Project Development Phase Delivery of Sprint -1

Team ID	PNT2022TMID17466
Project Name	Smart Farmer-IOT Enabled Smart FarmingApplication

In Sprint-1 we are going to develop the python code and Wokwi Online ESP32 Simulator and connecting to IBM Watson Platform

1. Introduction

The main aim of this project is to help farmers automate their farms by providing them with a Web App through which they can monitor the parameters of the field like Temperature, soil moisture, humidity and etc .And control the equipment like water motor and other devices remotely via internet without their actual presence in the field.

2. Problem Statement

Farmers are to be present at farm for its maintenance irrespective of the weather conditions. They have to ensure that the crops are well watered and the farm status is monitored by them physically. Farmer have to stay most of the time in field in order to get a good yield. In difficult times like in the presence of pandemic also they have to work hard in their fields risking their lives to provide food for the country.

3. Proposed Solution

In order to improve the farmer's working conditions and make them easier, we introduce IoT services to him in which we use cloud services and internet to enable farmer to continue his work remotely via internet. He can monitor the field parameters and control the devices in farm.

4 . Software Requirements

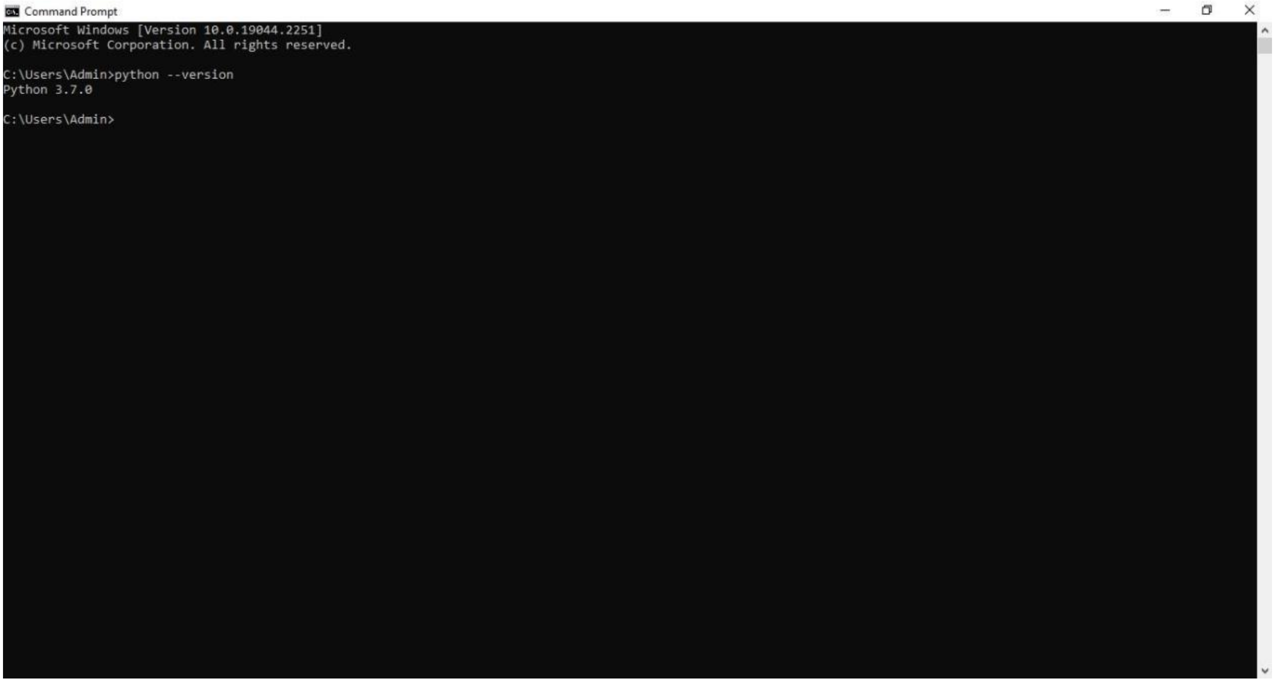
1.Python IDLE 3.7.0 (64-Bit)

