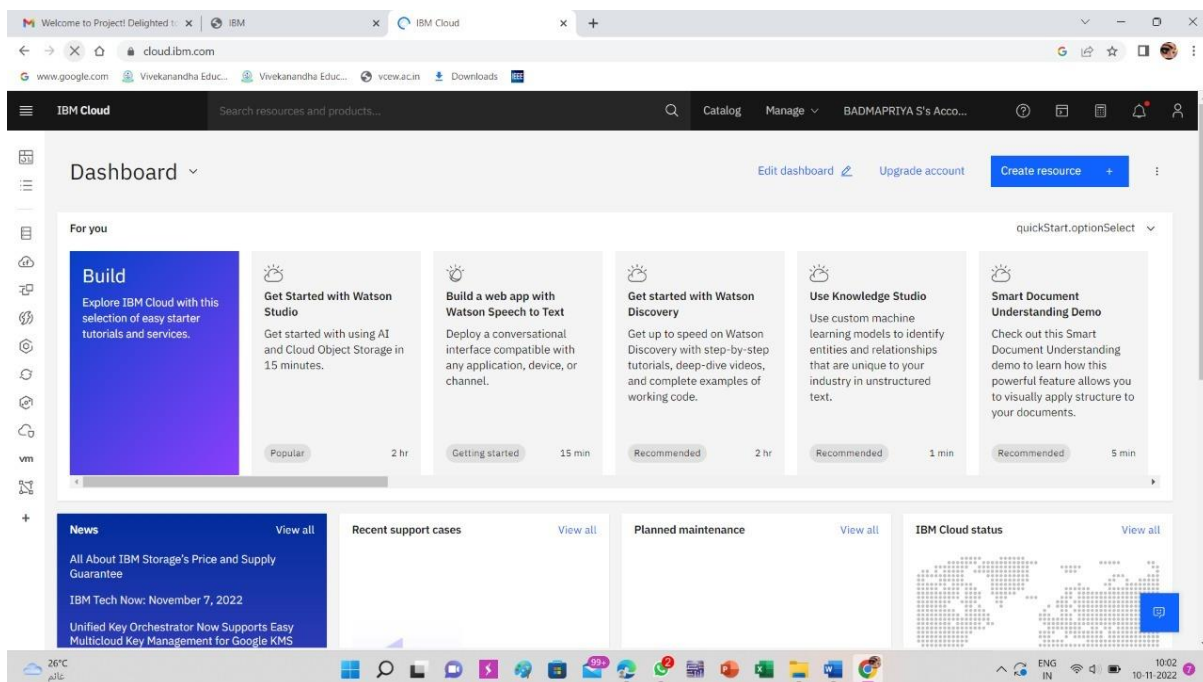


# REAL TIME RIVER WATER QUALITY MONITORING AND CONTROL SYSTEM

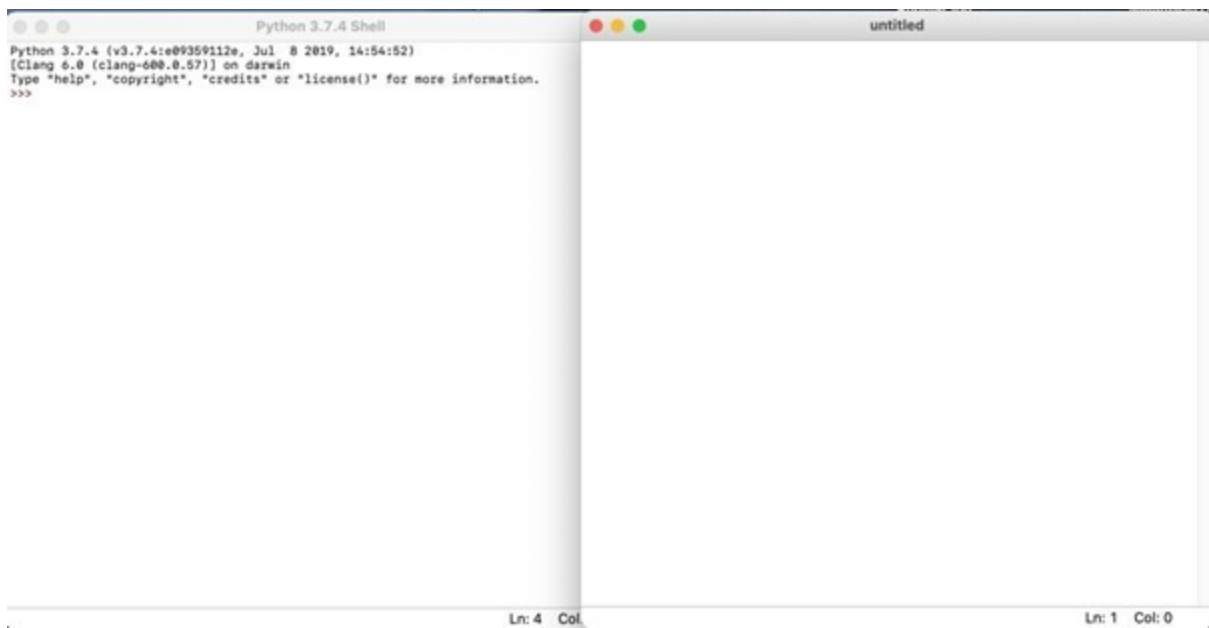
TEAM ID	PNT2022TMID23797
PROJECT NAME	REAL TIME RIVER WATER QUALITY MONITORING AND CONTROL SYSTEM

