Chapter 1

**INTRODUCTION**

* 1. Background

In a world where technology continues to reshape the way we interact with the environment around us, face recognition attendance systems have emerged as a pioneering solution to streamline attendance tracking processes. These systems blend cutting-edge facial recognition technology with data management software, offering a contactless and efficient way to manage attendance in various settings such as educational institutions, workplaces, and events. This article provides an overview of the evolution, components, benefits, challenges, and implications of face recognition attendance systems. The concept of using facial features for identification dates back several decades, but it wasn't until recent years that advancements in artificial intelligence and computer vision allowed for the development of reliable face recognition technology. With the availability of large datasets and improved deep learning algorithms, face recognition systems have gained remarkable accuracy and efficiency, paving the way for their integration into attendance management.

* 1. Problem Statement

In the realm of attendance tracking and management, traditional methods such as manual registers or card-based systems have proven to be error-prone, time-consuming, and susceptible to fraudulent activities. These inefficiencies not only result in inaccurate attendance records but also hinder the effective utilization of resources and administrative efforts. In light of these challenges, there is a pressing need for a reliable and modern solution that can revolutionize attendance tracking by leveraging the advancements in facial recognition technology and artificial intelligence. This problem statement seeks to explore the feasibility and effectiveness of integrating face recognition technology into attendance management systems. The objective is to develop a solution that not only overcomes the shortcomings of traditional methods but also addresses concerns related to data privacy, bias mitigation, hardware requirements, and overall system reliability. By examining these challenges and seeking innovative solutions, we aim to pave the way for a comprehensive face recognition attendance system that can redefine attendance tracking across various sectors.

* 1. Objective

The primary objective of implementing a face recognition attendance system is to revolutionize and optimize the process of tracking attendance in various environments, ranging from educational institutions to workplaces and events. This technology-driven solution aims to address the limitations and inefficiencies of traditional attendance tracking methods by harnessing the power of facial recognition technology and artificial intelligence.

Accuracy Enhancement: The foremost objective is to significantly improve the accuracy of attendance records. Face recognition systems can identify individuals with a high degree of precision, minimizing the chances of errors that are common in manual methods or card-based systems. This ensures that attendance data is reliable and trustworthy.

Efficiency and Time Savings: By automating the attendance tracking process, the system aims to save valuable time and resources for both administrators and attendees. Eliminating manual data entry and record-keeping reduces administrative workload and allows for more productive utilization of resources.

Contactless Operation: In a world increasingly concerned with hygiene and health, the objective is to provide a touch-free attendance solution. Attendees can simply walk in front of a camera, allowing for a seamless and contactless check-in process.

Real-time Monitoring: The system aims to enable real-time monitoring of attendance data. Administrators can access attendance records remotely and receive updates in real time, allowing for timely decision-making and intervention if necessary.

Fraud Prevention: One of the key goals is to deter attendance fraud and manipulation. Face recognition technology's biometric nature makes it difficult for individuals to impersonate others, enhancing the overall integrity of attendance records.

Security and Access Control: The objective is to enhance security by ensuring that only authorized individuals gain access to facilities or events. The face recognition system acts as an additional layer of authentication, reducing the risk of unauthorized entry.

Eco-friendly Solution: By reducing the need for paper-based registers and physical cards, the system contributes to environmental sustainability by minimizing paper usage and was Ease of Integration: The system aims to be seamlessly integrated into existing infrastructure. It should be compatible with various hardware setups, making adoption smooth for institutions and organizations.

Ethical Considerations: Ensuring privacy and obtaining informed consent from individuals are crucial objectives. The system aims to adhere to ethical standards and legal regulations regarding the collection and usage of biometric data.

Chapter 2

**LITERATURE SURVEY**

Facial recognition algorithms: The Eigenfaces algorithm is one of the earliest facial recognition algorithms and is based on Principal Component Analysis (PCA). Fisherfaces algorithm improves upon Eigenfaces by using Fisher Discriminant Analysis (FDA) to separate between different classes of faces. Local Binary Patterns (LBP) is another algorithm that uses texture information to recognize faces. Deep Learning-based approaches such as Convolutional Neural Networks (CNN) have shown promising results in facial recognition.

Face detection and tracking: Viola-Jones algorithm is a popular algorithm for face detection and tracking that uses Haar-like features and AdaBoost to classify faces. Histogram of Oriented Gradients (HOG) is another technique that uses gradient information to detect faces. Recently, CNN-based approaches have shown significant improvements in face detection and tracking.

Python libraries for facial recognition: OpenCV is a popular computer vision library that provides various functionalities for facial recognition, including face detection, recognition, and tracking. Dlib is another popular library that provides facial landmark detection and face recognition functionality. The face recognition library provides an easy-to-use API for face recognition and identification.

Performance evaluation: Accuracy, precision, recall, and F1 score are some of the commonly used metrics for evaluating the performance of facial recognition systems. The Labeled Faces in the Wild (LFW) dataset is a popular benchmark dataset for evaluating facial recognition systems.

Application scenarios: Facial recognition attendance systems are widely used in various industries, such as education, healthcare, and security. In education, facial recognition systems can help to automate attendance management and improve accuracy. In healthcare, facial recognition systems can be used to identify patients and ensure the correct medication is administered. In security, facial recognition systems can help to identify suspects and prevent crimes.

Ethical and privacy concerns: The use of facial recognition systems raises various ethical and privacy concerns, such as the potential for misuse, bias, and violation of privacy rights. It is essential to address these concerns and develop facial recognition systems that are transparent, fair, and respectful of privacy rights.

Overall, the literature survey provides a comprehensive understanding of the state-of-the-art in facial recognition attendance systems using Python and helps to identify the research gaps and potential areas for improvement

Chapter 3

**REQUIREMENT SPECIFICATION**

The main purpose for preparing this software is to give a general insight into the analysis and requirement of the existing system or situation and for determining the operating characteristics of the system.

## **3.1 General Description**

This project requires a computer system with the following software:

1. Operating System - Windows 7 or later (latest is best)
2. Python 3.7 (including all necessary libraries)
3. Microsoft Excel 2007 or later
4. Google Chrome (for cloud-related services)

**3.2 Functional Requirements**

In systems engineering and requirements engineering, functional requirements are the

requirements which are specified by a stakeholder or customer that define specific behavior or

functions that the system should incorporate. Functional requirements for the proposed system

are:

 User should be able to create a profile of his own.

 User should be able to edit and delete the student on his profile.

 User should be able to search for student through categories (author name, field Id).

 User should be able mark attendance.

 The system should be able to display student in a specified manner

**3.3 Non-Functional Requirements**

In systems engineering and requirements engineering, non-functional requirements are those

requirements that specifies criteria that can be used to judge the operation of a system. Non -

Functional requirements for the proposed system are:

 Performance

 Portability (able to run code on other devices too)

 Extensibility

 Reliability

 Compatibility

## **3.4 Specific Requirement**

This face recognition attendance system project requires OpenCV-a python library with a built-in LBPH face recognizer for training the dataset captured via the camera. Coming to the technical feasibility following are the requirements:

Hardware and Software Requirements:

Processor: Intel Processor IV (latest is recommended)

Ram: 4 GB (Higher would be good)

Hard disk: 40 GB

Monitor: RBG led

Keyboard: Basic 108-key keyboard

Mouse: Optical Mouse (or touchpad would work)

Camera: 1.5 Megapixels (Or more)

Chapter 4

**SYSTEM DESIGN**

The system design of a face recognition attendance system using Python and React involves the integration of advanced technologies to create a seamless and efficient solution. On the backend, Python serves as the core programming language, leveraging its powerful libraries and frameworks for facial recognition, data processing, and database management. The facial recognition algorithm, powered by deep learning frameworks like OpenCV and dlib, detects and extracts unique facial features, generating a distinct facial "signature" for each individual. This signature is then compared against a pre-existing database of authorized users' faceprints for accurate identification.

Python also facilitates the creation of a robust backend API using frameworks like Flask or Django, which handles data communication between the frontend and the backend. The API receives requests from the frontend, processes the facial recognition tasks, and communicates the results back to the React-based user interface.

On the frontend, React, a popular JavaScript library, is employed to design an intuitive and user-friendly interface for attendees and administrators. The frontend interface allows users to interact with the system, capture images via web cameras, and send the images to the backend for real-time face recognition. The React components display attendance records, real-time status updates, and alerts to administrators. Through this dynamic interaction between Python-based backend processing and React-based frontend presentation, the system achieves a seamless user experience and efficient attendance management.

The overall architecture leverages Python's extensive libraries, deep learning capabilities, and backend development frameworks in conjunction with React's responsive and interactive frontend design. The integration of these technologies forms a comprehensive face recognition attendance system that offers accuracy, real-time monitoring, and user-friendly interfaces for both attendees and administrators, ultimately streamlining attendance management processes.

4.1 System Architecture

A system architecture is a conceptual model that defines the structure, behavior, and views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system.

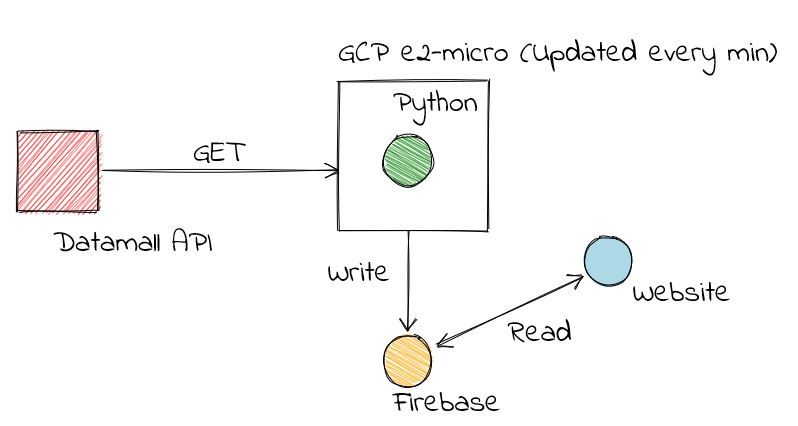


Figure 4.1.1

In the above figure, we can see that the system architecture works on the model Request Response. Firstly, we have React which is used to create various components under a similar category which in turn requests for the outline and data for its components. Python and fireFirebase as a backend in performing the service request. The Python in the execution of the command and the firebase helps in data fetching from the real-time database and storage bucket using the API service key configuration.