2.IBM Watson Platform

3.IBM Node-Red

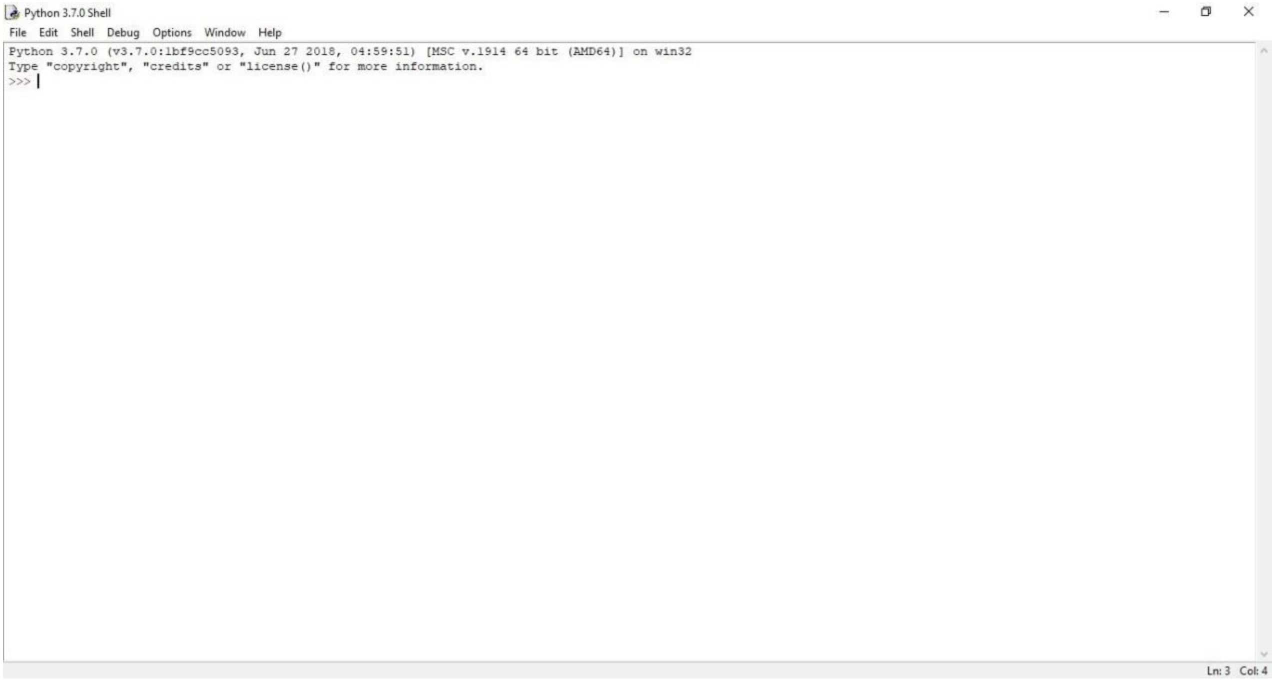
4. MIT App Inventor

First install the python 3.7.0 version idle . Go to command prompt and type python –version we can get version.



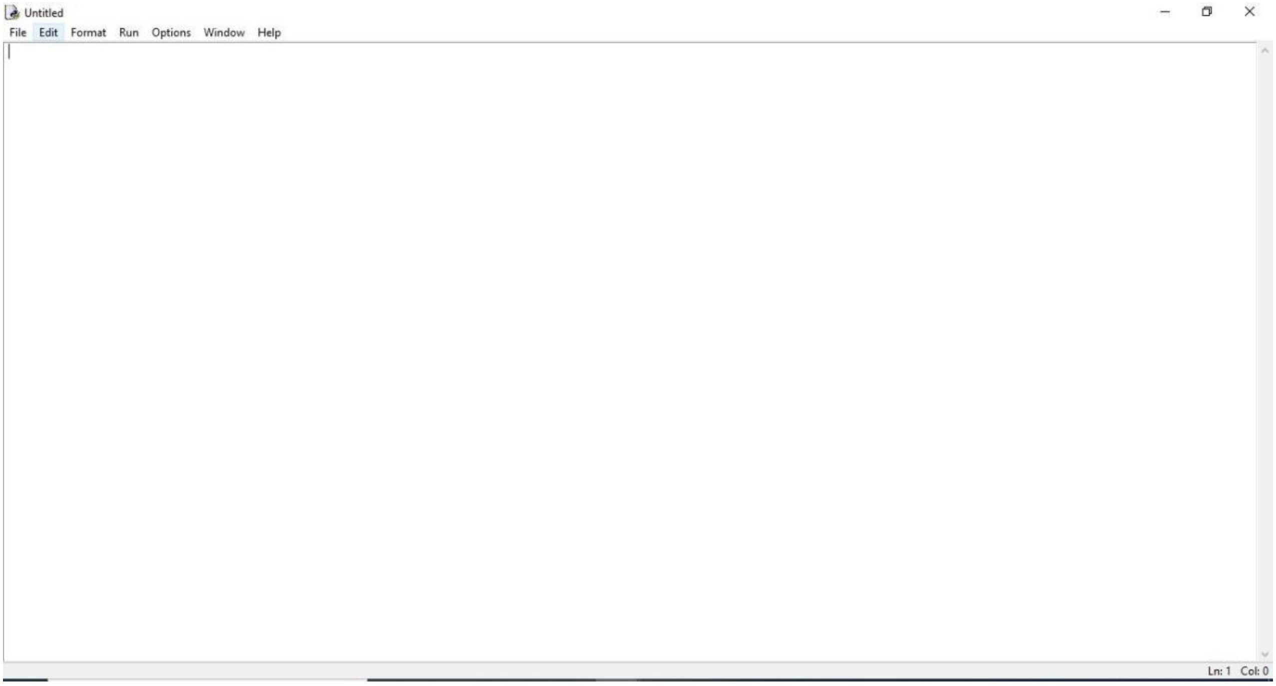
```
Microsoft Windows [Version 10.0.19044.2251]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Admin>python --version
Python 3.7.0

C:\Users\Admin>
```

After that open python idle we can see python shell.



