ARTIFICIAL INTELLIGENCE

LAB PROJECT SUBMISSION

Project Name

REAL TIME FACE ATTENDANCE SYSTEM USING DEEP LEARNING

Team Members

Yash (102166002)
Swaraj (102116075)
Prateek (102116066)
Devin (102116079)

GROUP : 2CS11

Submitted To :

Dr Varun Srivastava

INTRODUCTION

The primary goal of this project is to create an automated student attendance system based on facial recognition. The test pictures and training images of this suggested technique are restricted to frontal and upright facial images that only contain a single face to improve performance. To ensure no quality variation, the test photographs and training images must be taken using the same equipment. To be identified, the pupils must also register in the database. The user-friendly interface allows for immediate enrollment.

- **Background**

This report discusses the importance of face recognition in daily life and how it is achieved through human intellect and the analysis of visual information. However, humans have limitations in their ability to recognize faces, which is why computers with almost limitless memory, high processing speed, and power are used in face recognition systems. The report also highlights the widespread use of facial recognition technology in various fields such as airport security, criminal investigations, and social networking sites. Additionally, the report provides a brief history of face recognition research starting from the 1960s, including the development of methods such as the detection of facial features and principle component analysis.

- **Problem Statement**

The traditional method of recording student attendance has several issues and can be replaced with a facial recognition-based system that is simple and efficient. The system can overcome the problem of fraudulent approaches and eliminate the need for manual attendance taking. However, the training procedure for the system is time-consuming, and variations in illumination and head postures can impair its effectiveness. Therefore, a real-time functioning system with high precision and quick computation times must be developed.

- **Aims and Objectives**

The goal of this project is to create an automated student attendance system based on facial recognition. The following are expected results to meet the objectives: To identify the facial segment in the video frame.

- To identify the useful aspects on the observed face.
- To categorize the traits in order to identify the observed face.
- To note the indicated student's attendance

Literature Survey

Traditional methods of taking attendance in classrooms can be time-consuming, disruptive, and prone to errors. A facial recognition-based attendance system offers a simpler and more accurate solution. By using cameras to scan the faces of students as they enter the classroom, the system can quickly and easily mark attendance without the need for manual processes.

However, implementing a facial recognition attendance system comes with its own set of challenges. One of the main challenges is ensuring that the system can accurately identify individuals, even in varying lighting conditions or with different facial expressions. This requires a complex algorithm that can analyze facial features and compare them to a database of known faces.

To overcome these challenges, researchers have been working on developing more advanced facial recognition algorithms that can account for these variations and provide higher levels of accuracy. In addition, real-time functioning attendance systems must be created to identify students within specific time limits and avoid missing any students.

Overall, facial recognition-based attendance systems have the potential to greatly improve the efficiency and accuracy of attendance tracking in classrooms. With continued research and development, they can become even more effective and widespread in educational institutions.

Code and Explanation

Step 1: Data Collection and Preprocessing

The first step is to collect face images of the individuals who will be enrolled in the system. This can be done using a camera or other imaging device. The images should be preprocessed to ensure that they are of a consistent size and quality. You can use image processing libraries such as OpenCV or PIL to accomplish this.

Step 2: Face Detection

Once you have collected the images, the next step is to detect the faces within the images. The goal is to accurately identify the location of the face within the image. Here , we used the Haar Cascade Model for face detection

Step 3: Face Recognition

After detecting the face, the next step is to recognize it. You can use pre-trained face recognition models or train your own model using machine learning algorithms such as Support Vector Machines or Neural Networks. The goal is to match the detected face with a known face from the system's database via LBPH Algorithm.

