CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

Attendance being a very necessary side of administration may normally become an arduous, redundant activity, pushing itself to inaccuracies. The traditional approach of making roll calls proves itself to be a statute of limitations as it is very difficult to call names and maintain its record especially when the ratio of students is high. Every organization has its way of taking measures for the Attendance of students. Some organizations use document-oriented Approach and others have implemented these digital methods such as biometric fingerprinting techniques and card swapping techniques. however, these methods prove to be a statute of limitations as it subjects students to wait in a time-consuming queue. if the student fails to bring his id card then he will not be able to get attendance. evolving technologies have made many improvements in the changing world. The system of intelligent attendance is generally implemented with biometrics help. Recognition of face is one of the Biometric ways of improving this system. Face recognition proved to be a productive method for taking attendance.

CHAPTER 2
LITERATURE REVIEW

[2.1] AttenFace: A Real Time Attendance System Using Face Recognition

The current approach to marking attendance in colleges is tedious and time consuming. I propose AttenFace, a standalone system to analyze, track and grant attendance in real time using face recognition. Face recognition for each class is performed independently and in parallel, ensuring that the system scales with number of concurrent classes. The face recognition algorithm runs at 10 minute intervals on classroom snapshots, significantly reducing computation compared to direct processing of live camera feed. This method also provides students the flexibility to leave class for a short duration (such as for a phone call) without losing attendance for that class.

[2.2] Student Attendance System using Face Recognition

Face recognition is among the most productive image processing applications and has a pivotal role in the technical field. Recognition of the human face is an active issue for authentication purposes specifically in the context of attendance of students. Attendance system using face recognition is a procedure of recognizing students by using face biostatistics based on the high definition monitoring and other computer technologies. Attendance records can be easily manipulated by manual recording. This paper is therefore proposed to tackle all these problems.

[2.3] Face Recognition Based Smart Attendance System

Education institutes today are concerned about the consistency of students performance. One cause of this decrease in student performance is the inadequate attendance. There are several ways to mark your attendance, the most common ways to sign or call the students. It took longer and was problematic. From now on, a computer-based student attendance checking system is required that supports the faculty to keep records of attendance. We have used an intelligent attendance system based on face recognition in this project. We have proposed to implement a "Smart Attendance System for Face Recognition" through this large applications are incorporated.

[2.4] Face Recognition Attendance System Based on Real-Time Video Processing

With the advent of the era of big data in the world and the commercial value of face recognition technology, the prospects for face recognition technology are very bright and have great market demand. This article aims to design a face recognition attendance system based on real-time video processing. This article mainly sets four directions to consider the problems: the accuracy rate of the face recognition system in the actual check-in, the stability of the face recognition attendance system with real-time video processing, the truancy rate of the face recognition attendance system with real-time video processing and the interface settings of the face recognition attendance system using real-time video processing

[2.5] Automatic Attendance System using Face Recognition

Attendance in an educational institution for students is the most challenging part of the virtual platform. It wastes a lot of effort and time when it is done manually. There was also a lot of fake attendance due to lack of a good attendance system. The face is the most important thing which identifies a human. Therefore, in this paper, we work on an attendance system using a face detection algorithm in real-time using the frontal face recognition concept. In this paper, we describe an efficient algorithm of haar cascade using an open-source image processing framework known as OpenCV.

[2.6] Real-Time Smart Attendance System using Face Recognition Techniques

The management of the attendance can be a great burden on the teachers if it is done by hand. To resolve this problem, smart and auto attendance management system is being utilized. But authentication is an important issue in this system. The smart attendance system is generally executed with the help of biometrics. Face recognition is one of the biometric methods to improve this system. Being a prime feature of biometric verification, facial recognition is being used enormously in several such applications, like video monitoring and CCTV footage system, an interaction between computer & humans and access systems present indoors and network security.

CHAPTER 3

DESIGN CONCEPTS AND METHODOLOGIES

3.1 EXISTING SYSTEM

- We are using PCA and LDA algorithm used to recognition the face attendance system.
- But it is not accuracy to detect the faces. Also it will take some time to recognition the registered faces.
- So, we find out the mistake and improve the project in next level. There we have implemented a multiple face recognition system.

3.2 DISADVANTAGES

- Materials like metal & liquid can impact signal.
- Sometimes not as accurate or reliable as barcode scanners.
- Cost – RFID readers can be 10x more expensive than barcode readers.
- Implementation can be difficult & time consuming**.**

3.3 PROPOSED SYSTEM

- We proposed the new methodologies for an automated attendance system using video-based face recognition.
- Here input to the system is a video and output is an excel sheet with attendance of the students in the video.
- Automated attendance system can be implemented using various techniques of biometrics.
- Face recognition is one of them which do not involve human intervention.
- In this paper, attendance is registered from a video of students of a class by first performing Face Detection which separates faces from non-

faces, and then Face Recognition is carried out which finds the match of the detected face from the face database.

- If it is a valid match then attendance is registered to an excel sheet.

3.4 ADVANTAGES

- The database of the attendance management system can hold up to 2000 individual ‟sinformation.
- An excel sheet is created which contains the student attendance and is mailed to the respected faculty

3.5 SYSTEM ARCHITECTURE

The system architecture of a face recognition attendance system encompasses a sophisticated framework designed to revolutionize the way attendance is tracked and managed. Leveraging cutting-edge facial recognition technology, this architecture ensures accurate identification and verification of individuals by analyzing unique facial features.