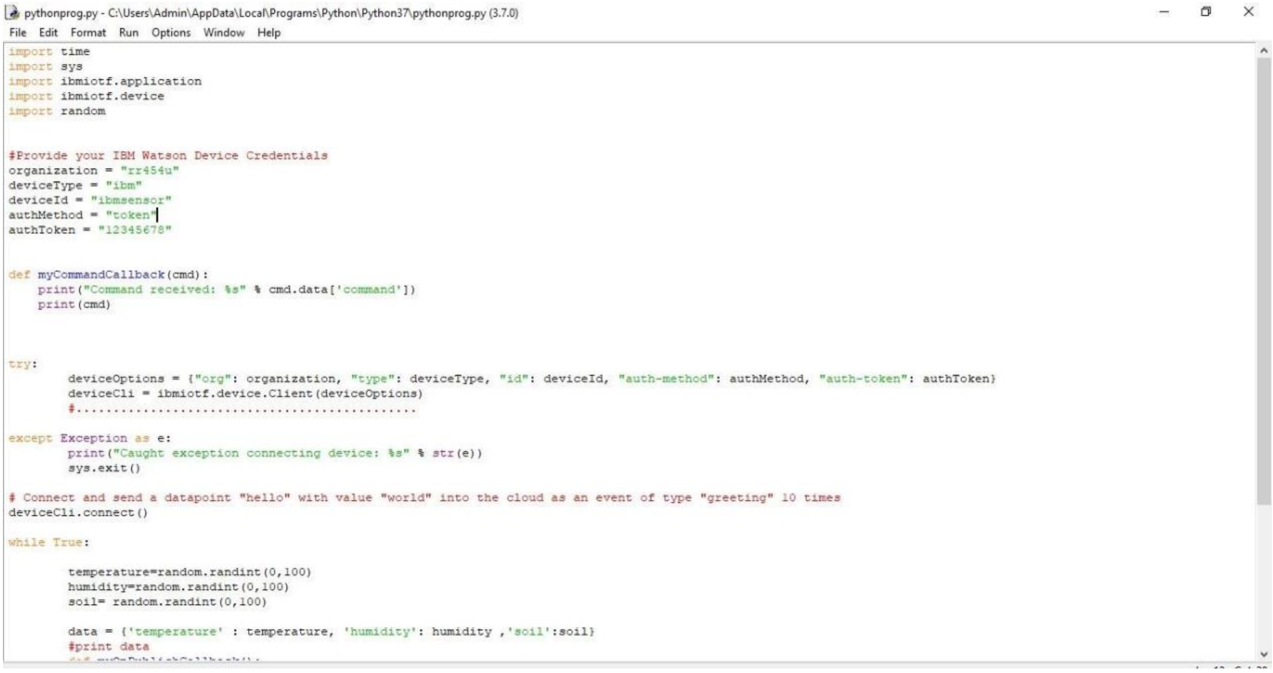
```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> 
```

Click on file and open new file the window appear as shown below.



Before writing the python script we have install pip ibmiotf install. After that we have to write the python code.

