**INTERNET OF THINGS(RSIP-2020)**

**PROJECT ON**

**SMART FIRE MANAGEMENT SYSTEM FOR INDUSTRIES**

**(Open CV) [Internship]**

**BATCH NO:03**

**BATCH NAME: MYTHICAL TECHIES**

* **J.Lakshmi Indrani(17481A0477)**
* **Ch.Yamini(17481A0441)**
* **B.Kusuma Naga Lalitha(17481A0420)**
* **A.Jyotsna(17481A0414)**

**1. INTRODUCTION**

**1.1 Overview**

* Purpose of the project
* Existing problem
* Proposed Solution
* Block Diagram
* Hardware/Software Designing
* Experimental Investigations
* Flowchart
* Result
* Pro's and con's
* Applications
* Conclusions
* Future Scope
* Bibilography

**1.2 Purpose**

The main purpose of our project is to prevent fire accidents and gas leakages in

industries . Recently this type of accident has been occured in vizag .To prevent such type

of accidents we came up with our project that is smart fire management system in

industries.

Our project includes flame detection , gas detection and high temperature

detection so that it can easily detect any fire or gas leakages. Based on temperature

readings if there is any high temperature or gas leakages the exhaust fans and sprinklers

will be turned ON. Continously video streaming will be done and if any fire is detected

automatically the camera starts capturing the images which can be used for further

investigations. The captured images will be stored in database.

**2 . LITERATURE SURVEY**

**2.1 Existing problem**

The problem with the existing systems are they may not contain all

the detectors such as high temperature detectors,gas leakage detectors . In present

systems it only includes fire detectors which detects the fire after happening of the

fire accidents this may cause damages or it may also leads to loss of human life.

In present systems it doesn't include video monitoring so that we can

not identify that the fire is detected . Fire accidents in industries has an impact on the

safety of workers, those living in the vicinity of the installation and on the environment.

**2.2 Proposed Solution**

In today's world,those responsible for life safety and property protection

face a wide array of threats,dangers and emergencies-from fire accidents.

An integrated solution can help optimize system performance and

functionality, thereby enhancing your life safety and property protection program. A

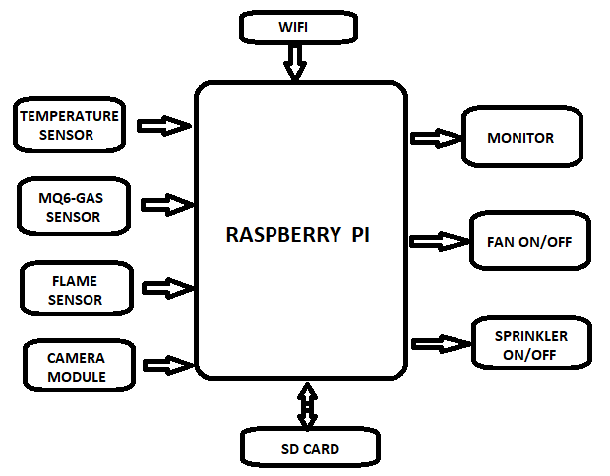
control center can enable system operators to view continuous monitering. The

integration of access control with the fire management system supports safe

evacuation.

**3 Theoretical Analysis**

**3.1 Block Diagram**



**3.2 Hardware/Software Designing**

In the software designing of our project the temperature,humidityand

gas values are continously generated. Here we used an IBM cloud service to create

a internet of things platform . In this platform we have created a virtual raspberrypi

device. After the device creation we will be provided with device credentials.

These credentials are uesd to integrate the device with python code.

Then by using node platform we integrated the device with node red

and created a webapp which continously displays the values. Using MIT app inventor

we have designed a mobile app for continous monitoring and alerts.

