CHAPTER-1

INTRODUCTION

Maintenance of attendance is incredibly necessary altogether at the institutes for checking the performance of employees/students. Some institutes take attending manually using paper or file-based approaches and a few have adopted ways of automatic attendance using some biometric techniques. The recent methodology for taking attendance is by calling out the name or roll number of the scholar to record attendance. It's a long and less efficient method of marking attending as a result of as all know the info written within the paper typically may be lost or is less accurate as a result of students ofresultten mark every other's attending proxy. Therefore, to unravel these issues and avoid errors, to recommend computerizing the method by providing a system that records and manages student's attending mechanically with no need for lecturers' interference.

Every biometric system consists of the enrollment process during which distinctive features of someone are stored within the database and then there are processes of identification and verification. These 2 processes compare the biometric feature of someone with previously stored biometric details captured at the time of enrollment. Biometric templates are of many varieties like Fingerprints, Eye Iris, Face, Signature, and Voice.

The system uses the face recognition approach for the automated

Attendance Management System.

By considering the truth of the system is going to be quicker and correct in marking the attendance of individual students. Face recognition consists of two steps, in the first step faces are detected in the image and then these detected faces are compared with the database for verification. Face detection is employed to find the position of face region and face recognition is employed for marking the attendance. The info can store the faces of scholars. Once the face of the scholar matches with one in each of the faces kept within the database then the attendance is recorded.

Various sturdy algorithms are developed that offer accurate performance to tackle face detection and recognition problems. These algorithms or methods are the most successfully and widely used for face detection and recognition applications.

Principal Component Analysis (PCA)

PCA is a dimensionality reduction technique that employs Eigenvalues and EigenVectors to reduce dimensionality and project a training sample/data on small feature space.

Linear Discriminant Analysis (LDA)

The methodology uses concepts of class. The goal is to perform dimensionality reduction while saving as much of the class discriminatory information as possible. LDA minimizes variation within each class and maximizes class separation.

Skin Color Based Approach

One of the simplest algorithms for detecting skin pixels is to use a skin color algorithm. Each pixel is classified as skin color and non-skin color. The classification is based on its color component, which is modeled by Gaussian probability density.

Artificial neural networks based algorithm

An artificial neural network (ANN) is mostly used as a method for recognition. ANN will be implemented once a face has been detected to identify and recognize who the person is by calculating the weight of the facial information.

CHAPTER-2

SURVEY ON OPENCV

Several algorithms and techniques for face recognition developed within the past few years by researchers. The Opencv has likewise been a functioning exploration subject for analysts till the current date. The examination has not many requests, for example, image processing, machine learning approach, pattern recognition, computer vision, neural network.

The authors described in the paper [1] *(Face Recognition Based Attendance Marking System), K.Senthamil Selvi, P.Chitrakala,* and *A.Antony Jenitha*, have described the concept to find real-time human faces, for that, Principal Component Analysis has been used to recognize the faces detected with a high accuracy rate. The system consists of a camera that captures the pictures of the worker and sends to the image enhancement module. Attendance is maintained on the server thus anyone will access for the functions like administration in the institution. On recognizing, the second attendance database contains data concerning the workers, and conjointly uses to mark attendance. Once opencv is compared to ancient attendance marking, the technique seems to save time and also helps to monitor the scholars.

In paper [2] *Face Recognition Based Student Attendance System with OpenCV* by *CH. Vinod Kumar* and, *Dr. K. Raja Kumar,* OpenCV is suggested that face recognition based (mostly on student attendance systems) with OpenCV takes the number of individuals present within the classroom and

takes attendance of each of them using face detection and recognition algorithms to see the particular identification of persons such that which of them are present. In the proposed system faces are detected and recognized using the Eigen object detector algorithm.

The process is done with the help of OpenCV with haar cascades that are present in the OpenCV inbuilt. For the process, the system associate HD digital camera is needed to require the input pictures in an exceedingly fastened space wherever the camera is located. The photographs that are taken from the camera are detected with a haar cascade. The frontal faces and eyes are then trained with the Eigen algorithm, the trained faces are stored in a database first, and compared to the trained images after comparing makes attendance to the recognized persons.

The objectives of the thesis [3] *Facial Recognition using OpenCV* by *Divyarajsinh N. Parmar* and *Brijesh B. Mehta*, are to supply a group of detection algorithms which will be later prepackaged in an easily-portable framework amongst the various processor architectures that tend to see in machines (computers) nowadays. A way is described in the paper, which supported the way to resize a picture along with keeping its aspect-ratio an equivalent. The algorithm uses a sort of face detector referred to as a Haar Cascade classifier, provided by OpenCV. Given a picture, which might come from a file or live video, the face detector examines every image location and classifies as a "Face" or "Not Face". The drawbacks like the image probably would not recognize them in a bright room problem (referred to as "lumination dependent"), and lots of different problems, like the face, ought to even be in a very consistent position are mentioned during the paper.

In the projected system [4] by *Shervin Emami* and *Valentin Petruț*, common ways like a holistic matching technique, feature extraction technique, and hybrid ways that are used. The paper relies on analysis in face recognition. The paper provides readers with an additional sturdy understanding of face recognition ways & applications. Face recognition systems establish people by their face photos.

Face recognition systems establish the presence of an accredited person rather than merely checking whether or not or not a sound identification or secret's being utilized or key's being employed or whether or not the user is aware of the key personal identification numbers(pins) or passwords. The face recognition system directly compares the face photos of individuals and does not use ID numbers to differentiate one from the others. Once the very best two matched faces are extremely just like the query face image, manual review is needed to create certain they're so different persons therefore on eliminating duplicates.

The paper [5] *Smart Attendance Monitoring System* by *Sudhir Bussa, Shruti Bharuka, Ananya Mani,* and *Sakshi Kaushik* presents the automatic attendance management system for convenience or data reliability. The system is developed by the integration of ubiquitous components to make a portable device for managing the students' attendance using Face Recognition technology. The system achieves high frame rates working only with the information present in a single grey-scale image. These alternative sources of information can also be integrated with the system to achieve even higher frame rates.

In the paper *Smart Attendance System using OPENCV based on Facial Recognition [6] by Sudhir Bussa, Ananya Mani, Shruti Bharuka,* and *Sakshi Kaushik*, the Open CV primarily based face recognition approach has been planned. The model integrates a camera that captures associate degree input image, an algorithmic rule for detecting a face from an input image, encoding and identifying the face, marking the attendance in a spreadsheet, and changing into a PDF file. The training database is formed by training the system with the faces of the approved students. The cropped pictures are then stored as a database with respective labels. The features are extracted using the LBPH algorithmic rule. The expected system engages the face recognition for automating the attendance system of scholars or staff without their involvement.

A digital camera is employed for capturing photographs of scholars and staff. The faces within the captured pictures square measure detected and compared with the photographs within the database and also the attendance is marked.

SMART ATTENDANCE SYSTEM WITH FACE RECOGNITION

The main intention of the project is to unravel the problems encountered within the previous attending system whereas reproducing a fresh innovative sensible system that may offer convenience to the institution. In the project, a sensible device will be developed that is capable of recognizing the identity of every individual and eventually record down the info into a database system. Apart from that, an interface will be developed to supply visual access to the data. The followings are the expected work to be done:

The targeted groups of the attendance monitoring system are the students and staff of an educational institution.

Users should be able to register through their already existing accounts.

The user’s face should be sensed and captured by the camera

The database of the attendance management system can hold an individual’s information.

The detected face’s data should be matched with the database

The facial recognition process can only be done for one person at a time.

CHAPTER-3

SMART ATTENDANCE SYSTEM USING FACE RECOGNITION

3.1 METHODOLOGY

The main intention of the project is to unravel the problems encountered within the previous attending system whereas reproducing a fresh innovative sensible system that may offer convenience to the institution. In the project, a sensible device will be developed that is capable of recognizing the identity of every individual and eventually record down the info into a database system. Apart from that, an interface will be developed to supply visual access to the data. The followings are the expected work to be done:

The targeted groups of the attendance monitoring system are the students and staff of an educational institution.

Users should be able to register through their already existing accounts.

The user's face should be sensed and captured by the camera

The database of the attendance management system can hold an individual's information.

The detected face's data should be matched with the database

The facial recognition process can only be done for one person at a time.

