Chapter 1

INTRODUCTION

1.1 Introduction

Attendance is prime important for both the teacher and student of an educational organization. So it is very important to keep record of the attendance. The problem arises when we think about the traditional process of taking attendance in class room. Calling name or roll number of the student for attendance is not only a problem of time consumption but also it needs energy. So an automatic attendance system can solve all above problems.

There are some automatic attendances making system which are currently used by much institution. One of such system is biometric technique and RFID system. Although it is automatic and a step ahead of traditional method it fails to meet the time constraint. The student has to wait in queue for giving attendance, which is time taking.

This project introduces an involuntary attendance marking system, devoid of any kind of interference with the normal teaching procedure. The system can be also implemented during exam sessions or in other teaching activities where attendance is highly essential. This system eliminates classical student identification such as calling name of the student, or checking respective identification cards of the student, which can not only interfere with the ongoing teaching process, but also can be stressful for students during examination sessions. In addition, the students have to register in the database to be recognized. The enrolment can be done on the spot through the user-friendly interface.

1.2 Background

Face recognition is crucial in daily life in order to identify family, friends or someone we are familiar with. We might not perceive that several steps have actually taken in order to identify human faces. Human intelligence allows us to receive information and interpret the information in the recognition process. We receive information through the image projected into our eyes, by specifically retina in the form of light. Light is a form of electromagnetic waves which are radiated from a source onto an object and projected to human vision. Robinson-Riegler,

G., & Robinson-Riegler, B. (2008) mentioned that after visual processing done by the human visual system, we actually classify shape, size, contour and the texture of the object in order to analyze the information. The analyzed information will be compared to other representations of objects or face that exist in our memory to recognize. In fact, it is a hard challenge to build an automated system to have the same capability as a human to recognize faces. However, we need large memory to recognize different faces, for example, in the Universities, there are a lot of students with different race and gender, it is impossible to remember every face of the individual without making mistakes. In order to overcome human limitations, computers with almost limitless memory, high processing speed and power are used in face recognition systems.

Nowadays, face recognition system is prevalent due to its simplicity and awesome performance. For instance, airport protection systems and FBI use face recognition for criminal investigations by tracking suspects, missing children and drug activities (Robert Silk, 2017). Apart from that, Facebook which is a popular social networking website implement face recognition to allow the users to tag their friends in the photo for entertainment purposes (Sidney Fussell, 2018). Furthermore, Intel Company allows the users to use face recognition to get access to their online account (Reichert, C., 2017). Apple allows the users to unlock their mobile phone, iPhone X by using face recognition (deAgonia, M., 2017).

The work on face recognition began in 1960. Woody Bledsoe, Helen Chan Wolf and Charles Bisson had introduced a system which required the administrator to locate eyes, ears, nose and mouth from images. The distance and ratios between the located features and the common reference points are then calculated and compared. The studies are further enhanced by Goldstein, Harmon, and Lesk in 1970 by using other features such as hair colour and lip thickness to automate the recognition. In 1988, Kirby and Sirovich first suggested principle component analysis (PCA) to solve face recognition problem. Many studies on face recognition were then conducted continuously until today (Ashley DuVal, 2012).

1.3 Problem Statement

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 a large number of students might find it difficult to have the attendance sheet being passed around the class. Thus, face recognition 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.

The paper proposed by Zhao, W et al. (2003) has listed the difficulties of facial identification. One of the difficulties of facial identification is the identification between known and unknown images. In addition, paper proposed by Pooja G.R et al. (2010) found out that the training process for face recognition student attendance system is slow and time-consuming.

Hence, there is a need to develop a real time operating student attendance system which means the identification process must be done within defined time constraints to prevent omission. The extracted features from facial images which represent the identity of the students have to be consistent towards a change in background, illumination, pose and expression. High accuracy and fast computation time will be the evaluation points of the performance.

1.4 Aims and Objectives

The objective of this project is to develop face recognition attendance system. Expected achievements in order to fulfill the objectives are:

- To detect the face segment from the video frame.
- To extract the useful features from the face detected.
- To classify the features in order to recognize the face detected.
- To record the attendance of the identified student.

1.5 Flow Chart

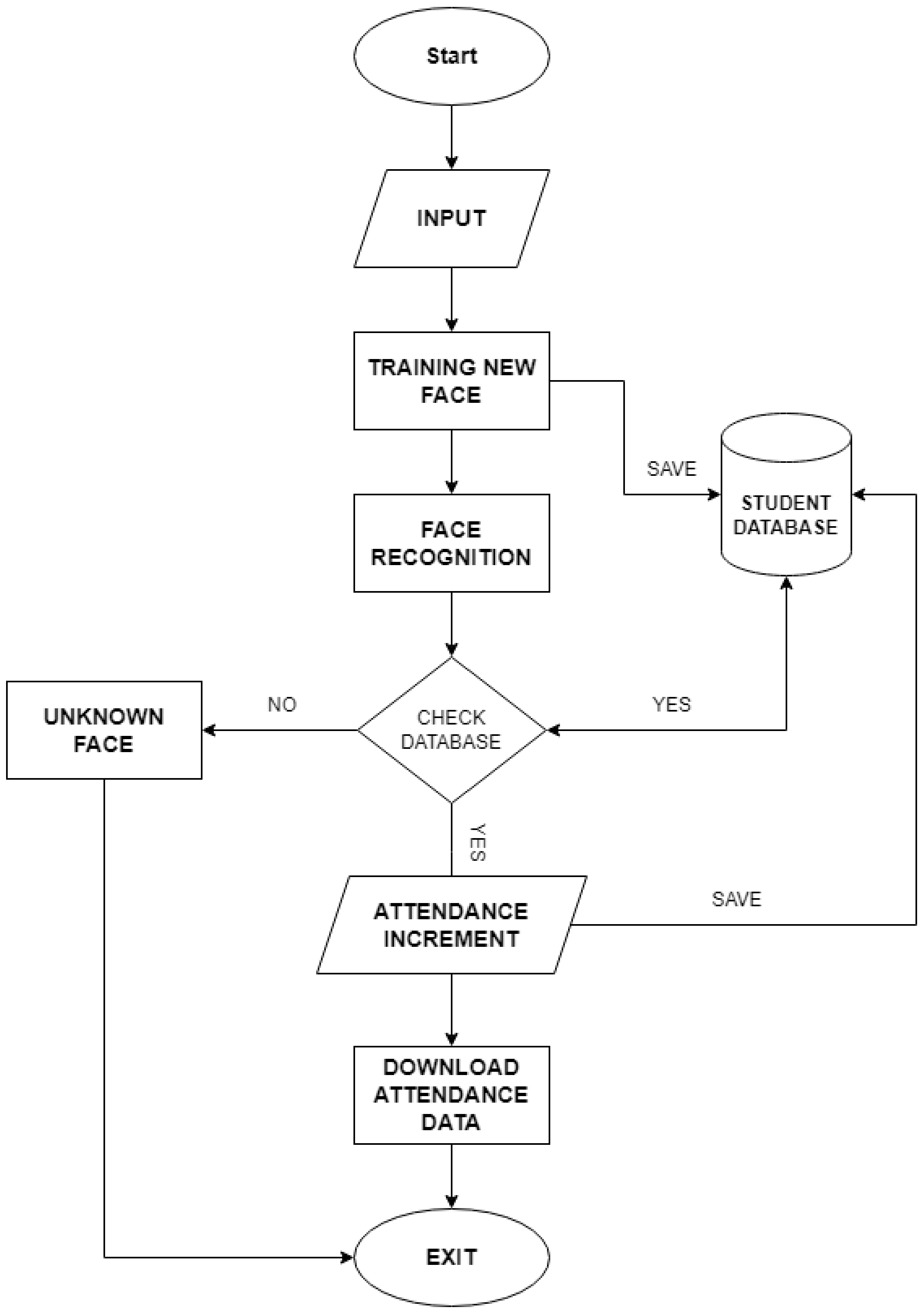


Fig: Flow Chart

1.6 Scope of the project

We are setting up to design a system comprising of two modules. The first module (face detector) is a mobile component, which is basically a camera application that captures student faces and stores them in a file using computer vision face detection algorithms and face extraction techniques. The second module is a desktop application that does face recognition of the captured images (faces) in the file, marks the students register and then stores the results in a database for future analysis.

