# Project Development Phase
# Sprint-1

## Python Script

| | |
|---|---|
| Date | 10 November 2022 |
| Team ID | PNT2022TMID51564 |
| Project Name | IOT BASED CROP PROTECTION SYSTEM FOR AGRICULTURE |

**Description:**

Instead of generating sensor values from the hardware circuits, we are using random module to generate sensor data and to automate IOT based crop protection system through the python code. The data generated from the python code are being stored in the IBM cloud.

**Python Code:**

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

#IBM Watson Device Credentials...
organization = "tw9ckq"
deviceType = "jade"
deviceId = "7010"
authMethod = "token"
authToken = "9944893843"

def myCommandCallback(cmd):
   print("Command received: %s" % cmd.data['command'])
   status=cmd.data['command']
   if status=="sprinkler_on":
      print ("sprinkler is turning ON")
   else :
      print ("sprinkler is turning OFF")

try:
   deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMethod, "auth-token": authToken}
   deviceCli = ibmiotf.device.Client(deviceOptions)

except Exception as e:
```

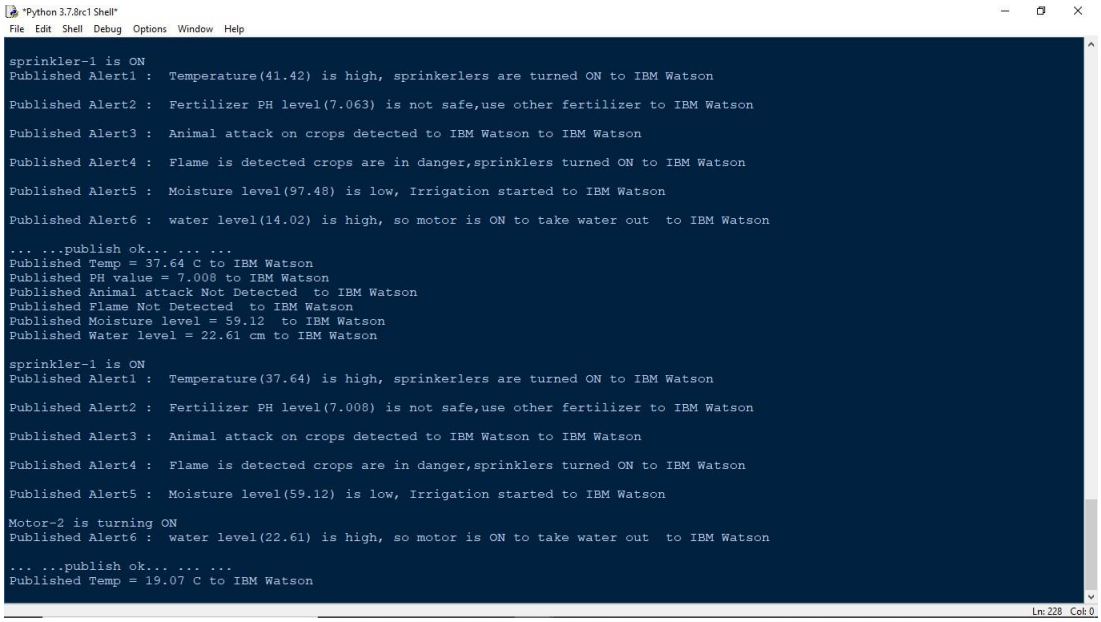
```
    print("Exception detected in connecting device: %s" % str(e))
    sys.exit()

#Connecting to IBM watson...
deviceCli.connect()

while True:
    #Getting values from sensors...
    temp_sensor = round( random.uniform(0,80),2)
    PH_sensor = round(random.uniform(1,14),3)
    camera = ["Detected","Not Detected","Not Detected","Not Detected","Not Detected","Not Detected",]
    camera_reading = random.choice(camera)
    flame = ["Detected","Not Detected","Not Detected","Not Detected","Not Detected","Not Detected",]
    flame_reading = random.choice(flame)
    moist_level = round(random.uniform(0,100),2)
    water_level = round(random.uniform(0,30),2)

    #storing the sensor data to send in json format to cloud…
    temp_data = { 'Temp' : temp_sensor }
    PH_data = { 'PH value' : PH_sensor }
    camera_data = { 'Animal attack' : camera_reading}
    flame_data = { 'Flame' : flame_reading }
    moist_data = { 'Moisture level' : moist_level}
    water_data = { 'Water level' : water_level}

    # publishing Sensor datas to IBM Watson for every 5-10 seconds…
    success = deviceCli.publishEvent("Temperature sensor", "json", temp_data, qos=0)
    sleep(1)

    if success:
        print ("... ...publish ok.....")
        print ("Published Temp = %s C" % temp_sensor, "to IBM Watson")
        success = deviceCli.publishEvent("PH sensor", "json", PH_data, qos=0)
        sleep(1)

    if success:
        print ("Published PH value = %s" % PH_sensor, "to IBM Watson")
        success = deviceCli.publishEvent("camera", "json", camera_data, qos=0)
        sleep(1)

    if success:
        print ("Published Animal attack %s " % camera_reading, "to IBM Watson")
        success = deviceCli.publishEvent("Flame sensor", "json", flame_data, qos=0)
        sleep(1)
```

