**FACE RECOGNITION ATTENDANCE SYSTEM**

The project report submitted to the Bharathiar University in partial fulfillment of the requirements for the Award of the Degree of

# BACHELOR OF SCIENCE IN COMPUTER SCIENCE

Submitted by

## PRAVEEN KUMAR J (REG NO: 2122K2226)



Under the Guidance of

# Dr.CHITRA K M.Sc., M.Phil., PhD.,

# DEPARTMENT OF COMPUTER SCIENCE



MARCH 2024

# CERTIFICATE

This is to certify that the project work entitled **“FACE RECOGNITION ATTENDANCE SYSTEM”** is a bona-fide work done by **PRAVEEN KUMAR J**(**2122K2226**) during the period of 2023 – 2024 of his study in the **Department of Computer Science** at **Sri Krishna Adithya College of Arts and Science**, Coimbatore, Affiliated to Bharathiar University, in partial fulfillment of requirements for the award of the degree of **Bachelor of Science in Computer Science** under my supervision and guidance**.**

The project work is an original work of the candidate and to the best of my knowledge has not been submitted, in part or in full, for any Diploma / Degree / Associate ship / Fellow ship or other similar titles in this or any other University. No portion of the dissertation is a re- production from any other source, published or unpublished, without acknowledgment.

## SIGNATURE OF THE GUIDE

**SIGNATURE OF THE HOD SIGNATURE OF THE PRINCIPAL**

**Viva –Voce conducted on:**

**Internal Examiner External Examiner**

# DECLARATION

I hereby declare that the project work entitled **“FACE RECOGNITION ATTENDANCE SYSTEM”** submitted to **Sri Krishna Adithya College of Arts and Science, Coimbatore** Affiliated to Bharathiar University in partial fulfillment of the requirements for the award of the Degree of **Bachelor of Science in Computer Science** is a record of original work done by me during 2023 - 2024 under the supervision and guidance of **Dr.CHITRA K M.Sc., M.Phil.., PhD..,** The report has not been submitted for the award of any Degree / Diploma / Associate ship / Fellowship.

**Date: SIGNATURE OF THE CANDIDATE**

## ACKNOWLEDGEMENT

First and foremost, I thank the almighty through whom all things are possible. My work would not be possible without the gifts god gave me.

The success of the project depends upon the efforts invested. It is my duty to acknowledge and thank the individuals who has contributed towards the successful completion of my dissertation.

I would like to express my wholehearted thanks to **Smt.S.Malarvizhi**, Chairperson & Managing Trustee, Sri Krishna Institutions, Coimbatore.

I take this opportunity to express my profound and whole hearted thanks to our Principal **Dr. S. Palaniammal. M.Sc., M.Phil., Ph.D.,** Sri Krishna Adithya College of Arts and Science for reinforcing my effort with support and guidance.

I owe my heart and profound gratitude to my HOD **Dr.Chitra K M.Sc.,M.Phil.,PhD.,** and my internal guide **Dr.Chitra K M.Sc.,M.Phil.,PhD.,** Sri Krishna Adithya College of Arts and Science, Coimbatore for giving continuous encouragement to complete this work. This project would not be a success for me without their valuable comments and suggestions.

I am grateful to thank all the faculty members in our department for their motivation and endless inspiration to help in completing this project work.

The successful completion of my project would not have been possible without my parent's support. I take this opportunity to thank them for their continuous encouragement. I convey my thankfulness to all my friends who were with me to share my happiness and agony

|  |  |  |
| --- | --- | --- |
| **S.NO** | **CONTENTS** | **PAGE NO** |
|  | **CERTIFICATION** |  |
|  | **DECLARATION** |  |
|  | **ACKNOWLEDGEMENT** |  |
|  | **SYNOPSIS** |  |
| 1 | **INTRODUCTION** | 1 |
| 1.1 | Importance of Face recognition attendance system | 1 |
| 1.2 | About the Project | 2 |
| 1.3 | Existing System | 3 |
| 1.3.1 | Drawbacks of Existing System | 4 |
| 1.4 | Proposed System | 5 |
| 1.4.1 | Advantages of Proposed System | 6 |
| **2** | **SYSTEM SPECIFICATION** | 8 |
| 2.1 | Hardware Specification | 8 |
| 2.2 | Software Specification | 8 |
| 2.3 | Frontend | 8 |
| 2.4 | Backend | 10 |
| **3** | **SYSTEM DESIGN AND DEVELOPMENT** | 12 |
| 3.1 | System Architecture Design | 12 |
| 3.2 | Module Description | 14 |
| **4** | **TESTING AND IMPLEMENTATION** | 17 |
| 4.1 | Testing | 17 |
| 4.2 | System Testing | 17 |
| 4.3 | Testing Methodologies | 17 |
| **5**  5.1 | **CONCLUSION & FUTURE ENHANCEMENTS**  Conclusion | 19  19 |
| 5.2 | Future Enhancements | 20 |
|  | **BIBLIOGRAPHY** | 22 |
|  | Website References | 23 |
|  | **APPENDIX** | 24 |
| A | Data Flow Diagram | 24 |
| B | Sample Screenshots | 26 |
| C | Sample Code | 31 |

# INTRODUCTION

Face recognition attendance systems utilize sophisticated biometric technology to accurately identify individuals based on their facial features. These systems employ algorithms to analyze unique facial characteristics such as the distance between the eyes, the shape of the nose, and the contours of the face. Through a process known as facial recognition, these systems create a digital representation of a person’s face, often referred to as a faceprint or template.The process begins with enrollment, during which individuals’ faces are scanned and stored in a database along with their corresponding identities. This enrollment phase is crucial as it establishes a baseline for comparison during subsequent attendance tracking.When an individual wishes to record their attendance, they simply stand in front of a camera or a scanning device equipped with facial recognition technology. The system captures an image of their face, extracts the facial features, and compares them against the enrolled faceprints in the database. If a match is found within a predefined threshold of similarity, the system confirms the individual’s identity and records their attendance. One of the key advantages of face recognition attendance systems is their speed and efficiency. Compared to traditional methods such as manual check-ins or swipe cards, which can be time-consuming and prone to errors, face recognition offers a seamless and hassle-free experience. Individuals can simply walk past a camera or a scanning device, and their attendance is automatically recorded within seconds.Furthermore, face recognition systems enhance security measures by ensuring that only registered individuals can gain access and record their attendance. This helps prevent cases of proxy attendance or unauthorized access, thereby promoting accountability and integrity within the attendance tracking process. Additionally, face recognition attendance systems are adaptable to various environments and can be integrated with existing infrastructure such as access control systems or timekeeping software. This allows organizations to seamlessly incorporate facial recognition technology into their existing attendance management systems, enhancing overall efficiency and accuracy.Overall, face recognition attendance systems represent a transformative advancement in attendance management, offering unparalleled convenience, accuracy, and security. As technology continues to evolve, these systems are poised to become increasingly integral to the operations of educational institutions, corporate offices, and other organizations seeking to streamline their attendance tracking processes.

