**IOT Based Automated Attendance System Based on Face Recognition**

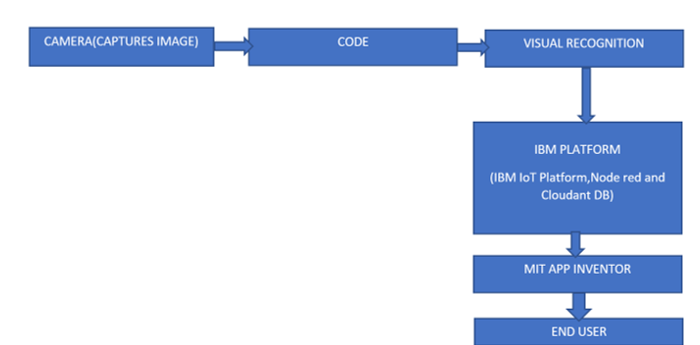
**Overview :** This is the IoT based automatic attendance system built using the IBM cloud platform,MIT app inventor,Fast2SMS and IDLE(Python).

**Purpose** : To properly execute the attendance process and store the information carefully,so that it can be later accessed by the institution and parents.

**Existing Problem** :Due to the manual attendance process in the educational system, now a days, there is a chance of manipulation of attendance data and also proxy.

**Proposed Solution** : By building this smart attendance system,this manipulation can be avoided as the person(student/faculty) is identified using Face Recognition and even data is stored securely,which can be accessed later.

**Block Diagram** :



**Hardware/Software Required** :

**IBM**: Acts as a platform between various services and is used to store and retrieve data.

**NODE-RED:**It is a development tool provided by IBM.It helps to integrate the hardware devices and application.

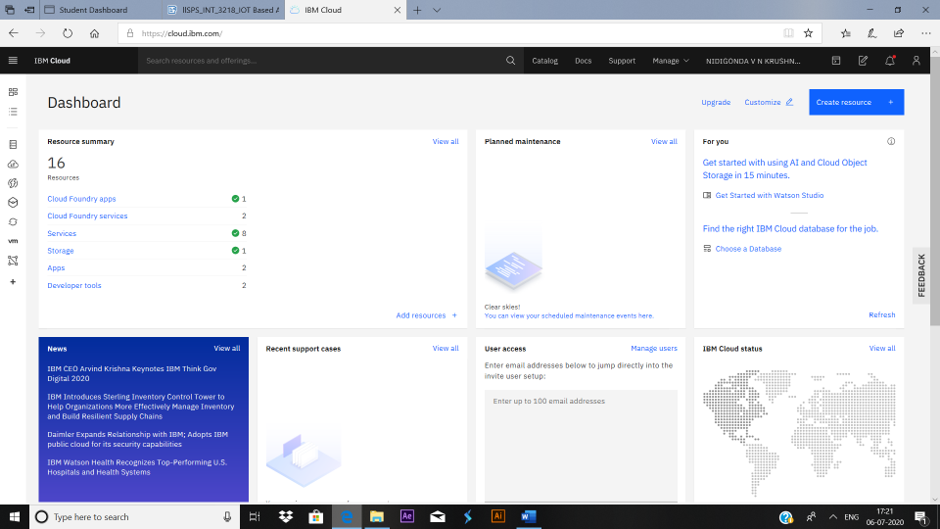
**FAST2SMS :** It helps us in sending the alerts/messages to multiple users at same instance of time.

**MIT APP INVENTOR:**It is an open-source platform which helps to create application software(apps) for android and ios.

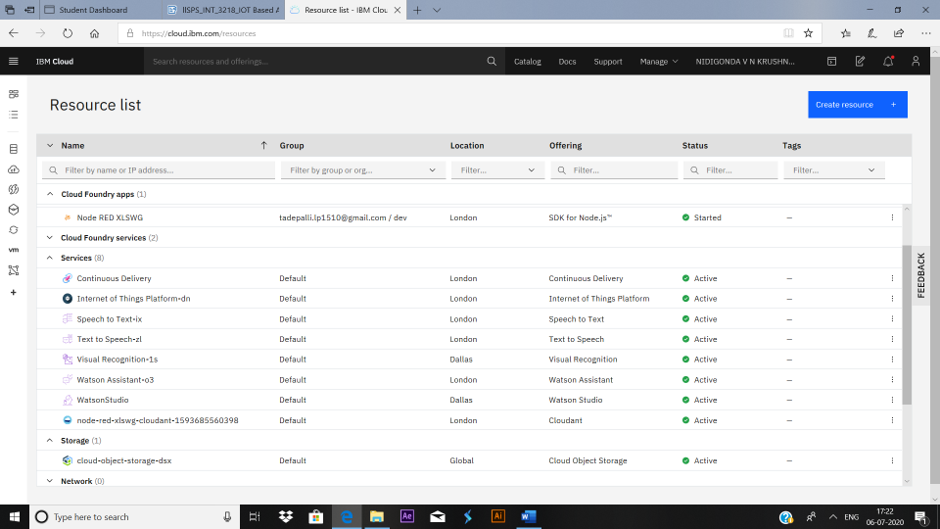
**EXPERIMENTAL INVESTIGATIONS AND THE PROJECT:**

First we train our visual recognition model to detect the features in the faces of students and staff by creating a custom model.Then by using opencv in python code, images are captured and when a face is detected, the image is used to check whether it is present in custom model or not by providing classifier id.If the image is detected , we update the attendence of that particular person in cloudant db,and also we pass this data to ibm iot platform.At the same time we create a node-red flow to display the attendence details in debug section. A ui is also created with button named 'DETAILS' which when pressed will display attendence records in python shell.A http request is also created which stores the data in json format,later this url is used in application development.Now by using MIT app inventor we configure a application,which displays the attendence details.And if the student was absent then message alerts will be sent to parents as well as class incharge by using fast2sms.

