

Project Development Phase

Sprint -2

Date	17 November 2022
Team ID	PNT2022TMID34928
Project Name	IOT BASED SMART CROP PROTECTION SYSTEM

Python Code:

```
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random
import cv2
import numpy as np
threatlst=["elephant","bear","tiger","horse","monkey"]
n=len(threatlst)
#Provide your IBM Watson Device Credentials
organization = "2ldaf5"
deviceType1 = "Sensor"
deviceId1 = "DHT"
authMethod = "token"
authToken1 = "NeVIAy2K16H)d9sXvz"

deviceType2 = "Sensor1"
deviceId2 = "Soil_moisture"
authToken2= "zwr247qk1Xca0w?QEs"
```

```
deviceType3 = "Actuator"  
deviceId3 = "Water_pump"  
authToken3= "Pze?D!@FjZeAtfMB4q"
```

```
deviceType4= "Sensor2"  
deviceId4= "PIR"  
authToken4= "i-yXXf?FnB011nEycG"
```

```
deviceType5= "Sensor3"  
deviceId5="Ultrasonic"  
authToken5="e&QzDxiHpQ4GaRyPGJ"
```

```
deviceType6="Detector"  
deviceId6="Camera"  
authToken6="f7LMx6-a(uhdnDcKa-
```

```
deviceType7="Output"  
deviceId7="LED"  
authToken7="qJIBVJHP9@Ihl8@CK3"
```

```
deviceType8="Actuator"  
deviceId8="Speaker"  
authToken8="TitkHSlzaTy_gxEtve"
```

Initialize GPIO

```
def myCommandCallback(cmd):  
    print("Command received: %s \n" % cmd.data['command'])  
    status=cmd.data['command']  
    if status=="ON":
```

```

        print ("Water Pump is turned ON \n")
    else :
        print ("Water Pump is turned OFF \n")
def myCommandCallback1(cmd):
    print("Command received: %s \n" % cmd.data['command1'])
    status=cmd.data['command1']
    if status=="ON":
        print ("LED is turned ON \n")
    else :
        print ("LED is turned OFF \n")
def myCommandCallback2(cmd):
    print("Command received: %s \n" % cmd.data['command2'])
    status=cmd.data['command2']
    if status=="ON":
        print ("Speaker is turned ON \n")
    else :
        print ("Speaker is turned OFF \n")
#Automatically turning on/off led and speaker
def threat(x):
    print("LED is turned ON")
    print("Speaker is turned ON")
    data8= {'Threat' : str(x)}
    def myOnPublishCallback8():
        print ("Threat allert is send to IBM Watson")
    success8 = deviceCli8.publishEvent("Threat status", "json", data8, qos=0,
on_publish=myOnPublishCallback8)
    if not success8:
        print("Not connected to IoT")
    time.sleep(1)

```

```

def cam():

net=cv2.dnn.readNet('C:/Users/hp/OneDrive/Desktop/opencvtrial/yolov3.weights','C:/User
s/hp/OneDrive/Desktop/opencvtrial/yolov3.cfg.txt')

classes=[]

with open('C:/Users/hp/OneDrive/Desktop/opencvtrial/coco.names','r') as f:
    classes=f.read().splitlines()

cap=cv2.VideoCapture('blackbear.mp4')

for i in range(50):
    _,img=cap.read()
    height,width,_=img.shape

    blob=cv2.dnn.blobFromImage(img,1/255,(416,416),(0,0,0),swapRB=True,crop=False)
    #(img,reduction the pixels size,size of the image,rgb colour)

    net.setInput(blob)

    output_layers_names=net.getUnconnectedOutLayersNames()

    layeroutput=net.forward(output_layers_names)

    boxes=[]

    confidences=[]

    class_ids=[]

    for output in layeroutput:
        for detection in output:
            scores=detection[5:]
            class_id=np.argmax(scores)
            confidence=scores[class_id]

            if confidence > 0.5:

                center_x=int(detection[0]*width)
                center_y =int(detection[1]*height)
                w=int(detection[2]*width)
                h=int(detection[3]*height)
                x=int(center_x - w/2)

```

```

        y=int(center_y - h/2)
        boxes.append([x,y,w,h])
        confidences.append((float(confidence)))
        class_ids.append(class_id)
        animal=classes[class_id]

indexes=cv2.dnn.NMSBoxes(boxes,confidences,0.5,0.4)
font=cv2.FONT_HERSHEY_COMPLEX
colors=np.random.uniform(0,255,size=(len(boxes),3))
for i in indexes.flatten():
    x,y,w,h=boxes[i]
    label=str(classes[class_ids[i]])
    confidence=str(round(confidences[i],2))
    color=colors[i]
    cv2.rectangle(img,(x,y),(x+w ,y+h),color,2)
    cv2.putText(img,label + " " +confidence,(x,y+20),font,2,(255,255,0),2)
cv2.imshow('Target Image',img)
key=cv2.waitKey(1)
if key ==ord('q'):
    break
print("Alert! detected animal is "+str(animal))
for x in threatlst :
    if animal==x :
        status1="YES"
        threat(status1)
cap.release()
cv2.destroyAllWindows()
data6 = { 'Intruded_Animal' : str(animal) }
def myOnPublishCallback6():