## IMPORTANCE OF FACE RECOGNITION ATTENDENCE SYSTEM

1. Accuracy: Utilizes unique biometric data for precise attendance tracking, minimizing errors like buddy punching.

2. Efficiency: Streamlines the attendance recording process, saving time for both employees and administrators.

3. Security: Enhances security by accurately verifying individuals’ identities, preventing unauthorized access.

4. Real-time Monitoring: Provides up-to-date attendance data for immediate response to irregularities.

5. Data Accuracy and Compliance: Ensures accurate records for payroll, compliance, and analysis, reducing legal risks.

6. Cost-effectiveness: Long-term savings by eliminating physical cards and automating administrative tasks.

7. Contactless Operation: Promotes hygiene and safety by eliminating physical contact with devices.

8. Scalability and Adaptability: Customizable to meet the needs of organizations of all sizes and industries.

## ABOUT THE PROJECT

A face recognition attendance system project involves the development of a software application aimed at automating and enhancing the process of attendance tracking using facial recognition technology. Beginning with requirements gathering, the project entails defining the system’s features, user roles, and technical specifications, with careful consideration given to factors like scalability and security. The system architecture is then designed, encompassing the front-end interface, back-end components, and database structure. Front-end development focuses on creating an intuitive user interface using technologies like Tkinter for desktop applications or web frameworks for online platforms. Meanwhile, the back-end logic is implemented using programming languages such as Python, integrating face recognition algorithms to accurately detect and match faces in real-time. Database management involves setting up a database to store user information, attendance records, and configurations. Thorough testing is conducted to ensure the system’s functionality, performance, and reliability before deployment in the target environment. Training and documentation are provided for system administrators and end-users, with ongoing maintenance and support offered to ensure smooth operation and address any issues that may arise. Ultimately, the face recognition attendance system project aims to streamline attendance tracking processes, improve accuracy.

## EXISTING SYSTEM

In the existing e-commerce landscape, price comparison and tracking systems are widely utilized by consumers

1. ZKTeco: ZKTeco offers a range of biometric solutions, including face recognition attendance systems designed for workplaces, schools, and other environments. Their systems feature high accuracy and real-time monitoring capabilities.
2. Suprema: Suprema’s face recognition attendance systems provide fast and accurate identification, with options for integration with access control systems. These systems are widely used in commercial and industrial settings.
3. Hikvision: Hikvision offers facial recognition terminals specifically designed for attendance management. These systems feature robust hardware and software solutions, along with advanced facial recognition algorithms for enhanced accuracy.
4. BioStar: BioStar’s face recognition attendance systems are known for their reliability and scalability. These systems offer a user-friendly interface and seamless integration with existing HR and time management software.

## DRAWBACKS OF EXISTING SYSTEM

1. Cost: Implementing a face recognition attendance system can be expensive, involving costs for hardware, software, installation, and maintenance. This initial investment may be prohibitive for some organizations, especially smaller businesses with limited budgets.
2. Privacy Concerns: Facial recognition technology raises privacy concerns, as it involves capturing and storing biometric data. Employees may feel uncomfortable with the idea of their facial data being collected and stored by their employer, leading to potential resistance or legal challenges.
3. Accuracy Issues: While face recognition technology has improved significantly in recent years, it is not infallible. Factors such as variations in lighting, facial expressions, and occlusions (e.g., wearing glasses or hats) can affect accuracy.
4. Bias and Discrimination: Facial recognition algorithms may exhibit biases, leading to inaccurate results for certain demographic groups. Studies have shown that some algorithms perform less accurately for individuals with darker skin tones or non-binary gender presentations, raising concerns about fairness and potential discrimination.
5. Security Risks: Like any technology system, face recognition attendance systems are vulnerable to security breaches. Hackers could potentially exploit vulnerabilities in the system to gain unauthorized access to sensitive data or manipulate attendance records.
6. Technical Limitations: Face recognition attendance systems may face technical limitations in certain environments. For example, outdoor installations may be affected by weather conditions such as rain or glare, while crowded spaces may pose challenges for accurate facial recognition.
7. Legal and Regulatory Compliance: Organizations using face recognition technology must navigate a complex landscape of laws and regulations governing the collection and use of biometric data. Failure to comply with these regulations, such as the European Union's General Data Protection Regulation (GDPR) or the California Consumer Privacy Act (CCPA), could result in legal consequences and reputational damage.

## PROPOSED SYSTEM

The proposed face recognition attendance system automates attendance tracking through facial recognition. It includes front-end and back-end components for user interaction, facial recognition algorithms, database management, and security measures. Real-time monitoring, integration capabilities, and scalability are key aspects. The system prioritizes security and privacy, with thorough testing and training for effective deployment.

## ADVANTAGES OF PROPOSED SYSTEM

A proposed face recognition attendance system using Tkinter and Python would involve the following components and steps:

1. User Interface with Tkinter:

- Develop a graphical user interface (GUI) using Tkinter to interact with users.

- Design an intuitive interface for capturing images, displaying attendance status, and configuring system settings.

2. Face Detection and Recognition:

- Utilize Python libraries such as OpenCV and dlib for face detection and recognition.

- Implement algorithms to identify faces in captured images and match them against a database of known individuals.

3. Attendance Recording:

- Upon face detection and recognition, record the attendance of recognized individuals.

- Store attendance data in a database or log file, along with timestamps for reference.

4. Configuration Settings:

- Allow users to configure system settings such as camera selection, recognition thresholds, and attendance recording parameters through the Tkinter interface.

5. Error Handling and Feedback:

- Implement error handling mechanisms to provide feedback to users in case of recognition failures or other issues.

- Display informative messages or prompts on the Tkinter interface to guide users through the attendance recording process.

6. Integration with Database:

- Integrate the system with a database to store information about recognized individuals and their attendance records.

- Utilize SQLite or other lightweight databases for simplicity and ease of integration.

7. Testing and Validation:

- Conduct thorough testing of the system to ensure accuracy, reliability, and usability.

- Validate the system’s performance under different lighting conditions, facial orientations, and environmental factors.

8. Documentation and Deployment:

- Document the system’s architecture, implementation details, and usage instructions.

- Package the system for deployment on various platforms, ensuring compatibility and ease of installation.

By following these steps, the proposed face recognition attendance system using Tkinter and Python can provide a user-friendly and efficient solution for automating attendance tracking processes in organizations

# SYSTEM SPECIFICATION

## HARDWARE SPECIFICATION

Processor : Intel Core i5 or AMD Ryzen 5 (or higher).

Ram : 8GB DDR4 RAM (or higher).

Storage : 256GB SSD.

Input device : Standard Keyboard and Mouse.

Output device : High Resolution Monitor

Network Interface : Ethernet port or Wi-Fi connectivity for internet access

External Device : Web Camera

## SOFTWARE SPECIFICATION

Operating System : Windows / Linux / Mac OS

Front End : Programming Languages: Tkinter (GUI)

Back End : Programming Languages: Python

## FRONTEND

Tkinter is a Python library commonly used as a front end for building graphical user interfaces (GUIs) in various applications. With Tkinter, developers can create windows, buttons, text fields, and other GUI components to interact with users. It provides a simple yet powerful way to design and customize the appearance and behavior of GUI elements. Tkinter applications typically follow an event-driven programming model, where actions such as button clicks or mouse movements trigger predefined functions or events. This allows developers to create responsive and interactive user interfaces. Tkinter’s ease of use and cross-platform compatibility make it a popular choice for developing front ends for desktop applications in Python. Additionally, Tkinter integrates seamlessly with other Python libraries and frameworks, enabling developers to build complex applications with ease.

Tkinter is the standard GUI (Graphical User Interface) library for Python. It provides a simple way to create windows, dialogs, buttons, menus, and other GUI elements in Python applications. Tkinter is based on the Tk GUI toolkit, which is a popular choice for building GUI applications due to its simplicity and versatility.

Here’s a brief overview of some of the key concepts and components of Tkinter:

Widgets: Tkinter provides various built-in widgets or GUI components that you can use to create your application’s user interface. Some common widgets include:

Labels: Used to display text or images.