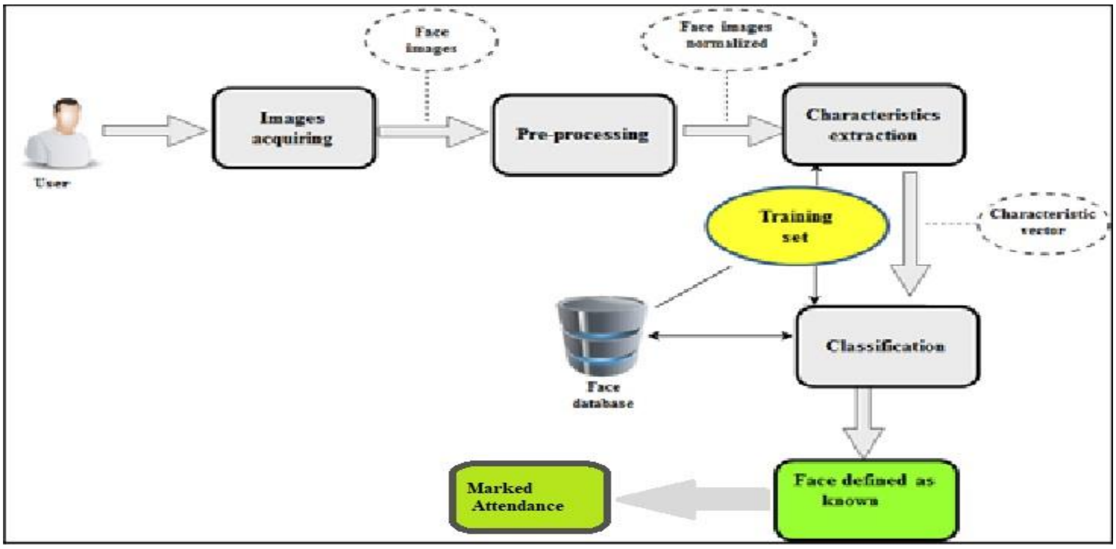


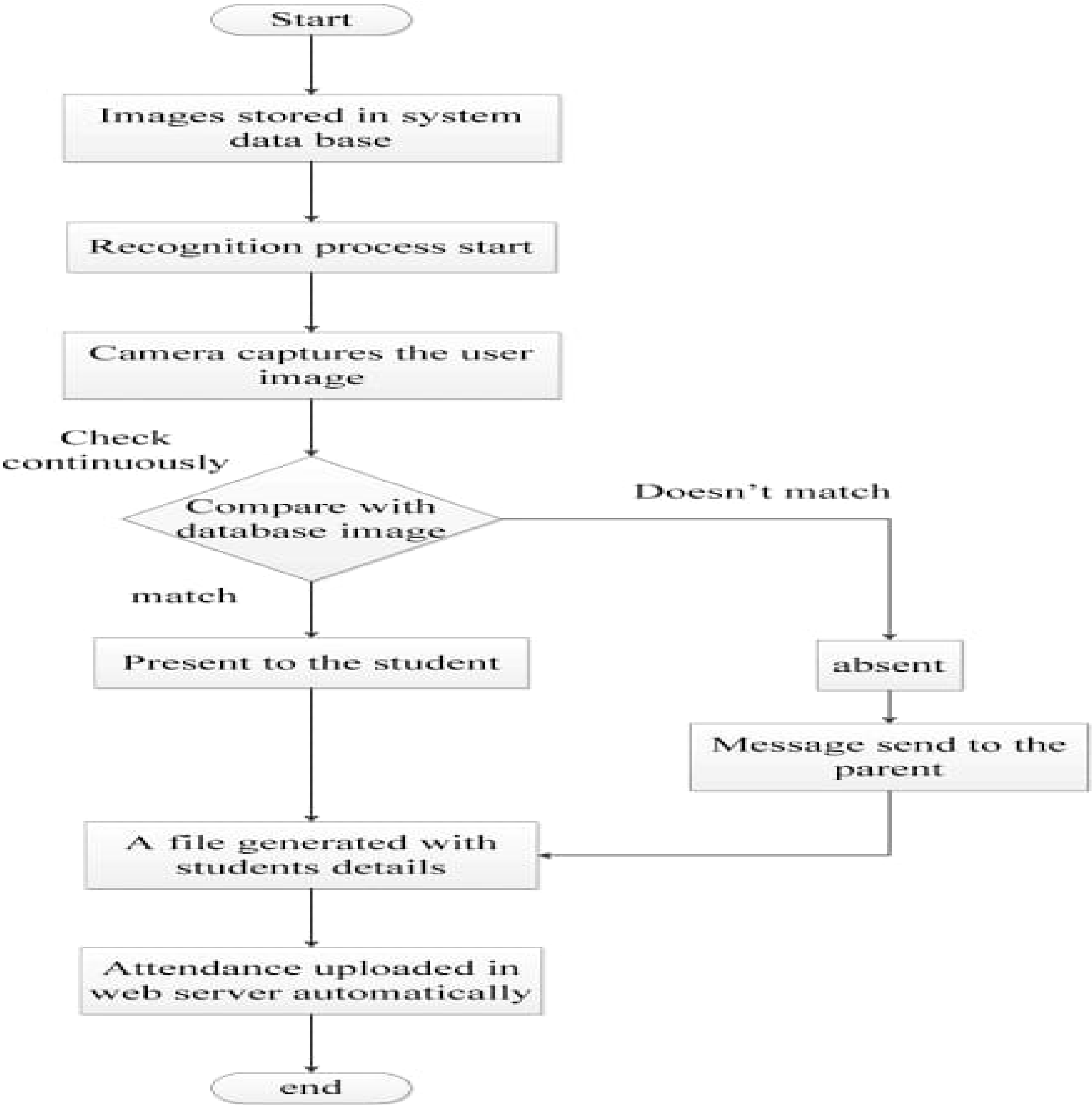
Figure 3.1 System Architecture

CHAPTER 4

MODULE DESCRIPTION

4.1 LIST OF MODULES:

- Creating database
- Video recording
- Face Detection
- Face Recognition
- Registering the attendance



Start
Images stored in system data base
Recognition process start
Camera captures the user image
Check continuously
Compare with database image
Doesn't match
match
Present to the student
absent
Message send to the parent
A file generated with students details
Attendance uploaded in web server automatically
end

Figure 4.1 List of module

4.2 BLOCK DIAGRAM:

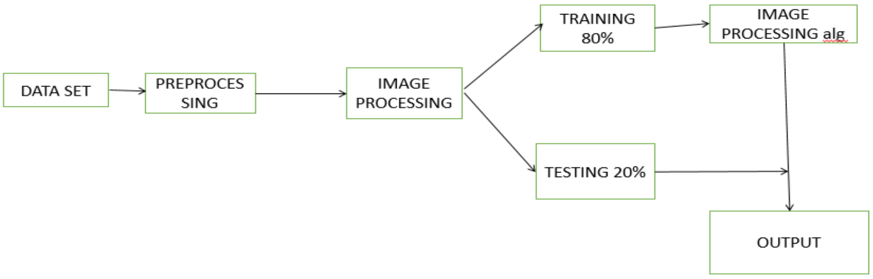


Figure 4.2 Block Diagram

4.3 FLOW DIAGRAM:

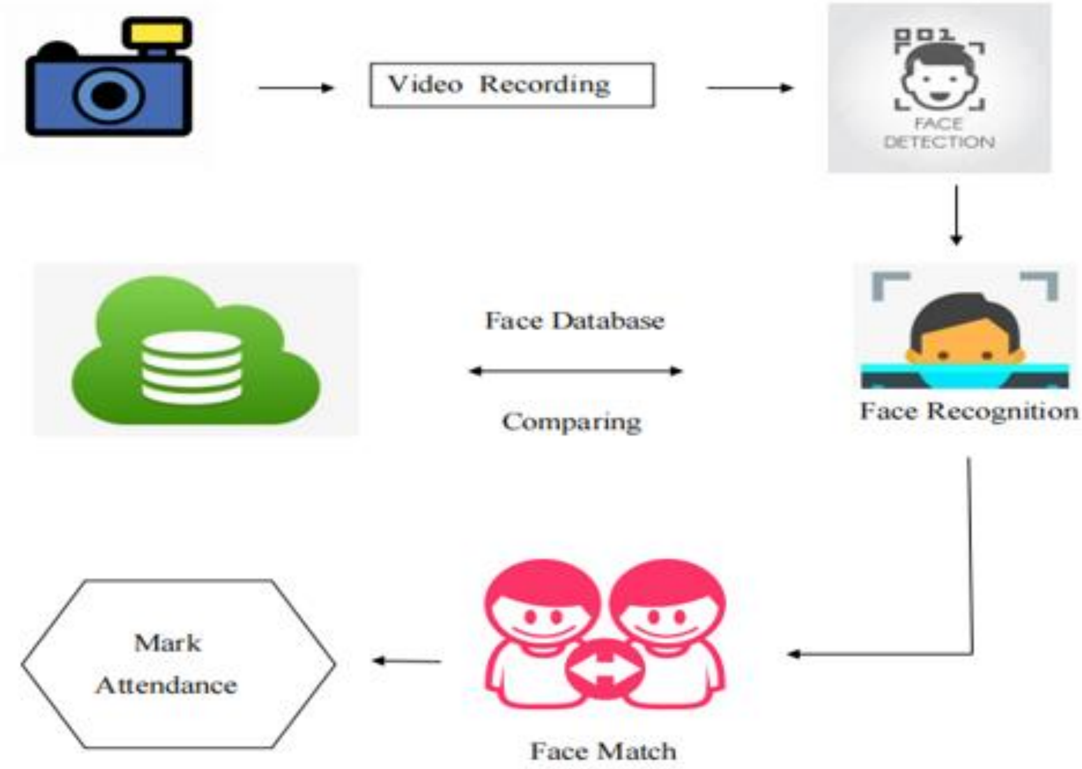


Figure 4.3 Flow Diagram

4.4 CREATING DATABASE:

- The database is the training set of our system and is created in such a way that it contains images of enrolled students.
- These images are cropped to get the region of interest which is the face of the student
- Labeled Faces in the Wild is a database of face photographs designed for studying the problem of unconstrained face recognition.

4.4 VIDEO RECORDING:

- We must have a very good quality camera to get the efficient detection and recognition.
- It will capture the video.
- The video into convert the multiple frames. It will helpful for more accurate to produce the results.
- Facial recognition is a way of identifying or confirming an individual's identity using their face.
- Facial recognition systems can be used to identify people in photos, videos, or in real-time.
- Facial recognition is a category of biometric security.

4.5 FACE DETECTION:

- We are train the collected video frames using Deep leaning techniques.
- Because, it will train to multiple times and push a very fast results.

- Deep Learning is a one of the best algorithm for making a project and it is also give an accurate output.

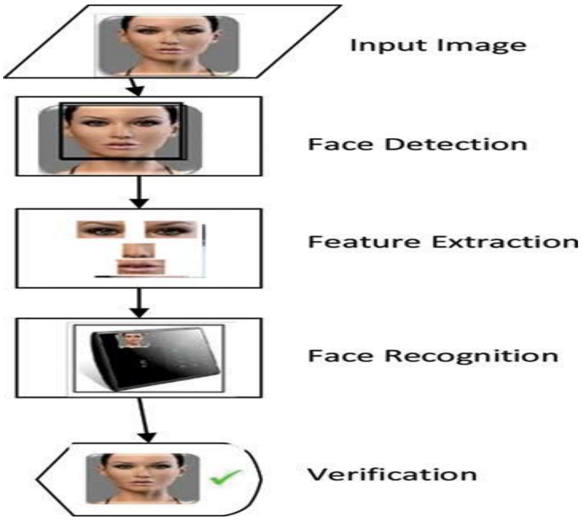


Figure 4.4 Face Detection

- Face detection is a computer technology being used in a variety of applications that identifies human faces in digital images.
- Face detection also refers to the psychological process by which humans locate and attend to faces in a visual scene.

4.6 FACE RECOGNITION:

- This is the most important module of our system which is used to perform the comparison between the test images and the training images.
- the use of face detection come first to determine and isolate face before it can be recognized
- Faces are made of thousands of fine lines and features that must be matched.

4.7 REGISTERING THE ATTENDANCE:

- After completion of the face recognition module, next comes the module toregister the attendance.
- If the detected face has been recognized, then it marks the attendance in the excel sheet.

4.8 ALGORITHM USED:

4.8.1 OPEN CV:

- OpenCV-Python is a library of Python bindings designed to solve computer vision problems. Python is a general purpose programming language started by Guido van

Rossum that became very popular very quickly, mainly because of its simplicity and code readability.

- OpenCV is the huge open-source library for the computer vision, machine learning, and image processing and now it plays a major role in real-time operation which is very important in today's systems. By using it, one can process images and videos to identify objects, faces, or even handwriting of a human

4.8.2 Haar Cascade algorithm

- It is an Object Detection Algorithm used to identify faces in an image or a real time video.
- The algorithm uses edge or line detection features proposed by Viola and Jones in their research paper "Rapid Object Detection using a Boosted Cascade
- The algorithm is given a lot of positive images consisting of faces, and a lot of negative images not consisting of any face to train on them.