In the first step image is captured from the camera. There are illumination effects in the captured image because of different lighting conditions and some noise which is to be removed before going to the next steps. Histogram normalization is used for contrast enhancement in the spatial domain. Median filter is used for removal of noise in the image. There are other techniques like FFT and low pass filter for noise removal and smoothing of the images, but median filter gives good results.

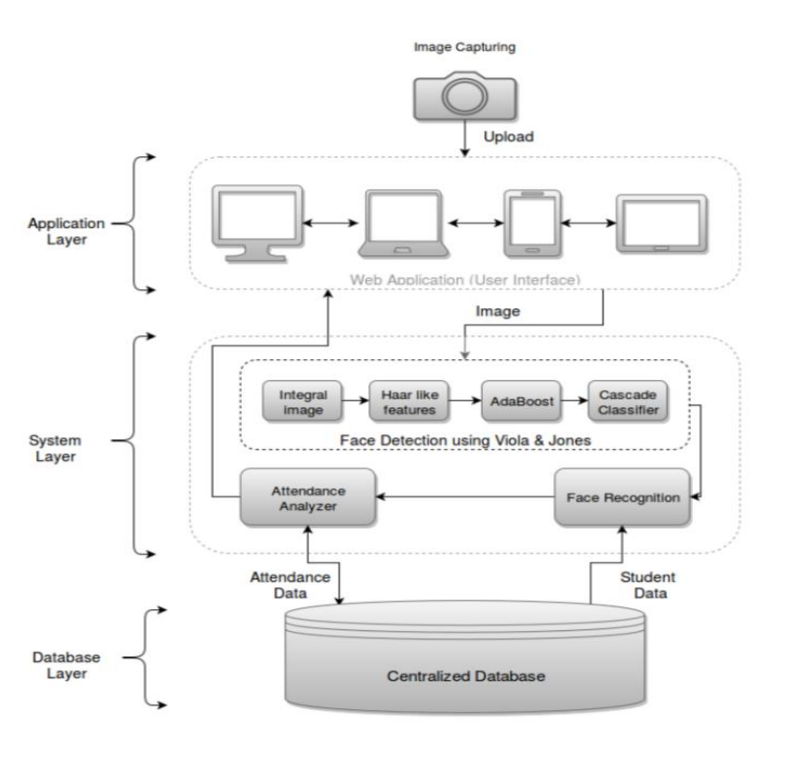
**DATABASE DESIGN**

Fig 4.1.2

**DESIGN NOTATION**

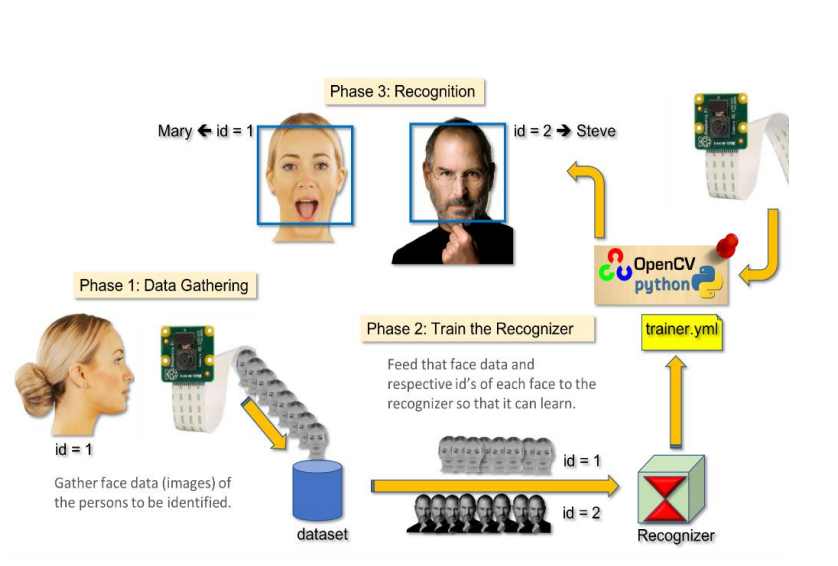
****

Fig 4.1.3

**DETAILED DESIGN**

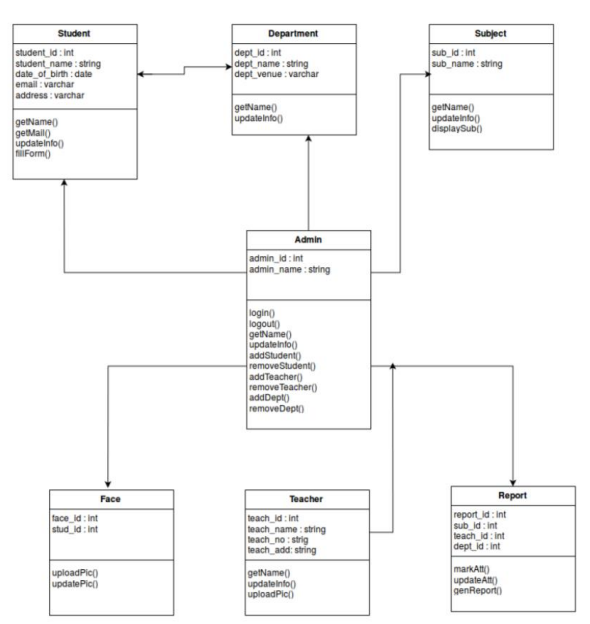
****

Fig 4.1.4

**4.2 Use Case Diagram**

Use case diagrams are used to gather the requirements of a system including internal and

external influences. These requirements are mostly design requirements. Hence, when a system

is analyzed to gather its functionalities, use cases are prepared and actors are identified. When

the initial task is complete, use case diagrams are modelled to present the outside view.

In brief, the purposes of use case diagrams can be said to be as follows −

 Used to gather the requirements of a system.

 Used to get an outside view of a system.

 Identify the external and internal factors influencing the system.

 Show the interaction among the requirements are actors.

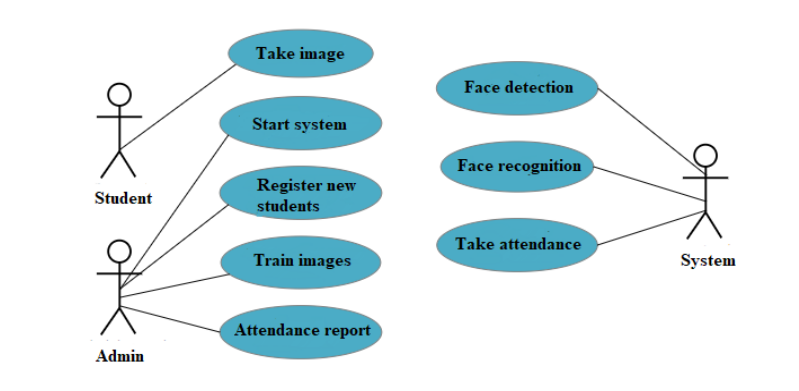


Figure 4.2.1

|  |  |
| --- | --- |
| UseCase | Creating a Profile |
| Actors | User |
| PreCondition | The application must be open on the web browser |
| Scenario | 1. User opens the application 2. Selects the login page 3. User tries to create a new profile by giving a username and password 4. User profile is created and start to take attendance |
| Post Condition | User should be logged in to his profile after creating it |

Table 4.2.1

|  |  |
| --- | --- |
| UseCase | Logging into the Profile |
| Actors | User |
| PreCondition | The application should be open on the web browser |
| Scenario | 1. Go to login page 2. Enter credentials 3. User should be redirected to his dashboard |
| Post Condition | User is logged into his profile |

Table 4.2.2

|  |  |
| --- | --- |
| UseCase | Adding and deleting the student |
| Actors | User |
| PreCondition | The application must be open on the web browser |
| Scenario | 1. User should be logged into his/her profile 2. User clicks on to add or delete the student 3. The added or deleted student is retrived on the dashboard when user click the fetch button |
| Post Condition | User actions should be stimulated |

Table 4.2.3

Chapter 5

**IMPLEMENTATION**

**5.1 Front End**

Front-end web development, also known as client-side development is the practice of producing HTML, CSS and JavaScript for a website or Web Application so that a user can see and interact with them directly. The challenge associated with front end development is that the tools and techniques used to create the front end of a website change constantly and so the developer needs to constantly be aware of how the field is developing.

**ReactJS**

ReactJS, also known as React or React.js, is an open-source JavaScript library for building user interfaces. It is used for handling view layer in single page applications and mobile applications development. It is maintained by Facebook, Instagram and a community of developers and corporations. React strives to provide speed, simplicity and scalability. Some of its most notable features are JSX, Stateful components, Virtual Document Object Model.

