Lovely Professional University Jalandhar, Punjab, India

**Task:Face Authentication Based Attendance System**

**Submitted By : Bavigadda Meghana**

**Gmail:meghanabaviggada@gmail.com**

# 1. Introduction

## Problem Statement:

Traditional attendance systems such as manual registers and ID card verification are time-consuming and prone to errors and proxy attendance. These methods require physical interaction and continuous supervision, which makes them inefficient for large groups. There is a need for an automated and contactless attendance system that can accurately identify individuals and record their attendance without manual effort.

## Objective of the Project:

The main objective of this project is to develop a face authentication-based attendance system that can register a user's face, recognize the user in real time using a camera, and automatically record attendance with date and time. The system aims to minimize duplicate entries, reduce proxy attendance, and improve efficiency and reliability.

## Project Description:

This project implements a Face Authentication Attendance System using Python, OpenCV, and the face recognition library. The system captures facial images through a live camera feed and converts them into numerical facial encodings. These encodings are stored in a database for future comparison. During recognition, the system matches the detected face with stored encodings and identifies the user. Once a user is successfully authenticated, the system records the user's name along with the current date and time in an attendance file.

## Scope of the Project:

The proposed system is suitable for small to medium-scale environments such as classrooms, offices, and training centers. It provides a contactless and automated alternative to traditional attendance systems. The project focuses on real-time face recognition and basic spoof prevention by using live camera input.

# 2. System Architecture & Workflow

## System Architecture

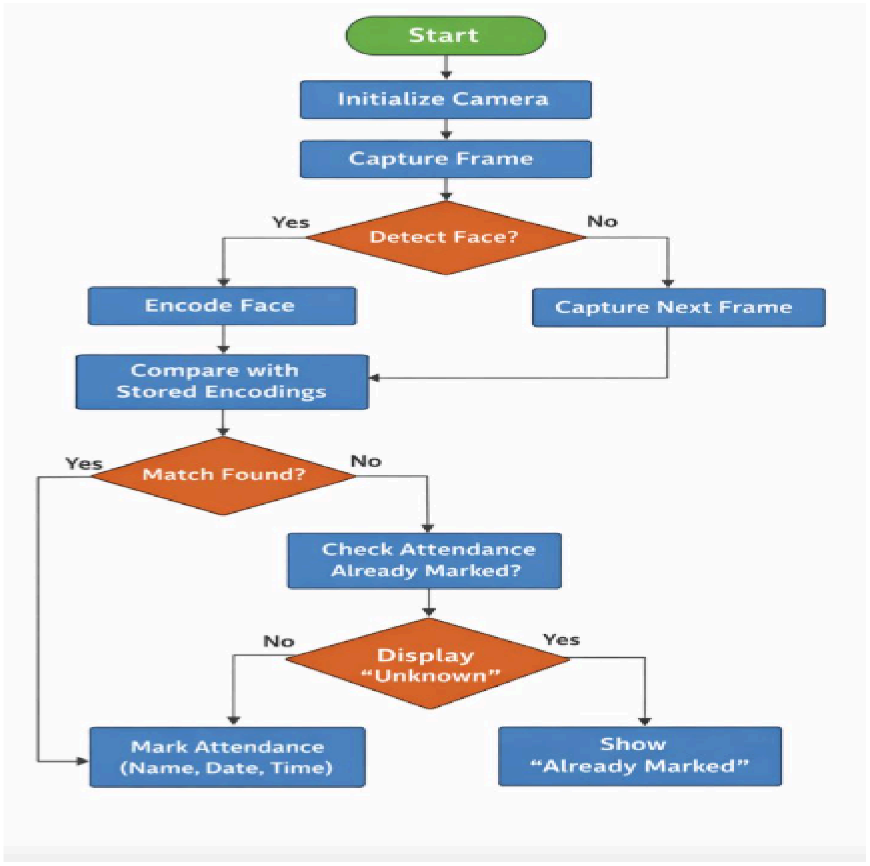
The Face Authentication Attendance System is developed using Python and follows a modular architecture. The system is divided into several functional modules, each responsible for a specific task in the face recognition and attendance process.

