# INTRUDER DETECTION AND ALERTING SYSTEM

Team -C11

PRESENTED BY:

19485A0424-A.Kavya Sri

18481A04G0- P.Siva Shankar

18481A04G1- P.Mahesh Babu

18481A04G2-P.Divya Koti

18481A04G3-P.Dhanush Raj

# 1.Introduction:

## a. Overview:

House is a residential building, an asset, as well as a place to store wealth. Therefore, security becomes one of mandatory considerations in keeping the house from undesirable events or accidents.

This is to design and implement a home security system with human detection capability. Traditional home security systems, i.e., Closed-Circuit Television (CCTV) can only capture and record videos without the ability of giving warning feedback if there is any suspicious object. Therefore, an additional object detection and warning method is required.

Internet of Things (IoT) is a network of indata without interference or with minimal human intervention. This technology has been widely used for smart city application, personal health monitoring, manufacturing and smart lighting.An IoT system for monitoring the presence of intruders in a house using the combination of motion detection and object recognition.After the motion of object is detected, the web camera takes the picture of the suspicious spot.

## b. Purpose:

Intruder detection and alerting system captures and alerts the user when the theives or intruders entered through back end of the house, whereas cctv can also captures but not able to alert the user in which loss has been occurred and then the user came to know that. As we all know that "prevention is better than cure". The cctv is compared with cure and this alerting system is compared with prevention as it alerting the user before the loss has been occurred.

# 2.Literature Survey:

## a. Existing Problem:

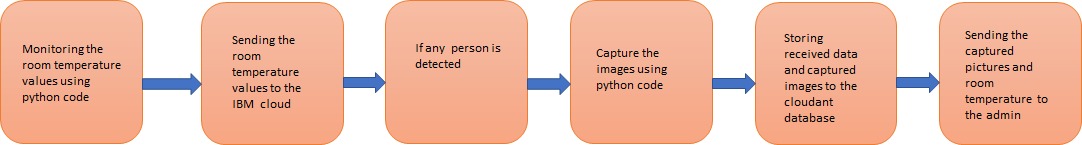
CCTV is a device for monitoring the situation around an office area, house and building. CCTV is also useful for monitoring the situation around a house, both when the residents are at home and when they are not at home. Despite the benefits, there are some problems related to the use of CCTV. Firstly, CCTV does not produce any notification and warning whenever it captures any suspicious object. Secondly, CCTV streams continuously to capture events that occur in the home environment even when there is no suspicious object or activity. Therefore, the streaming requires huge consumption of bandwidth and storage media due to the continuous video streaming and storing.

## b. Proposed Solution:

To avoid the problem, Intruder Detection and Alerting System is proposed. Through this, as per our requirement the device is on/off. If any intruder is detected, the image will be sent to the user as an alert.

# 3.Theoritical Analysis:

## a.Block diagram:



## b.Hardware/Software designing:

We follow a step-by-step procedure to set up all the interfaces required for our project and develop the code in python to send random sensor data to the cloud. The following software is required:

1.Python Idle (with specified packages installed)