```
  if success:
    print ("Published Flame %s " % flame_reading,  "to IBM Watson")
    success = deviceCli.publishEvent("Moisture sensor", "json", moist_data, qos=0)
    sleep(1)

  if success:
    print ("Published Moisture level = %s " % moist_level, "to IBM Watson")
    success = deviceCli.publishEvent("Water sensor", "json", water_data, qos=0)
    sleep(1)

  if success:
    print ("Published Water level = %s cm" % water_level,  "to IBM Watson")
    print ("")

  #Automation to control sprinklers by present temperature an to send alert message to IBM Watson…
  if (temp_sensor > 35):
    print("sprinkler-1 is ON")
    success = deviceCli.publishEvent("Alert1", "json",{ 'alert1' : "Temperature(%s) is high, sprinkerlers are
turned ON" %temp_sensor }, qos=0)
    sleep(1)

  if success:
    print( 'Published Alert1 : ',  "Temperature(%s) is high, sprinkerlers are turned ON" %temp_sensor,"to IBM
Watson")
    print("")
  else:
    print("sprinkler-1 is OFF")
    print("")

   #To send alert message if farmer uses the unsafe fertilizer to crops…

  if (PH_sensor > 7.5 or PH_sensor < 5.5):
    success = deviceCli.publishEvent("Alert2", "json",{ 'alert2' : "Fertilizer PH level(%s) is not safe,use other
fertilizer" %PH_sensor } , qos=0)
    sleep(1)

  if success:
    print('Published Alert2 : ' , "Fertilizer PH level(%s) is not safe,use other fertilizer" %PH_sensor,"to IBM
Watson")
    print("")

   #To send alert message to farmer that animal attack on crops…
  if (camera_reading == "Detected"):
    success = deviceCli.publishEvent("Alert3", "json", { 'alert3' : "Animal attack on crops detected" }, qos=0)
    sleep(1)
```

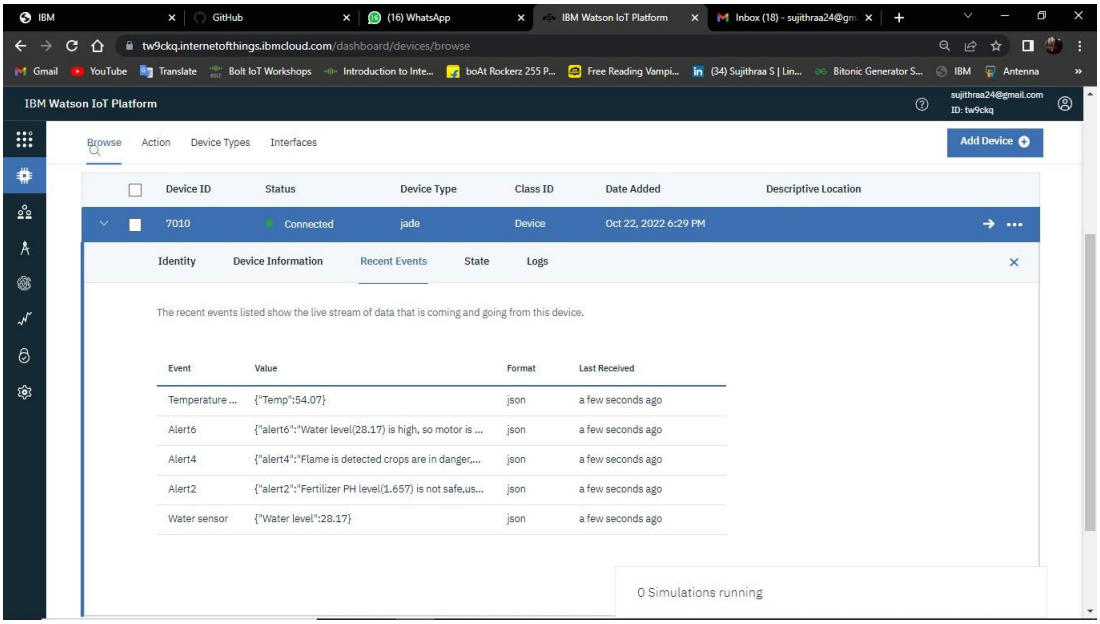
```
  if success:
    print('Published Alert3 : ' , "Animal attack on crops detected","to IBM Watson","to IBM Watson")
    print("")

  #To send alert message if flame detected on crop land and turn ON the splinkers to take immediate action…
  if (flame_reading == "Detected"):
    print("sprinkler-2 is ON")
    success = deviceCli.publishEvent("Alert4", "json", { 'alert4' : "Flame is detected crops are in
danger,sprinklers turned ON" }, qos=0)
    sleep(1)

  if success:
    print( 'Published Alert4 : ' , "Flame is detected crops are in danger,sprinklers turned ON","to IBM Watson")
    print("")

  else:
    print("sprinkler-2 is OFF")
    print("")

