

PROJECT DEVELOPMENT PHASE

SPRINT 1

Date	16 November 2022
Team ID	PNT2022TMID13514
Project name	Gas Leakage Monitoring & Alerting System for Industries

ANALYZE THE PREREQUISITES

Needed prerequisites for real time river water quality monitoring and control system using Internet Of Things (IoT) were

- ❖ IBM Watson IoT Platform
- ❖ Node-RED Service
- ❖ Cloudant DB

Python code:

```
import time

import sys

import ibmiotf.application

import ibmiotf.device

import random

organization = "l27fmg"

deviceType = "12345"

deviceId = "123456"

authMethod="token"

authToken = "123456789"

def myCommandCallback(cmd):

    print("Command received: %s" % cmd.data['command'])

    status=cmd.data['command']

    if status=="lighton":

        print ("led is on")

    elif status == "lightoff":
```

```

        print ("led is off")

    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:

    gasconcentration = random.randint(90,110)

    Humidity = random.randint(90,110)

    Temperature = random.randint(90,110)

    data = {'gasconcentration' : gasconcentration, 'Humidity' : Humidity, 'Temperature'
:Temperature}

    def myOnPublishCallback():

        print(" GasConcentration = %s PPM" % gasconcentration, "to IBM Watson")

        print(" Humidity = %s%%" % Humidity, "to IBM Watson")

        print(" Temperature = %s C" % Temperature, "to IBM Watson")

        success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0,
on_publish=myOnPublishCallback)

        if not success:

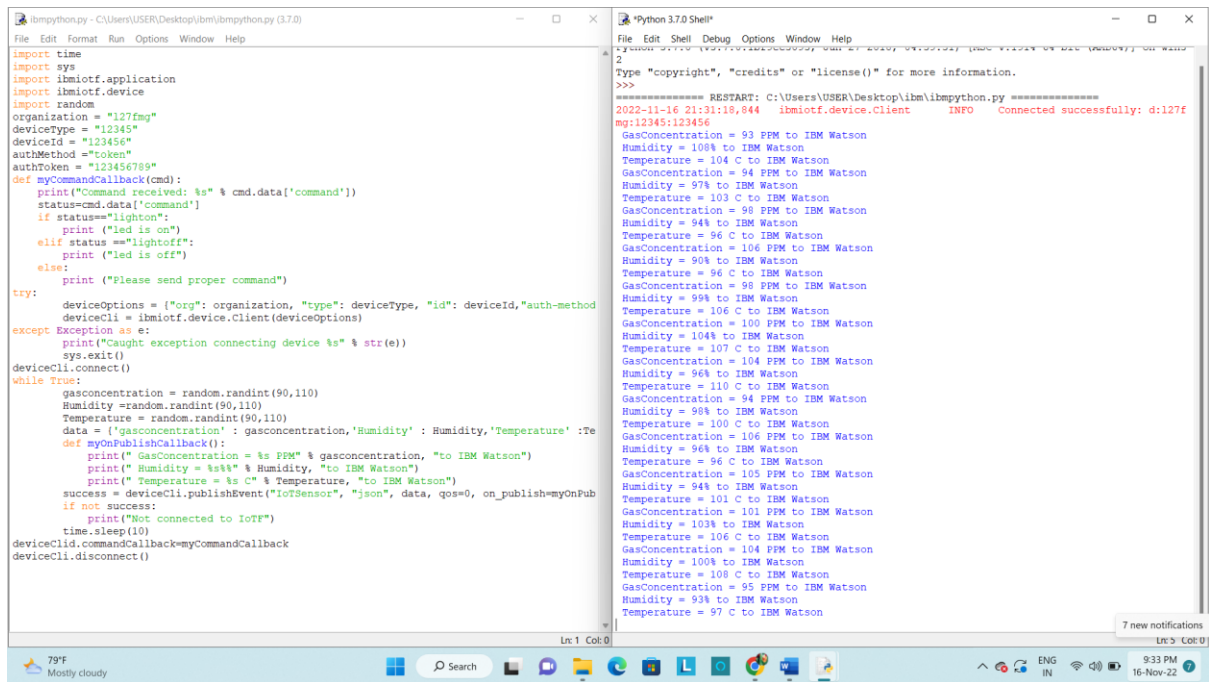
            print("Not connected to IoTF")

            time.sleep(10)

deviceCli.commandCallback=myCommandCallback

deviceCli.disconnect()

```



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

organization = "127fmg"
deviceType = "12345"
deviceId = "123456"
authMethod = "token"
authToken = "123456789"

def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status=cmd.data['command']
    if status=="lighton":
        print ("led is on")
    elif status=="lightoff":
        print ("led is off")
    else:
        print ("Please send proper command")

