**FACIAL ATTENDANCE SYSTEM**

**CONTENT**

**1. ABSTRACTION ………………………………………………..**

**2. INTRODUCTION …………………………………………………**

2.1 PROJECT OBJECTIVE **………………………………………………………**

2.2BACKGROUND …………………………………………………………

2.3 PROBLEM STATEMENT **………………………………………………………**

2.4 AIMS AND OBJECTIVES **………………………………………………………..**

2.5SCOPE OF THE PROJECT **……………………………………………………..**

**3 LITERATURE REIVIEW …………………………………………………..**

3.1 Face Recognition Based Attendance Marking System **……………………………………………………**

Fingerprint Based Attendance System Using **………………………………………………..**

RFID based Student Attendance System …………………………………………………

**4. MODEL IMPLEMENTATION AND ANALASIS ………………………………………………..**

4.1 Design requirement ……………………………………………………..

4.2 The creation of the database …………………………………………………..

4.3 The process of the attendance taking ……………………………………………………..

4.4 Structure of the csv file …………………………………………….

4.5Experimental results ……………………………………….

**5. CODE IMPLEMENTATION ANALASYS ………………………………………….**

5.1Sign up.py …………………………………………..

5.2 Sign in.py …………………………………………

5.3 Visitor.py …………………………………………

**6. CONCLUSION**

**ACKNOWLEDMENT**

We would like to express our deep guide to our project guide **Mr.C** **Sharath** his guidance with unsurpassed knowledge and immense encouragement. We are grateful to our M. SRINIVAS SIR CEO of Mempage technologies pvt Ltd for providing us with the required facilities for the competition of the project work

We would like to thanks our team members and friends and colleagues for their encouragement and support throughout the project period. At last but not least we thank everyone for supporting us directly or indirectly in completing this project successfully

**PROJECT EMPLOYEES**

N.chiranjeevudu

k.bhaskar

G.Raju

A.Swapna

P.Prabhas

**Chapter-1**

**ABSTRACTION**

Automatic face recognition technology have made many improvements in the changing world. Smart attendance using real time face recognition is a real world solution which comes with day to day activities of handling student attendance system. Face recognition based attendance system s process of recognizing the employee and student faces for taking attendance by using face biometrics based on high definition monitor video and other information technology. In my face recognition project a computer system will be able to find and recognize human faces fast and precisely in images or videos that are being captured through a surveillance camera or web camera. Numerous algorithms and techniques have been developed for improving the performance of face recognition. Bur the concept to be frames of the video into images so that the face of the employee and student can be easily recognized for their attendance so that the attendance database can be easily reflected automatically

**Keywords:**

Opencv, numpy, pandas, Face recognition model, HaarCascade classifier, KNN model.

**Chapter -2**

**INTRODUCTION**

**2.1 Project Objective:**

Attendance is of prime importance for both employees and the organization an organization. So it is very important to keep record of the attendance. The problem arises when we think about the traditional process of taking attendance in offices and employee work places. Every time check the employees how many people attend office Calling and name or roll number of the employee/student for attendance is not only a problem of time consumption but also it needs energy. So an automatic attendance system can solve all above problems. There are some automatic attendances making system which are currently used by much institution. One of such system is biometric technique and RFID system. Although it is automatic and a step ahead of traditional method it fails to meet the time constraint. The employees or student has to wait in queue for giving attendance, which is time taking. This project introduces an involuntary attendance marking system, devoid of any kind of interference with the normal teaching procedure. The system can be also implemented during exam sessions or in other teaching activities where attendance is highly essential. This system eliminates classical employee identification such as calling name of the employee, or checking respective identification cards of the student, which can not only interfere with the ongoing teaching process, but also can be stressful for students during examination sessions. In addition, the students have to register in the database to be recognized. The enrolment can be done on the spot through the user friendly interface

2.2 **Background:**

Face recognition is crucial in daily life in order to identify family, friends or someone we are familiar with. We might not perceive that several steps have actually taken in order to identify human faces. Human intelligence allows us to receive information and interpret the information in the recognition process. We receive information through the image projected into our eyes, by specifically retina in the form of light. Light is a form of electromagnetic waves which are radiated from a source onto an object and projected to human vision. The analysed information will be compared to other representations of objects or face that exist in our memory to recognize. In fact, it is a hard challenge to build an automated system to have the same capability as a human to recognize faces. However, we need large memory to recognize different faces, for example, in the MNC companies and Universities, there are a lot of people with different face and gender, it is impossible to remember every face of the individual without making mistakes. In order to overcome human limitations, computers with almost limitless memory, high processing speed and

Power are used in face recognition systems. The human face is a unique representation of individual identity. Thus, face recognition is defined as a biometric method in which identification of an individual is performed by comparing real-time capture image with stored images in the database of that person

The work on face recognition began in 1960. Woody Bledsoe, Helen Chan Wolf and Charles Bisson had introduced a system which required the administrator to locate eyes, ears, nose and mouth from images. The distance and ratios between the located features and the common reference points are then calculated and compared. The studies are further enhanced by Goldstein, Harmon, and Lesk in 1970 by using other features such as hair colour and lip thickness to automate the recognition. In 1988, Kirby and Sirovich first suggested principle component analysis (PCA) to solve face recognition problem. Many studies on face recognition were then conducted continuously until today

**2.3 Problem Statement:**

Traditional employee/student attendance marking technique is often facing a lot of trouble. The face recognition student attendance system emphasizes its simplicity by eliminating classical student attendance marking technique such as calling student names or checking respective identification cards. There are not only disturbing the teaching process but also causes distraction for students during exam sessions. Apart from calling names, attendance sheet is passed around the classroom during the lecture sessions. The lecture class especially the class with a large number of students might find it difficult to have the attendance sheet being passed around the class. Thus, face recognition attendance system is proposed in order to replace the manual signing of the presence of students which are burdensome and causes students get distracted in order to sign for their attendance. Furthermore, the face recognition based automated student attendance system able to overcome the problem of fraudulent approach and lecturers does not have to count the number of students several times to ensure the presence of the students.

Hence, there is a need to develop a real time operating student attendance system which means the identification process must be done within defined time constraints to prevent omission. The extracted features from facial images which represent the identity of the students have to be consistent towards a change in background, illumination, pose and expression. High accuracy and fast computation time will be the evaluation points of the performance.

**Identification Errors:**

Facial Recognition technology doesn’t always work as well as it should. Facial Recognition systems can be impacted by poor lighting or low image quality. The data may not match up with the person’s nodal points because of camera angles being obscured, this creates an error when matching face prints cannot be verified in the database

**POSE VARIATIONS:**

Variation in pose causes significant problems in detecting a face. Pose variation can be due to change in observing angle of the observer and also due to rotation in the head position. These variations can cause a serious problem in identifying the input image. Many of the systems can tolerate small variations such as small rotations in angles. But it will be difficult when it comes to large rotational angles. The database usually consists of face images of frontal view of the faces. Since the existing FRSs are very sensitive to pose variation, pose correction is essential and could be achieved by means of efficient techniques aiming to rotate the face and/or to align it to the image's axis.