```

```

    print ("Published Intruded Animal is "+str(animal) ,"to IBM Watson")

    success6 = deviceCli6.publishEvent("Alert", "json", data6, qos=0,
on_publish=myOnPublishCallback6)

    if not success6:

        print("Not connected to IoT")

    time.sleep(1)

try:

    deviceOptions1 = {"org": organization, "type": deviceType1, "id": deviceId1, "auth-
method": authMethod, "auth-token": authToken1}

    deviceCli1 = ibmiotf.device.Client(deviceOptions1)

    #.....

    deviceOptions2 = {"org": organization, "type": deviceType2, "id": deviceId2, "auth-
method": authMethod, "auth-token": authToken2}

    deviceCli2 = ibmiotf.device.Client(deviceOptions2)

    #.....

    deviceOptions3 = {"org": organization, "type": deviceType3, "id": deviceId3, "auth-
method": authMethod, "auth-token": authToken3}

    deviceCli3 = ibmiotf.device.Client(deviceOptions3)

    #.....

    deviceOptions4 = {"org": organization, "type": deviceType4, "id": deviceId4, "auth-
method": authMethod, "auth-token": authToken4}

    deviceCli4 = ibmiotf.device.Client(deviceOptions4)

    #.....

    deviceOptions5 = {"org": organization, "type": deviceType5, "id": deviceId5, "auth-
method": authMethod, "auth-token": authToken5}

    deviceCli5 = ibmiotf.device.Client(deviceOptions5)

    #.....

    deviceOptions6 = {"org": organization, "type": deviceType6, "id": deviceId6, "auth-
method": authMethod, "auth-token": authToken6}

    deviceCli6 = ibmiotf.device.Client(deviceOptions6)

    #.....

```

```
deviceOptions7 = {"org": organization, "type": deviceType7, "id": deviceId7, "auth-  
method": authMethod, "auth-token": authToken7}
```

```
deviceCli7 = ibmiotf.device.Client(deviceOptions7)
```

```
#.....
```

```
deviceOptions8 = {"org": organization, "type": deviceType8, "id": deviceId8, "auth-  
method": authMethod, "auth-token": authToken8}
```

```
deviceCli8 = ibmiotf.device.Client(deviceOptions8)
```

```
#.....
```

```
except Exception as e:
```

```
    print("Caught exception connecting device: %s" % str(e))
```

```
    sys.exit()
```

```
deviceCli1.connect()
```

```
deviceCli2.connect()
```

```
deviceCli3.connect()
```

```
deviceCli4.connect()
```

```
deviceCli5.connect()
```

```
deviceCli6.connect()
```

```
deviceCli7.connect()
```

```
deviceCli8.connect()
```

```
print("\n")
```

```
while (True):
```

```
    #Get Sensor Data from DHT11
```

```
    temp=random.randint(0,45)
```

```
    Humid=random.randint(0,100)
```

```
    data1 = { 'Temperature' : temp , 'Humidity': Humid}
```

```
    def myOnPublishCallback1():
```

```
        print ("Published Temperature  = %s C" % temp, "Humidity  = %s %" % Humid, "to  
IBM Watson\n")
```

```
        success1 = deviceCli1.publishEvent("DHT Sensor", "json", data1, qos=0,  
on_publish=myOnPublishCallback1)
```

```
        if not success1:
```

```

    print("Not connected to IoT\n")
time.sleep(1)
#Get Sensor Data from SOIL Moisture
Soil_moisture=random.randint(0,100)
data2 = { 'Soil_moisture' : Soil_moisture}
def myOnPublishCallback2():
    print ("Published Soil_moisture = %s %" % Soil_moisture, "to IBM Watson\n")
    success2 = deviceCli2.publishEvent("Soil Moisture Sensor", "json", data2, qos=0,
on_publish=myOnPublishCallback2)
    if not success2:
        print("Not connected to IoT")
        time.sleep(1)
#Automatically turning on/off water pump
if Soil_moisture <= 20:
    print("Water pump is turned ON\n")
    deviceCli3.commandCallback = myCommandCallback
#Get Sensor Data from PIR
pir=random.randint(0,1)
data4 = { 'PIR_output' : pir}
#Movement detection by PIR
if(pir==1):
    print("movement is detected\nultrasonic sensor turned ON")
    def myOnPublishCallback4():
        print ("Published PIR output  = %s " % pir, "to IBM Watson")
        success4 = deviceCli4.publishEvent("PIR Sensor", "json", data4, qos=0,
on_publish=myOnPublishCallback4)
        if not success4:
            print("Not connected to IoT")
            time.sleep(1)
#Get Sensor data from ultrasonic sensor

