

# **SMART ATTENDANCE SYSTEM**

Enroll. No. (s) - 9919103096, 9919103121, 9919103150

Names of Student (s) - Rishabh Rajpurohit, P Paul Jonathan, Yash Raj Gairola



**Department of CSE/IT**  
**Jaypee Institute of Information Technology, Noida**

**December 2022**

Submitted in partial fulfillment of the Degree of Bachelor of Technology

in

Computer Science Engineering

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING & INFORMATION TECHNOLOGY

JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY, NOIDA

## ACKNOWLEDGEMENT

We would like to place on record our deep sense of gratitude to Dr. Neeraj Jain, Assistant Professor, Jaypee Institute of Information Technology, India for his generous guidance, help and useful suggestions.

We express our sincere gratitude to Dr. Neeraj Jain, Dept. of Computer Science and Engineering, JIIT, India, for his guidance, continuous encouragement and supervision throughout the course of present work.

We also wish to thank our classmates for their insightful comments and constructive suggestions to improve the quality of this project work. A blend of gratitude, pleasure and great satisfaction, is what we feel to convey our indebtedness to all those who have directly or indirectly contributed to the successful completion of our project work.

Name: Rishabh Rajpurohit  
Enrolment No.: 9919103096

Name: P Paul Jonathan  
Enrolment No.: 9919103121

Name: Yash Raj Gairola  
Enrolment No.: 9919103150

## DECLARATION

We hereby declare that this submission is our own work and that, to the best of our knowledge and beliefs, it contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma from a university or other institute of higher learning, except where due acknowledgment has been made in the text.

Date: 09 December, 2022

Name: Rishabh Rajpurohit  
Enrolment No.: 9919103096

Name: P Paul Jonathan  
Enrolment No.: 9919103121

Name: Yash Raj Gairola  
Enrolment No.: 9919103150

## CERTIFICATE

This is to certify that the work titled “Smart Attendance System Using Facial Biometrics” submitted by RISHABH RAJPUROHIT, P PAUL JONATHAN, and YASH RAJ GAIROLA of B.Tech of Jaypee Institute of Information Technology, Noida has been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of any other degree or diploma.

Dr. Neeraj Jain  
Assistant Professor (Senior Grade)  
09 December, 2022

## ABSTRACT

In modern times, face recognition has become one of the key aspects of computer vision. There are at least two reasons for this trend; the first is the commercial and law enforcement applications, and the second is the availability of feasible technologies after years of research. Due to the very nature of the problem, computer scientists, neuroscientists and psychologists all share a keen interest in this field. In plain words, it is a computer application for automatically identifying a person from a still image or video frame. In this paper we proposed an automated attendance management system. This system is based on face detection and recognition algorithms, which automatically detects the student when he enters the classroom and marks the attendance by recognizing him.

Chapter No. Topics	TABLE OF CONTENTS	Page No.
<b>Chapter-1 Introduction</b>		<b>7</b>
1.1 General Introduction		7
1.2 Problem Statement		8
1.3 Significance/Novelty of the Proposed Solution		8
1.4 Empirical Study/Field Survey/Experimental Study		8
HAAR CASCADE		8
LBPH		9
GTTS		9
PLAYSOUND		9
1.5 Brief Description of the Solution Approach		9
1.6 Comparison of existing approaches to the problem framed		10
<b>Chapter-2 Literature Survey</b>		<b>11</b>
2.1 Summary of Papers studied		11
Paper 1 - A Novel Technique to Detect Faces in a Group Photo		11
Paper 2 - Face Recognition from Group Photograph		11
Paper 3 - Two faces are better than one: Face recognition in group photographs		12
Paper 4 - Face Recognition System		12
Paper 5 - Face Recognition with Local Binary Patterns		13
Paper 6 - Face Recognition and Identification using Deep Learning Approach		13
2.2 Integrated summary of the Literature Studied		13
<b>Chapter-3 Requirement Analysis and Solution Approach</b>		<b>14</b>
3.1 Solution Approach (overall and module wise detailed description of your algorithm or hardware)		14
Fig-1 USE CASE DIAGRAM		14
Fig-2 ACTIVITY DIAGRAM		15
Fig-3 COMPONENT DIAGRAM		15
Fig-4 STATE DIAGRAM		16
<b>Chapter-4 Modeling and Implementation Details</b>		<b>17</b>
4.1 Implementation details and Issues		17
main.py		17
Train.py		20
Lbph.py		23
Screenshots:		27
Issues		29
<b>Results</b>		<b>29</b>
<b>Conclusion</b>		<b>29</b>
<b>References</b>		<b>30</b>

# Chapter-1 Introduction

## 1.1 General Introduction

Human Beings can distinguish between a particular face depending on a number of factors. One of the main objectives of computer vision is to create such a face recognition system that can emulate and eventually surpass this capability of humans. Though it is much easier to install face recognition system in a large scale setting, the actual implementation is very challenging as it needs to account for all possible appearance variations caused by change in illumination, facial features, variations in pose, image resolution, sensor noise, viewing distance, occlusions, etc. recently available face detection methods mainly rely on two approaches. The first one is a local face recognition system which uses facial features of a face e.g. nose, mouth, eyes etc. to associate the face with a person.

The second approach or global face recognition system uses the whole face to identify a person. The above two approaches have been implemented one way or another by various algorithms. The recent development of artificial neural networks and its possible applications in face recognition systems have attracted many researchers into this field. The intricacy of a face features originate from continuous changes in the facial features that take place over time. Regardless of these changes we are able to recognize a person very easily. Thus the idea of imitating this skill inherent in human beings by machines can be very rewarding. Though the idea of developing an intelligent and self-learning may require supply of sufficient information to the machine. Considering all the above mentioned points and their implications we have tried to gain some experience with some of the most commonly available face recognition algorithms and also compare and contrast the use of neural networks in this field.

Attendance is the measure for calculating productivity whether you are at the workplace or college/university. It is meant to keep track of every individual's availability. The traditional ways of recording an attendance involved maintaining a register and marking people present/absent on particular days. The process of maintaining attendance registers is a cumbersome task and is extremely difficult to track historical records as most of the registers find themselves stacked in cupboards in a degraded state. With the advent of technology and its revolutionary changes, recording an attendance too has become tech-savvy. The smart attendance system in today's world is the best way to record attendance without any irregularities. Moreover, the historical records can be easily accessed at the touch of a button within a few minutes.

## 1.2 Problem Statement

The traditional System of maintaining attendance in some schools still uses Registers and is a slow procedure as compared to our proposed system. In most of the universities too, the webkiosk system is manual, i.e. the faculty announces the names of students one by one but as compared to schools attendance is still stored in a centralized database because of a huge database of student data. A smart attendance system is very crucial and important for any kind of business. A good and efficient attendance system helps in monitoring the punctuality of employees and managing the absence of people. A smart attendance system enables setting up the attendance workflows and maintaining a proper validation of employee time-sheets.

## 1.3 Significance/Novelty of the Proposed Solution

Our smart attendance system is very useful in busy working environments like schools. We have introduced the text to speech functionality which has many imminent benefits. One of the major benefits of this component is that our system will be able to mark attendance of visually impaired students as well. It also makes our system more presentable and easy to use. Our system is very well structured and due to that it can be easily enhanced into a more complex system. Our format of attendance record is very intricate and understandable. The system can be configured in mobile devices and the managers can have a track of the employee's location if required. The smart attendance system saves a lot of time and money and is a highly secure process in maintaining the database on the server. We aim to make a smart attendance system based on facial recognition. Our aim is to make this system more convenient and feasible. We plan to replace webcam with raspberry pi camera as raspberry pi is quite cheap and portable.

## 1.4 Empirical Study/Field Survey/Experimental Study

### HAAR CASCADE

It is an Object Detection Algorithm used to identify faces in an image or a real time video. The algorithm uses edge or line detection features proposed by Viola and Jones in their research paper "Rapid Object Detection using a Boosted Cascade of Simple Features" published in 2001. The algorithm is given a lot of positive images consisting of faces, and a lot of negative images not consisting of any face to train on them. The objective is to find out the sum of all the image pixels lying in the darker area of the haar feature and the sum of all the image pixels lying in the lighter area of the haar feature. And then find out their differences.



