SPRINT-2

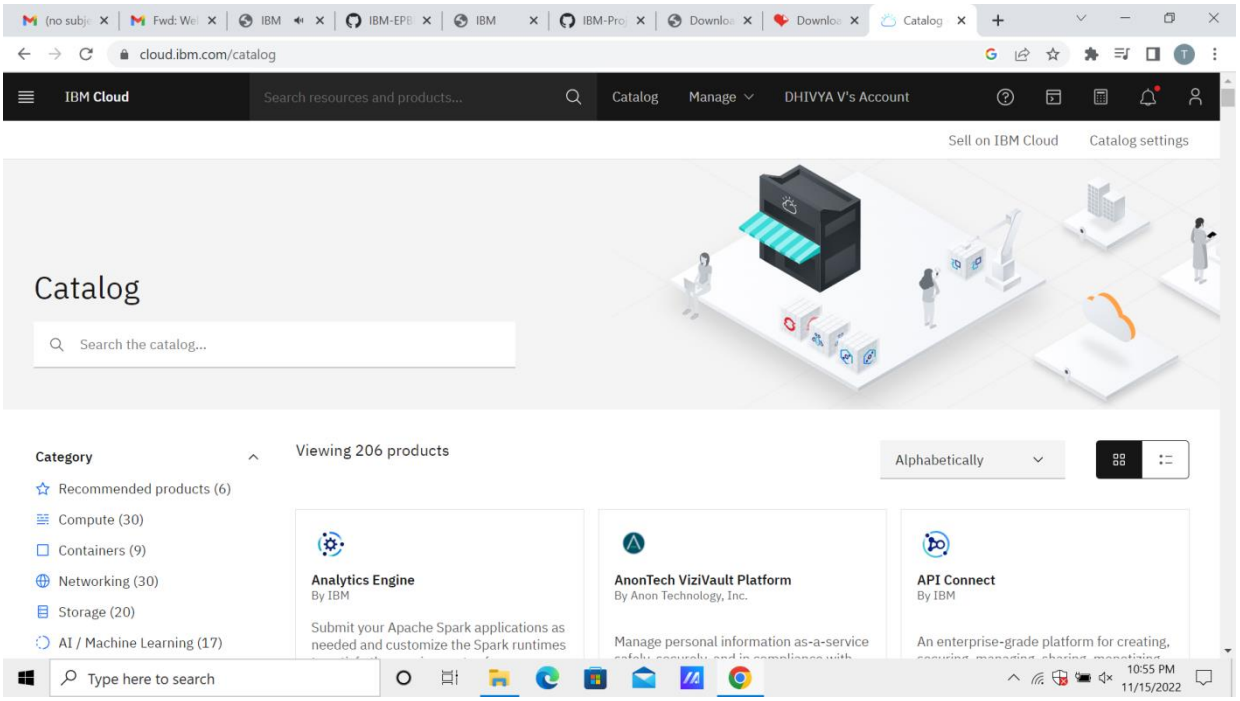
TEAM ID: PNT2022TMID18899

