FACE DETECTION BASED ATTENDANCE MONITORING SYSTEM

A MINI PROJECT REPORT

18CSC305J - Artificial Intelligence

Submitted by

Sambhav	**[RA2111003011790]**
Yash Rana	**[RA2111003011796]**
Trilok Dhawan	**[RA2111003011805]**

Under the Guidance of

Dr. Lubin Balasubramanian

Assistant Professor, Department of Computing Technologies

in partial fulfillment for the award for the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

of

FACULTY OF ENGINEERING AND TECHNOLOGY

S.R.M. Nagar, Kattankulathur, Chengalpattu District

MAY 2024

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Under Section 3 of UGC Act, 1956)

BONAFIDECERTIFICATE

Certified that Mini project report titled "**FACE DETECTION BASED ATTENDANCE MONITORING SYSTEM**" is the bonafide work of **Sambhav [RA2111003011790], Yash Rana [RA2111003011796], Trilok Dhawan [RA2111003011805]** who carried out the minor project under my supervision. Certified further, that to the best of my knowledge, the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE	***SIGNATURE***
Dr. Lubin Balasubramanian	Dr. M. Pushpalatha
Assistant Professor	Head of The Department
Department of Computing Technologies	Department of Computing Technologies

ABSTRACT

Face detection, a fundamental component in computer vision, has garnered significant attention in recent years due to its wide-ranging applications in various domains such as security, surveillance, human-computer interaction, and biometrics. Traditional methods for face detection relied heavily on handcrafted features and complex algorithms, which often struggled to perform robustly under diverse environmental conditions.

In contrast, the emergence of Artificial Intelligence (AI) techniques, particularly deep learning, has revolutionized the field of face detection. Deep learning models, especially Convolutional Neural Networks (CNNs), have demonstrated remarkable capabilities in automatically learning discriminative features directly from raw pixel data, enabling more accurate and efficient face detection systems.

This paper provides a comprehensive overview of recent advancements in face detection using AI techniques. It covers key concepts, methodologies, and architectures employed in AI-based face detection systems. Additionally, it discusses various challenges and issues associated with face detection, including occlusion, pose variation, illumination changes, and diversity in facial appearances.

Furthermore, the paper explores algorithms and approaches, including Single Shot Multibox Detector (SSD), Faster R-CNN, and RetinaNet, which have shown remarkable performance in detecting faces with high accuracy and speed. It also examines the influence of large-scale datasets, such as WIDER FACE and CelebA, in training robust face detection models.Moreover, the paper discusses emerging trends and future directions in AI-based face detection research, including the integration of multimodal information, such as depth data and thermal imaging, for enhanced performance and robustness. Additionally, it addresses ethical considerations and privacy concerns associated with the widespread deployment of face detection technology.

Overall, this review provides valuable insights into the state-of-the-art techniques, challenges, and future prospects of face detection using AI, paving the way for further advancements in this critical area of computer vision research.

TABLE OF CONTENTS

ABSTRACT **3**

TABLE OF CONTENTS **4**

LIST OF FIGURES **5**

ABBREVIATIONS **6**

1. INTRODUCTION **7**

2. LITERATURE SURVEY **9**

3. PROTOTYPE/APPLICATION DEVELOPED **12**

4. PROBLEM STATEMENT **14**

5. METHODOLOGY **15**

6. SYSTEM ARCHITECTURE AND DESIGN **17**

6.1 Architecture Overview 17

6.2 Architecture Design 18

7. CODING AND TESTING **19**

8. SCREENSHOTS AND RESULT **28**

9. REFERENCES **31**

LIST OF FIGURES

Figure 6.1	**Architecture of the Project**	**18**
Figure 8.1	**Graphical User Interface**	**28**
Figure 8.2	**User Data/Information collection**	**28**
Figure 8.3	**Target User Images Collection**	**29**
Figure 8.4	**User Face Recognition**	**29**
Figure 8.5	**User Details Database**	**30**
Figure 8.6	**Student Attendance Database**	**30**

ABBREVIATIONS

AI	Artificial Intelligence
SSD	Single Shot Multibox Detector
CNN	Convolutional Neural Network
GPT	Generative Pre-trained Transformer
FD	Face Detection
UI	User Interface
IoT	Internet of Things
API	Application Programming Interface
DALL-E	Histogram of Oriented Gradients
PDF	Portable Document Format
RPN	Region Proposal Network

CHAPTER 1
INTRODUCTION

Face Recognition is a popular image processing technology because of its widespread usage. Face recognition may be used to identify people in an organization for attendance purposes. The maintenance and evaluation of attendance records is critical in every organization's performance review. The aim of creating an attendance monitoring system is to automate the conventional method of taking attendance. With less human interaction, the Automated Attendance Management System conducts the everyday tasks of attendance marking and review. When the intensity is greater, the traditional form of attendance marking becomes very time consuming and complicated. Automation of Attendance System has an advantage over conventional methods in that it saves time and can also be used for monitoring. This also aids in the prevention of false participation. Other biometric techniques, such as those mentioned below, can also be used to digitalize the attendance process:

1. Log Book entry

2. Fingerprint based System

3. IRIS Recognition

4. RFID based System

5. Face Recognition

Facial recognition is the most unique, efficient, precise, and cost-effective of all the techniques described above.

Face Detection: The process of locating and identifying human faces within digital images or videos.

- Artificial Intelligence (AI): The simulation of human intelligence processes by machines, especially computer systems, to perform tasks that typically require human intelligence, such as learning, problem-solving, and decision-making.

- Deep Learning: A subset of machine learning techniques inspired by the structure and function of the human brain, particularly neural networks with multiple layers (deep neural

networks). Deep learning algorithms can automatically learn representations of data through multiple levels of abstraction.

- Convolutional Neural Networks (CNNs): A class of deep neural networks, most commonly applied to analyzing visual imagery. CNNs are designed to automatically and adaptively learn spatial hierarchies of features from input images.

- Single Shot Multibox Detector (SSD): An object detection algorithm that simultaneously predicts multiple bounding boxes and their corresponding class probabilities for objects in an image. SSD is known for its speed and accuracy in detecting objects, including faces.

- Faster R-CNN: A state-of-the-art object detection model that improves upon earlier region-based convolutional neural network (R-CNN) architectures by integrating a region proposal network (RPN) to efficiently generate region proposals.