## LBPH

Local Binary Pattern (LBP) is a simple yet very efficient texture operator which labels the pixels of an image by thresholding the neighborhood of each pixel and considers the result as a binary number. It was first described in 1994 (LBP) and has since been found to be a powerful feature for texture classification. It has further been determined that when LBP is combined with histograms of oriented gradients (HOG) descriptors, it improves the detection performance considerably on some datasets. Using the LBP combined with histograms we can represent the face images with a simple data vector. As LBP is a visual descriptor it can also be used for face recognition tasks.

## GTTS

gTTs (*Google Text-to-Speech*), a Python library and CLI tool to interface with Google Translate's text-to-speech API. Writes spoken mp3 data to a file, a file-like object (bytestring) for further audio manipulation, or stdout. It features flexible pre-processing and tokenizing.

## PLAYSOUND

The playsound module is a cross platform module that can play audio files. This module doesn't have any dependencies.

## 1.5 Brief Description of the Solution Approach

During training, the students are made to sit in front of the camera or the photos are uploaded to the database, they are preprocessed by extracting faces and storing it in a black and white format, and resized to 150x150 images, then the local binary pattern histogram algorithm is run which gives a value for each of the 22500 pixels of each image in the database which allows us to plot a 22500 dimensional space which contains each image in the training data. [6]

During runtime, a student's face is captured and the same preprocessing process is applied to this and it is mapped on to the 22500 space and the closest point is found and the classification of that point is given as the classification of the current point and gives a surety value for which we have taken 85% as the threshold, also known as the confidence level [4], in addition to this we have also added text to speech support so that the faculty taking the attendance will know when to coordinate the students and the students will also know that who has been marked present, this is also useful for visually impaired people, and since we are also displaying text on the terminal window, it is also useful to auditorily impaired people. When the model recognizes the student it stores it in the database. Our project enables an organization to smartly manage and store attendance of employees/staff or students in case of educational organizations. The user has to stand within 50 cm of the device to be able to successfully detect his face with high accuracy[4]. The Attendance folder in our project directory stores a record of csv files named as per the current date. The fields of csv files include columns like Subject Name, Class Time and Class Type including Lectures, Tutorials or Labs and at last Faculty name. Redundancy in records is avoided using a python list which is checked everytime we recognize a new face and if the face is not found in the python list we mark him/her present and store the details accordingly.

## 1.6 Comparison of existing approaches to the problem framed

Biometric-based techniques are promising options for recognizing individuals in recent years. Biometric-based technologies include identification based on physiological characteristics (such as face, fingerprints, finger geometry, hand geometry, hand veins, palm, iris, retina, ear and voice) and behavioral traits (such as gait, signature and keystroke dynamics). Face Recognition appears to offer several advantages over the above mentioned methods, a few of which are outlined here:

- a. Almost all these technologies require some voluntary action by the user, i.e The user needs to place his hand on a hand-rest for fingerprinting or hand geometry detection and has to stand in a fixed position in front of a camera for iris or retina identification. However, face recognition can be done without any explicit action or participation on the part of the user since face images can be acquired from a distance by a camera. This is particularly beneficial for security and surveillance purposes.
- b. Data acquisition in general is a drawback with problems for other biometrics techniques that rely on hands and fingers that can be rendered useless if the epidermis tissue is damaged in some way . Iris and retina identification require expensive equipment and are much too sensitive to any body motion. Voice recognition is susceptible to background noises in public places. However, facial images can be easily obtained with a couple of inexpensive fixed cameras. Good face recognition algorithms and appropriate pre-processing of the images can compensate for noise and slight variations in orientation, scale and illumination.
- c. Finally, technologies that require multiple individuals to use the same equipment to capture their biological characteristics potentially expose the user to the transmission of germs and impurities from other users .However, face recognition is totally non-intrusive and does not carry any such health risks. See [2]

# Chapter-2 Literature Survey

## 2.1 Summary of Papers studied

### Paper 1 - A Novel Technique to Detect Faces in a Group Photo

*By Saravanan Chandran, Assistant Professor, National Institute of Technology, Durgapur, West Bengal, India.*

This paper discusses the various methods to detect faces and combines them to make a new method to detect the area of all faces in an image. The algorithms described were, Firstly, H.H.K Tin's approach which involved feature extraction based on facial features using Principal Component Analysis, Secondly, the paper talks about H. Naeem's method is based on the EigenFaces algorithm. A set of images were represented as long vectors. This produced face space with each image representing a single point. The eigenfaces helped to detect lower dimensional space in which shorter vectors will describe face images. Thirdly, A. Mall and S. Ghosh proposed neural network based face detection approach, Fourthly, Hausdorff Distance is discussed, proposed by O.Jesorsky which achieved between 91-94% accuracy which measures the Hausdorff Distance between the given image and the images present in the training sample and classifies it as the nearest. After this in a short section adaboost, CNN, and other methods are used and a conclusion is reached that the more computationally intensive an algorithm is, the rate of detecting the face increases long with the time complexity as a tradeoff, so a balance must be made when using such methods. The proposed new method in this paper involves 3 phases, pre-processing: in this case it involves using a smoothing filter to reduce the noise in the image, in step 2 the image is converted to black and white to reduce the number of color channels and so reducing the complexity by a third, in the final step, the core algorithm used is the boundary identification algorithm, which aims to solve the boundary problem for any given image and locates the edges of the image. The bounded regions in this new processed image are given to the model which compares every boundary to a library of possible facial boundary patterns and gives a result based on the h value which quantifies the similarity of the boundary with the already known and identifies it as a face or not. According to the paper, this technique has produced the following result: true face region identification ratio is 70.83%, false face region identification ratio is 8.33% and the face regions not identified ratio is 29.17%.

### Paper 2 - Face Recognition from Group Photograph

*Kavita Shelke, Asst. Prof. Department of Computer Engg, FCRIT,Vashi, Navi Mumbai, India*

The paper first about the problem of having many faces in a single frame of video like for example in a surveillance video camera and thus the need for recognising multiple faces in the same frame. The paper then talks about the benefits of facial recognition as a method of biometric identification over other methods in terms of cost, biological safety, and need for voluntary action. The next section is about the applications of facial recognition, i.e Information Security, Access Management, Biometrics, Law Enforcement, Personal Security, and Entertainment. The proposed algorithm is given as follows and is explained in further detail at each step. Input Images: color photos of any group photos

Skin Detection Algorithm for Face Detection: this is the main algorithm used in this model, which consists of the following steps

- Finding faces with controlled background
- Finding faces by color
- Finding faces by motion
- Finding faces in unconstrained scenes
- Geometry based
- Appearance based
- Edge based
- Color Analysis in the following color spaces - RGB, HSV, YCbCr

Gray Level Co-occurrence Matrix for Face Recognition: 5 principal features of faces are extracted from the images using the matrices.

### Paper 3 - Two faces are better than one: Face recognition in group photographs

*Ohil K. Manyam, David Kriegman, University of California San Diego, La Jolla Neeraj Kumar, Peter Belhumeur, Columbia University, New York*

The paper describes the lack of uniformity of conditions in the photos of the same person or different people taken across different environments, and implies that group photos have an advantage in which all the faces in the image have similar lighting conditions and we can make comparisons of different faces, these facial comparisons are known as relative features, for example height difference, color differences, are called describable visual attributes. The model used in this paper is the Bayesian recognition approach, which consists of a baseline model, a conditional probability model, and a joint probability model. The datasets used for training this model are, the Buffy Dataset, which is a dataset consisting of 120,000 frames extracted from the TV show Buffy the Vampire; and a personal photo album belonging to the authors. The paper also acknowledges the difficulty of data scarcity in reference to group photos and the sheer number of models which would be needed to build for doing this task, i.e pairwise feature extraction. The paper states that the result of this approach led to an 88-90% accuracy on testing data.

### Paper 4 - Face Recognition System

*Shivam Singh, Prof. S. Graceline Jasmine, Department of SCSE, Vellore Institute of Technology, Chennai, Tamil Nadu, India*

Keywords: <https://www.ijert.org/research/face-recognition-system-IJERTV8IS050150.pdf>, paper uses haar\_cascade for frontal face detection, preprocessing: face image is resized to 100x100 histogram normalization used to make image more clear, db\_storage: facial feature are stored in the database, post processing: names are shown in video output, This paper compares a few algorithms for facial recognition which are: Neural Networks, Principal Component Analysis Eigenfaces Fisherfaces, Local Binary Pattern Histograms. Basically , here they used KLT algorithm, viola jones used haar cascade classifier and PCA for feature extraction. In this project , they have also run some experiments on other facial recognition algorithms. They have found that the PCA algorithm outperforms all the other algorithms. And he has also acknowledged that poor lighting is a disadvantage here.