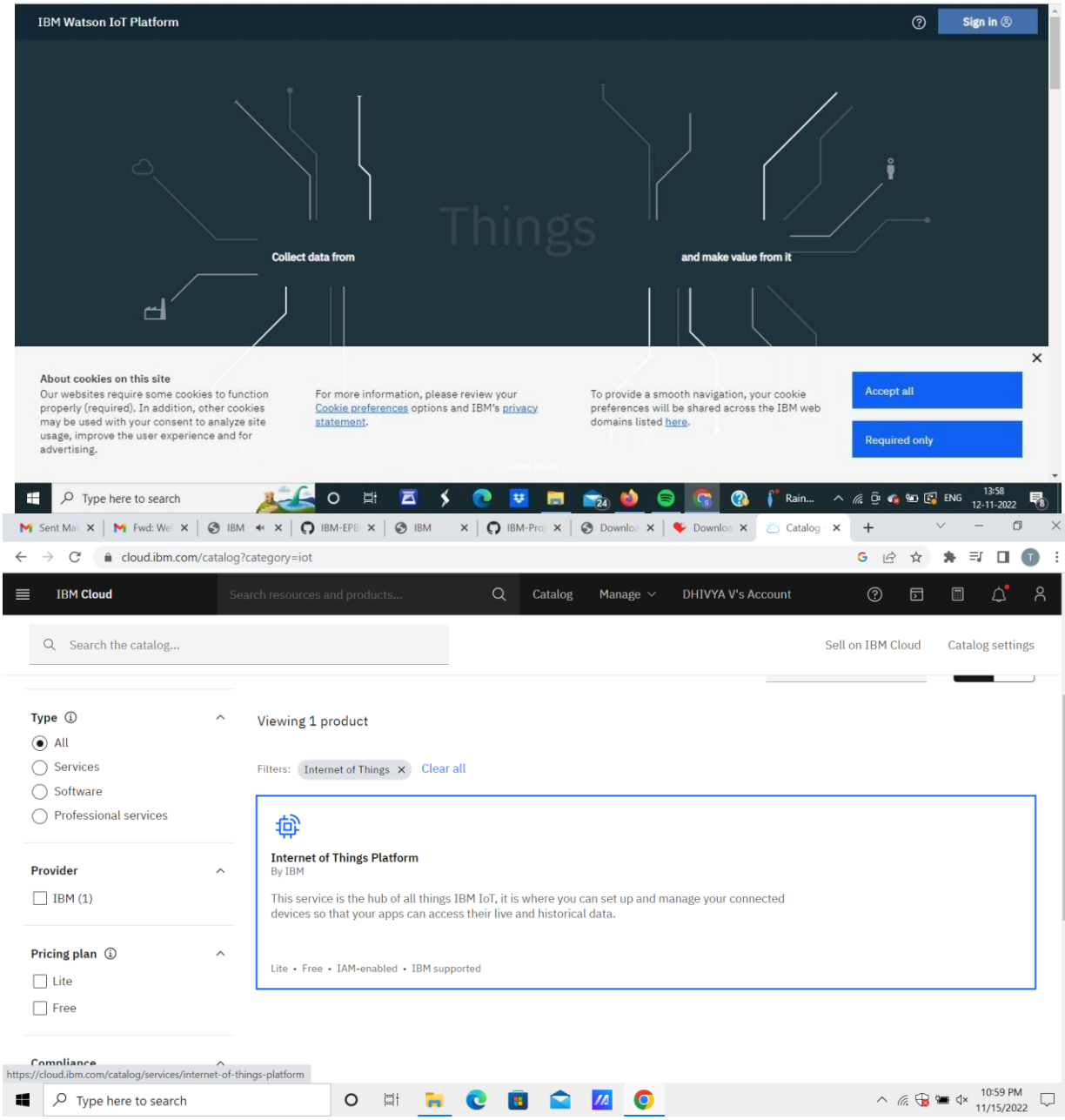
PROJECT NAME : IoT based smart crop protection system for Agriculture

