

MINI PROJECT

STUDENT ATTENDANCE SYSTEM BASED ON THE FACE RECOGNITION OF WEBCAM'S IMAGE OF THE CLASSROOM

CSBS SEM V

ADITI GUPTA - 1914110284 - 05

ADITYA SINGH - 1914110286 - 07

SARTHAK GOEL - 1914110316 - 34

HARSHADA BHOSALE - 2014111517 - 68

CERTIFICATE

This is to certify that Aditi Gupta, Aditya Singh, Sarthak Goel, and Harshada Bhosale, students of Computer Science and Business Systems Semester 5 have successfully completed the Mini Project on 'STUDENT ATTENDANCE SYSTEM BASED ON THE FACE RECOGNITION OF WEBCAM'S IMAGE OF THE CLASSROOM' under the guidance of Prof. Manisha Kasar for the year 2021-2022.

Student's Signature

Teacher's Signature

ACKNOWLEDGEMENT

In the competitive, technocratic world, we often forget that technology can and should primarily be used for good. Advances in Artificial Intelligence, Machine Learning, and NLP have the potential to make life easier for millions of people belonging to various backgrounds and transform countless occupations for the better.

We were fortunate to be guided in this endeavor by Prof. Manisha Kasar, who helped us stay true to our goal .It is because of this guidance that we were able to complete our project.

We are indebted to our parents and would like to thank them and everyone else who supported us in the completion of this project. Finally, we hope to advance this project further and refine our model as humanly as possible.

TABLE OF CONTENTS

Page No.

1. Introduction	
2. Review of Literature	
3. Objective of the Project.....	
4. System Design.....	
5. Methodology for implementation.....	
(Formulation/Algorithm)	
6. Sample inputs and outputs.....	
7. Conclusion.....	

Appendix:- Program Source code with adequate comments.

1. INTRODUCTION

In today's networked world, the need to maintain the security of information or physical property is becoming both increasingly important and increasingly difficult. From time to time we hear about the crimes of credit card fraud, computer break-ins by hackers, or security breaches in a company or government building. In most of these crimes, the criminals were taking advantage of a fundamental flaw in the conventional access control systems: the systems do not grant access by "who we are", but by "what we have", such as ID cards, keys, passwords, PIN numbers, or mother's maiden name.

None of these means are really defined us. Recently, technology became available to allow verification of "true" individual identity. This technology is based in a field called "biometrics".

Biometric access control is an automated method of verifying or recognizing the identity of a living person on the basis of some physiological characteristics, such as fingerprints or facial features, or some aspects of the person's behavior, like his/her handwriting style or keystroke patterns. Since biometric systems identify a person by biological characteristics, they are difficult to forge. Face recognition is one of the few biometric methods that possess the merits of both high accuracy and low intrusiveness. It has the accuracy of a physiological approach without being intrusive. For this reason, since the early '70s (Kelly, 1970), face recognition has drawn the attention of researchers in fields from security, psychology, and image processing, to computer vision.

2. Review of Literature

Face recognition is one of the few biometric methods that possess the merits of both high accuracy and low intrusiveness. It has the accuracy of a physiological approach without being intrusive. Over the past 30 years, many researchers have proposed different face recognition techniques, motivated by the increased number of real

world applications requiring the recognition of human faces. There are several problems that make automatic face recognition a very difficult task. However, the face image of a person inputs to the database is usually acquired under different conditions. The importance of automatic face recognition is much cope with numerous variations of images of the same face due to changes in the following parameters such as :

- A. Pose
- B. Illumination
- C. Expression
- D. Motion
- E. Facial hair
- F. Glasses
- G. Background of image.

Face recognition technology is a well advance that can be applied for many commercial applications such as personal identification, security system, image-film processing, psychology, computer interaction, entertainment system, smart card, law enforcement, surveillance, and so on. Face recognition can be done in both a still image and video sequence which has its origin in still-image face recognition. Different approaches of face recognition for still images can be categorized into three main groups such as :

- I. Holistic approach
- II. Feature-based approach
- III. Hybrid approach product

- I. Holistic approach:- In a holistic approach or global feature, the whole face region is taken into account as input data into the face detection system. Examples of holistic methods are eigenfaces (most widely used method for face recognition), probabilistic eigenfaces, fisher faces, support vector machines, nearest feature lines (NFL) and independent-

component analysis approaches. They are all based on principal component analysis (PCA) techniques that can be used to simplify a dataset into a lower dimension while retaining the characteristics of the dataset.

- II. Feature-based approach:- In feature-based approaches or local features that are the features on the face such as nose, and then eyes are segmented and then used as input data for the structural classifier. Pure geometry, dynamic link architecture, and hidden Markov model methods belong to this category. One of the most successful of these systems is the Elastic Bunch Graph Matching (EBGM) system [40],[41], which is based on DLA.

Wavelets, especially Gabor wavelets, play a building block role for facial representation in these graph matching methods. A typical local feature representation consists of wavelet coefficients for different scales and rotations based on fixed wavelet bases. These locally estimated wavelet coefficients are robust to illumination change, translation, distortion, rotation, and scaling. The grid is appropriately positioned over the image and is stored with each grid point's locally determined jet in figure 2(a), and serves to represent the pattern classes. Recognition of a new image takes place by transforming the image into the grid of jets and matching all stored model graphs to the image. Confirmation of the DLA is done by establishing and dynamically modifying links between vertices in the model domain.

- III. Hybrid approach:- The idea of this method comes from how the human visual system perceives both holistic and local features. The key factors that influence the performance of the hybrid approach include how to determine which features should be combined and how to combine, so as to preserve

their advantages and avert their disadvantages at the same time.

These problems have a close relationship with the multiple classifier systems (MCS) and ensemble learning in the field of machine learning. Unfortunately, even in these fields, these problems remain unsolved. In spite of this, numerous efforts made in these fields indeed provide us some insights into solving these problems, and these lessons can be used as guidelines in designing a hybrid face recognition system. a hybrid approach that uses both holistic and local information for recognition may be an effective way to reduce the complexity of classifiers and improve their generalization capability.

3. Objective of the Project

To implement a Student attendance system based on the face recognition of the webcam's image in the classroom.

4. System Design

A throughout survey has revealed that various methods and combinations of these methods can be applied in the development of a new face recognition system. Among the many possible approaches, we have decided to use a combination of knowledge-based methods for the face detection part and a neural network approach for the face recognition part. The main reason for this selection is their smooth applicability and reliability issues. Our face recognition system approach is given in Figure

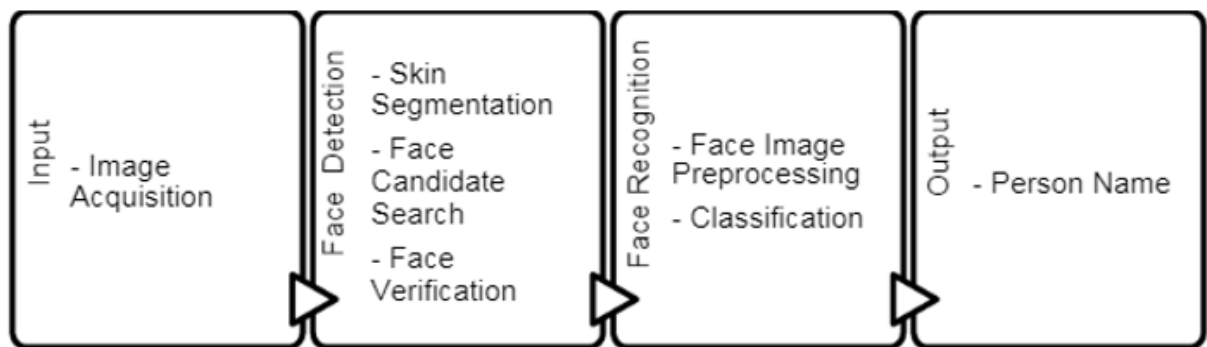


Figure 2 Face Recognition Approach

4.1. Input Part

The input part is a prerequisite for the face recognition system. Image acquisition operation is performed in this part. Live captured images are converted to digital data for performing image-processing computations. These captured images are sent to a face detection algorithm.

