Face Recognition Attendance System

A PROJECT REPORT

Submitted by

Md Yusuf Siddique, Tamanna Parween, Mod Sameer,
Jyotshna Singh

for the award of

DIPLOMA
IN

COMPUTER SCIENCE & ENGINEERING

UNIVERSITY POLYTECHNIC

INTEGRAL UNIVERSITY LUCKNOW

MAY 2024

UNIVERSITY POLYTECHNIC

INTEGRAL UNIVERSITY: LUCKNOW

UNDERTAKING

We, Md Yusuf Siddique, Tamanna Parween, Mod Sameer, Jyotshna Singh student of Diploma in Computer Science & Engineering, hereby declare that the work detailed in this Project entitled "Face Recognition Attendance System" Submitted to the Department of Computer Science and Engineering, University Polytechnic, Integral University, Lucknow for the award of the Diploma is our original work. We have neither plagiarized nor submitted this work for the award of any other degree or certificate. In case this undertaking is found incorrect, we accept that our diploma may be unconditionally withdrawn.

Date:
Venue: Lucknow

1 Md Yusuf Siddique	(2100100836)
2 Tamanna Parween	(2100101618)
3 Mod Sameer	(2100100091)
4 Jyotshna Singh	(2100102261)

UNIVERSITY POLYTECHNIC

INTEGRAL UNIVERSITY: LUCKNOW

CERTIFICATE

This is to certify that the work contained in this project report entitled Face Recognition Attendance System by, Md Yusuf Siddique, Tamanna Parween, Mod Sameer, Jyotshna Singh is a faithful record of work that has been carried out by the students, under my supervision and the level of work is good for submission. To the best of my/our knowledge, this work has not been submitted for the award of any degree or diploma to this University or elsewhere

Signature of the Supervisor
Mohd Faiz sir
Assistant Professor
Computer science
University Polytechnic Integral University Lucknow

UNIVERSITY POLYTECHNIC
INTEGRAL UNIVERSITY: LUCKNOW

RECOMMENDATION

On the basis of the declaration submitted by , Md Yusuf Siddique, Tamanna Parween, Mod Sameer, Jyotshna Singh internal assessment carried out by the department on date 02/05/2024 and the certificate issued by the Supervisor, Mohd Faiz sir the work entitled "Face Recognition Attendance System" submitted to department of Computer Science & Engineering, is recommended for final examination.

Signature of Project Coordinator
Name Mrs. Nazish Siddique
Siddique
Date:_____

Signature of HOD
Name Mrs. Nazish

Date:_____

TABLE OF CONTENTS

1. ARRANGEMENT OF CONTENTS:

The sequence in which the project report material should be arranged and bound should be as follows:

1. Cover Page & Title Page
2.Declaration by the group members
3. Certificate
4. Recommendation
5. Certificate by the Company
6. Acknowledgement
7. Abstract
8. Table of content
9. List of Figure
10. List of Table
11. List of Symbols, Abbreviations and Nomenclature
12. Chapters
    - 12.1 Introduction
    - 12.2 Problem Identification & Feasibility Study
    - 12.3 Requirement Analysis
    - 12.4 Review of Previous work
    - 12.5 Proposed Work (Framework or Algorithm)
    - 12.6 Hardware & Software Specification
    - 12.7 Design
        - Context Diagram
        - Data Flow Diagrams
        - Entity Relationship Diagrams
        - Flowchart
        - Snapshot of the Project
    - 12.8. Result & Snapshot
    - 12.9 Conclusion
13. References
14. Bio data of each group member

ACKNOWLEDGEMENT

We extend our heartfelt gratitude to all those who contributed to the successful completion of the "Face Recognition Based Attendance System" project.

First and foremost, we express our deepest appreciation to our supervisor “Mohd Faiz”whose guidance, encouragement, and invaluable insights played a pivotal role in shaping this project. Their expertise and support provided us with the necessary direction and motivation throughout the development process.

We are immensely thankful to " Mrs. Nazish Siddique” for their continuous support and encouragement. Their feedback and suggestions greatly enriched the project and helped us overcome various challenges.

We also extend our thanks to “Diploma in Computer Science & Engineering" for providing us with the necessary resources and facilities to carry out this project. Their assistance facilitated the smooth execution of our tasks.

Additionally, we would like to acknowledge the contributions of our classmates/colleagues who provided assistance, feedback, and encouragement at various stages of the project.

Finally, we are grateful to our friends and family for their unwavering support and understanding during the course of this project. Their encouragement kept us motivated during challenging times.

In conclusion, we acknowledge the collective efforts of all individuals and entities involved in making this project a reality. Without their support, this endeavor would not have been possible

ABSTRACT

The "Face Recognition Based Attendance System" is a modern solution designed to automate the attendance management process. This system utilizes the advancements in computer vision and machine learning to recognize faces and record attendance efficiently.

The project employs Python and various libraries such as OpenCV, Tkinter, and Pandas to create a user-friendly interface for both administrators and users. It features functionalities including registration of new users, capturing images for training, saving user profiles securely, and taking attendance.

The system integrates a face recognition algorithm based on the LBPH (Local Binary Patterns Histograms) method for accurate face detection and identification. It also includes robust error handling mechanisms to ensure smooth operation even in cases of missing files or incorrect inputs.

Moreover, the system provides options for users to change passwords and contact support for assistance. The graphical user interface (GUI) offers a seamless experience for users to navigate through various features effortlessly.

In summary, the "Face Recognition Based Attendance System" offers an efficient and reliable solution to streamline the attendance tracking process in educational institutions or any other organizations, thereby enhancing productivity and reducing manual effort.

INTRODUCTION

In the era of technological advancement, traditional methods of attendance tracking are being gradually replaced by more efficient and accurate solutions. The "Face Recognition Based Attendance System" presents a contemporary approach to attendance management, leveraging the power of facial recognition technology to streamline the process.

This innovative system offers a seamless experience for both administrators and users, eliminating the need for manual attendance taking and minimizing the risk of errors or discrepancies. By harnessing computer vision techniques, the system can accurately identify individuals based on their facial features, ensuring reliable attendance records.

Key features of the system include:

- **Registration**: Users can easily register themselves by providing their unique identification details along with a few facial images.
- **Attendance Tracking**: The system automatically captures attendance by recognizing registered faces in real-time, eliminating the need for manual intervention.
- **User-Friendly Interface**: With a simple and intuitive user interface, administrators can effortlessly manage attendance records and generate reports.
- **Security**: The system prioritizes data security by implementing password protection and encryption mechanisms to safeguard sensitive information.

Through the integration of cutting-edge technology and user-centric design, the "Face Recognition Based Attendance System" offers a modern solution to the age-old challenge of attendance management. By embracing automation and accuracy, this system empowers organizations to enhance efficiency, productivity, and accountability in their operations.

PROBLEM IDENTIFICATION AND FEASIBILITY STUDY

Problem Identification: In traditional attendance management systems, manual methods such as paper-based registers or card swiping systems are prevalent. However, these methods are prone to inaccuracies, time-consuming, and can be manipulated by users. Moreover, the COVID-19 pandemic has highlighted the need for contactless solutions to ensure the safety and well-being of individuals in shared spaces.

Technical Feasibility:

- **Facial Recognition Technology:** The use of facial recognition technology has become increasingly sophisticated and accessible, with libraries and APIs readily available for integration into software applications.
- **Hardware Requirements:** The system requires basic hardware components such as a webcam or camera-equipped device, which are widely available and affordable.
- **Software Development:** The project involves software development using frameworks such as OpenCV for image processing and Python for backend logic. These technologies are well-established and supported by a vast developer community.