## PREREQUISITES:

## IBM CLOUD SERVICES

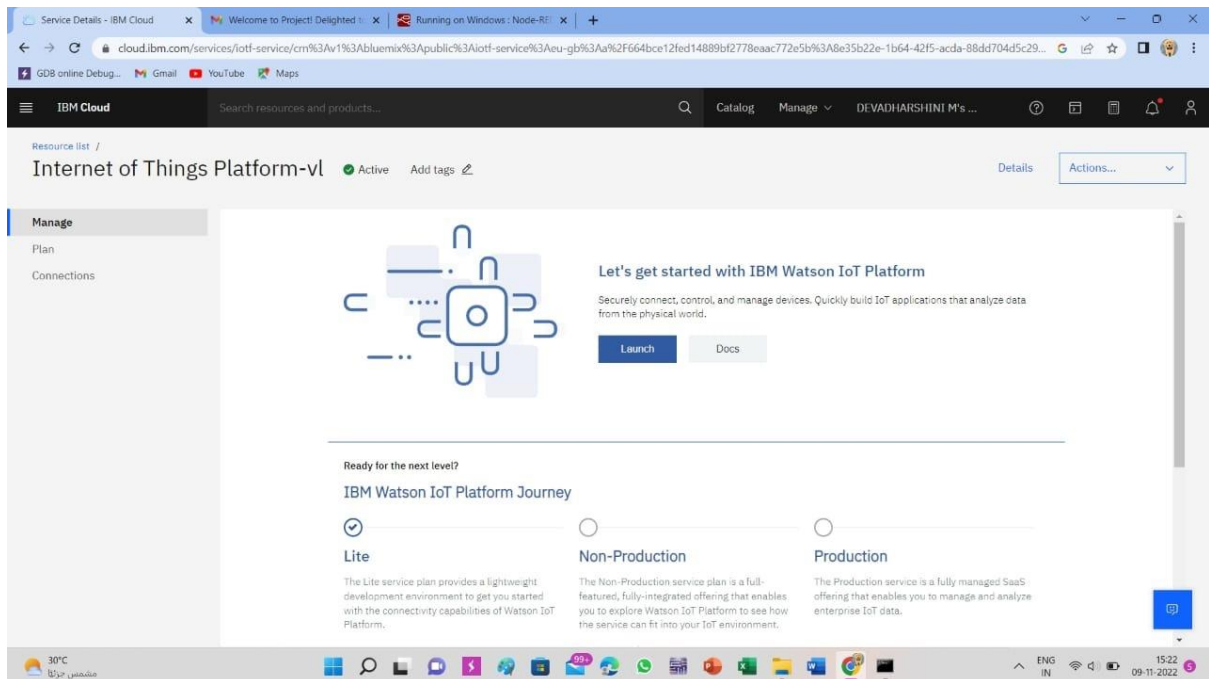


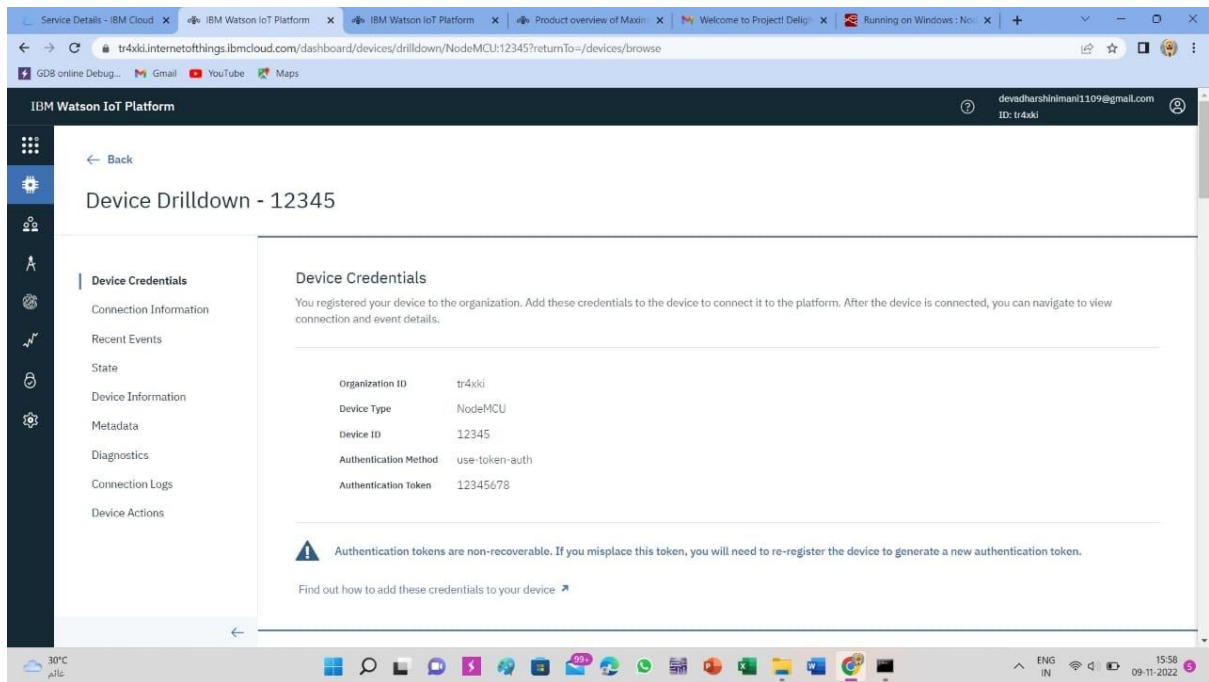