3.2 SOFTWARE ENGINEERING CONCEPTS

Software Engineering has emerged as a discipline just like civil, mechanical, and electrical engineering, wherever **engineering principles such**

as modeling, prototyping, designing, developing, testing, delivery, installation, and maintenance square measure the broad steps plain-woven along to succeed the final engineering objective. Whereas these engineering disciplines square measure a lot of or less a science, based mostly on arithmetic, material sciences, physics, and chemistry, the constant isn't true for software system engineering.

Software system engineering tends to be nearer to science owing to its approach being scientific, systematic, and also the use of standards, protocols tools, and techniques, Further; the concepts encourages the principles of sensible and design. In the view of software system engineering at the core level is a subject area.

3.3 SOFTWARE PROCESS MODEL

The agile project delivery framework is the approach that may be used for the event of the system during the development of the project. DSDM is an agile methodology approach primarily used as a code development method. The event of the project is requiring user involvement to own timely visible results. data gathered from the literature review shows researchers using completely different algorithms in face detection and recognition. However, because the software process model is a current analysis space, the project needs progressive implementation in smaller functionalities that can be placed along at the tip of a whole system. The project objectives laid out in the proposal will solely be achieved with the experience of the user, as functionalities are going to be prioritized so as of importance aboard continuous user involvement. In contrast to alternative approaches (Waterfall) wherever the stages of implantations are clearly outlined, preferred to use the approach which can

adapt simply to changes created throughout the implementation.

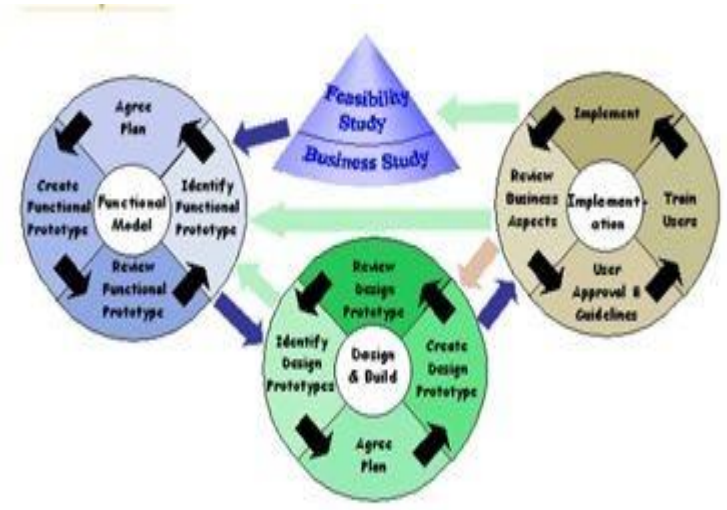


fig:3.1:process model

DSDM DEVELOPMENT PROCESS

3.4 CONCEPTUAL MODEL

The stages in the proposed Smart Attendance System with Face Recognition are as shown in the figure below. The block diagram and also the conceptual model of the attendance system.

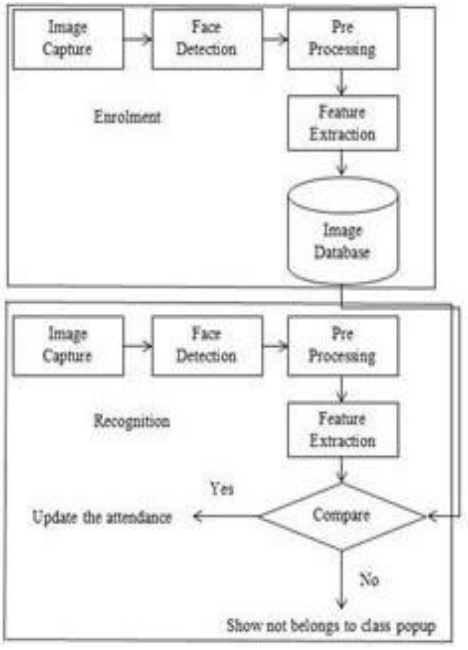


Fig:3.2:conceptual model

3.5 OVERVIEW OF ARCHITECTURE

Smart Attendance System with Face Recognition has the architecture as the following step. The camera needs to install in the front which can capture an entire face of the student inside the class. In the first phase after the camera has been captured; the captured image is transferred into the system as an input. The image capture from the camera sometimes comes with darkness or brightness which needs to do an enhancement on architecture such as convert to a gray image.

Face detection performs locating and extracting face image operations for face recognition systems using the 68-points shape predictor. The model design of the system will form a blueprint for the implementation that will be put together to achieve the projects and best performance for the final product.

The system design consists of activities that fit between software requirements analysis and software construction.

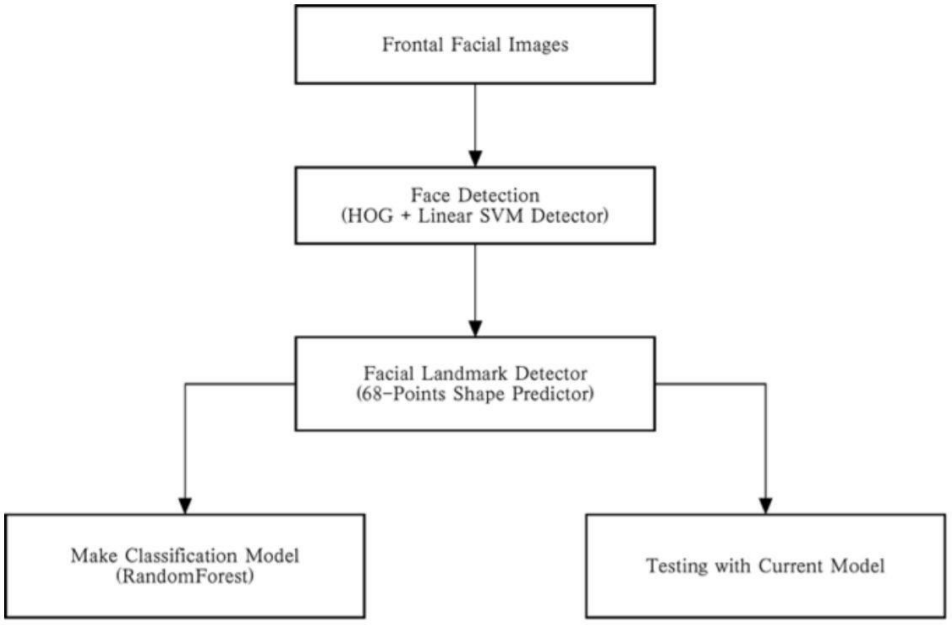


Fig:3.3:The Framework of basic Face Detector

3.6 USER CASE DIAGRAM

The use case diagram in the image represents a system designed for face recognition and attendance calculation, likely within an application intended for automatic attendance tracking. Here's a breakdown of the main use cases and components involved:

- Upload Image - The use case allows users to upload an image to the system, which will be used for face detection and recognition.
- Capture Image - In addition to uploading, there is an option for capturing an image, possibly using a webcam or camera integrated into the system.
- Automatic Detection - The use case involves automatically identifying faces in an uploaded or captured image. User case diagram likely includes processes for detecting and localizing faces in the image.

- Calculate Attendance - Once faces are detected, the system calculates attendance by matching detected faces with a database of registered individuals.
- Store Image - After uploading or capturing, the system might store the image for future reference or analysis.
- Feature Extraction - The use case represents the process of extracting key facial features for accurate recognition and comparison.
- Create Database - Users can add individuals to a database, creating a collection of known faces that the system can refer to during recognition.
- Load Database - The involvement of retrieving the database of known individuals to match with detected faces.
- Train Database - The step could include a training process for the system to better recognize faces over time by learning features of new individuals added to the database.
- Perform Recognition - The use case handles the actual recognition task, comparing detected faces with the database to identify individuals.
- Calculate Recognition Rate - The system may calculate the success or accuracy rate of recognition, possibl`y for performance evaluation.
- Recognize from Webcam - The use case allows for real-time face recognition directly from a webcam feed.
- Exit - The use case provides a way to exit or close the system.