```



```

distance=random.randint(0,500)
data5 = { 'Distance:' : distance}
def myOnPublishCallback5():
    print ("Published distance = %s m" % distance, "to IBM Watson\n")
    success5 = deviceCli5.publishEvent("Ultrasonic Sensor", "json", data5, qos=0,
on_publish=myOnPublishCallback5)
    if not success5:
        print("Not connected to IoT")
    time.sleep(1)
    #turning on camera
    if(distance<=200):
        print("camera turned on\n")
        cam()
    deviceCli7.commandCallback = myCommandCallback1
    deviceCli8.commandCallback = myCommandCallback2
deviceCli1.disconnect()
deviceCli2.disconnect()
deviceCli4.disconnect()
deviceCli5.disconnect()
deviceCli6.disconnect()

```

Python Output :

```
opentrial.py - C:\Users\hp\OneDrive\Desktop\opencvtrial\opentrial.py (3.7.0)
File Edit Format Run Options Window Help

import time
import sys
import ibmiotf.application
import ibmiotf.device
import random
import cv2
import numpy as np

threatlist=["elephant","bear","tiger"]
n=len(threatlist)

#Provide your IBM Watson Device Credentials
organization = "21daf5"
deviceType1 = "Sensor"
deviceId1 = "DHT"
authMethod = "token"
authToken1 = "NeVIAy2K16H)d9sXvz"

deviceType2 = "Sensor1"
deviceId2 = "Soil_moisture"
authToken2 = "zwr247qk1Xca0w?QEs"

deviceType3 = "Actuator"
deviceId3 = "Water_pump"
authToken3 = "Pze7D18FJ2eAtfMB4q"

deviceType4 = "Sensor2"
deviceId4 = "PIR"
authToken4 = "i-yXf?FnB01nBycG"

deviceType5 = "Sensor3"
deviceId5 = "Ultrasonic"
authToken5 = "esQzDx1HpQ4GaRyP6J"

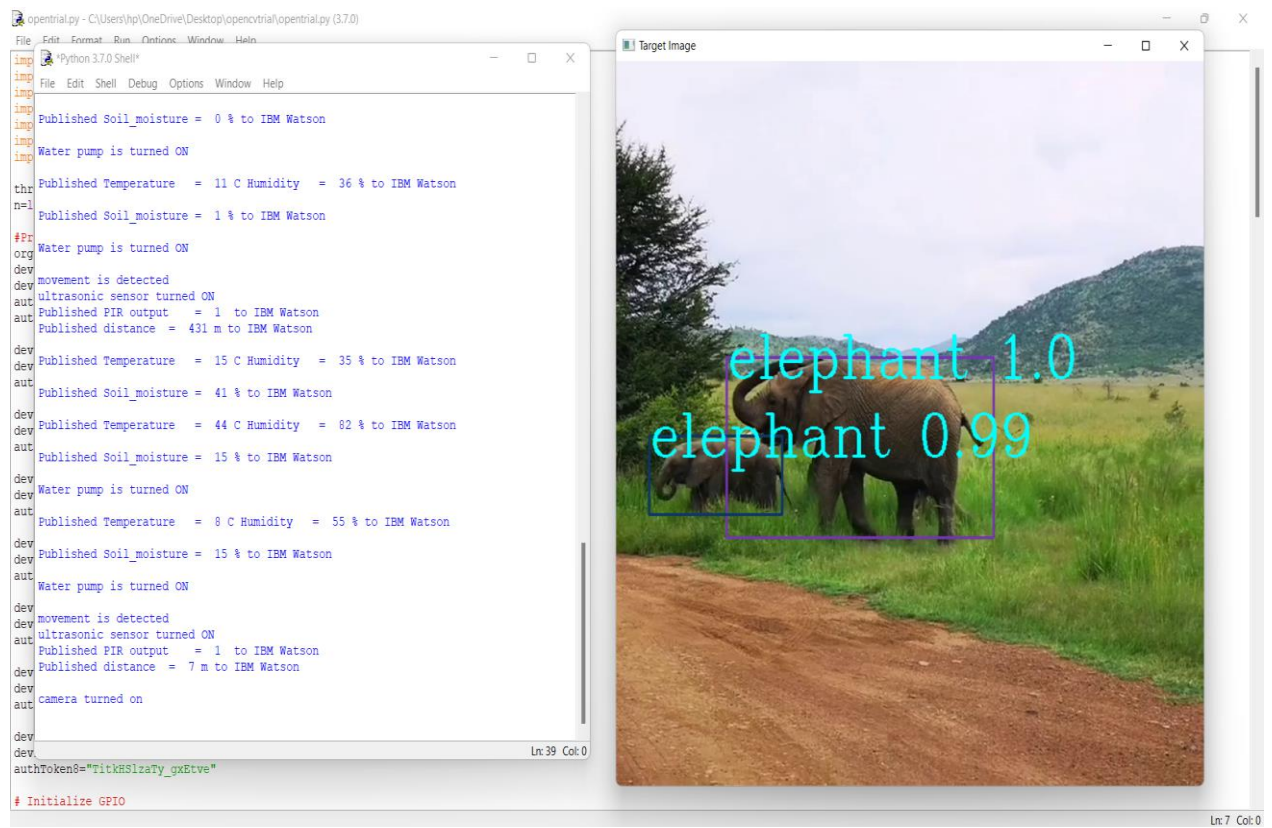
deviceType6 = "Detector"
deviceId6 = "Camera"
authToken6 = "f7LMx6-a (uhdnDcKa-"

deviceType7 = "Output"
deviceId7 = "LED"
authToken7 = "q3IBVJHF9@Ih18@CK3"

deviceType8 = "Actuator"
deviceId8 = "Speaker"
authToken8 = "TtkHSLzafy_gxEtve"

# Initialize GPIO

Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
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\hp\OneDrive\Desktop\opencvtrial\opentrial.py =====
2022-11-17 18:35:21,833 ibmiotf.device.Client INFO Connected successfully: d:21daf5:Sensor1:Soil_moisture
2022-11-17 18:35:23,161 ibmiotf.device.Client INFO Connected successfully: d:21daf5:Actuator:Water_pump
2022-11-17 18:35:25,417 ibmiotf.device.Client INFO Connected successfully: d:21daf5:Sensor2:PIR