**Variation in illumination**

Variation in illumination Variations of illuminations could reduce the efficiency of FRS. For moderate levels of lighting of the background, face detection and recognition are much difficult to perform. Variation in illumination can vary the total magnitude of light intensity being reflected back from an object. On the other hand, higher light levels could lead to over‐exposure of the face and (partially) undetectable facial patterns. There have been many algorithms such as equalization techniques that are available now to get rid of this problem to an extent. Sometimes even multiple algorithms can be used in a face recognition system to tolerate the issue of illumination. But in case of extents, it is not desirable to depend on these techniques.

**AGE VARIATIONS**

Another reason for the changes in the appearance of the face could be the aging of the human face and could affect the entire process of face recognition; if the time between each image capture is large, there will be significant changes in the person. As per various study conducted by scientists, in every 10 years there will be significant changes in an individual’s face appearance. It is not just the shape and lines of a face that gets modified over time; there will be changes in hairstyles as well.

**Occlusions:**

Variation in facial appearance can also be caused due to presence of objects that such as occlusion that partially cover the face. This makes it a difficult task for the system to classify the image. Although the face is found, it may be difficult to recognize it due to some hidden facial parts, making it difficult to recognize features. This challenge can be seen in real world application where acquiring persons talking on the phone, wearing glasses, scarf, hats etc or having their faces covered with hands.

**2.4 Aims and Objectives**:

The objective of this project is to develop face recognition attendance system. Expected achievements in order to fulfil the objectives are:

● To detect the face segment from the video frame.

● To extract the useful features from the face detected.

● To classify the features in order to recognize the face detected.

● To record the attendance of the identified employee.

**2.5 Scope of the project:**

We are setting up to design a system comprising of two modules. The first module (face detector) is a mobile component, which is basically a camera application that captures student faces and stores them in a file using computer vision face detection algorithms and face extraction techniques. The second module is a desktop application that does face recognition of the captured images (faces) in the file, marks the students register and then stores the results in a database for future analysis.

**CHAPTER-3**

**LITERATURE REVIEW**

**3.1 Attendance System Using NFC Technology with Embedded Camera on Mobile Device:**

According to research journal Attendance System Using NFC (Near Field Communication) Technology with Embedded Camera on Mobile Device (The attendance system is improved by using NFCtechnology and mobile application. According to the research paper, each student is given a NFC tag that has a unique ID during their enrolment into the college. Attendance of each class will then be taken by touching or moving these tags on the lecturer mobile phone. The embedded camera on the phone will then capture the student’s face to send all the data to the college server to do validation and verification. The advantages of this method is where the NFC is simple to use, and the speed of connection establishment is very high. It indeed speeds up the attendance taking process a lot. However, this system couldn’t automatically spot the violation when the NFC tag is not personally tagged by the original owner. Apart from that, the convenience of the system which uses the mobile phone as the NFC reader was actually an inconvenience to the lecturer. Imagine if the lecturer had forgotten to bring their mobile phones to work, what would be the backup procedure for the attendance to be recorded? Moreover, most of the lecturer will not likely to prefer their personal smart phones to be used in this way due to privacy matter. Hence, unique information about the student like biometrics or face recognition, which is guanine for a student should be used in replacement of the NFC tag. This will ensure attendance to be taken originally by the actual student.

**3.2 Face Recognition Based Attendance Marking System**

The second research journals “Face Recognition Based Attendance Marking System is based on the identification of face recognition to solve the previous attendance system’s issues. This system uses camera to capture the images of the employee to do face detection and recognition. The captured image is compared one by one with the face database to search for the worker’s face where attendance will be marked when a result is found in the face database. The main advantage of this system is where attendance is marked on the server which is highly secure where no one can mark the attendance of other. Moreover, in this proposed system, the face detection algorithm is improved by using the skin classification technique to increase the accuracy of the detection process. Although more efforts are invested in the accuracy of the face detection algorithm, the system is yet not portable. This system requires a standalone computer which will need a constant power supply that makes it not portable. This type of system is only suitable for marking staff s attendance as they only need to report their presence once a day, unlike students which require to report their attendance at every class on a particular day, it will be inconvenient if the attendance marking system is not portable. Thus, to solve this issue, the whole attendance management system can be developed on an portable module so that it can be work just by executing the python program.

**3.3 Fingerprint Based Attendance System Using Microcontroller and LabView**

The third research journal “Fingerprint Based Attendance System Using Microcontroller and LabView proposed a solution of using fingerprint to mark the attendance. This system is using 2 microcontrollers to deal with the fingerprint recognition process. Firstly, the fingerprint pattern will be obtained through a fingerprint sensor, then the information will be transmitted to microcontroller 1. Next microcontroller 1 will pass the information to microcontroller 2 to do the checking with the database that resides in it. After finding a employee’s match, the details are sent to the PC through serial communication to be displayed. This design is good as it accelerates development while maintaining design flexibility and simplifies testing. But again, this system is attached to a PC which make it not portable. Other than that, the database information cannot be accessible easily. Meaning that, for the parents whom are interested in knowing their employee’s attendance cannot easily or conveniently access the information. Therefore, to provide accessibility of the employee’s information to the legitimate concerned party, the information can be uploaded to a web server for easy access. While the authentication for the appropriate access can be enforced through a login screen.

**3.4 RFID based Student Attendance System**

According to the fourth research journal “RFID based Student Attendance System the proposed solution is almost similar to the first research journal where RFID technology is used to improve the older attendance system. In this system, a tag and a reader is again used as a method of tracking the attendance of the students. The difference between the first journals with this is where attendance’s information can be accessed through a web portal. It provides more convenient for information retrieval. Again, this system is imperfect in the sense that, firstly, it is not portable, as the RFID reader can only work when it is connected to a PC. Secondly, the RFID tag is not a guanine information that can uniquely identify a student, thus, resulting in the inaccuracy of the collected attendance information

Now we are going to taking attendance through the face recognition based attendance system

**CHAPTER-4**

**MODEL IMPEMENTATION**

**AND ANALASYS**

**4.1 Model implementation**

The main components used in the implementation approach are open source computer vision library (OpenCV). One of OpenCV’s goals is to provide a simple to-use computer vision infrastructure that helps people build fairly sophisticated vision applications quickly. OpenCV library contains over 500 functions that span many areas in vision. The primary technology behind Face recognition is OpenCV. The user stands in front of the camera keeping a minimum distance of 50cm and his image is taken as an input. The frontal face is extracted from the image then converted to gray scale and stored. The Principal component Analysis (PCA) algorithm is performed on the images and the eigen values are stored in an xml file. When a user requests for recognition the frontal face is extracted from the captured video frame through the camera. The eigen value is re-calculated for the test face and it is matched with the stored data for the closest neighbour.

**4.2 Design Requirements:**

We used some tools to build the system. Without the help of these tools it would not be possible to make it done. Here we will discuss about the most important one.