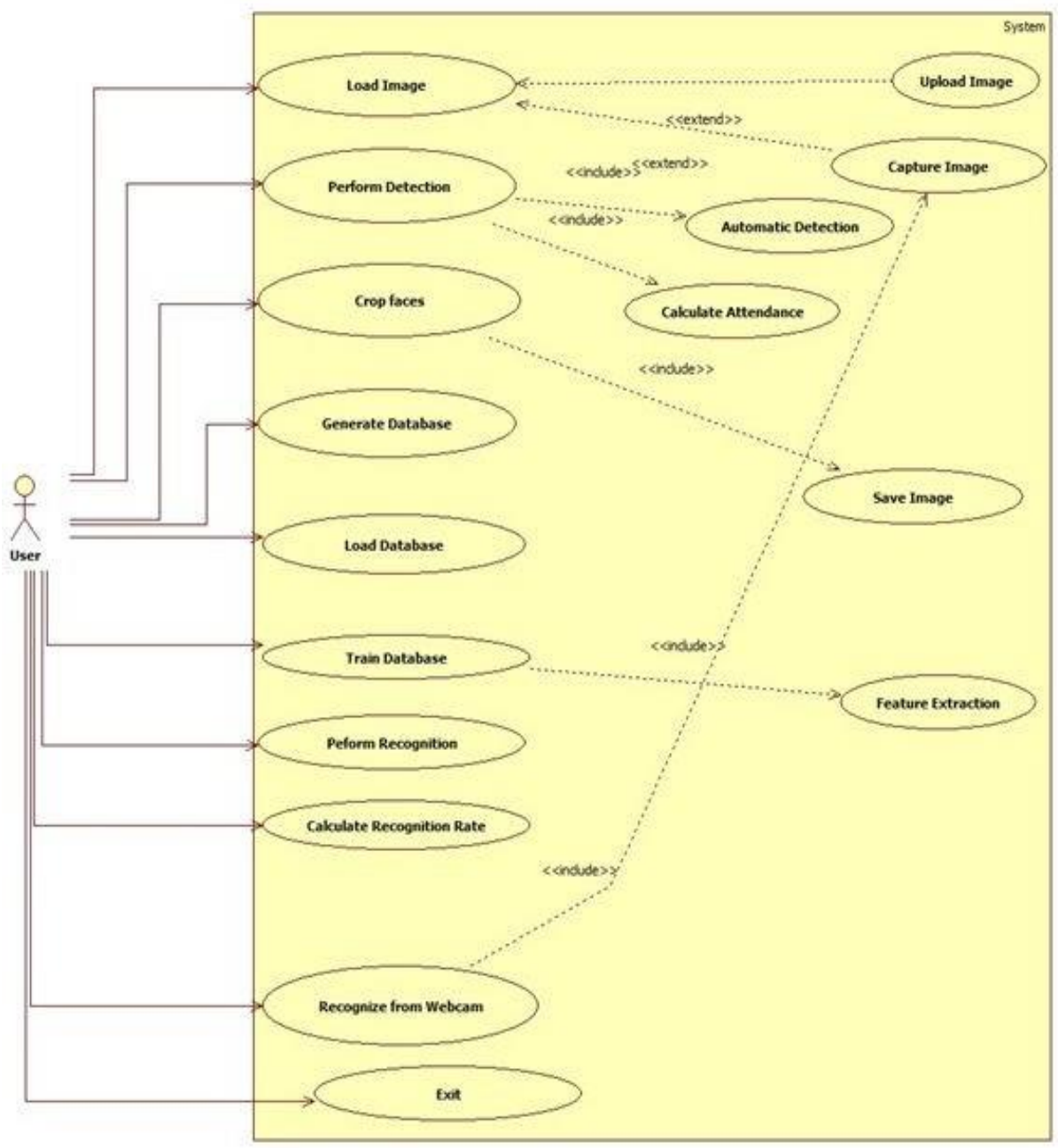


Fig:3.4:Use case diagram

3.7 THE USER STORY

As a user, the client wants a system where they can load an image that will automatically detect the number of faces on the image. The system should have the option to capture an image using an inbuilt webcam on a laptop. As a user,

the system should be able to crop the faces on an image after detection and store them on a folder/dataset that will be used for recognition purposes in the second phase of the system. The system should be able to automatically count the number of faces detected on the image.

As a user, the client requests the second phase of the system to be able to match faces stored on a dataset against input images which are either detected from the first phase or captured by an input device (camera). When the face is detected while taking attendance, the user story is recognized and details of name and time of attendance taken are entered in the database.

The system should be able to integrate with the school’s existing database,including student rosters and course schedules.The system should allow the teacher to create or import class schedules for attendance tracking.

The system should should support the automatic attendance recording.Through facial recognition, RFID cards , or biometric methods(fingerprint,etc.,).

Users should be able to check in at the start of the class by simply standing infront of the scanner or camera.

CHAPTER-4

SMART ATTENDANCE SYSTEM USING MACHINE LEARNING WITH OPENCV

The software technologies used in the mini-project include:

Python is the artificial language for the project, and additionally one in every of the extremely powerful and renowned programming languages better-known for its large use in machine learning and computing.

Visual Studio code editor, because the editor is incredibly reliable with loads of options enclosed in it. It's a robust tool for developers as virtually each programming language may be executed in it.

- Spyder
- Numpy - could be a library for Python, adding support for multi-dimensional arrays and matrices, in conjunction with an enormous assortment of high-level mathematical functions to operate on these arrays.
- Pandas - is a fast, powerful, flexible, and easy to use open-source data analysis and manipulation tool, built on top of the Python programming language.
- Haar Cascade - is a machine learning object detection algorithm used to identify objects in an image or video and based on the concept of features proposed by Paul Viola and Michael Jones in

their paper "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001.

- Datetime - It’s a combination of date and time along with the attributes year, month, day, hour, minute, second, microsecond, and info.
- Face_Recognition - Recognize and manipulate faces from Python or the command line with the world’s simplest face recognition library.
- OpenCV - a library of programming functions primarily geared toward real-time computer vision.

4.1 LOCAL BINARY PATTERNS HISTOGRAM ALGORITHM

Local Binary Pattern (LBP) can be a clear yet productive surface that puts marks on parts of the picture by impeding the area, all things considered, and seeing the outcome as a paired width. The algorithm was resolved that when LBP is joined with histograms of characterized angles (HOG) directed, which improves the securing execution of more informational indexes. Utilizing LBP joined with histograms represent to confront pictures with a vector of direct information.

4.1.1.PARAMETERS:LBPH USES 4 PARAMETERS:

Radius: the radius is used to create a local binary pattern and represents the radius around the center pixel. The frequency is set to 1.

Neighbors: the number of sample points to form a circular area for a binary. Keep in mind: the more points you enter, the higher the computer cost. The frequency is set to 8.

Grid X: the number of cells in a horizontal direction. The more cells, the better grid, the vector size of the emerging element. The frequency is set to 8.

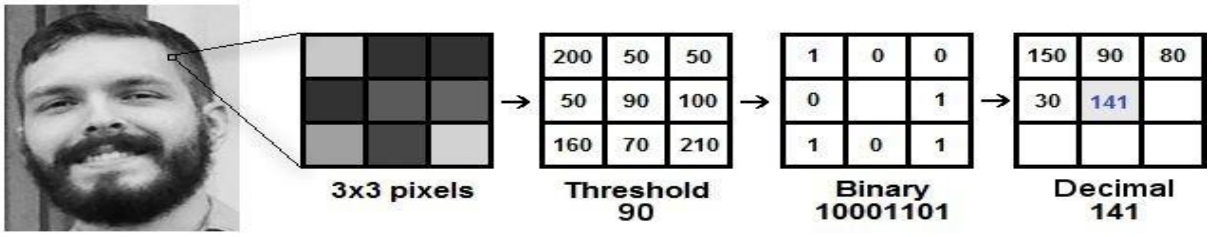
Grid Y - the number of cells in a straight line. The more cells, the better grid, the vector size of the emerging element. The frequency is set to 8.

4.1.2.ALGORITHM TRAINING

The first step is to train the model. To do so, use a database that contains photos of people would like to inform. Next, an ID (also a number or personal name) for all images is added, so the algorithm can use the data to identify the input image and give the result. photos of the same person must have the same ID. Since the training set has already been created, let’s take a look at the LBPH process steps.

