**A**

**Semester Project-II**

**Report**

**On**

**“Attendance Management System Using Face Recognition”**

**By**

1. Parag Krishna Nikum

2. Raj Ashok Sonawane

3. Tanvi Ritesh Badgujar

4. Nikita Laxmikant Wagh



**Department of Computer Science & Engineering (Data Science)**

**The Shirpur Education Society’s**

**R. C. Patel Institute of Technology, Shirpur - 425405.**

**[2024-25]**

**A**

**Semester Project-II Report**

**On**

**“Attendance Management System Using Face Recognition”**

In partial fulfillment of requirements for the degree of

Bachelor of Technology

In

Computer Science & Engineering (Data Science)

**Submitted By**

1. Parag Krishna Nikum

2. Raj Ashok Sonawane

3. Tanvi Ritesh Badgujar

4. Nikita Laxmikant Wagh

**Under the Guidance of**

Prof. Ujwala M. Patil



**The Shirpur Education Society’s**

**R. C. Patel Institute of Technology, Shirpur - 425405.**

**Computer Science & Engineering (Data Science)**

**[2024-25]**



**The Shirpur Education Society’s**

**R. C. Patel Institute of Technology**

**Shirpur, Dist. Dhule (M.S.)**

**Computer Science & Engineering (Data Science)**

**CERTIFICATE**

This is to certify that the Semester Project-II entitled “**Attaindance Management System using face recognition ”** has been carried out by team:

**1. Parag Krishna Nikum**

**2. Raj Ashok Sonawane**

**3. Tanvi Ritesh Badgujar**

**4. Nikita Laxmikant Wagh**

under the guidance of **Prof**. **Ujwala S.Patil** in partial fulfillment of the requirement for the degree of Bachelor of Technology in Department of Computer Science & Engineering (Data Science) (Semester-IV) of Dr. Babasaheb Ambedkar Technological University, Lonere during the academic year 2024-25.

**Date:**

**Place: Shirpur**

**Guide Semester Project-II Coordinator**

**Prof. Ujwala M. Patil Prof. S. P. Salunkhe**

**H.O.D. Director**

**Prof. Dr. Ujwala. M. Patil Prof. Dr. J. B. Pat**

**ACKNOWLEDGEMENT**

It is a matter of great pride and satisfaction for us to present this project report on “Attendance Management System Using Face Recognition” which has been the outcome of sincere and dedicated efforts by our entire team.

We are grateful to our esteemed guide, **Prof. Ujwala M. Patil** , for her invaluable guidance, encouragement, and support throughout the course of this project. Her mentorship not only helped us in successfully completing this project but also in gaining deep insights into the field of machine learning and computer vision.

We also express our sincere gratitude to **Prof. S. P. Salunkhe**, our Semester Project Coordinator, for her constant monitoring, helpful feedback, and organizational support which enabled smooth progress throughout the semester.

Our heartfelt thanks go to **Prof. Ujwala M. Patil**, Head of the Department of Artificial Intelligence and Machine Learning , for her constant encouragement and provision of academic resources that greatly helped us in executing the project effectively.

We also take this opportunity to thank **Prof. Dr. J. B. Pat**, Director of R.C.Patel Institute of Technology , for providing an excellent academic environment and the required infrastructural facilities for completing our project work.

Last but not the least, we extend our sincere thanks to all the faculty members of our department, technical staff, and our classmates who directly or indirectly helped us at various stages of this project.

We feel privileged and honored to have had the opportunity to work on this innovative project which has helped us enhance our technical skills, team coordination, and practical knowledge

**INDEX**

|  |  |  |  |
| --- | --- | --- | --- |
| **Chapter No.** | | **Topic** | **Page No.** |
|  |  | ABSTRACT | 1 |
| 1 |  | INTRODUCTION | 2 |
|  | 1.1 | CONCEPTS FROM CURRENT SEMESTER SUBJECTS | 2 |
|  | 1.2 | 1.2 PROBLEM STATEMENT | 4 |
| 2 |  | LITERATURE SURVEY | 6 |
|  | 2.1 | REVIEW OF EXISTING SYSTEMS | 6 |
|  | 2.2 | LIMITATIONS OF EXISTING SYSTEMS | 8 |
|  | 2.3 | CONSIDERATIONS FOR SIMPLE-LEVEL PROJECT | 9 |
| 3 |  | RELATED CONCEPTS | 11 |
|  | 3.1 | PYTHON PROGRAMMING AND SCRIPT EXECUTION | 11 |
|  | 3.2 | IMAGE PROCESSING AND COMPUTER VISION | 12 |
|  | 3.3 | MACHINE LEARNING AND FACIAL RECOGNITION | 13 |
|  | 3.4 | GUI DEVELOPMENT USING TKINTER | 14 |
|  | 3.5 | FILE HANDLING AND LOCAL STORAGE | 15 |
|  | 3.6 | MODULAR ARCHITECTURE | 16 |
| 4 |  | SOFTWARE AND HARDWARE REQUIREMENTS | 17 |
|  | 4.1 | SOFTWARE REQUIREMENTS | 17 |
|  | 4.2 | HARDWARE REQUIREMENTS | 19 |
|  | 4.3 | SYSTEM CONFIGURATION SUMMARY | 20 |
|  | 4.4 | PERFORMANCE EXPECTATIONS AND JUSTIFICATION | 21 |
| 5 |  | IMPLEMENTATION DETAILS | 22 |
|  | 5.1 | ALGORITHM | 22 |
|  | 5.2 | STRUCTURE OF PROJECT FILES | 24 |
|  | 5.3 | SNAPSHOTS OF OUTPUT | 24 |
|  | 5.4 | RESULT DISCUSSION | 25 |
|  |  | CONCLUSION |  |
|  |  | BIBLIOGRAPHY |  |

**FIGURE INDEX**

|  |  |  |
| --- | --- | --- |
| **Figure No.** | **Name** | **Page No.** |
| **1** | home dashboard | 27 |
| **2** | Register image | 27 |
| **3** | Take attaindance | 28 |
| **4** | Show attaindance | 28 |

**ABSTRACT**

*This project report presents the development of an advanced and efficient attendance management system that utilizes face recognition technology to automatically identify and mark the attendance of students. Traditional attendance systems are often time-consuming, inefficient, and vulnerable to fraudulent practices like proxy attendance. The face recognition-based attendance system addresses these challenges by offering a secure, automated, and contactless solution.*