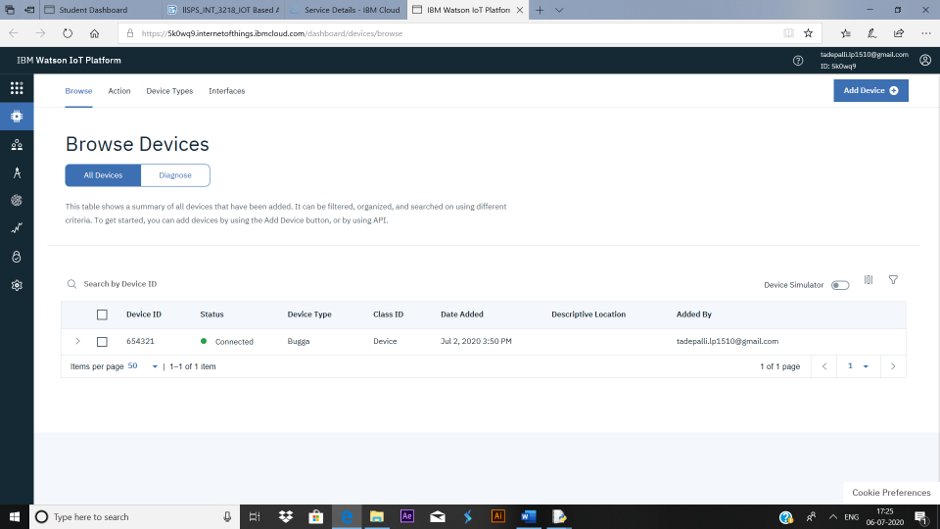
**IBM CLOUD PLATFORM :**

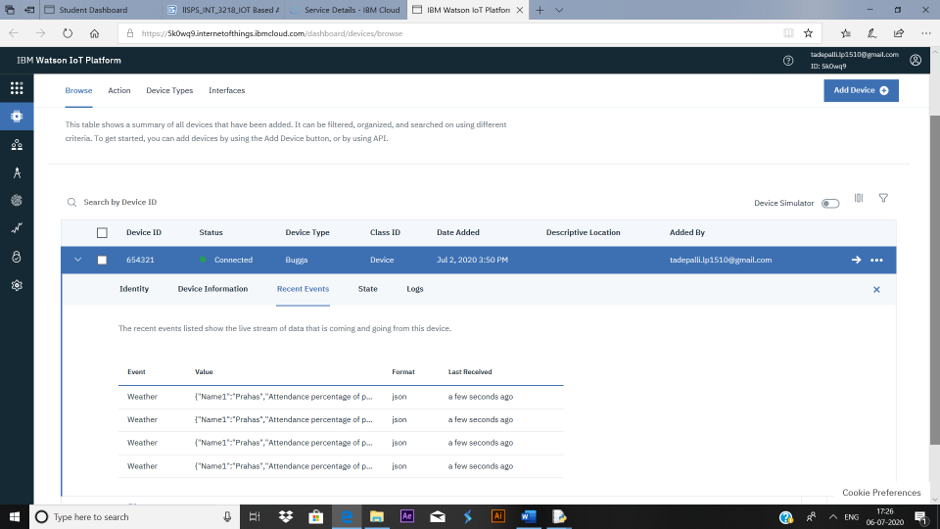


**LIST OF SERVICES,SOFTWARES AND STORAGES CREATED IN IBM PLATFORM :**

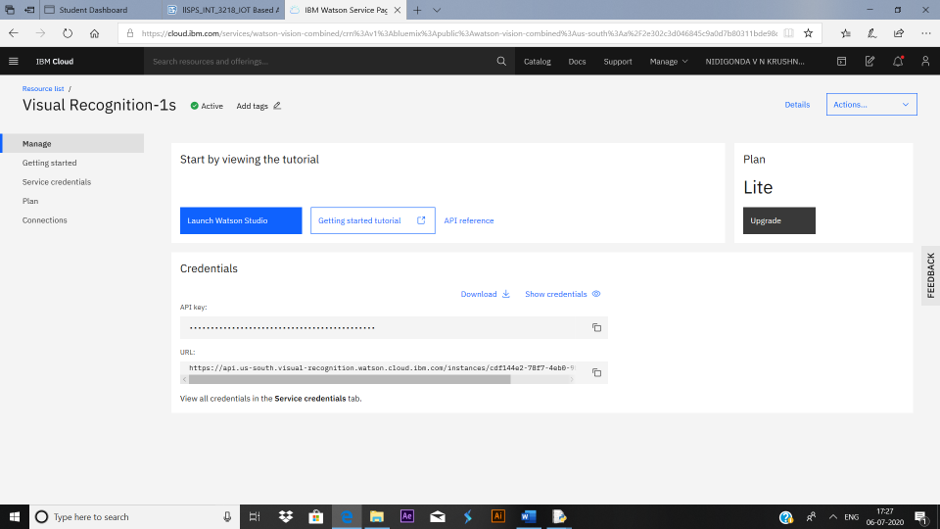


**IBM IOT SERVICE :**

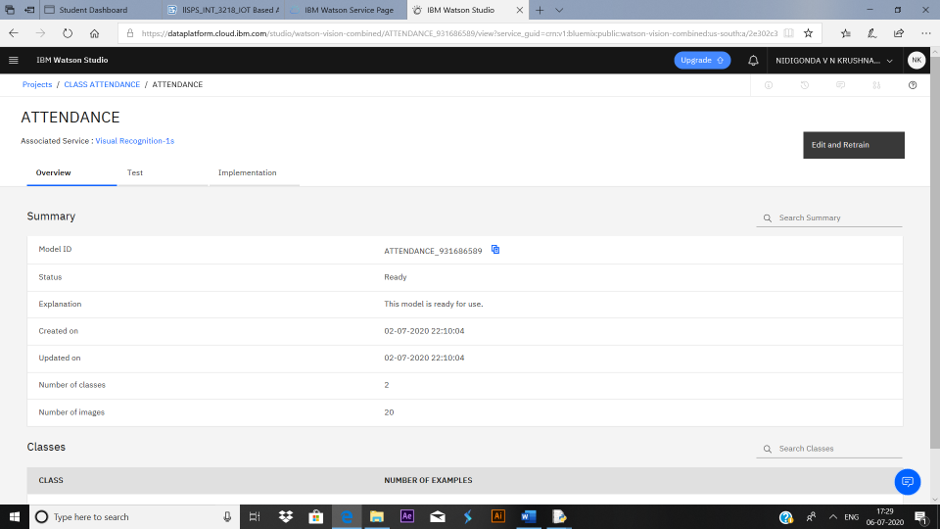