## Paper 5 - Face Recognition with Local Binary Patterns

*Timo Ahonen, Abdenour Hadid, and Matti Pietikäinen Machine Vision Group, Infotech Oulu PO Box 4500, FIN-90014 University of Oulu, Finland*

In this paper, they have described an approach of recognising faces using local binary pattern histogram. They described how the approach of lbpf works, in lbpf there are several formulas for calculating distance between two histograms to find out the classification. In this paper, they have run some performance tests on several algorithms and distance measures, using the CSU face identification and evaluation system using the ferret test. The results of the test were that in LBP category, fb, fc and dup distance measures were outperforming the others with 79% accuracy and PCA was 65%, BIC 37% and 42% for EBGM.

## Paper 6 - Face Recognition and Identification using Deep Learning Approach

*KH Teoh, RC Ismail, SZM Naziri, R Hussin, MNM Isa and MSSM Basir, School of Computing and Informatics, Albukhary International University, Kedah Malaysia, School of Microelectronic Engineering, Universiti Malaysia Perlis, Perlis, Malaysia, Department of Electrical Engineering, Politeknik Mukah, Sarawak, Malaysia.*

Basically, authors here used haar cascade for face detection. They have found that there are several cases in which haar cascade does not detect faces which are a distance below 60 cm from the camera tha haar cascade do not detect faces. For the model they have used convolutional and deep neural networks for the model which have accuracy of 91.7% in recognising images and 86.7% accuracy in video.

## 2.2 Integrated summary of the Literature Studied

Since, from all these research papers we can see that, lbph has the highest accuracy rate, across 10,000 + images and also is the easiest to implement. so, we have used lbph in our project. All the research papers have used haar cascade (viola jones algorithm) for face detection. One paper used an interesting approach by comparing different types of distances between histograms, to find out which distance is more optimized and give better results. for example, chi square distance, euclidean distance, manhattan distance etc. Chi square had the best results which was fascinating to know .