2022-11-17 18:35:26,859 ibmiotf.device.Client INFO Connected successfully: d:21daf5:Sensor3:Ultrasonic
2022-11-17 18:35:29,436 ibmiotf.device.Client INFO Connected successfully: d:21daf5:Detector:Camera
2022-11-17 18:35:30,536 ibmiotf.device.Client INFO Connected successfully: d:21daf5:Output:LED
2022-11-17 18:35:31,653 ibmiotf.device.Client INFO Connected successfully: d:21daf5:Actuator:Speaker
Published Temperature = 40 C Humidity = 0 % to IBM Watson
Published Soil_moisture = 26 % to IBM Watson
movement is detected
ultrasonic sensor turned ON
Published PIR output = 1 to IBM Watson
Published distance = 280 m to IBM Watson
Published Temperature = 19 C Humidity = 18 % to IBM Watson
Published Soil_moisture = 34 % to IBM Watson
movement is detected
ultrasonic sensor turned ON
Published PIR output = 1 to IBM Watson
Published distance = 446 m to IBM Watson
Published Temperature = 13 C Humidity = 54 % to IBM Watson
Published Soil_moisture = 57 % to IBM Watson
Published Temperature = 45 C Humidity = 42 % to IBM Watson
|
Ln: 5 Col: 0
```



```

opentrial.py - C:\Users\spi\OneDrive\Desktop\opencvtrial\opentrial.py (3.7.0)
File Edit Format Run Plugins Window Help
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help

imp Published Temperature = 15 C Humidity = 35 % to IBM Watson
imp
imp Published Soil_moisture = 41 % to IBM Watson
imp
thr Published Temperature = 44 C Humidity = 82 % to IBM Watson
n=1 Published Soil_moisture = 15 % to IBM Watson
#Pr
org Water pump is turned ON
dev Published Temperature = 8 C Humidity = 55 % to IBM Watson
dev Published Soil_moisture = 15 % to IBM Watson
aut
dev Water pump is turned ON
dev movement is detected
aut ultrasonic sensor turned ON
dev Published PIR output = 1 to IBM Watson
dev Published distance = 7 m to IBM Watson
aut camera turned on
dev Alert! detected animal is elephant
dev LED is turned ON
aut Speaker is turned ON
dev Threat alert is send to IBM Watson
dev Published Intruded Animal is elephant to IBM Watson
aut Published Temperature = 25 C Humidity = 16 % to IBM Watson
dev Published Soil_moisture = 15 % to IBM Watson
dev Water pump is turned ON
aut
dev Published Temperature = 19 C Humidity = 16 % to IBM Watson
dev Published Soil_moisture = 2 % to IBM Watson
aut
dev
dev
dev authToken="TickHSlzaty_gxEtve"
# Initialize GPIO

```

IBM Watson Output:

IBM Watson IoT Platform

Browse Action Device Types Interfaces

Add Device +

Search by Device ID

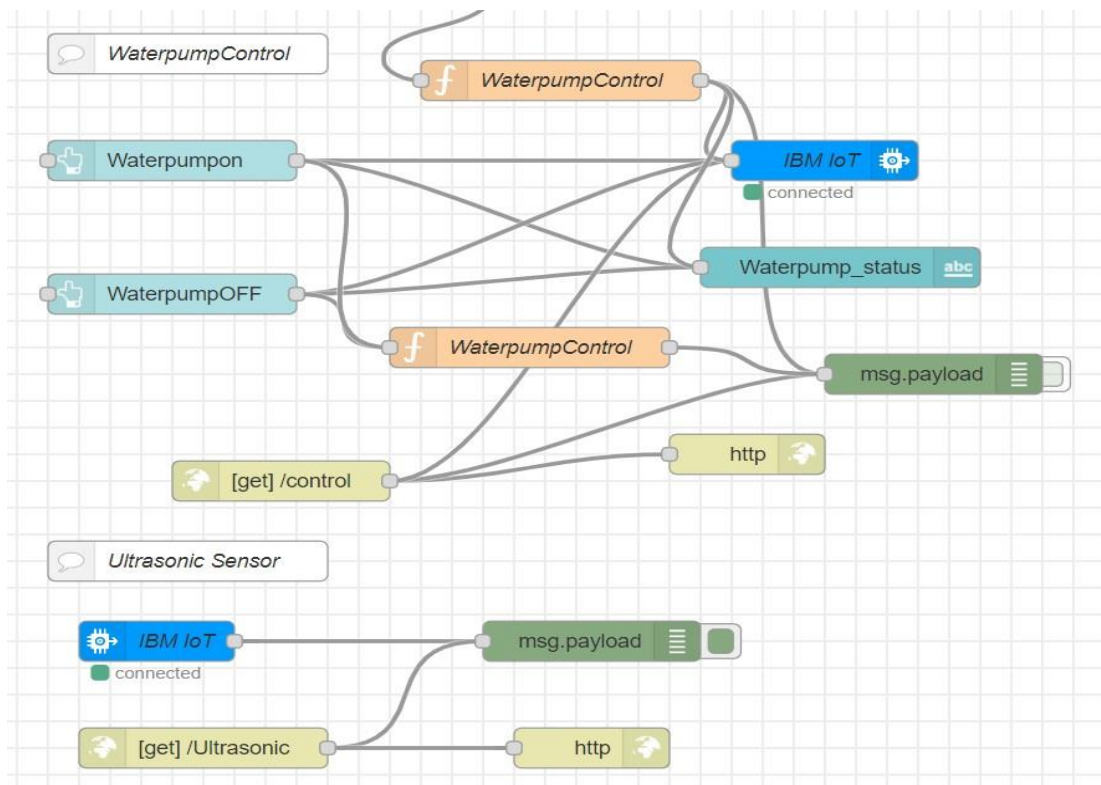
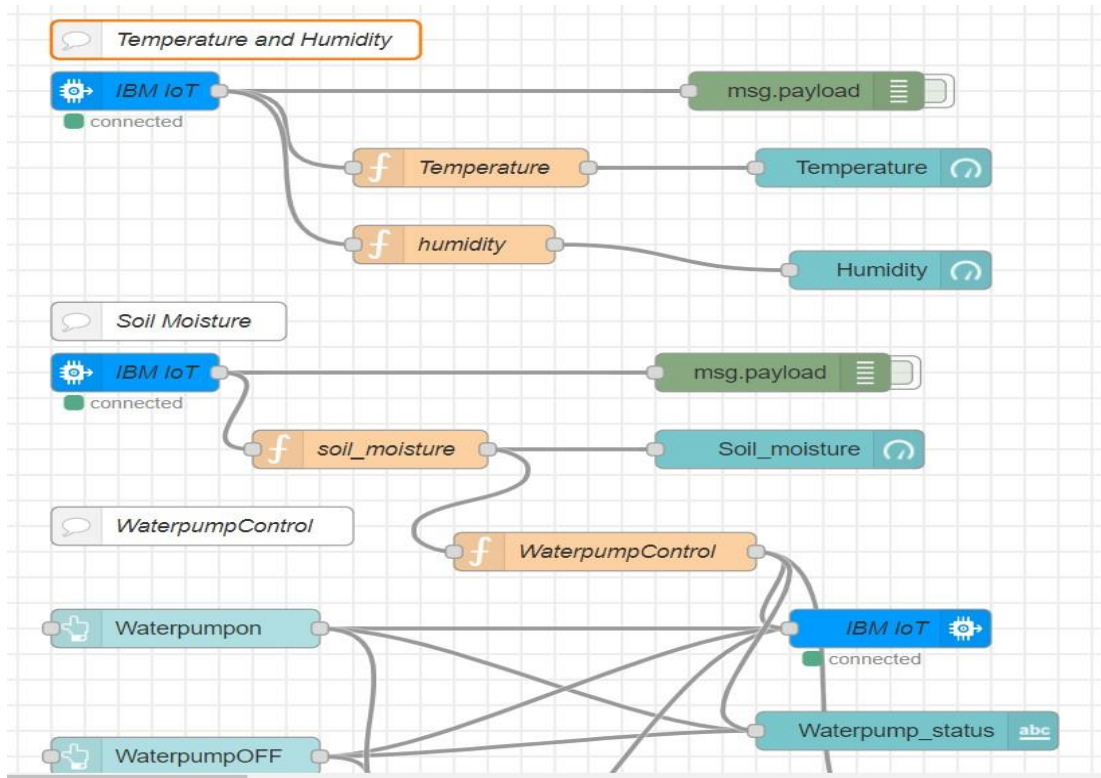
Device Simulator ☒

