

PROJECT DEVELOPMENT PHASE

Sprint 1

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

The project Development phase is divided into four parts (i.e., the four sprints). In this we are seeing the sprint 1 we have developed a python code to push our data into the cloud by creating our own python script. The code follows below

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 Filtration 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 filtration
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 filtration 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' : "Crtial 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 reservoir release of water into the stream "
%water_rate,"to IBM Watson" )
    print("")
```

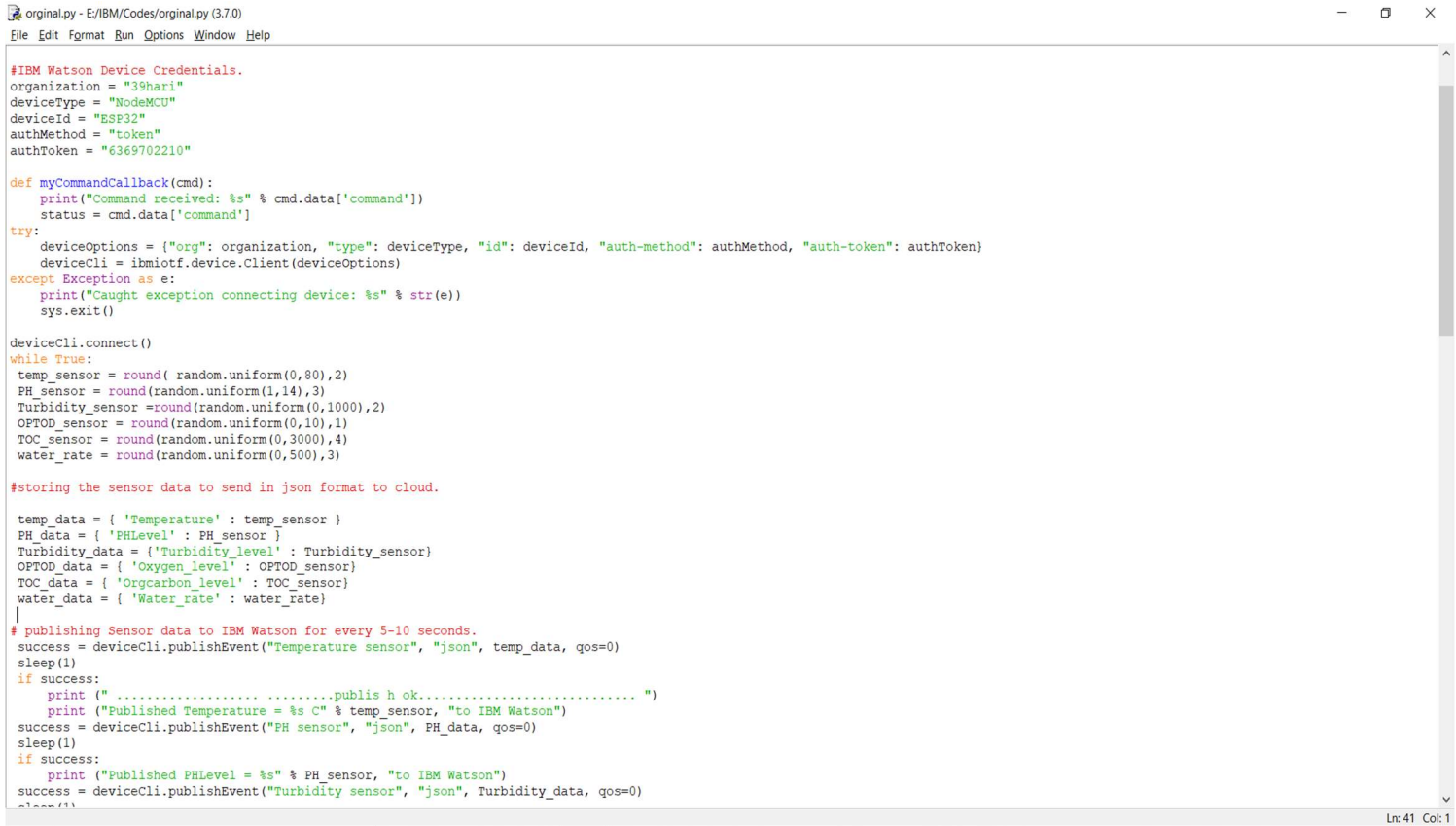
else:

```
    print("Sludge is opened")
    print("")
```

deviceCli.commandCallback = myCommandCallback

Disconnect the device and application from the cloud

```
deviceCli.disconnect()
```

A screenshot of a Python script in a text editor. The script is titled 'original.py - E:/IBM/Codes/original.py (3.7.0)'. It contains code for connecting to IBM Watson IoT, publishing sensor data, and handling alerts. The code includes comments and print statements for debugging. The script is currently at line 41, column 1.

