CAPSTONE 2019-2020

(Project Semester August-December 2019)

(Face Recognition Attendance System)

Submitted by

DEEPA DEVARAJAN: 11601746

VISHAL VERMA:11615139

PIKESH PATEL: 11606271

ANKIT BHARDWAJ: 11615912

Programme: **B-Tech (CSE)**

Course Code: **CSE439**

Under the Guidance of

(Mr. Puneet Kumar)

Discipline of CSE/IT

LOVELY SCHOOL OF COMPUTER SCIENCE LOVELY PROFESSIONAL UNIVERSITY, PHAGWARA

PAC FORM:

INDIA'S LARGEST UNIVERSITY

LOVELY PROFESSIONAL UNIVERSITY

Transforming Education, Transforming India

TOPIC APPROVAL PERFORMA

School of Computer Science and Engineering (SCSE)

Program: P192-ND::Integrated B.Tech. - M.Tech. (Computer Science & Engineering)

COURSE CODE: CSE439 **REGULAR/BACKLOG:** Regular **GROUP NUMBER:** CSERGC0195

Supervisor Name: Puneet Kumar **UID:** 25190 **Designation:** Assistant Professor

Qualification: ____________________ **Research Experience:** ____________________

SR.NO.	NAME OF STUDENT	REGISTRATION NO	BATCH	SECTION	CONTACT NUMBER
1	Ankit Bhardwaj	11615912	2016	K1607	9015590578
2	Vishal	11615139	2016	K1637	7986659276
3	Pikesh Patel	11606271	2016	K1612	8602400701
4	Deepa Devarajan	11601746	2016	K1637	9137066346

SPECIALIZATION AREA: Programming-I **Supervisor Signature:** ____________________

PROPOSED TOPIC: Face Recognition based Attendance System

Qualitative Assessment of Proposed Topic by PAC		
Sr.No.	**Parameter**	**Rating (out of 10)**
1	Project Novelty: Potential of the project to create new knowledge	7.85
2	Project Feasibility: Project can be timely carried out in-house with low-cost and available resources in the University by the students.	7.15
3	Project Academic Inputs: Project topic is relevant and makes extensive use of academic inputs in UG program and serves as a culminating effort for core study area of the degree program.	7.69
4	Project Supervision: Project supervisor's is technically competent to guide students, resolve any issues, and impart necessary skills.	7.85
5	Social Applicability: Project work intends to solve a practical problem.	7.31
6	Future Scope: Project has potential to become basis of future research work, publication or patent.	7.00

PAC Committee Members		
PAC Member (HOD/Chairperson) Name: Sawal Tandon	UID: 14770	Recommended (Y/N): Yes
PAC Member (Allied) Name: Ravi Kant Sahu	UID: 16920	Recommended (Y/N): Yes
PAC Member 3 Name: Virrat Devaser	UID: 14591	Recommended (Y/N): Yes

Final Topic Approved by PAC: **Face Recognition based Attendance System**

Overall Remarks: Approved

PAC CHAIRPERSON Name: 11024::Amandeep Nagpal **Approval Date:** 03 May 2019

DECLARATION

We hereby declare that the project work entitle Face recognition attendance system Application is an authentic record of our own carried out as requirements of Capstone Project for the award of B. Tech degree in Computer Science and Engineering from Lovely Professional University, Phagwara, under the guidance of Mr. Puneet Kumar during August to December 2019. All the information which we are going to cover in this capstone project report is based on our own intensive work and is genuine.

Project Group Number: KC195

Name of Student 1: Deepa Devarajan

Registration No: 11601746

Name of Student 2: Vishal

Registration No: 11615139

Name of Student 3: Pikesh Patel

Registration No: 11606271

Name of Student 4: Ankit Bhardwaj

Registration No: 11615912

(Signature of Student 1)

Date:

(Signature of Student 2)

Date:

(Signature of Student 3)

Date:

(Signature of Student 4)

Date:

CERTIFICATE

This is to certify that the declaration statement made by this group of students is to the best of my knowledge and belief. They will complete this Capstone Project under my guidance and supervision. The present work is the result of original investigation, effort, and study. No part of the work has ever been submitted for any other degree at any University. The Capstone project is fit for the submission and partial fulfilment of the conditions for the award of B-Tech degree in Computer Science and Engineering from Lovely Professional University, Phagwara.

Signature and Name of the Mentor:

Design Assistant Professor

School of Computer Science and Engineering,

Lovely Professional University,

Phagwara, Punjab.

Date:

ACKNOWLEDGEMENT

We the students of B-Tech CSE, Lovely Professional University, are here to acknowledge that we have completed our Project Report under the guidance of Mr. Puneet Kumar. We are highly indebted to our teacher for his guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in complete the task. We hope that we can develop better application upon the experience and knowledge that we have gained by designing this application and we hope to learn more about the technologies which are used to develop applications.

Deepa Devarajan

Vishal Verma

Pikesh Patel

Ankit Bhardwaj

TABLE OF CONTENT

S.no. Title Page no.

Introduction

Face Recognition Attendance System (FRAS) is the complete system to record attendance without much interference in very less time. It saves time and effort of the teacher in the university where we only get one hours of time for one class. FRAS record attendance with the help of face recognition. It marks the attendance of all students in the class by obtaining data from the database and match with present face data and saves the result in the database (Excel Sheet). This system makes the attendance process easy and minimizes the interference of the teachers. Which provides them more time to teach and take full advantage of their period time.

The main objective of this project is to develop face recognition based automated student attendance system. In order to achieve better performance, the test images and training images of this proposed approach are limited to frontal and upright facial images that consist of a single face only. The test images and training images must be captured by using the same device to ensure no quality difference. In addition, the students must register in the database to be recognized. The enrolment can be done on the spot through the user-friendly interface

Profile of the problem Rationale/scope of the study

It will be waste of time to mark attendance in every period. Taking attendance is one of the important and daily tasks which our university teachers must do. This takes an average of 10% time of our period, sometimes more than that. While taking attendance teachers must perform many tasks if we differentiate them there will be 4 - 5 steps. First, when they come to the class, they need to open their laptops they call each roll numbers and sometimes a student cannot able to listen or missed their chance. After that teachers need to call all the absentees for in case there may be any student present and mistakenly marked absent. In the end, the teacher needs to headcount all the present student, headcount should match the number of present students if it's not there may be any case of proxy. To find that student teacher has to go through all the steps again.

Traditional student attendance marking technique is often facing a lot of trouble. The face recognition student attendance system emphasizes its simplicity by eliminating classical student attendance marking technique such as calling student names or checking respective identification cards. There are not only disturbing the teaching process but also causes distraction for students during exam sessions. Apart from calling names, attendance sheet is passed around the classroom during the lecture sessions. The lecture class especially the class with many students might find it difficult to have the attendance sheet being passed around the class.

Thus, face recognition student attendance system is proposed in order to replace the manual signing of the presence of students which are burdensome and causes students get distracted in order to sign for their attendance. Furthermore, the face recognition based automated student attendance system able to overcome the problem of fraudulent approach and lecturers does not have to count the number of students several times to ensure the presence of the students.

Existing System

3.1 Introduction

The system is being developed for deploying an easy and a secure way of taking down attendance. The software would first capture an image of all the authorized persons and stores the information into database. The system would then store the image by mapping it into a face coordinate structure. Next time whenever the registered person enters the premises the system recognizes the person and marks his attendance.

3.2 Existing System

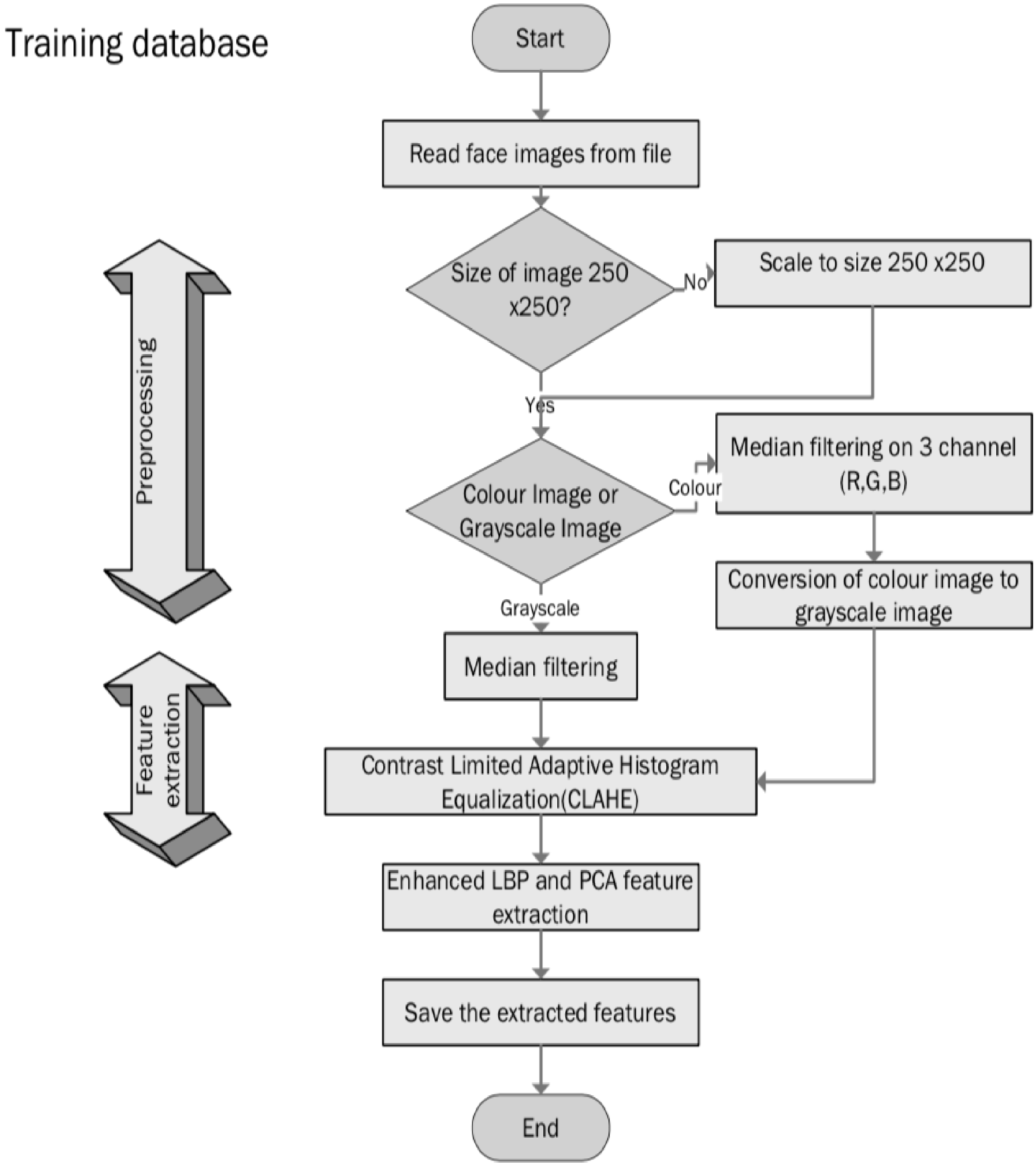
For now, there is not very used existing system that is present for marking attendance online via facial recognition however facial recognition software is widely used for security purpose and person counting in casino and other public places. No doubt later this system can be implemented when there will be growth in reliability of facial detection software's.

Coming to our project which has been made using open cv library, the software identifies 80 nodal points on a human face. In this context, nodal points are endpoints used to measure variables of a person's face, such as the length or width of the nose, the depth of the eye sockets and the shape of the cheekbones. The system works by capturing data for nodal points on a digital image of an individual's face and storing the resulting data as a faceprint. The faceprint is then used as a basis for comparison with data captured from faces in an image or video.

