A FIELD PROJECT REPORT

On

“Face Recognition System Based Attendance”

Submitted
by

221FA04102 R.Sowmya	221FA04534 M.Lasya
221FA04551 N.Tejaswar Rao	221FA04561 Y.Sahith

Under the guidance of

Dr. Sajida Sultana

Assistant Professor
Dept of CSE
Vignan University

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

VIGNAN'S FOUNDATION FOR SCIENCE, TECHNOLOGY AND RESEARCH Deemed to be UNIVERSITY

Vadlamudi, Guntur.

ANDHRA PRADESH, INDIA, PIN-522213.

VIGNAN'S
FOUNDATION FOR SCIENCE, TECHNOLOGY & RESEARCH
(Deemed to be University) - Estd. u/s 3 of UGC Act 1956

CERTIFICATE

This is to certify that the Field Project entitled **"Face Recognition System Based Attendance"** that is being submitted by 221FA04102(R.Sowmya), 221FA04534(M.Lasya), 221FA04551(N.Tejaswar Rao), 221FA04561(Y.Sahith) for partial fulfilment of Field Project is a bonafide work carried out under the supervision of Ms. Sajida Sultana.Sk , Assistant Professor, Department of CSE.

Guide name& Signature	[illegible] Dr. S. V. Phani Kumar	[illegible] Dr.K.V. Krishna Kishore
Assistant/Associate/Professor, CSE	HOD,CSE	Dean, SoCI

VIGNAN'S
FOUNDATION FOR SCIENCE, TECHNOLOGY & RESEARCH
(Deemed to be University) - Estd. u/s 3 of UGC Act 1956

DECLARATION

We hereby declare that the Field Project entitled **"Face Recognition System Based Attendance"** that is being submitted by 221FA04102(R.Sowmya), 221FA04534(M.Lasya), 221FA04551(N.Tejaswar Rao), 221FA04561(Y.Sahith) in partial fulfilment of Field Project course work. This is our original work, and this project has not formed the basis for the award of any degree. We have worked under the supervision of Ms. Sajida Sultana.Sk ,, Assistant Professor, Department of CSE.

By

221FA04102 (R.Sowmya),
221FA04534 (M.Lasya),
221FA04551 (N.Tejaswar Rao),
221FA04561 (Y.Sahith)

Date:

ABSTRACT

Historically, keeping track of attendance has been a labor-intensive and prone to error procedure that frequently calls for human inputs and checks. Automation has become the go-to method for modern answers to these problems, and face recognition technology is one of the most promising ones. This system detects and recognizes faces collected by a real-time camera stream, using machine learning techniques to automate the attendance process. In contrast to traditional systems that use pre-registered datasets, this method uses on-site manual picture training. When people arrive, their faces are photographed by the system, which uses this information to train the model in real time and help adapt the recognition process to the particular conditions and surroundings.

The system is made to work with a live webcam to take pictures of people in real time under different circumstances. Various lighting conditions were examined during the development process to guarantee reliable and precise facial recognition. To assess the system's performance in low-light or dimly illuminated areas, for instance, it was tested without direct light focus. The model was also trained to identify faces with subtle angles, wearers of spectacles, and variations in facial emotions. The system's capacity to adjust to varying lighting conditions and facial expressions improves its dependability and guarantees constant performance in many settings, rendering it more suitable for real-world use. This technology's automated, non-intrusive operation, which streamlines the procedure by removing the need for people to carry identifying tools or make physical contact with a device, is one of its main advantages. Furthermore, the system's adaptability to different locations, including corporate offices, educational institutions, and event sites, is enhanced by its flexibility to be modified on-site. The system's integration of sophisticated machine learning techniques guarantees dependable, secure, and effective attendance tracking. This solves the drawbacks of conventional systems and enhances both productivity and security. Our face recognition system is designed with Python programming and the **OpenCV library**. The **Local Binary Patterns Histograms (LBPH)** algorithm is the particular face recognition technique used in this implementation.

Key words - LBPH (Local Binary Patterns Histograms), Haar Cascade, Attendance Monitoring,Tkinter,Real-timeVideoCapture

Table of Contents

LIST OF FIGURES

CHAPTER-1

INTRODUCTION

INTRODUCTION

1. Introduction

It uses an advanced Face Recognition Attendance Monitoring System that is intended to automate the process of tracking attendance in learning environments. By utilizing sophisticated computer vision algorithms, this technology makes it possible to identify students in real time based just on their facial traits. The Haar Cascade Classifier for face detection and Local Binary Patterns Histograms (LBPH) for face identification are the main techniques used in this application.

1.1 What is face recognition?

A face recognition attendance system is a technical advancement that automates the attendance marking process by utilizing facial recognition technology.This technology records attendance without requiring face-to-face interaction by taking pictures of people's faces and comparing them to a database of registered users that has already been pre-stored.

1.2 Algorithms used

1.2.1 What is LBPH?

After face detection is finished, the discovered faces are recognized using the LBPH algorithm. LBPH is a powerful technique that uses Local Binary Patterns (LBP), a coding system that gives binary values to pixels based on their intensity in relation to their neighboring pixels, to analyze the texture of faces. A histogram is created from this texture data, and it provides a concise representation of the features found in the facial image. During the training phase, each student's multiple labeled images are gathered, transformed into LBP representations, and their histograms are stored. When a face is recognized in real-time, the system computes its histogram and compares it with the previously stored histograms to identify the most confidently closest match.

1.2.2 What is HCC?

One well-known method in computer vision is the Haar Cascade algorithm, which is very useful for identifying objects in pictures and movies. It functions as the first stage of the face recognition procedure in this attendance system. The system detects the existence of a face using a sequence of trained classifiers based on rectangular Haar-like features, which record variations in intensity within an image. The classifier analyzes regions for face-like characteristics by swiping a detection window across the picture while it processes the incoming video feed from a webcam. In the event that a face is found, the algorithm provides the detected region's coordinates, which are used in the recognition process's further stages.

CHAPTER -2
LITERATURE SURVEY

2.1 Literature Survey

Kennedy Okokpujie, Etinosa Noma-Osaghae,et al.[1] The research journal “Face Recognition Based Attendance System with GSM notification” is based on the identification of face recognition to solve the previous attendance system’s issues.This paper presents a stress-free non-intrusive way of taking class attendance using face as the biometric It also has the added novelty of relaying vital information about class attendance to handheld devices via any available cellular network.

During enrolment, a camera was used to acquire facial images that were made into templates using Fisher faces algorithm. These templates were stored in a database. During verification or attendance taking, facial features extracted from acquired face images and stored picture templates were compared using Fisher Linear Discrimination algorithm for any match within the pre-set threshold. Vital information about collated attendance reports were sent via a cellular network to designated handheld devices.The designed and implemented system had 54.17% accuracy during verification when lighting was varied without any variation in facial expression during enrolment. The system had 70.83% accuracy during verification when facial expressions were varied along with variations in lighting conditions during enrolment.

Xiaoxiang Xu, Li Zhang, Fanzhang Li [2] The research journal “multi-scale feature extraction for single face recognition” proposed a solution for single face recognition of the attendance.Single sample face recognition has always been a hot but difficult issue in face recognition. The existing methods solve this issue from selecting robust features or generating virtual samples. By considering selecting robust features and generating virtual samples simultaneously, this paper proposes a multi-scale support vector transformation (MSSVT) based method to generate multi-scale virtual samples for single image recognition. Experimental results on three face data sets verify that the proposed algorithm retains most information and has the best performance compared with other related algorithm. Face recognition has always been one of the hotspots in the fields of pattern recognition, image processing, machine vision and neural network. The so-called face recognition is a computer system that detects the static image or the face image in dynamic video, matches 7 with the stored face database in computer, and then performs single or multiple face identification

Changjiang Jiang,Mingyi Wang,et al.[3] The research journal “a face recognition algorithm based on sparse representation and feature fusion”. In order to improve the accuracy of face recognition, a face recognition algorithm based on sparse representation and feature fusion is proposed.Firstly, the training samples and test samples are pre-processed by grey image conversion, scale scaling, histogram equalization, smooth filtering and so on, and the LBP, Gabor and HOG features of face images are extracted. And then the RSC classification test is carried on the partial samples. A loss function is defined according to the recognition result and the classification residual, then the weight vector is obtained by using the regularized least square method to minimize the loss function. Finally, the final residual is calculated according to the weight vector so as to obtain the final classification result.The experimental results on AR face dataset and LFW face dataset show that the recognition rate of our algorithm is obviously higher than the single feature recognition method, and it is robust to illumination, occlusion and expression.

SamridhiDev, Tushar Patnaik[4] the research journal “Student Attendance System using Face Recognition” . This paper is therefore proposed to tackle all these problems. The proposed system makes the use of Haar classifiers, KNN, CNN, SVM, Generative adversarial networks, and Gabor filters. After face recognition attendance reports will be generated and stored in excel format. The system is tested under various conditions like illumination, head movements, the variation of distance between the student and cameras. After vigorous testing overall complexity and accuracy are calculated. The Proposed system proved to be an efficient and robust device for taking attendance in a classroom without any time consumption and manual work. The system developed is costefficient and need less installation. The development of this system is aimed to accomplish digitization of the traditional system of taking attendance by calling names and maintaining pen-paper records. Present strategies for taking attendance are tedious and timeconsuming. Attendance records can be easily manipulated by manual recording. The traditional process of making attendance and present biometric systems are vulnerable to proxies. Attendance system using face recognition is a procedure of recognizing students by using face biostatistics based on the high-definition monitoring and other computer technologies.

Nico Surantha , Boy Sugijakko[5] Lightweight face recognition-based portable attendance system with liveness detection**,** In order to improve security against spoofing attacks, the researchers in this study created a lightweight face recognition attendance system that incorporates liveness detection. The team added a liveness detection step before subject recognition since they were aware of how readily impersonators may fool typical facial recognition systems. Because the system is meant to run on a Raspberry Pi, it can be easily moved around and implemented in a variety of settings. They examined a number of pre-trained models in order to attain the best results, and they ultimately decided on MobileNetV2 due to its efficacy and efficiency. Because the model was trained utilizing a transfer learning methodology, it was able to swiftly adjust to the particular needs of the task. The suggested approach produced remarkable outcomes, with an accuracy rate of 95% and an average processing time of less than 0.6seconds

Bao-Thien Nguyen-Tat, Minh-Quoc Bui, Vuong M. Ngo[6] Automating human resources' attendance tracking highlights how revolutionary computer vision and facial recognition technologies may be. Automated solutions that improve accuracy and efficiency are increasingly being used in place of traditional attendance systems, which are frequently human and prone to errors. Research shows that even in difficult situations like changing lighting or partial occlusions, facial recognition systems can reliably identify people. Furthermore, studies show that incorporating these technologies into current HR frameworks can increase data accuracy, decrease administrative workloads, and expedite attendance tracking.

Joshita Malla,et al.[7] Existing security systems confront serious issues from the growing complexity of artificial intelligence (AI)-generated faces, especially in crucial settings like airport security checks.Research shows that many facial recognition algorithms have trouble telling the difference between real faces and high-quality deepfakes, which raises fundamental questions about how well they work to prevent unwanted access. This calls for the creation of sophisticated machine learning techniques that can swiftly adapt to the changing nature of these threats and learn from them. The dependability of the technology used for security, whether in online transactions or physical security settings, needs to be strictly guaranteed as AI-generated faces get more realistic. In order to address the ethical and technological ramifications of this quickly developing field, research is concentrated on developing strong AI solutions that accurately identify fake faces.

Olufemi S. Ojo,et al.[8] Real-time Face-based Gender Identification System Using Pelican Support Vector Machine with important applications in targeted marketing, human-computer interaction, and surveillance, gender recognition from video is a new area of study that automatically classifies people's gender. This paper introduces a gender identification system that combines a Support Vector Machine (SVM) classifier with the Pelican optimization technique. To optimize classification performance, the Pelican optimizer improves the choice of SVM parameters, such as kernel functions. The suggested method produces impressive results by using preprocessing and feature extraction approaches like Local Binary Pattern (LBP), showing a 95% accuracy and 94.4% sensitivity on benchmark datasets containing labeled gender information. When compared to other approaches, these results highlight how well the Pelican Optimization Algorithm-SVM model performs in enhancing gender categorization from video data.

Gabriela Moise, Elena S. Nicoară[9] Ethical aspects of automatic emotion recognition in online learning**,** The emergence of AI technology has a big impact on education, especially when it comes to online and e-learning platforms, which frequently can't adapt the learning process to each student's unique characteristics and emotional states. Automatic emotion identification systems are becoming more and more popular in online learning environments as a result of the increased awareness of the role emotions play in improving learning outcomes. The research does, however, show a significant lack of representation in legislative frameworks governing AI technologies with relation to ethical implications in affective computing. This study draws attention to a number of hazards and ethical issues with using automatic emotion recognition in classroom settings.The work offers a model for responsible emotion detection systems intended to handle ethical difficulties and give a specific usage by recognizing both the positive implications and complicated ethical challenges.