Python code to connect the IBM Watson platform

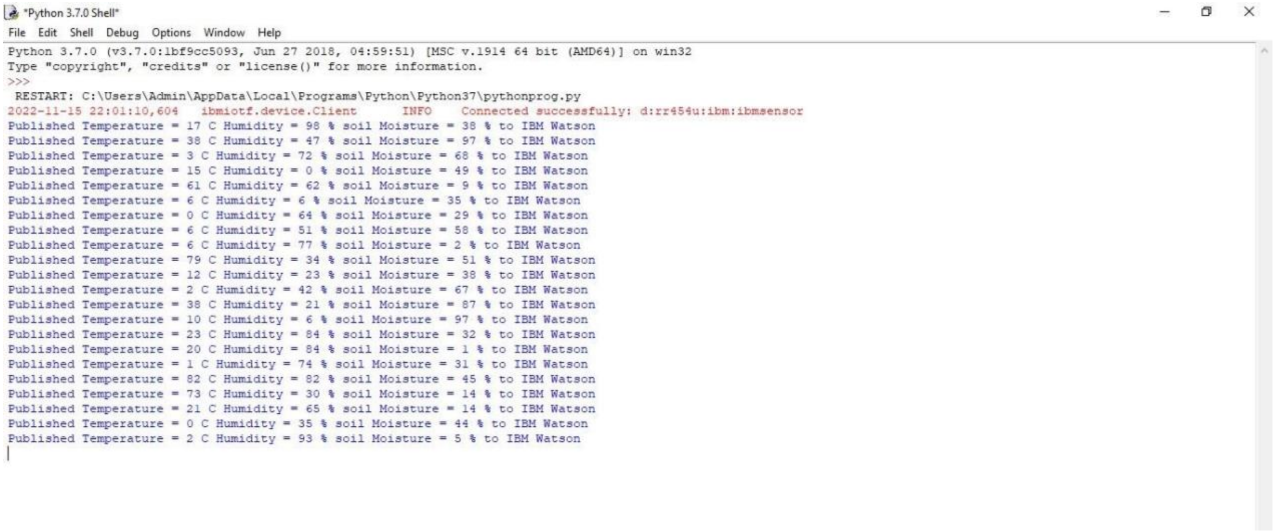

pythonprog.py - C:\Users\Admin\AppData\Local\Programs\Python\Python37\pythonprog.py (3.7.0)
File
Edit
Format
Run
Options
Window
Help

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


#Provide your IBM Watson Device Credentials
organization = "rr454u"
deviceType = "ibm"
deviceId = "ibmsensor"
authMethod = "token"
authToken = "12345678"


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

# Connect and send a datapoint "hello" with value "world" into the cloud as an event of type "greeting" 10 times
deviceCli.connect()

while True:

        temperature=random.randint(0,100)
        humidity=random.randint(0,100)
        soil= random.randint(0,100)

        data = {'temperature' : temperature, 'humidity': humidity ,'soil':soil}
        #print data
```

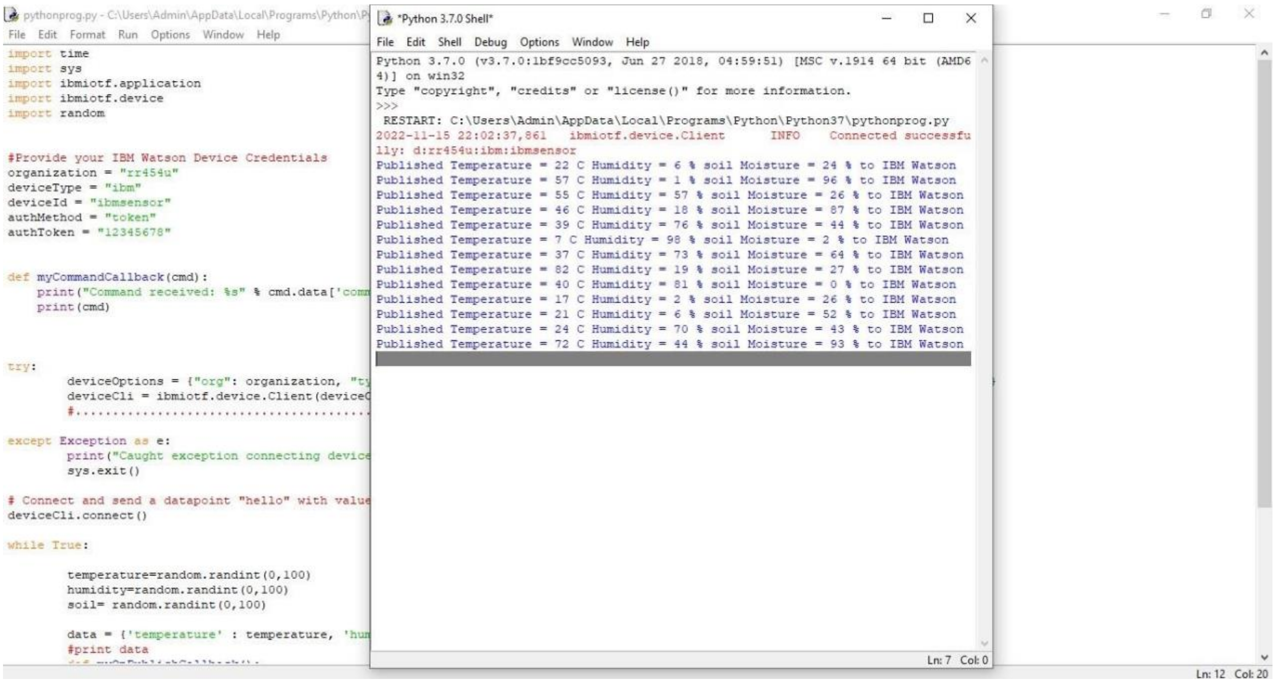
CODE:

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


#Provide your IBM Watson Device Credentials organization
= "9lglg1 " deviceType = "
Arduino " deviceId = "1234567" authMethod
= "token" authToken = "123456789 "   def
myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
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()

# Connect and send a datapoint "hello" with value "world" into
the cloud as an event of type "greeting" 10 times
deviceCli.connect()  while True:
        temperature=random.randint(0,100)
humidity=random.randint(0,100)         soil=
random.randint(0,100)
         data = {'temperature' : temperature,
'humidity': humidity ,'soil':soil}         #print data
def myOnPublishCallback():
            print ("Published Temperature = %s C" % temperature,
"Humidity = %s %%" % humidity, "soil Moisture = %s %%"% soil,"to
IBM Watson")

        success = deviceCli.publishEvent("IoTSensor", "json",
data, qos=0, on_publish=myOnPublishCallback)         if not
success:             print("Not connected to IoTF")
time.sleep(1)                   deviceCli.commandCallback =
myCommandCallback

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

Simulation output in the python idle:



```
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\Admin\AppData\Local\Programs\Python\Python37\pythonprog.py
2022-11-15 22:01:10,604   ibmiotf.device.Client      INFO    Connected successfully: d:rr454u:ibm:ibmsensor
Published Temperature = 17 C Humidity = 98 % soil Moisture = 38 % to IBM Watson
Published Temperature = 38 C Humidity = 47 % soil Moisture = 97 % to IBM Watson
Published Temperature = 3 C Humidity = 72 % soil Moisture = 68 % to IBM Watson
Published Temperature = 15 C Humidity = 0 % soil Moisture = 49 % to IBM Watson
Published Temperature = 61 C Humidity = 62 % soil Moisture = 9 % to IBM Watson
Published Temperature = 6 C Humidity = 6 % soil Moisture = 35 % to IBM Watson
Published Temperature = 0 C Humidity = 64 % soil Moisture = 29 % to IBM Watson
Published Temperature = 6 C Humidity = 51 % soil Moisture = 58 % to IBM Watson
Published Temperature = 6 C Humidity = 77 % soil Moisture = 2 % to IBM Watson
Published Temperature = 79 C Humidity = 34 % soil Moisture = 51 % to IBM Watson
Published Temperature = 12 C Humidity = 23 % soil Moisture = 38 % to IBM Watson
Published Temperature = 2 C Humidity = 42 % soil Moisture = 67 % to IBM Watson
Published Temperature = 38 C Humidity = 21 % soil Moisture = 87 % to IBM Watson
Published Temperature = 10 C Humidity = 6 % soil Moisture = 97 % to IBM Watson
Published Temperature = 23 C Humidity = 84 % soil Moisture = 32 % to IBM Watson
Published Temperature = 20 C Humidity = 84 % soil Moisture = 1 % to IBM Watson
Published Temperature = 1 C Humidity = 74 % soil Moisture = 31 % to IBM Watson
Published Temperature = 82 C Humidity = 82 % soil Moisture = 45 % to IBM Watson
Published Temperature = 73 C Humidity = 30 % soil Moisture = 14 % to IBM Watson
Published Temperature = 21 C Humidity = 65 % soil Moisture = 14 % to IBM Watson
Published Temperature = 0 C Humidity = 35 % soil Moisture = 44 % to IBM Watson
Published Temperature = 2 C Humidity = 93 % soil Moisture = 5 % to IBM Watson
```

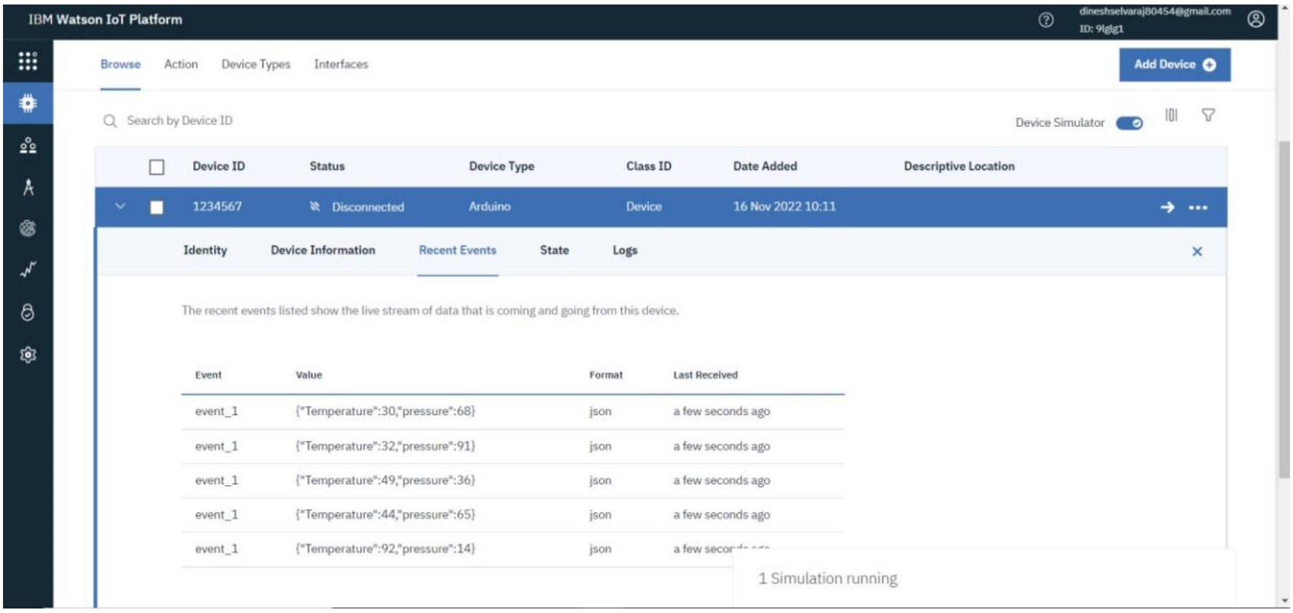


pythonprog.py - C:\Users\Admin\AppData\Local\Programs\Python\P
File Edit Format Run Options Window Help
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