The main components of the system are:

1. **Camera Input Module:**
   This module uses OpenCV to capture real-time video frames from the webcam. The captured frames are sent to the face detection module for further processing.
2. **Face Detection and Encoding Module:**
   This module uses the face_recognition library to detect faces in each frame and convert them into numerical feature vectors known as face encodings.
3. **Face Database Module:**
   The generated face encodings along with corresponding user names are stored in a serialized file (pickle file). This database is used as reference data for recognition.
4. **Face Recognition Module:**
   The system compares the face encoding obtained from the live camera feed with the stored encodings in the database using distance-based comparison. The closest match is selected to identify the user.
5. **Attendance Management Module:**
   Once a face is recognized, the system records the user's name, current date, and time in an attendance file (CSV/Excel format). The system also supports punch-in and punch-out functionality and prevents multiple entries for the same user in a single session.
6. **User Interface Module:**
   The system displays the recognized user name and attendance status on the screen through console output and camera window.

## Workflow of the System

1. The system starts and activates the webcam using OpenCV.
2. Video frames are continuously captured from the camera.
3. Faces are detected from each frame using the face_recognition library.
4. Facial features are extracted and converted into face encodings.
5. The extracted encodings are compared with the stored encodings in the face database.

6. If a match is found, the user is identified and the system checks whether attendance is already marked.
7. If attendance is not marked, the system records the user's name, date, and time as punch-in or punch-out.
8. If no match is found, the face is labeled as "Unknown.
9. The process continues in real time until the system is stopped
10. 

The flowchart represents the working of the Face Authentication Attendance System. The system starts by initializing the camera and capturing video frames. It checks whether a face is detected in the frame. If a face is detected, the system encodes the facial features and compares them with the stored face encodings. If a match is found, the system verifies whether the attendance is already marked. If not marked, the system records the user's name along with date and time. If no face is detected or no match is found, the system continues capturing the next frame.

## 3. Methodology:

**Face Registration Process:**

The face registration process is used to enroll a new user into the system. During this phase, the system captures facial images of the user and generates unique face encodings.

- The user stands in front of the camera.
- The camera captures multiple frames of the user's face.
- The system detects the face region from each frame.
- Facial features are extracted and converted into numerical encodings using the face_recognition library.
- The generated encodings are stored along with the user's name in a serialized file (pickle file).
- These stored encodings form the face database used for future recognition.

**Face Recognition Process:**

The recognition process identifies a user by comparing live camera input with the stored face database.

- The webcam captures live video frames.
- Faces are detected from each frame.
- The detected face is encoded into a feature vector.
- The extracted encoding is compared with the known encodings stored in the database.
- The distance between encodings is calculated to determine the closest match.
- If the distance is within the threshold limit, the face is recognized and the corresponding user name is retrieved.
- If no suitable match is found, the system labels the face as "Unknown."

**Attendance Marking Process**

- Once a face is recognized, the attendance module records the user's details.

- The system checks whether the recognized user’s attendance is already marked for the current session.
- If attendance is not marked, the system records the user’s name, current date, and current time as punch-in or punch-out.
- If attendance is already marked, the system displays a message indicating that the attendance has already been recorded.
- The attendance data is stored in a CSV or Excel file for future reference.

### Spoof Prevention Approach

The system performs basic spoof prevention by restricting face recognition to live camera input only. Static images or screenshots are not directly used for recognition. This ensures that face authentication is performed in real-time and reduces the possibility of simple spoofing using photographs.

## 4. Tools & Technologies Used

This project is implemented using the following tools and technologies:

### Programming Language

Python: Python is used as the primary programming language due to its simplicity, extensive library support, and suitability for computer vision and machine learning applications.

### Libraries and Frameworks

- OpenCV (cv2): Used for capturing real-time video frames from the webcam and for basic image processing operations.
- face_recognition: Used for detecting faces and generating facial encodings. This library is built on top of dlib and provides high accuracy face recognition functionality.
- NumPy: Used for numerical computations and handling image arrays.
- Pandas:Used to create and manage the attendance file in CSV/Excel format.

- Pickle:Used to store and load the face encodings and user names as serialized files.
- Datetime:Used to fetch the current date and time for marking attendance.