Abraham Woubie,et al.[10] Maintaining Privacy in Face Recognition Using Federated Learning Method, Modern face recognition algorithms frequently use large image datasets that are gathered from users, however because these datasets contain sensitive personal data, privacy issues are frequently raised. This study investigates the use of federated learning, which uses decentralized edge devices to train a shared model without requiring the sharing of personal information in order to address these problems. In order to increase data diversity without transmission, each device can independently train its own model and create fake data using generative adversarial networks. The use of federated learning in supervised and unsupervised face recognition not only protects user privacy but also maintains performance levels equivalent to individual models, according to experimental results using CelebA datasets.This strategy emphasizes the crucial equilibrium between guaranteeing privacy protection and attaining a high level of accuracy in facial recognition software.

N. Bergman,et al.[11] Biometric identification of dairy cows via real-time facial recognition, the uncontrolled movements of animals, biometric techniques that were previously used for human identification are now being modified to identify dairy cows. However, this presents considerable hurdles in terms of accuracy and robustness. This study uses cutting-edge deep-learning algorithms to present a novel method for multiple-cow face detection and classification from video. Two annotated datasets were gathered: one for 77 cows' facial detection and another for their facial categorization. The system was set up at a dairy farm, four meters above a feeding zone. The system's mean average precision for facial detection using the YOLOv5 method was 97.8%, and its classification accuracy using a Vision-Transformer model was 96.3%. The system can handle real-time video at a rate of 50 frames per second and process video frames having up to 10 cow faces in less than 20 milliseconds.

Kritagya Painuly,et al.[12] Efficient Real-Time Face recognition-based attendance systems with deep learning, Using deep learning algorithms, effective real-time facial recognition-based attendance systems seek to improve the precision and timeliness of attendance management in a variety of institutions. To guarantee high reliability in face recognition, this suggested method combines real-time processing and information gathering with reliable face detection and model selection. The system successfully handles the difficulties related to attendance tracking by utilizing cutting-edge approaches, such as dlib, cvzone, and YOLO algorithms for face detection, as well as blur detection methods to increase system efficiency.

Muhammad Ahmad Nawaz Ul Ghani 1,et al.[13] Enhancing Security and Privacy in Distributed Face Recognition Systems through Blockchain and GAN Technologies, In order to overcome these obstacles and guarantee precise face recognition, this work presents a unique architecture that blends distributed computing, blockchain, and Generative Adversarial Networks (GANs). The suggested solution provides scalable and secure facial analysis while protecting user privacy by leveraging cutting-edge tools like Ray Cluster for distributed computing and Dlib for face analysis. The creation of a sustainable solution for privacy-aware face recognition and the application of adaptable privacy computing techniques through Blockchain networks are two significant contributions. The StyleGAN model achieved an astounding accuracy rate of 93.84% on high-resolution images from the CelebA-HQ dataset.

2.2 Motivation

This code automates attendance management in educational institutions by implementing an effective face recognition-based attendance tracking system that makes use of cutting-edge deep learning algorithms.It seeks to improve the precision and dependability of attendance tracking by incorporating features for picture capture, facial recognition, and user identification. The system offers a user-friendly interface for smooth operation and places a high priority on user privacy and data protection.

CHAPTER-3
PROPOSED WORK

3.PROPOSED WORK

Using OpenCV for image processing and facial recognition, and the Tkinter toolkit for its graphical user interface (GUI), the accompanying Python code builds a Face Recognition-Based Attendance Monitoring System. By automatically recognizing students' faces and documenting their presence in real-time, this system efficiently streamlines the attendance recording process.With fields for entering a student's ID and name, action buttons for taking pictures, saving profiles, and tracking attendance, and frames that arrange functions like registration and attendance tracking, the GUI is made to make user interaction easier.

The program activates the webcam and takes multiple pictures of the student's face, which are saved in a designated training images folder to create a dataset for the recognition algorithm. This feature of the system is crucial because it allows new students to be registered by capturing their facial images. After photos are taken, the application converts them to grayscale for processing and looks for required files, such as the Haar Cascade classifier for facial recognition. Real-time functionality is supported for facial recognition using an effective Local Binary Patterns Histograms (LBPH) recognizer.

In order to streamline and lower errors in attendance management, the trained model detects students' faces as they appear in the webcam's frame, matches them with registered profiles, and logs their presence together with the time and date in a CSV file. Only authorized users can make major changes thanks to the application's password protection function, which improves security for crucial tasks like saving profiles and changing passwords. All information is methodically saved in CSV files for simple maintenance and retrieval, including student profiles and attendance logs.

In addition to having strong error-handling features to handle possible problems, like missing files or incorrect password entries, the application offers real-time feedback through a dynamic display that updates attendance data as students are identified. Informing message boxes also assist users, which enhances the user experience overall.

3.1 Input data

- **Student Registration**: Through the GUI, students must manually enter their ID and name in order for the system to function. When recording attendance and registering photos, this data is crucial for creating a personalized profile for every student.

3.2 Data Preprocessing

1. **Image Capture and Transformation:**

    - **Face Detection:** The system uses Haar Cascades to detect faces in real-time throughout the image capturing process. Before executing the detection method, the acquired image must be preprocessed by converting it to grayscale using cv2.cvtColor. By simplifying the data and lowering dimensionality, grayscale photos speed up processing.

    - **Image Saving:** Using the student's ID and a special serial number as a guide, the taken face photos are stored in an organized manner. This orderly storage is made possible by the preprocessing step known as the "structured approach," which also makes it simpler to obtain and process images for training or recognition stages.

2. **Training Data Preparation:**
    - **Loading and Converting Images:**Images are loaded and converted to grayscale in the getImagesAndLabels method in order to get them ready for model training. In this step color information that is not needed for the recognition task is discarded and the necessary features are extracted

    - **Normalization:** Any best practice in the preprocessing step would usually involve normalizing picture pixel values, even though this isn't stated in the code explicitly. This ensures that the model can learn from variations in brightness and contrast in an effective manner. By lessening the impact of changing lighting conditions, normalizing the dataset would improve training. Although it isn't in the current code, this may be added either during training or after image capture.

3. **Data Validation:**
    The system makes sure that the data required for processing is available before any actions are carried out by verifying the existence of required files (such as the Haar Cascade classifier) and the training data. To guarantee the integrity of the data the system is processing, this validation step is essential.

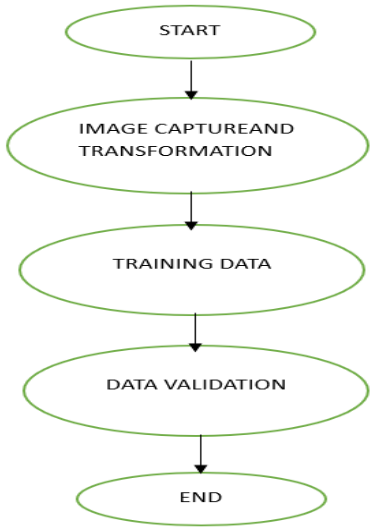


Figure 3.1 Flow diagram for Face Recognition System Based attendance

3.3 Model Building

In comparison to conventional approaches, the suggested system greatly increases efficiency and accuracy by automating the attendance taking process using facial recognition technology. This paradigm combines database administration, safe data processing, user interface design, and computer vision algorithms to enable facial recognition-based real-time attendance tracking.

3.3.1 Interface for Users:

The Face Recognition-Based Attendance Monitoring System's User Interface (UI) is created with Python's Tkinter toolkit, a powerful framework that makes creating graphical user interfaces easier. Clear functionality and user experience are ensured by the UI's separate frames devoted to student registration and attendance tracking. Users come across input areas in these frames where they can enter vital data such student names and IDs. Alongside this direct input feature are useful buttons that enable taking pictures, storing profiles, and starting attendance-tracking procedures. Additionally, the design includes dynamic displays that show real-time attendance statistics, giving users instant feedback and improving the system's overall usability.

3.3.2 Image Capture and Processing:

The system makes use of OpenCV, a potent computer vision framework that permits real-time image processing, for both image capture and processing. A webcam is used to take real-time pictures during the student registration process to guarantee that the information gathered is up-to-date and pertinent. The system uses the Haar Cascade Classifier, which effectively detects faces within images using machine learning approaches, to improve the accuracy of face detection. To enable accurate operation during live capture, this classifier is trained on a range of positive and negative data. Additionally, the collected photos are transformed to grayscale, which reduces the quantity of data that needs to be examined and increases processing efficiency.

3.3.3 Algorithm for Facial Recognition:

The facial recognition algorithm, which uses Local Binary Patterns Histograms (LBPH) for effective face recognition, is the brains behind the attendance system.Because it examines the local binary patterns of the facial features that are captured in the pictures, LBPH is especially useful for gaining a more sophisticated knowledge of a person's face. Face matching and feature extraction are the two primary stages of the recognition process. The system recognizes distinct facial features from the training images during feature extraction, and each image is represented as a histogram of local patterns. This approach streamlines the comparison procedure during face matching, in which the system determines identity by comparing the histogram of a recently obtained face with the database's stored histograms.

3.3.4 Attendance Management:

The system's attendance management feature is intended to make real-time student attendance recording more efficient. When a student's face is successfully identified, the system incorporates the feature into the live video feed and immediately tracks their attendance. With a timestamp attached to each attendance entry, a safe and verifiable record is produced, which is then saved in a CSV file for convenient maintenance and access. Not only does this automation make teaching easier, but it also improves the accuracy of attendance records, making it possible to track student engagement accurately over time.

3.3.5 Security and Password Protection Measures:

A crucial component of the system is security, particularly when it comes to sensitive functions like accessing student accounts and changing passwords. Strong password protection methods are put in place to combat this. In order to protect user credentials from unwanted access, the system uses hashed passwords, which are kept in a specific text file (TrainingImageLabel/psd.txt). To further improve the security of sensitive data and system activities, user input validation is also used to guarantee that only authorized actions can be taken.

3.3.6 Persistence of Data:

CSV files, which are a dependable way to store attendance data and student information, are used to achieve data permanence in the system. Because this method enables persistent storage, data may be efficiently maintained between sessions and retrieved with ease. The system's use of CSV format for these records makes data handling easier for administrators and teachers, enabling them to view and edit attendance data as needed while keeping it structured.

3.3.7 Mechanism for Real-Time Processing and Feedback:

For effective attendance tracking, the system is made to process and offer feedback in real-time. As pupils are identified, attendance records are dynamically updated via a live video feed. Teachers can quickly view a student's attendance status thanks to this instant feedback method, which improves the user experience overall. In order to ensure that the attendance status is clearly displayed in real-time and to streamline the entire process of taking attendance while reducing the possibility of errors, the GUI updates dynamically as students are identified. This feature of the system enhances the responsiveness and engagement of the attendance management system in addition to increasing operational efficiency.

3.4 Methodology of the system

The Face Recognition-Based Attendance Monitoring System automates the attendance process in educational environments by utilizing sophisticated algorithms and computer vision techniques. The methods used in this system is thoroughly described in the parts that follow.
This system's main objective is to provide automated attendance tracking through the use of facial recognition technology.The system reduces the manual overhead associated with standard attendance systems by using a webcam to take real-time photographs of students, process these images for facial recognition, and report attendance automatically.

3.4.1 System Components:

The system is made up of a number of essential parts, each of which is vital to its operation

1. **Design of User Interfaces:**

    - Python's Tkinter library is used in the development of the user interface, which offers a graphical user interface. It has buttons for taking pictures and saving profiles, input sections for entering student names and IDs, and a display area for showing attendance information.

    - The interface's smart design minimizes user errors during interactions by providing clear labels and directions.

2. **Capturing and Processing Images:**
    - The system uses a webcam to take pictures using OpenCV, a powerful image processing package.

    - The recorded photos are subjected to real-time face detection using a Haar Cascade Classifier. By recognizing facial traits, this classifier allows the system to concentrate on the areas of the face for additional processing.

    - The collected photos are converted to grayscale to maximize processing performance, as color information is not often necessary for good face identification.

3. **Algorithm for Facial Recognition:**

    - For facial recognition, the system uses the Local Binary Patterns Histograms (LBPH)

.Face matching and feature extraction are the two phases of this algorithm's operation.

- Feature Extraction: From the grayscale pictures, distinct facial features are taken out, and each face is shown as a histogram of regional patterns. Important facial features that are essential for recognition tasks are captured by this representation.

- Face Matching: The faces that are photographed during attendance tracking are compared to the histograms that are kept in a database. To find out if the detected face matches any registered faces, a recognition threshold is established. The related student's attendance is recorded if a match is discovered.