**4 Experimental Investigation**

In this project firstly we have created an IBM cloud account. After

the creation of our account we have created a Internet of Things platform in which

we also created a vitual raspberrypi device. Then we have written a python code in

python IDLE and connected the Internet of things platform . Then we have created

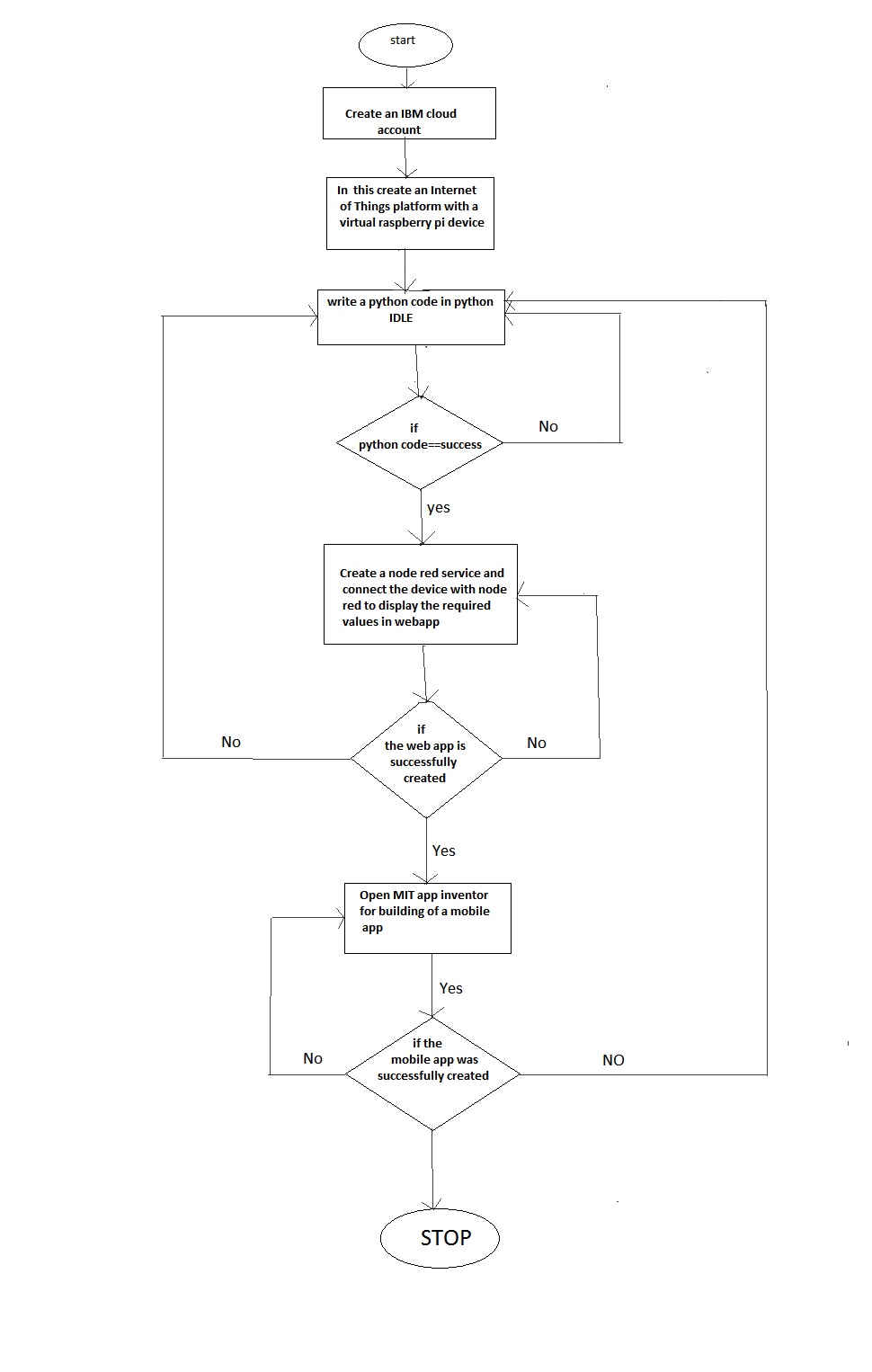
a node red service and designed a webapp.

Finally we have created a mobile app by using MIT app inventor.