Ln: 7 Col: 0
Ln: 12 Col: 20

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


#Provide your IBM Watson Device Credentials
organization = "rr454u"
deviceType = "ibm"
deviceId = "ibmsensor"
authMethod = "token"
authToken = "12345678"


def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['comm
    print(cmd)



try:
        deviceOptions = {"org": organization, "ty
        deviceCli = ibmiotf.device.Client(deviceC
        #.....................................

except Exception as e:
        print("Caught exception connecting device
        sys.exit()

# Connect and send a datapoint "hello" with value
deviceCli.connect()

while True:

        temperature=random.randint(0,100)
        humidity=random.randint(0,100)
        soil= random.randint(0,100)

        data = {'temperature' : temperature, 'hum
        #print data
```

```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD6
4)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
 RESTART: C:\Users\Admin\AppData\Local\Programs\Python\Python37\pythonprog.py
2022-11-15 22:02:37,861   ibmiotf.device.Client      INFO    Connected successfu
lly: d:rr454u:ibm:ibmsensor
Published Temperature = 22 C Humidity = 6 % soil Moisture = 24 % to IBM Watson
Published Temperature = 57 C Humidity = 1 % soil Moisture = 96 % to IBM Watson
Published Temperature = 55 C Humidity = 57 % soil Moisture = 26 % to IBM Watson
Published Temperature = 46 C Humidity = 18 % soil Moisture = 87 % to IBM Watson
Published Temperature = 39 C Humidity = 76 % soil Moisture = 44 % to IBM Watson
Published Temperature = 7 C Humidity = 98 % soil Moisture = 2 % to IBM Watson
Published Temperature = 37 C Humidity = 73 % soil Moisture = 64 % to IBM Watson
Published Temperature = 82 C Humidity = 19 % soil Moisture = 27 % to IBM Watson
Published Temperature = 40 C Humidity = 81 % soil Moisture = 0 % to IBM Watson
Published Temperature = 17 C Humidity = 2 % soil Moisture = 26 % to IBM Watson
Published Temperature = 21 C Humidity = 6 % soil Moisture = 52 % to IBM Watson
Published Temperature = 24 C Humidity = 70 % soil Moisture = 43 % to IBM Watson
Published Temperature = 72 C Humidity = 44 % soil Moisture = 93 % to IBM Watson
```