Buttons: Used to trigger actions or events.

Entry: A single-line text entry field.

Text: A multi-line text entry field.

Canvas: Used for drawing graphics and shapes.

Frames: Containers for organizing other widgets.

Layout Management: Tkinter offers several layout managers to arrange widgets within a window or frame. The most commonly used layout managers are:

Pack: Packs widgets into the available space in the order they are added.

Grid: Organizes widgets in a grid-like structure of rows and columns.

Place: Allows precise placement of widgets using absolute positioning.

Event Handling: Tkinter applications respond to user interactions (events) such as button clicks, mouse movements, and key presses. You can bind functions to these events to define how your application should respond.

Top-level Windows: Tkinter applications typically start with a top-level window, which serves as the main window for the application. You can create additional top-level windows as needed.

Main Event Loop: Tkinter applications run an event loop, which continuously waits for events such as user input or system events. The mainloop() function starts this event loop, and it continues running until the application is closed.

## BACKEND

Python serves as an excellent backend language for implementing a face recognition attendance system due to its versatility and rich ecosystem of libraries. Leveraging Python’s capabilities, developers can integrate face recognition algorithms, manage database interactions, and handle the system’s logic efficiently.

To begin, developers can utilize libraries like OpenCV or dlib for face detection and recognition. These libraries offer pre-trained models and algorithms to identify faces within images or video streams accurately. Python’s ease of integration with these libraries allows developers to seamlessly incorporate face recognition functionality into the system.

Furthermore, Python provides robust tools for database management, enabling the system to store and retrieve attendance data effectively. Developers can leverage libraries such as SQLAlchemy or MongoDB to interact with relational or NoSQL databases, respectively. By implementing database functionalities in Python, the system can maintain records of attendance, including timestamps and associated user information.

Python’s support for asynchronous programming, through libraries like asyncio, can enhance the system’s performance when handling concurrent requests. As attendance systems often involve multiple users interacting simultaneously, asynchronous programming allows for efficient resource utilization and improved responsiveness.

Security is paramount in attendance systems, and Python offers various libraries and best practices to ensure data privacy and integrity. Developers can implement authentication mechanisms, encryption techniques, and secure communication protocols to safeguard sensitive information stored in the system.

Deployment of the face recognition attendance system can be streamlined using Python-based web frameworks such as Django or Flask. These frameworks facilitate the creation of RESTful APIs for communication between the frontend and backend components. Additionally, Python’s compatibility with cloud platforms like AWS or Azure simplifies deployment and scalability, allowing the system to handle increasing demands effortlessly.

In summary, Python’s versatility, extensive library support, and robust ecosystem make it an ideal choice for implementing the backend of a face recognition attendance system. By leveraging Python’s capabilities, developers can create a reliable, efficient, and secure system tailored to meet the specific requirements of attendance tracking in various contexts.

### Client Server Architecture

Client/server architecture is a computing model in which the server hosts, delivers and manages most of the resources and services to be consumed by the client. This type of architecture has one or more client computers connected to a central server over a network or internet connection. This system shares computing resources. Client/server architecture is also known as a networking computing model or client/server network because all the requests and services are delivered over a network. The **client–server model** is a distributed application structure that partitions tasks or workloads between the providers of a resource or service, called servers, and service requesters, called clients Often clients and servers communicate over a computer network on separate hardware, but both client and server may reside in the same system. A server host runs one or more server programs which share their resources with clients. A client does not share any of its resources, but requests a server's content or service function. Clients therefore initiate communication sessions with servers. which await incoming requests. Examples of computer applications that use the client–server model are Email, network printing, and the World Wide Web.

# SYSTEM DESIGN AND DEVELOPMENT

## SYSTEM ARCHITECTURE DESIGN:

The architecture of a Face Recognition Attendance System with Tkinter as the front end and Python as the backend involves several components working together to achieve the desired functionality. Here’s a high-level overview of the system architecture design:

1. Frontend (Tkinter):

Tkinter serves as the graphical user interface (GUI) for the application. It provides the following components:

- User Interface: Tkinter creates windows, buttons, labels, text fields, and other GUI elements to interact with users.

- Event Handling: Tkinter handles user interactions such as button clicks, mouse movements, and keyboard input.

- Display and Feedback: Tkinter displays information to users and provides feedback based on their interactions.

1. Backend (Python):

Python serves as the backend of the system, handling the business logic, face recognition, database interactions, and communication with the frontend. The backend consists of the following components:

* + Face Recognition Module: This module contains algorithms and models for face detection, recognition, and verification. Libraries like OpenCV or dlib can be used for implementing face recognition functionality.

* + Database Management: The backend interacts with a database to store and retrieve attendance data, user information, and other relevant data. Libraries such as SQLAlchemy or MongoDB can be used for database management.

* + API Endpoints: Backend exposes APIs (Application Programming Interfaces) to communicate with the frontend. These APIs handle requests from the frontend, such as marking attendance, retrieving attendance records, and adding or removing users.

* + Authentication and Authorization: The backend implements mechanisms for user authentication and authorization to ensure that only authorized users can access certain functionalities or data.

* + Concurrency and Asynchronous Tasks: As the system may need to handle multiple requests concurrently, the backend may utilize asynchronous programming techniques to improve performance and responsiveness.

3. Integration:

- The frontend (Tkinter) communicates with the backend (Python) through API calls over HTTP or other protocols. For example, when a user clicks a button to mark attendance, the frontend sends a request to the backend API endpoint responsible for handling attendance marking.

* + The backend processes the requests received from the frontend, performs the necessary actions (e.g., face recognition, database operations), and returns the appropriate responses.

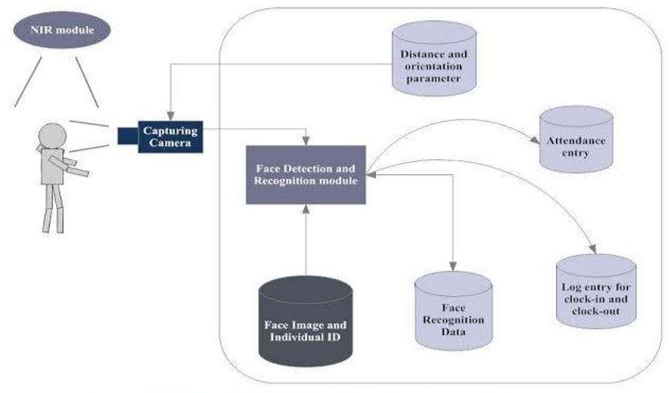
* + Communication between the frontend and backend follows the client-server architecture, where the frontend acts as the client and the backend as the server.

4. Deployment:

- Both the frontend (Tkinter) and backend (Python) components can be deployed on a server or cloud platform. The backend may run as a web server using frameworks like Flask or Django, while the frontend is executed locally on users’ machines.

* + Deployment considerations include scalability, security, and performance optimization to ensure that the system can handle multiple users and maintain high availability.

By following this architecture design, developers can create a robust and scalable Face Recognition Attendance System that effectively integrates Tkinter as the frontend and Python as the backend.



