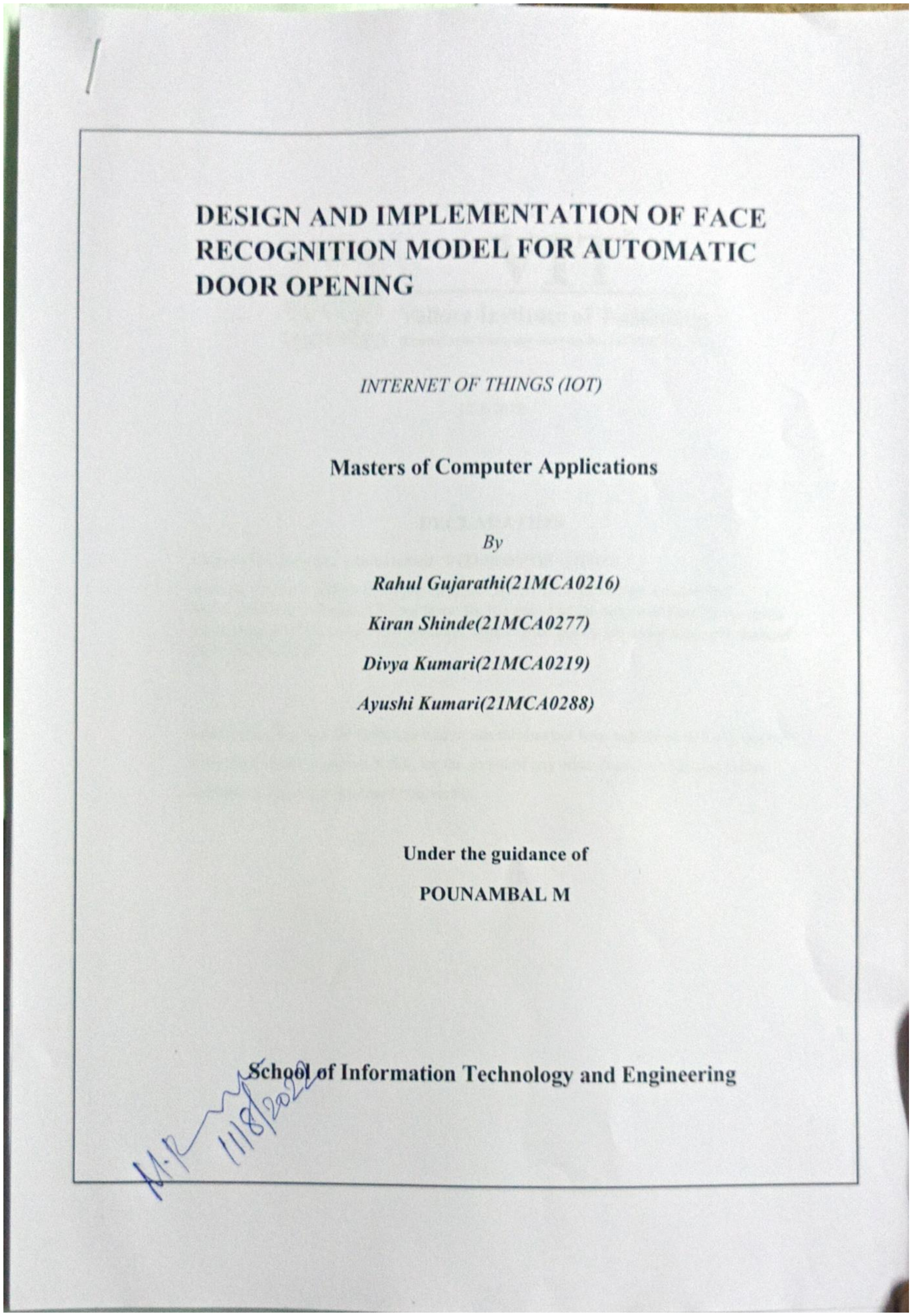


DESIGN AND IMPLEMENTATION OF FACE RECOGNITION MODEL FOR AUTOMATIC DOOR OPENING

INTERNET OF THINGS (IOT)

Masters of Computer Applications

By

Rahul Gujarathi(21MCA0216)

Kiran Shinde(21MCA0277)

Divya Kumari(21MCA0219)

Ayushi Kumari(21MCA0288)

Under the guidance of

POUNAMBAL M

School of Information Technology and Engineering

[illegible] 11/8/2022

School of Information Technology and Engineering

VIT, Vellore

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

12/8/2022

DECLARATION

I hereby declare that the this entitled "INTERNET OF THINGS

DESIGN AND IMPLEMENTATION OF FACE RECOGNITION MODEL FOR AUTOMATIC DOOR OPENING" submitted by our team, for the award of the degree of Specify the name of the degree VIT is a record of bonafide work carried out by me under the supervision of POUNAMBAL M

I further declare that the work reported in this this has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place: Vellore

Date: 12/08/2022

Signature of the Candidates

CERTIFICATE

This is to certify that the thesis entitled “INTERNET OF THINGS

DESIGN AND IMPLEMENTATION OF FACE RECOGNITION MODEL FOR AUTOMATIC DOOR OPENING

submitted

RAHUL GUJARATHI (21MCA0216)

KIRAN SHINDE (21MCA0277)

AYUSHI KUMARI (21MCA0288)

DIVYA KUMARI (21MCA0219)

School of Information Technology and Engineering VIT, for the award of the degree of Bachelor of Computer Applications is a record of bonafide work carried out by him under my supervision.

The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university. The Project report fulfils the requirements and regulations of VIT and in my opinion meets the necessary standards for submission.

TABLE OF CONTENTS

ABSTRACT:

The main objective of the thesis is to develop an AI-based face recognition Model (which is implemented following the Deep Learning algorithm) for the security system for making decisions to lock or unlock the door system and to deploy the developed AI Model in a Docker Container on an IoT platform. The main aim of the project would be to achieve the edge computing concept that brings the Artificial Intelligence (through our AI model) to the low power Internet of Things (IoT) devices with the help of containerization concept. Containerisation would be similar to the virtualisation. Docker containers are easy to port on various IoT devices (Firefly rk3399). Along with the portability, Docker includes all the dependencies and modules required for running the application in a container.

- Method: The methodology of developing the containerised AI model. We have chosen the method of training the algorithm such that it detects the faces captured by our camera, which is connected with the help of CSI connector. The algorithm includes the concept of Deep Learning which is a subset of Artificial Intelligence. The method consists of several steps, for example, Deep learning Algorithm detects the faces from the image, and then the image is converted to a set of gradients. These gradients can be converted again to landmarks.

- to consider the focal points of the image and then the training step is performed using the Support Vector Machine classifier. Finally, the authorised user is recognised.

INTRODUCTION:

Machine Learning Methods: Machine Learning can be done in multiple ways as per the users/programmer requirement. Currently, a highly used method for Machine Learning has Supervised Learning. Majorly used is Unsupervised, whereas Reinforcement and

Semisupervised are used on rare occasions. Supervised Machine Learning: These algorithms are written in a way that they can apply what has been learned in the past to new data using examples, which in turn may be allowed to predict future events. The past data, commonly known as examples, are used as input, output desired is known, and these are the starting steps for these algorithms to start learning. The analysis of this labelled example leads to the learning algorithm producing a function that can successfully make predictions about the future outputs. The system can provide targets for any new input after sufficient training.