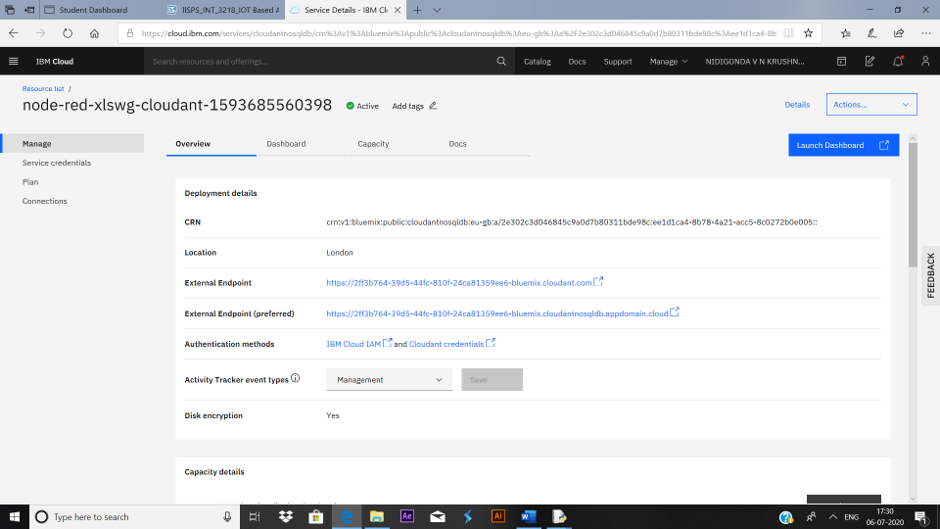
**IBM VISUAL RECOGNITION SERVICE :**



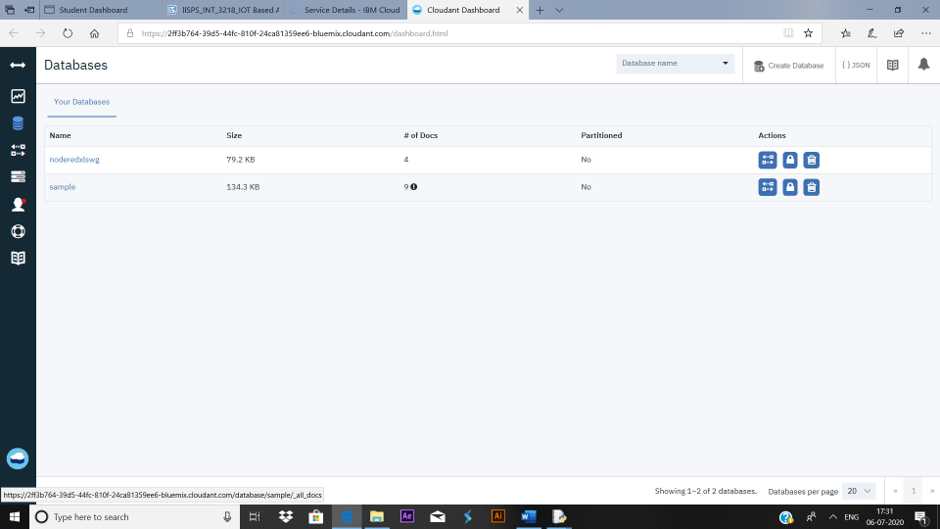
**IBM WATSON STUDIO :**



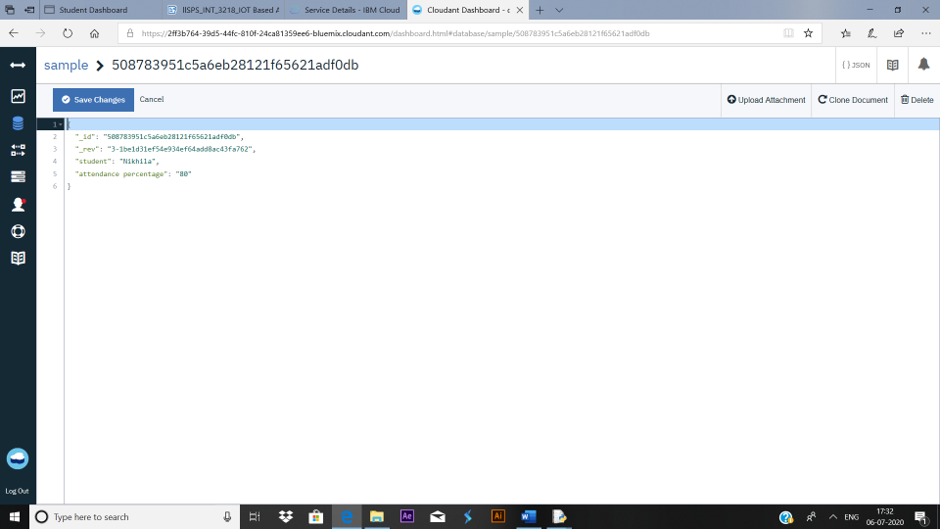
**IBM CLOUDANT DB:**



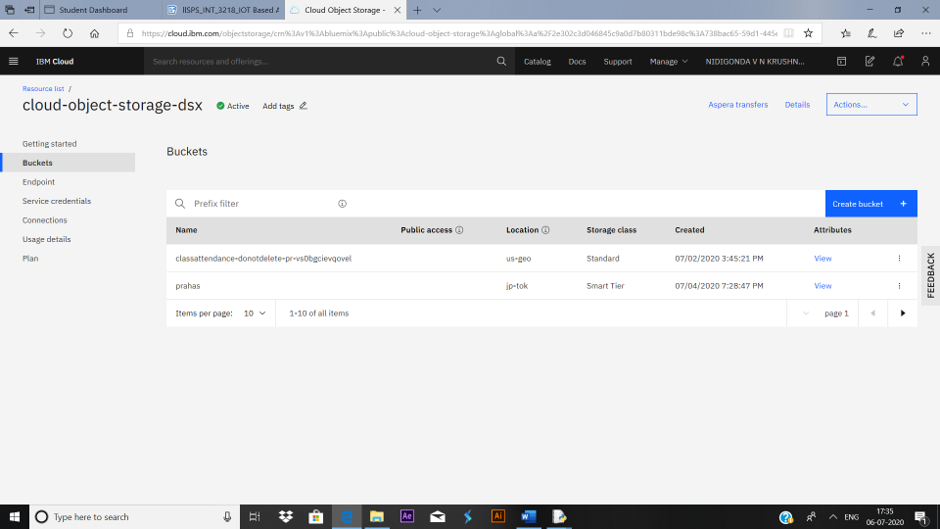