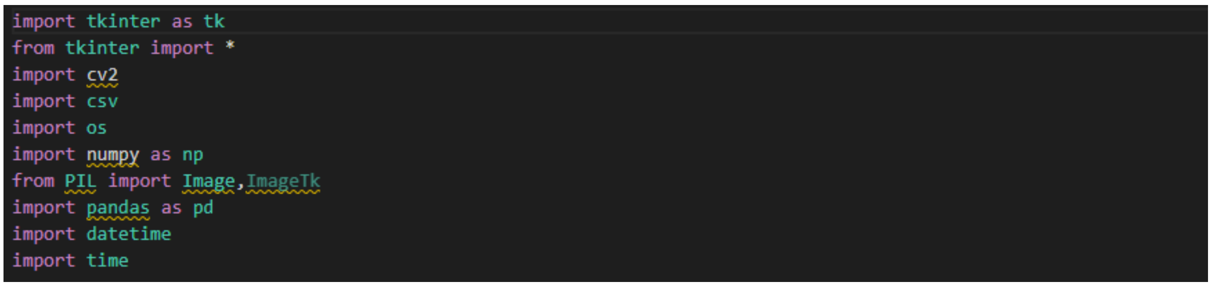
Step 4: Attendance Tracking

Once a face has been recognized, the system can track attendance by recording the time and date of the recognition event. This can be accomplished using a simple database or a cloud-based service.

Step 5: User Interface

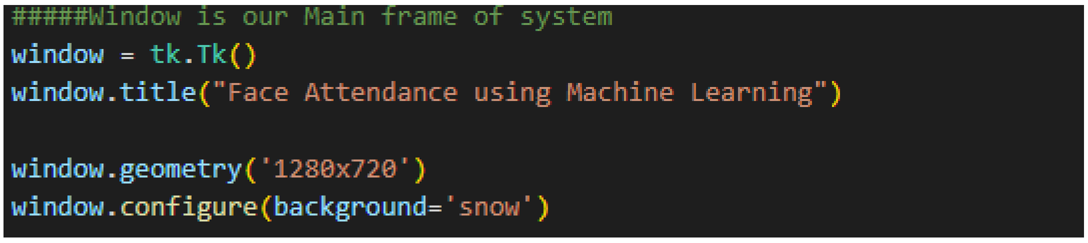
To make the system easy to use, you can create a user interface that displays the attendance records and allows administrators to manage the system. The Tkinter UI makes it easy for users to view attendance records and perform other tasks such as adding new
users to the system.

IMPORTING DEPENDENCIES



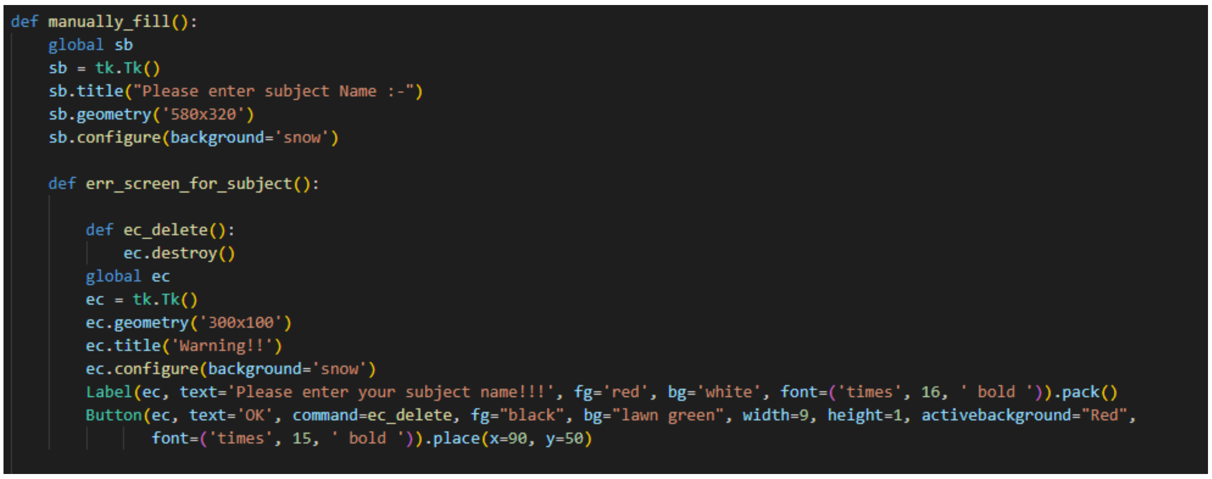
```
import tkinter as tk
from tkinter import *
import cv2
import csv
import os
import numpy as np
from PIL import Image,ImageTk
import pandas as pd
import datetime
import time
```

First of all we need to download the dependencies which will be needed in our project such as Tkinter for GUI, OpenCV, Numpy for mathematical calculation , Pandas for data analysis and management etc.