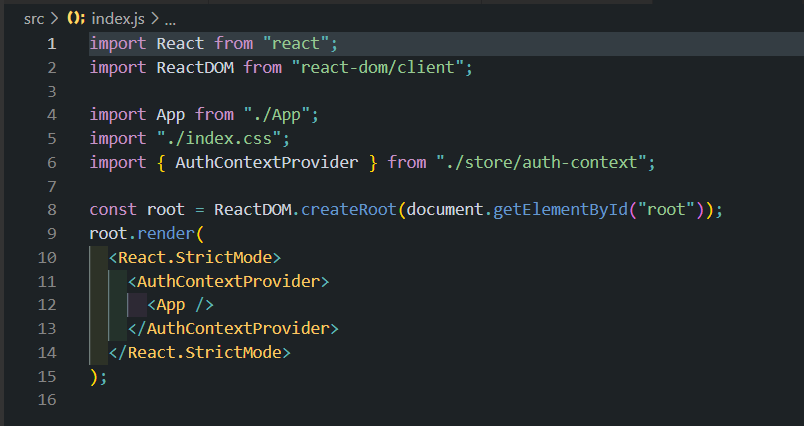


Figure 5.1.1

**JSX**

JavaScript XML (JSX) is an extension to the ECMAScript syntax without any defined

semantics. (JSX 2014) React embraces the fact that rendering logic is inherently coupled with

another UI logic. Instead of separating technologies, React uses loosely coupled units called

components that contain both. JSX is optional and not required to use React.

**Virtual Document Object**

Model The HTML DOM was originally intended for static pages and thus was not optimized

for creating dynamic UI. When the DOM updates, it has to update every node and re-paint the page with the corresponding CSS and layout. It is common for a single page application to contain thousands of dynamically generated nodes that have event listeners attached to them. In dynamic pages, the HTML DOM must check for changes in every node data at a regularinterval. This is considerably reducing application performance. The Virtual DOM was invented as a solution to this inefficiency. The Virtual DOM is an abstraction of the HTML DOM. It is lightweight and detached from the browser. It can be updated without affecting the actual DOM. React has Virtual DOM built in a module called ReactDOM. When updates are supplied, React uses a process called reconciliation, using an algorithm that compares and contrasts changes to know what elements needs updating. React then only change those elements, leaving the others unaffected.

**Stateful Components**

React allows users to split the UI into independent, reusable pieces called React Components. React components implement a render method that takes input data and returns what to display. Each component has several lifecycle methods that can be overridden to execute code at particular times during the process. Methods can be called using Reacts API. (React) State is a plain JavaScript object that is used to record and react to user events. Each class-based component defined has its own state object. Whenever a component state is changed, the component, and all of its child components, immediately re-renders. States hold values throughout the component and can be passed down to child components as props.

**States Hook**

The state is an instance of React Component Class can be defined as an object of a set of

observable properties that control the behavior of the component. In other words, the State of a component is an object that holds some information that may change over the lifetime of the component.

**Refs Hook**

Refs are a function provided by React to access the DOM element and the React element that you might have created on your own. They are used in cases where we want to change the value of a child component, without making use of props and all. They also provide us with good functionality as we can use call-backs with them. While building client-side apps, a team of Facebook developers realized that the DOM is slow (The Document Object Model (DOM) is an application programming interface (API) for HTML and XML documents. It defines the logical structure of documents and the way a document is accessed and manipulated.). So, to make it faster, React implements a virtual DOM that is basically a DOM tree representation in JavaScript. So, when it needs to read or write to the DOM, it will use the virtual representation of it. Then the virtual DOM will try to find the most efficient way to update the browser’s DOM. Unlike browser DOM elements, React elements are plain objects and are cheap to create. React DOM takes care of updating the DOM to match the React elements. The reason for this is that JavaScript is very fast and it’s worth keeping a DOM tree in it to speed up its manipulation. Although React was conceived to be used in the browser, because of its design it can also be used in the server with Node.js.

**useEffect Hook**

The motivation behind the introduction of useEffect Hook is to eliminate the side-effects of using class-based components. Tasks like updating the DOM, fetching data from API endpoints, setting up subscriptions or timers, etc can be led to unwanted side-effects. Since the render method is too quick to produce a side-effect one needs to use life cycle methods to observe the side effects. Consider updating the document title for a simple counter component to the current value. On the initial render, we set the current clicked value to 0 clicks. So, this section is coded into the componentDidMount() method which is executed only once in the component life cycle. Then we create a button to increment the count state value by one on every click. As the count value state changes, we also need to update the document title again and for that, we need to write the same piece of code in componentDidUpdate(). The componentDidupdate() method is perfect for updating the counter value at any time the state changes but the repetition of code is one of the side-effects. The useEffect() is used for causing side effects in functional components and it is also capable for handling componentDidMount(), componentDidUpdate() and componentWillUnmount() life-cycle methods of class based components into functional component.

More than two –third of automotive customers indicate that Service

Convenience is a determining in selecting a brand or purchasing from a Specific

dealership .Consequently, Customer Service and service management is of vital

relevance to ensure ongoing Customer loyalty and retention and ultimately,

dealer Profitability.

More than two –third of automotive customers indicate that Service

Convenience is a determining in selecting a brand or purchasing from a Specific

dealership .Consequently, Customer Service and service management is of vital

relevance to ensure ongoing Customer loyalty and retention and ultimately,

dealer Profitability.

More than two –third of automotive customers indicate that Service

Convenience is a determining in selecting a brand or purchasing from a Specific

dealership .Consequently, Customer Service and service management is of vital

relevance to ensure ongoing Customer loyalty and retention and ultimately,

dealer Profitability.

More than two –third of automotive customers indicate that Service

Convenience is a determining in selecting a brand or purchasing from a Specific

dealership .Consequently, Customer Service and service management is of vital

relevance to ensure ongoing Customer loyalty and retention and ultimately,

dealer Profitability.

More than two –third of automotive customers indicate that Service

Convenience is a determining in selecting a brand or purchasing from a Specific

dealership .Consequently, Customer Service and service management is of vital

relevance to ensure ongoing Customer loyalty and retention and ultimately,

dealer Profitability.

More than two –third of automotive customers indicate that Service

Convenience is a determining in selecting a brand or purchasing from a Specific

dealership .Consequently, Customer Service and service management is of vital

relevance to ensure ongoing Customer loyalty and retention and ultimately,

dealer Profitability.

More than two –third of automotive customers indicate that Service

Convenience is a determining in selecting a brand or purchasing from a Specific

dealership .Consequently, Customer Service and service management is of vital

relevance to ensure ongoing Customer loyalty and retention and ultimately,

dealer Profitability.

More than two –third of automotive customers indicate that Service

Convenience is a determining in selecting a brand or purchasing from a Specific

dealership .Consequently, Customer Service and service management is of vital

relevance to ensure ongoing Customer loyalty and retention and ultimately,

dealer Profitability.

More than two –third of automotive customers indicate that Service

Convenience is a determining in selecting a brand or purchasing from a Specific

dealership .Consequently, Customer Service and service management is of vital

relevance to ensure ongoing Customer loyalty and retention and ultimately,

dealer Profitability.

More than two –third of automotive customers indicate that Service

Convenience is a determining in selecting a brand or purchasing from a Specific

dealership .Consequently, Customer Service and service management is of vital

relevance to ensure ongoing Customer loyalty and retention and ultimately,

dealer Profitability.

More than two –third of automotive customers indicate that Service

Convenience is a determining in selecting a brand or purchasing from a Specific

dealership .Consequently, Customer Service and service management is of vital

relevance to ensure ongoing Customer loyalty and retention and ultimately,

dealer Profitability.

More than two –third of automotive customers indicate that Service

Convenience is a determining in selecting a brand or purchasing from a Specific

dealership .Consequently, Customer Service and service management is of vital

relevance to ensure ongoing Customer loyalty and retention and ultimately,

dealer Profitability.

More than two –third of automotive customers indicate that Service

Convenience is a determining in selecting a brand or purchasing from a Specific

dealership .Consequently, Customer Service and service management is of vital

relevance to ensure ongoing Customer loyalty and retention and ultimately,

dealer Profitability.

More than two –third of automotive customers indicate that Service

Convenience is a determining in selecting a brand or purchasing from a Specific

dealership .Consequently, Customer Service and service management is of vital

relevance to ensure ongoing Customer loyalty and retention and ultimately,

dealer Profitability.

**ReactDOM**

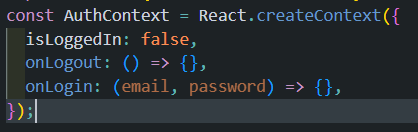
Before React, Developers directly manipulated the DOM elements which resulted in frequent DOM manipulation, and each time an update was made the browser had to recalculate and repaint the whole view according to the particular CSS of the page, which made the total process to consume a lot of time. As a betterment, React brought into the scene the virtual DOM. The Virtual DOM can be referred to as a copy of the actual DOM representation that is used to hold the updates made by the user and finally reflect it over to the original Browser DOM at once consuming much lesser time.

**Router**

React Router is a standard library for routing in React. It enables the navigation among views of various components in a React Application, allows changing the browser URL, and keeps the UI in sync with the URL. The main Components of React Router are: • BrowserRouter: BrowserRouter is a router implementation that uses the HTML5 history API (pushState, replaceState and the popstate event) to keep your UI in sync with the URL. It is the parent component that is used to store all of the other components. • Routes: It’s a new component introduced in the v6 and a upgrade of the component. The main advantages of Routes over Switch are: • Relative s and s • Routes are chosen based on the best match instead of being traversed in order. • Route: Route is the conditionally shown component that renders some UI when its path matches the current URL. • Link: Link component is used to create links to different routes and implement navigation around the application. It works like HTML

**Context API**

In a typical React application, data is passed top-down (parent to child) via props, but such usage can be cumbersome for certain types of props (e.g. locale preference, UI theme) that are required by many components within an application. Context provides a way to share values like these between components without having to explicitly pass a prop through every level of the tree.