This will helps us to view the temperature ,humidity and gas values . It also includes

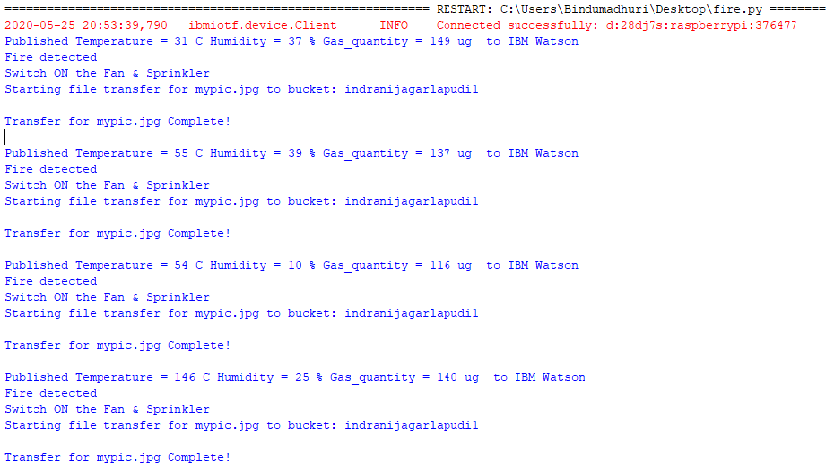
some buttons which helps us to control the device from mobile