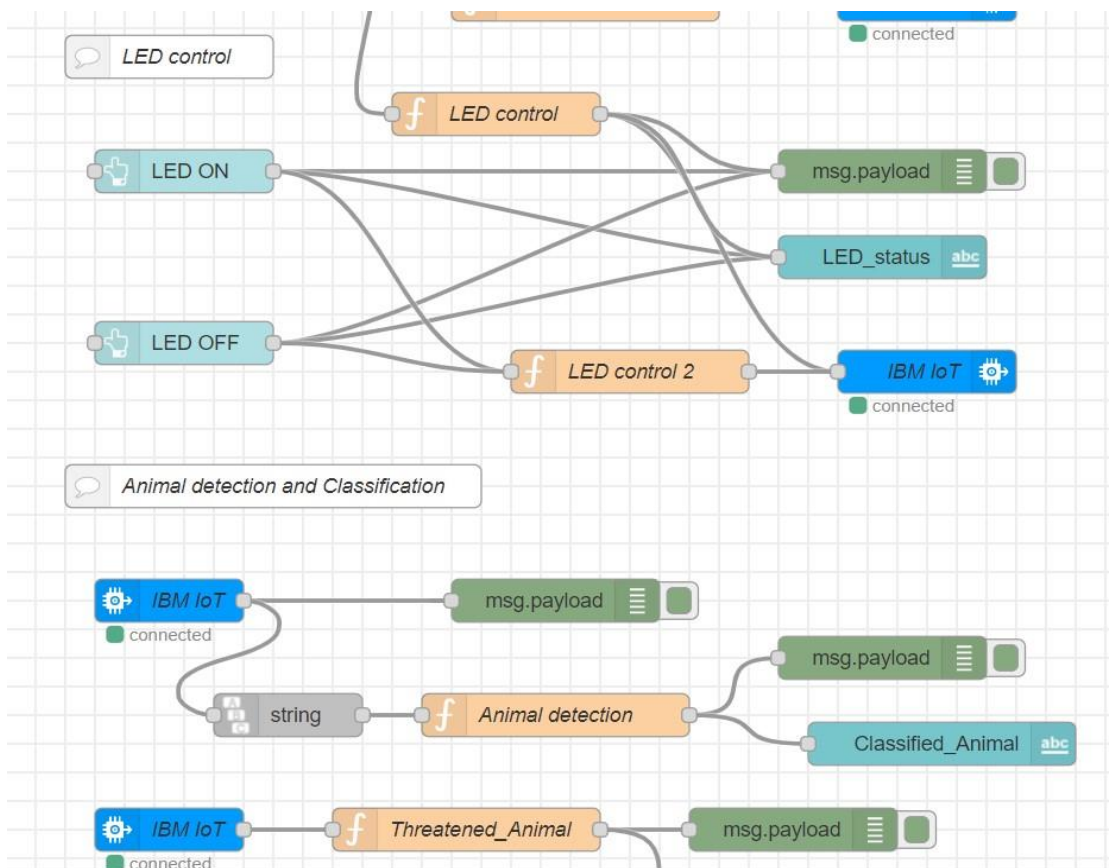
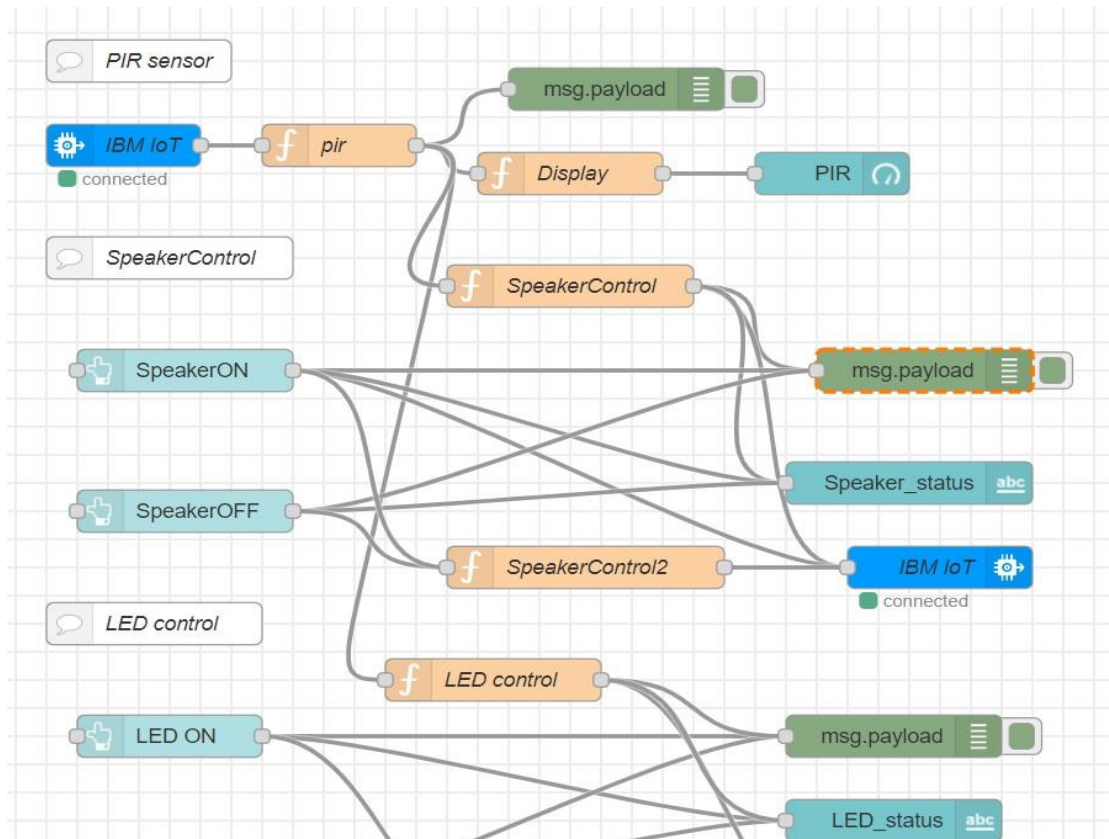
	Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
>	962819106039	Disconnected	device962819106039	Device	Oct 22, 2022 5:04 PM	
>	Camera	Connected	Detector	Device	Nov 12, 2022 9:13 PM	
>	DHT	Connected	Sensor	Device	Nov 6, 2022 10:29 PM	
>	LED	Connected	Output	Device	Nov 12, 2022 11:19 PM	
>	PIR	Connected	Sensor2	Device	Nov 12, 2022 1:06 PM	
>	Soil_moisture	Connected	Sensor1	Device	Nov 6, 2022 10:51 PM	
>	Speaker	Connected	Actuator	Device	Nov 16, 2022 8:32 PM	
>	Ultrasonic	Connected	Sensor3	Device	Nov 12, 2022 1:11 PM	
>	Water_pump	Connected	Actuator	Device	Nov 6, 2022 10:44 PM	