## SOFTWARE



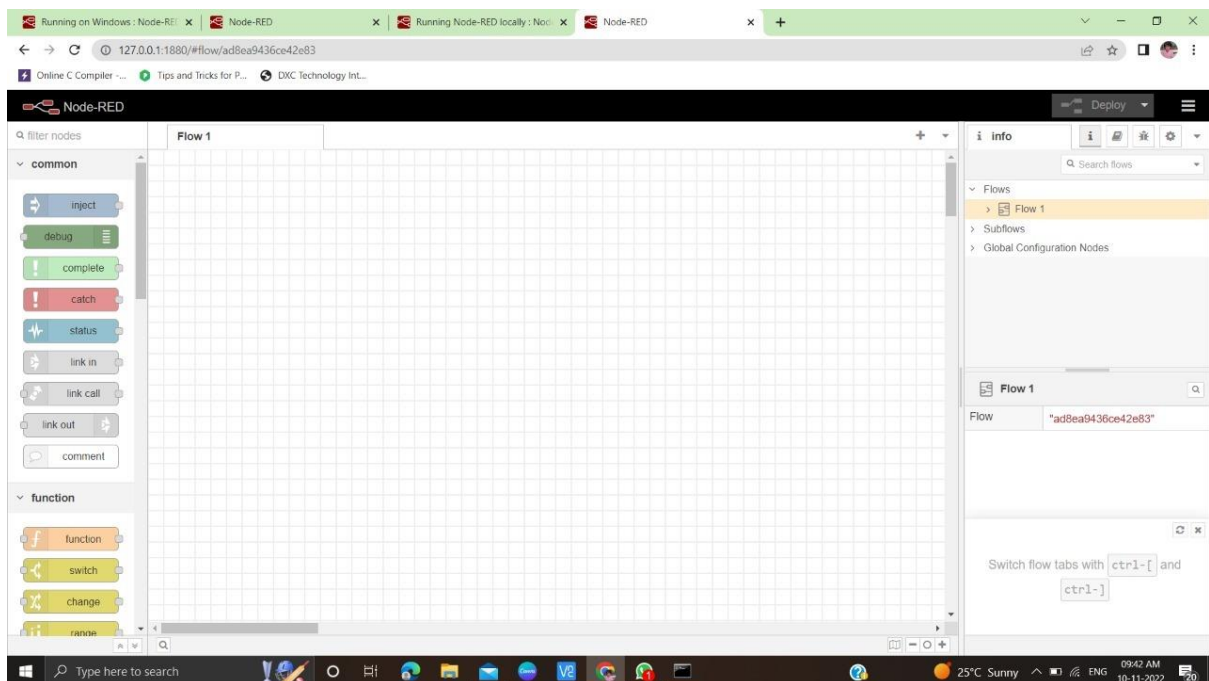
## **CREATE AN CONFIGURE IBM CLOUD SERVICES:**