**5 FLOWCHART**

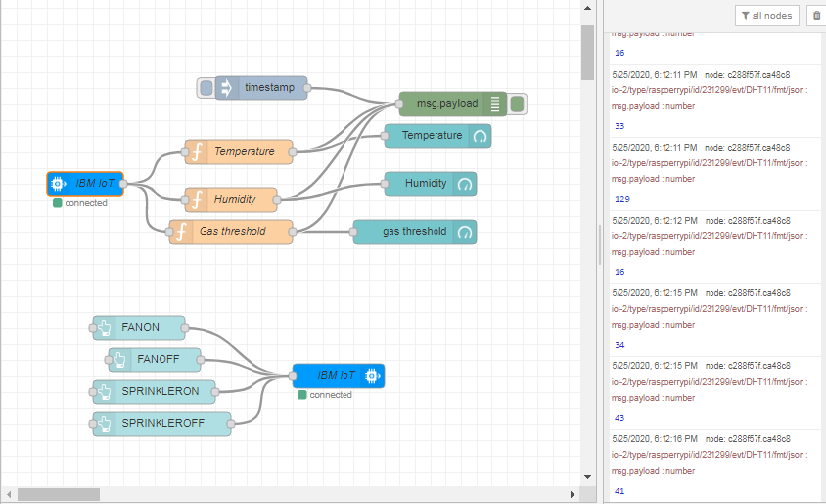


**6 RESULT**

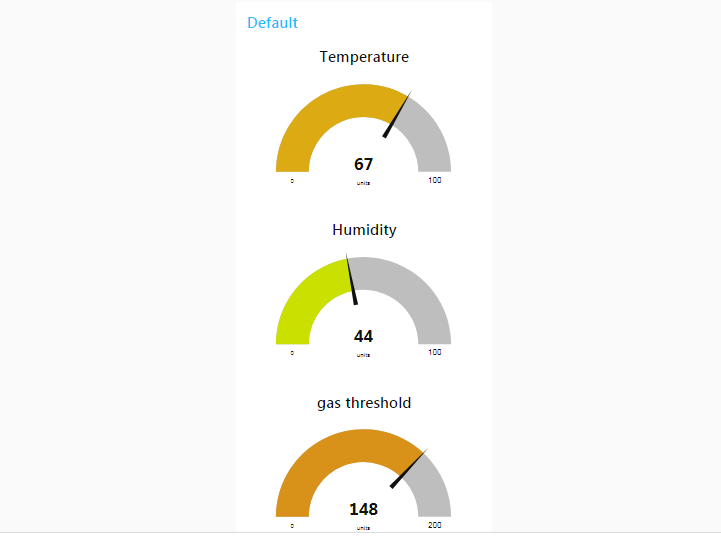
**Python code Output:**



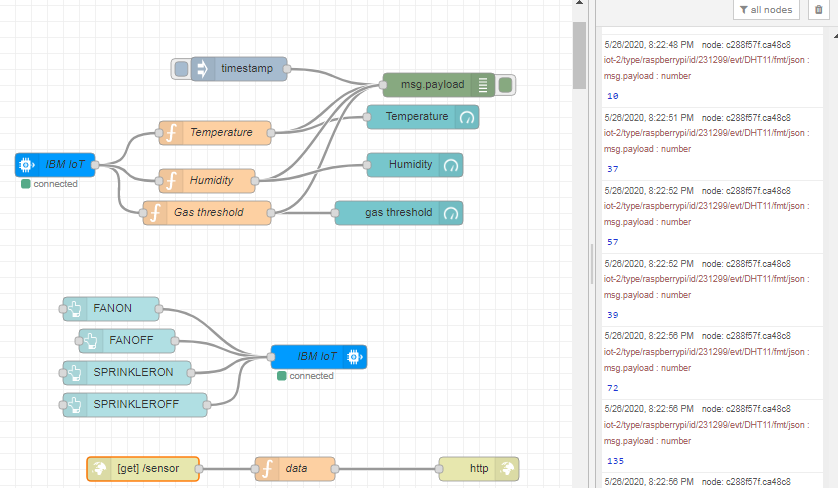
**Node-Red Flow:**



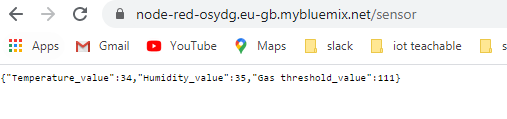
**Node Red UI:**



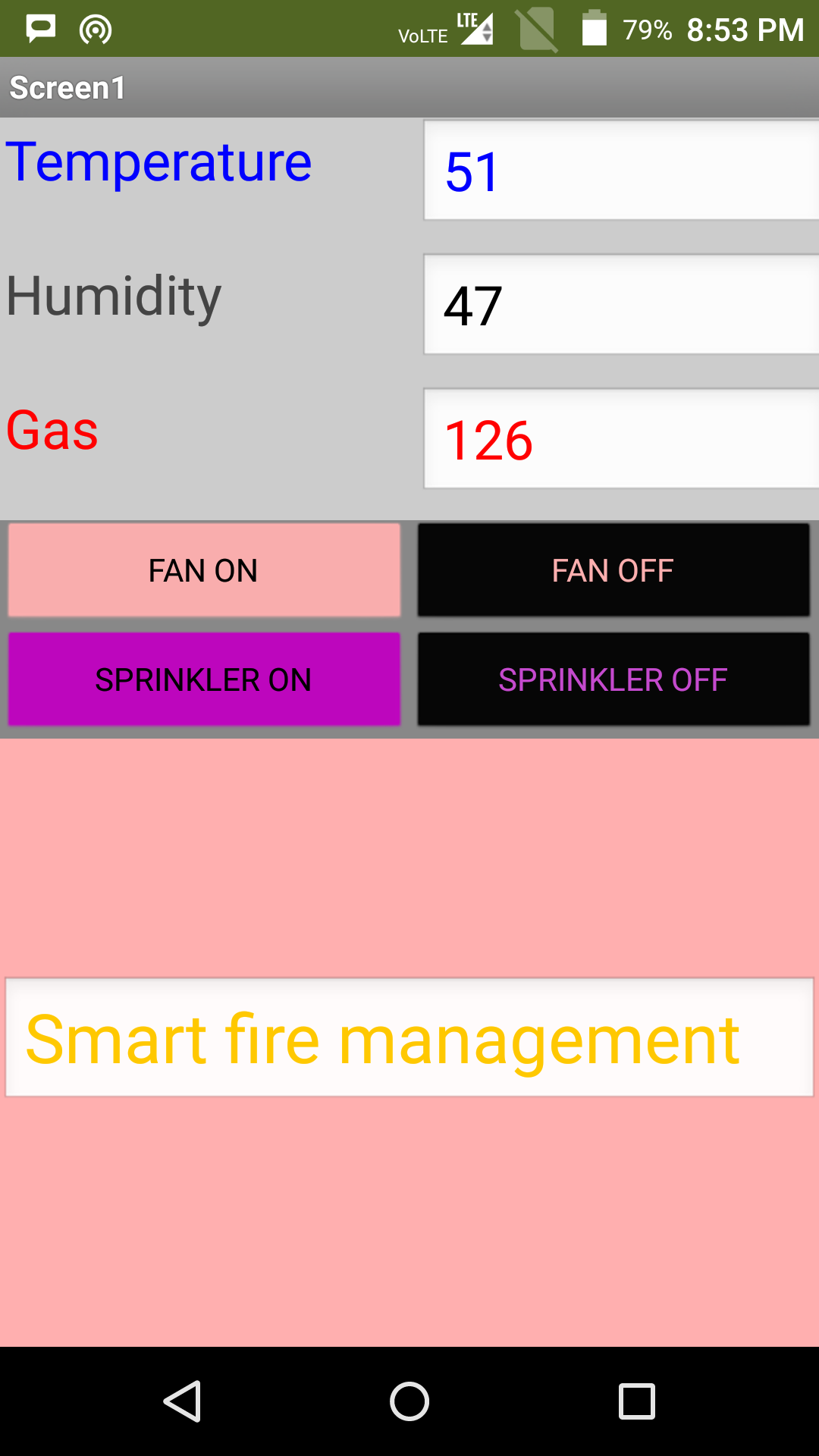
**HTTP Node-Red Flow:**



**HTTP Page:**



**MIT App:**



**7 PRO'S AND CON'S**

**Advantages:**

1. These systems are highly cost effective.

2. It gives maximum safety for industries.

3. These can avoid major damages or destructions in industries.

**Disadvantages:**

1. cannot stop electric fires.

2. There may be a huge damage incase of improper working of system.

3. The installation of these systems is expensive.

**9 CONCLUSION**

The project depicts the neccesity and an efficient solution for fire

safety. Internet of Things was the main concept used and the project mainly

builds on the techniques which are already present and also it has overcome

many obstacles present in the previous systems. But still there are few

disadvantages and some improvements are needed for more efficient working.

**10 FUTURE SCOPE**

The designed fire management system is simple but it has wide area of

applications in industrial safety especially in developing countries. Using this

system quick and reliable alert is posible to initiate preventive measures to avert