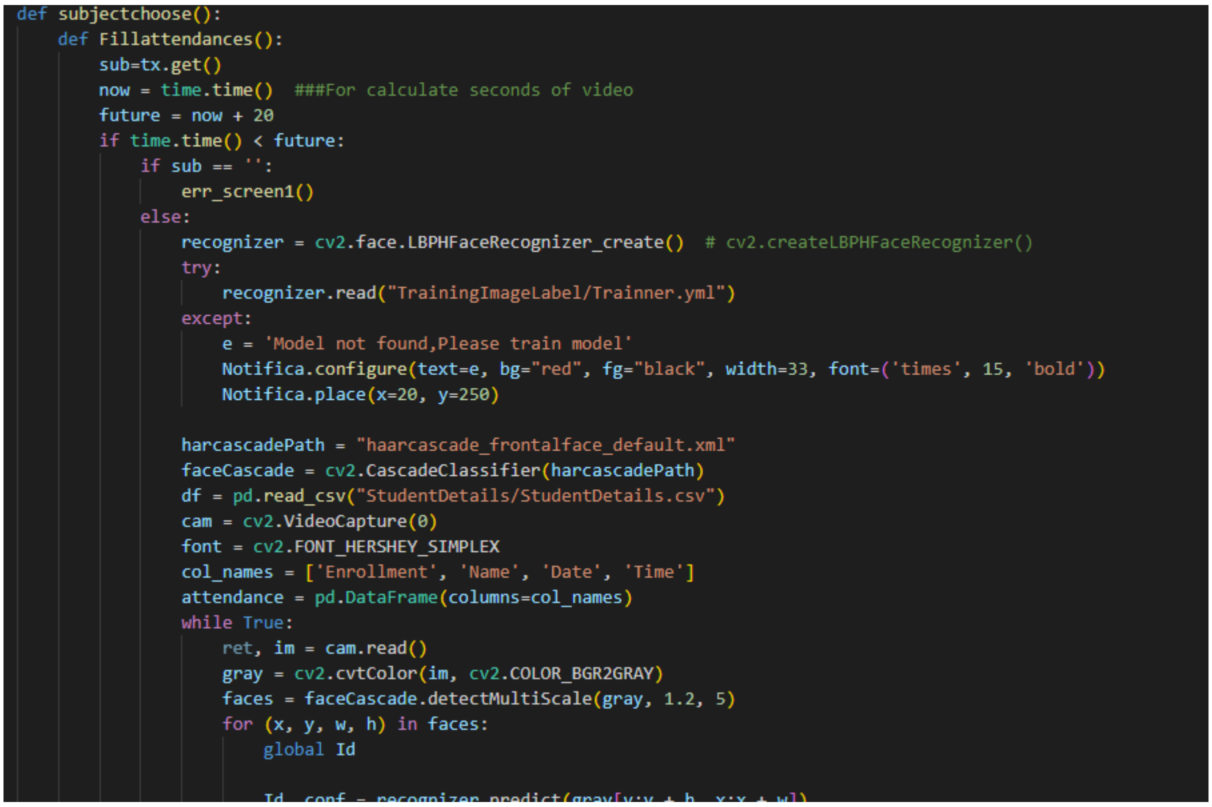
```python
#####Window is our Main frame of system
window = tk.Tk()
window.title("Face Attendance using Machine Learning")

window.geometry('1280x720')
window.configure(background='snow')
```

