Final Deliverables

FINAL CODE

Date	**11 November 2022**
Team ID	**PNT2022TMID47383**
Project Name	**Real-Time River Water Quality Monitoring and Control System**

Python Code:

```
import random
import ibmiotf.application
import ibmiotf.device
from time import sleep
import sys
 #IBM Watson Device Credentials.
organization = "39hari"
deviceType = "NodeMCU"
deviceId = "ESP32"
authMethod = "token"
authToken = "6369702210"
def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status = cmd.data['command']
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()
#Connecting to IBM watson.
    deviceCli.connect()
while True:
 temp_sensor = round( random.uniform(0,80),2)
 PH_sensor = round(random.uniform(1,14),3)
 Turbidity_sensor =round(random.uniform(0,1000),2)
 OPTOD_sensor = round(random.uniform(0,10),1)
 TOC_sensor = round(random.uniform(0,3000),4)
 water_rate = round(random.uniform(0,500),3)

#storing the sensor data to send in json format to cloud.

 temp_data = { 'Temperature' : temp_sensor }
 PH_data = { 'PHLevel' : PH_sensor }
 Turbidity_data = {'Turbidity Level' : Turbidity_sensor}
 OPTOD_data = { 'Oxygen Level' : OPTOD_sensor}
 TOC_data = { 'Water Level' : TOC_sensor}
 water_data = { 'Water rate' : water_rate}

# publishing Sensor data to IBM Watson for every 5-10 seconds.
 success = deviceCli.publishEvent("Temperature sensor", "json", temp_data, qos=0)
 sleep(1)
 if success:
    print (" ..... .....publis h ok..... ")
```

```
    print ("Published Temperature = %s C" % temp_sensor, "to IBM Watson")
 success = deviceCli.publishEvent("PH sensor", "json", PH_data, qos=0)
 sleep(1)
 if success:
    print ("Published PHLevel = %s" % PH_sensor, "to IBM Watson")
 success = deviceCli.publishEvent("Turbidity sensor", "json", Turbidity_data, qos=0)
 sleep(1)
 if success:
    print ("Published Turbidity Level %s " % Turbidity_sensor, "to IBM Watson")
 success = deviceCli.publishEvent("optod sensor", "json", OPTOD_data, qos=0)
 sleep(1)
 if success:
    print ("Published Flame %s " % OPTOD_sensor, "to IBM Watson")
 success = deviceCli.publishEvent("Moisture sensor", "json", TOC_data, qos=0)
 sleep(1)
 if success:
    print ("Published Moisture Level = %s " % TOC_sensor, "to IBM Watson")
 success = deviceCli.publishEvent("Water sensor", "json", water_data, qos=0)
 sleep(1)
 if success:
    print ("Published Water rate = %s cm" % water_rate, "to IBM Watson")
    print ("")

#Automation to RO plants and water treatment plants by present temperature an to send alert message to IBM Watson.
 if (temp_sensor > 35):

    print("Intake to Filteration is Closed")
 success = deviceCli.publishEvent("Alert1", "json",{ 'alert1' : "Temperature(%s) is high, Intake is plant is closed "
%temp_sensor } , qos=0)
 sleep(1)
 if success:
    print( 'Published alert1 : ', "Temperature(%s) is high, Intake to plant is closed" %temp_sensor,"to IBM Watson")
    print("")
 else:
    print("Intake to plant is resumed ")
    print("")

#To send alert message if the ph is more acidic or basic to the local water board authorities.
 if (PH_sensor > 7.5 or PH_sensor < 5.5):
    success = deviceCli.publishEvent("Alert2", "json",{ 'alert2' : "Water is highly acidic/basis alert is sent to
authorities" %PH_sensor } , qos=0)
    sleep(1)
 if success:
    print('Published alert2 : ' , "Harmfull for human consumption" %PH_sensor,"to IBM Watson")
    print("")
 else:
    print('Published alert2 : ' , "Ph neutral safe for consumption" %PH_sensor,"to IBM Watson")
    print("")
#To send alert message depending upon the impurity level.

if (Turbidity_sensor > 500):
   print("Impurities too high diversion to Industrial use")
   success = deviceCli.publishEvent("Alert3", "json", { 'alert3' : "Water is in taken for the industrial use " }, qos=0)
   sleep(1)
if success:
```

```
   print( 'Published alert3 : ' , "Impurities is high amount can't be used for human consumption or for the filteration
process","to IBM Watson")
   print("")
else:
   if (Turbidity_sensor < 20):
      print("Admissable level can be used for human consumption")
      success = deviceCli.publishEvent("Alert3", "json", { 'alert3' : "instructs the public to utilize the water" }, qos=0)
      sleep(1)
      if success:
         print( 'Published alert3 : ' , "Directly used without filteration for basic needs","to IBM Watson")
         print("")
      else:
         print("Switched to Ro and sedemantation process")
         print("")
#To send alert message depending upon the oxygen content present in water.

if (OPTOD_sensor < 2):
   print("low level of dissolved oxygen in river water")
   success = deviceCli.publishEvent("Alert3", "json", { 'alert4' : "Crital level of dissolved oxygen content in water  " },
qos=0)
   sleep(1)
if success:
   print( 'Published alert4 : ' , "Freshwater creatures may be dying required help","to IBM Watson")
   print("alert is send marine department")
else:
   if (OPTOD_sensor > 8):
      print("High oxygen content present")
      success = deviceCli.publishEvent("Alert4", "json", { 'alert3' : "no action is to be taken:indicates hoe fresh is
water" }, qos=0)
      sleep(1)
      if success:
         print( 'Published alert4 : ' , "good to consume since it is fresh","to IBM Watson")
         print("")
      else:
         print("Permissible level to sustain minimum no of creatures in water")
         print("")
#Automation to detect the amount of organic carbon an to send alert message to IBM Watson.
if (TOC_sensor > 900):
   print(" high amount of organic carbon")
   success = deviceCli.publishEvent("Alert1", "json",{ 'alert1' : "Water needs immediate treatment"} , qos=0)
   sleep(1)
if success:
   print( 'Published alert1 : ', "Authorities alerted prevented loss in lives","to IBM Watson")
   print("")
else:
   print("Water consumption is stoped")
   print("")
if (water_rate > 180):
   print("sludge gates are opened")
   success = deviceCli.publishEvent("Alert6", "json", { 'alert6' : "WReservoiur authorites are alerted and NDRF is
informed to be in stand_by " %water_rate }, qos=0)
   sleep(1)
if success:
   print('Published alert6 : ' , "water rate is high so it indicates rain or reserviour release of water into the stream "
%water_rate,"to IBM Watson" )
   print("")
```

```
else:
    print("Sludge is opened")
    print("")
deviceCli.commandCallback = myCommandCallback
# Disconnect the device and application from the cloud
deviceCli.disconnect()
```