danger of fire hazards and minimize losses of life and property. Large industrial

or residential area can be monitored through the proposed system installing

multiple modules. The ystem can be further developed with added features.

**11 BIBILOGRAPHY**

[https://cloud.ibm.com](https://cloud.ibm.com/)

<https://node-red-osydg.eu-gb.mybluemix.net/ui/#!/0?socketid=efwocTVpSTd6U3JzAAA1>

<https://node-red-osydg.eu-gb.mybluemix.net/sensor>

<http://appinventor.mit.edu/>

**APPENDIX**

**A. Source code**

import time

import sys

import ibmiotf.application

import ibmiotf.device

import random

import cv2

import numpy as np

import datetime

import ibm\_boto3

from ibm\_botocore.client import Config, ClientError

#video\_file = "video\_1.mp4"

video = cv2.VideoCapture(0)

# Constants for IBM COS values

COS\_ENDPOINT = "https://s3.jp-tok.cloud-object-storage.appdomain.cloud" # Current list avaiable at https://control.cloud-object-storage.cloud.ibm.com/v2/endpoints

COS\_API\_KEY\_ID = "pysZjjSmGSalxoPvSuxBDxcwfJTXkO75Sk7tOoPEzJwi" # eg "W00YiRnLW4a3fTjMB-odB-2ySfTrFBIQQWanc--P3byk"

COS\_AUTH\_ENDPOINT = "https://iam.cloud.ibm.com/identity/token"

COS\_RESOURCE\_CRN = "crn:v1:bluemix:public:cloud-object-storage:global:a/dca1c353e5f94823b188880cafdd1179:d0ac4f37-dc9e-480b-bf74-d6d5295c398c::" # eg "crn:v1:bluemix:public:cloud-object-storage:global:a/3bf0d9003abfb5d29761c3e97696b71c:d6f04d83-6c4f-4a62-a165-696756d63903::"