- RetinaNet: Another advanced object detection model known for its one- stage architecture and focal loss mechanism, which addresses the class imbalance problem inherent in object detection tasks.

- WIDER FACE Dataset: A large-scale face detection benchmark dataset, containing images with a wide range of face variations in terms of scale, pose, expression, occlusion, and illumination. It is commonly used for training and evaluating face detection algorithms.

CHAPTER 2
LITERATURE SURVEY

The literature survey encompasses a comprehensive review of existing research and studies relevant to the field of social media content creation, with a focus on leveraging AI and advanced technologies to streamline the process. Below are five notable studies identified from the literature, along with a detailed explanation of each:

- ***"Face Recognition-Based Smart Attendance Monitoring System in Classroom" [1]:*** This paper presents an automated attendance monitoring system that utilizes artificial intelligence to enhance the efficiency and accuracy of the attendance process in educational settings. The system employs the Histogram of Oriented Gradient (HOG) for facial feature detection and the Haar Cascade classifier for recognition. It addresses the challenges of traditional attendance methods, such as proxy attendance and administrative burden, by capturing live video feeds and marking attendance in real-time. The paper also discusses the scalability of the system and the ethical considerations related to privacy and data security. The integration of AI into education, as demonstrated by this system, shows great potential in overcoming longstanding attendance challenges.

- ***"Student Attendance Monitoring System Using Face Recognition" [2]:*** This paper outlines a model that employs the Haarcascade classifier and the Local Binary Pattern Histogram (LBPH) algorithm for face recognition, implemented in Python and OpenCV. It introduces a tkinter GUI interface for user interaction and emphasizes the practicality of real-time face recognition for managing large groups of students' attendance. The proposed model aims to improve face recognition performance and offers a practical solution to the tedious and error-prone conventional method of attendance tracking.

- ***"A Survey on Facial Recognition-Based Attendance Management System" [3]:*** This comprehensive survey paper examines the current state-of-the-art in facial recognition-based attendance management systems. It covers various aspects, including the underlying technology, implementation challenges, benefits, ethical considerations, and future prospects. The paper provides insights into the latest research and developments in this field, offering a well-rounded perspective on the evolving landscape of attendance management systems that employ facial recognition technology.

- ***"Face Recognition-Based Smart Attendance Monitoring and Tracking System" [4]:*** The paper provides an extensive overview of face recognition systems, including their history, methodology, working principle, and applications. It discusses the automatic face recognition system's significance in various fields such as security, surveillance, and identity verification. The main intention of the system is to perform automatic human face recognition for institutions or organizations to have the attendance of their students or employees. The paper also explores the use of the Histogram of Oriented Gradients (HOG) algorithm and its deployment for object detection and face recognition.

Table 2.1 Literature Survey

Author(s)	Title	Dataset	Methods	Remarks
P. Pramod Kumar, R. Akshay, K. Sagar	Face Recognition-Based Smart Attendance Monitoring System in Classroom	Custom dataset of student facial images	Histogram of Oriented Gradient (HOG) for facial feature detection, Haar Cascade classifier for recognition	The system captures live video feeds in real-time, marking student attendance and preventing proxy attendance. It is scalable and could integrate with student management systems, but privacy and data security are concerns
E. Charan Sai, Shaik Althaf Hussain, Syed Khaja, Amara Shyam	Student Attendance Monitoring System Using Face Recognition	Not specified	Haarcascade classifier for face detection, Local Binary Pattern	The model aims to improve face recognition performance and

			Histogram (LBPH) algorithm for face recognition	offers a practical solution to conventional attendance methods
Onkar Akirke, Atharva Patange, Devesh Sonawane, Chinmay Yenugwar, Prof. S.S.Bhong	A Survey on Facial Recognition-Based Attendance Management System	Not applicable (survey)	Review of various facial recognition technologies and their application in attendance systems	This survey covers the technology, implementation challenges, benefits, ethical considerations, and future prospects of facial recognition-based attendance systems

Summary: The papers mentioned above collectively provide a detailed understanding of the advancements in face recognition technologies for attendance monitoring systems. They cover a range of topics from technical details to practical applications and address the challenges and potential improvements in this field. For anyone interested in developing or researching attendance systems, these papers would be an invaluable resource. The use of face recognition technology in attendance systems offers several advantages over traditional methods. It provides a non-intrusive, contactless way to record attendance, which is particularly beneficial in the context of health concerns such as the COVID-19 pandemic. Additionally, it can significantly reduce the time and effort required for attendance tracking, freeing up time for educators to focus on teaching. However, there are also challenges associated with the implementation of such systems. One of the main concerns is the accuracy of face recognition algorithms, especially in varying lighting conditions and with different facial expressions. There is also the issue of privacy and data protection, as biometric data is considered sensitive information.

Future research in this area is likely to focus on improving the accuracy and reliability of face recognition algorithms, developing more robust systems that can handle a wide range of conditions, and addressing privacy concerns through secure data handling practices.

In conclusion, face recognition-based attendance monitoring systems represent a significant advancement in the field of educational technology. They offer the potential to streamline the attendance process, improve accuracy, and reduce administrative burdens. As the technology continues to evolve, it is expected that these systems will become increasingly prevalent in educational institutions around the world.

CHAPTER 3

PROTOTYPE / APPLICATION DEVELOPED

Phase 1 - Workflow & Algorithm Used

- Introduction to Phase 1
- Workflow diagram illustrating the steps involved in face detection
- Description of each step in the workflow
- Overview of algorithms used in each phase, such as preprocessing, feature extraction, and detection
- Highlight the importance of each step in achieving accurate and robust face detection

Phase 2 - Evaluation Metrics & Performance Analysis

- Introduction to Phase 2
- Explanation of evaluation metrics used to assess the performance of the face detection system, such as precision, recall, F1-score, and mean Average Precision (mAP)
- Description of the benchmark datasets used for evaluation, such as WIDER FACE or FDDB
- Presentation of performance analysis results, including quantitative metrics and qualitative assessments

Phase 3 - Results & Discussion (1/3)

- Introduction to Phase 3
- Presentation of experimental results, including detection accuracy, speed, and robustness
- Comparison of the performance of different algorithms and techniques used in the face detection system
- Discussion of key findings and observations from the evaluation phase

Phase 3 - Results & Discussion (2/3)

- Detailed analysis of the strengths and weaknesses of the face detection system
- Identification of potential challenges encountered during the evaluation process
- Discussion of factors influencing the performance of the system, such as dataset characteristics, algorithm parameters, and computational resources