**LIST OF DATABASES CREATED :**



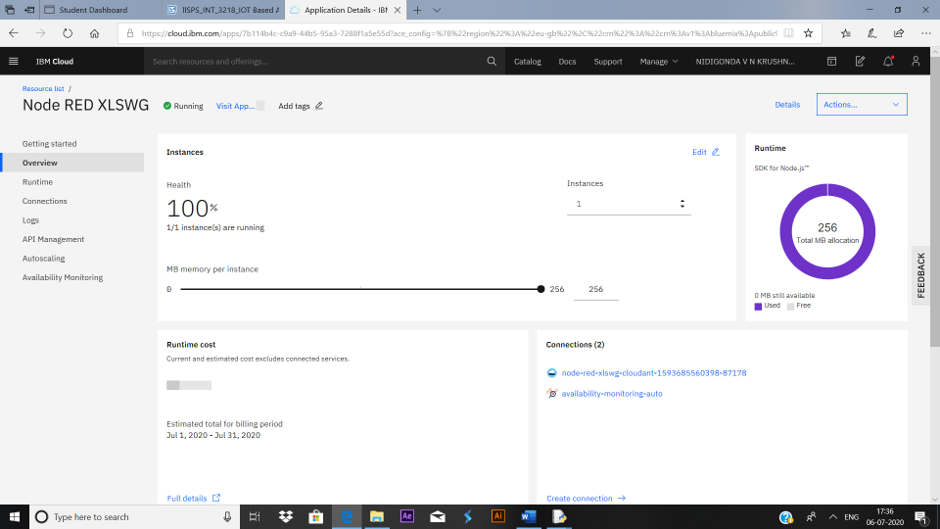
**DOCUMENT CREATED IN THE DATABASE:**

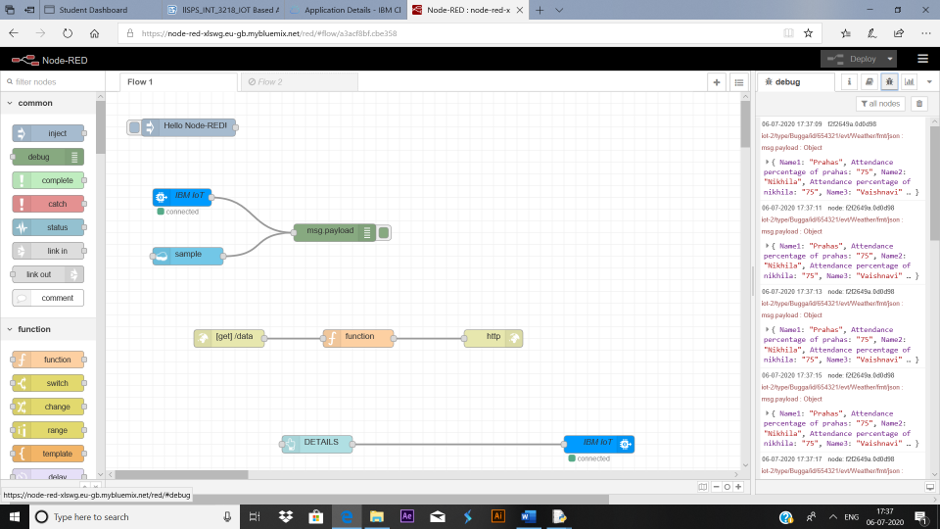


**CLOUD STORAGE SERVICE IN IBM:**

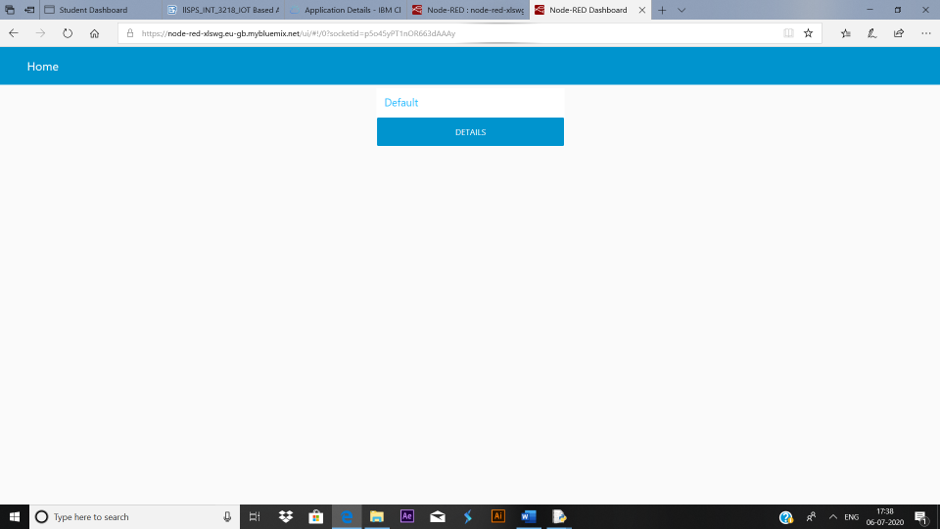