The Supervised Algorithm has the capability to infer on its errors and mistakes by comparing the algorithm's output to the given output. **Unsupervised Machine Learning algorithms** are used when the input data gave is not bound by any structure or any labelled examples like in the algorithms of Supervised Machine Learning. These algorithms are developed to intuitively discover a structure or any form of label within the given unlabelled data. Deep Learning: Like ML, "deep Learning" is likewise a technique that concentrates highlights or characteristics from crude data sets. The central matter of contrast is DL does this by using a multi-layer Artificial neural network with many shrouded layers stacked in a steady progression. DL additionally has, to some degree, progressively modern algorithms and requires more powerful computational resources. These are exceptionally created computers with high-performance CPUs or GPUs. Most deep learning techniques utilize neural network designs, which is the reason deep learning models are regularly alluded to as deep neural systems. The expression "deep" means the quantity of concealed layers in the neural network. Conventional neural systems just contain 2-3 hidden layers, while deep neural networks can have upwards of 150.

Artificial neural networks (ANNs) are computing systems that are actually inspired by biological neural networks. Artificial neural networks (ANNs) are the systems within the neural networks. Such frameworks learn (logically enhance their capacity) to do assignments by thinking about precedents, by and large without undertaking explicit programming. Typically, neurons are organized in layers. After some time, consideration concentrated on coordinating explicit mental capacities, prompting deviations from science, for example, back propagation, or passing data in the switch bearing and altering the system to mirror that data. The actual goal of the neural network is to find solutions in the same way that a human brain would resolve. The input can lead to output either by using a linear relationship or a non-linear relationship. Considering an example, a deep neural network that is trained to recognize the dog breeds would check the picture and confirms the probability that the dog, which is in the image, is a particular breed.

Face Recognition Techniques: The face recognition techniques can be divided into three categories:

- Techniques that operate on intensity image
- Techniques that deal with video sequences.
- Techniques that require other sensory data such as 3D information or infra-red imagery. Considering the intensity of images as criteria, this technique can be divided into two categories:
- Feature-based approach
- Holistic-based approach Feature-based approach Feature-based methodology will process the input picture to distinguish and remove facial highlights, for example, eyes, mouth, nose, and so forth and after that, it figures out the geometric relationship among those facial focuses, along these lines lessening the input facial picture to a vector of the geometric elements. This methodology is sub partitioned into:
- Geometric feature-based matching: These techniques are dependent on the computation of a set of components from the image of a face. The complete set can be portrayed as a vector. The vector represents the position and the span of the primary facial highlights like the nose, eyes, eyebrows, mouth, jaw, and the blueprint of the Face. The pros of the approach are, it defeats the issue of occlusion, and it doesn't require extensive computational time. The disadvantage is that it doesn't provide a high degree of precision.
- Elastic bunch graph: This approach is based on dynamic link structures. A diagram for an individual face is produced utilizing a lot of fiducially focuses on the Face, each fiducially point is a hub of a completely associated diagram and is named with the Gabor channels' reaction. A representative set of such graphs is combined into a stacklike structure called face bunch graph. The recognition of a new face image is done by comparing its image graph to those of all the known face images and the one with the highest similar values are selected as closed matching. Advantages:
- By focusing only on the bounded areas, they do not modify or damage any information in the images
- This approach generates improvised recognition outcomes than the feature based approach. Disadvantages: The approach utilizes an immense unit of interaction between the test and training images
- When there is a massive difference in the pore, scale, and illumination, this technique doesn't carry out productively.

Hybrid approach: The hybrid approach is achieved by synthesizing multiple techniques to benefit from developing prominent outputs. Adopting numerous technologies will enable us to compensate for the cons of one approach by the pros of another technique.

MODULES DESCRIPTION:

Face recognition from sensory:

Abundant efforts were put forth on face recognition for a 2- Dimensional (2D) intensity images. Yet, a complete study hasn't been done in identifying an undividable by accomplishing other methods of sensing like 3- Dimensional (3D) or range data IR imagery. 3D based model techniques: This method will aid in accomplishing the characteristics which are depended on the shape of face and outline cheeks without disturbing the changes occurred due to lighting, orientation, and background clusters that will influence the 2D system. Examples of 3D include scanning systems, stereo vision systems, structured light systems, reverse rendering/shape from shading, etc.

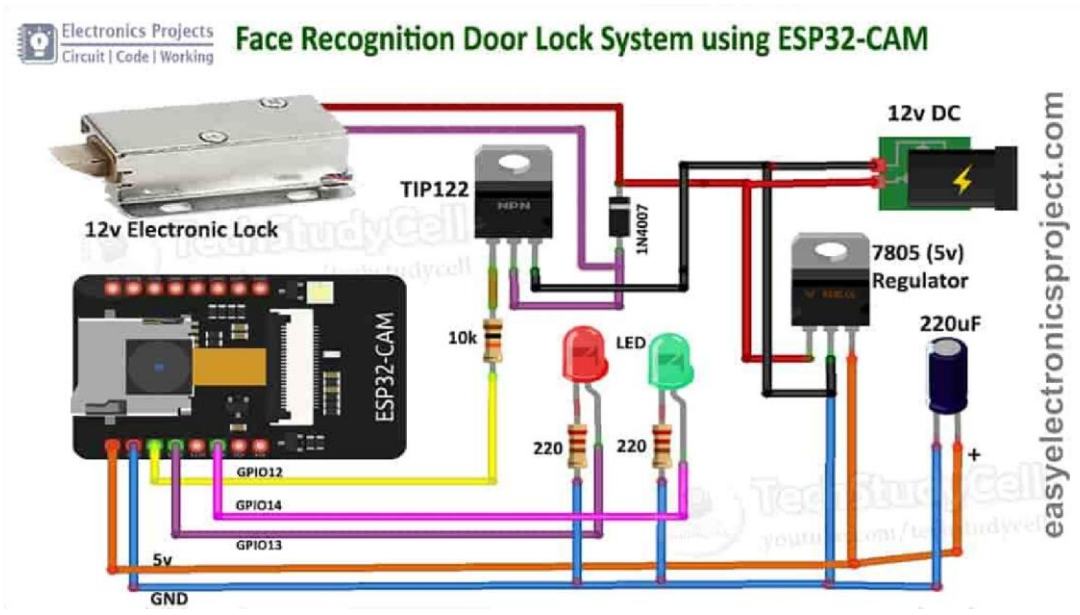
<u>Advantages:</u>

- It accomplishes the characteristics by illustrating the shape of the Face.

