Attendance Recording System using Facial Recognition

***Project Report***

*Submitted in partial fulfillment of the*

*Requirements for the award of the degree of*

# Major Project - II

In

Fourth year – eighth semester of

**Bachelor of Technology** specialization In

**Big Data**

Under

**Dr. Shaurya Gupta**

By

|  |  |  |
| --- | --- | --- |
| **SAPID** | **Roll No** | **Name** |
| 500086349 | R2142201647 | Khushi Bansal |
| 500084615 | R214220407 | Dhanraj Bhosale |
| 500085562 | R2142201474 | Ritika Kumari |



DATA SCIENCE CLUSTER

SCHOOL OF COMPUTER SCIENCE

UNIVERSITY OF PETROLEUM AND ENERGY STUDIES, BIDHOLI,

DEHRADUN, UTTARAKHAND, INDIA

**TABLE OF CONTENT**

|  |  |  |
| --- | --- | --- |
| S. No. | Content | Page |
| 1 | Abstract | 3 |
| 2 | Introduction | 3-4 |
| 3 | Problem identification | 4 |
| 4 | Existing system issue | 5 |
| 5 | Proposed system design | 5-6 |
| 6 | Algorithm discussed | 6 |
| 7 | Methodology | 7-8 |
| 8 | Conclusion | 9 |
| 9 | References | 9 |

# 1.Abstract

Current attendance recording methods often lack efficiency and accuracy, leading to wasted time, potential errors, and even security concerns. This proposes an innovative solution: an automated attendance system powered by facial recognition technology. Utilizing a webcam, the system captures participant images, identifies faces, and compares them against stored authorized individual encodings. Upon successful recognition, participants are automatically marked present with precise timestamps, ensuring robust documentation.

This system delivers significant advantages:

* Enhanced Efficiency: Instructors regain valuable time previously spent on attendance marking, allowing them to dedicate more focus to teaching.
* Improved Accuracy: By eliminating manual errors and preventing proxy attendance, the system guarantees the authenticity of recorded data.
* Heightened Security: Unauthorized individuals are denied access, fostering a more secure classroom environment.
* User-Friendly Experience: Contactless and seamless participation provides a convenient experience for all involved.

We will be Implementing using Python libraries like face\_recognition and opencv, the system efficiently processes images and performs recognition. A comprehensive attendance list, complete with timestamps for each participant, can be exported for further analysis or reporting.

# 2. Introduction

Imagine a world where attendance no longer requires tedious signing papers or battling noisy roll calls. Imagine a system that effortlessly verifies your presence with a simple glance. This vision becomes reality with Automated Attendance Recording Systems powered by Facial Recognition (FRS).

FRS eliminates the inefficiencies and frustrations of traditional methods, offering a plethora of benefits:

* Effortless Efficiency: Forget tedious manual processes. FRS streamlines attendance, freeing up valuable time for instructors and students.
* Unwavering Accuracy: Say goodbye to human error and proxy attendance. FRS ensures meticulous and reliable records, boosting accountability.
* Enhanced Security: Unfamiliar faces are flagged, fostering a secure environment where only authorized individuals gain access.
* Seamless Convenience: No more scrambling for pens or fumbling with papers. FRS offers a contactless and intuitive experience for everyone.

This technology, implemented using readily available tools like Python libraries, captures images, identifies faces, and compares them with a secure database. Recognized individuals are instantly marked present, with timestamps providing detailed records.

But the journey doesn't end here. Future advancements promise even greater potential:

Machine Learning Mastery: Imagine FRS adapting to diverse lighting conditions with the help of machine learning, ensuring even higher accuracy.

Unmasking Spoofing: Advanced features like liveness detection can thwart attempts to fool the system, further strengthening its security.

Privacy at the Forefront: Ethical considerations and robust data security measures are paramount, ensuring responsible implementation and user trust.

**3. Problem identification**

Accuracy Concerns: Recognition accuracy can be affected by various factors like lighting, head pose, and facial expressions (Abhishek et al., 2018).

Privacy Issues: Data collection and storage raise privacy concerns, requiring careful consideration and ethical guidelines (Prabhakar et al., 2020).

Cost and Technical Requirements: Implementing FR systems may require initial investment in hardware and software, and ongoing maintenance (Molla et al., 2017).

Ethical Considerations: Bias in algorithms and potential discrimination based on facial features demand careful handling (Garvie, 2019).

Comparative Analysis:

Several studies have compared FR with other attendance technologies like RFID cards and fingerprint scanners. While FR often shows comparable or higher accuracy, its user-friendliness and lower risk of physical contact make it a preferred option.

**4. Existing System Issue**

Inefficient and Error-Prone Attendance Recording

Traditional attendance recording methods, such as roll calls and manual signing, suffer from several limitations:

Inefficiency: These methods require significant time and effort from teachers or instructors, diverting their focus from teaching and learning.

