

# Face Recognition based attendance taking system for employees

Priyanganu Bhattacharyya, Sutanuka Bhattacharjee, Arjama Dutta  
NSHM Knowledge Campus, Kolkata

## Abstract

An effective workplace is all about its employees and their participation. To keep a track on employee involvement in the workplace, attendance taking systems have evolved through the years. From manual check-ins to biometric scanning – it has come a long way. To maintain the pace of evolving technologies, “Face-Recognition based attendance taking system” plays an important role. It completely automates the attendance recording by making use of the advanced libraries of Python 3.12.4. Dlib, Face-recognition, OpenCV, Openpyxl are the responsible libraries for this task. The working principle of this system works in few steps. It utilizes a webcam to capture the facial structure. The database stores pre-recorded pictures of the employees in various angles to make the database more efficient. Then it compares the captured image to those of the database. If a match is found, the data is then linked to an Excel file to keep track of the attendance. Factors like employee name and time of arrival are main components of that file. The system minimizes human intervention, reduces errors, and makes manipulation infeasible. It represents a strong alternative to traditional methods. Nevertheless, it requires sufficiently strong lights to perform accurate facial detection and operates best in a Wi-Fi-enabled environment. Thus, automation of the attendance process brings innumerable benefits like increasing efficiency, data accuracy, hence enhanced productivity. It becomes an indispensable tool in today's work environment.

**Keywords:** Face-recognition, OpenCV, Haar Cascade Classifier, LBP Classifiers, Advantages of Face-recognition system.

## Introduction

Since the introduction of Industry 4.0 (I4) in 2011 at the Hannover Fair in Germany, automation and machine learning (ML) have piqued the interest of researchers to apply them to industry, agriculture, and other services. This field forms an important part of modern business and research. ML can improve computing performance in processes pertaining to a single factory or system, a chain of factories, or multi-systems used in any organization. I4 will benefit human society when it synergizes artificial intelligence (AI) with automation in production. [1]

Attendance tracking has evolved significantly over the years, from manual sign-in sheets to biometric systems such as fingerprint scanners and facial recognition. Traditional methods, such as manual attendance registers or card-based systems, are time-consuming, prone to errors, and susceptible to manipulation (e.g., proxy attendance). In contrast, biometric systems offer a more secure and efficient way to track attendance. Among biometric technologies, facial recognition has gained prominence due to its non-intrusive nature and high accuracy.

Facial recognition systems analyse unique facial features to identify individuals, making them ideal for applications such as security, access control, and attendance management. Python, with its extensive libraries and ease of use, has become the language of choice for developing facial recognition systems. Libraries such as OpenCV, Dlib, and Face-recognition provide powerful tools for face detection, feature extraction, and matching, enabling developers to build robust and scalable attendance systems.

### **Problem Statement:**

Many organizations still rely on outdated attendance systems that are inefficient, inaccurate, and vulnerable to manipulation. Manual systems are slow and prone to errors, while card-based systems can be cheated or suffer from issues such as lost or misplaced cards. These limitations negatively impact productivity, compliance, and organizational effectiveness.

To address these challenges, there is a growing need for an automated attendance system that is secure, accurate, and efficient. A face recognition-based attendance system offers a modern solution by leveraging facial biometrics to identify employees and record their attendance automatically. Built using Python, this system eliminates the need for manual intervention, reduces errors, and provides a tamper-proof record of attendance.

### **Scope of the system:**

Face detection is a type of identification. When we see any person face, then we will get information like gender and age, etc. Face detection is used in applications such as human-machine interaction, gender classification, surveillance system, bio-metrics, etc., it is very difficult to detect the face. [2]. The proposed system is designed to automate attendance tracking using facial recognition technology. Its key features include:

1. Automatic Attendance Management: Eliminates human errors and reduces the time required for check-ins and check-outs.

2. **Accuracy and Reliability:** Compares facial features to identify employees, reducing the possibility of proxies or manipulation.
3. **Scalability:** Suitable for use in corporate offices, schools, factories, and government buildings.
4. **Real-Time Recording:** Records attendance in real-time, providing up-to-date information on employee presence.
5. **Integration with Other Systems:** Can be integrated with payroll systems to automate salary processing based on attendance data.
6. **Security and Privacy:** Encrypts facial data to ensure compliance with privacy regulations.