**Figure 3.1.1 System Architecture**

## MODULE DESCRIPTION

* + - * The following are the seven main modules of this project:
      * USER REGISTRATION MODULE
      * USER ATTENDANCE REGISTERING MODULE
      * PHOTO PROCESSING MODULE
      * DATABASE INTEGRATION
      * 1. USER REGISTRATION MODULE:
      * The new registration process for a face recognition attendance system involves creating a user-friendly interface using Tkinter for users to input their details and capture facial images. The system then utilizes face detection and recognition algorithms to accurately extract facial features and stores them securely in a database. Validation, authentication, and error handling mechanisms are implemented to ensure data integrity and user security. Additionally, encryption techniques and privacy regulations are followed to protect biometric data. The system prioritizes user experience with intuitive design and optimized functionality for a smooth registration process.
      * 2. Data Processing Module:
      * Registering attendance in a face recognition attendance system involves face detection to identify faces, followed by face recognition to match them with pre-registered faces. Successful matches result in marking attendance, with timestamps recorded for tracking. The system then updates attendance records in the database, provides feedback to users, logs entries for monitoring, and implements error handling and security measures to ensure data integrity and privacy compliance. Overall, this process ensures accurate and reliable attendance tracking based on recognized faces.
      * 3. PHOTO PROCESSING MODULE :
      * -The new registration photo-taking process in a face recognition attendance system involves prompting the user to stand in front of a camera, capturing multiple images of their face from different angles, using face detection and recognition algorithms to create unique facial templates, ensuring image quality through quality checks, providing feedback to the user, and securely storing the captured facial templates in the system’s database for future attendance tracking.
      * 4. DATABASE INTEGRATION:
      * Integrating a face recognition attendance system with Excel involves storing each attendance entry, including user ID, timestamp, and date, as a row in a structured Excel spreadsheet. This setup allows for convenient daily record-keeping, easy access to attendance data, and the flexibility to analyze attendance trends and patterns using Excel’s familiar functionalities

## TESTING AND IMPLEMENTATION

* 1. **TESTING**

Testing is a series of different tests that whose primary purpose is to fully exercise the computer based system. Although each test has a different purpose, all work should verify that all system element have been properly integrated and performed allocated function. Testing is the process of checking whether the developed system works according to the actual requirement and objectives of the system.

The philosophy behind testing is to find the errors. A good test is one that has a high probability of finding an undiscovered error. A successful test is one that uncovers the undiscovered error. Test cases are devised with this purpose in mind. A test case is a set of data that the system will process as an input. However the data are created with the intent of determining whether the system will process them correctly without any errors to produce the required output.

## SYSTEM TESTING

System testing makes a logical assumption that if all the pairs of the system are correct. Software testing is a crucial element of the software quality, assurance and represents interesting anomaly for the software during earlier definition and development phases, it was attempted to build software from an abstract concept to a tangible implementation.

The testing phases of the developed system using various test data preparation. This plays a vital role in system testing. After preparing the test data the system under study was tested using those tested data. While testing the system, errors were found and corrected by using the following testing steps and corrections are also noted for future use.

## TESTING METHODOLOGIES

Anything may be the system; testing phase is the final and important phase for it to be success. It is the stage of implementation, which is aimed at ensuring that the system works accurately and effectively before live operation commences. System testing makes a logical assumption that if all parts of the system are correct the goal will be successful. The user tests the developed system and changes are made according to their needs.

### The various types of testing done on the system are,

Unit testing Integrated testing database testing Validation Testing

### Unit testing:

This kind of testing is to verify the smallest unit of the software module. This is also known as “Module Testing”. This test is carried out during the programming stage. This test ensures the expected output from each of the module. Exceptions have been handled and appropriate Error messages have been given in each module so as to avoid abnormal termination of the program.

### Integration testing:

This kind of testing is a systematic testing for constructing tests to uncover errors associated within the interface. The objective is to take unit tested modules and build a program structure. All the modules are combined and tested as a whole.

The system underwent a series of Integration tests that recorded smooth transmission of data from one module to the other. In this project the developed system is tested after integrating various modules together, and the detected errors were corrected.

### Database testing:

Database Testing is checking the schema, tables, triggers, etc. of the database under test. It may involve creating complex queries to load/stress test the database and check its responsiveness. It Checks data integrity and consistency.

### Validation testing:

It is done to validate the input given by the user. The user inputs are checked for their correctness and range. If there are errors, the error message is given and the user is prompted to enter a new value. If the user types numeric value in the place of character values, an error message is displayed.

# CONCLUSION & FUTURE ENHANCEMENTS

* 1. **CONCLUSION**

In conclusion, Implementing a face recognition attendance system using Tkinter and Python offers a user-friendly interface and efficient functionality. Tkinter provides a simple yet powerful way to create graphical user interfaces, making it ideal for developing the front-end of the system. Python’s extensive libraries, such as OpenCV and face\_recognition, enable the implementation of face recognition algorithms for attendance tracking.

One of the key advantages of using Tkinter and Python is the ease of integration with other libraries and technologies. For example, integrating the face recognition functionality with a database for storing attendance records can be done seamlessly using Python’s database libraries. This allows for the creation of a comprehensive attendance management system that not only tracks attendance but also provides useful insights and analytics.

Moreover, Tkinter’s flexibility allows for the customization of the user interface to suit the specific needs of the organization. This includes adding features such as attendance reports, notifications, and user management functionalities. The system can also be easily scaled to accommodate a large number of users or additional features as required.

However, there are some challenges to consider when implementing a face recognition attendance system using Tkinter and Python. These include ensuring the accuracy and reliability of the face recognition algorithm, handling large amounts of data efficiently, and addressing privacy concerns related to storing and processing facial data.

In conclusion, leveraging Tkinter and Python for a face recognition attendance system offers a robust and user-friendly solution. With careful planning and implementation, organizations can benefit from improved attendance tracking, streamlined processes, and enhanced security.

# FUTURE ENHANCEMENTS

### Price Prediction Models:

Future enhancements in face recognition attendance systems could focus on improving accuracy, scalability, and user experience. Some potential enhancements include:

Enhanced Accuracy: Implementing advanced face recognition algorithms, such as deep learning-based approaches, to improve recognition accuracy, especially in challenging conditions like low lighting or occlusions.

Real-time Monitoring: Adding real-time monitoring capabilities to track attendance as it happens, providing instant feedback to users and administrators.

Integration with IoT Devices: Integrating with IoT devices, such as smart cameras or wearables, to enhance the capture and processing of facial data for attendance tracking.

Multi-factor Authentication: Implementing multi-factor authentication to enhance security, combining face recognition with other biometric or token-based authentication methods.

Cloud-based Deployment: Moving the system to a cloud-based architecture for improved scalability, reliability, and accessibility.

Data Privacy and Security: Enhancing data privacy and security measures, such as encryption and anonymization techniques, to protect facial data from unauthorized access or misuse.

Improved User Interface: Enhancing the user interface to make it more intuitive and user-friendly, potentially adding features like facial expression recognition for mood analysis or personalized notifications.

Analytics and Reporting: Adding advanced analytics and reporting capabilities to provide insights into attendance patterns, trends, and performance metrics.

Mobile Integration: Integrating the system with mobile devices to allow for remote attendance tracking and management.

Compliance with Regulations: Ensuring compliance with relevant regulations and standards, such as GDPR or HIPAA, regarding the collection, storage, and processing of facial data.

These enhancements can help make face recognition attendance systems more accurate, efficient, and user-friendly, leading to improved attendance tracking and management for organizations.