4.2. Face Detection Part

Face detection performs locating and extracting face image operations for the face recognition system. The face detection part algorithm is given in the figure given below.

Our experiments reveal that skin segmentation, as the first step for face detection, reduces the computational time for searching the whole image. While segmentation is applied, only the segmented region is searched whether the segment includes any face or not.

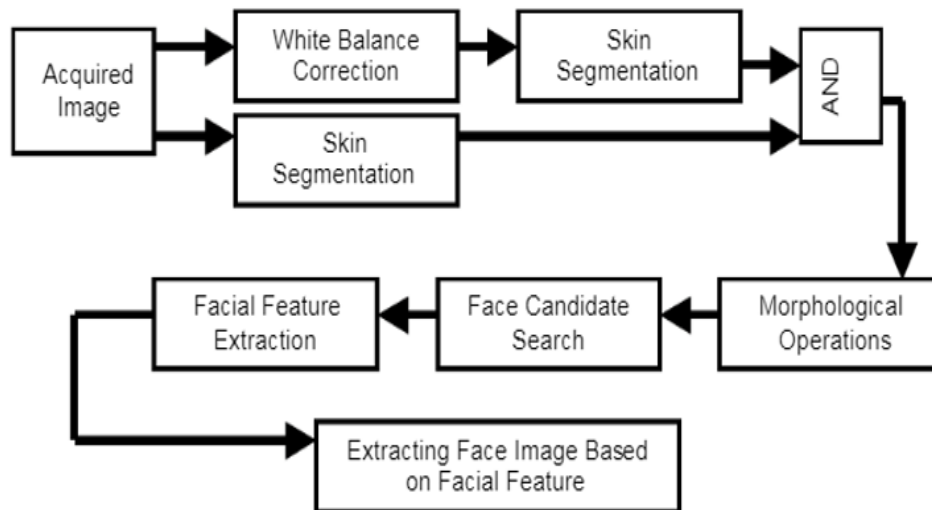


Figure 3 Algorithm of Face Detection Part

For this reason, skin segmentation is applied as the first step of the detection part. RGB color space is used to describe skin-like color [4]. The white balance of images differs due to change in the lighting conditions of the environment while acquiring images. This situation creates non-skin objects that belong to skin objects.

Therefore, the white balance of the acquired image should be corrected before segmenting it [18]. Results of segmentation on the original image and white balance corrected image is given in Figure 4 and 5.

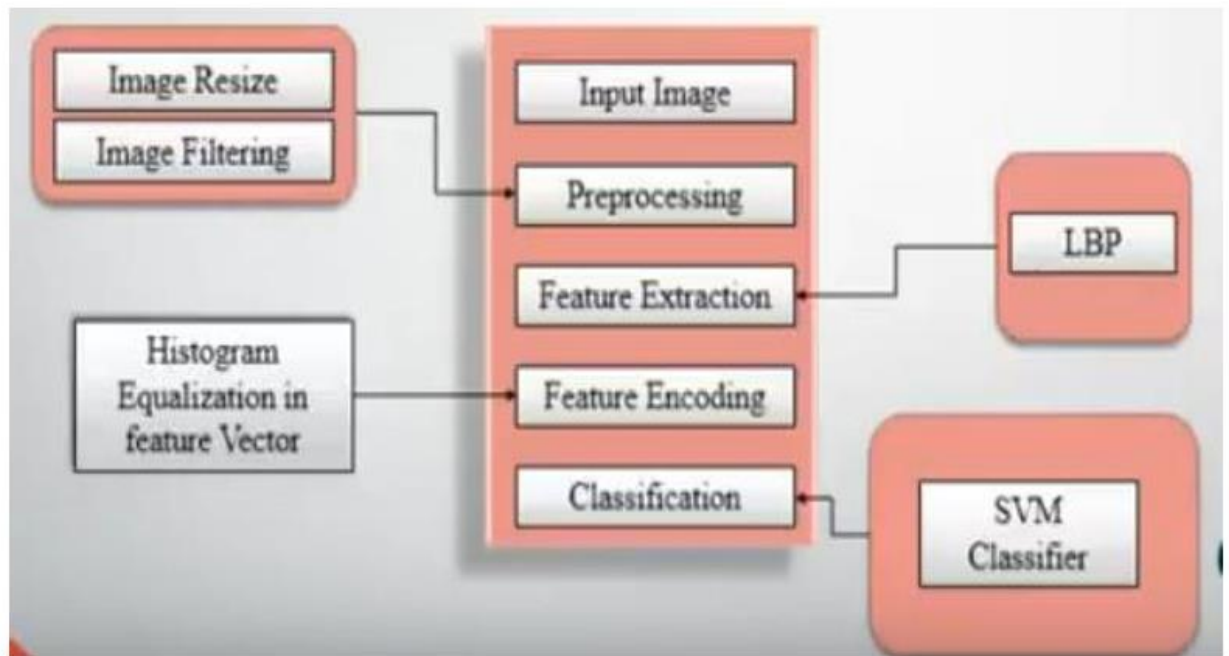
After “and operation” is applied on segmented images, some morphological operations are applied on the final skin image to search face candidates. Noisy like small regions

elimination, closing operations are performed. Then, face candidates are chosen with two conditions which are the ratio of the bounding box of candidates and covering some gaps inside the candidate region. The ratio of the bounding box should lie between 0.3 and 1.5

Based on these conditions, face candidates are extracted from the input images with modified bounding boxes from the original bounding box. The height of the bounding box was modified as 1.28 times bigger than the width of the bounding box because chest and neck parts will be eliminated if the candidate includes them. This modification value has been determined experimentally.

These face candidates will be sent to the facial feature extraction part to validate the candidates. Final verification of candidate and face image extraction, the facial feature extraction process is applied. The facial feature is one of the most significant features of the face. Facial features are eyebrows, eyes, mouth, nose, nose tip, cheek, etc. The property is used to extract the eyes and mouth which, two eyes and mouth generate isosceles triangle, and the distance between eye to eye and midpoint of eyes distance to mouth is equal [2]. Laplacian of Gaussian (LoG) filter and some other filtering operations are performed to extract facial features of face candidates.

5. Methodology for implementation (Formulation/Algorithm)



A similar separation of pattern recognition algorithms into four groups is proposed by Jain and colleges. We can group face recognition methods into three main groups. The following approaches are proposed: ^