## **Core Content**

### **Overview of the system:**

The face recognition-based attendance system operates in the following steps:

1. **Employee Registration:** Employees register by providing their facial data and personal details (e.g., name, employee ID). Multiple images are captured from different angles to ensure robust recognition.
2. **Facial Data Encoding:** The system extracts unique facial features (e.g., distance between eyes, nose, and mouth) and stores them as embeddings in a database.
3. **Attendance Recording:** When an employee arrives, the system captures their live image via a webcam, compares it with the stored embeddings, and logs their attendance in real-time.
4. **Data Storage:** Attendance data is recorded in an Excel file, which can be accessed by employers for reporting and analysis.

## **Face Detection Techniques**

The system relies on face detection algorithms to identify human faces in images. Two primary techniques are discussed:

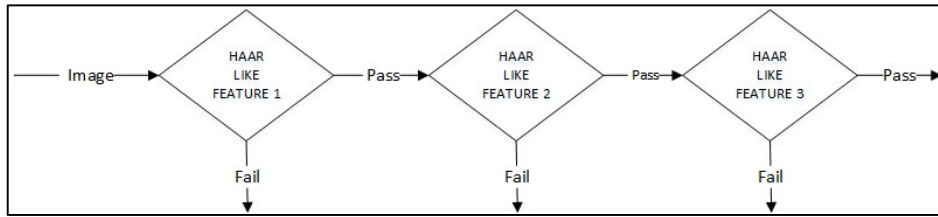
### **Haar Cascade Classifier:**

The Haar Cascade Classifier is a machine learning-based approach that uses Haar-like features to detect faces. It works by analysing patterns of intensity differences in images and applying a cascade of classifiers to discard non-face regions. While fast and efficient, the Haar Cascade Classifier is sensitive to variations in lighting, pose, and scale.

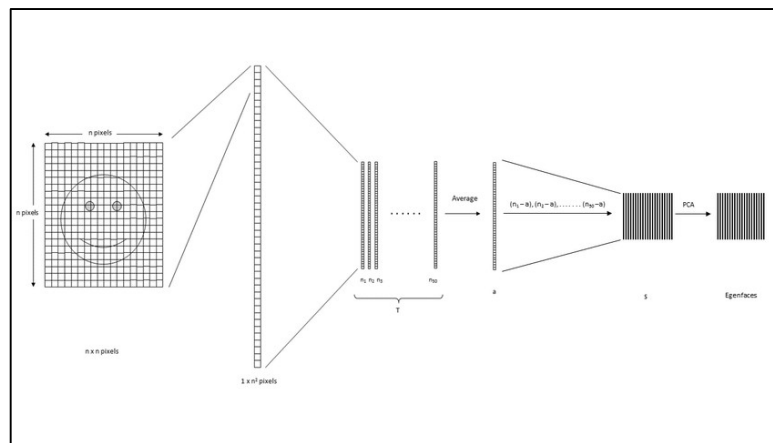
Paul Viola and Michael Jones proposed Haar Cascade Algorithm, which is productively used for Object Detection. This Algorithm is based on a Machine Learning approach in which lots of images are used, whether positive or negative, to train the classifier.

- **Positive Images:** Positive Images are a type of image that we want our classifier to identify.

- **Negative Images:** Negative Images are a type of image that contains something else, i.e., it does not contain the objects we want to detect. [3]