4. **Management of Attendance:**

When facial recognition is successful, the system instantly logs attendance information. This includes:

- retrieving the current date and timestamp in order to record the moment the student's attendance is confirmed.

- storing the student ID, name, date, and time of attendance in a CSV file with other attendance data. This offers a trustworthy attendance record that may be retrieved at a later time for administrative or reporting needs.

5. **Features of Security:**

To safeguard private user information, the system employs a number of security measures:

- Password Protection: Mechanisms for user authentication are implemented. Secure passwords are kept in a hashed format to keep unwanted access at bay.

- The integrity of attendance records is preserved since only authorized workers are able to alter attendance data or gain access to critical areas of the system.

- 4.1 Password Management: Users can safely modify their passwords through a special interface in the application. To increase security, the application verifies the previous password before permitting any modifications.

6. **Real-time processing and feedback:**

A crucial component of the system is real-time processing, which provides instant feedback and functionality:

- For the purpose of tracking attendance, a live video feed is kept up to date. The application recognizes faces in the feed and modifies the UI accordingly.

- To improve the user experience, the system gives users feedback messages, such as alerts in the event of problems or confirmations of successful image captures.

7. **Persistence of Data:**

The system uses CSV files to store attendance information and student records over time:

- The structured CSV format in which all attendance logs are stored makes it simple to manage and retrieve records over time. Data is easily accessible for review or analysis thanks to its file-based storage, which guarantees that it is not lost between sessions.

3.4.2 Algorithms used

- **The algorithms used in image processing:**

Mathematical ideas are essential to many image processing tasks in the context of facial identification. Among the notable mathematical operations are:

Converting Images:

By using the **formula gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)**, a colored image is
transformed into a grayscale image. The mathematical representation of this operation is:

$$GrayValue=0.2989\times R+0.5870\times G+0.1140\times B$$

where the values of the red, green, and blue pixels are denoted by \(R\), \(G\), and \(B\), respectively.
In order to provide a single intensity value that represents brightness—a crucial component of face
recognition—this algorithm averages the color channels.

- **Detection Faces using Haar Cascade:**

In order to detect faces, the system uses a Haar-like feature classifier, which basically assesses the changes in pixel intensity in different areas of the image. Rectangle features can be computed quickly by using the concept of integral pictures to define the computation for feature evaluation mathematically.

- **Face Detection Systems:**

The Local Binary Patterns Histogram (LBPH) approach, which the code employs, entails the following mathematical procedures:

Patterns of Local Binary (LBP):

The LBP operator uses the pixel intensity in a nearby neighborhood to convert a picture into a binary number. Regarding a pixel P The LBP has eight surrounding neighbors and can be mathematically stated as follows:

$$LBP = \sum_{i=0}^{7} (gi - gc) * 2i$$

Where:
gi is the grayscale version of the ith adjacent pixel

gc is the central pixel's gray level

An essential component of face recognition, this expression creates a binary pattern that represents the local texture surrounding the pixel.

Training Data Representation:

Arrays of pixel values from the facial photos make up the recognizer's training data. These pixel values' mathematical representation can be organized as follows:

$Faces{=}I1{,}I2{,}..{,}In$

symbolizes the picture matrix for every single face that was taken throughout the training stage.

Calculating Distance to Identify Faces:

A distance metric is frequently used to assess how similar the input face and the training face models are when the model predicts a face. A popular metric in machine learning, the Euclidean distance is defined as follows:

$$d = \sqrt{\sum_{i=1}^{n} (ai - bi)^2}$$

where two distinct faces are represented by the feature vectors a and b. The proximity of two images (or their feature representations) to one another in the feature space is indicated by this distance.

- **Logging Attendance:**

  Datetime functions are used to record timestamps while recording attendance. Mathematical processes involving time and date, such as determining the current date, formatting that may require integer conversions, and string manipulations, are reflected in the resulting data.

3.5 Model Evaluation

3.5.1 User Interface:

- **Technology:** Tkinter and Python

- **Functionality:** The student registration and attendance tracking frames in the GUI include:
  Enter the name and student ID in these areas.
  Buttons for taking attendance, storing profiles, and taking pictures.
  Data on attendance is shown dynamically.

3.5.2 Capturing and Processing Images

- **Technology:**OpenCV is the technology.

- **Functionality:** Takes real-time pictures for student registration using a webcam.
  Real-time face detection during image acquisition is accomplished with the Haar Cascade Classifier.
  To guarantee processing efficiency, images are reduced to grayscale and stored in a structured way for training.

3.5.3 Algorithm for Facial Recognition

- **Technology:**Local Binary Patterns Histograms (LBPH) was the algorithm used.

- The goal is to efficiently identify faces in photos by examining local binary patterns.

- **Procedure:**
  Feature Extraction: Uses LBPH to extract distinctive facial features from photos. A histogram of local patterns is used to represent each image.Face matching is the process of comparing the histograms of the photographed face with those that are kept in the database.

3.5.4 Management of Attendance

- **Functionality:** The system instantly logs attendance when facial recognition is successful

- Timestamps and attendance are recorded, forming a safe record in a CSV file.

3.5.5 Security and Password Protection Measures

- **Functionality:** Strengthens security for critical processes (such as profile access and password changes).uses passwords that have been hashed and saved in a text file called TrainingImageLabel/psd.txt.

- To stop illegal activity, the system verifies user input.

3.5.6 Persistence of Data

- **Technology:** CSV files for long-term preservation
- **Functionality:** Student information and attendance data are saved in CSV format, making it simple to manage and retrieve data between sessions.

3.5.7 Mechanism for Real-Time Processing and Feedback

- **Functionality:** Using a live video feed, the system updates attendance in real time.
- As students are identified, the GUI dynamically refreshes to display their attendance status as soon as possible.

Face recognition model evaluation is a thorough analysis of the models' performance using a rangeof indicators and methodologies. Usually, the procedure starts by dividing the dataset into subsets for training and testing. The training set is used to construct the model, and the testing set assesses how well it performs on data that hasn't been seen yet.

Accuracy, which quantifies the percentage of cases that are successfully classified, and precision, which shows the caliber of positive predictions, are important performance indicators. Recall, which measures the model's capacity to locate pertinent cases, and the F1 score, which strikes a compromise between precision and recall, are additional crucial metrics. Furthermore, a confusion matrix is frequently used to depict true positives, true negatives, false positives, and false negatives while evaluating models.

While benchmarking against common datasets like MegaFace or LFW (Labeled Faces in the Wild) enables uniform comparisons between models, ROC curves and the area under the curve (AUC) offer insights into model performance at different classification thresholds. To evaluate the model's performance under dynamic circumstances such changing illumination and face emotions, real-world testing is crucial.

Additionally, ongoing post-deployment monitoring guarantees the model's efficacy throughout time, taking into account changes in operational settings and demography. Last but not least, ethical issues are crucial, highlighting the necessity of assessing equity and possible biases in model performance across various demographic groups, including age, gender, and race. All things considered, the assessment of face recognition models is a complex procedure that strikes a compromise between technical performance and moral considerations to encourage efficient and fair use of technology.

CHAPTER – 4
IMPLEMENTATION

4.1 Environment Setup

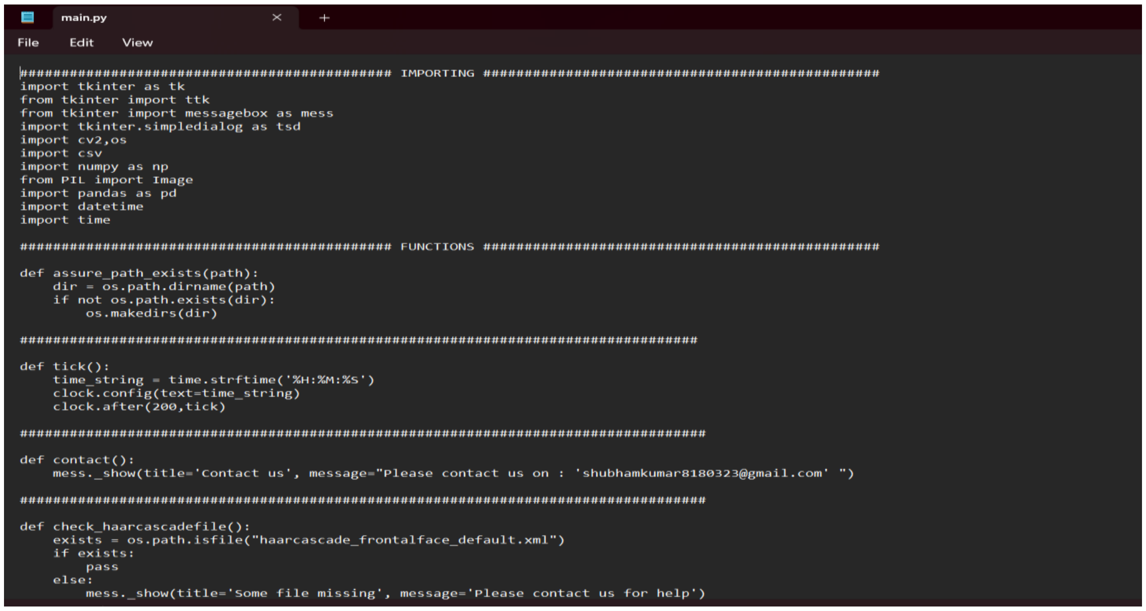
To use and implement the Face Recogniton System Based Attendance, you will set up a python Environment with the following libraries:

- Python is utilized because of its ease of use and adaptability, which allow for quick prototyping and development. It also has a strong ecosystem of libraries and frameworks that make jobs like image processing and GUI development easier.

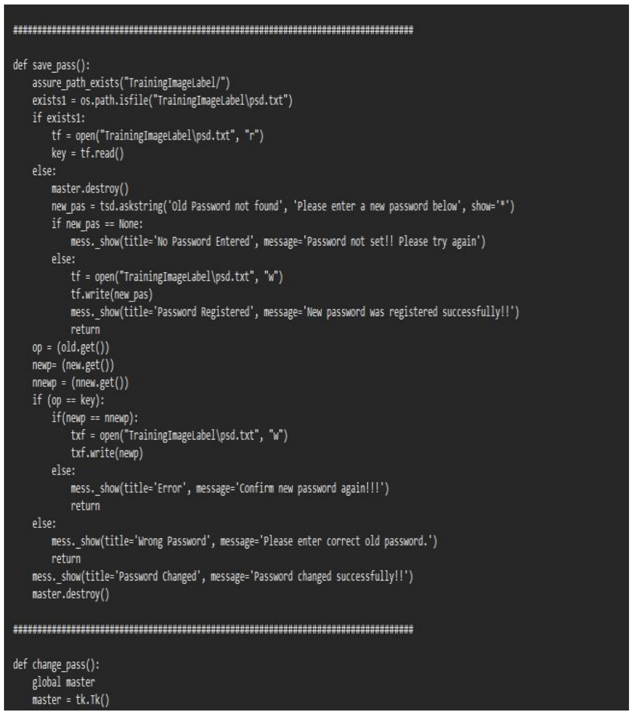
- **The libraries :**
    - tkinter,
    - tkinter.ttk,
    - tkinter.
    - messagebox,
    - tkinter.
    - simpledialog,
    - cv2,
    - os,
    - csv,
    - numpy, and
    - PIL

Time, date, image, and pandas.