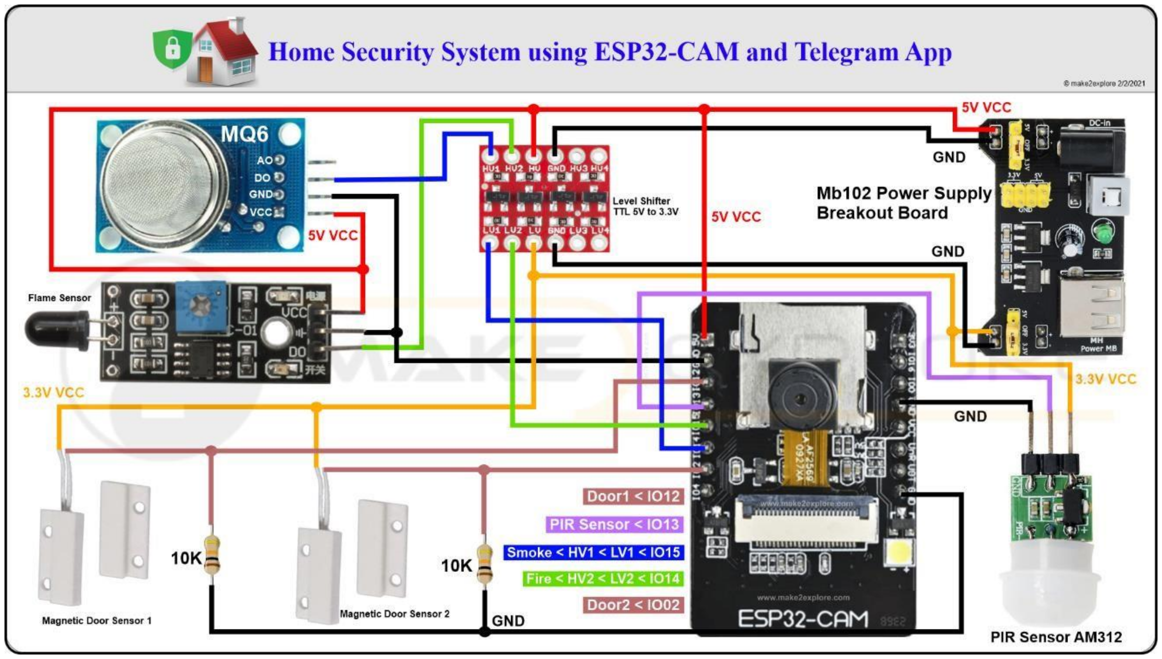
<u>Disadvantages:</u>

- This technique is difficult to understand and is highly expensive.

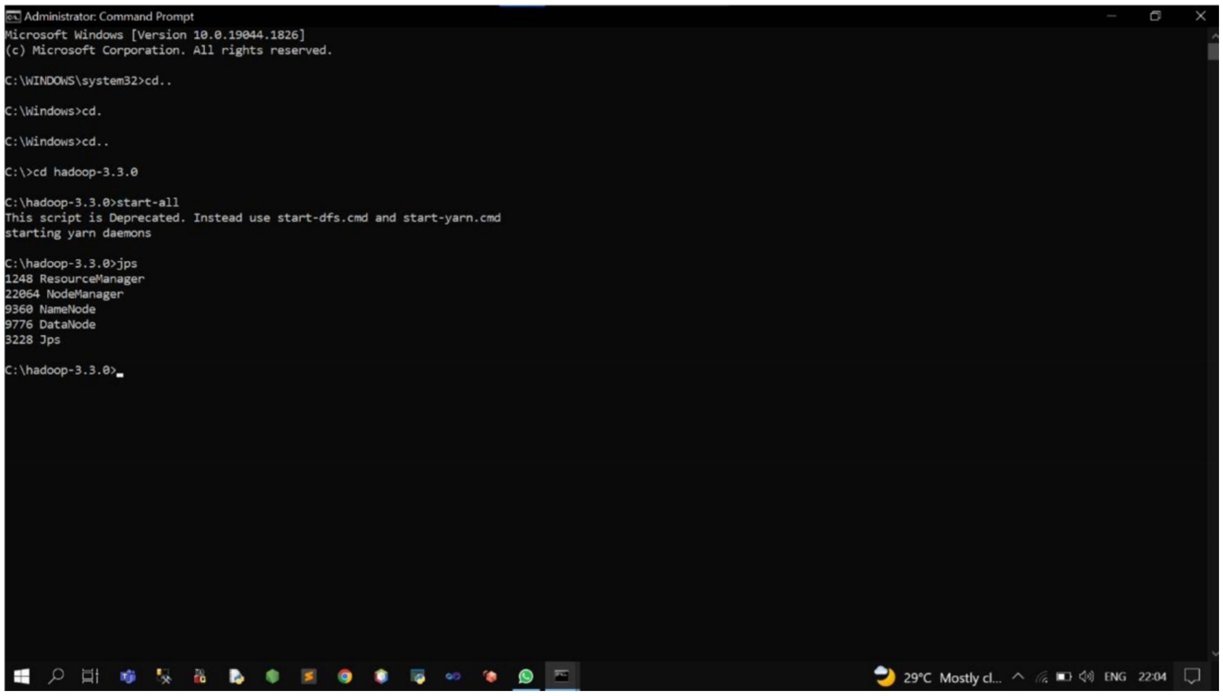
MODULE ARCHITECTURE:



HARDWARE REQUIREMENTS:

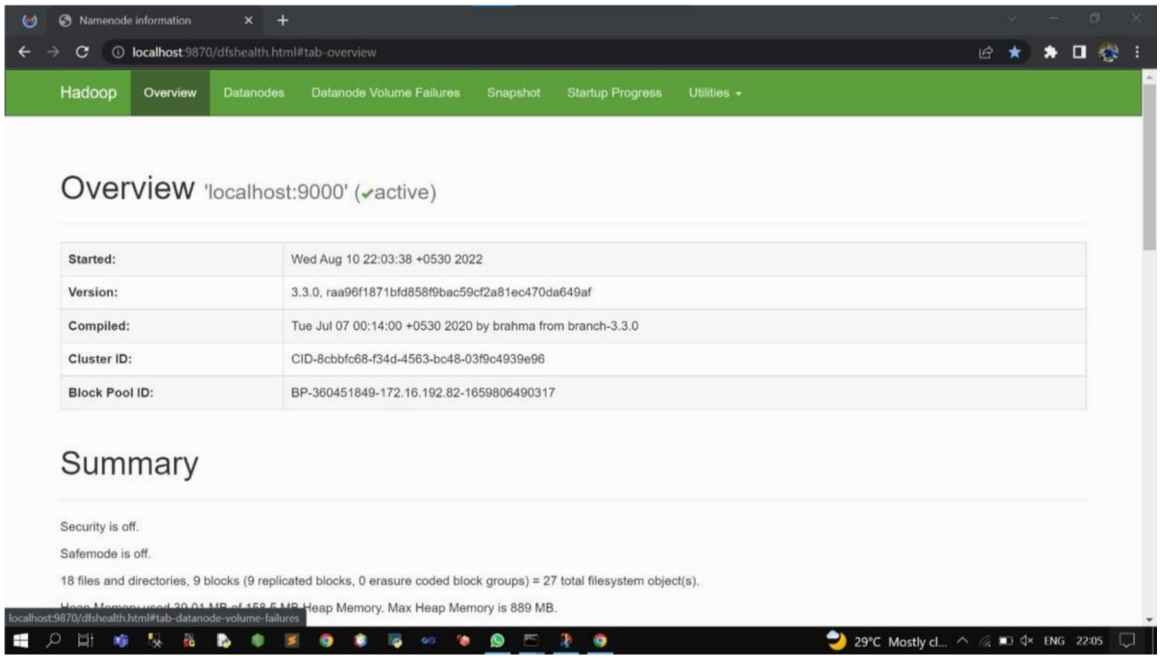


HADOOP INSTALLATION:



Administrator: Command Prompt

```
Microsoft Windows [Version 10.0.19044.1826]
(c) Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>cd..

C:\Windows>cd.

C:\Windows>cd..

C:\>cd hadoop-3.3.0

C:\hadoop-3.3.0>start-all
This script is Deprecated. Instead use start-dfs.cmd and start-yarn.cmd
starting yarn daemons

C:\hadoop-3.3.0>jps
1248 ResourceManager
22064 NodeManager
9360 NameNode
9776 DataNode
3228 Jps

C:\hadoop-3.3.0>
```



Overview 'localhost:9000' (✓active)

Started:	Wed Aug 10 22:03:38 +0530 2022
Version:	3.3.0, raa96f1871bfd858f9bac59cf2a81ec470da649af
Compiled:	Tue Jul 07 00:14:00 +0530 2020 by brahma from branch-3.3.0
Cluster ID:	CID-8cbbfc68-f34d-4563-bc48-03f9c4939e96
Block Pool ID:	BP-360451849-172.16.192.82-1659806490317

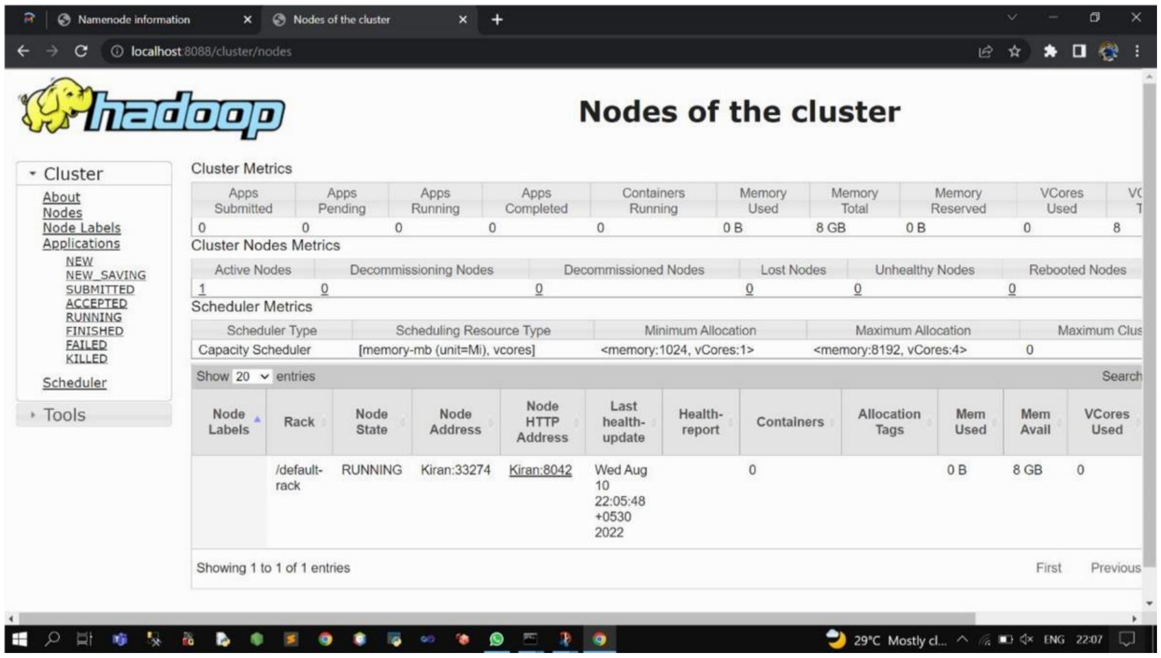
Summary

Security is off.

Safemode is off.

18 files and directories, 9 blocks (9 replicated blocks, 0 erasure coded block groups) = 27 total filesystem object(s).

Heap Memory. Max Heap Memory is 889 MB.



Nodes of the cluster

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used
0	0	0	0	0	0 B	8 GB	0 B	0

Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes
1	0	0	0	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation
Capacity Scheduler	[memory-mb (unit=Mi), vcores]	<memory:1024, vCores:1>	<memory:8192, vCores:4>

Show 20 entries

Node Labels	Rack	Node State	Node Address	Node HTTP Address	Last health-update	Health-report	Containers	Allocation Tags	Mem Used	Mem Avail	VCores Used
	/default-rack	RUNNING	Kiran:33274	Kiran:8042	Wed Aug 10 22:05:48 +0530 2022		0		0 B	8 GB	0

Showing 1 to 1 of 1 entries

STEPS OF HADOOP:

Download Hadoop version 3.3.0 from Create a folder in C:\ “Hadoop” Move the downloaded tar.gz file to the folder C:\Hadoop Open Windows powershell Navigate to the folder cd C:\Hadoop And unzip the hadoop downloaded file

After the unzip command is completed, a new folder hadoop-3.3.0 is created under the destination folder.

Check java version it should be higher than 1.8 In Environment variables Set

User variables, if it has space between change it (Ex: Program Files into Program

Download the Binary tar.gz archive and unzip apache-maven-3.8.2-bin.zip or tar xzvf apache-maven 3.8.2-bin.tar.gz Set the Environment Path - Add the bin directory of the created directory apache-maven-3.8.2 to the PATH environment variable

To check if its installed: in Command prompt C:\WINDOWS\system32>mvn

--version Apache Maven 3.8.2

(ea98e05a04480131370aa0c110b8c54cf726c06f) Maven home: C:\apachemaven-3.8.2 Java version: 13.0.2, vendor: Oracle Corporation, runtime:

C:\Progra~1\Java\jdk-13.0.2 Default locale: en_IN, platform encoding:

Cp1252 OS name: "windows 10", version: "10.0", arch: "amd64", family: "windows"

Set the ZLIB_HOME environment variable to the directory containing the headers. set ZLIB_HOME=C:\zlib-1.2.7 At runtime, zlib1.dll must be accessible on the PATH. Hadoop has been tested with zlib 1.2.7, built using Visual Studio 2010 out of contrib\vstudio\vc10 in the zlib 1.2.7 source tree.

Set Environmental Variables In user variables: HADOOP_HOME C:\Hadoop\hadoop-3.3.0\bin

Edit file core-site.xml in C:\Hadoop\hadoop-3.3.0\etc\hadoop fs.default.name hdfs://localhost:9000 Edit file hdfs-site.xml in C:\Hadoop\hadoop3.3.0\etc\hadoop dfs.replication 1 dfs.namenode.name.dir file:///C:/Hadoop/hadoop-3.3.0/data/namenode dfs.datanode.data.dir /C:/Hadoop/hadoop-3.3.0/data/datanode Edit file mapred-site.xml in

C:\Hadoop\hadoop-3.3.0\etc\hadoop mapreduce.framework.name yarn mapreduce.application.classpath

%HADOOP_HOME%/share/hadoop/mapreduce/,%HADOOP_HOME%/sha re/hadoop/mapre duce/lib

examples/,%HADOOP_HOME%/share/hadoop/common/,%HADOOP_HO ME%/share/hadoop/com

mon/lib/,%HADOOP_HOME%/share/hadoop/yarn/,%HADOOP_HOME%/s hare/hadoop/yarn/lib/,

%HADOOP_HOME%/share/hadoop/hdfs/,%HADOOP_HOME%/share/had

oop/hdfs/lib/ Edit file yarn-site.xml in C:\Hadoop\hadoop-3.3.0\etc\hadoop yarn.nodemanager.aux-services mapreduce_shuffle yarn.nodemanager.envwhitelist