**Fig 1: Haar Cascade Classifier flowchart [4]**

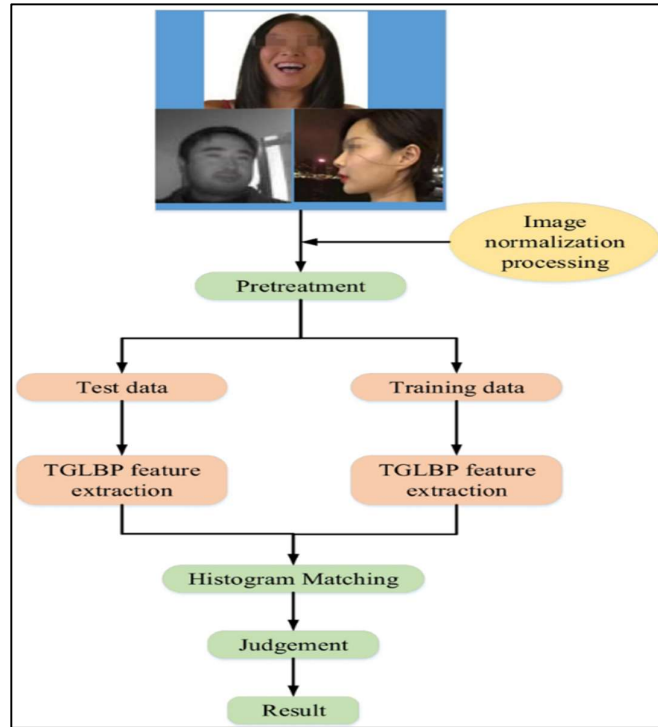


**Fig 2: Pixels of the image are reordered to perform calculations for Eigenface [4]**

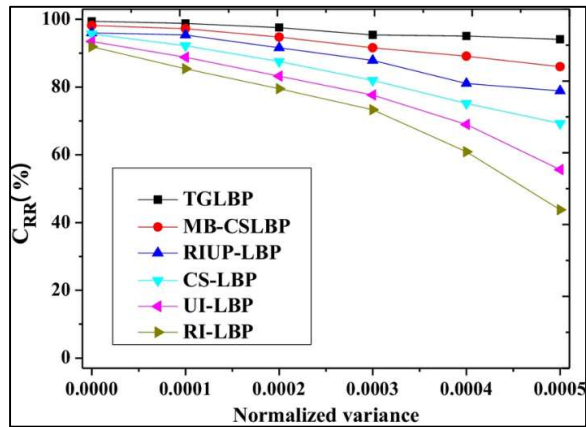
**Local Binary Patterns (LBP) Classifier:** The LBP Classifier is a texture-based method that detects faces by analyzing local texture patterns. It is robust to changes in lighting but struggles with noise and pose variations. Both techniques are implemented using the OpenCV library.

**LBP Features:** For each pixel in the image, LBP compares the intensity of the center pixel with its neighboring pixels (typically an 8-pixel neighborhood).

- If a neighboring pixel's intensity is greater than or equal to the center pixel, it is assigned a value of 1; otherwise, it is 0.
- The binary values are concatenated to form an 8-bit binary number, which is then converted to a decimal value (LBP value).



**Fig 3: LBP Algorithm flowchart [5]**



**Fig 4: Noise addition experiment curves of algorithms on the CAS-PEAL background set [5]**

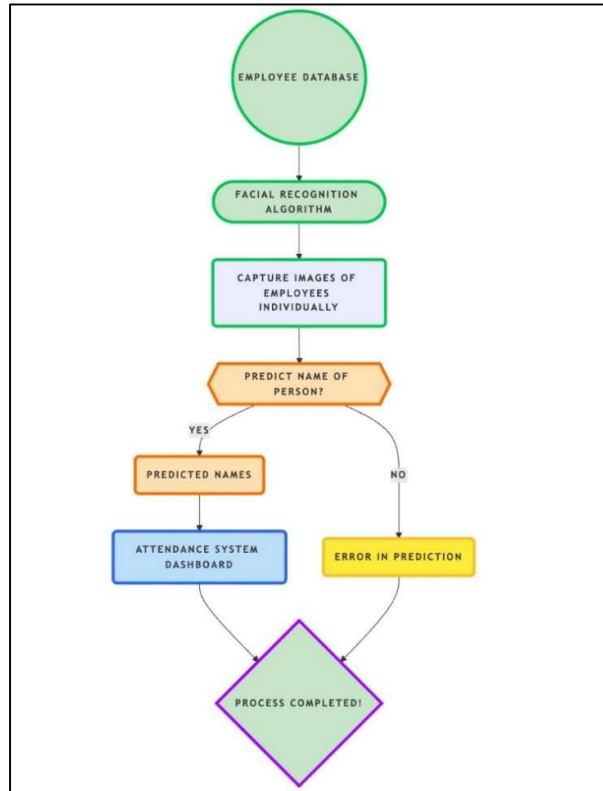
## Working Mechanism

The system works in three major steps. All the steps are correlated to give the desired output of the attendance system. The steps are:

- 1. Database set up** – The database is set by loading the existing photographs of individual employees.
- 2. Face recognition** – The employee is required to stand in front of the image capturing device/camera. The system will capture the image and compare it

with the existing images in the database to trace the similarity in the Haar like features.

**3. Logging Attendance** – The date and time of attendance is logged in a csv (comma separated values) format datafile for further retrieval and analysis.



**Fig 5: Flowchart of working mechanism of the project**

## Technological Framework

### Main Back-end Language: Python 3.12.4

Python is a high-level general purpose programming language:

- Because code is automatically compiled to byte code and executed, Python is suitable for use as a scripting language, Web application implementation language, etc.
- Because Python can be extended in C and C++, Python can provide the speed needed for even compute intensive tasks.
- Because of its strong structuring constructs (nested code blocks, functions, classes, modules, and packages) and its consistent use of objects and object-

oriented programming, Python enables us to write clear, logical applications for small and large tasks. [6]

With all these features in hand, Python 3.12.4 provides a wide variety of libraries to work with. Hence, a complex program like Face recognition becomes much easier for the robust features of Python.

### **Corresponding Front-end Languages: HTML, CSS**

**HTML** is a Markup Language. It is used in the project to build a simple yet efficient user interface.

**CSS** is a Stylesheet based Language. It provided the graphic user interactivity in the system.

### **IDE: Visual Studio Code**

A versatile and well managed IDE is provided by the Visual Studio Code. It's memory efficient approach makes it easier to organize all the languages and framework into one.

### **Libraries in use:**

Python Libraries provide a wide set of actions depending on specific tasks. It is very helpful in data encapsulation and providing abstraction to the author. The libraries which are used in the project are:

- i. OpenCV-python
- ii. Cmake
- iii. Dlib
- iv. Face-recognition
- v. Face-recognition-models
- vi. Pandas
- vii. Numpy 1.26.4
- viii. Flask
- ix. setuptools

### **OpenCV Library in Python**

OpenCV (Open Source Computer Vision Library) is a powerful open-source library designed for computer vision and image processing tasks. It provides tools to analyze, manipulate, and process images and videos, making it essential for applications like object detection, face recognition, and augmented reality.

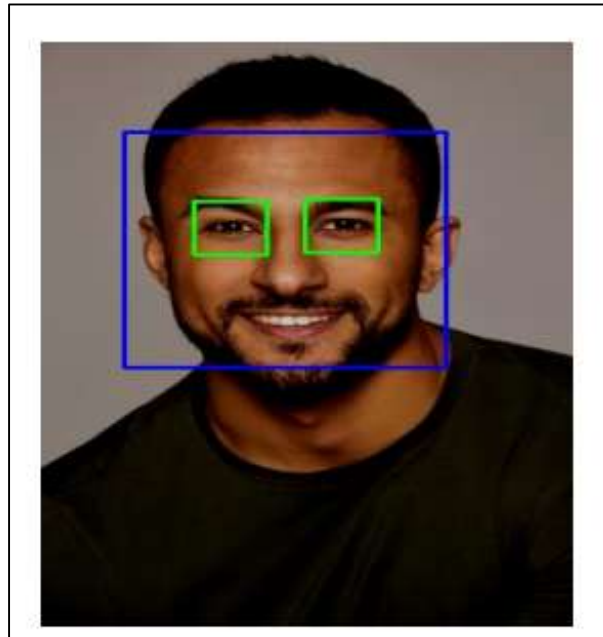
OpenCV was started at Intel in 1999 by Gary Bradski for the purposes of accelerating research in and commercial applications of computer vision in the world and, for Intel,

creating a demand for ever more powerful computers by such applications. Vadim Pisarevsky joined Gary to manage Intel's Russian software OpenCV team. Over time the OpenCV team moved on to other companies and other Research. Several of the original team eventually ended up working in robotics and found their way to Willow Garage. In 2008, Willow Garage saw the need to rapidly advance robotic perception capabilities in an open way that leverages the entire research and commercial community and began actively supporting OpenCV, with Gary and Vadim once again leading the effort. [7]