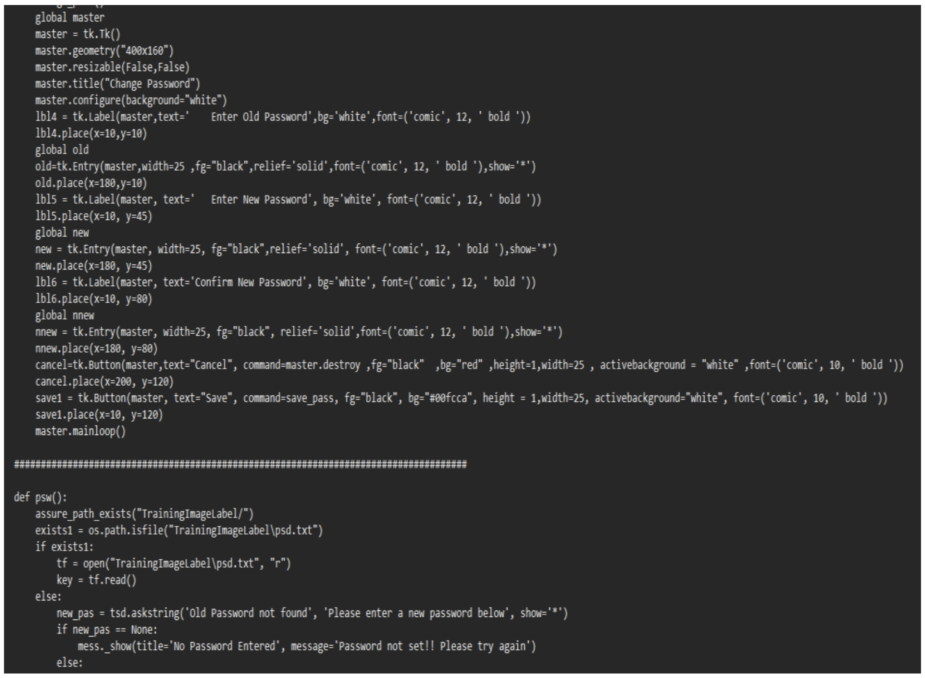
4.2 Code Implementation



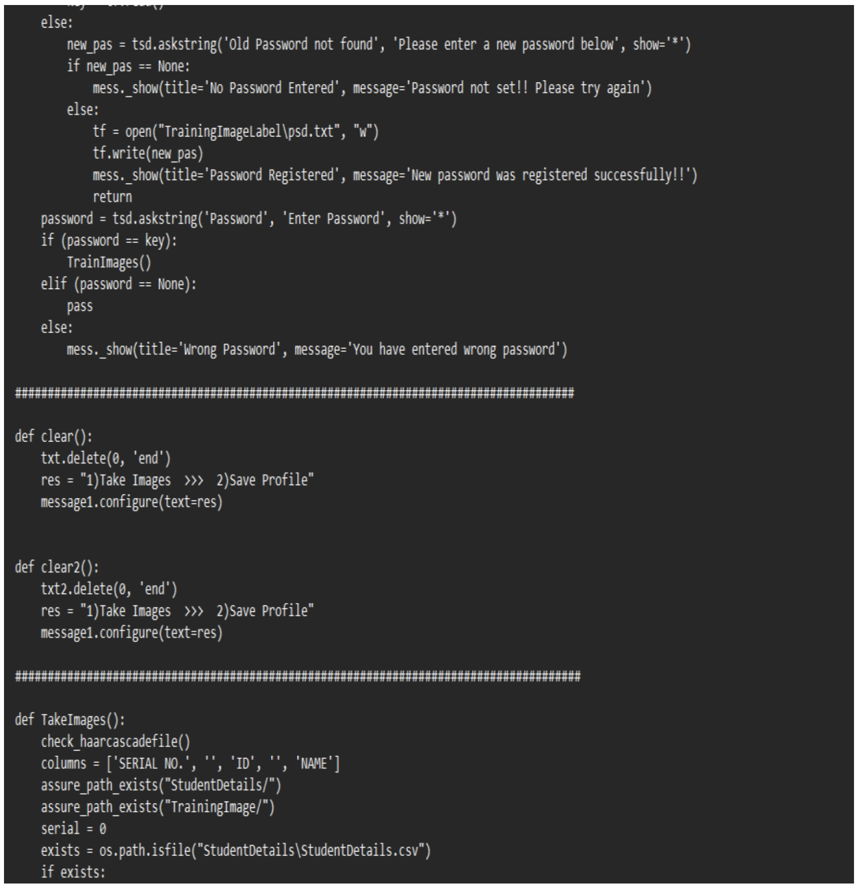
```
############################################# IMPORTING ################################################
import tkinter as tk
from tkinter import ttk
from tkinter import messagebox as mess
import tkinter.simpledialog as tsd
import cv2,os
import csv
import numpy as np
from PIL import Image
import pandas as pd
import datetime
import time

############################################# FUNCTIONS ################################################

def assure_path_exists(path):
    dir = os.path.dirname(path)
    if not os.path.exists(dir):
        os.makedirs(dir)

###################################################################################

def tick():
    time_string = time.strftime('%H:%M:%S')
    clock.config(text=time_string)
    clock.after(200,tick)

####################################################################################

def contact():
    mess._show(title='Contact us', message="Please contact us on : 'shubhamkumar8180323@gmail.com' ")

####################################################################################

def check_haarcascadefile():
    exists = os.path.isfile("haarcascade_frontalface_default.xml")
    if exists:
        pass
    else:
        mess._show(title='Some file missing', message='Please contact us for help')
```



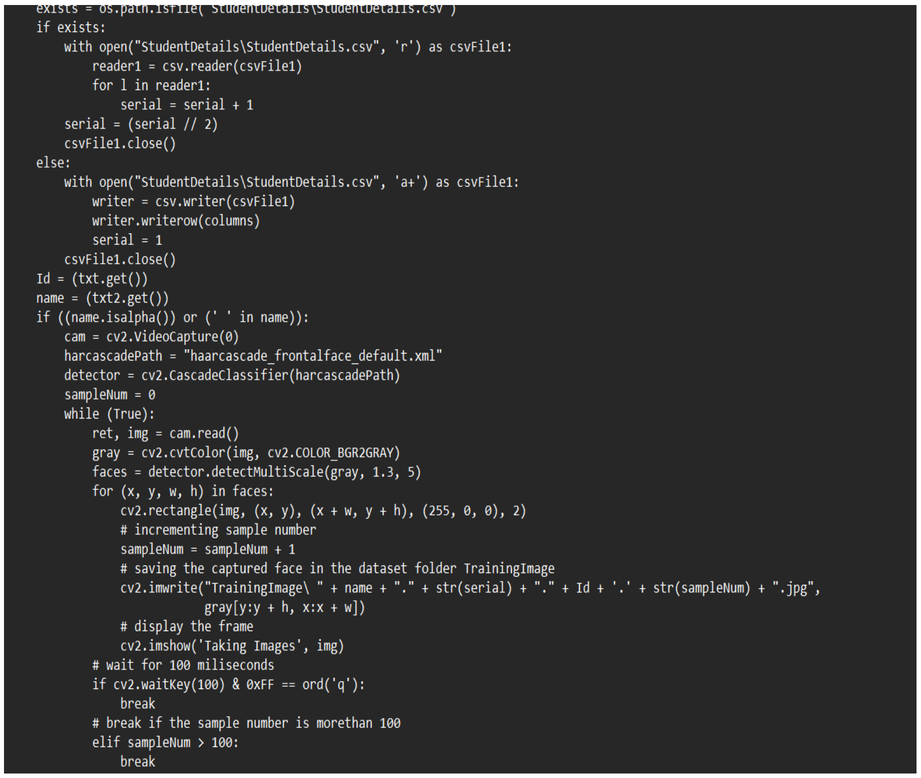
```
###################################################################################

def save_pass():
    assure_path_exists("TrainingImageLabel/")
    exists1 = os.path.isfile("TrainingImageLabel\psd.txt")
    if exists1:
        tf = open("TrainingImageLabel\psd.txt", "r")
        key = tf.read()
    else:
        master.destroy()
        new_pas = tsd.askstring('Old Password not found', 'Please enter a new password below', show='*')
        if new_pas == None:
            mess._show(title='No Password Entered', message='Password not set!! Please try again')
        else:
            tf = open("TrainingImageLabel\psd.txt", "w")
            tf.write(new_pas)
            mess._show(title='Password Registered', message='New password was registered successfully!!')
            return
    op = (old.get())
    newp= (new.get())
    nnewp = (nnew.get())
    if (op == key):
        if(newp == nnewp):
            txf = open("TrainingImageLabel\psd.txt", "w")
            txf.write(newp)
        else:
            mess._show(title='Error', message='Confirm new password again!!!')
            return
    else:
        mess._show(title='Wrong Password', message='Please enter correct old password.')
        return
    mess._show(title='Password Changed', message='Password changed successfully!!')
    master.destroy()

###################################################################################

def change_pass():
    global master
    master = tk.Tk()
```



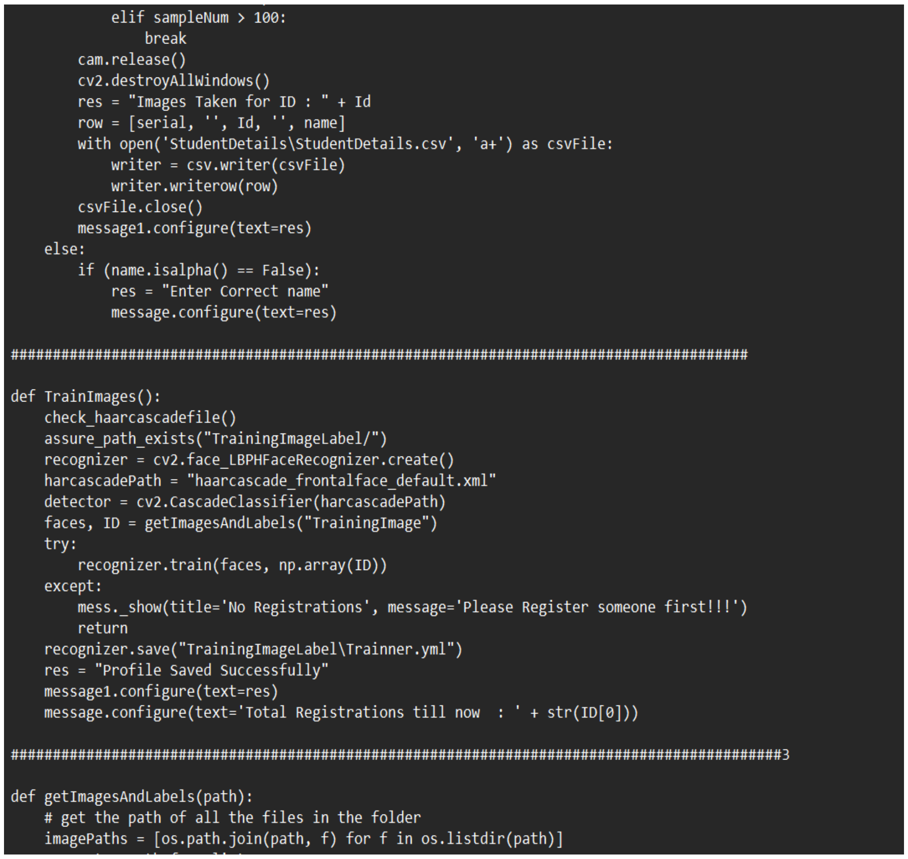
```
    global master
    master = tk.Tk()
    master.geometry("400x160")
    master.resizable(False,False)
    master.title("Change Password")
    master.configure(background="white")
    lbl4 = tk.Label(master,text='    Enter Old Password',bg='white',font=('comic', 12, ' bold '))
    lbl4.place(x=10,y=10)
    global old
    old=tk.Entry(master,width=25 ,fg="black",relief='solid',font=('comic', 12, ' bold '),show='*')
    old.place(x=180,y=10)
    lbl5 = tk.Label(master, text='   Enter New Password', bg='white', font=('comic', 12, ' bold '))
    lbl5.place(x=10, y=45)
    global new
    new = tk.Entry(master, width=25, fg="black",relief='solid', font=('comic', 12, ' bold '),show='*')
    new.place(x=180, y=45)
    lbl6 = tk.Label(master, text='Confirm New Password', bg='white', font=('comic', 12, ' bold '))
    lbl6.place(x=10, y=80)
    global nnew
    nnew = tk.Entry(master, width=25, fg="black", relief='solid',font=('comic', 12, ' bold '),show='*')
    nnew.place(x=180, y=80)
    cancel=tk.Button(master,text="Cancel", command=master.destroy ,fg="black"  ,bg="red" ,height=1,width=25 , activebackground = "white" ,font=('comic', 10, ' bold '))
    cancel.place(x=200, y=120)
    save1 = tk.Button(master, text="Save", command=save_pass, fg="black", bg="#00fcca", height = 1,width=25, activebackground="white", font=('comic', 10, ' bold '))
    save1.place(x=10, y=120)
    master.mainloop()

#####################################################################################

def psw():
    assure_path_exists("TrainingImageLabel/")
    exists1 = os.path.isfile("TrainingImageLabel\psd.txt")
    if exists1:
        tf = open("TrainingImageLabel\psd.txt", "r")
        key = tf.read()
    else:
        new_pas = tsd.askstring('Old Password not found', 'Please enter a new password below', show='*')
        if new_pas == None:
            mess._show(title='No Password Entered', message='Password not set!! Please try again')
        else:
```