*The system is designed using Python programming language with the integration of OpenCV library for computer vision functionalities. It uses Haar Cascade classifiers for face detection and the Local Binary Pattern Histogram (LBPH) algorithm for face recognition. The application captures facial images of students, stores them in a dataset, and trains a model to recognize individuals in real-time using webcam input. Once a face is recognized, the system logs the attendance of that student automatically into a CSV file along with the date and time.*

*The project incorporates various essential computing concepts including GUI development with Tkinter, image processing, machine learning, and file system management. The student registration module allows for easy input of student details and collection of face samples. A training module processes the images and builds a robust recognition model. The real-time attendance module ensures accurate and quick identification during classroom sessions.*

*Additionally, the system provides functionalities to view past attendance records, filter data by date or subject, and export the reports for administrative use. It ensures data security, minimizes manual effort, and streamlines the attendance process for educational institutions.*

*This project demonstrates a practical implementation of theoretical concepts learned in coursework, contributing to the digital transformation of academic processes. The system not only reduces administrative workload but also enhances transparency, reliability, and efficiency in student attendance management.*

***Domain:*** *Computer Vision, Machine Learning, Attendance Automation*

**CHAPTER 1**

**INTRODUCTION**

In today’s rapidly advancing technological world, educational institutions are increasingly adopting digital tools to enhance administrative efficiency and academic performance. One of the most essential yet frequently time-consuming tasks in academia is managing student attendance. Traditional attendance systems, which include manual roll-call or physical registers, are susceptible to numerous issues such as human error, inefficiency, proxy attendance, and a lack of data analytics.

With the evolution of artificial intelligence, particularly in the domain of computer vision, face recognition technology offers a highly efficient, contactless, and intelligent method of recording attendance. This technology eliminates the flaws associated with traditional systems and provides a smart, real-time, and secure solution.

The project titled **“Attendance Management System Using Face Recognition”** is a software application that aims to replace manual and semi-automated attendance systems with an intelligent, camera-based facial recognition approach. Developed using Python, OpenCV, and machine learning techniques, this system captures and processes facial features to automatically identify individuals and record their attendance.

This desktop-based application integrates concepts from several core subjects of the current semester and provides a real-world application of theoretical knowledge gained in Database Management Systems (DBMS), Design and Analysis of Algorithms (DAA), and Artificial Intelligence (AI).

**1.1 Concepts from Current Semester Subjects**

The development and functionality of this project are heavily grounded in the academic principles covered in the current semester. The subjects that directly contribute to the foundation and success of the project include:

**1.1.1 Database Management Systems (DBMS)**

DBMS plays a pivotal role in efficiently organizing, storing, retrieving, and manipulating data. In the context of this project, DBMS principles are applied to manage student information, attendance records, and image datasets.

* **Data Modeling and Relational Schema**: Student data is modeled and stored in structured tabular formats such as CSV files, simulating relational databases. The schema design includes attributes such as student ID, name, image path, and attendance timestamps.
* **Normalization**: The database structure is normalized to eliminate redundancy and ensure data consistency. For example, attendance records are separated from student details, maintaining a one-to-many relationship.
* **Querying and Retrieval**: SQL-like operations are mimicked using Python’s pandas and CSV modules, enabling data filtering, sorting, and exporting capabilities.
* **Security and Integrity**: Basic data validation is enforced during input to prevent anomalies, and proper file handling ensures that student information is not lost or corrupted.

**1.1.2 Design and Analysis of Algorithms (DAA)**

The reliability and performance of a face recognition system largely depend on the efficiency of its underlying algorithms. DAA principles are utilized in designing and analyzing various components of the system, especially face detection and recognition.

* **Face Detection Using Haar Cascade Classifier**: This uses a tree-based algorithm trained on positive and negative images to quickly detect faces in real-time, showcasing practical application of greedy and recursive algorithms.
* **Face Recognition Using LBPH**: The Local Binary Pattern Histogram algorithm is chosen for its robustness, low computational complexity, and suitability for grayscale image processing.
* **Algorithm Optimization**: The overall system is structured into modular algorithms — including image capture, training, and prediction — to ensure scalability and maintainability. Each algorithm is analyzed for time and space complexity to enhance efficiency.
* **Flow Control and Error Handling**: Algorithmic logic is implemented to manage unexpected inputs, camera errors, and recognition mismatches, contributing to the robustness of the system.

**1.1.3 Artificial Intelligence (AI)**

Artificial Intelligence forms the core of the system's capability to identify and distinguish between individuals. The AI components are focused on learning from facial data and making accurate predictions during real-time use.

* **Pattern Recognition**: AI techniques are used to identify unique facial features such as eyes, nose, and jaw structure, which form the basis for recognition.
* **Training and Classification**: The system uses supervised learning. During training, labeled images of students are processed, and the LBPH algorithm builds a classification model based on the patterns identified.
* **Decision-Making and Adaptation**: AI enables the model to make intelligent decisions — whether to recognize, ignore, or request re-capture of an image — based on confidence scores and feature matching thresholds.
* **Automation and Self-improvement**: Although basic, the system demonstrates the potential for AI models to be retrained over time with more data, improving accuracy and learning from variations in lighting, expressions, and camera angles.

Together, these subject areas contribute to a holistic learning experience, demonstrating how academic theories can be applied to build a practical and functional real-world application.

**1.2 Problem Statement**

**1.2.1 Title:**

Development of an Automated Attendance Management System Using Face Recognition

**1.2.2 Justification:**

Attendance management is an indispensable part of institutional operations, particularly in educational settings. Conventional systems such as manual registers, fingerprint scanners, or ID cards come with a plethora of limitations:

* **Manual Register Systems**: Susceptible to human error, time delays, and manipulation through proxy attendance.
* **Biometric Systems**: Although digital, they pose hygiene issues, especially in post-pandemic contexts, and require costly hardware and regular maintenance.
* **RFID Cards and Barcode Systems**: Can be easily lost, forgotten, or misused by swapping cards.

The increasing need for contactless, intelligent systems in educational institutions motivates the development of this project. The justification for implementing a face recognition-based system is founded on several grounds:

* **Accuracy**: Facial recognition reduces the chances of error and proxy attendance significantly.
* **Efficiency**: It automates the attendance process, reducing classroom time wasted on roll calls.
* **Security**: Student data is securely stored and retrieved, ensuring privacy and compliance.
* **Scalability**: Can be deployed in classrooms, exam halls, or even integrated into online platforms.
* **Real-Time Processing**: Attendance can be marked instantly and exported for further analysis.