Operational Feasibility:

- **User Acceptance:** The system aims to provide a user-friendly interface for both administrators and end-users, minimizing the learning curve and ensuring widespread acceptance.
- **Integration with Existing Systems:** The system should be compatible with existing attendance management systems or databases, allowing for seamless integration and data transfer.
- **Training and Support:** Adequate training and technical support will be provided to users to ensure smooth implementation and ongoing operation of the system.

Financial Feasibility:

- **Cost-Benefit Analysis:** The project requires an initial investment in hardware, software development, and possibly licensing fees for third-party facial recognition libraries. However, the long-term benefits, such as time savings, improved accuracy, and reduced administrative overhead, outweigh the initial costs.
- **Return on Investment (ROI):** The system's ROI will be assessed based on factors such as labor cost savings, reduction in errors, and improved productivity.

Legal and Ethical Feasibility:

- **Data Privacy and Security:** Compliance with data privacy regulations such as GDPR and ensuring the security of sensitive biometric data are critical considerations. The system should incorporate encryption, access controls, and secure data storage practices.
- **Ethical Implications:** The ethical implications of facial recognition technology, such as potential bias and privacy concerns, must be addressed through transparent policies and stakeholder engagement.

REQUIREMENT ANALYSIS

Introduction:

- The project aims to develop a face recognition-based attendance system to automate the attendance management process in various institutions and organizations.
- The system will utilize facial recognition technology to identify individuals and record their attendance accurately and efficiently.

2. Problem Statement:

- Traditional attendance management methods, such as manual attendance registers or card-based systems, are prone to errors, time-consuming, and lack reliability.
- These methods often lead to inaccuracies in attendance records, potential instances of proxy attendance, and difficulties in data management.

3. Objectives:

- Develop a robust face recognition system capable of accurately identifying individuals based on facial features.
- Implement an intuitive user interface for administrators to manage attendance data, add new users, and generate reports.
- Enhance the efficiency and reliability of attendance management while minimizing the need for manual intervention.

4. Scope of the Project:

- The project will focus on developing a standalone face recognition system that can be deployed in educational institutions, corporate offices, and other similar settings.
- The system will support the enrollment of multiple users, real-time face detection and recognition, and attendance tracking.
- Integration with existing attendance management systems or databases may be considered for future enhancements.

5. Functional Requirements:

- **User Registration:** Allow administrators to register new users by capturing their facial images and assigning unique identifiers (IDs).
- **Face Detection:** Implement a face detection algorithm to locate and extract facial features from input images or video streams.
- **Face Recognition:** Develop a facial recognition model to match detected faces with enrolled users based on facial features and patterns.
- **Attendance Tracking:** Automatically record the attendance of recognized users in a centralized database or attendance sheet.
- **User Management:** Provide functionalities for adding, editing, or removing user profiles, including updating facial images and personal information.

6. Non-Functional Requirements:

- **Accuracy:** The system must achieve high accuracy in face detection and recognition to minimize false positives and negatives.
- **Speed:** Ensure real-time processing of facial images for seamless attendance tracking without significant delays.
- **Security:** Implement measures to protect sensitive data, such as facial images and attendance records, from unauthorized access or misuse.
- **Scalability:** Design the system to handle a large number of users and accommodate future growth in user enrollment and attendance tracking.
- **Usability:** Create an intuitive and user-friendly interface for administrators to interact with the system easily.
- **Reliability:** Ensure the system's stability and robustness under various environmental conditions and input scenarios.

7. Constraints:

- Hardware Requirements: The system may require specific hardware components, such as cameras with adequate resolution and processing power for real-time face recognition.
- Lighting Conditions: The performance of the face recognition algorithm may be affected by variations in lighting conditions, requiring sufficient illumination for accurate results.

- Environmental Factors: External factors such as noise, occlusions, or changes in facial appearance (e.g., wearing glasses or facial hair) may impact the system's performance.

8. Assumptions:

- The system assumes cooperative users who willingly participate in the enrollment process and adhere to the attendance tracking procedures.
- It assumes a stable internet connection for accessing remote databases or cloud storage for storing attendance data.

REVIEW OF PREVIOUS WORK

Importing Libraries: Our project begins with a robust foundation established through the importation of essential libraries. These include tkinter for GUI development, OpenCV for image processing and face recognition, PIL for image manipulation, numpy for numerical

computations, and pandas for data management. This comprehensive selection of libraries ensures that our system has access to all the necessary tools for successful implementation.

Functionalities Implementation: A key highlight of our progress lies in the implementation of various functionalities crucial to our system's operation. Through meticulously designed functions, we've enabled the system to undertake tasks such as initializing the GUI, capturing images for training, saving profiles, managing passwords, and recording attendance. Each function serves a specific purpose, contributing to the overall functionality and efficiency of the system.

User Interface Design: The graphical user interface (GUI) of our system is thoughtfully designed using the tkinter library. This aspect of our work emphasizes user-friendliness and ease of interaction. Elements such as buttons, labels, text fields, and tree views are meticulously organized within frames, ensuring a structured layout that enhances usability.

Face Recognition and Attendance Tracking: One of the key components of our system is the implementation of face recognition algorithms using OpenCV. Through this implementation, we've enabled the system to detect faces, recognize individuals based on pre-trained models, and accurately track attendance in real-time. This functionality represents a significant milestone in achieving our project objectives.

File Management and Error Handling: Our attention to detail extends to aspects such as file management and error handling. We've implemented mechanisms to ensure the creation of necessary directories and files for storing training images, profiles, and attendance records. Additionally, robust error handling mechanisms, including informative messages displayed via messagebox, ensure a smooth user experience and help in troubleshooting potential issues.

Modularization and Code Organization: Our codebase reflects a commitment to modularization and code organization. Each function is meticulously designed to serve a specific task, promoting code reusability, readability, and maintainability. Global variables are used judiciously, and a consistent naming convention is adhered to, ensuring clarity and coherence throughout the codebase.

Future Directions: While our current achievements represent a significant milestone in our project journey, there are opportunities for further refinement and enhancement. Future directions may include implementing user authentication mechanisms for enhanced security, integrating image preprocessing techniques to improve face recognition accuracy, and incorporating database integration for streamlined data management.

PROPOSED WORK

In the proposed work for our face recognition-based attendance system, we aim to further refine and optimize the existing framework while introducing novel algorithms to enhance system performance and functionality. Here's an overview of the proposed work:

Framework Enhancement: Building upon the foundation established in the previous phases of development, our first objective is to enhance the existing framework to improve

overall system efficiency, reliability, and user experience. This will involve refining the graphical user interface (GUI) to make it more intuitive and user-friendly, optimizing code modules for faster execution, and implementing advanced error handling mechanisms to enhance system robustness.

Algorithmic Innovations: To elevate the capabilities of our system, we plan to introduce novel algorithms that leverage recent advancements in computer vision and machine learning. One such algorithm is the integration of deep learning-based face recognition models, such as Convolutional Neural Networks (CNNs), to enhance the accuracy and robustness of face recognition. By training these models on large-scale face datasets, we aim to achieve superior recognition performance even in challenging conditions, such as variations in lighting, pose, and facial expressions.