### CREATE IBM WATSON IOT PLATFORM AND DEVICE





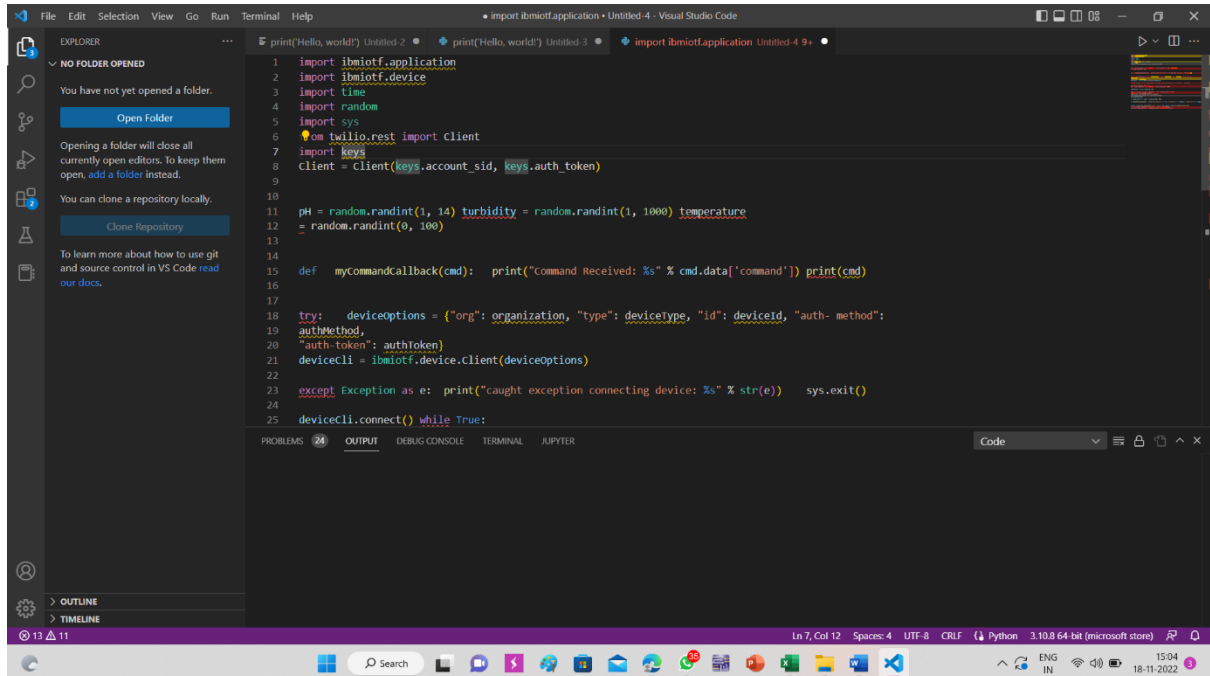
## CREATE NODE-RED SERVICE



# DEVELOP THE PYTHON SCRIPT

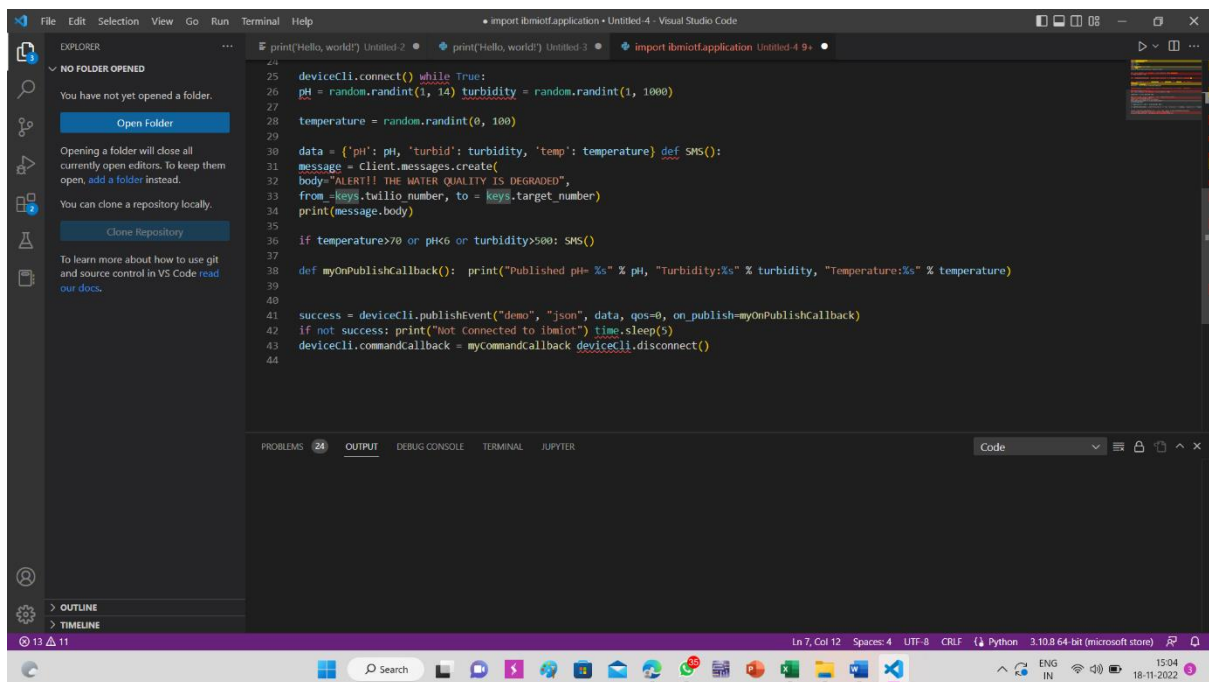
## DEVELOP A PYTHON SCRIPT

### CODING



The screenshot shows the Visual Studio Code editor with a Python script titled 'import ibmiotfApplication Untitled-4 9+'. The script imports necessary modules and sets up a Twilio client for sending SMS messages. It also initializes an IBM IoT device client with random sensor data (pH, turbidity, temperature). The script includes a callback function for receiving commands and a try-except block for handling connection errors.