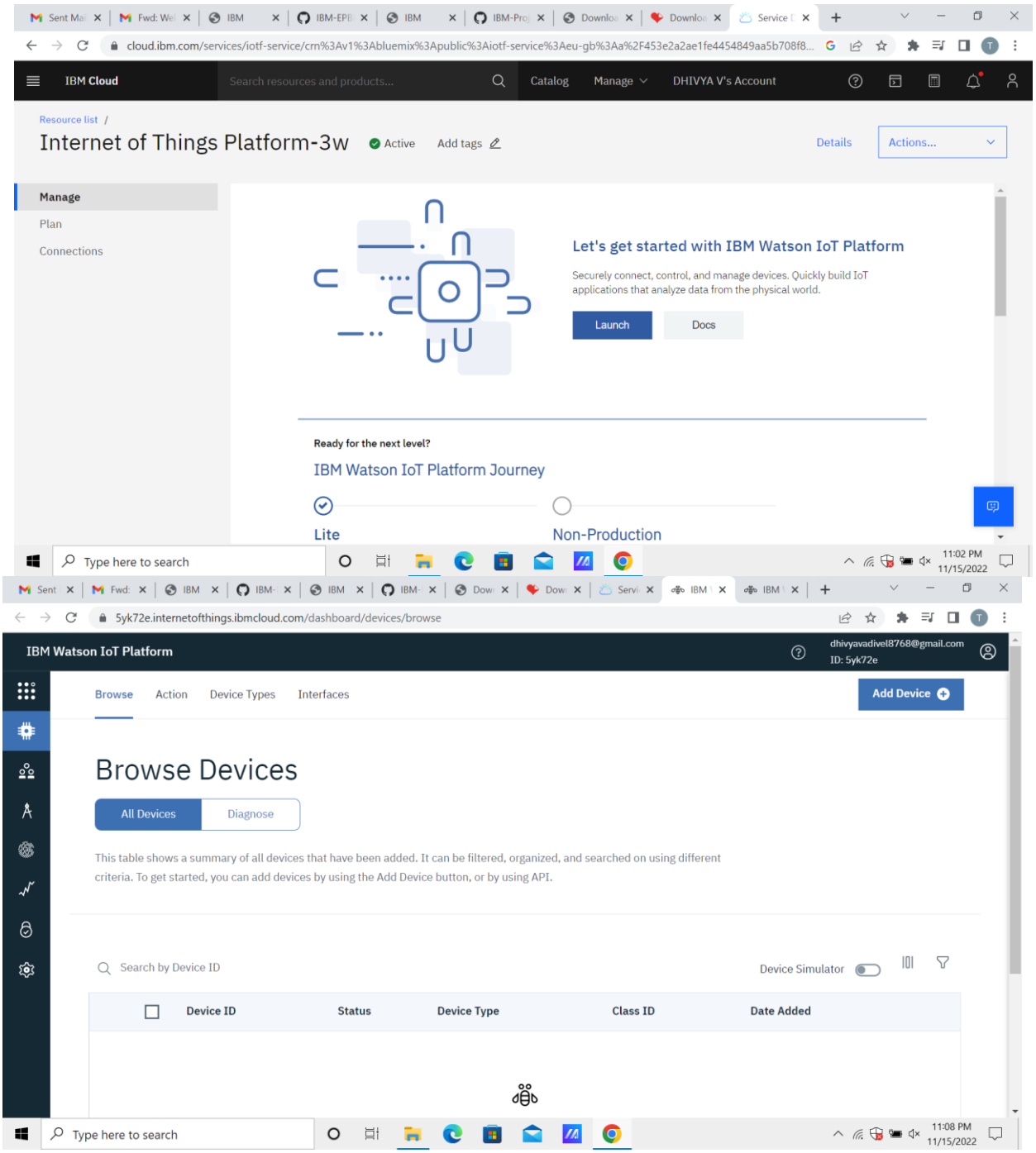
STEPS:

The random sensor data's are generated and automation has been implemented through the python code instead of using hardware to implement IOT based crop protection system. And the python code need to upload the data's in IBM cloud are written in this python script.

CREATION OF IBM WATSON CLOUD PLATFROM:




IBM Watson IoT Platform
Sign in
Things
Collect data from
and make value from it
About cookies on this site
Our websites require some cookies to function properly (required). In addition, other cookies may be used with your consent to analyze site usage, improve the user experience and for advertising.
For more information, please review your Cookie preferences options and IBM's privacy statement.
To provide a smooth navigation, your cookie preferences will be shared across the IBM web domains listed here.
Accept all
Required only
Type here to search
cloud.ibm.com/catalog?category=iot
IBM Cloud
Search resources and products...
Catalog
Manage
DHIVYA V's Account
Search the catalog...
Sell on IBM Cloud
Catalog settings
Type
All
Services
Software
Professional services
Provider
IBM (1)
Pricing plan
Lite
Free
Compliance
Viewing 1 product
Filters:
Internet of Things
Clear all
Internet of Things Platform
By IBM
This service is the hub of all things IBM IoT, it is where you can set up and manage your connected devices so that your apps can access their live and historical data.
Lite • Free • IAM-enabled • IBM supported
https://cloud.ibm.com/catalog/services/internet-of-things-platform
Type here to search
10:59 PM
11/15/2022



CODE:

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

```
import sys
#IBM Watson Device Credentials.
organization = "3xaptt"
deviceType = "NodeMCU"
deviceId = "12345"
authMethod = "use-token-auth"
authToken = "12345678"
def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status=cmd.data['command']
    if status=="sprinkler_on":
        print ("sprinkler is ON")
    else :
        print ("sprinkler is OFF")
#print(cmd)



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:
#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 = { 'Temperature' : temp_sensor }
 PH_data = { 'PH Level' : 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 data 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 Temperature = %s C" % temp_sensor, "to IBM Watson")

 success = deviceCli.publishEvent("PH sensor", "json", PH_data, qos=0)
 sleep(1)
 if success:
   print ("Published PH Level = %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.
```

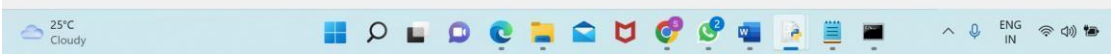
```
 if (temp_sensor > 35):
    print("sprinkler-1 is ON")
 success = deviceCli.publishEvent("Alert1", "json",{ 'alert1' :
"Temperature(%s) is high, sprinkerlers are turned ON"
%temp_sensor }
, qos=0)
 sleep(1)
 if success:
   print( 'Published alert1 : ', "Temperature(%s) is high,
sprinkerlers are turned 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)
```

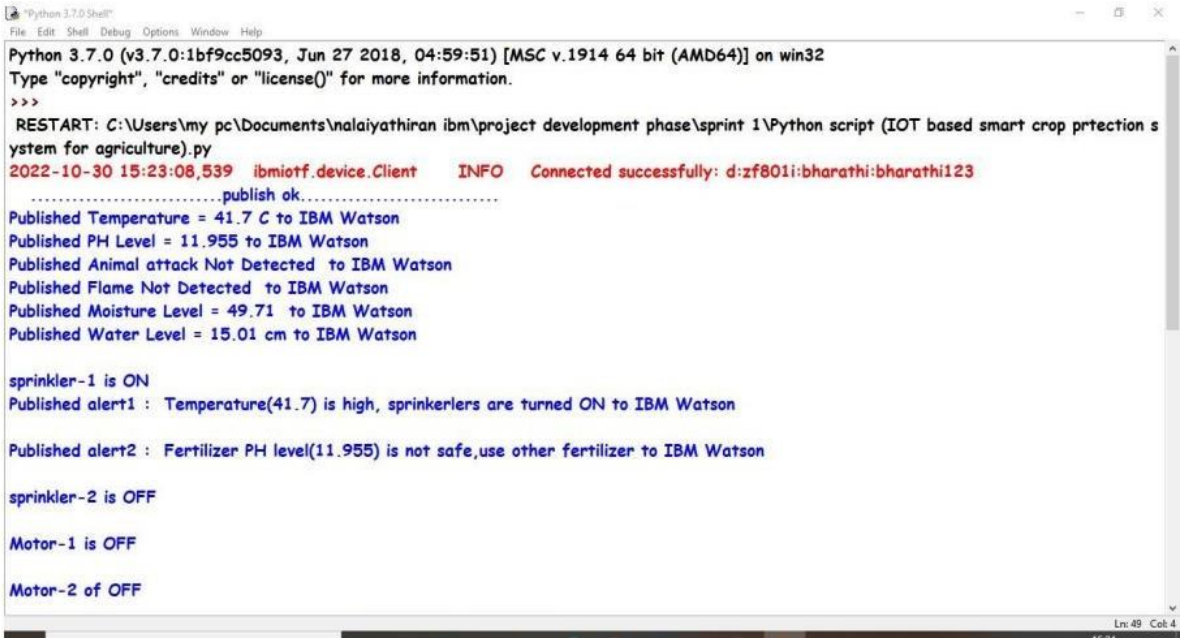
```
    sleep(1)
 if success:
    print( 'Published alert4 : ' , "Flame is detected crops are in danger,sprinklers turned ON","to IBM Watson")
    print("")
 else:
    print("sprinkler-2 is OFF")
    print("")
 #To send alert message if Moisture level is LOW and to Turn ON Motor-1 for irrigation.
 if (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("")
```