This is our main window (frame) of the project using tkinter tk class with the title “Face attendance using Machine Learning” with window size and color.



```python
def manually_fill():
    global sb
    sb = tk.Tk()
    sb.title("Please enter subject Name :-")
    sb.geometry('580x320')
    sb.configure(background='snow')

    def err_screen_for_subject():

        def ec_delete():
            ec.destroy()
        global ec
        ec = tk.Tk()
        ec.geometry('300x100')
        ec.title('Warning!!')
        ec.configure(background='snow')
        Label(ec, text='Please enter your subject name!!!', fg='red', bg='white', font=('times', 16, ' bold ')).pack()
        Button(ec, text='OK', command=ec_delete, fg="black", bg="lawn green", width=9, height=1, activebackground="Red",
               font=('times', 15, ' bold ')).place(x=90, y=50)
```

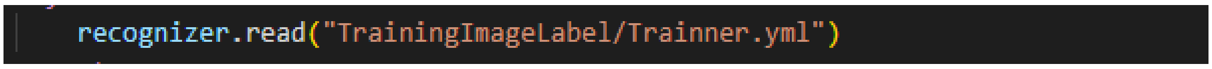
This part of code is used to fill attendance manually in the program and the code continued more further and we will explain the automatic attendance system next.



```
def subjectchoose():
    def Fillattendances():
        sub=tx.get()
        now = time.time()  ###For calculate seconds of video
        future = now + 20
        if time.time() < future:
            if sub == '':
                err_screen1()
            else:
                recognizer = cv2.face.LBPHFaceRecognizer_create()  # cv2.createLBPHFaceRecognizer()
                try:
                    recognizer.read("TrainingImageLabel/Trainner.yml")
                except:
                    e = 'Model not found,Please train model'
                    Notifica.configure(text=e, bg="red", fg="black", width=33, font=('times', 15, 'bold'))
                    Notifica.place(x=20, y=250)

                harcascadePath = "haarcascade_frontalface_default.xml"
                faceCascade = cv2.CascadeClassifier(harcascadePath)
                df = pd.read_csv("StudentDetails/StudentDetails.csv")
                cam = cv2.VideoCapture(0)
                font = cv2.FONT_HERSHEY_SIMPLEX
                col_names = ['Enrollment', 'Name', 'Date', 'Time']
                attendance = pd.DataFrame(columns=col_names)
                while True:
                    ret, im = cam.read()
                    gray = cv2.cvtColor(im, cv2.COLOR_BGR2GRAY)
                    faces = faceCascade.detectMultiScale(gray, 1.2, 5)
                    for (x, y, w, h) in faces:
                        global Id

                        Id, conf = recognizer.predict(gray[y:y + h, x:x + w])
```

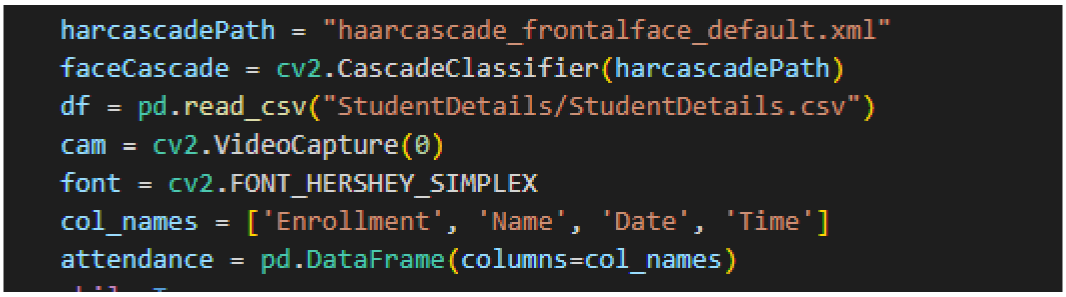
Here the automatic attendance system starts, it is a python function that uses openCv library to capture video, detect faces in the camera by Cascade Classifier and recognize them using a trained face recognition model and records the attendance in the CSV file.

Subject Choose function has a fill attendance function which is called when user enters subject name in the GUI window and clicks the button. Subject name is obtained from a text input named tx and the function checks if the user has entered any subject name.



