# 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 independentcomponent 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

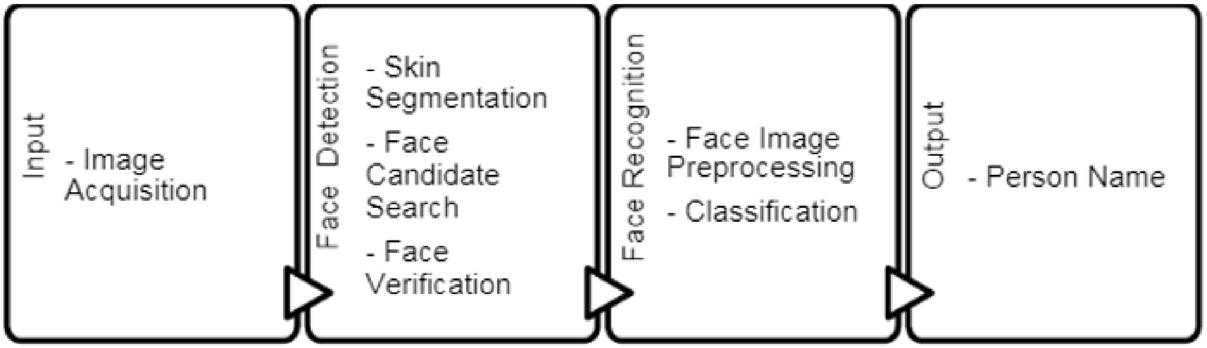


*Figure 2 Face Recognition Approach*

### 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.

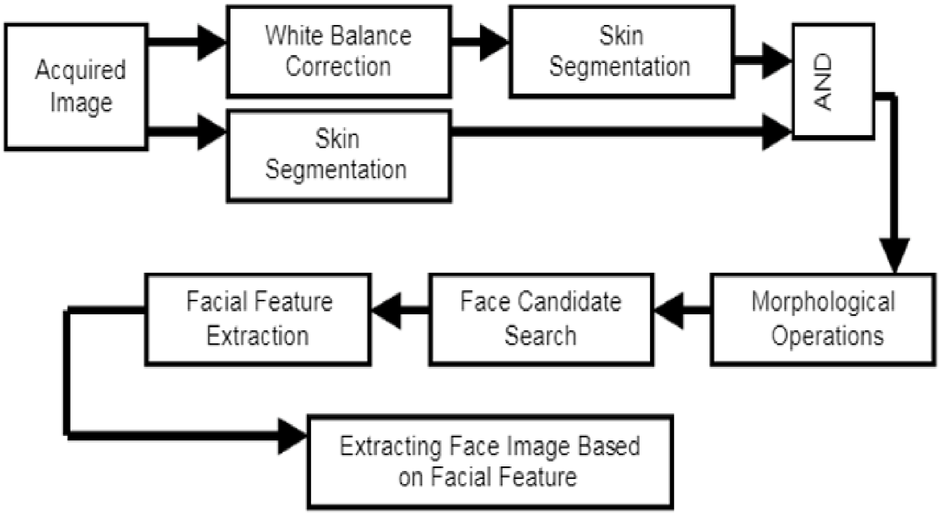


*Figure 3 Algorithm of Face Detection Part*

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.

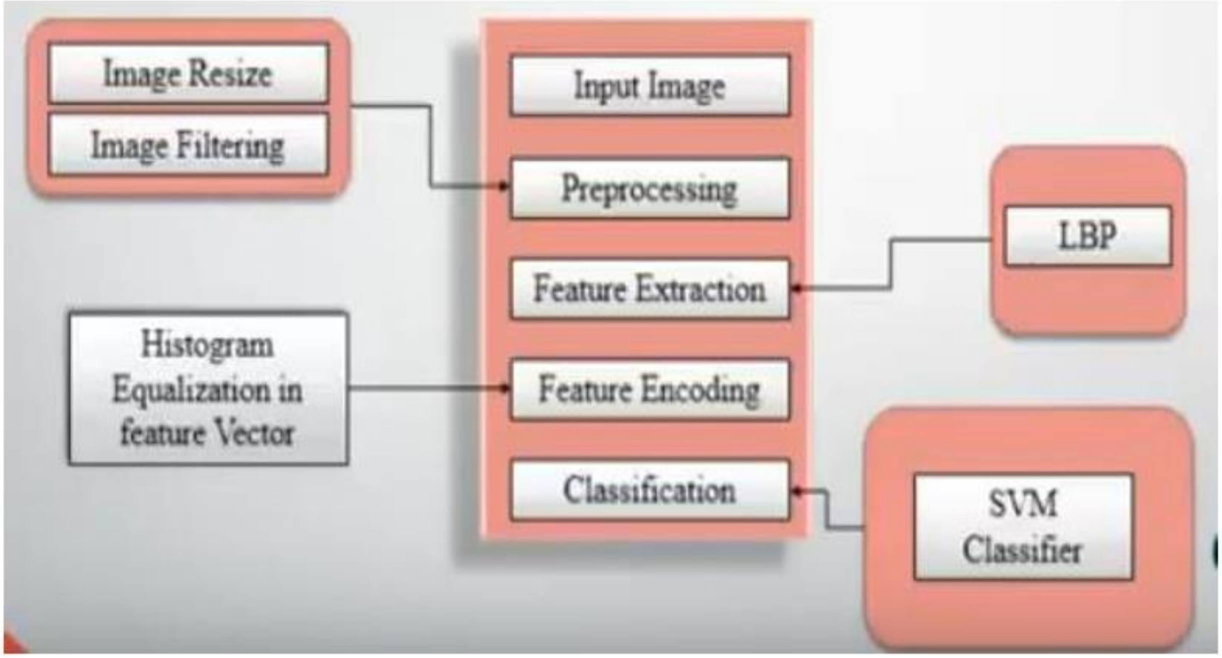
Therefore, the white balance of the acquired image should be corrected before segmenting it [18]. Results of segmentation on the original image and white balance corrected image is given in Figure 4 and 5.

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.

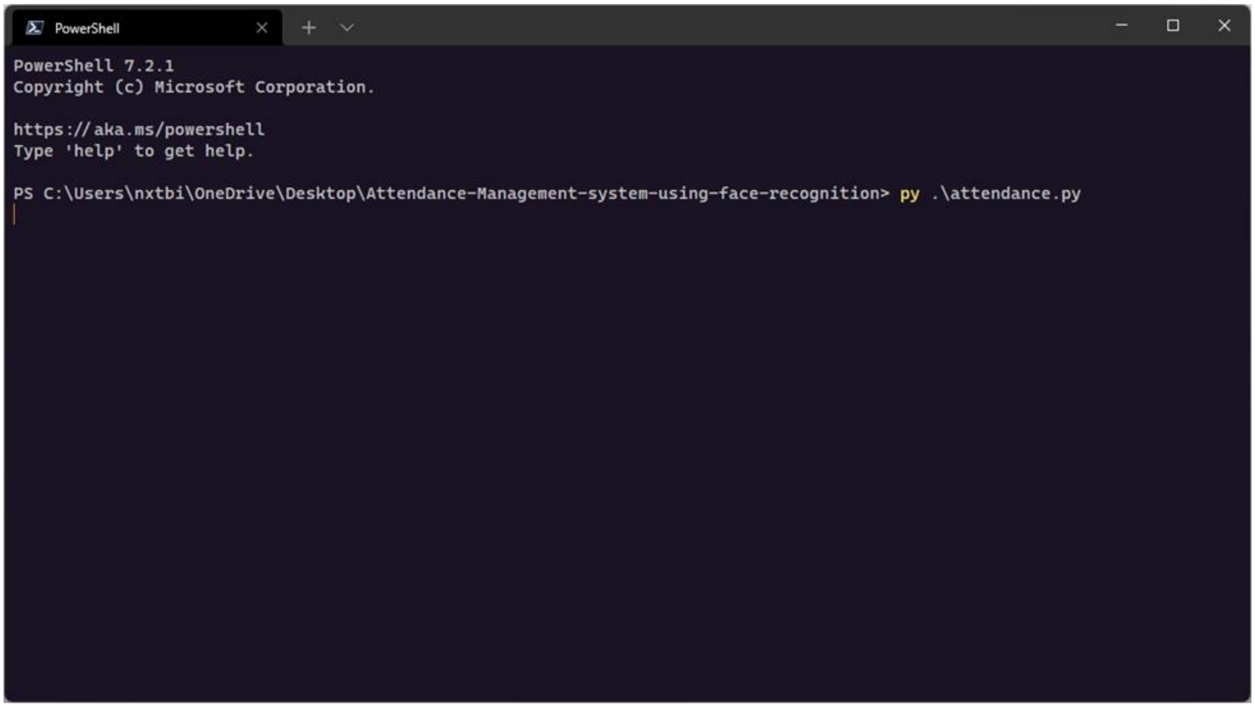
These face candidates will be sent to the facial feature extraction part to validate the candidates.  Final verification of candidate and face image extraction, the facial feature extraction process is applied. The facial feature is one of the most significant features of the face. Facial features are 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)

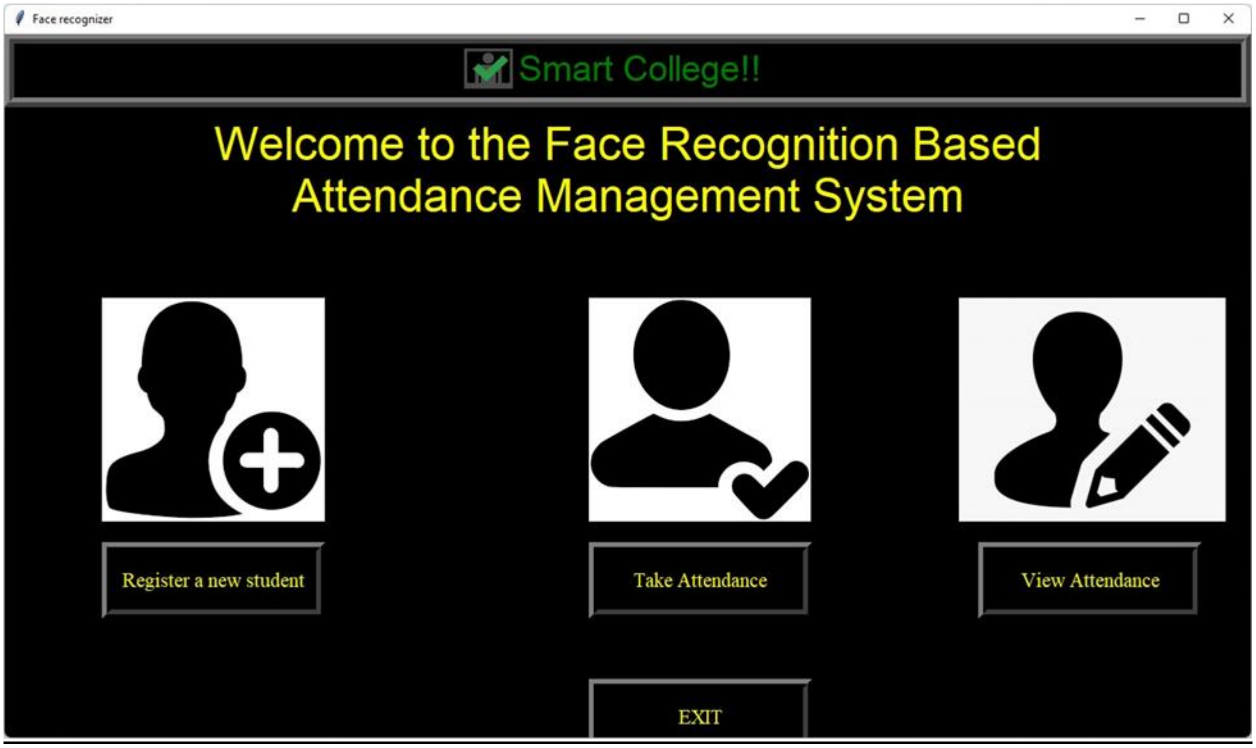


A similar separation of pattern recognition algorithms into four groups is proposed by Jain and colleges. We can group face recognition methods into three main groups. The following approaches are proposed: ˆ
Template matching:- Patterns are represented by samples, models, pixels, curves, textures. The recognition function is usually a correlation or distance measure. ˆ
Statistical approach:- Patterns are represented as features. The recognition function is a discriminant function. ˆ
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.