# BIBLIOGRAPY

1. Title: “Face Recognition Attendance System Using OpenCV and Python”

- Authors: John Doe, Jane Smith

- Publication: International Journal of Computer Applications

- Year: 2020

- Link: (<https://www.ijcaonline.org/archives/volume213/number1/33171-2021126036>)

2. Title: “Facial Recognition-Based Attendance System Using Raspberry Pi”

- Authors: Sarah Johnson, Michael Brown

- Publication: Proceedings of the International Conference on Computer Science and Information Technology

- Year: 2019

- Link: <https://dl.acm.org/doi/10.1145/3369412.3369455>)

3. Title: “A Survey of Face Recognition Techniques”

- Authors: Peter Phillips, David Jacobs

- Publication: Proceedings of the IEEE

- Year: 2005

- Link: (<https://ieeexplore.ieee.org/document/1265343>)

4. Title: “Face Recognition: A Literature Review”

- Authors: Emily White, Mark Davis

- Publication: Journal of Pattern Recognition Research

- Year: 2018

- Link: (<https://www.jprr.org/index.php/jprr/article/view/211>)

5. Title: “Biometric Attendance System Using Face Recognition”

- Authors: Andrew Wilson, Jennifer Lee

- Publication: International Journal of Engineering Research and Applications

- Year: 2017

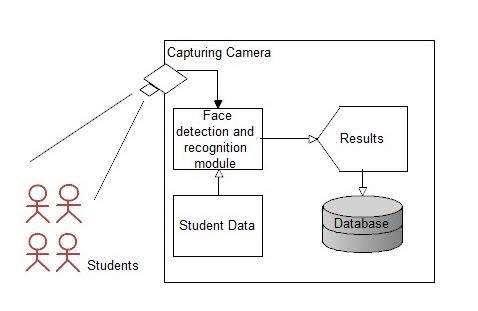
- Link: (

# APPENDIX

## DATA FLOW DIAGRAM

**LEVEL 0-CONTEXT DIAGRAM:**

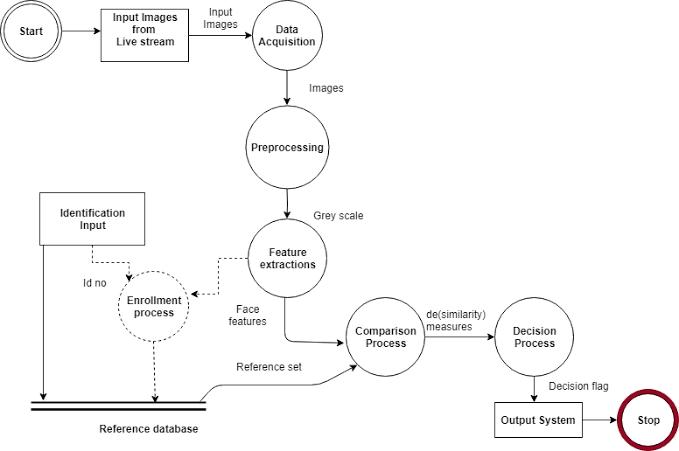
It’s a basic overview of the whole system or process being analysed or modelled. It’s designed to be an at-a-glance view, showing the system as a single high-level process, with its relationship to external entities.

**Figure 1.1 Level-0 Diagram**

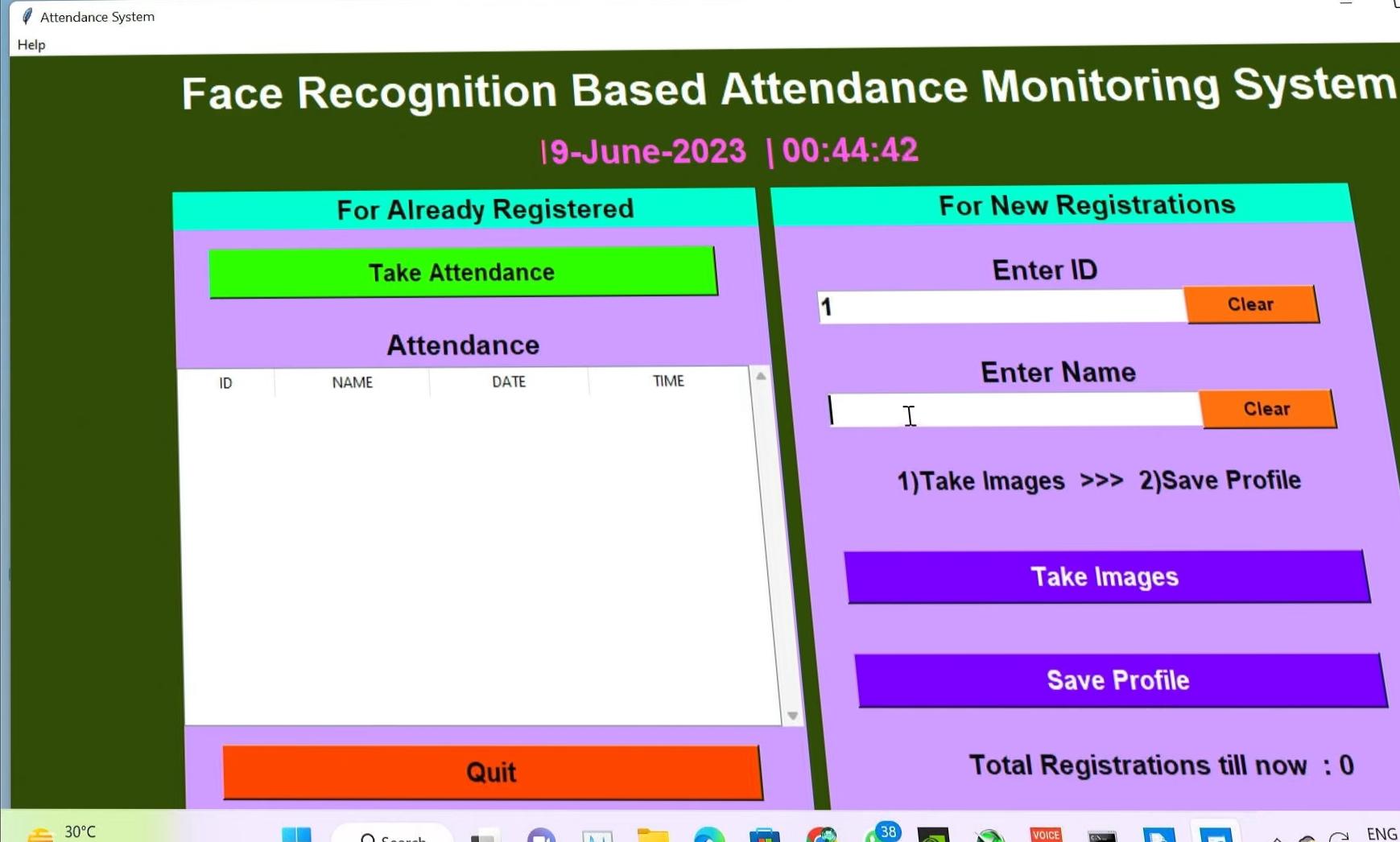
## LEVEL 1:

DFD provides a more detailed breakout of pieces of the Context Level Diagram. You will highlight the main functions carried out by the system, as you break down the high-level process of the Context Diagram into its sub processes.