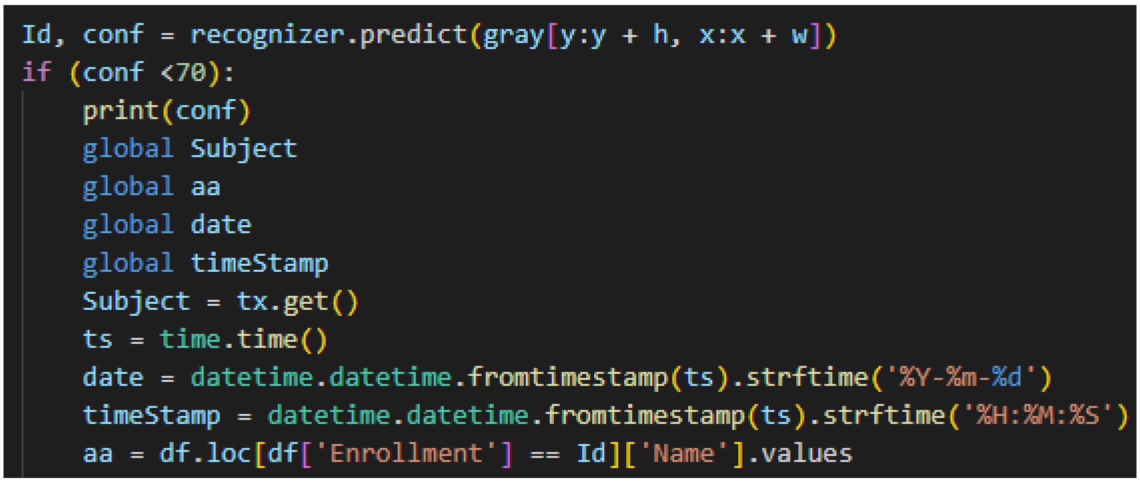
```
    recognizer.read("TrainingImageLabel/Trainner.yml")
```

After getting input for the subject name then it proceeds to load a pretrained face recognition model by using Trainner.yml using the LBPH model it then loads a haar cascade frontal face.xml to detect faces from the video stream then the function also reads a CSV file that contains list of enrolled students with details such as ID and name.



```
harcascadePath = "haarcascade_frontalface_default.xml"
faceCascade = cv2.CascadeClassifier(harcascadePath)
df = pd.read_csv("StudentDetails/StudentDetails.csv")
cam = cv2.VideoCapture(0)
font = cv2.FONT_HERSHEY_SIMPLEX
col_names = ['Enrollment', 'Name', 'Date', 'Time']
attendance = pd.DataFrame(columns=col_names)
```

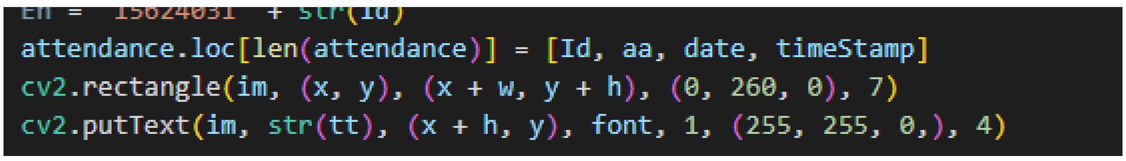
The function then initializes the video camera using openCV video capture class, it capture each frame and detects face by haar cascade classifier then recognizing by face recognition model.