# Chapter-3 Requirement Analysis and Solution Approach

## 3.1 Solution Approach (overall and module wise detailed description of your algorithm or hardware)

Fig-1 USE CASE DIAGRAM

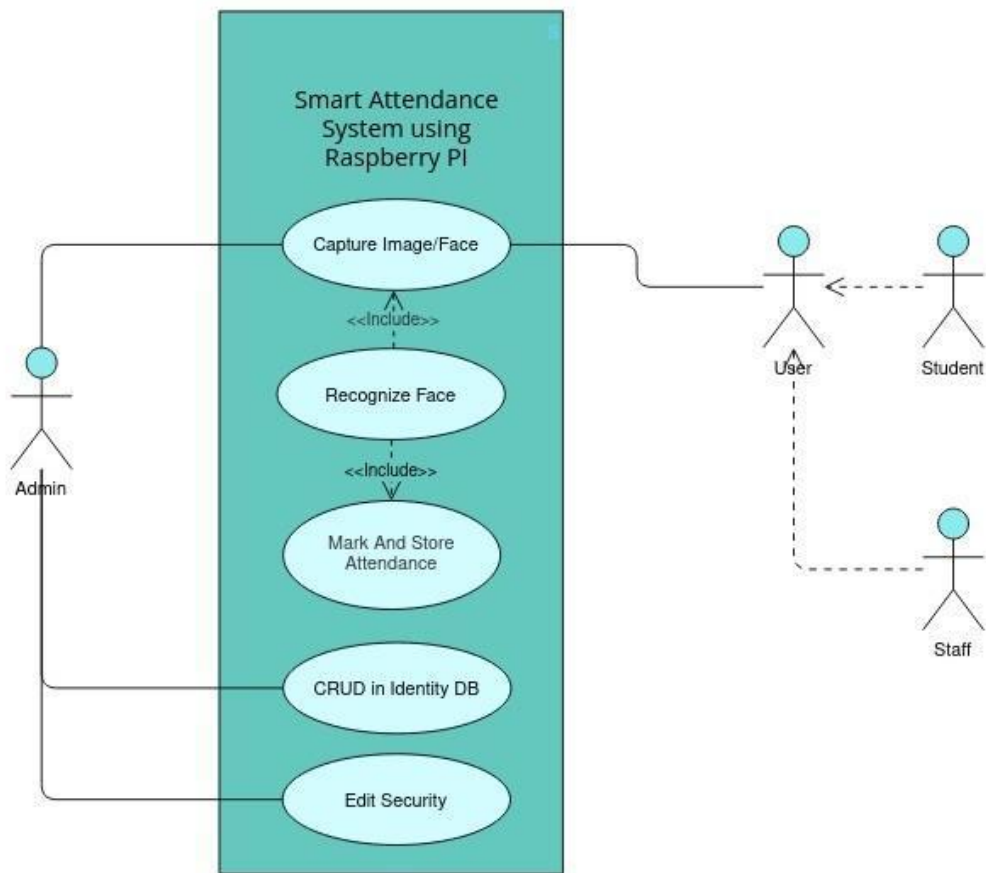


Fig-2 ACTIVITY DIAGRAM

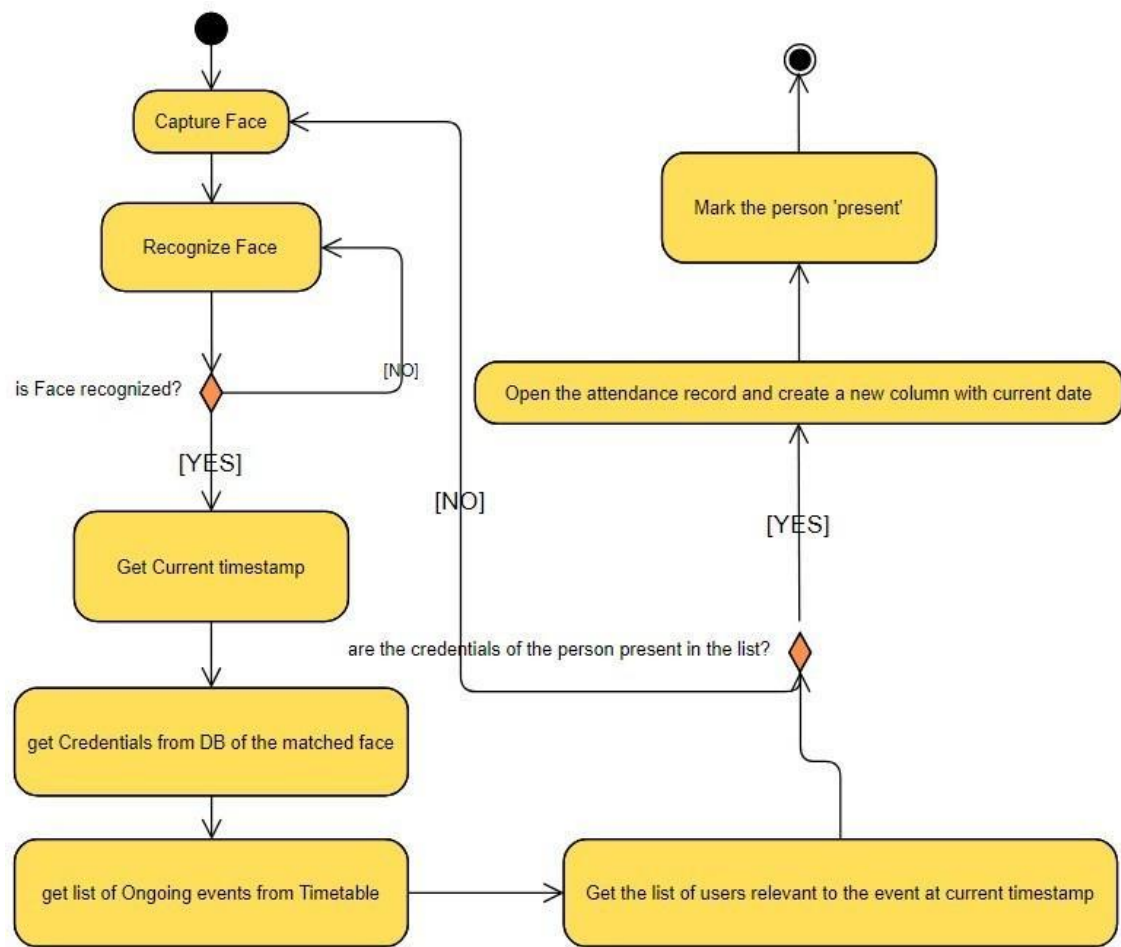


Fig-3 COMPONENT DIAGRAM

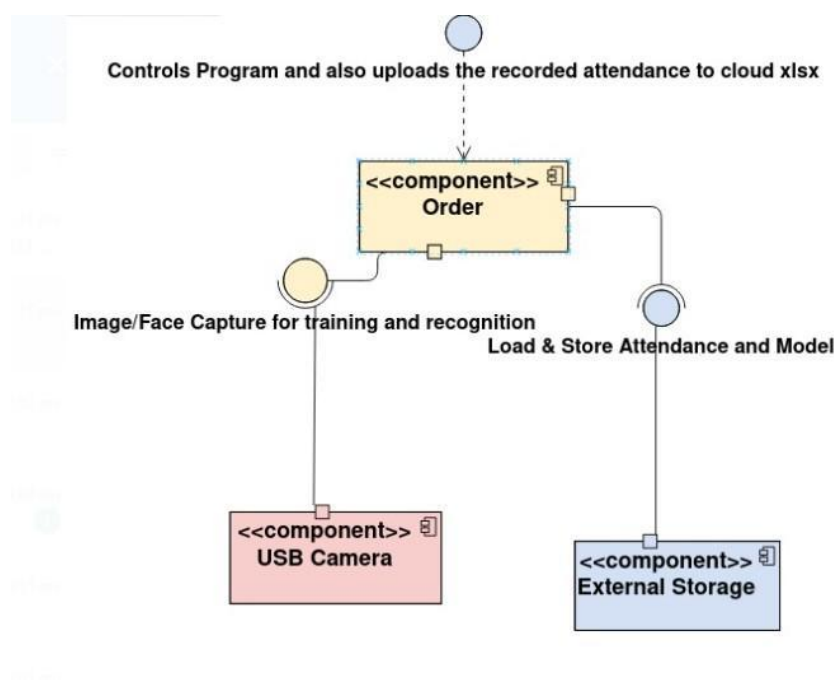
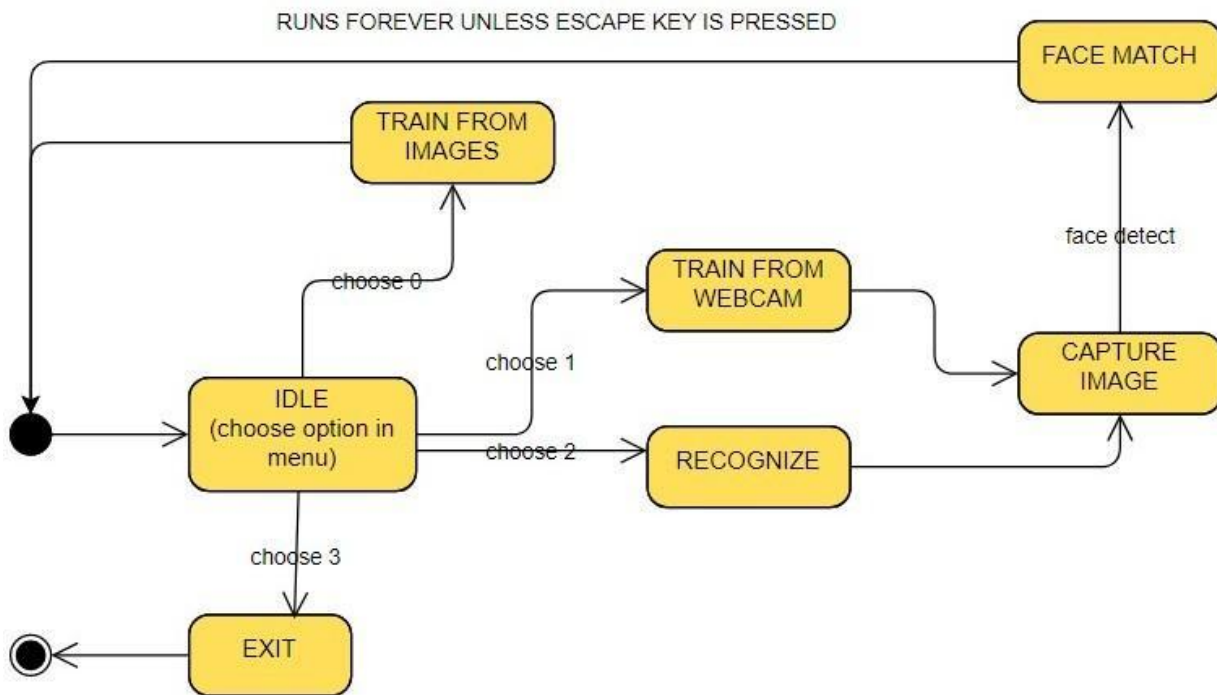


Fig-4 STATE DIAGRAM



Our code is divided into 4 modules which are described as follows:

- Lbph.py: This module contains 2 main functions, local binary pattern: which gives the lbph for a particular pixel, and trainlbph: which gives an array of lbph values for the entire 150x150 image. [6]
- Haar\_cascade\_frontalface\_default.xml: this is a .xml containing filters for nose, eyes, mouth, and other facial features which are run pixel-by-pixel across the image to find the location of these facial features and use them to triangulate the location of the face.[1, 2, 3, 4, 5, 6]
- Train.py: this module contains two functions for training from a directory containing saved photographs of students, sorted into directories based on enrollment number(or name as is convenient) and for training directly from the camera feed The program asks for a label name, The camera is turned on and the program captures images from the camera feed and detects faces using haar\_cascade and stored in the data folder under the label name given recognizing the person in front of the camera, the lbph module takes all the images and returns the lbph array of all the images and it is stored as a numpy array, which is our model[6], there is also another function play\_sound, which takes in a string, converts it to an audio file (using google's text to speech api) and plays the audio file.
- Main.py: This is the module, which runs and manages all of the program and the other modules. The main module consists a function which makes a .csv file to store the daily attendance of students and for every lecture and an infinite loop which asks for the credentials of the faculty recording the attendance and other details such as type of class, class timing and course name, there is also a LoadModelAndRun function which loads the model and captures an image from the camera, identifies faces in it and applies preprocessing on the first face detected and finds the closest lbph and returns an accuracy value or surety index which quantifies the closeness of the test face to one of the training images[1], after this the detected person's details are stored in the .csv file and the program sleeps for a time to allow the next student to come forward .
- attendance-<date>.csv: this is the csv file which is created everyday, the columns of this csv are: S-No, Enrollment-No, Time-Stamp(Hour:Min:Sec), Subject-Name, Class-Type, Faculty-Name, and Class-Time.