Chapter 2

Literature Review

2.1 Student Attendance System

RFID (Radio Frequency Identification) card system, fingerprint system and iris recognition system. RFID card system is implemented due to its simplicity. However, the user tends to help their friends to check in as long as they have their friend's ID card. The fingerprint system is indeed effective but not efficient because it takes time for the verification process so the user has to line up and perform the verification one by one. However, for face recognition, the human face is always exposed and contain less information compared to iris. Iris recognition system which contains more detail might invade the privacy of the user. Voice recognition is available, but it is less accurate compared to other methods. Hence, face recognition system is suggested to be implemented in the student attendance system.

System Type	Advantage	Disadvantage
RFID card system	Simple	Fraudulent usage
Fingerprint system	Accurate	Time-consuming
Voice recognition system		Less accurate
Iris recognition system	Accurate	Privacy Invasion

Table 2.1: Advantages & Disadvantages of Different Biometric System.

2.2 Digital Image Processing

Digital Image Processing is the processing of images which are digital in nature by a digital computer. Digital image processing techniques are motivated by three major applications mainly:

- Improvement of pictorial information for human perception
- Image processing for autonomous machine application
- Efficient storage and transmission.

2.3 Image Representation in a Digital Computer

An image is a 2-Dimensional light intensity function

$\mathbf{f}\ (\mathbf{x}, \mathbf{y}) = \mathbf{r}\ (\mathbf{x}, \mathbf{y}) \times \mathbf{i}\ (\mathbf{x}, \mathbf{y})$ - (2.0)

Where, r (x, y) is the reflectivity of the surface of the corresponding image point. I (x, y) Represents the intensity of the incident light. A digital image f(x, y) is discretized both in spatial co-ordinates by grids and in brightness by quantization. Effectively, the image can be represented as a matrix whose row, column indices specify a point in the image and the element value identifies gray level value at that point. These elements are referred to as pixels.

Typically following image processing applications, the image
size which is used is$\mathbf{256} \times \mathbf{256}$, elements, $\mathbf{640} \times \mathbf{480}$ pixels or $\mathbf{1024}$
$\times\ \mathbf{1024}$ pixels. Quantization of these matrix pixels is done at 8 bits for black and white images and 24 bits for colored images (because of the three color planes Red, Green and Blue each at 8 bits).

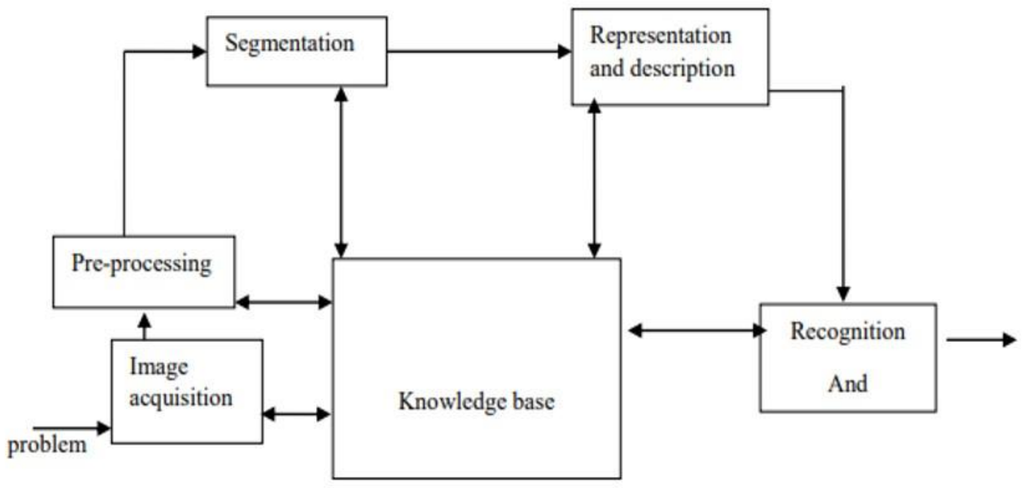
2.4 Steps in Digital Image Processing

Digital image processing involves the following basic tasks:

- Image Acquisition - An imaging sensor and the capability to digitize the signal produced by the sensor.

- Preprocessing – Enhances the image quality, filtering, contrast enhancement etc.
- Segmentation – Partitions an input image into constituent parts of objects.
- Description/feature Selection – extracts the description of image objects suitable for further computer processing.
- Recognition and Interpretation – Assigning a label to the object based on the information provided by its descriptor.

  Interpretation assigns meaning to a set of labelled objects.
- Knowledge Base – This helps for efficient processing as well as inter module cooperation.



2.5 Definition of Terms and History

Face Detection

Face detection is the process of identifying and locating all the present faces in a single image or video regardless of their position, scale, orientation, age and expression. Furthermore, the detection should be irrespective of extraneous illumination conditions and the image and video content.

Face Recognition

Face Recognition is a visual pattern recognition problem, where the face, represented as a three dimensional object that is subject to varying illumination, pose and other factors, needs to be identified based on acquired images.

Face Recognition is therefore simply the task of identifying an already detected face as a known or unknown face and in more advanced cases telling exactly whose face it is.

Difference between Face Detection and Face Recognition

Face detection answers the question, where is the face? It identifies an object as a “face” and locates it in the input image. Face Recognition on the other hand answers the question who is this? Or whose face is it? It decides if the detected face is someone known or unknown based on the database of faces it uses to validate this input image.It can therefore be seen that face detections output (the detected face) is the input to the face recognizer and the face Recognition’s output is the final decision i.e. face known or face unknown.

Face Detection

A face Detector has to tell whether an image of arbitrary size contains a human face and if so, where it is. Face detection can be performed based on several cues: skin color (for faces in color images and videos, motion (for faces in videos), facial/head shape, facial appearance or a combination of these parameters. Most face detection algorithms are appearance based without using other cues. An input image is scanned at all possible locations and scales by a sub window. Face detection is posed as classifying the pattern in the sub window either as a face or a non-face. The face/non-face classifier is learned from face and non-face training examples using

Detection Method	Advantage	Disadvantage
Viola Jones Algorithm	1. High detection Speed. 2. High Accuracy	1.Limited Head Pose. 2.Not able to detect dark faces.
Local Binary Pattern Histogram	1.Simple computation. 2.High tolerance against the monotonic	1.Only used for binary and grey images. 2.Overall performance is inaccurate

Table: Advantages & Disadvantages of Face Detection Methods.

Viola-Jones Algorithm

Viola-Jones algorithm which was introduced by P. Viola, M. J. Jones (2001) is the most popular algorithm to localize the face segment from static images or video frame. Basically the concept of Viola-Jones algorithm consists of four parts. The first part is known as Haar feature, second part is where integral image is created, followed by implementation of Adaboost on the third part and lastly cascading process.

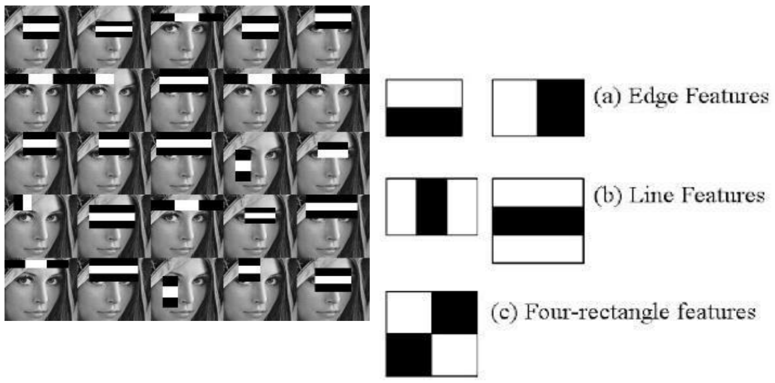
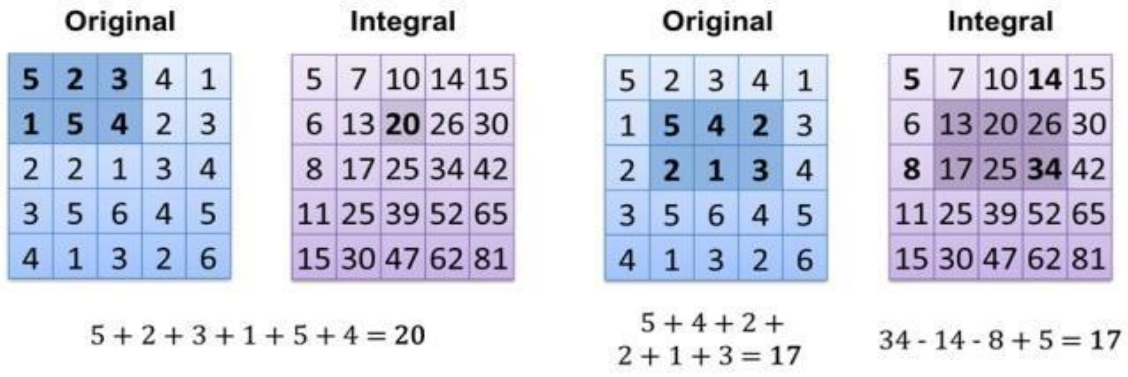


Fig: Haar Feature

Viola-Jones algorithm analyses a given image using Haar features consisting of multiple rectangles (Mekha Joseph et al., 2016). In the fig shows several types of Haar features. The features perform as window function mapping onto the image. A single value result, which representing each feature can be computed by subtracting the sum of the white rectangle(s) from the sum of the black rectangle(s) (Mekha Joseph et al., 2016).