The driving force behind the whole program is the OpenCV library. It uses cascading technique to implement the face recognition using Haar Cascade Algorithm. It analyses the images by their HSV components, where H stands for Hue, S stands for Saturation and V stands for Value. In OpenCV, the Hue component ranges from 0 to 179 as it takes the image input as a 8 bit unsigned array.

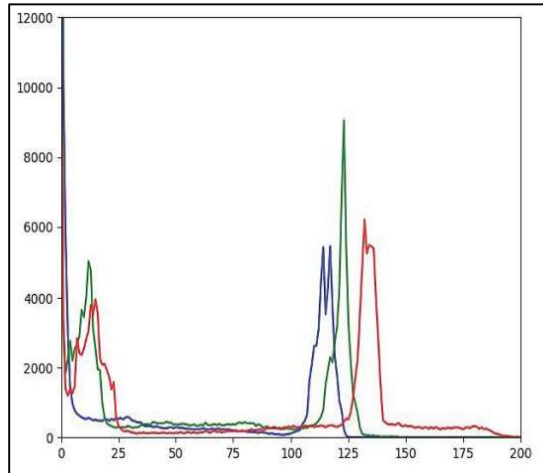
```
rgb_small_frame = cv2.cvtColor(small_frame , cv2.COLOR_BGR2RGB)
```

In this code snippet, the `cv2.cvtColor()` command changes the color space from RGB (Red-Green-Blue) to HSV color space. As OpenCV takes an image input in the BGR (Blue-Green-Red) color mode by default, the mode is changed to RGB using `cv2.COLOR_BGR2RGB` command. Thus, the further analysis of Haar-like features is done in the system.