2.IBM cloud

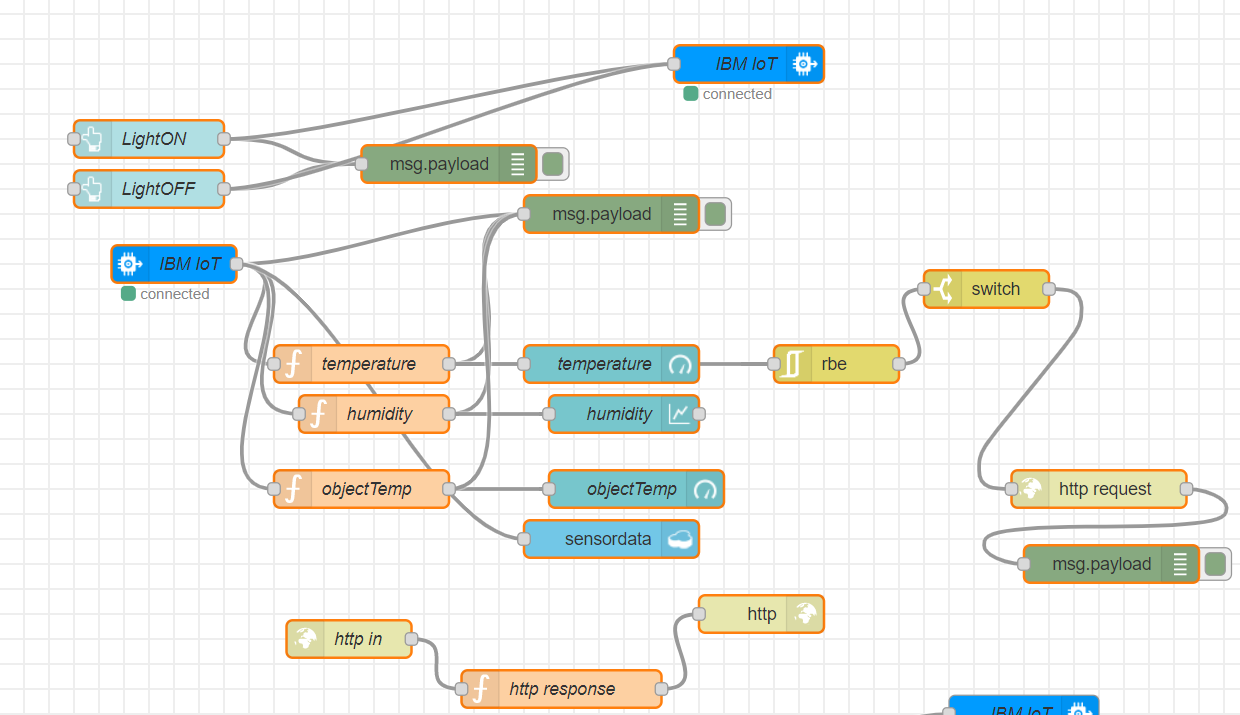
3.Node Red service

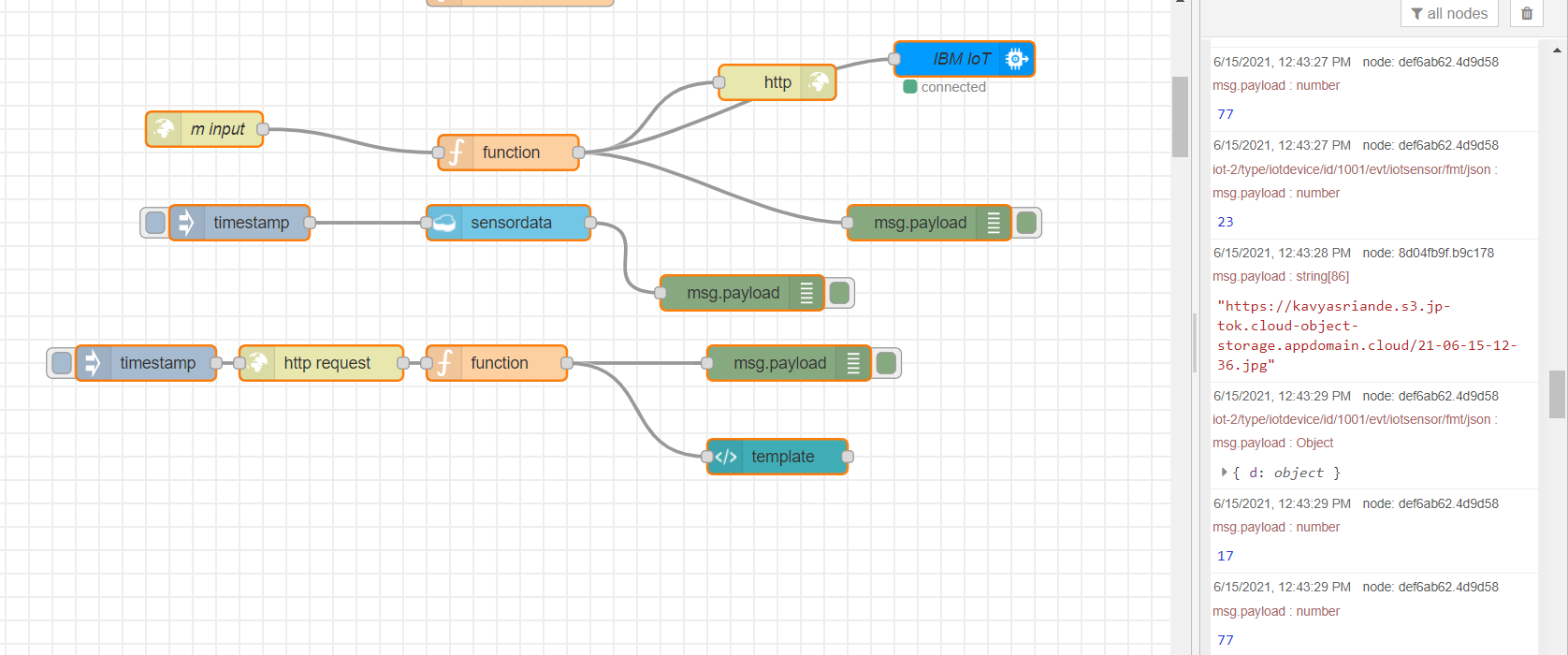
# 4.Experimental investigations:

To successfully detect the person and motion of that person , some sensors are placed in this sysem. Since hardware is not available ,  a python code snippet is used to generate random values such as temperature and to capture the face of a person. This randomly generated data is sent to IBM IoT platform.