This system not only modernizes attendance tracking but also aligns with the broader goals of digital transformation in education. It serves as a meaningful application of computer science theories and prepares students for the challenges of developing AI-integrated software systems in their careers.

**CHAPTER 2**

**LITERATURE SURVEY**

In this chapter, we will review existing systems related to the "Attendance Management System Using Face Recognition" and analyze their limitations. We will also discuss the considerations necessary for developing a simple-level project for attendance management.

**2.1 Review of Existing Systems**

Attendance management systems have been evolving, with a shift from manual systems to more automated solutions. While many educational institutions still rely on manual methods for recording attendance, there are advanced systems that utilize technologies like RFID, biometrics, and face recognition for better accuracy and efficiency. Below, we review the existing systems that are commonly used:

**2.1.1 Manual Attendance Systems**

In many educational institutions, manual attendance is still the norm. In this system, teachers or administrators take attendance by calling the students' names or having them sign an attendance sheet. This approach has several significant drawbacks:

* **Time-Consuming**: Manual attendance recording is a time-consuming process, especially in large classes. It takes up valuable instructional time and may lead to delays.
* **Inaccuracy**: Manual systems are prone to errors, such as misheard names, incorrect entries, or the possibility of students marking each other's attendance.
* **Lack of Automation**: There is no automation in the process, which leads to human errors, inaccurate records, and difficulties in tracking attendance over time.
* **Fraud**: Students may mark each other's attendance, or fake attendance sheets may be used, which compromises the integrity of the attendance data.

Given these limitations, manual attendance systems are becoming less viable, especially as the scale of educational institutions increases.

**2.1.2 Biometric-Based Attendance Systems**

In contrast to manual attendance, biometric systems—especially fingerprint and facial recognition systems—are being adopted for automating attendance. These systems use physical attributes (fingerprints or facial features) for identification and tracking. Some advantages of biometric systems include:

* **Accuracy**: Biometric systems are much more accurate than manual systems. They minimize the chances of human error and fraud.
* **Security**: The data used for identification (such as fingerprints or facial recognition) is unique to each individual, making it difficult for students to fake their attendance.
* **Efficiency**: These systems allow for quick and automated attendance tracking, freeing up time for teaching activities.

However, biometric systems also have their limitations, such as the need for specialized hardware (like fingerprint scanners or cameras) and concerns about privacy and data security.

**2.1.3 Face Recognition-Based Attendance Systems**

Face recognition technology has become a popular choice in attendance management systems because of its non-intrusive nature and high accuracy. In face recognition systems, a camera captures the student's face, and the system identifies the student based on their facial features. Advantages include:

* **Non-Intrusive**: Students do not need to carry cards or scan fingerprints, making the system convenient and passive.
* **Efficiency**: Attendance is automatically recorded as soon as the student is recognized, saving time and reducing errors.
* **Scalability**: These systems can be scaled easily to accommodate large numbers of students, making them suitable for institutions with a significant student population.

However, face recognition systems may face challenges such as lighting conditions, face masks, or significant changes in appearance, which may affect their accuracy.

**2.1.4 Cloud-Based Attendance Systems**

Cloud-based systems are another emerging trend in attendance management. These systems store data on remote servers, allowing administrators and teachers to access attendance records from anywhere. Some benefits of cloud-based systems include:

* **Remote Access**: Attendance data is accessible from anywhere, making it easier for administrators to manage and review records.
* **Real-Time Updates**: Data is updated in real time, providing immediate access to attendance information.

However, cloud-based systems also come with their own challenges, such as reliance on internet connectivity and the need for ongoing subscription costs for cloud storage.

**2.2 Limitations of Existing Systems**

While existing attendance management systems (especially manual systems) offer various methods for tracking attendance, they come with several limitations that can hinder efficiency and accuracy. Below are the key limitations of current systems:

**2.2.1 Dependency on Manual Intervention**

In manual attendance systems, there is a heavy dependency on teachers or administrators to record attendance, which is time-consuming and prone to human error. Teachers may forget to take attendance or make errors when writing down student names, leading to inaccuracies in the attendance records.

**2.2.2 Inaccuracy and Fraud**

In manual systems, the risk of inaccuracies or fraudulent attendance is high. Students can mark each other’s attendance, or teachers may forget to take attendance for some students. This compromises the integrity of the attendance system and can lead to incorrect attendance records, which can affect academic performance and administrative reporting.

**2.2.3 Lack of Automation and Reporting Features**

Manual systems lack automation, which means that attendance data must be manually entered into a digital system or reported on paper. This results in inefficiencies in tracking attendance over time. Furthermore, manual systems often do not have built-in features for generating reports or analyzing trends, which makes it difficult to evaluate attendance patterns or identify potential issues.

**2.2.4 Limited Security and Privacy Concerns**

Biometric and face recognition systems are not immune to privacy concerns. Storing sensitive biometric data, such as fingerprints or facial images, raises significant privacy issues, especially if proper security protocols are not in place. Unauthorized access or breaches of this data can lead to severe consequences for both institutions and students.

**2.2.5 Scalability Challenges**

Many existing systems, particularly manual or RFID-based systems, struggle with scalability. For instance, manual attendance taking becomes increasingly difficult to manage as the number of students grows. RFID-based systems may also require a significant investment in hardware, which can be difficult to scale to large institutions.

**2.3 Considerations for Simple-Level Project**

For a simple-level project like the "Attendance Management System Using Face Recognition," several considerations need to be taken into account to ensure a balance between feasibility, functionality, and ease of implementation. The following factors are key:

**2.3.1 Simplicity in Design**

The project must focus on core functionalities and avoid unnecessary complexity. For example, face recognition can be implemented using simple algorithms like Local Binary Pattern Histogram (LBPH), which is computationally efficient and works well in many real-world scenarios. This avoids the need for complex deep learning models that would require more powerful hardware.

**2.3.2 Cost-Effectiveness**

The system should be designed to be cost-effective. By utilizing existing hardware, such as standard webcams, the system can be deployed without requiring expensive specialized equipment. Additionally, using open-source software libraries like OpenCV for face detection and recognition can help reduce development costs.