**4.3 PYTHON IDE:**

Python is easy to learn and an incredible programming language. It is effective object-oriented programming that has a simple high-level data structure. Python’s syntax is easy to use and a lot of libraries make this better than other object-oriented programming languages**.**

**4.4 OpenCV's :**

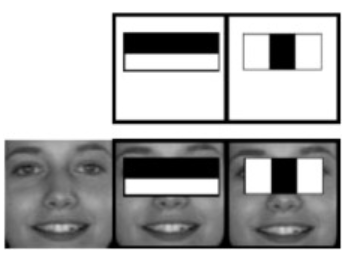
OpenCV: We used OpenCV 3 dependency for python 3. OpenCV is library where there are lots of image processing functions are available. This is very useful library for image processing. Even one can get expected outcome without writing a single code. The library is cross-platform and free for use under the open-source BSD license.

**4.5 NUMPY:**

It is a very popular scientific computing with python. It is an open-source module for python. Which gives quick precompiled capacities to numerical and mathematical routines. Moreover, Numpy advances the programming Language Python with incredible data structures for effective calculation of multi-dimensional matrices

**4.6 HAARSCASCADE FRONTAL FACE:**

The Haar Cascade algorithm is trained on a large dataset of positive and negative images of the object being detected. During training, the algorithm learns the features that distinguish the object from the background, such as edges, lines, and corners.

****

**SOFTWARE DEVELOPMENT**

There are two major system flows in the software development section as shown below:

• The creation of the face database

• The process of attendance taking

Both processes mentioned above are essential because they made up the backbone of the attendance management system. In this section, the process of both flows will be briefly described. Meanwhile, their full functionality, specific requirements and also the methods/approach to accomplish such objectives will be discussed in the upcoming chapter.

**The creation of the face database:**

The face database is an important step to be done before any further process can be initiated. This is because the face database acts as a comparison factor during the recognition process which will be discussed in later section. In the process above, a csv file is created to aid the process of image labelling because there will be more than one portrait stored for each student, thus, in order to group their portraits under the name of the same person, labels are used to distinguish them. After that, those images will be inserted into a recognizer to do its training. Since the training process is very time consuming as the face database grew larger, the training is only done right after there is a batch of new addition of student’s portraits to ensure the training is done as minimum as possible.

**The creation of the face database flow:**

**Image acquisition**

The image of the employee’s face will be captured for a specific amount of to be stored into the files on the Static folder

**Face detection**

The captured image will first undergo a face detection algorithm to ensure the system can identify a face in every images

**Image pre processing**

Captured image with a confirmed face detected in it will then undergo cropping colour conversion before actually being stored into the files

**Creation of csv file**

Then by using the path the stored image a csv file is created with each image being labelled accordingly. The csv file is created to aid the process of list insertion their corresponding labels

**Training the recognizer**

The image in the created list retrieved from the faces will then be pumped into a recognizer to do training

**Save the trained data**

After training the process is done the trained sets of data will be stored into a file which be retrieved during the recognition process to ensure the training process only done for the minimum time

**The process of attendance taking**

**Access the attendance management system**

The attendance taking office can be started after the office selected the related date and time table id for the current working session

**Initiate python script**

After receiving the button click action from done in the web page a python script will be initiated called

**Acquire portrait**

After the system capture the employee image it identifies the facial features with whom it matches

**Load the trained data**

In python script the first step is to load the trained data (.xml)

File which is saved during the face database creation process

**Mark the attendance**

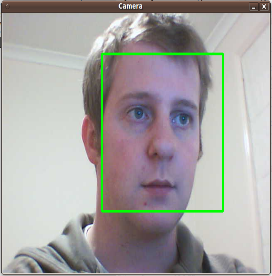
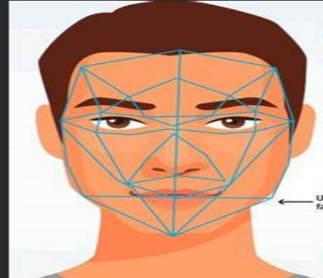
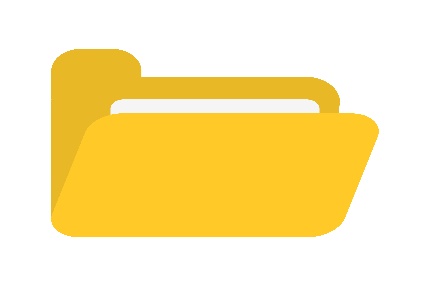
After identifying the appropriate employee from the capturing process a record of the current attendance will be added into the attendance table managed by database

**Recognise the faces**

If the captured image features matches the image features in the faces then it will display identifies the person by name

**METHODOLOGY:**

Before the attendance management system can work, there are a set of data needed to be inputted into the system which essentially consist of the individual’s basic information which is their ID and their faces. The first procedure of portrait acquisition can be done by using the Camera to capture the faces of the individual. In this process the system will first detect the presence of a face in the captured image, if there are no face detected, the system will prompt the user to capture their face again until it meets certain number of portraits which will be 10 required portraits in this project for each student. The decision of storing only 10 portrait per student is due to the consideration of the limited storage space in the raspberry pi because the total amount of students in the university is considered heavy. Then, the images will undergo several pre-processing procedures to obtain a grayscale image and cropped faces of equal sized images because those are the prerequisites of using the Eigen Faces Recognizer. Both of the processes mentioned above can be represented in the diagram below.

**  **

**Image Acquisition and Pre-processing procedures:**

After the images are being processed, they are stored into a file in a hierarchy manner. In this project, all the faces will be stored in a hierarchy manner under the „database‟ folder. When expanding through the database folder, there will consist of many sub-folders which each of them will represent an individual where a series of face portrait belonging to the same individual will be stored in that particular sub-folder. The subfolders that represent each individual will be named upon the ID no. of that individual which is unique for every single individual in the institution. The whole process of image retrieval, pre-processing, storing mechanism is done by the script named

**Structure of the content in the csv file**

After having sufficient images in the database, those images will then be inserted into a training mechanism. There are generally 3 different types of training mechanism provided in OpenCV. The recognizer that will be focused in this project will be the Eigen Faces recognizer. The concept behind Eigen Faces is simple – it recognizes a particular face by catching the maximum deviation in a face and then turning those identified variations into information to be compared when a new face arrives. In the training process, the faces folder file will be read to provide the path to all of the images where those images and labels will be loaded into a list variable. Then, the list will be passed into the training function where the training

**Experimental Results:**

The step of the experiments process are given below:

**Face Detection:**

Start capturing images through web camera of the client side: Begin:

● Pre-process the captured image and extract face image

● calculate the Eigen value of the captured face image and compared with Eigen values of existing faces in the database.

●If Eigen value does not matched with existing ones , save the new face image information to the face database (xml file).

● If Eigen value matched with existing one then recognition step will done.