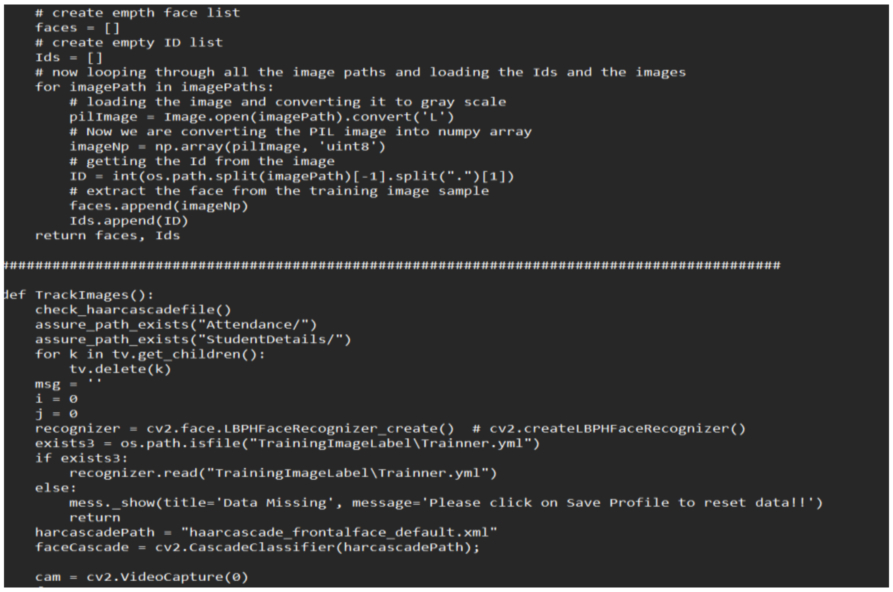
```
    else:
        new_pas = tsd.askstring('Old Password not found', 'Please enter a new password below', show='*')
        if new_pas == None:
            mess._show(title='No Password Entered', message='Password not set!! Please try again')
        else:
            tf = open("TrainingImageLabel\psd.txt", "w")
            tf.write(new_pas)
            mess._show(title='Password Registered', message='New password was registered successfully!!')
            return
    password = tsd.askstring('Password', 'Enter Password', show='*')
    if (password == key):
        TrainImages()
    elif (password == None):
        pass
    else:
        mess._show(title='Wrong Password', message='You have entered wrong password')

######################################################################################

def clear():
    txt.delete(0, 'end')
    res = "1)Take Images  >>>  2)Save Profile"
    message1.configure(text=res)


def clear2():
    txt2.delete(0, 'end')
    res = "1)Take Images  >>>  2)Save Profile"
    message1.configure(text=res)

#######################################################################################

def TakeImages():
    check_haarcascadefile()
    columns = ['SERIAL NO.', '', 'ID', '', 'NAME']
    assure_path_exists("StudentDetails/")
    assure_path_exists("TrainingImage/")
    serial = 0
    exists = os.path.isfile("StudentDetails\StudentDetails.csv")
    if exists:
```



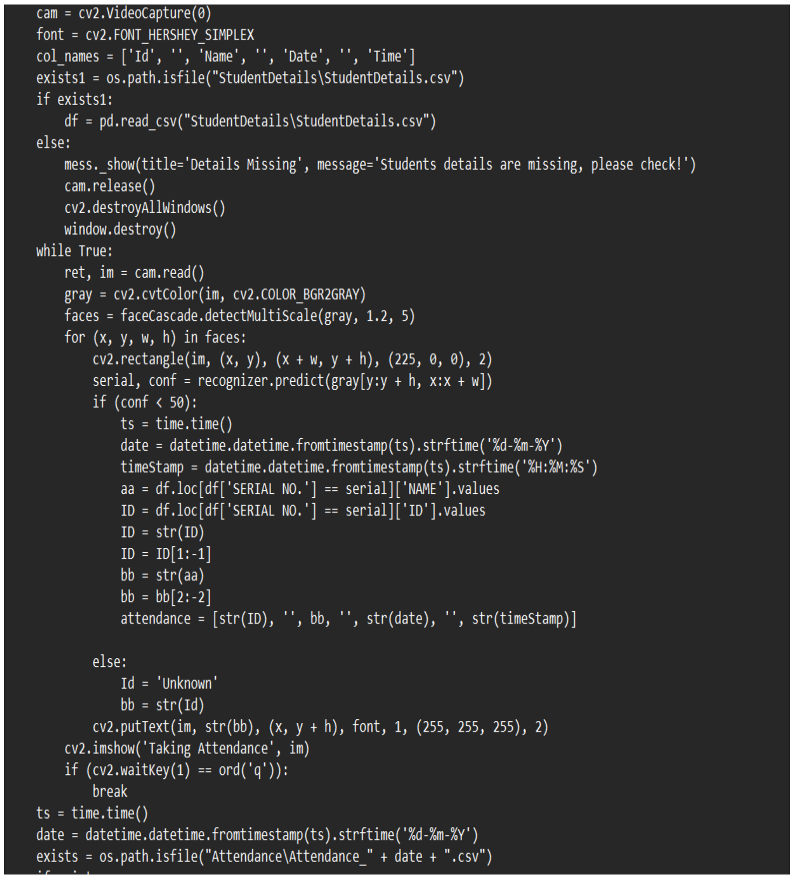
```
    exists = os.path.isfile("StudentDetails\StudentDetails.csv")
    if exists:
        with open("StudentDetails\StudentDetails.csv", 'r') as csvFile1:
            reader1 = csv.reader(csvFile1)
            for l in reader1:
                serial = serial + 1
        serial = (serial // 2)
        csvFile1.close()
    else:
        with open("StudentDetails\StudentDetails.csv", 'a+') as csvFile1:
            writer = csv.writer(csvFile1)
            writer.writerow(columns)
            serial = 1
        csvFile1.close()
    Id = (txt.get())
    name = (txt2.get())
    if ((name.isalpha()) or (' ' in name)):
        cam = cv2.VideoCapture(0)
        harcascadePath = "haarcascade_frontalface_default.xml"
        detector = cv2.CascadeClassifier(harcascadePath)
        sampleNum = 0
        while (True):
            ret, img = cam.read()
            gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
            faces = detector.detectMultiScale(gray, 1.3, 5)
            for (x, y, w, h) in faces:
                cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 2)
                # incrementing sample number
                sampleNum = sampleNum + 1
                # saving the captured face in the dataset folder TrainingImage
                cv2.imwrite("TrainingImage\ " + name + "." + str(serial) + "." + Id + '.' + str(sampleNum) + ".jpg",
                            gray[y:y + h, x:x + w])
                # display the frame
                cv2.imshow('Taking Images', img)
            # wait for 100 miliseconds
            if cv2.waitKey(100) & 0xFF == ord('q'):
                break
            # break if the sample number is morethan 100
            elif sampleNum > 100:
                break
```



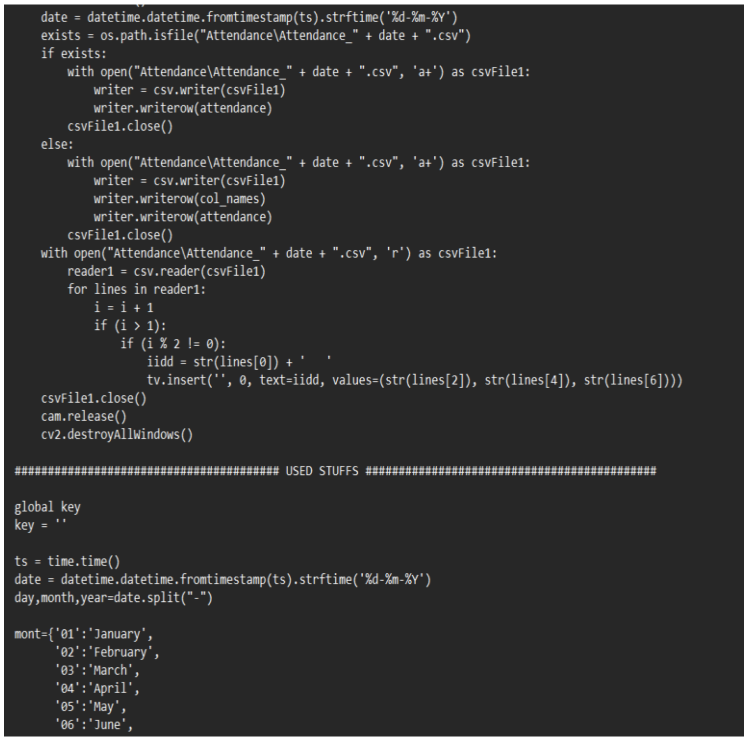
```
            elif sampleNum > 100:
                break
        cam.release()
        cv2.destroyAllWindows()
        res = "Images Taken for ID : " + Id
        row = [serial, '', Id, '', name]
        with open('StudentDetails\StudentDetails.csv', 'a+') as csvFile:
            writer = csv.writer(csvFile)
            writer.writerow(row)
        csvFile.close()
        message1.configure(text=res)
    else:
        if (name.isalpha() == False):
            res = "Enter Correct name"
            message.configure(text=res)

########################################################################################

def TrainImages():
    check_haarcascadefile()
    assure_path_exists("TrainingImageLabel/")
    recognizer = cv2.face_LBPHFaceRecognizer.create()
    harcascadePath = "haarcascade_frontalface_default.xml"
    detector = cv2.CascadeClassifier(harcascadePath)
    faces, ID = getImagesAndLabels("TrainingImage")
    try:
        recognizer.train(faces, np.array(ID))
    except:
        mess._show(title='No Registrations', message='Please Register someone first!!!')
        return
    recognizer.save("TrainingImageLabel\Trainner.yml")
    res = "Profile Saved Successfully"
    message1.configure(text=res)
    message.configure(text='Total Registrations till now  : ' + str(ID[0]))

############################################################################################3

def getImagesAndLabels(path):
    # get the path of all the files in the folder
    imagePaths = [os.path.join(path, f) for f in os.listdir(path)]
```



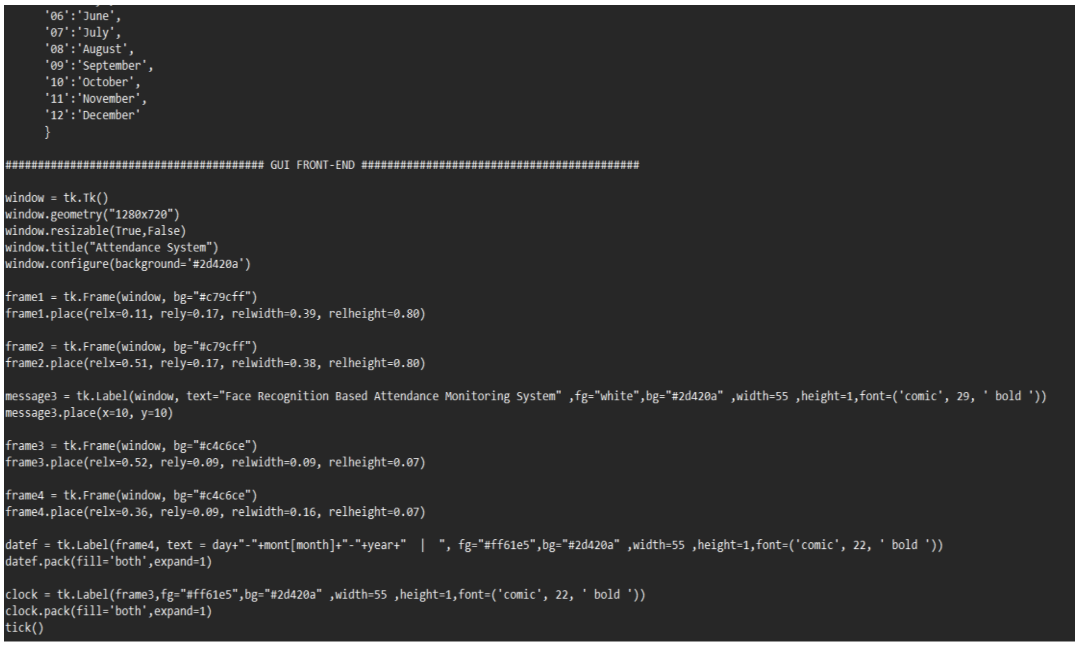
```
    # create empth face list
    faces = []
    # create empty ID list
    Ids = []
    # now looping through all the image paths and loading the Ids and the images
    for imagePath in imagePaths:
        # loading the image and converting it to gray scale
        pilImage = Image.open(imagePath).convert('L')
        # Now we are converting the PIL image into numpy array
        imageNp = np.array(pilImage, 'uint8')
        # getting the Id from the image
        ID = int(os.path.split(imagePath)[-1].split(".")[1])
        # extract the face from the training image sample
        faces.append(imageNp)
        Ids.append(ID)
    return faces, Ids

##########################################################################################

def TrackImages():
    check_haarcascadefile()
    assure_path_exists("Attendance/")
    assure_path_exists("StudentDetails/")
    for k in tv.get_children():
        tv.delete(k)
    msg = ''
    i = 0
    j = 0
    recognizer = cv2.face.LBPHFaceRecognizer_create()  # cv2.createLBPHFaceRecognizer()
    exists3 = os.path.isfile("TrainingImageLabel\Trainner.yml")
    if exists3:
        recognizer.read("TrainingImageLabel\Trainner.yml")
    else:
        mess._show(title='Data Missing', message='Please click on Save Profile to reset data!!')
        return
    harcascadePath = "haarcascade_frontalface_default.xml"
    faceCascade = cv2.CascadeClassifier(harcascadePath);

    cam = cv2.VideoCapture(0)
```