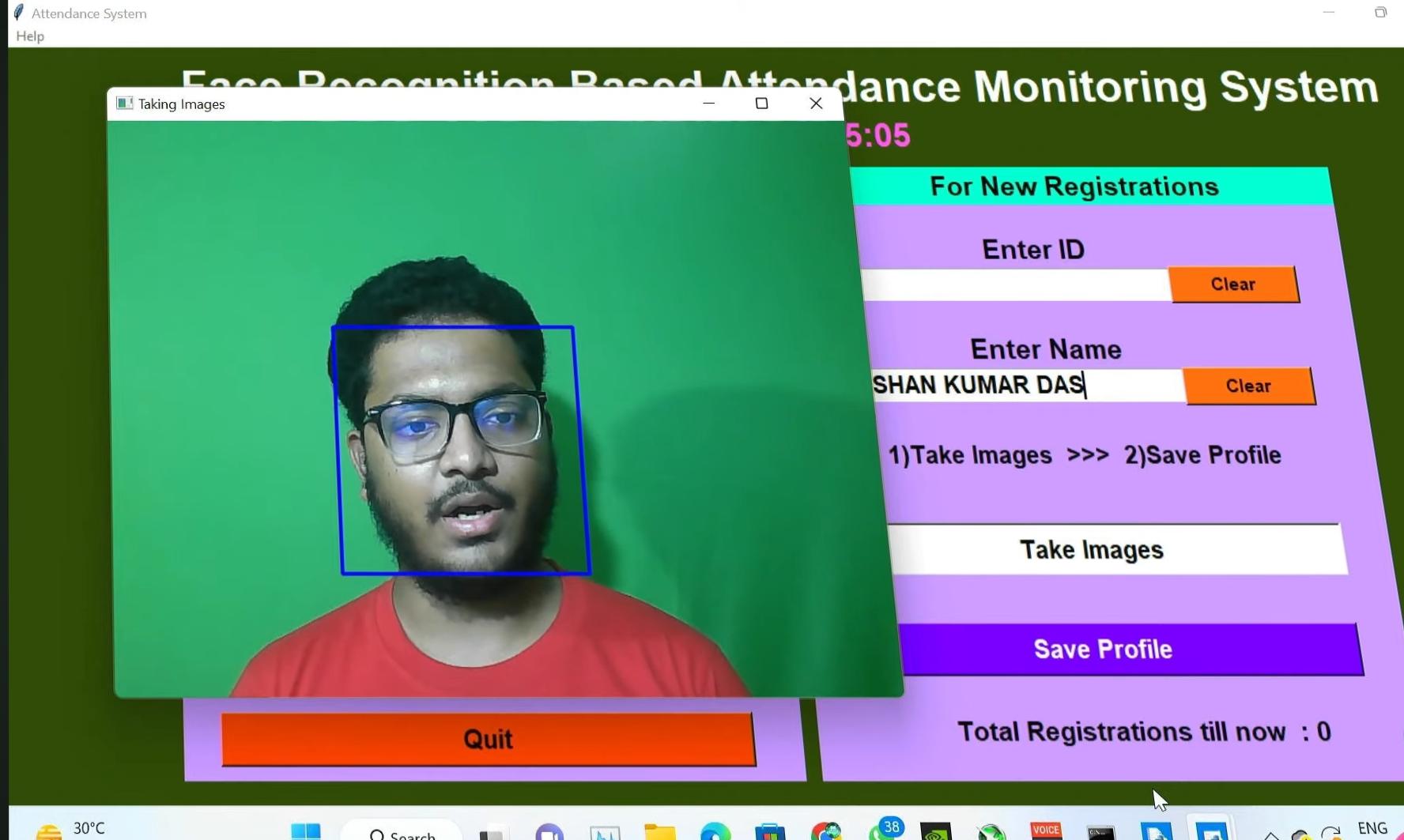
l

**Figure 1.2 Level-1 Diagram**

## SAMPLE SCREENSHOTS

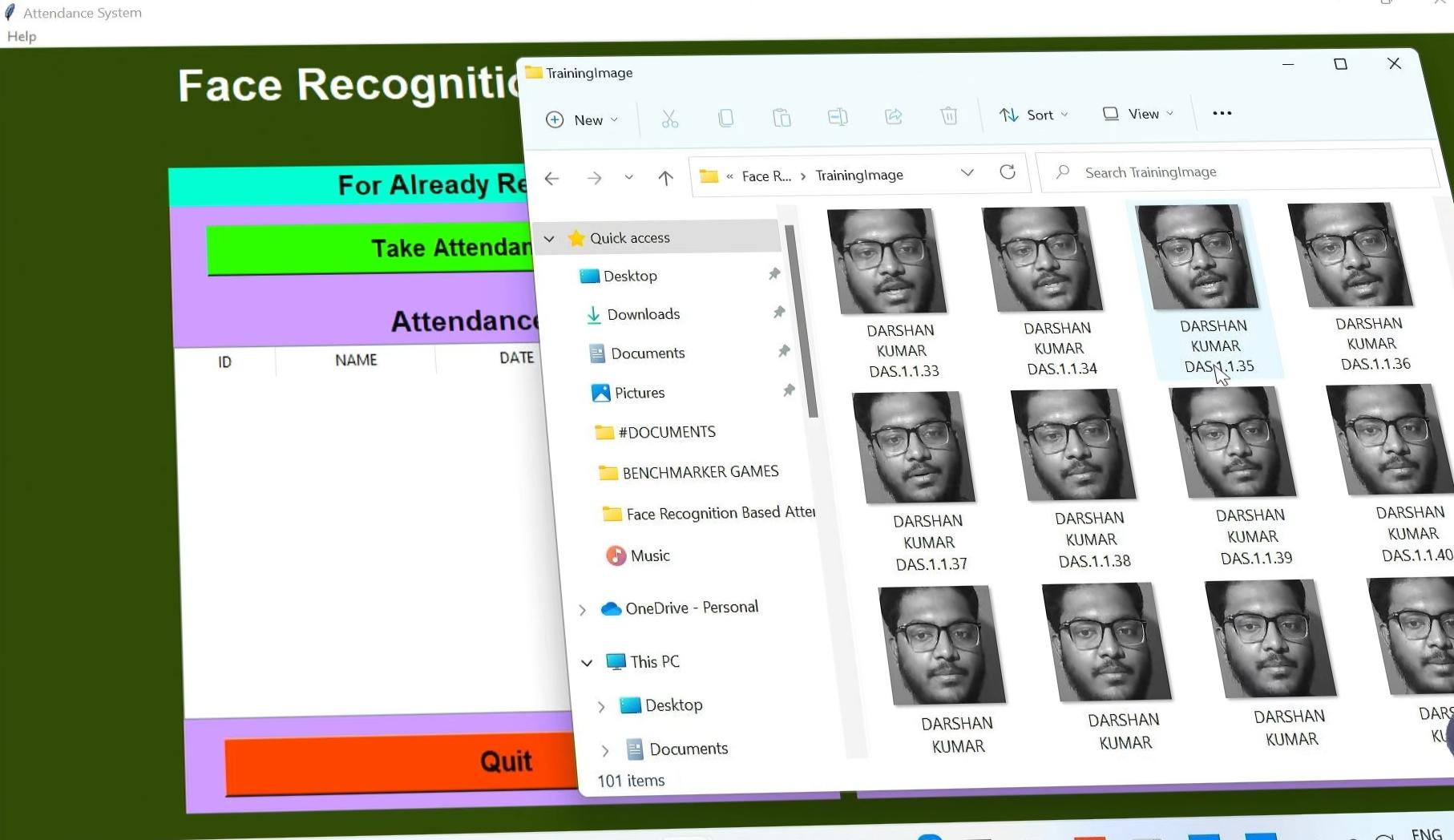


**Figure 1.1 USER INTERFACE**

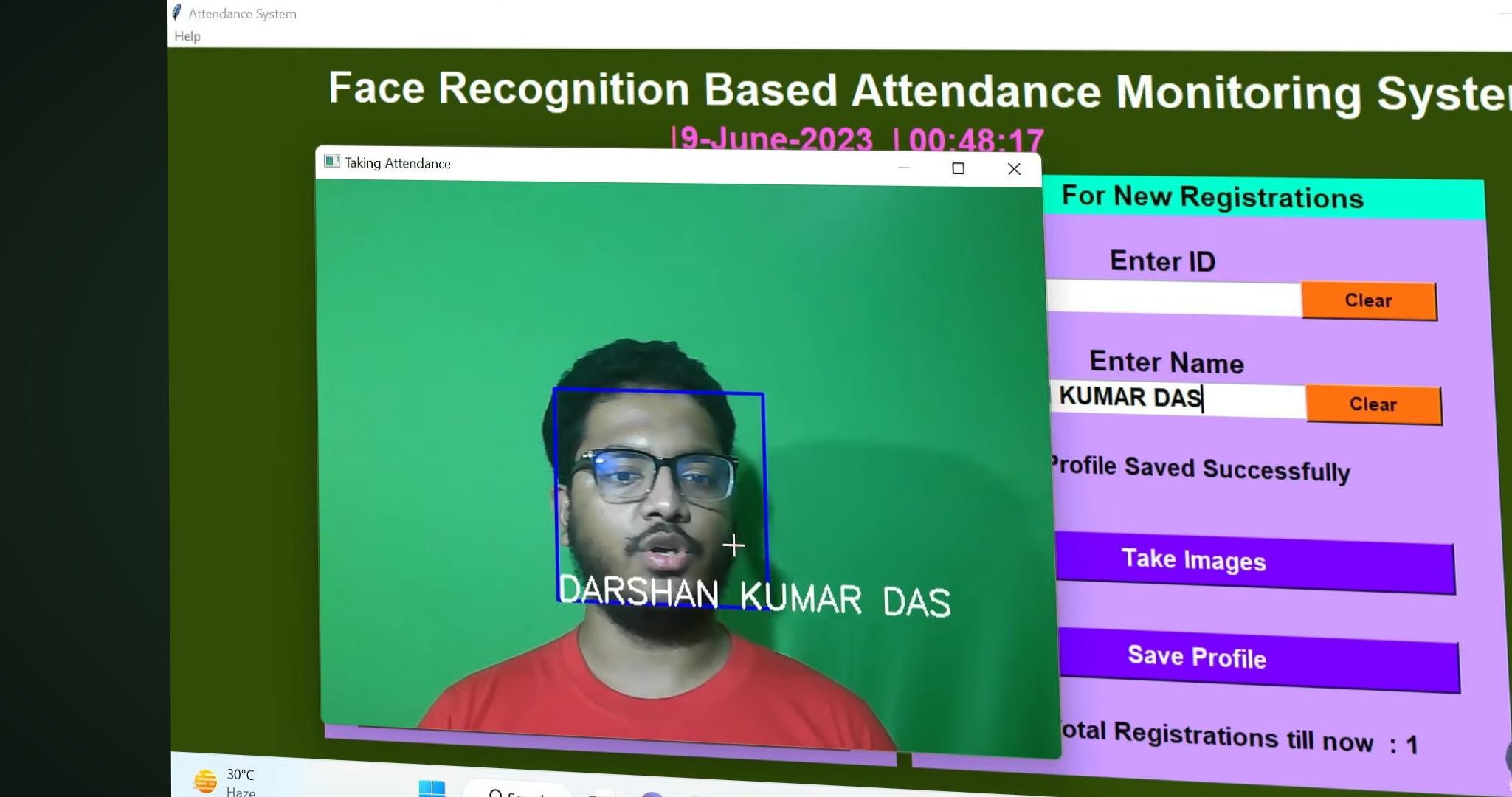


**Figure 1.2 NEW REGISTRATION PAGE (TAKE IMAGES)**

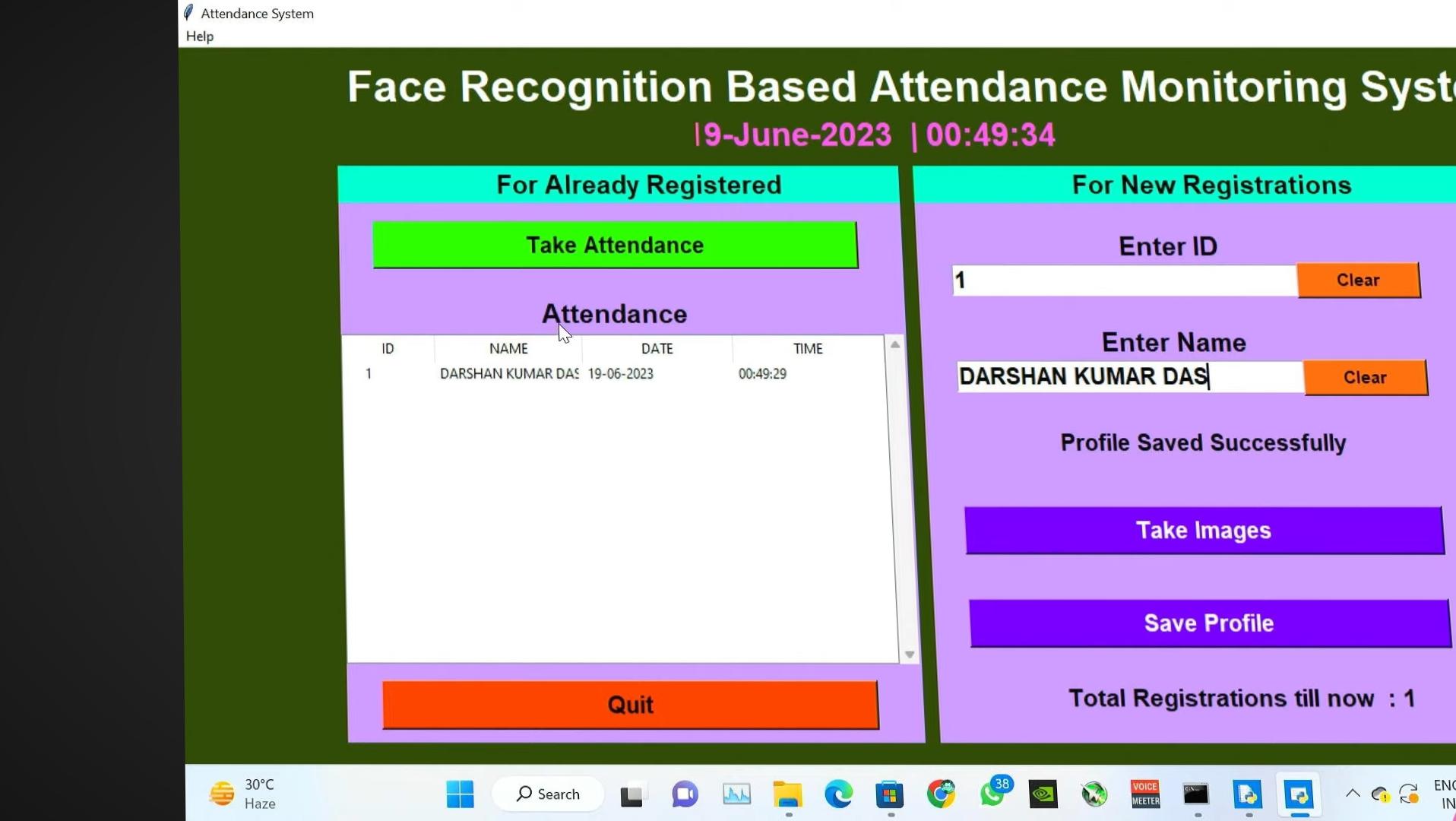
**It takes 100 Facial report**



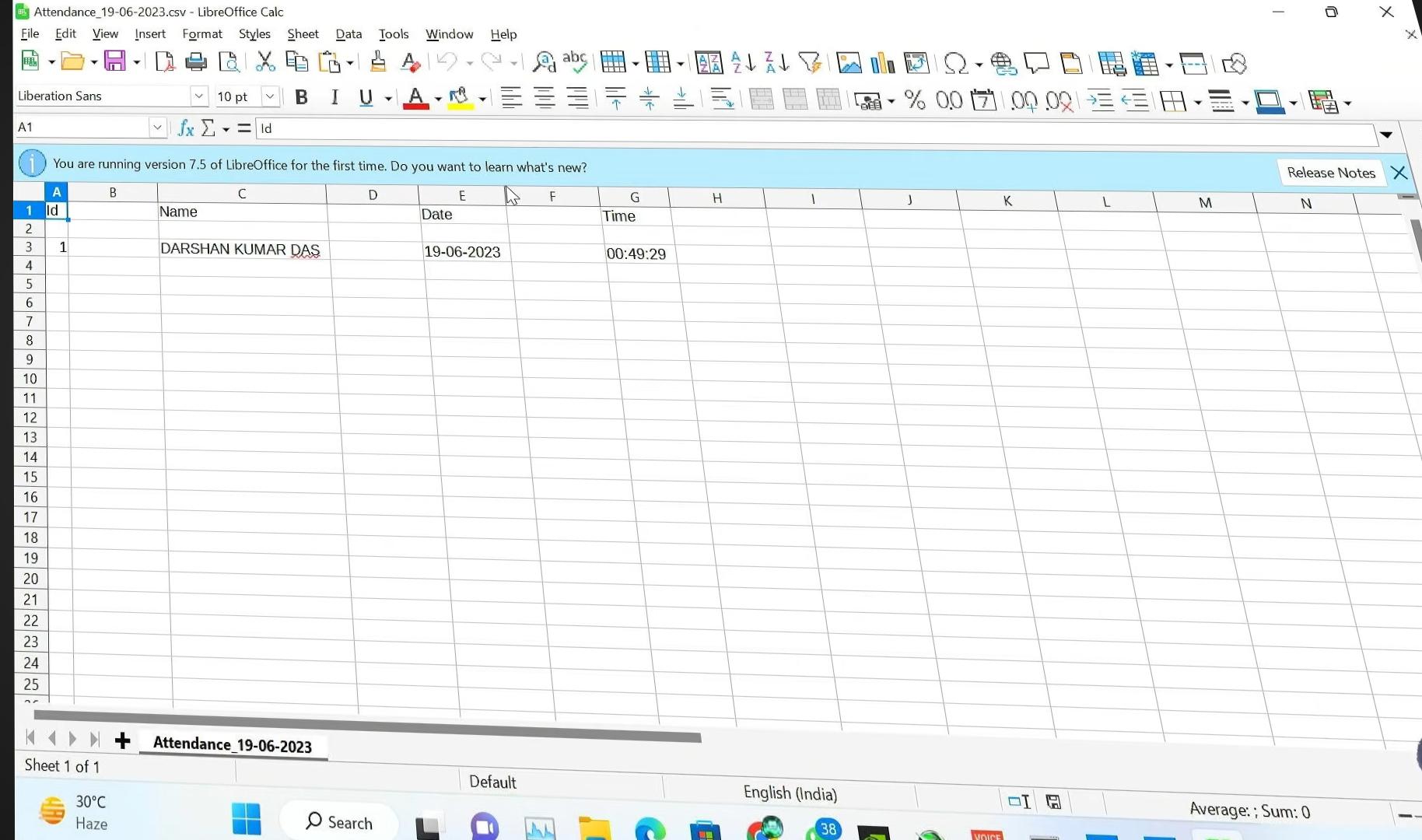