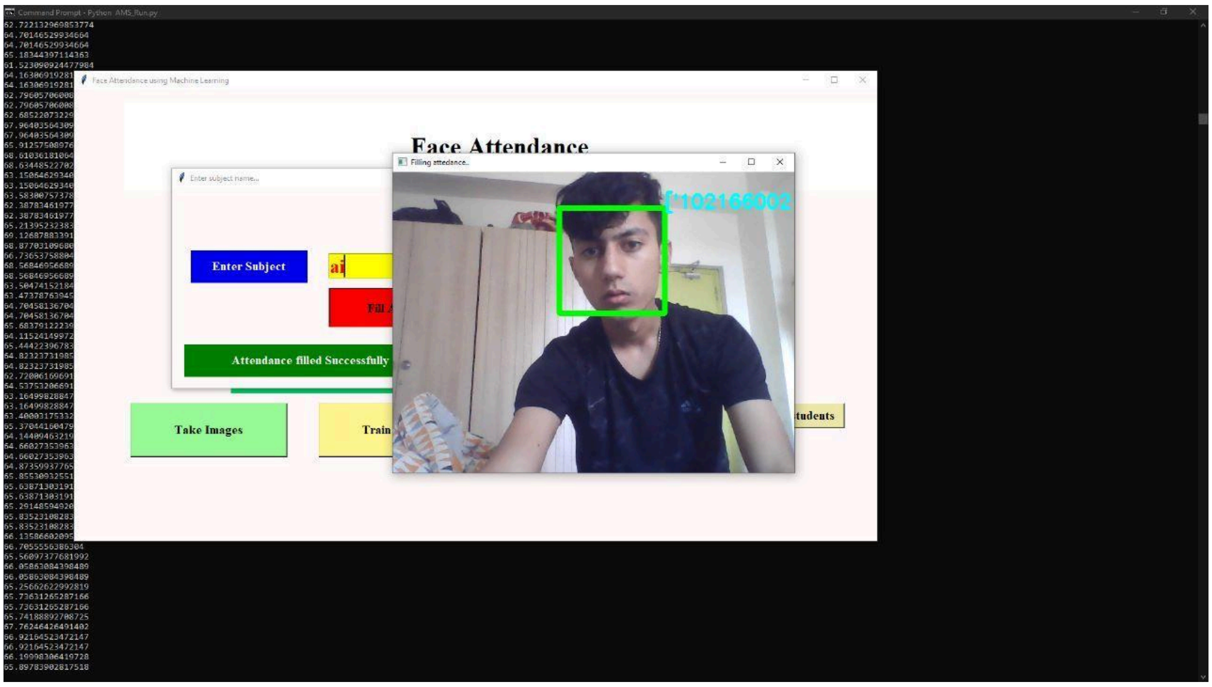
```
Id, conf = recognizer.predict(gray[y:y + h, x:x + w])
if (conf <70):
    print(conf)
    global Subject
    global aa
    global date
    global timeStamp
    Subject = tx.get()
    ts = time.time()
    date = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d')
    timeStamp = datetime.datetime.fromtimestamp(ts).strftime('%H:%M:%S')
    aa = df.loc[df['Enrollment'] == Id]['Name'].values
```

If a face is recognized with a confidence score of less than 70 the function extracts the id and the name from csv file and marks the attendance with subject name, date and time in a pandas dataframe named attendance.

The function then draws a rectangle around user's face and displays the student's name with the roll number along using OpenCV put text function.

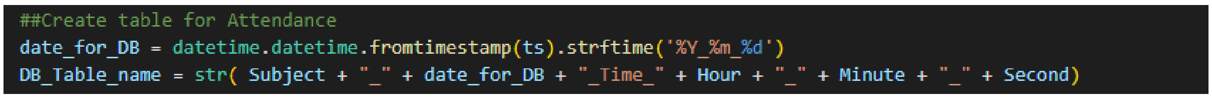


```
En = '15624031' + str(Id)
attendance.loc[len(attendance)] = [Id, aa, date, timeStamp]
cv2.rectangle(im, (x, y), (x + w, y + h), (0, 260, 0), 7)
cv2.putText(im, str(tt), (x + h, y), font, 1, (255, 255, 0,), 4)
```