5	**2**	**3**	4	1
1	**5**	**4**	2	3
2	2	1	3	4
3	5	6	4	5
4	1	3	2	6

5	7	10	14	15
6	13	**20**	26	30
8	17	25	34	42
11	25	39	52	65
15	30	47	62	81

5	2	3	4	1
1	**5**	**4**	**2**	3
2	**2**	**1**	**3**	4
3	5	6	4	5
4	1	3	2	6

5	7	10	**14**	15
6	13	20	26	30
8	17	25	**34**	42
11	25	39	52	65
15	30	47	62	81

Fig: Integral of Image

The value of integrating image in a specific location is the sum of pixels on the left and the top of the respective location. In order to illustrate clearly, the value of the integral image at location 1 is the sum of the pixels in rectangle A. The values of integral image at the rest of the locations are cumulative. For instance, the value at location 2 is summation of A and B, (A + B), at location 3 is summation of A and C, (A + C), and at location 4 is summation of all the regions, (A + B + C + D). Therefore, the sum within the D region can be computed with only addition and subtraction of diagonal at location 4 + 1 − (2 + 3) to eliminate rectangles A, B and C.

Local Binary Patterns Histogram

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) 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.

LBPH algorithm work step by step: LBPH algorithm work in 5 steps.

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 parameters radius and neighbors.

The image below shows this procedure:

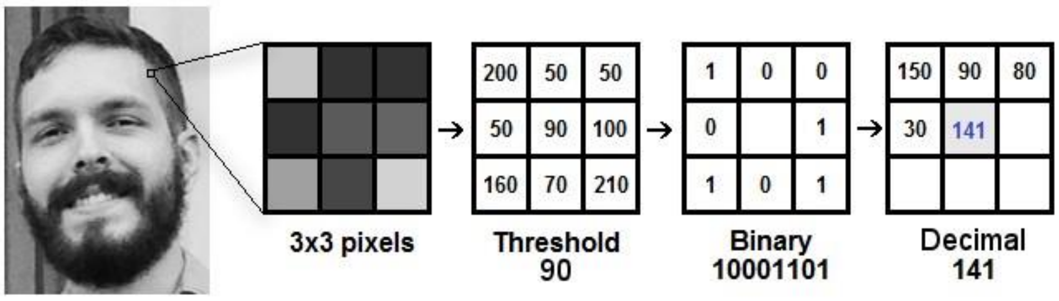


Fig: LBP Operation

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 final 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 actually 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.

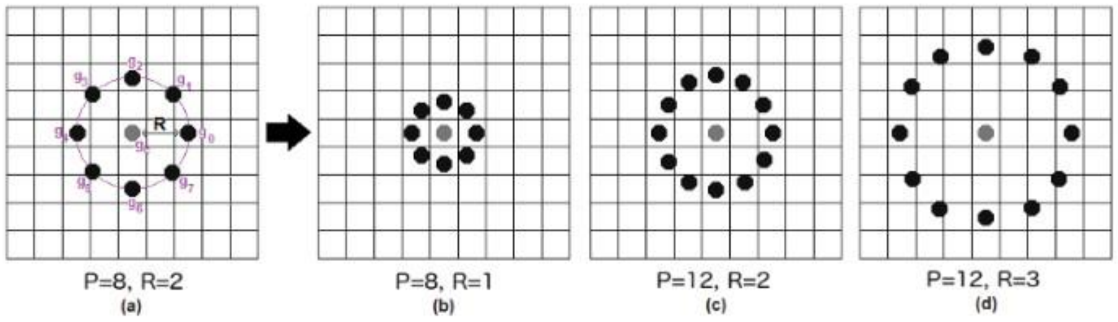


Fig: The LBP operation Radius Change

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:

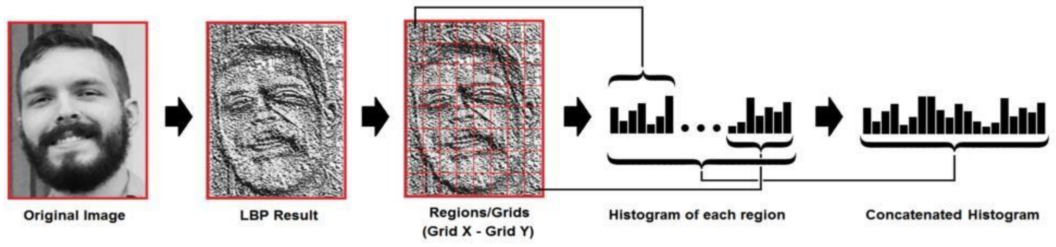


Fig: Extracting The Histogram

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.

5. **Performing the 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. Note: don't be fooled about the 'confidence' name, as lower confidences are better because it means the distance between the two histograms is closer.
- 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.

Chapter 3

Model Implementation & Analysis

3.1 Introduction

Face detection involves separating image windows into two classes; one containing faces (turning the background (clutter). It is difficult because although commonalities exist between faces, they can vary considerably in terms of age, skin color and facial expression. The problem is further complicated by differing lighting conditions, image qualities and geometries, as well as the possibility of partial occlusion and disguise. An ideal face detector would therefore be able to detect the presence of any face under any set of lighting conditions, upon any background. The face detection task can be broken down into two steps. The first step is a classification task that takes some arbitrary image as input and outputs a binary value of yes or no, indicating whether there are any faces present in the image. The second step is the face localization task that aims to take an image as input and output the location of any face or faces within that image as some bounding box with (x, y, width, height). After taking the picture the system will compare the equality of the pictures in its database and give the most related result.

We will use open CV platform and will do the coding in python language.

3.2 Model Implementation

The main components used in the implementation approach are open source computer vision library (OpenCV). One of OpenCV's goals is to provide a simple-to-use computer vision infrastructure that helps people build fairly sophisticated vision applications quickly. OpenCV library contains over 500 functions that span many areas in vision. The primary technology behind Face recognition is OpenCV. The user stands in front of the camera keeping a minimum distance of 50cm and his image is taken as an input. The frontal face is extracted from the image then converted to rgb and stored. When a user requests for recognition the frontal face is extracted from the captured video frame through the camera. The eigen value is re-calculated for the test face and it is matched with the stored data for the closest neighbor.

3.3 Design Requirements

We used some tools to build the system. Without the help of these tools it would not be possible to make it done. Here we will discuss about the most important one.

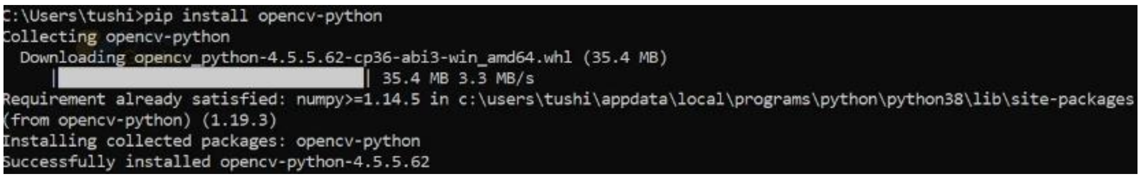
3.3.1 Software Implementation

OpenCV: We used OpenCV 3 dependency for python 3. OpenCV is library where there are lots of image processing functions are available. This is very useful library for image processing. Even one can get expected outcome without writing a single code. The library is cross-platform and free for use under the open-source BSD license. Example of some supported functions are given bellow:

- **Derivation**: Gradient / laplacian computing, contours delimitation
- **Hough transforms:** lines, segments, circles, and geometrical shapes detection
- **Histograms**: computing, equalization, and object localization with back projection algorithm
- **Segmentation**: thresholding, distance transform, foreground / background detection, watershed segmentation
- **Filtering**: linear and nonlinear filters, morphological operations

- **Cascade detectors**: detection of face, eye, car plates
- **Interest points**: detection and matching
- **Video processing:** optical flow, background subtraction, camshaft (object tracking)
- **Photography**: panoramas realization, high definition imaging (HDR), image inpainting

So it was very important to install OpenCV. But installing OpenCV 3 is a complex process. How we did it is given below:



```
C:\Users\tushi>pip install opencv-python
Collecting opencv-python
  Downloading opencv_python-4.5.5.62-cp36-abi3-win_amd64.whl (35.4 MB)
     |████████████████████████████████| 35.4 MB 3.3 MB/s
Requirement already satisfied: numpy>=1.14.5 in c:\users\tushi\appdata\local\programs\python\python38\lib\site-packages
(from opencv-python) (1.19.3)
Installing collected packages: opencv-python
Successfully installed opencv-python-4.5.5.62
```

Fig: Installing OpenCV

2.Python IDE: There are lots of IDEs for python. Some of them are PyCharm, VScode etc. PyCharm and VScode both are very excellent and free but we used VScode as it feature- rich than PyCharm.