# Chapter-4 Modeling and Implementation Details

## 4.1 Implementation details and Issues

```
main.py

import cv2
import numpy as numpy
import lbph as lr
import os
from time import sleep
from gtts import gTTS
from playsound import playsound
from datetime import date,datetime
import pandas as pd
from train import TrainFromSavedPhotos,TrainFromWebcam

recognized__students__list = list()
date_today_compressed = date.today().strftime("%a-%d-%m-%y")
date_today_detailed = date.today().strftime("%a-%d-%B-%Y")

attendance__dir = "C:\\Users\\Rishabh Rajpurohit\\Documents\\majorP\\Attendance"

sub__name = "default subject"
cls__type = "lecture"
faculty__name = "default faculty"
cls__time = "9am"
s__no = 1

if f'Attendance-{date_today_detailed}.csv' not in os.listdir(attendance__dir):
    with open(f'Attendance/Attendance-{date_today_detailed}.csv','w') as f:
        f.write('S-No,Enrollment-No,Time-Stamp(Hour:Min:Sec),Subject-Name,Class-Type,Faculty-Name,Class-Time')

savedModelLocation = 'C:\\Users\\Rishabh Rajpurohit\\Documents\\majorP\\code\\trained_face_model.npy'
baseDir = 'C:\\Users\\Rishabh Rajpurohit\\Documents\\majorP\\res\\data'
fn_haar = 'C:\\Users\\Rishabh Rajpurohit\\Documents\\majorP\\code\\haarcascade_frontalface_default.xml'
fn_dir = 'C:\\Users\\Rishabh Rajpurohit\\Documents\\majorP\\res\\database'

sound__file = 'C:\\Users\\Rishabh Rajpurohit\\Documents\\majorP\\testvoice.mp3'
def play__sound(s):
    mytext = s
    language = 'en'
    myobj = gTTS(text=mytext, lang=language, slow=False)
    myobj.save(sound__file)
    playsound(sound__file)
```

```
os.remove(sound__file)
```

```
def LoadModelAndRun():
```

```
    s__no = 1
```

```
    recognized__students__list.clear()
```

```
    sub__name, cls__type, faculty__name, cls__time = input("\nEnter the following Details:\nFormat:
```

```
<Sub_name>,<Class_type>,<Faculty_Name>,<Class_time>\n--->").split(',')
```

```
    try:
```

```
        trained_face_recognizer=numpy.load(savedModelLocation)
```

```
    except:
```

```
        print("\n\nTrain the model first!")
```

```
    return
```

```
# Load prebuilt model for Frontal Face
```

```
(im_width, im_height) = (68, 68)
```

```
# Part 2: Use fisherRecognizer on camera stream
```

```
(images, lables, names, id) = ([], [], {}, 0)
```

```
for (subdirs, dirs, files) in os.walk(fn_dir):
```

```
    for subdir in dirs:
```

```
        names[id] = subdir
```

```
        subjectpath = os.path.join(fn_dir, subdir)
```

```
        for filename in os.listdir(subjectpath):
```

```
            path = subjectpath + '/' + filename
```

```
            lable = id
```

```
            images.append(cv2.imread(path, 0))
```

```
            lables.append(int(lable))
```

```
        id += 1
```

```
face_cascade = cv2.CascadeClassifier(fn_haar)
```

```
webcam = cv2.VideoCapture(0,cv2.CAP_DSHOW)
```

```
while True:
```

```
    (_, im) = webcam.read()
```

```
    gray = cv2.cvtColor(im, cv2.COLOR_BGR2GRAY)
```

```
    faces = face_cascade.detectMultiScale(gray, 1.3, 5)
```

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

```
        cv2.rectangle(im,(x,y),(x+w,y+h),(255,0,0),2)
```

```
        face = gray[y:y + h, x:x + w]
```

```
        face_resize = cv2.resize(face, (im_width, im_height))
```

```
        prediction=lr.predict_lbph(face_resize,trained_face_recognizer,lables)
```

```
        cv2.rectangle(im, (x, y), (x + w, y + h), (0, 255, 0), 3)
```

```
        if (prediction[1])<=100 and (prediction[1])>85:
```

```
            current__time = datetime.now().strftime('%H:%M:%S')
```

```
            df = pd.read_csv(f'Attendance/Attendance-{date_today_detailed}.csv')
```

```
            if(recognized__students__list.count(names[prediction[0]])==0):
```

```

        recognized__students__list.append(names[prediction[0]])
        with open(f'Attendance/Attendance-{date_today_detailed}.csv', 'a') as f:

f.write(f'\n{s__no},{names[prediction[0]]},{current__time},{sub__name},{cls__type},{faculty__name},{cls__time
}')

        s__no += 1
        print('%s - %s' % (names[prediction[0]], "marked PRESENT"))
        cv2.putText(im, '%s - %.0f%s' % (names[prediction[0]], prediction[1], "%"), (x-10, y-10),
cv2.FONT_HERSHEY_PLAIN, 1, (0, 255, 0))
        play__sound(str(names[prediction[0]]) + "marked PRESENT")
        #play__sound("next student, please come forward")

    else:
        cv2.putText(im, 'not recognized', (x-10, y-10), cv2.FONT_HERSHEY_PLAIN, 1, (0, 255, 0))

#cv2.imshow('OpenCV', im)
key = cv2.waitKey(10)
if key == 27:
    break
cv2.destroyAllWindows()

def init():
    while True:
        #play__sound("Choose 0 to train from Saved Photos, Choose 1 to train from Webcam, Choose 2 to Run
Recognition and Choose 3 to Exit.")
        print('_____ \n\n0.Train from Saved Photos\n1.Train From
Webcam \n2.Run \n3.Reset Model \n4.Exit:\n')
        user=input("-->")
        if user == '0':
            TrainFromSavedPhotos()

        elif user == '1':
            TrainFromWebcam()

        elif user=='2':
            try:
                LoadModelAndRun()
            except:
                break
            finally:
                init()

        elif user=='3':
            os.remove(savedModelLocation)

```

```

elif user == '4':
    print("Exiting!")
    break
else:
    print("Enter Valid input!\n\n")
    #play__sound("Please enter a valid input")

```

```

init()

```

## Train.py

```

import os,time
import cv2
import numpy
import lbph as lr
from playsound import playsound
from gtts import gTTS

```

```

savedModelLocation = 'C:\\Users\\Rishabh Rajpurohit\\Documents\\majorP\\code\\trained_face_model.npy'
baseDir = 'C:\\Users\\Rishabh Rajpurohit\\Documents\\majorp\\res\\data'
fn_haar = 'C:\\Users\\Rishabh Rajpurohit\\Documents\\majorp\\code\\haarcascade_frontalface_default.xml'
fn_dir = 'C:\\Users\\Rishabh Rajpurohit\\Documents\\majorp\\res\\database'

```

```

sound__file = 'C:\\Users\\Rishabh Rajpurohit\\Documents\\majorp\\testvoice.mp3'

```

```

def play__sound(s):
    mytext = s
    language = 'en'
    myobj = gTTS(text=mytext, lang=language, slow=False)
    myobj.save(sound__file)
    playsound(sound__file)
    os.remove(sound__file)

```

```

def TrainFromSavedPhotos():
    persons = os.listdir(baseDir)
    print("Fetching Data...")
    #play__sound('fetching data')
    for person in persons:
        images = os.listdir(baseDir+"\\ "+person)
        count = 0
        size = 4
        fn_name = person

```

```

path = os.path.join(fn_dir, fn_name)
if not os.path.isdir(path):
    os.mkdir(path)
(im_width, im_height) = (68, 68)
haar_cascade = cv2.CascadeClassifier(fn_haar)

for image in images:
    pathOfImg = baseDir+"\\"+person+"\\"+image
    im = cv2.imread(pathOfImg)
    gray = cv2.cvtColor(im, cv2.COLOR_BGR2GRAY)
    mini = cv2.resize(gray, ((int)(gray.shape[1] / size), (int)(gray.shape[0] / size)))
    faces = haar_cascade.detectMultiScale(mini)
    faces = sorted(faces, key=lambda x: x[3])
    if faces:
        count=count+1
        face_i = faces[0]
        (x, y, w, h) = [v * size for v in face_i]
        face = gray[y:y + h, x:x + w]
        face_resize = cv2.resize(face, (im_width, im_height))
        pin=sorted([int(n[:n.find('.')]) for n in os.listdir(path) if n[0]!='.']+[-1])[-1] + 1
        cv2.imwrite('%s/%s.png' % (path, pin), face_resize)
        cv2.rectangle(im, (x, y), (x + w, y + h), (0, 255, 0), 3)
        cv2.putText(im, fn_name, (x - 10, y - 10), cv2.FONT_HERSHEY_TRIPLEX,2,(0, 255, 0))
        if(count>=20):
            break;

print('Training...')
#play__sound('the model is now training')

# Create a list of images and a list of corresponding names
(images, lables, names, id) = ([], [], {}, 0)
for (subdirs, dirs, files) in os.walk(fn_dir):
    for subdir in dirs:
        names[id] = subdir
        subjectpath = os.path.join(fn_dir, subdir)
        for filename in os.listdir(subjectpath):
            path = subjectpath + '/' + filename
            lable = id
            images.append(cv2.imread(path, 0))
            lables.append(int(lable))
        id += 1
(im_width, im_height) = (68, 68)

# Create a Numpy array from the two lists above
(images, lables) = [numpy.array(lis) for lis in [images, lables]]

