mqttc.publish(pubTopic2,pH)
            #mqttc.publish(pubTopic3,round(turb,2))

            print("Published")
            time.sleep(10)

        except RuntimeError as error:
            print(error.args[0])
            time.sleep(2.0)
        except Exception as error:
            print("Error encountered!")
            time.sleep(5.0)
    mqttc.loop_forever()
if __name__=='__main__':
    run()
```

2.Executing the developed python script to send value to IOT Watson platform by MQTT protocol



3.Sending the obtained values to Web UI dashboard and designed App

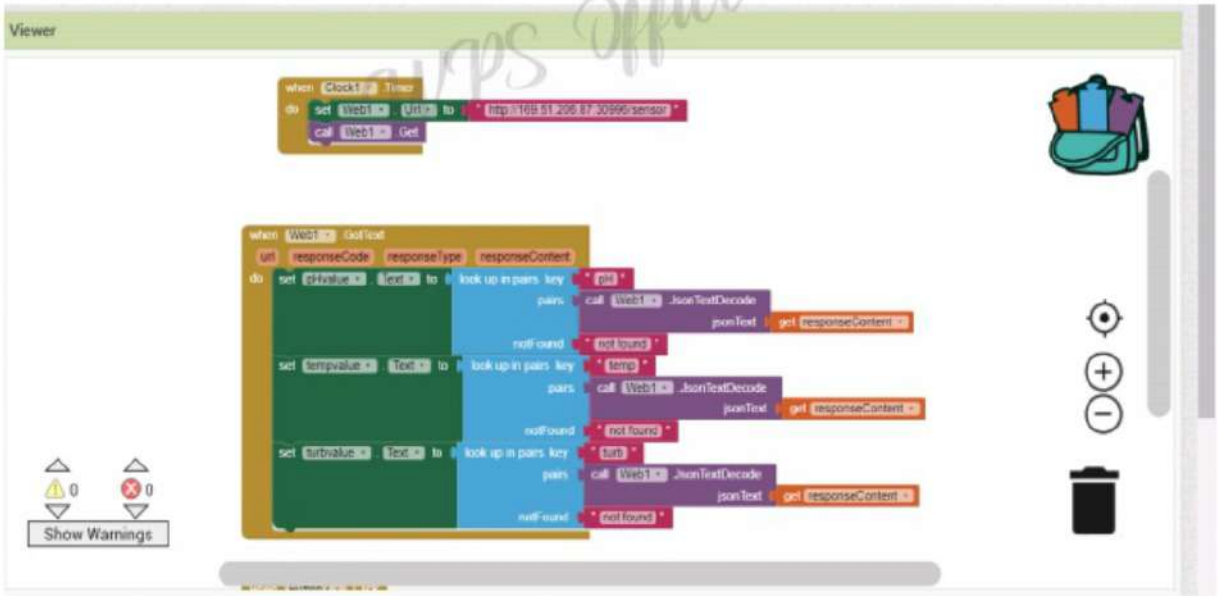


4.Payload defined to obtain all the parameters in mobile app

5.JSON object obtained in the specified URL

5.Mobile UI frontend to receive the data from Node-red

6.Configuring MIT mobile app backend to receive the data from Node-Red



7.Web UI dashboard

8.Checking in mobile app whether data correctly received or not(**Waterflow is not added)**