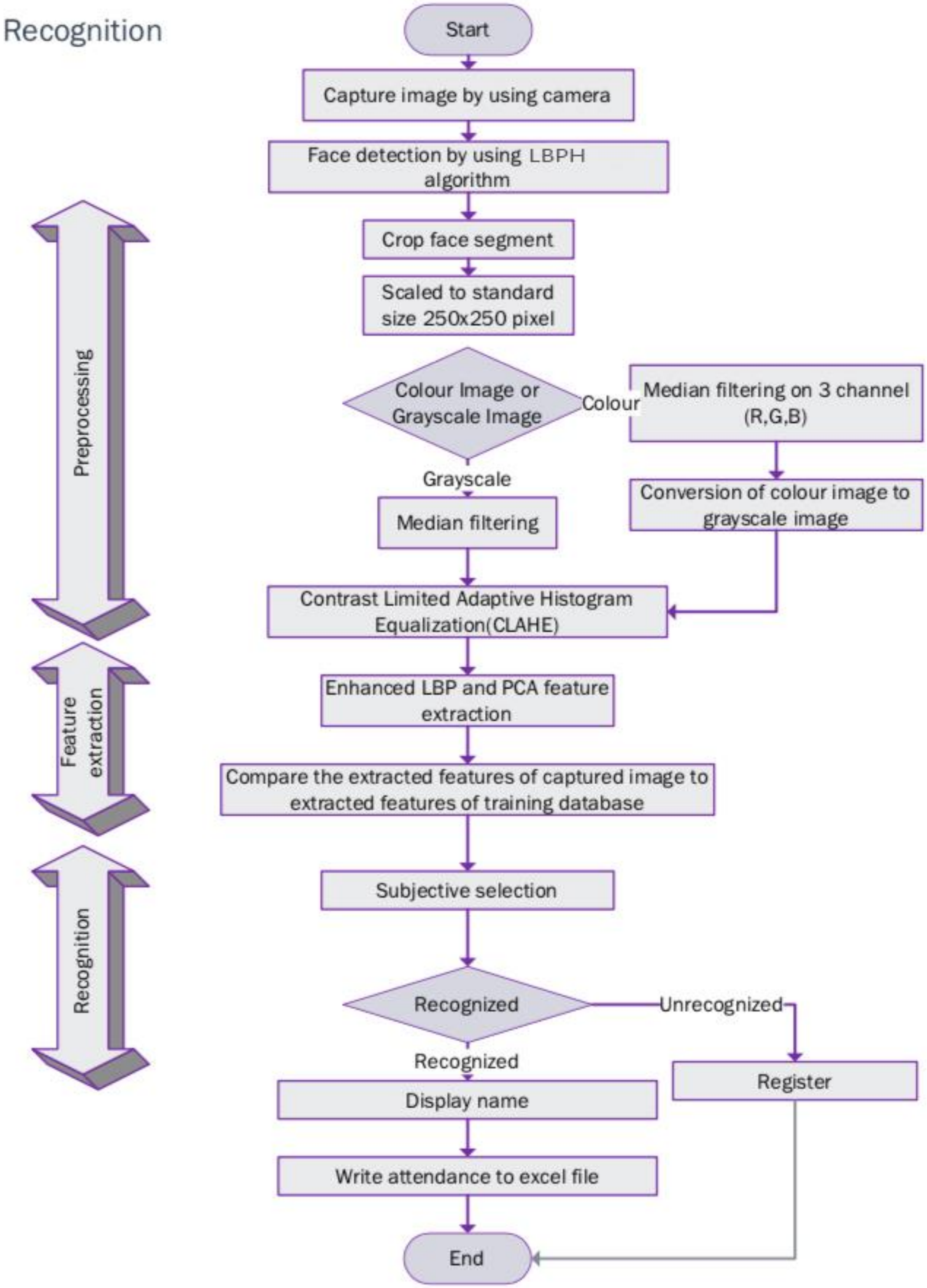
Face recognition consists of two steps, in first step faces are detected in the image and then these detected faces are compared with the database for verification. Several methods have been proposed for face detection.

The efficiency of face recognition algorithm can be increased with the fast face detection algorithm. Our system utilized the detection of faces in the classroom image. Face recognition techniques can be Divided into two types Appearance based which use texture features that is applied to whole face or some specific Regions, other is Feature is using classifiers which uses geometric features like mouth, nose, eyes, eye brows, cheeks and relation between them.

3.3 DFD for current system


Training database
Start
Read face images from file
Size of image 250 x250?
No
Scale to size 250 x250
Yes
Preprocessing
Colour Image or Grayscale Image
Colour
Median filtering on 3 channel (R,G,B)
Conversion of colour image to grayscale image
Grayscale
Median filtering
Feature extraction
Contrast Limited Adaptive Histogram Equalization(CLAHE)
Enhanced LBP and PCA feature extraction
Save the extracted features
End

After Training System DFD:


Recognition
Start
Capture image by using camera
Face detection by using LBPH algorithm
Crop face segment
Scaled to standard size 250x250 pixel
Colour Image or Grayscale Image
Colour
Median filtering on 3 channel (R,G,B)
Conversion of colour image to grayscale image
Grayscale
Median filtering
Contrast Limited Adaptive Histogram Equalization(CLAHE)
Enhanced LBP and PCA feature extraction
Compare the extracted features of captured image to extracted features of training database
Subjective selection
Recognized
Unrecognized
Register
Recognized
Display name
Write attendance to excel file
End
Preprocessing
Feature extraction
Recognition

3.4 What’s new in the system to be developed

The system we will be developing would be successfully able to accomplish the task of marking the attendance in the classroom automatically and output will be obtained in an excel sheet as desired in real-time. However, in order to develop a dedicated system which can be implemented in an educational institution, a very efficient algorithm which is insensitive to the lighting conditions of the classroom must be developed. Also, a camera of the optimum resolution must be utilized in the system.

Another important aspect where we can work towards is creating an online database of the attendance and automatic updating of the attendance into it keeping in mind the growing popularity of Internet of Things and cloud. This can be done by creating a standalone module which can be installed in the classroom having access to internet, preferably a wireless system. These developments can greatly improve the applications of the project.

Problem Analysis

This project involves taking attendance of the student using a biometric face recognition software. The main objectives of this project are:

- Capturing the dataset: Herein, we need to capture the facial images of a student and store it into the database.
- Training the dataset: The dataset needs to be trained by feeding it with algorithms so that it correctly identifies the face.
- Face recognition: Based on the data captured, the model is then tested. If the face is present in the database, it should be correctly identified. If not,
- Marking attendance: Marking the attendance of the right person into the excel sheet. The model must be trained well to increase its accuracy.

4.1 Product definition

The system is being developed for deploying an easy and a secure way of taking down attendance. The software would first capture an image of all the authorized persons and stores the information into database. The system would then store the image by mapping it into a face coordinate structure. Next time whenever the registered person enters the premises the system recognizes the person and marks his attendance.

4.2 Feasibility Analysis

When it comes to feasibility, it is feasible for mid-size organization to a large organization as we are dealing with time saving and manpower, we can say that this is a one time investment kind of service where we install the system and camera and then all we need to do is use the system and keep maintaining and improve the features.

Currently either manual or biometric attendance system are being used in which manual is hectic and time consuming. The biometric serves one at a time, so there is a need of such system which could automatically mark the attendance of many persons at the same time.

This system is cost efficient, no extra hardware required just a daily laptop, mobile or tablet, etc. Hence it is easily deployable.

There could be some price of cloud services when the project is deployed on cloud.

The work of administration department to enter the attendance is reduced and stationary cost so every institute or organization will opt for such time and money saving system.

Not only in institutes or organizations, it can also be used at any public places or entry-exit gates for advance surveillance.

4.3 Project Plan:

ACTIVITY	**TIME PERIOD**
Requirement Gathering	10 Days
Planning	10 Days
Design	10 Days
Implementation	90 Days
Testing	10 Days

Software Requirement Analysis

5.1 Introduction

The main purpose for preparing this software is to give a general insight into the analysis and requirement of the existing system or situation and for determining the operating characteristics of the system.

5.2 General Description

This project requires a computer system with the following software's:

- Operating System - Windows 7 or later (latest is best)
- Python 3.7 (including all necessary libraries)
- Microsoft Excel 2007 or later
- Google chrome (for cloud related services)

5.3 Specific Requirement

This face recognition attendance system project requires OpenCV-a python library with built-in LBPH face recognizer for training the dataset captured via the camera.

Coming to the technical feasibility following are the requirements:
Hardware and Software Requirements

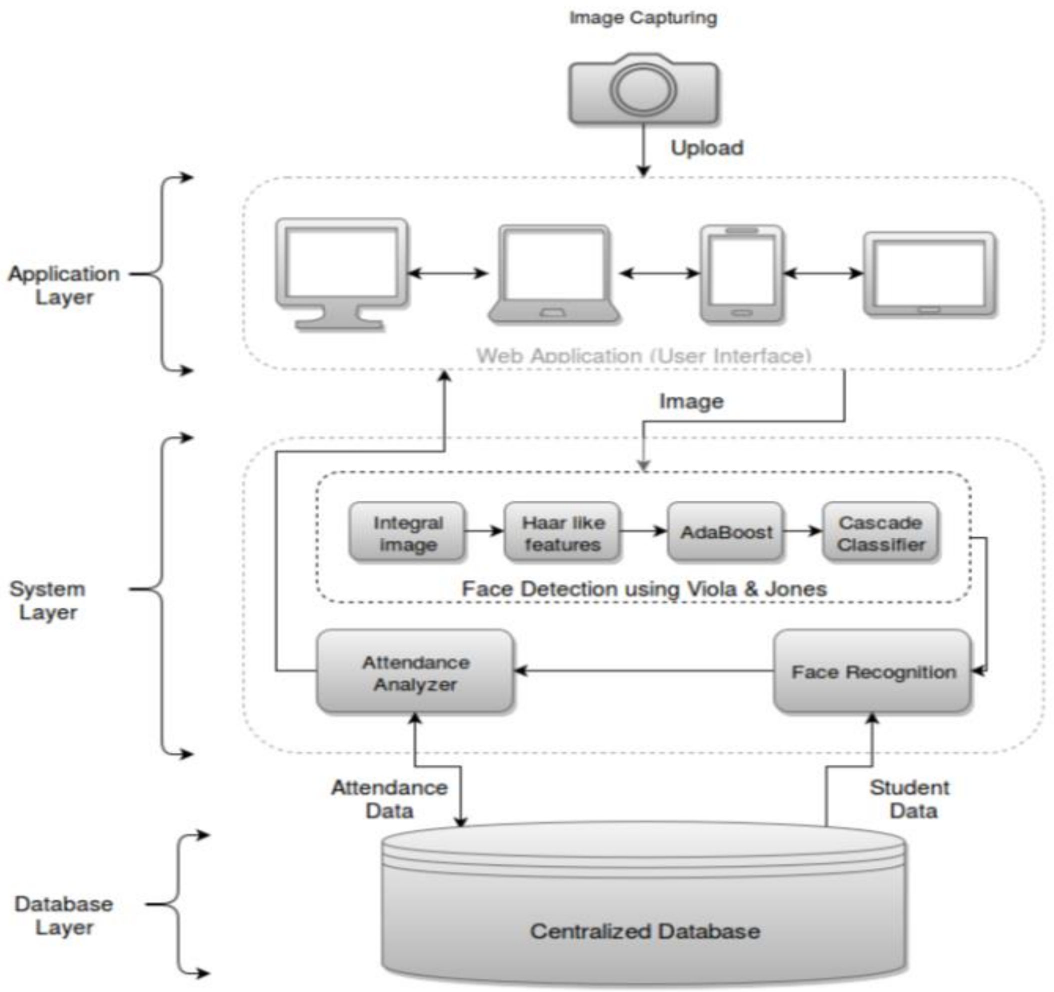
Processor	:	Intel Processor IV (latest is recommended)
Ram	:	4 GB (Higher would be good)
Hard disk	:	40 GB
Monitor	:	RBG led
Keyboard	:	Basic 108 key keyboard
Mouse	:	Optical Mouse (or touchpad would work)
Camera	:	1.5 Mega pixel (Or more)