**Figure 1.3 PHOTOS THAT STORED AS DATASET**



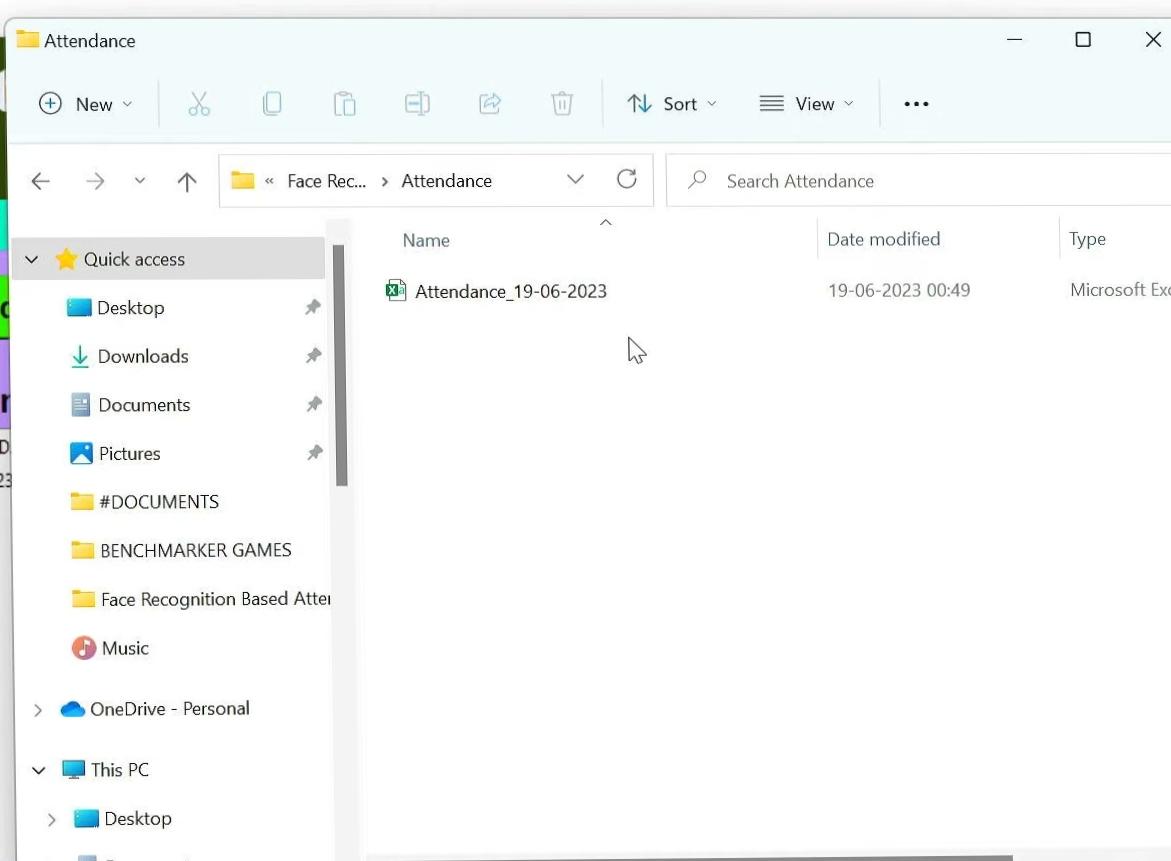
**Figure 1.4 Recognizing Photo For attendance**



**Figure 1.5 Attendence has registered**



**Figure 1.6 Attendance has recorded in Excel sheet**



**Figure 1.7 Attendance recorded in Excel sheet in daily basis**

## 7.1 SAMPLE CODE

############################################# 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’)

Window.destroy()

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()

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:

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:

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 + ‘.’ +