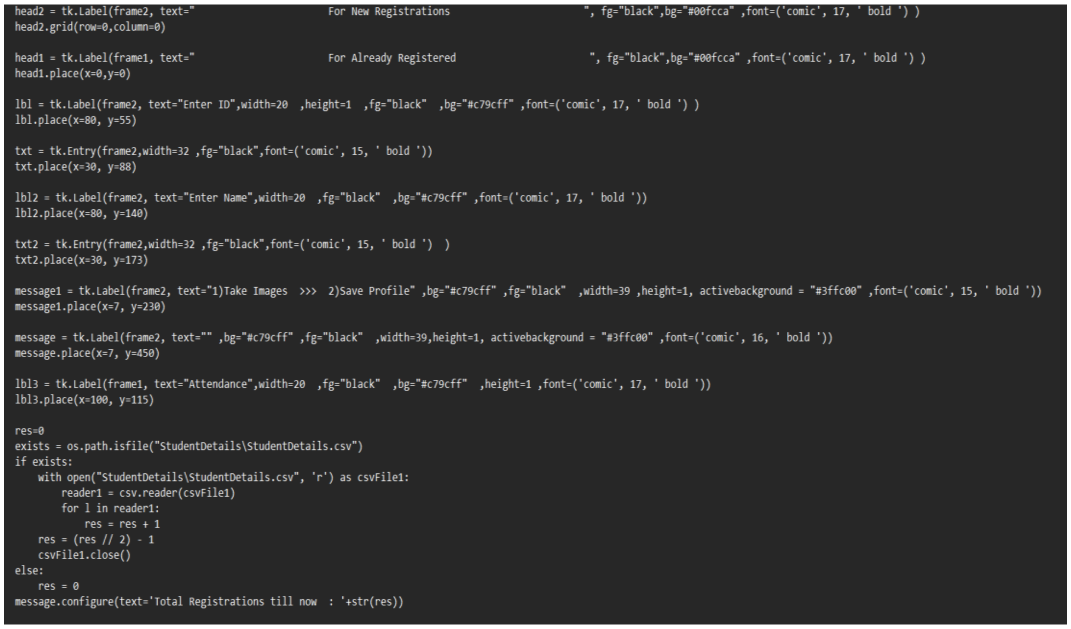
```
    cam = cv2.VideoCapture(0)
    font = cv2.FONT_HERSHEY_SIMPLEX
    col_names = ['Id', '', 'Name', '', 'Date', '', 'Time']
    exists1 = os.path.isfile("StudentDetails\StudentDetails.csv")
    if exists1:
        df = pd.read_csv("StudentDetails\StudentDetails.csv")
    else:
        mess._show(title='Details Missing', message='Students details are missing, please check!')
        cam.release()
        cv2.destroyAllWindows()
        window.destroy()
    while True:
        ret, im = cam.read()
        gray = cv2.cvtColor(im, cv2.COLOR_BGR2GRAY)
        faces = faceCascade.detectMultiScale(gray, 1.2, 5)
        for (x, y, w, h) in faces:
            cv2.rectangle(im, (x, y), (x + w, y + h), (225, 0, 0), 2)
            serial, conf = recognizer.predict(gray[y:y + h, x:x + w])
            if (conf < 50):
                ts = time.time()
                date = datetime.datetime.fromtimestamp(ts).strftime('%d-%m-%Y')
                timeStamp = datetime.datetime.fromtimestamp(ts).strftime('%H:%M:%S')
                aa = df.loc[df['SERIAL NO.'] == serial]['NAME'].values
                ID = df.loc[df['SERIAL NO.'] == serial]['ID'].values
                ID = str(ID)
                ID = ID[1:-1]
                bb = str(aa)
                bb = bb[2:-2]
                attendance = [str(ID), '', bb, '', str(date), '', str(timeStamp)]

            else:
                Id = 'Unknown'
                bb = str(Id)
            cv2.putText(im, str(bb), (x, y + h), font, 1, (255, 255, 255), 2)
        cv2.imshow('Taking Attendance', im)
        if (cv2.waitKey(1) == ord('q')):
            break
    ts = time.time()
    date = datetime.datetime.fromtimestamp(ts).strftime('%d-%m-%Y')
    exists = os.path.isfile("Attendance\Attendance_" + date + ".csv")
```



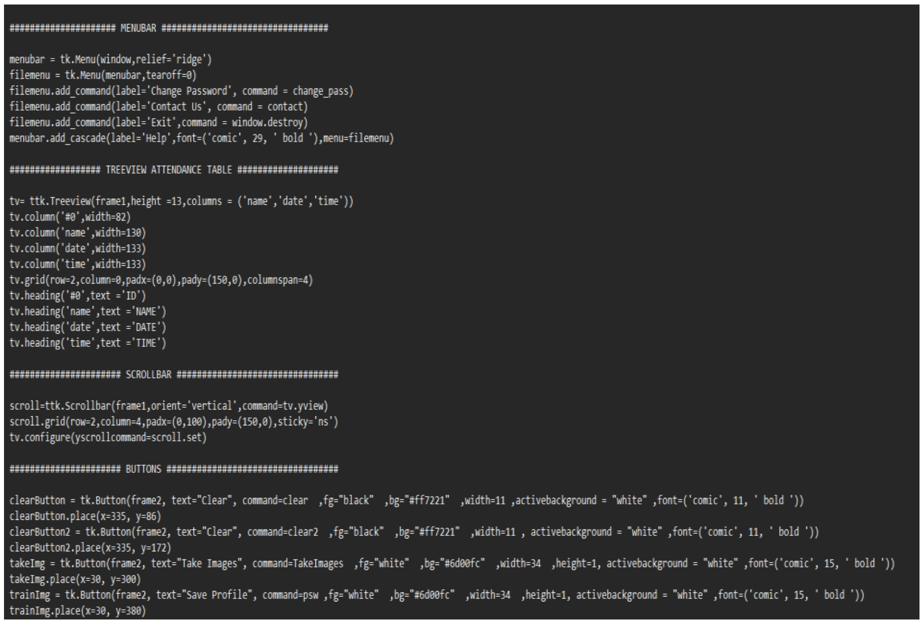
```
    date = datetime.datetime.fromtimestamp(ts).strftime('%d-%m-%Y')
    exists = os.path.isfile("Attendance\Attendance_" + date + ".csv")
    if exists:
        with open("Attendance\Attendance_" + date + ".csv", 'a+') as csvFile1:
            writer = csv.writer(csvFile1)
            writer.writerow(attendance)
        csvFile1.close()
    else:
        with open("Attendance\Attendance_" + date + ".csv", 'a+') as csvFile1:
            writer = csv.writer(csvFile1)
            writer.writerow(col_names)
            writer.writerow(attendance)
        csvFile1.close()
    with open("Attendance\Attendance_" + date + ".csv", 'r') as csvFile1:
        reader1 = csv.reader(csvFile1)
        for lines in reader1:
            i = i + 1
            if (i > 1):
                if (i % 2 != 0):
                    iidd = str(lines[0]) + '   '
                    tv.insert('', 0, text=iidd, values=(str(lines[2]), str(lines[4]), str(lines[6])))
    csvFile1.close()
    cam.release()
    cv2.destroyAllWindows()

######################################## USED STUFFS ############################################

global key
key = ''

ts = time.time()
date = datetime.datetime.fromtimestamp(ts).strftime('%d-%m-%Y')
day,month,year=date.split("-")

mont={'01':'January',
      '02':'February',
      '03':'March',
      '04':'April',
      '05':'May',
      '06':'June',
```



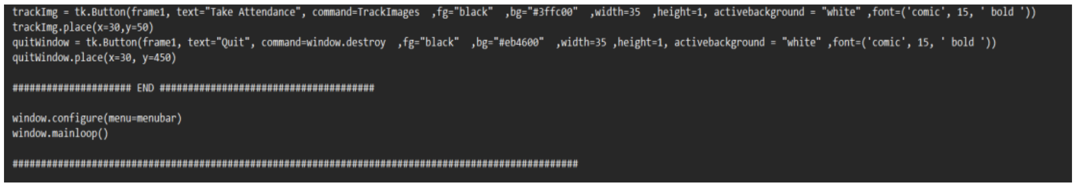
```
      '06':'June',
      '07':'July',
      '08':'August',
      '09':'September',
      '10':'October',
      '11':'November',
      '12':'December'
      }

######################################## GUI FRONT-END ###########################################

window = tk.Tk()
window.geometry("1280x720")
window.resizable(True,False)
window.title("Attendance System")
window.configure(background='#2d420a')

frame1 = tk.Frame(window, bg="#c79cff")
frame1.place(relx=0.11, rely=0.17, relwidth=0.39, relheight=0.80)

frame2 = tk.Frame(window, bg="#c79cff")
frame2.place(relx=0.51, rely=0.17, relwidth=0.38, relheight=0.80)

message3 = tk.Label(window, text="Face Recognition Based Attendance Monitoring System" ,fg="white",bg="#2d420a" ,width=55 ,height=1,font=('comic', 29, ' bold '))
message3.place(x=10, y=10)

frame3 = tk.Frame(window, bg="#c4c6ce")
frame3.place(relx=0.52, rely=0.09, relwidth=0.09, relheight=0.07)

frame4 = tk.Frame(window, bg="#c4c6ce")
frame4.place(relx=0.36, rely=0.09, relwidth=0.16, relheight=0.07)

datef = tk.Label(frame4, text = day+"-"+mont[month]+"-"+year+"  |  ", fg="#ff61e5",bg="#2d420a" ,width=55 ,height=1,font=('comic', 22, ' bold '))
datef.pack(fill='both',expand=1)

clock = tk.Label(frame3,fg="#ff61e5",bg="#2d420a" ,width=55 ,height=1,font=('comic', 22, ' bold '))
clock.pack(fill='both',expand=1)
tick()
```



```
head2 = tk.Label(frame2, text="                       For New Registrations                       ", fg="black",bg="#00fcca" ,font=('comic', 17, ' bold ') )
head2.grid(row=0,column=0)

head1 = tk.Label(frame1, text="                       For Already Registered                       ", fg="black",bg="#00fcca" ,font=('comic', 17, ' bold ') )
head1.place(x=0,y=0)

lbl = tk.Label(frame2, text="Enter ID",width=20  ,height=1  ,fg="black"  ,bg="#c79cff" ,font=('comic', 17, ' bold ') )
lbl.place(x=80, y=55)

txt = tk.Entry(frame2,width=32 ,fg="black",font=('comic', 15, ' bold '))
txt.place(x=30, y=88)

lbl2 = tk.Label(frame2, text="Enter Name",width=20  ,fg="black"  ,bg="#c79cff" ,font=('comic', 17, ' bold '))
lbl2.place(x=80, y=140)

txt2 = tk.Entry(frame2,width=32 ,fg="black",font=('comic', 15, ' bold ')  )
txt2.place(x=30, y=173)

message1 = tk.Label(frame2, text="1)Take Images  >>>  2)Save Profile" ,bg="#c79cff" ,fg="black"  ,width=39 ,height=1, activebackground = "#3ffc00" ,font=('comic', 15, ' bold '))
message1.place(x=7, y=230)

message = tk.Label(frame2, text="" ,bg="#c79cff" ,fg="black"  ,width=39,height=1, activebackground = "#3ffc00" ,font=('comic', 16, ' bold '))
message.place(x=7, y=450)

lbl3 = tk.Label(frame1, text="Attendance",width=20  ,fg="black"  ,bg="#c79cff"  ,height=1 ,font=('comic', 17, ' bold '))
lbl3.place(x=100, y=115)

res=0
exists = os.path.isfile("StudentDetails\StudentDetails.csv")
if exists:
    with open("StudentDetails\StudentDetails.csv", 'r') as csvFile1:
        reader1 = csv.reader(csvFile1)
        for l in reader1:
            res = res + 1
    res = (res // 2) - 1
    csvFile1.close()
else:
    res = 0
message.configure(text='Total Registrations till now  : '+str(res))
```



```
##################### MENUBAR ################################

menubar = tk.Menu(window,relief='ridge')
filemenu = tk.Menu(menubar,tearoff=0)
filemenu.add_command(label='Change Password', command = change_pass)
filemenu.add_command(label='Contact Us', command = contact)
filemenu.add_command(label='Exit',command = window.destroy)
menubar.add_cascade(label='Help',font=('comic', 29, ' bold '),menu=filemenu)

################## TREEVIEW ATTENDANCE TABLE ####################

tv= ttk.Treeview(frame1,height =13,columns = ('name','date','time'))
tv.column('#0',width=82)
tv.column('name',width=130)
tv.column('date',width=133)
tv.column('time',width=133)
tv.grid(row=2,column=0,padx=(0,0),pady=(150,0),columnspan=4)
tv.heading('#0',text ='ID')
tv.heading('name',text ='NAME')
tv.heading('date',text ='DATE')
tv.heading('time',text ='TIME')

###################### SCROLLBAR ################################

scroll=ttk.Scrollbar(frame1,orient='vertical',command=tv.yview)
scroll.grid(row=2,column=4,padx=(0,100),pady=(150,0),sticky='ns')
tv.configure(yscrollcommand=scroll.set)

###################### BUTTONS ##################################

clearButton = tk.Button(frame2, text="Clear", command=clear  ,fg="black"  ,bg="#ff7221"  ,width=11 ,activebackground = "white" ,font=('comic', 11, ' bold '))
clearButton.place(x=335, y=86)
clearButton2 = tk.Button(frame2, text="Clear", command=clear2  ,fg="black"  ,bg="#ff7221"  ,width=11 , activebackground = "white" ,font=('comic', 11, ' bold '))
clearButton2.place(x=335, y=172)
takeImg = tk.Button(frame2, text="Take Images", command=TakeImages  ,fg="white"  ,bg="#6d00fc"  ,width=34  ,height=1, activebackground = "white" ,font=('comic', 15, ' bold '))
takeImg.place(x=30, y=300)
trainImg = tk.Button(frame2, text="Save Profile", command=psw ,fg="white"  ,bg="#6d00fc"  ,width=34  ,height=1, activebackground = "white" ,font=('comic', 15, ' bold '))
trainImg.place(x=30, y=380)
```



```
trackImg = tk.Button(frame1, text="Take Attendance", command=TrackImages  ,fg="black"  ,bg="#3ffc00"  ,width=35  ,height=1, activebackground = "white" ,font=('comic', 15, ' bold '))
trackImg.place(x=30,y=50)
quitWindow = tk.Button(frame1, text="Quit", command=window.destroy  ,fg="black"  ,bg="#eb4600"  ,width=35 ,height=1, activebackground = "white" ,font=('comic', 15, ' bold '))
quitWindow.place(x=30, y=450)

#################### END ######################################

window.configure(menu=menubar)
window.mainloop()

####################################################################################################
```