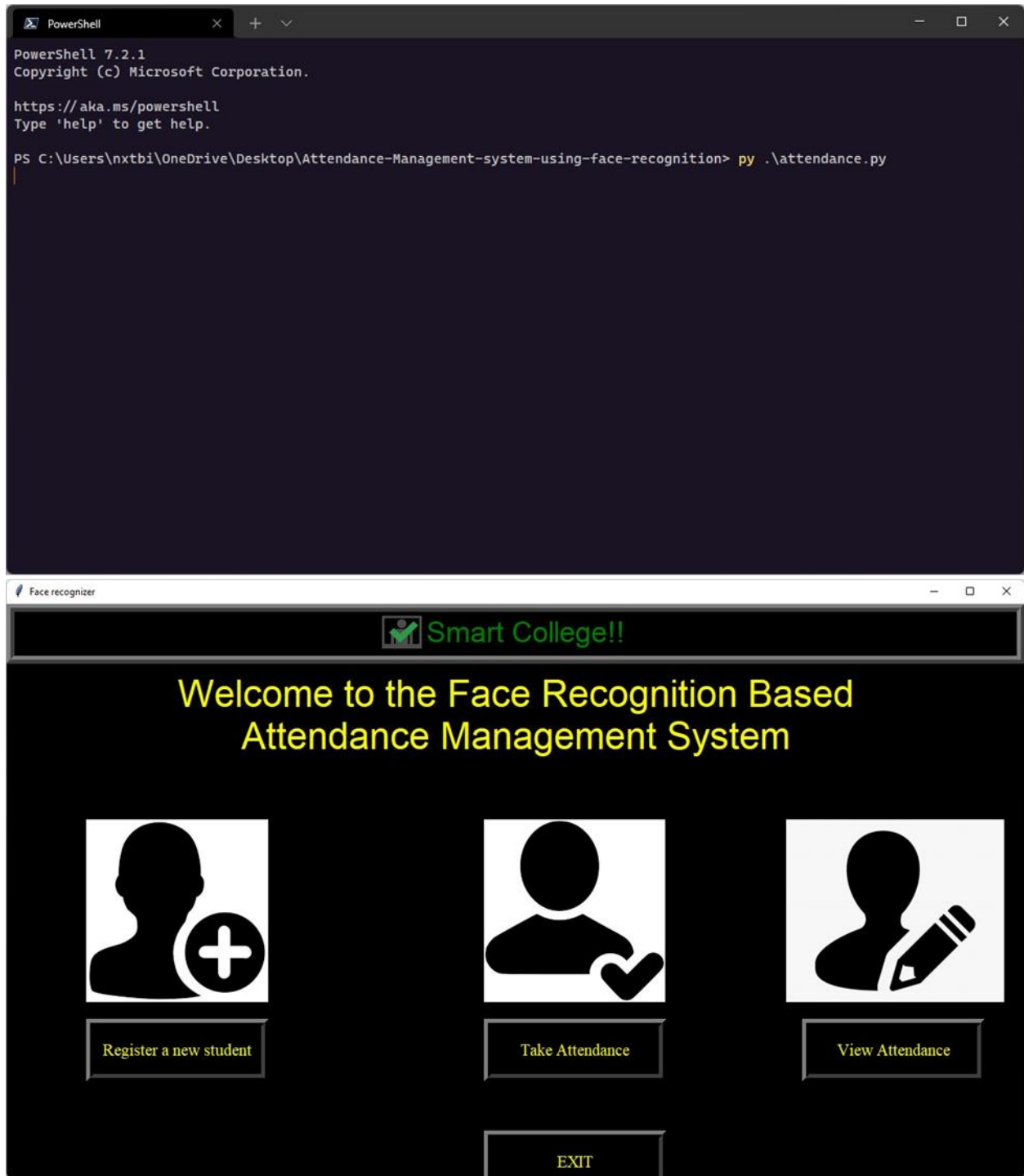
Template matching:- Patterns are represented by samples, models, pixels, curves, textures. The recognition function is usually a correlation or distance measure. ^

Statistical approach:- Patterns are represented as features. The recognition function is a discriminant function. ^

Neural networks. The representation may vary. There is a network function in some point.

Note that many algorithms, mostly current complex algorithms, may fall into more than one of these categories. The most relevant face recognition algorithms will be discussed later under this classification.

6. SAMPLE INPUTS AND OUTPUTS



Take Student Image..

Register Your Face

Enter the details

Enrollment No	1
Name	Adi
Notification	

Take Image Train Image

Take Student Image..

Register Your Face

Enter the details

Enrollment No	
Name	
Notification	Images Saved for ER No:1 Name:Adi

Take Image Train Image

Take Student Image..

Register Your Face

Enter the details

Enrollment No	<input type="text"/>
Name	<input type="text"/>
Notification	Image Trained successfully

Take Image

Train Image

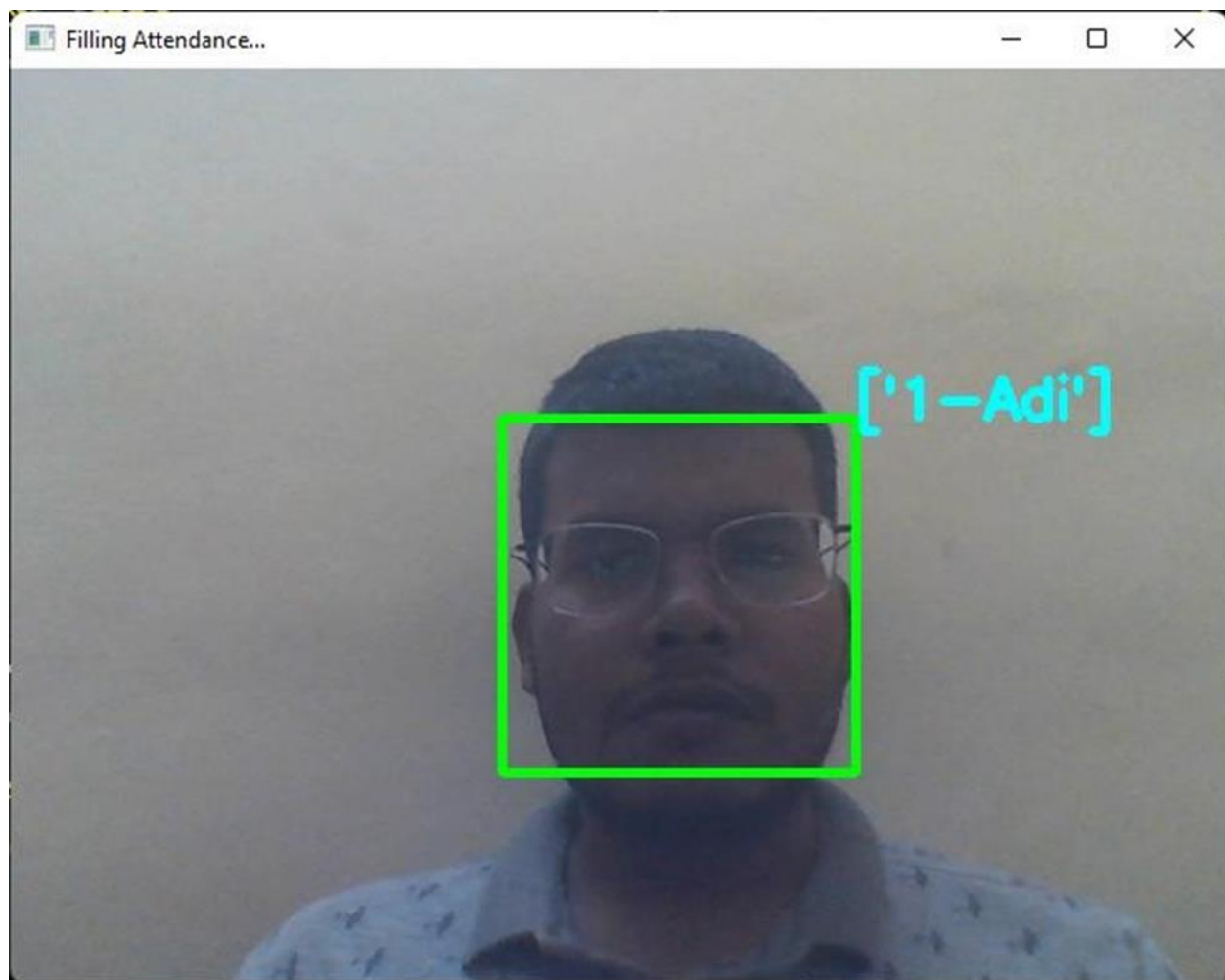
Subject...

Enter the Subject Name

Enter Subject	<input type="text" value="ml"/>
---------------	---------------------------------

Fill Attendance

Check Sheets



Attendance of ml		
Enrollment	Name	2022-01-05
1	['Adi']	1

Subject...