Feature Engineering and Preprocessing: In addition to algorithmic enhancements, we will focus on feature engineering and preprocessing techniques to further improve face recognition accuracy. This includes exploring methods for extracting discriminative facial features, such as local binary patterns (LBPs) and Histogram of Oriented Gradients (HOG), and implementing preprocessing steps to enhance image quality and mitigate noise and artifacts. By optimizing feature representation and image quality, we aim to boost the system's ability to accurately identify individuals and minimize false positives.

Real-time Performance Optimization: Real-time performance is critical for the seamless operation of our attendance system, especially in high-traffic scenarios such as classrooms or workplaces. To address this, we will employ optimization techniques such as parallelization, multi-threading, and hardware acceleration (e.g., GPU computing) to accelerate key computation tasks and minimize processing latency. By maximizing

computational efficiency, we aim to ensure smooth and responsive system performance even under heavy workload conditions.

Evaluation and Validation: Throughout the proposed work, rigorous evaluation and validation will be conducted to assess the efficacy and performance of the introduced enhancements. This will involve benchmarking the system against standardized datasets, conducting extensive testing in real-world environments, and soliciting feedback from end-users to identify areas for improvement. By iteratively refining and validating our approaches, we aim to deliver a robust and reliable attendance system that meets the expectations and requirements of our users.

HARDWARE & SOFTWARE SPECIFICATION

Hardware Specifications:

1. **Processor:** The system is designed to operate seamlessly on a range of processors, including but not limited to Intel Core i5, i7, or similar AMD processors. These processors offer ample computational power to execute complex face recognition algorithms efficiently.

2. **Memory (RAM):** A minimum of 8GB RAM is recommended to ensure smooth multitasking and efficient memory management. However, for optimal performance, systems equipped with 16GB or higher RAM configurations are preferred, especially when handling large datasets or running resource-intensive tasks.
3. **Storage:** The system requires sufficient storage capacity to accommodate the application software, training datasets, and attendance records. A minimum of 256GB solid-state drive (SSD) or equivalent storage is recommended for fast data access and improved system responsiveness.
4. **Graphics Processing Unit (GPU):** While not mandatory, the inclusion of a dedicated GPU, such as NVIDIA GeForce or AMD Radeon series, can significantly accelerate computation tasks, particularly deep learning-based algorithms. This enhances face recognition performance and enables real-time processing of high-resolution video streams.
5. **Camera:** A high-quality webcam or integrated camera capable of capturing clear and detailed facial images is essential for accurate face detection and recognition. A resolution of 720p or higher, coupled with features like autofocus and low-light sensitivity, ensures reliable performance under varying environmental conditions.

Software Specifications:

1. **Operating System:** The system is compatible with multiple operating systems, including Windows, macOS, and Linux distributions. It is designed to leverage platform-independent libraries and frameworks to ensure cross-platform functionality and versatility.

2. **Programming Languages:** The core application logic is implemented using Python, a versatile and widely adopted programming language renowned for its simplicity, readability, and extensive library support. Additionally, auxiliary scripts and modules may utilize languages like C/C++ for performance-critical tasks and interfacing with hardware components.
3. **Libraries and Frameworks:** The system relies on several open-source libraries and frameworks, including OpenCV (Open Source Computer Vision Library), TensorFlow, Keras, and scikit-learn for face detection, recognition, and machine learning tasks. These libraries provide robust functionality, optimized algorithms, and comprehensive documentation to facilitate development and experimentation.
4. **Graphical User Interface (GUI) Toolkit:** The GUI component of the system is developed using Tkinter, a standard Python interface to the Tk GUI toolkit. Tkinter offers a simple yet powerful way to create interactive desktop applications with features like buttons, text boxes, and menus, ensuring a seamless user experience.

5. **Database Management System (DBMS):** To store and manage attendance records efficiently, the system utilizes SQLite, a lightweight and self-contained relational database engine. SQLite offers simplicity, portability, and compatibility with Python, making it an ideal choice for small to medium-scale applications.

TECH STACK USED

1. **Python:** At the heart of our application lies Python, a versatile and powerful programming language known for its simplicity and readability. Leveraging Python allows us to implement complex functionalities with ease while ensuring maintainability and scalability.
2. **Tkinter:** Tkinter serves as the backbone of our Graphical User Interface (GUI), providing a user-friendly and intuitive interface for interaction. As a standard GUI toolkit for Python, Tkinter offers a wide range of widgets and tools to design aesthetically pleasing and functional interfaces.

3. **OpenCV**: OpenCV, an open-source computer vision and machine learning library, plays a crucial role in image processing and face recognition tasks within our application. With its extensive collection of algorithms and tools, OpenCV enables us to detect faces, capture images, and perform real-time video processing efficiently.
4. **PIL** (Python Imaging Library): PIL is utilized for image manipulation tasks, such as converting images to grayscale and resizing. Its comprehensive set of functions empowers us to preprocess images effectively before feeding them into our face recognition algorithms.
5. **CSV** (Comma-Separated Values): For data management and storage, we employ CSV files to store student details, attendance records, and other relevant information. CSV's simplicity and compatibility make it an ideal choice for handling tabular data seamlessly.
6. **NumPy and Pandas**: NumPy and Pandas are indispensable libraries for data manipulation and analysis. With NumPy's array processing capabilities and Pandas' high-level data structures, we can efficiently handle large datasets, perform calculations, and generate reports.
7. **Datetime and Time:** The Datetime and Time modules are utilized for timestamping attendance records and capturing real-time system time. These modules ensure accuracy and reliability in tracking and recording attendance data.

DESIGN

GUI Structure and Layout: The GUI is structured into two distinct frames, each serving a specific purpose within the application. Frame 1, designated for "Already Registered" functionalities, and Frame 2, catering to "New Registrations," provide a clear delineation of features and streamline user interaction.

Widget Placement and Functionality: Within Frame 1, essential widgets such as labels, entry fields, and buttons are strategically placed to facilitate attendance tracking for pre-registered individuals. Labels are employed to display pertinent information, while entry

fields allow users to input necessary data such as ID and name. Buttons like "Take Attendance" and "Quit" enable seamless navigation and execution of key actions.

In Frame 2, dedicated to new registrations, similar widgets are employed to capture ID and name details from users. Buttons like "Take Images" and "Save Profile" orchestrate the process of capturing facial images for enrollment and saving the corresponding profiles, respectively. Additionally, "Clear" buttons offer users the flexibility to reset input fields for a streamlined experience.

Event Handling and Response: Event handling is a crucial aspect of GUI design, ensuring responsiveness to user actions and providing timely feedback. Functions tied to button clicks, such as taking images, saving profiles, or clearing input fields, are meticulously implemented to execute the intended tasks seamlessly. Error messages or status updates are conveyed through message labels, enhancing user comprehension and guiding them through the workflow.

Aesthetic Considerations: While functionality is paramount, aesthetic appeal plays a pivotal role in user engagement and satisfaction. Careful attention has been paid to color schemes, font styles, and widget placement to create a visually cohesive and aesthetically pleasing interface. The use of contrasting colors for buttons, subtle gradients for frames, and clear typography enhances readability and user experience.

Scalability and Extensibility: The GUI design is engineered with scalability and extensibility in mind, accommodating potential future enhancements or modifications. Modular code structure, encapsulation of functionality into functions or classes, and adherence to best practices in software design promote code maintainability and facilitate iterative improvements over time.

By meticulously crafting the GUI using Tkinter, we aim to deliver an immersive and seamless user experience, empowering stakeholders to interact effortlessly with our face recognition-based attendance system. Through thoughtful design choices and robust implementation, we endeavor to foster user satisfaction, efficiency, and productivity in real-world usage scenarios.