5.2 Backend

Back-End refers to the server-side development. It focuses on databases, scripting, website architecture. It contains behind-the-scene activities that occur when performing any action on a website. It can be an account login or making a purchase from an online store. Code written by back-end developers helps browsers to communicate with database information.

**Capturing the dataset:** The first and foremost module of the project is capturing the dataset. When building an on-site face recognition system and when you must have physical access to a specific individual to assemble model pictures of their face, you must make a custom dataset. Such a framework would be necessary in organizations where individuals need to physically appear and attend regularly. To retrieve facial images and make a dataset, we may accompany them to an exceptional room where a camcorder is arranged to (1) distinguish the (x, y)- directions of their face in a video stream and (2) store the frames containing their face to database. We may even play out this process over a course of days or weeks in order to accumulate instances of their face in:

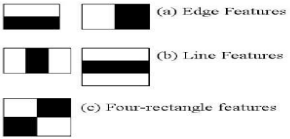
1. Distinctive lighting conditions
2. Times of day
3. Mind-sets and passionate states to make an increasingly differing set of pictures
4. illustrative of that specific individual's face

This Python script will:

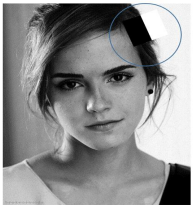
1. Access the camera of system
2. Detect faces
3. Write the frame containing the face to database

**Face detection using OpenCV:** In this project we have used OpenCV, to be precise, the Haar-cascade classifier for face detection. Haar Cascade is an AI object detection algorithm proposed by Paul Viola and Michael Jones in their paper "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001. It is an AI based methodology where a cascade function is prepared from a great deal of positive and negative pictures (where positive pictures are those where the item to be distinguished is available, negative is those where it isn't). It is then used to tell objects in different pictures. Fortunately, OpenCV offers pre-trained Haar cascade algorithms, sorted out into classifications (faces, eyes, etc.), based on the pictures they have been prepared on.

Presently let us perceive how this algorithm solidly functions. The possibility of Haar cascade is separating features from pictures utilizing a sort of 'filter', like the idea of the convolutional kernel. These filters are called Haar features and resemble that:

****

The idea is passing these filters on the picture, investigating one part at the time. At that point, for every window, all the pixel powers of, separately, white and dark parts are added. Ultimately, the value acquired by subtracting those two summations is the value of the feature extracted. In a perfect world, an extraordinary value of a feature implies it is pertinent. On the off chance that we consider the Edge (a) and apply it to the accompanying B&W pic:

****

We will get a noteworthy value, subsequently the algorithm will render an edge highlight with high likelihood. Obviously, the genuine intensities of pixels are never equivalent to white or dark, and we will frequently confront a comparable circumstance:



By and by, the thought continues as before: the higher the outcome (that is, the distinction among highly contrasting black and white summations), the higher the likelihood of that window of being a pertinent element.

**Training Database (LBPH algorithm):**

As it is one of the easier face recognitions algorithms, I think everyone can understand it without major difficulties.

**Introduction**: Local Binary Pattern (LBP) is a simple yet very efficient texture operator which labels the pixels of an image by thresholding the neighborhood of each pixel and considers the result as a binary number.

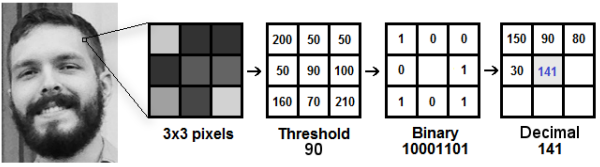
It was first described in 1994 (LBP) and has since been found to be a powerful feature for texture classification. It has further been determined that when LBP is combined with histograms of oriented gradients (HOG) descriptor, it improves the detection performance considerably on some datasets.

Using the LBP combined with histograms we can represent the face images with a simple data vector.

DETAILED EXPLANATION OF WORKING OF LBPH

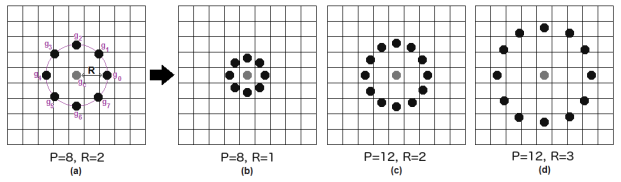
1. **Parameters:** the LBPH uses 4 parameters:
   * **Radius:** the radius is used to build the circular local binary pattern and represents the radius around the central pixel. It is usually set to 1.
   * **Neighbors:** the number of sample points to build the circular local binary pattern. Keep in mind: the more sample points you include, the higher the computational cost. It is usually set to 8.
   * **Grid X:** the number of cells in the horizontal direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. It is usually set to 8.
   * **Grid Y:** the number of cells in the vertical direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. It is usually set to 8.
2. **Training the Algorithm:** First, we need to train the algorithm. To do so, we need to use a dataset with the facial images of the people we want to recognize. We need to also set an ID (it may be a number or the name of the person) for each image, so the algorithm will use this information to recognize an input image and give you an output. Images of the same person must have the same ID. With the training set already constructed, let’s see the LBPH computational steps.
3. **Applying the LBP operation:** The first computational step of the LBPH is to create an intermediate image that describes the original image in a better way, by highlighting thefacial characteristics. To do so, the algorithm uses a concept of a sliding window, based on the parameter’s **radius** **and neighbors**.

The image below shows this procedure:

****

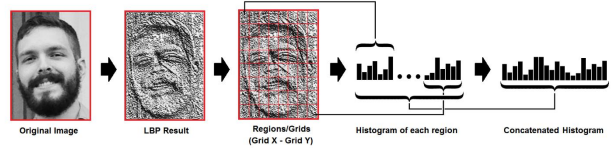
Based on the image, let’s break it into several small steps so we can understand it easily:

* Suppose we have a facial image in grayscale.
* We can get part of this image as a window of 3x3 pixels.
* It can also be represented as a 3x3 matrix containing the intensity of each pixel (0~255).
* Then, we need to take the central value of the matrix to be used as the threshold.
* This value will be used to define the new values from the 8 neighbors.
* For each neighbor of the central value (threshold), we set a new binary value. We set 1 for values equal or higher than the threshold and 0 for values lower than the threshold.
* Now, the matrix will contain only binary values (ignoring the central value). We need to concatenate each binary value from each position from the matrix line by line into a new binary value (e.g., 10001101). Note: some authors use other approaches to concatenate the binary values (e.g., clockwise direction), but the result will be the same.
* Then, we convert this binary value to a decimal value and set it to the central value of the matrix, which is a pixel from the original image.
* At the end of this procedure (LBP procedure), we have a new image which represents better the characteristics of the original image.
* Note: The LBP procedure was expanded to use a different number of radius and neighbors, it is called Circular LBP.



It can be done by using bilinear interpolation. If some data point is between the pixels, it uses the values from the 4 nearest pixels (2x2) to estimate the value of the new data point.

1. **Extracting the Histograms:** Now, using the image generated in the last step, we can use the Grid X and Grid Y parameters to divide the image into multiple grids, as can be seen in the following image:



Based on the image above, we can extract the histogram of each region as follows:

* As we have an image in grayscale, each histogram (from each grid) will contain only 256 positions (0~255) representing the occurrences of each pixel intensity.
* Then, we need to concatenate each histogram to create a new and bigger histogram. Supposing we have 8x8 grids, we will have 8x8x256=16.384 positions in the final histogram. The final histogram represents the characteristics of the image original image.

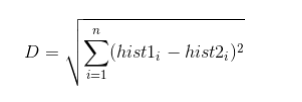
After detecting the face, image need to crop as it has only face nothing else focused. To do so Python Imaging Library (PIL) or also known as Pillow is used. PIL is a free library for python programming language that adds support for opening, manipulating, and saving many different images file formats. Capabilities of pillow:

* per-pixel manipulations,
* masking and transparency handling,
* image filtering, such as blurring, contouring, smoothing, or edge finding,
* image enhancing, such as sharpening, adjusting brightness, contrast or color,
* adding text to images and much more.

**Face recognition**

In this step, the algorithm is already trained. Each histogram created is used to represent each image from the training dataset. So, given an input image, we perform the steps again for this new image and creates a histogram which represents the image.

* So, to find the image that matches the input image we just need to compare two histograms and return the image with the closest histogram.
* We can use various approaches to compare the histograms (calculate the distance between two histograms), for example: Euclidean distance, chi-square, absolute value, etc. In this example, we can use the Euclidean distance (which is quite known) based on the following formula:



* So, the algorithm output is the ID from the image with the closest histogram. The algorithm should also return the calculated distance, which can be used as a ‘confidence’ measurement.
* We can then use a threshold and the ‘confidence’ to automatically estimate if the algorithm has correctly recognized the image. We can assume that the algorithm has successfully recognized if the confidence is lower than the threshold defined.

**Marking the attendance**

The face(s) that have been recognized are marked as present into the database. Then the entire attendance data is written into an excel sheet that is dynamically created using pywin32 library of python. First, an instance is created for excel application and then new excel workbook is created and active worksheet of that file is fetched. Then data is fetched from database and written in the excel sheet.

**FireBase for storage**

Firebase Realtime Database and Cloud Storage are powerful tools provided by Google's Firebase platform that can be used to efficiently store and manage various types of data for your applications. In your case, you are using these services to store student and teacher data in the Realtime Database and attendance records along with student photos in the Cloud Storage bucket.

The Firebase Realtime Database is a NoSQL cloud-hosted database that allows you to store and synchronize data in real-time. It's designed to handle real-time updates, making it ideal for applications that require collaborative or dynamic data. You've chosen to use it to store student and teacher data, which may include information such as names, IDs, courses, and contact details.

