FINAL

 CODE

```
import time

 import sys

import ibmiotf.application

 import ibmiotf.device

 import random

. #Provide your IBM Watson Device Credentials

organization = "uo60re"


deviceType = "AKASH"

 deviceId = "1234"

 authMethod = "token"
 authToken = "12345678"

# Initialize GPIO



def myCommandCallback(cmd):
 print("Command received: %s" %
cmd.data['command']) status=cmd.data['command']
 if status=="lighton":
print ("led is on")
else:
 print ("led is off")
 #print(cmd) cmd):
 print("Command received: %s" %
 deviceType = "AKASH"
 deviceId = "1234"
authMethod = "token"
authToken = "12345678"

# Initialize GPIO def myCommandCallback(
cmd.data['command'])
status=cmd.data['command'] if status=="lighton":
print ("led is on")
 else:
print ("led 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()

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

deviceCli.connect()



while True:

#Get Sensor Data from DHT11

temp=random.randint(60,100) Turbidity=random.randint(0,100)


phvalue=random.randint(2,14)
```

```
data = { 'temp' : temp, 'Turbidity':
 Turbidity,'phvalue': phvalue}
#print data
def myOnPublishCallback():
 print ("Published temp = %s
'C" % temp, "Turbidity = %s %%" % Turbidity,"phvalue = %s %%" % phvalue, "to IBM Watson")

success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0, on_publish=myOnPublishCallback)
if not success:
print("Not connected to
IoTF")
time.sleep(10)



deviceCli.commandCallback = myCommandCallback



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