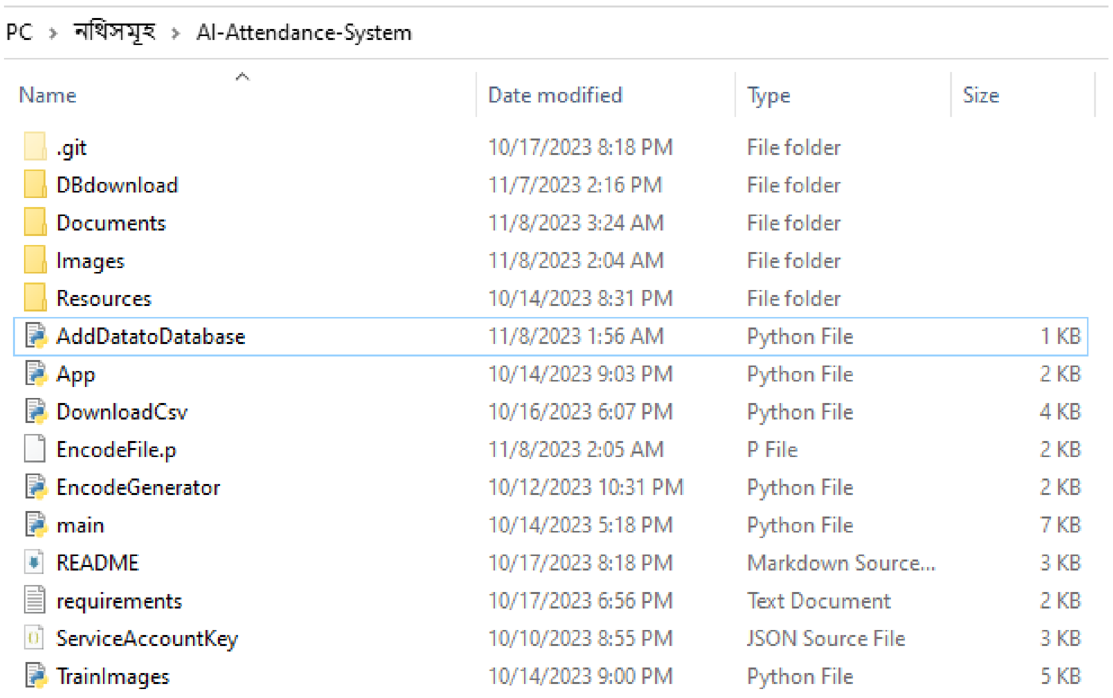
3.iVcam: Free webcam application. iVCam is a free application that allows people to turn their handheld devices into webcams for their computers and laptops.

Chapter 4

Code Implementation

4.1 Code Implementation

All our code is written in Python language. First here is our project directory structure and files.

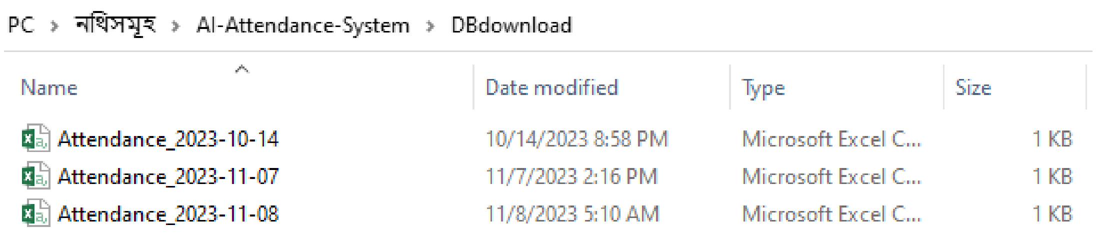


PC › নথিসমূহ › AI-Attendance-System

Name	Date modified	Type	Size
.git	10/17/2023 8:18 PM	File folder	
DBdownload	11/7/2023 2:16 PM	File folder	
Documents	11/8/2023 3:24 AM	File folder	
Images	11/8/2023 2:04 AM	File folder	
Resources	10/14/2023 8:31 PM	File folder	
AddDatatoDatabase	11/8/2023 1:56 AM	Python File	1 KB
App	10/14/2023 9:03 PM	Python File	2 KB
DownloadCsv	10/16/2023 6:07 PM	Python File	4 KB
EncodeFile.p	11/8/2023 2:05 AM	P File	2 KB
EncodeGenerator	10/12/2023 10:31 PM	Python File	2 KB
main	10/14/2023 5:18 PM	Python File	7 KB
README	10/17/2023 8:18 PM	Markdown Source...	3 KB
requirements	10/17/2023 6:56 PM	Text Document	2 KB
ServiceAccountKey	10/10/2023 8:55 PM	JSON Source File	3 KB
TrainImages	10/14/2023 9:00 PM	Python File	5 KB

All those file in the project directory.

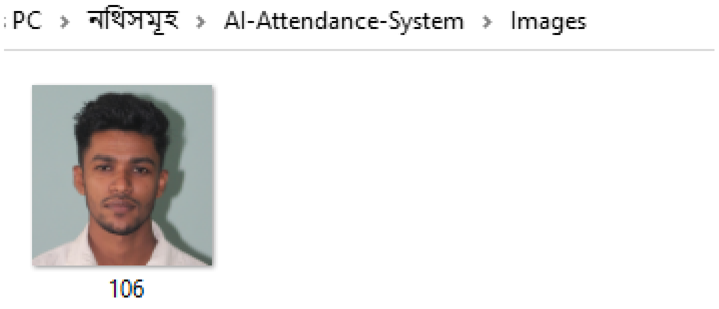
DBdownload: Where downloaded attendance file is saved.



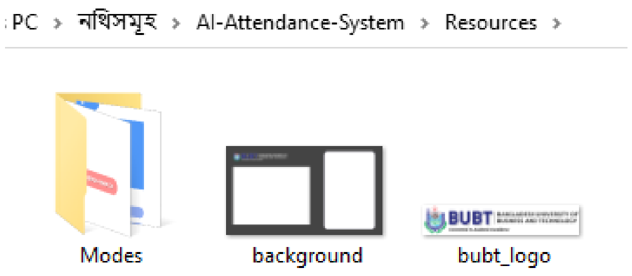
PC › নথিসমূহ › AI-Attendance-System › DBdownload

Name	Date modified	Type	Size
Attendance_2023-10-14	10/14/2023 8:58 PM	Microsoft Excel C...	1 KB
Attendance_2023-11-07	11/7/2023 2:16 PM	Microsoft Excel C...	1 KB
Attendance_2023-11-08	11/8/2023 5:10 AM	Microsoft Excel C...	1 KB

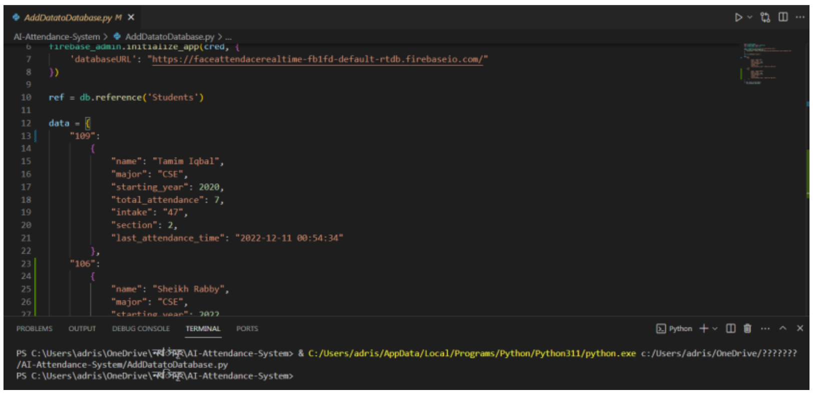
Images: Captured Face Data is stored here.