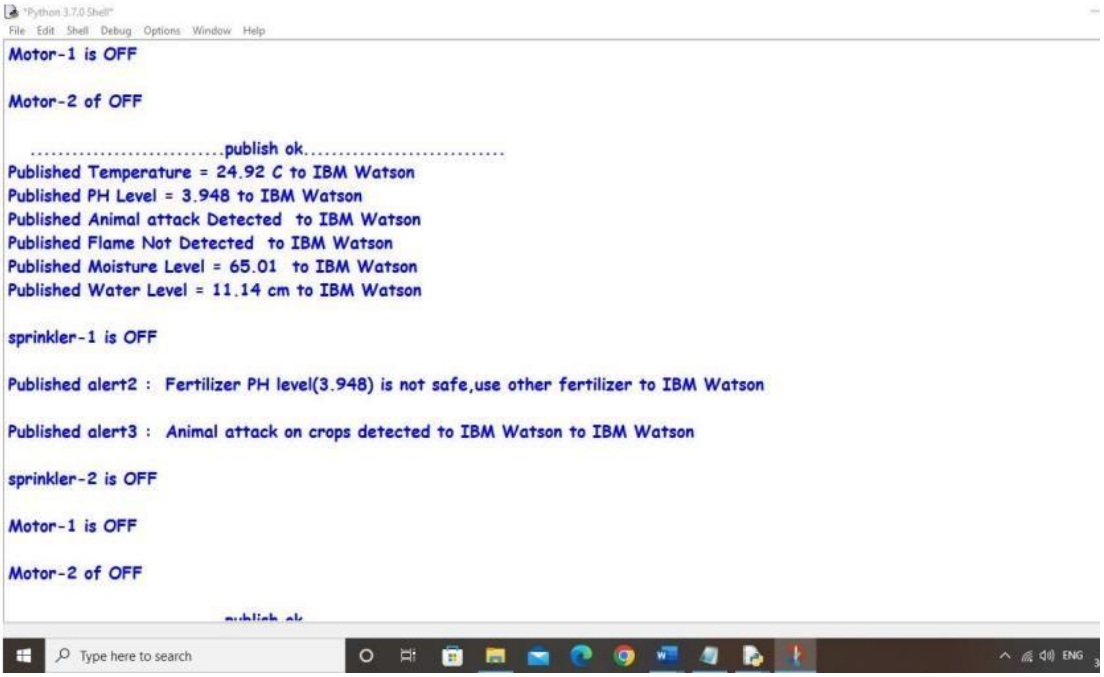
```
 #To send alert message if Water level is HIGH and to Turn ON Motor-2 to take water out.
if (water_level > 20):
    print("Motor-2 is ON")
    success = deviceCli.publishEvent("Alert6", "json", { 'alert6' : "Water level(%s) is high, so motor is ON to take water out "
%water_level }, qos=0)
    sleep(1)
    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 of OFF")
            print("")
 #command recived by farmer
deviceCli.commandCallback = myCommandCallback
# Disconnect the device and application from the cloud
deviceCli.disconnect()
```

```
*ibm ss.py - C:/Users/Sakthi Sneghaa/Desktop/ibm ss.py (3.11.0)*
File  Edit  Format  Run  Options  Window  Help
      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 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 of OFF")
          print("")
 #command recived by farmer
deviceCli.commandCallback = myCommandCallback
# Disconnect the device and application from the cloud
deviceCli.disconnect()
```



OUTPUT




"Python 3.7.0 Shell"
File Edit Shell Debug Options Window Help
Motor-1 is OFF
Motor-2 of OFF
.....publish ok.....
Published Temperature = 24.92 C to IBM Watson
Published PH Level = 3.948 to IBM Watson
Published Animal attack Detected to IBM Watson
Published Flame Not Detected to IBM Watson
Published Moisture Level = 65.01 to IBM Watson
Published Water Level = 11.14 cm to IBM Watson
sprinkler-1 is OFF
Published alert2 : Fertilizer PH level(3.948) is not safe,use other fertilizer to IBM Watson
Published alert3 : Animal attack on crops detected to IBM Watson to IBM Watson
sprinkler-2 is OFF
Motor-1 is OFF
Motor-2 of OFF
Type here to search
ENG