Inaccuracy: Manual errors, proxy attendance, and forgetting to sign are common, leading to unreliable data and potential discrepancies.

Security Issues: Physical attendance markers like attendance sheets can be tampered with, and unauthorized individuals might gain access to classrooms.

Lack of Scalability: Manually handling attendance becomes increasingly challenging with larger class sizes, leading to logistical difficulties.

Contact-Based Interaction: Traditional methods involve physical contact with attendance sheets or signing devices, raising concerns during pandemics or hygiene-sensitive environments.

These limitations necessitate a more efficient, accurate, secure, and scalable solution for attendance recording.

**5. Proposed System Design**

The proposed system design based on the code involves several main components or modules:

**1.** **Face Detection:** This module is responsible for detecting faces in the input image or video frame. It uses the OpenCV library to perform face detection, which is a crucial step in the face recognition process.

**2.** **Feature Extraction:** After face detection, the system extracts features from the detected faces. This can be done using various techniques, such as Eigenfaces, Local Binary Patterns (LBP), or Histogram of Oriented Gradients (HOG). In the provided code, the `fac**e\_recognition.face\_encodings()` function is used to extract features from the faces.**

**3. Face Encoding:** The extracted features are then converted into numerical encodings, which can be compared with other face encodings to determine if they match. The `face\_recognition.face\_encodings()` function also returns face encodings, which are stored in the `known\_face\_encoding` list in the provided code.

**4. Face Matching:** The system compares the face encodings of the detected faces with the known face encodings to determine if there is a match. This is done using the `face\_recognition.compare\_faces()` function in the provided code.

**5. Face Recognition:** If a match is found, the system can recognize the face and take appropriate actions, such as marking attendance or granting access.

**6. Database Management:** The system needs to store and manage the known face encodings and corresponding names. In the provided code, this is done using the `known\_face\_encoding` and `known\_faces\_names` lists.

**7. User Interaction:** The system can provide feedback to the user, such as displaying the recognized face and its name on the video frame or providing an alert when a match is found.

**8. CSV File Writing:** The system can also write the recognized faces and their corresponding times to a CSV file for further analysis or reporting.

These components work together to create a facial recognition system that can be integrated into various applications, such as attendance tracking, security systems, or social media

**6. Algorithm Discussed:**

1. The algorithm used in the code is a combination of face detection and face recognition techniques. The code uses the face\_recognition library, which is a Python library for performing face recognition. The library is based on the **Local Binary Patterns Histogram (LBPH) algorithm**, which is a popular face recognition algorithm.

2. The code first loads images of known faces and extracts their face encodings using the face\_recognition library. These encodings are then stored in the `known\_face\_encoding` list.

3. Next, the code performs face detection on the video frames captured from the webcam. It uses the `face\_locations` and `face\_encodings` functions from the face\_recognition library to detect faces and extract their encodings.

4. The extracted face encodings are then compared with the known face encodings using the `compare\_faces` function from the face\_recognition library[6]. If a match is found, the corresponding face name is added to the `face\_names` list, and the attendee's name is displayed on the video frame along with the message "Present".

5.The code also keeps track of the students who have attended the class and writes their attendance to a CSV file.

6.The algorithm used in the code is based on the Local Binary Patterns Histogram (LBPH) algorithm for face recognition and the face\_recognition library for face detection and encoding.

**7. Methodology :**

The program is a Python script that uses the face\_recognition library to perform face recognition on live video from a webcam. Here is a step-by-step explanation of how the code works:

**1. Importing Libraries:** The script starts by importing the necessary libraries, including `cv2` for OpenCV, `numpy` for numerical operations, `csv` for working with CSV files, `os` for interacting with the operating system, and `datetime` for handling date and time.

**2. Initializing Video Capture:** The script then initializes a video capture object by calling `cv2.VideoCapture(0)`, which captures video from the default camera (camera index 0).

**3. Loading Known Faces:** The script loads images of known faces from the `photos` directory using the `face\_recognition.load\_image\_file()` function. For each image, it extracts the face encoding using `face\_recognition.face\_encodings()`. These encodings and corresponding face names are stored in `known\_face\_encoding` and `known\_faces\_names` lists, respectively.

**4. Initializing Variables:** The script initializes empty lists for storing face locations, encodings, and names. It also sets a flag `s` to True, indicating the first frame of the video has been processed.

**5. Creating a CSV File:** The script creates a CSV file with the current date as the filename using `datetime.now().strftime("%Y-%m-%d")`. It then opens the file for writing using `csv.writer()`.

**6. Processing Video Frames:** The script enters a loop that continuously captures video frames using `video\_capture.read()`. For each frame, it resizes the frame to a quarter of its original size using `cv2.resize()`, converts it to RGB format using `cv2.cvtColor()`, and loads it into the `rgb\_small\_frame` variable.

**7. Face Detection:** If the `s` flag is True, the script performs face detection on the resized frame using `face\_recognition.face\_locations()` and `face\_recognition.face\_encodings()`. These functions detect faces and extract their encodings.