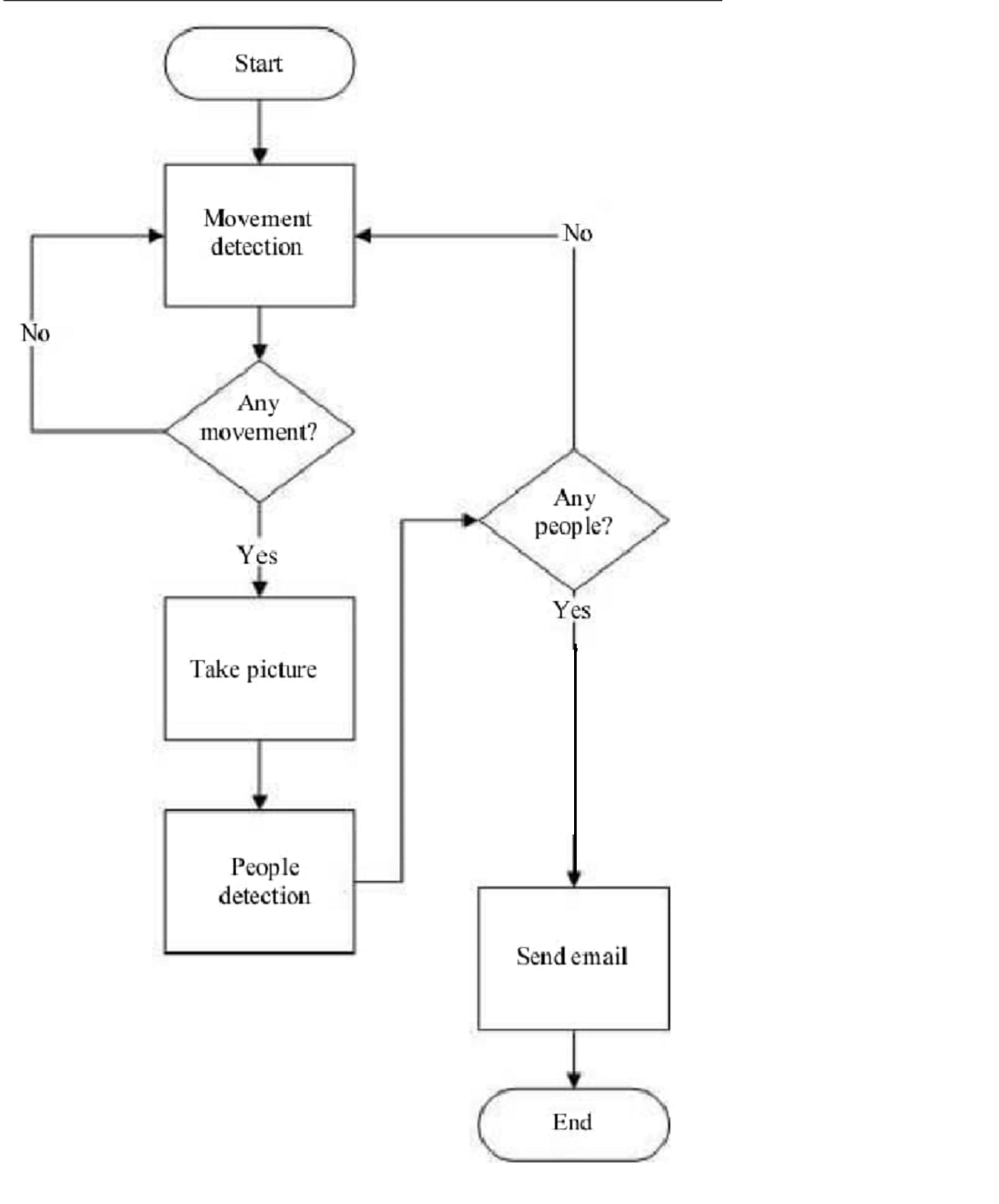
Using node red service , a node red flow is created which retrieves the data from IBM IoT platform.So, the node red gives the parameters like temperature, and whenever a person is detected camera captures the image of a person. The photo is stored by the system in the temporary folder, namely directory /home/pi/Home Security/images. The original color photo is transformed into a grayscale one. The trick is to take all the pixels in the image then from the color of each pixel, information about the 3 basic colors, namely red, blue and green, is obtained. These three basic colors will be added, then divided by three so that the average values are added. This average value will be used to give color to the pixels of the image so that the color becomes grayscale. The received data is stored in cloudant base and sent to the user via email.