JAVA_HOME,HADOOP_COMMON_HOME,HADOOP_HDFS_HOME,H

ADOOP_CONF_DIR,CLASS

PATH_PREPEND_DISTCACHE,HADOOP_YARN_HOME,HADOOP_H OME,PATH,LANG,TZ,HADOOP_MAPR ED_HOME Edit hadoop-env file, set JAVA_HOME=C:\Progra~1\Java\jdk-13.0.2

IMPLIMENTATION:

FRONT END OF SYSTEM:

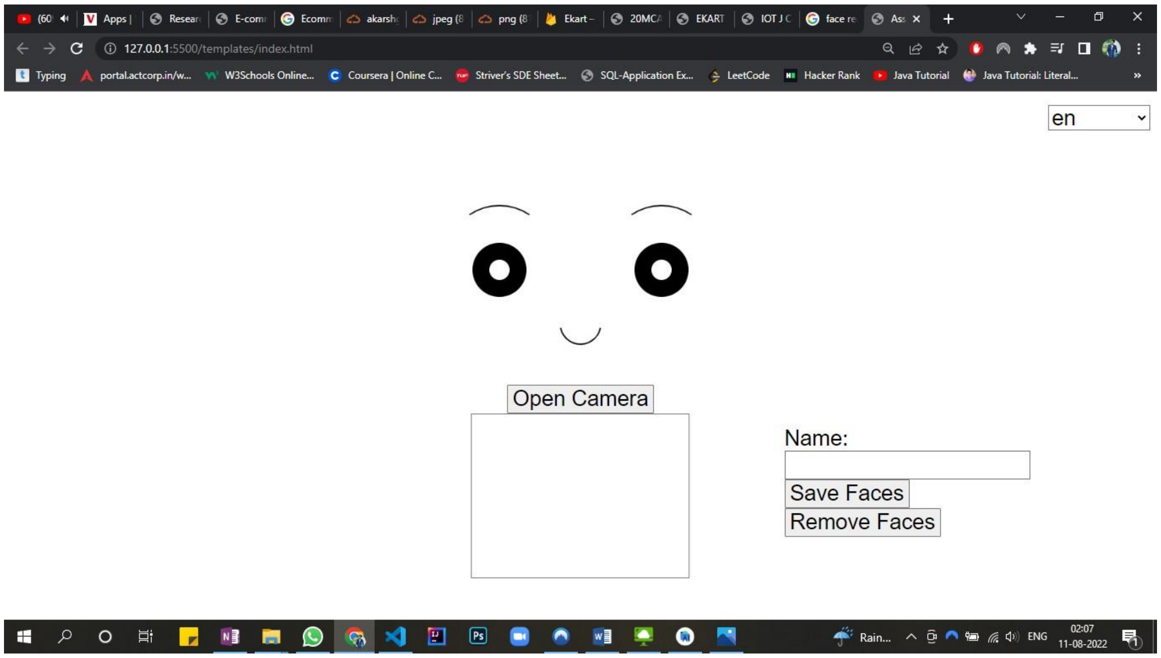
```
<!DOCTYPE html>
<html lang="en" >

<head>
  <meta charset="UTF-8">
  <title>Assignment2</title>
      <link type="text/css" rel="stylesheet" href="{{ url_for('static',
filename='css/style.css') }}">
      <style>
*{
          font-family: Helvetica;
font-size: 32px;           margin:
0px;
```

```
        }
html, body {
          height: 100%;
          width: 100%;
        }
div.wrapper {
display: flex;
          flex-direction: column;
align-items: center;
          justify-content: flex-start;
        }
select#lang {
margin-top: 20px;
margin-left: 90%;
        }
div#subtitle {
margin-top: 20px;
        }
div.wrapper2 {             display:
flex;           flex-direction:
row;           align-items:
center;           justify-content:
center;           margin-top:
20px;
        }
div.wrapper3 {             display:
flex;           flex-direction:
column;           align-items:
center;           margin-left:
150px;
        }
        div.wrapper4 {
width: 10px;           align-
items: left;           margin-
left: 140px;
        }
        canvas#cameraCanvas {
border: 2px gray solid;
transform: scaleX(-1);
        }
```

```
      </style>
</head>

<body>
```

```
  <html>
  <head>
  </head>

  <body>
    <div class="wrapper">
      <select id="lang"></select>
      <div id="subtitle">Hi, I am a robot!</div>
      <canvas id="faceCanvas" width="400" height="300"></canvas>
    </div>

    <div class="wrapper2">

      <div class="wrapper3">
        <input id="startCamera" type="button" value="Open Camera">
        <video hidden id="myVideo" width="640" height="480"></video>
        <canvas id="cameraCanvas" width="320" height="240"></canvas>
        <div id="lightIndicator"></div>
      </div>

      <div class="wrapper4">
        Name:
        <input id="nameinput">
        <input id="saveFace" type="button" value="Save Faces">
        <input id="removeFace" type="button" value="Remove Faces">
      </div>

    </div>
  </body>

  <script>     class Camera {        constructor() {
this.video = document.getElementById("myVideo");
this.canvas = document.getElementById("cameraCanvas");
this.context = this.canvas.getContext("2d");
this.blackThres = 10;
        this.lightTextDiv = document.getElementById("lightIndicator");
         this.frame = 100;
        this.send_cnt = 0;
            this.left = 0;
           this.right = 0;
        this.top = 0;
this.bottom = 0;          this.squareData
= new Array(0);
      }

      /* This function checks and sets up the camera */        startVideo()
{          console.log("hello");          console.log(this);         if
(navigator.mediaDevices && navigator.mediaDevices.getUserMedia)
{
navigator.mediaDevices
            .getUserMedia({video: true})
            .then(this.handleUserMediaSuccess.bind(this));
        }
window.setInterval(communicate.recognize.bind(communicate), 2000);
      }

      /* This function initiates the camera this.video */
handleUserMediaSuccess(stream) {         this.video.src =
window.URL.createObjectURL(stream);
this.video.play();
```

```
        /* We will capture 10 image every second */
window.setInterval(this.captureImageFromVideo.bind(this), this.frame);
      }

      /* This function captures the this.video */
captureImageFromVideo() {       // Draw the images
        this.context.drawImage(this.video, 0, 0, this.canvas.width,
this.canvas.height);

        // Get the image data
        var dataObj = this.context.getImageData(0, 0, this.canvas.width,
this.canvas.height);         var data = dataObj.data;
        var grayImg = []
        // Generate the gray image
         for (var i = 0, j = 0; i < data.length; i += 4, j += 1) {
          var gray = 0.333 * (data[i] + data[i + 1] + data[i + 2]);
grayImg[j] = gray;
      }
      //this.context.putImageData(dataObj, 0, 0);

        // Detect the light level of the image
this.detectLightLevel(grayImg);

        // Send the img to database
communicate.detect(this.canvas.toDataURL())

        // Draw the square
this.drawSquare();
      }
switchCamera() {}

      // Detect the level of light
detectLightLevel(data) {          var light = 0;
for (var i = 0; i < data.length; i += 1) {
light += data[i];
        }         light /=
data.length;
                 if(light < this.blackThres)
this.lightTextDiv.innerHTML = "Are you sure your camera is not
blocked?";         else           this.lightTextDiv.innerHTML =
Math.floor(light);
      }
      drawSquare()
      {
```