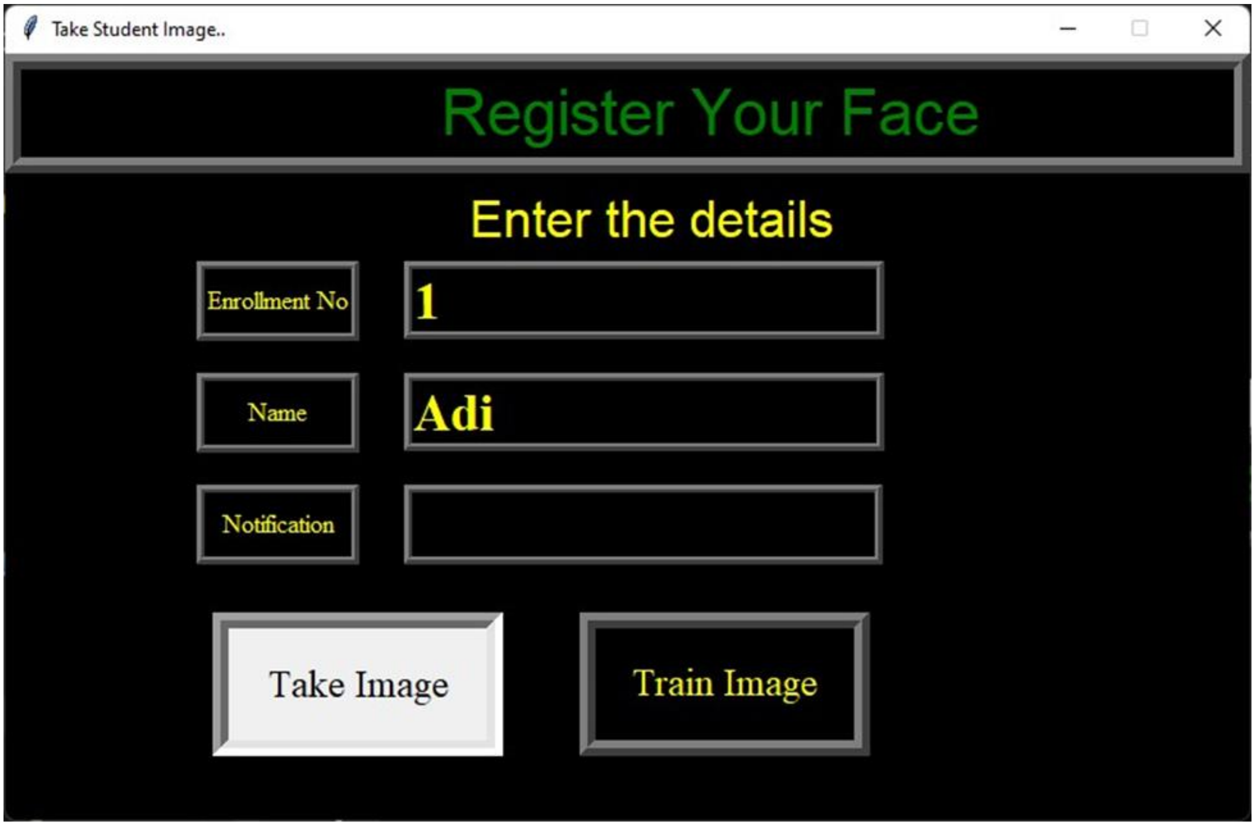
# 6. SAMPLE INPUTS AND OUTPUTS



```
PowerShell 7.2.1
Copyright (c) Microsoft Corporation.

https://aka.ms/powershell
Type 'help' to get help.

PS C:\Users\nxtbi\OneDrive\Desktop\Attendance-Management-system-using-face-recognition> py .\attendance.py
```



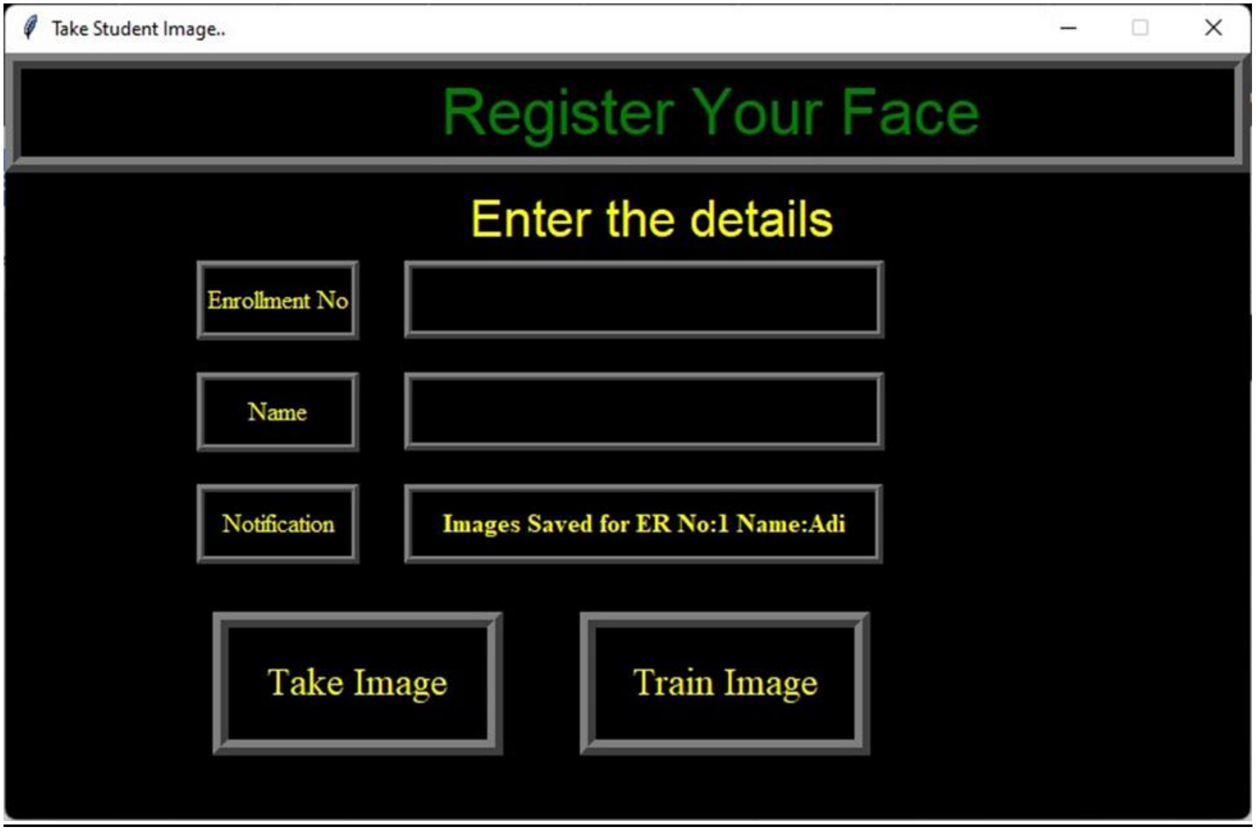
Welcome to the Face Recognition Based Attendance Management System



Enrollment No: 1

Name: Adi

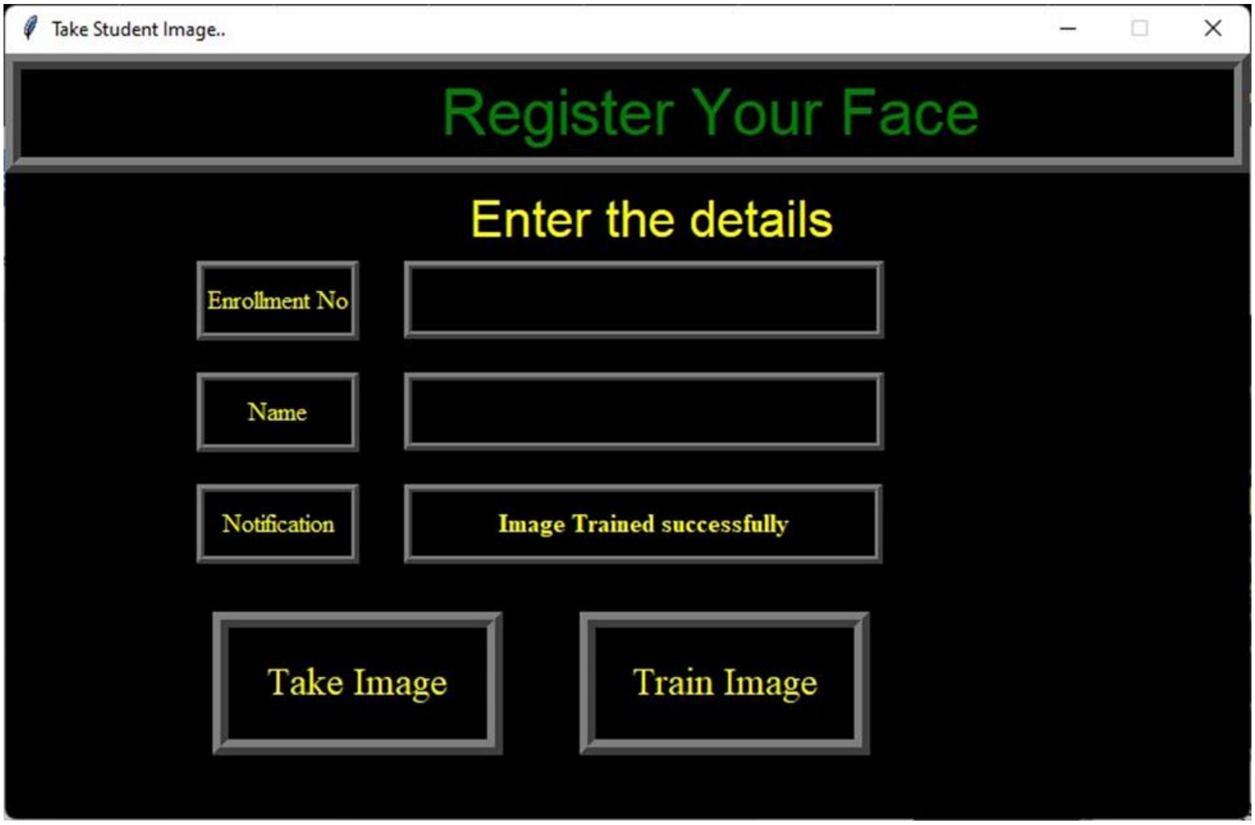
Notification:



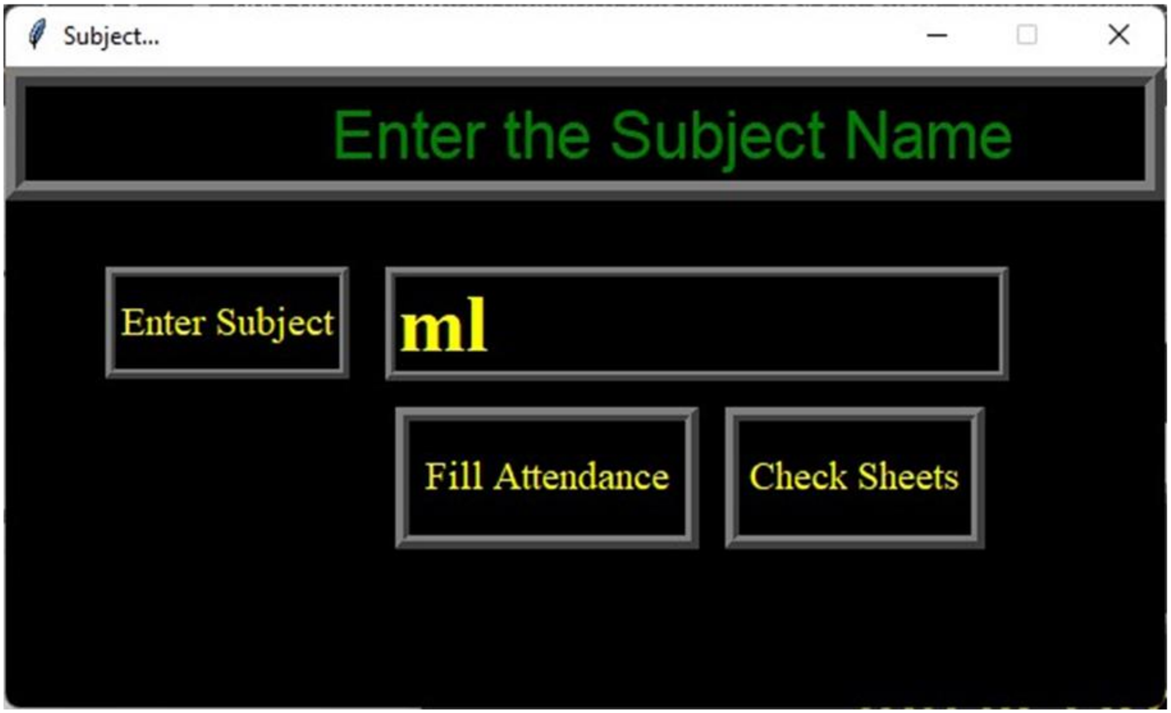
Enrollment No:

Name:

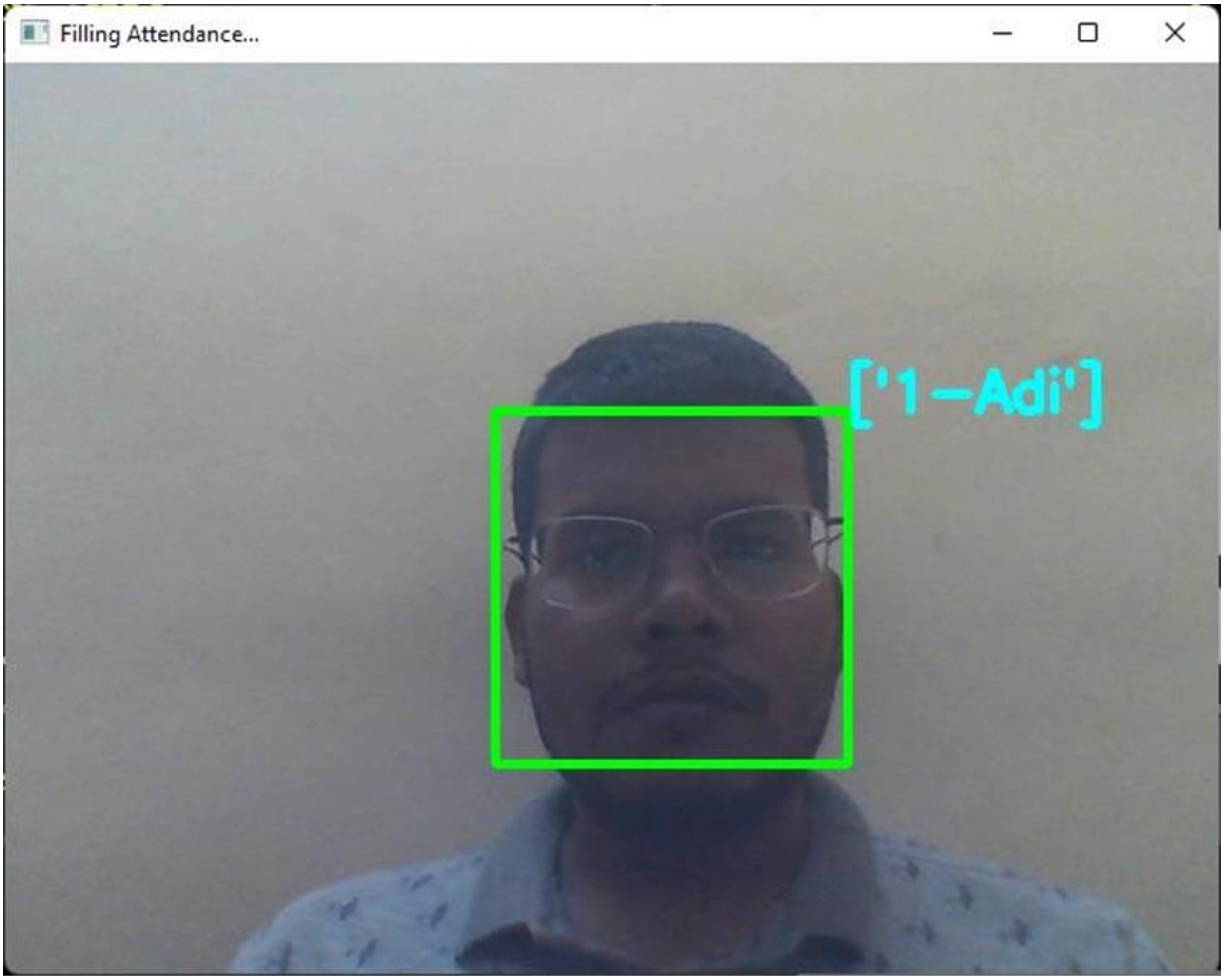
Notification: Images Saved for ER No:1 Name:Adi

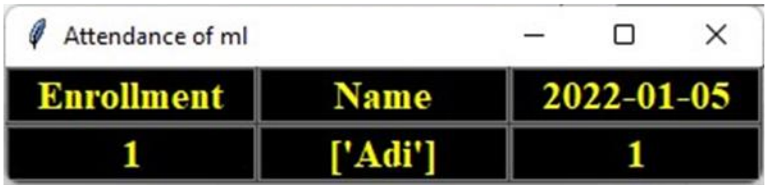


Notification: Image Trained successfully

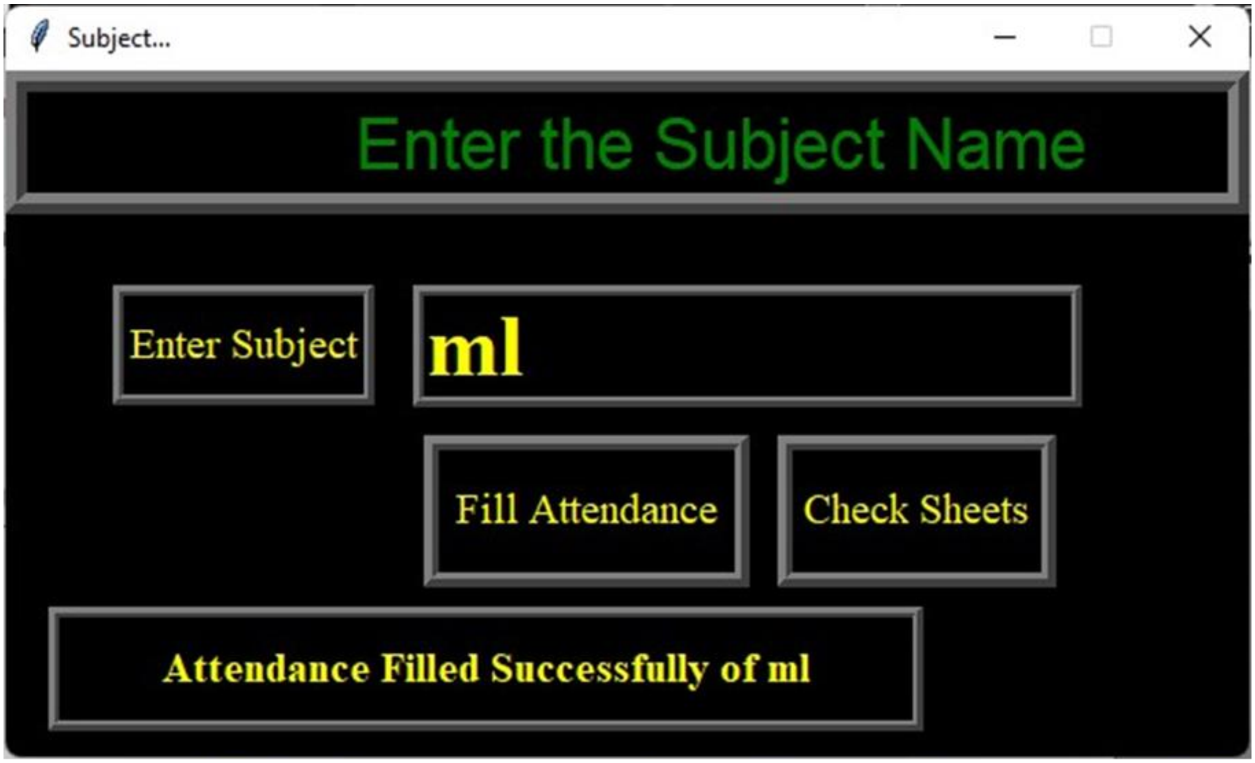


Enter Subject: ml





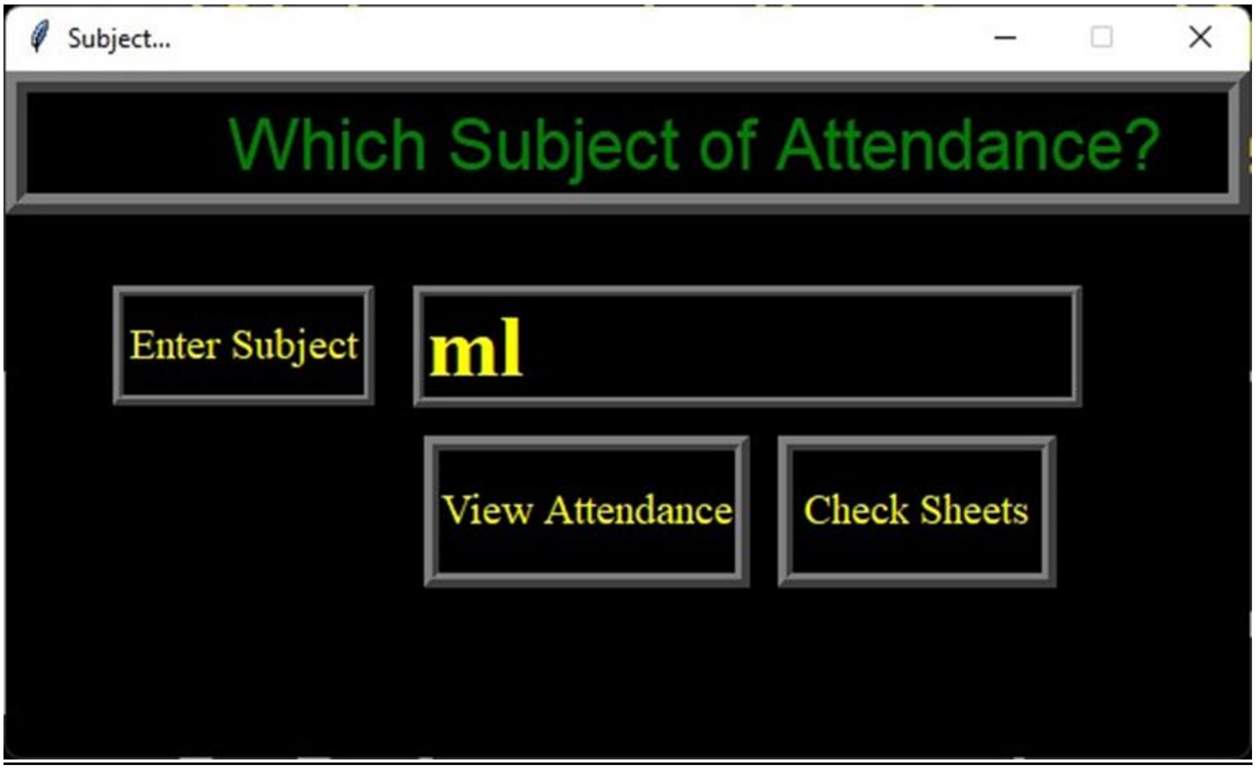
| Enrollment | Name | 2022-01-05 |
|---|---|---|
| 1 | ['Adi'] | 1 |



Enter the Subject Name

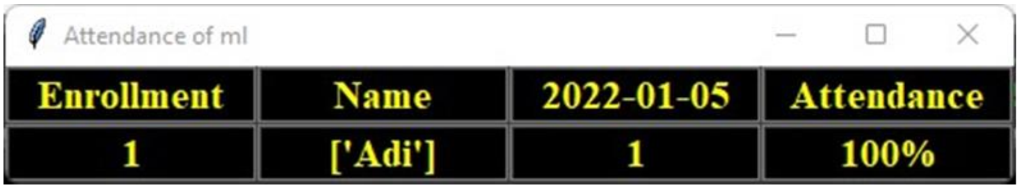
Enter Subject: ml

Attendance Filled Successfully of ml



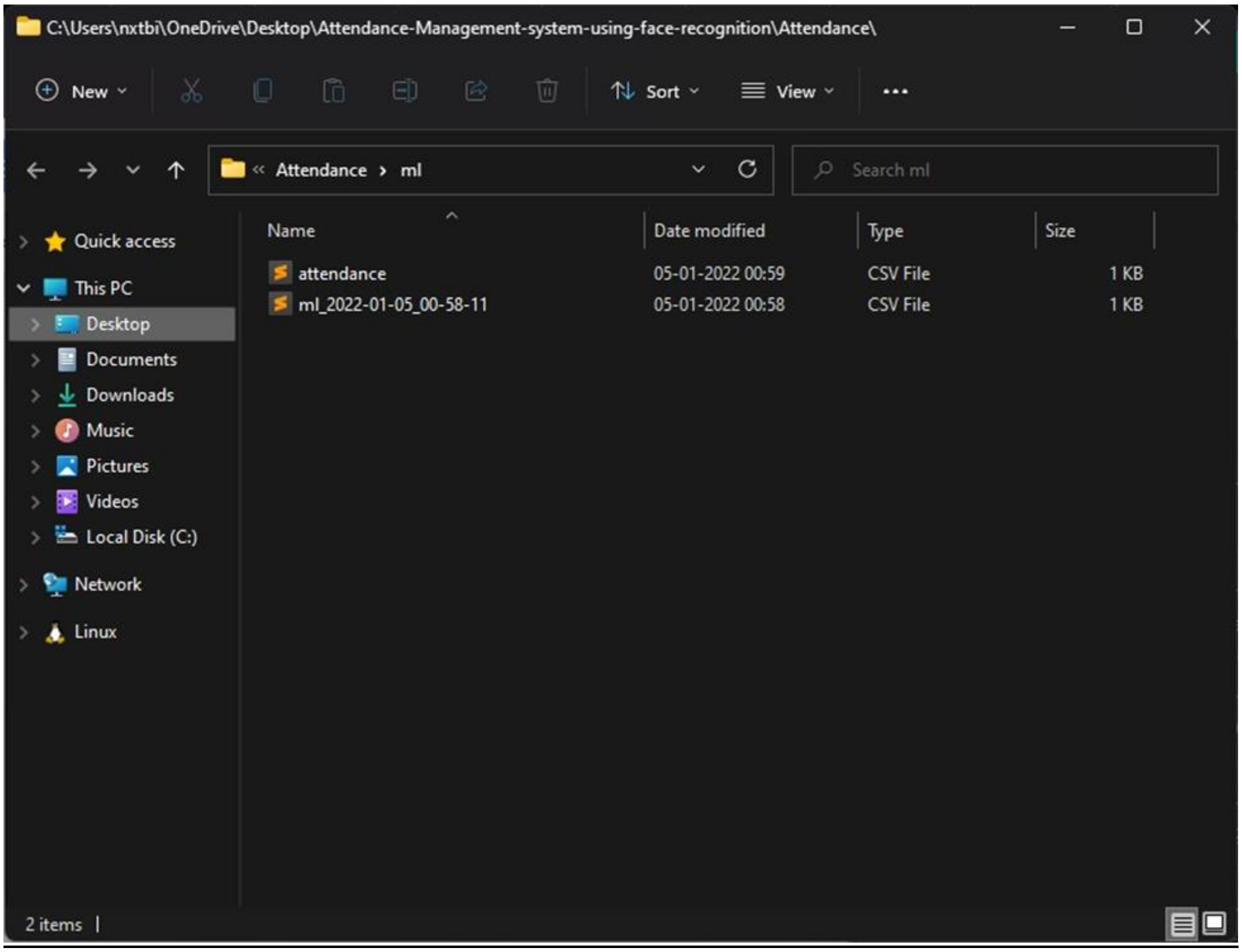
Which Subject of Attendance?