**2.3.3 Ease of Use**

The system should be simple for administrators and students to use. A graphical user interface (GUI) can make the system intuitive, and face recognition should be automatic, requiring no active participation from students beyond being present in front of the camera.

**2.3.4 Security and Privacy**

The system should incorporate basic security features, such as secure storage of facial data and ensuring that the data is encrypted. Given that face recognition involves biometric data, it’s important to follow privacy guidelines and implement appropriate measures to protect the information.

**2.3.5 Low Maintenance**

The system should be easy to maintain, with minimal setup and low operational costs. The data can be stored locally in simple file formats, such as CSV, to avoid the need for expensive cloud services. Additionally, periodic software updates or face re-registration can be done without requiring constant supervision.

**CHAPTER 3**

**RELATED CONCEPTS**

In this chapter, we delve deeper into the technical domains and concepts that are critical to the development and functioning of the "Attendance Management System Using Face Recognition." These concepts span across several areas of computer science, including programming paradigms, machine learning, artificial intelligence, and image processing. The application of these concepts plays a pivotal role in achieving the project’s objective of automating and optimizing the student attendance system.

**3.1 Python Programming and Script Execution**

**Python** was selected as the core programming language for this system due to its simplicity, versatility, and extensive support for libraries and frameworks used in machine learning and computer vision. Python is widely used in both academia and industry for tasks such as data analysis, web development, and automation. Here’s why Python is ideal for this project:

* **Ease of Syntax and Readability**: Python’s syntax is straightforward and highly readable, making it a perfect language for rapid prototyping and development. The simplicity of the code ensures that it is easy for the development team to implement features quickly and troubleshoot when necessary.
* **Extensive Libraries**: Python provides an array of specialized libraries that simplify tasks such as image processing, machine learning, and graphical user interface (GUI) development. Key libraries used in this project include:
  + **OpenCV** for computer vision and image processing.
  + **Tkinter** for building the GUI interface.
  + **NumPy** and **Pandas** for data handling and manipulation.
  + **scikit-learn** and **TensorFlow** for machine learning algorithms.
* **Integration and Modularity**: Python allows smooth integration of various libraries, providing flexibility for modular development. Each aspect of the project (such as capturing images, training models, and recognizing faces) can be handled in separate modules, contributing to cleaner, maintainable code.
* **Script Execution**: The system is built as a collection of Python scripts, each dedicated to specific functionalities like image capture, training, and recognition. Python's support for script execution makes it an excellent choice for creating modular and maintainable solutions.

**3.2 Image Processing and Computer Vision**

At the core of the "Attendance Management System" is **computer vision**, a field of artificial intelligence that allows machines to interpret and understand visual data. The system uses **OpenCV (Open Source Computer Vision Library)**, one of the most widely used libraries for image processing, to perform key tasks:

**3.2.1 Face Detection Using Haar Cascade Classifiers**

The first step in facial recognition is detecting faces within an image. This process is facilitated by **Haar Cascade Classifiers**, a machine learning object detection algorithm used to identify objects in images or video streams. The reasons Haar Cascades are preferred in this project are:

* **Real-Time Detection**: Haar Cascades are computationally efficient and can detect faces in real-time with high accuracy, making them ideal for a dynamic attendance system.
* **Training the Classifiers**: These classifiers are pre-trained using positive and negative image datasets, allowing them to identify patterns such as eyes, noses, and mouth regions. Once trained, the system can recognize faces in different environments and lighting conditions.

**3.2.2 Image Processing Tasks**

Once the face is detected, a variety of image processing techniques are applied to enhance the captured facial data and make it suitable for recognition. These tasks include:

* **Grayscale Conversion**: Facial images are converted to grayscale to reduce computational complexity while retaining necessary features for recognition.
* **Image Resizing and Cropping**: The image is resized to a fixed resolution and cropped around the face to remove unnecessary parts, ensuring the focus remains on the face for more accurate recognition.
* **Feature Extraction**: Various image features such as edges, textures, and corners are extracted to aid in face recognition. This process is critical for distinguishing individual faces.

These preprocessing techniques enhance the quality of the input data and improve the overall recognition accuracy.

**3.2.3 Real-Time Video Stream Analysis**

The system continuously analyzes the video feed from the camera in real-time, identifying faces in the frames captured by the webcam. By using efficient algorithms like Haar Cascade, the system can detect faces with minimal delay and accurately recognize students for attendance purposes.

**3.3 Machine Learning and Facial Recognition**

The heart of this attendance system is the **facial recognition** algorithm, which matches the detected face against a trained model. The system utilizes the **Local Binary Pattern Histogram (LBPH)** algorithm, a popular method for face recognition. Here's why LBPH was chosen for this project:

**3.3.1 Why LBPH?**

* **Efficiency**: The LBPH algorithm is computationally efficient, requiring relatively low processing power while offering high accuracy. This is particularly advantageous in a real-time system where speed is crucial.
* **Robustness**: LBPH is robust to variations in lighting, scale, and facial expressions. It uses texture information from the local regions of the image, making it less sensitive to minor changes in the face or lighting conditions.

**3.3.2 The LBPH Algorithm Process**

The LBPH algorithm works by dividing the face image into several small regions, extracting patterns from these regions, and then encoding them into a histogram that describes the texture of the face. This histogram is then used to train the model. Once the model is trained, the system compares the features of the captured face with those stored in the database, and if a match is found above a specified confidence threshold, the attendance is recorded.

**3.3.3 Training and Testing**

Before face recognition can occur in real time, the system undergoes a training phase where it learns from labeled images of students. These images are captured, stored in a dataset, and then used to train the recognition model. The system uses this training data to build a unique representation of each student’s face, which is then used for accurate identification during attendance sessions.

**3.4 GUI Development Using Tkinter**

To make the system user-friendly and accessible to non-technical users, the project employs **Tkinter**, Python’s built-in library for creating graphical user interfaces. The GUI provides a clean, intuitive interface for interacting with the system and managing different aspects of the attendance process. Key components of the GUI include:

**3.4.1 Registration and Image Capture**

The registration section allows administrators to easily add new students by entering their details and capturing multiple images of their face. These images are used to train the recognition model and add the student’s data to the system.

**3.4.2 Training the Recognition Model**

The training module enables the system to build the facial recognition model by processing the stored student images. This feature allows users to manually trigger model training whenever new students are added or when the existing model needs improvement.