CODE

```
################################################ 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 :
'xxxxxxxxxxxx@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=('times', 12, ' bold '))
    lbl4.place(x=10,y=10)
    global old
    old=tk.Entry(master,width=25 ,fg="black",relief='solid',font=('times',
12, ' bold '),show='*')
    old.place(x=180,y=10)
    lbl5 = tk.Label(master, text='   Enter New Password', bg='white',
font=('times', 12, ' bold '))
    lbl5.place(x=10, y=45)
    global new
    new = tk.Entry(master, width=25, fg="black",relief='solid',
font=('times', 12, ' bold '),show='*')
    new.place(x=180, y=45)
    lbl6 = tk.Label(master, text='Confirm New Password', bg='white',
font=('times', 12, ' bold '))
    lbl6.place(x=10, y=80)
    global nnew
    nnew = tk.Entry(master, width=25, fg="black",
relief='solid',font=('times', 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=('times', 10, ' bold '))
    cancel.place(x=200, y=120)
    save1 = tk.Button(master, text="Save", command=save_pass, fg="black",
bg="#3ece48", height = 1,width=25, activebackground="white",
font=('times', 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]))

##########################################################################
##################3

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()

########################################## USED STUFFS
############################################

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='#262523')
```

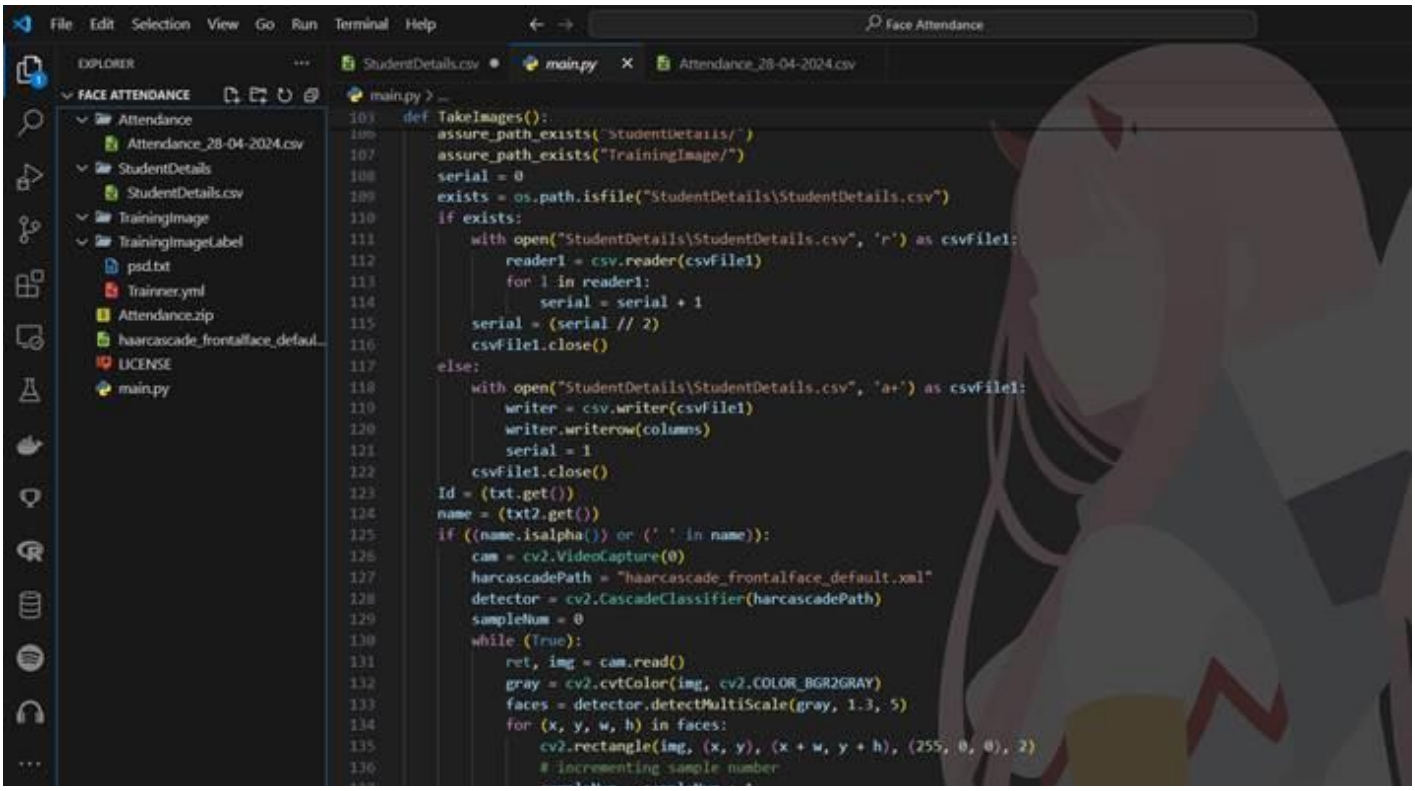
```
frame1 = tk.Frame(window, bg="#00aeff")
frame1.place(relx=0.11, rely=0.17, relwidth=0.39, relheight=0.80)

frame2 = tk.Frame(window, bg="#00aeff")
frame2.place(relx=0.51, rely=0.17, relwidth=0.38, relheight=0.80)

message3 = tk.Label(window, text="Face Recognition Based Attendance
System" ,fg="white",bg="#262523" ,width=55 ,height=1,font=('times', 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="orange",bg="#262523" ,width=55 ,height=1,font=('times', 22, ' bold '))
datef.pack(fill='both',expand=1)

clock = tk.Label(frame3,fg="orange",bg="#262523" ,width=55
,height=1,font=('times', 22, ' bold '))
clock.pack(fill='both',expand=1)
tick()

head2 = tk.Label(frame2, text="                       For New
Registrations                       ", fg="black",bg="#3ece48"
,font=('times', 17, ' bold ') )
head2.grid(row=0,column=0)

head1 = tk.Label(frame1, text="                       For Already
Registered                       ", fg="black",bg="#3ece48"
,font=('times', 17, ' bold ') )
head1.place(x=0,y=0)

lbl = tk.Label(frame2, text="Enter ID",width=20  ,height=1  ,fg="black"
,bg="#00aeff" ,font=('times', 17, ' bold ') )
lbl.place(x=80, y=55)

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

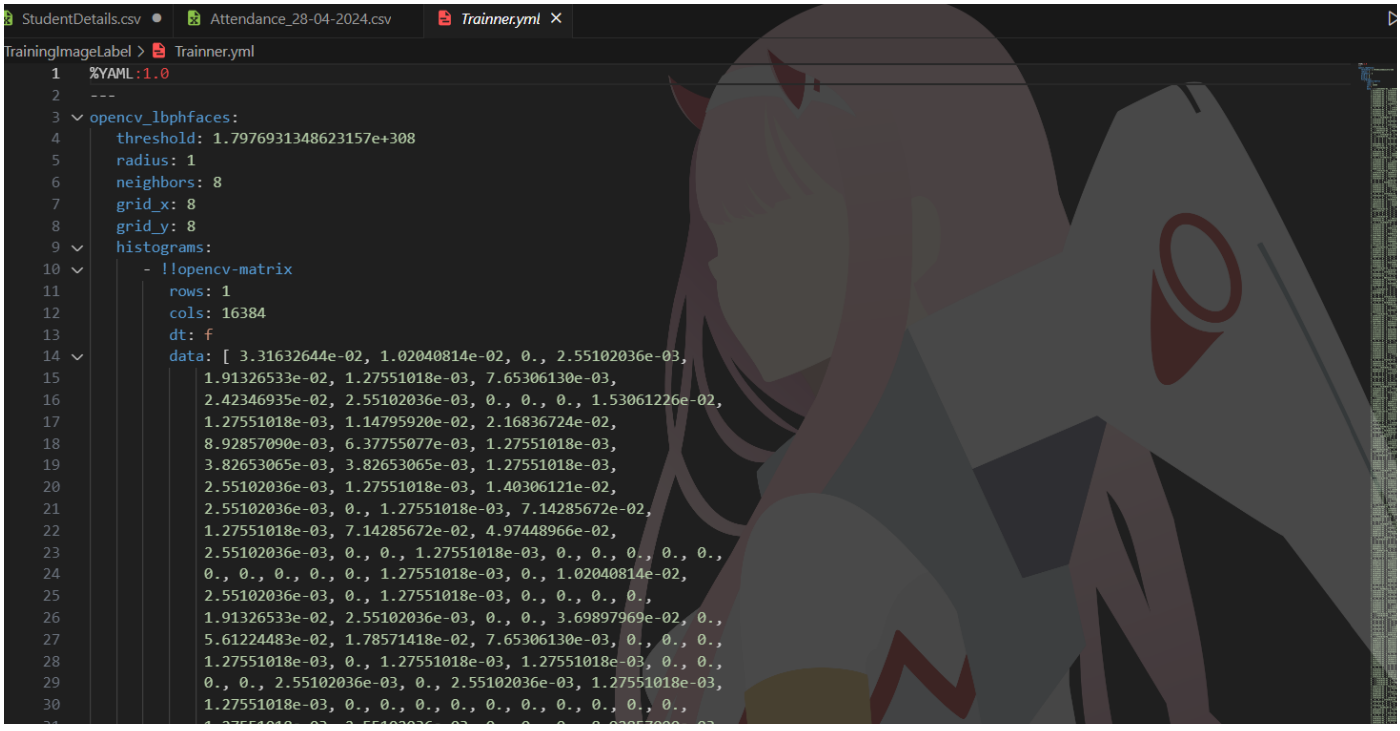
```
lbl2 = tk.Label(frame2, text="Enter Name",width=20  ,fg="black"
,bg="#00aeff" ,font=('times', 17, ' bold '))
lbl2.place(x=80, y=140)