```
#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']

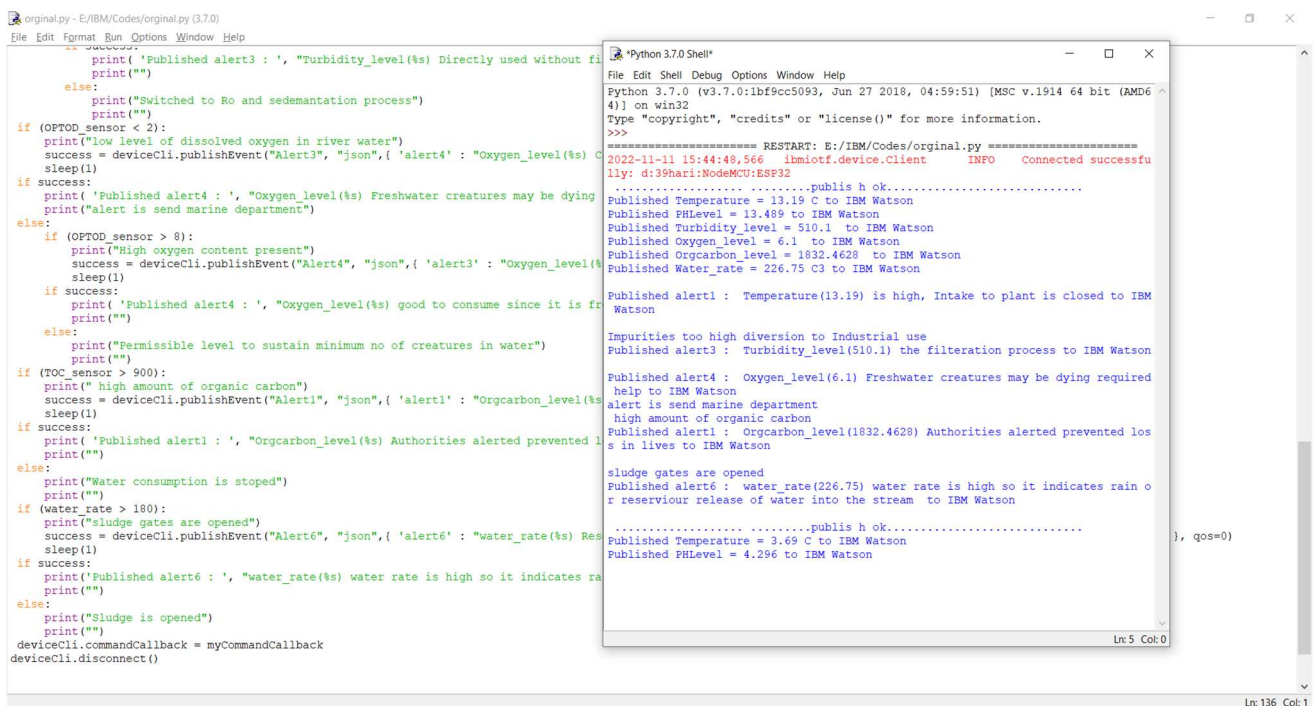
try:
    deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMethod, "auth-token": authToken}
    deviceCli = ibmiotf.device.Client(deviceOptions)
except Exception as e:
    print("Caught exception connecting device: %s" % str(e))
    sys.exit()

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 = { 'Orgcarbon_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)
```

A screenshot showing a Python script in a text editor and its execution output in a terminal window. The script is titled 'original.py - E:/IBM/Codes/original.py (3.7.0)'. The terminal window shows the output of the script, including sensor data and alerts. The script is currently at line 136, column 1.