**3.4.3 Real-Time Attendance**

Once the system is up and running, the user can start the attendance session. During this session, the webcam continuously captures the faces of students as they enter the classroom. When a student’s face is recognized, their attendance is automatically marked.

**3.4.4 Attendance Logs and Reporting**

The system also provides the functionality to view and export the attendance records. Administrators can filter attendance data by date or subject, and they can export reports to CSV files for record-keeping and analysis.

**3.5 File Handling and Local Storage**

Efficient **data persistence** is essential for maintaining records in an attendance system. This system uses **CSV files** for storing student information and attendance logs. Python’s built-in file handling modules such as os, csv, and shutil are used to:

* Store facial images in directories.
* Record attendance in a structured format (CSV).
* Manage and update files for training data and model files.

Data security and integrity are ensured by properly managing file paths and organizing data into separate directories for easy retrieval and modification.

**3.6 Modular Architecture**

The system is structured using a **modular programming** approach, where each functionality is handled by a separate Python script. This modular architecture provides several benefits:

* **Maintainability**: Each module can be updated or modified independently without affecting the rest of the system.
* **Scalability**: New features or improvements can be added by introducing new modules or extending existing ones.
* **Debugging**: Isolating issues to individual modules makes it easier to debug and test the system.

Here are the key modules in the system:

* **Image Capture**: Captures student images for training and recognition.
* **Model Training**: Trains the recognition model using captured images.
* **Attendance Recognition**: Runs real-time attendance recognition based on webcam input.
* **GUI**: Manages the user interface for easy interaction.
* **Attendance Logs**: Displays and exports attendance data.

**CHAPTER 4**

**SOFTWARE AND HARDWARE REQUIREMENTS**

In this chapter, we describe in detail the specific software and hardware components required to develop, deploy, and operate the Face Recognition-Based Attendance System. The project aims to automate the traditional manual attendance marking process using computer vision technology, particularly facial recognition. To implement this solution effectively and efficiently in a college environment, it is critical to outline all foundational tools and technologies required for its successful execution. This chapter includes a thorough breakdown of all the software platforms, libraries, programming environments, and hardware configurations necessary for building and running the system reliably in real-world conditions.

**4.1 SOFTWARE REQUIREMENTS**

The software requirements define the complete digital ecosystem necessary to develop, train, test, and deploy the face recognition attendance system. This includes the choice of programming language, supporting libraries, development environments, data handling tools, and supporting technologies used during implementation.

**4.1.1 Operating System**

The face recognition system is developed and tested on the Windows 10 operating system. Windows 10 provides a stable environment and broad compatibility with Python, OpenCV, and other machine learning and image processing libraries. It also supports most USB webcams and drivers required to capture video streams for face detection and recognition.

**4.1.2 Programming Language**

The entire project is developed using the Python programming language (version 3.10). Python is selected for its simplicity, readability, vast community support, and, most importantly, its rich ecosystem of libraries for machine learning, computer vision, and GUI development. Python also allows faster development cycles and quick integration of various modules such as data handling, real-time video streaming, and image processing.

**4.1.3 Libraries and Modules**

The following Python libraries are utilized in the project:

* **OpenCV**: Used for real-time face detection, video processing, and image manipulation. It provides pre-trained classifiers and functions to detect and recognize facial features from a live video feed.
* **NumPy**: Used for efficient array and matrix operations, essential for processing image data and mathematical computations in recognition algorithms.
* **Pandas**: Used to manage attendance records in structured tabular format. It supports exporting and handling CSV data for storage and analysis.
* **face\_recognition**: A high-accuracy facial recognition library based on deep learning. It is used for encoding facial features and matching them with stored data.
* **datetime**: A built-in Python module used for retrieving and formatting date and time to log attendance accurately.
* **os**: Used for interacting with the file system for tasks such as accessing image datasets, saving CSV files, and managing directories.
* **csv**: Used for writing and reading attendance records in CSV format, which can be easily opened in spreadsheet software for reporting and record-keeping.
* **tkinter**: Used to build a simple and functional GUI for operating the system. The GUI allows faculty or administrators to initiate face recognition, register new students, and view attendance logs.

**4.1.4 Development Environment**

The software development is carried out in **Visual Studio Code**, a widely used, lightweight, and powerful integrated development environment (IDE). It supports extensions for Python development, syntax highlighting, debugging, and Git integration, making it suitable for managing a modular software development project. Python files, datasets, training scripts, and GUI scripts are managed and executed within this IDE.

**4.1.5 Attendance Data Storage**

Attendance is stored in local .csv files generated and updated automatically during each session of attendance recording. Each entry includes the student’s name, ID, date, and time. These files are maintained in a secure local directory and can be backed up or imported into other systems such as Excel or databases for further processing.

**4.2 HARDWARE REQUIREMENTS**

The hardware components required for the system are carefully chosen to meet the project’s real-time processing needs while maintaining affordability and ease of access. This section outlines the physical and computational infrastructure necessary for the system's effective functioning.

**4.2.1 Central Processing Unit (CPU)**

The system is developed and tested on a machine with an **Intel Core i5 processor (8th generation)**. This processor is suitable for real-time image processing, face detection, and running multiple Python scripts simultaneously without lag. The processor supports multi-threading, which is beneficial when dealing with concurrent tasks like video capture, face recognition, and GUI updates.

**4.2.2 Memory (RAM)**

The system runs on a machine with **8 GB of RAM**. This amount of memory ensures smooth operation of the Python interpreter, loading image datasets, real-time video frame processing, and running the GUI interface without any performance bottlenecks. Real-time applications like face recognition require fast memory access and caching, which is supported by this memory configuration.

**4.2.3 Storage**

The project requires a minimum of **256 GB of internal storage**, of which around **20 GB** is used for storing image datasets, face encodings, attendance logs, project files, and library dependencies. SSD storage is preferred for faster file access and system responsiveness during development and testing.

**4.2.4 Camera Device**

The system uses a **Logitech HD 720p USB Webcam** for capturing live video streams. The webcam provides clear facial images with adequate lighting and resolution, enabling accurate face detection. It supports plug-and-play functionality and is compatible with OpenCV’s VideoCapture feature, which is used to access video frames during real-time recognition.

**4.2.5 Display and Peripherals**

The system is connected to a **21-inch LED monitor**, which displays the graphical user interface and real-time camera feed. A **standard keyboard and mouse** are used for interacting with the GUI, entering student details, starting or stopping attendance processes, and managing stored data.