Firebase Cloud Storage is a scalable object storage service that allows you to store and retrieve user-generated content such as photos, videos, and files. I've chosen to use it to store attendance CSV files and student photos.

**Chapter 6**

**SYSTEM TESTING**

System Testing is a type of software testing that is performed on a complete integrated system to evaluate the compliance of the system with the corresponding requirements. In system testing, integration testing passed components are taken as input. System testing detects defects within both the integrated units and the whole system. The result of system testing is the observed behavior of a component or a system when it is tested. System Testing is carried out on the whole system in the context of either system requirement specifications or functional requirement specifications or in the context of both. System testing tests the design and behavior of the system and also the expectations of the customer. System Testing is basically performed by a testing team that is independent of the development team that helps to test the quality of the system impartial.

**6.1 Types of System Testing**

**Performance Testing** - Performance Testing is a type of software testing that is carried out to test the speed, scalability, stability and reliability of the software product or application. In this project Speed is tested as per the time in which a request from the client side if fulfilled. Stability is measured on how the server is acting to the requests that are made in responding to it without crashing. Reliability is measured on how reliable is our project in case of a power failure whether it retrieves the data at the time of power failure or not

**Load Testing** - Load Testing is a type of software Testing which is carried out to determine the behavior of a system or software product under extreme load. As we deploy our project over a webserver load testing can be done by seeing on how many number of requests it can intake before crashing or failing to respond to the requests.

**Stress Testing** - Stress Testing is a type of software testing performed to check the robustness of the system under the varying loads.

**Scalability Testing** - Scalability Testing is a type of software testing which is carried out to check the performance of a software application or system in terms of its capability to scale up or scale down the number of user request load. We’ve created a website that can be according to requirements in the future, we can also add commenting on other’s blog and share it to what’s app or any other social media by clicking on share button. Towards the future out project can be Upscaled as well as downscaled to include less features to be able to deploy it over the web..