Resources: UI design files.



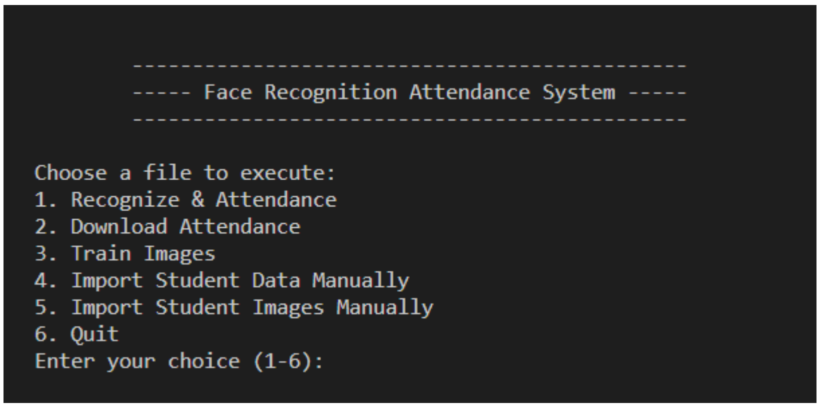
AddDataToDatabase.py: Python script to upload large data to Firebase server.



```python
firebase_admin.initialize_app(cred, {
    'databaseURL': "https://faceattendacerealtime-fb1fd-default-rtdb.firebaseio.com/"
})

ref = db.reference('Students')

data = {
    "109":
        {
            "name": "Tamim Iqbal",
            "major": "CSE",
            "starting_year": 2020,
            "total_attendance": 7,
            "intake": "47",
            "section": 2,
            "last_attendance_time": "2022-12-11 00:54:34"
        },
    "106":
        {
            "name": "Sheikh Rabby",
            "major": "CSE",
            "starting_year": 2022
```

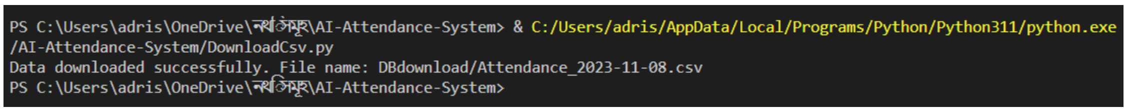
```
PS C:\Users\adris\OneDrive\নথিসমূহ\AI-Attendance-System> & C:/Users/adris/AppData/Local/Programs/Python/Python311/python.exe c:/Users/adris/OneDrive/???????
/AI-Attendance-System/AddDatatoDatabase.py
PS C:\Users\adris\OneDrive\নথিসমূহ\AI-Attendance-System>
```

App.py: Script to start the Application.



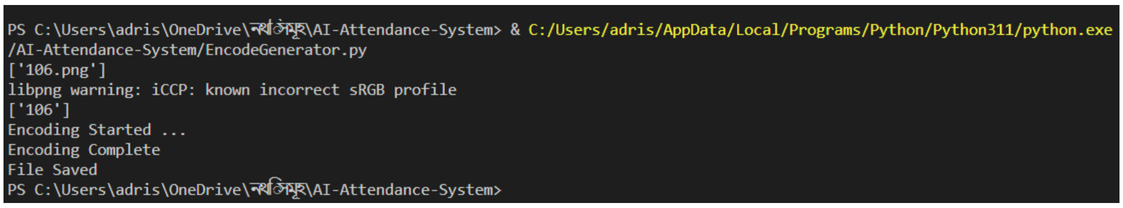
```
          ----------------------------------------------
          ----- Face Recognition Attendance System -----
          ----------------------------------------------

Choose a file to execute:
1. Recognize & Attendance
2. Download Attendance
3. Train Images
4. Import Student Data Manually
5. Import Student Images Manually
6. Quit
Enter your choice (1-6):
```

DownloadCsv.py: Script to download attendance data from firebase server.



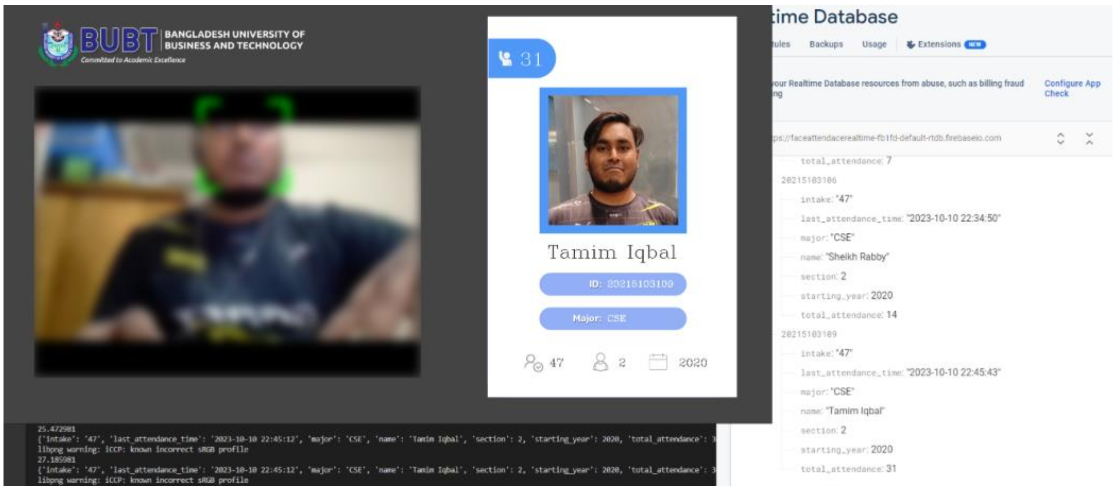
```
PS C:\Users\adris\OneDrive\নথিসমূহ\AI-Attendance-System> & C:/Users/adris/AppData/Local/Programs/Python/Python311/python.exe
/AI-Attendance-System/DownloadCsv.py
Data downloaded successfully. File name: DBdownload/Attendance_2023-11-08.csv
PS C:\Users\adris\OneDrive\নথিসমূহ\AI-Attendance-System>
```

Encodegenerator.py: Script to upload face images to firebase server.



```
PS C:\Users\adris\OneDrive\নথিসমূহ\AI-Attendance-System> & C:/Users/adris/AppData/Local/Programs/Python/Python311/python.exe
/AI-Attendance-System/EncodeGenerator.py
['106.png']
libpng warning: iCCP: known incorrect sRGB profile
['106']
Encoding Started ...
Encoding Complete
File Saved
PS C:\Users\adris\OneDrive\নথিসমূহ\AI-Attendance-System>
```

Main.py: Main script file that recognize faces and attendance system.



```
25.472981
{'intake': '47', 'last_attendance_time': '2023-10-10 22:45:12', 'major': 'CSE', 'name': 'Tamim Iqbal', 'section': 2, 'starting_year': 2020, 'total_attendance': 3
libpng warning: iCCP: known incorrect sRGB profile
27.185981
{'intake': '47', 'last_attendance_time': '2023-10-10 22:45:12', 'major': 'CSE', 'name': 'Tamim Iqbal', 'section': 2, 'starting_year': 2020, 'total_attendance': 3
libpng warning: iCCP: known incorrect sRGB profile
```