**NODE RED SOFTWARE AND THE FLOWS :**

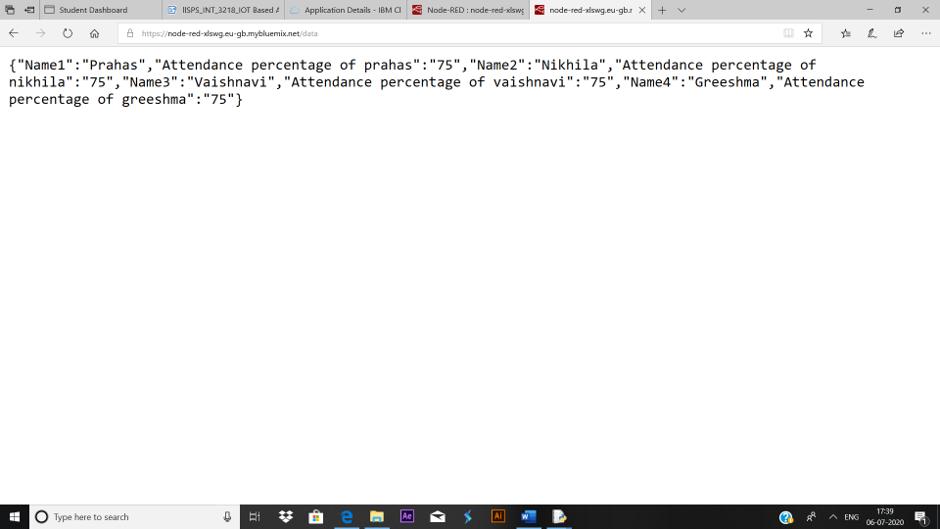




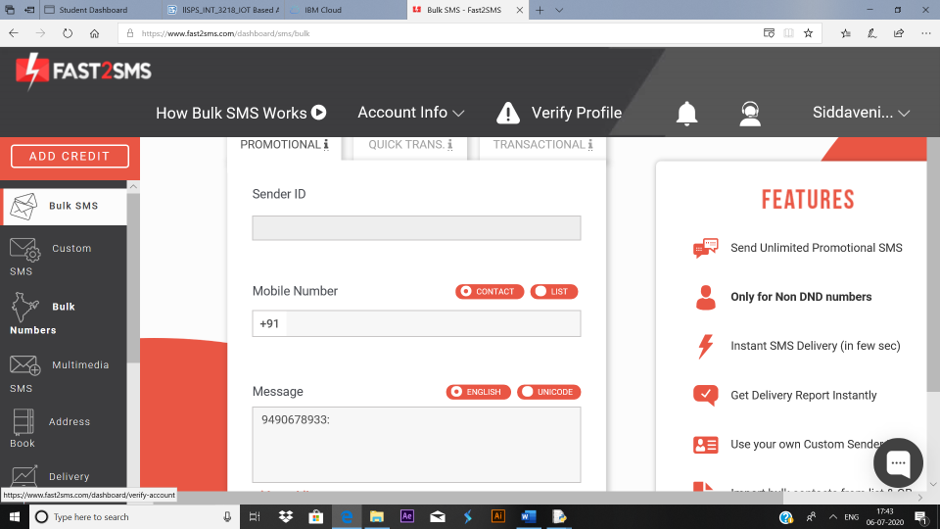
**NODE RED UI WINDOW :**



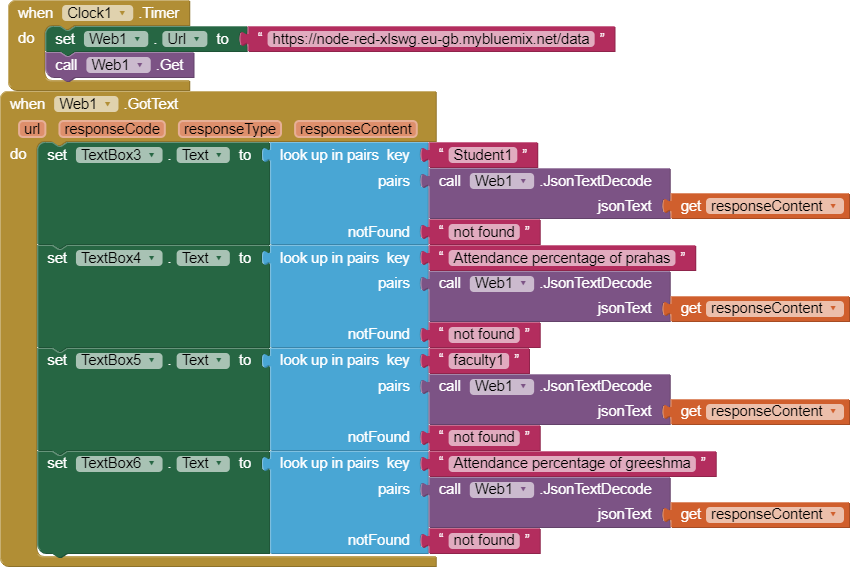
**DATA IN THE HTTP URL RESPONSE FOR THE REQUEST CREATED :**