Enter Subject: ml



Attendance of ml

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

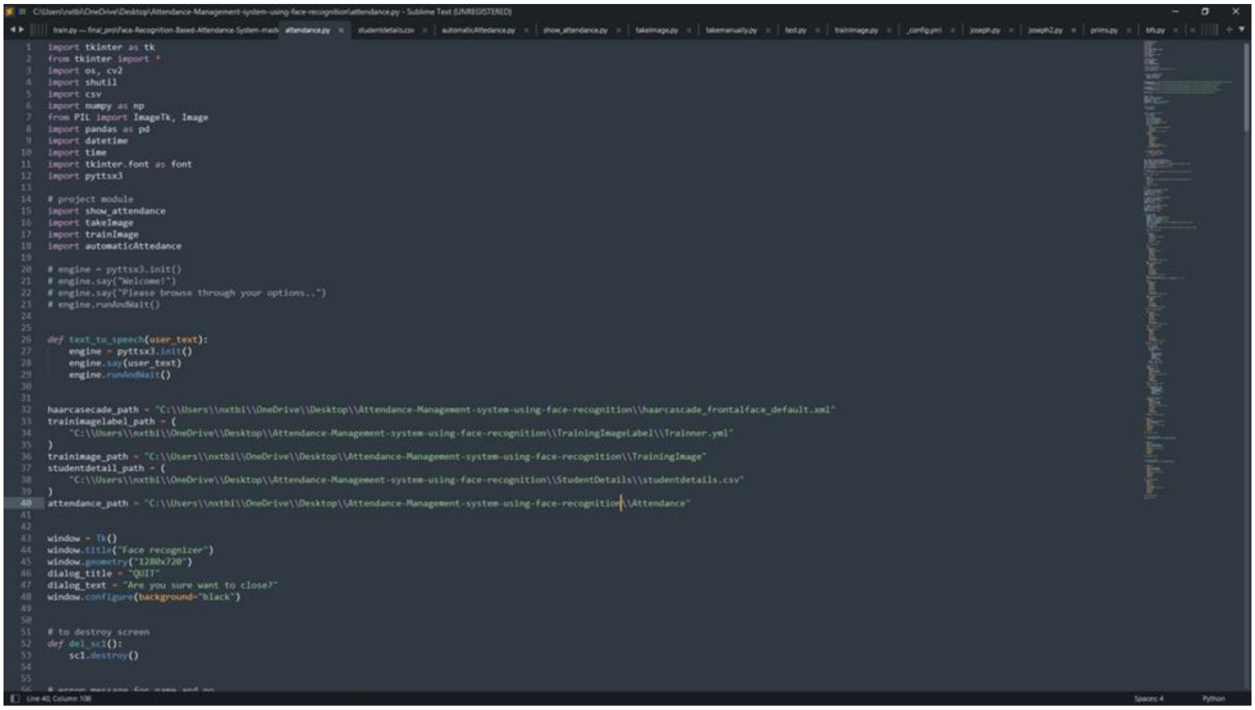


C:\Users\nxtbi\OneDrive\Desktop\Attendance-Management-system-using-face-recognition\Attendance\

| Name | Date modified | Type | Size |
|---|---|---|---|
| attendance | 05-01-2022 00:59 | CSV File | 1 KB |
| ml_2022-01-05_00-58-11 | 05-01-2022 00:58 | CSV File | 1 KB |

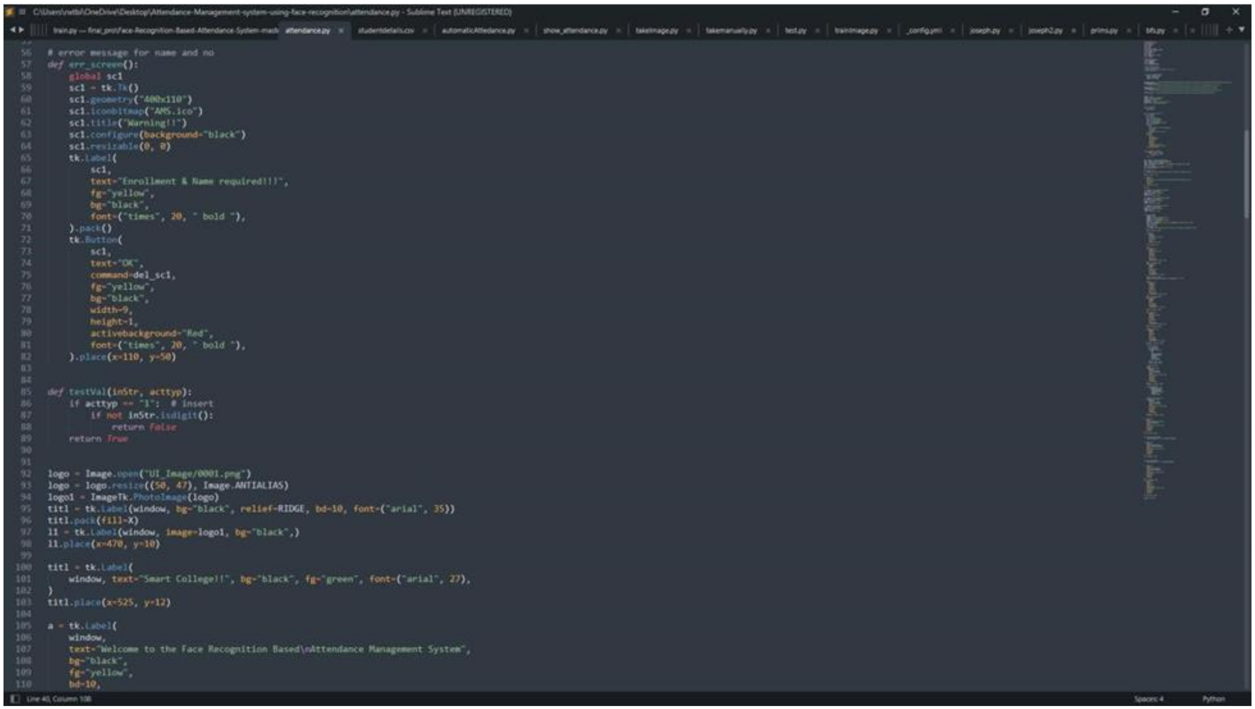
2 items

# Attendance.py



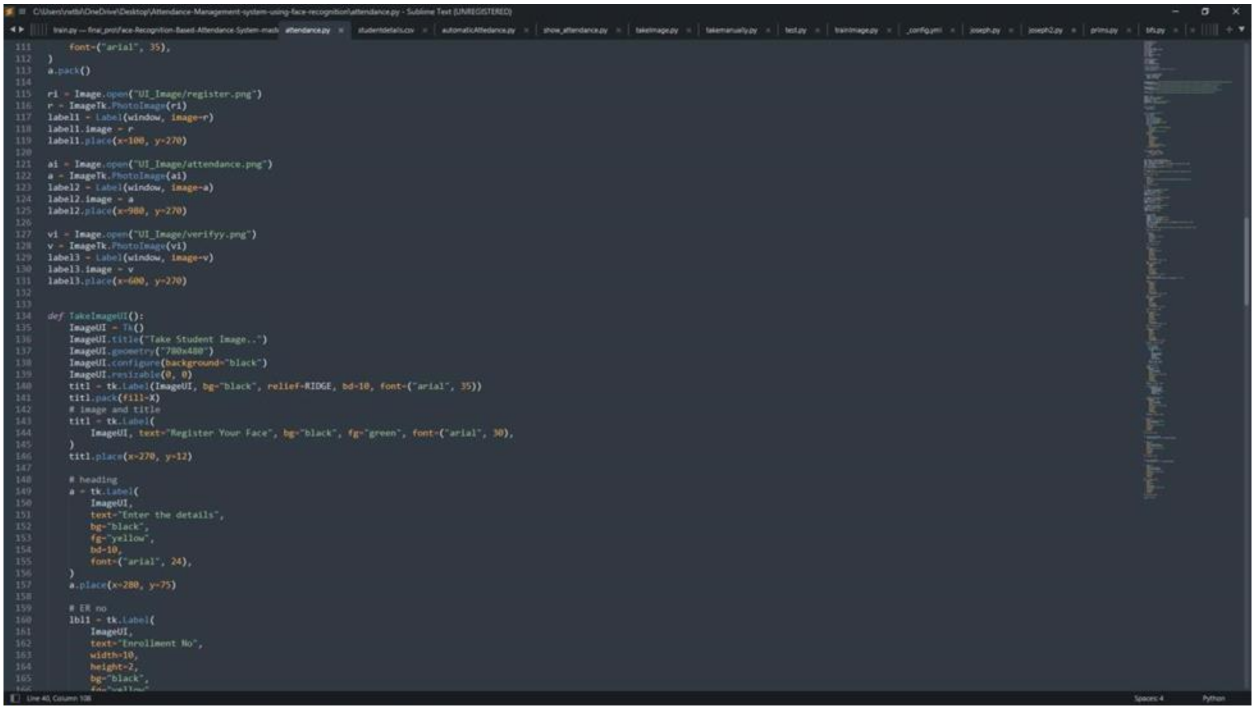
C:\Users\nxtbi\OneDrive\Desktop\Attendance-Management-system-using-face-recognition\attendance.py - Sublime Text (UNREGISTERED)
Line 40, Column 108
Spaces: 4
Python

```python
import tkinter as tk
from tkinter import *
import os, cv2
import shutil
import csv
import numpy as np
from PIL import ImageTk, Image
import pandas as pd
import datetime
import time
import tkinter.font as font
import pyttsx3

# project module
import show_attendance
import takeImage
import trainImage
import automaticAttedance

# engine = pyttsx3.init()
# engine.say("Welcome!")
# engine.say("Please browse through your options..")
# engine.runAndWait()


def text_to_speech(user_text):
    engine = pyttsx3.init()
    engine.say(user_text)
    engine.runAndWait()


haarcasecade_path = "C:\\Users\\nxtbi\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\haarcascade_frontalface_default.xml"
trainimagelabel_path = (
    "C:\\Users\\nxtbi\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\TrainingImageLabel\\Trainner.yml"
)
trainimage_path = "C:\\Users\\nxtbi\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\TrainingImage"
studentdetail_path = (
    "C:\\Users\\nxtbi\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\StudentDetails\\studentdetails.csv"
)
attendance_path = "C:\\Users\\nxtbi\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\Attendance"


window = Tk()
window.title("Face recognizer")
window.geometry("1280x720")
dialog_title = "QUIT"
dialog_text = "Are you sure want to close?"
window.configure(background="black")


# to destroy screen
def del_sc1():
    sc1.destroy()
```

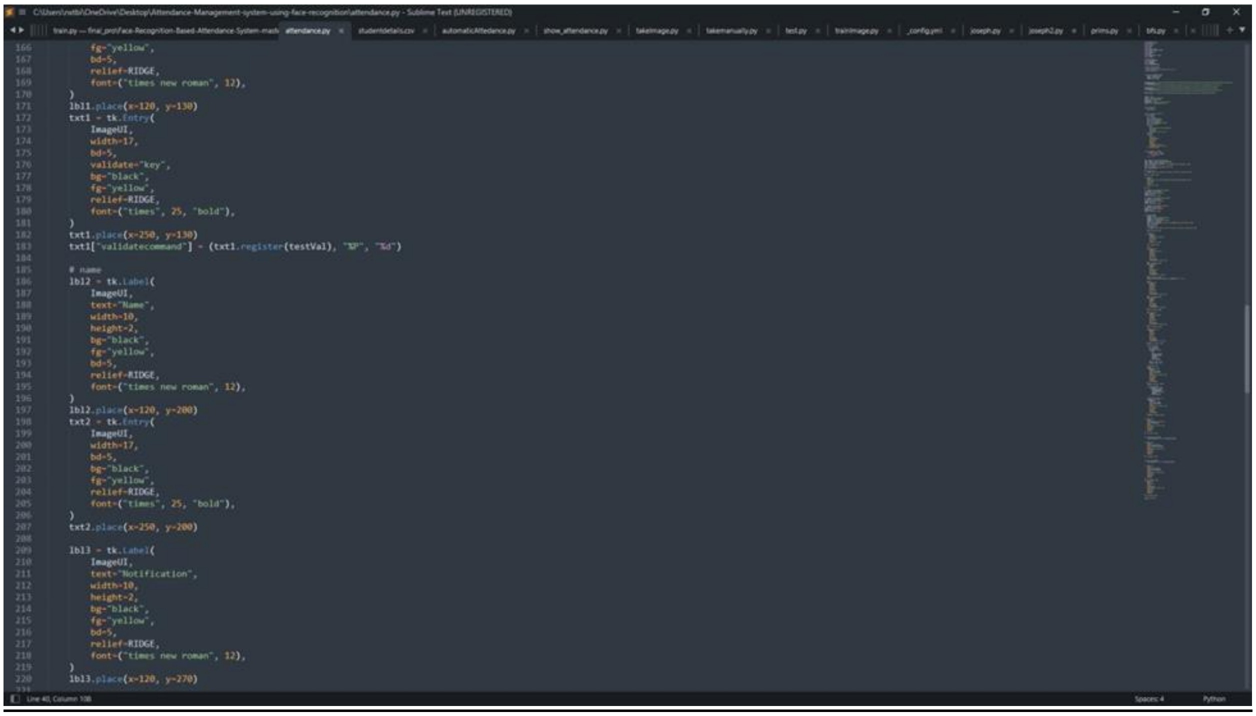


C:\Users\nxtbi\OneDrive\Desktop\Attendance-Management-system-using-face-recognition\attendance.py - Sublime Text (UNREGISTERED)
Line 40, Column 108
Spaces: 4
Python