Enter the Subject Name

Enter Subject **ml**

Fill Attendance Check Sheets

Attendance Filled Successfully of ml

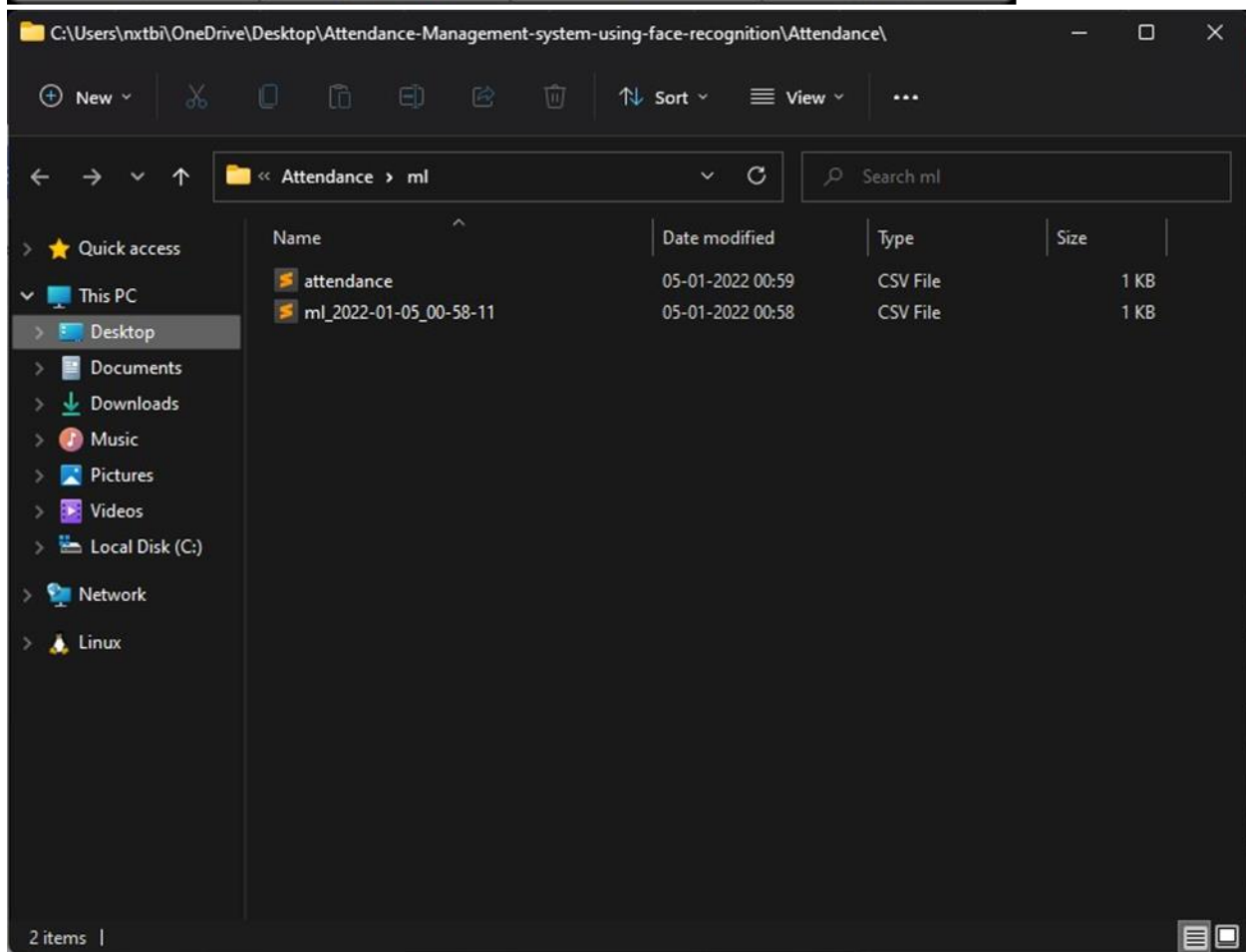
Subject...

Which Subject of Attendance?