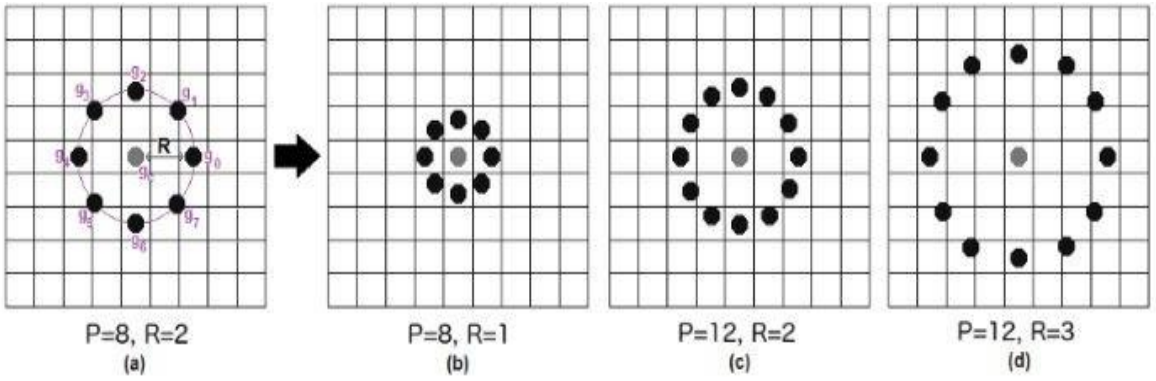
4.1.3.APPLYING FOR LBP OPERATION

Next thing is to create an intermediate image that defines the original image on a highway, with the light of the face. To do so, algorithmic law uses a window view, supporting the range of frames and neighbors.



The above fig 4.1 shows the LBP Operations

- A gray face picture.
- To find part of the image as a 3x3 pixel window.
- LBP can also be represented as a 3x3 matrix containing the thickness of each pixel (0 ~ 255).
- After that, need to take the median value of the matrix that will be used as the limit.
- The number will be used to describe new values from eight neighbors.
- For each median of the middle value (limit), set a new binary value. Set 1 value equal to or higher than the limit and 0 value less than the limit.
- Now, the matrix will contain only binary values (regardless of average value). To estimate each binary value from each location from the matrix line by line to the new binary value (e.g. 10001101). Note: some authors use other methods to synchronize binary values (e.g. clock direction), but the result will be the same.
- After that, set 1 convert the binary number to a decimal value and set to the center value of the matrix, which is a pixel from the first image.
- At the end of the process (LBP process), have a new image that better represents the features of the original image.



The above fig 4.2 shows the process of LBP

The process can be done by using bilinear interpolation. If a certain point of data is within a pixel, the image uses values from the nearest 4 (2x2) pixels to measure the point of the new data point.

4.1.4.PERFORMING FACE RECOGNITION

At the point, the algorithm is already trained. Each created histogram is used to represent each image from the training database. So, when that are given an image to insert, to perform steps again for the new image and create a histogram representing the image.

So to get an image similar to an input image that just need to compare the two histograms and replace the image with the nearest histogram.

The model can use the various methods to compare histograms (calculating the distance between two histograms), for example, Euclidean distance, square-chi, total value, etc.

So the output of the calculation is an ID from a picture with a close-by histogram. The calculation ought to likewise restore a determined reach, which

can be utilized as a proportion of 'certainty'. Note: don't be tricked by the word 'certainty', since low camouflage is better since the process implies that the separation between the two histograms is nearer.

Then use the threshold limit and 'confidence' to automatically adjust if the algorithm has detected the image correctly. Can assume that the algorithm has detected success when confidence is below the defined limit.

4.5.WORKING

In the model, that will be providing the users the options to choose a given function from the above possible functions, the pseudocode for the same is given below:

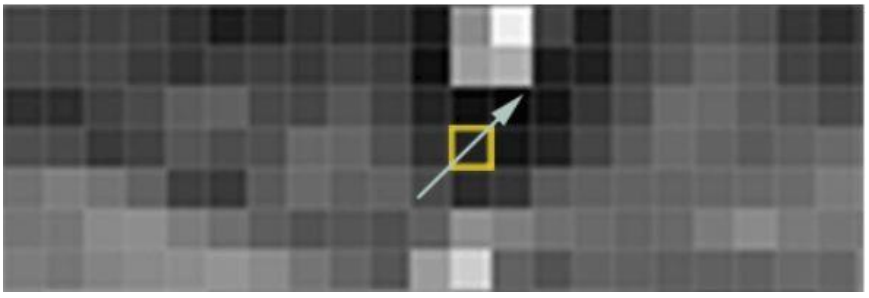
a. Check Camera
b. Capture Faces
c. Train Images
d. Recognize & Attendance

STEP-1: FINDING ALL THE FACES

To start by looking out for faces in an image, tend to start with making the image black and white as a result of color data isn’t required to go looking out for faces. Then investigate every single pixel in the image one at a time. For every single pixel, that might wish to appear at the pixels that directly encompass it. The goal is to work out however dark the component is compared to the pixels directly encompassing it.

Then like to draw the arrow showing during which direction the image is

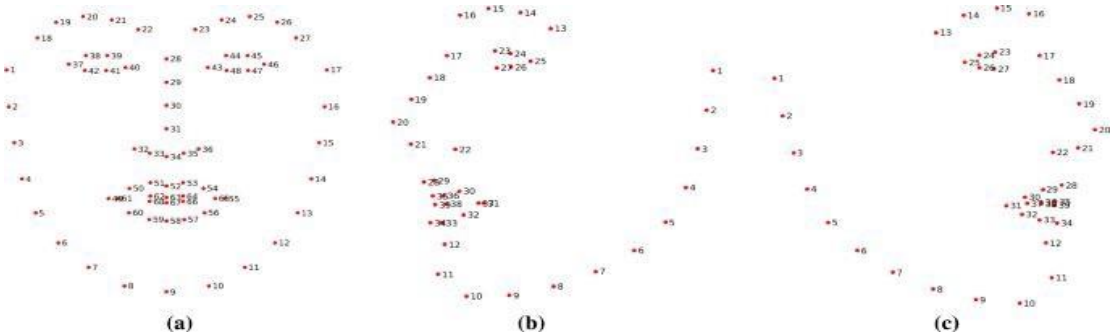
obtaining darker. If you repeat that method for every single component within the image, you finish up with each component being replaced by an arrow. These arrows are referred to as gradients and that they show the flowing light to dark across the whole image:



The above 4.3 is the pixel during the face recognition

STEP-2: POSING AND PROJECTING FACES

For the process, that have planned to use an algorithm known as face landmark estimation. The essential plan is to return up with sixty-eight specific points (landmarks) — the upper part of the chin, the skin fringe of every eye, the inner fringe of the eyebrow, etc. Then train a machine-learning model to be able to realize these sixty-eight specific points on any face:



The above 4.4 represents the Face positons and projections

Given below is the python code for recognizing the 68-points in a person's face using the face-recognition library.

import face_recognition

image = face_recognition.load_image_file("your_file.jpg") face_locations = face_recognition.face_locations(image)

Thus can locate the eyes and mouth, that can also rotate, scale, and shear the image which results in proper centering of the eyes and mouth to its best.

Therfore,face recognition does not depend on the rotation of the face, that will be able to center the eyes and mouth in roughly the same position in the image.

STEP-3:ENCODING FACES

In encoding faces, the face image would be encoded. The plan to find the most reliable way to measure a face. Tend to conjointly train the model during encoding of the method. If the system output provides over one rectangle, that indicates the position of the face, the space of center points of those rectangles has been calculated. If the distance is smaller than a pre-set threshold, the mean of those rectangles is going to be computed and set as the final position of the detected face.

```
# ----------- train images function ---------------
def TrainImages():
    recognizer = cv2.face_LBPHFaceRecognizer.create()
    harcascadePath = "haarcascade_frontalface_default.xml"
    detector = cv2.CascadeClassifier(harcascadePath)
    faces, Id = getImagesAndLabels("TrainingImage")
    Thread(target = recognizer.train(faces, np.array(Id))).start()
    # Below line is optional for a visual counter effect
    Thread(target = counter_img("TrainingImage")).start()
    recognizer.save("TrainingImageLabel"+os.sep+"Trainner.yml")
    print("All Images")
```

The above fig 4.5 is to Train the images in python

The machine learning-based approach in which a cascade function is trained from a lot of positive and negative images. OpenCV comes with a trainer as well as a detector. If there is a need to train the classifier for any object like a car, planes, etc. you can use OpenCV to create one.

STEP-4:FINDING THE PERSON'S NAME FROM ENCODING

The last step is going to be the easiest in the whole process. All wanted to do is identify the person in the database who has the closest measurements to the test image.

In other words, the steps involved in face detection for each of the faces found in an image are as
follows:

1. Find face in an image
2. Analyze facial features
3. Compare against known faces
4. Make a prediction

STEP-5:RECORDING ATTENDANCE

As suggested above, the face is detected by following the mentioned steps and in the final step the face is recognized and the attendance is marked for the recognized person, along with a timestamp. The system can be evaluated bypassing the face recognition function. The face recognition function returns the index of the recognized face on the database.