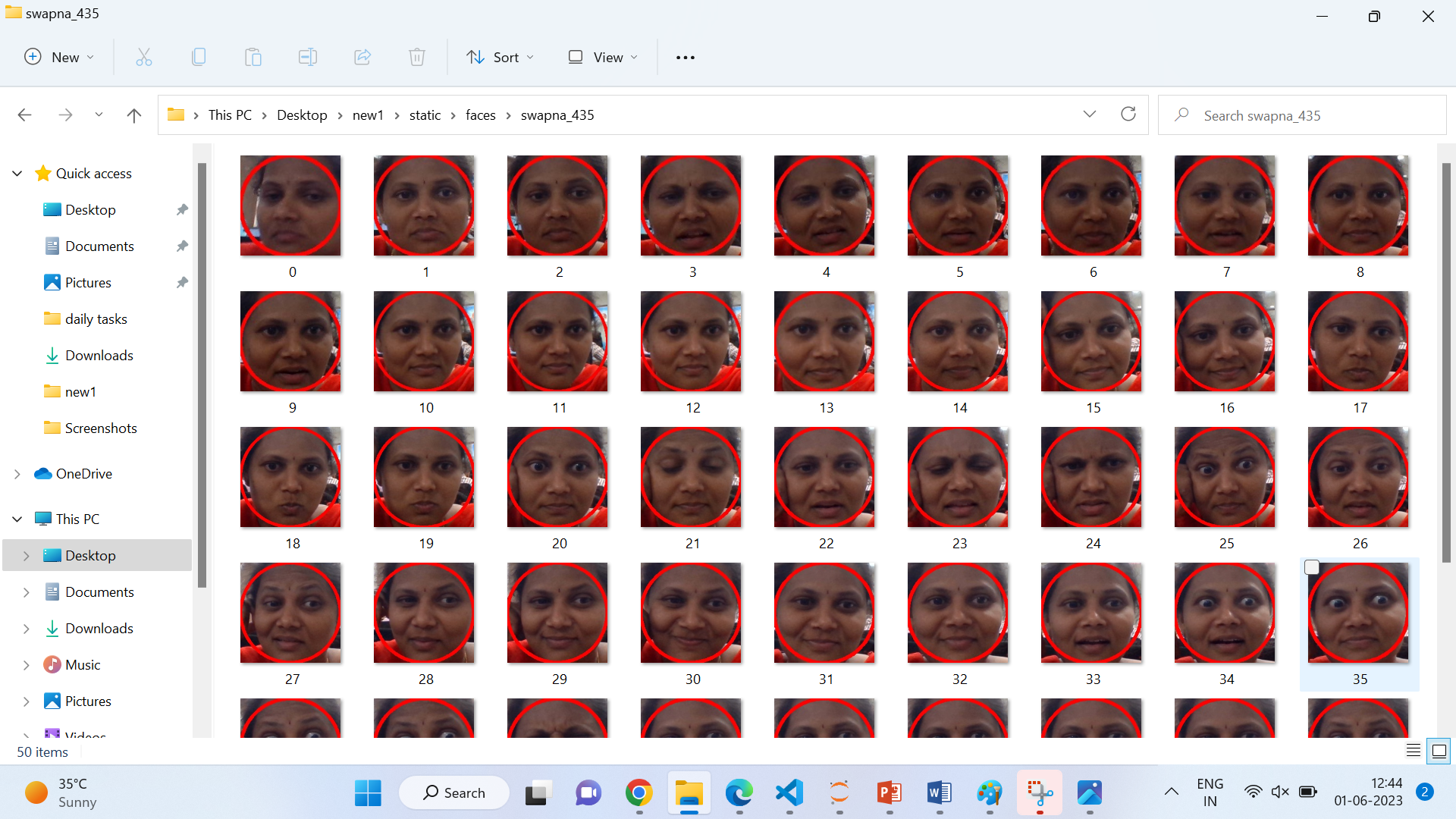
**Face Recognition:**

● Find the face information of matched face image in from the database.

● Update the log table with corresponding face image and system time that makes completion of attendance for an individual employee

This section presents the results of the experiment conducted to capture the face into a grey scale image of 50x50 pixels.

**HERE OUR DATA SET SAMPLE**



All our code is written in Python language. First here is our project directory structure and files

**CHAPTER- 4**

**CODE IMPLEMETATION**

**CODE IMPLEMENTATION:**

**SIGNUP.PY**

**SIGN IN .PY**

**VISITOR.PY**

**Sign up.py:**

In this code we are using date function method to retrieve the current date and then formats it using the start time function method. The format string specified the desired format month two digits represent the day in two digits and year also represents two digits. For example if the current date is June 4, 2023 the function will return 06-04-2023.

The date function also uses the method to retrieve the current date. However the method is used with a different format string ’%d-%B-%y. %d in this string represent two digits %B represent the full moth named and &Y rearrests the full year in four digits. For example if the current date is June 5-2023 the function will return 05-june-2023.

Check if a directory named “attendance” exists in in the current directory the os path isdir function true if the given path is an existing directory. if the directory does not exist the condition id true

Creates a directory named attendance using the os make directory

Code execute only if the “attendance” directory does not exists. the function creates intermediate directories as needed, so if the directories already exists it will not overwrite or modify it.

The extract attendance function attempts to read a csv file containing attendance date for the current date. It extracts specific columns from the file and returns them, along with the length of the DataFrame. If an exception occurs indicating an empty file or missing columns, it prints an error message and returns none for the columns and a length of 0. If directory named ‘faces’ exists inside the static directory.

Creates the in static folder automatically created a faces folder directory using makedir. Again this line is executed only if the static/faces directory does not already exists

If file named “attendance-{current-date}.csv” does not exists in the “attendance” directory. The list dir function returns a list of files and directories in the specified path. If the file does not exists the condition is true.

Then open a file named attendance –current date.csv in write mode (w) inside the attendance directory. To create or open a file with statement ensures that the file is automatically closed after the code is executed.

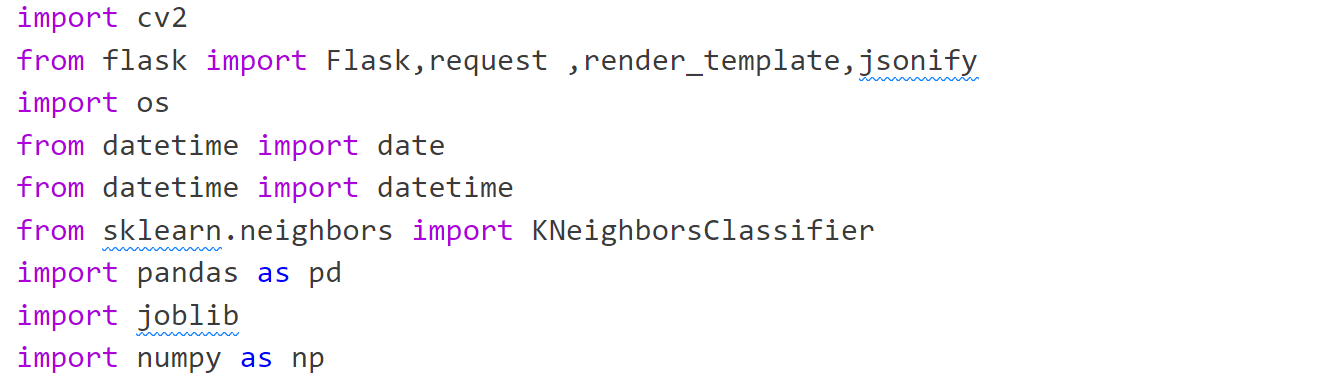
Check if the necessary directories attendance and static/faces and the attendance file for the current date do not exists. If they doesn’t exist then the code creates the directories and initializes the attendance file by writing the column headers. Ensure that the required file and directory structure is in place before further operations are performed

Then a list of files and directories present in the specified directories. In this case it returns a list of files in the static/faces directory. The function of length is then added to calculate the length of the returned list which corresponds to the number of files in the directory

This can be useful for if you want to know the total number of registered faces stored in the static/faces directory.