Figure 4.2.1 Face Recognition Code

CHAPTER – 5
Experimentation and Result Analysis

Experimentation and Result Analysis

A number of crucial processes are included in the Face Recognition Attendance System experimentation process, which aims to assess the code's performance and functionality. The system must first be tested to see if it can take good pictures in a range of environmental settings. This involves evaluating the face identification algorithm's accuracy using OpenCV's Haar cascade classifier, especially in various lighting scenarios and facial orientations. In order to analyze how well the system detects and reliably captures the photos of many individuals for later recognition, an experiment can be set up where the people exhibit themselves in various contexts.

Data management and user engagement are the third area of experimentation. This involves assessing the system's ability to process user input, such as registration information and attendance records, and whether the data is correctly saved to CSV files without corruption or loss. Surveys can be used to gather user input after an interaction in order to evaluate usability and effectiveness.

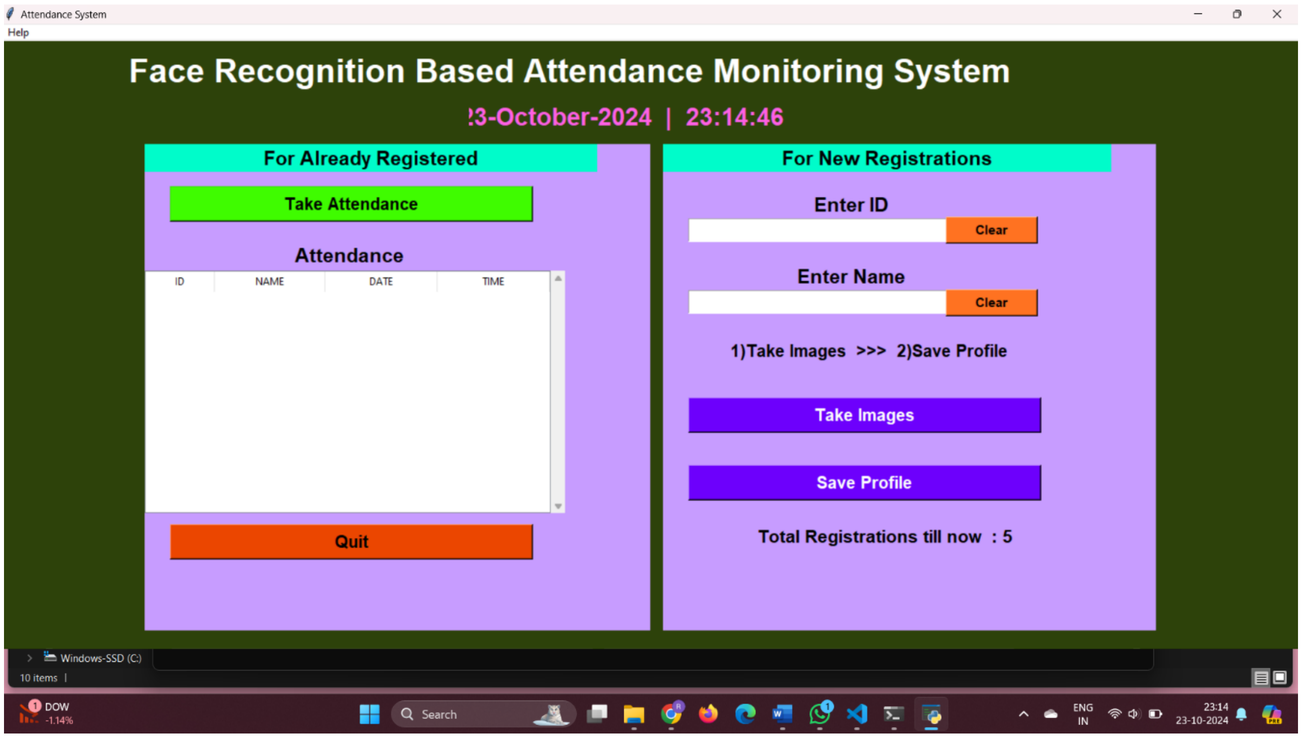
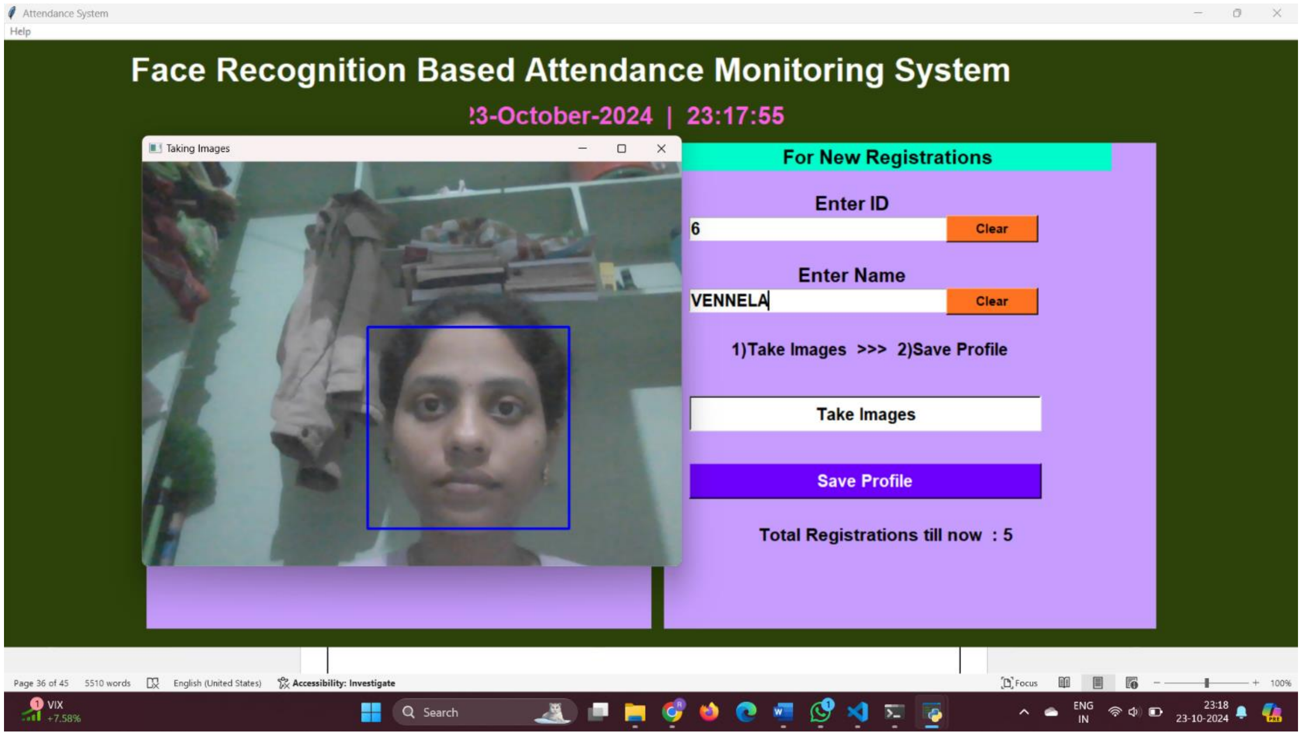


