REAL TIME RIVER WATER QUALITY MONITORING AND CONTROL SYSTEM

TEAM ID:PNT2022TMID33851

PYTHON SCRIPT

```
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random

#Provide your IBM WatSON Credentials

organization = "eo8548" #replace the ORG ID
deviceType = "sensor"#replace the Device type wi
deviceId = "12"#replace Device ID
authMethod = "token"
authToken = "123456789" #Replace the authtoken

# Initialize GPIO

#Receives Command from Node-red

def myCommandCallback(cmd):
 print ("Command received: %s" % cmd.data['command'])
 status=cmd.data['command']
 if status=="turbidity<50":
    if status =="temp<17 " :
       if status == "ph>=7" :
          if status=="do>7:
             print("Portable water for drinking and bio-life")
          print("Acidic")
      print ("Affects bio-life")
print ("Unfit for consumption")
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()
# Connect and send a datapoint "hello" with value "world" into the cloud as an event of
type "greeting" 10 times
deviceCli.connect()
while True:
#Get Sensor Data from DHT11
 temp=random.randint(0,100)
 ph=random.randint(0,14)
 turbidity=random.randint(0,100)
 do=random.randint(0,10)
 data = { 'temp' : temp, 'ph': ph, 'turbidity': turbidity,'do':do }
#print data
 def myOnPublishCallback():
  print ("Published Temperature = %s C" % temp, "ph = %s %%" % ph, "turbidity = %s
%%"% turbidity, "do=%s %%"% do,"to IBM Watson")
  success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0,
on_publish=myOnPublishCallback)
  if not success:
    print("Not connected to IoTF")
  time.sleep(5)
  deviceCli.commandCallback = myCommandCallback
# Disconnect the device and application from the cloud
deviceCli.disconnect()
```