Enter Subject **ml**

View Attendance Check Sheets

Attendance of ml			
Enrollment	Name	2022-01-05	Attendance
1	['Adi']	1	100%



Attendance.py

```
1 import tkinter as tk
2 from tkinter import *
3 import os, cv2
4 import shutil
5 import csv
6 import numpy as np
7 from PIL import ImageTk, Image
8 import pandas as pd
9 import datetime
10 import time
11 import tkinter.font as font
12 import pyttsx3
13
14 # project module
15 import show_attendance
16 import takeImage
17 import trainImage
18 import automaticAttendance
19
20 # engine = pyttsx3.init()
21 # engine.say("Welcome!")
22 # engine.say("Please browse through your options..")
23 # engine.runAndWait()
24
25
26 def test_text_to_speech(user_text):
27     engine = pyttsx3.init()
28     engine.say(user_text)
29     engine.runAndWait()
30
31
32 haarcascade_path = "C:\\Users\\nxtbi\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\haarcascade_frontalface_default.xml"
33 trainImageLabel_path = {
34     "C:\\Users\\nxtbi\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\TrainingImageLabel\\Trainer.yml"
35 }
36 trainImage_path = "C:\\Users\\nxtbi\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\TrainingImage"
37 studentdetail_path = {
38     "C:\\Users\\nxtbi\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\StudentDetails\\studentdetails.csv"
39 }
40 attendance_path = "C:\\Users\\nxtbi\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\Attendance"
41
42
43 window = Tk()
44 window.title("Face recognizer")
45 window.geometry("1280x720")
46 dialog_title = "QUIT"
47 dialog_text = "Are you sure want to close?"
48 window.config(background="black")
49
50
51 # to destroy screen
52 def del_scl():
53     scl.destroy()
54
55
56 # error message for name and no
57 def err_screen():
58     global scl
59     scl = Tk()
60     scl.geometry("400x100")
61     scl.config(background="black")
62     scl.title("Warning!")
63     scl.config(background="black")
64     scl.resizable(0, 0)
65     tk.Label(
66         scl,
67         text="Enrollment & Name required!!!",
68         fg="yellow",
69         bg="black",
70         font=("times", 20, "bold"),
71     ).pack()
72     tk.Button(
73         scl,
74         text="OK",
75         command=del_scl,
76         fg="yellow",
77         bg="black",
78         width=9,
79         height=1,
80         activebackground="red",
81         font=("times", 20, "bold"),
82     ).place(x=110, y=10)
83
84
85 def testVal(inStr, acttyp):
86     if acttyp == "1": # insert
87         if not inStr.isdigit():
88             return False
89         return True
90
91
92 logo = Image.open("UI_Image\\NMI.png")
93 logo = logo.resize((50, 42), Image.ANTIALIAS)
94 logo1 = ImageTk.PhotoImage(logo)
95 titl = tk.Label(window, bg="black", relief=RIDGE, bd=10, font=("arial", 35))
96 titl.pack(fill=X)
97 ti = tk.Label(window, image=logo1, bg="black",)
98 ti.place(x=470, y=10)
99
100 titl = tk.Label(
101     window,
102     text="Smart College!!",
103     bg="black",
104     fg="green",
105     font=("arial", 27),
106 )
107 titl.place(x=525, y=12)
108
109 a = tk.Label(
110     window,
111     text="Welcome to the Face Recognition Based Attendance Management System",
112     bg="black",
113     fg="yellow",
114     bd=10,
115 )
116
117
118 # error message for name and no
119 def err_screen():
120     global scl
121     scl = Tk()
122     scl.geometry("400x100")
123     scl.config(background="black")
124     scl.title("Warning!")
125     scl.config(background="black")
126     scl.resizable(0, 0)
127     tk.Label(
128         scl,
129         text="Enrollment & Name required!!!",
130         fg="yellow",
131         bg="black",
132         font=("times", 20, "bold"),
133     ).pack()
134     tk.Button(
135         scl,
136         text="OK",
137         command=del_scl,
138         fg="yellow",
139         bg="black",
140         width=9,
141         height=1,
142         activebackground="red",
143         font=("times", 20, "bold"),
144     ).place(x=110, y=10)
145
146
147 def testVal(inStr, acttyp):
148     if acttyp == "1": # insert
149         if not inStr.isdigit():
150             return False
151         return True
152
153
154 logo = Image.open("UI_Image\\NMI.png")
155 logo = logo.resize((50, 42), Image.ANTIALIAS)
156 logo1 = ImageTk.PhotoImage(logo)
157 titl = tk.Label(window, bg="black", relief=RIDGE, bd=10, font=("arial", 35))
158 titl.pack(fill=X)
159 ti = tk.Label(window, image=logo1, bg="black",)
160 ti.place(x=470, y=10)
161
162 titl = tk.Label(
163     window,
164     text="Smart College!!",
165     bg="black",
166     fg="green",
167     font=("arial", 27),
168 )
169 titl.place(x=525, y=12)
170
171 a = tk.Label(
172     window,
173     text="Welcome to the Face Recognition Based Attendance Management System",
174     bg="black",
175     fg="yellow",
176     bd=10,
177 )
```

```
111 font=("arial", 35),
112 )
113 a.pack()
114
115 r1 = Image.open("UI_Image/register.png")
116 r = ImageTk.PhotoImage(r1)
117 label1 = Label(window, image=r)
118 label1.image = r
119 label1.place(x=100, y=270)
120
121 a1 = Image.open("UI_Image/attendance.png")
122 a = ImageTk.PhotoImage(a1)
123 label2 = Label(window, image=a)
124 label2.image = a
125 label2.place(x=500, y=270)
126
127 v1 = Image.open("UI_Image/verify.png")
128 v = ImageTk.PhotoImage(v1)
129 label3 = Label(window, image=v)
130 label3.image = v
131 label3.place(x=600, y=270)
132
133
134 def TakeImageUI():
135     imageUI = Tk()
136     imageUI.title("Take Student Image..")
137     imageUI.geometry("780x480")
138     imageUI.configure(background="black")
139     imageUI.resizable(0, 0)
140     titl = tk.Label(imageUI, bg="black", relief=RIDGE, bd=10, font=("arial", 35))
141     titl.pack(fill=X)
142     # image and title
143     titl = tk.Label(
144         imageUI, text="Register Your Face", bg="black", fg="green", font=("arial", 30),
145     )
146     titl.place(x=270, y=12)
147
148     # heading
149     a = tk.Label(
150         imageUI,
151         text="Enter the details",
152         bg="black",
153         fg="yellow",
154         bd=10,
155         font=("arial", 24),
156     )
157     a.place(x=280, y=75)
158
159     # ID no
160     lb11 = tk.Label(
161         imageUI,
162         text="Enrollment No",
163         width=10,
164         height=2,
165         bg="black",
166         fg="yellow",
167         bd=5,
168         relief=RIDGE,
169         font=("times new roman", 12),
170     )
171     lb11.place(x=120, y=130)
172     txt1 = tk.Entry(
173         imageUI,
174         width=17,
175         bd=5,
176         validate="key",
177         bg="black",
178         fg="yellow",
179         relief=RIDGE,
180         font=("times", 25, "bold"),
181     )
182     txt1.place(x=250, y=130)
183     txt1["validatecommand"] = (txt1.register(testVal), "%P", "%d")
184
185     # name
186     lb12 = tk.Label(
187         imageUI,
188         text="Name",
189         width=10,
190         height=2,
191         bg="black",
192         fg="yellow",
193         bd=5,
194         relief=RIDGE,
195         font=("times new roman", 12),
196     )
197     lb12.place(x=120, y=200)
198     txt2 = tk.Entry(
199         imageUI,
200         width=17,
201         bd=5,
202         bg="black",
203         fg="yellow",
204         relief=RIDGE,
205         font=("times", 25, "bold"),
206     )
207     txt2.place(x=250, y=200)
208
209     lb13 = tk.Label(
210         imageUI,
211         text="Notification",
212         width=10,
213         height=2,
214         bg="black",
215         fg="yellow",
216         bd=5,
217         relief=RIDGE,
218         font=("times new roman", 12),
219     )
220     lb13.place(x=120, y=270)
```

```
221 message = tk.Label(  
222     imageObj,  
223     text="",  
224     width=32,  
225     height=2,  
226     bg="black",  
227     fg="yellow",  
228     relief=RIDGE,  
229     font=("times", 12, "bold"),  
230 )  
231 message.place(x=250, y=220)  
232  
233  
234  
235  
236 def take_image():  
237     t1 = txt1.get()  
238     t2 = txt2.get()  
239     takeImage.takeImage(  
240         t1,  
241         t2,  
242         haarcascade_path,  
243         trainImage_path,  
244         message,  
245         err_screen,  
246         text_to_speech,  
247     )  
248     txt1.delete(0, "end")  
249     txt2.delete(0, "end")  
250  
251 # take Image button  
252 # image  
253 takeImg = tk.Button(  
254     imageObj,  
255     text="take Image",  
256     command=take_image,  
257     bd=10,  
258     font=("times new roman", 10),  
259     bg="black",  
260     fg="yellow",  
261     height=2,  
262     width=12,  
263     relief=RIDGE,  
264 )  
265 takeImg.place(x=130, y=350)  
266  
267 def train_image():  
268     trainImage.trainImage(  
269         haarcascade_path,  
270         trainImage_path,  
271         trainImageLabel_path,  
272         message,  
273         text_to_speech,  
274     )  
275 # train Image function call  
276 trainImg = tk.Button(  
277     imageObj,  
278     text="train Image",  
279     command=train_image,  
280     bd=10,  
281     font=("times new roman", 10),  
282     bg="black",  
283     fg="yellow",  
284     height=2,  
285     width=12,  
286     relief=RIDGE,  
287 )  
288 trainImg.place(x=360, y=350)  
289  
290  
291 r = tk.Button(  
292     window,  
293     text="Register a new student",  
294     command=takeImageObj,  
295     bd=10,  
296     font=("times new roman", 16),  
297     bg="black",  
298     fg="yellow",  
299     height=2,  
300     width=17,  
301 )  
302 r.place(x=100, y=520)  
303  
304  
305 def automatic_attendance():  
306     automaticAttendance.subjectChoose(text_to_speech)  
307  
308  
309 r = tk.Button(  
310     window,  
311     text="Take Attendance",  
312     command=automatic_attendance,  
313     bd=10,  
314     font=("times new roman", 16),  
315     bg="black",  
316     fg="yellow",  
317     height=2,  
318     width=17,  
319 )  
320 r.place(x=600, y=520)  
321  
322  
323 def view_attendance():  
324     show_attendance.subjectChoose(text_to_speech)  
325  
326  
327 r = tk.Button(  
328     window,  
329     text="View Attendance",  
330     command=view_attendance,  
331     bd=10,  
332     font=("times new roman", 16),  
333     bg="black",  
334     fg="yellow",  
335     height=2,  
336     width=17,  
337 )  
338 r.place(x=100, y=620)  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000
```

```
275 # train Image function call  
276 trainImg = tk.Button(  
277     imageObj,  
278     text="train Image",  
279     command=train_image,  
280     bd=10,  
281     font=("times new roman", 10),  
282     bg="black",  
283     fg="yellow",  
284     height=2,  
285     width=12,  
286     relief=RIDGE,  
287 )  
288 trainImg.place(x=360, y=350)  
289  
290  
291 r = tk.Button(  
292     window,  
293     text="Register a new student",  
294     command=takeImageObj,  
295     bd=10,  
296     font=("times new roman", 16),  
297     bg="black",  
298     fg="yellow",  
299     height=2,  
300     width=17,  
301 )  
302 r.place(x=100, y=520)  
303  
304  
305 def automatic_attendance():  
306     automaticAttendance.subjectChoose(text_to_speech)  
307  
308  
309 r = tk.Button(  
310     window,  
311     text="Take Attendance",  
312     command=automatic_attendance,  
313     bd=10,  
314     font=("times new roman", 16),  
315     bg="black",  
316     fg="yellow",  
317     height=2,  
318     width=17,  
319 )  
320 r.place(x=600, y=520)  
321  
322  
323 def view_attendance():  
324     show_attendance.subjectChoose(text_to_speech)  
325  
326  
327 r = tk.Button(  
328     window,  
329     text="View Attendance",  
330     command=view_attendance,  
331     bd=10,  
332     font=("times new roman", 16),  
333     bg="black",  
334     fg="yellow",  
335     height=2,  
336     width=17,  
337 )  
338 r.place(x=100, y=620)  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000
```

```
110 window,
111 text="Take Attendance",
112 command=automatic_attendance,
113 bd=10,
114 font=("times new roman", 16),
115 bg="black",
116 fg="yellow",
117 height=2,
118 width=17,
119 )
120 r.place(x=600, y=520)
121
122
123 def view_attendance():
124     show_attendance.subjectchose(text_to_speech)
125
126
127 r = tk.Button(
128     window,
129     text="View Attendance",
130     command=view_attendance,
131     bd=10,
132     font=("times new roman", 16),
133     bg="black",
134     fg="yellow",
135     height=2,
136     width=17,
137 )
138 r.place(x=1000, y=520)
139
140 r = tk.Button(
141     window,
142     text="EXIT",
143     bd=10,
144     command=quit,
145     font=("times new roman", 16),
146     bg="black",
147     fg="yellow",
148     height=2,
149     width=17,
150 )
151 r.place(x=600, y=660)
152 window.mainloop()
153
```