Figure 5.1 Face Recognition Based Attendance Monitoring System



Face Recognition Based Attendance Monitoring System

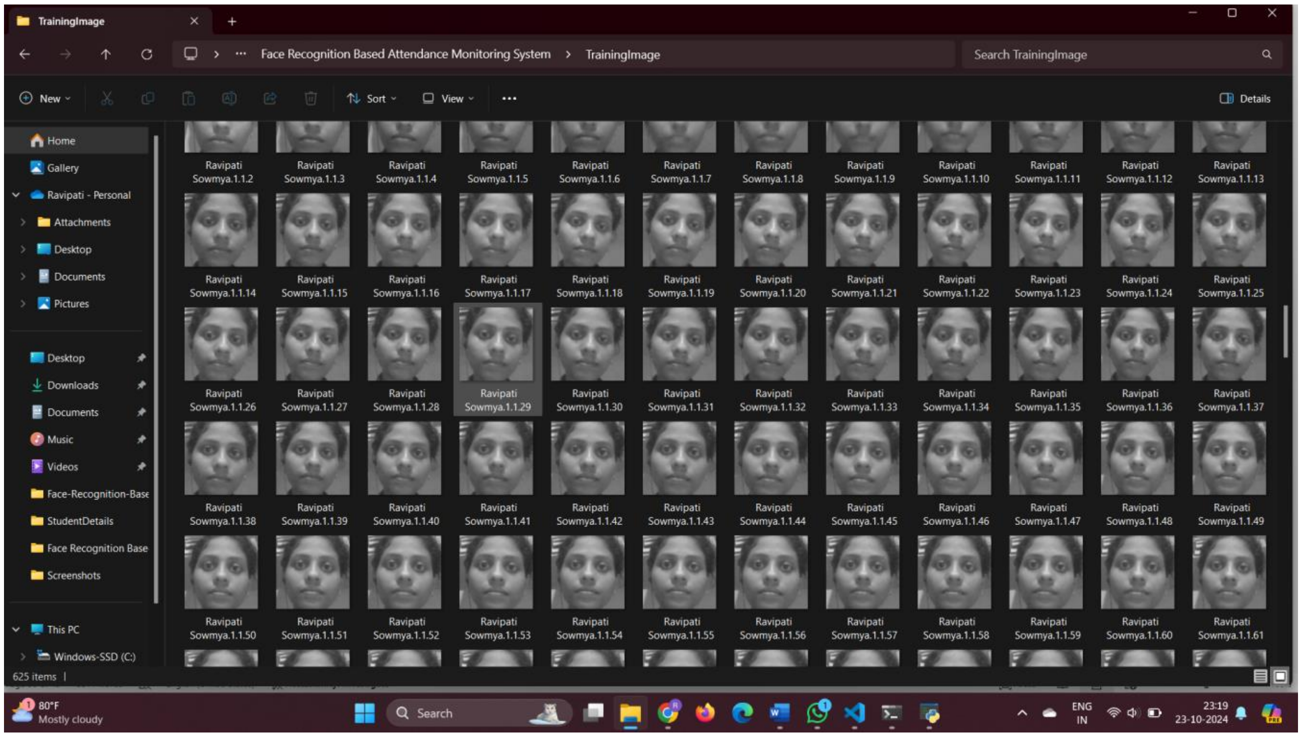
!3-October-2024 | 23:17:55

Enter ID: 6

Enter Name: VENNELA

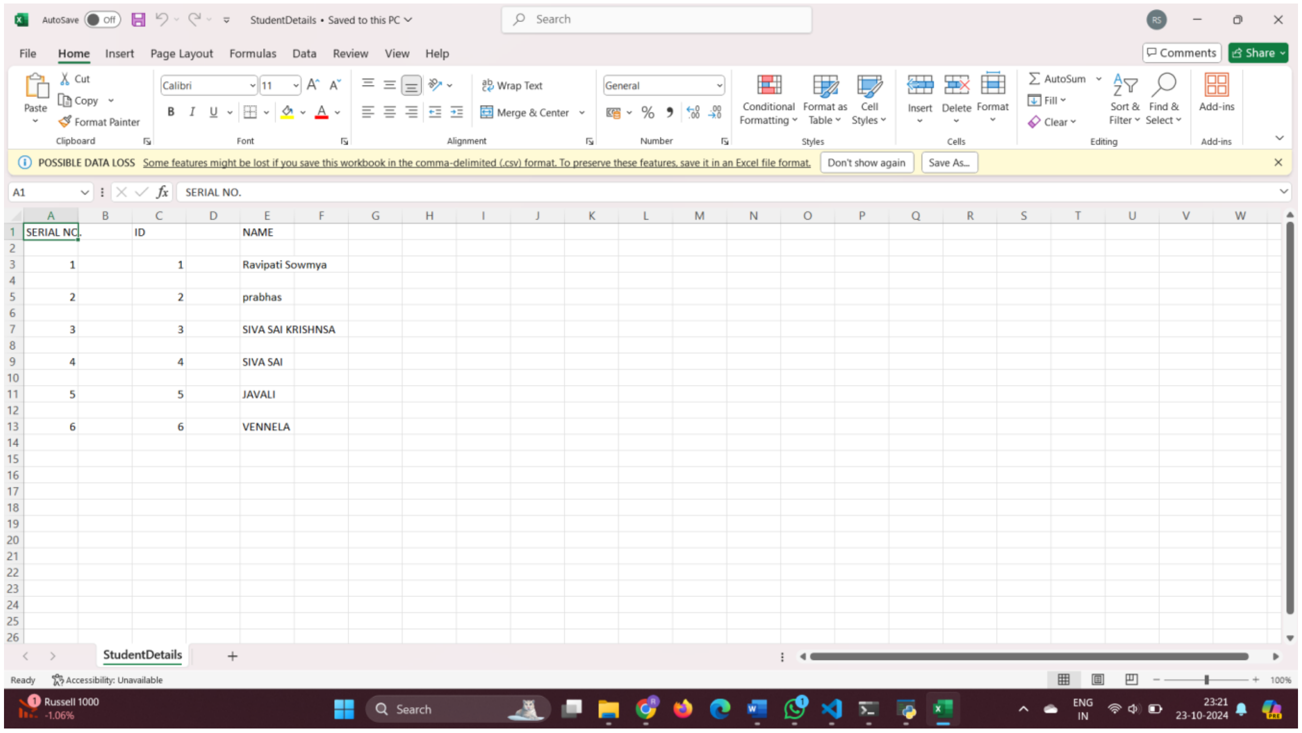
Total Registrations till now : 5

Figure 5.2 Taking image



Ravipati Sowmya.1.1.2, Ravipati Sowmya.1.1.3, Ravipati Sowmya.1.1.4, Ravipati Sowmya.1.1.5, Ravipati Sowmya.1.1.6, Ravipati Sowmya.1.1.7, Ravipati Sowmya.1.1.8, Ravipati Sowmya.1.1.9, Ravipati Sowmya.1.1.10, Ravipati Sowmya.1.1.11, Ravipati Sowmya.1.1.12, Ravipati Sowmya.1.1.13
Ravipati Sowmya.1.1.14, Ravipati Sowmya.1.1.15, Ravipati Sowmya.1.1.16, Ravipati Sowmya.1.1.17, Ravipati Sowmya.1.1.18, Ravipati Sowmya.1.1.19, Ravipati Sowmya.1.1.20, Ravipati Sowmya.1.1.21, Ravipati Sowmya.1.1.22, Ravipati Sowmya.1.1.23, Ravipati Sowmya.1.1.24, Ravipati Sowmya.1.1.25
Ravipati Sowmya.1.1.26, Ravipati Sowmya.1.1.27, Ravipati Sowmya.1.1.28, Ravipati Sowmya.1.1.29, Ravipati Sowmya.1.1.30, Ravipati Sowmya.1.1.31, Ravipati Sowmya.1.1.32, Ravipati Sowmya.1.1.33, Ravipati Sowmya.1.1.34, Ravipati Sowmya.1.1.35, Ravipati Sowmya.1.1.36, Ravipati Sowmya.1.1.37
Ravipati Sowmya.1.1.38, Ravipati Sowmya.1.1.39, Ravipati Sowmya.1.1.40, Ravipati Sowmya.1.1.41, Ravipati Sowmya.1.1.42, Ravipati Sowmya.1.1.43, Ravipati Sowmya.1.1.44, Ravipati Sowmya.1.1.45, Ravipati Sowmya.1.1.46, Ravipati Sowmya.1.1.47, Ravipati Sowmya.1.1.48, Ravipati Sowmya.1.1.49
Ravipati Sowmya.1.1.50, Ravipati Sowmya.1.1.51, Ravipati Sowmya.1.1.52, Ravipati Sowmya.1.1.53, Ravipati Sowmya.1.1.54, Ravipati Sowmya.1.1.55, Ravipati Sowmya.1.1.56, Ravipati Sowmya.1.1.57, Ravipati Sowmya.1.1.58, Ravipati Sowmya.1.1.59, Ravipati Sowmya.1.1.60, Ravipati Sowmya.1.1.61

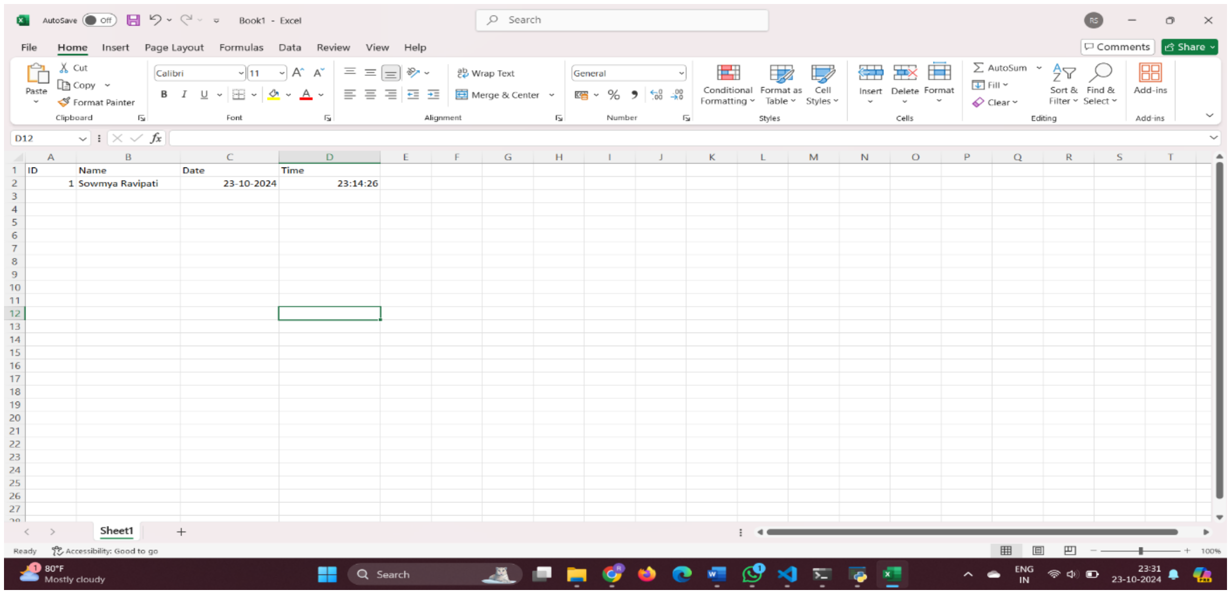
Figure 5.3 Image training


POSSIBLE DATA LOSS Some features might be lost if you save this workbook in the comma-delimited (.csv) format. To preserve these features, save it in an Excel file format.

SERIAL NO.	ID	NAME
1	1	Ravipati Sowmya
2	2	prabhas
3	3	SIVA SAI KRISHNSA
4	4	SIVA SAI
5	5	JAVALI
6	6	VENNELA

Figure 5.4 Student Details

The student details get automatically saved in the PC after training of the images in the excel sheet.



ID	Name	Date	Time
1	Sowmya Ravipati	23-10-2024	23:14:26

Figure 5.5 Attendance sheet

The attendance is gets marked after the whole process of Face Recognition System and stores in an excel sheet.

CHAPTER – 6
CONCLUSION AND FUTURE ENHANCEMENT

6.1 Conclusion

The Face Recognition Attendance System, which was created in Python, is a prime example of how cutting-edge computer vision technology can be seamlessly integrated with user-friendly interfaces to simplify attendance management. **The Haar Cascade and LBPH (Local Binary Patterns Histograms) algorithms** are essential components of this system.

A machine learning object detection technique called the Haar Cascade classifier is used to discover and identify faces in live video streams, guaranteeing that every participant is precisely identified prior to their attendance being noted. The **LBPH technique**, which effectively classes and detects faces based on local features, considerably improves this fundamental identification capability and enables efficient recognition model training with minimum storage needs.

These two strong algorithms work together to guarantee excellent **facial detection** and identification accuracy, reducing the errors that come with human attendance systems. The user-friendly **GUI**, which is based on the **Tkinter framework**, makes it easy for educators and students to navigate and do tasks like managing profiles and taking pictures.

Furthermore, data integrity is improved by security measures like password protection, which makes the system especially suitable for use in educational settings where privacy and accuracy are essential. By utilizing these technologies, the system enhances the overall educational experience through its useful, effective design in addition to automating the administration of attendance.

6.2 Future Enhancement

Future studies in the field of facial recognition-based attendance systems will be able to take advantage of state-of-the-art tools and techniques to improve system functionality, user experience, and moral considerations. The following cutting-edge methods have promise for further research:

In order to improve accuracy and usefulness, future research in facial recognition-based attendance systems will concentrate on a few crucial developments.To ensure accurate identification, improved facial recognition algorithms that make use of machine learning techniques like transformer-based models and **convolutional neural networks (CNNs)** are essential for overcoming obstacles like occlusions and changing lighting conditions. Furthermore, by combining contextual data and other biometric inputs, richer user profiles may be produced, greatly increasing attendance accuracy and reducing threats like identity theft. Through the use of online learning algorithms, the system will be able to adapt its recognition model on the fly in response to human interactions, maintaining its efficacy in dynamic contexts.

Moreover, it will be crucial to create frameworks that put user privacy, permission, and data security first, with explainable **AI (XAI)** promoting openness and confidence in data use. Attendance systems can provide a more customized and adaptable user experience by taking into account contextual elements like time and place, which can streamline procedures and increase overall dependability. By continuously adjusting to user interactions and presenting attendance marking as a decision-making problem, reinforcement learning will maximize attendance while improving efficiency and personalization. Lastly, research should focus on developing scalable attendance systems that can handle big groups and work well with current management software. API development will help to improve overall utility across a range of applications and facilitate interaction.

CHAPTER-7
REFERENCES

References

[1] Kennedy Okokpujie, Etinosa Noma-Osaghae,et al. The research journal “Face Recognition Based Attendance System with GSM notification”

[2] Xiaoxiang Xu, Li Zhang, Fanzhang Li.The research journal “multi-scale feature extraction for single face recognition”,2018.

[3] Changjiang Jiang,Mingyi Wang,et al. The research journal “a face recognition algorithm based on sparse representation and feature fusion”,2019.

[4] SamridhiDev, Tushar Patnaik “the research journal “Student Attendance System using Face Recognition”,2020.

[5] Nico Surantha , Boy Sugijakko “Lightweight face recognition-based portable attendance system with liveness detection”, 2024.

[6] Bao-Thien Nguyen-Tat, Minh-Quoc Bui, Vuong M. Ngo “Automating human resources' Attendance”,2024.

[7] Joshita Malla,et al. “Improved Facial Recognition Algorithms: Advancements in machine learning, such as convolutional neural networks (CNNs) and transformer-based model”,2024.

[8] Olufemi S. Ojo,et al. “Real-time Face-based Gender Identification System Using Pelican Support Vector Machine”,2024.

[9] Gabriela Moise, Elena S. Nicoară, “Ethical aspects of automatic emotion recognition in online learning”,2024.

[10] Abraham Woubie,et al. “Maintaining Privacy in Face Recognition Using Federated Learning Method”,2024.

[11] N. Bergman,et al. “Biometric identification of dairy cows via real-time facial recognition”,2024.

[12] Kritagya Painuly,et al. “Efficient Real-Time Face recognition-based attendance systems with deep learning”,2024.

[13] Muhammad Ahmad Nawaz Ul Ghani 1,et al. “Enhancing Security and Privacy in Distributed Face Recognition Systems through Blockchain and GAN”,2024.