```

```

trained_face_recognizer=lr.train_lbph(images)
print('done')
#play__sound('the model is now trained')
numpy.save(savedModelLocation,trained_face_recognizer)

def TrainFromWebcam():
    count = 0
    size = 4
    #play__sound('Please enter your name or enrollment number')
    fn_name = input('Enter Your Enrollment Number: ')
    path = os.path.join(fn_dir, fn_name)
    if not os.path.isdir(path):
        os.mkdir(path)
    (im_width, im_height) = (68, 68)
    haar_cascade = cv2.CascadeClassifier(fn_haar)
    webcam = cv2.VideoCapture(0)

    print("-----Ensure that the room is well lit-----")
    print("-----Taking pictures-----")
    #play__sound("Ensure the room is well lit and the distance between face and camera is not more than half a meter
    for better accuracy")
    # The program loops until it has a few images of the face.

    while count < 20:
        (rval, im) = webcam.read()
        gray = cv2.cvtColor(im, cv2.COLOR_BGR2GRAY)
        mini = cv2.resize(gray, ((int)(gray.shape[1] / size), (int)(gray.shape[0] / size)))
        faces = haar_cascade.detectMultiScale(mini)
        faces = sorted(faces, key=lambda x: x[3])
        if faces:
            face_i = faces[0]
            (x, y, w, h) = [v * size for v in face_i]
            face = gray[y:y + h, x:x + w]
            face_resize = cv2.resize(face, (im_width, im_height))
            pin=sorted([int(n[:n.find('.')]) for n in os.listdir(path) if n[0]!='.' ]+[0])[-1] + 1
            cv2.imwrite('%s/%s.png' % (path, pin), face_resize)
            cv2.rectangle(im, (x, y), (x + w, y + h), (0, 255, 0), 3)
            cv2.putText(im, fn_name, (x - 10, y - 10), cv2.FONT_HERSHEY_TRIPLEX,2,(0, 255, 0))
            time.sleep(0.38)
            count += 1

    #cv2.imshow('OpenCV', im)
    key = cv2.waitKey(10)

```

```

    if key == 27:
        break
print(str(count) + " images taken and saved to " + fn_name + " folder in database ")
cv2.destroyAllWindows()
webcam.release()
size = 4
print('Training...')
#play__sound('the model is now training')
# Create a list of images and a list of corresponding names
(images, labes, names, id) = ([], [], {}, 0)
for (subdirs, dirs, files) in os.walk(fn_dir):
    for subdir in dirs:
        names[id] = subdir
        subjectpath = os.path.join(fn_dir, subdir)
        for filename in os.listdir(subjectpath):
            path = subjectpath + '/' + filename
            lable = id
            images.append(cv2.imread(path, 0))
            labes.append(int(lable))
        id += 1
(im_width, im_height) = (68, 68)

# Create a Numpy array from the two lists above
(images, labes) = [numpy.array(lis) for lis in [images, labes]]
trained_face_recognizer=lr.train_lbph(images)
print('done')
#play__sound('the model is now trained')
numpy.save(savedModelLocation,trained_face_recognizer)

```

## Lbph.py

```

import cv2, sys, numpy, os,time
def train_lbph(images):
    images_lbp=localBinaryPattern(images)
    images_histograme=getHistograms(images_lbp)
    return images_histograme

def localBinaryPattern_wh(images):
    (s1,s2,s3)=images.shape
    images_processed=numpy.zeros((s1,(s2-4),(s3-4)))
    window=3
    for i in range(s1):
        temp2=numpy.zeros((s2,s3))
        for j in range(s2-window-1):
            for k in range(s3-window-1):

```

```

        temp2[i,j:j+window,k:k+window]=images[i,j:j+window,k:k+window]
        temp2[i,j:j+window,k:k+window]=getPattern_wh(temp2[i,j:j+window,k:k+window])
    images_processed[i]=temp2
return images_processed

```

```

def localBinaryPattern(images):
    (s1,s2,s3)=images.shape
    images_processed=numpy.zeros((s1,(s2-4),(s3-4)))
    count=0
    temp1=numpy.zeros((1,(s2-4)*(s3-4)))
    temp2=numpy.zeros((s2,s3))
    window=3
    for i in range(s1):
        count=0
        temp1=numpy.zeros((1,(s2-4)*(s3-4)))
        for j in range(s2-window-1):
            for k in range(s3-window-1):
                temp2=images[i,j:j+window,k:k+window]
                temp1[0,count]=int(getPattern(temp2))
                count=count+1
        temp1=temp1[0].reshape((s2-4),(s3-4))
        images_processed[i]=temp1
    return images_processed

```

```

def bin2int(x):
    y = 0
    for i,j in enumerate(x):
        y += j<<i
    return y

```

```

def getElementsClockwise(mat):
    (d,r)=mat.shape
    l=0
    u=0
    count=0
    theta=numpy.zeros((1,r*d))
    direction=0
    while l<=(r-1) and u<=(d-1):
        if direction==0:
            for i in range(l,r):
                theta[0,count]=mat[u,i]
                count=count+1
            u=u+1
            direction=direction+1
        elif direction==1:

```



```

    for i in range(u,d):
        theta[0,count]=mat[i,r-1]
        count=count+1
    r=r-1
    direction=direction+1
elif direction==2:
    for i in range(r-1,l-1,-1):
        theta[0,count]=mat[d-1,i]
        count=count+1
    d=d-1
    direction=direction+1
elif direction==3:
    for i in range(d-1,u-1,-1):
        theta[0,count]=mat[i,l]
        count=count+1
    l=l+1
    direction=direction+1
if direction%4 ==0:
    direction=0
return theta

def getPattern_wh(im_portion):
    temp=numpy.zeros((3,3))
    #im_portion1=getElementsClockwise(im_portion)
    for i in range(3):
        for j in range(3):
            if im_portion[i,j]>=im_portion[1,1]:
                temp[i,j]=1
    return temp

def getPattern(im_portion):
    temp=numpy.zeros((1,8))
    im_portion1=getElementsClockwise(im_portion)
    for i in range(8):
        if im_portion1[0,i]<im_portion1[0,8]:
            temp[0,i]=1
    temp_int=temp.tolist()
    temp2_bool=[bool(m) for m in temp_int[0]]
    temp2=bin2int(temp2_bool[:-1])
    return temp2

def getHistograms(images_lbp):
    (s1,s2,s3)=images_lbp.shape
    sd=(int)((s2/8)*(s3/8)*256)
    result=numpy.zeros((s1,sd))

```

```

for i in range(s1):
    result[i]=getVals(images_lbp[i],s2,s3)
return result

def getVals(img_por,s2,s3):
    sd=(int)((s2/8)*(s3/8))
    temp=numpy.zeros((sd,256))
    idx=0
    for j in range(0,s2-8,8):
        for k in range(0,s3-8,8):
            unique,counts=numpy.unique(img_por[j:j+8,k:k+8].reshape(1,8*8,order='F'),return_counts=True)
            unique_int=[int(l) for l in unique.tolist()]
            temp[idx,unique_int]=counts
            idx=idx+1
    return temp.reshape((1,sd*256))

def predict_lbph(input_image,recognizer,labels):
    (s1,s2)=input_image.shape
    (d1,d2)=recognizer.shape
    temp=numpy.zeros((1,s1,s2))
    temp[0]=input_image
    input_histogamed=train_lbph(temp)
    (minval,index,distance)=(10000,0,0)
    for i in range(d1):
        distance=numpy.linalg.norm(recognizer[i,:]-input_histogamed)
        if distance<minval:
            index=i
            minval=distance
    return (labels[index],minval)

def predict_lbph_multi(input_image,recognizer,labels):
    (s3,s1,s2)=input_image.shape
    (d1,d2)=recognizer.shape
    predictions=numpy.zeros((s3))
    minval_confidence = numpy.zeros((s3))
    input_histogamed=train_lbph(input_image)
    for j in range(s3):
        (minval, index, distance) = (10000, 0, 0)
        for i in range(d1):
            distance=numpy.linalg.norm(recognizer[i,:]-input_histogamed[j,:])
            if distance<minval:
                index=i
                minval=distance
        predictions[j]=labels[index]
        minval_confidence[j]=minval

```

```
return (predictions)
```

Screenshots:

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

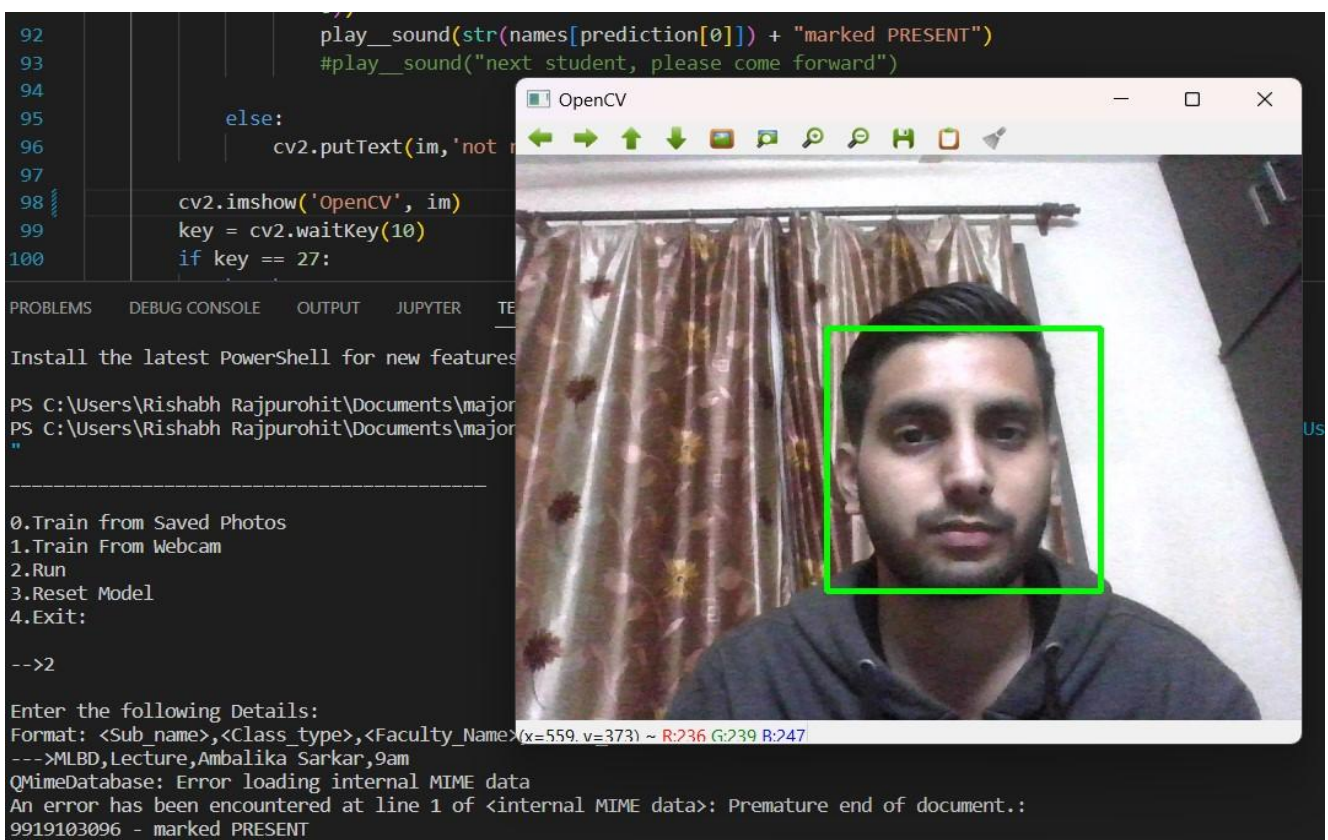
PS C:\Users\Rishabh Rajpurohit\Documents\majorP> conda activate major
PS C:\Users\Rishabh Rajpurohit\Documents\majorP> & "C:/Users/Rishabh Rajpurohit/anaconda3/

-----

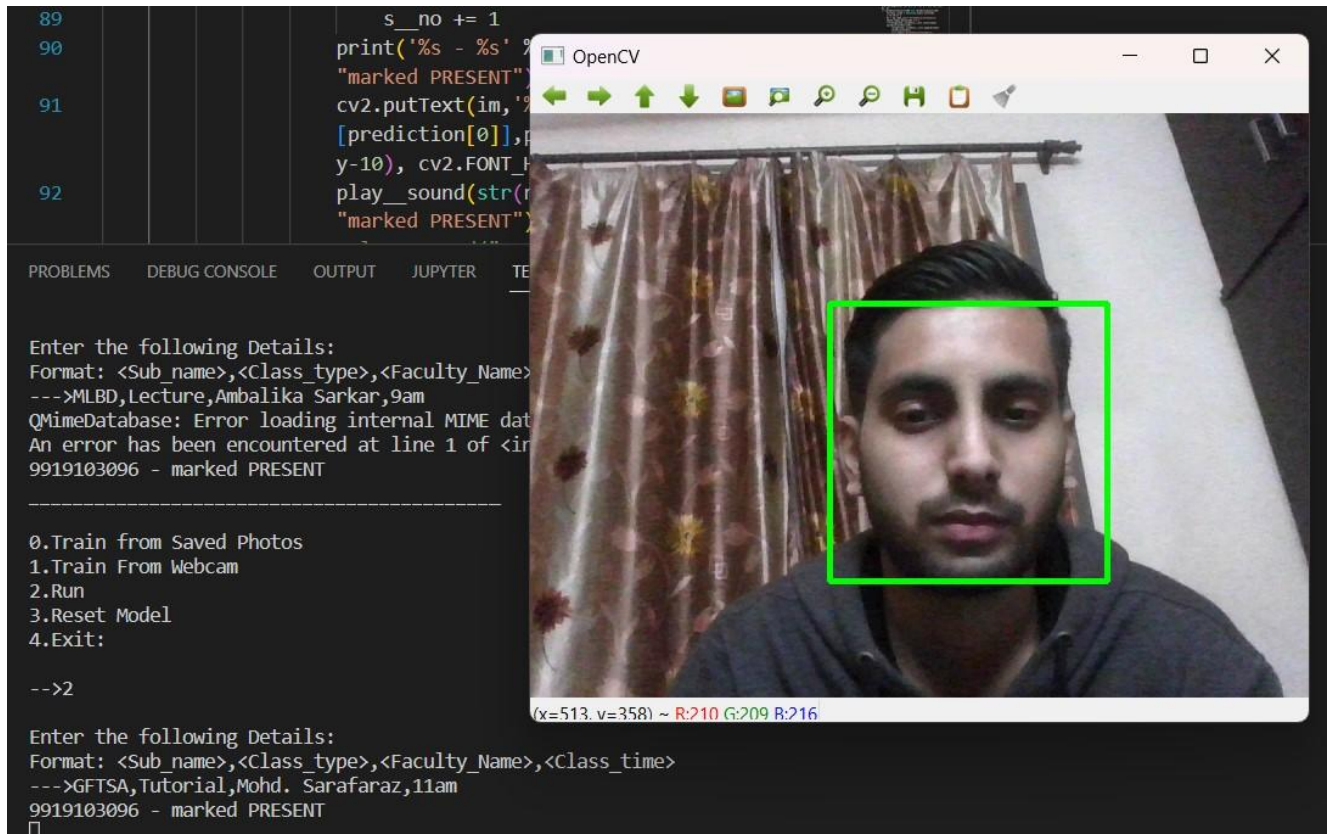
0.Train from Saved Photos
1.Train From Webcam
2.Run
3.Reset Model
4.Exit:

-->
```

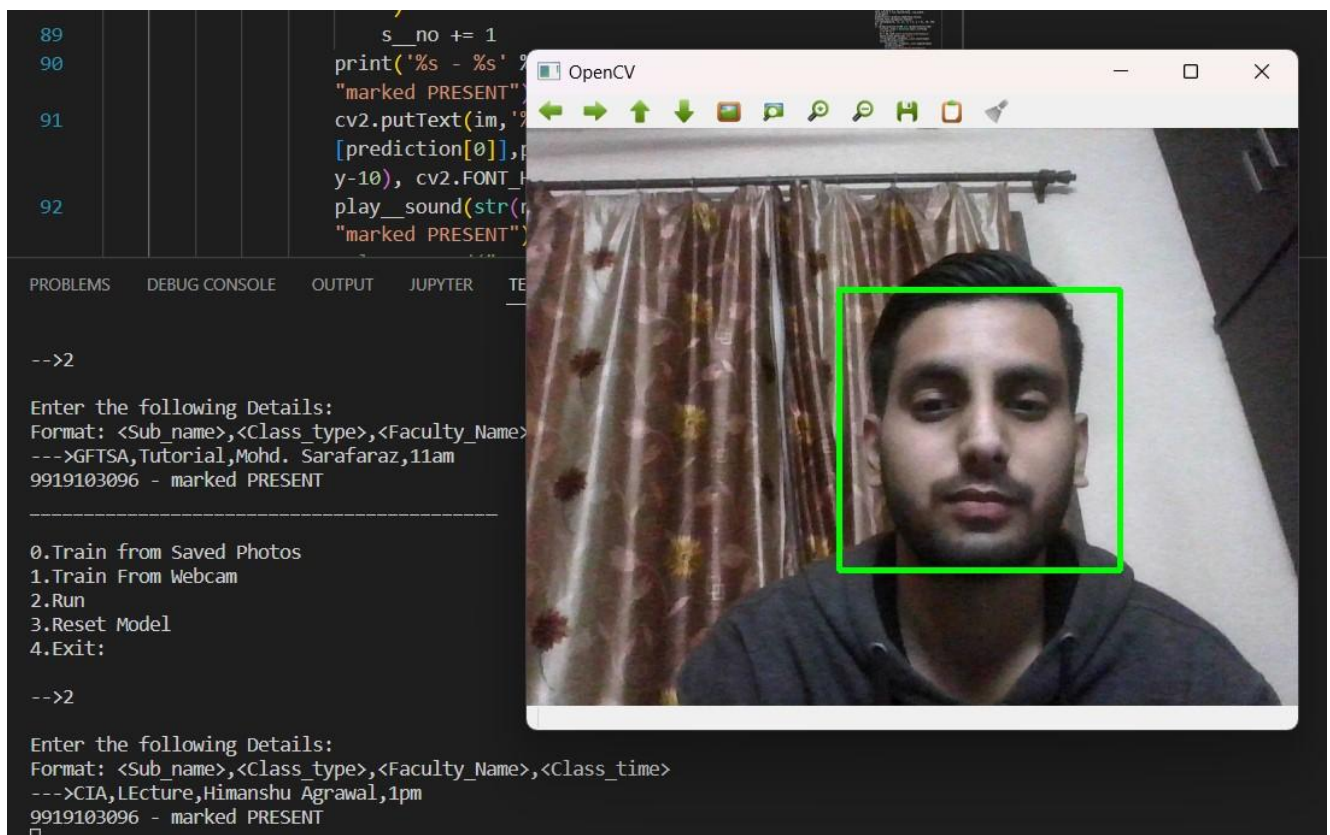
(Different Modes in our system / welcome screen and menu)



(adding details of class 1 - MLBD and marked present because It is already in the database)  
This mode can be safely exited by pressing ctrl+c on the keyboard.

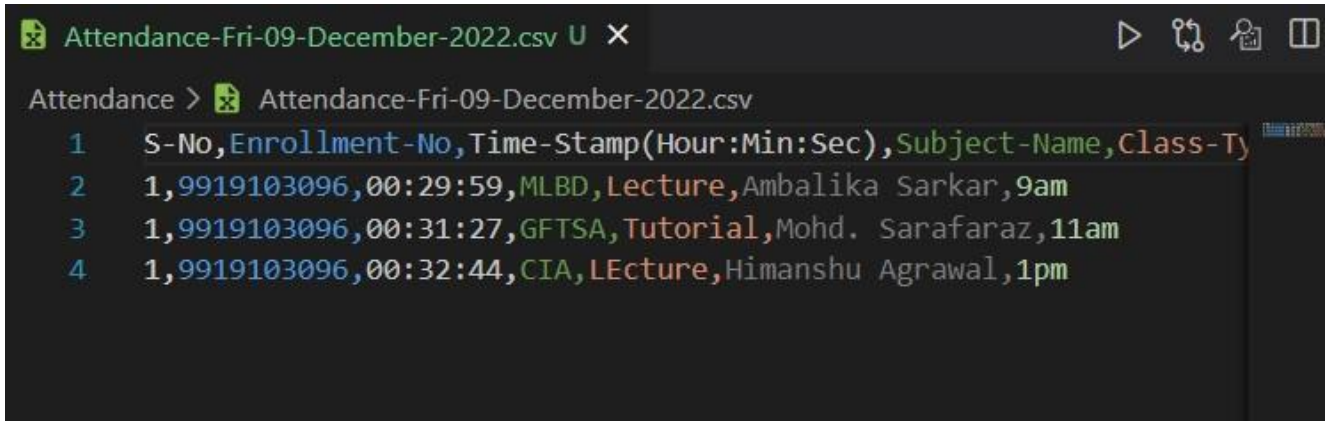


(Now GFSTA class)



(Now CIA class)





```
Attendance > Attendance-Fri-09-December-2022.csv
1 S-No,Enrollment-No,Time-Stamp(Hour:Min:Sec),Subject-Name,Class-Ty
2 1,9919103096,00:29:59,MLBD,Lecture,Ambalika Sarkar,9am
3 1,9919103096,00:31:27,GFTSA,Tutorial,Mohd. Sarafaraz,11am
4 1,9919103096,00:32:44,CIA,Lecture,Himanshu Agrawal,1pm
```

(At last this is how our Attendance Record Looks like)

## Issues

- We had firstly aimed to implement this project on a Raspberry Pi, but due to low memory, hardware, and voltage problems, the library opencv-contrib could not be installed in a feasible amount of time.
- The accuracy of the lbph model is 85%, but in the future we can switch to another algorithm if necessary and this will be very easy to do as our code is quite modular in nature.

## Results

So as we successfully demonstrated that our system can detect and recognize faces present in the database and act accordingly and it can also generate structured Attendance records each day in a separate file. Also Attendance of different classes are well segregated in groups each starting from Serial No. 1. The last entry in each group can be used to deduce the total strength of class. In the future we can integrate data handling and analysis functionality like Generating Reports both weekly and monthly. In short the possibilities are endless.

## Conclusion

Automated Attendance Systems based on face recognition techniques thus proved to be time saving and secured. This system can also be used to identify an unknown person. In real time scenarios PCA outperforms other algorithms with better recognition rate and low false positive rate. The future work is to improve the recognition rate of algorithms when there are unintentional changes in a person like a tonsuring head, using a scarf or a beard. The system developed only recognizes face up to 30 degrees angle variations which has to be improved further. Gait recognition can be fused with ace recognition systems in order to achieve better performance of the system. Poor lighting conditions may affect image quality which indirectly degrades system performance. Our system will perform well in a well lit environment.

## References

- [1] S. Chandran (2012, September). "A Novel Technique to Detect Faces in a Group Photo". *International Journal of Computer Applications* [Online]. vol. 54, issue 1. Available: [https://www.researchgate.net/publication/258652446\\_A\\_Novel\\_Technique\\_to\\_Detect\\_Faces\\_in\\_a\\_Group\\_Photo](https://www.researchgate.net/publication/258652446_A_Novel_Technique_to_Detect_Faces_in_a_Group_Photo).
- [2] K. Shelke, "Face Recognition from Group Photograph", *International Journal of Engineering and Innovative Technology (IJEIT)*, vol. 3, issue 1, pp. 265-269, July 2013. Available: [https://www.ijeit.com/Vol%203/Issue%201/IJEIT1412201307\\_47.pdf](https://www.ijeit.com/Vol%203/Issue%201/IJEIT1412201307_47.pdf)
- [3] O. K. Manyam, N. Kumar, P. Belhumeur, D. Kriegman, "Two faces are better than one: Face recognition in group photographs", *International Joint Conference on Biometrics, IJCB*, 2011
- [4] K. H. Teoh, et al., "Face Recognition and Identification using Deep Learning Approach", *Journal of Physics: Conference Series*, 2020. Available: <https://iopscience.iop.org/article/10.1088/1742-6596/1755/1/012006/pdf>
- [5] S. Singh, S. G. Jasmine, "Face Recognition System", *International Journal of Engineering Research & Technology (IJERT)*, vol. 8, issue 5, pp. 263-263, May 2019. Available: <https://www.ijert.org/research/face-recognition-system-IJERTV8IS050150.pdf>
- [6] T. Ahonen, A. Hadid, M. Pietikainen. "Face Recognition with Local Binary Patterns", in *Conf. European Conference on Computer Vision*, Prague, Czech Republic, May 11-14, 2004, pp.470-482. Available: [https://www.researchgate.net/publication/221304831\\_Face\\_Recognition\\_with\\_Local\\_Binary\\_Patterns](https://www.researchgate.net/publication/221304831_Face_Recognition_with_Local_Binary_Patterns)