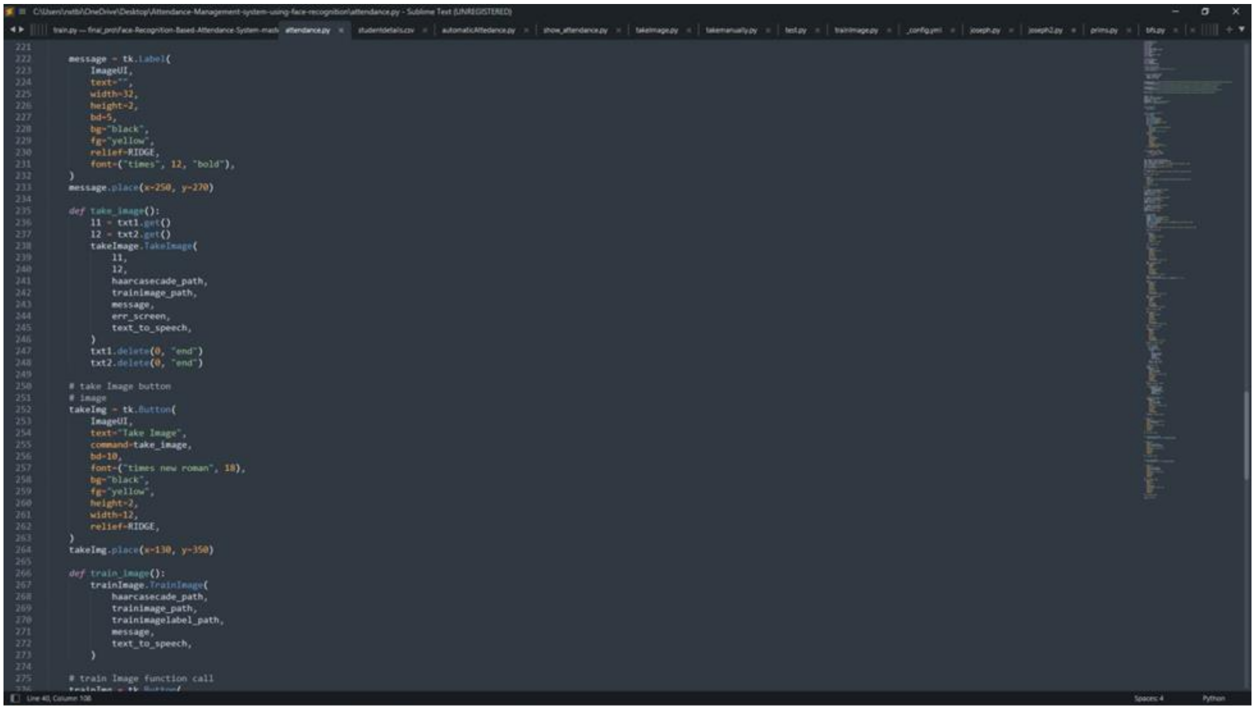
```python
# error message for name and no
def err_screen():
    global sc1
    sc1 = tk.Tk()
    sc1.geometry("400x110")
    sc1.iconbitmap("AMS.ico")
    sc1.title("Warning!!")
    sc1.configure(background="black")
    sc1.resizable(0, 0)
    tk.Label(
        sc1,
        text="Enrollment & Name required!!!",
        fg="yellow",
        bg="black",
        font=("times", 20, " bold "),
    ).pack()
    tk.Button(
        sc1,
        text="OK",
        command=del_sc1,
        fg="yellow",
        bg="black",
        width=9,
        height=1,
        activebackground="Red",
        font=("times", 20, " bold "),
    ).place(x=110, y=50)


def testVal(inStr, acttyp):
    if acttyp == "1":  # insert
        if not inStr.isdigit():
            return False
    return True


logo = Image.open("UI_Image/0001.png")
logo = logo.resize((50, 47), Image.ANTIALIAS)
logo1 = ImageTk.PhotoImage(logo)
titl = tk.Label(window, bg="black", relief=RIDGE, bd=10, font=("arial", 35))
titl.pack(fill=X)
l1 = tk.Label(window, image=logo1, bg="black",)
l1.place(x=470, y=10)

titl = tk.Label(
    window, text="Smart College!!", bg="black", fg="green", font=("arial", 27),
)
titl.place(x=525, y=12)

a = tk.Label(
    window,
    text="Welcome to the Face Recognition Based\nAttendance Management System",
    bg="black",
    fg="yellow",
    bd=10,
```


C:\Users\nettb\OneDrive\Desktop\Attendance-Management-system-using-face-recognition\attendance.py - Sublime Text (UNREGISTERED)
attendance.py
studentdetails.csv
automaticAttedance.py
show_attendance.py
Line 40, Column 108
Spaces: 4
Python

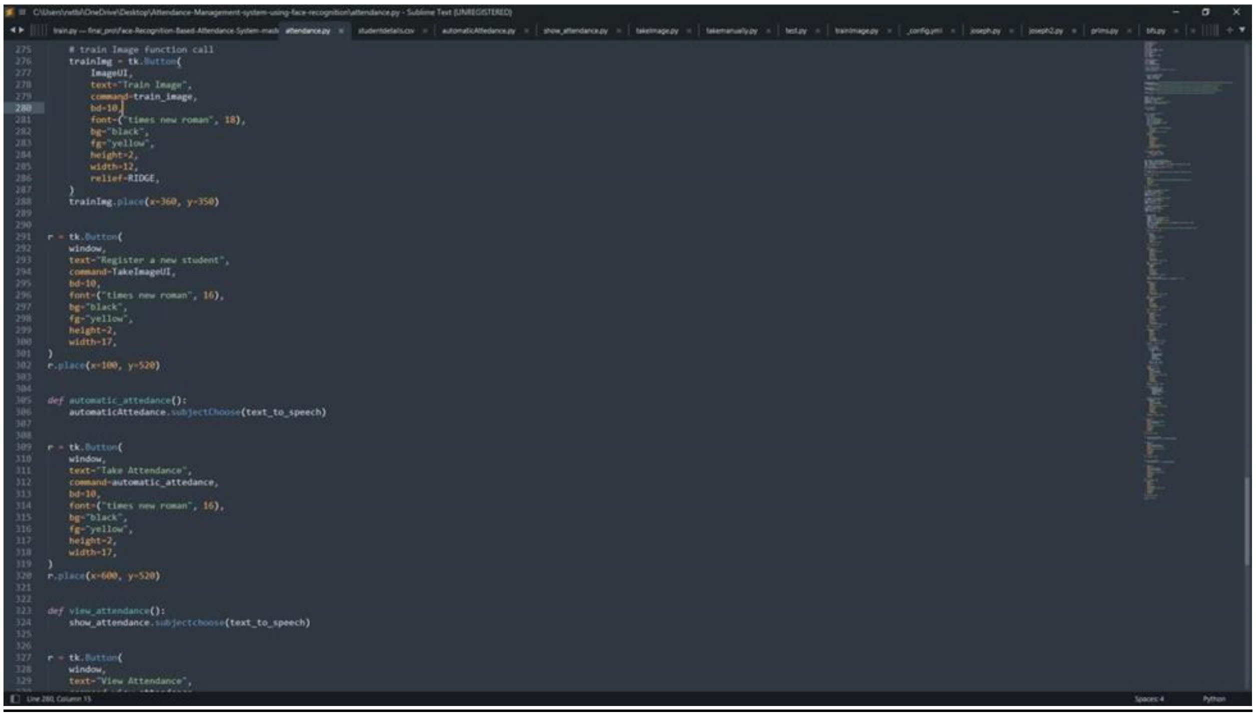
```
    font=("arial", 35),
)
a.pack()

ri = Image.open("UI_Image/register.png")
r = ImageTk.PhotoImage(ri)
label1 = Label(window, image=r)
label1.image = r
label1.place(x=100, y=270)

ai = Image.open("UI_Image/attendance.png")
a = ImageTk.PhotoImage(ai)
label2 = Label(window, image=a)
label2.image = a
label2.place(x=980, y=270)

vi = Image.open("UI_Image/verifyy.png")
v = ImageTk.PhotoImage(vi)
label3 = Label(window, image=v)
label3.image = v
label3.place(x=600, y=270)


def TakeImageUI():
    ImageUI = Tk()
    ImageUI.title("Take Student Image..")
    ImageUI.geometry("780x480")
    ImageUI.configure(background="black")
    ImageUI.resizable(0, 0)
    titl = tk.Label(ImageUI, bg="black", relief=RIDGE, bd=10, font=("arial", 35))
    titl.pack(fill=X)
    # image and title
    titl = tk.Label(
        ImageUI, text="Register Your Face", bg="black", fg="green", font=("arial", 30),
    )
    titl.place(x=270, y=12)

    # heading
    a = tk.Label(
        ImageUI,
        text="Enter the details",
        bg="black",
        fg="yellow",
        bd=10,
        font=("arial", 24),
    )
    a.place(x=280, y=75)

    # ER no
    lbl1 = tk.Label(
        ImageUI,
        text="Enrollment No",
        width=10,
        height=2,
        bg="black",
```


C:\Users\nettb\OneDrive\Desktop\Attendance-Management-system-using-face-recognition\attendance.py - Sublime Text (UNREGISTERED)
attendance.py
studentdetails.csv
automaticAttedance.py
show_attendance.py
Line 40, Column 108
Spaces: 4
Python

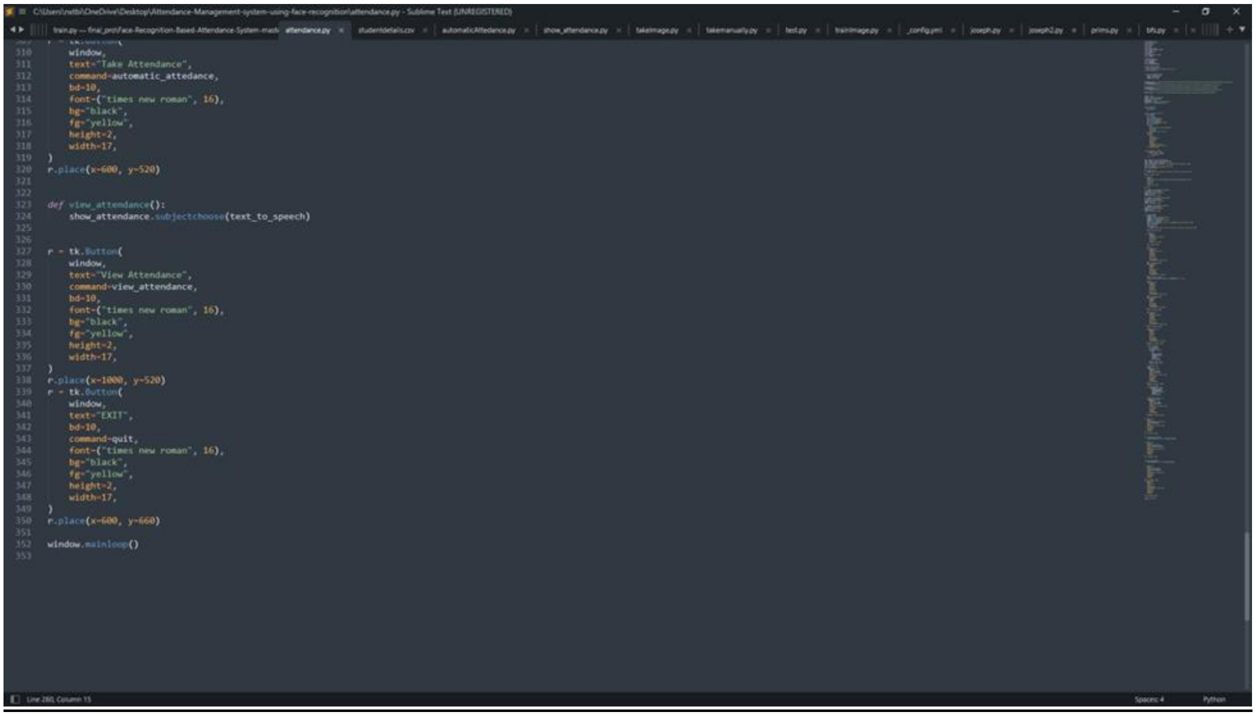
```
        fg="yellow",
        bd=5,
        relief=RIDGE,
        font=("times new roman", 12),
    )
    lbl1.place(x=120, y=130)
    txt1 = tk.Entry(
        ImageUI,
        width=17,
        bd=5,
        validate="key",
        bg="black",
        fg="yellow",
        relief=RIDGE,
        font=("times", 25, "bold"),
    )
    txt1.place(x=250, y=130)
    txt1["validatecommand"] = (txt1.register(testVal), "%P", "%d")

    # name
    lbl2 = tk.Label(
        ImageUI,
        text="Name",
        width=10,
        height=2,
        bg="black",
        fg="yellow",
        bd=5,
        relief=RIDGE,
        font=("times new roman", 12),
    )
    lbl2.place(x=120, y=200)
    txt2 = tk.Entry(
        ImageUI,
        width=17,
        bd=5,
        bg="black",
        fg="yellow",
        relief=RIDGE,
        font=("times", 25, "bold"),
    )
    txt2.place(x=250, y=200)

    lbl3 = tk.Label(
        ImageUI,
        text="Notification",
        width=10,
        height=2,
        bg="black",
        fg="yellow",
        bd=5,
        relief=RIDGE,
        font=("times new roman", 12),
    )
    lbl3.place(x=120, y=270)
```



```

    message = tk.Label(
        ImageUI,
        text="",
        width=32,
        height=2,
        bd=5,
        bg="black",
        fg="yellow",
        relief=RIDGE,
        font=("times", 12, "bold"),
    )
    message.place(x=250, y=270)

    def take_image():
        l1 = txt1.get()
        l2 = txt2.get()
        takeImage.TakeImage(
            l1,
            l2,
            haarcasecade_path,
            trainimage_path,
            message,
            err_screen,
            text_to_speech,
        )
        txt1.delete(0, "end")
        txt2.delete(0, "end")

    # take Image button
    # image
    takeImg = tk.Button(
        ImageUI,
        text="Take Image",
        command=take_image,
        bd=10,
        font=("times new roman", 18),
        bg="black",
        fg="yellow",
        height=2,
        width=12,
        relief=RIDGE,
    )
    takeImg.place(x=130, y=350)

    def train_image():
        trainImage.TrainImage(
            haarcasecade_path,
            trainimage_path,
            trainimagelabel_path,
            message,
            text_to_speech,
        )

    # train Image function call
```