Design

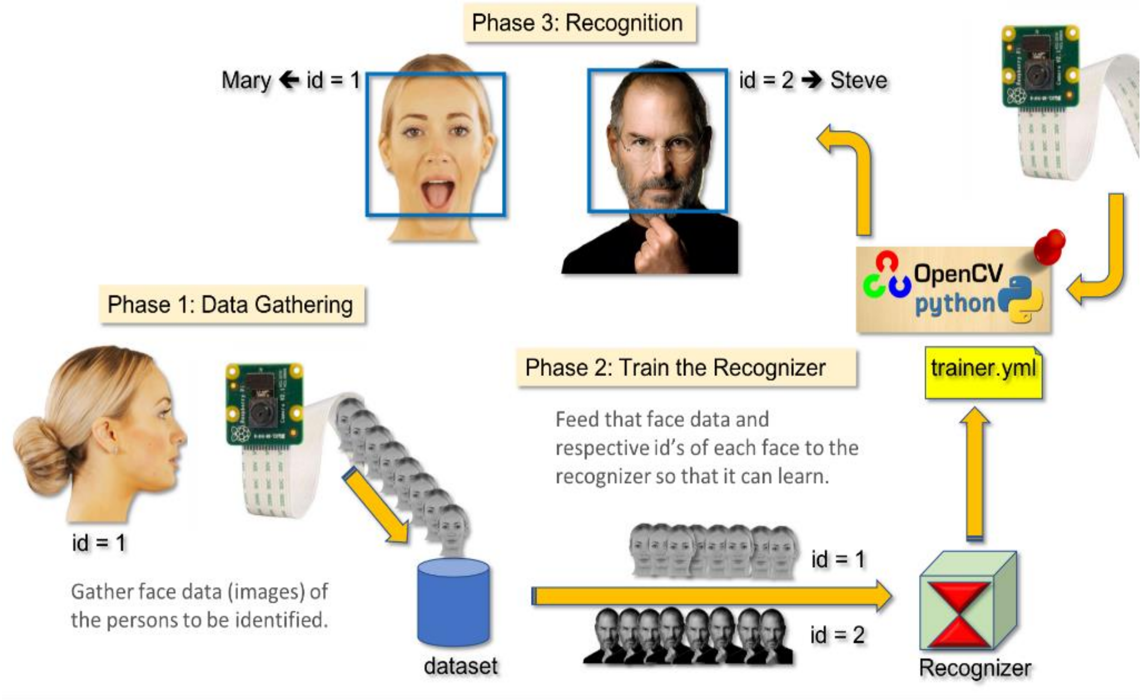
6.1 System design

In the first step image is captured from the camera. There are illumination effects in the captured image because of different lighting conditions and some noise which is to be removed before going to the next steps. Histogram normalization is used for contrast enhancement in the spatial domain. Median filter is used for removal of noise in the image. There are other techniques like FFT and low pass filter for noise removal and smoothing of the images, but median filter gives good results.

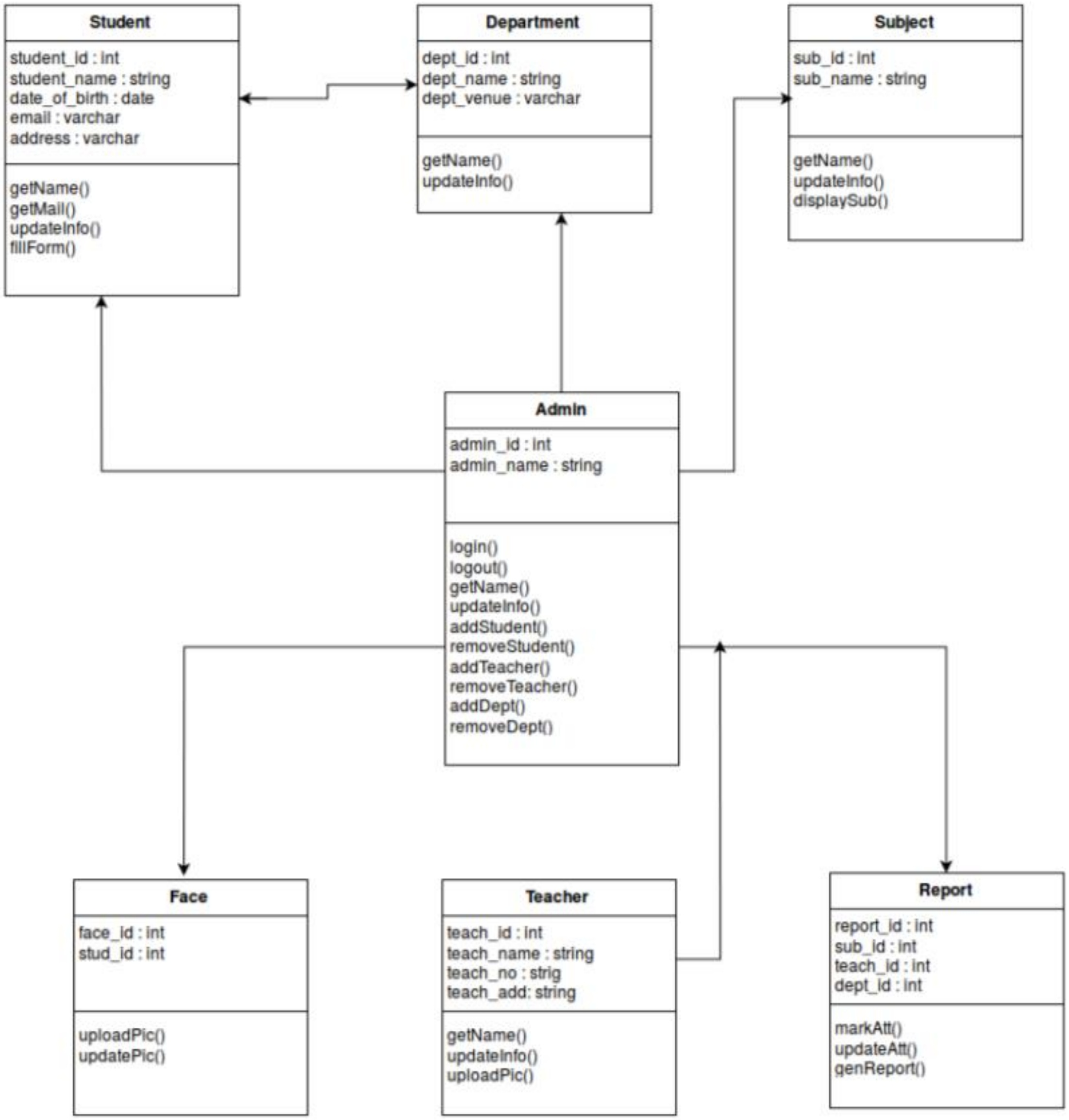
Database Design:



6.2 Design Notation


Phase 3: Recognition
Mary ← id = 1
id = 2 → Steve
OpenCV
python
Phase 1: Data Gathering
Phase 2: Train the Recognizer
trainer.yml
Feed that face data and respective id's of each face to the recognizer so that it can learn.
id = 1
Gather face data (images) of the persons to be identified.
dataset
id = 1
id = 2
Recognizer

6.3 Detailed Design



Testing

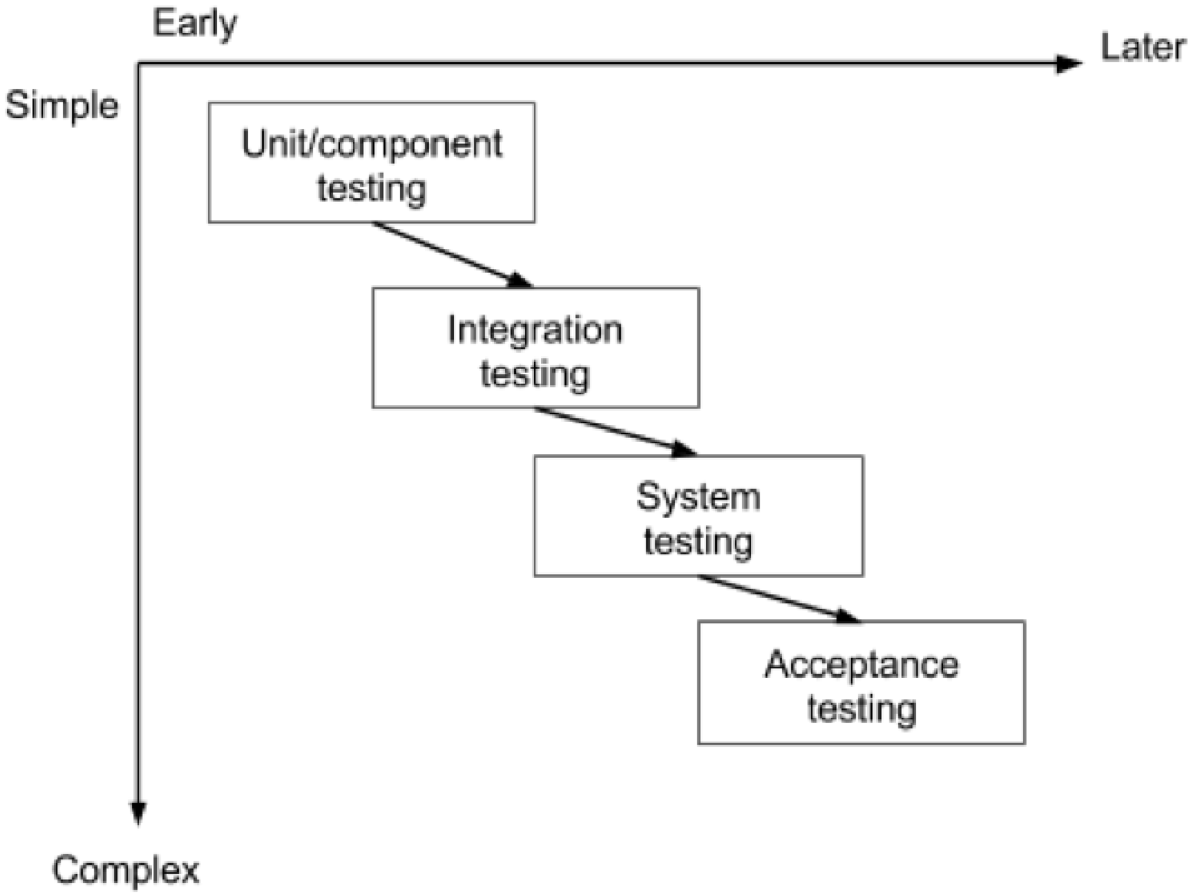
7.1 Functional Testing

Functional testing is done at every function level for example there is function named as assure_path_exists which is responsible to create directory for dataset, that function is tested if the directory is created or not. Similarly, all the functions are tested separately before integrating.