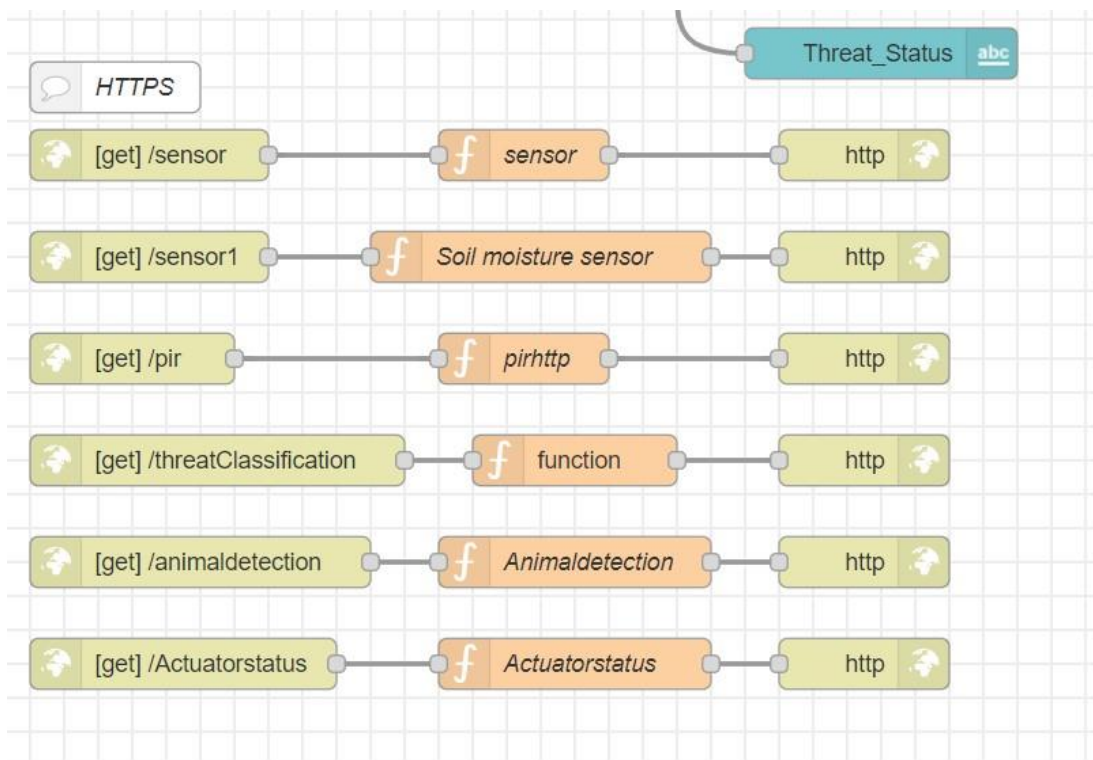
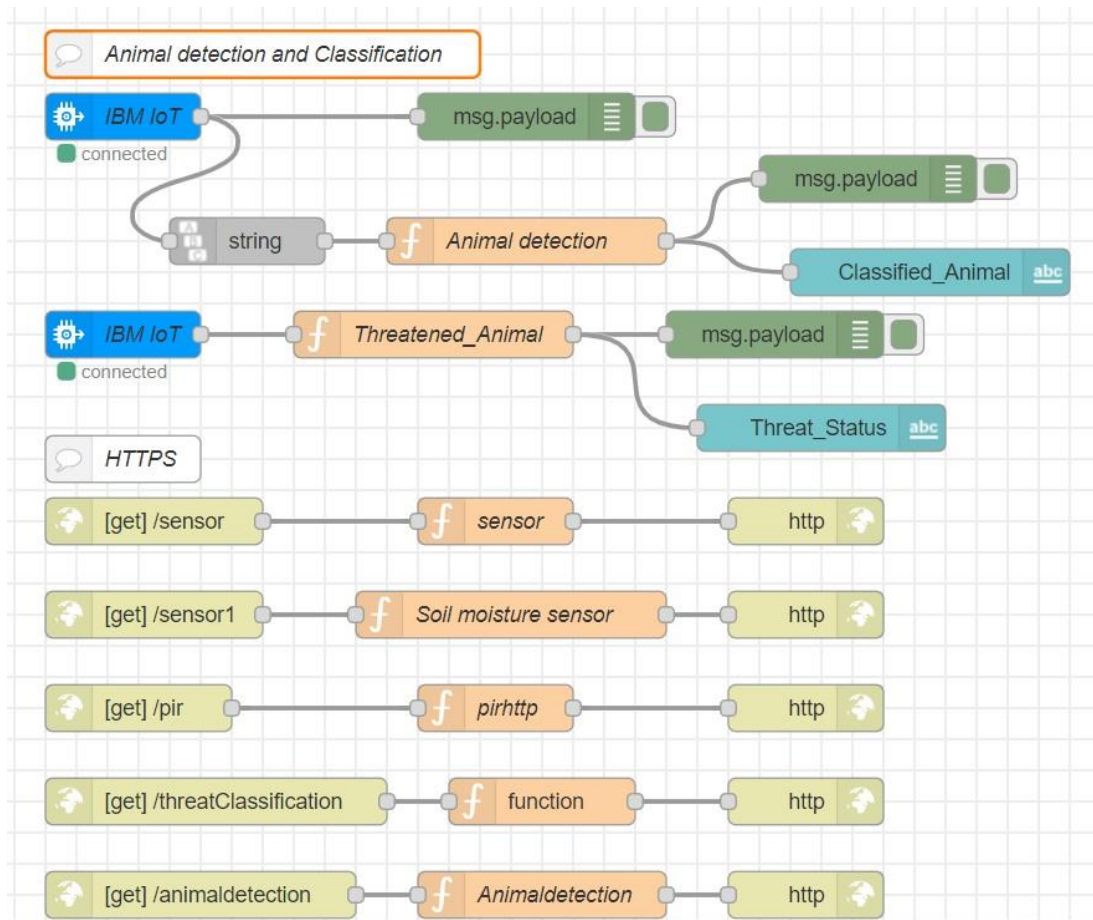
Items per page 50 | 1-9 of 9 Items

1 of 1 page < 1 >

Node Red Flow:







Node Red Web UI :