4.9 LOCAL BINARY PATTERN:

There are lots of different types of texture descriptors are used to extract features of an image. Local Binary Pattern, also known as LBP, is a simple and grayscale invariant texture descriptor measure for classification. In LBP, a binary code is generated at each pixel by thresholding it's neighbourhood pixels to either 0 or 1 based on the value of the centre pixel. The rule for finding LBP of an image is as follows:

- Set a pixel value as center pixel.
- Collect its neighbourhood pixels (Here I am taking a 3 x 3 matrix so; total number of neighbourhood pixel is 8)
- Threshold it's neighbourhood pixel value to 1 if its value is greater than or equal to centre pixel value otherwise threshold it to 0.
- After thresholding, collect all threshold values from neighbourhood either clockwise or anti-clockwise. The collection will give you an 8-digit binary code. Convert the binary code into decimal.
- Replace the center pixel value with resulted decimal and do the same process for all pixel values present in image.

CHAPTER 5

SYSTEM SPECIFICATION

5.1 Hardware requirements

- Operating System : Windows 7 , 8, 10, 11 (64 bit)
- Software : Python 3.7
- Tools : Anaconda (Jupyter Note Book IDE, VS code)

5.2 Software Requirements

- Hard Disk : 512GB and Above
- RAM : 4GB, 8GB and Above
- Processor : I3, I5 and Above

CHAPTER 6
RESULTS AND DISCUSSION

6.1 Machine learning

Introduction:

In this blog, we will discuss the workflow of a Machine learning project this includes all the steps required to build the proper machine learning project from scratch.

We will also go over data pre-processing, data cleaning, feature exploration and feature engineering and show the impact that it has on Machine Learning Model Performance. We will also cover a couple of the pre-modelling steps that can help to improve the model performance.

Python Libraries that would be need to achieve the task:

1. Numpy
2. Pandas
3. Sci-kit Learn
4. Matplotlib

Understanding the machine learning workflow

We can define the machine learning workflow in 3 stages.

1. Gathering data
2. Data pre-processing
3. Researching the model that will be best for the type of data
4. Training and testing the model
5. Evaluation

Okay but first let’s start from the basics

6.2 What is the machine learning Model?

The machine learning model is nothing but a piece of code; an engineer or data scientist makes it smart through training with data. So, if you give garbage to the model, you will get garbage in return, i.e. the trained model will provide false or wrong prediction

6.2.1 Gathering Data

The process of gathering data depends on the type of project we desire to make, if we want to make an ML project that uses real-time data, then we can build an IoT system that using different sensors data. The data set can be collected from various sources such as a file, database, sensor and many other such sources but the collected data cannot be used directly for performing the analysis process as there might be a lot of missing data, extremely large values, unorganized text data or noisy data. Therefore, to solve this problem Data Preparation is done.

We can also use some free data sets which are present on the internet. Kaggle and UCI Machine learning Repository are the repositories that are used the most for making Machine learning models. Kaggle is one of the most visited websites that is used for practicing machine learning algorithms, they also host competitions in which people can participate and get to test their knowledge of machine learning.

6.2.2 Data pre-processing

Data pre-processing is one of the most important steps in machine learning. It is the most important step that helps in building machine learning models more accurately. In machine learning, there is an 80/20 rule. Every data scientist should spend 80% time for data per-processing and 20% time to actually perform the analysis.

What is data pre-processing?

Data pre-processing is a process of cleaning the raw data i.e. the data is collected in the real world and is converted to a clean data set. In other words, whenever the data is gathered from different sources it is collected in a raw format and this data isn't feasible for the analysis.

Therefore, certain steps are executed to convert the data into a small clean data set, this part of the process is called as data pre-processing.

Why do we need it?

As we know that data pre-processing is a process of cleaning the raw data into clean data, so that can be used to train the model. So, we definitely need data pre-processing to achieve good results from the applied model in machine learning and deep learning projects.Most of the real-world data is messy, some of these types of data are:

6.3 Types of data

6.3.1 Missing data: Missing data can be found when it is not continuously created or due to technical issues in the application (IOT system).

6.3.2 Noisy data: This type of data is also called outliners, this can occur due to human errors (human manually gathering the data) or some technical problem of the device at the time of collection of data.

6.3.3 Inconsistent data: This type of data might be collected due to human errors (mistakes with the name or values) or duplication of data.

Three Types of Data

1. Numeric e.g. income, age
2. Categorical e.g. gender, nationality
3. Ordinal e.g. low/medium/high

How can data pre-processing be performed?

These are some of the basic pre — processing techniques that can be used to convert raw data

6.4 Pre — processing techniques

6.4.1 Conversion of data: As we know that Machine Learning models can only handle numeric features, hence categorical and ordinal data must be somehow converted into numeric features.

6.4.2 Ignoring the missing values: Whenever we encounter missing data in the data set then we can remove the row or column of data depending on our need. This method is known to be efficient but it shouldn’t be performed if there are a lot of missing values in the dataset.

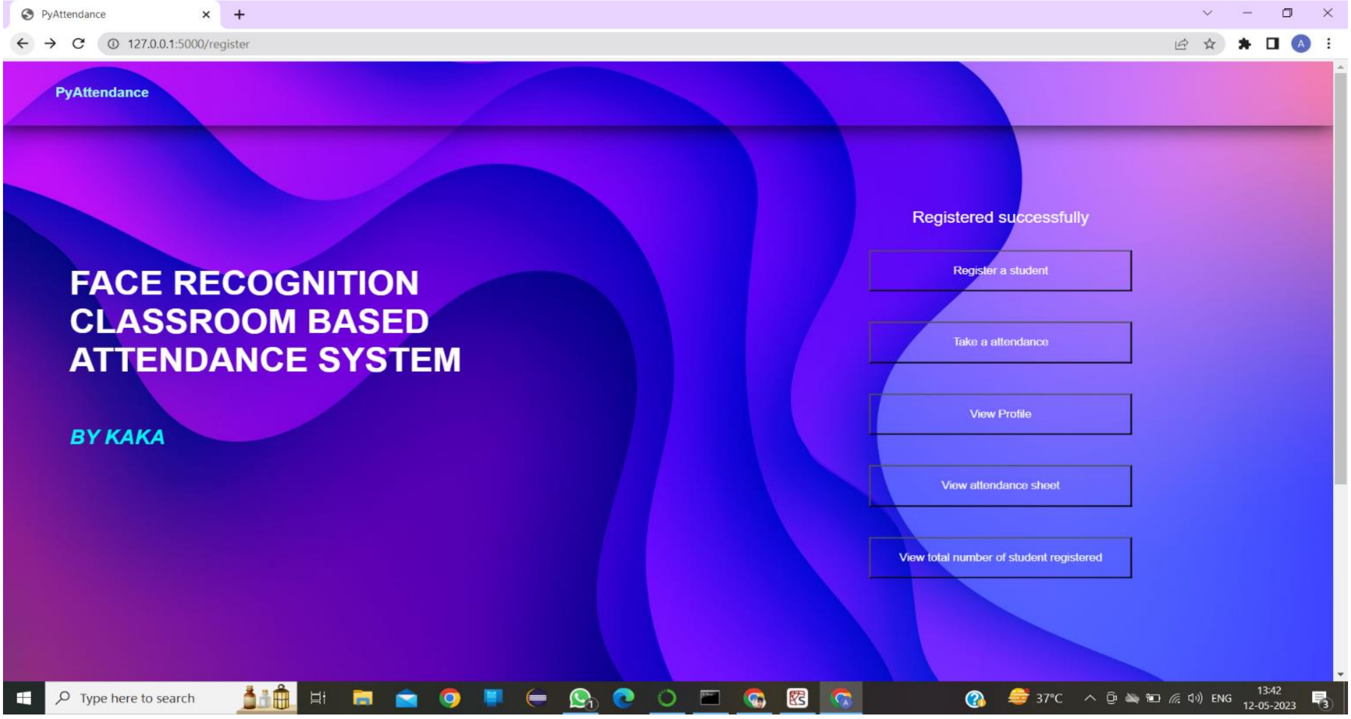
6.4.3 Filling the missing values: Whenever we encounter missing data in the data set then we can fill the missing data manually, most commonly the mean, median or highest frequency value is used.

6.4.4 Machine learning: If we have some missing data then we can predict what data shall be present at the empty position by using the existing data.