**8. Face Matching:** For each detected face, the script compares its encoding to the known face encodings using `face\_recognition.compare\_faces()`. If a match is found, the corresponding face name is added to the `face\_names` list

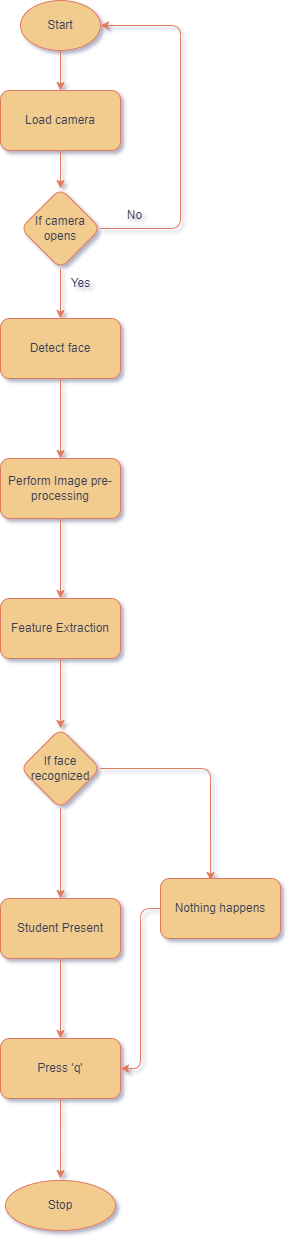
**9. Displaying Results:** The script then displays the detected faces and their names on the video frame using `cv2.putText()`. If a known face is detected, it prints the name of the face to the console.

**10. Writing to CSV:** The script writes the detected faces and their corresponding times to the CSV file using `csv.writer().writerow()`.

**11. User Interaction:** The script displays the video frame using `cv2.imshow()` and waits for a 'q' key press using `cv2.waitKey()`.

**12. Cleaning Up:** Finally, the script releases the video capture object, closes all windows, and closes the CSV file.

In summary, the code uses the face\_recognition library to detect faces in real-time video from a webcam, compares them to known faces, and writes the results to a CSV file.



**8. Conclusion**

The implementation of a facial recognition-based attendance recording system presents a revolutionary advancement in traditional attendance tracking methods. Through its seamless integration of cutting-edge technology, this system offers unparalleled accuracy, efficiency, and convenience. By harnessing the power of facial recognition algorithms, organizations can streamline their attendance management processes, mitigate the risks associated with manual data entry errors, and enhance overall security measures. Moreover, this system fosters a more transparent and accountable environment, facilitating better monitoring and analysis of attendance patterns.

**9. References**

[1] Smitha, Pavithra S Hegde, Afshin: Face Recognition based Attendance Management System, International Journal of Engineering and

Technical Research V9(05), June 2020

[2] Enjie Jiang, “A review of the comparative studies on traditional and intelligent face recognition methods”, (CVIDL), 2020.

[3] Dilpreet Singh Brar, Amit Kumar, Pallavi, Usha Mittal, Pooja Rana, “Face Detection for Real World Application”, IEEE,2021.

[4] Bharath Tej Chinimilli, Anjali T, Akhil Kotturi, h Vihas Reddy Kaipu, Jathin Varma Mandapati “Face Recognition based

Attendance System using Haar Cascade and Local Binary Pattern Histogram Algorithm", IEEE, 2020.Aini, N., Noor, R. M., & Othman, M. S. (2019). A Review of Automated Attendance System Using Biometric: Fingerprints, Irises, and Faces. Journal of Telecommunication, Electronic Computer Engineering, 11(2), 81-86.

[5] Abhishek, K., Agarwal, S., & Gupta, K. (2018). A Deep Learning Framework for Face Recognition with Pose and Illumination Variations. Proceedings of the International Conference on Advances in Computing, Communications and Informatics (ICACCI), 852-857.

[6] Garvie, C. (2019). The Ethics of Facial Recognition. Journal of Applied Philosophy, 36(3), 306-323.

[7] Gadekar, S. P., Patil, S. M., & Yeole, S. S. (2019). Face Recognition Attendance System Using OpenCV. International Journal of Engineering and Advanced Research Technology, 3(4), 501-504.

[8] Hava, A., Kocaman, I., & Keskin, C. (2017). A Mobile Real-time Attendance System Using Face Recognition on Android Platform. International Journal of Engineering and Technology, 9(8), 2963-2967.

[9] Molla, M. I., Hossain, M. B., & Amin, M. B. (2017). A Comparative Study of Biometric Attendance Systems. International Journal of Computer Applications, 176(8), 29-34.

[10] Noor, A. S. M., Zakaria, N. A., & Abdullah, A. H. (2015). An Efficient Attendance Management System Using Facial Recognition Technique. Applied Soft Computing, 37, 840-848.