**Sign up.py**

First we import required libraries



Once you import cv2, you can use its functions and classes to perform various computer vision tasks and integrate them into your Python code. To install OpenCv module

Open a terminal or command prompt in vs code.

**Pip install opencv-python**

In this code we are using date function method to retrieve the current date and then formats it using the start time function method. The format string specified the desired format month two digits represent the day in two digits and year also represents two digits. For example if the current date is June 4, 2023 the function will return 06-04-2023.

The date function also uses the method to retrieve the current date. However the method is used with a different format string ’%d-%B-%y. %d in this string represent two digits %B represent the full moth named and &Y rearrests the full year in four digits. For example if the current date is June 5-2023 the function will return 05-june-2023.

After that we are creating one variable called face-detector, it creates an instance of the cascade classifier class and assigns it to the variable and it is the cascade classifier object is initialized with the Harcascade classifier model for faces detection by providing the path to the XML file. Face detector object can then be used to detect faces in images using the detect multiscale method which applies the cascade classifier algorithm to image, so finally initializes a cascade classifier object for face detection using the haar cascade classifier algorithm. The XML file containing the pre-trained mode is specified by providing the file path. This cascade classifier object face-detector can be used to detect the faces in images by calling the detect multiscale function method on the images.

Then we take images and converts it to grayscale and uses the face detector cascade classifier to detect faces in the grey scale image. The function returns a list of rectangles representing the coordinates and sizes of the detect faces. Then code loops over the images in the "static/faces" directory, extracts the face features, and stores them along with their corresponding labels. It then trains a KNN classifier using the extracted features and labels. Finally, the trained model is saved to a file for future use in face recognition tasks.

The identify face function loads a trained face recognition model from a file and uses it to predict the identities of the faces in the input facearray. The predicted labels are then returned by the function. This allows you to identify the faces based on the trained model's classification.

It initializes an empty list called userimagedir to store the paths of user image folders. Opens the CSV file with the current date in append mode. The 'a' mode allows writing to the file while preserving its existing contents. The file will be used to store attendance information. This line closes the CSV file after opening it briefly. After that generates a random number between 0 and 999 and appends it to the 'static/faces/' directory to create a user image folder path.

Each user will have a separate image folder. After adds the user image folder path to the userimagedir list.

**if not os.path.isdir(userimagefolder)**

This line adds the user image folder path to the userimagedir list.

**os.makedirs(userimagefolder)**

If the user image folder does not exist, this line creates it using the os.makedirs() function.

The function creates any necessary intermediate directories as needed.

Cap= cv2.VideoCapture(0) this line initializes a video capture object to capture frames from a camera. The argument 0 specifies the index of the camera to be used manually

i,j,=0,0 this line initializes two variables I and j to keep track of captured images

starts an infinite loop to continuously capture frames from the camera and capture the image frame by frame, and extract the faces function

This starts a loop over each detected face in the faces list.

The variables x, y, w, and h represent the coordinates and dimensions of a face bounding box.

calculates the centre coordinates of the face bounding box.

calculates the radius of a circle that encompasses the face bounding box, draws a circle around the detected face in the captured frame.

The circle is drawn with a red color (0, 0, 255) and a thickness of 6, adds a text overlay to the captured frame.

The text indicates the number of images captured so far out of the desired limit of 50.

if j is divisible by 10, which means every 10th face will be captured, generates a unique image name by concatenating the current image count (i) with the file extension '.jpg', saves the captured face image within the user's image folder.

The image is cropped to the bounding box coordinates before saving and increments the image count (i) to keep track of the number of captured images and increments the loop counter (j) for every iteration. if the loop counter (j) reaches 500.

It determines the maximum number of frames to capture before stopping the loop then breaks the loop, ending the image capture process it to retrieve attendance data from a CSV file.

The function returns various attendance-related information, such as names, dates, rolls, start times, end times, and the total number of records.

returns a JSON response indicating the success of the '/add' endpoint and captures face images from a camera, detects faces in each frame, saves the captured face images to user-specific folders, and updates the attendance CSV file. The function returns a JSON response indicating the success of the '/add' endpoint.

After the image captures we are gives employee details retrieves the value of the 'username' parameter from the request's query parameters.

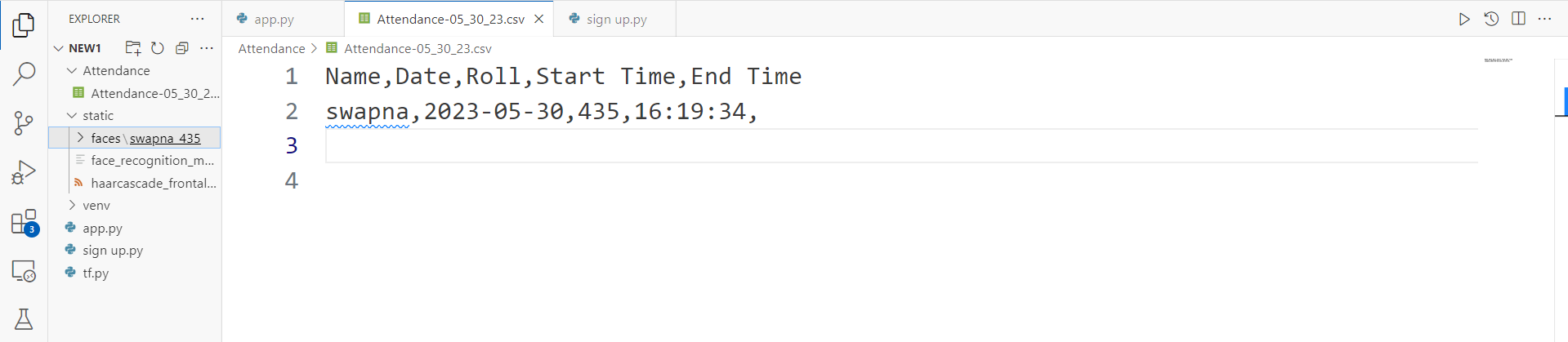
The value is converted to a string and assigned to the newusername variable, user id variable with current date and at present time. If the combination of newusername and newuserid is not equal to Name, Roll, it ensure that the username and userid are not the default values.