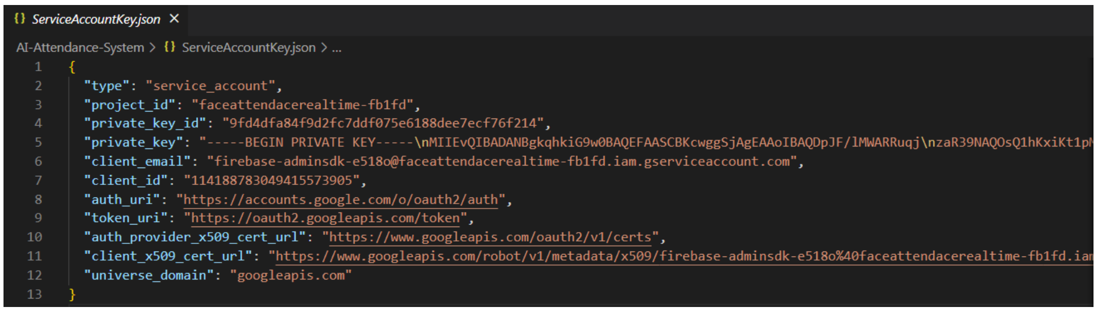
your Realtime Database resources from abuse, such as billing fraud
ng

ps://faceattendacerealtime-fb1fd-default-rtdb.firebaseio.com

```
    total_attendance: 7
20215103106
    intake: "47"
    last_attendance_time: "2023-10-10 22:34:50"
    major: "CSE"
    name: "Sheikh Rabby"
    section: 2
    starting_year: 2020
    total_attendance: 14
20215103109
    intake: "47"
    last_attendance_time: "2023-10-10 22:45:43"
    major: "CSE"
    name: "Tamim Iqbal"
    section: 2
    starting_year: 2020
    total_attendance: 31
```

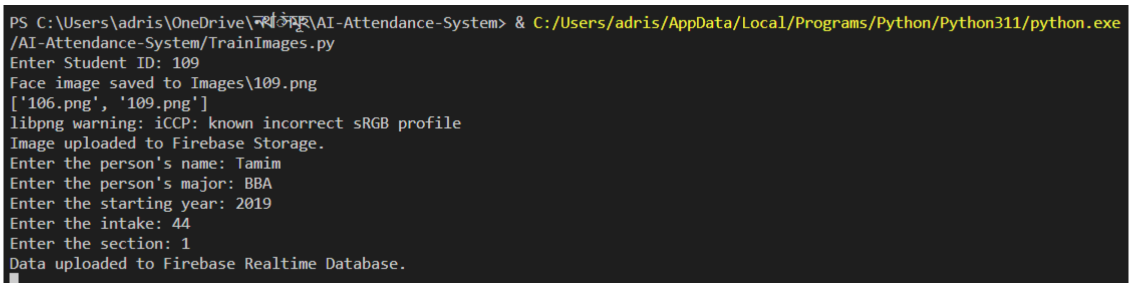
Requirements.txt: Required python library list that we need to run the application

ServiceAccountKey.json: json file holds keys to connect firebase database and storage with application.



```
{
  "type": "service_account",
  "project_id": "faceattendacerealtime-fb1fd",
  "private_key_id": "9fd4dfa84f9d2fc7ddf075e6188dee7ecf76f214",
  "private_key": "-----BEGIN PRIVATE KEY-----\nMIIEvQIBADANBgkqhkiG9w0BAQEFAASCBKcwggSjAgEAAoIBAQDpJF/1MWARRuqj\nzaR39NAQOsQ1hKxiKt1p
  "client_email": "firebase-adminsdk-e518o@faceattendacerealtime-fb1fd.iam.gserviceaccount.com",
  "client_id": "114188783049415573905",
  "auth_uri": "https://accounts.google.com/o/oauth2/auth",
  "token_uri": "https://oauth2.googleapis.com/token",
  "auth_provider_x509_cert_url": "https://www.googleapis.com/oauth2/v1/certs",
  "client_x509_cert_url": "https://www.googleapis.com/robot/v1/metadata/x509/firebase-adminsdk-e518o%40faceattendacerealtime-fb1fd.ia
  "universe_domain": "googleapis.com"
}
```

TrainImages.py: Script that detects & captures faces and takes userdata and uploads them in firebase server.



```
PS C:\Users\adris\OneDrive\নথিসমূহ\AI-Attendance-System> & C:/Users/adris/AppData/Local/Programs/Python/Python311/python.exe
/AI-Attendance-System/TrainImages.py
Enter Student ID: 109
Face image saved to Images\109.png
['106.png', '109.png']
libpng warning: iCCP: known incorrect sRGB profile
Image uploaded to Firebase Storage.
Enter the person's name: Tamim
Enter the person's major: BBA
Enter the starting year: 2019
Enter the intake: 44
Enter the section: 1
Data uploaded to Firebase Realtime Database.
```

4.1.1 main.py

```
import os
import pickle
import numpy as np
import face_recognition
import cvzone
import firebase_admin
from firebase_admin import credentials
from firebase_admin import db
from firebase_admin import storage
import numpy as np
from datetime import datetime
cred = credentials.Certificate("ServiceAccountKey.json")
firebase_admin.initialize_app(cred, {
    'databaseURL': "https://faceattendacerealtime-fb1fd-default-rtdb.firebaseio.com/",
    'storageBucket': "faceattendacerealtime-fb1fd.appspot.com"
})
bucket = storage.bucket()
cap = cv2.VideoCapture(0)
cap.set(3, 640)
cap.set(4, 480)
imgBackground = cv2.imread('Resources/background.png')
folderModePath = 'Resources/Modes'
modePathList = os.listdir(folderModePath)
imgModeList = []
```

```
for path in modePathList:
    imgModeList.append(cv2.imread(os.path.join(folderModePath, path)))
# print(len(imgModeList))
# Load the encoding file
print("Loading Encode File ...")
file = open('EncodeFile.p', 'rb')
encodeListKnownWithIds = pickle.load(file)
file.close()
encodeListKnown, studentIds = encodeListKnownWithIds
# print(studentIds)
print("Encode File Loaded")
modeType = 0
counter = 0
id = -1
imgStudent = []
while True:
    success, img = cap.read()
    imgS = cv2.resize(img, (0, 0), None, 0.25, 0.25)
    imgS = cv2.cvtColor(imgS, cv2.COLOR_BGR2RGB)
    faceCurFrame = face_recognition.face_locations(imgS)
    encodeCurFrame = face_recognition.face_encodings(imgS, faceCurFrame)
    imgBackground[162:162 + 480, 55:55 + 640] = img
    imgBackground[44:44 + 633, 808:808 + 414] = imgModeList[modeType]
    if faceCurFrame:
        for encodeFace, faceLoc in zip(encodeCurFrame, faceCurFrame):
            matches = face_recognition.compare_faces(encodeListKnown, encodeFace)
            faceDis = face_recognition.face_distance(encodeListKnown, encodeFace)
            # print("matches", matches)
            # print("faceDis", faceDis)
            matchIndex = np.argmin(faceDis)
            # print("Match Index", matchIndex)
            if matches[matchIndex]:
                # print("Known Face Detected")
                # print(studentIds[matchIndex])
                y1, x2, y2, x1 = faceLoc
                y1, x2, y2, x1 = y1 * 4, x2 * 4, y2 * 4, x1 * 4
                bbox = 55 + x1, 162 + y1, x2 - x1, y2 - y1
                imgBackground = cvzone.cornerRect(imgBackground, bbox, rt=0)
                id = studentIds[matchIndex]
                if counter == 0:
                    cvzone.putTextRect(imgBackground, "Loading", (275, 400))
                    cv2.imshow("Face Attendance", imgBackground)
                    cv2.waitKey(1)
                    counter = 1
                    modeType = 1
        if counter != 0:
            if counter == 1:
                # Get the Data
                studentInfo = db.reference(f'Students/{id}').get()
                print(studentInfo)
                # Get the Image from the storage
                blob = bucket.get_blob(f'Images/{id}.png')
                array = np.frombuffer(blob.download_as_string(), np.uint8)
                imgStudent = cv2.imdecode(array, cv2.COLOR_BGRA2BGR)
                # Update data of attendance
                datetimeObject = datetime.strptime(studentInfo['last_attendance_time'],
                                                   "%Y-%m-%d %H:%M:%S")
                secondsElapsed = (datetime.now() - datetimeObject).total_seconds()
                print(secondsElapsed)
                if secondsElapsed > 30:
                    ref = db.reference(f'Students/{id}')
                    studentInfo['total_attendance'] += 1
                    ref.child('total_attendance').set(studentInfo['total_attendance'])
                    ref.child('last_attendance_time').set(datetime.now().strftime("%Y-%m-%d
%I:%M:%S"))
                else:
                    modeType = 3
                    counter = 0
                    imgBackground[44:44 + 633, 808:808 + 414] = imgModeList[modeType]

            if modeType != 3:
                if 30 < counter < 50:
                    modeType = 2
                imgBackground[44:44 + 633, 808:808 + 414] = imgModeList[modeType]
                if counter <= 30:
                        cv2.putText(imgBackground, str(studentInfo['total_attendance']), (861,
125),
                                cv2.FONT_HERSHEY_COMPLEX, 1, (255, 255, 255), 1)
                    cv2.putText(imgBackground, str(studentInfo['major']), (1006, 550),
                                cv2.FONT_HERSHEY_COMPLEX, 0.5, (255, 255, 255), 1)
                    cv2.putText(imgBackground, str(id), (1006, 493),
                                cv2.FONT_HERSHEY_COMPLEX, 0.5, (255, 255, 255), 1)
                    cv2.putText(imgBackground, str(studentInfo['intake']), (910, 625),
                                cv2.FONT_HERSHEY_COMPLEX, 0.6, (100, 100, 100), 1)
                    cv2.putText(imgBackground, str(studentInfo['section']), (1025, 625),
                                cv2.FONT_HERSHEY_COMPLEX, 0.6, (100, 100, 100), 1)
                    cv2.putText(imgBackground, str(studentInfo['starting_year']), (1125, 625),
                                cv2.FONT_HERSHEY_COMPLEX, 0.6, (100, 100, 100), 1)

                                                (w, h), _ = cv2.getTextSize(studentInfo['name'],
cv2.FONT_HERSHEY_COMPLEX, 1, 1)
                    offset = (414 - w) // 2
                    cv2.putText(imgBackground, str(studentInfo['name']), (808 + offset, 445),
                                cv2.FONT_HERSHEY_COMPLEX, 1, (50, 50, 50), 1)
                    imgBackground[175:175 + 216, 909:909 + 216] = imgStudent
                counter += 1
                if counter >= 50:
                    counter = 0
                    modeType = 0
                    studentInfo = []
                    imgStudent = []
                    imgBackground[44:44 + 633, 808:808 + 414] = imgModeList[modeType]
    else:
        modeType = 0
        counter = 0
    # cv2.imshow("Webcam", img)
    cv2.imshow("Face Attendance", imgBackground)
    cv2.waitKey(1)
```