Chapter 7

**RESULTS**

**7.1 Frontend Snapshots**

**Home Page**

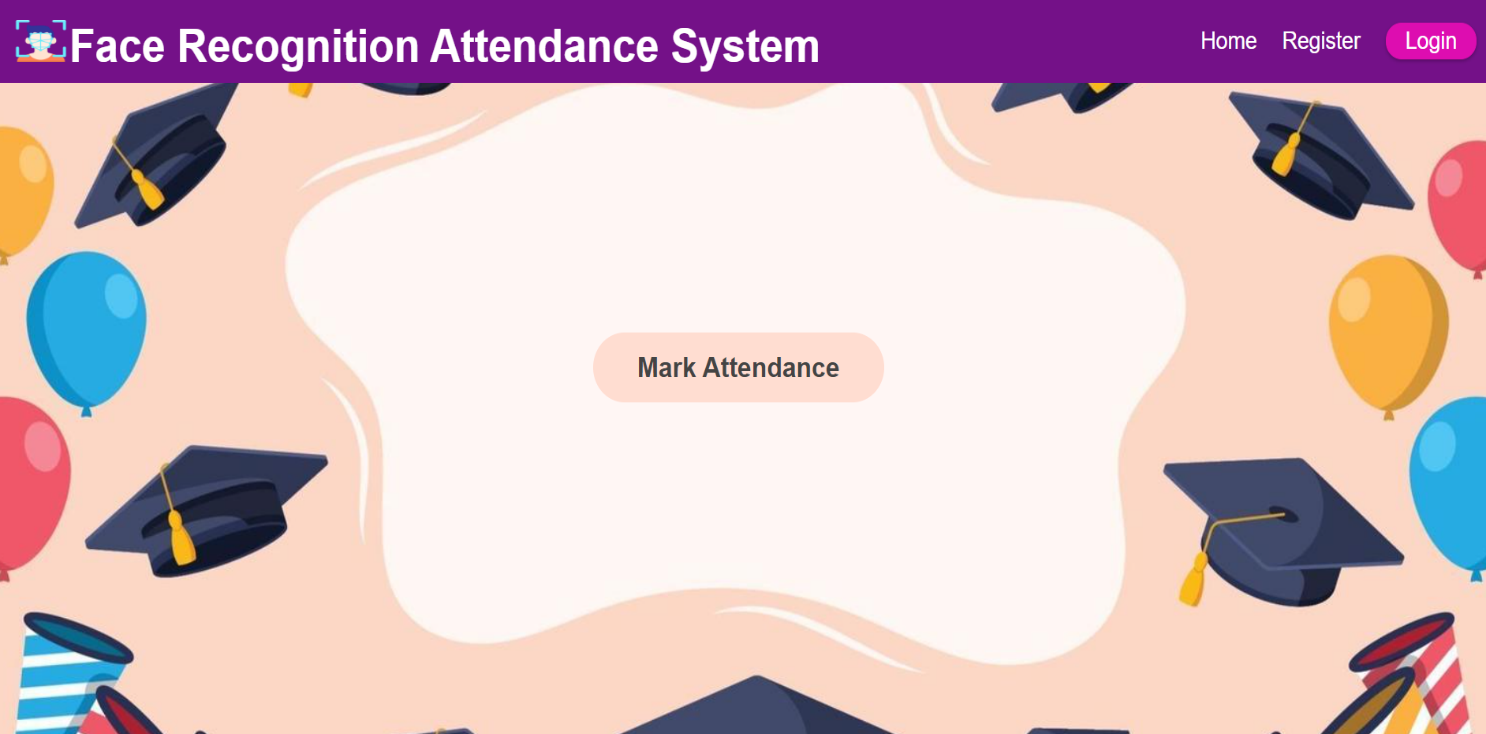
****

Fig 7.1.1

**Login Page**

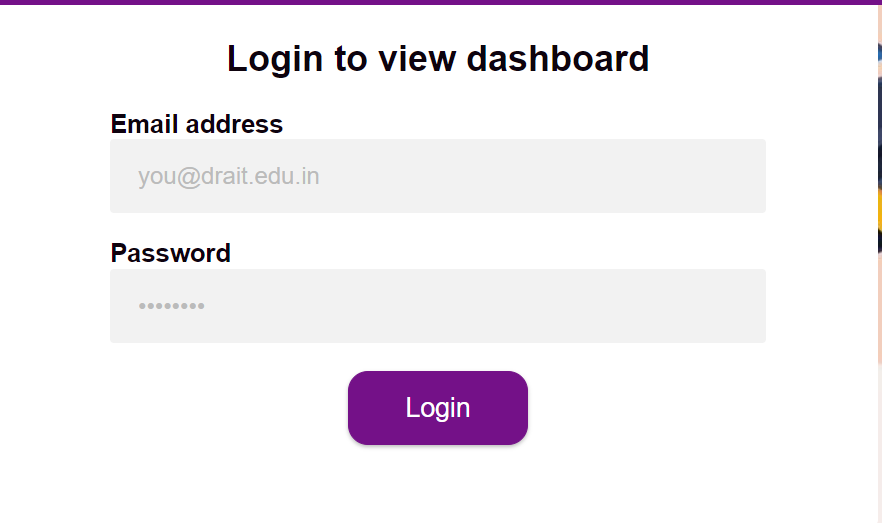
****

Fig 7.1.2

**Signup**

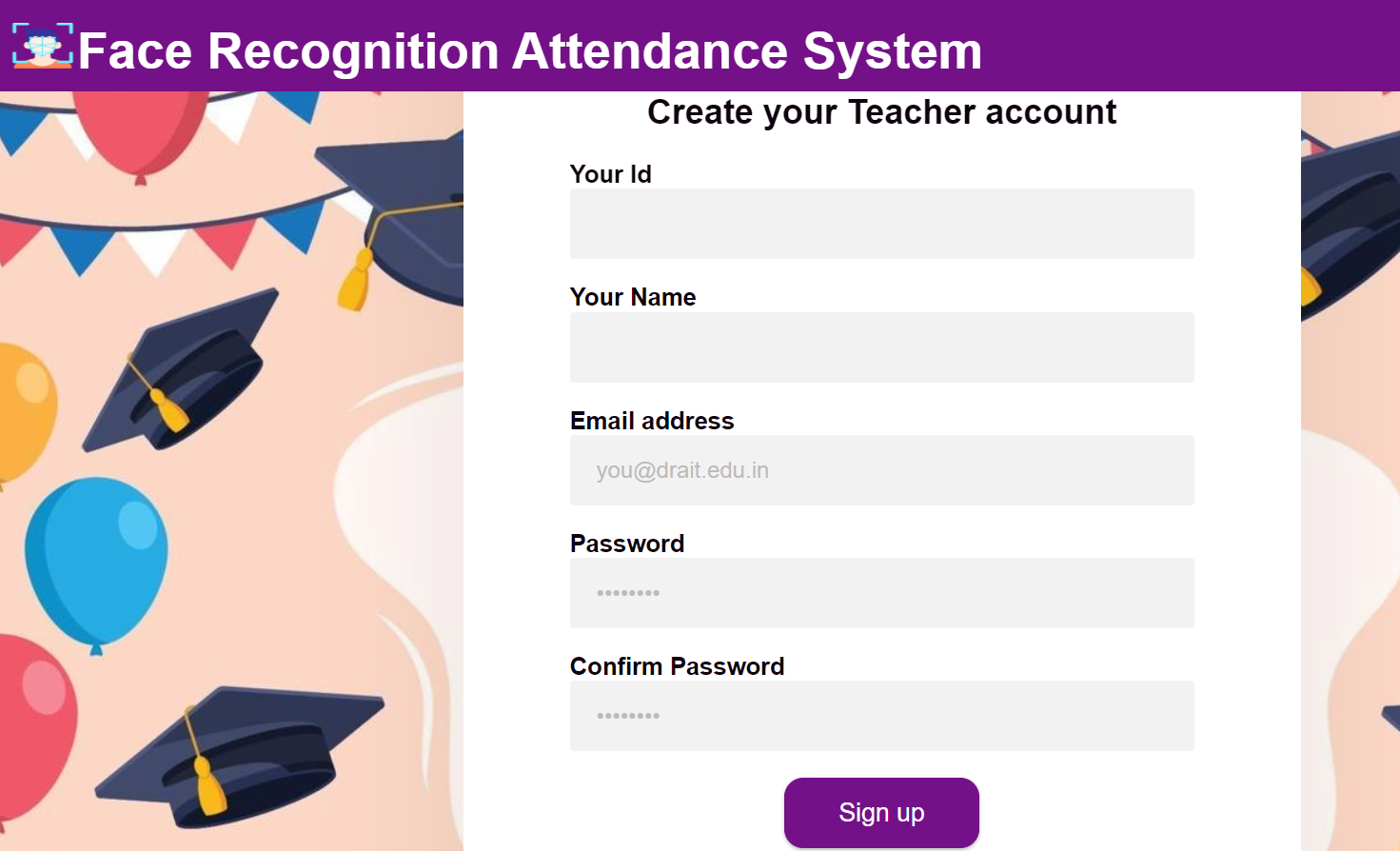
****

Fig 7.1.3

**DashBoard**

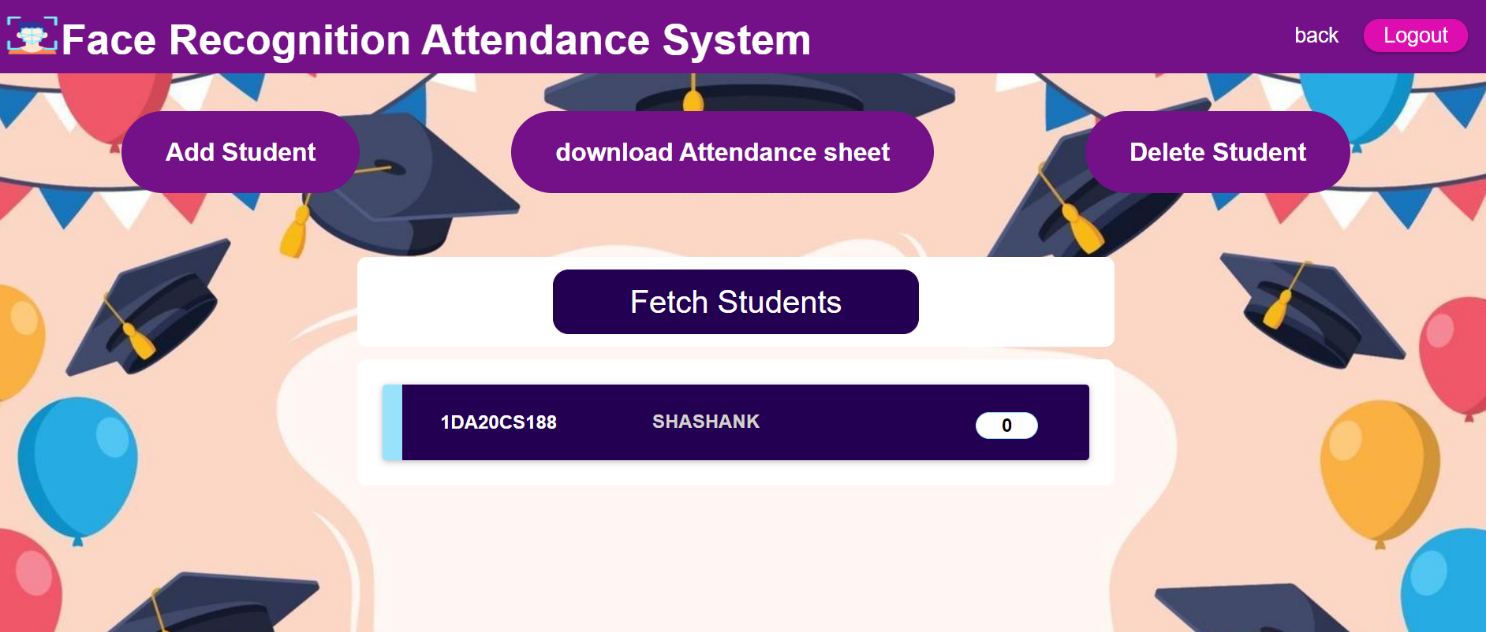
****

Fig 7.1.4

**Delete Student**

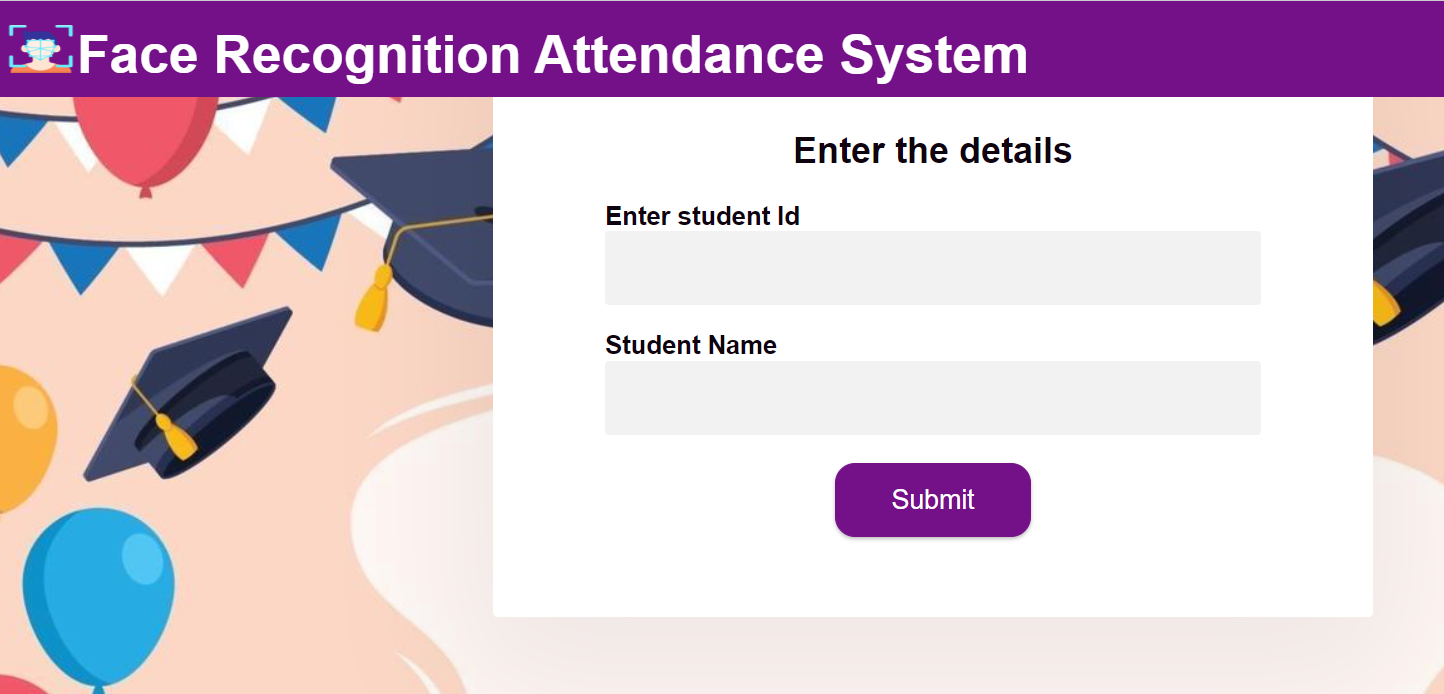
****

Fig 7.1.5

**Add Student**

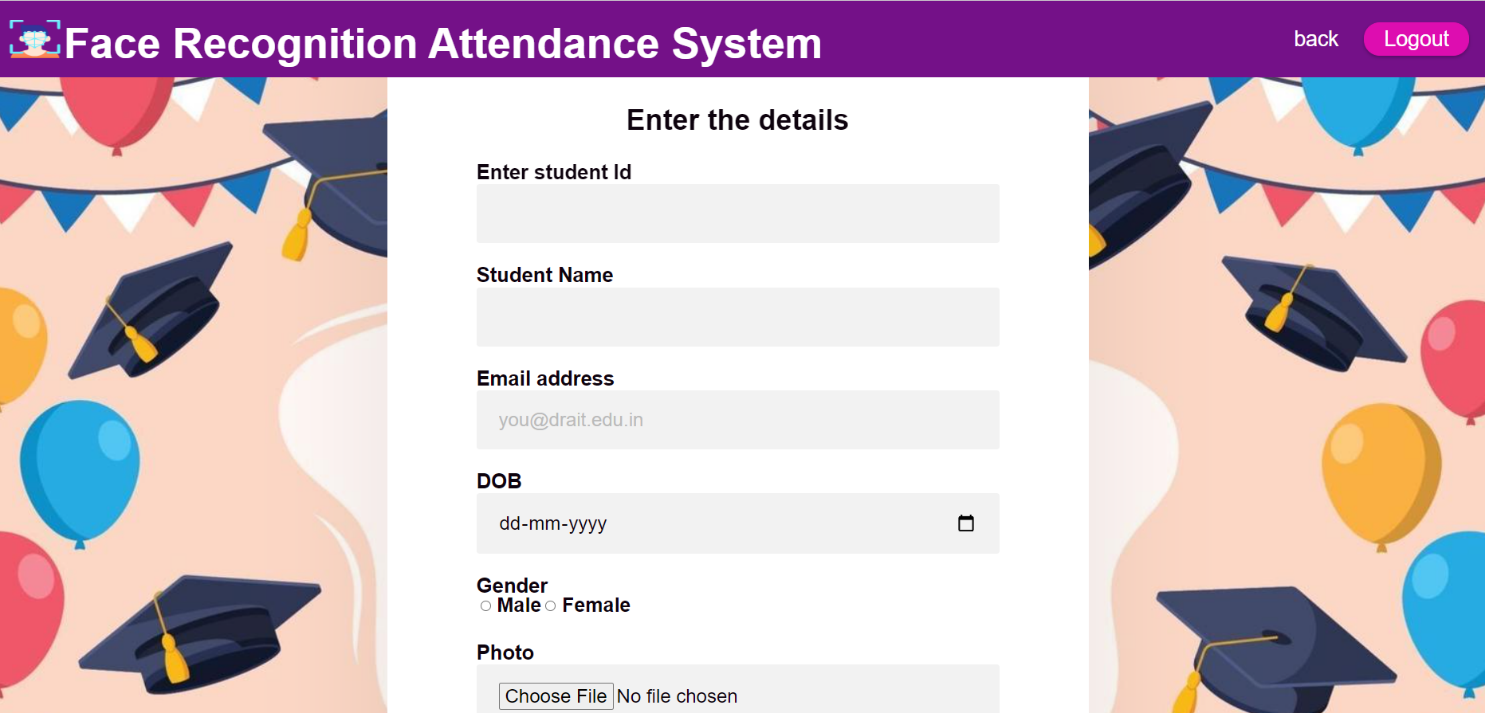
****

Fig 7.1.6

**7.2 Backend Snapshots**

**Firebase Storage**

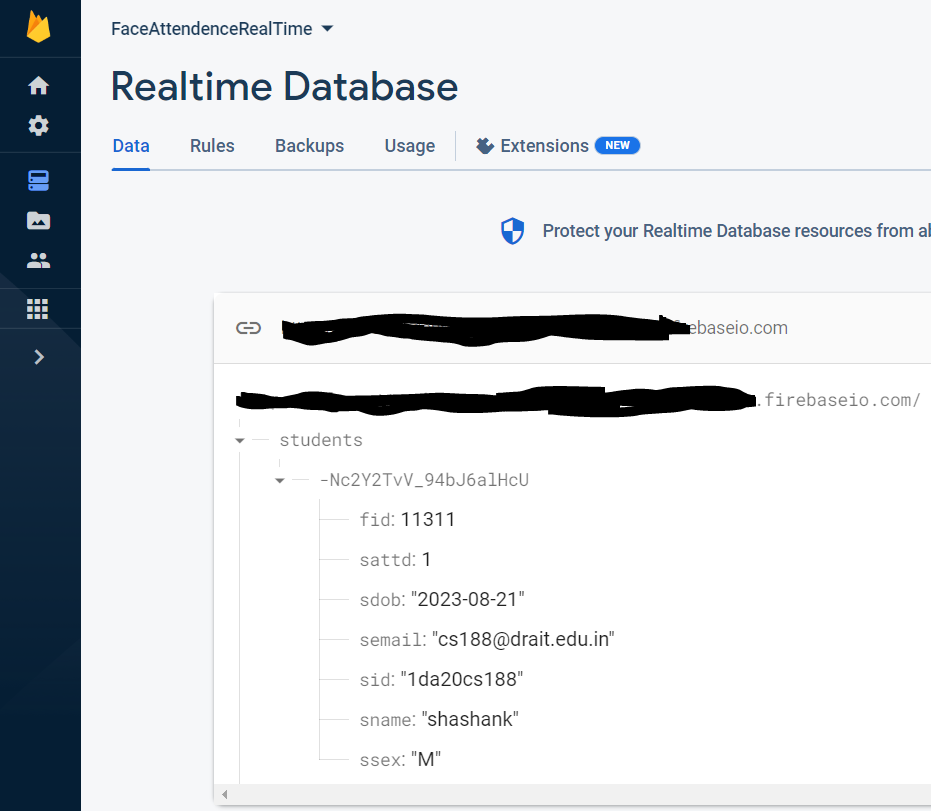
****

Fig 7.2.1

**Firebase Storage Bucket**

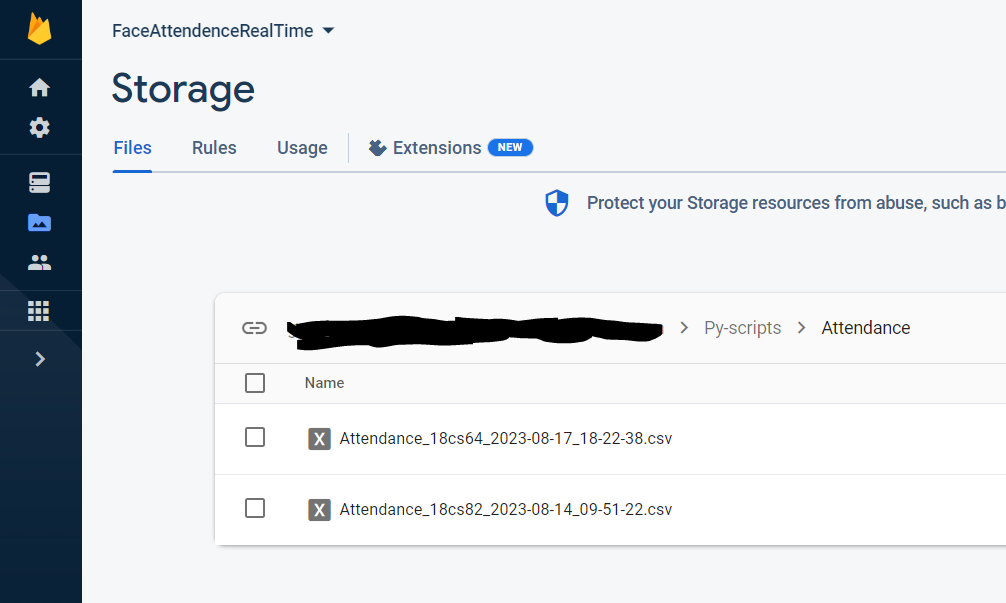
****

Fig 7.2.2

**Npm server**

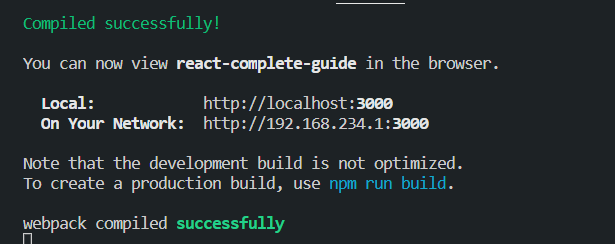


Fig 7.2.3

**Python flask server**

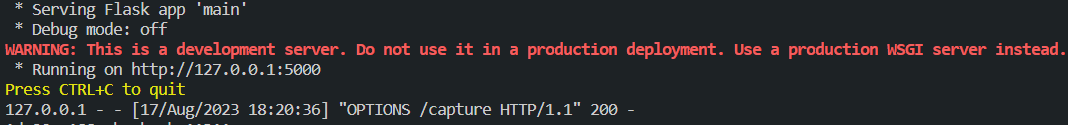


Fig 7.2.4

**7.3 Face Recognition Snapshots**

Recognize Face

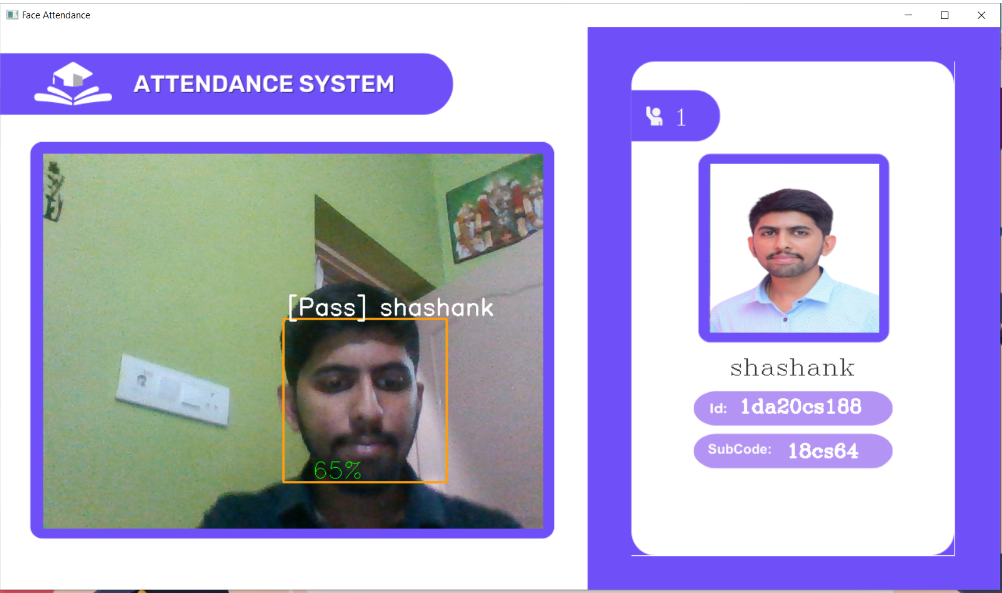


Fig 7.3.1

Check marked

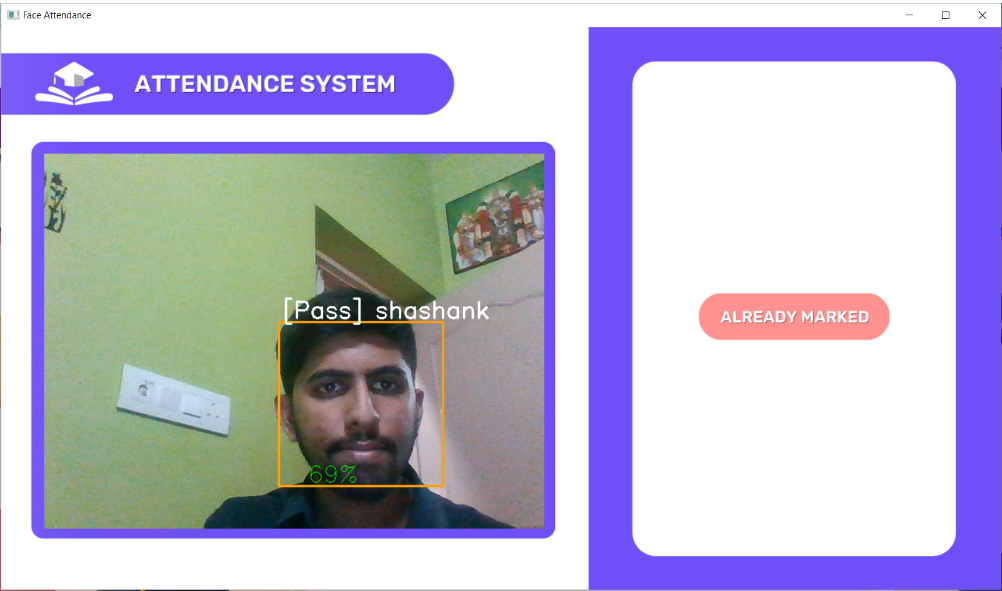


Fig 7.3.2

Done Training face

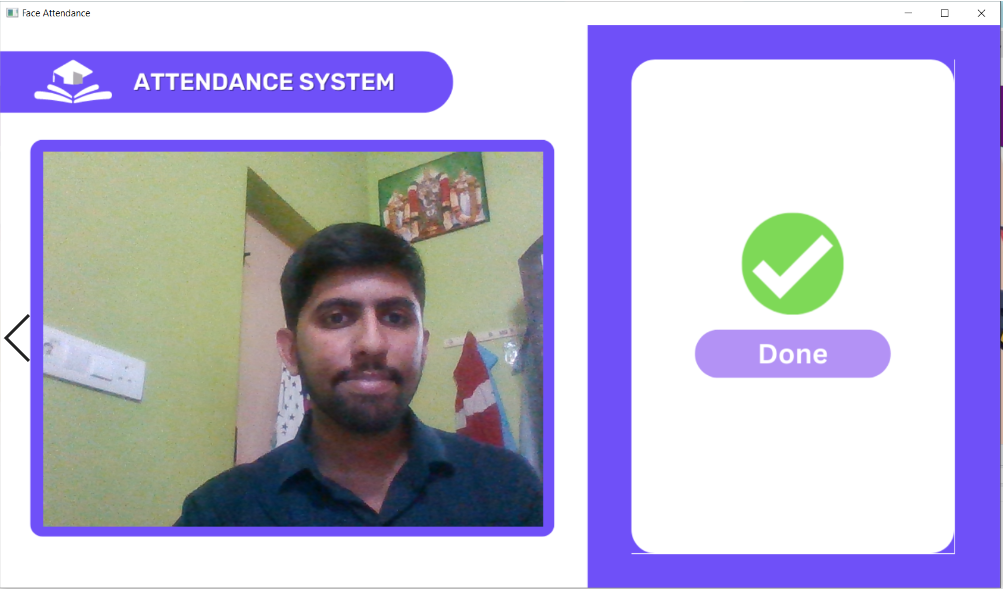


Fig 7.3.3

Mark Attendance

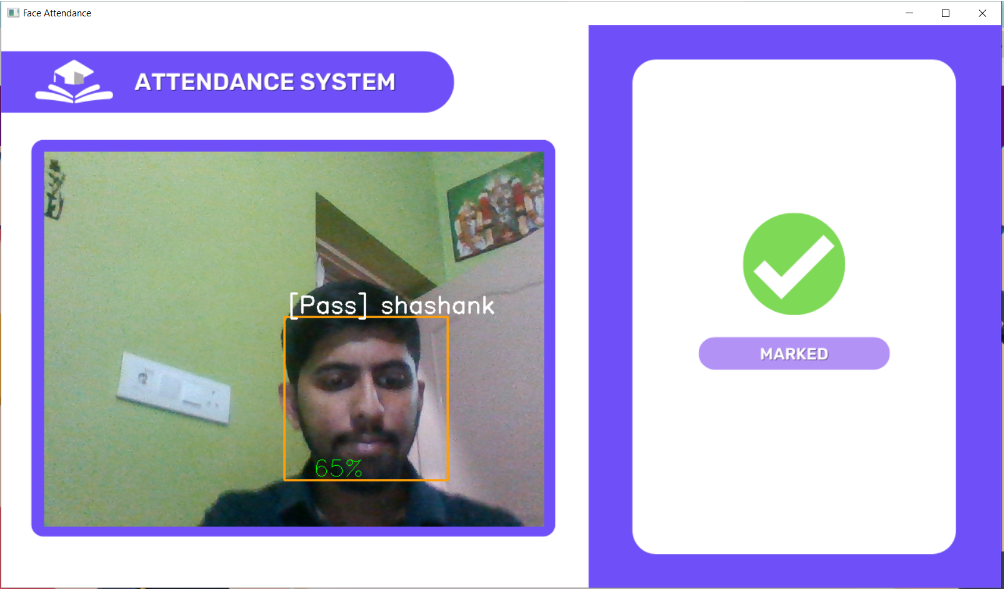


Fig 7.3.4

**APPLICATIONS**

Face recognition attendance systems have a wide range of applications across various industries due to their accuracy, convenience, and potential for reducing administrative work. Here are some of the key applications of face recognition attendance systems:

1. Corporate and Office Environments: Face recognition attendance systems can be used to automate employee attendance tracking in corporate offices. This helps in streamlining payroll processes and reducing time-consuming manual attendance management.

2. Educational Institutions: Schools, colleges, and universities can use face recognition attendance systems to monitor student attendance. This can improve accountability and ensure accurate record-keeping for both students and teachers.

3. Healthcare Facilities: Hospitals and clinics can implement face recognition systems to track the attendance of medical staff, ensuring that the right personnel are present during critical shifts. This can enhance patient care and overall operational efficiency.