Phase 3 - Results & Discussion (3/3)

- Overview of qualitative results, including visual examples of successful and challenging face detection scenarios
- Interpretation of results in the context of real-world applications and use-case scenarios
- Discussion of implications for future research and development efforts

Phase 4 - Conclusion & Future Enhancements

- Summary of key findings and conclusions drawn from the study
- Recap of the achievements and limitations of the face detection system
- Identification of potential areas for future enhancement and research, such as algorithm optimization, dataset augmentation, and integration with other technologies
- Closing remarks and acknowledgments

CHAPTER 4
PROBLEM STATEMENT

4.1 Problem Definition

It becomes more difficult to mark attendance for each student when there are so many students in an organization and it is a time-consuming one. The Existing system of any institute is manual entry for the students. This system faces the issue of wasting time and it becomes complicated when the strength is more. It is very tedious job to carry out the attendance in log books and to maintain the records. Face recognition is a difficult issue in computer vision. Some of the problems to deal with include lighting issues, posing issues, scale variability, low image capture accuracy, and partially occluded faces are all issues that need to be addressed. As a result, face recognition algorithms must be resistant to changes in the above parameters. Existing techniques don't work well when there's a change in lighting, background, or rotation. As a result, the drawbacks listed above must be addressed. The project's goal is to design and construct a system that is less vulnerable to light, rotates invariantly, scales invariantly, and is robust enough to be used in real-world scenarios.

4.2 Problem Solution

The approach suggested in this project is to use facial recognition technology to monitor attendance. The computer captures camera video streams and senses faces in image format. The identified faces will be linked to the student database, and the attendance will be recorded in an Excel spreadsheet. Using these Excel boards, we will create a graph that displays the average attendance of the whole class/individual student.

CHAPTER 5
METHODOLOGY

Methodology Steps

The methodology comprises the following steps:

1. Requirement Analysis:

- Identify the specific requirements for the content creation system, including platform-specific constraints and user needs.

2. Data Collection:

- Gather a dataset of images or video footage containing faces of individuals who will be enrolled in the attendance system.

- Ensure the dataset includes variations in lighting conditions, facial expressions, poses, and occlusions to improve the robustness of the system.

3. Data Preprocessing:

- Preprocess the collected data to standardize the images, normalize lighting conditions, and remove noise.

- Techniques such as image resizing, histogram equalization, and noise reduction may be applied to enhance the quality of the images.

4. Face Detection Model Selection:

- Choose a suitable face detection model or algorithm based on the requirements of the attendance system.
- Deep learning-based models like SSD, Faster R-CNN, or RetinaNet are commonly used for accurate and robust face detection.

5. Model Training:

- Train the selected face detection model using the preprocessed dataset.

- Utilize transfer learning on pre-trained models or train from scratch depending on the availability and size of the dataset.

6. Integration with Attendance System:

- Integrate the trained face detection model into the attendance system architecture.
- Develop modules for capturing live video streams or images from cameras, processing the input data, and performing face detection in real-time.

7. Attendance Logging:

- Implement mechanisms for logging attendance records based on the detected faces.
- Associate detected faces with individuals enrolled in the system and record their attendance status along with timestamps.

8. User Interface Development:

- Design and develop a user-friendly interface for the attendance system.
- Provide functionalities for enrolling new users, viewing attendance reports, and managing system settings.

9. Testing and Evaluation:

- Test the face detection attendance system in real-world scenarios to evaluate its accuracy, efficiency, and reliability.
- Collect feedback from users and stakeholders to identify any issues or areas for improvement.

10. Deployment and Maintenance:

- Deploy the face detection attendance system in the target environment, ensuring compatibility with hardware and software components.
- Monitor system performance and address any issues through regular maintenance and updates.

CHAPTER 6
SYSTEM ARCHITECTURE AND DESIGN

6.1 Architecture Overview

The architecture of the face recognition-based attendance system follows a sequential process designed to capture and recognize faces for attendance purposes. Here's a detailed explanation of each step:

- **Acquisition of Image:** The system starts by acquiring a new image from the camera. This is the initial step where the camera captures the image of the individual whose attendance needs to be recorded.
- **Color to Grayscale Conversion:** The acquired colored image is converted into grayscale. This simplification step is crucial because grayscale images require less computational power for processing compared to colored images.
- **Face Detection:** A Cascade Classifier is used to detect faces in the grayscale image. This classifier is a machine learning object detection algorithm used to identify objects in an image or video.
- **Eye Detection:** Within the detected face, the system uses another Cascade Classifier to detect the eyes. This step is important for confirming that a face is present and properly oriented in the image.
- **Normalization:** Once the face and eyes are detected, the system normalizes the face images in terms of size and orientation. Normalization ensures that the faces are uniform, which is essential for accurate recognition.
- **Contrast and Lighting Enhancements:** The system then enhances the contrast and lighting of the face images. These enhancements improve the visibility and clarity of facial features, which aids in better recognition.
- **Facial Recognition using PCA Classifier:** The Principal Component Analysis (PCA) classifier is employed for recognizing faces. PCA is a statistical technique used to emphasize variation and bring out strong patterns in a dataset. It compares the pre-processed face images with a collection of pre-stored face samples.
- **Recognition Training & Face Samples Collection:** The recognized faces are added

to a collection, which aids in training and improving the accuracy of the system over time. As more face samples are collected, the system can learn and adapt, increasing its effectiveness in recognizing faces accurately.

This architecture outlines a comprehensive process for automating attendance using face recognition technology, from image capture to processing and recognition. It highlights the importance of preprocessing steps like normalization and enhancements, as well as the use of machine learning algorithms for accurate face detection and recognition. The system's ability to learn and improve over time is a key feature that ensures its long-term reliability and efficiency.