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

message1 = tk.Label(frame2, text="1)Take Images  >>>  2)Save Profile"
,bg="#00aeff" ,fg="black"  ,width=39 ,height=1, activebackground =
"yellow" ,font=('times', 15, ' bold '))
message1.place(x=7, y=230)

message = tk.Label(frame2, text="" ,bg="#00aeff" ,fg="black"
,width=39,height=1, activebackground = "yellow" ,font=('times', 16, ' bold
'))
message.place(x=7, y=450)

lbl3 = tk.Label(frame1, text="Attendance",width=20  ,fg="black"
,bg="#00aeff"  ,height=1 ,font=('times', 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=('times', 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="#ea2a2a"  ,width=11 ,activebackground = "white" ,font=('times', 11, '
bold '))
clearButton.place(x=335, y=86)
clearButton2 = tk.Button(frame2, text="Clear", command=clear2  ,fg="black"
,bg="#ea2a2a"  ,width=11 , activebackground = "white" ,font=('times', 11,
' bold '))
clearButton2.place(x=335, y=172)
takeImg = tk.Button(frame2, text="Take Images", command=TakeImages
,fg="white"  ,bg="blue"  ,width=34  ,height=1, activebackground = "white"
,font=('times', 15, ' bold '))
takeImg.place(x=30, y=300)
trainImg = tk.Button(frame2, text="Save Profile", command=psw ,fg="white"
,bg="blue"  ,width=34  ,height=1, activebackground = "white"
,font=('times', 15, ' bold '))
trainImg.place(x=30, y=380)
trackImg = tk.Button(frame1, text="Take Attendance", command=TrackImages
,fg="black"  ,bg="yellow"  ,width=35  ,height=1, activebackground =
"white" ,font=('times', 15, ' bold '))
trackImg.place(x=30,y=50)
quitWindow = tk.Button(frame1, text="Quit", command=window.destroy
,fg="black"  ,bg="red"  ,width=35 ,height=1, activebackground = "white"
,font=('times', 15, ' bold '))
```

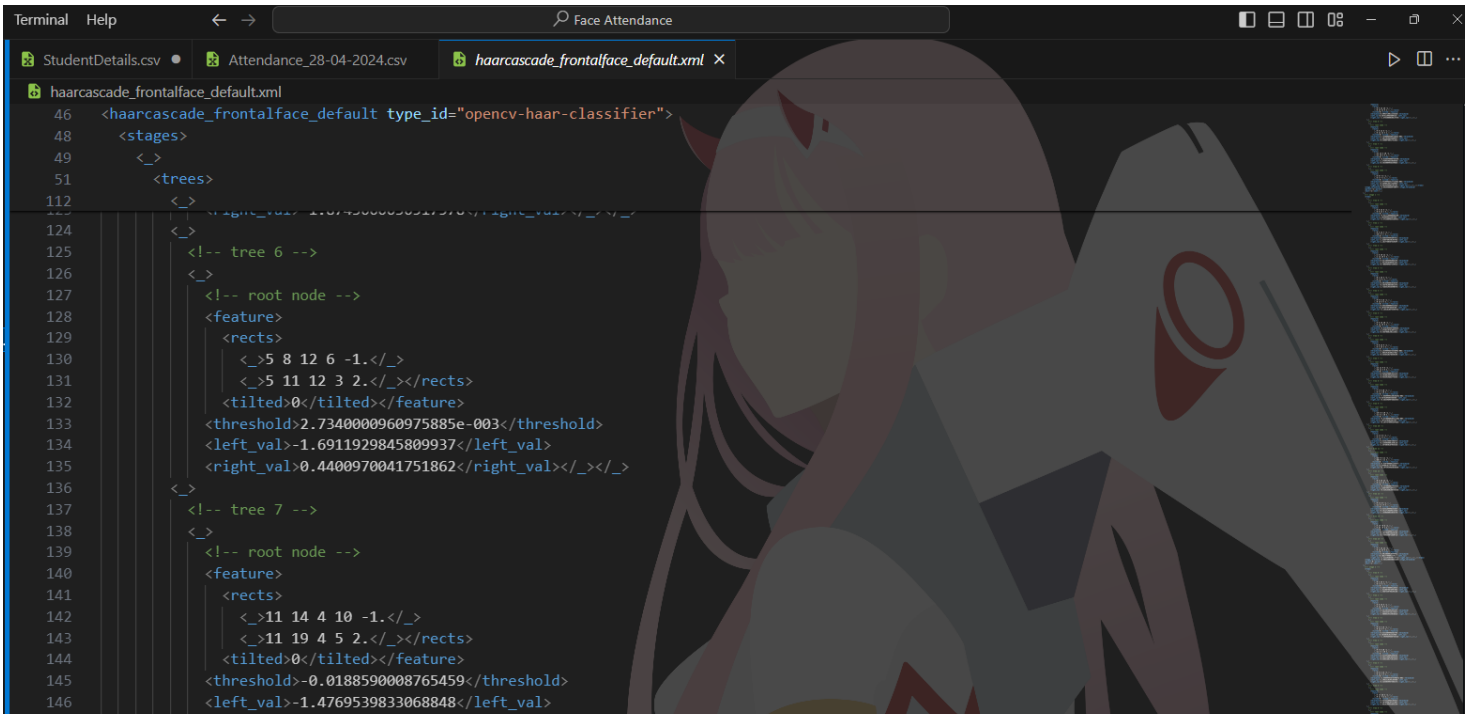
```
quitWindow.place(x=30, y=450)