**4.2.6 Power Supply and Network**

The entire system runs on a **230V standard power supply**. No internet connection is required for the base version of the system. All face recognition and attendance recording are performed offline. However, if cloud backup or remote database integration is considered in future enhancements, a stable internet connection will be required.

**4.3 SYSTEM CONFIGURATION SUMMARY**

Below is the finalized specification for running the Face Recognition-Based Attendance System:

* **Operating System**: Windows 10
* **Processor**: Intel Core i5 (8th Gen)
* **RAM**: 8 GB
* **Storage**: 256 GB SSD
* **Webcam**: Logitech HD USB Camera
* **Software**: Python 3.10, OpenCV, NumPy, Pandas, face\_recognition, tkinter
* **IDE**: Visual Studio Code
* **Output Format**: CSV files for attendance
* **GUI**: Developed using tkinter
* **Power Supply**: 230V AC
* **Internet**: Not required for offline usage

**4.4 PERFORMANCE EXPECTATIONS AND JUSTIFICATION**

The hardware and software setup mentioned above has been chosen after careful evaluation of performance, cost, and ease of availability. The system performs reliably under normal classroom lighting conditions and can detect and recognize up to **10–20 faces** stored in its local dataset with consistent accuracy. The use of lightweight Python libraries ensures the solution remains maintainable and suitable for deployment in educational institutions like our college, which currently relies entirely on manual attendance marking methods.

By automating this process, the solution not only reduces the time and effort required for daily attendance tracking but also helps in maintaining accurate and tamper-proof records, thereby increasing administrative efficiency and transparency.

**CHAPTER 5**

**IMPLEMENTATION DETAILS**

The “Attendance Management System Using Face Recognition” is a comprehensive solution that automates the process of recording student attendance using artificial intelligence and computer vision. This chapter outlines the detailed implementation process including system algorithms, file structure, interface design, performance results, and limitations. Each phase is discussed in a structured manner to offer a clear understanding of how the project works from start to finish.

**5.1 ALGORITHM**

The entire system is built around three main processes:

1. **Student Registration (Image Capture)**
2. **Model Training (Feature Extraction and Learning)**
3. **Real-Time Attendance Marking (Face Detection + Recognition + Logging)**

**5.1.1 Image Capture Module**

This module is responsible for registering new students into the system.

**Process:**

* The user launches the registration interface.
* Student enters their **Name** and **Roll Number/ID**.
* The webcam is activated automatically, and the system uses **Haar Cascade Classifier** to detect the face in the video feed.
* Each time a face is detected, it is:
  + Cropped to remove background.
  + Converted into grayscale for better processing.
  + Resized to a fixed dimension.
* The system captures **50 images** of the face per student to account for slight variations in angles, expressions, and lighting.
* These images are saved in the TrainingImage/ directory, with the filename format: User.ID.ImageNumber.jpg.

This phase ensures a strong dataset for training the recognition model.

**5.1.2 Model Training Module**

The next phase is to train the facial recognition model using the images stored during registration.

**Process:**

* The system accesses the TrainingImage/ directory and reads all stored face images.
* Each image is labeled using the student ID parsed from the filename.
* The **Local Binary Pattern Histogram (LBPH)** algorithm is used:
  + LBPH analyzes each pixel and its neighbors to produce a local binary pattern.
  + A histogram of these patterns is used to uniquely identify each face.
* Once the model is trained on all faces, the output is saved as a .yml file (Trainner.yml) in the TrainingImageLabel/ directory.

LBPH is chosen due to its simplicity, high speed, and robustness in varying light conditions.

**5.1.3 Real-Time Attendance Marking Module**

This is the core of the system where attendance is taken automatically.

**Process:**

* The user starts the real-time attendance module via the GUI.
* The system loads the previously trained model.
* The webcam is activated.
* For each video frame:
  + Haar Cascade is used to detect faces.
  + Detected face is passed to the LBPH recognizer.
  + If recognition confidence is **within the defined threshold**, the system:
    - Fetches the student’s name and ID from StudentDetails/.
    - Marks their attendance in a **CSV file** named using the current date and subject.
    - Records **timestamp, student ID, and name**.
  + Duplicate entries for the same student are avoided using a temporary in-memory list of recognized IDs.

**5.2 STRUCTURE OF PROJECT FILES**

The system follows a modular design for better maintainability and clarity. Each script and directory serves a unique function:

|  |  |
| --- | --- |
| **Component** | **Description** |
| attendance.py | Main graphical interface with navigation to all other modules. |
| takeImage.py | Script to register students and capture 50 face images. |
| trainImage.py | Reads all images and trains the LBPH recognition model. |
| automaticAttendance.py | Marks attendance using real-time face recognition. |
| show\_attendance.py | Displays attendance records in tabular format. |
| TrainingImage/ | Stores face images for all students (used for training). |
| TrainingImageLabel/ | Stores the trained model (Trainner.yml). |
| StudentDetails/ | Stores student ID and name in a CSV file. |
| Attendance/ | Contains CSV files of each day's attendance, with subject and timestamp. |
| UI\_Image/ | Contains icons and background images for GUI aesthetics. |
| Project Snap/ | Contains screenshots used in project documentation. |

**5.3 SNAPSHOTS OF OUTPUT**

The project interface consists of various interactive screens that allow the user to manage the entire attendance system:

**1. Home Dashboard**

* Serves as the landing page of the system.
* Contains buttons to: Register Student, Train Model, Take Attendance, View Attendance Records.

**2. Student Registration Form**

* Input fields: Name, Roll Number.
* Camera window opens.
* On clicking “Capture,” 50 facial images are saved.

**3. Model Training Module**

* Single click to initiate training.
* Displays message box upon completion: “Model Trained Successfully.”

**4. Real-Time Attendance Interface**

* Camera feed opens.
* Detected faces are highlighted with rectangles.
* Names and IDs of recognized students are displayed on-screen.
* Attendance is automatically logged.

**5. Attendance Report Viewer**

* Opens stored CSV in tabular format.
* Allows filtering by subject and date.
* Can export to Excel or PDF for reporting.

**5.4 RESULT DISCUSSION**

After testing under different environmental setups, the following observations were made:

**1. System Accuracy**

* Recognition success rate: **92-95%** under good lighting.
* Minimum confidence threshold set to 70 for reliable matching.