Python output Showing in IBM Watson platform:



□	Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
□	1234567	Disconnected	Arduino	Device	16 Nov 2022 10:11	

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

Event	Value	Format	Last Received
event_1	{"Temperature":30,"pressure":68}	json	a few seconds ago
event_1	{"Temperature":32,"pressure":91}	json	a few seconds ago
event_1	{"Temperature":49,"pressure":36}	json	a few seconds ago
event_1	{"Temperature":44,"pressure":65}	json	a few seconds ago
event_1	{"Temperature":92,"pressure":14}	json	a few secor

1 Simulation running

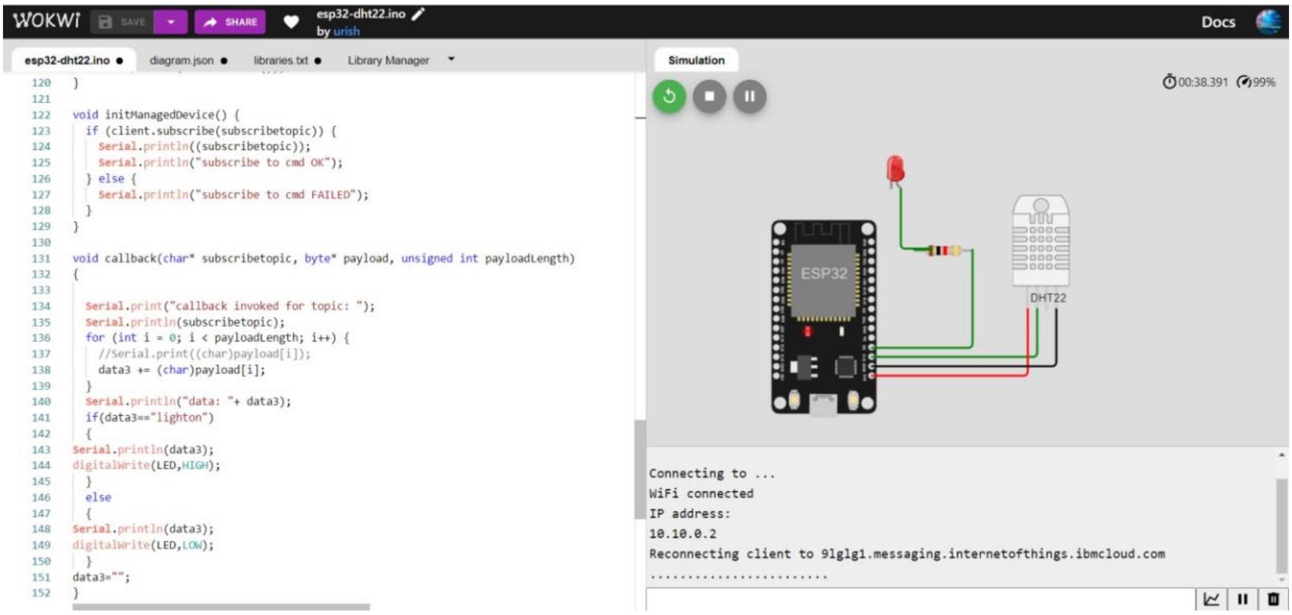
WOKWI Online Simulator ESP32 : https://wokwi.com/projects/322410731508073042

```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQtt
```

```
#include "DHT.h"// Library for dht11
#define DHTPIN 15     // what pin we're connected to
#define DHTTYPE DHT22   // define type of sensor DHT 11
#define LED 2

DHT dht (DHTPIN, DHTTYPE);// creating the instance by passing pin and typr of dht
connected  void callback(char* subscribetopic, byte* payload, unsigned int
payloadLength);
//-------credentials of IBM Accounts------

#define ORG "rr454u"//IBM ORGANITION ID
#define DEVICE_TYPE "sensor"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "sensor_1"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "12345678"     //Token
String data3; float h, t;

//-------- Customise the above values -------- char server[] = ORG
".messaging.internetofthings.ibmcloud.com";// Server Name char publishTopic[] =
"iot-2/evt/Data/fmt/json";// topic name and type of event perform and format in
which data to be send char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd
REPRESENT command type AND COMMAND IS TEST OF FORMAT STRING
char authMethod[] = "use-token-auth";// authentication method char
token[] = TOKEN; char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client
id

//-----------------------------------------
WiFiClient wifiClient; // creating the instance for wificlient
PubSubClient client(server, 1883, callback ,wifiClient); //calling the predefined
client id by passing parameter like server id,portand wificredential


void setup()// configureing the ESP32
{
  Serial.begin(115200);   dht.begin();
pinMode(LED,OUTPUT); delay(10);
Serial.println(); wificonnect();
mqttconnect();
}  void loop()// Recursive
Function
{    h = dht.readHumidity();   t
= dht.readTemperature();
  Serial.print("Temperature:");
  Serial.println(t);
  Serial.print("Humidity:");
  Serial.println(h);
PublishData(t, h);   delay(1000);
if (!client.loop()) {
mqttconnect();
  }
}
```

```
/*....................................retrieving to
Cloud...............................*/   void
PublishData(float temp, float humid) {
mqttconnect();//function call for connecting to ibm
  /*        creating the String in in form JSon to update the data to ibm
cloud    */
  String payload = "{\"Temperature\":";
payload += temp;
  payload += "," "\"Humidity\":";
payload += humid;   payload +=
"}";


  Serial.print("Sending payload: ");
  Serial.println(payload);
      if (client.publish(publishTopic, (char*)
payload.c_str())) {
    Serial.println("Publish ok");// if it sucessfully upload data on the cloud
then it will print publish ok in Serial monitor or else it will print publish
failed    } else {
    Serial.println("Publish failed");  }

}

void mqttconnect() {   if (!client.connected())
{
    Serial.print("Reconnecting      client      to      ");
Serial.println(server);                               while
(!!!client.connect(clientId,    authMethod,    token))    {
Serial.print(".");      delay(500);
    }
initManagedDevice();
     Serial.println();
  } } void wificonnect() //function defination for wificonnect
{
  Serial.println();
  Serial.print("Connecting to ");

  WiFi.begin("Wokwi-GUEST", "", 6);//passing the wifi credentials to establish the
connection   while (WiFi.status() != WL_CONNECTED) {     delay(500);
Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}  void initManagedDevice()
{
  if (client.subscribe(subscribetopic)) {
Serial.println((subscribetopic));
    Serial.println("subscribe to cmd OK");
  } else {
    Serial.println("subscribe to cmd FAILED");
  } }   void callback(char* subscribetopic, byte* payload, unsigned int
payloadLength)   {

  Serial.print("callback invoked for topic: ");  Serial.println(subscribetopic);
  for (int i = 0; i < payloadLength; i++) {
    //Serial.print((char)payload[i]);     data3
+= (char)payload[i];
  }
  Serial.println("data: "+ data3);
if(data3=="lighton")
  {
Serial.println(data3); digitalWrite(LED,HIGH);
  }   else
  {
Serial.println(data3); digitalWrite(LED,LOW);
  } data3="";
}
```