WEB SERVER:

```
import cv2 from ImageProcess import
FaceProcess import requests import
threading as th import time import
numpy as np from flask import Flask
from flask import request from
flask_cors import * import json from
PIL import Image import base64 import
io


# If `entrypoint` is not defined in app.yaml, App Engine will look for an
app
# called `app` in `main.py`.
app = Flask(__name__)

# Enable CORS
CORS(app, supports_credentials=True)

# Users can change the following parameters
url_stranger =
'https://maker.ifttt.com/trigger/strangerdetect/with/key/cUXwhgUTcUmlTgd6D
```

```
hhawX' url_package
=
'https://maker.ifttt.com/trigger/PackageCall/with/key/cUXwhgUTcUmlTgd6Dhha
wX' filename = 'dataset/known_face.csv'
# Users shouldn't change the following parameters
stranger = 0 package = 1 check_flag = True flags =
[True, True]

# Send Message with IFTTT APIs def
send_message(url):
    response = requests.post(url)
print(response.content)

# Check the status of elements
def check_status():     global
flags     while check_flag:
if False in flags:
            index = flags.index(False)
flags[index] = True
time.sleep(30)

# Configure the face recognition process face_rgn =
FaceProcess(resize_frame=1, recognize_threshold=0.4,
detect_interval=0) face_rgn.load_database(filename)


# Configure the camera
# camera = cv2.VideoCapture(0)
# camera.set(cv2.CAP_PROP_FRAME_WIDTH, 1280.0) #
camera.set(cv2.CAP_PROP_FRAME_HEIGHT, 720.0)


# Check the flags
# th.Thread(target=check_status).start()


# Configure the window
# cv2.namedWindow('Detection', cv2.WINDOW_NORMAL)
# cv2.setWindowProperty('Detection', cv2.WND_PROP_FULLSCREEN,
cv2.WINDOW_FULLSCREEN)

@app.route('/') def
root():
    return "Hello"

@app.route('/test', methods=['POST']) def
test():
    print(request.data)     news = dict(request.form)     image_bytes =
io.BytesIO(base64.b64decode(news['img'].split(',')[1]))     im =
Image.open(image_bytes)     image = np.array(im)
     print(image)
     return "MY POST"

@app.route('/detect', methods=['GET', 'POST'])
def detect():     if request.method == 'POST':
# Get news from POST          news =
json.loads(request.data)         img_url =
news['img'].split(',')[1]

        # Convert url to image
img = url_to_image(img_url)

        # Detect people and recognize them
        face_positions = face_rgn.detect_people(img)

        # Return the results         face_pos =
{"face_pos": face_positions}         return
json.dumps(face_pos)     else:
        face_names = face_rgn.recognize_people()
face_nam = {"face_name": face_names}
         return json.dumps(face_nam)

@app.route('/save', methods=['POST']) def
save_remove():
    # Get news from POST     news =
json.loads(request.data)     status
= False     if news["mode"] ==
"remove":
        face_rgn.delete_data(filename, news["name"])
status = True     elif news["mode"] == "add":
        img_url = news['img'].split(',')[1]         img =
url_to_image(img_url)          status =
face_rgn.save_database(filename, news["name"], img)
     rtn_message = {"status": status}
return json.dumps(rtn_message)


def url_to_image(img_url):
    image_bytes = io.BytesIO(base64.b64decode(img_url))
im = Image.open(image_bytes)     img = np.array(im)
return cv2.cvtColor(img, cv2.COLOR_RGBA2RGB)


if __name__ == "__main__":
    # This is used when running locally only. When deploying to Google App
    # Engine, a webserver process such as Gunicorn will serve the app.
This
    # can be configured by adding an `entrypoint` to app.yaml.
    app.run(host='127.0.0.1', port=8081, debug=True)
pass
```

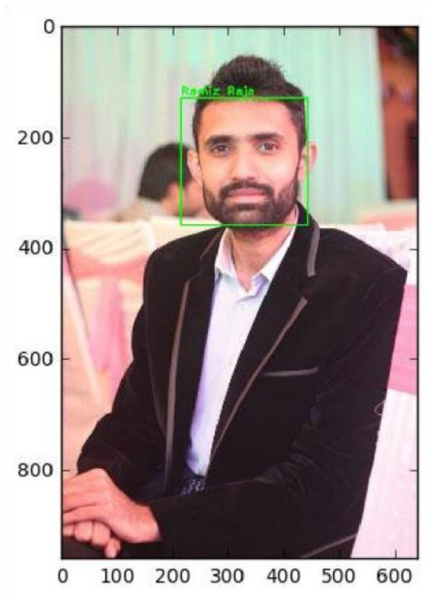
FACE RECOGNITION:

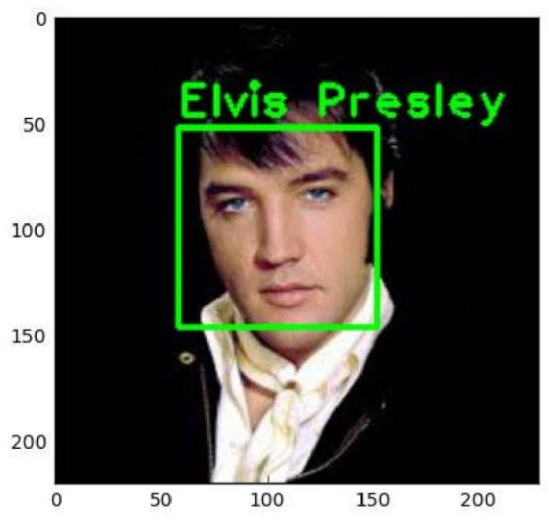
```
import face_recognition as fr
import os import cv2
import face_recognition
import numpy as np from
time import sleep
```

```
def get_encoded_faces():
    """     looks through the faces folder and
encodes all     the faces

    :return: dict of (name, image encoded)
    """
    encoded = {}
     for dirpath, dnames, fnames in
os.walk("./faces"):
        for f in fnames:             if
f.endswith(".jpg") or f.endswith(".png"):
face = fr.load_image_file("faces/" + f)
encoding = fr.face_encodings(face)[0]
encoded[f.split(".")[0]] = encoding
     return encoded


def unknown_image_encoded(img):
    """     encode a face given the
file name
    """     face =
fr.load_image_file("faces/" + img)
encoding = fr.face_encodings(face)[0]
     return encoding


def classify_face(im):
    """     will find all of the faces in a given image
and label     them if it knows what they are

    :param im: str of file path
    :return: list of face names
    """     faces = get_encoded_faces()
faces_encoded = list(faces.values())
known_face_names = list(faces.keys())
     img = cv2.imread(im, 1)
    #img = cv2.resize(img, (0, 0), fx=0.5, fy=0.5)
```

```
#img = img[:,:,::-1]
```

```
    face_locations = face_recognition.face_locations(img)
    unknown_face_encodings = face_recognition.face_encodings(img,
face_locations)
     face_names = []      for face_encoding in
unknown_face_encodings:
        # See if the face is a match for the known face(s)
matches = face_recognition.compare_faces(faces_encoded,
face_encoding)
        name = "Unknown"

        # use the known face with the smallest distance to the new face
face_distances = face_recognition.face_distance(faces_encoded,
face_encoding)         best_match_index = np.argmin(face_distances)
if matches[best_match_index]:
            name = known_face_names[best_match_index]
         face_names.append(name)
         for (top, right, bottom, left), name in zip(face_locations,
face_names):
            # Draw a box around the face             cv2.rectangle(img,
(left-20, top-20), (right+20, bottom+20), (255, 0, 0), 2)

            # Draw a label with a name below the face
cv2.rectangle(img, (left-20, bottom -15), (right+20, bottom+20), (255,
0, 0), cv2.FILLED)             font = cv2.FONT_HERSHEY_DUPLEX
cv2.putText(img, name, (left -20, bottom + 15), font, 1.0, (255, 255,
255), 2)


    # Display the resulting image
while True:

        cv2.imshow('Video', img)         if
cv2.waitKey(1) & 0xFF == ord('q'):
            return face_names


print(classify_face("test"))
```