# Create resource

cos = ibm\_boto3.resource("s3",

ibm\_api\_key\_id=COS\_API\_KEY\_ID,

ibm\_service\_instance\_id=COS\_RESOURCE\_CRN,

ibm\_auth\_endpoint=COS\_AUTH\_ENDPOINT,

config=Config(signature\_version="oauth"),

endpoint\_url=COS\_ENDPOINT

)

#Provide your IBM Watson Device Credentials

organization = "28dj7s"

deviceType = "raspberrypi"

deviceId = "376477"

authMethod = "token"

authToken = "12345678"

# Initialize GPIO

def myCommandCallback(cmd):

print("Command received: %s" % cmd.data)

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()

def multi\_part\_upload(bucket\_name, item\_name, file\_path):

try:

print("Starting file transfer for {0} to bucket: {1}\n".format(item\_name, bucket\_name))

# set 5 MB chunks

part\_size = 1024 \* 1024 \* 5

# set threadhold to 15 MB

file\_threshold = 1024 \* 1024 \* 15

# set the transfer threshold and chunk size

transfer\_config = ibm\_boto3.s3.transfer.TransferConfig(

multipart\_threshold=file\_threshold,

multipart\_chunksize=part\_size

)

# the upload\_fileobj method will automatically execute a multi-part upload

# in 5 MB chunks for all files over 15 MB

with open(file\_path, "rb") as file\_data:

cos.Object(bucket\_name, item\_name).upload\_fileobj(

Fileobj=file\_data,

Config=transfer\_config

)

print("Transfer for {0} Complete!\n".format(item\_name))

except ClientError as be:

print("CLIENT ERROR: {0}\n".format(be))

except Exception as e:

print("Unable to complete multi-part upload: {0}".format(e))

while True:

hum=random.randint(10,50)

#print(hum)

temp =random.randint(30,150)

gas=random.randint(40,150)

#Send Temperature & Humidity to IBM Watson

data = { 'Temperature' : temp, 'Humidity': hum , 'Gas\_threshold': gas}

#print (data)

def myOnPublishCallback():

print ("Published Temperature = %s C" % temp, "Humidity = %s %%" % hum, "Gas\_quantity = %s ug "%gas, "to IBM Watson")

success = deviceCli.publishEvent("DHT11", "json", data, qos=0, on\_publish=myOnPublishCallback)

if not success:

print("Not connected to IoTF")

time.sleep(2)

deviceCli.commandCallback = myCommandCallback

(grabbed, frame) = video.read()

if not grabbed:

break

blur = cv2.GaussianBlur(frame, (21, 21), 0)

hsv = cv2.cvtColor(blur, cv2.COLOR\_BGR2HSV)

lower = [18, 50, 50]

upper = [35, 255, 255]

lower = np.array(lower, dtype="uint8")

upper = np.array(upper, dtype="uint8")

mask = cv2.inRange(hsv, lower, upper)

output = cv2.bitwise\_and(frame, hsv, mask=mask)

no\_red = cv2.countNonZero(mask)

cv2.imshow("output", output)

#print("output:", frame)

if ((int(no\_red) > 20000)|(temp>=70)|(gas>=80)):

print ('Fire detected')

print ('Switch ON the Fan & Sprinkler')

else:

print ('Fire not detected')

#capture the first frame

check,frame=video.read()

gray=cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)

#drawing rectangle boundries for the detected fire

cv2.imshow('Fire detection', frame)

global picname

picname=datetime.datetime.now().strftime("%y-%m-%d-%H-%M")

cv2.imwrite(picname+".jpg",frame)

multi\_part\_upload("indranijagarlapudi1","mypic.jpg",picname+".jpg")

#print(int(no\_red))

#print("output:".format(mask))

if cv2.waitKey(1) & 0xFF == ord('q'):

break

cv2.destroyAllWindows()

video.release()

# Disconnect the device and application from the cloud

deviceCli.disconnect()