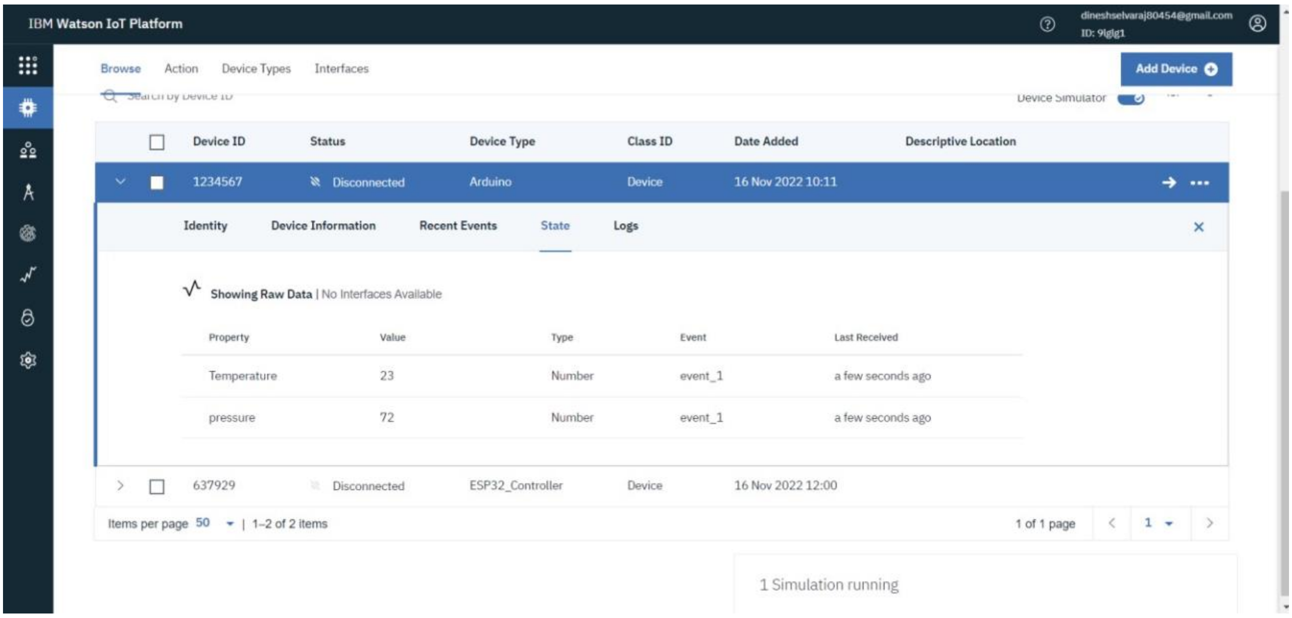
Simulation Output in the Wokwi web site:



```
}

void initManagedDevice() {
  if (client.subscribe(subscribetopic)) {
    Serial.println((subscribetopic));
    Serial.println("subscribe to cmd OK");
  } else {
    Serial.println("subscribe to cmd FAILED");
  }
}

void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)
{

  Serial.print("callback invoked for topic: ");
  Serial.println(subscribetopic);
  for (int i = 0; i < payloadLength; i++) {
    //Serial.print((char)payload[i]);
    data3 += (char)payload[i];
  }
  Serial.println("data: "+ data3);
  if(data3=="lighton")
  {
Serial.println(data3);
digitalWrite(LED,HIGH);
  }
  else
  {
Serial.println(data3);
digitalWrite(LED,LOW);
  }
data3="";
}
```

```
Connecting to ...
WiFi connected
IP address:
10.10.0.2
Reconnecting client to 9lglg1.messaging.internetofthings.ibmcloud.com
........................
```

Wokwi Simulation Output in the IBM Watson Platform:



□	Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
□	1234567	Disconnected	Arduino	Device	16 Nov 2022 10:11	
□	637929	Disconnected	ESP32_Controller	Device	16 Nov 2022 12:00	

Showing Raw Data | No Interfaces Available

Property	Value	Type	Event	Last Received
Temperature	23	Number	event_1	a few seconds ago
pressure	72	Number	event_1	a few seconds ago

Items per page 50 | 1–2 of 2 items

1 of 1 page 1

1 Simulation running