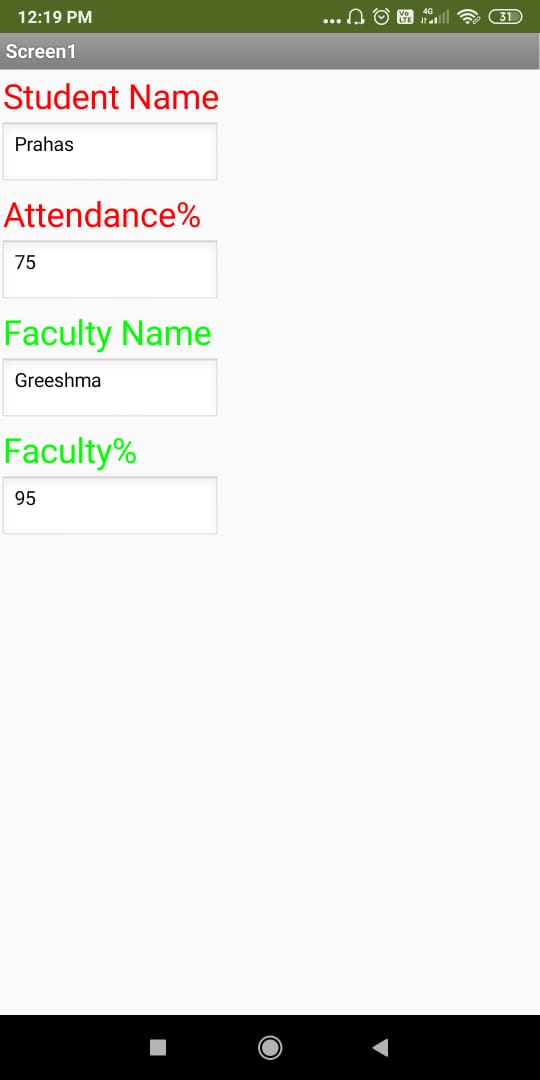


**FAST2SMS :**

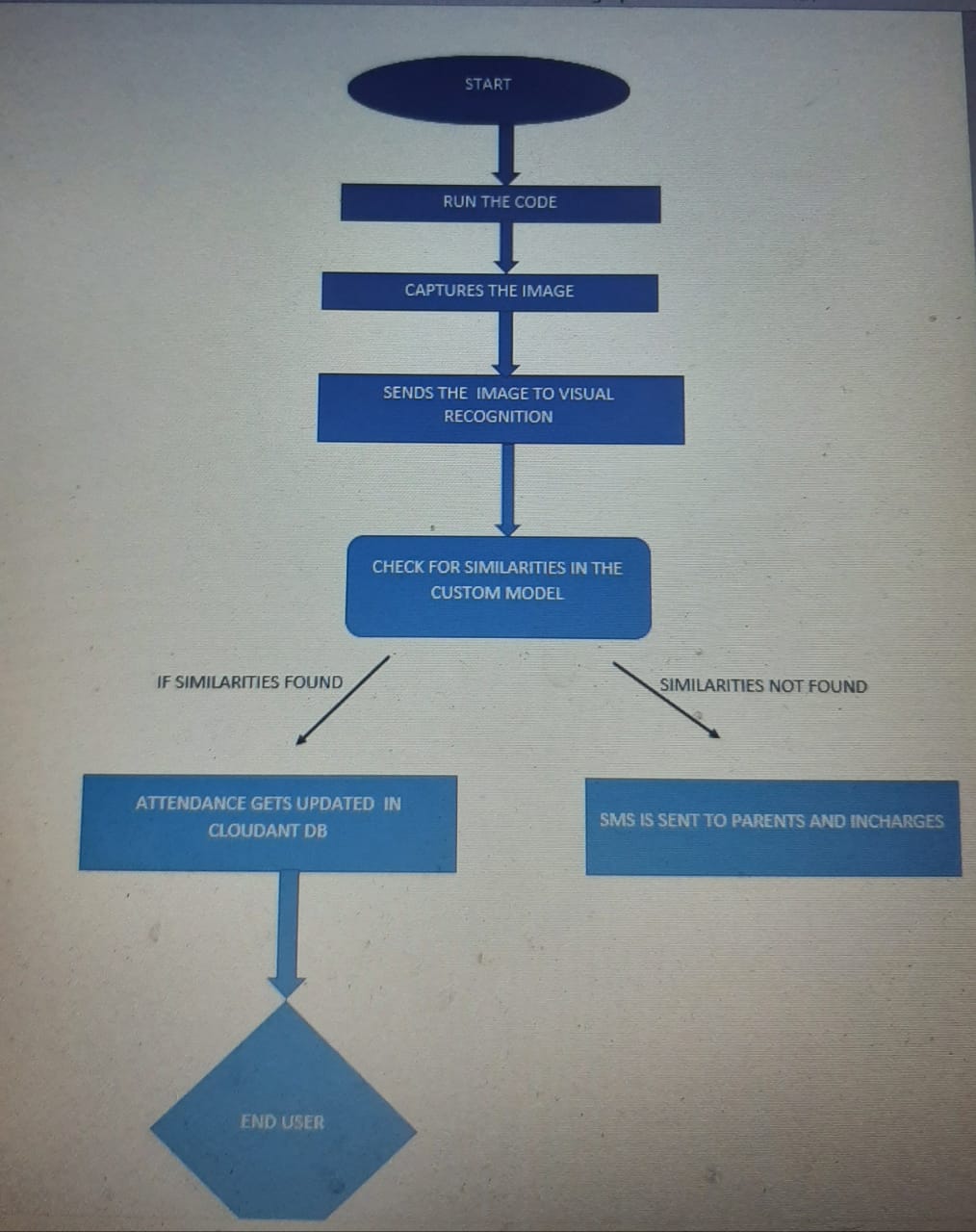


**MIT APP INVENTOR AND THE APPLICATION :**





**FLOWCHART :**



**RESULT :**

Hence, using the python code and the IBM cloud platform, we capture the images and store the data of the attendance of students and faculty.Attendance details are sent to parents, H.O.D if required.Even classes are recorded using openCV code for further access to the students and also application is built by which details of attendance can be accessed by Students, Teachers, Parents and H.O.D.

**ADVANTAGES** :

>> Ensures "no manipulation" attendance system.

>> Easy to store and retrieve information whenever required.

>> Changes to any data also can be done easily.

**DISADVANTAGES** :

>> Any wrong information recorded may cause issues in the institution where attendance is an important aspect.

>> Not all people can easily adapt to this system.

**APPLICATIONS** :

>> In all educational institutions.

>> In governmental and non-governmental organisations for the attendance details of staff.

**CONCLUSION :**

The "IoT Based Smart Attendance System" provides the chance for every educational institution and offices to maintain transparency in the attendance details of every student, faculty or employee.

**FUTURE SCOPE :**

Educational sector and employment sector are two main pillars of the Indian Economy.

We need to maintain transparency in each and every aspect related to these sectors.

For ensuring the transparency in this aspect of attendance, this type of smart projects, play a major role.If attendance records are maintained properly,they can be accessed in the future if necessary. In offices where salary is based on attendance or in educational institutions where attendance is compulsory for promotion to further levels of education, this system helps a lot. Accountability in these both sectors also increases.