In this project, switching on and off the lights through the web app is also implemented.



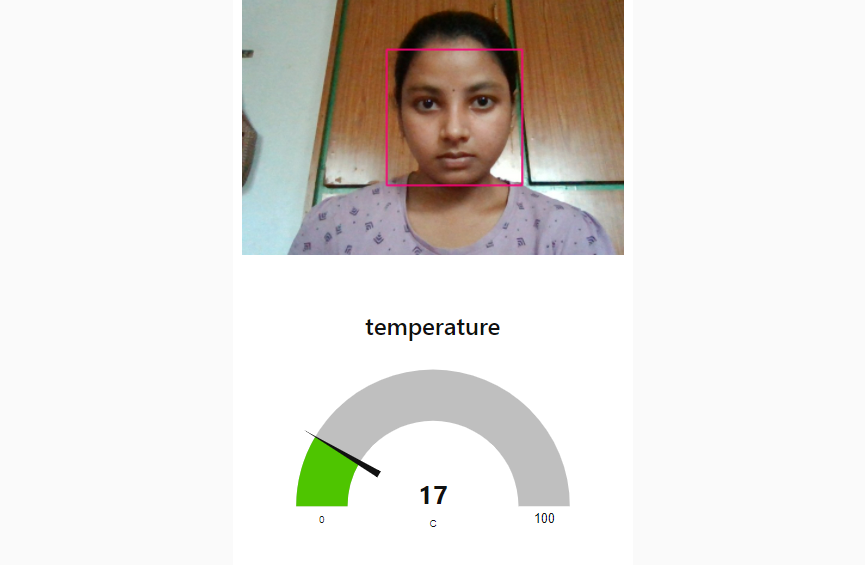


# 5.Flow chart:



# 6.Result:

The result of this project fulfills our purpose of detecting the intruder and alerting the user by sending the image of intruder through IBM iot cloud platform. So that there is a chance of an immediate action for the user on intruder.



# 7. Advantages and disadvantages:

## a.Advantages:

* Easier to use and maintain.
* An immediate notification of intruder image to the user.
* More efficient than CCTV.

## B.Disadvantages:

* In case of power failure it may not work.
* In case of poor network connection it may not alert the user.
* In case of worst climatic conditions the camera may not capture properly.

# 8. Applications:

* Personal health monitoring.
* Smart City and smart homes.
* Banks during night times.

# 9.Conclusion:

Home security plays an important role in terms of wealth or in any other way.Even the CCTV footage cannot alert the user that there is an intuder entering, and the continuous streaming is useless and which causes maximum band width. So this system is activated only when the intruder is entered and immediately captures and sends the image of an intruder, so that the user can be alerted through this intruder detection and alerting system.

# 10.Future scope:

This project has been used beyond at homes and banks as it increases the security and plays a major role of alerting the user before the loss has been occurred.

# 11.Bibliography:

https://youtu.be/1ZmJUHcqjXE

# 12.Appendix:

## a.Source code:

## import datetime

## import ibm\_boto3

## from ibm\_botocore.client import Config, ClientError

## import cv2

## import numpy as np

## import sys

## import ibmiotf.application

## import ibmiotf.device

## import random

## import time

## from cloudant.client import Cloudant

## from cloudant.error import CloudantException

## from cloudant.result import Result, ResultByKey

## #Provide your IBM Watson Device Credentials

## organization = "ykmjw0"

## deviceType = "iotdevice"

## deviceId = "1001"

## authMethod = "token"

## authToken = "1234567890"

## def myCommandCallback(cmd):

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

## print(cmd.data['command'])

## 

## if(cmd.data['command']=="open"):

## print("door open")

## 

## if(cmd.data['command']=="close"):

## print("door close")

## 

## if(cmd.data['command']=="lighton"):

## print("light on")

## 

## if(cmd.data['command']=="lightoff"):

## print("light off")

## 

## if(cmd.data['command']=="fanon"):

## print("fan on")

## if(cmd.data['command']=="fanoff"):

## print("fan off")

## 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()

## face\_classifier=cv2.CascadeClassifier("haar-face.xml")

## #It will read the first frame/image of the video

## video=cv2.VideoCapture(0)

## 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 = "UnNzqTZspDeKxls0pFOnKSGJqY5SkB21SGnYAjfddVCo" # 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/96cc5eebcff045d3a9f1692d6c990c84:9e4c25d8-daa2-4ce2-b6a2-e4e7eeb8bf78::s" # eg "crn:v1:bluemix:public:cloud-object-storage:global:a/3bf0d9003abfb5d29761c3e97696b71c:d6f04d83-6c4f-4a62-a165-696756d63903::"