automaticAttendance.py

```
C:\Users\mishra\Desktop\Attendance-Management-system-using-face-recognition\automaticAttendance.py - Sublime Text (UNREGISTERED)
1 from tkinter import *
2 import os, cv2
3 import shutil
4 import csv
5 import numpy as np
6 from PIL import ImageTk, Image
7 import pandas as pd
8 import datetime
9 import time
10 import tkinter.ttk as ttk
11 import tkinter.font as font
12
13
14 haarcascade_path = "C:\\Users\\mishra\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\haarcascade_frontalface_default.xml"
15 trainimage_label_path = {
16     "C:\\Users\\mishra\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\TrainingImageLabel\\trainer.yml"
17 }
18 trainimage_path = "C:\\Users\\mishra\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\TrainingImage"
19 studentdetail_path = {
20     "C:\\Users\\mishra\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\StudentDetails\\studentdetails.csv"
21 }
22 attendance_path = "C:\\Users\\mishra\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\Attendance"
23 # for choose subject and fill attendance
24 def subject_choose(text_to_speech):
25     def fillAttendance():
26         sub = tk.get()
27         now = time.time()
28         future = now + 20
29         print(now)
30         print(future)
31         if sub == "1":
32             t = "Please enter the subject name!!!"
33             text_to_speech(t)
34         else:
35             try:
36                 recognizer = cv2.face.LBPHFaceRecognizer_create()
37                 recognizer.read(trainimage_label_path)
38             except:
39                 e = "Model not found, please train model"
40                 Notification.configure(
41                     text=e,
42                     bg="black",
43                     fg="yellow",
44                     width=33,
45                     font=("times", 15, "bold"),
46                 )
47                 Notification.place(x=20, y=250)
48                 text_to_speech(e)
49                 faceCascade = cv2.CascadeClassifier(haarcascade_path)
50                 df = pd.read_csv(studentdetail_path)
51                 cam = cv2.VideoCapture(0, cv2.CAP_DSHOW) # capture device = camera
52                 font = cv2.FONT_HERSHEY_SIMPLEX
53                 col_names = ["enrollment", "name"]
54                 attendance = pd.DataFrame(columns=col_names)
55
56
57 attendance = pd.DataFrame(columns=col_names)
58 while True:
59     im = cam.read()
60     gray = cv2.cvtColor(im, cv2.COLOR_BGR2GRAY)
61     faces = faceCascade.detectMultiScale(gray, 1.2, 5)
62     for (x, y, w, h) in faces:
63         global Id
64         Id, conf = recognizer.predict(gray[y:y+h, x:x+w])
65         if conf < 70:
66             print(conf)
67             global Subject
68             global aa
69             global date
70             global timeStamp
71             Subject = tk.get()
72             ts = time.time()
73             date = datetime.datetime.fromtimestamp(ts).strftime(
74                 "%Y-%m-%d"
75             )
76             timeStamp = datetime.datetime.fromtimestamp(ts).strftime(
77                 "%H:%M:%S"
78             )
79             aa = df.loc[df["Enrollment"] == Id][["Name"]].values
80             tt = str(Id) + "-" + aa
81             # Im = 1080x1600 str(Id)
82             attendance.loc[len(attendance)] = [
83                 Id,
84                 aa,
85             ]
86             cv2.rectangle(im, (x, y), (x + w, y + h), (0, 255, 0), 4)
87             cv2.putText(
88                 im, str(tt), (x + h, y), font, 1, (255, 255, 0), 4
89             )
90         else:
91             Id = "Unknown"
92             tt = str(Id)
93             cv2.rectangle(im, (x, y), (x + w, y + h), (0, 25, 255), 2)
94             cv2.putText(
95                 im, str(tt), (x + h, y), font, 1, (0, 25, 255), 4
96             )
97         if time.time() > future:
98             break
99
100 attendance = attendance.drop_duplicates(
101     ["Enrollment"], keep="first"
102 )
103 cv2.imshow("Filling Attendance...", im)
104 key = cv2.waitKey(30) & 0xFF
105 if key == 27:
106     break
107
108 ts = time.time()
109 print(aa)
110 # attendance["date"] = date
```

```
110 # attendance["date"] = date
111 # attendance["Attendance"] = "P"
112 attendance[date] = 1
113 date = datetime.datetime.fromtimestamp(ts).strftime("%Y-%m-%d")
114 timeStamp = datetime.datetime.fromtimestamp(ts).strftime("%H:%M:%S")
115 Hour, Minute, Second = timeStamp.split(":")
116 # filename = "Attendance" + Subject + ".csv"
117 path = os.path.join(attendance_path, Subject)
118 filename = {
119     f"{path}/"
120     + Subject
121     + "-"
122     + date
123     + "-"
124     + Hour
125     + "-"
126     + Minute
127     + "-"
128     + Second
129     + ".csv"
130 }
131 attendance = attendance.drop_duplicates(["Enrollment"], keep="first")
132 print(attendance)
133 attendance.to_csv(filename, index=False)
134
135 # = "Attendance Filled Successfully of " + Subject
136 Notifica.configure(
137     text="",
138     bg="black",
139     fg="yellow",
140     width=33,
141     relief=RIDGE,
142     bd=5,
143     font=("times", 15, "bold"),
144 )
145 text_to_speech(m)
146
147 Notifica.place(x=20, y=250)
148
149 cam.release()
150 cv2.destroyAllWindows()
151
152 import csv
153 import tkinter
154
155 root = tkinter.Tk()
156 root.title("Attendance of " + Subject)
157 root.configure(background="black")
158 cs = os.path.join(path, filename)
159 print(cs)
160 with open(cs, newline="") as file:
161     reader = csv.reader(file)
162     r = 0
163     for col in reader:
164         r = r + 1
165
166 for col in reader:
167     c = 0
168     for row in col:
169         label = tkinter.Label(
170             root,
171             width=10,
172             height=1,
173             fg="yellow",
174             font=("times", 15, "bold"),
175             bg="black",
176             text=row,
177             relief=tkinter.RIDGE,
178         )
179         label.grid(row=r, column=c)
180         c = c + 1
181         r = r + 1
182     root.mainloop()
183     print(attendance)
184 except:
185     f = "No Face found for attendance"
186     text_to_speech(f)
187     cv2.destroyAllWindows()
188
189 ###window is frame for subject chooser
190 subject = Tk()
191 # window.iconbitmap("AMS.ico")
192 subject.title("Subject...")
193 subject.geometry("500x120")
194 subject.resizable(0, 0)
195 subject.configure(background="black")
196 # subject_logo = Image.open("UI_Image/0004.png")
197 # subject_logo = subject_logo.resize((50, 47), Image.ANTIALIAS)
198 # subject_logo = ImageTk.PhotoImage(subject_logo)
199 titl = tk.Label(subject, bg="black", relief=RIDGE, bd=10, font=("arial", 30))
200 titl.pack(fill=X)
201 # l1 = tk.Label(subject, image=subject_logo, bg="black",)
202 # l1.place(x=100, y=10)
203 titl = tk.Label(
204     subject,
205     text="Enter the Subject Name",
206     bg="black",
207     fg="green",
208     font=("arial", 25),
209 )
210 titl.place(x=100, y=12)
211 Notifica = tk.Label(
212     subject,
213     text="Attendance Filled Successfully",
214     bg="yellow",
215     fg="black",
216     width=33,
217     height=2,
218     font=("times", 15, "bold"),
219 )
220
```



```
def Attf():
    sub = tk.get()
    if sub == "":
        t = "Please enter the subject name!!!"
        text_to_speak(t)
    else:
        os.startfile(
            f"C:\\Users\\nrbh1\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\Attendance\\{sub}"
        )

    attf = tk.Button(
        subject,
        text="Check Sheets",
        command=Attf,
        bd=7,
        font=("times new roman", 15),
        bg="black",
        fg="yellow",
        height=2,
        width=10,
        relief=RIDGE,
    )
    attf.place(x=300, y=170)

    sub = tk.Label(
        subject,
        text="Enter Subject",
        width=10,
        height=2,
        bg="black",
        fg="yellow",
        bd=5,
        relief=RIDGE,
        font=("times new roman", 15),
    )
    sub.place(x=50, y=100)

    tx = tk.Entry(
        subject,
        width=15,
        bd=5,
        bg="black",
        fg="yellow",
        relief=RIDGE,
        font=("times", 30, "bold"),
    )
    tx.place(x=190, y=100)

    fill_a = tk.Button(
        subject,
        text="Fill Attendance",
        command=FillAttendance,
        bd=7,
        font=("times new roman", 15),
        bg="black",
    )

    height=2,
    bg="black",
    fg="yellow",
    bd=5,
    relief=RIDGE,
    font=("times new roman", 15),
    )
    sub.place(x=50, y=100)

    tx = tk.Entry(
        subject,
        width=15,
        bd=5,
        bg="black",
        fg="yellow",
        relief=RIDGE,
        font=("times", 30, "bold"),
    )
    tx.place(x=190, y=100)

    fill_a = tk.Button(
        subject,
        text="Fill Attendance",
        command=FillAttendance,
        bd=7,
        font=("times new roman", 15),
        bg="black",
        fg="yellow",
        height=2,
        width=12,
        relief=RIDGE,
    )
    fill_a.place(x=195, y=170)
    subject.mainloop()
```