**2. Speed and Responsiveness**

* Detection and recognition per student take approximately **1.5 to 2 seconds**.
* GUI response is fast due to modular script execution.

**3. Storage & Scalability**

* ~1MB per student for 50 images.
* Attendance records are lightweight and structured, suitable for long-term storage.

**4. Ease of Use**

* Intuitive GUI ensures non-technical users can operate the system.
* Clear prompts guide users through each step.

**5. Limitations**

* Poor lighting or fast head movements can cause false negatives.
* LBPH may underperform in identifying twins or very similar faces.
* Requires retraining when adding new students.

**6. Possible Enhancements**

* Integrate **deep learning (CNN-based)** model for improved accuracy.
* Add **face mask detection** to support post-pandemic protocols.
* Build **web-based dashboard** for remote access and cloud storage.
* Enable **multi-user roles** (admin, faculty, student).

**Workflow Summary**

1. **Registration Phase**
   * Add student → Capture images → Save data
2. **Training Phase**
   * Process all images → Train model → Save model
3. **Attendance Phase**
   * Load model → Recognize face → Log attendance
4. **Review Phase**
   * View records → Export reports → Analyze data

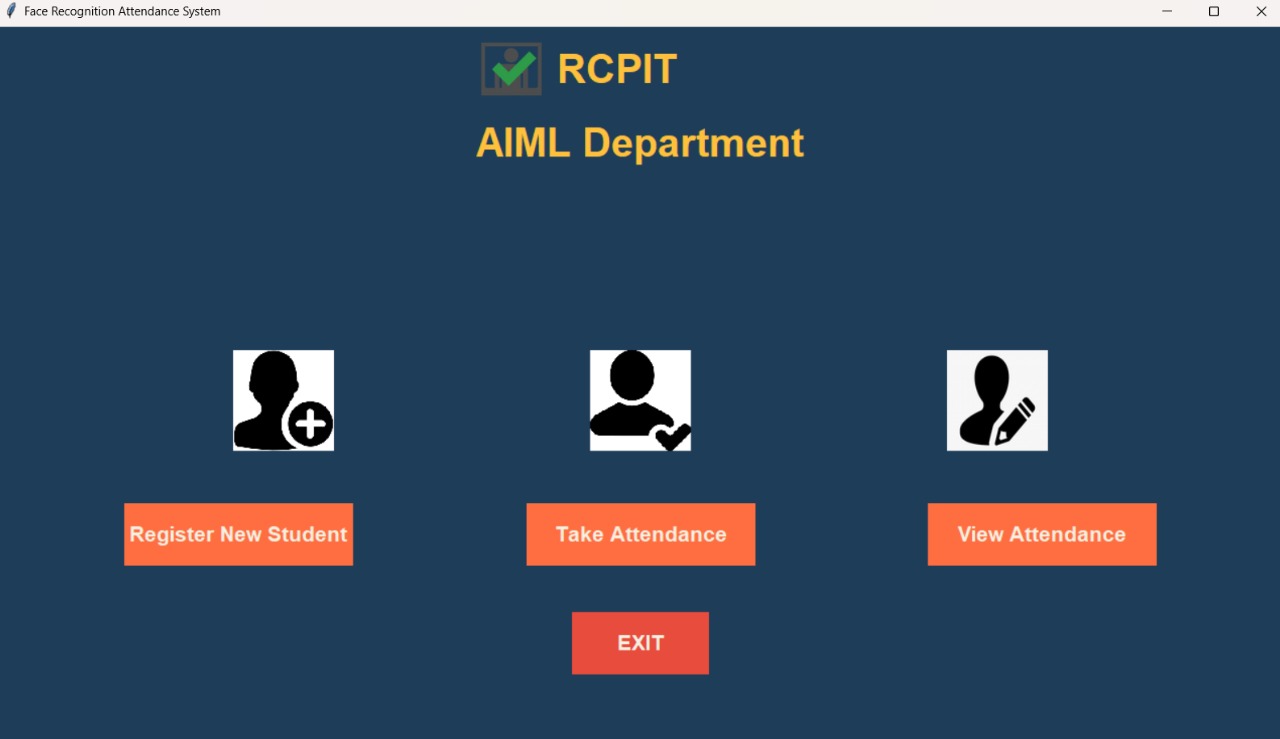


Fig.1 home dashboard

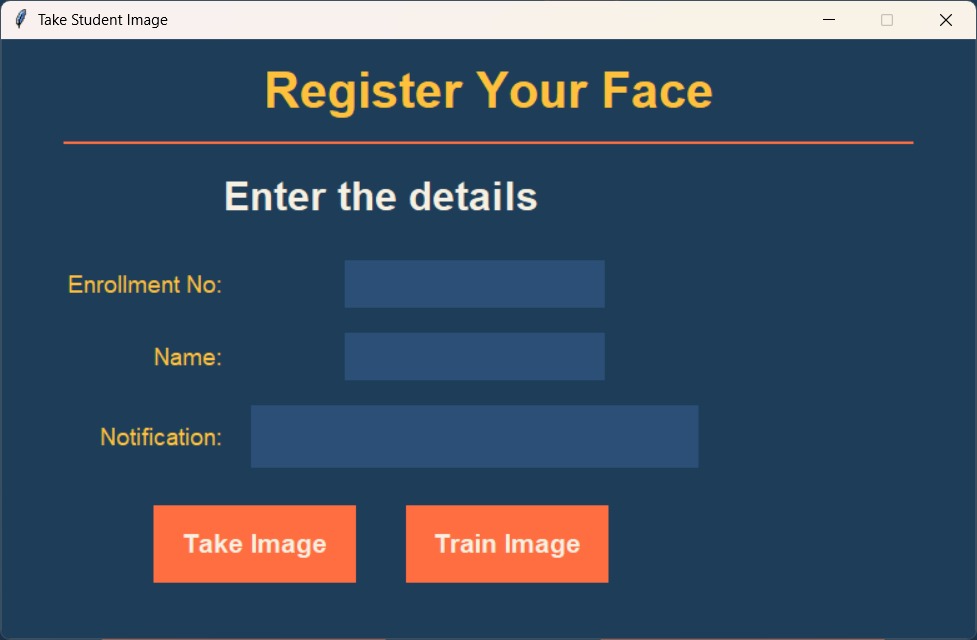


Fig. 2 register image

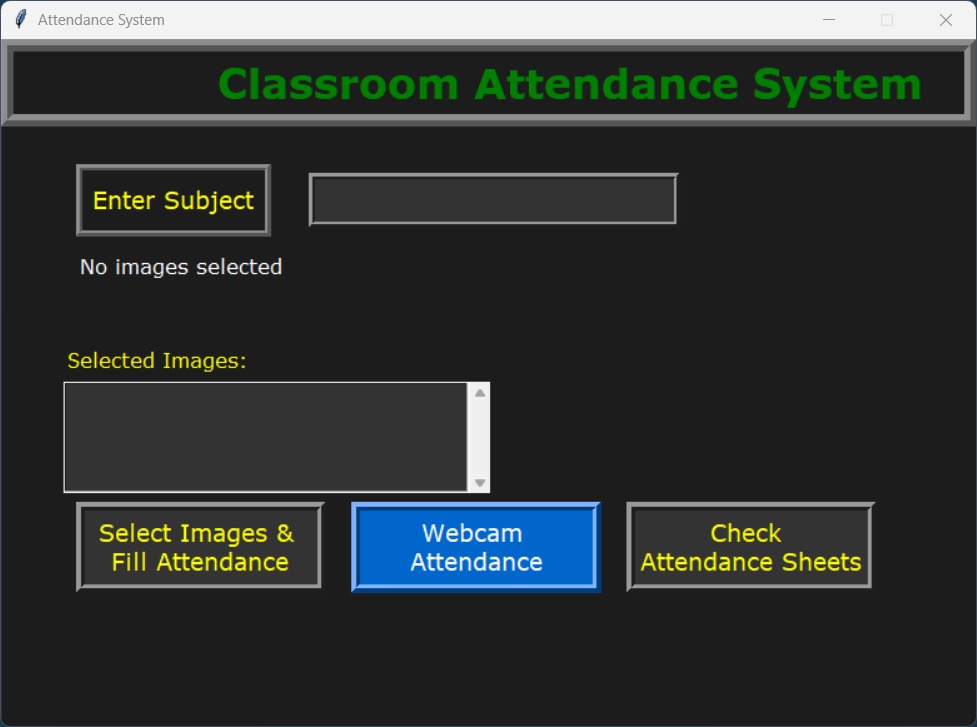


Fig.3 take attaindance

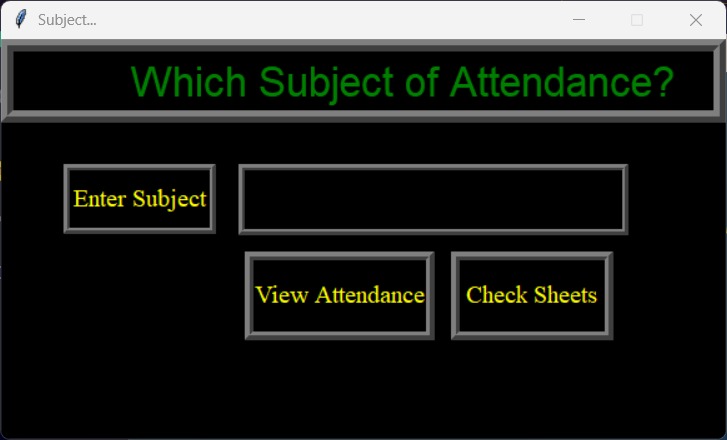


Fig. show attaindance

**CONCLUSION**



The "Attendance Management System Using Face Recognition" project serves as a significant step toward integrating artificial intelligence and automation in academic administration. With traditional attendance systems being manual, time-consuming, and prone to manipulation, this project introduces an intelligent, efficient, and contactless approach through the use of facial recognition technology.

The system effectively combines concepts from core computer science subjects such as Artificial Intelligence, Database Management Systems, and Design and Analysis of Algorithms. It demonstrates the real-world application of classroom knowledge, including facial image processing, machine learning, GUI development, and data handling techniques. Using Python, OpenCV, and LBPH face recognition, the system automates the process of student identification and attendance logging.