  #To send alert message if Moisture level is LOW and to Turn ON Motor-1 for irrigation…
  if (moist_level < 20):
    print("Motor-1 is ON")
    success = deviceCli.publishEvent("Alert5", "json", { 'alert5' : "Moisture level(%s) is low, Irrigation started"
%moist_level }, qos=0)
    sleep(1)

  if success:
    print('Published Alert5 : ' , "Moisture level(%s) is low, Irrigation started" %moist_level,"to IBM Watson" )
    print("")
  else:
    print("Motor-1 is OFF")
    print("")

  #To send alert message if Water level is HIGH and to Turn ON Motor-2 to take water out…
  if (water_level > 20):
    print("Motor-2 is turning ON")
    success = deviceCli.publishEvent("Alert6", "json", { 'alert6' : "Water level(%s) is high, so motor is ON to take
water out "  %water_level }, qos=0)
    sleep(1)

  if success:
    print('Published Alert6 : ' , "water level(%s) is high, so motor is ON to take water out " %water_level,"to
IBM Watson" )
    print("")
```

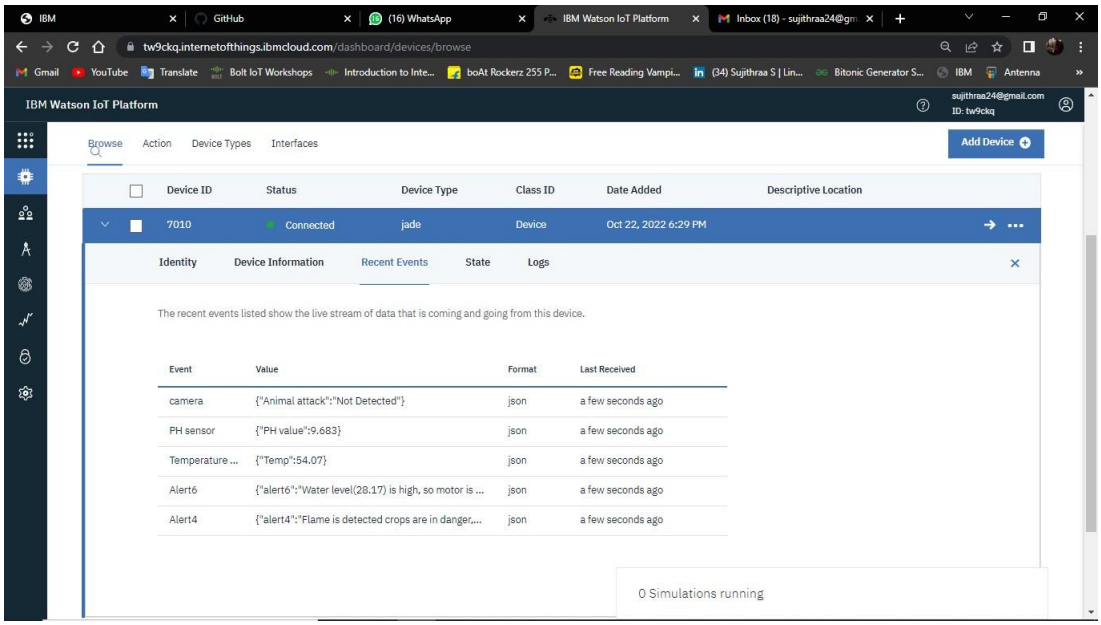
```
    else:
        print("Motor-2 is turning OFF")
        print("")

#command recived by farmer
deviceCli.commandCallback = myCommandCallback

# Disconnect the device and application from the cloud
deviceCli.disconnect()
```

## Python Script Output:




## IBM Watson Output:


IBM Watson IoT Platform
Browse
Action
Device Types
Interfaces
Add Device
Device ID
Status
Device Type
Class ID
Date Added
Descriptive Location
7010
Connected
jade
Device
Oct 22, 2022 6:29 PM
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
Temperature ...
{"Temp":54.07}
json
a few seconds ago
Alert6
{"alert6":"Water level(28.17) is high, so motor is ...
json
a few seconds ago
Alert4
{"alert4":"Flame is detected crops are in danger,...
json
a few seconds ago
Alert2
{"alert2":"Fertilizer PH level(1.657) is not safe,us...
json
a few seconds ago
Water sensor
{"Water level":28.17}
json
a few seconds ago
0 Simulations running


IBM Watson IoT Platform
Browse
Action
Device Types
Interfaces
Add Device
Device ID
Status
Device Type
Class ID
Date Added
Descriptive Location
7010
Connected
jade
Device
Oct 22, 2022 6:29 PM
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
camera
{"Animal attack":"Not Detected"}
json
a few seconds ago
PH sensor
{"PH value":9.683}
json
a few seconds ago
Temperature ...
{"Temp":54.07}
json
a few seconds ago
Alert6
{"alert6":"Water level(28.17) is high, so motor is ...
json
a few seconds ago
Alert4
{"alert4":"Flame is detected crops are in danger,...
json
a few seconds ago
0 Simulations running