CHAPTER-5

SOFTWARE REQUIREMENT SPECIFICATION

5.1 HARDWARE REQUIREMENTS

- Computer : Supports Python

- Web-camera : High Definition
- RAM : 16 GB RAM

5.2 SOFTWARE REQUIREMENTS

Platform :

- Libraries : Open CV,NumPy,pandas
- Tools : Visual Studio Code or Spyder
- Database : Excel-Sheet
- Coding Language : Python

5.3 REQUIREMENTS ELICITATION

The requirements are the descriptions of the system services and constraints. In the requirement, elicitation process requirements are collected for software from customers, users, and stakeholders. In this paper, we discussed the concept of needs elicitation and process elicitation in software engineering. Needs elicitation are known through situational characteristics and sources of information. In situational characteristics, a service provider must know the types of stakeholders (domain expert, not domain expert, homogeneous, heterogeneous), the domain of the system being developed (existing system or new system), the scope of the system, know the environment, followed approach, know the problem, etc. In sources of information, a service provider knows competitors, technical literature, expert advice, surveys, etc. All this must prepare you for the elicitation process.

5.3.1 FUNCTIONAL REQUIREMENTS

- Capture face images via webcam or external USB camera

- Faces on an image must be detected.
- The faces must be detected in bounding boxes.
- Resize the cropped faces to match faces the size required for recognition
- Store the cropped faces to a folder
- Load faces on database
- Train faces for recognition
- Perform recognition for faces stored on the database.
- Display the name of the output image above the image in the plot area

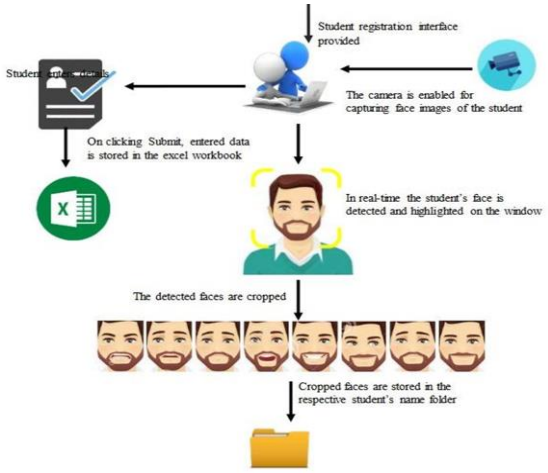


Fig:5.1: Process Of Image Processing

5.3.2:NON-FUNCTIONAL REQUIREMENTS

- **Accuracy and Precision:** the system should perform its process in accuracy and Precision to avoid problems.

- **Modifiability:** the system should be easy to modify, any wrong should be correct.
- **Security:** the system should be secure and save student's privacy.
- **Usability:** the system should be easy to deal with and simple to understand.
- **Maintainability:** the maintenance group should be able to fix any problem that occurs suddenly.
- **Speed and Responsiveness:** Execution of operations should be fast

5.4 SPECIFIC REQUIREMENTS

FACE RECOGNITION

- This module will have a GUI interface used to train the system by taking pictures of each student and assigning names.
- It will show an image box and a button to start the camera
- The main menu has a present button to mark the attendance of the detected face in the camera.
- The main menu will also have delete, update, and close buttons to alter the information.

UPDATE ATTENDANCE

- When a student's image is re-entered while taking attendance the time of attendance is updated.
- It will also show a picture box and an update button.

DISPLAY RECORDS

- This module will be used to display the information of students including their attendance status.
- It will also have a search bar to search for students on the list.

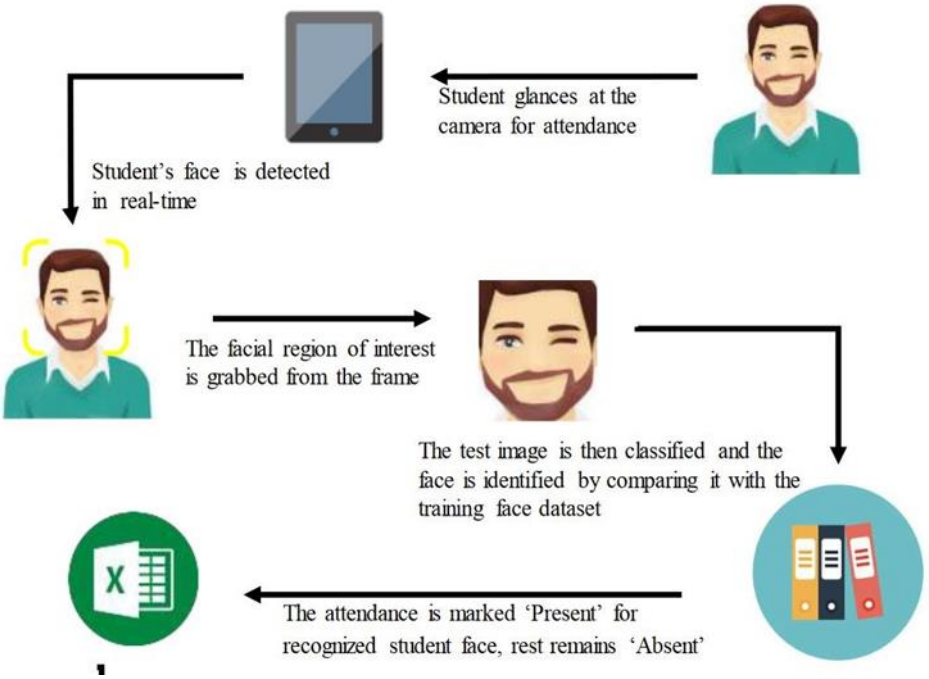


Fig:5.2:Face Detection Process And Store The Proocess

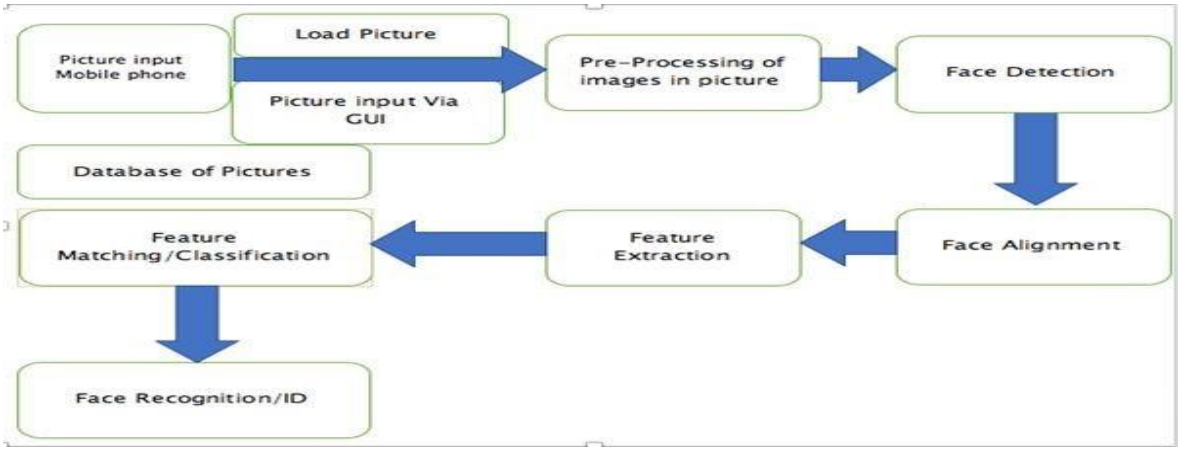


Fig:5.3: Flow Diagram For Recognition

CHAPTER-6

IMPLEMENTATION AND RESULT

6.1 INTRODUCTION

All the code is written in Python language. Below are the important python codes which are the core of the implementation of the software engineering project.

a.Dataset: Where all the face images are saved.

b.main.py: Main program file to run the program.

c.train_image.py: Train the captured images and work on datasets.

d.recognize.py: To recognize and mark attendance

e.automail.py: To send mail to a specific mail id.

Creating a smart attendance system:

6.2 PREPROCESSING DETAILS

Python Libraries

- OpenCV: for capturing and processing images.
- face_recognition: for recognizing faces.
- NumPy: for handling array data.
- pandas: for managing attendance records.

Install these libraries using:

pip install opencv-python face-recognition numpy pandas

Dataset

- Collect images of each person who will be recognized by the system.
- Store each image in a folder with the person's name.