6.4.5 Outliers detection: There are some error data that might be present in our data set that deviates drastically from other observations in a data set. [Example: human weight = 800 Kg; due to mistyping of extra 0]

OUTPUT

1. **HOME PAGE:** It consists Student Registration , Take Attendance , View Profile , View Attendance sheet and No Of Students Registered



127.0.0.1:5000/register

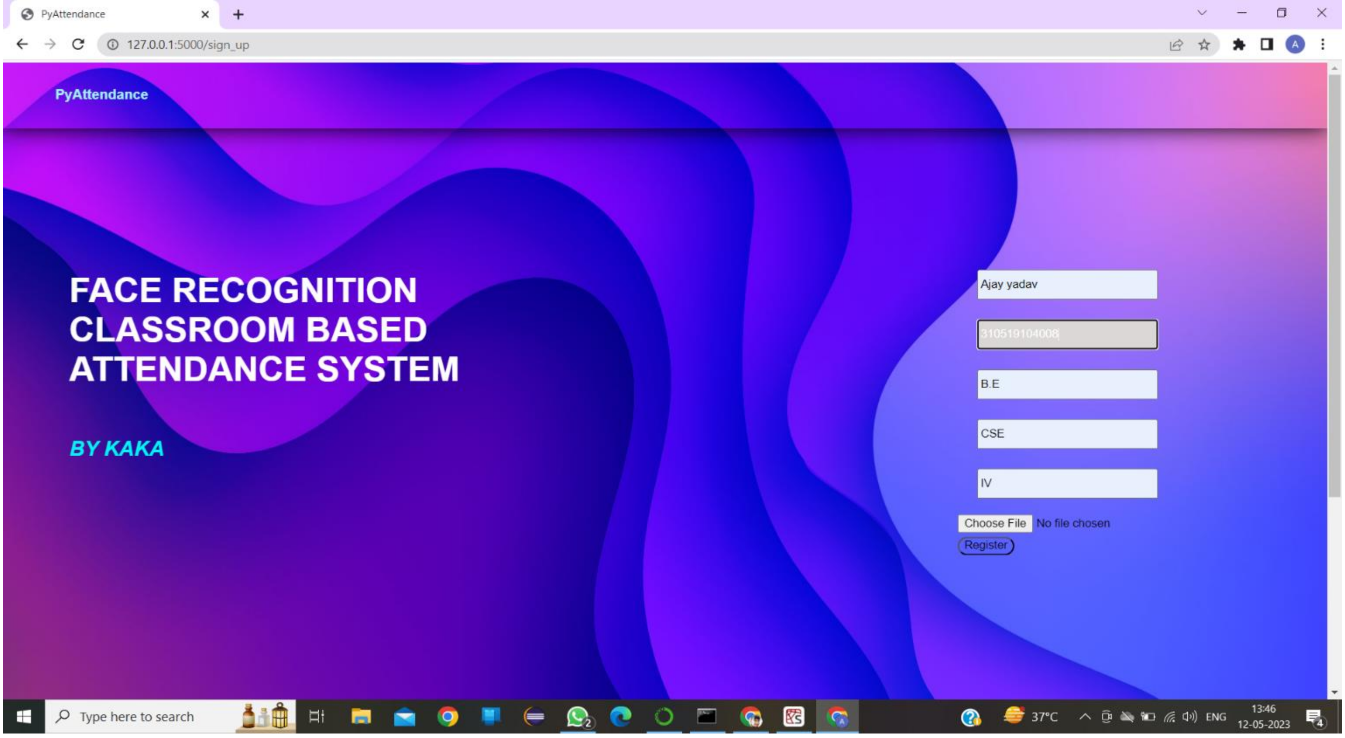
PyAttendance

FACE RECOGNITION CLASSROOM BASED ATTENDANCE SYSTEM

BY KAKA

Registered successfully

2 STUDENT REGISTRATION PAGE: Registration should be done by the Faculty and they required some basic details about students



127.0.0.1:5000/sign_up

PyAttendance

FACE RECOGNITION CLASSROOM BASED ATTENDANCE SYSTEM

BY KAKA

Ajay yadav

310519104008

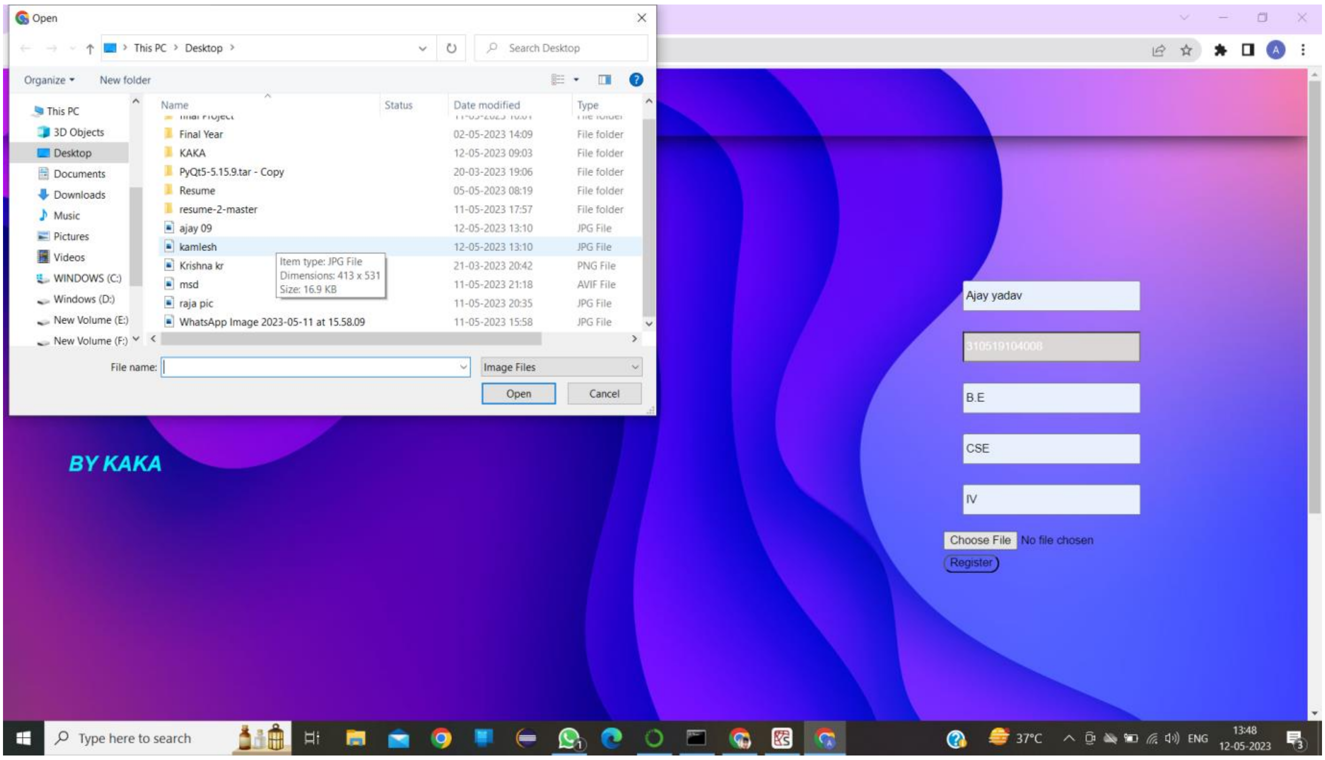
B.E

CSE

IV

Choose File No file chosen

3. IMAGE SELECTION: During Registration Student photo is needed



Name	Status	Date modified	Type
Final Year		02-05-2023 14:09	File folder
KAKA		12-05-2023 09:03	File folder
PyQt5-5.15.9.tar - Copy		20-03-2023 19:06	File folder
Resume		05-05-2023 08:19	File folder
resume-2-master		11-05-2023 17:57	File folder
ajay 09		12-05-2023 13:10	JPG File
kamlesh		12-05-2023 13:10	JPG File
Krishna kr		21-03-2023 20:42	PNG File
msd		11-05-2023 21:18	AVIF File
raja pic		11-05-2023 20:35	JPG File
WhatsApp Image 2023-05-11 at 15.58.09		11-05-2023 15:58	JPG File

Item type: JPG File
Dimensions: 413 x 531
Size: 16.9 KB

File name:

Image Files

Open   Cancel

Ajay yadav

310519104008

B.E

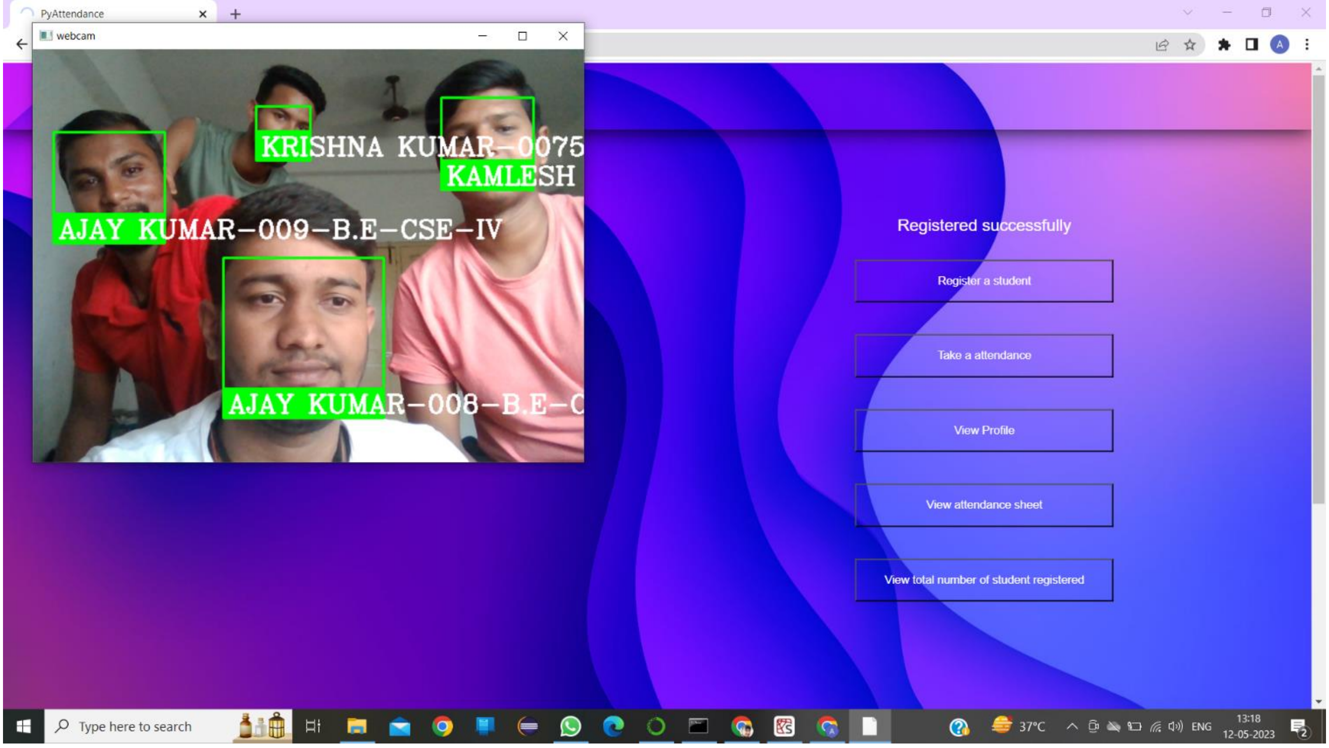
CSE

IV

Choose File No file chosen

Register

4. STUDENT PROFILE : It shows Students Details by capturing Students face



Registered successfully

Register a student

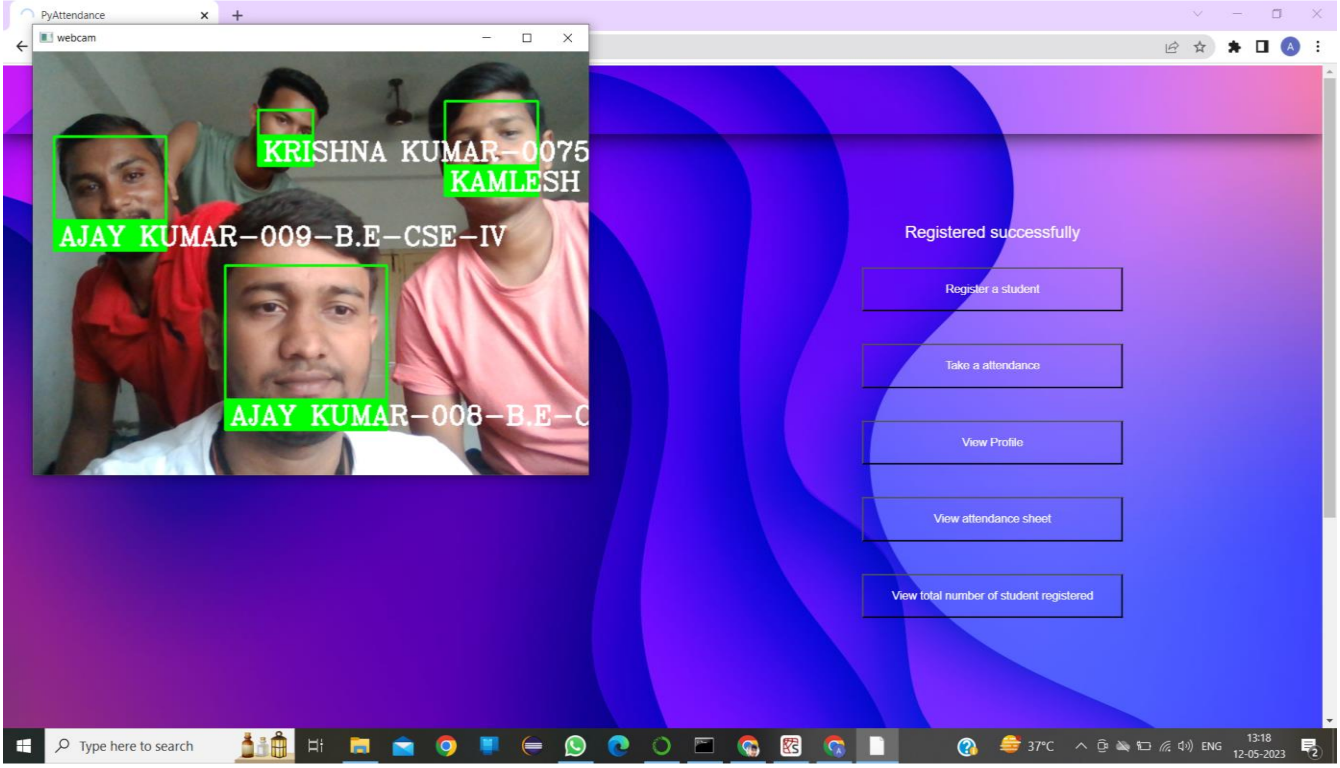
Take a attendance

View Profile

View attendance sheet

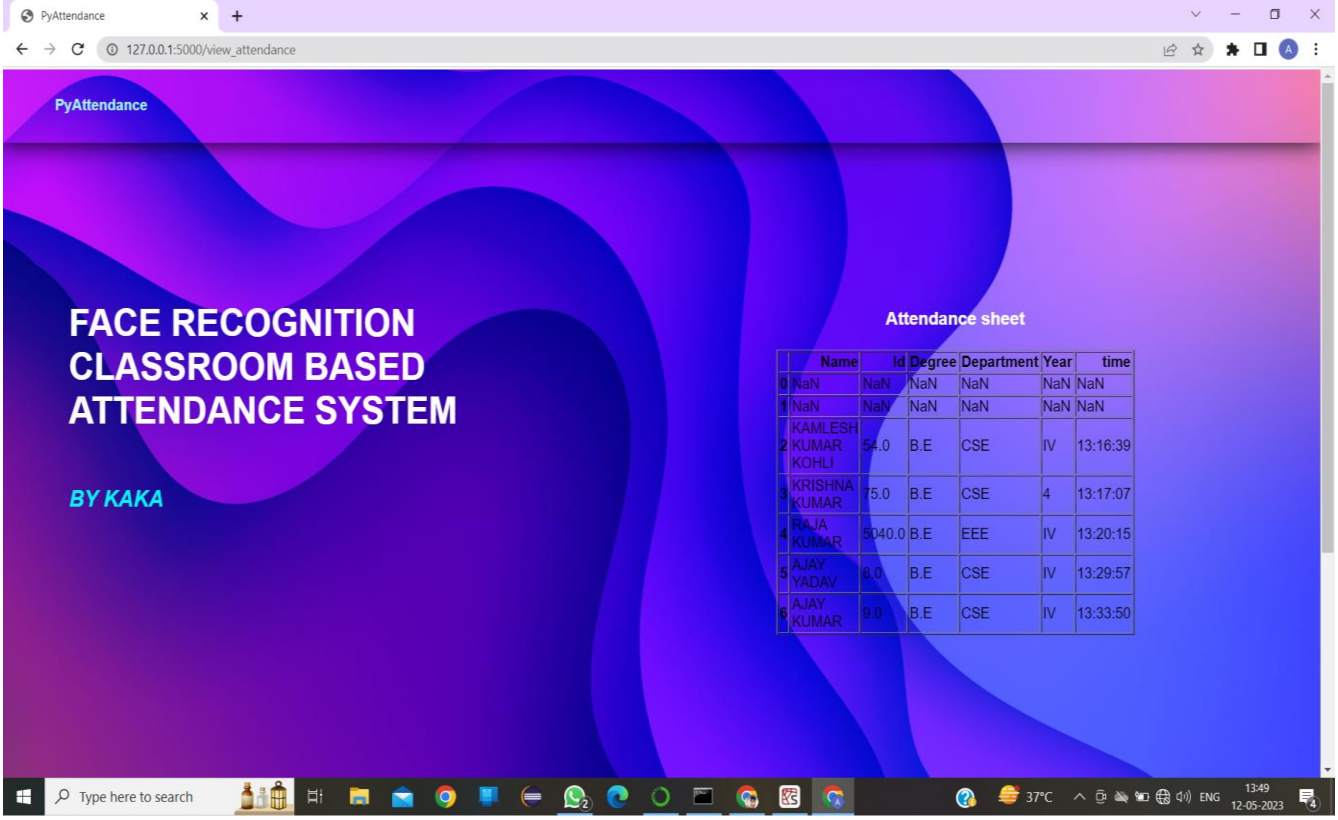
View total number of student registered

5. TAKING MULTIPLE ATTENDACE: It is able to take Multiple Attendance at a single time



Registered successfully

6. ATTENDANCE SHEETS: Faculty can check student Attendance sheet in web page also



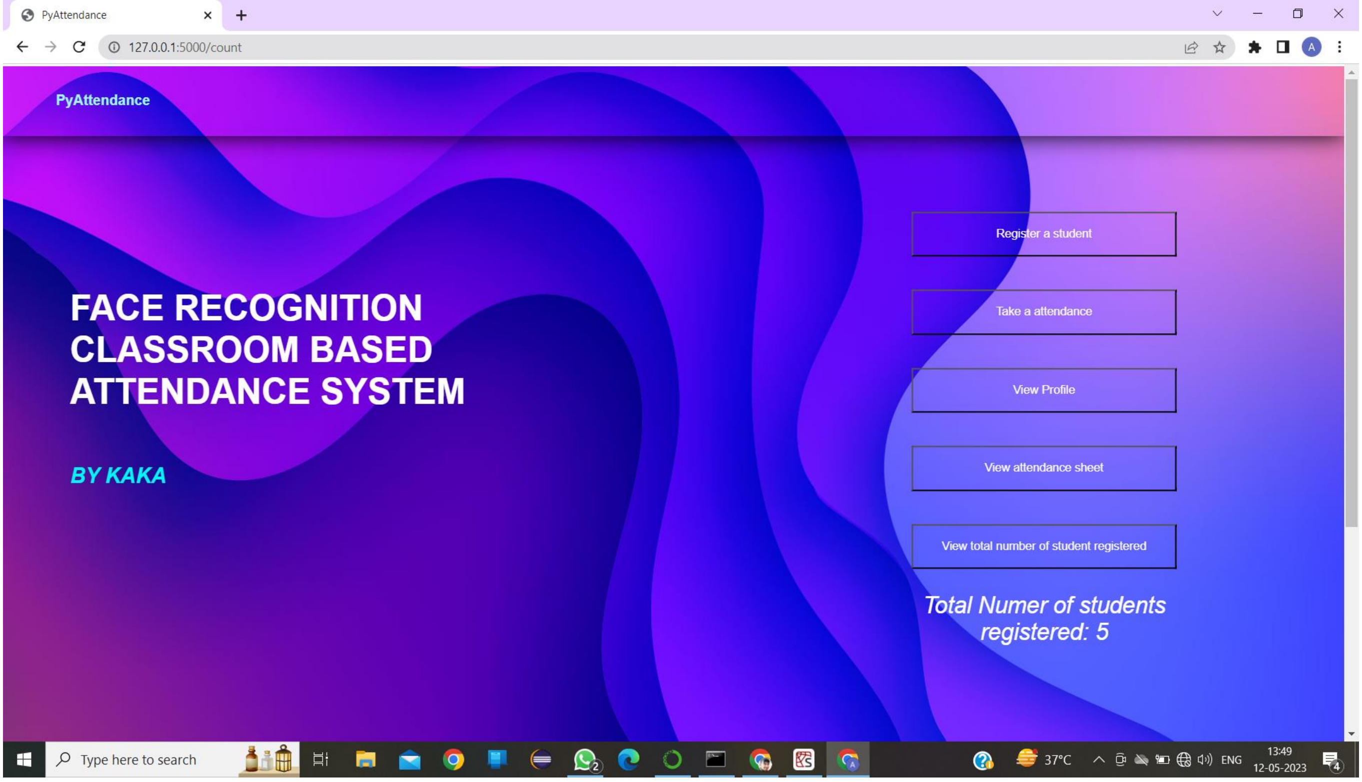
FACE RECOGNITION CLASSROOM BASED ATTENDANCE SYSTEM

BY KAKA

Attendance sheet

	Name	Id	Degree	Department	Year	time
0	NaN	NaN	NaN	NaN	NaN	NaN
1	NaN	NaN	NaN	NaN	NaN	NaN
2	KAMLESH KUMAR KOHLI	54.0	B.E	CSE	IV	13:16:39
3	KRISHNA KUMAR	75.0	B.E	CSE	4	13:17:07
4	RAJA KUMAR	5040.0	B.E	EEE	IV	13:20:15
5	AJAY YADAV	8.0	B.E	CSE	IV	13:29:57
6	AJAY KUMAR	9.0	B.E	CSE	IV	13:33:50

7. **REGISTERED STUDENTS:** It shows Total Number of Students registered

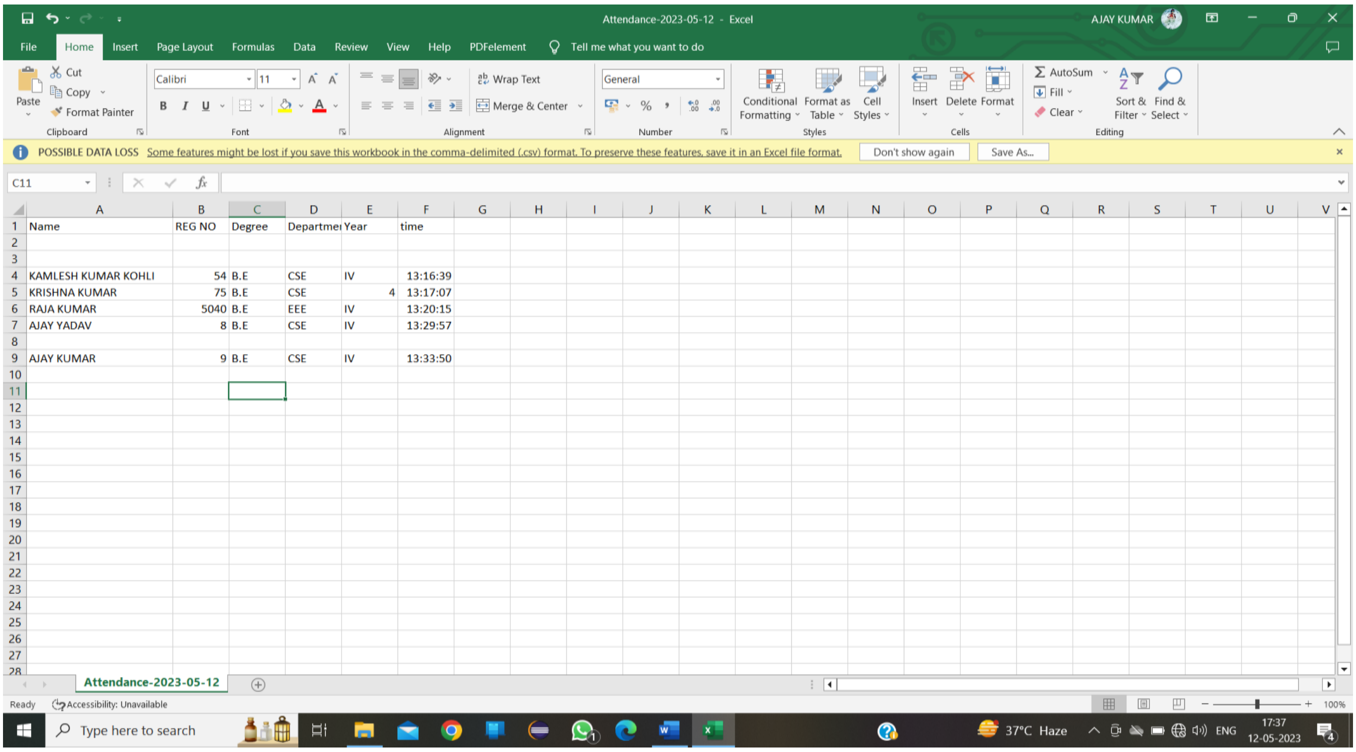


FACE RECOGNITION CLASSROOM BASED ATTENDANCE SYSTEM

BY KAKA

Total Numer of students registered: 5

8. EXCEL ATTENDANCE SHEET: It Excel sheet for students attendance which can download by csv file



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

	A	B	C	D	E	F
1	Name	REG NO	Degree	Departmei	Year	time
2						