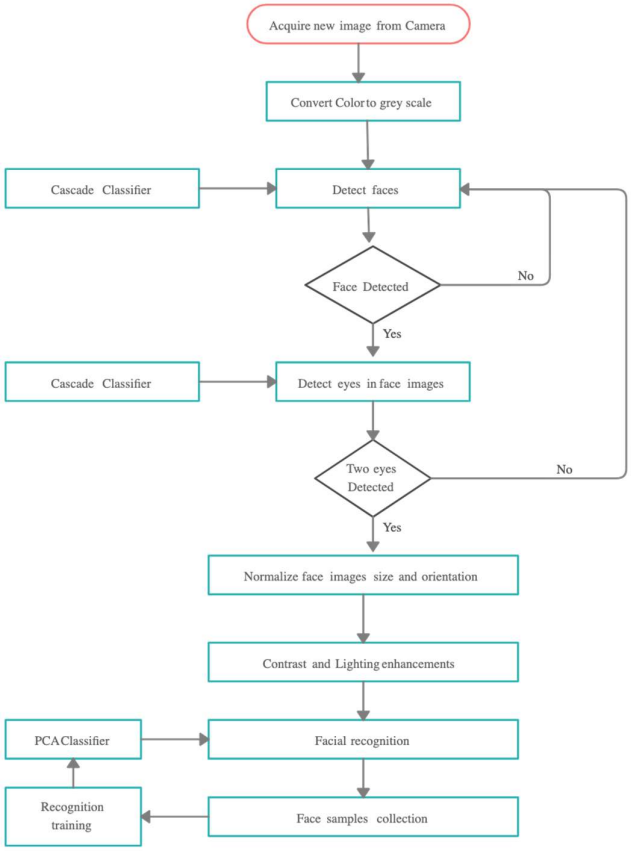


Figure 6.1: Architecture of the Project

CHAPTER 7

CODING AND TESTING

Coding:

```
#IMPORTING
import tkinter as tk
from tkinter import ttk
from tkinter import messagebox as mess
import tkinter.simpledialog as tsd
import cv2,os
import csv
import numpy as np
from PIL import Image
import pandas as pd
import datetime
import time

#FUNCTIONS

def assure_path_exists(path):
    dir = os.path.dirname(path)
    if not os.path.exists(dir):
        os.makedirs(dir)



def tick():
    time_string = time.strftime('%H:%M:%S')
    clock.config(text=time_string)
    clock.after(200,tick)



def contact():
    mess._show(title='Contact us', message="Please contact us on :
'dhawantrilok05@gmail.com' ")



def check_haarcascadefile():
    exists = os.path.isfile("haarcascade_frontalface_default.xml")
    if exists:
        pass
    else:
        mess._show(title='Some file missing', message='Please contact us for help')
        window.destroy()



def save_pass():
    assure_path_exists("TrainingImageLabel/")
    exists1 = os.path.isfile("TrainingImageLabel\psd.txt")
    if exists1:
        tf = open("TrainingImageLabel\psd.txt", "r")
        key = tf.read()
    else:
        master.destroy()
        new_pas = tsd.askstring('Old Password not found', 'Please enter a new
```

```
password below', show='*')
        if new_pas == None:
            mess._show(title='No Password Entered', message='Password not set!!
Please try again')
        else:
            tf = open("TrainingImageLabel\psd.txt", "w")
            tf.write(new_pas)
            mess._show(title='Password Registered', message='New password was
registered successfully!!')
            return
    op = (old.get())
    newp= (new.get())
    nnewp = (nnew.get())
    if (op == key):
        if(newp == nnewp):
            txf = open("TrainingImageLabel\psd.txt", "w")
            txf.write(newp)
        else:
            mess._show(title='Error', message='Confirm new password again!!!')
            return
    else:
        mess._show(title='Wrong Password', message='Please enter correct old
password.')
        return
    mess._show(title='Password Changed', message='Password changed successfully!!')
    master.destroy()



def change_pass():
    global master
    master = tk.Tk()
    master.geometry("400x160")
    master.resizable(False,False)
    master.title("Change Password")
    master.configure(background="white")
    lbl4 = tk.Label(master,text='    Enter Old Password',bg='white',font=('comic',
12, ' bold '))
    lbl4.place(x=10,y=10)
    global old
    old=tk.Entry(master,width=25 ,fg="black",relief='solid',font=('comic', 12, '
bold '),show='*')
    old.place(x=180,y=10)
    lbl5 = tk.Label(master, text='   Enter New Password', bg='white', font=('comic',
12, ' bold '))
    lbl5.place(x=10, y=45)
    global new
    new = tk.Entry(master, width=25, fg="black",relief='solid', font=('comic', 12, '
bold '),show='*')
    new.place(x=180, y=45)
    lbl6 = tk.Label(master, text='Confirm New Password', bg='white', font=('comic',
12, ' bold '))
    lbl6.place(x=10, y=80)
    global nnew
    nnew = tk.Entry(master, width=25, fg="black", relief='solid',font=('comic', 12,
' bold '),show='*')
    nnew.place(x=180, y=80)
    cancel=tk.Button(master,text="Cancel", command=master.destroy ,fg="black"
,bg="red" ,height=1,width=25 , activebackground = "white" ,font=('comic', 10, ' bold
'))
    cancel.place(x=200, y=120)
    save1 = tk.Button(master, text="Save", command=save_pass, fg="black",
```

```
bg="#00fcca", height = 1,width=25, activebackground="white", font=('comic', 10, '
bold '))
    save1.place(x=10, y=120)
    master.mainloop()



def psw():
    assure_path_exists("TrainingImageLabel/")
    exists1 = os.path.isfile("TrainingImageLabel\psd.txt")
    if exists1:
        tf = open("TrainingImageLabel\psd.txt", "r")
        key = tf.read()
    else:
        new_pas = tsd.askstring('Old Password not found', 'Please enter a new
password below', show='*')
        if new_pas == None:
            mess._show(title='No Password Entered', message='Password not set!!
Please try again')
        else:
            tf = open("TrainingImageLabel\psd.txt", "w")
            tf.write(new_pas)
            mess._show(title='Password Registered', message='New password was
registered successfully!!')
            return
    password = tsd.askstring('Password', 'Enter Password', show='*')
    if (password == key):
        TrainImages()
    elif (password == None):
        pass
    else:
        mess._show(title='Wrong Password', message='You have entered wrong
password')



def clear():
    txt.delete(0, 'end')
    res = "1)Take Images  >>>  2)Save Profile"
    message1.configure(text=res)


def clear2():
    txt2.delete(0, 'end')
    res = "1)Take Images  >>>  2)Save Profile"
    message1.configure(text=res)



def TakeImages():
    check_haarcascadefile()
    columns = ['SERIAL NO.', '', 'ID', '', 'NAME']
    assure_path_exists("StudentDetails/")
    assure_path_exists("TrainingImage/")
    serial = 0
    exists = os.path.isfile("StudentDetails\StudentDetails.csv")
    if exists:
        with open("StudentDetails\StudentDetails.csv", 'r') as csvFile1:
            reader1 = csv.reader(csvFile1)
            for l in reader1:
                serial = serial + 1
        serial = (serial // 2)
```

```
        csvFile1.close()
    else:
        with open("StudentDetails\StudentDetails.csv", 'a+') as csvFile1:
            writer = csv.writer(csvFile1)
            writer.writerow(columns)
            serial = 1
        csvFile1.close()
    Id = (txt.get())
    name = (txt2.get())
