

# PROJECT DEVELOPMENT DELIVERY

## SPRINT-1

Date	29 OCTOBER 2022
Team ID	PNT2022TMID33093
ProjectName	IOT Based Smart crop protection system for Agriculture

## Create IBM Watson IOT Platform

### IBM WATSON CLOUD

The screenshot shows a web browser window with multiple tabs open, including WhatsApp, Sprint 4.pdf, Sprint 3.pdf, Sprint 2.pdf, PREREQUISITES, Welcome to Proj, and Catalog - IBM Cl. The address bar shows the URL cloud.ibm.com/catalog?category=iot. The page header includes the IBM Cloud logo, a search bar, and navigation links for Catalog, Manage, and Rithika S's Account. The main content area is titled 'Catalog' and features a search bar. Below the search bar, there is a filter section for 'Internet of Things' with options for Type (All, Services, Software, Professional services) and Provider. The main product listing shows 'Internet of Things Platform' by IBM. The bottom of the screen displays the Windows taskbar with the search bar and system tray icons.

The screenshot shows the IBM Cloud catalog page for the Internet of Things Platform. The page is titled "Internet of Things Platform" and describes it as the hub for all things IBM IoT. The "Create" tab is active, showing a "Select a location" dropdown set to "Frankfurt (eu-de)" and a "Select a pricing plan" section. A table lists the "Lite" plan, which includes up to 500 registered devices and 200 MB of data metric, for a price of "Free". A summary panel on the right shows the plan details and a warning about existing Lite plan instances. The bottom of the screen shows the Windows taskbar with various application icons.

WhatsApp x Sprint 4.pdf x Sprint 3.pdf x Sprint 2.pdf x PREREQUISITES x Welcome to Proj x Internet of Thing x

cloud.ibm.com/catalog/services/internet-of-things-platform

IBM Cloud Search resources and products... Catalog Manage Rithika S's Account

Internet of Things Platform

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.

Create About

Type Service

Provider IBM

Last updated 08/15/2022

Category Internet of Things

Compliance IAM-enabled

Location Frankfurt London

Select a location

Frankfurt (eu-de)

Select a pricing plan

Displayed prices do not include tax. Monthly prices shown are for country or location: [United States](#)

Plan	Features	Pricing
Lite	Includes up to 500 registered devices, and a maximum of 200 MB of each data metric Maximum of 500 registered devices Maximum of 500 application bindings	Free

Summary

Internet of Things Platform Free

Location: Frankfurt

Plan: Lite

Service name: Internet of Things Platform-ar

Resource group: Default

Existing Lite plan instance

You can have only 1 Lite plan instance of this service per resource group. [Delete](#) your current Lite plan instance in Default resource group to create a new one, or [view the existing instance](#).

☐ I have read and agree to the following license agreements: [Terms](#)

Type here to search

01:40 09-11-2022

## Device credentials information:

The screenshot shows the "Add Device" wizard in the IBM Watson IoT Platform. The wizard is a multi-step process with four steps: Identity, Device Information, Security, and Summary. The "Device Information" step is currently active, showing fields for "Device Type" (ABCD), "Device ID" (1234), and "Security Token" (12345678). A "View Metadata" button is located below the "Device ID" field. The "Summary" step is the final step in the process, indicated by a blue circle. The bottom of the screen shows "Back" and "Finish" buttons.

krz3g7.internetofthings.ibmcloud.com/dashboard/devices/browse/add

IBM Watson IoT Platform

Browse Action Device Types Interfaces

Add Device

Identity Device Information Security Summary

Verify that the following information is correct then select Finish

Device Type  
ABCD

Device ID  
1234

[View Metadata](#)

Security Token  
12345678

[Back](#) [Finish](#)

← → ↻ krz3g7.internetofthings.ibmcloud.com/dashboard/devices/drilldown/ABCD:1234?returnTo=/devices/browse

IBM Watson IoT Platform k.sakthipriya.23eceb@psvpec.in ID: krz3g7

← Back

## Device Drilldown - 1234

Device Credentials

Connection Information

Recent Events

State

Device Information

Metadata

Diagnostics

Connection Logs

Device Actions

### Device Credentials

You registered your device to the organization. Add these credentials to the device to connect it to the platform. After the device is connected, you can navigate to view connection and event details.

Organization ID	krz3g7
Device Type	ABCD
Device ID	1234
Authentication Method	use-token-auth
Authentication Token	12345678

**!** Authentication tokens are non-recoverable. If you misplace this token, you will need to re-register the device to generate a new authentication token.

