

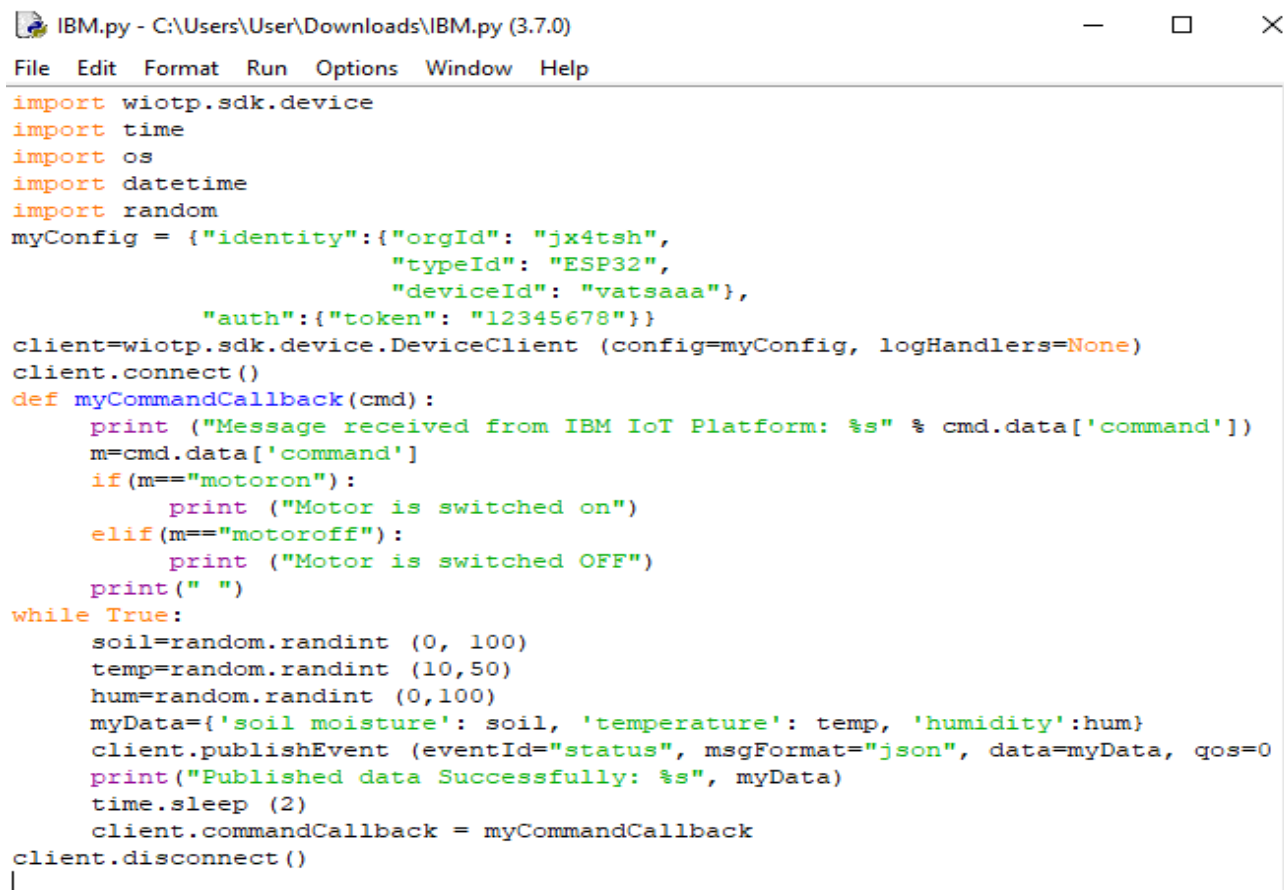
# Smart Farmer - IoT Enabled Smart Farming Application

## SPRINT-2

Team ID	PNT2022TMID35924
Project name	Smart farmer- IoT enabled smart farming

In this session, Python code has been written to replicate the sensors which are installed in the farming field. The output of the Python code has been transferred to the IBM cloud. The code along with the output has been attached below.

### Code:



```
IBM.py - C:\Users\User\Downloads\IBM.py (3.7.0)
File Edit Format Run Options Window Help
import wiotp.sdk.device
import time
import os
import datetime
import random
myConfig = {"identity":{"orgId": "jx4tsh",
                        "typeId": "ESP32",
                        "deviceId": "vatsaaa"},
            "auth":{"token": "12345678"}}
client=wiotp.sdk.device.DeviceClient (config=myConfig, logHandlers=None)
client.connect ()
def myCommandCallback (cmd):
    print ("Message received from IBM IoT Platform: %s" % cmd.data['command'])
    m=cmd.data['command']
    if(m=="motoron"):
        print ("Motor is switched on")
    elif(m=="motoroff"):
        print ("Motor is switched OFF")
    print(" ")
while True:
    soil=random.randint (0, 100)
    temp=random.randint (10,50)
    hum=random.randint (0,100)
    myData={'soil moisture': soil, 'temperature': temp, 'humidity':hum}
    client.publishEvent (eventId="status", msgFormat="json", data=myData, qos=0)
    print("Published data Successfully: %s", myData)
    time.sleep (2)
    client.commandCallback = myCommandCallback
client.disconnect ()
```

## Output:

### IBM cloud window:

The screenshot displays the IBM Watson IoT Platform interface. The top navigation bar includes 'Browse', 'Action', 'Device Types', and 'Interfaces'. The main content area shows details for a device named 'vatsaaa', which is 'Connected' and an 'ESP32' device. The 'Recent Events' tab is selected, showing a table of events.

Event	Value	Format	Last Received
status	{"soil moisture":99,"temperature":19,"humidity":...	json	a few seconds ago
status	{"soil moisture":48,"temperature":42,"humidity":...	json	a few seconds ago
status	{"soil moisture":67,"temperature":28,"humidity":...	json	a few seconds ago
status	{"soil moisture":66,"temperature":45,"humidity":...	json	a few seconds ago

The screenshot shows a Python 3.7.0 Shell window with the following output:

```
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: C:\Users\User\Downloads\IBM.py =====
2022-11-21 21:14:58,308 wiotp.sdk.device.client.DeviceClient INFO Connected successfully: d:jx4tsh:ESP32:vatsaaaPublished data Successfully: %s {'soil moisture': 66, 'temperature': 45, 'humidity': 15}
Published data Successfully: %s {'soil moisture': 67, 'temperature': 28, 'humidity': 86}
Published data Successfully: %s {'soil moisture': 48, 'temperature': 42, 'humidity': 64}
Published data Successfully: %s {'soil moisture': 99, 'temperature': 19, 'humidity': 59}
Published data Successfully: %s {'soil moisture': 97, 'temperature': 31, 'humidity': 59}
Published data Successfully: %s {'soil moisture': 86, 'temperature': 23, 'humidity': 98}
Published data Successfully: %s {'soil moisture': 26, 'temperature': 28, 'humidity': 66}
```

**Sprint 2 video duration:**

**Video 1:** 4 minutes 31 seconds

**Video 2:** 7 minutes 52 seconds

**Difficulties faced:**

- ❖ Tinker cad software is not suitable for our project as we can't connect Tinker cad with IBM cloud.
- ❖ Wokwi software is also not suitable for implementing our project as Wokwi doesn't have all the required sensors.