    if ((name.isalpha()) or (' ' in name)):
        cam = cv2.VideoCapture(0)
        harcascadePath = "haarcascade_frontalface_default.xml"
        detector = cv2.CascadeClassifier(harcascadePath)
        sampleNum = 0
        while (True):
            ret, img = cam.read()
            gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
            faces = detector.detectMultiScale(gray, 1.3, 5)
            for (x, y, w, h) in faces:
                cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 2)
                # incrementing sample number
                sampleNum = sampleNum + 1
                # saving the captured face in the dataset folder TrainingImage
                cv2.imwrite("TrainingImage\ " + name + "." + str(serial) + "." + Id
+ '.' + str(sampleNum) + ".jpg",
                            gray[y:y + h, x:x + w])
                # display the frame
                cv2.imshow('Taking Images', img)
            # wait for 100 miliseconds
            if cv2.waitKey(100) & 0xFF == ord('q'):
                break
            # break if the sample number is morethan 100
            elif sampleNum > 100:
                break
        cam.release()
        cv2.destroyAllWindows()
        res = "Images Taken for ID : " + Id
        row = [serial, '', Id, '', name]
        with open('StudentDetails\StudentDetails.csv', 'a+') as csvFile:
            writer = csv.writer(csvFile)
            writer.writerow(row)
        csvFile.close()
        message1.configure(text=res)
    else:
        if (name.isalpha() == False):
            res = "Enter Correct name"
            message.configure(text=res)



def TrainImages():
    check_haarcascadefile()
    assure_path_exists("TrainingImageLabel/")
    recognizer = cv2.face_LBPHFaceRecognizer.create()
    harcascadePath = "haarcascade_frontalface_default.xml"
    detector = cv2.CascadeClassifier(harcascadePath)
    faces, ID = getImagesAndLabels("TrainingImage")
    try:
        recognizer.train(faces, np.array(ID))
    except:
        mess._show(title='No Registrations', message='Please Register someone
first!!!')
```

```
        return
    recognizer.save("TrainingImageLabel\Trainner.yml")
    res = "Profile Saved Successfully"
    message1.configure(text=res)
    message.configure(text='Total Registrations till now  : ' + str(ID[0]))



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



def TrackImages():
    check_haarcascadefile()
    assure_path_exists("Attendance/")
    assure_path_exists("StudentDetails/")
    for k in tv.get_children():
        tv.delete(k)
    msg = ''
    i = 0
    j = 0
    recognizer = cv2.face.LBPHFaceRecognizer_create()  #
cv2.createLBPHFaceRecognizer()
    exists3 = os.path.isfile("TrainingImageLabel\Trainner.yml")
    if exists3:
        recognizer.read("TrainingImageLabel\Trainner.yml")
    else:
        mess._show(title='Data Missing', message='Please click on Save Profile to
reset data!!')
        return
    harcascadePath = "haarcascade_frontalface_default.xml"
    faceCascade = cv2.CascadeClassifier(harcascadePath);

    cam = cv2.VideoCapture(0)
    font = cv2.FONT_HERSHEY_SIMPLEX
    col_names = ['Id', '', 'Name', '', 'Date', '', 'Time']
    exists1 = os.path.isfile("StudentDetails\StudentDetails.csv")
    if exists1:
        df = pd.read_csv("StudentDetails\StudentDetails.csv")
    else:
        mess._show(title='Details Missing', message='Students details are missing,
please check!')
        cam.release()
        cv2.destroyAllWindows()
```

```
        window.destroy()
    while True:
        ret, im = cam.read()
        gray = cv2.cvtColor(im, cv2.COLOR_BGR2GRAY)
        faces = faceCascade.detectMultiScale(gray, 1.2, 5)
        for (x, y, w, h) in faces:
            cv2.rectangle(im, (x, y), (x + w, y + h), (225, 0, 0), 2)
            serial, conf = recognizer.predict(gray[y:y + h, x:x + w])
            if (conf < 50):
                ts = time.time()
                date = datetime.datetime.fromtimestamp(ts).strftime('%d-%m-%Y')
                timeStamp = datetime.datetime.fromtimestamp(ts).strftime('%H:%M:%S')
                aa = df.loc[df['SERIAL NO.'] == serial]['NAME'].values
                ID = df.loc[df['SERIAL NO.'] == serial]['ID'].values
                ID = str(ID)
                ID = ID[1:-1]
                bb = str(aa)
                bb = bb[2:-2]
                attendance = [str(ID), '', bb, '', str(date), '', str(timeStamp)]

            else:
                Id = 'Unknown'
                bb = str(Id)
            cv2.putText(im, str(bb), (x, y + h), font, 1, (255, 255, 255), 2)
        cv2.imshow('Taking Attendance', im)
        if (cv2.waitKey(1) == ord('q')):
            break
    ts = time.time()
    date = datetime.datetime.fromtimestamp(ts).strftime('%d-%m-%Y')
    exists = os.path.isfile("Attendance\Attendance_" + date + ".csv")
    if exists:
        with open("Attendance\Attendance_" + date + ".csv", 'a+') as csvFile1:
            writer = csv.writer(csvFile1)
            writer.writerow(attendance)
        csvFile1.close()
    else:
        with open("Attendance\Attendance_" + date + ".csv", 'a+') as csvFile1:
            writer = csv.writer(csvFile1)
            writer.writerow(col_names)
            writer.writerow(attendance)
        csvFile1.close()
    with open("Attendance\Attendance_" + date + ".csv", 'r') as csvFile1:
        reader1 = csv.reader(csvFile1)
        for lines in reader1:
            i = i + 1
            if (i > 1):
                if (i % 2 != 0):
                    iidd = str(lines[0]) + '   '
                    tv.insert('', 0, text=iidd, values=(str(lines[2]),