4.1.2 TrainImages.py

```
import cv2
import os
import numpy as np
from datetime import datetime
import firebase_admin
from firebase_admin import credentials
from firebase_admin import db
from firebase_admin import storage
cred = credentials.Certificate("ServiceAccountKey.json")
firebase_admin.initialize_app(cred, {
    'databaseURL': "https://faceattendacerealtime-fb1fd-default-rtdb.firebaseio.com/",
    'storageBucket': "faceattendacerealtime-fb1fd.appspot.com"
})
# Firebase Storage bucket
bucket = storage.bucket()
def capture_and_upload_face(output_directory, output_filename):
    # Load pre-trained Haar Cascade face detection model
    face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades +
'haarcascade_frontalface_default.xml')
    # Open a connection to the default camera (usually the built-in webcam)
    camera = cv2.VideoCapture(0)

    # Check if the camera is opened successfully
    if not camera.isOpened():
        print("Error: Could not open camera.")
        return
    frame_count = 0
    face_detected_count = 0
    while True:
        # Capture a frame from the camera
        ret, frame = camera.read()
        if not ret:
            print("Error: Could not capture frame.")
            break
        # Convert the frame to grayscale for face detection
        gray_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        # Perform face detection
        faces = face_cascade.detectMultiScale(gray_frame, scaleFactor=1.1,
minNeighbors=5, minSize=(30, 30))
        # Increment the frame count
        frame_count += 1

        # Check if a face is detected
        if len(faces) > 0:
            face_detected_count += 1
            for (x, y, w, h) in faces:
                # Crop the face to 216x216 pixels
                cropped_face = frame[y:y+h, x:x+w]
                cropped_face = cv2.resize(cropped_face, (216, 216))
                # Create the output directory if it doesn't exist
                if not os.path.exists(output_directory):
                    os.makedirs(output_directory)
                # Save the captured face to the output directory with the provided filename
                output_path = os.path.join(output_directory, f"{output_filename}.png")
                cv2.imwrite(output_path, cropped_face)
                print("Face image saved to", output_path)
                # Upload the face image to Firebase Storage
                image_url = upload_to_firebase(output_path, output_filename)
                # Capture additional data
                person_data = {
                    'name': input("Enter the person's name: "),
                    'major': input("Enter the person's major: "),
                    'starting_year': input("Enter the starting year: "),
                    'total_attendance': 0,
                    'intake': input("Enter the intake: "),
                    'section': input("Enter the section: "),
                    'last_attendance_time': datetime.now().strftime("%Y-%m-%d
%I:%M:%S"),
                }
                # Upload data to Firebase Realtime Database
                upload_data_to_firebase(output_filename, person_data)
            # Break the loop and stop capturing frames
            break
        # Check if a face has been detected in more than 10 frames
        if face_detected_count > 2:
            break
        # Check for user input to exit
        key = cv2.waitKey(1)
        if key == 27:  # ESC key
            break
    camera.release()
    # Destroy OpenCV windows
    cv2.destroyAllWindows()
def upload_to_firebase(image_path, output_filename):
    try:
        # Upload the image to Firebase Storage
        folderPath = 'Images'
        pathList = os.listdir(folderPath)
        print(pathList)
        imgList = []
        studentIds = []
        for path in pathList:
            imgList.append(cv2.imread(os.path.join(folderPath, path)))
            studentIds.append(os.path.splitext(path)[0])
            fileName = f'{folderPath}/{path}'
            bucket = storage.bucket()
            blob = bucket.blob(fileName)
            blob.upload_from_filename(fileName)
        print("Image uploaded to Firebase Storage.")
        return blob.public_url
    except Exception as e:
        print("Error uploading image to Firebase Storage:", str(e))
        return None

def upload_data_to_firebase(filename, data):
    try:
        # Initialize Firebase Realtime Database
        ref = db.reference('Students')
        ref.child(filename).set(data)
        print("Data uploaded to Firebase Realtime Database.")
    except Exception as e:
        print("Error uploading data to Firebase Realtime Database:", str(e))
if __name__ == "__main__":
    output_directory = 'Images'  # Change this to your desired output directory
    output_filename = input("Enter Student ID: ")
    capture_and_upload_face(output_directory, output_filename)
```

4.1.3 AddDatatoDatabase.py

```
import firebase_admin
from firebase_admin import credentials
from firebase_admin import db
cred = credentials.Certificate("ServiceAccountKey.json")
firebase_admin.initialize_app(cred, {
    'databaseURL': "https://faceattendacerealtime-fb1fd-default-rtdb.firebaseio.com/"
})
ref = db.reference('Students')
data = {
    "109":
        {
            "name": "Tamim Iqbal",
            "major": "CSE",
            "starting_year": 2020,
            "total_attendance": 7,
            "intake": "47",
            "section": 2,
            "last_attendance_time": "2022-12-11 00:54:34"
        },
    "106":
        {
            "name": "Sheikh Rabby",
            "major": "CSE",
            "starting_year": 2022,
            "total_attendance": 0,
            "intake": "47",
            "section": 2,
            "last_attendance_time": "2022-12-11 00:54:34"
        },
}
for key, value in data.items():
    ref.child(key).set(value)
```

4.1.4 App.py

```
import os
def execute_file(file_number):
    files = ["main.py",
"DownloadCsv.py","TrainImages.py","AddDatatoDatabase.py","EncodeGenerator.py
",]
    if 1 <= file_number <= len(files):
        file_to_execute = files[file_number - 1]
        os.system(f"python {file_to_execute}")
    else:
        print("Invalid input. Please choose a valid option.")


if __name__ == "__main__":
    while True:
        print("\t")
        print("\t")
        print("\t--------------------------------------------")
        print("\t----- Face Recognition Attendance System -----")
        print("\t--------------------------------------------")
        print("\t")
        print("Choose a file to execute:")
        print("1. Recognize & Attendance")
        print("2. Download Attendance")
        print("3. Train Images")
        print("4. Import Student Data Manually")
        print("5. Import Student Images Manually")
        print("6. Quit")

        user_input = input("Enter your choice (1-6): ")

        if user_input.lower() == 'q' or user_input == '6':
            break
        elif user_input.isdigit():
            execute_file(int(user_input))
        else:
            print("Invalid input. Please enter a number (1-6).")
```

4.1.5 EncodeGenerator.py

```
import cv2
import face_recognition
import pickle
import os
import firebase_admin
from firebase_admin import credentials
from firebase_admin import db
from firebase_admin import storage

cred = credentials.Certificate("ServiceAccountKey.json")
firebase_admin.initialize_app(cred, {
    'databaseURL': "https://faceattendacerealtime-fb1fd-default-rtdb.firebaseio.com/",
    'storageBucket': "faceattendacerealtime-fb1fd.appspot.com"
})

# Importing student images
folderPath = 'Images'
pathList = os.listdir(folderPath)
print(pathList)
imgList = []
studentIds = []
for path in pathList:
    imgList.append(cv2.imread(os.path.join(folderPath, path)))
    studentIds.append(os.path.splitext(path)[0])

    fileName = f'{folderPath}/{path}'
    bucket = storage.bucket()
    blob = bucket.blob(fileName)
    blob.upload_from_filename(fileName)



    # print(path)
    # print(os.path.splitext(path)[0])
print(studentIds)

def findEncodings(imagesList):
    encodeList = []
    for img in imagesList:
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        encode = face_recognition.face_encodings(img)[0]
        encodeList.append(encode)

    return encodeList

print("Encoding Started ...")
encodeListKnown = findEncodings(imgList)
encodeListKnownWithIds = [encodeListKnown, studentIds]
print("Encoding Complete")

file = open("EncodeFile.p", 'wb')
pickle.dump(encodeListKnownWithIds, file)
file.close()
print("File Saved")
```

4.2 Summary

In this long yet useful chapter we managed to cover the entire structure of how the system has been developed and how it functions to give the best outcome.