As all the sectors of economy are starting to prefer technological or electronics intervention rather than human intervention. Educational institutions and offices can take this system as an example, adapt it and further become more technology dependent.

**BIBLIOGRAPHY** :

>>Some of the websites like www.fast2sms.com, cloud.ibm.com and appinventor.mit.edu.

>> Also the readily available "PYTHON IDLE" is used for coding in this project.

**PROJECT DELIVARABLES :**

>>Web Application

>> Project Report

>> Project Video

**APPENDIX :**

**CODE FOR IBM IOT :**

import time  
import sys  
import ibmiotf.application  
import ibmiotf.device  
import random

organization="5k0wq9"  
deviceType="Bugga"  
deviceId="654321"  
authMethod="token"  
authToken="87654321"

def myCommandCallback(cmd):  
 if cmd.data:  
 print("Name1:Prahas","Attendance percentage of prahas:75",  
 "Name2:'Nikhila","Attendance percentage of nikhila:75",  
 "Name3:Vaishnavi","Attendance percentage of vaishnavi:75",  
 "faculty1:Greeshma","Attendance percentage of greeshma:95")  
try:  
 deviceOptions={"org":organization,"type":deviceType,"id":deviceId,"auth-method":authMethod,"auth-token":authToken}  
 deviceCli=ibmiotf.device.Client(deviceOptions)

except Exception as e:  
 print(str(e))  
 sys.exit()

deviceCli.connect()

while True:  
 data={'Name1':'Prahas','Attendance percentage of prahas':'75','Name2':'Nikhila','Attendance percentage of nikhila':'75','Name3':'Vaishnavi','Attendance percentage of vaishnavi':'75','faculty1':'Greeshma','Attendance percentage of greeshma':'95'}  
 def myOnPublishCallback():  
 print(data)  
 success=deviceCli.publishEvent("Weather","json",data,qos=0,on\_publish=myOnPublishCallback)  
 if not success:  
 print("not connected")  
 time.sleep(2)  
 deviceCli.commandCallback=myCommandCallback

deviceCli.disconnect()

**CODE FOR OPENCV VIDEOCAPTURE:**

import cv2  
cap=cv2.VideoCapture(0)

print(cap.isOpened())  
rec=cv2.VideoWriter\_fourcc(\*'XVID')

out=cv2.VideoWriter('output.avi',rec,20.0,(640,480))  
while(cap.isOpened()):  
 ret,frame=cap.read()

out.write(frame)

cv2.imshow("videocapture",frame)