7.2 **Structural Testing**

Structural testing is performed after we integrated each module. After integrating the path creation function and capture image function then we tested that after capturing image the photos were saved in the correct directory that was created by assure_path_exists.

7.3 Levels of testing



Unit testing has been performed on the project by verifying each module independently, isolating it from the others. Each module is fulfilling the requirements as well as desired functionality. Integration testing would be performed to check if all the functionalities are working after integration.

7.4 Testing the project

Testing early and testing frequently is well worth the effort. By adopting an attitude of constant alertness and scrutiny in all your projects, as well as a systematic approach to testing, the tester can pinpoint any faults in the system sooner, which translates in less time and money wasted later.

Implementation

8.1 Implementation of the project

The complete project is implemented using python 3.7 (or later), main library used was OpenCV, SQLite3, pywin32, Pillow, ms excel is used for database purpose to store the attendance.

8.1.1 Capturing the dataset

The first and foremost module of the project is capturing the dataset. When building an on-site face recognition system and when you must have physical access to a specific individual to assemble model pictures of their face, you must make a custom dataset. Such a framework would be necessary in organizations where individuals need to physically appear and attend regularly.

To retrieve facial images and make a dataset, we may accompany them to an exceptional room where a camcorder is arranged to (1) distinguish the (x, y)- directions of their face in a video stream and (2) store the frames containing their face to database. We may even play out this process over a course of days or weeks in order to accumulate instances of their face in:

- Distinctive lighting conditions
- Times of day

- Mind-sets and passionate states to make an increasingly differing set of pictures illustrative of that specific individual's face.

This Python script will:

1. Access the camera of system

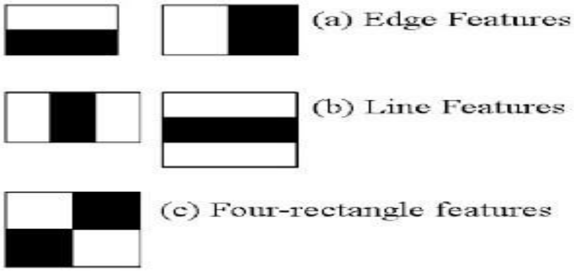
2. Detect faces

3. Write the frame containing the face to database

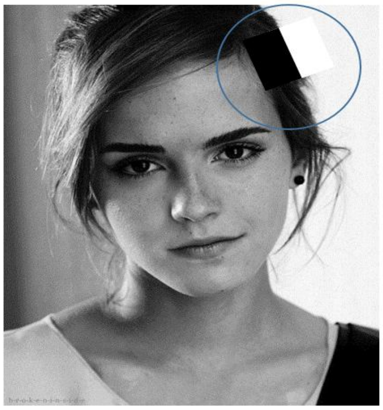
Face detection using OpenCV:

In this project we have used OpenCV, to be precise, the Haar-cascade classifier for face detection. Haar Cascade is an AI object detection algorithm proposed by Paul Viola and Michael Jones in their paper "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001. It is an AI based methodology where a cascade function is prepared from a great deal of positive and negative pictures (where positive pictures are those where the item to be distinguished is available, negative is those where it isn't). It is then used to tell objects in different pictures. Fortunately, OpenCV offers pre-trained Haar cascade algorithms, sorted out into classifications (faces, eyes, etc.), based on the pictures they have been prepared on.

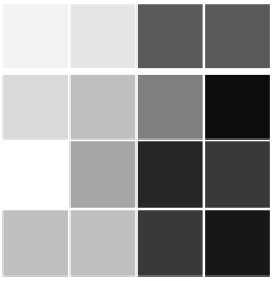
Presently let us perceive how this algorithm solidly functions. The possibility of Haar cascade is separating features from pictures utilizing a sort of 'filter', like the idea of the convolutional kernel. These filters are called Haar features and resemble that:



The idea is passing these filters on the picture, investigating one part at the time. At that point, for every window, all the pixel powers of, separately, white and dark parts are added. Ultimately, the value acquired by subtracting those two summations is the value of the feature extracted. In a perfect world, an extraordinary value of a feature implies it is pertinent. On the off chance that we consider the Edge (a) and apply it to the accompanying B&W pic:



We will get a noteworthy value, subsequently the algorithm will render an edge highlight with high likelihood. Obviously, the genuine intensities of pixels are never equivalent to white or dark, and we will frequently confront a comparable circumstance:



By and by, the thought continues as before: the higher the outcome (that is, the distinction among highly contrasting black and white summations), the higher the likelihood of that window of being a pertinent element.

To give you a thought, even a 24x24 window results in more than 160000 highlights, and windows inside a picture are a ton. How to make this procedure progressively effective? The arrangement turned out with the idea of Summed-area table, otherwise called Integral Image. It is an information structure and algorithm for producing the sum of values in a rectangular subset of a matrix. The objective is diminishing the amount of calculations expected to get the summations of pixel intensities in a window.

Following stage additionally includes proficiency and enhancement. Other than being various, features may likewise be unessential. Among the features we get (that are more than 160000), how might we choose which ones are great? The response to this inquiry depends on the idea of Ensemble strategy: by consolidating numerous algorithms, weak by definition, we can make a solid algorithm. This is cultivated utilizing Ad boost which both chooses the best features and prepares the classifiers that utilize them. This algorithm builds a strong classifier as a simple mix of weighted basic weak classifiers.

We are nearly done. The last idea which should be presented is a last component of advancement. Even though we decreased our 160000+ highlights to an increasingly reasonable number, the latter is still high: applying every one of the features on every one of the windows will consume a great deal of time. That is the reason we utilize the idea of Cascade of classifiers: rather than applying every one of the features on a window, it bunches the features into various phases of classifiers and applies individually. On the off chance that a window comes up short (deciphered: the distinction among white and dark summations is low) the primary stage (which typically incorporates few features), the algorithm disposes it: it won't think about outstanding features on it. If it passes, the algorithm applies the second phase of features and continues with the procedure.

Storing the data:

In order to store the captured dataset, we use SQLite. SQLite is a software library that implements a self-contained, serverless, zero-configuration, transactional SQL database engine. SQLite is the most widely deployed SQL database engine in the world.

SQLite is renowned for its extraordinary element zero-design, which implies no intricate arrangement or organization is required. In SQLite, sqlite3 command is utilized to make another SQLite database. You don't have to have any unique benefit to make a database. Consider a situation when you have different databases accessible and you need to utilize any of them at once. SQLite ATTACH DATABASE command is utilized to choose a specific database, and after this command, all SQLite statements will be executed under the appended database.

SQLite CREATE TABLE statement is utilized to make another table in any of the given database. Making a primary table includes naming the table and characterizing its columns and every column's information type.

SQLite INSERT INTO Statement is utilized to include new tuples of information into a table in the database.

SQLite-Python

SQLite3 can be incorporated with Python utilizing sqlite3 module, which was written by Gerhard Haring. It furnishes an SQL interface agreeable with the DB-API 2.0 determination depicted by PEP 249. You don't have to introduce this module independently because it is delivered as a matter of course alongside Python version 2.5.x onwards.

To utilize sqlite3 module, you should initially make an association object that speaks to the database and afterward alternatively you can make a cursor object, which will help you in executing all the SQL explanations.

The face recognition systems can operate basically in two modes:

- Verification or authentication of a facial image: it basically compares the input facial image with the facial image related to the user which is requiring the authentication. It is basically a 1x1 comparison.
- Identification or facial recognition: it basically compares the input facial image with all facial images from a dataset with the aim to find the user that matches that face. It is basically a 1xN comparison.

There are different types of face recognition algorithms, for example:

- Eigenfaces (1991)
- Local Binary Patterns Histograms (LBPH) (1996)
- Fisher faces (1997)
- Scale Invariant Feature Transform (SIFT) (1999)
- Speed Up Robust Features (SURF) (2006)

This project uses the LBPH algorithm.

8.1.2 Training Database (LBPH algorithm):

As it is one of the easier face recognitions algorithms, I think everyone can understand it without major difficulties.

Introduction: Local Binary Pattern (LBP) is a simple yet very efficient texture operator which labels the pixels of an image by thresholding the neighbourhood 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) descriptor, 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.

DETAILED EXPLANATION OF WORKING OF LBPH

1. **Parameters**: the LBPH uses 4 parameters:

- **Radius**: the radius is used to build the circular local binary pattern and represents the radius around the central pixel. It is usually set to 1.

- **Neighbors**: the number of sample points to build the circular local binary pattern. Keep in mind: the more sample points you include, the higher the computational cost. It is usually set to 8.

- **Grid X**: the number of cells in the horizontal direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. It is usually set to 8.

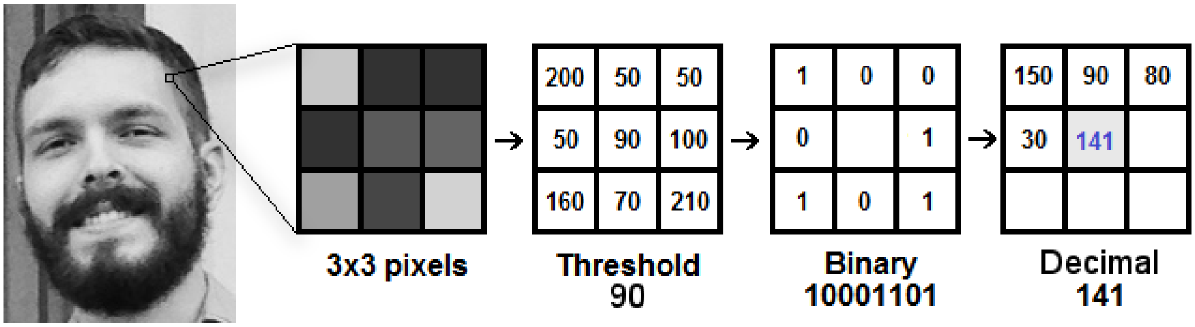
- **Grid Y**: the number of cells in the vertical direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. It is usually set to 8.

2. **Training the Algorithm**: First, we need to train the algorithm. To do so, we need to use a dataset with the facial images of the people we want to recognize. We need to also set an ID (it may be a number or the name of the person) for each image, so the algorithm will use this information to recognize an input image and give you an output. Images of the same person must have the same ID. With the training set already constructed, let's see the LBPH computational steps.

3. **Applying the LBP operation**: The first computational step of the LBPH is to create an intermediate image that describes the original image in a better way, by highlighting the

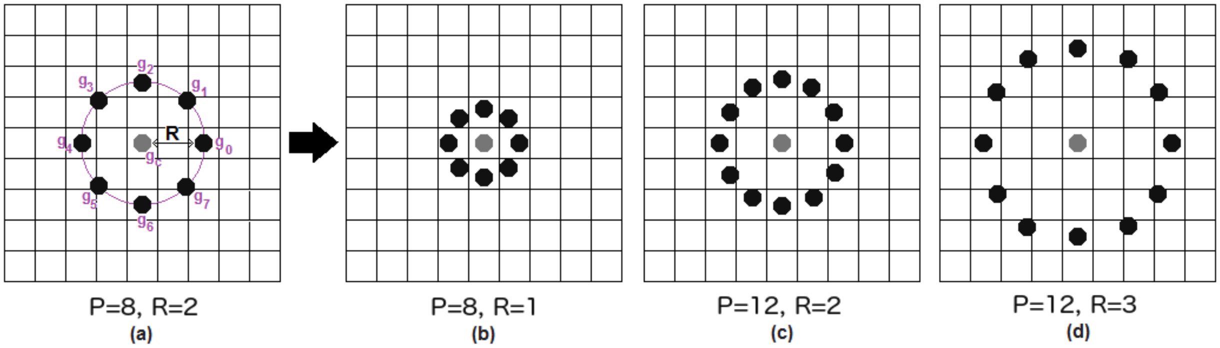
facial characteristics. To do so, the algorithm uses a concept of a sliding window, based on the parameter's **radius** and **neighbors**.