3						
4	KAMLESH KUMAR KOHLI	54	B.E	CSE	IV	13:16:39
5	KRISHNA KUMAR	75	B.E	CSE	4	13:17:07
6	RAJA KUMAR	5040	B.E	EEE	IV	13:20:15
7	AJAY YADAV	8	B.E	CSE	IV	13:29:57
8						
9	AJAY KUMAR	9	B.E	CSE	IV	13:33:50

CHAPTER 7
CONCLUSION AND FUTURE ENHANCEMENTS

7.1 Conclusion:

In conclusion, face recognition classroom-based attendance systems using Python offer an efficient and accurate way to manage attendance in classrooms. With the use of machine learning algorithms and computer vision techniques, these systems can accurately identify and verify students' identities, eliminating the need for manual attendance tracking. Moreover, these systems are user-friendly and can be easily integrated with existing classroom technologies. The use of these systems can save valuable time and resources for educational institutions, allowing teachers to focus on teaching and students on learning. Therefore, face recognition classroom-based attendance systems using Python hold great promise for the future of education.

7.2 Future Enhancements

At the same time grants a certain amount of leniency in the attendance calculation, but There is always scope for improvement. Face recognition techniques are not completely accurate, and the system may sometimes be unable to identify students. For more reliability of staff to take attendance we will create it as Android App later, and build one more app which given to their parents which can help to monitor their kids attendance from home .