4. Government Offices: Government agencies can utilize face recognition attendance systems to monitor employee attendance and maintain accurate records, ensuring efficient public service delivery.

5. Construction Sites: Face recognition attendance systems can help manage the attendance of construction workers, subcontractors, and site staff. This aids in better project management and compliance with safety regulations.

6. Retail Industry: Retail businesses can use face recognition attendance systems for employee time and attendance tracking in stores. This helps in efficient workforce management and scheduling.

7. Manufacturing Facilities: Face recognition systems can be integrated into manufacturing plants to monitor worker attendance, optimize production processes, and ensure a safe working environment.

8. Events and Conferences: Face recognition attendance systems can be employed to manage attendee registration and track participation at events, conferences, and seminars.

9. Gyms and Fitness Centers: Fitness clubs can use face recognition attendance systems to automate member check-ins, track attendance, and manage access to facilities.

10. Security and Access Control: Face recognition technology can enhance security by allowing access only to authorized personnel in restricted areas. This is particularly useful for organizations that require high levels of security.

**CONCLUSION**

In conclusion, face recognition attendance system in Python is a powerful technology that offers a reliable and efficient way to track attendance. It utilizes advanced algorithms to identify and verify an individual's identity through their facial features, making it a secure and accurate method for recording attendance.By implementing this system, organizations can save time and resources that would otherwise be spent on manual attendance tracking. Furthermore, it can also help to reduce errors and eliminate the possibility of attendance fraud.Python provides various libraries and tools that can be used to build a face recognition attendance system. Some of the popular libraries for face recognition include OpenCV, Dlib, and Face Recognition.