Steps to Develop the System

Step 1: Import Libraries

```
import cv2
import numpy as np
```

import face_recognition
import pandas as pd
from datetime import datetime

Step 2: Load and Encode Images

- Store images in a folder and load them into your script.
- Encode each image so the system can be compared later.

```
import os
# Folder containing images of people
image_path = 'Images'
images = []
class_names = []

# Load images and names
for img_name in os.listdir(image_path):
    img = cv2.imread(f'{image_path}/{img_name}')
    images.append(img)
    class_names.append(os.path.splitext(img_name)[0])

# Encode images
def encode_images(images):
    encodings = []
    for img in images:
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        encode = face_recognition.face_encodings(img)[0]
        encodings.append(encode)
    return encodings
```

```
known_encodings = encode_images(images)
```

Step 3: Initialize Attendance CSV

- Create a CSV file to store attendance records.

```
def mark_attendance(name):
    with open('attendance.csv', 'r+') as f:
    data = f.readlines()
    recorded_names = [line.split(',')[0] for line in data]

# Check if person already marked
    if name not in recorded_names:
        now = datetime.now()
        time_str = now.strftime('%H:%M:%S')
        date_str = now.strftime('%Y-%m-%d')
        f.write(f'{name},{time_str},{date_str}\n')
```

Step 4: Capture Video and Recognize Faces

- Capture live video and detect faces.
- Compare detected faces with known encodings to mark attendance.

```
cap = cv2.VideoCapture(0)

while True:
    ret, frame = cap.read()
    small_frame = cv2.resize(frame, (0, 0), fx=0.25, fy=0.25)
```

```
        rgb_frame = cv2.cvtColor(small_frame, cv2.COLOR_BGR2RGB)

# Detect face locations and encodings
        face_locations = face_recognition.face_locations(rgb_frame)
        face_encodings = face_recognition.face_encodings(rgb_frame,
                                face_locations)

    for encode_face, face_loc in zip(face_encodings, face_locations):
        matches = face_recognition.compare_faces(known_encodings,
                        encode_face)
        face_distance = face_recognition.face_distance(known_encodings,
                                encode_face)
         if True in matches:
                match_index = np.argmin(face_distance)
                name = class_names[match_index].upper()
                mark_attendance(name)



# Display name on screen
          top, right, bottom, left = face_loc
          top, right, bottom, left = top * 4, right * 4, bottom * 4, left * 4
          cv2.rectangle(frame, (left, top), (right, bottom), (0, 255, 0), 2)
          cv2.putText(frame,name, (left + 6, bottom – 6),cv2.FONT_HERSHEY
          SIMPLEX,1, (0, 255, 0), 2)

         cv2.imshow('Attendance System', frame)
         if cv2.waitKey(1) & 0xFF == ord('q'):
```

```
            break
    cap.release()
    cv2.destroyAllWindows()
```

Explanation of Code

1.Encoding Images: Each image is converted into an encoding, a unique identifier that helps the system recognize the face.
2.Marking Attendance: Once a face is matched, the system logs the name, date, and time into attendance.csv.
3.Live Recognition: The system continuously captures frames, detects faces, and checks for matches against known encodings.

CSV File Structure

Your attendance.csv will look like:

```
Name,Time,Date
Arshad Mohamed,10:15:30,2024-11-11
Dinesh Kumar,10:16:05,2024-11-11
```

Additional Improvements

1.GUI Integration: Use tkinter or other libraries to create a user interface.
2.Database Storage: Store attendance data in a SQL database for easier querying and management.
3.Performance Optimization: Use threading to handle camera input separately, improving frame rates.

The setup will give you a functional attendance system using face recognition. Let me know if you'd like help with specific enhancements or troubleshooting!

6.3 SOURCE CODE

```
import cv2
import numpy as np
import face_recognition
import os
from datetime import datetime

# from PIL import ImageGrab

path = 'Training_images'
images = []
classNames = []
myList = os.listdir(path)
print(myList)
for cl in myList:
    curImg = cv2.imread(f'{path}/{cl}')
    images.append(curImg)
    classNames.append(os.path.splitext(cl)[0])
print(classNames)


def findEncodings(images):
    encodeList = []
```

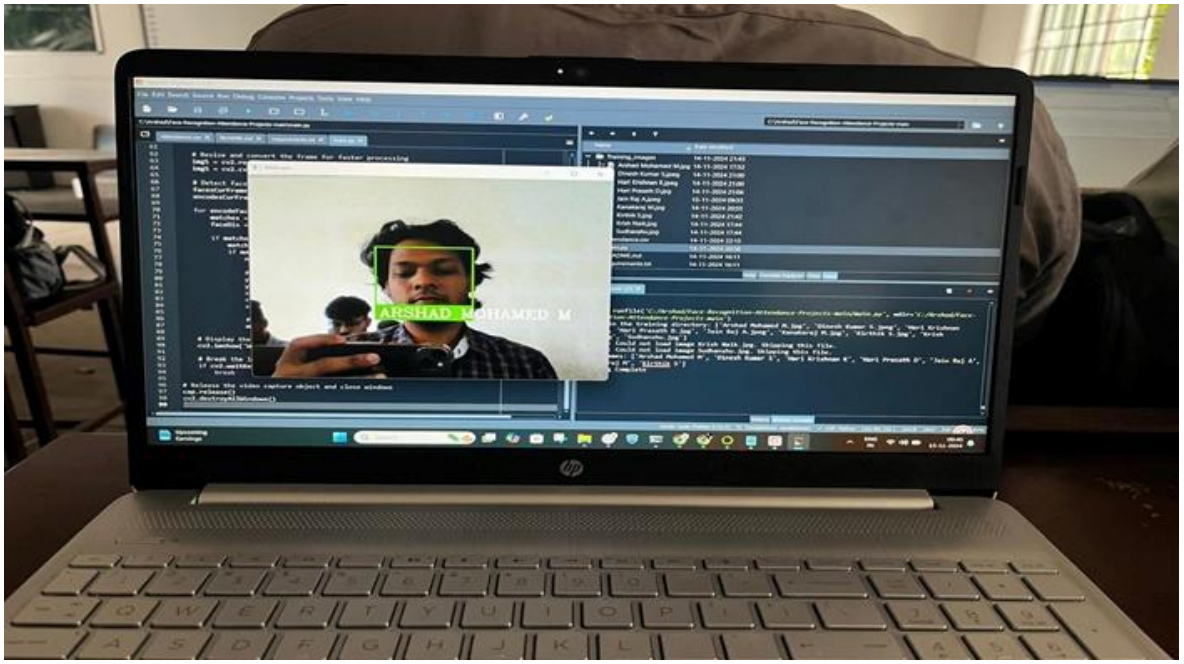
```
    for img in images:
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        encode = face_recognition.face_encodings(img)[0]
        encodeList.append(encode)
    return encodeList


def markAttendance(name):
    with open('Attendance.csv', 'r+') as f:
        myDataList = f.readlines()


        nameList = []
        for line in myDataList:
            entry = line.split(',')
            nameList.append(entry[0])
            if name not in nameList:
                now = datetime.now()
                dtString = now.strftime('%H:%M:%S')
                f.writelines(f'\n{name},{dtString}')

#### FOR CAPTURING SCREEN RATHER THAN WEBCAM
# def captureScreen(bbox=(300,300,690+300,530+300)):
#     capScr = np.array(ImageGrab.grab(bbox))
#     capScr = cv2.cvtColor(capScr, cv2.COLOR_RGB2BGR)
#     return capScr

encodeListKnown = findEncodings(images)
print('Encoding Complete')
```

```
cap = cv2.VideoCapture(0)

while True:
    success, img = cap.read()
# img = captureScreen()
    imgS = cv2.resize(img, (0, 0), None, 0.25, 0.25)
    imgS = cv2.cvtColor(imgS, cv2.COLOR_BGR2RGB)

    facesCurFrame = face_recognition.face_locations(imgS)
    encodesCurFrame=face_recognition.face_encodings(imgS, facesCurFrame)


    for encodeFace, faceLoc in zip(encodesCurFrame, facesCurFrame):
        matches=face_recognition.compare_faces(encodeListKnown,
encodeFace)
        faceDis= face_recognition.face_distance(encodeListKnown, encodeFace)
# print(faceDis)
        matchIndex = np.argmin(faceDis)

        if matches[matchIndex]:
            name = classNames[matchIndex].upper()
         # print(name)
            y1, x2, y2, x1 = faceLoc
            y1, x2, y2, x1 = y1 * 4, x2 * 4, y2 * 4, x1 * 4
            cv2.rectangle(img, (x1, y1), (x2, y2), (0, 255, 0), 2)
            cv2.rectangle(img, (x1, y2 - 35), (x2, y2), (0, 255, 0), cv2.FILLED)
            cv2.putText(img,name,(x1+6,y2-6),
            cv2.FONT_HERSHEY_COMPLEX, 1, (255, 255, 255), 2)
            markAttendance(name)
```

```
cv2.imshow('Webcam', img)
cv2.waitKey(1)
```