show_attendance.py

```
56     r += 1
57     root.mainloop()
58     print(newdf)
59
60     subject = tk()
61     # window geometry ("APP_TITLE")
62     subject.title("Subject...")
63     # subject geometry ("500x300")
64     subject.geometry("500x300")
65     # subject resizable (0, 0)
66     subject.resizable(0, 0)
67     # subject logo = image.open("UI_Image/0004.png")
68     # subject_logo = subject_logo.resize((50, 47), Image.ANTIALIAS)
69     # subject_logo = ImageTk.PhotoImage(subject_logo)
70     title = tk.Label(subject, bg="black", relief=RIDGE, bd=10, font=("arial", 30))
71     title.pack(fill=X)
72     # l1 = tk.Label(subject, image=subject_logo, bg="black",)
73     # l1.place(x=100, y=10)
74     title = tk.Label(subject,
75                     text="Which Subject of Attendance?",
76                     bg="black",
77                     fg="green",
78                     font=("arial", 25),
79                     )
80     title.place(x=100, y=12)
81
82     def Attf():
83         sub = tx.get()
84         if sub == "":
85             t = "Please enter the subject name!!!"
86             text_to_speech(t)
87         else:
88             os.startfile(
89                 f"C:\\Users\\nithi\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\Attendance\\{sub}"
90             )
91
92     attf = tk.Button(
93         subject,
94         text="Check Sheets",
95         command=Attf,
96         bd=7,
97         font=("times new roman", 15),
98         bg="black",
99         fg="yellow",
100         height=2,
101         width=10,
102         relief=RIDGE,
103     )
104     attf.place(x=300, y=170)
105
106     sub = tk.Label(
107         subject,
108         text="Enter Subject",
109         width=10,
110         height=1,
111     )
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
262
```

```
117         bd=7,
118         font=("times new roman", 15),
119         bg="black",
120         fg="yellow",
121         height=2,
122         width=10,
123         relief=RIDGE,
124     )
125     attf.place(x=360, y=170)
126
127     sub = tk.Label(
128         subject,
129         text="Enter Subject",
130         width=10,
131         height=2,
132         bg="black",
133         fg="yellow",
134         bd=5,
135         relief=RIDGE,
136         font=("times new roman", 15),
137     )
138     sub.place(x=50, y=100)
139
140     tx = tk.Entry(
141         subject,
142         width=15,
143         bd=5,
144         bg="black",
145         fg="yellow",
146         relief=RIDGE,
147         font=("times", 30, "bold"),
148     )
149     tx.place(x=190, y=100)
150
151     fill_a = tk.Button(
152         subject,
153         text="View Attendance",
154         command=calculate_attendance,
155         bd=7,
156         font=("times new roman", 15),
157         bg="black",
158         fg="yellow",
159         height=2,
160         width=12,
161         relief=RIDGE,
162     )
163     fill_a.place(x=195, y=170)
164     subject.mainloop()
165
```

takelImage.py

```
1 import cv
2 import cv2
3 import numpy as np
4 import pandas as pd
5 import datetime
6 import time
7
8
9
10 # take image of user
11 def TakeImage(l1, l2, haarcascade_path, trainimage_path, message, err_screen, text_to_speech):
12     if (l1 == "") and (l2 == ""):
13         t="Please Enter the your Enrollment Number and Name."
14         text_to_speech(t)
15     elif l1 == "":
16         t="Please Enter the your Enrollment Number."
17         text_to_speech(t)
18     elif l2 == "":
19         t="Please Enter the your Name."
20         text_to_speech(t)
21     else:
22         try:
23             cam = cv2.VideoCapture(0)
24             detector = cv2.CascadeClassifier(haarcascade_path)
25             Enrollment = l1
26             Name = l2
27             sampleNum = 0
28             directory = Enrollment + "-" + Name
29             path = os.path.join(trainimage_path, directory)
30             os.mkdir(path)
31             while True:
32                 ret, img = cam.read()
33                 gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
34                 faces = detector.detectMultiScale(gray, 1.3, 5)
35                 for (x, y, w, h) in faces:
36                     cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 2)
37                     sampleNum = sampleNum + 1
38                     cv2.imwrite(
39                         f'{path}\{
40                             + Name
41                             + "-"
42                             + Enrollment
43                             + "-"
44                             + str(sampleNum)
45                             + ".jpg",
46                         gray[y : y + h, x : x + w],
47                     )
48                     cv2.imshow("Frame", img)
49                     if cv2.waitKey(1) & 0xFF == ord('q'):
50                         break
51                     elif sampleNum > 50:
52                         break
53             cam.release()
54             cv2.destroyAllWindows()
55             row = [Enrollment, Name]
56             with open(
57                 "C:\\Users\\nxbt1\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\StudentDetails\\studentdetails.csv",
58                 "a",
59             ) as csvfile:
60                 writer = csv.writer(csvfile, delimiter=",")
61                 writer.writerow(row)
62                 csvfile.close()
63             res = "Images Saved for ID No:" + Enrollment + " Name:" + Name
64             message.configures(text=res)
65             text_to_speech(res)
66         except FileNotFoundError as f:
67             f = "Student Data already exists"
68             text_to_speech(f)
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

C:\Users\rishi\OneDrive\Desktop\Attendance-Management-system-using-face-recognition\takenmanually.py - Sublime Text (UNREGISTERED)

C:\Users\rishi\OneDrive\Desktop\Attendance-Management-system-using-face-recognition\takemanually.py - Sublime Text (UNREGISTERED)

Downloaded from <http://ajph.org/> on November 10, 2015