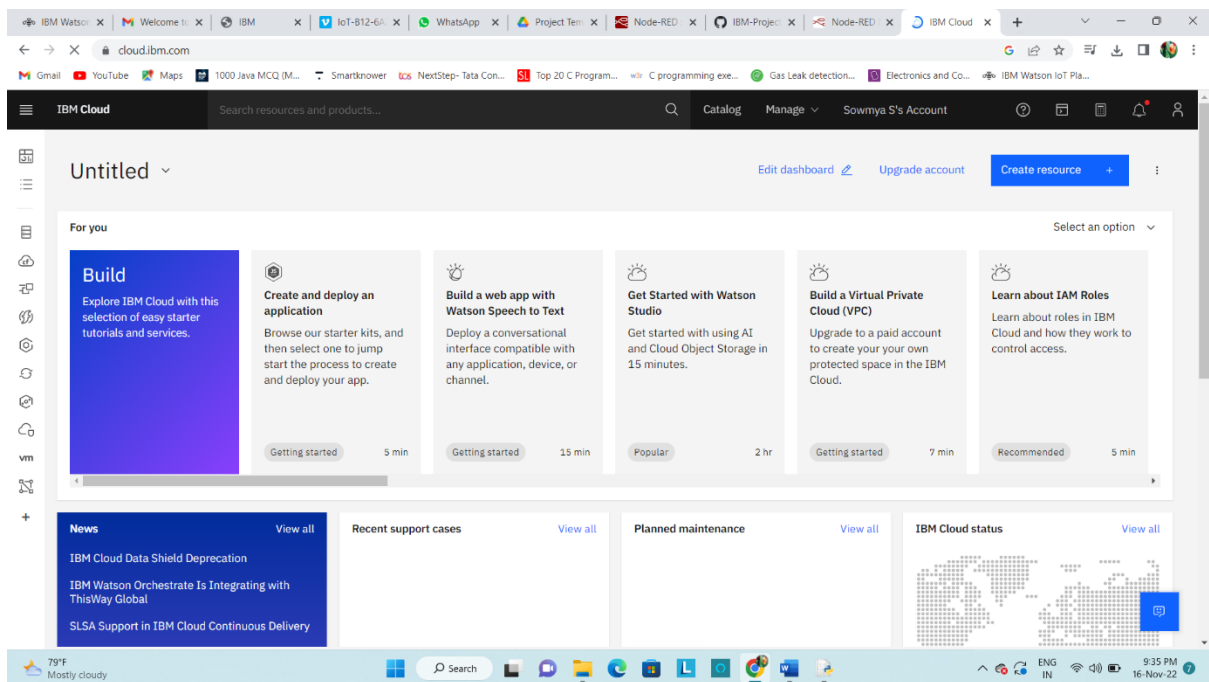
try:
    deviceOptions = {"org": organization, "type": deviceType, "id": deviceId,"auth-method":authMethod}
    deviceCli = ibmiotf.device.Client(deviceOptions)
except Exception as e:
    print("Caught exception connecting device %s" % str(e))
    sys.exit()

deviceCli.connect()

while True:
    gasconcentration = random.randint(90,110)
    Humidity =random.randint(90,110)
    Temperature = random.randint(90,110)
    data = {'gasconcentration': gasconcentration,'Humidity' : Humidity,'Temperature' :Temperature}
    def myOnPublishCallback():
        print(" GasConcentration = %s PPM" % gasconcentration, "to IBM Watson")
        print(" Humidity = %s%%" % Humidity, "to IBM Watson")
        print(" Temperature = %s C" % Temperature, "to IBM Watson")
    success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0, on_publish=myOnPub
    if not success:
        print("Not connected to IoT")
        time.sleep(10)
    deviceCli.commandCallback=myCommandCallback
    deviceCli.disconnect()
```

Code runs successfully and random output values are generated

Creation of IBM cloud



Procedure for the creation of IBM IOT watson

IBM Watson IoT Platform

Browse Action Device Types Interfaces

Add Device

Browse Devices

All Devices Diagnose

This table shows a summary of all devices that have been added. It can be filtered, organized, and searched on using different criteria. To get started, you can add devices by using the Add Device button, or by using API.

Search by Device ID

Device Simulator

Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
123456	Disconnected	12345	Device	Nov 12, 2022 10:49 AM	

Items per page 50 | 1-1 of 1 item

1 of 1 page

Device creation

IBM Watson IoT Platform

Browse Action Device Types Interfaces

Add Device

Browse Devices

All Devices Diagnose

This table shows a summary of all devices that have been added. It can be filtered, organized, and searched on using different criteria. To get started, you can add devices by using the Add Device button, or by using API.

Search by Device ID

Device Simulator

Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
123456	Disconnected	12345	Device	Nov 12, 2022 10:49 AM	

Items per page 50 | 1-1 of 1 item

1 of 1 page

Identity Device Information Recent Events State Logs

Device ID 123456

Device Type 12345

Date Added Nov 12, 2022 10:49 AM

Added By sowmyasekar2306@gmail.com

Connection Status Disconnected

Generation of random values in IBM Watson

The screenshot displays the IBM Watson IoT Platform dashboard. The top navigation bar includes tabs for 'Browse', 'Action', 'Device Types', and 'Interfaces'. The 'Recent Events' tab is selected, showing a table of live data streams. The table has four columns: 'Event', 'Value', 'Format', and 'Last Received'. It lists four events from an 'IoTSensor' device, each providing a JSON string of sensor data (gas concentration, humidity, and temperature) in 'json' format, received 'a few seconds ago'. The bottom of the dashboard shows a pagination control for '1 of 1 page' and a system status bar at the very bottom indicating '79°F Mostly cloudy' and the time '9:38 PM 16-Nov-22'.

Event	Value	Format	Last Received
IoTSensor	{"gasconcentration":100,"Humidity":109,"Tempe...	json	a few seconds ago
IoTSensor	{"gasconcentration":94,"Humidity":104,"Temper...	json	a few seconds ago
IoTSensor	{"gasconcentration":104,"Humidity":108,"Tempe...	json	a few seconds ago
IoTSensor	{"gasconcentration":98,"Humidity":109,"Temper...	json	a few seconds ago