###################### END ######################################

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

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

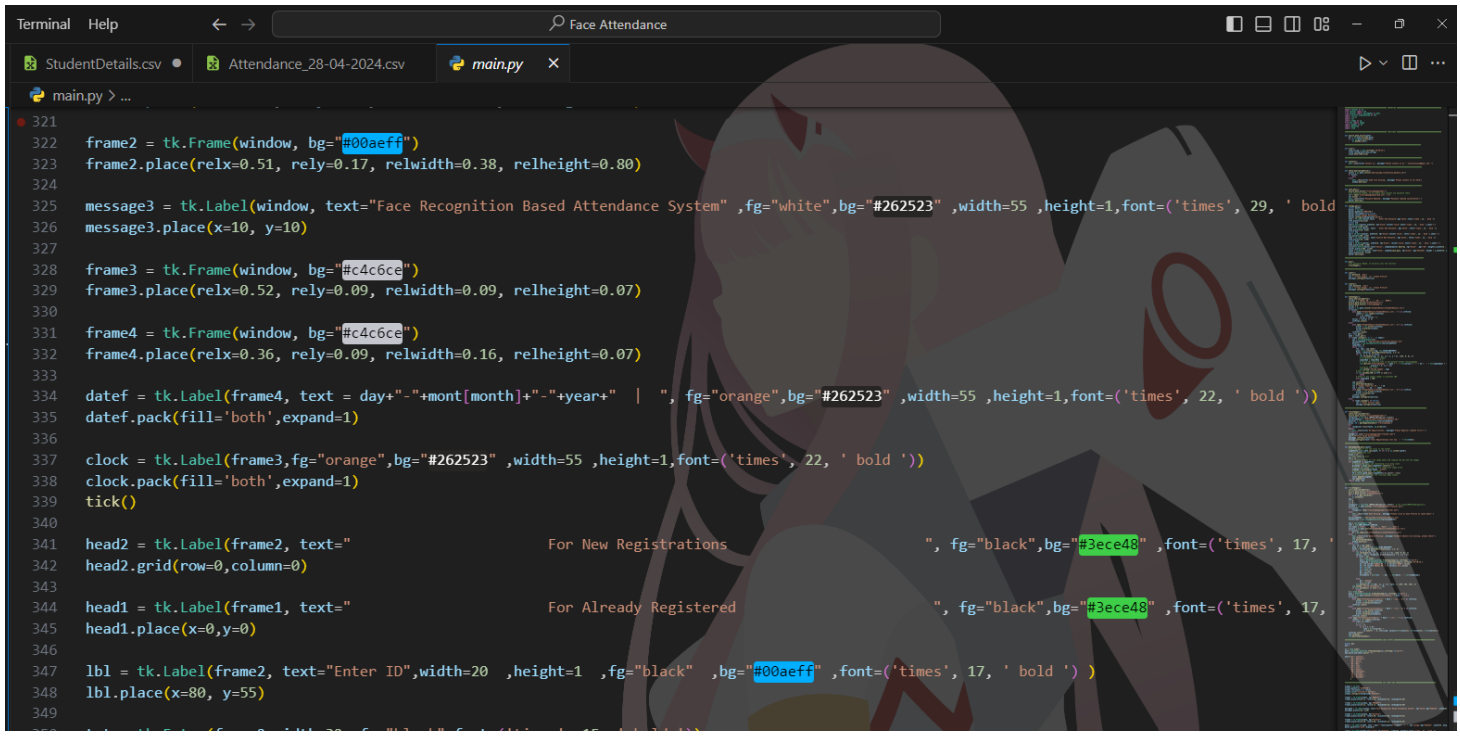
SNAPSHOTS


File Edit Selection View Go Run Terminal Help
Face Attendance
EXPLORER
FACE ATTENDANCE
Attendance
Attendance_28-04-2024.csv
StudentDetails
StudentDetails.csv
TrainingImage
TrainingImageLabel
psd.txt
Trainner.yml
Attendance.zip
haarcascade_frontalface_defaul...
LICENSE
main.py
StudentDetails.csv
main.py
Attendance_28-04-2024.csv
def TakeImages():
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

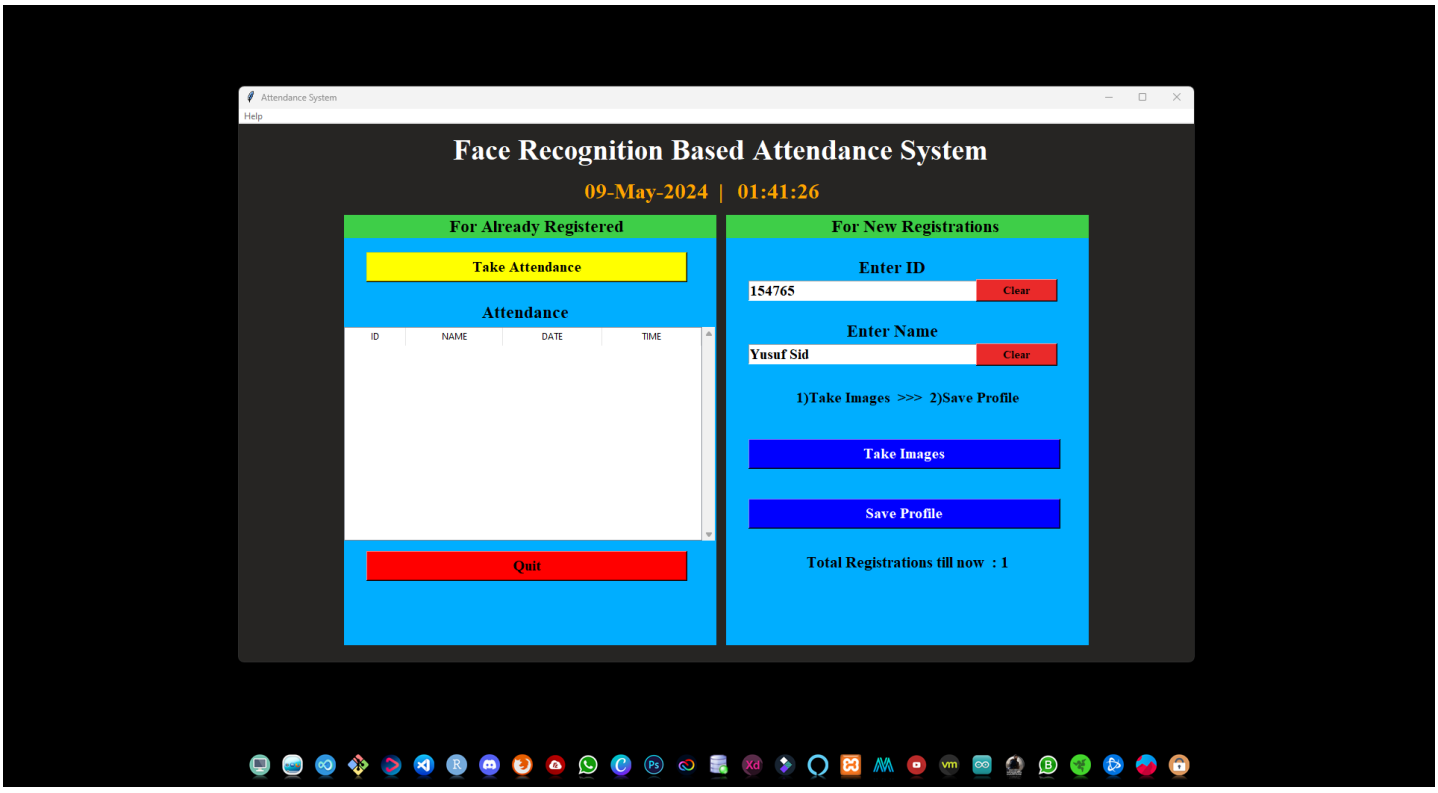

StudentDetails.csv
Attendance_28-04-2024.csv
Trainner.yml
TrainingImageLabel > Trainner.yml
%YAML:1.0
---
opencv_lbphfaces:
threshold: 1.7976931348623157e+308
radius: 1
neighbors: 8
grid_x: 8
grid_y: 8
histograms:
- !!opencv-matrix
rows: 1
cols: 16384
dt: f
data: [3.31632644e-02, 1.02040814e-02, 0., 2.55102036e-03,
1.91326533e-02, 1.27551018e-03, 7.65306130e-03,
2.42346935e-02, 2.55102036e-03, 0., 0., 1.53061226e-02,
1.27551018e-03, 1.14795920e-02, 2.16836724e-02,
8.92857090e-03, 6.37755077e-03, 1.27551018e-03,
3.82653065e-03, 3.82653065e-03, 1.27551018e-03,
2.55102036e-03, 1.27551018e-03, 1.40306121e-02,
2.55102036e-03, 0., 1.27551018e-03, 7.14285672e-02,
1.27551018e-03, 7.14285672e-02, 4.97448966e-02,
2.55102036e-03, 0., 0., 1.27551018e-03, 0., 0., 0., 0., 0.,
0., 0., 0., 0., 1.27551018e-03, 0., 1.02040814e-02,
2.55102036e-03, 0., 1.27551018e-03, 0., 0., 0., 0.,
1.91326533e-02, 2.55102036e-03, 0., 0., 3.69897969e-02, 0.,
5.61224483e-02, 1.78571418e-02, 7.65306130e-03, 0., 0., 0.,
1.27551018e-03, 0., 1.27551018e-03, 1.27551018e-03, 0., 0.,
0., 0., 2.55102036e-03, 0., 2.55102036e-03, 1.27551018e-03,
1.27551018e-03, 0., 0., 0., 0., 0., 0., 0., 0.,

SNAPSHOTS


Terminal Help
Face Attendance
StudentDetails.csv
Attendance_28-04-2024.csv
haarcascade_frontalface_default.xml
haarcascade_frontalface_default.xml
<haarcascade_frontalface_default type_id="opencv-haar-classifier">
<stages>
<_>
<trees>
<_>
<_>
<!-- tree 6 -->
<_>
<!-- root node -->
<feature>
<rects>
<_>5 8 12 6 -1.</_>
<_>5 11 12 3 2.</_></rects>
<tilted>0</tilted></feature>
<threshold>2.7340000960975885e-003</threshold>
<left_val>-1.6911929845809937</left_val>
<right_val>0.4400970041751862</right_val></_></_>
<_>
<!-- tree 7 -->
<_>
<!-- root node -->
<feature>
<rects>
<_>11 14 4 10 -1.</_>
<_>11 19 4 5 2.</_></rects>
<tilted>0</tilted></feature>
<threshold>-0.0188590008765459</threshold>
<left_val>-1.4769539833068848</left_val>

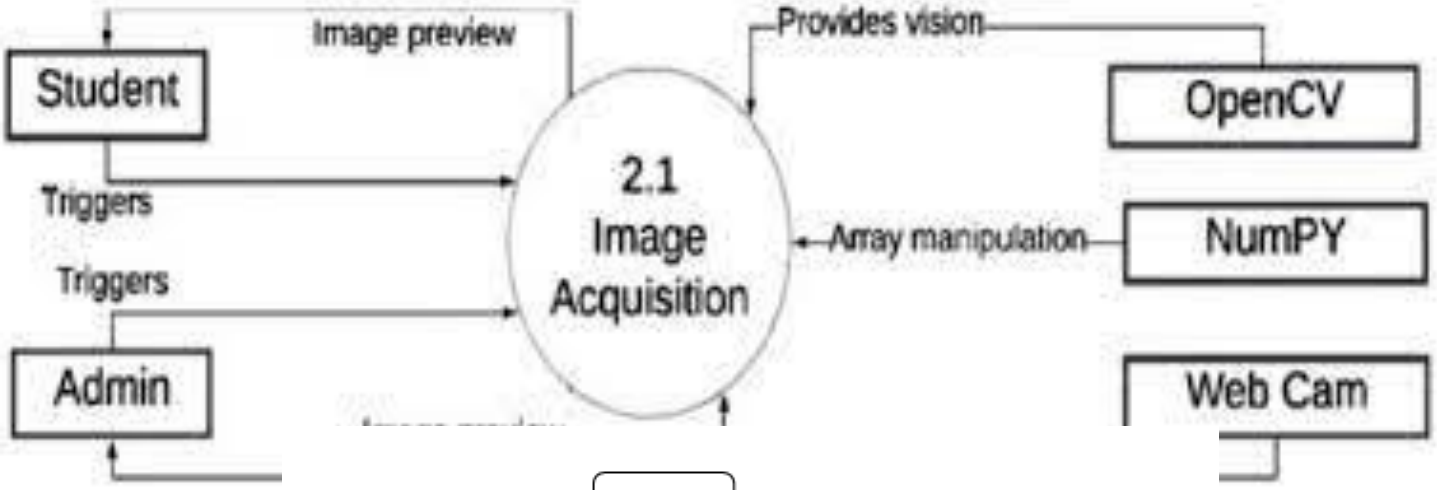

Terminal Help
Face Attendance
StudentDetails.csv
Attendance_28-04-2024.csv
main.py
main.py > ...
frame2 = tk.Frame(window, bg="#00aeff")
frame2.place(relx=0.51, rely=0.17, relwidth=0.38, relheight=0.80)
message3 = tk.Label(window, text="Face Recognition Based Attendance System" ,fg="white",bg="#262523" ,width=55 ,height=1,font=('times', 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="orange",bg="#262523" ,width=55 ,height=1,font=('times', 22, ' bold '))
datef.pack(fill='both',expand=1)
clock = tk.Label(frame3,fg="orange",bg="#262523" ,width=55 ,height=1,font=('times', 22, ' bold '))
clock.pack(fill='both',expand=1)
tick()
head2 = tk.Label(frame2, text="For New Registrations ", fg="black",bg="#3ece48" ,font=('times', 17, '
head2.grid(row=0,column=0)
head1 = tk.Label(frame1, text="For Already Registered ", fg="black",bg="#3ece48" ,font=('times', 17,
head1.place(x=0,y=0)
lbl = tk.Label(frame2, text="Enter ID",width=20 ,height=1 ,fg="black" ,bg="#00aeff" ,font=('times', 17, ' bold '))
lbl.place(x=80, y=55)

