

Project Development Phase

Delivery of Sprint -1

Team ID	PNT2022TMID48383
Project Name	Smart Farmer-IOT Enabled Smart Farming Application

In Sprint-1 we are going to develop the python code .

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)

Installing pip ibmiotf packages in python idle.After that we have to write the python code.

```
C:\Windows\System32\cmd.exe
ERROR: No matching distribution found for ibmiotf

C:\Users\THIRUMURUGAN.V\AppData\Local\Programs\Python\Python39-32\Scripts>pip install ibmiotf
Collecting ibmiotf
  Downloading ibmiotf-0.4.0.tar.gz (71 kB)
    | 71 kB 13 kB/s
Collecting iso8601>=0.1.12
  Downloading iso8601-1.1.0-py3-none-any.whl (9.9 kB)
Collecting pytz>=2017.3
  Downloading pytz-2022.6-py2.py3-none-any.whl (498 kB)
    | 498 kB 109 kB/s
Collecting paho-mqtt>=1.3.1
  Downloading paho-mqtt-1.6.1.tar.gz (99 kB)
    | 99 kB 380 kB/s
Collecting requests>=2.18.4
  Downloading requests-2.28.1-py3-none-any.whl (62 kB)
    | 62 kB 137 kB/s
Collecting requests_toolbelt>=0.8.0
  Downloading requests_toolbelt-0.10.1-py2.py3-none-any.whl (54 kB)
    | 54 kB 155 kB/s
Collecting urllib3<1.27,>=1.21.1
  Downloading urllib3-1.26.12-py2.py3-none-any.whl (140 kB)
    | 140 kB 60 kB/s
Collecting idna<4,>=2.5
  Downloading idna-3.4-py3-none-any.whl (61 kB)
    | 61 kB 8.1 kB/s
Collecting certifi>=2017.4.17
  Downloading certifi-2022.9.24-py3-none-any.whl (161 kB)
    | 161 kB 117 kB/s
Collecting charset-normalizer<3,>=2
  Downloading charset-normalizer-2.1.1-py3-none-any.whl (39 kB)
Using legacy 'setup.py install' for ibmiotf, since package 'wheel' is not installed.
Using legacy 'setup.py install' for paho-mqtt, since package 'wheel' is not installed.
Installing collected packages: iso8601, pytz, paho-mqtt, urllib3, idna, certifi, charset-normalizer, requests, requests-toolbelt, ibmiotf
  Running setup.py install for paho-mqtt ... done
  WARNING: The script normalizer.exe is installed in 'c:\users\thirumurugan.v\appdata\local\programs\python\python39-32\Scripts' which is not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
  Running setup.py install for ibmiotf ... done
Successfully installed certifi-2022.9.24 charset-normalizer-2.1.1 ibmiotf-0.4.0 idna-3.4 iso8601-1.1.0 paho-mqtt-1.6.1 pytz-2022.6 requests-2.28.1 requests-toolbelt-0.10.1 urllib3-1.26.12
WARNING: You are using pip version 20.2.3; however, version 22.3.1 is available.
You should consider upgrading via the 'c:\users\thirumurugan.v\appdata\local\programs\python\python39-32\python.exe -m pip install --upgrade pip' command.

C:\Users\THIRUMURUGAN.V\AppData\Local\Programs\Python\Python39-32\Scripts>pip install ibmiotf_
```

```
python code.py - C:\Users\THIRUMURUGAN.V\Desktop\python code.py (3.9.1)
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 = "49x4b9"
deviceType = "weather_monitoring"
deviceId = "weather_today"
authMethod = "token"
authToken = "Qp4oHg7bZHhaQeigMA"

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")

Ln: 1 Col: 0
```

```
python code.py - C:\Users\THIRUMURUGAN.V\Desktop\python code.py (3.9.1)
File Edit Format Run Options Window Help

authMethod = "token"
authToken = "Qp4oHg7bZHhaQeigMA"

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 IoT")
    time.sleep(1)

    deviceCli.commandCallback = myCommandCallback

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

Ln: 1 Col: 0
```

CODE:

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

#Provide your IBM Watson Device Credentials
organization = "49x4b9"
deviceType = "weather_monitoring"
deviceId = "weather_today"
authMethod = "token"
authToken = "Qp4oHg?bZHhaQeigMA"
```

```

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 IoT")
    time.sleep(1)

deviceCli.commandCallback = myCommandCallback

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

```

Simulation output in the python idle:

```

Published Temperature = 62 C Humidity = 85 % soil Moisture = 21 % to IBM Watson
Published Temperature = 36 C Humidity = 64 % soil Moisture = 68 % to IBM Watson
Published Temperature = 7 C Humidity = 25 % soil Moisture = 100 % to IBM Watson
Published Temperature = 44 C Humidity = 60 % soil Moisture = 11 % to IBM Watson
Published Temperature = 30 C Humidity = 12 % soil Moisture = 35 % to IBM Watson
Published Temperature = 83 C Humidity = 40 % soil Moisture = 70 % to IBM Watson
Published Temperature = 37 C Humidity = 22 % soil Moisture = 41 % to IBM Watson
Published Temperature = 0 C Humidity = 30 % soil Moisture = 15 % to IBM Watson
Published Temperature = 65 C Humidity = 88 % soil Moisture = 10 % to IBM Watson
Published Temperature = 97 C Humidity = 34 % soil Moisture = 70 % to IBM Watson
Published Temperature = 8 C Humidity = 30 % soil Moisture = 5 % to IBM Watson
Published Temperature = 34 C Humidity = 37 % soil Moisture = 69 % to IBM Watson
Published Temperature = 75 C Humidity = 17 % soil Moisture = 2 % to IBM Watson
Published Temperature = 80 C Humidity = 84 % soil Moisture = 60 % to IBM Watson
Published Temperature = 16 C Humidity = 9 % soil Moisture = 39 % to IBM Watson
Published Temperature = 34 C Humidity = 99 % soil Moisture = 35 % to IBM Watson
Published Temperature = 72 C Humidity = 8 % soil Moisture = 87 % to IBM Watson
Published Temperature = 77 C Humidity = 10 % soil Moisture = 30 % to IBM Watson
Published Temperature = 73 C Humidity = 4 % soil Moisture = 84 % to IBM Watson
Published Temperature = 49 C Humidity = 99 % soil Moisture = 78 % to IBM Watson
Published Temperature = 96 C Humidity = 31 % soil Moisture = 72 % to IBM Watson
Published Temperature = 89 C Humidity = 86 % soil Moisture = 11 % to IBM Watson
Published Temperature = 49 C Humidity = 81 % soil Moisture = 53 % to IBM Watson
Published Temperature = 19 C Humidity = 23 % soil Moisture = 31 % to IBM Watson
Published Temperature = 29 C Humidity = 34 % soil Moisture = 64 % to IBM Watson
Published Temperature = 44 C Humidity = 67 % soil Moisture = 58 % to IBM Watson
Published Temperature = 52 C Humidity = 82 % soil Moisture = 73 % to IBM Watson
Published Temperature = 1 C Humidity = 88 % soil Moisture = 0 % to IBM Watson
Published Temperature = 61 C Humidity = 26 % soil Moisture = 48 % to IBM Watson
Published Temperature = 32 C Humidity = 100 % soil Moisture = 100 % to IBM Watson
Published Temperature = 46 C Humidity = 74 % soil Moisture = 81 % to IBM Watson
Published Temperature = 45 C Humidity = 23 % soil Moisture = 26 % to IBM Watson
Published Temperature = 64 C Humidity = 63 % soil Moisture = 62 % to IBM Watson
Published Temperature = 46 C Humidity = 89 % soil Moisture = 8 % to IBM Watson
Published Temperature = 100 C Humidity = 11 % soil Moisture = 87 % to IBM Watson
Published Temperature = 21 C Humidity = 73 % soil Moisture = 42 % to IBM Watson
Published Temperature = 56 C Humidity = 92 % soil Moisture = 29 % to IBM Watson
Published Temperature = 8 C Humidity = 30 % soil Moisture = 50 % to IBM Watson
Published Temperature = 34 C Humidity = 77 % soil Moisture = 52 % to IBM Watson

```

RESTART: C:\Users\THIRUMURUGAN.V\Desktop\python code..py

Wokwi Link: <https://wokwi.com/projects/348283018223288915>

OUTPUT & SIMULATION:

The screenshot displays the Wokwi web interface for an Arduino simulation. On the left, the 'sketch.ino' file contains the following code:

```
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 void callback(char* topic, byte* payload, unsigned int payloadLength)
4 #define ORG "92zbf"
5 #define DEVICE_TYPE "esp32"
6 #define DEVICE_ID "12345"
7 #define TOKEN "12345678"
8 String data;
9 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
10 char publishTopic[] = "iot-2/evt/Data/fmt/json";
11 char subscribeTopic[] = "iot-2/cmd/test/fmt/String";
12 char authMethod[] = "use-token-auth";
13 char token[] = TOKEN;
14 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
15 WiFiClient wifiClient;
16 PubSubClient client(server, 1883, callback, wifiClient);
17 const int trigPin = 5;
18 const int echoPin = 18;
19 #define SOUND_SPEED 0.034
20 long duration;
21 float distance;
22 void setup() {
23   Serial.begin(115200);
24   pinMode(trigPin, OUTPUT);
25   pinMode(echoPin, INPUT);
26   wifiConnect();
27   mqttConnect();
28 }
29 void loop()
30 {
```

The simulation window on the right shows a visual representation of the ESP32 and HC-SR04 sensor connected by wires. Below the visual, the simulation output displays the following information:

- IP address: 10.10.0.2
- Reconnecting client to 92zbf.messaging.internetofthings.ibmcloud.com
- iot-2/cmd/test/fmt/String
- subscribe to cmd OK
- Distance (cm): 399.92

Alert to IBM cloud:

The screenshot shows the IBM Watson IoT Platform dashboard. The top navigation bar includes links for Browse, Action, Device Types, and Interfaces. A table displays the following data:

Device ID	Status	Device Type	Class ID	Date Added
1234	Disconnected	ABCD	Device	Nov 19, 2022 10:21 AM
ABCD_2	Connected	ABCD	Device	Nov 19, 2022 10:27 AM
Weather_3	Disconnected	Weather	Device	Nov 18, 2022 3:24 PM

The interface also includes a search bar for Device ID, a 'Device Simulator' toggle, and a notification stating '1 Simulation running'.