Open the attendance file with the current date in append mode it allows writing to the file while preserving its existing contents. The file will be used to store attendance information, and attendance csv file containing the new user name current date ,new user id and current time the values are formatted as a string and append to the file train model function trains the face recognition model using the collected face images and extract the attendance function to retrieve attendance data from the csv file it returns various attendance related information such as names, dates, rolls, star times, end times and the total number of records and finally try block exicutes read the attendance CSV file using pd.read\_csv. If the file is empty or does not exist, an EmptyDataError exception is raised. In such a case, an error message is printed, and the function returns without further execution.

**SAMPLE IMAGES FOR SING UP:**

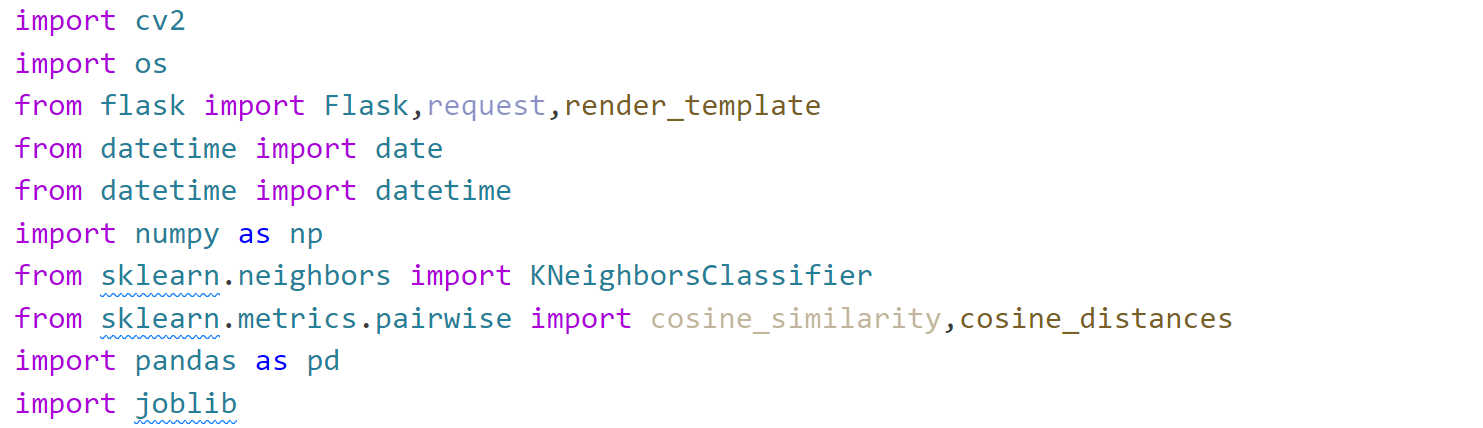
****

****

****

**Sign in .py**

First we are importing the required libraries



Once you import cv2, you can use its functions and classes to perform various computer vision tasks and integrate them into your Python code. To install OpenCv module

Open a terminal or command prompt in vs code.

**Pip install opencv-python**

If you have multiple versions of Python installed, make sure to use the appropriate pip command associated with the Python version you intend to use with OpenCV.

Wait for the installation to complete. This will download and install the OpenCV package and its dependencies.

Once the installation is finished, you should have OpenCV installed and ready to use in your Python environment. You can import it in your code using the import cv2 statement.

By importing the KNeighborsClassifier class, you can create an instance of this class and use it to train a k-nearest neighbors classifier model. This algorithm is a type of supervised learning, where the classification of a new data point is based on the class labels of its nearest neighbors in the training data.

The cosine\_similarity function calculates the cosine similarity between two vectors. Given two vectors X and Y, the cosine similarity measures the cosine of the angle between them. It provides a similarity score between -1 and 1, where 1 indicates perfect similarity, 0 indicates no similarity, and -1 indicates perfect dissimilarity. The function takes the input vectors and returns a similarity matrix.

The cosine\_distances function, on the other hand, calculates the cosine distances between two vectors. This distance metric is complementary to the cosine similarity. Instead of measuring similarity, it measures dissimilarity. The cosine distance between two vectors X and Y is defined as 1 - cosine\_similarity(X, Y). It provides a distance score between 0 and 2, where 0 indicates perfect similarity, and 2 indicates perfect dissimilarity. The function takes the input vectors and returns a distance matrix.

In the provided code, it seems that the cosine\_distances function is used to compute the cosine distance between the identify\_faces vector (representing a known face) and the face vector (representing a detected face). The resulting similarity value is compared against a threshold of 0.14 to determine if the detected face is authorized or unknown.

In this code we are using date function method to retrieve the current date and then formats it using the start time function method. The format string specified the desired format month two digits represent the day in two digits and year also represents two digits. For example if the current date is June 4, 2023 the function will return 06-04-2023.

The date function also uses the method to retrieve the current date. However the method is used with a different format string ’%d-%B-%y. %d in this string represent two digits %B represent the full moth named and &Y rearrests the full year in four digits. For example if the current date is June 5-2023 the function will return 05-june-2023.

Already we trained the images and store the images folder wise now extract the username and userid from the given name parameter. The name is expected to be in the format "username\_userid", so the split('\_') method is used to separate the username and userid based on the underscore. capture the current date and time using the date.today and datetime.now functions respectively. The time is formatted as "%H:%M:%S" to represent the hours, minutes, and seconds. if the given userid is not present in the 'Roll' column of the attendance DataFrame df. If the condition is true, it means that the person's attendance record does not exist yet, and a new record is appended to the CSV file. The information includes the username, current date, userid, and current time, which are written as a new line in the CSV file.

If the person's attendance record already exists (i.e., the userid is present in the 'Roll' column of the DataFrame), this block is executed. The code retrieves the index of the existing record where the 'Roll' column matches the userid. The 'End Time' value for that record is updated with the current time, saves the modified DataFrame back to the attendance CSV file, overwriting the previous file content ensures that the index column is not included in the saved CSV file.

Overall, this function adds attendance information to the CSV file by appending new records or updating existing records with the end time.

if the file 'face\_recognition\_model.pkl' is not present in the 'static' directory. It verifies if the trained face recognition model exists.

If the model file is not found, the code returns a rendered HTML template with a message stating that there is no trained model in the static folder, and it prompts the user to add a new face, initializes the video capture object to capture frames from a camera with index 1. Here, the index 1 indicates the second camera connected to the system.

It assumes that a camera is available and accessible for capturing video frame continuously capture frames and perform face recognition

captures a frame from the video feed using the video capture object cap. The read() function returns two values: ret, a Boolean indicating whether the frame was successfully read, and frame, the captured frame to detect faces in the captured frame. It returns a list of face bounding box coordinates.

if any faces were detected in the captured frame, extracts the coordinates of the first detected face from the faces list, calculate the centre and radius of a circle to be drawn around the detected face, extracts the detected face region from the frame, resizes it to a fixed size of (50, 50), and assigns it to the face variable.