**Development Environment**

- Google Colab / Local System:The project can be executed either on a local machine or on Google Colab with webcam support.
- Operating System:Windows / Linux (any system with camera support

**4.4 File Formats Used**

- Pickle (.pkl): For storing face encodings
- CSV / Excel (.csv / .xlsx): For storing attendance records

# 5. Implementation Details

**Face Data Collection and Storage**

The system captures facial images through a webcam and processes them using the face_recognition library. For each detected face, facial features are extracted and converted into numerical face encodings. These encodings are associated with the user's name and stored in a serialized pickle file. This file acts as the face database and is loaded during the recognition phase.

**Face Detection and Encoding**

The face detection process is carried out using the face_recognition.face_locations() function, which identifies the coordinates of faces in a video frame. The detected face regions are then passed to the face_recognition.face_encodings() function to generate a 128-dimensional feature vector representing the facial characteristics of the user.

### Face Matching and Identification

During recognition, the system compares the encoding obtained from the live camera feed with the encodings stored in the face database. The comparison is performed using distance-based matching. The system selects the closest match and verifies whether the distance is within an acceptable threshold. If a valid match is found, the corresponding user name is retrieved; otherwise, the face is labeled as “Unknown.”

### Attendance File Handling

The attendance is recorded using a CSV or Excel file. When a user is recognized, the system checks whether their attendance has already been marked in the current session. If not, a new entry is added with the user’s name, current date, and current time. This prevents duplicate attendance entries and supports punch-in and punch-out functionality.

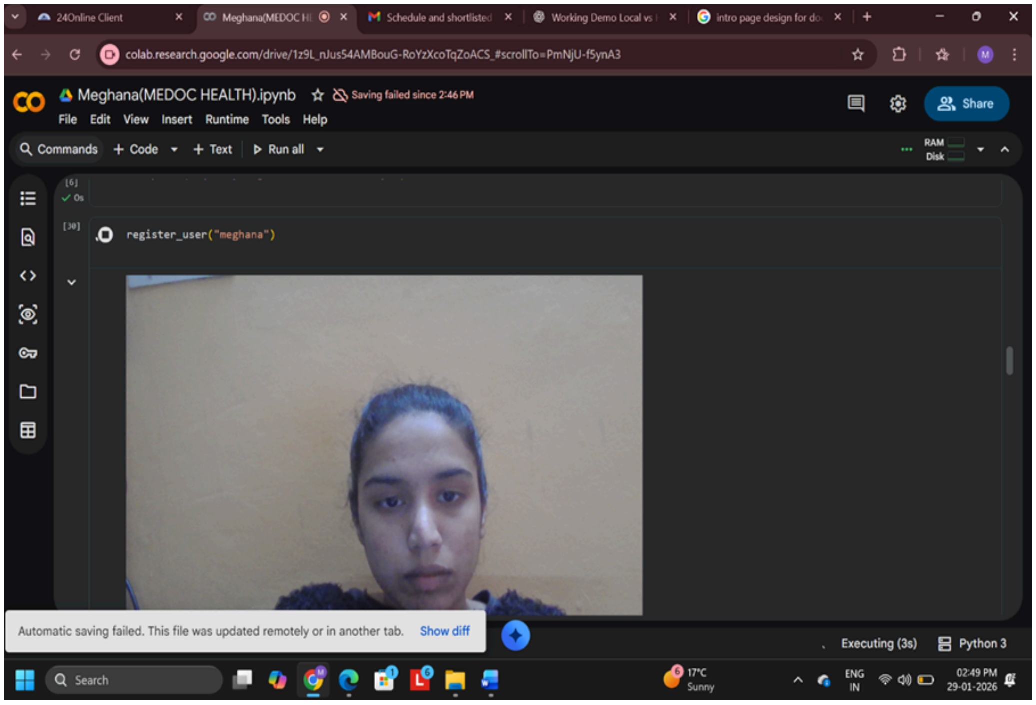
### Real-Time Processing

The system continuously captures frames from the webcam in a loop. Each frame is processed independently for face detection and recognition. The recognized user’s name and status are displayed in real time on the video window. The process continues until the user terminates the program.

# 6. Results & Output

### Face Recognition Output

When a registered user appears in front of the camera, the system detects the face and displays the user’s name on the video frame. If the detected face matches one of the stored face encodings, the system identifies the user correctly. If the face is not present in the database, the system labels it as “Unknown.”