Chapter 5

Working Plan

5.1 Introduction

In this chapter, we observe the entire work structure, meaning how the scheduling was maintained throughout the developmental phase. We shall also see the financial foundation of this project and furthermore the feasibility study should be also discussed.

5.2 Work Breakdown Structure

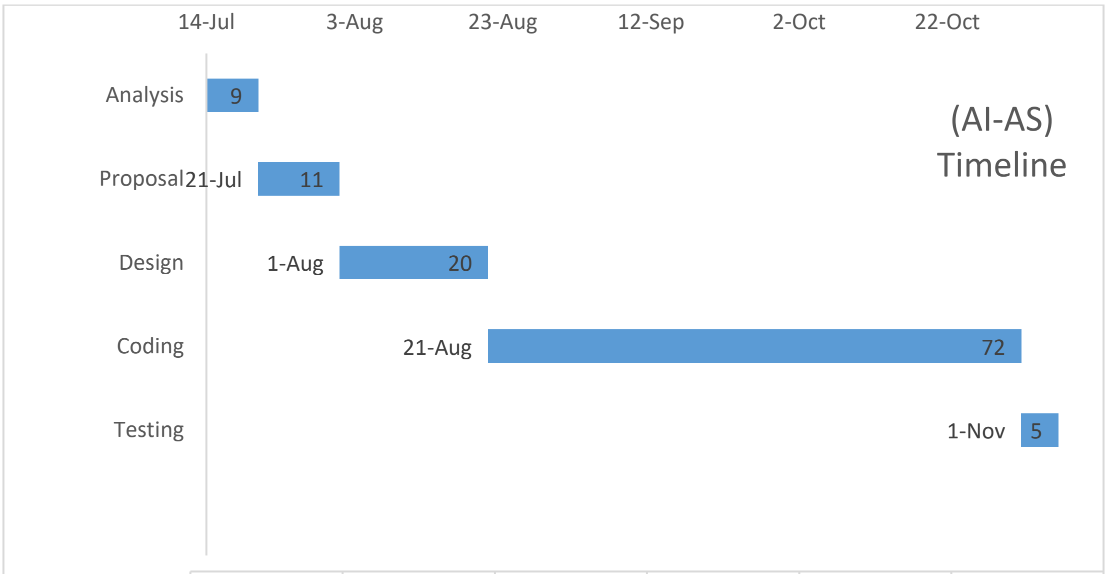
In order to develop this system, we gave enormous importance to scheduling because we believed if we want to provide the best of quality in a given period of time then we must give due importance to scheduling which also helped us to achieve a better results.

The figure below focuses the weekly work we had accomplished.

Week	Proposed Work
Week.1	Study related works
Week.1	Study in Python
Week.2	Study related works using OpenCV
Week.3	Study related works using processing
Week.3	Study Firebase database
Week.3	Study image processing
Week.4	Prototype design
Week.4	Finalize Prototype design
Week.5	Runnable with basic commands(Input, Output)
Week.5	Designing Lookahead table

Week.6	Creating environment for image processing
Week.7	Integrating all together
Week.7	Start coding
Week.8	Coding for basic instructions (Compare, Result, Accuracy measure etc.)
Week.8	Coding for single face detection
Week.9	Single face detection and Compare with database
Week.9	Multiple Face detection and Compare
Week.10	Attendance collection
Week.10	File Generate base on collective data
Week.10	Daily file generation of attendance

5.3.Gantt Chart



		Testing	Coding	Design	Proposal	Analysis
Start Date		1-Nov	21-Aug	1-Aug	21-Jul	12-Jul
🟧 Duration (Days)		5	72	20	11	9

Fig: Gantt Chart

5.5.Feasibility Study

Depending on the results of the initial investigation the survey is now expanded to a more detailed feasibility study. “**FEASIBILITY STUDY**” is a test of system proposal according to its workability, impact of the organization, ability to meet needs and effective use of the resources. It focuses on these major questions:

1. What are the user’s demonstrable needs and how does a candidate System meets them?
2. What resources are available for given candidate system?
3. What are the likely impacts of the candidate system on the organization?
4. Whether it is worth to solve the problem?

During feasibility analysis for on our project, following primary areas of interest are to be considered. Investigation and generating ideas about a new system does the following steps:

Steps in feasibility analysis

1. Form a project team and appoint a project leader.
2. Enumerate potential proposed system.
3. Define and identify characteristics of proposed system.
4. Determine and evaluate performance and cost effectively of each proposed system.
5. Weight system performance and cost data.
6. Select the best-proposed system.
7. Prepare and report final project directive to management.

Technical feasibility

A study of available resource that may affect the ability to achieve an acceptable system. This evaluation determines whether the technology needed for the proposed system is available or not.

- Can the work for the project be done with current equipment existing software technology & available personal?

- Can the system be upgraded if developed?
- If new technology is needed then what can be developed?

This is concerned with specifying equipment and software that will successfully satisfy the user requirement.

Economic feasibility

Economic justification is generally the "Bottom Line" consideration for most systems. Economic justification includes a broad range of concerns that includes cost benefit analysis. In this we weight the cost and the benefits associated with the candidate system and if it suits the basic purpose of the organization i.e. profit making, the project is making to the analysis and design phase.
The financial and the economic questions during the preliminary investigation are verified to estimate the following:

- The cost to conduct a full system investigation.
- The cost of hardware and software for the class of application being considered.
- The benefits in the form of reduced cost.
- The proposed system will give the minute information, as a result the performance is improved which in turn may be expected to provide increased profits.
- This feasibility checks whether the system can be developed with the available funds.

Operational Feasibility

It is mainly related to human organizations and political aspects.

The points to be considered are:

- What changes will be brought with the system?
- What organization structures are disturbed?
- What new skills will be required?

- Do the existing staff members have these skills? If not, can they be trained in due course of time?

  The system is operationally feasible as it very easy for the users to operate it.

Schedule feasibility

Time evaluation is the most important consideration in the development of project. The time schedule required for the developed of this project is very important since more development time effect machine time, cost and cause delay in the development of other systems.

5.6 Summary

To conclude, we discussed the scheduling processes of developing this system. Additionally we have also identified how feasible the system is through the lens of evaluating using various feasibility studies.

Chapter 6

Future Work

6.1 Introduction

This chapter discusses the future scope or the implementation of this robot. To incerease the scope of this device we can add some new features. As technology is becoming more advance it will be mendatory to change the sturctute some day with better replacement and sometimes based on customer requirements.

6.2 Future Scope of Work

There are so many future scope on this project. Some of them are

- Can improve security
- Can use Neural Network for high accuracy • Can used in big factory or employee attendance
- Can build on fully web base system.

6.3 Summary

This chapter has described the possible future applications of the design. But there are a lot of possibilities with the designed device. The device may need some research for different applications, though the principle of the designed system will remain as it is.

Chapter 7

Result

7.1 Introduction

This chapter of the report contains the results that we achieved throughout the course of using this system.

Results Achieved

From initiation through conclusion of developing this system the following results has been achieved. They are as follows:

- The system can be administered by a non-IT technician.
- The system is market ready for commercial use.
- The system has the capacity to carry up to a thousand faces to recognize.
- The system can serve as much people as they want within an organization.

7.2 Summary

This chapter has covered the different types of results that we have managed to obtain throughout the course of using this system.