The image below shows this procedure:



Based on the image above, let's break it into several small steps so we can understand it easily:

- Suppose we have a facial image in grayscale.
- We can get part of this image as a window of 3x3 pixels.
- It can also be represented as a 3x3 matrix containing the intensity of each pixel (0~255).
- Then, we need to take the central value of the matrix to be used as the threshold.
- This value will be used to define the new values from the 8 neighbors.
- For each neighbor of the central value (threshold), we set a new binary value. We set 1 for values equal or higher than the threshold and 0 for values lower than the threshold.
- Now, the matrix will contain only binary values (ignoring the central value). We need to concatenate each binary value from each position from the matrix line by line into a new binary value (e.g. 10001101). Note: some authors use other approaches to concatenate the binary values (e.g. clockwise direction), but the result will be the same.
- Then, we convert this binary value to a decimal value and set it to the central value of the matrix, which is a pixel from the original image.
- At the end of this procedure (LBP procedure), we have a new image which represents better the characteristics of the original image.
- **Note**: The LBP procedure was expanded to use a different number of radius and neighbors, it is called Circular LBP.



P=8, R=2
(a)

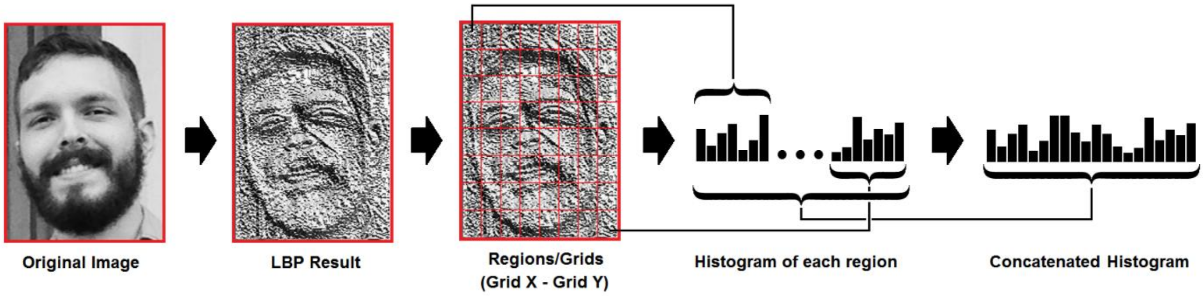
P=8, R=1
(b)

P=12, R=2
(c)

P=12, R=3
(d)

It can be done by using **bilinear interpolation**. If some data point is between the pixels, it uses the values from the 4 nearest pixels (2x2) to estimate the value of the new data point.

4. Extracting the Histograms: Now, using the image generated in the last step, we can use the **Grid X** and **Grid Y** parameters to divide the image into multiple grids, as can be seen in the following image:



Based on the image above, we can extract the histogram of each region as follows:

- As we have an image in grayscale, each histogram (from each grid) will contain only 256 positions (0~255) representing the occurrences of each pixel intensity.

- Then, we need to concatenate each histogram to create a new and bigger histogram. Supposing we have 8x8 grids, we will have 8x8x256=16.384 positions in the final histogram. The final histogram represents the characteristics of the image original image.

After detecting the face, image need to crop as it has only face nothing else focused. To do so Python Imaging Library (PIL) or also known as Pillow is used. PIL is a free library for python programming language that adds support for opening, manipulating, and saving many different image file formats.

Capabilities of pillow:

- per-pixel manipulations,
- masking and transparency handling,
- image filtering, such as blurring, contouring, smoothing, or edge finding,
- image enhancing, such as sharpening, adjusting brightness, contrast or color,
- adding text to images and much more.

8.1.3 Face recognition

In this step, the algorithm is already trained. Each histogram created is used to represent each image from the training dataset. So, given an input image, we perform the steps again for this new image and creates a histogram which represents the image.

- So, to find the image that matches the input image we just need to compare two histograms and return the image with the closest histogram.

- We can use various approaches to compare the histograms (calculate the distance between two histograms), for example: **Euclidean distance**, **chi-square**, **absolute value**, etc. In this example, we can use the Euclidean distance (which is quite known) based on the following formula:

$$D = \sqrt{\sum_{i=1}^{n}(hist1_i - hist2_i)^2}$$

- So, the algorithm output is the ID from the image with the closest histogram. The algorithm should also return the calculated distance, which can be used as a '**confidence**' measurement.

- We can then use a threshold and the ‘confidence’ to automatically estimate if the algorithm has correctly recognized the image. We can assume that the algorithm has successfully recognized if the confidence is lower than the threshold defined.

Conclusions

- LBPH is one of the easiest face recognition algorithms.
- It can represent local features in the images.
- It is possible to get great results (mainly in a controlled environment).
- It is robust against monotonic grey scale transformations.

It is provided by the OpenCV library (Open Source Computer Vision Library).

8.1.4 Marking the attendance

The face(s) that have been recognized are marked as present into the database. Then the entire attendance data is written into an excel sheet that is dynamically created using pywin32 library of python. First, an instance is created for excel application and then new excel workbook is created and active worksheet of that file is fetched. Then data is fetched from database and written in the excel sheet.

8.2 Post implementation and software maintained

After the implementation of the software post implementation can be done by adding more features in the software like SMS tracking of the attendance of particular student for example if he is absent then a automatic scheduled SMS can be sent to their parent.

About the software maintenance part only the features can be maintained and improved over time and database could be large over time so a better data structure can be used for faster fetching of data could be done, for that purpose cloud storage can be used to minimize the latency of fetching data of a student.

Project Legacy

9.1 Current status of the project

Current status of the project contains all the separate modules of capture, training, recognizer and result. The first module that is capture data set is created to fetch the details of students, write them in database, capture the photo of the student and name the files accordingly to train the photos to recognize for the attendance later. After capturing the dataset, the next module comes that is train the dataset that train the photos captured using LBPH algorithm. Next module marks the attendance and write attendance in database as well as excel file. Last module opens the excel file to see the attendance.

9.2 Remaining Area of concern

Most of the remaining area of concern are the technical hurdles that comes when taking images of group of students, for that we can do is use better machines that can handle recognizing multiple people at a time, also the camera and its orientation and most important lightening conditions are important for that HDR cameras can be used which can handle backlight and produce better results.

The ambient light or artificial illumination affects the accuracy of face recognition systems. This is because the basic idea of capturing an image depends on the reflection of the light off an object and the higher the amount of illumination, the higher the recognition accuracy.

Another difficulty which prevents face recognition (FR) systems from achieving good recognition accuracy is the camera angle (pose). The closer the image pose is to the front view, the more accurate the recognition.

For face recognition, changes in facial expression are also problematic because they alter details such as the distance between the eyebrows and the iris in case of anger or surprise or change the size of the mouth in the case of happiness.

Simple non-permanent cosmetic alterations are also a common problem in obtaining good recognition accuracy. One example of such cosmetic alterations is make-up which tends to slightly modify the features. These alterations can interfere with contouring techniques used to perceive facial shape and alter the perceived nose size and shape or mouth size. Other color enhancements and eye alterations can convey misinformation such as the location of eyebrows, eye contrast and cancelling the dark area under the eyes leading to potential change in the appearance of a person.

Although wearing eyeglasses is necessary for many human vision problems and the certain healthy people also wear them for cosmetic reasons but wearing glasses hides the eyes which contain the greatest amount of distinctive information. It also changes the holistic facial appearance of a person.

9.3 Technical and Management lessons learnt

Technical lesson that we learnt is code should be in module so that our team can work better on modules and it can be integrated easily and if there is any error we don't have to go through all the code and another lesson that learnt is that report should be written while we code the modules that gives precise report over the code.

While testing the code sometimes we make changes to the original code and later it becomes a mess when we want to add or remove that tested part. For that we created a separate test.py file for testing all kinds changes that we did in the original source code.

Coming to Management lessons most of the problems we faced is while creating the report file and integration of code, while integration many errors came that we are not aware of so what we can do is integrate the completed modules at the time when they are completed , so that they create less errors while integrating with other modules.

User Manual: A complete (help guide) of the software developed

To run the software first we need to capture the image which we do using open cv video capture function that open camera for few seconds and capture only facial part of the image using frontal face haarcascades classifier, A haarcascades is classifier which is used to detect particular objects form source that is image or video, in our case it was frontal face detection so the cascade file contains the information about the facial symmetry of human face.

After capturing the face, the images need to train, for that there is a separate code that used for training the images what the trainer does is it set the image to the person who's the image is. For example, after training the images are named as user.UID.count.

And the last step is recognizing or mark attendance for that open the recognizer python file and it will open a camera window that will capture the person in the frame and the good thing about this algorithm is it can recognize more than 1 person in the frame depending on the lighting and image condition. After that the recognized person are marked present in the excel sheet as well as the database.

How to use the software

Step 1: Installation

Install the setup in the system. Allow the setup to be installed. A wizard opens. Choose the default download location and click **Next**. Continue the installation by clicking **Next** and at the end click **Finish** to close the wizard. After installation, a folder would appear in that folder one text file is present named dependencies. Open that text file and install 'python' software or one can download on its own from internet, any 3.x version would work. While installing python, choose the option of manual installation and then choose the option to set up the python in system environment. After installation is complete, follow the indexing and install the rest of dependencies. To install the dependencies open "command prompt". To open command prompt press "Window + S", Window Search will be open then type "cmd" in search box and choose option "run as administrator" from right menu shown up after searching. After opening "command prompt", copy the commands from dependencies.txt file and paste in 'cmd'. Make sure system is connected to internet.

Step 2: Run the application(.exe)

After successful installation, a folder is created and inside the folder, run the application file(.exe). A UI opens.

Step 3: Capturing the dataset

In order to enroll a student into the database, click the capture dataset option that appears on the UI. The system runs a program which captures the facial images. Make sure the lighting is fair and your face is in front of the camera so that the system can capture and train the images properly.

Step 4: Training the dataset

After capturing the images, they need to be trained by applying face recognition algorithms like LBPH. This is performed by the train dataset option.

Step 5: Mark the attendance

Click the mark attendance option. The camera opens. Position your face in front of the camera and make sure the lighting is good.

Step 6: Attendance Sheet

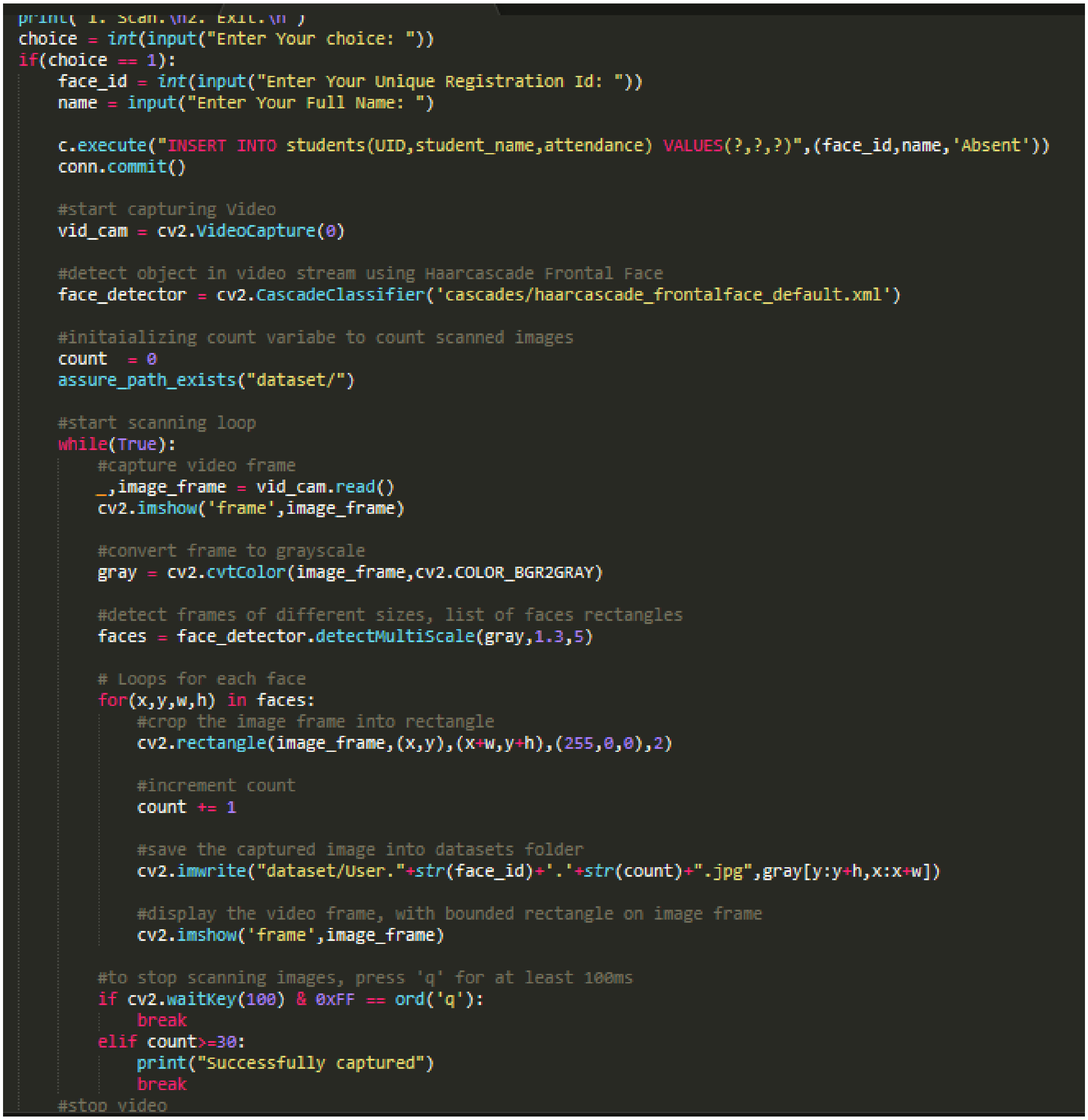
To check if your attendance has been marked, click the attendance sheet option, an excel sheet opens with three fields-name, ID and attendance. The faces which were recognized by the system in the previous step are marked present, with the rest marked as absent.

Step 7: Exit

After the completion of the process of marking the attendance, click **Exit** to close the application.

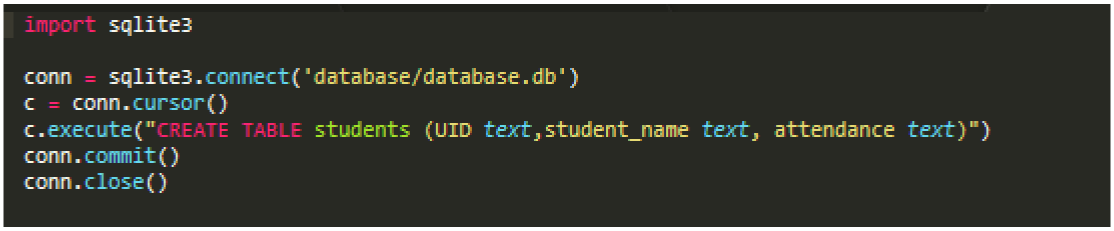
Source code or snapshot:

capturing Dataset:



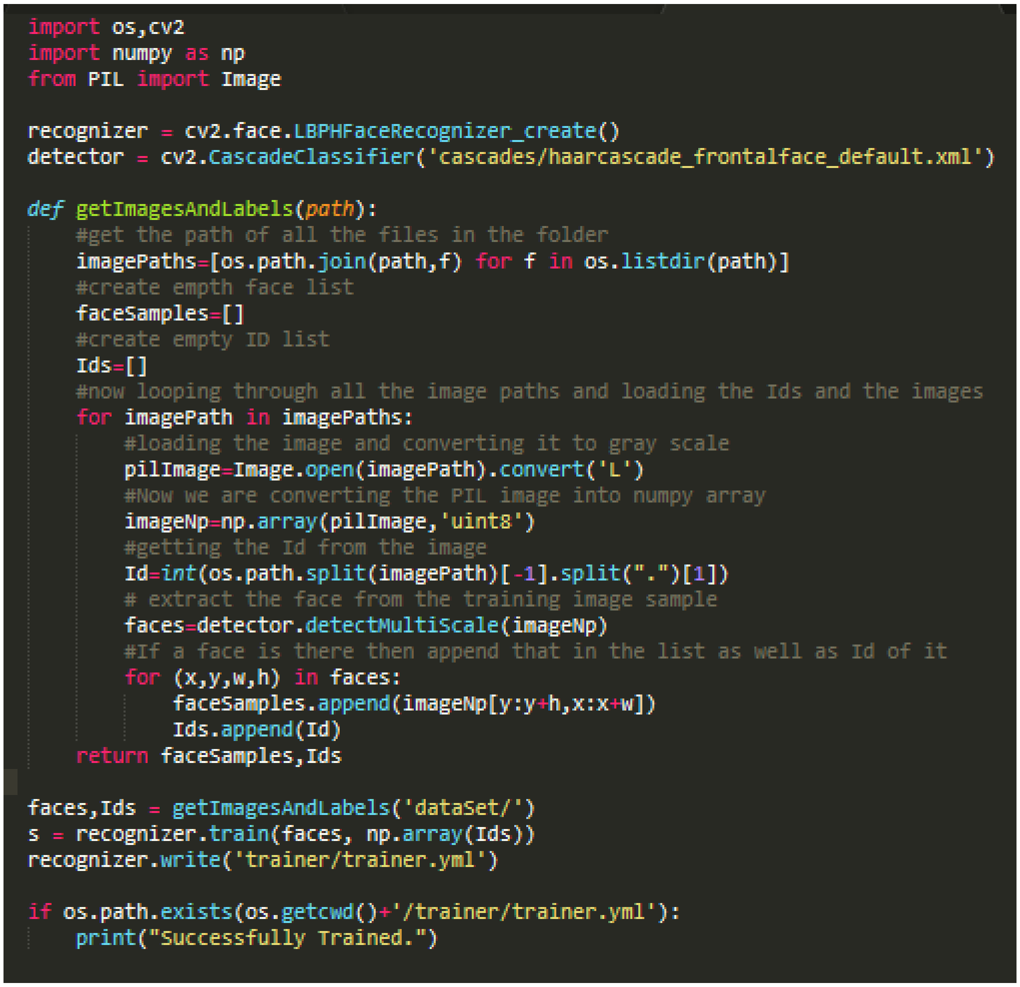
```
print('1. Scan.\n2. Exit.\n')
choice = int(input("Enter Your choice: "))
if(choice == 1):
    face_id = int(input("Enter Your Unique Registration Id: "))
    name = input("Enter Your Full Name: ")

    c.execute("INSERT INTO students(UID,student_name,attendance) VALUES(?,?,?)",(face_id,name,'Absent'))
    conn.commit()

    #start capturing Video
    vid_cam = cv2.VideoCapture(0)

    #detect object in video stream using Haarcascade Frontal Face
    face_detector = cv2.CascadeClassifier('cascades/haarcascade_frontalface_default.xml')

    #initaializing count variabe to count scanned images
    count  = 0
    assure_path_exists("dataset/")

    #start scanning loop
    while(True):
        #capture video frame
        _,image_frame = vid_cam.read()
        cv2.imshow('frame',image_frame)

        #convert frame to grayscale
        gray = cv2.cvtColor(image_frame,cv2.COLOR_BGR2GRAY)

        #detect frames of different sizes, list of faces rectangles
        faces = face_detector.detectMultiScale(gray,1.3,5)

        # Loops for each face
        for(x,y,w,h) in faces:
            #crop the image frame into rectangle
            cv2.rectangle(image_frame,(x,y),(x+w,y+h),(255,0,0),2)

            #increment count
            count += 1

            #save the captured image into datasets folder
            cv2.imwrite("dataset/User."+str(face_id)+'.'+str(count)+".jpg",gray[y:y+h,x:x+w])

            #display the video frame, with bounded rectangle on image frame
            cv2.imshow('frame',image_frame)

        #to stop scanning images, press 'q' for at least 100ms
        if cv2.waitKey(100) & 0xFF == ord('q'):
            break
        elif count>=30:
            print("Successfully captured")
            break
    #stop video
```

1. Creating Database:



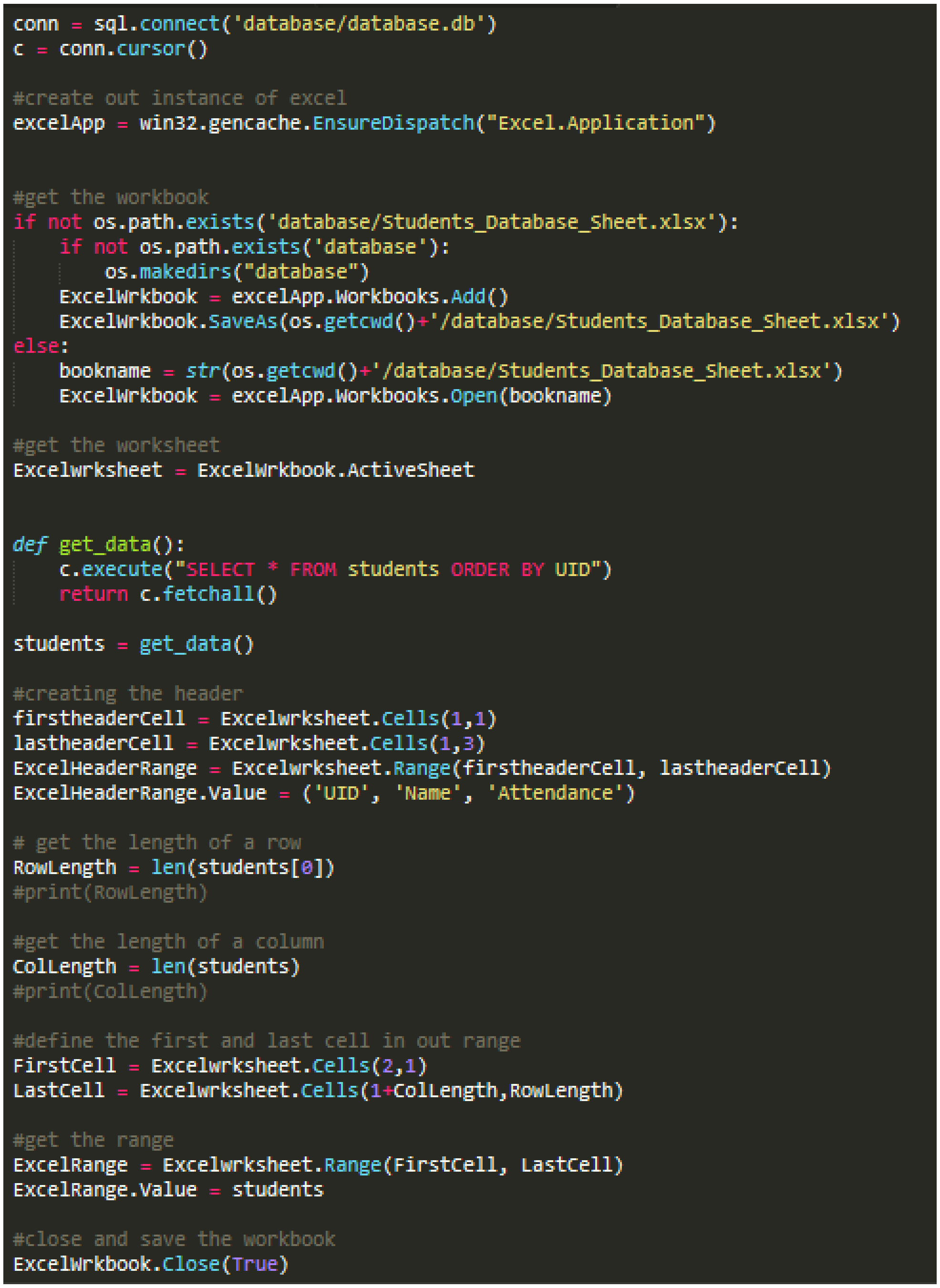
```
import sqlite3

conn = sqlite3.connect('database/database.db')
c = conn.cursor()
c.execute("CREATE TABLE students (UID text,student_name text, attendance text)")
conn.commit()
conn.close()
```