**Fig 6: Considering the single face object (Singh et al.)**

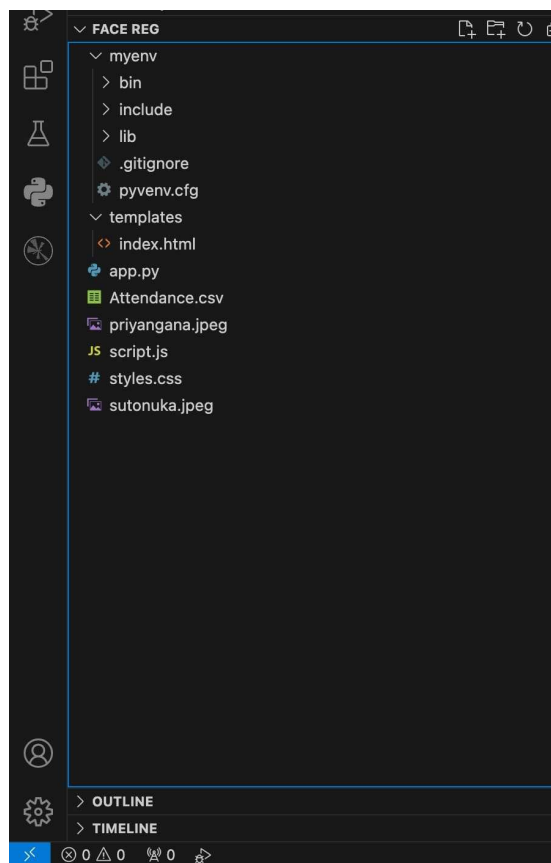




**Fig 7: Histogram of the RGB graph for the image (Singh et al.)**

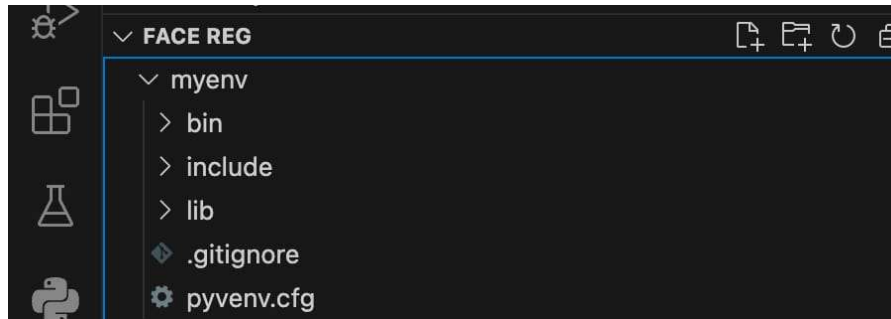
## Implementation

**1. Folder Creation:** A dedicated folder for the project is created in VSCode interface. It will have the employee images as contents.



**Fig 8: Main folder creation**

**2. Creation of virtual environment:** A virtual environment named ‘myenv’ is created for optimized library installation. Activate the environment using terminal of VSCode.



**fig 9: Virtual environment is created named ‘myenv’**

**3. Install the libraries:** The ‘pip install’ command is used to install all the libraries. After successful installation ‘pip list’ command will view the following list of installed libraries:

Package	Version
blinker	1.9.0
click	8.1.8
cmake	3.31.4
dlib	19.24.2
et_xmlfile	2.0.0
face-recognition	1.3.0
face_recognition_models	0.3.0
Flask	3.1.0
itsdangerous	2.2.0
Jinja2	3.1.5
MarkupSafe	3.0.2
numpy	1.26.4
opencv-python	4.10.0.84
openpyxl	3.1.5
pandas	2.2.3
pillow	11.1.0
pip	24.3.1
python-dateutil	2.9.0.post0
pytz	2024.2
setuptools	75.8.0
six	1.17.0
tzdata	2024.2
Werkzeug	3.1.3

**fig 10: List of libraries installed for the project**

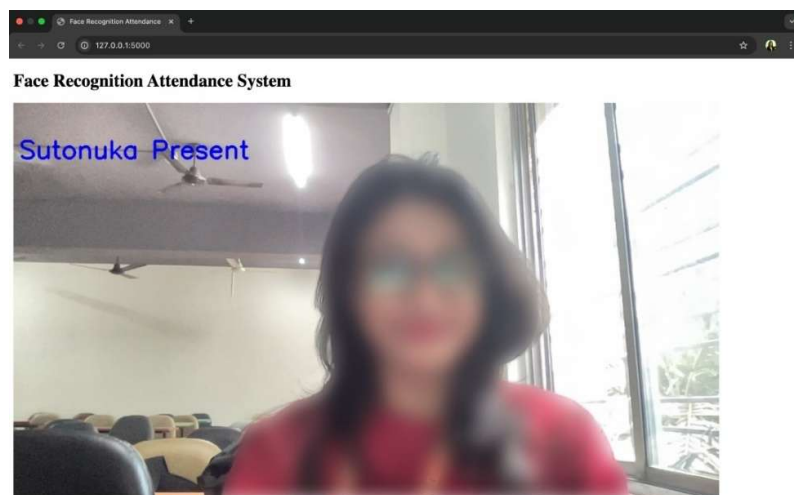
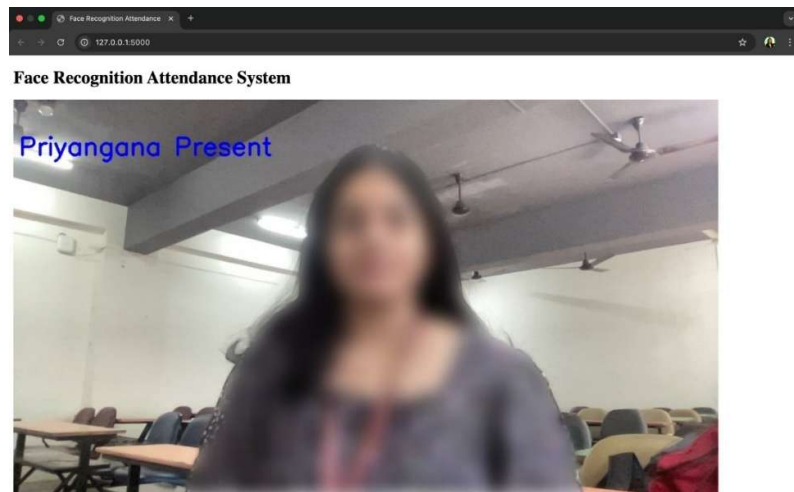
**4. Results:** After compiling, debugging and running the program the terminal will display to redirect to the mentioned port using a web browser.