## client = Cloudant("apikey-v2-2yedcnk14s51dz107nyihcsysmnqme9zf6n6krz5lh7a", "9491a52e99289c069785314e361e9d9f", url="https://apikey-v2-2yedcnk14s51dz107nyihcsysmnqme9zf6n6krz5lh7a:9491a52e99289c069785314e361e9d9f@f9c4774e-7384-4712-b5c0-767673786baf-bluemix.cloudantnosqldb.appdomain.cloud")

## client.connect()

## database\_name = "newdatabase"

## # 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

## )

## 

## 

## 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:

## 

## #capture the first frame

## check,frame=video.read()

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

## #detect the faces from the video using detectMultiScale function

## faces=face\_classifier.detectMultiScale(gray,1.3,5)

## 

## #drawing rectangle boundries for the detected face

## for(x,y,w,h) in faces:

## cv2.rectangle(frame, (x,y), (x+w,y+h), (127,0,255), 2)

## cv2.imshow('Face detection', frame)

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

## picname=picname+".jpg"

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

## cv2.imwrite(picname,frame)

## person=1

## my\_database = client.create\_database(database\_name)

## multi\_part\_upload("kavyasriande",picname,pic+".jpg")

## if my\_database.exists():

## print("'{database\_name}' successfully created.")

## json\_document = {

## "\_id": pic,

## "link":"https://kavyasriande.s3.jp-tok.cloud-object-storage.appdomain.cloud/"+picname

## }

## new\_document = my\_database.create\_document(json\_document)

## if new\_document.exists():

## print("Document '{new\_document}' successfully created.")

## time.sleep(1)

## t=34

## h=45

## data = {"d":{ 'temperature' : t, 'humidity': h, 'person': person}}

## #print data

## def myOnPublishCallback():

## print ("Published data 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

## person=0

## #waitKey(1)- for every 1 millisecond new frame will be captured

## Key=cv2.waitKey(1)

## if Key==ord('q'):

## #release the camera

## video.release()

## #destroy all windows

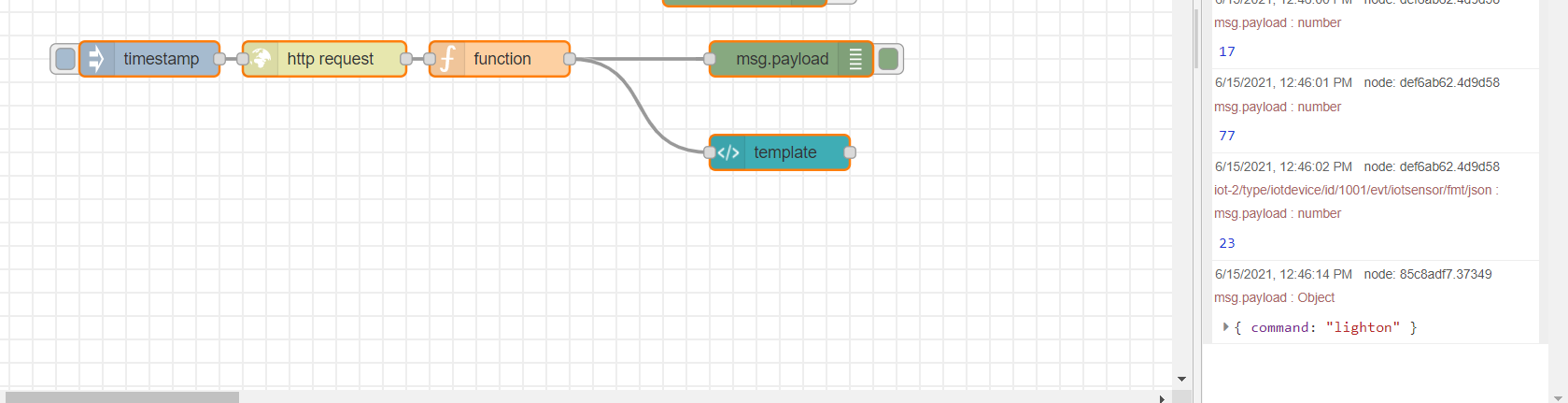
## cv2.destroyAllWindows()

## break

## deviceCli.disconnect()

## 

## b.UI Output:

When the light ON is pressed in UI:

When the light OFF is pressed in UI:

