VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

CHENNAI

Title:

Face Recognition Attendance System Using LBPH and Open CV

To:

Reena Roy R

By:

Chikkam Venkat Satya Dhiraj 21BCE5717

Abstract:

The Facial Recognition Attendance System (FRAS) revolutionizes attendance tracking across various sectors, including educational institutions, businesses, and organizations. Leveraging cutting-edge technology like LBPH (Local Binary Pattern Histogram), FRAS streamlines the attendance process by automatically identifying individuals based on their facial features. This eliminates the need for manual attendance taking, saving valuable time and resources.

At the core of FRAS's effectiveness lies its sophisticated algorithms, which meticulously analyze facial characteristics to ensure precise identification. By harnessing the power of advanced computer vision, FRAS can distinguish between individuals with remarkable accuracy, mitigating the risk of attendance errors or malpractice.

One of the key benefits of FRAS is its ability to enhance operational efficiency. By automating attendance tracking, FRAS relieves administrative staff from the repetitive task of recording attendance manually. This not only reduces the burden on personnel but also minimizes the likelihood of human errors, leading to more reliable attendance records.

Moreover, FRAS facilitates prompt decision-making by generating real-time attendance reports. These reports offer insights into attendance patterns, trends, and discrepancies, empowering administrators to take timely actions as needed. Whether it's monitoring student attendance, tracking employee punctuality, or ensuring compliance with regulations, FRAS equips organizations with valuable data for informed decision-making.

The versatility of FRAS extends across various industries and sectors. In educational settings, FRAS can streamline attendance management for schools, colleges, and universities, fostering a more efficient learning environment. In the corporate world, FRAS aids in workforce management, helping businesses track employee attendance accurately and effortlessly. Additionally, FRAS finds applications in security and access control, where it enhances safety measures by verifying individuals' identities in real-time.

Despite its myriad benefits, FRAS also acknowledges the importance of privacy and ethical considerations. With growing concerns about data security and individual rights, FRAS prioritizes responsible deployment practices. It adheres to stringent privacy protocols, ensuring that facial recognition data is handled securely and ethically. Furthermore, FRAS incorporates features like consent mechanisms and data anonymization to safeguard user privacy and uphold ethical standards.

In summary, the Facial Recognition Attendance System (FRAS) represents a paradigm shift in attendance tracking, offering unparalleled efficiency, accuracy, and convenience across diverse industries. By harnessing advanced technology while adhering to privacy and ethical principles, FRAS paves the way for a more streamlined and responsible approach to attendance management.

Introduction:

The Face Recognition Attendance System (FRAS) is set to revolutionize attendance management by offering a seamless, automated solution that eliminates the need for manual data entry. By utilizing sophisticated facial recognition technology, FRAS identifies individuals with precision, drastically reducing the chances of errors or inaccuracies that often plague traditional attendance methods. This cutting-edge system leverages advanced algorithms to ensure high reliability and consistency in tracking attendance, providing organizations with a robust and efficient tool for managing their records.

With FRAS, operational efficiency receives a significant boost. Traditional attendance processes often require manual input, which can be time-consuming and prone to human error. FRAS overcomes these challenges by automating the identification process, allowing organizations to allocate resources to

more productive tasks. This leads to a reduction in administrative overhead and creates a more streamlined workflow.

One of the standout features of FRAS is its ability to mitigate attendance malpractice. By automating the identification process, FRAS reduces the potential for attendance fraud or manipulation. This is especially important in educational environments, where accurate attendance tracking is crucial for both regulatory compliance and student success. With FRAS, institutions can maintain integrity in their attendance records, ensuring a fair and transparent system.

Real-time reporting is another key advantage of FRAS. The system generates up-to-the-minute attendance reports, enabling quick decision-making and providing administrators with valuable insights into attendance patterns. This real-time capability helps schools and other educational institutions identify trends, monitor student engagement, and address issues promptly. By offering immediate access to attendance data, FRAS empowers educators and administrators to take proactive measures to enhance the learning environment.

The implementation of FRAS in educational settings has the potential to transform how attendance is managed. By reducing administrative tasks and providing reliable data, educators can focus more on teaching and supporting students. Furthermore, the accuracy and efficiency of FRAS contribute to a safer and more secure environment, as the system can also be integrated with security and access control measures.

In summary, the Face Recognition Attendance System (FRAS) is a groundbreaking solution that promises to reshape attendance tracking in education. Its automation, accuracy, and real-time reporting capabilities offer numerous benefits, from reducing administrative workload to improving the integrity of attendance records. As educational institutions embrace this innovative technology, FRAS stands poised to become a key component in creating efficient and effective learning environments.

Literature Review:

1. Thai-Viet Dang - Smart Attendance System based on improved Facial Recognition -2023. The study introduces a streamlined deep learning technique for facial recognition, employing an upgraded Face Net model with MobileNetV2 backbone and SSD subsection. Utilizing depth-wise separable convolution, it achieves high accuracy and processing speed, ideal for practical use on low-capacity hardware, paving the way for smart automated attendance systems.
2. Face Recognition based Attendance System using Haar Cascade and Local Binary Pattern Histogram Algorithm

   This paper addresses the use of various attendance systems, highlighting the superiority of face recognition. It focuses on reducing false positives in face recognition attendance systems through thresholding, achieving a 77% student recognition rate with a 28% false positive rate, while also robustly recognizing individuals with glasses or facial hair, with a 60% recognition rate for unknown persons.
3. Design of Attendance System Based on Face Recognition and Android Platform.

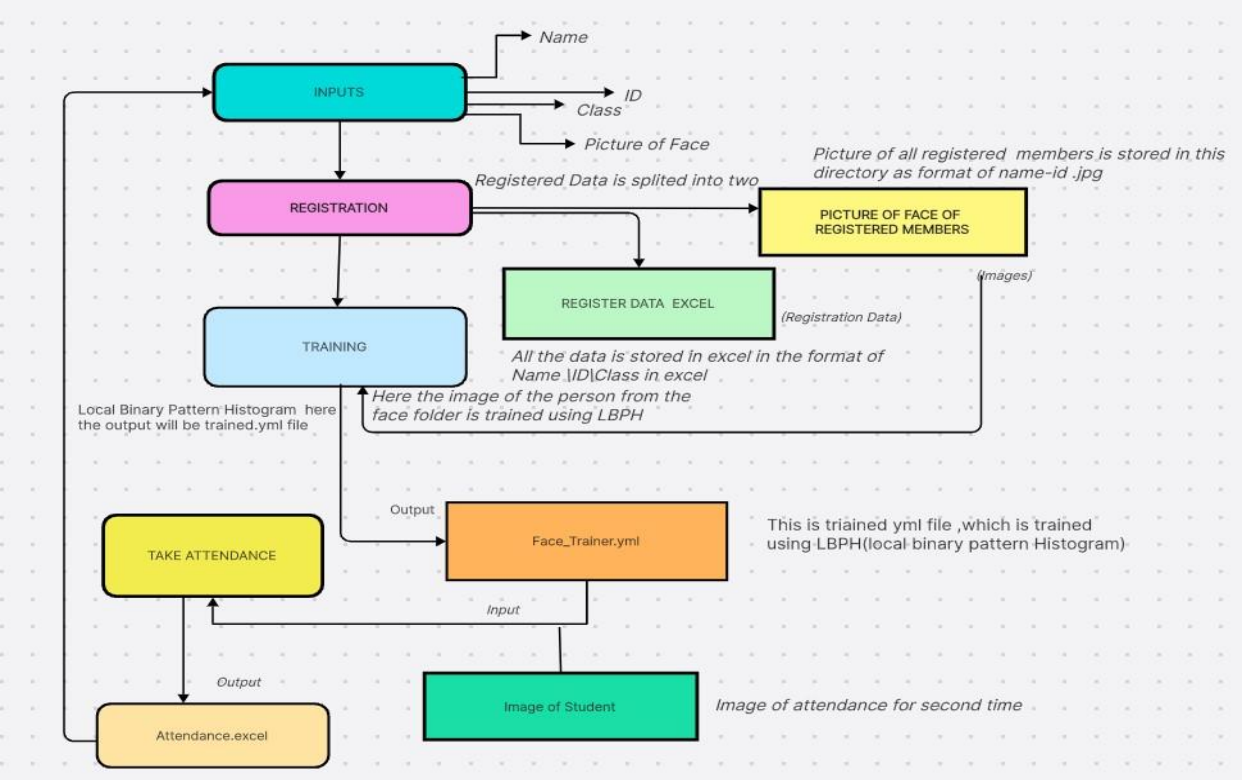
   This paper presents a face recognition-based attendance solution to overcome manual attendance limitations, utilizing mobile platforms and facial recognition technology. It outlines three main modules: information input, sign-in, and record, detailing face detection principles, classifier construction, and Android platform implementation, confirming feasibility through accuracy experiments.
4. Automatic Attendance System Using Face Recognition Technique

   Mayur Surve, Priya Joshi, Sujata Jamadar, Minakshi Vharkate

This paper emphasizes the significance of attendance systems in educational institutions and highlights the challenges of manual methods. By leveraging face recognition technology, it aims to enhance accuracy and efficiency, particularly focusing on implementing a system using the Haar Cascade Algorithm, known for its superior accuracy. This system features a userfriendly interface for image capture, dataset creation, and training, enabling automatic attendance recording with student identification and timestamp.

5. Face Recognition Attendance System Based on Real-Time Video Processing
   This paper investigates the creation of a real-time video-based attendance system employing face recognition technology, influenced by the era of big data and rising commercial interest. It tackles accuracy, stability, truancy reduction, and interface design, providing a fresh solution that improves efficiency and diminishes truancy within educational settings.
6. Student Attendance System using Face Recognition.
   Samridhi Dev; Tushar Patnaik
   This paper introduces an innovative attendance system utilizing cutting-edge face recognition technologies to modernize traditional attendance procedures. Through rigorous testing, it showcases its effectiveness in streamlining attendance processes, ensuring accuracy, and minimizing manual efforts.
7. Real Time Automatic Attendance System for Face Recognition Using Face API and OpenCV.
   Sikandar Khan, Adeel Akram & Nighat Usman
   This paper addresses the inefficiencies of manual attendance tracking by proposing a modern solution utilizing smartphones for faculty members. Leveraging advanced object detection algorithms like YOLO V3 and Microsoft Azure's face API ensures accurate and efficient attendance monitoring, with automatic email notifications for students, parents, and faculty members at the end of each month.
8. Face Recognition Based Smart Attendance System
   A Arjun Raj; Mahammed Shoheb; K Arvind; K S Chethan
   This paper introduces a computerized attendance system leveraging facial recognition technology to address attendance inconsistencies in educational institutions. Utilizing Raspberry Pi, OpenCV, and Dlib, the system ensures accurate attendance tracking, notifies parents of absences via GSM, and provides students with a user-friendly Android application for monitoring their attendance.
9. Face Recognition Smart Attendance System using Deep Transfer Learning.
   Khawla Alhanaee a, Mitha Alhammadi a, Nahla Almenhali a, Maad Shatnawi a
   This paper presents a facial recognition attendance system using deep learning CNNs, leveraging transfer learning for efficient training and achieving high prediction accuracy. It addresses the complexity of attendance management systems by providing a robust solution based on widely-used biometric authentication techniques.

Methodology:



Flow of the project

The Local Binary Pattern Histogram (LBPH) method is a popular technique for face recognition and analysis. It is renowned for its simplicity, effectiveness, and computational efficiency, making it widely used in facial recognition systems like the one used in FRAS. Here's a detailed overview of the LBPH method:

What is LBPH?

LBPH stands for Local Binary Pattern Histogram. It is a texture-based approach to facial recognition that was originally developed for texture classification but has since been applied extensively in face recognition. The method works by analyzing the local patterns of pixels in an image and converting them into a histogram that represents the image's texture. This histogram can then be used for comparison and recognition tasks.

How LBPH Works:

LBPH involves several key steps:

1. Grayscale Conversion: The image is first converted to grayscale, as the LBPH method relies on analyzing pixel intensity. This step simplifies the analysis and reduces computational complexity.

2. Local Binary Patterns (LBP) Calculation: The LBP algorithm divides the image into small regions and compares the pixel intensity in each region to its neighboring pixels. Specifically:

- For each pixel, a binary pattern is created by comparing its intensity to that of its eight neighbors (pixels surrounding it).
- If the central pixel's intensity is greater than or equal to the neighbor's intensity, a 1 is recorded. Otherwise, a 0 is recorded.

- This results in an 8-bit binary code (0 to 255), representing the pattern of relative intensities around the pixel.

3. Region-wise Histograms: The image is divided into a grid of smaller regions (e.g., 8x8). The binary patterns for all pixels within a region are counted to create a histogram for each region. These regional histograms capture the texture information for different parts of the face.

4. Concatenation of Regional Histograms: The histograms from all regions are concatenated to form a single histogram representing the entire image. This comprehensive histogram is the "fingerprint" that represents the unique texture features of the face.

5. Comparison and Recognition: The resulting histogram can then be compared to histograms from a database of known faces to determine the closest match. Typically, a distance metric such as the ChiSquare distance or the Euclidean distance is used for this comparison. The smaller the distance, the more similar the two faces are.

Advantages of LBPH:

1. Simplicity and Efficiency: LBPH is computationally lightweight, making it suitable for real-time applications and environments with limited computing resources.

2. Robustness: It can handle variations in lighting and facial expressions to some extent, providing reliable results across different conditions.

3. Flexibility: LBPH works with grayscale images, reducing the need for complex preprocessing, and it can be adapted to various grid and histogram configurations to suit different applications.

Applications of LBPH: • LBPH is used in a variety of

applications, including:

- Facial Recognition Systems: For identifying and verifying individuals in attendance systems, access control, and security.
- Biometrics: In biometric systems where texture analysis plays a key role.
- Object Recognition: In applications that require texture-based analysis.

Overall, LBPH is a powerful yet straightforward method for facial recognition, offering a good balance of accuracy and computational efficiency. Its robust design and flexible nature make it a popular choice in many applications, from security systems to automated attendance tracking.

OpenCV:

OpenCV (Open Source Computer Vision Library) is an open-source computer vision and machine learning software library. It provides a comprehensive set of tools and functions for computer vision and image processing tasks. Developed by Intel in 1999, OpenCV has grown into one of the most widely used libraries for computer vision applications, with support for multiple programming languages, including C++, Python, Java, and MATLAB/Octave. Below is a detailed description of OpenCV, its features, and common applications. **Key Features of OpenCV**

- Rich Functionality: OpenCV offers a vast array of functions for tasks like image processing, feature detection, object tracking, machine learning, and more. It is designed to be easy to use, with intuitive functions and a comprehensive documentation.
- Cross-Platform Compatibility: OpenCV is cross-platform and can run on various operating systems, including Windows, Linux, macOS, Android, and iOS.

- Real-Time Performance: OpenCV is optimized for real-time computer vision applications. It includes support for parallel processing and GPU acceleration, allowing developers to create high-performance applications.
- Community and Ecosystem: OpenCV has a large community of developers and contributors, leading to extensive third-party support, tutorials, and resources. Its ecosystem includes additional libraries and tools that integrate with OpenCV to extend its functionality.

Core Components of OpenCV:

- Core Module: This module provides the basic data structures and functions for matrix operations, mathematical operations, random number generation, and other foundational elements.
- Image Processing Module: This module contains functions for common image processing tasks like resizing, cropping, rotation, color conversion, filtering, and morphological operations.
- Video Analysis Module: Includes functions for video capturing, frame-by-frame processing, background subtraction, object tracking, and optical flow analysis.
- Feature Detection and Description Module: Provides various algorithms for feature detection and description, such as SIFT, SURF, ORB, FAST, and BRIEF.
- Machine Learning Module: Contains a collection of machine learning algorithms, including decision trees, random forests, k-nearest neighbors, support vector machines, and neural networks.
- Deep Learning Module (DNN): This module enables the use of pre-trained deep learning models within OpenCV, allowing developers to perform tasks like object detection, image segmentation, and facial recognition.
- High-Level Graphical Functions: Includes functions for drawing shapes, text, and graphical overlays on images and videos.

Common Applications of OpenCV:

- Image Processing: Tasks like image filtering, enhancement, edge detection, and transformation are commonly performed using OpenCV.
- Object Detection and Recognition: OpenCV is widely used for object detection applications, such as facial recognition, vehicle detection, and license plate recognition.
- Computer Vision for Robotics: OpenCV is employed in robotics for tasks like SLAM (Simultaneous Localization and Mapping), robot vision, and autonomous navigation.
- Augmented Reality (AR): OpenCV can be used to develop AR applications by overlaying virtual objects onto real-world images.
- Medical Imaging: OpenCV finds applications in medical imaging for tasks like image analysis, medical scans processing, and automated diagnosis.
- Automotive and Transportation: OpenCV is used in autonomous vehicles for lane detection, traffic sign recognition, and obstacle detection.

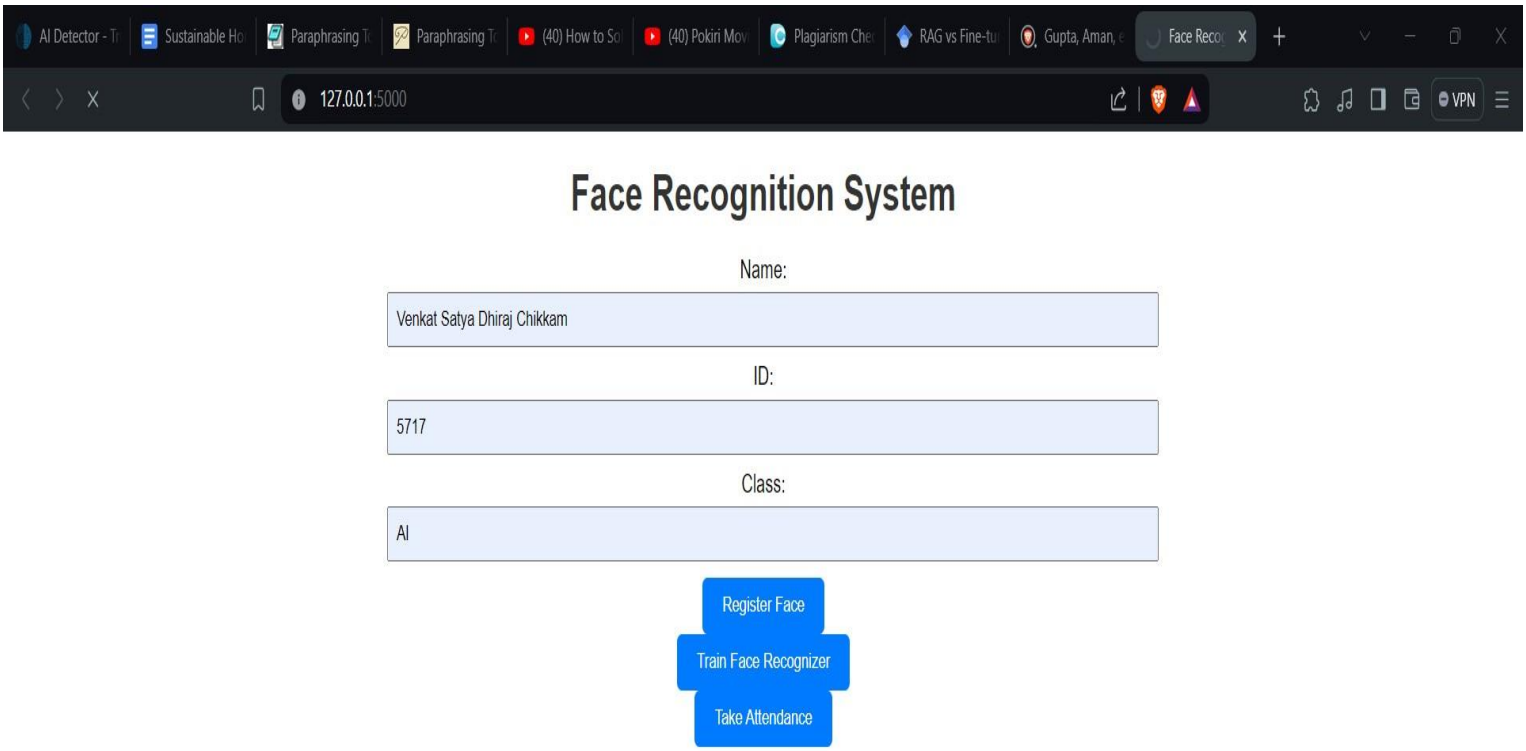
Getting Started with OpenCV:

- Installation: OpenCV can be installed using package managers like pip for Python, or by building from source for C++ and other platforms.

- Learning Resources: There are numerous tutorials, books, and online courses available to learn OpenCV. The official OpenCV documentation provides detailed explanations and examples for most functions and modules.
- Community Support: OpenCV has a large and active community. Online forums, GitHub repositories, and other resources offer support for troubleshooting and guidance on OpenCVrelated projects.

In summary, OpenCV is a versatile and powerful library for computer vision and image processing applications. Its rich functionality, cross-platform compatibility, and extensive community support make it a popular choice among developers, researchers, and engineers in various industries.

Results:



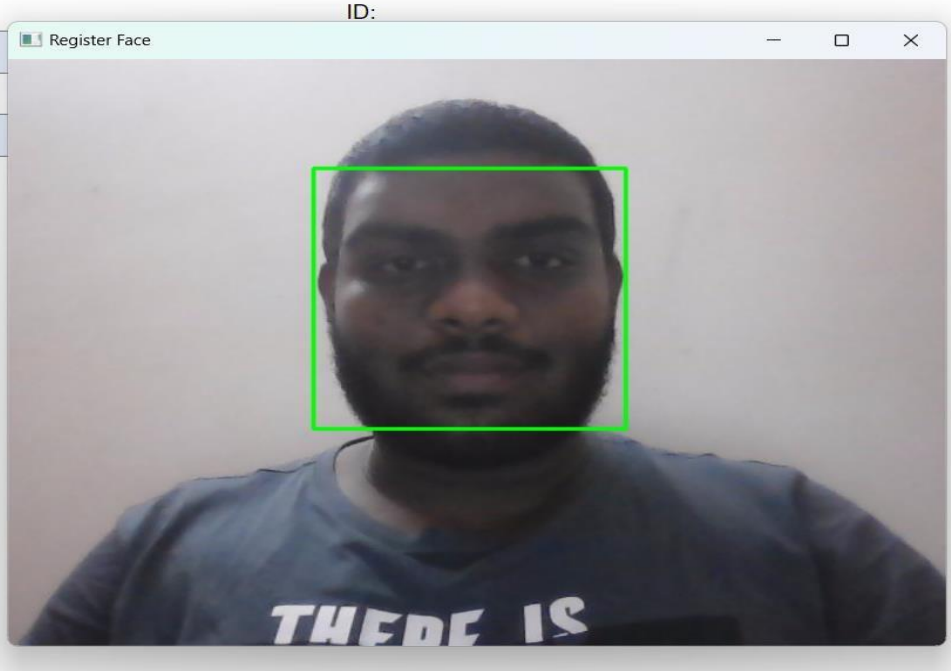
Name:

Venkat Satya Dhiraj Chikkam

ID:

5717

AI

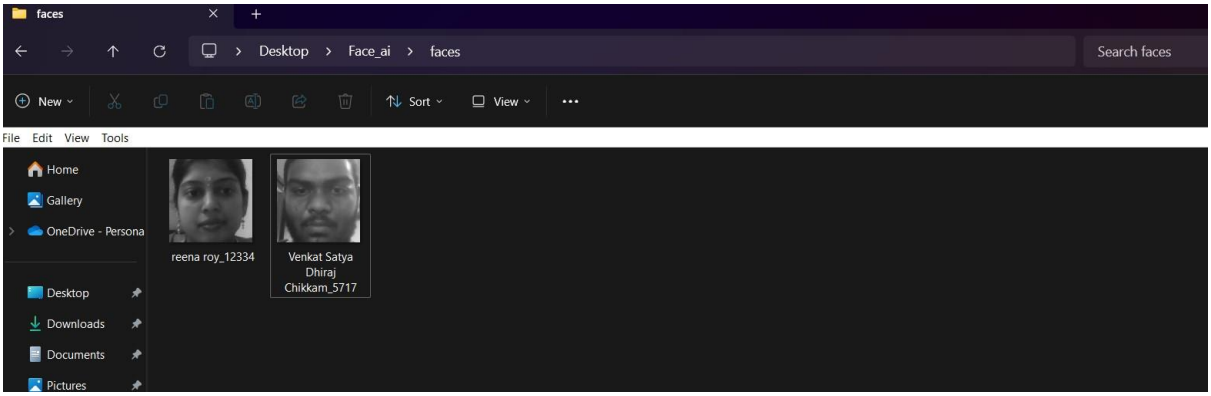


	A	B	C
1	Name	ID	Class
2	reena roy	12334	AI
3	Venkat Satya Dhiraj	5717	AI
4			
5			
6			

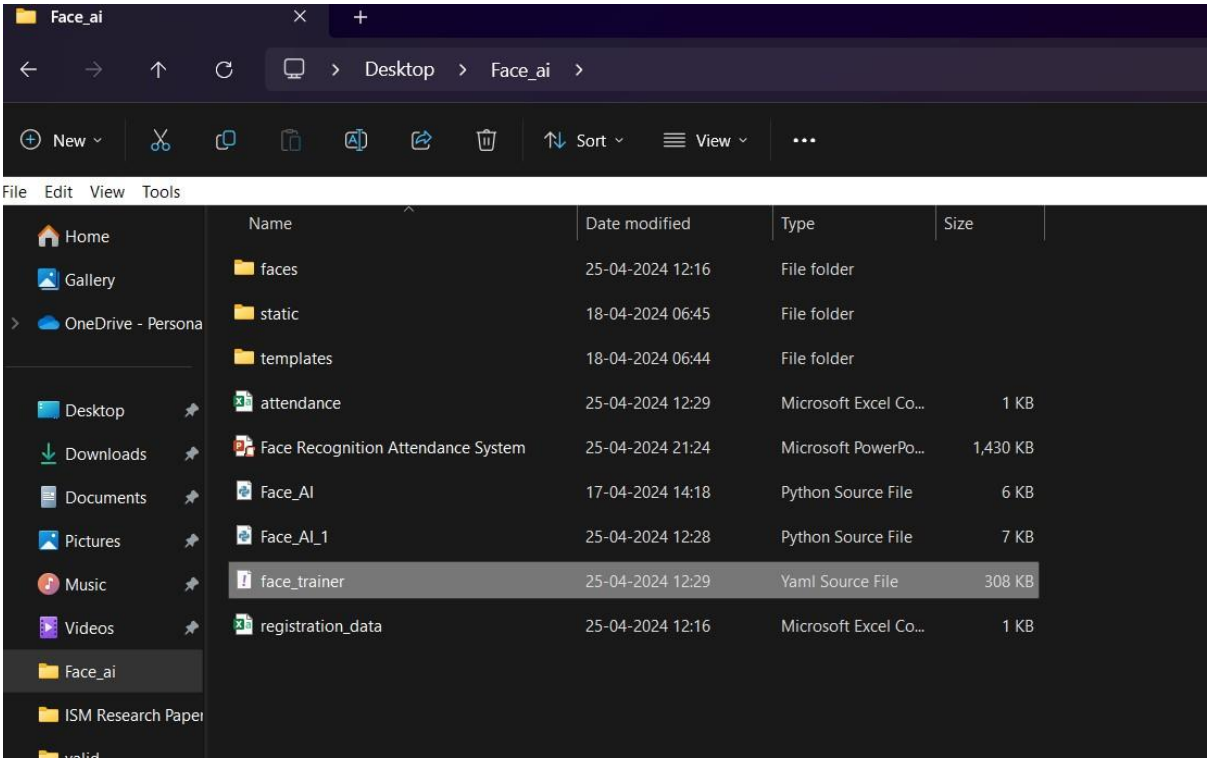
Registration excel

	A	B
1	Name	24-04-2024
2	reena roy	
3	Venkat Satya Dhiraj	Absent
4		
5		
6		

Attendance excel



Faces are registered in the face folder then these are used for training



Face_trainer is the folder generated after training

Conclusion:

The Facial Recognition Attendance System (FRAS) is a transformative solution for attendance tracking in educational institutions, businesses, and organizations. Utilizing Local Binary Pattern Histogram (LBPH) technology, FRAS automates the identification process by recognizing individuals based on their facial features. This automation eliminates the need for manual attendance recording, saving time and reducing errors.

FRAS's sophisticated algorithms ensure accurate identification, reducing attendance malpractice risks. By automating attendance tracking, FRAS eases the administrative burden, providing real-time reports that help in decision-making and monitoring attendance trends. This makes it valuable across various industries, from education to corporate environments, and even in security applications.

FRAS is versatile, improving efficiency in schools, colleges, and businesses, and contributing to security through real-time identity verification. Despite its benefits, FRAS acknowledges privacy and ethical concerns, implementing strict protocols to ensure data security, user consent, and ethical practices.

Overall, FRAS revolutionizes attendance management with its blend of advanced technology, efficiency, and commitment to privacy. Its implementation offers a streamlined, accurate, and ethical approach to attendance tracking.

References:

1.Smart Attendance System based on improved Facial Recognition -2023 https://journal.umy.ac.id/index.php/jrc/article/view/16808.

2. Face Recognition based Attendance System using Haar Cascade and Local Binary Pattern Histogram Algorithm
https://ieeexplore.ieee.org/abstract/document/9143046
Bharath Tej Chinimilli; Anjali T.; Akhil Kotturi; Vihas Reddy Kaipu; Jathin Varma Mandapati

3. Design of Attendance System Based on Face Recognition and Android Platform
https://ieeexplore.ieee.org/abstract/document/9239722
Xiaojun Bai; Feihu Jiang; Tianyi Shi; Yuang Wu

4. Automatic Attendance System Using Face Recognition Technique
https://d1wqtxts1xzle7.cloudfront.net/87419263/A2644059120libre.pdf?1655095435=&response-contentdisposition=inline%3B+filename%3DAutomatic_Attendance_System_Using_Face_R.pdf&Expires=1713379081&Signature=DM4hbZtnjSnyMSTR6UsM20P6AETTeYAULsBCVdMb8qc3Q~4CFLB9oZqZJGuCMa78ZHDgSJuSmCDnc6u5usQPymZeU6ZR5Jk4XOQ27JRejVHPlDAE0OQ9oCrfpyiEPmwoPicJq7C2gZfRAkxrPJKeqeU8h~y2dy045El4dUD-57NSxXLWWRDXicRkHlXwYEo~KCPPhwSCnUgijkN6w8JqnHnyvZDXiNFeXTyLCLULtNuov2H-PAXKrq9qZNKP-5H0sp8f6Vbt63APh8YB7WJXD176QmPdCbbIJwsWjyB~juefshcB29U6oS21PMJ~cUFBVJxwaOohXYV9Bgt6wG9w__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA

5. Face Recognition Attendance System Based on Real-Time Video Processing
https://ieeexplore.ieee.org/abstract/document/9138372

6. Student Attendance System using Face Recognition
https://ieeexplore.ieee.org/abstract/document/9215441

7. Real Time Automatic Attendance System for Face Recognition Using Face API and OpenCV https://link.springer.com/article/10.1007/s11277-020-07224-2

8. Face Recognition Based Smart Attendance System
https://ieeexplore.ieee.org/abstract/document/9160184

9. Face Recognition Smart Attendance System using Deep Transfer Learning
https://www.sciencedirect.com/science/article/pii/S1877050921019232

Appendix:

Source code:

```
import cv2 import os import pandas as pd import
numpy as np from datetime import datetime from
flask import Flask, render_template, request
 app =
Flask(__name__)

# Constants
START_TIME = "12:00"
END_TIME = "13:00"

# Function to register a face def
register_face(name, student_id, student_class):
# Create directory if it doesn't exist     if not
os.path.exists("faces"):
        os.makedirs("faces")

    # Initialize OpenCV's face recognizer     face_cascade =
cv2.CascadeClassifier(cv2.data.haarcascades +
'haarcascade_frontalface_default.xml')
video_capture = cv2.VideoCapture(0)
     while
True:
        ret, frame = video_capture.read()         gray =
cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)         faces =
face_cascade.detectMultiScale(gray, scaleFactor=1.1,
minNeighbors=5, minSize=(30, 30))
         for (x, y, w, h) in
faces:
            cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 2)
            cv2.imwrite(f"faces/{name}_{student_id}.jpg", gray[y:y+h, x:x+w])
         cv2.imshow('Register Face',
frame)
         if cv2.waitKey(1) & 0xFF ==
ord('q'):
            break

    print(f"{name}'s face registered successfully!")

    # Save registration information to a CSV file     registration_data
= pd.DataFrame({'Name': [name], 'ID': [student_id],
'Class': [student_class]})
```

```
    registration_data.to_csv('registration_data.csv',  mode='a',  header=not
os.path.exists('registration_data.csv'), index=False)

video_capture.release()
cv2.destroyAllWindows()

# Function to train face recognizer def
train_face_recognizer():     # Load registration data
registration_data = pd.read_csv('registration_data.csv')
    # Initialize OpenCV's face recognizer
face_recognizer = cv2.face.LBPHFaceRecognizer_create()
    # Initialize OpenCV's face detector     face_cascade =
cv2.CascadeClassifier(cv2.data.haarcascades +
'haarcascade_frontalface_default.xml')
     faces =
[]     ids =
[]

    # Load face images and corresponding IDs
for index, row in registration_data.iterrows():
        face_id = row['ID']         face_path =
f'faces/{row["Name"]}_{row["ID"]}.jpg'         if
os.path.exists(face_path):             face_image = cv2.imread(face_path,
cv2.IMREAD_GRAYSCALE)             faces.append(face_image)
ids.append(int(face_id))  # Convert ID to int for compatibility
    # Train the face recognizer
face_recognizer.train(faces, np.array(ids))

    # Save the trained model
    face_recognizer.save('face_trainer.yml')

# Function to take attendance def
take_attendance():
    start_time = datetime.strptime(START_TIME, "%H:%M")
end_time = datetime.strptime(END_TIME, "%H:%M")

    # Load registration data     registration_data =
pd.read_csv('registration_data.csv')
    # Initialize face recognizer     face_recognizer =
cv2.face.LBPHFaceRecognizer_create()
face_recognizer.read('face_trainer.yml')
```

```
    # Initialize OpenCV's face detector     face_cascade =
cv2.CascadeClassifier(cv2.data.haarcascades +
'haarcascade_frontalface_default.xml')
video_capture = cv2.VideoCapture(0)

    # Define attendance data     current_date =
datetime.now().strftime("%Y-%m-%d")     attendance_file =
'attendance.csv'     attendance_data =
pd.DataFrame(columns=['Name', current_date])
     if
os.path.exists(attendance_file):
        attendance_data = pd.read_csv(attendance_file)
     while
True:
        ret, frame = video_capture.read()         gray =
cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)         faces =
face_cascade.detectMultiScale(gray, scaleFactor=1.1,
minNeighbors=5, minSize=(30, 30))
         for (x, y, w, h) in
faces:
            roi_gray = gray[y:y + h, x:x + w]
            # Recognize faces             id_, confidence =
face_recognizer.predict(roi_gray)             if confidence >=
45:
                font = cv2.FONT_HERSHEY_SIMPLEX                 name =
registration_data.loc[registration_data['ID'] == id_]['Name'].values[0]
cv2.putText(frame, f'{name} - {confidence}%', (x, y), font, 1,
(255, 255, 255), 2, cv2.LINE_AA)
                # Mark attendance in Excel sheet if within the time
range                 current_time = datetime.now().strftime("%H:%M")
if start_time.time() <= datetime.strptime(current_time,
"%H:%M").time() <= end_time.time():                     if
name in attendance_data['Name'].values:
                        attendance_data.loc[attendance_data['Name'] == name,
current_date] = 'Present'                     else:
new_row = pd.DataFrame({'Name': [name], current_date:
['Absent']})                         attendance_data =
pd.concat([attendance_data, new_row], ignore_index=True)
else:                     if name not in attendance_data['Name'].values:
new_row = pd.DataFrame({'Name': [name], current_date:
['Absent']})                                         attendance_data  =
pd.concat([attendance_data, new_row], ignore_index=True)
```

```
            cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 2)
        cv2.imshow('Take Attendance',
frame)
        if cv2.waitKey(1) & 0xFF ==
ord('q'):
            break

video_capture.release()
cv2.destroyAllWindows()

    # Save attendance to a CSV file
attendance_data.to_csv(attendance_file, index=False)
# Routes
@app.route("/") def
index():
    return render_template("index.html")

@app.route("/register_face", methods=["POST"]) def
do_register_face():
    name = request.form["name"]     student_id =
request.form["id"]     student_class = request.form["class"]
register_face(name, student_id, student_class)     return
render_template("index.html", message="Face registered
successfully!")

@app.route("/train_face_recognizer") def
do_train_face_recognizer():
    train_face_recognizer()     return render_template("index.html",
message="Face recognizer trained successfully!")

@app.route("/take_attendance") def
do_take_attendance():
    take_attendance()     return render_template("index.html",
message="Attendance taken successfully!")
 if __name__ ==
"__main__":
app.run(debug=True)
```