Overall, the face recognition attendance system in Python is a valuable tool that can help streamline attendance tracking processes and enhance security in various settings, including schools, offices, and other organizations.

**FUTURE ENHANCEMENT**

**Improved accuracy**

Facial recognition technology is already quite accurate, but there is always room for improvement. With the advancement of deep learning techniques and the availability of larger and more diverse datasets, facial recognition systems are likely to become even more accurate in the future.

**Better privacy protection**

As concerns about privacy and data security continue to grow, facial recognition technology will need to adapt to meet these challenges. Future enhancements may includestronger encryption, more secure storagemethods, and better control over who has access to facial recognition data.

**Increased speed and efficiency**

As facial recognition technology becomes more widespread, there will be a growing need for faster and more efficient systems. Future enhancements may include improvements in hardware and software that enable faster processing and analysis of facial data.

**Improved recognition in different environments**

Facial recognition technology can sometimes struggle in low-light conditions, or when faces are partially obscured by clothing or accessories. Future enhancements may include improvements in lighting, camera technology, and algorithms that can better recognize faces under a variety of conditions.

**REFERENCES**

* <https://www.instructables.com/id/Real-time-Face-Recognition-an-End-to-end-Project/>
* <https://pdfs.semanticscholar.org/b690/8248f52c917c8202e5bb1d39a19c0e018813.pdf>
* <https://towardsdatascience.com/face-recognition-for-beginners-a7a9bd5eb5c2>
* <https://www.tutorialspoint.com/>
* <https://www.pyimagesearch.com/2018/06/11/how-to-build-a-custom-face-recognition-dataset>
* <https://medium.com/dataseries/face-recognition-with-opencv-haar-cascade-a289b6ff042a>
* <https://towardsdatascience.com/face-recognition-how-lbph-works-90ec258c3d6b>
* <https://www.w3schools.com/>
* <https://firebase.google.com/>
* https://react.dev/