[Find out how to add these credentials to your device](#)

← → ↻ krz3g7.internetofthings.ibmcloud.com/dashboard/devices/drilldown/ABCD:1234?returnTo=/devices/browse

IBM Watson IoT Platform k.sakthipriya.23eceb@psvpec.in ID: krz3g7

← Back

## Device Drilldown - 1234

Device Credentials

Connection Information

Recent Events

State

Device Information

Metadata

Diagnostics

Connection Logs

Device Actions

### Connection Information

Basic connection information about this device.

Device ID	1234
Device Type	ABCD
Date Added	Nov 14, 2022 12:12 PM
Added By	k.sakthipriya.23eceb@psvpec.in
Connection Status	Disconnected

### Recent Events

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

Event	Value	Format	Last Received
-------	-------	--------	---------------

**IBM Watson Output:**

The screenshot displays the IBM Watson IoT Platform interface. The top navigation bar shows the user 'k.sakthipriya.23eceb@psvpec.in' with ID 'krz3g7'. The main content area is titled 'Browse' and shows a list of devices. The selected device '1234' is shown in a detailed view with the following information:

Event	Value	Format	Last Received
IoTSensor	{"temp":101,"Humid":97}	json	a few seconds ago
IoTSensor	{"temp":104,"Humid":92}	json	a few seconds ago
IoTSensor	{"temp":93,"Humid":69}	json	a few seconds ago
IoTSensor	{"temp":96,"Humid":66}	json	a few seconds ago
IoTSensor	{"temp":99,"Humid":100}	json	a few seconds ago

## Create a Device and Configure the IBM IOT Platform

### Python Code:

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

#IBM Watson Device Credentials.
organization = "krz3g7"
deviceType = "ABCD"
deviceId = "1234"
authMethod = "token"
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, sprinklers are
turned ON" %temp_sensor }
, qos=0)
    sleep(1)
    if success:
        print( 'Published alert1 : ', "Temperature(%s) is high, sprinklers are turned ON" %temp_sensor,"to
IBM Watson")

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, sprinklers are
turned ON" %temp_sensor }
, qos=0)
    sleep(1)
    if success:
        print( 'Published alert1 : ', "Temperature(%s) is high, sprinklers are turned ON" %temp_sensor,"to

```



IBM Watson")

## Python Output:

```
*Python 3.7.4 Shell*
File Edit Shell Debug Options Window Help
stem {IBM}\sprint 1.py
>>>
RESTART: C:/Users/CHELLA/Desktop/BER PROJECT/IOT based smart crop protection sy
stem {IBM}/PY1.py
2022-11-14 12:19:24,359 ibmiotf.device.Client INFO Connected successfu
lly: d:krz3g7:ABCD:1234
Published Temperature = 101 C Humidity = 76 % to IBM Watson
Published Temperature = 109 C Humidity = 68 % to IBM Watson
Published Temperature = 97 C Humidity = 62 % to IBM Watson
Published Temperature = 108 C Humidity = 61 % to IBM Watson
Published Temperature = 99 C Humidity = 100 % to IBM Watson
Published Temperature = 96 C Humidity = 66 % to IBM Watson
Published Temperature = 93 C Humidity = 69 % to IBM Watson
Published Temperature = 104 C Humidity = 92 % to IBM Watson
Published Temperature = 101 C Humidity = 97 % to IBM Watson
Published Temperature = 92 C Humidity = 88 % to IBM Watson
Published Temperature = 107 C Humidity = 68 % to IBM Watson
Published Temperature = 101 C Humidity = 76 % to IBM Watson
Published Temperature = 106 C Humidity = 71 % to IBM Watson
Published Temperature = 97 C Humidity = 68 % to IBM Watson
Published Temperature = 110 C Humidity = 93 % to IBM Watson
Published Temperature = 95 C Humidity = 78 % to IBM Watson
Published Temperature = 95 C Humidity = 89 % to IBM Watson
Published Temperature = 96 C Humidity = 84 % to IBM Watson
Published Temperature = 103 C Humidity = 63 % to IBM Watson
Published Temperature = 92 C Humidity = 84 % to IBM Watson
Published Temperature = 97 C Humidity = 83 % to IBM Watson
Published Temperature = 95 C Humidity = 64 % to IBM Watson
Published Temperature = 93 C Humidity = 70 % to IBM Watson
Published Temperature = 100 C Humidity = 60 % to IBM Watson
Published Temperature = 108 C Humidity = 91 % to IBM Watson
Published Temperature = 104 C Humidity = 81 % to IBM Watson
Published Temperature = 93 C Humidity = 81 % to IBM Watson
Published Temperature = 100 C Humidity = 94 % to IBM Watson
Published Temperature = 108 C Humidity = 86 % to IBM Watson
Published Temperature = 99 C Humidity = 72 % to IBM Watson
Published Temperature = 110 C Humidity = 83 % to IBM Watson
```