2. Training the Dataset:



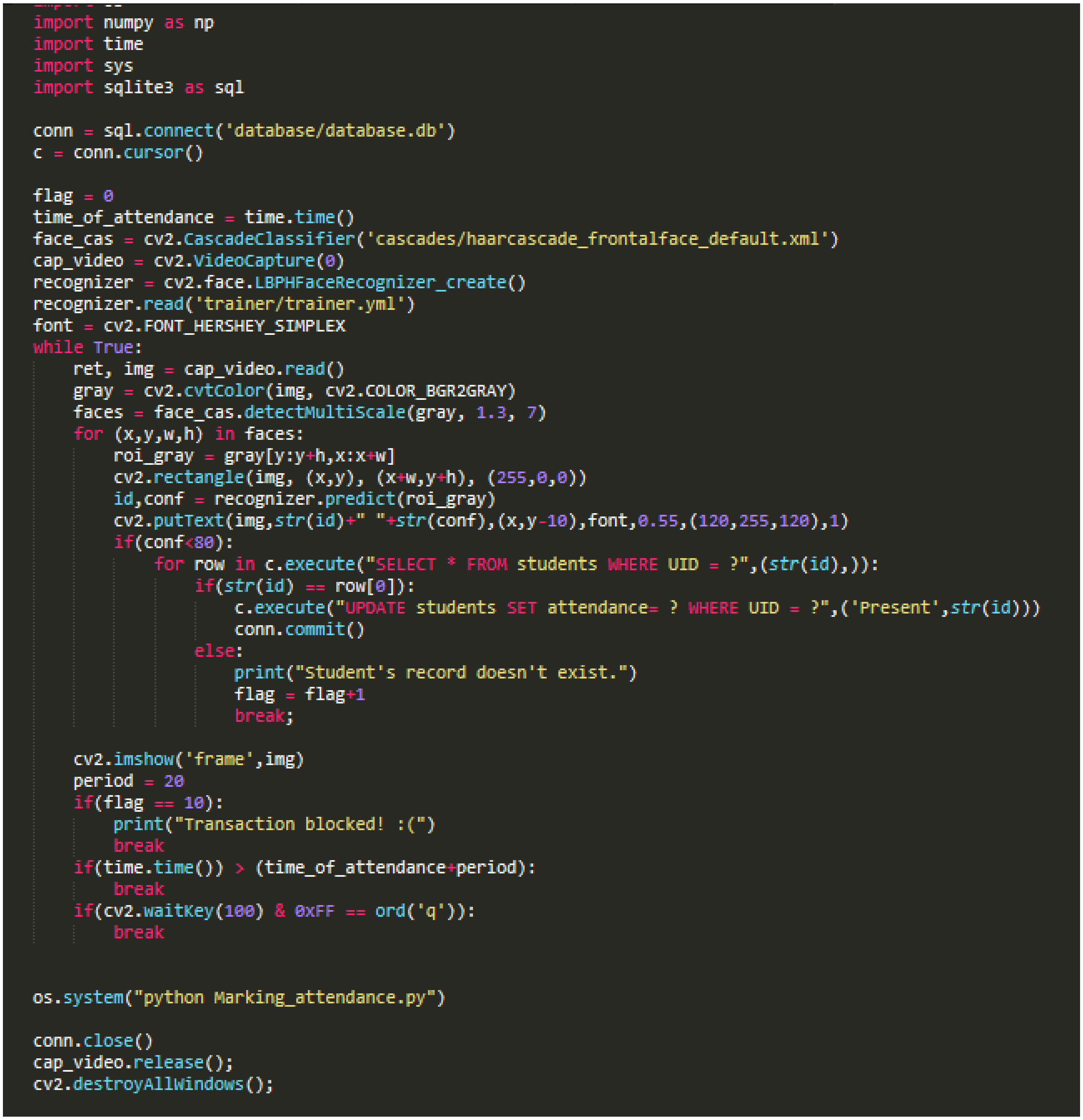
```
import os,cv2
import numpy as np
from PIL import Image

recognizer = cv2.face.LBPHFaceRecognizer_create()
detector = cv2.CascadeClassifier('cascades/haarcascade_frontalface_default.xml')

def getImagesAndLabels(path):
    #get the path of all the files in the folder
    imagePaths=[os.path.join(path,f) for f in os.listdir(path)]
    #create empth face list
    faceSamples=[]
    #create empty ID list
    Ids=[]
    #now looping through all the image paths and loading the Ids and the images
    for imagePath in imagePaths:
        #loading the image and converting it to gray scale
        pilImage=Image.open(imagePath).convert('L')
        #Now we are converting the PIL image into numpy array
        imageNp=np.array(pilImage,'uint8')
        #getting the Id from the image
        Id=int(os.path.split(imagePath)[-1].split(".")[1])
        # extract the face from the training image sample
        faces=detector.detectMultiScale(imageNp)
        #If a face is there then append that in the list as well as Id of it
        for (x,y,w,h) in faces:
            faceSamples.append(imageNp[y:y+h,x:x+w])
            Ids.append(Id)
    return faceSamples,Ids

faces,Ids = getImagesAndLabels('dataSet/')
s = recognizer.train(faces, np.array(Ids))
recognizer.write('trainer/trainer.yml')

if os.path.exists(os.getcwd()+'/trainer/trainer.yml'):
    print("Successfully Trained.")
```

3. Writing database in excel:



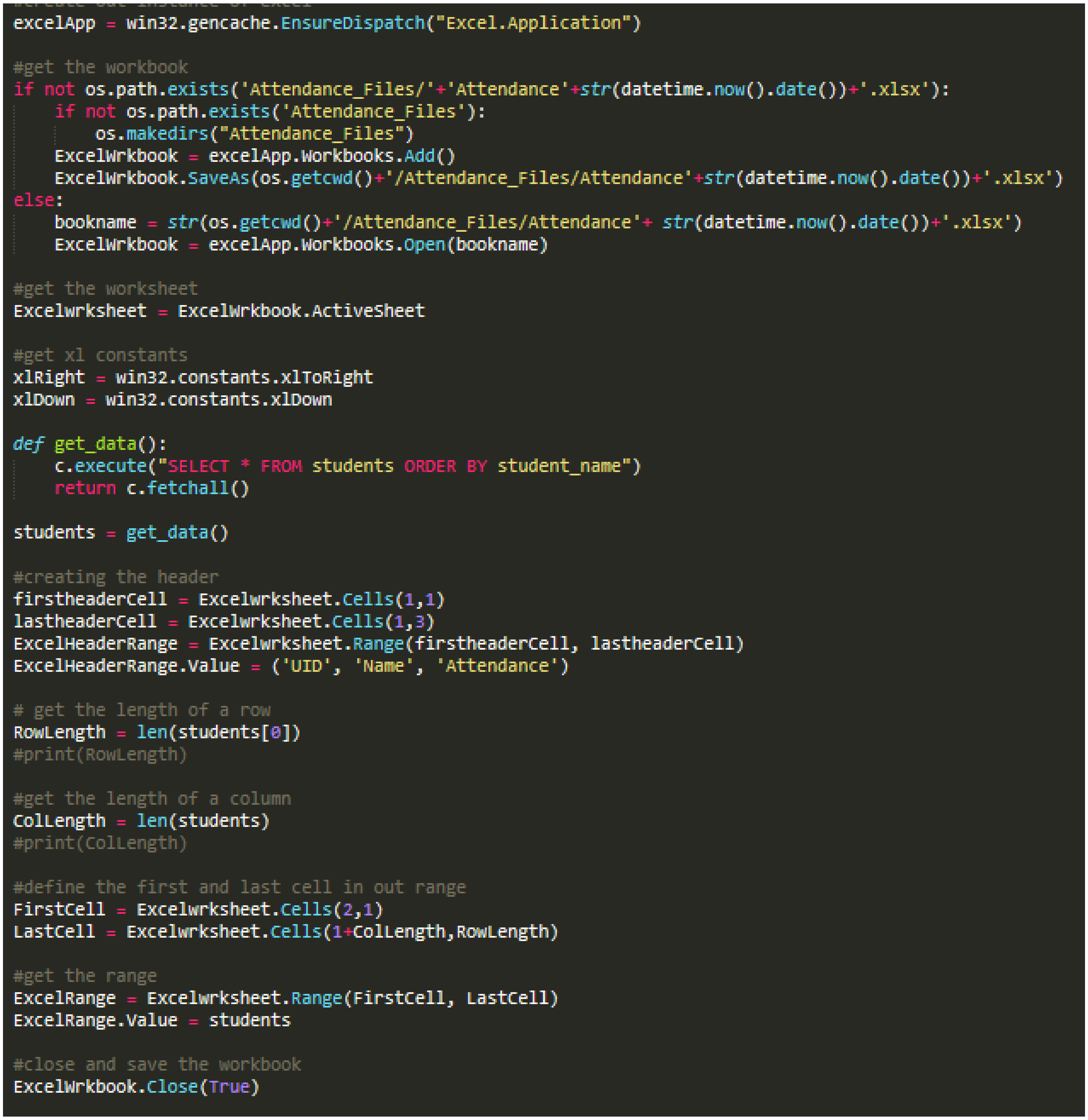
```python
conn = sql.connect('database/database.db')
c = conn.cursor()

#create out instance of excel
excelApp = win32.gencache.EnsureDispatch("Excel.Application")


#get the workbook
if not os.path.exists('database/Students_Database_Sheet.xlsx'):
    if not os.path.exists('database'):
        os.makedirs("database")
    ExcelWrkbook = excelApp.Workbooks.Add()
    ExcelWrkbook.SaveAs(os.getcwd()+'/database/Students_Database_Sheet.xlsx')
else:
    bookname = str(os.getcwd()+'/database/Students_Database_Sheet.xlsx')
    ExcelWrkbook = excelApp.Workbooks.Open(bookname)

#get the worksheet
Excelwrksheet = ExcelWrkbook.ActiveSheet


def get_data():
    c.execute("SELECT * FROM students ORDER BY UID")
    return c.fetchall()

students = get_data()

#creating the header
firstheaderCell = Excelwrksheet.Cells(1,1)
lastheaderCell = Excelwrksheet.Cells(1,3)
ExcelHeaderRange = Excelwrksheet.Range(firstheaderCell, lastheaderCell)
ExcelHeaderRange.Value = ('UID', 'Name', 'Attendance')

# get the length of a row
RowLength = len(students[0])
#print(RowLength)

#get the length of a column
ColLength = len(students)
#print(ColLength)

#define the first and last cell in out range
FirstCell = Excelwrksheet.Cells(2,1)
LastCell = Excelwrksheet.Cells(1+ColLength,RowLength)

#get the range
ExcelRange = Excelwrksheet.Range(FirstCell, LastCell)
ExcelRange.Value = students

#close and save the workbook
ExcelWrkbook.Close(True)
```

4. Recognizer:



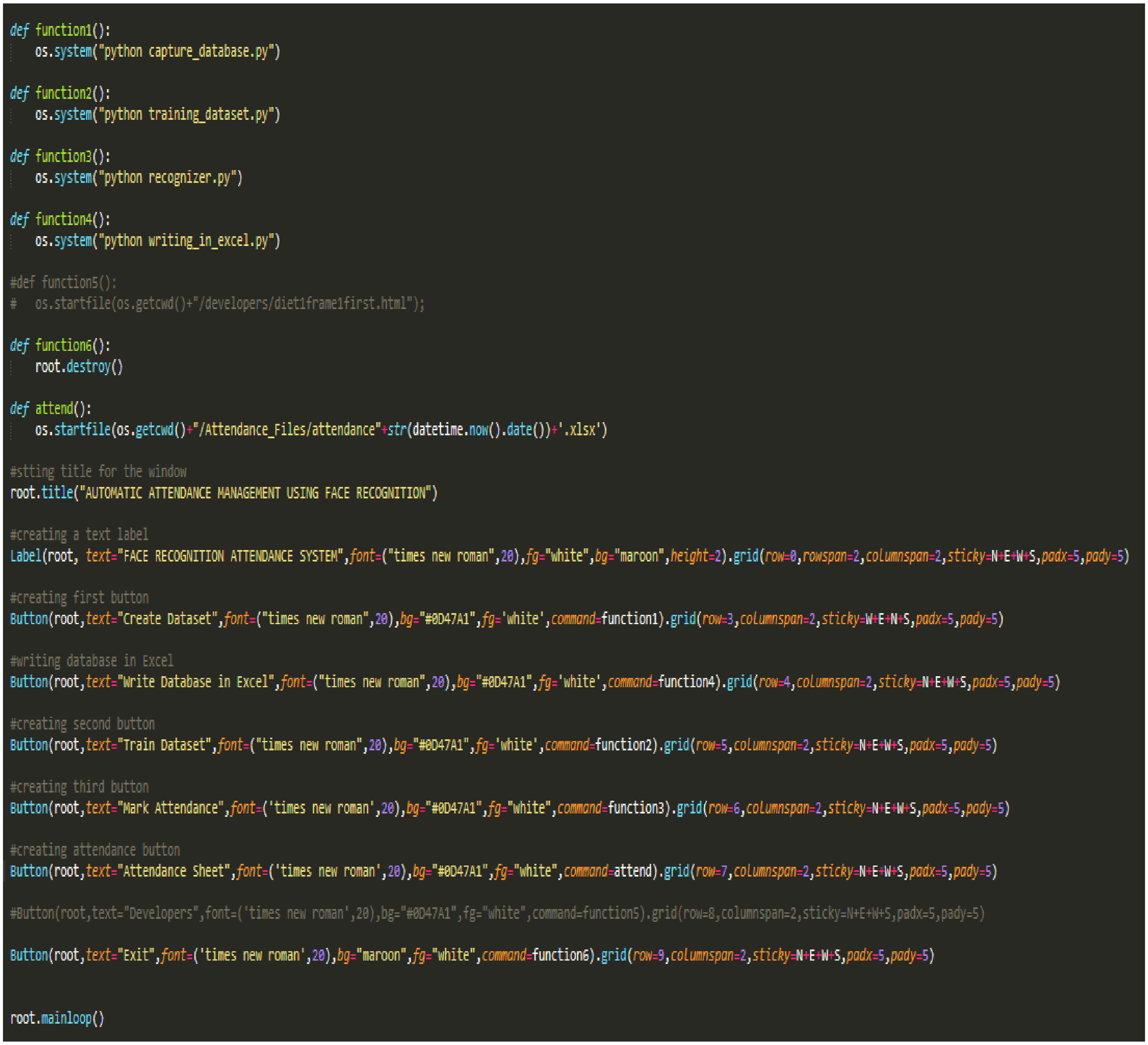
```
import os
import numpy as np
import time
import sys
import sqlite3 as sql

conn = sql.connect('database/database.db')
c = conn.cursor()

flag = 0
time_of_attendance = time.time()
face_cas = cv2.CascadeClassifier('cascades/haarcascade_frontalface_default.xml')
cap_video = cv2.VideoCapture(0)
recognizer = cv2.face.LBPHFaceRecognizer_create()
recognizer.read('trainer/trainer.yml')
font = cv2.FONT_HERSHEY_SIMPLEX
while True:
    ret, img = cap_video.read()
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    faces = face_cas.detectMultiScale(gray, 1.3, 7)
    for (x,y,w,h) in faces:
        roi_gray = gray[y:y+h,x:x+w]
        cv2.rectangle(img, (x,y), (x+w,y+h), (255,0,0))
        id,conf = recognizer.predict(roi_gray)
        cv2.putText(img,str(id)+" "+str(conf),(x,y-10),font,0.55,(120,255,120),1)
        if(conf<80):
            for row in c.execute("SELECT * FROM students WHERE UID = ?",(str(id),)):
                if(str(id) == row[0]):
                    c.execute("UPDATE students SET attendance= ? WHERE UID = ?",('Present',str(id)))
                    conn.commit()
                else:
                    print("Student's record doesn't exist.")
                    flag = flag+1
                    break;

    cv2.imshow('frame',img)
    period = 20
    if(flag == 10):
        print("Transaction blocked! :(")
        break
    if(time.time()) > (time_of_attendance+period):
        break
    if(cv2.waitKey(100) & 0xFF == ord('q')):
        break


os.system("python Marking_attendance.py")

conn.close()
cap_video.release();
cv2.destroyAllWindows();
```

5. Marking Attendance:



```
#create our instance of excel
excelApp = win32.gencache.EnsureDispatch("Excel.Application")

#get the workbook
if not os.path.exists('Attendance_Files/'+'Attendance'+str(datetime.now().date())+'.xlsx'):
    if not os.path.exists('Attendance_Files'):
        os.makedirs("Attendance_Files")
    ExcelWrkbook = excelApp.Workbooks.Add()
    ExcelWrkbook.SaveAs(os.getcwd()+'/Attendance_Files/Attendance'+str(datetime.now().date())+'.xlsx')
else:
    bookname = str(os.getcwd()+'/Attendance_Files/Attendance'+ str(datetime.now().date())+'.xlsx')
    ExcelWrkbook = excelApp.Workbooks.Open(bookname)

#get the worksheet
Excelwrksheet = ExcelWrkbook.ActiveSheet

#get xl constants
xlRight = win32.constants.xlToRight
xlDown = win32.constants.xlDown

def get_data():
    c.execute("SELECT * FROM students ORDER BY student_name")
    return c.fetchall()

students = get_data()

#creating the header
firstheaderCell = Excelwrksheet.Cells(1,1)
lastheaderCell = Excelwrksheet.Cells(1,3)
ExcelHeaderRange = Excelwrksheet.Range(firstheaderCell, lastheaderCell)
ExcelHeaderRange.Value = ('UID', 'Name', 'Attendance')

# get the length of a row
RowLength = len(students[0])
#print(RowLength)

#get the length of a column
ColLength = len(students)
#print(ColLength)

#define the first and last cell in out range
FirstCell = Excelwrksheet.Cells(2,1)
LastCell = Excelwrksheet.Cells(1+ColLength,RowLength)

#get the range
ExcelRange = Excelwrksheet.Range(FirstCell, LastCell)
ExcelRange.Value = students

#close and save the workbook
ExcelWrkbook.Close(True)
```

6. GUI:



```
def function1():
    os.system("python capture_database.py")

def function2():
    os.system("python training_dataset.py")

def function3():
    os.system("python recognizer.py")

def function4():
    os.system("python writing_in_excel.py")

#def function5():
#    os.startfile(os.getcwd()+"/developers/diet1frame1first.html");

def function6():
    root.destroy()

def attend():
    os.startfile(os.getcwd()+"/Attendance_Files/attendance"+str(datetime.now().date())+'.xlsx')

#stting title for the window
root.title("AUTOMATIC ATTENDANCE MANAGEMENT USING FACE RECOGNITION")

#creating a text label
Label(root, text="FACE RECOGNITION ATTENDANCE SYSTEM",font=("times new roman",20),fg="white",bg="maroon",height=2).grid(row=0,rowspan=2,columnspan=2,sticky=N+E+W+S,padx=5,pady=5)

#creating first button
Button(root,text="Create Dataset",font=("times new roman",20),bg="#0D47A1",fg='white',command=function1).grid(row=3,columnspan=2,sticky=W+E+N+S,padx=5,pady=5)

#writing database in Excel
Button(root,text="Write Database in Excel",font=("times new roman",20),bg="#0D47A1",fg='white',command=function4).grid(row=4,columnspan=2,sticky=N+E+W+S,padx=5,pady=5)

#creating second button
Button(root,text="Train Dataset",font=("times new roman",20),bg="#0D47A1",fg='white',command=function2).grid(row=5,columnspan=2,sticky=N+E+W+S,padx=5,pady=5)

#creating third button
Button(root,text="Mark Attendance",font=('times new roman',20),bg="#0D47A1",fg="white",command=function3).grid(row=6,columnspan=2,sticky=N+E+W+S,padx=5,pady=5)

#creating attendance button
Button(root,text="Attendance Sheet",font=('times new roman',20),bg="#0D47A1",fg="white",command=attend).grid(row=7,columnspan=2,sticky=N+E+W+S,padx=5,pady=5)

#Button(root,text="Developers",font=('times new roman',20),bg="#0D47A1",fg="white",command=function5).grid(row=8,columnspan=2,sticky=N+E+W+S,padx=5,pady=5)

Button(root,text="Exit",font=('times new roman',20),bg="maroon",fg="white",command=function6).grid(row=9,columnspan=2,sticky=N+E+W+S,padx=5,pady=5)


root.mainloop()
```

Bibliography and Resource

LBPH - Local Binary Pattern Histogram (**LBPH**) **algorithm** is a simple solution on face recognition problem, which can recognize both front face and side face.

HDR **-** High-dynamic-range.

OpenCV - python library for computer vision.

Harcascade - A Harcascade is basically a classifier which is used to detect the object for which

 it has been trained for.

https://lpuin-my.sharepoint.com/:u:/g/personal/vishal_11615139_lpu_in/ESZ0oMNjXq1Jlaq2yCUAlt0BR7YNkLoPyOzfSfHtnK23Cw?e=jR6fIk

https://www.instructables.com/id/Real-time-Face-Recognition-an-End-to-end-Project/

https://pdfs.semanticscholar.org/b690/8248f52c917c8202e5bb1d39a19c0e018813.pdf

https://towardsdatascience.com/face-recognition-for-beginners-a7a9bd5eb5c2

https://www.tutorialspoint.com/

https://www.pyimagesearch.com/2018/06/11/how-to-build-a-custom-face-recognition-dataset/

https://medium.com/dataseries/face-recognition-with-opencv-haar-cascade-a289b6ff042a

https://towardsdatascience.com/face-recognition-how-lbph-works-90ec258c3d6b