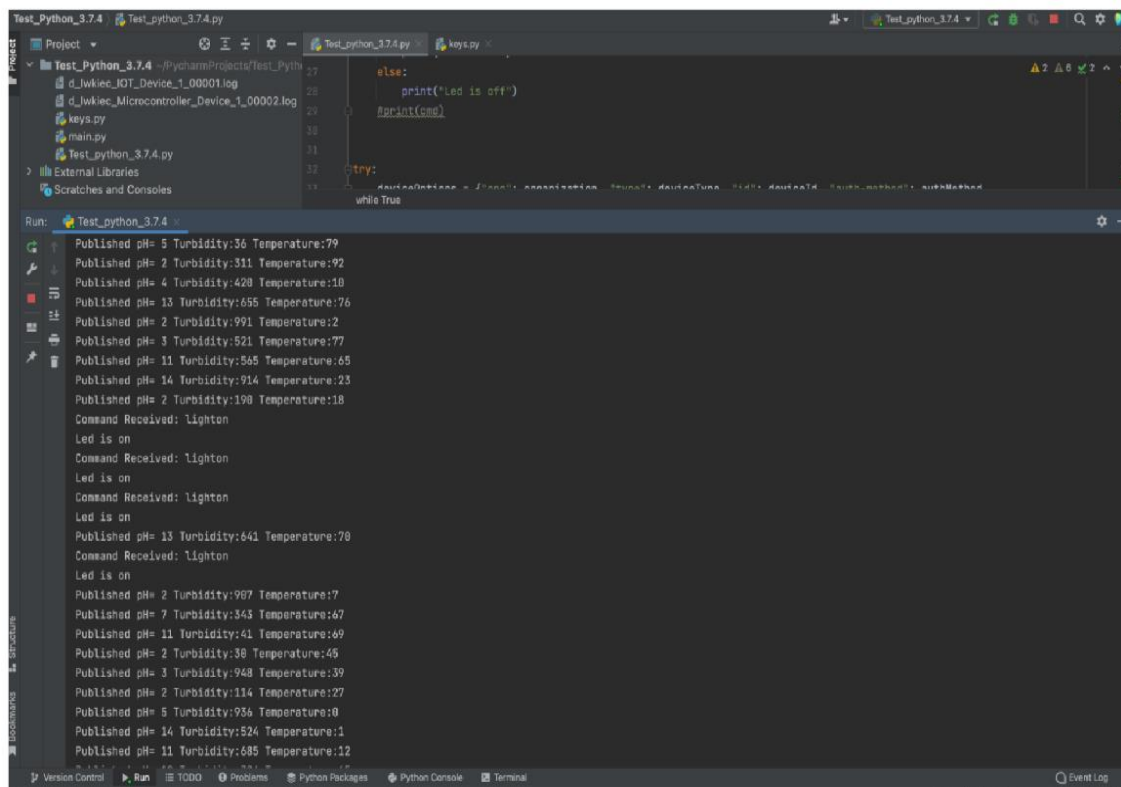
```
1 import ibmiotf.application
2 import ibmiotf.device
3 import time
4 import random
5 import sys
6 from twilio.rest import Client
7 import keys
8 client = Client(keys.account_sid, keys.auth_token)
9
10
11 pH = random.randint(1, 14) turbidity = random.randint(1, 1000) temperature
12 = random.randint(0, 100)
13
14
15 def myCommandCallback(cmd): print("Command Received: %s" % cmd.data['command']) print(cmd)
16
17
18 try: deviceOptions = {"org": organization, "type": devicetype, "id": deviceid, "auth-method":
19 authMethod,
20 "auth-token": authToken}
21 deviceCli = ibmiotf.device.Client(deviceOptions)
22
23 except Exception as e: print("caught exception connecting device: %s" % str(e)) sys.exit()
24
25 deviceCli.connect() while True:
```



The screenshot shows the continuation of the Python script in Visual Studio Code. It includes logic for creating and publishing messages to the IBM IoT Platform. The script checks for specific conditions (pH > 70, turbidity > 500) and sends an SMS alert if they are met. It also publishes sensor data to the IoT Platform and includes a disconnect function.

```
24
25 deviceCli.connect() while True:
26 pH = random.randint(1, 14) turbidity = random.randint(1, 1000)
27
28 temperature = random.randint(0, 100)
29
30 data = {'pH': pH, 'turbid': turbidity, 'temp': temperature} def SMS():
31 message = Client.messages.create(
32 body="ALERT!! THE WATER QUALITY IS DEGRADED",
33 from_=keys.twilio_number, to = keys.target_number)
34 print(message.body)
35
36 if temperature>70 or pH<6 or turbidity>500: SMS()
37
38 def myOnPublishCallback(): print("Published pH= %s" % pH, "Turbidity:%s" % turbidity, "Temperature:%s" % temperature)
39
40
41 success = deviceCli.publishEvent("demo", "json", data, qos=0, on_publish=myOnPublishCallback)
42 if not success: print("Not connected to ibmiot") time.sleep(5)
43 deviceCli.commandCallback = myCommandCallback deviceCli.disconnect()
44
```

