**Python (source code)**

**import time**

import sys

import ibmiotf.application

import ibmiotf.device

import random

import json

#Provide your IBM Watson Device Credentials

organization = "fxocv2"

deviceType = "iotdevice"

deviceId = "1001"

authMethod = "token"

authToken = "7382651854"

# Initialize the device client.

#taking predefined values for jarweight and cylinderweight assuming 40 and 35 values respevtively.

#here we are assuming after 20 loops the leakage sensor will be detected.

jar=40

cy=35

lek=0

fan="off"

leak="off"

i=0

def myCommandCallback(cmd):

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

if cmd.data['command']=='fanon':

print("Fan ON IS RECEIVED")

elif cmd.data['command']=='fanoff':

print("Fan OFF IS RECEIVED")

if cmd.command == "setInterval":

if 'interval' not in cmd.data:

print("Error - command is missing required information: 'interval'")

else:

interval = cmd.data['interval']

elif cmd.command == "print":

if 'message' not in cmd.data

print("Error - command is missing required information: 'message'")

else:

print(cmd.data['message'])

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()

#we are assuming after cy=0 and jar=0 then the cylinder and jar status as empty.

while True:

cy=cy-1

jar=jar-1

lek=lek+1

if(cy==0):

print("cylinder is empty")

cy=35

if(jar==0):

print("jar is empty")

jar=40

#here we assumed after 20 loops cylinder will be leaked.and sensor will be detected.becoz as we are not using any sensor.

if(lek==20):

print("gas is leaking so switch on the fan")

lek=0

data = {"d":{ 'cylinderweight' : cy, 'jarweight' : jar, 'leakagesensor' : lek }}

print (data)

def myOnPublishCallback():

print ("Published cylinderweight = %s %%" % cy,"jarweight = %s %%" % jar, "leakagesensor = %s %%" % lek, "to IBM Watson")

success = deviceCli.publishEvent("Data", "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()

output of python code