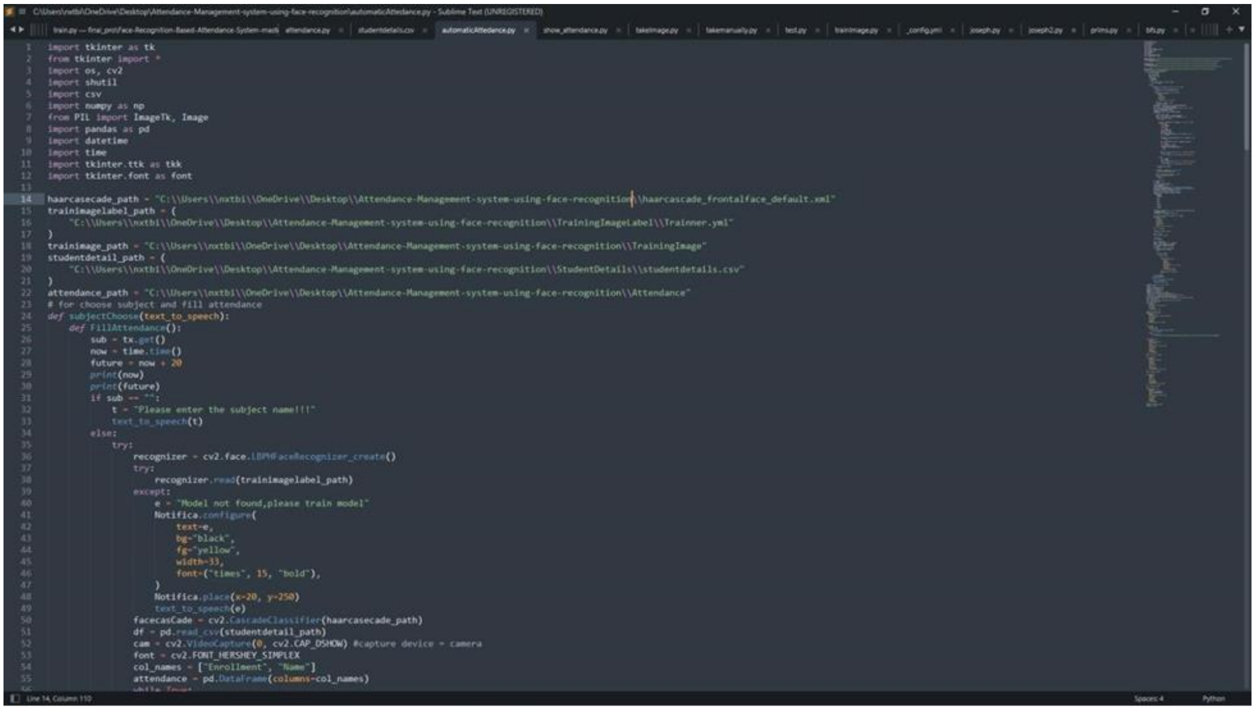


```
    # train Image function call
    trainImg = tk.Button(
        ImageUI,
        text="Train Image",
        command=train_image,
        bd=10,
        font=("times new roman", 18),
        bg="black",
        fg="yellow",
        height=2,
        width=12,
        relief=RIDGE,
    )
    trainImg.place(x=360, y=350)


r = tk.Button(
    window,
    text="Register a new student",
    command=TakeImageUI,
    bd=10,
    font=("times new roman", 16),
    bg="black",
    fg="yellow",
    height=2,
    width=17,
)
r.place(x=100, y=520)


def automatic_attedance():
    automaticAttedance.subjectChoose(text_to_speech)


r = tk.Button(
    window,
    text="Take Attendance",
    command=automatic_attedance,
    bd=10,
    font=("times new roman", 16),
    bg="black",
    fg="yellow",
    height=2,
    width=17,
)
r.place(x=600, y=520)


def view_attendance():
    show_attendance.subjectchoose(text_to_speech)


r = tk.Button(
    window,
    text="View Attendance",
```

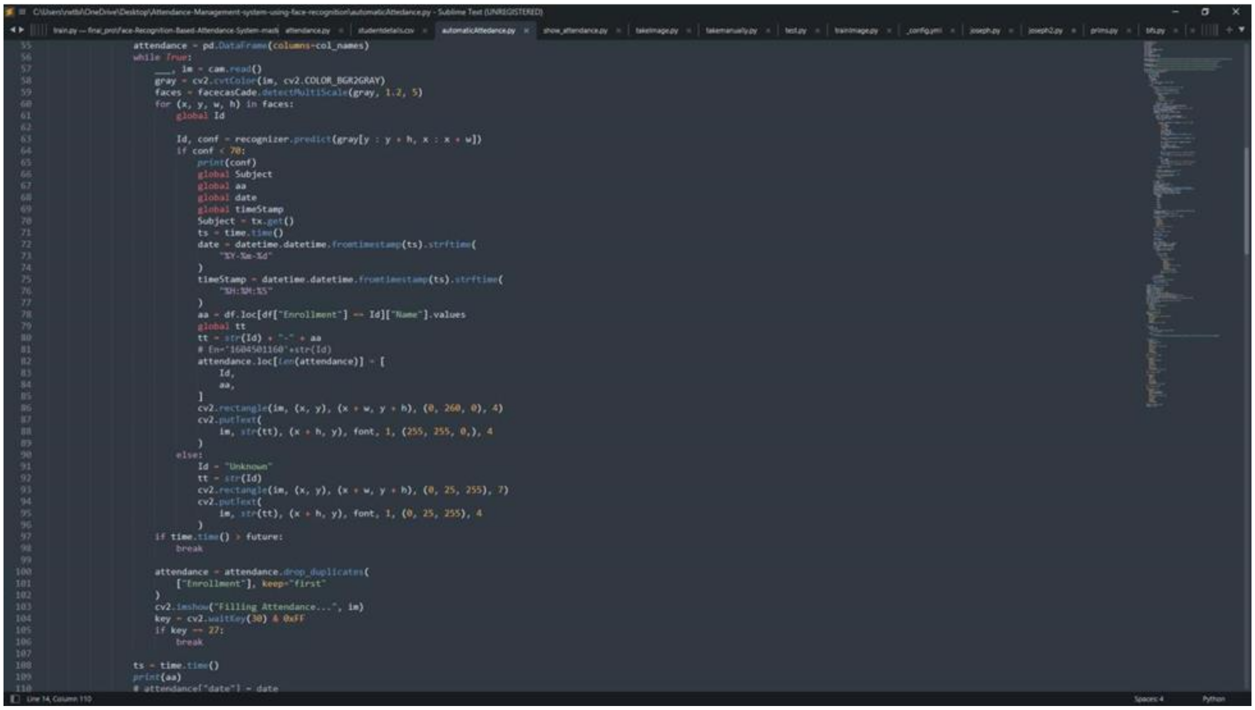


```
r = tk.Button(
    window,
    text="Take Attendance",
    command=automatic_attedance,
    bd=10,
    font=("times new roman", 16),
    bg="black",
    fg="yellow",
    height=2,
    width=17,
)
r.place(x=600, y=520)


def view_attendance():
    show_attendance.subjectchoose(text_to_speech)


r = tk.Button(
    window,
    text="View Attendance",
    command=view_attendance,
    bd=10,
    font=("times new roman", 16),
    bg="black",
    fg="yellow",
    height=2,
    width=17,
)
r.place(x=1000, y=520)
r = tk.Button(
    window,
    text="EXIT",
    bd=10,
    command=quit,
    font=("times new roman", 16),
    bg="black",
    fg="yellow",
    height=2,
    width=17,
)
r.place(x=600, y=660)

window.mainloop()
```

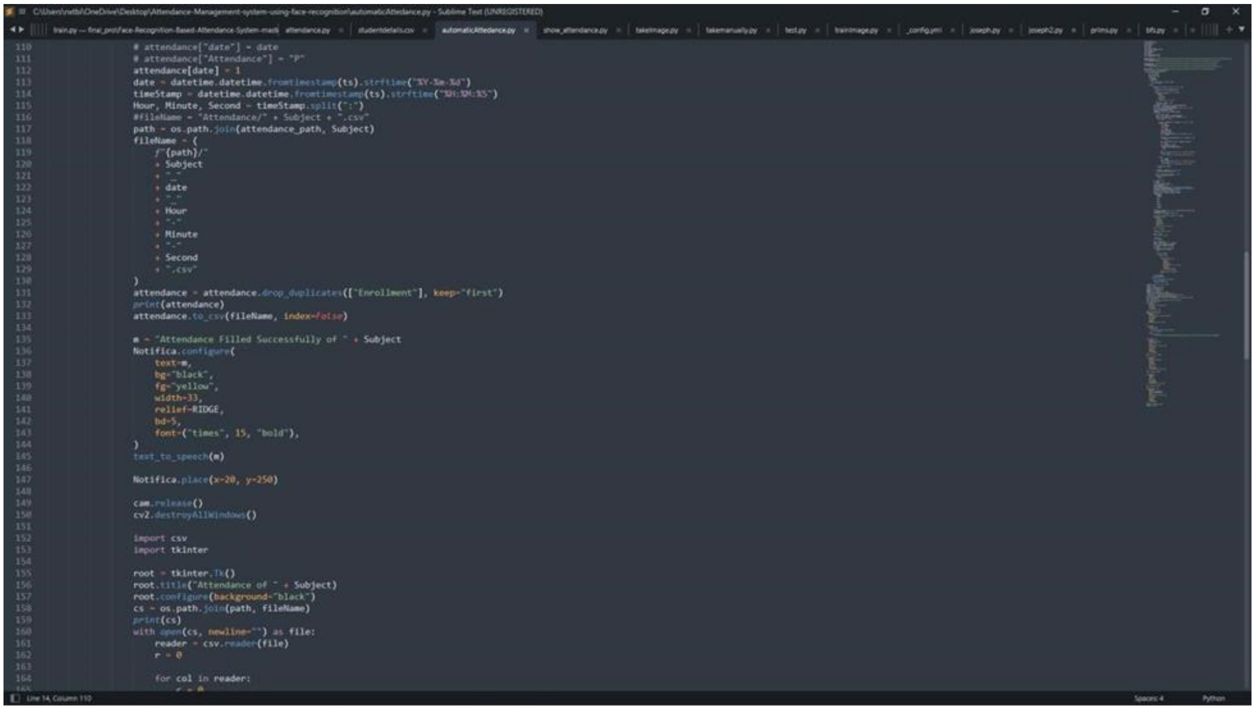
# automaticAttedance.py


C:\Users\nxtbi\OneDrive\Desktop\Attendance-Management-system-using-face-recognition\automaticAttedance.py - Sublime Text (UNREGISTERED)
train.py — final_proj\Face-Recognition-Based-Attendance-System-mas
attendance.py
studentdetails.csv
automaticAttedance.py
show_attendance.py
takeImage.py
takemanually.py
test.py
trainImage.py
_config.yml
joseph.py
joseph2.py
prims.py
bfs.py
Line 14, Column 110
Spaces: 4
Python

