Develop a Python Script

Team ID	PNT2022TMID06691
Project Name	Real-Time River Water Quality Monitoring and Control System

Python Script:

```
import random

import time

import sys

import ibmiotf.application

import ibmiotf.device


# Provide your IBM Watson Device Credentials

organization = "um5y3e"  # repalce it with organization ID

deviceType = "ESP32"  # replace it with device type

deviceId = "13448"  # repalce with device id

authMethod = "token"

authToken = "8883686824"  # repalce with token


def myCommandCallback(cmd):

    print("Command received: %s" % cmd.data)

    if cmd.data['command'] == 'lighton':

        print("LIGHT ON")

    elif cmd.data['command'] == 'lightoff':

        print("LIGHT OFF")
```

```
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(0,100)
    cond = random.randint(0,100)
    temp = random.randint(0,100)
    oxy = random.randint(0,100)
    turb = random.randint(0,100)
    # Send Temperature & Humidity to IBM Watson
    data = {'Temperature': temp, 'PH': ph, 'Conductivity': cond, 'Oxygen': oxy, "Turbidity": turb}
 # print data
    def myOnPublishCallback():
        print("Published data",data, "to IBM Watson")
 success = deviceCli.publishEvent("event", "json", data, 0, myOnPublishCallback)
    if not success:
        print("Not connected to IoTF")
    time.sleep(5)
  deviceCli.commandCallback = myCommandCallback
```

Python Script running in Jupyter Notebook:

jupyter Untitled Last Checkpoint: 3 minutes ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

Run Code

In [*]:
```
import random
import time
import sys
import ibmiotf.application
import ibmiotf.device


# Provide your IBM Watson Device Credentials

organization = "um5y3e"  # repalce it with organization ID
deviceType = "ESP32"  # replace it with device type
deviceId = "13448"  # repalce with device id
authMethod = "token"
authToken = "8883686824"  # repalce with token


def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data)
    if cmd.data['command'] == 'lighton':
        print("LIGHT ON")
    elif cmd.data['command'] == 'lightoff':
        print("LIGHT OFF")


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()
```

```
deviceCli.connect()

while True:
    ph = random.randint(0,100)
    cond = random.randint(0,100)
    temp = random.randint(0,100)
    oxy = random.randint(0,100)
    turb = random.randint(0,100)
    # Send Temperature & Humidity to IBM Watson
    data = {'Temperature': temp, 'PH': ph, 'Conductivity': cond, 'Oxygen': oxy, "Turbidity": turb}


    # print data
    def myOnPublishCallback():
        print("Published data",data, "to IBM Watson")


    success = deviceCli.publishEvent("event", "json", data, 0, myOnPublishCallback)
    if not success:
        print("Not connected to IoTF")
    time.sleep(5)

    deviceCli.commandCallback = myCommandCallback
```