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

cap.release()  
cv2.destroyAllWindows()

**CODE FOR VISUAL RECOGNITION :**

import json  
from watson\_developer\_cloud import VisualRecognitionV3

visual\_recognition = VisualRecognitionV3(  
 '2018-03-19',  
 iam\_apikey='2mez20oDaG6NyRGvosBuuQO4rIozeasNrdocFMYzY7\_O')  
   
with open('hani.jpg', 'rb') as images\_file:  
 classes = visual\_recognition.classify(  
 images\_file,  
 threshold='0.6',  
 classifier\_ids='ATTENDANCE\_931686589').get\_result()  
print(json.dumps(classes, indent=2))

**CODE FOR IMAGE CAPTURING WHEN FACE IS DETECTED:**

import cv2

import numpy as np

import datetime

face\_classifier=cv2.CascadeClassifier("haarcascade\_frontalface\_default.xml")

eye\_classifier=cv2.CascadeClassifier("haarcascade\_eye.xml")

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

video=cv2.VideoCapture(0)

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)

eyes=eye\_classifier.detectMultiScale(gray,1.3,5)

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")

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

for(ex,ey,ew,eh) in eyes:

cv2.rectangle(frame, (ex,ey), (ex+ew,ey+eh), (127,0,255), 2)

cv2.imshow('Face detection', frame)

#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

**CODE FOR SENDING MESSAGE ALERTS USING FAST2SMS:**

import requests

r=requests.get('https://www.fast2sms.com/dev/bulk?authorization=lfBwdoqVEMsGYPa Khk517J9InvytCpTujQH4DgeFxZ3LRm8ASUUdM6zjECX1GubLtyivqHIZ8Bf9gKPa&send er\_id=FSTSMS&message=This message is to inform you that your child did not come to college today&language=english&route=p&numbers=9398659128,6304514195,9676926774,9381274122')

print(r.status\_code)

**CODE FOR INTEGRATING OPENCV ,VISUAL RECOGNITION AND IBM PLATFORM:**

import cv2

import numpy as np

import datetime

import json

import time

import time

import sys

import ibmiotf.application

import ibmiotf.device

import random

#Provide your IBM IoT Device Credentials

organization="5k0wq9"

deviceType="Bugga"

deviceId="654321"

authMethod="token"

authToken="87654321"

from watson\_developer\_cloud import VisualRecognitionV3

from cloudant.client import Cloudant

from cloudant.error import CloudantException

from cloudant.result import Result, ResultByKey

import requests

face\_classifier=cv2.CascadeClassifier("haarcascade\_frontalface\_default.xml")

eye\_classifier=cv2.CascadeClassifier("haarcascade\_eye.xml")

#VisualRecognition Service Credentials

visual\_recognition = VisualRecognitionV3(

'2018-03-19',

iam\_apikey='qDP8ul5pmj-JeA-X59xTLeI9UNpncNoSRWRCimBcO-hZ')

#Provide CloudantDB credentials such as username,password and url

client=Cloudant("2ff3b764-39d5-44fc-810f-24ca81359ee6-bluemix", "0acf9043d2850635230da1c615cb01f501659cf87cca8c0142243c1e20b9c33f", url="https://2ff3b764-39d5-44fc-810f-24ca81359ee6-bluemix:0acf9043d2850635230da1c615cb01f501659cf87cca8c0142243c1e20b9c33f@2ff3b764-39d5-44fc-810f-24ca81359ee6-bluemix.cloudantnosqldb.appdomain.cloud")

client.connect()

#Provide your database name

database\_name = "atomated\_attendence"

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

if my\_database.exists():

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

# Initialize GPIO

def myCommandCallback(cmd):

if cmd.data:

print("Name1:Prahas","Attendance percentage of prahas:75",

"\nName2:Nikhila","Attendance percentage of nikhila:75",

"\nName3:Vaishnavi","Attendance percentage of vaishnavi:75",

"\nName4:Greeshma","Attendance percentage of greeshma:75")

try:

deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMethod, "auth-token": authToken}

deviceCli = ibmiotf.device.Client(deviceOptions)#create client

#.............................................

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()#connect client to platform

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

video=cv2.VideoCapture(0)

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)

eyes=eye\_classifier.detectMultiScale(gray,1.3,5)

print(faces)

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")

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

with open(picname+'.jpg', "rb") as images\_file:

classes = visual\_recognition.classify(

images\_file,

threshold='0.6',

classifier\_ids='Attendence\_834401690').get\_result()

if len(classes['images'][0]['classifiers'][0]['classes'])<1:

print('Face not detected')

continue

img=classes['images'][0]['classifiers'][0]['classes'][0]['class']

if img=='Others':

continue

else:

print(img+' IS PRESENT')

attendence=random.randint(75,80)

#print(json.dumps(classes, indent=2))

json\_document={"Student\_Name":img,"AttendencePercentage":attendence}

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

# Check that the document exists in the database.

if new\_document.exists():

print(f"Document successfully created.")

data={'Student\_Name':img,'AttendencePercentage':attendence}

def myOnPublishCallback():

print(data)

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

if not success:

print("not connected")

time.sleep(2)

deviceCli.commandCallback=myCommandCallback

for(ex,ey,ew,eh) in eyes:

cv2.rectangle(frame, (ex,ey), (ex+ew,ey+eh), (127,0,255), 2)

cv2.imshow('Face detection', frame)

#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()

----------THE END----------

**?**