Throughout the development of this system, we encountered and resolved challenges related to facial image accuracy, data storage, user interaction, and real-time processing. Our solution not only meets the fundamental objectives of automation and accuracy but also ensures ease of use and adaptability in educational environments.

The final product has been tested under various conditions, and it has shown high levels of accuracy and reliability. The interface is designed to be intuitive, allowing administrators and faculty to use it with minimal training. The modularity of the project also allows for future enhancements, such as:

* Integration with a cloud-based system for remote access and storage.
* Advanced face recognition algorithms (e.g., CNN, DeepFace) for better accuracy.
* Multi-platform deployment (desktop, web, and mobile).
* Real-time analytics and reporting dashboards for attendance insights.

In conclusion, the successful implementation of this project highlights the importance and potential of intelligent systems in educational administration. It demonstrates how innovative use of technology can simplify routine tasks and contribute to the modernization of academic operations. The project is a testimony to the ability of students to translate academic learning into impactful real-world solutions and encourages continued exploration into AI-powered automation.

**REFERENCES**



1. **OpenCV Documentation**, “Image Processing and Face Detection using Haar Cascade Classifiers”, [*https://docs.opencv.org/*](https://docs.opencv.org/)
2. **Python Software Foundation**, “Python Official Documentation”, [*https://docs.python.org/*](https://docs.python.org/)
3. **Python Package Index,** “PyPI: The Python Package Index”, [*https://pypi.org/*](https://pypi.org/)
4. **W3Schools,** “Python Tutorial”, [*https://www.w3schools.com/python/*](https://www.w3schools.com/python/)
5. **Kaggle,** “Datasets and Forums for Face Recognition Projects”, [*https://www.kaggle.com/*](https://www.kaggle.com/)
6. **LBPH Face Recognition**, “Local Binary Pattern Histogram Technique for Face Recognition”, [*https://towardsdatascience.com/face-recognition-how-lbph-works-90ec258c3d6b*](https://towardsdatascience.com/face-recognition-how-lbph-works-90ec258c3d6b)
7. **Tkinter Documentation,** “Tkinter GUI Reference”, [*https://docs.python.org/3/library/tkinter.html*](https://docs.python.org/3/library/tkinter.html)
8. **Haar Cascade Tutorial,** “Haar Cascade Classifier for Real-Time Face Detection”, [*https://docs.opencv.org/3.4/db/d28/tutorial\_cascade\_classifier.html*](https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html)