PROJECT DEVELOPMENT PHASE

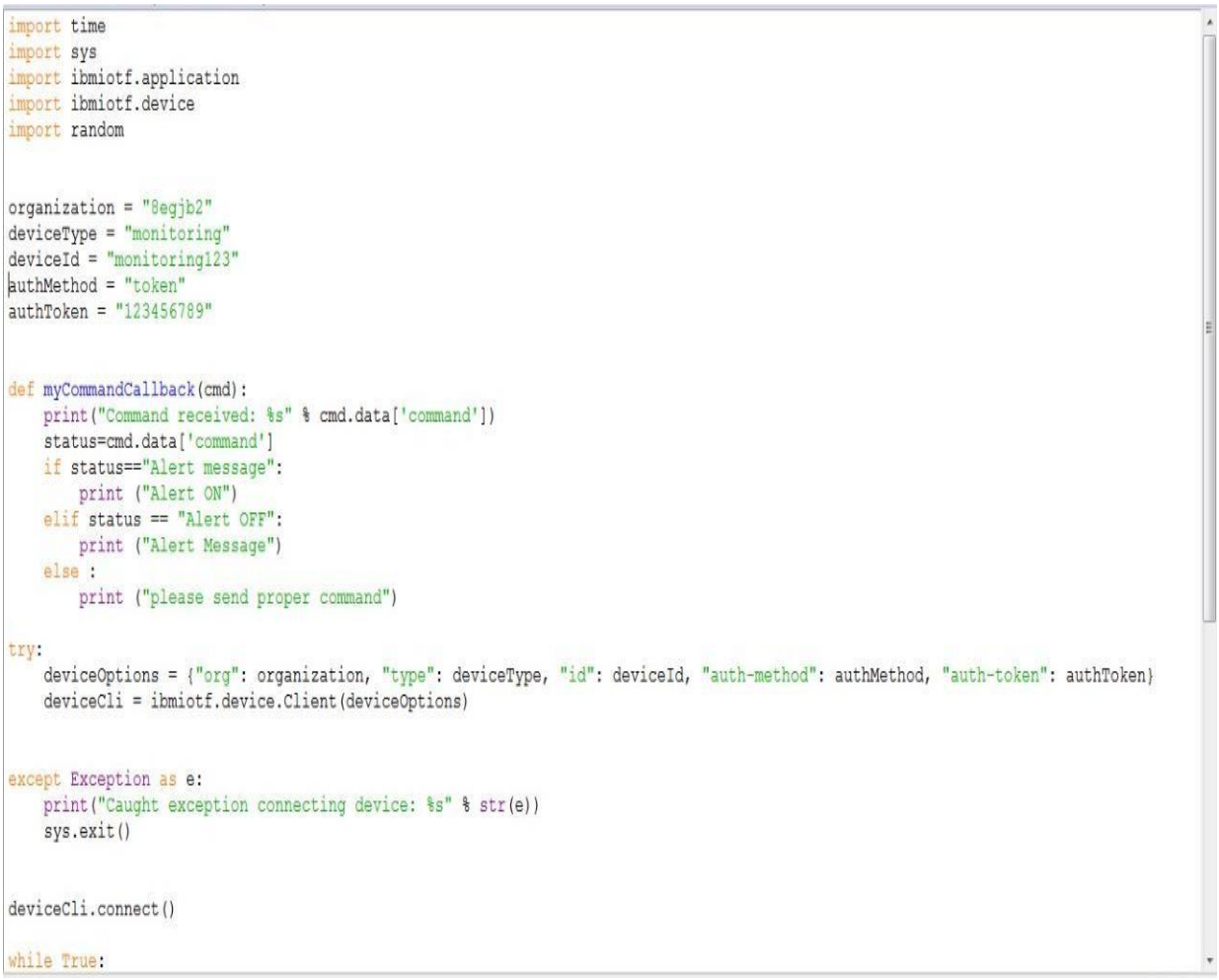
DELIVERY OF SPRINT – 1

Team ID	PNT2022TMID46761
Team Name	Real-Time River Water Quality Monitoring and Control System

PYTHON SCRIPT

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


organization = "8egjb2"
deviceType = "monitoring"
deviceId = "monitoring123"
authMethod = "token"
authToken = "123456789"


def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status=cmd.data['command']
    if status=="Alert message":
        print ("Alert ON")
    elif status == "Alert OFF":
        print ("Alert Message")
    else :
        print ("please send proper 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()


deviceCli.connect()

while True:
```

```
        Temp=random.randint(0,100)
        pH=random.randint(0,14)
        Turbidity=random.randint(0,100)

        data = { 'Temp' : Temp, 'pH' : pH, 'Turbidity': Turbidity }

        def myOnPublishCallback():
            print ("Published, Temperature = %s %%" % Temp, "pH_Value = %s pH" % pH,
"Turbidity_Value = %s %%" % Turbidity, "to IBM Watson")

        success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0,
on_publish=myOnPublishCallback)
        if not success:
            print("Not connected to IoTF")
        time.sleep(10)

        deviceCli.commandCallback = myCommandCallback


deviceCli.disconnect()
```



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


organization = "8egjb2"
deviceType = "monitoring"
deviceId = "monitoring123"
authMethod = "token"
authToken = "123456789"


def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status=cmd.data['command']
    if status=="Alert message":
        print ("Alert ON")
    elif status == "Alert OFF":
        print ("Alert Message")
    else :
        print ("please send proper 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()


deviceCli.connect()

while True:
```

```
        Temp=random.randint(0,100)
        pH=random.randint(0,14)
        Turbidity=random.randint(0,100)

        data = { 'Temp' : Temp, 'pH' : pH, 'Turbidity': Turbidity }

        def myOnPublishCallback():
            print ("Published, Temperature = %s %%" % Temp, "pH_Value = %s pH" % pH, "Turbidity_Value = %s %%" % Turbidity, "to IBM Wa

        success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0, on_publish=myOnPublishCallback)
        if not success:
            print("Not connected to IoTF")
        time.sleep(10)

        deviceCli.commandCallback = myCommandCallback


deviceCli.disconnect()
```