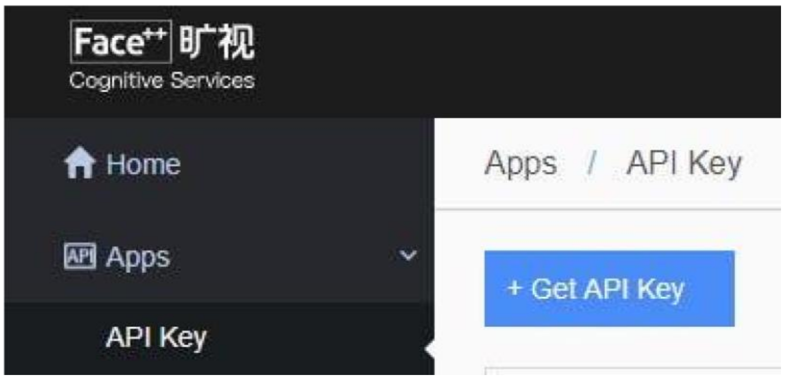


CLOUD IMPLEMENTATION:

Before jumping into coding, we need to set a few things up.

1. Bolt Cloud API Credentials
If you haven't already, to to cloud.boltiot.com and set up an account. Once you're logged in, link your WiFi Module with the Bolt Cloud. For that, download the Bolt IoT Setup App on your mobile phone. Follow the instructions given in the app to link your device with your account. This involves pairing the Bolt with local WiFi network. Once successfully linked, your dashboard will display your device. You can now get your Device ID and and API Key from the dashboard.

2. FacePlusPlus API Credentials
Another API service that we rely on in this project is *FacePlusPlus API*. It's a free platform that offers various kinds of image recognition services. We use it for facial identification. Create an account and go to the FacePlusPlus console. Go to *API Key* under *Apps* and click on *+Get API Key*. Note down the newly generated API Key and API Secret.



Now you should have the following ready:

```
private readonly string BOLT_DEVICE_ID = "BOLTXXXXXX";
private readonly string BOLT_API_KEY = "XXXXXXXXXXXXXXXX
XXXXXXXXXXXXXX";

private readonly string FPP_API_KEY = "XXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXX";
private readonly string FPP_API_SECRET = "XXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX";
```

We create a new global instance of the Bolt class called `myBolt`, through which we'll do all the future communications with the WiFi Module:

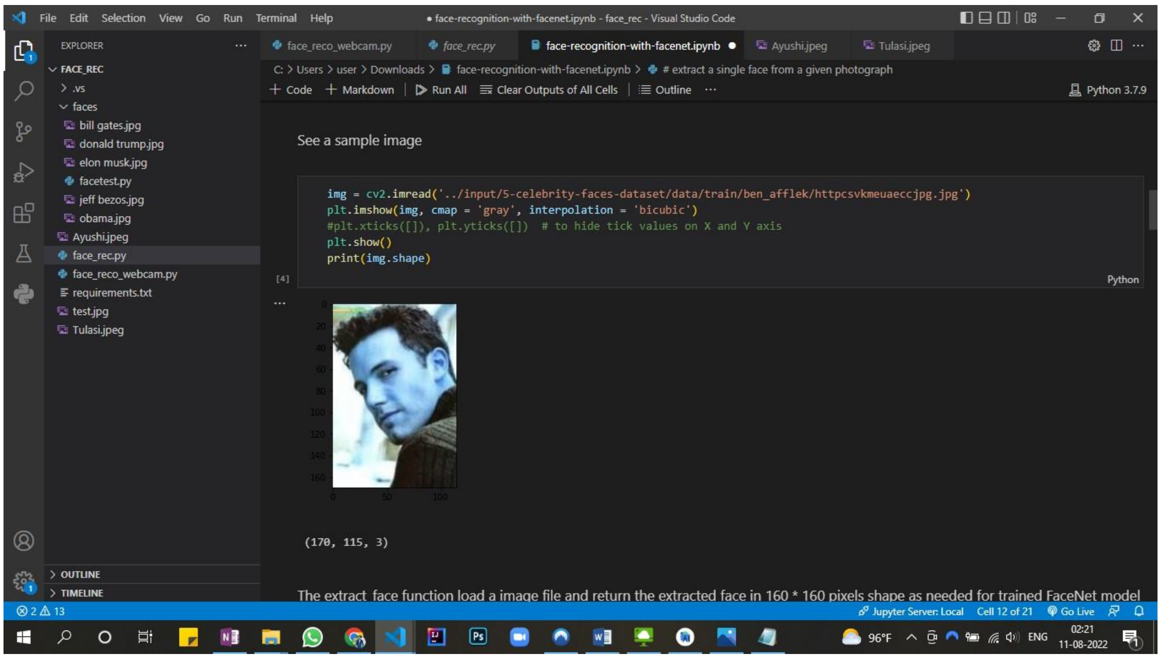
```
myBolt = new Bolt(BOLT_API_KEY, BOLT_DEVICE_ID);
```

That said, now let's see how our application performs some of the core functions.

<u>DATA ANALYTICS AND OUTCOMES:</u>

The detection accuracy of the system reaches 97.8%, the false detection rate is 0.8%, and the leakage rate is 0.8% the detection rate is 1.4%, and the accuracy of face recognition is 96.7%, which can well meet the requirements of contemporary face recognition.

