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 = "lwkiec"

deviceType = "Microcontroller\_Device\_1" deviceId = "00002"

authMethod = "token" authToken = "sushi@123"

pH = random.randint(1, 14) turbidity = random.randint(1, 1000)

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, 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()