Meghana(MEDOC HEALTH).ipynb
Saving failed since 2:46 PM
File Edit View Insert Runtime Tools Help
Commands
+ Code
+ Text
Run all
Share
Executing (3s)
Python 3

```
register_user("meghana")
```

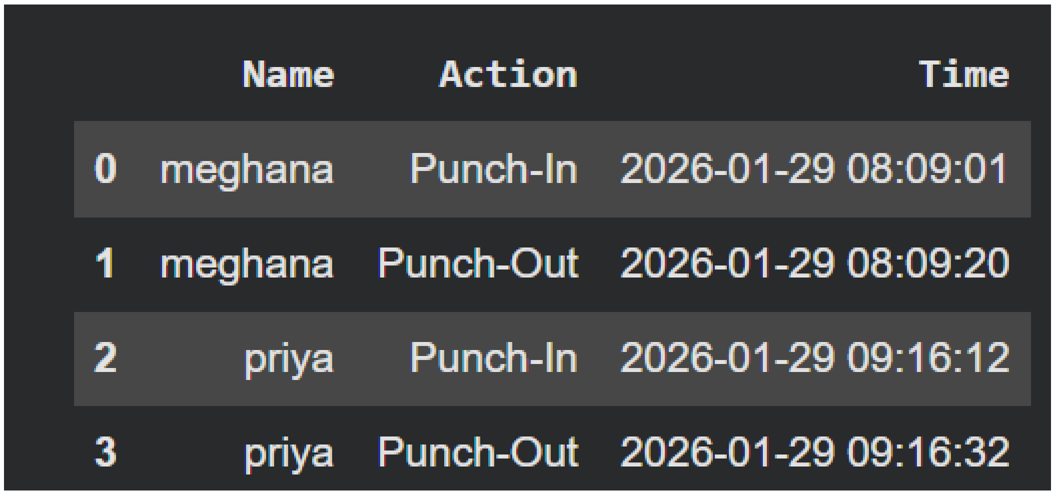
Automatic saving failed. This file was updated remotely or in another tab. Show diff

## Attendance Record Output

Once a user is recognized, the system records the attendance by storing the user's name, current date, and time in an attendance file. The attendance file is generated in CSV or Excel format and can be opened for verification.

## Sample Attendance Table:

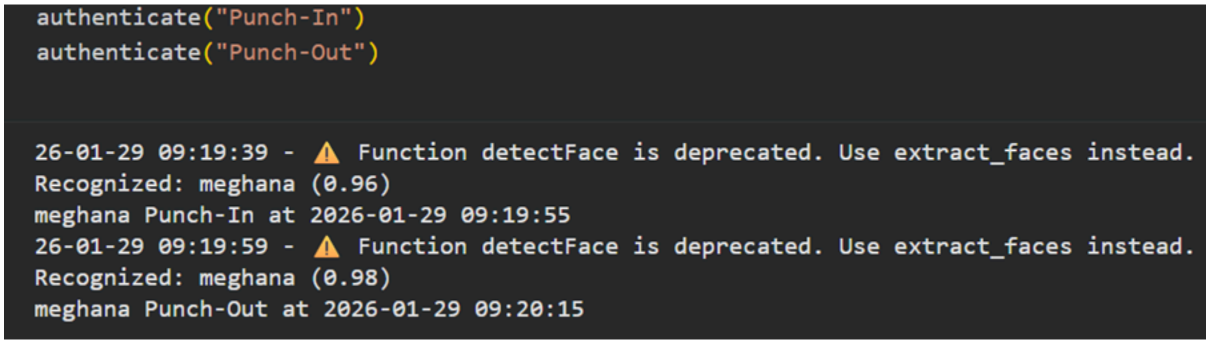
| | Name | Action | Time |
|---|---|---|---|
| 0 | **meghana** | **Punch-In** | **2026-01-29 08:09:01** |
| 1 | **meghana** | **Punch-Out** | **2026-01-29 08:09:20** |
| 2 | **priya** | **Punch-In** | **2026-01-29 09:16:12** |
| 3 | **priya** | **Punch-Out** | **2026-01-29 09:16:32** |



|   | Name | Action | Time |
|---|---|---|---|
| 0 | meghana | Punch-In | 2026-01-29 08:09:01 |
| 1 | meghana | Punch-Out | 2026-01-29 08:09:20 |
| 2 | priya | Punch-In | 2026-01-29 09:16:12 |
| 3 | priya | Punch-Out | 2026-01-29 09:16:32 |