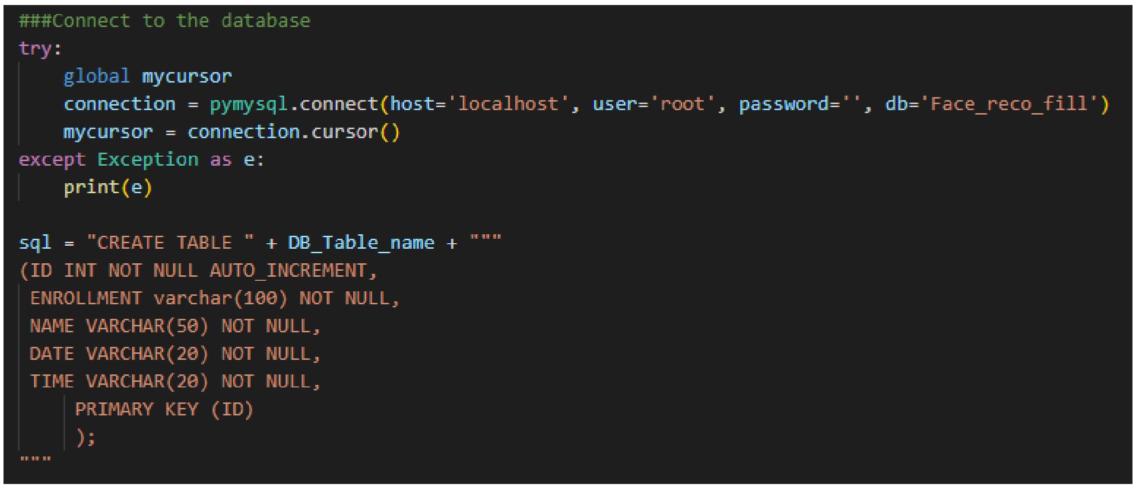
```
62.722132969853774
64.70146529934664
64.70146529934664
65.18344397114363
61.523090924477984
64.16306919281
64.16306919281
62.79605706008
62.79605706008
62.68522073229
67.96403564309
67.96403564309
65.91257508976
68.61036181064
68.63448522702
63.15064629340
63.15064629340
63.58300757378
62.38783461977
62.38783461977
65.21395232383
69.12687883391
68.87703109680
66.73653758804
68.56846956689
68.56846956689
63.50474152184
63.47378763945
64.70458136704
64.70458136704
65.68379122239
64.11524149972
65.44422396783
64.82323731985
64.82323731985
62.72086169691
64.53753206691
63.16499828847
63.16499828847
63.40003175332
65.37044160479
64.14409463219
64.66027353963
64.66027353963
64.87359937765
65.85530932551
65.63871303191
65.63871303191
65.29148594920
65.83523108283
65.83523108283
66.13586602095
66.7055556386304
65.56097377681992
66.05863084398489
66.05863084398489
65.25662622992819
65.73631265287166
65.73631265287166
65.74188892708725
67.76246426491402
66.92164523472147
66.92164523472147
66.19998306419728
65.89783902817518
```

The function continues capturing frames and recognizing faces until a timeout of 20 seconds or the user presses ‘escape’ key.



```
##Create table for Attendance
date_for_DB = datetime.datetime.fromtimestamp(ts).strftime('%Y_%m_%d')
DB_Table_name = str( Subject + "_" + date_for_DB + "_Time_" + Hour + "_" + Minute + "_" + Second)
```

This part of code creates a database for the attendance and marks it with timestamp.