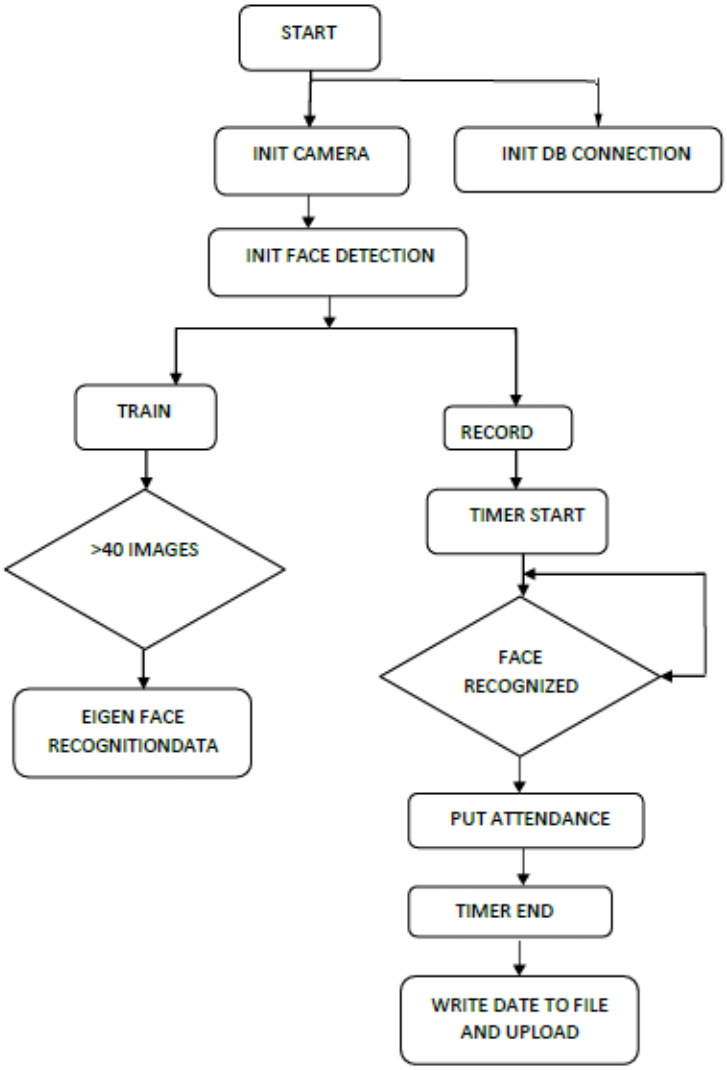
SNAPSHOTS


Attendance System
Help
Face Recognition Based Attendance System
09-May-2024 | 01:41:26
For Already Registered
Take Attendance
Attendance
ID
NAME
DATE
TIME
Quit
For New Registrations
Enter ID
154765
Clear
Enter Name
Yusuf Sid
Clear
1)Take Images >>> 2)Save Profile
Take Images
Save Profile
Total Registrations till now : 1

DFD AND FLOWCHARTS

FUTURE SCOPE





Enhanced Recognition Algorithms: Continuously refining and optimizing the face recognition algorithms can improve accuracy and reliability, leading to more precise attendance tracking.

Integration with Biometric Systems: Integrating our system with biometric authentication methods such as fingerprint or iris scanning can enhance security and authentication capabilities.

Mobile Application Development: Developing a mobile application version of our attendance system can increase accessibility and convenience, allowing users to take attendance on-the-go using their smartphones.

Cloud Integration: Implementing cloud-based storage and processing solutions can enable seamless synchronization of attendance data across multiple devices and locations, enhancing scalability and data accessibility.

Advanced Reporting Features: Introducing advanced reporting features such as customizable attendance reports, analytics dashboards, and automated notifications can provide valuable insights and streamline administrative tasks.

Multi-factor Authentication: Incorporating multi-factor authentication mechanisms, such as OTP (One-Time Password) or biometric verification, can further enhance security and prevent unauthorized access.

Machine Learning Integration: Leveraging machine learning techniques for predictive analytics, anomaly detection, and pattern recognition can enable more intelligent and proactive attendance management.

Integration with Learning Management Systems (LMS): Integrating our attendance system with existing Learning Management Systems can streamline administrative workflows and provide a unified platform for managing student data and attendance records.

CONCLUSION

In conclusion, the development and implementation of the Face Recognition Based Attendance System represent a significant milestone in addressing the challenges of traditional attendance management methods. Through meticulous planning, thorough analysis, and systematic execution, we have successfully crafted a robust and efficient solution to streamline attendance tracking processes.

Our project offers not only a simplified approach to attendance recording but also enhances security, accuracy, and efficiency. By harnessing cutting-edge technologies such as face recognition and machine learning, we have engineered a system that provides reliable and tamper-proof attendance tracking capabilities.

Moreover, the intuitive design and user-friendly interface of the graphical user interface (GUI) ensure accessibility and ease of use for both administrators and end-users. The system's scalability and adaptability allow for seamless integration with existing infrastructure and future expansions.

While our project has achieved its primary objectives, there is still ample room for further refinement and enhancement. Future iterations of the system could explore advanced functionalities such as biometric integration, cloud-based storage, and predictive analytics to elevate its capabilities and user experience.

In summary, the Face Recognition Based Attendance System holds immense potential for transforming attendance management practices in educational institutions and organizations. With ongoing innovation and development, it stands poised to become a cornerstone of modern attendance tracking solutions, driving efficiency, accountability, and productivity across various sectors.

REFERENCES

1. "OpenCV Library." OpenCV, https://opencv.org/.
2. "tkinter Documentation." Python Software Foundation, https://docs.python.org/3/library/tkinter.html.
3. "PIL (Python Imaging Library)." PythonWare, https://pythonware.com/products/pil/.
4. "NumPy Documentation." NumPy, https://numpy.org/doc/.
5. "Pandas Documentation." pandas, https://pandas.pydata.org/docs/.
6. "datetime Module." Python Software Foundation, https://docs.python.org/3/library/datetime.html.
7. "Time Module." Python Software Foundation, https://docs.python.org/3/library/time.html.
8. "CSV Module." Python Software Foundation, https://docs.python.org/3/library/csv.html.
9. "LBPH (Local Binary Patterns Histograms) Face Recognizer." OpenCV, https://docs.opencv.org/3.4/d8/d8c/tutorial_lbph_faces.html.
10. "Haar Cascade Classifier." OpenCV, https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html.

BIO DATA OF TEAMMATES

Name: - Md Yusuf Siddique (**Team Leader**)

Branch:- Diploma in Computer Science & Engineering

College:- Integral University Polytechnic

Mobile. No:-+91 8887656434

Address:-Kushinagar (up)

E-mail:-yusufsidd87842@gmail.com

Name: - Tamanna Parween

Branch:- Diploma in Computer Science & Engineering

College:- Integral University Polytechnic

Mobile. No:-+91 8541929720

Address:-Siwan (Bihar)

E-mail:- parweentamanna519@gmail.com

Name: - Mod Sameer

Branch:- Diploma in Computer Science & Engineering

College:- Integral University Polytechnic

Mobile. No:-+91 9076669826

Address:-Adil Nagar (Lucknow)

E-mail:- Modhsameer03@gmail.com

Name: - Jyotshna Singh

Branch:- Diploma in Computer Science & Engineering

College:- Integral University Polytechnic

Mobile. No:-+91 7903663698

Address:-Gopalganj (Bihar)

E-mail:- jyotshnasingh663@gmail.com