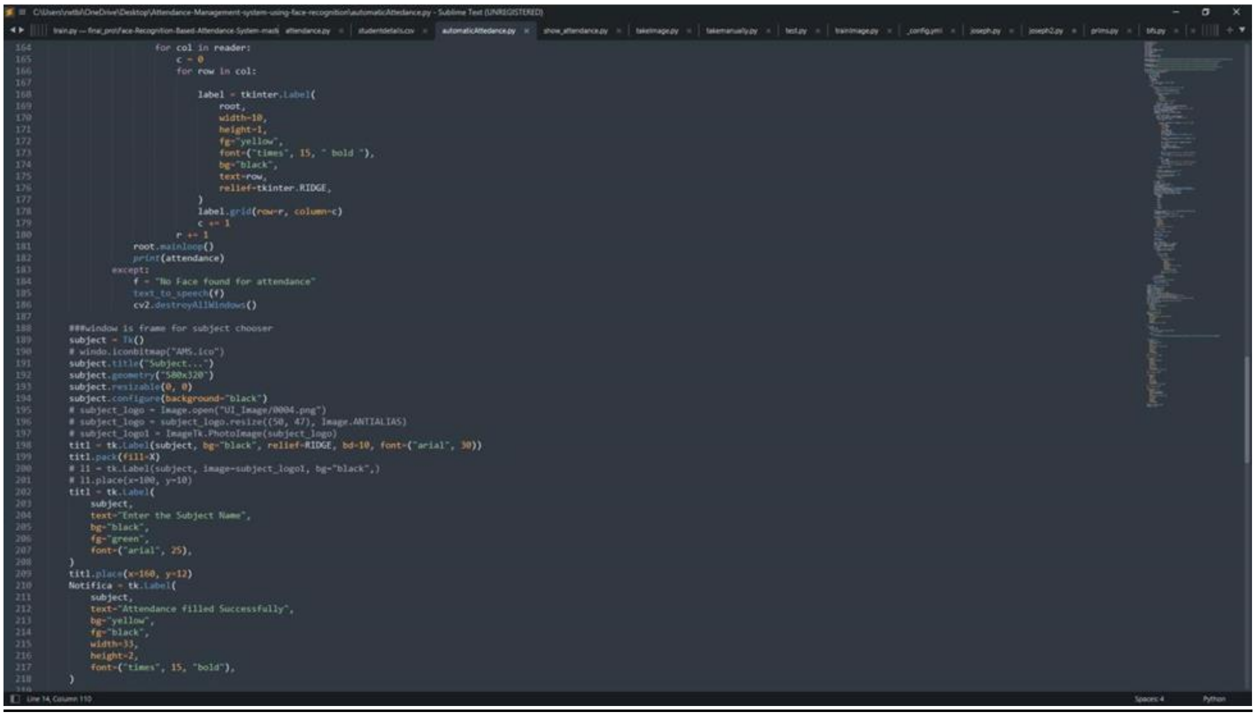
```python
import tkinter as tk
from tkinter import *
import os, cv2
import shutil
import csv
import numpy as np
from PIL import ImageTk, Image
import pandas as pd
import datetime
import time
import tkinter.ttk as tkk
import tkinter.font as font

haarcasecade_path = "C:\\Users\\nxtbi\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\haarcascade_frontalface_default.xml"
trainimagelabel_path = (
    "C:\\Users\\nxtbi\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\TrainingImageLabel\\Trainner.yml"
)
trainimage_path = "C:\\Users\\nxtbi\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\TrainingImage"
studentdetail_path = (
    "C:\\Users\\nxtbi\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\StudentDetails\\studentdetails.csv"
)
attendance_path = "C:\\Users\\nxtbi\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\Attendance"
# for choose subject and fill attendance
def subjectChoose(text_to_speech):
    def FillAttendance():
        sub = tx.get()
        now = time.time()
        future = now + 20
        print(now)
        print(future)
        if sub == "":
            t = "Please enter the subject name!!!"
            text_to_speech(t)
        else:
            try:
                recognizer = cv2.face.LBPHFaceRecognizer_create()
                try:
                    recognizer.read(trainimagelabel_path)
                except:
                    e = "Model not found,please train model"
                    Notifica.configure(
                        text=e,
                        bg="black",
                        fg="yellow",
                        width=33,
                        font=("times", 15, "bold"),
                    )
                    Notifica.place(x=20, y=250)
                    text_to_speech(e)
                facecasCade = cv2.CascadeClassifier(haarcasecade_path)
                df = pd.read_csv(studentdetail_path)
                cam = cv2.VideoCapture(0, cv2.CAP_DSHOW) #capture device = camera
                font = cv2.FONT_HERSHEY_SIMPLEX
                col_names = ["Enrollment", "Name"]
                attendance = pd.DataFrame(columns=col_names)
```


C:\Users\nxtbi\OneDrive\Desktop\Attendance-Management-system-using-face-recognition\automaticAttedance.py - Sublime Text (UNREGISTERED)
train.py — final_proj\Face-Recognition-Based-Attendance-System-mas
attendance.py
studentdetails.csv
automaticAttedance.py
show_attendance.py
takeImage.py
takemanually.py
test.py
trainImage.py
_config.yml
joseph.py
joseph2.py
prims.py
bfs.py
Line 14, Column 110
Spaces: 4
Python

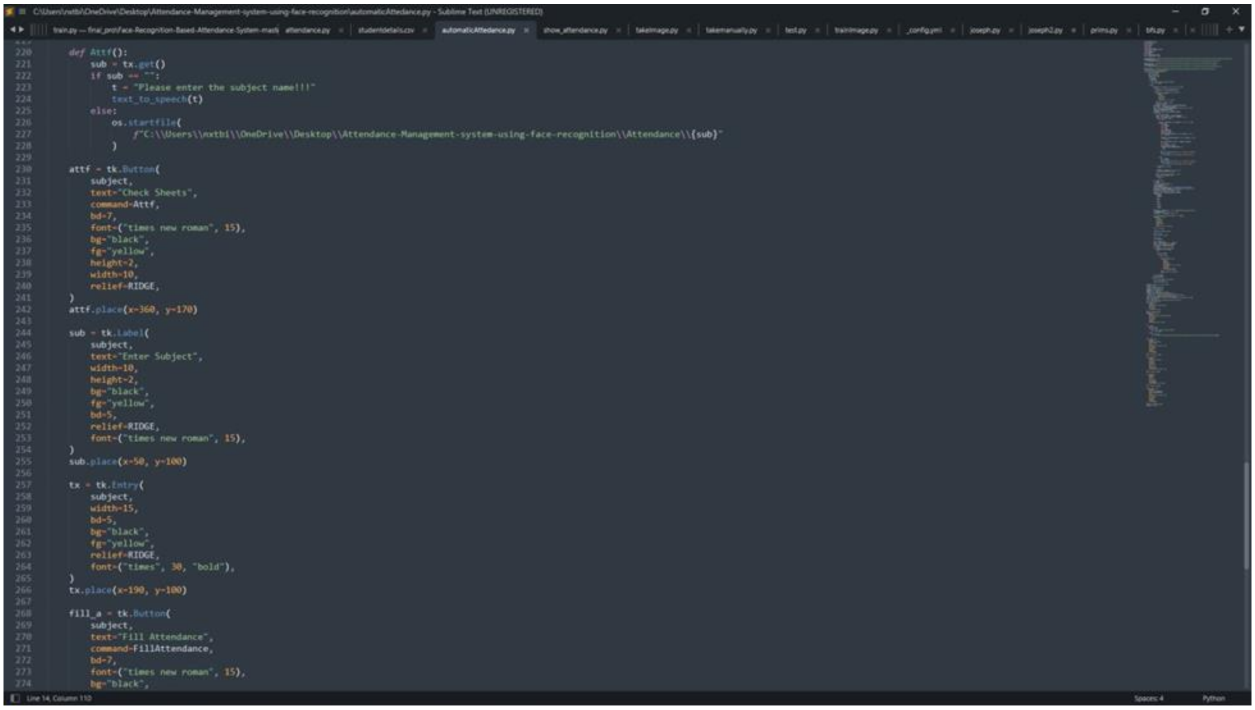
```python
                attendance = pd.DataFrame(columns=col_names)
                while True:
                    ___, im = cam.read()
                    gray = cv2.cvtColor(im, cv2.COLOR_BGR2GRAY)
                    faces = facecasCade.detectMultiScale(gray, 1.2, 5)
                    for (x, y, w, h) in faces:
                        global Id

                        Id, conf = recognizer.predict(gray[y : y + h, x : x + w])
                        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
                            global tt
                            tt = str(Id) + "-" + aa
                            # En='1604501160'+str(Id)
                            attendance.loc[len(attendance)] = [
                                Id,
                                aa,
                            ]
                            cv2.rectangle(im, (x, y), (x + w, y + h), (0, 260, 0), 4)
                            cv2.putText(
                                im, str(tt), (x + h, y), font, 1, (255, 255, 0,), 4
                            )
                        else:
                            Id = "Unknown"
                            tt = str(Id)
                            cv2.rectangle(im, (x, y), (x + w, y + h), (0, 25, 255), 7)
                            cv2.putText(
                                im, str(tt), (x + h, y), font, 1, (0, 25, 255), 4
                            )
                    if time.time() > future:
                        break

                    attendance = attendance.drop_duplicates(
                        ["Enrollment"], keep="first"
                    )
                    cv2.imshow("Filling Attendance...", im)
                    key = cv2.waitKey(30) & 0xFF
                    if key == 27:
                        break

                ts = time.time()
                print(aa)
                # attendance["date"] = date
```


C:\Users\vetbi\OneDrive\Desktop\Attendance-Management-system-using-face-recognition\automaticAttedance.py - Sublime Text (UNREGISTERED)
train.py — final_proj\Face-Recognition-Based-Attendance-System-mask
attendance.py
studentdetails.csv
automaticAttedance.py
show_attendance.py
takeImage.py
takemanually.py
test.py
trainImage.py
_config.yml
joseph.py
joseph2.py
prims.py
bfs.py
Line 14, Column 110
Spaces: 4
Python

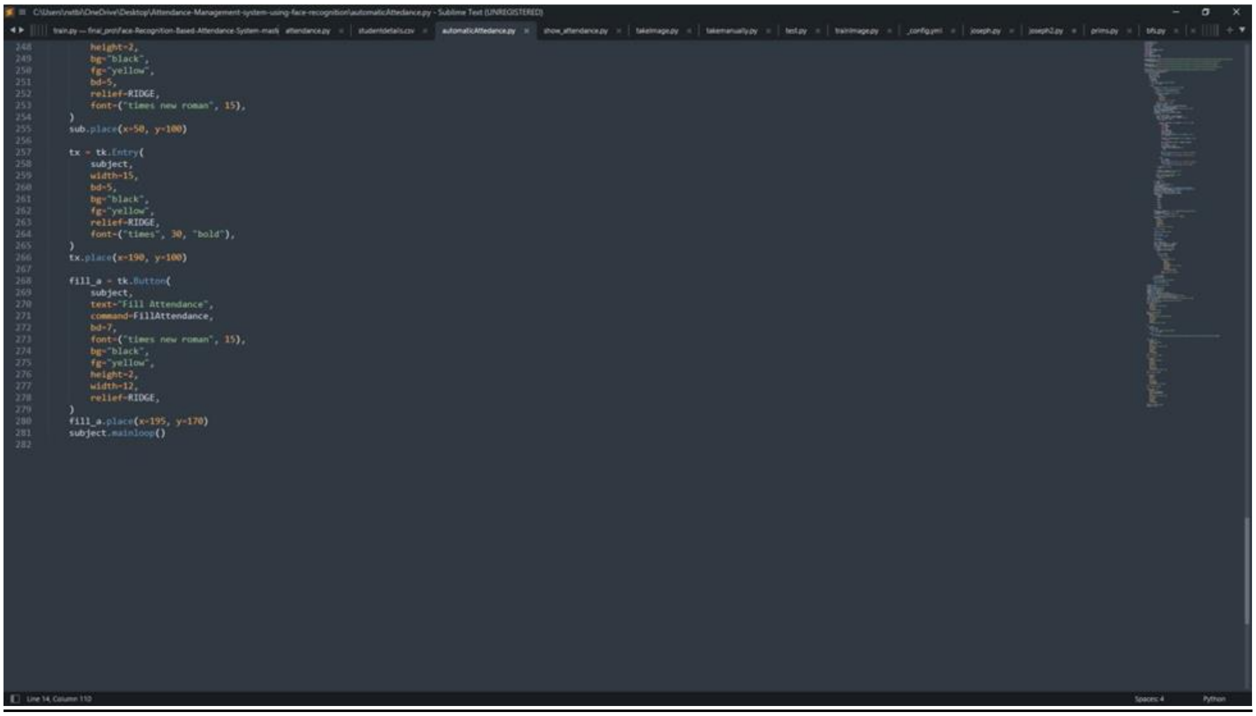
```
                # attendance["date"] = date
                # attendance["Attendance"] = "P"
                attendance[date] = 1
                date = datetime.datetime.fromtimestamp(ts).strftime("%Y-%m-%d")
                timeStamp = datetime.datetime.fromtimestamp(ts).strftime("%H:%M:%S")
                Hour, Minute, Second = timeStamp.split(":")
                #fileName = "Attendance/" + Subject + ".csv"
                path = os.path.join(attendance_path, Subject)
                fileName = (
                    f"{path}/"
                    + Subject
                    + "_"
                    + date
                    + "_"
                    + Hour
                    + "-"
                    + Minute
                    + "-"
                    + Second
                    + ".csv"
                )
                attendance = attendance.drop_duplicates(["Enrollment"], keep="first")
                print(attendance)
                attendance.to_csv(fileName, index=False)

                m = "Attendance Filled Successfully of " + Subject
                Notifica.configure(
                    text=m,
                    bg="black",
                    fg="yellow",
                    width=33,
                    relief=RIDGE,
                    bd=5,
                    font=("times", 15, "bold"),
                )
                text_to_speech(m)

                Notifica.place(x=20, y=250)

                cam.release()
                cv2.destroyAllWindows()

                import csv
                import tkinter

                root = tkinter.Tk()
                root.title("Attendance of " + Subject)
                root.configure(background="black")
                cs = os.path.join(path, fileName)
                print(cs)
                with open(cs, newline="") as file:
                    reader = csv.reader(file)
                    r = 0

                    for col in reader:
```