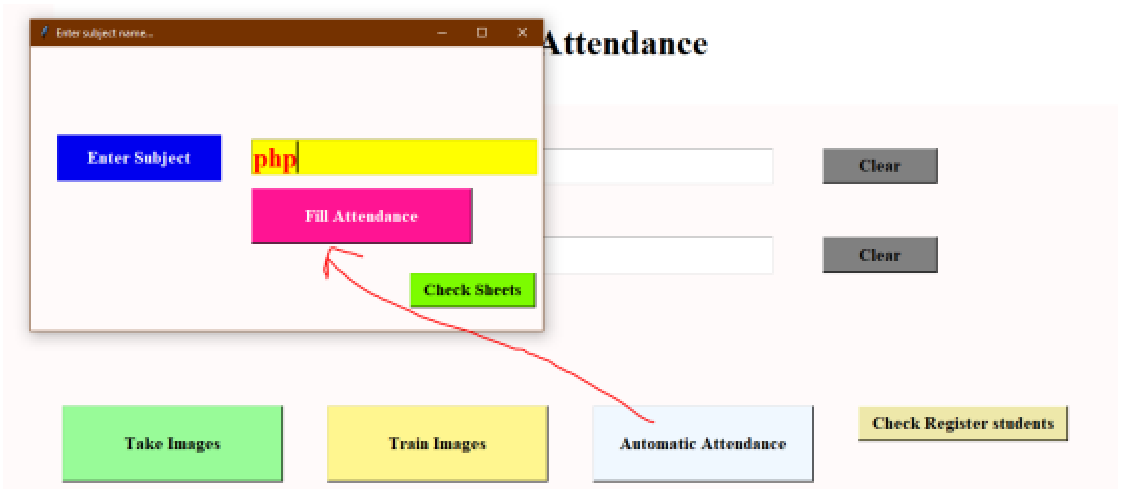
```
###Connect to the database
try:
    global mycursor
    connection = pymysql.connect(host='localhost', user='root', password='', db='Face_reco_fill')
    mycursor = connection.cursor()
except Exception as e:
    print(e)

sql = "CREATE TABLE " + DB_Table_name + """
(ID INT NOT NULL AUTO_INCREMENT,
 ENROLLMENT varchar(100) NOT NULL,
 NAME VARCHAR(50) NOT NULL,
 DATE VARCHAR(20) NOT NULL,
 TIME VARCHAR(20) NOT NULL,
     PRIMARY KEY (ID)
     );
"""
```

We create a database for mySQL, we establish connection to the database using pymysql library and create a table which is used to mark attendance.

```
###windo is frame for subject chooser
windo = tk.Tk()
windo.title("Enter subject name...")
windo.geometry('580x320')
windo.configure(background='snow')
Notifica = tk.Label(windo, text="Attendance filled Successfully", bg="Green", fg="white", width=33,
                        height=2, font=('times', 15, 'bold'))
```

This part of code is a python script that uses a tkinter library to create the GUI for system.

The GUI has subject chooser window and admin window
The subject chooser window is launched .



The window includes Enter Subject column and also “Fill Attendance” button and includes a check sheet button too.

```
def trainimg():
    recognizer = cv2.face.LBPHFaceRecognizer_create()
    global detector
    detector = cv2.CascadeClassifier("haarcascade_frontalface_default.xml")
    try:
        global faces,Id
        faces, Id = getImagesAndLabels("TrainingImage")
    except Exception as e:
        l='please make "TrainingImage" folder & put Images'
        Notification.configure(text=l, bg="SpringGreen3", width=50, font=('times', 18, 'bold'))
        Notification.place(x=350, y=400)
```

This part of code includes the training of model by the images stored and the training image folder which captures images of people and store there.

Result

After evaluating the performance of our model, if we observe that it is performing well on the training and validation data, we can confidently use this model to predict outcomes on new and unseen data. In other words, we can use this model for marking attendance in various fields.

Future Scope

The future scope of face recognition with attendance in AI is vast and promising. Here are some potential areas where this technology could be applied:

1. Security: Face recognition attendance systems can be used for security purposes, such as in access control systems or in law enforcement to identify suspects.

2. Education: Face recognition attendance systems can be used in schools and universities to track student attendance, reducing the administrative burden on teachers and improving accuracy.

3. Healthcare: Face recognition attendance systems can be used in hospitals and clinics to track staff attendance, ensuring that there are enough personnel to care for patients and reducing errors in payroll processing.

4. Retail: Face recognition attendance systems can be used in retail environments to track employee attendance and reduce time theft or fraud.

5. Events: Face recognition attendance systems can be used at events to manage attendance and provide better security and safety measures.

6. Marketing: Face recognition attendance systems can be used in marketing campaigns to track customer attendance and demographics, providing valuable insights for targeted marketing efforts.

7. Travel: Face recognition attendance systems can be used in airports and other travel hubs to track passenger attendance and reduce wait times.

Overall, the future scope of face recognition with attendance in AI is vast and has the potential to improve efficiency, accuracy, and security across a wide range of industries and applications.