

SPRINT 2

IBM CLOUD

The screenshot shows the IBM Watson IoT Platform interface. On the left, a code editor displays a Python script for a Raspberry Pi 4 microcontroller. The script generates random data for pH, turbidity, and temperature, and publishes it to the IoT platform. The right side of the interface shows a list of recent events, each containing a timestamp, pH value, turbidity value, and temperature value.

```
Test_Python_3.7.4 --PythonProjects/Test_Pyth
d:\Jewlec_Microcontroller_Device_1_00002.log
main.py
Test_Python_3.7.4.py
External Libraries
Scratches and Consoles

Run: Test_Python_3.7.4
Published pH: 4 Turbidity: 242 Temperature: 71
Published pH: 12 Turbidity: 564 Temperature: 54
Published pH: 2 Turbidity: 571 Temperature: 98
Published pH: 7 Turbidity: 677 Temperature: 65
Published pH: 8 Turbidity: 352 Temperature: 13
Published pH: 5 Turbidity: 862 Temperature: 88
Published pH: 3 Turbidity: 834 Temperature: 7
Published pH: 9 Turbidity: 213 Temperature: 89
Published pH: 14 Turbidity: 677 Temperature: 22
Published pH: 11 Turbidity: 292 Temperature: 160
Published pH: 2 Turbidity: 53 Temperature: 21
Published pH: 6 Turbidity: 499 Temperature: 69
Published pH: 11 Turbidity: 238 Temperature: 26
Published pH: 2 Turbidity: 443 Temperature: 43
Published pH: 6 Turbidity: 986 Temperature: 91
Published pH: 5 Turbidity: 593 Temperature: 85
Published pH: 14 Turbidity: 388 Temperature: 86
Published pH: 4 Turbidity: 532 Temperature: 8
Published pH: 3 Turbidity: 654 Temperature: 65
```

IBM Watson IoT Platform

Browse Action Device Types Interfaces Add Device

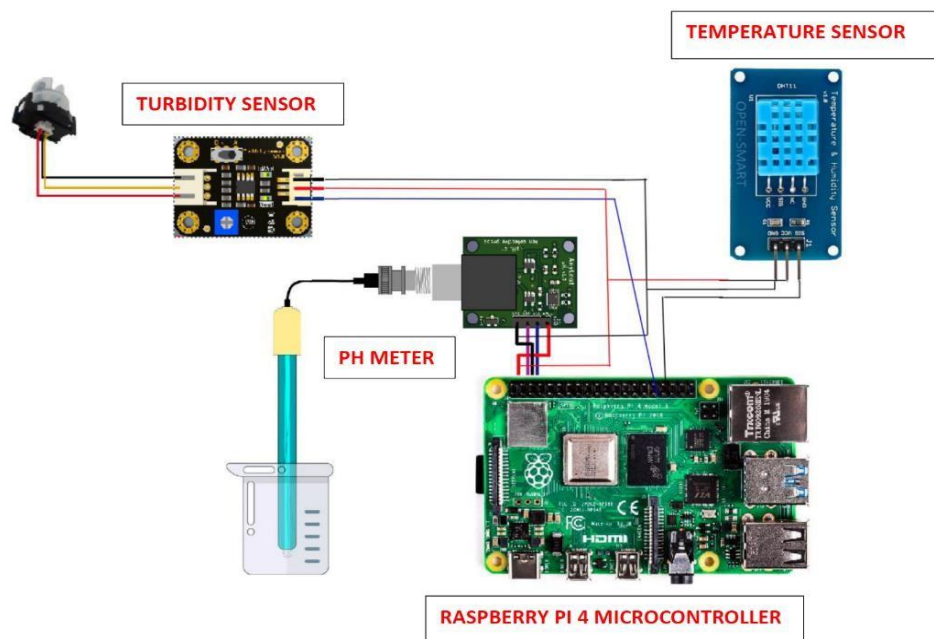
The recent events listed show the live stream of data that is coming an

Event	Value
demo	("pH":12,"turbid":93,"temp":87)
demo	("pH":7,"turbid":873,"temp":94)
demo	("pH":3,"turbid":204,"temp":19)
demo	("pH":11,"turbid":304,"temp":77)
demo	("pH":13,"turbid":16,"temp":50)

00003 Disconnected Micro_controller_2 Devi

Items per page 50 | 1-3 of 3 items 1 of 1 page < 1 >

CIRCUIT DAIGRAM



CODING

```
Untitled - Notepad
File Edit Format View Help
import ibmiotf.application
import ibmiotf.device
import time
import random
import sys
from twilio.rest import Client
import keys
Client = Client(keys.account_sid, keys.auth_token)

organization = "8n6m92"
deviceType = "Node"
deviceId = "868283"
authMethod = "use-token-auth"
authToken = "86828329"

pH = random.randint(1, 14)
turbidity = random.randint(1, 100)
temperature = random.randint(0, 100)

def myCommandCallback(cmd):
    print("Command Received: %s" % cmd.data['command'])
    print(cmd)

try:
    deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMethod,
                    "auth-token": authToken}
    deviceCli = ibmiotf.device.Client(deviceOptions)

except Exception as e:
    print("caught exception connecting device: %s" % str(e))
    sys.exit()

deviceCli.connect()

while True:
    pH = random.randint(1, 14)
    turbidity = random.randint(1, 100)
    temperature = random.randint(0, 100)
```

```
Untitled - Notepad
File Edit Format View Help
def myCommandCallback(cmd):
    print("Command Received: %s" % cmd.data['command'])
    print(cmd)

try:
    deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMethod,
                    "auth-token": authToken}
    deviceCli = ibmiotf.device.Client(deviceOptions)

except Exception as e:
    print("caught exception connecting device: %s" % str(e))
    sys.exit()

deviceCli.connect()

while True:
    pH = random.randint(1, 14)
    turbidity = random.randint(1, 100)
    temperature = random.randint(0, 100)

    data = {'pH': pH, 'turbid': turbidity, 'temp': temperature}
    def SMS():
        message = Client.messages.create(
            body="ALERT!! THE WATER QUALITY IS DEGRADED",
            from_=keys.twilio_number,
            to = keys.target_number)
        print(message.body)

    if temperature>70 or pH<6 or turbidity>500:
        SMS()

    def myOnPublishCallback():
        print("Published pH= %s" % pH, "Turbidity:%s" % turbidity, "Temperature:%s" % temperature)

    success = deviceCli.publishEvent("demo", "json", data, qos=0, on_publish=myOnPublishCallback)
    if not success:
        print("Not Connected to ibmiot")
        time.sleep(5)
    deviceCli.commandCallback = myCommandCallback

deviceCli.disconnect()
```