APPENIDX 1

SOURCE CODE

REGISTER STUDENTS

```
{% extends 'base.html' %}
{% block content %}
<div class="split right">
   <div class="centered"  style="display: flex; justify-content: center;">
<div style="float: right; margin-right: 0%; margin-left: 65%;">
   <div style="margin-top: 20%;">
   <form action="/register" method="post" enctype="multipart/form-data">
      <input type="text" placeholder="Enter your name" name="name" 
style="width: 200px; height: 30px; background-color: rgb(219, 215, 215); 
margin-top: 2%; color: white;">
      <br>
      <br>
      <input type="text" placeholder="Enter the id" name="id" style="width: 
200px; height: 30px; background-color: rgb(219, 215, 215);margin-top: 2%; 
color: white;">
      <br>
      <br>
      <input type="text" placeholder="Enter the degree" name="degree" 
style="width: 200px; height: 30px; background-color: rgb(219, 215, 
215);margin-top: 2%; color: white;">
      <br>
      <br>
```

```
        <input type="text" placeholder="Enter the department"
name="department" style="width: 200px; height: 30px; background-color:
rgb(219, 215, 215);margin-top: 2%; color: white;">
        <br>
        <br>
        <input type="text" placeholder="Enter the Year" name="year"
style="width: 200px; height: 30px; background-color: rgb(219, 215,
215);margin-top: 2%; color: white;">
        <br>
        <br>
        <input type="file" name="image" accept="image/*" />

        <button type="submit" style="background-color: transparent; border-
radius: 12px; display: flex; justify-content: center;margin-top:
2%;">Register</button>

    </form>
</div>
</div>
</div>
</div>

{% endblock %}
```

TAKE A ATTENDANCE

```
{% extends 'base.html' %}
{% block content %}
<div class="split right">
    <div class="centered">
<div style="float: right; margin-right: 10%; margin-left: 50%;">
```

```
    <div align="center" style="margin-top: 20%;">
        <table style="border-collapse: collapse; width: 20%; background-color:
#e5989b; color: white;">
            <h1 style="color: white; font-size: larger;">Attendance sheet</h1>
            <h1 style="color: white;">
            <!--Displaying the converted table-->
                 {% for table in tables %}
                <h2 style="color: white;">{{titles[loop.index]}}</h2>
                {{ table|safe }}
                {% endfor %}
            </h1>
        </table>
    </div>
</div>
</div>
</div>

{% endblock %}
```

STYLE PAGE

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>PyAttendance</title>
</head>
<STYle>
    header img {
```

```
    height: 80px;
    margin-left: 40px;
}
body {
    height: 125vh;
    background-image: url(static/images/vector1.jpg);
    background-size: cover;
    font-family: sans-serif;
    margin-top: 80px;
    padding: 30px;
}

main {
    color: white;
}

header {
    background-color: transparent;
    color: rgb(255, 94, 0);
    position: fixed;
    top: 0;
    left: 0;
    right: 0;
    height: 80px;
    display: flex;
    align-items: center;
    box-shadow: 0 0 25px 0 black;
}
```

```
header * {
    display: inline;
}

header li {
    margin: 20px;
}

header li a {
    color: white;
    text-decoration: none;
}
/* Split the screen in half */
.split {
  height: 100%;
  width: 50%;
  position: fixed;
  z-index: 1;
  top: 0;
  overflow-x: hidden;
  padding-top: 20px;
}

/* Control the left side */
.left {
  left: 0;


}
```

```
/* Control the right side */
.right {
  right: 0;

}

/* If you want the content centered horizontally and vertically */
.centered {
  position: absolute;
  top: 50%;
  left: 50%;
  transform: translate(-50%, -50%);
  text-align: center;
}

/* Style the image inside the centered container, if needed */
.centered img {
  width: 150px;
  border-radius: 50%;
}
</STYle>
<body style="background-image: url(static/images/vector1.jpg);background-repeat: no-repeat; background-size: cover;">
    <header>
        <nav>
            <ul>
                <li><a href="{{ url_for('home') }}" style="font-style: Impact; font-weight: bolder; font: size larger ; color:  #9df8fd;">PyAttendance</a></li>
            </ul>
        </nav>
    </header>
    <div class="split left">
            <section>
            <main>
               <DIV style="margin-left: 10%;margin-top: 30%;">

               <h1 style="width: 600px; font-size: 40px;">FACE RECOGNITION
CLASSROOM BASED ATTENDANCE SYSTEM </h1>
                <p style="width: 500px; float: left; font-size: 24px; color: rgb(2, 246,
246); font-weight: bold; font-style: italic;">BY KAKA</p>

               </DIV>
            </main>
           </section>
      </div>
body {
    height: 125vh;
    background-image: url(static/images/vector1.jpg);
    background-size: cover;
    font-family: sans-serif;
    margin-top: 80px;
    padding: 30px;
}

main {
    color: white;
}
```

```
header {
    background-color: transparent;
    color: rgb(255, 94, 0);
    position: fixed;
    top: 0;
    left: 0;
    right: 0;
    height: 80px;
    display: flex;
    align-items: center;
    box-shadow: 0 0 25px 0 black;
}

header * {
    display: inline;
}

header li {
    margin: 20px;
}

header li a {
    color: white;
    text-decoration: none;
}
/* Split the screen in half */
.split {
  height: 100%;
  width: 50%;
  position: fixed;
  z-index: 1;
  top: 0;
  overflow-x: hidden;
  padding-top: 20px;
}

/* Control the left side */
.left {
  left: 0;

}

/* Control the right side */
.right {
  right: 0;

}

/* If you want the content centered horizontally and vertically */
.centered {
  position: absolute;
  top: 50%;
  left: 50%;
  transform: translate(-50%, -50%);
  text-align: center;
}
```

```
    {% block content %}
    {% endblock %}

</body>
</html>
```

HOME PAGE

```
{% extends 'base.html' %}
{% block content %}
<div class="split right">
  <div class="centered">
    <section><div style="float: right; margin-right: 0%; margin-left: 65%;">
      <div style="display: flex; justify-content: center; margin-top: 5%;">
      <p style="color: white; font-size: larger;">{{msg}}</p>

        <br></div>
      <div style="display: flex; justify-content: center; margin-top: 3%;">

      <form action="/sign_up" method="post">
        <button type="submit"  style="width: 300px; height: 50px; background-color: transparent; color: white;">Register a student</button>
      </form>
    </div>
      <br>
      <br>
      <div style="display: flex; justify-content: center;">
      <form action="/take_attendance" method="post">
        <button type="submit"  style="width: 300px; height: 50px; background-color: transparent; color: white;">Take a attendance</button>
      </form>
    </div>
    <br>
      <br>
      <div style="display: flex; justify-content: center;">
      <form action="/view_profile" method="post">
        <button type="submit"  style="width: 300px; height: 50px;
background-color: transparent; color: white;">View Profile</button>
      </form>
    </div>
    <section><div style="float: right; margin-right: 0%; margin-left: 65%;">
      <div style="display: flex; justify-content: center; margin-top: 5%;">
      <p style="color: white; font-size: larger;">{{msg}}</p>

        <br></div>
      <div style="display: flex; justify-content: center; margin-top: 3%;">

      <form action="/sign_up" method="post">
        <button type="submit"  style="width: 300px; height: 50px;
background-color: transparent; color: white;">Register a student</button>
      </form>
    </div>


    <br>
    <br>
    <div style="display: flex; justify-content: center;">
      <form action="/view_attendance" method="post">
        <button type="submit"  style="width: 300px; height: 50px;
background-color: transparent; color: white;">View attendance sheet</button>
      </form>
    </div>
    <br>
    <br>
    <div style="display: flex; justify-content: center;">
      <form action="/count" method="post">
        <button type="submit"  style="width: 300px; height: 50px;
background-color: transparent; color: white;">View total number of student
registered</button>
      </form>
    </div>
    <div style="display: flex; justify-content: center;">
      {% if show == 0 %}
      <p style="font-style: italic; font-size: 26px; color: white;">Total Numer
of students registered: {{count}}</p>
      {% endif %}
    </div>
    </div>
  </section>
  </div>
 </div>
 {% endblock %}
```

CSS CODE

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
```

```
</head>
<STYle>
   header img {
   height: 80px;
   margin-left: 40px;
}
body {
   height: 125vh;
   background-image: url(static/images/vector1.jpg);
   background-size: cover;
   font-family: sans-serif;
   margin-top: 80px;
   padding: 30px;
}

main {
   color: white;
}

header {
   background-color: transparent;
   color: rgb(255, 94, 0);
   position: fixed;
   top: 0;
   left: 0;
   right: 0;
   height: 80px;
   display: flex;
   align-items: center;
```

```
    box-shadow: 0 0 25px 0 black;
}

header * {
    display: inline;
}

header li {
    margin: 20px;
}

header li a {
    color: white;
    text-decoration: none;
}
/* Split the screen in half */
.split {
  height: 100%;
  width: 50%;
  position: fixed;
  z-index: 1;
  top: 0;
  overflow-x: hidden;
  padding-top: 20px;
}

/* Control the left side */
.left {
  left: 0;
```

```
}

/* Control the right side */
.right {
  right: 0;

}

/* If you want the content centered horizontally and vertically */
.centered {
  position: absolute;
  top: 50%;
  left: 50%;
  transform: translate(-50%, -50%);
  text-align: center;
}

/* Style the image inside the centered container, if needed */
.centered img {
  width: 150px;
  border-radius: 50%;
}
</STYle>
<body style="background-image: url(static/images/vector1.jpg);background-repeat: no-repeat; background-size: cover;">
    <header>
        <nav>
            <ul>
```

```
               <li><a href="{{ url_for('home') }}" style="font-style: Impact; font-weight: bolder; font-size: larger; text-shadow: 2px 2px #f485f8;">FACE ATTENDANCE SYSTEM</a></li>
           </ul>
        </nav>
    </header>
    <div class="split left">
           <section>
           <main>
              <DIV style="margin-left: 10%;margin-top: 30%;">

              <h1 style="width: 600px; font-size: 40px;">FACE RECOGNITION BASED ATTENDANCE SYSTEM</h1>
               <p style="width: 500px; float: left; font-size: 24px;">A facial recognition attendance system incorporates facial recognition technology to recognize and verify an student's facial features and to record attendance automatically.</p>

              </DIV>
           </main>
          </section>
      </div>

      {% block content %}
      {% endblock %}
```

```
</body>
</html>
```