## OUTPUT

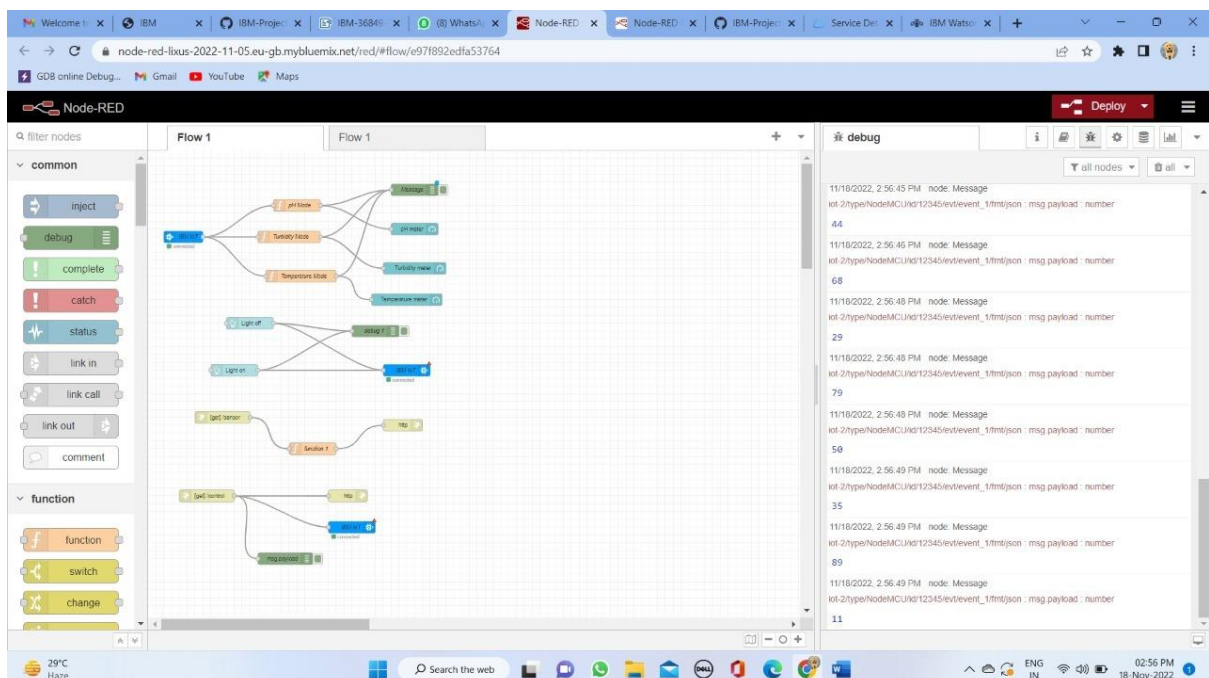


```
Test_python_3.7.4
Project
  Test_python_3.7.4
    d_jwkiec_IOT_Device_1_00001.log
    d_jwkiec_Microcontroller_Device_1_00002.log
    keys.py
    main.py
    Test_python_3.7.4.py
  External Libraries
  Scratches and Consoles

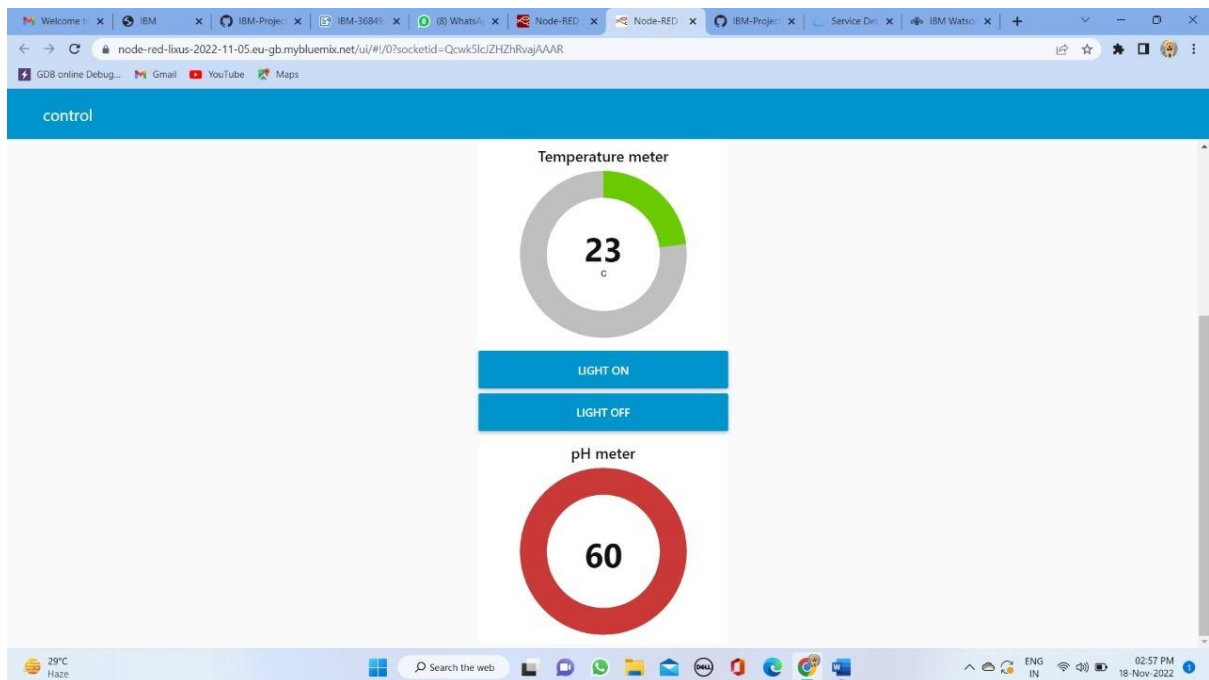
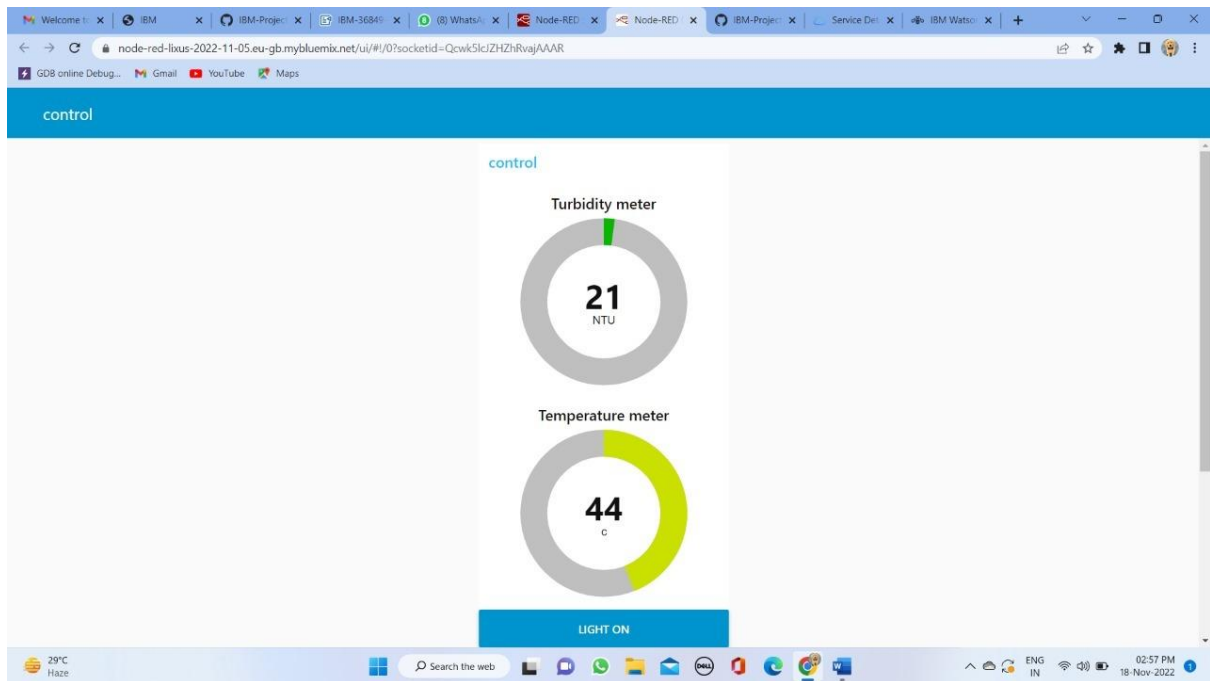
Run: Test_python_3.7.4
Published pH= 5 Turbidity:36 Temperature:79
Published pH= 2 Turbidity:311 Temperature:92
Published pH= 4 Turbidity:428 Temperature:10
Published pH= 13 Turbidity:695 Temperature:76
Published pH= 2 Turbidity:991 Temperature:2
Published pH= 3 Turbidity:521 Temperature:77
Published pH= 11 Turbidity:565 Temperature:65
Published pH= 14 Turbidity:914 Temperature:23
Published pH= 2 Turbidity:198 Temperature:18
Command Received: lighton
Led is on
Command Received: lighton
Led is on
Command Received: lighton
Led is on
Published pH= 13 Turbidity:641 Temperature:70
Command Received: lighton
Led is on
Published pH= 2 Turbidity:997 Temperature:7
Published pH= 7 Turbidity:543 Temperature:67
Published pH= 11 Turbidity:41 Temperature:69
Published pH= 2 Turbidity:38 Temperature:45
Published pH= 3 Turbidity:948 Temperature:39
Published pH= 2 Turbidity:114 Temperature:27
Published pH= 5 Turbidity:936 Temperature:0
Published pH= 14 Turbidity:524 Temperature:1
Published pH= 11 Turbidity:685 Temperature:12
```