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

pip 24.3.1
python-dateutil 2.9.0.post0
pytz 2024.2
setuptools 75.8.0
six 1.17.0
tzdata 2024.2
Werkzeug 3.1.3
myenvsutonuka@Sutonukas-MacBook-Air Face Reg % cd /Users/sutonuka/Desktop/Face\ Reg ; /usr/bin/env /Users/sutonuka/D
esktop/Face\ Reg/myenv/bin/python /Users/sutonuka/.vscode/extensions/ms-python.debugpy-2024.14.0-darwin-arm64/bundled
/libs/debugpy/adapter/../../debugpy/launcher 49691 -- /Users/sutonuka/Desktop/Face\ Reg/app.py
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 178-948-280
```

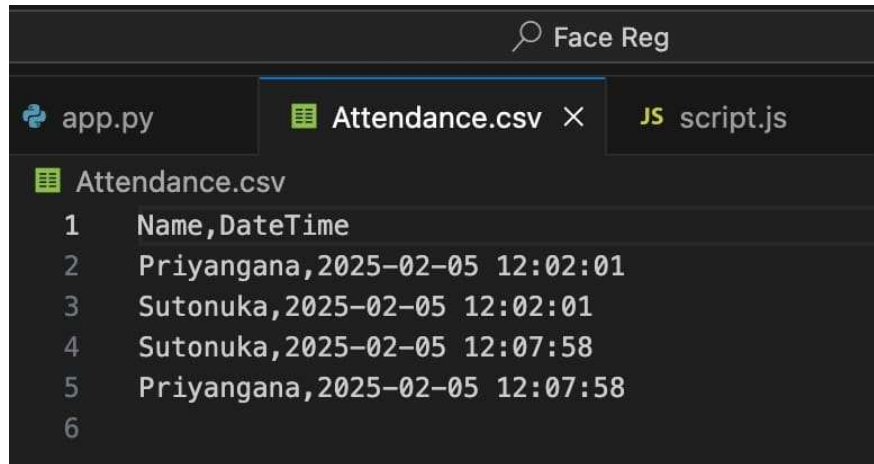
**Fig 11: Terminal displaying to redirect to the given web address**

After redirecting to the web page, the attendance is taken by the face -recognition model.



**Fig 12 & 13: Face-recognition model**

Finally, the attendance is saved in a csv file named 'attendance.csv' by the comma separated values.