```
110 ENR = tk.Label(
111     MW,
112     text="Enter Enrollment",
113     width=15,
114     height=2,
115     fg="white",
116     bg="blue2",
117     font=("times", 15, "bold"),
118 )
119 ENR.place(x=30, y=100)
120
121 STU_NAME = tk.Label(
122     MW,
123     text="Enter Student name",
124     width=15,
125     height=2,
126     fg="white",
127     bg="blue2",
128     font=("times", 15, "bold"),
129 )
130 STU_NAME.place(x=30, y=200)
131
132 global ENR_ENTRY
133 ENR_ENTRY = tk.Entry(
134     MW,
135     width=20,
136     validate="key",
137     bg="yellow",
138     fg="red",
139     font=("times", 23, "bold"),
140 )
141 ENR_ENTRY["validatecommand"] = (ENR_ENTRY.register(testVal), "%P", "%d")
142 ENR_ENTRY.place(x=290, y=105)
143
144 def remove_enr():
145     ENR_ENTRY.delete(first=0, last=22)
146
147 STUDENT_ENTRY = tk.Entry(
148     MW, width=20, bg="yellow", fg="red", font=("times", 23, "bold")
149 )
150 STUDENT_ENTRY.place(x=290, y=205)
151
152 def remove_student():
153     STUDENT_ENTRY.delete(first=0, last=22)
154
155 #####get important variable
156
157 def enter_data_db():
158     global index
159     global d
160     ENROLLMENT = ENR_ENTRY.get()
161     STUDENT = STUDENT_ENTRY.get()
162     if ENROLLMENT == "":
163         enr_screen()
164     elif STUDENT == "":
165         enr_screen()
166     else:
167         if index == 0:
168             d = {
169                 "Enrollment": ENROLLMENT, "Name": STUDENT, "Date": 1
170             }
171             index += 1
172             ENR_ENTRY.delete(0, "end")
173             STUDENT_ENTRY.delete(0, "end")
174         else:
175             d[index] = {"Enrollment": ENROLLMENT, "Name": STUDENT, "Date": 1}
176             index += 1
177             ENR_ENTRY.delete(0, "end")
178             STUDENT_ENTRY.delete(0, "end")
179             # TODO: Implement CSV code
180             print(d)
181
182 def create_csv():
183     df = pd.DataFrame(d)
184     csv_name = (
185         "C:\\Users\\nrbt1\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\Attendance(Manually)\\\"
186         + subb
187         + ".-."
188         + Date
189         + ".-."
190         + Hour
191         + ".-."
192         + Minute
193         + ".-."
194         + Second
195         + ".csv"
196     )
197     df.to_csv(csv_name)
198     0 = "CSV created Successfully"
199     Notifi.configure(
200         text=0,
201         bg="Green",
202         fg="white",
203         width=31,
204         font=("times", 19, "bold"),
205     )
206     Notifi.place(x=180, y=380)
207     import csv
208     import tkinter
209
210     root = tkinter.Tk()
211     root.title("Attendance of " + subb)
212     root.configure(background="snow")
213     with open(csv_name, newline="") as file:
214         reader = csv.reader(file)
215         r = 0
216
217         for col in reader:
```

```
217         for col in reader:
218             c = 0
219             for row in col:
220                 # I've added some styling
221                 label = tkinter.Label(
222                     root,
223                     width=15,
224                     height=1,
225                     fg="black",
226                     font=("times", 13, "bold "),
227                     bg="light green",
228                     text=row,
229                     relief=tkinter.RIDGE,
230                 )
231                 label.grid(row=r, column=c)
232                 c += 1
233             r += 1
234         root.mainloop()
235
236         Notifi = tk.Label(
237             MW,
238             text="CSV created Successfully",
239             bg="green",
240             fg="white",
241             width=33,
242             height=2,
243             font=("times", 19, "bold"),
244         )
245
246         clear_enroll = tk.Button(
247             MW,
248             text="Clear",
249             command=remove_enr,
250             fg="black",
251             bg="deep pink",
252             width=10,
253             height=1,
254             activebackground="Red",
255             font=("times", 15, "bold "),
256         )
257         clear_enroll.place(x=690, y=100)
258
259         clear_student = tk.Button(
260             MW,
261             text="Clear",
262             command=remove_student,
263             fg="black",
264             bg="deep pink",
265             width=10,
266             height=1,
267             activebackground="Red",
268             font=("times", 15, "bold "),
269         )
270         clear_student.place(x=690, y=200)
271
272         # I've added some styling
```

```
272         DATA_SUB = tk.Button(
273             MW,
274             text="Enter Data",
275             command=enter_data_DB,
276             fg="black",
277             bg="light green",
278             width=20,
279             height=2,
280             activebackground="Red",
281             font=("times", 15, "bold "),
282         )
283         DATA_SUB.place(x=170, y=300)
284
285         MAKE_CSV = tk.Button(
286             MW,
287             text="Convert to CSV",
288             command=create_csv,
289             fg="black",
290             bg="red",
291             width=20,
292             height=2,
293             activebackground="Red",
294             font=("times", 15, "bold "),
295         )
296         MAKE_CSV.place(x=570, y=300)
297         # TODO remove check sheet
298         def attr():
299             import subprocess
300
301             subprocess.Popen(
302                 r"explorer /select,\"C:/Users/natbl/OneDrive/Desktop/Attendance-Management-system-using-face-recognition/Attendance(Manually)\"
303             )
304
305         attr = tk.Button(
306             MW,
307             text="Check Sheets",
308             command=attr,
309             fg="black",
310             bg="light green",
311             width=12,
312             height=1,
313             activebackground="Red",
314             font=("times", 14, "bold "),
315         )
316         attr.place(x=730, y=410)
317
318         MW.mainloop()
319
320         SUB = tk.Label(
321             MW,
322             text="Enter Subject",
323             width=15,
324             height=2,
325             fg="white",
326             bg="blue2",
```

```
298 def attf():
299     import subprocess
300
301     subprocess.Popen(
302         r"explorer /select,\"C:/Users/nathl/Desktop/Attendance-Management-system-using-face-recognition/Attendance(Manually)\"
303     )
304
305     attf = tk.Button(
306         MW,
307         text="Check Sheets",
308         command=attf,
309         fg="black",
310         bg="light green",
311         width=22,
312         height=1,
313         activebackground="Red",
314         font=("times", 14, "bold"),
315     )
316     attf.place(x=730, y=410)
317
318     MW.mainloop()
319
320 SUB = tk.Label(
321     sb,
322     text="Enter Subject",
323     width=15,
324     height=2,
325     fg="white",
326     bg="black",
327     font=("times", 15, "bold"),
328 )
329 SUB.place(x=30, y=100)
330
331 global SUB_ENTRY
332
333 SUB_ENTRY = tk.Entry(
334     sb, width=20, bg="yellow", fg="red", font=("times", 23, "bold")
335 )
336 SUB_ENTRY.place(x=250, y=105)
337
338 fill_manual_attendance = tk.Button(
339     sb,
340     text="Fill Attendance",
341     command=fill_attendance,
342     fg="white",
343     bg="deep pink",
344     width=20,
345     height=2,
346     activebackground="Red",
347     font=("times", 15, "bold"),
348 )
349 fill_manual_attendance.place(x=250, y=160)
350 sb.mainloop()
351
```


trainImage.py

```
1 import cv2
2 import os, cv2
3 import numpy as np
4 import pandas as pd
5 import datetime
6 import time
7 from PIL import ImageTk, Image
8
9
10 # Train Image
11 def trainImage(harcascade_path, trainimage_path, trainimagelabel_path, message, text_to_speech):
12     recognizer = cv2.face.LBPHFaceRecognizer_create()
13     detector = cv2.CascadeClassifier(harcascade_path)
14     faces, Id = getImagesAndLabels(trainimage_path)
15     recognizer.train(faces, np.array(Id))
16     recognizer.save(trainimagelabel_path)
17     res = "Image Trained successfully" # + ",".join(str(f) for f in Id)
18     message.config(text=res)
19     text_to_speech(res)
20
21
22 def getImagesAndLabels(path):
23     # imagePath = [os.path.join(path, f) for d in os.listdir(path) for f in d]
24     newdir = [os.path.join(path, d) for d in os.listdir(path)]
25     imagePath = []
26     for i in range(len(newdir)):
27         os.path.join(newdir[i], f)
28         for f in os.listdir(newdir[i]):
29             faces = []
30             Ids = []
31             for imagePath in imagePath:
32                 pilImage = Image.open(imagePath).convert("L")
33                 imageObj = np.array(pilImage, "uint8")
34                 Id = int(os.path.splitext(imagePath)[-1].split("_")[1])
35                 faces.append(imageObj)
36                 Ids.append(Id)
37     return faces, Ids
```

7. CONCLUSION

Face recognition systems are part of facial image processing applications and their significance as a research area are increasing recently. Implementations of the system are crime prevention, video surveillance, person verification, and similar security activities. The goal is reached by face detection and recognition methods. Knowledge-Based face detection methods are used to find, locate and extract faces in acquired images. Implemented methods are skin color and facial features. Neural network is used for face recognition. RGB color space is used to specify skin color values, and segmentation decreases searching time of face images. Facial components on face candidates are appeared with implementation of LoG filter. LoG filter shows good performance on extracting facial components under different illumination conditions. FFNN is performed to solve pattern recognition problems since face recognition is a kind of pattern recognition. Classification result is accurate. Classification is also flexible and correct when extracted face image is small oriented, closed eye, and small smiled.

Proposed algorithm is capable of detecting multiple faces, and performance of the system has acceptable good results.