**System Behavior**

- The system successfully recognizes registered users in real time.
- Attendance is marked only once per session to avoid duplicate entries.
- Unknown faces are handled separately and displayed as "Unknown."
- The system works efficiently under normal lighting conditions and standard camera resolution.



```
authenticate("Punch-In")
authenticate("Punch-Out")
```

```
26-01-29 09:19:39 - ⚠ Function detectFace is deprecated. Use extract_faces instead.
Recognized: meghana (0.96)
meghana Punch-In at 2026-01-29 09:19:55
26-01-29 09:19:59 - ⚠ Function detectFace is deprecated. Use extract_faces instead.
Recognized: meghana (0.98)
meghana Punch-Out at 2026-01-29 09:20:15
```

## 7. Accuracy & Performance Analysis

### Recognition Accuracy

The system achieves good recognition accuracy for registered users under normal lighting conditions and frontal face orientation. Since the system uses the face_recognition library, which is based on deep learning models trained on large face datasets, it is capable of extracting distinctive facial features and matching them effectively.

In controlled conditions (proper lighting and clear face visibility), the system correctly recognizes authorized users and marks their attendance. Unknown users are correctly identified as "Unknown," which prevents unauthorized attendance marking.

### Performance

The system processes video frames in real time using OpenCV and performs face detection and recognition efficiently. The average response time between face detection and attendance marking is low, making the system suitable for real-time applications such as classroom or office attendance.

### Factors Affecting Accuracy:

The accuracy of the system depends on the following factors:

- Lighting conditions
- Camera resolution and quality
- Face orientation and distance from the camera
- Facial expressions and occlusions (mask, spectacles, etc.)
- Under poor lighting or extreme face angles, the recognition accuracy may reduce.

#### Observations

- The system successfully marks Punch-In and Punch-Out time for registered users.
- Duplicate entries are avoided by checking existing attendance records.
- The system performs reliably in real-time with minimal delay.

# 8. Limitations of the System:

Although the Face Authentication Attendance System performs well under normal conditions, it has certain limitations:

- Lighting Sensitivity:The system works best under proper lighting conditions. In very low light ,face detection and recognition accuracy may decrease.
- Face Angle and Distance:The system may fail to recognize faces if the face is not directly facing the camera or if the user is too far from the camera.
- Spoofing Possibility:The system uses basic spoof prevention by relying on live camera input only. However, it does not include advanced liveness detection techniques such as blink detection or depth sensing, which means it may be vulnerable to spoofing using high-quality photographs or videos.
- Camera Dependency:The accuracy of the system depends on the quality and resolution of the camera used. Low-quality cameras may reduce recognition performance.
- Scalability:The system is suitable for small to medium-scale usage. For very large databases with hundreds or thousands of users, the recognition speed may reduce due to increased comparison time.

# 9. Conclusion

This project successfully implements a Face Authentication Attendance System using Python, OpenCV, and the face_recognition library. The system is capable of registering users, recognizing faces in real time, and automatically recording attendance with accurate date and time information.

The proposed system reduces the need for manual attendance marking and minimizes the possibility of proxy attendance. It provides a contactless and efficient solution for attendance management in classrooms and office environments. Experimental results show that the system performs well under normal lighting conditions and standard camera settings.

Although the system has certain limitations related to lighting conditions and spoofing prevention, it demonstrates the practical application of face recognition technology in real-world attendance systems.

# 10. Future Enhancements

The current system provides a basic and functional face authentication attendance solution. In the future, the following enhancements can be implemented to improve accuracy, security, and usability:

- Liveness Detection:Advanced spoof prevention techniques such as blink detection, facial movement analysis, or depth sensing can be added to ensure that only live faces are authenticated.
- Cloud-Based Database:The attendance records and face encodings can be stored in a cloud database to allow centralized access and better scalability.
- Mobile and Web Application:A web or mobile interface can be developed to make the system more user-friendly and accessible from different devices.
- Multiple Face Handling:The system can be extended to detect and recognize multiple faces simultaneously and mark attendance for multiple users in a single frame.
- Improved Accuracy Using Deep Learning Models:More advanced deep learning-based face recognition models can be integrated to improve recognition accuracy under varying lighting and pose conditions.
- Report Generation:Automatic generation of daily or monthly attendance reports in PDF or Excel format can be added for easy record keeping and analysis