str(lines[4]), str(lines[6])))
    csvFile1.close()
    cam.release()
    cv2.destroyAllWindows()



global key
key = ''

ts = time.time()
date = datetime.datetime.fromtimestamp(ts).strftime('%d-%m-%Y')
day,month,year=date.split("-")
```

```
mont={'01':'January',
      '02':'February',
      '03':'March',
      '04':'April',
      '05':'May',
      '06':'June',
      '07':'July',
      '08':'August',
      '09':'September',
      '10':'October',
      '11':'November',
      '12':'December'
      }

#GUI FRONT-END

window = tk.Tk()
window.geometry("1280x720")
window.resizable(True,False)
window.title("Attendance System")
window.configure(background='#2d420a')

frame1 = tk.Frame(window, bg="#c79cff")
frame1.place(relx=0.11, rely=0.17, relwidth=0.39, relheight=0.80)

frame2 = tk.Frame(window, bg="#c79cff")
frame2.place(relx=0.51, rely=0.17, relwidth=0.38, relheight=0.80)

message3 = tk.Label(window, text="Face Recognition Based Attendance Monitoring
System" ,fg="white",bg="#2d420a" ,width=55 ,height=1,font=('comic', 29, ' bold '))
message3.place(x=10, y=10)

frame3 = tk.Frame(window, bg="#c4c6ce")
frame3.place(relx=0.52, rely=0.09, relwidth=0.09, relheight=0.07)

frame4 = tk.Frame(window, bg="#c4c6ce")
frame4.place(relx=0.36, rely=0.09, relwidth=0.16, relheight=0.07)

datef = tk.Label(frame4, text = day+"-"+mont[month]+"-"+year+"  |  ",
fg="#ff61e5",bg="#2d420a" ,width=55 ,height=1,font=('comic', 22, ' bold '))
datef.pack(fill='both',expand=1)

clock = tk.Label(frame3,fg="#ff61e5",bg="#2d420a" ,width=55 ,height=1,font=('comic',
22, ' bold '))
clock.pack(fill='both',expand=1)
tick()

head2 = tk.Label(frame2, text="                       For New Registrations
", fg="black",bg="#00fcca" ,font=('comic', 17, ' bold ') )
head2.grid(row=0,column=0)

head1 = tk.Label(frame1, text="                       For Already Registered
", fg="black",bg="#00fcca" ,font=('comic', 17, ' bold ') )
head1.place(x=0,y=0)

lbl = tk.Label(frame2, text="Enter ID",width=20  ,height=1  ,fg="black"
,bg="#c79cff" ,font=('comic', 17, ' bold ') )
lbl.place(x=80, y=55)

txt = tk.Entry(frame2,width=32 ,fg="black",font=('comic', 15, ' bold '))
txt.place(x=30, y=88)
```

```
lbl2 = tk.Label(frame2, text="Enter Name",width=20  ,fg="black"  ,bg="#c79cff"
,font=('comic', 17, ' bold '))
lbl2.place(x=80, y=140)

txt2 = tk.Entry(frame2,width=32 ,fg="black",font=('comic', 15, ' bold ')  )
txt2.place(x=30, y=173)

message1 = tk.Label(frame2, text="1)Take Images  >>>  2)Save Profile" ,bg="#c79cff"
,fg="black"  ,width=39 ,height=1, activebackground = "#3ffc00" ,font=('comic', 15, '
bold '))
message1.place(x=7, y=230)

message = tk.Label(frame2, text="" ,bg="#c79cff" ,fg="black"  ,width=39,height=1,
activebackground = "#3ffc00" ,font=('comic', 16, ' bold '))
message.place(x=7, y=450)

lbl3 = tk.Label(frame1, text="Attendance",width=20  ,fg="black"  ,bg="#c79cff"
,height=1 ,font=('comic', 17, ' bold '))
lbl3.place(x=100, y=115)

res=0
exists = os.path.isfile("StudentDetails\StudentDetails.csv")
if exists:
    with open("StudentDetails\StudentDetails.csv", 'r') as csvFile1:
        reader1 = csv.reader(csvFile1)
        for l in reader1:
            res = res + 1
    res = (res // 2) - 1
    csvFile1.close()
else:
    res = 0
message.configure(text='Total Registrations till now  : '+str(res))

#MENUBAR

menubar = tk.Menu(window,relief='ridge')
filemenu = tk.Menu(menubar,tearoff=0)
filemenu.add_command(label='Change Password', command = change_pass)
filemenu.add_command(label='Contact Us', command = contact)
filemenu.add_command(label='Exit',command = window.destroy)
menubar.add_cascade(label='Help',font=('comic', 29, ' bold '),menu=filemenu)

#TREEVIEW ATTENDANCE TABLE

tv= ttk.Treeview(frame1,height =13,columns = ('name','date','time'))
tv.column('#0',width=82)
tv.column('name',width=130)
tv.column('date',width=133)
tv.column('time',width=133)
tv.grid(row=2,column=0,padx=(0,0),pady=(150,0),columnspan=4)
tv.heading('#0',text ='ID')
tv.heading('name',text ='NAME')
tv.heading('date',text ='DATE')
tv.heading('time',text ='TIME')

#SCROLLBAR

scroll=ttk.Scrollbar(frame1,orient='vertical',command=tv.yview)
scroll.grid(row=2,column=4,padx=(0,100),pady=(150,0),sticky='ns')
tv.configure(yscrollcommand=scroll.set)
```

```
#BUTTONS

clearButton = tk.Button(frame2, text="Clear", command=clear  ,fg="black"
,bg="#ff7221"  ,width=11 ,activebackground = "white" ,font=('comic', 11, ' bold '))
clearButton.place(x=335, y=86)
clearButton2 = tk.Button(frame2, text="Clear", command=clear2  ,fg="black"
,bg="#ff7221"  ,width=11 , activebackground = "white" ,font=('comic', 11, ' bold '))
clearButton2.place(x=335, y=172)
takeImg = tk.Button(frame2, text="Take Images", command=TakeImages  ,fg="white"
,bg="#6d00fc"  ,width=34  ,height=1, activebackground = "white" ,font=('comic', 15,
' bold '))
takeImg.place(x=30, y=300)
trainImg = tk.Button(frame2, text="Save Profile", command=psw ,fg="white"
,bg="#6d00fc"  ,width=34  ,height=1, activebackground = "white" ,font=('comic', 15,
' bold '))
trainImg.place(x=30, y=380)
trackImg = tk.Button(frame1, text="Take Attendance", command=TrackImages
,fg="black"  ,bg="#3ffc00"  ,width=35  ,height=1, activebackground = "white"
,font=('comic', 15, ' bold '))
trackImg.place(x=30,y=50)
quitWindow = tk.Button(frame1, text="Quit", command=window.destroy  ,fg="black"
,bg="#eb4600"  ,width=35 ,height=1, activebackground = "white" ,font=('comic', 15, '
bold '))
quitWindow.place(x=30, y=450)

#END

window.configure(menu=menubar)
window.mainloop()

####################################################################################
################
```