**identified\_person = identify\_face(face.reshape(1, -1))[0]:**

it calls the identify\_face function to recognize the person in the captured face. converts the face image into a 1D array and passes it to the identify\_face function.

The function returns the predicted label (identified person) for the face, and it is assigned to the identified\_person variable.

reads the image of the identified person (identified by the label) from the corresponding directory in the 'static/faces' folder, resizes the identified person's face image to the same size as the captured face (50x50) and reshapes it into a 1D array, reshapes the captured face into a 1D array.

To calculates the cosine distance between the identified person's face and the captured face.

The cosine\_distances function from the scikit-learn library is used for this calculation.

The resulting similarity value represents the dissimilarity between the faces, with lower values indicating higher similarity.

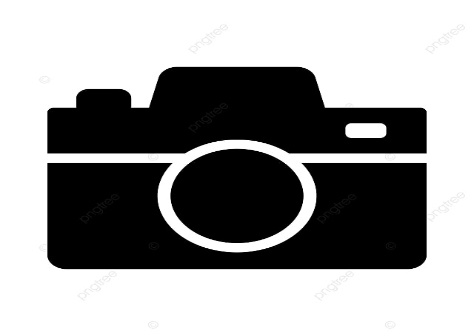
if the similarity value exceeds a threshold of 0.14.

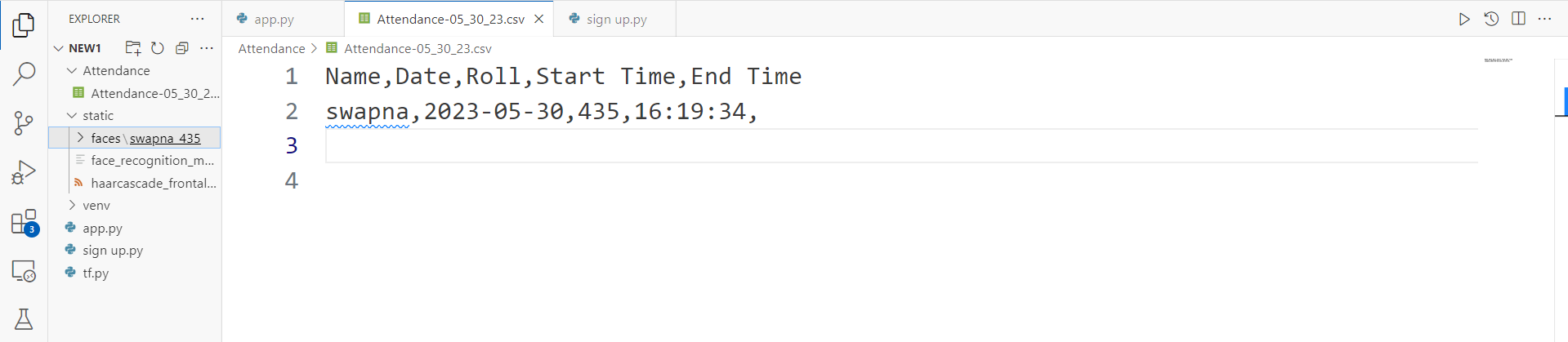
If the condition is true, it means the captured face is recognized as an authorized person.

The code updates the d dictionary with a 'Status' key indicating that the recognition was successful.

It also calls the add\_attendance function to add the attendance record for the identified person. If the captured face is not recognized as an authorized person executed the else block. It simply adds a text overlay to the frame indicating that the person is unknown

**SAMPLE IMAGES OF SIGNIN:**

** **

****

**Visitor.py**

Create two empty lists faces and labels these are two lists will be used to store the face images and corresponding labels for training the model. Create variable user list its retrieves the list of the subdirectories in the static/faces directory. Each sub directory represents a different user, then we are entered for loop it nested loops iterate over each user’s subdirectories and each image file with in that directory, and Reads the image file using OpenCV's imread function.

The image path is constructed based on the current user and image name, resizes the image to a fixed size of (50, 50) pixels using OpenCV's resize function.

The resized image is assigned to the resized\_face variable, add the flattened (1D) version of the resized face image to the faces list. The corresponding user label (subdirectory name) is added to the labels list. Converts the faces list into a NumPy array. The shape of the array will be (num\_samples, num\_features), where num\_samples is the total number of face images and num\_features is the flattened size of each face image. creates an instance of the KNeighborsClassifier class from scikit-learns neighbors module.

The n neighbors parameter is set to 1, indicating that the algorithm will consider only the closest neighbour for classification, trains the KNN classifier by fitting the face images (faces) and their corresponding labels (labels).

The algorithm learns the relationship between the face images and their associated users. saves the trained KNN model as a binary file using the joblib library.

The model file is saved with the name 'face\_recognition\_model.pkl' in the 'static' directory.

The saved model can be loaded later for face recognition tasks.

In summary, the train\_model function iterates through the user directories, reads the face images, resizes them, and constructs the training dataset. It then trains a KNN classifier on the dataset and saves the trained model for future use.

Add\_attendance is used to add attendance records to a csv file. The function takes a parameter name which contains the information of the visitor. The name is split using underscores to extract the username, Mobile Number, and Visitor Purpose. It tries to read the CSV file with the attendance data using pd.read\_csv. If the file is empty or doesn't exist, it catches the pd.errors. EmptyDataError exception and prints an error message. If the Mobile Number is not already present in the Mobile Number column of the DataFrame, it means the visitor is new and their attendance record needs to be added. The function opens the CSV file in append mode and writes a new line with the visitor's details including username, current date, Mobile Number, Visitor Purpose, and current time.

If the Mobile Number is already present in the DataFrame, it means the visitor is exiting, and the function updates the corresponding row's 'Exit Time' with the current time.

Finally, the updated DataFrame is saved back to the CSV file using df.to\_csv().This function is responsible for maintaining the attendance records by adding new entries and updating exit times for existing entries. The file face\_recognition\_model.pkl exists in the static folder. If it doesn't exist, it renders a template called home.html with some parameters (totalreg, datetoday2, and mess) and returns the rendered template. This is likely done to indicate that a trained model is required before proceeding. If the model file exists, the function proceeds to capture video from a webcam using cv2.VideoCapture (0). The number 1 indicates the index of the webcam device to be used. Inside the while loop, it reads frames from the video capture using cap.read function. The variable ret indicates whether a frame was successfully read.

If faces are detected in the frame using extract\_faces function, the first detected face's coordinates are extracted. A rectangle is drawn around the face using cv2.rectangle .

The face region is resized to a smaller size (5x2) using cv2.resize ().

The face is passed to the identify\_face function to determine the identity of the person. The identified person's name is stored in the variable identified\_person. The add\_attendance function is called with identified\_person as the argument to update the attendance records.

The person's name is displayed on the frame using cv2.putText .The frame with the drawn rectangle and text is displayed using cv2.imshow. If the escape key (key code 27) is pressed (cv2.waitKey (3)==27), the loop breaks.

Finally, the video capture is released using cap.release and all windows are closed using cv2.destroyAllWindows().

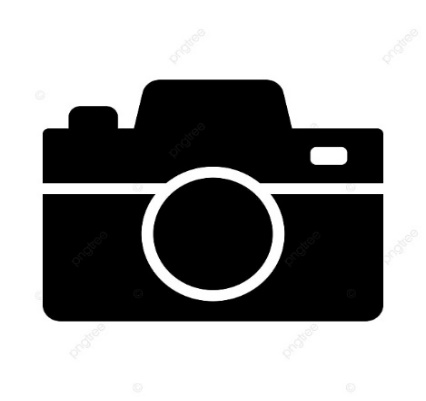
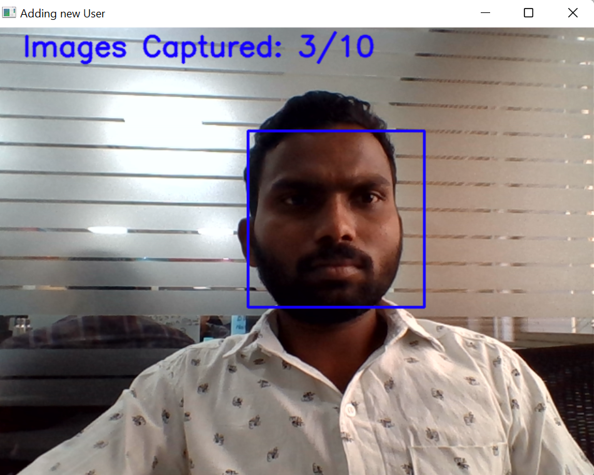
This code captures video from a webcam, detects faces, identifies the persons using a trained model, and updates the attendance records for the identified persons. add data() that handles data submission from a form. Here's an explanation of the code: The function retrieves the values submitted in the form fields newusername, Mobile Number, and visitor purpose using request Form. The current date and time are obtained using the date.today and datetime.now.strftime("%H:%M:%S") functions, respectively. A new name is created by concatenating newusername and Mobile Number with an underscore.

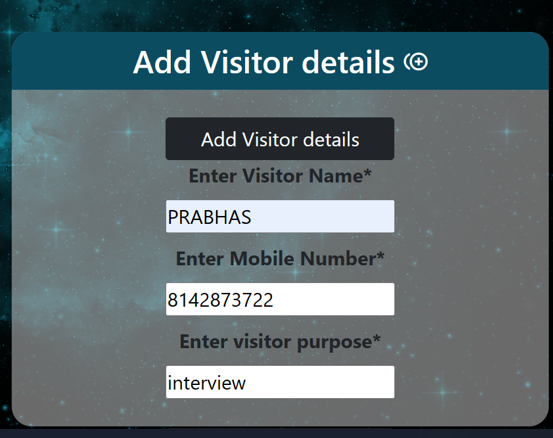
The image directory (userimagedir[-1]) is renamed to the new name by using os.rename(). It modifies the path of the image directory.

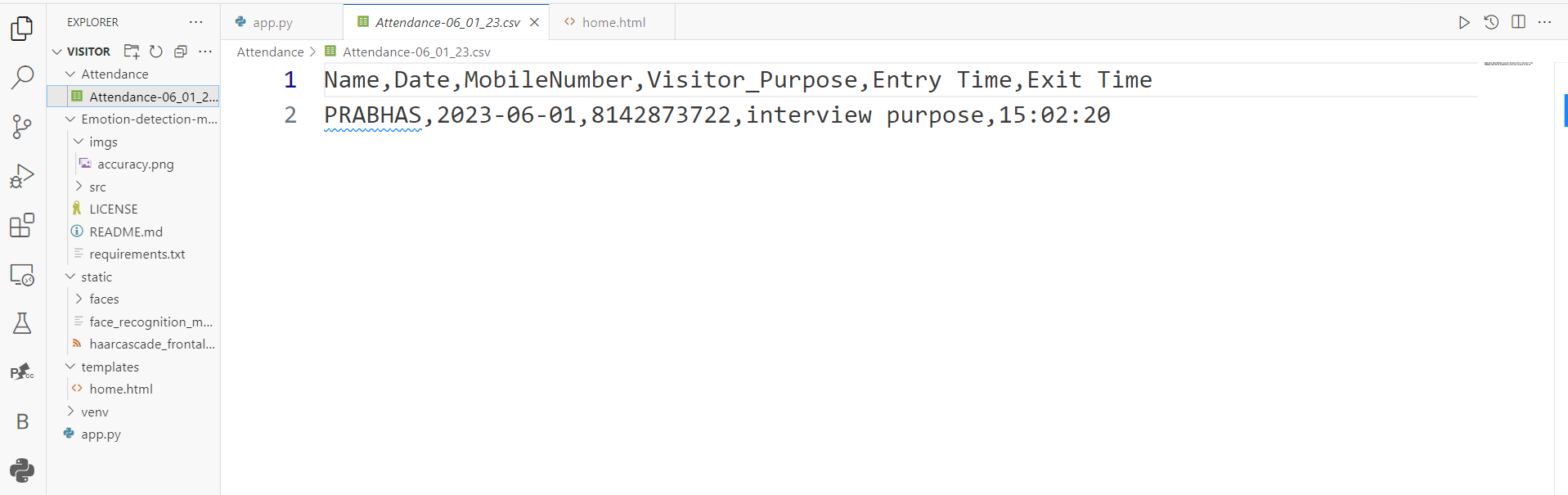
If the combination of newusername, Mobile Number, and visitor purpose is not present in the list ['Name', 'Roll'], the attendance details are appended to the CSV file (Attendance/Attendance-{datetoday()}.csv). The attendance details include newusername, current date, Mobile Number, visitor purpose, and current\_time. The code then proceeds to train the model by calling the train\_model function. After training the model, attendance details are extracted using the extract\_attendance function and assigned to variables names, date, rolls, visitor purpose, times, etime, and l. Finally, the rendered template home.html is returned with the attendance details and other parameters (totalreg, datetoday2) to be displayed on the page.

This code handles the form submission, updates the attendance records, trains the model, and renders the updated attendance details on the home page.

**Sample images of visitor:**

** **

****

****

**Conclusion:**

Before the development of this project. There are many loopholes in the process of taking attendance using the old method which caused many troubles to most of the institutions. Therefore, the facial recognition feature in the attendance monitoring system can not only ensure attendance to be taken accurately and also eliminated the flaws in the previous system. By using technology to conquer the defects cannot merely save resources but also reduces human intervention in the whole process by handling all the complicated task to the machine. The only cost to this solution is to have sufficient space in to store all the faces into the database storage. Fortunately, there is such existence of micro SD that can compensate with the volume of the data. In this project, the face database is successfully built. Apart from that, the face recognizing system is also working well. At the end, the system not only resolve troubles that exist in the old model but also provide convenience to the user to access the information collected by mailing the attendance sheet to the respected faculty