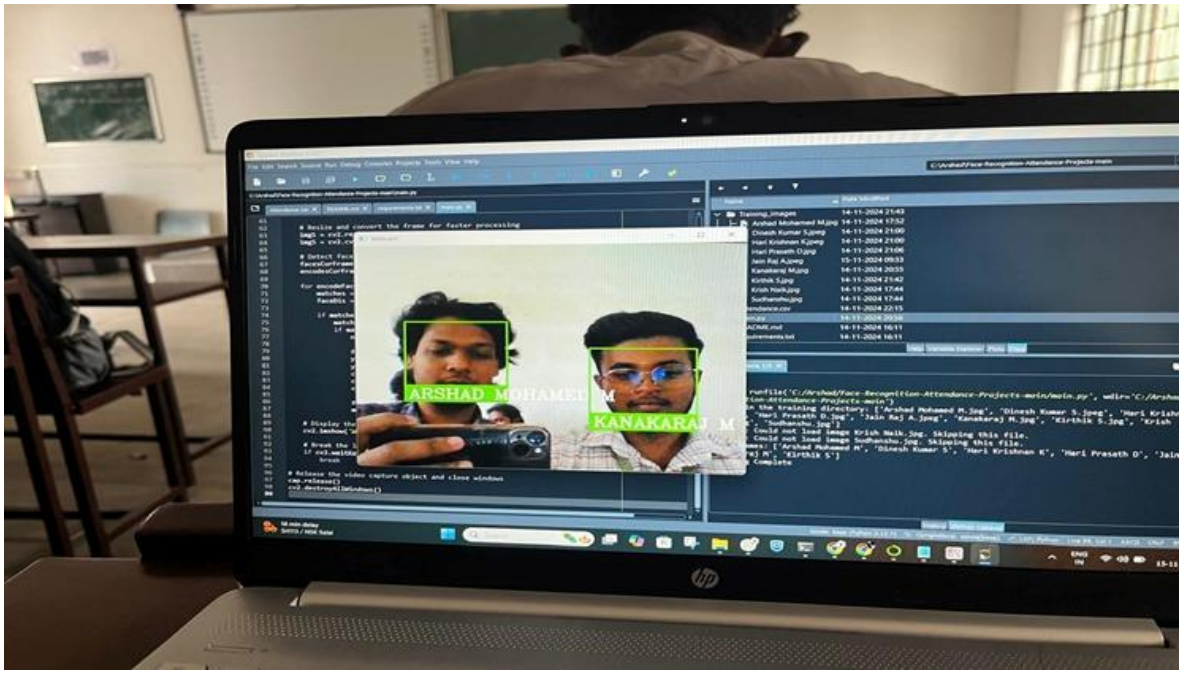
6.4 RESULT

6.4.1 RECOGNIZE FACE



The above fig 6.1 allows the users to recognize the face

6.4.2 RECOGNIZE MULTIPLE FACES

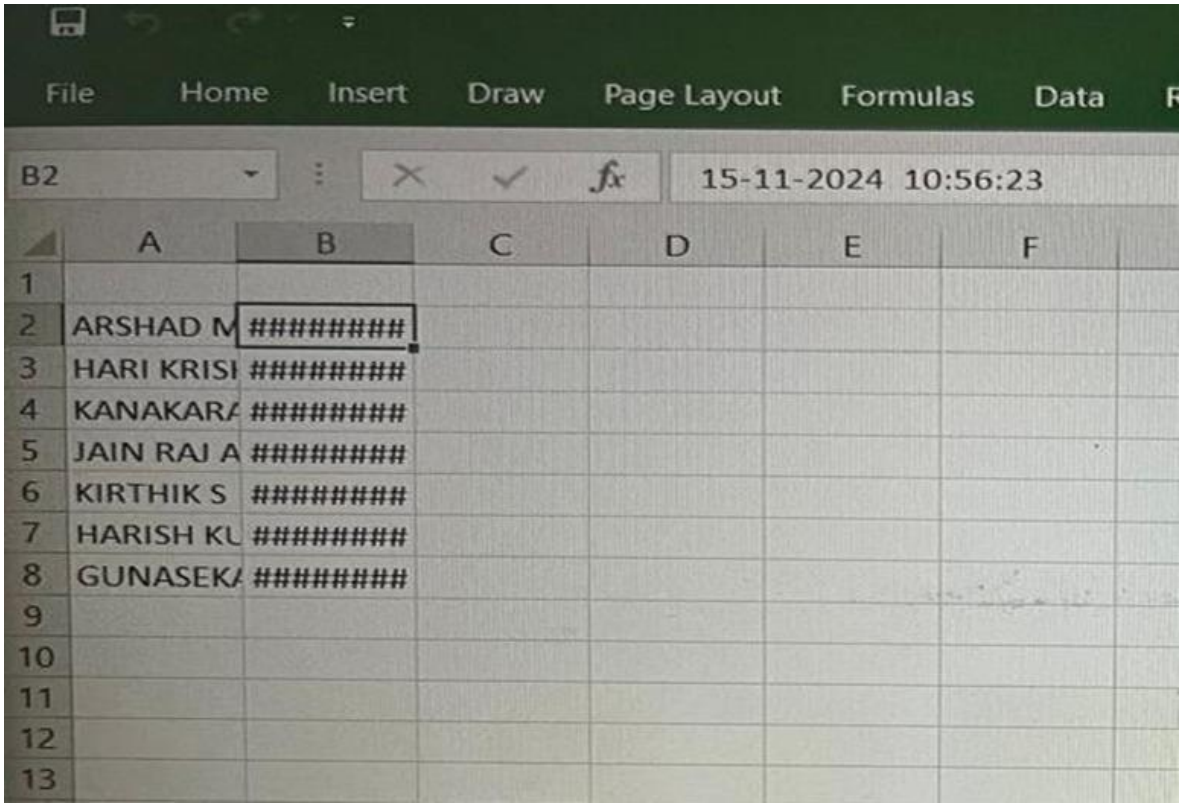


The above fig 6.2 Recognize the multiple Faces

6.4.3 TRAINING WITH MULTIPLE EXPRESSIONS OF OUR FACES

The above fig 6.3 are the Trained Images to Recognize the Faces

6.4.4 DATA STORAGE



The above fig 6.4 shows the Attendance are noted in the Excel-sheet

CHAPTER-7
CONCLUSION AND FUTURE ENHANCEMENT

7.1 CONCLUSION

The system is capable of detecting the faces from the captured image from HD Video to investigate and discover the face. Face detection determines when a picture, a face is found and it's being done by scanning the various image scales and extracting the precise patterns to discover the face. The entire range of pictures saved within the folder can actuate the numeric attendance throughout the class. The images within the folder are going to be matched by the recognition part to faces already held on and trained on the database images. The functions may facilitate confirmation of attendance for a specific student. then conjointly modify the lecturer to require full management of the class by calling out every student by name once it's needed.

Confirmation of attendance for a specific student. Then the system will conjointly modify the lecturer to require full management of the class by calling out every student by name once it's needed.

REQUIREMENT TRACEABILITY MATRIX

A traceability matrix is a document that associates any two-gauge records that require a many-to-numerous relationship to check the fulfillment of the relationship. The Matrix is utilized to follow the prerequisites and to check the current venture necessities are met.

Requirement Traceability Matrix, RTM catches all necessities proposed by the customer client and advancement group client and their detectability in a solitary archive conveyed after the life-cycle.In different words, RTM is a report that guides and follows client necessities with experiments. The fundamental reason for the Requirement Traceability Matrix is to see that all experiments are covered so no usefulness should miss while testing.

The Requirement Traceability Matrix,Analyzes the fundamental records that require a multiple relationships to process the fulfillment of the relationship.The Matrix propsed the solitary life-cycle in the Requirement Traceability Matrix.