```
print('Published alert3 : ', "Turbidity_level(%s) Directly used without fi
print("")
else:
    print("Switched to Ro and sedimentation process")
    print("")

if (OPTOD_sensor < 2):
    print("Low level of dissolved oxygen in river water")
    success = deviceCli.publishEvent("Alert3", "json", { 'alert4' : "Oxygen_level(%s)
    sleep(1)
if success:
    print("Published alert4 : ', "Oxygen_level(%s) Freshwater creatures may be dying
    print("alert is send marine department")
else:
    if (OPTOD_sensor > 8):
        print("High oxygen content present")
        success = deviceCli.publishEvent("Alert3", "json", { 'alert3' : "Oxygen_level(%s
        sleep(1)
    if success:
        print('Published alert4 : ', "Oxygen_level(%s) good to consume since it is fr
        print("")
    else:
        print("Permissible level to sustain minimum no of creatures in water")
        print("")
if (TOC_sensor > 900):
    print("high amount of organic carbon")
    success = deviceCli.publishEvent("Alert1", "json", { 'alert1' : "Orgcarbon_level(%s
    sleep(1)
if success:
    print('Published alert1 : ', "Orgcarbon_level(%s) Authorities alerted prevented
    print("")
else:
    print("Water consumption is stoped")
    print("")
if (water_rate > 180):
    print("Sludge gates are opened")
    success = deviceCli.publishEvent("Alert6", "json", { 'alert6' : "water_rate(%s) Res
    sleep(1)
if success:
    print('Published alert6 : ', "water_rate(%s) water rate is high so it indicates ra
    print("")
else:
    print("Sludge is opened")
    print("")
deviceCli.commandCallback = myCommandCallback
deviceCli.disconnect()
```

```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: E:/IBM/Codes/original.py =====
2022-11-11 15:44:48,566 ibmiotf.device.Client INFO Connected successfully: d:39hari:NodeMCU:ESP32
.....publis h ok.....
Published Temperature = 13.19 C to IBM Watson
Published PHLevel = 13.489 to IBM Watson
Published Turbidity_level = 510.1 to IBM Watson
Published Oxygen_level = 6.1 to IBM Watson
Published Orgcarbon_level = 1832.4628 to IBM Watson
Published Water_rate = 226.75 C3 to IBM Watson
Published alert1 : Temperature(13.19) is high, Intake to plant is closed to IBM Watson
Impurities too high diversion to Industrial use
Published alert3 : Turbidity_level(510.1) the filtration process to IBM Watson
Published alert4 : Oxygen_level(6.1) Freshwater creatures may be dying required help to IBM Watson
alert is send marine department
high amount of organic carbon
Published alert1 : Orgcarbon_level(1832.4628) Authorities alerted prevented loss in lives to IBM Watson
sludge gates are opened
Published alert6 : water_rate(226.75) water rate is high so it indicates rain or reservoir release of water into the stream to IBM Watson
.....publis h ok.....
Published Temperature = 3.69 C to IBM Watson
Published PHLevel = 4.296 to IBM Watson
), qos=0
```

The above is our python script and the successful debugging and execution of it.

The next step is to connect it to the an IOT services enabled by the IBM cloud.

```
C:\windows\py.exe
2022-11-11 16:09:49,481 ibmiotf.device.Client INFO Connected successfully: d:39hari:NodeMCU:ESP32
.....publis h ok.....
Published Temperature = 24.63 C to IBM Watson
Published PHLevel = 2.254 to IBM Watson
Published Turbidity_level = 802.25 to IBM Watson
Published Oxygen_level = 9.8 to IBM Watson
Published Orgcarbon_level = 2586.1287 to IBM Watson
Published Water_rate = 282.384 C3 to IBM Watson

Published alert1 : Temperature(24.63) is high, Intake to plant is closed to IBM Watson

Impurities too high diversion to Industrial use
Published alert3 : Turbidity_level(802.25) the filtration process to IBM Watson

Published alert4 : Oxygen_level(9.8) Freshwater creatures may be dying required help to IBM Watson
alert is send marine department
high amount of organic carbon
Published alert1 : Orgcarbon_level(2586.1287) Authorities alerted prevented loss in lives to IBM Watson

sludge gates are opened
Published alert6 : water_rate(282.384) water rate is high so it indicates rain or reservoir release of water into the stream to IBM W

