

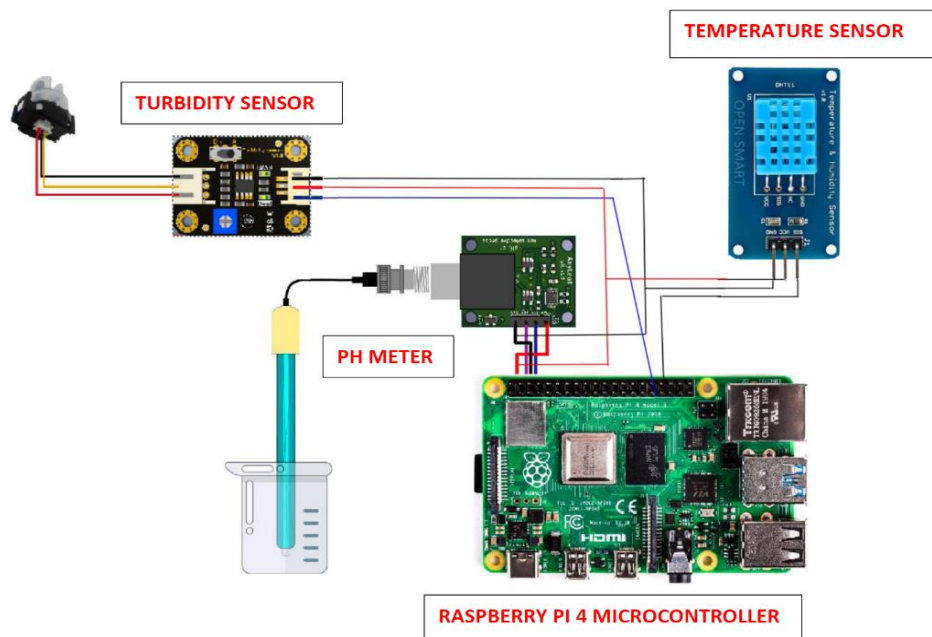
## SPRINT 2

### IBM CLOUD

The screenshot displays the IBM Watson IoT Platform interface. On the left, a code editor shows a Python script named `Test_Python_3.7.4.py` that generates random data for pH, turbidity, and temperature. The script is running on a Raspberry Pi, as indicated by the `Run` tab output. The output shows a series of published data points, such as `Published pH= 4 Turbidity:242 Temperature:71`. On the right, the IBM Watson IoT Platform console shows the recent events listed, which are the live stream of data coming from the device. The events are displayed in a table with columns `Event` and `Value`.

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)

### CIRCUIT DAIGRAM



# CODING

```
untitled.txt - Notepad
File Edit View

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 = "m03gv5"
devicetype = "NodeMCU"
deviceId = "12345"
authMethod = "use-token-auth"
authToken = "123456789"
pH = random.randint(1, 14)
turbidity = random.randint(1, 1000)
temperature = random.randint(0, 100)

def myCommandcallback (cmd) :
    print ("Command Received: Xs" % cmd.data["command"])
    print (cmd)

try:
    deviceoptions = {"org": organization, "type": devicetype, "id": deviceId, "auth-method": authMethod,
    "auth-token": authToken}

    devicecli =ibmotf .device.Client (deviceoptions)

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

devicecli.connect()

while True:
    random.randint(1, 14)
Ln 40, Col 22
100% Unix (LF) UTF-8
25°C Cloudy Search 19:18 20-11-2022
File Edit View Close

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, 1000)
    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()
#Twilio Account Credentials
account_sid = 'ACa0eb9bf43aa629b503bdd01d0962d465'
auth_token = '48c1a0ae0472038ab36d45e0d9fb6e7'
twilio_number = '+19804095xxx'
target_number = '+919940555xxx'
Ln 70, Col 31
100% Windows (CRLF) UTF-8
25°C Partly cloudy Search 22:58 17-11-2022
```