Table1: Functional Requirements Tables

Functional Requirement - ID	**Functional Requirements Description**
FR-1	Face Detection
FR-2	Capturing Faces and Training Model
FR-3	Storing images in the database
FR-4	Face Recognition
FR-5	Recording Attendance
FR-6	Emailing Attendance

Table2:Test Cases

Test Case - ID	**Test Case Description**
TC-1	Web camera detecting faces live
TC-2	Capturing multiple images
TC-3	Storing images with student details
TC-4	Encoding faces
TC-5	Training model with training images
TC-6	Recognizing faces registered in the database
TC-7	Marking attendance along with time
TC-8	Emailing the attendance along with details of timestamp

Table3: REQUIREMENTS TRACEABILITY MATRIX

Functional Requirements Document FRD			**TestCase Document**
FR - ID	**FR Use case**	**Priority**	**TC - ID**
FR -1	Face Detection	High	TC-1
FR -2	Capturing Faces and Training Model	High	TC-2 TC-4 TC-5
FR -3	Storing images in the database	High	TC-3
FR -4	Face Recognition	High	TC-6
FR -5	Recording	High	TC-7

	Attendance		
FR -6	Emailing Attendance	Medium	TC-8

Table 4: Detection and Recognition rate concerning face orientation

Face Orientations	Detection Rate	Recognition Rate
O^0 (Frontal face)	98.7 %	95%
18°	80.0 %	78%
54°	59.2 %	58%
72°	0.00 %	0.00%
90°(Profile face)	0.00 %	0.00%

Here performed a set of experiments to demonstrate the efficiency of the proposed method. 50 different images of 5 persons are used in the training set. The evaluation of the face recognition part of the system produced results that were not as expected. The evaluation showed that the recognition part can achieve approximately 70-80% recognition rate based on the image resolution. With a poor image resolution, there is a high chance the system will fail. However, to evaluate the system against images taken in controlled backgrounds, there is a high chance of excellent results.

The random sequence of input images when tested will most likely be the sequence from the output image. Although the recognition does not change the

overall percentage of recognition, detection was not possible to tell the user at what percentage they could decide a face is a face, in other words, detection hardly fails to recognize a face. The only way out was to carry out a test on all five input images and at least with three matches, the user can confirm a face, based on the output match displayed side by side. The performance of the system has impacted the reliability of the system. Because detection is still an ongoing research area, the system will not be available for use at the end of the project. However, the detection can be used for research purposes by the supervisor and experimented with in a lecture room before approval.

The detection and recognition rate of facial recognition systems is significantly influenced by face orientation. Orientation refers to the angle and position of the face relative to the camera, including yaw (left-right turn), pitch (up-down tilt), and roll (side tilt). Here's how face orientation impacts these systems:

Impact on Detection

1. Frontal Orientation: Facial detection systems typically perform best when the face is directly facing the camera (0° yaw, pitch, and roll). The orientation maximizes the visibility of facial features.

2. Yaw (Horizontal Rotation): As the face turns away from the camera (e.g., looking to the side), fewer facial features are visible, reducing the detection rate. Performance usually drops significantly after ±30° yaw.

3. Pitch (Vertical Tilt): Tilting the face upwards or downwards can obscure key facial features such as the eyes or mouth. Detection rates decline as the pitch

angle increases beyond ±20°.

4. Roll (Side Tilt): Rotating the face sideways can distort the alignment of facial features, but systems tend to tolerate moderate roll angles (e.g., ±15°).

5. Extreme Angles: Detection becomes challenging when the face is at extreme angles, such as profile views (±90° yaw) or when obscured by objects.

Impact on Recognition

1. Feature Distortion: As orientation deviates from frontal, the appearance of facial features changes due to perspective and occlusion, which can degrade recognition accuracy.

2. Profile Views: Recognition systems typically struggle with profile views because key features, such as both eyes, are not fully visible.

3. Dataset Bias: Many recognition algorithms are trained on datasets with predominantly frontal or slightly rotated faces, limiting their ability to handle extreme orientations.

4. Alignment Techniques: Modern systems use techniques like 3D face modeling or feature alignment to normalize orientation and improve recognition for non-frontal faces.

5. Occlusion: Extreme orientations often result in partial occlusion of facial

features (e.g., one eye or part of the mouth), further reducing recognition accuracy.

7.2 FUTURE ENHANCEMENT

The entire project has been developed from the requirements to a complete system alongside evaluation and testing. The system developed has achieved its aim and objectives. More careful analysis is required on a project intrinsically. The ways used may be combined with others to attain nice results. Completely different ways are enforced within the past in keeping with the literature review. The conclusion to set the parameters of the particular part of the system based on a very small class size was due to the failures obtained from the recognition part of the system. The size of the image is very important in face recognition as every pixel counts. The algorithm that used would be analyzed with images of different sizes and these images would be showing students in a classroom setting with natural sitting positions showing faces of different sizes.

The basic idea is, no matter how the face is turned, should be able to center the eyes and mouth in roughly the same position in the image, thus recognizing the person and marking their attendance.

Login practicality would be enforced on the system for security functions.

Data confidentiality is incredibly vital. At the beginning of every year, the photographs of the latest people can be taken and kept by the organization.

Each person will have the right to be told regarding the employment of their faces for a face recognition attendance system. The aligning should be in

line with government laws on moral problems and information protection laws and rights. The individuals can consent to their pictures used for the aim of attendance.

In future work, that need to improve face detection effectiveness with the help of the interaction among the system, the scholars, and the faculty. On the other hand, the system is improved by desegregation video-streaming service and lecture archiving system, to give additional profound applications in the field of distance education, course management system (CMS), and support for college development (FD). Improve the same technique thus as tend to run the system with additional than two students on a bench and permitting them to vary their positions.

- Can improve security by adding administrator login.
- Can use Neural Network for high accuracy.
- Can be used in a big factory or employee attendance.
- Can build on a fully web-based system.

Table 5: Work Division

Sr. no.	Task
1.	Installations of Anaconda,Spyder ((python environment) along with OpenCV libraries
2.	Conduct a literature survey to identify the algorithms suitable for developing the model
3.	Developing Design and Architecture of the model,By resource implementation and initial Prototyping

4.	Implementing necessary changes with given feedback to counter underlying problems and tackle necessities
5.	Review and Evaluation of the model with full-scale testing and corrections
6.	System testing to improve the features
7.	Complete documentation of the project with details in the Project Report

REFERENCES

[1] K.Senthamil Selvi et al, “Face Recognition Based Attendance Marking System” in International Journal of Computer Science and Mobile Computing, Vol.3 Issue.2, February- 2014.

[2] CH. Vinod Kumar and Dr. K. Raja Kumar, “Face Recognition Based Student Attendance System with OpenCV” in International Journal of Advanced Technology and Innovative Research Volume. 08, issue.24, December-2016.

[3] Shervin EMAMI and Valentin Petrut SUCIU, “Facial Recognition using OpenCV”, ResearchGate article March 2017.

[4] Divyarajsinh N. Parmar and Brijesh B. Mehta, “Face Recognition Methods & Applications”, an article in International Journal of Computer Applications in Technology · January 2018.

[5] Sudhir Bussa, Shruti Bharuka, Ananya Mani, and Sakshi Kaushik, “Smart Attendance System using OPENCV based on Facial Recognition”, Vol. 9 Issue 03, March-2020.

[6] Centre for Information Technology and Engineering, Manonmaniam Sundaranar University, “Software Engineering – Concepts and Implementations”.

[7] Ali Rehman Shinwari, Asadullah Jalali Balooch, Ala Abdulhakim Alariki, Sami Abduljalil Abdulhak, International Conference on Advanced Communication Technology (ICACT), “A Comparative Study of Face Recognition Algorithms under Facial Expression and Illumination”, 2020 21st.

[8] Bobby R. BruceEmail author Jonathan M. AitkenEmail author Justyna PetkeEmail author,

[9] Sudhir Bussa , Ananya Mani , Shruti Bharuka , Sakshi Kaushik, Department of Electronics and Telecommunication, Bharati Vidyapeeth, University, College of Engineering, Dhankawadi, “Smart Attendance System using OPENCV based on Facial Recognition”, Volume 09, Issue 03, March 2020.

[10] Mamata S.Kalas, Associate Professor, Department Of It, Kit’s College Of Engg., Kolhapur, “Real-time face Detection And Tracking Using Opencv”, 5th & 6th April 2020