.....publis h ok.....
Published Temperature = 39.01 C to IBM Watson
Published PHLevel = 4.796 to IBM Watson
Published Turbidity_level = 321.48 to IBM Watson
Published Oxygen_level = 2.1 to IBM Watson
Published Orgcarbon_level = 221.4432 to IBM Watson
Published Water_rate = 171.929 C3 to IBM Watson

Intake to Filtration is Closed
```

File Edit View History Bookmarks Tools Help

IBM Watson IoT Platform

https://39hari.internetofthings.ibmcloud.com/dashboard/devices/browse

IBM Watson IoT Platform

Browse Action Device Types Interfaces

Device ID Status Device Type Class ID

ESP32 Connected NodeMCU Device

Identity Device Information Recent Events State Logs

The recent events listed show the live stream of data that is coming and going from this device

Event	Value	Format
Alert1	{"alert1": "Temperature(14.88) is high, Intake is p..."}	json
Flowrate sen...	{"Water_rate": 81.53}	json
TOC sensor	{"Orgcarbon_level": 710.638}	json
optod sensor	{"Oxygen_level": 2.9}	json
Turbidity sen...	{"Turbidity_level": 571.03}	json

Python 3.7.0 Shell

File Edit Shell Debug Options Window Help

===== RESTART: E:/IBM/Codes/origanal.py =====

2022-11-11 15:47:48,477 ibmiotf.device.Client INFO Connected successfully: d:39hari:NodeMCU:ESP32

.....publis h ok.....

Published Temperature = 37.74 C to IBM Watson

Published PHLevel = 4.869 to IBM Watson

Published Turbidity_level = 862.47 to IBM Watson

Published Oxygen_level = 1.0 to IBM Watson

Published Orgcarbon_level = 1686.9192 to IBM Watson

Published Water_rate = 350.297 C3 to IBM Watson

Intake to Filtration is Closed

Published alert1 : Temperature(37.74) is high, Intake to plant is closed to IBM Watson

Impurities too high diversion to Industrial use

Published alert3 : Turbidity_level(862.47) the filtration process to IBM Watson

low level of dissolved oxygen in river water

Published alert4 : Oxygen_level(1.0) Freshwater creatures may be dying required help to IBM Watson

alert is send marine department

high amount of organic carbon

Published alert1 : Orgcarbon_level(1686.9192) Authorities alerted prevented loss in lives to IBM Watson

sludge gates are opened

Published alert6 : water_rate(350.297) water rate is high so it indicates rain or reservoir release of water into the stream to IBM Watson

.....publis h ok.....

Published Temperature = 14.88 C to IBM Watson

Published PHLevel = 11.67 to IBM Watson

Published Turbidity_level = 571.03 to IBM Watson

Published Oxygen_level = 2.9 to IBM Watson

Published Orgcarbon_level = 710.638 to IBM Watson

Published Water_rate = 81.53 C3 to IBM Watson

Ln: 5 Col: 0

0 Simulations running

File Edit View History Bookmarks Tools Help

IBM Watson IoT Platform

https://39hari.internetofthings.ibmcloud.com/dashboard/devices/browse

IBM Watson IoT Platform 830119106034@smartinternz.com ID: 39hari

Browse Action Device Types Interfaces Add Device

Search by Device ID Device Simulator

Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
ESP32	Connected	NodeMCU	Device	Nov 11, 2022 11:59 AM	

Identity Device Information Recent Events State Logs

The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
PH sensor	{"PHLevel":12.566}	json	a few seconds ago
Temperature ...	{"Temperature":56.28}	json	a few seconds ago
Alert1	{"alert1":"Orgcarbon_level(2316.6651) Water n..."}	json	a few seconds ago
Alert3	{"alert3":"Turbidity_level(757.2) Water industrial..."}	json	a few seconds ago
Alert1	{"alert1":"Temperature(28.68)is high, Intake is p..."}	json	a few seconds ago

0 Simulations running

Thus, we have done our sprint 1 part and the remaining of the process will be seen in the following sprints.