## DEVELOP A WEB APPLICATION USING NODE-RED SERVICES

### DEVELOP THE WEB APPLICATION USING NODE-RED

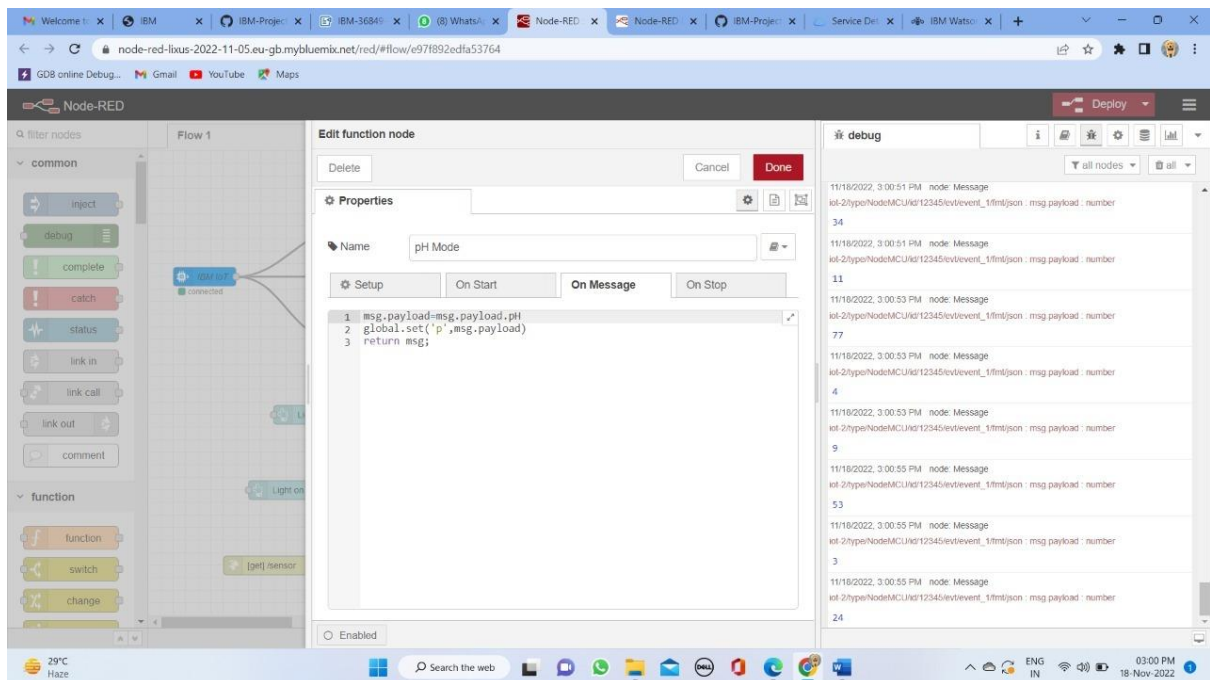


## USING DASHBOARD NODES FOR CREATING UI(WEB APP)



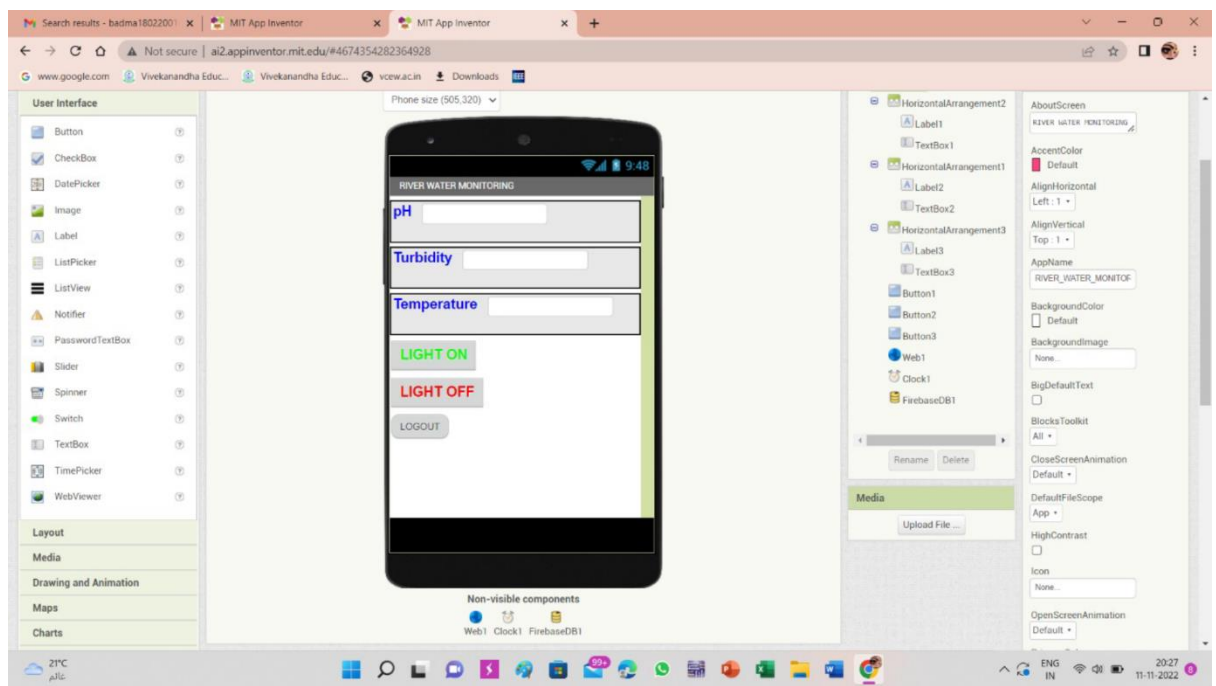


## CREATE AN HTTP REQUESTS TO COMMUNICATE WITH MOBILE APP



## BUILDING MOBILE APP

### DESIGN YOUR UI TO DISPLAY THE WATER,TURBIDITY,PH VALUES



CONFIGURE THE APPLICATION TO RECEIVE THE DATA FROM CLOUD