APP.PY

```
from flask import Flask, render_template, request, redirect, url_for
import cv2
import os
import csv
import numpy as np
import os
from PIL import Image
import pickle
import pandas as pd
import cv2
import numpy as np
import face_recognition
import os
from datetime import date
from datetime import datetime
from werkzeug.utils import secure_filename

today = date.today()

UPLOAD_FOLDER = '/ImagesAttendence'




app = Flask(__name__)
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
```

```
@app.route('/')
def home():
    return render_template("home.html")


@app.route('/register', methods=['POST', 'GET'])
def register():
     if request.method == 'POST':
        name = request.form['name']
        id = request.form['id']
        degree = request.form['degree']
        department = request.form['department']
        year = request.form['year']
        image_file = request.files['image']
        image_file.save('ImagesAttendence/'+name+'-'+id+ '-'+ degree+'-
'+department+'-'+year+'.jpg')
        path = 'ImagesAttendence'
        images = []
        classNames = []
        myList = os.listdir(path)
        print(myList)
        for cl in myList:
            curImg = cv2.imread(f'{path}/{cl}')
            images.append(curImg)
            classNames.append(os.path.splitext(cl)[0])
            print(classNames)

        def findEncodings(images):
            encodeList = []
            for img in images:
                img = cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
                encode = face_recognition.face_encodings(img)[0]
                encodeList.append(encode)
            return encodeList
        encodeListKnown = findEncodings(images)
        print('Encoding Complete')
        msg = 'Registered successfully'
        return render_template("home.html", msg = msg)
@app.route('/sign_up', methods=['POST', 'GET'])
def sign_up():
    return render_template("register.html")



@app.route('/take_attendance', methods=['POST', 'GET'])
def take_attendance():
    if request.method == "POST":
        fields = ['Name','Id','Degree','Department','Year','time']
        filename = 'Attendance-' + str(today) + '.csv'
        if not os.path.exists('Attendance-' + str(today) + '.csv'):
            with open(filename, 'w') as csvfile:
```

```
        csvwriter = csv.writer(csvfile)
        csvwriter.writerow(fields)
path = 'ImagesAttendence'
images = []
classNames = []
myList = os.listdir(path)
print(myList)
for cl in myList:
    curImg = cv2.imread(f'{path}/{cl}')
    images.append(curImg)
    classNames.append(os.path.splitext(cl)[0])
print(classNames)

def findEncodings(images):
    encodeList = []
    for img in images:
        img = cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
        encode = face_recognition.face_encodings(img)[0]
        encodeList.append(encode)
    return encodeList

def markAttendance(name):
    path = 'Attendance-' + str(today) + '.csv'
    s = name
    for j in range(0,5):
        i = s.find('-')
        if j == 0:
            name = s[:i]
        elif j == 1:
```

```
                    id = s[:i]
                elif j == 2:
                    degree = s[:i]
                elif j == 3:
                    department = s[:i]
                elif j == 4:
                    year = s
                s = s[i+1:]
            with open(path,'r+') as f:
                myDataList = f.readlines()
                nameList = []
                for line in myDataList:
                    entry = line.split(',')
                    nameList.append(entry[0])
                if name not in nameList:
                    now = datetime.now()
                    dtString = now.strftime('%H:%M:%S')

f.writelines(f'\n{name},{id},{degree},{department},{year},{dtString}')



        encodeListKnown = findEncodings(images)
        print('Encoding Complete')

        cap = cv2.VideoCapture(0)

        while True:
            success, img = cap.read()
            imgS = cv2.resize(img,(0,0),None,0.25,0.25)
            imgS = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

            faceCurFrame = face_recognition.face_locations(imgS)
            encodesCurFrame =
face_recognition.face_encodings(imgS,faceCurFrame)

            for encodeFace,faceLoc in zip(encodesCurFrame,faceCurFrame):
                matches =
face_recognition.compare_faces(encodeListKnown,encodeFace)
                faceDis =
face_recognition.face_distance(encodeListKnown,encodeFace)
        #print(faceDis)
                matchIndex = np.argmin(faceDis)

                if matches[matchIndex]:
                    name = classNames[matchIndex].upper()
                    markAttendance(name)
                else:
                    name = 'unknown'
        #print(name)
                    y1,x2,y2,x1 = faceLoc
        #y1,x2,y2,x1 = y1*4,x2*4,y2*4,x1*4
                    cv2.rectangle(img,(x1,y1),(x2,y2),(0,255,0),2)
                    cv2.rectangle(img,(x1,y2-35),(x2,y2),(0,255,0),cv2.FILLED)
                    cv2.putText(img,name,(x1+6,y2-
6),cv2.FONT_HERSHEY_COMPLEX,1,(255,255,255),2)
```

```
            cv2.imshow('webcam',img)
            cv2.waitKey(1)

    return render_template("mark_attendance.html")

@app.route('/count', methods=['POST', 'GET'])
def count():
    if request.method == "POST":
        import os
        directory_path = "ImagesAttendence"
        count = 0

        for item in os.listdir(directory_path):
            item_path = os.path.join(directory_path, item)
            count += 1
    return render_template("home.html", show = 0,count = count)

@app.route('/view_attendance', methods=['POST', 'GET'])
def view_attendance():
    if request.method == "POST":
        data = pd.read_csv('Attendance-' + str(today) + '.csv')
    return render_template('attendance.html', tables=[data.to_html()], titles=[''])

@app.route('/profile', methods=['POST', 'GET'])
def profile():
    if request.method == "POST":
        path = 'ImagesAttendence'
        images = []
```

```
classNames = []
myList = os.listdir(path)
print(myList)
for cl in myList:
    curImg = cv2.imread(f'{path}/{cl}')
    images.append(curImg)
    classNames.append(os.path.splitext(cl)[0])
print(classNames)

def findEncodings(images):
    encodeList = []
    for img in images:
        img = cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
        encode = face_recognition.face_encodings(img)[0]
        encodeList.append(encode)
    return encodeList



encodeListKnown = findEncodings(images)
print('Encoding Complete')

cap = cv2.VideoCapture(0)

while True:
    success, img = cap.read()
    imgS = cv2.resize(img,(0,0),None,0.25,0.25)
    imgS = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
```

```
            faceCurFrame = face_recognition.face_locations(imgS)
            encodesCurFrame =
face_recognition.face_encodings(imgS,faceCurFrame)
            for encodeFace,faceLoc in zip(encodesCurFrame,faceCurFrame):
                matches =
face_recognition.compare_faces(encodeListKnown,encodeFace)
                faceDis =
face_recognition.face_distance(encodeListKnown,encodeFace)
        #print(faceDis)
                matchIndex = np.argmin(faceDis)

                if matches[matchIndex]:
                    name = classNames[matchIndex].upper()
                #else:
                    #name = 'unknown'
        #print(name)
                    y1,x2,y2,x1 = faceLoc
        #y1,x2,y2,x1 = y1*4,x2*4,y2*4,x1*4
                    cv2.rectangle(img,(x1,y1),(x2,y2),(0,255,0),2)
                    cv2.rectangle(img,(x1,y2-35),(x2,y2),(0,255,0),cv2.FILLED)
                    cv2.putText(img,name,(x1+6,y2-
6),cv2.FONT_HERSHEY_COMPLEX,1,(255,255,255),
            cv2.imshow('webcam',img)
            cv2.waitKey(1)

    return render_template("mark_attendance.html")
if __name__ == "__main__":
    app.run(debug=True)
```

REFERENCES

[1] “Real-Time Smart Attendance System using Face Recognition Techniques”(Shreyak Sawhney; Karan Kacker; Samyak Jain; Shailendra Narayan Singh; Rakesh Garg, 29 July 2019)

[2] “Student Attendance System using Face Recognition” (Samridhi Dev; Tushar Patnaik, 07 October 2020)

[3] “Face Recognition Based Smart Attendance System” (A Arjun Raj; Mahammed Shoheb; K Arvind; K S Chethan, 2020)

[4] “Face Recognition Attendance System Based on Real-Time Video Processing”(Hao Yang; Xiaofeng Hank, 2020)

[5] “Automatic Attendance System using Face Recognition” (Susanta Kumar Sarangi; Arunesh Paul; Harshit Kishor; Kritath Pandey, 21 December 2021)

[6] “AttenFace: A Real Time Attendance System Using Face Recognition (Ashwin Rao, 14 Nov 2022)