```
Face Reg
app.py Attendance.csv X JS script.js
Attendance.csv
1 Name,DateTime
2 Priyangana,2025-02-05 12:02:01
3 Sutonuka,2025-02-05 12:02:01
4 Sutonuka,2025-02-05 12:07:58
5 Priyangana,2025-02-05 12:07:58
6
```

**Fig 14: Attendance logged in csv format**

#### **Challenges:**

- Lighting Conditions: Effective detection requires adequate lighting.
- Hardware Requirements: Requires a webcam and sufficient processing power.
- Connectivity: Operates effectively only within Wi-Fi-enabled areas.
- Technical dependency: The system may collapse if there is any technical malfunction, which may lead to potential data loss.

### **Implementation Guidelines**

- **Hardware Requirements:**
  - Laptop/PC with i3 processor or higher.
  - 4 GB RAM or higher, 100 GB ROM or higher and a functional webcam.
- **Software Requirements:**
  - Python 3.12.4.
  - HTML
  - CSS
  - Visual Studio Code or preferred IDE.
- **Setup:**
  - Create a dedicated folder for the project and a virtual environment within the IDE.
  - Install necessary libraries like OpenCV, Dlib, Openpyxl and Face-recognition.
  - Set up database by including the pictures.

- Configure the system for real-time attendance marking.

## Conclusion:

In the era of Industry 4.0, the focus on automation is relevant in every aspect of workplace and organization. Based on face recognition technology, Face Recognition System works out a comprehensive and progressive solution for addressing employee attendance.[8] Due to the factors of enhanced accuracy and reduced human intervention with improved data security and reliability, it is a better solution compared to other conventional methods of attendance. Despite the limitations such as lighting condition and dependency on the hardware, the advantages exceed these problems a lot, which makes this method a perfect solution for the modern organization, which is searching for an effective way to manage the attendance problem.

## Future Scope:

- Connecting the system with Cloud Environment will help with Data Designing and organizing. Which in turn will ease the process of Data retrieval and analysis to take future decisions. A dedicated database will also help in data privacy and security.
- In various highly confidential organizations, face detection can be merged with other biometric prompts like fingerprints, will help to establish multi-facto authentication system to ensure data integrity.
- Using Face detection system with a proper database in workplace will help to trace out employee traits by analysing their attendance performance, which will help in HRM (Human Resource Management).
- Convolutional Neural Network (CNN) helps in image analysis. Application of Face recognition system may help in detailed facial structure analysis when it comes to restrained conditions like lighting or angle.[9]
- Real-time data processing feature of the system will help in building smart office environment by implementation of IoT (Internet of Things).

## Reference

1. L. T. H. Phuc, H. J. Jeon, N. T. N. Truong, and J. J. Hak, "Applying the haar-cascade algorithm for detecting safety equipment in safety management systems for multipleworking environments," *Electronics* (Switzerland), vol. 8, no. 10, Oct. 2019, doi: 10.3390/electronics8101079.
2. A. Singh and F. Furtado, "Modified Haar-Cascade Model for Face Detection Issues", doi: 10.22105/riej.2020.226857.1129.
3. "Haar Cascade Algorithm - Tpoint Tech." Accessed: Feb. 17, 2025. [Online]. Available: <https://www.tpointtech.com/haar-cascade-algorithm>
4. ["(PDF) Face Detection & Face Recognition Using Open Computer Vision Classifies." Accessed: Feb. 17, 2025. [Online]. Available: [https://www.researchgate.net/publication/318900718\\_Face\\_Detection\\_Face\\_Recognition\\_Using\\_Open\\_Computer\\_Vision\\_Classifies](https://www.researchgate.net/publication/318900718_Face_Detection_Face_Recognition_Using_Open_Computer_Vision_Classifies)

5. H. Qu and Y. Wang, "Application of Optimized Local Binary Pattern Algorithm in Small Pose Face Recognition under Machine Vision," *Multimed Tools Appl*, vol. 81, no. 20, pp. 29367–29381, Aug. 2022, doi: 10.1007/S11042-021-11809-9.
6. "A Python Book: Beginning Python, Advanced Python, and Python Exercises." Accessed: Feb. 17, 2025. [Online]. Available: [https://web.archive.org/web/20120623165941/http://cutter.rexx.com/~dkuhlman/python\\_book\\_01.html#introduction-python-101-beginning-python](https://web.archive.org/web/20120623165941/http://cutter.rexx.com/~dkuhlman/python_book_01.html#introduction-python-101-beginning-python)
7. "(PDF) Facial Recognition using OpenCV." Accessed: Feb. 17, 2025. [Online]. Available: [https://www.researchgate.net/publication/267426877\\_Facial\\_Recognition\\_using\\_OpenCV](https://www.researchgate.net/publication/267426877_Facial_Recognition_using_OpenCV)
8. A. V B, Z. V Ayisha P, M. Chettiyam Veettil, and B. Tech Students, "SAM-SMART ATTENDANCE MARKING SYSTEM USING FACIAL RECOGNITION." [Online]. Available: [www.irjmets.com](http://www.irjmets.com)
9. W. Zhao, R. Chellappa, P. J. Phillips, A. Rosenfeld, and ; R Chellappa, "Face Recognition: A Literature Survey," 2003. [Online]. Available: <http://www.eyematic.com/>