CHAPTER 8
SCREENSHOTS AND RESULTS

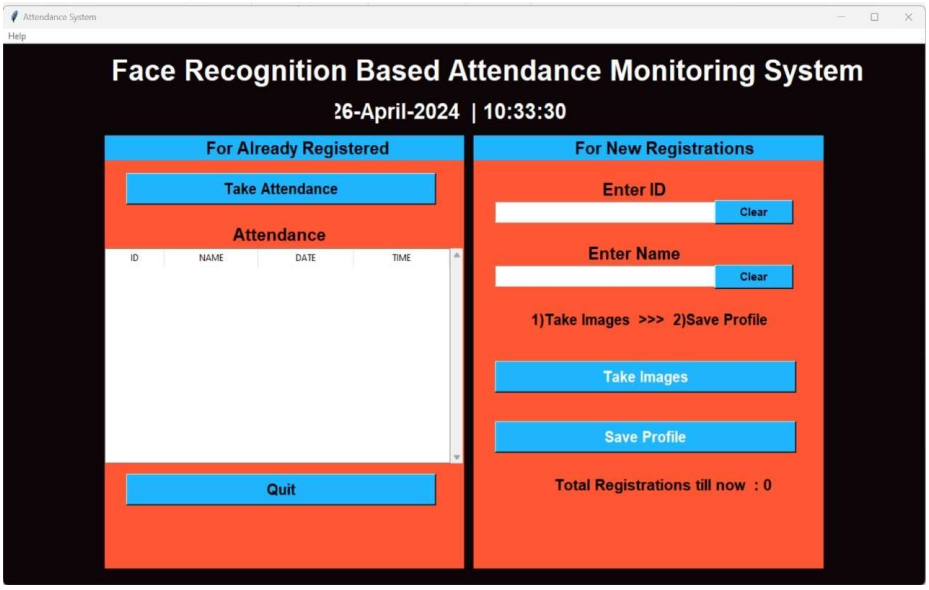


Figure 8.1: Graphical User Interface

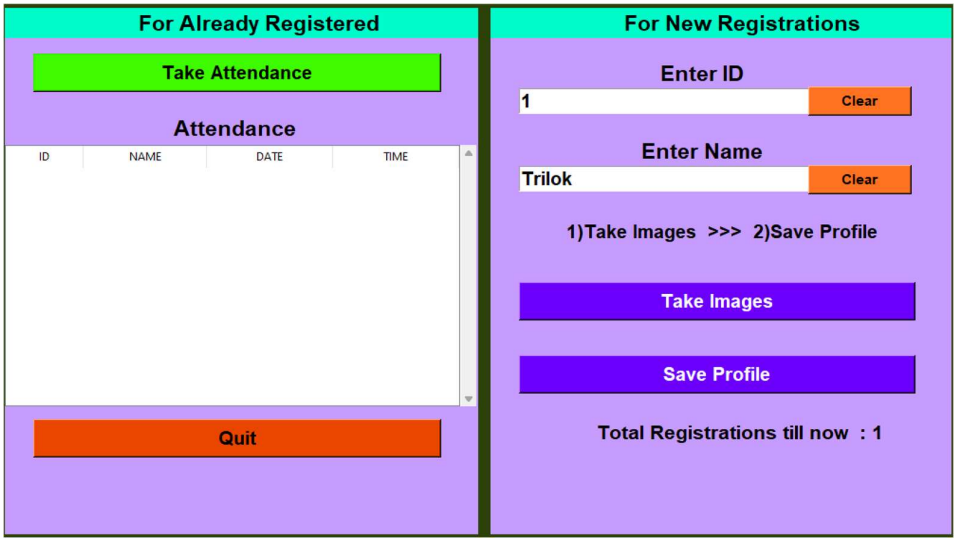


Figure 8.2: User Data/Information Collection

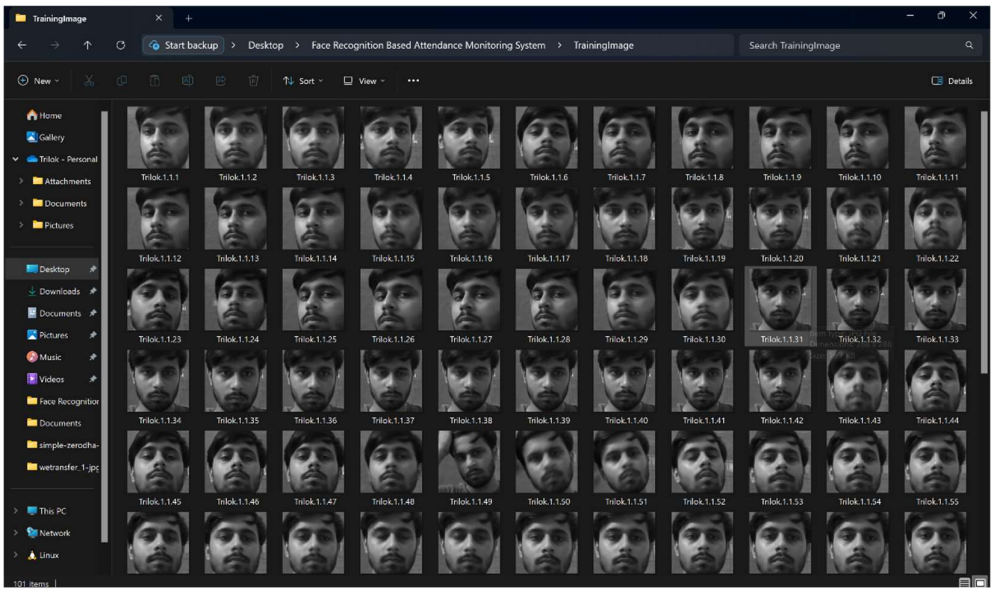


Figure 8.3: Target User Image Collection

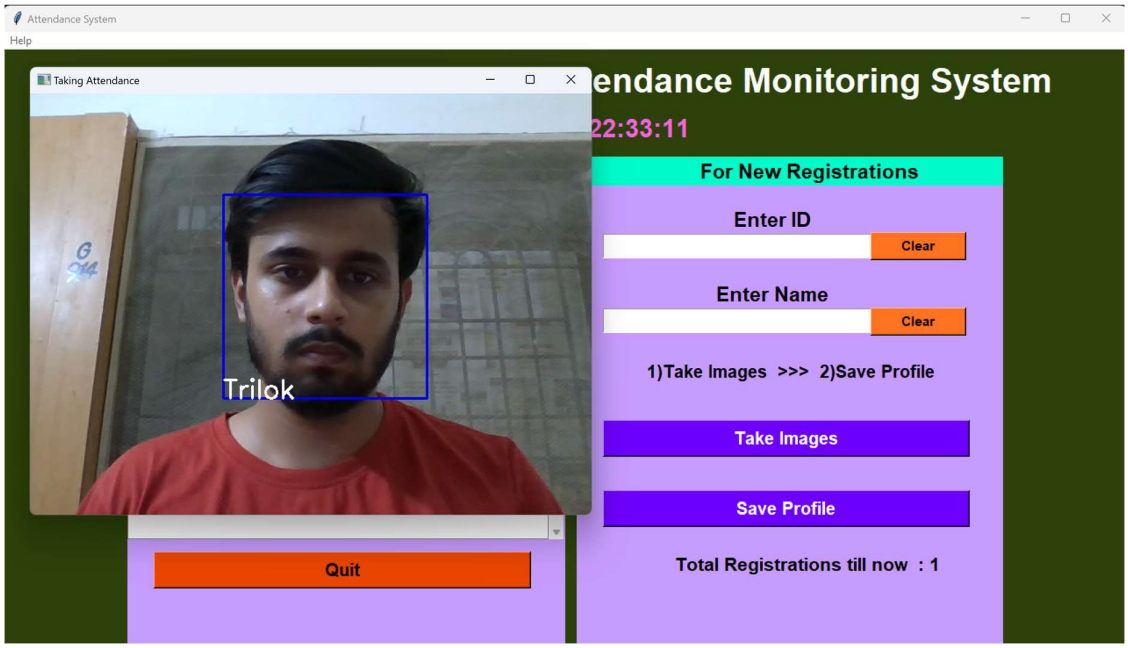
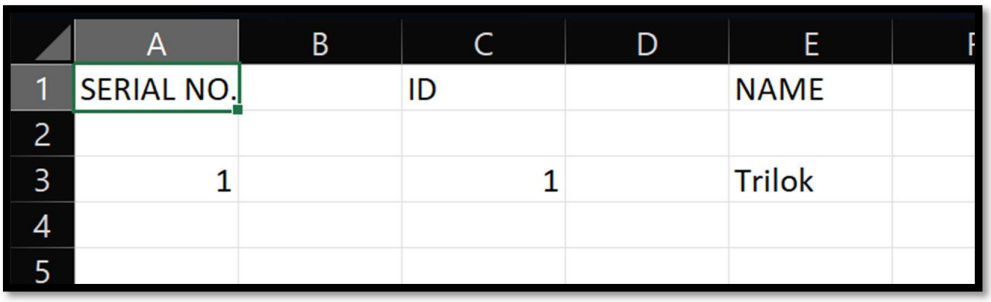
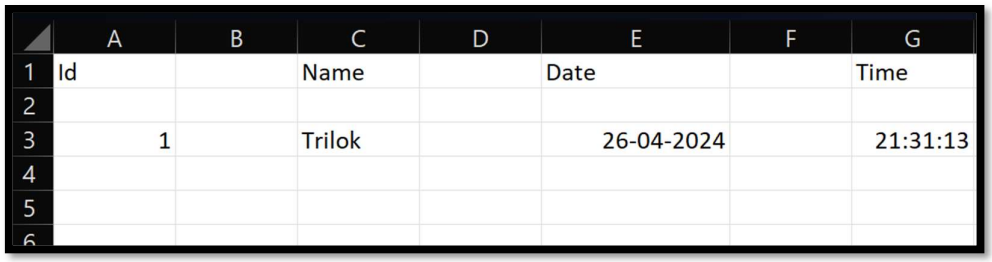


Figure 8.4: User Face Recognition



	A	B	C	D	E	F
1	SERIAL NO.		ID		NAME	
2						
3	1		1		Trilok	
4						
5						

Figure 8.5: Student Details Database



	A	B	C	D	E	F	G
1	Id		Name		Date		Time
2							
3	1		Trilok		26-04-2024		21:31:13
4							
5							
6							

Figure 8.6: Student Attendance Database

CHAPTER 9

REFERENCES

1. Kumar, P. P., Akshay, R., & Sagar, K. (2024). Face Recognition-Based Smart Attendance Monitoring System in Classroom. In Trends in Sustainable Computing and Machine Intelligence (pp. 407-422). https://link.springer.com/chapter/10.1007/978-981-99-9436-6_28
2. Sai, E. C., Hussain, S. A., Khaja, S., & Shyam, A. (2021). Student Attendance Monitoring System Using Face Recognition. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3851056
3. Kulkarni, S., & Choudhari, D. (2023). Attendance Monitoring System using Face Recognition. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4669196
4. Chatterjee, S., Das, D., Adhikary, D., Dutta, M., Mitra, I., Ghosh, M., & Bhunia, P. K. (2023). Face Recognition-Based Smart Attendance Monitoring and Tracking System. *JETIR Research Journal*. https://www.jetir.org/papers/JETIR2306977.pdf
5. Akbar, Md Sajid, Et Al. "Face Recognition And RFID Verified Attendance System." 2018 International Conference On Computing, Electronics & Communications Engineering (ICCECE). IEEE, 2018
6. Patel, R., & Patel, A. (2020). "Automatic Attendance System Using Face Recognition." International Journal of Advanced Research in Computer Science, 11(2), 94-99.
7. 2. Hu, P., Zhang, Z., & Zhou, L. (2018). "Face Attendance System Based on Deep Learning." 2018 International Conference on Machine Learning and Intelligent Systems (MLIS).