[redacted]

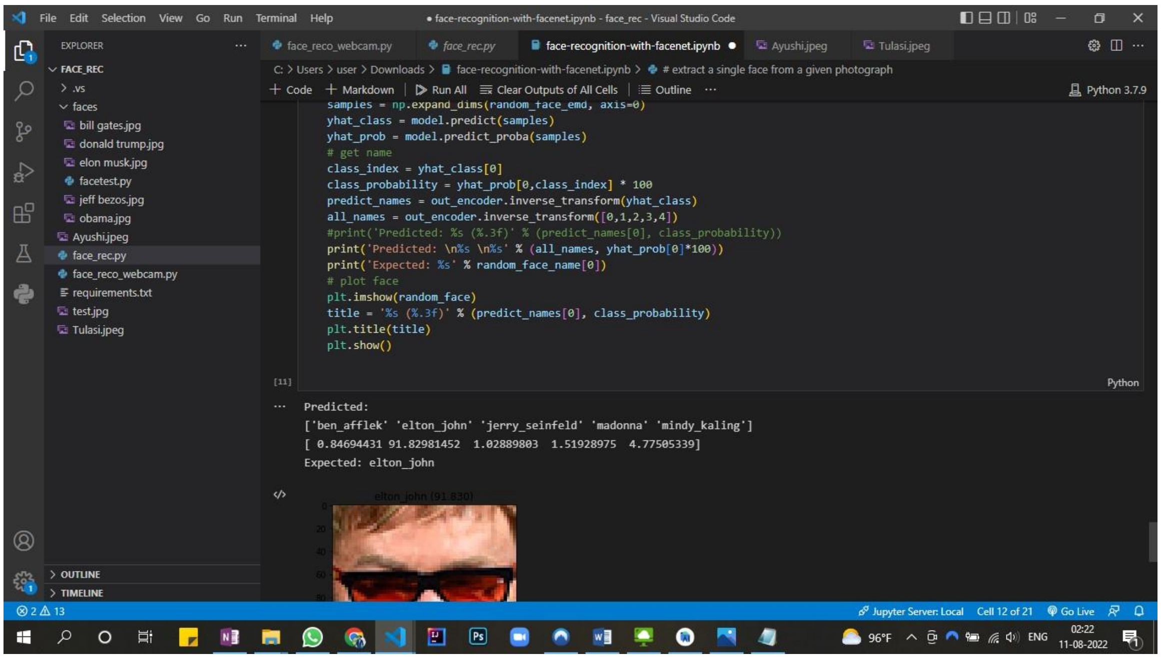


See a sample image

```
img = cv2.imread('../input/5-celebrity-faces-dataset/data/train/ben_afflek/httpcsvkmeuaeccjpg.jpg')
plt.imshow(img, cmap = 'gray', interpolation = 'bicubic')
#plt.xticks([]), plt.yticks([])  # to hide tick values on X and Y axis
plt.show()
print(img.shape)
```

```
(170, 115, 3)
```

The extract_face function load a image file and return the extracted face in 160 * 160 pixels shape as needed for trained FaceNet model

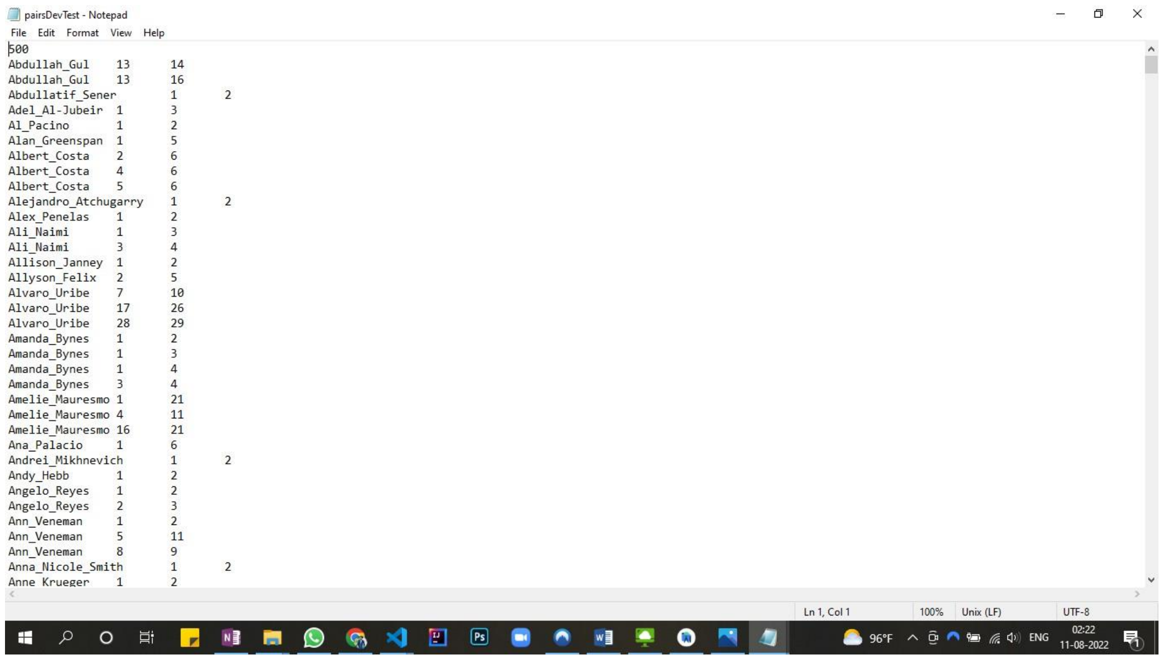


```
samples = np.expand_dims(random_face_emd, axis=0)
yhat_class = model.predict(samples)
yhat_prob = model.predict_proba(samples)
# get name
class_index = yhat_class[0]
class_probability = yhat_prob[0,class_index] * 100
predict_names = out_encoder.inverse_transform(yhat_class)
all_names = out_encoder.inverse_transform([0,1,2,3,4])
#print('Predicted: %s (%.3f)' % (predict_names[0], class_probability))
print('Predicted: \n%s \n%s' % (all_names, yhat_prob[0]*100))
print('Expected: %s' % random_face_name[0])
# plot face
plt.imshow(random_face)
title = '%s (%.3f)' % (predict_names[0], class_probability)
plt.title(title)
plt.show()
```

```
Predicted:
['ben_afflek' 'elton_john' 'jerry_seinfeld' 'madonna' 'mindy_kaling']
[ 0.84694431 91.82981452  1.02889803  1.51928975  4.77505339]
Expected: elton_john
```

TESTING DATA SET:

[redacted]



pairsDevTest - Notepad

```
500
Abdullah_Gul	13	14
Abdullah_Gul	13	16
Abdullatif_Sener	1	2
Adel_Al-Jubeir	1	3
Al_Pacino	1	2
Alan_Greenspan	1	5
Albert_Costa	2	6
Albert_Costa	4	6
Albert_Costa	5	6
Alejandro_Atchugarry	1	2
Alex_Penelas	1	2
Ali_Naimi	1	3
Ali_Naimi	3	4
Allison_Janney	1	2
Allyson_Felix	2	5
Alvaro_Uribe	7	10
Alvaro_Uribe	17	26
Alvaro_Uribe	28	29
Amanda_Bynes	1	2
Amanda_Bynes	1	3
Amanda_Bynes	1	4
Amanda_Bynes	3	4
Amelie_Mauresmo	1	21
Amelie_Mauresmo	4	11
Amelie_Mauresmo	16	21
Ana_Palacio	1	6
Andrei_Mikhnevich	1	2
Andy_Hebb	1	2
Angelo_Reyes	1	2
Angelo_Reyes	2	3
Ann_Veneman	1	2
Ann_Veneman	5	11
Ann_Veneman	8	9
Anna_Nicole_Smith	1	2
Anne_Krueger	1	2
```

RESULT:

The detection accuracy of the system reaches 97.8%, the false detection rate is 0.8%, and the leakage rate is 0.8% the detection rate is 1.4%, and the accuracy of face recognition is 96.7%, which can well meet the requirements of contemporary face recognition.

THANKING YOU