C:\Users\vetbi\OneDrive\Desktop\Attendance-Management-system-using-face-recognition\automaticAttedance.py - Sublime Text (UNREGISTERED)
train.py — final_proj\Face-Recognition-Based-Attendance-System-mask
attendance.py
studentdetails.csv
automaticAttedance.py
show_attendance.py
takeImage.py
takemanually.py
test.py
trainImage.py
_config.yml
joseph.py
joseph2.py
prims.py
bfs.py
Line 14, Column 110
Spaces: 4
Python

```
                    for col in reader:
                        c = 0
                        for row in col:

                            label = tkinter.Label(
                                root,
                                width=10,
                                height=1,
                                fg="yellow",
                                font=("times", 15, " bold "),
                                bg="black",
                                text=row,
                                relief=tkinter.RIDGE,
                            )
                            label.grid(row=r, column=c)
                            c += 1
                        r += 1
                root.mainloop()
                print(attendance)
            except:
                f = "No Face found for attendance"
                text_to_speech(f)
                cv2.destroyAllWindows()

    ###window is frame for subject chooser
    subject = Tk()
    # windo.iconbitmap("AMS.ico")
    subject.title("Subject...")
    subject.geometry("580x320")
    subject.resizable(0, 0)
    subject.configure(background="black")
    # subject_logo = Image.open("UI_Image/0004.png")
    # subject_logo = subject_logo.resize((50, 47), Image.ANTIALIAS)
    # subject_logo1 = ImageTk.PhotoImage(subject_logo)
    titl = tk.Label(subject, bg="black", relief=RIDGE, bd=10, font=("arial", 30))
    titl.pack(fill=X)
    # l1 = tk.Label(subject, image=subject_logo1, bg="black",)
    # l1.place(x=100, y=10)
    titl = tk.Label(
        subject,
        text="Enter the Subject Name",
        bg="black",
        fg="green",
        font=("arial", 25),
    )
    titl.place(x=160, y=12)
    Notifica = tk.Label(
        subject,
        text="Attendance filled Successfully",
        bg="yellow",
        fg="black",
        width=33,
        height=2,
        font=("times", 15, "bold"),
    )
```

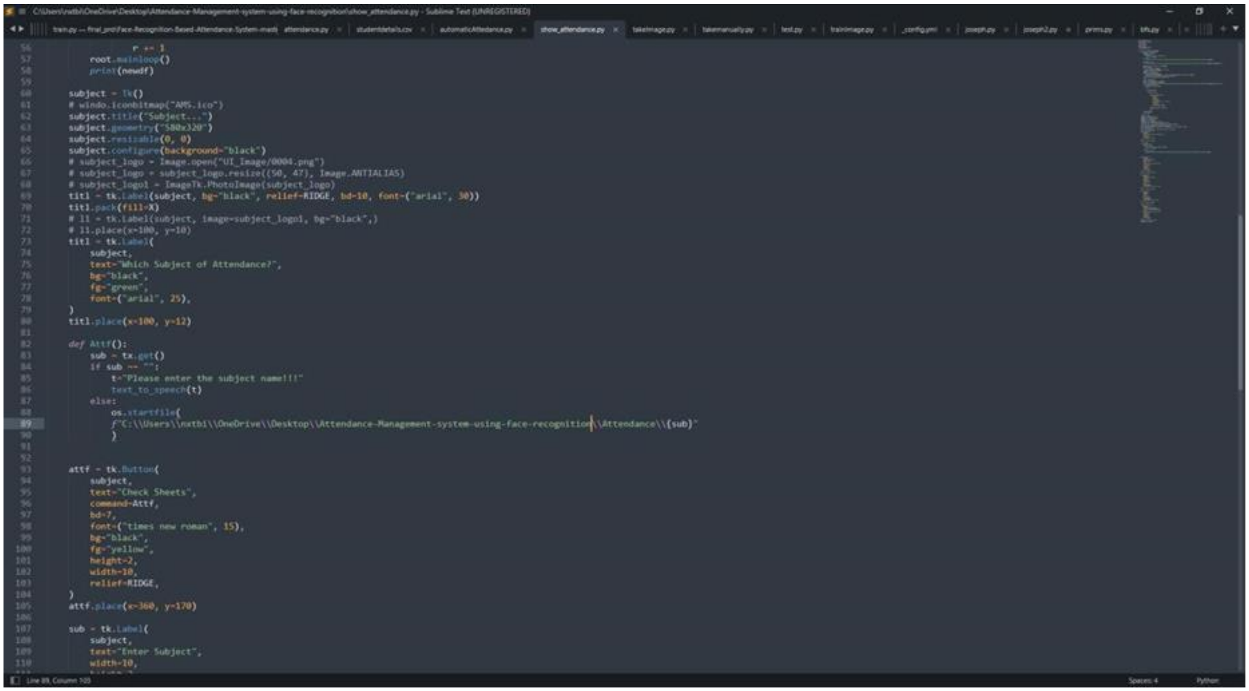

C:\Users\nxtbi\OneDrive\Desktop\Attendance-Management-system-using-face-recognition\automaticAttedance.py - Sublime Text (UNREGISTERED)
train.py — final_proj\Face-Recognition-Based-Attendance-System-mast
attendance.py
studentdetails.csv
automaticAttedance.py
show_attendance.py
takeImage.py
takemanually.py
test.py
trainImage.py
_config.yml
joseph.py
joseph2.py
prims.py
bfs.py
Line 14, Column 110
Spaces: 4
Python

```
    def Attf():
        sub = tx.get()
        if sub == "":
            t = "Please enter the subject name!!!"
            text_to_speech(t)
        else:
            os.startfile(
                f"C:\\Users\\nxtbi\\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=360, 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",
```

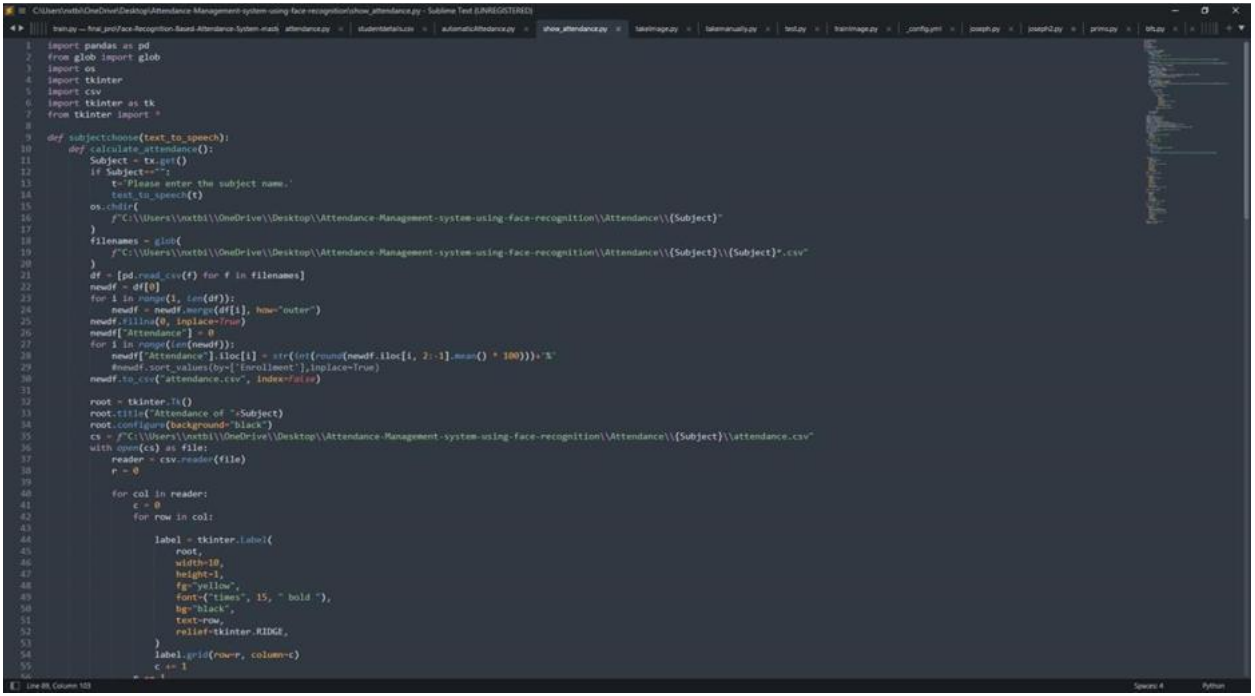

C:\Users\nxtbi\OneDrive\Desktop\Attendance-Management-system-using-face-recognition\automaticAttedance.py - Sublime Text (UNREGISTERED)
train.py — final_proj\Face-Recognition-Based-Attendance-System-mast
attendance.py
studentdetails.csv
automaticAttedance.py
show_attendance.py
takeImage.py
takemanually.py
test.py
trainImage.py
_config.yml
joseph.py
joseph2.py
prims.py
bfs.py
Line 14, Column 110
Spaces: 4
Python

```
        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()
```

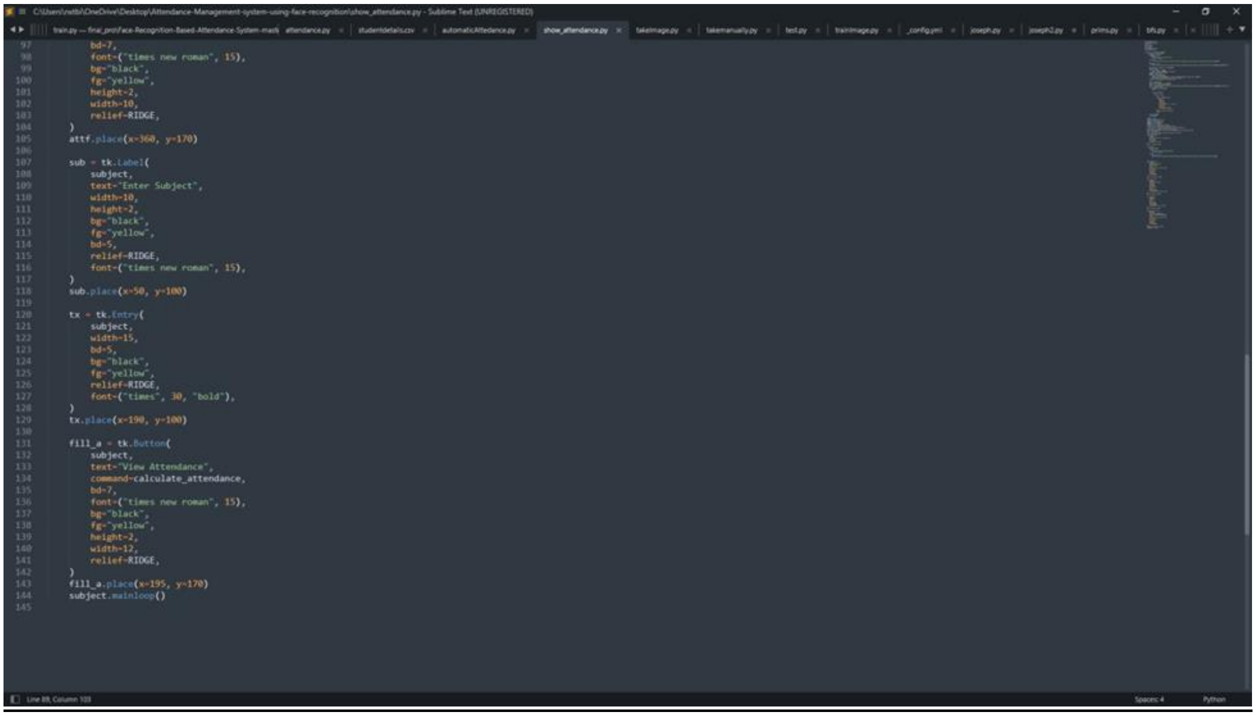
# show_attendance.py


C:\Users\nxtbi\OneDrive\Desktop\Attendance-Management-system-using-face-recognition\show_attendance.py - Sublime Text (UNREGISTERED)
train.py — final_proj\Face-Recognition-Based-Attendance-System-mast
attendance.py
studentdetails.csv
automaticAttedance.py
show_attendance.py
takeImage.py
takemanually.py
test.py
trainImage.py
_config.yml
joseph.py
joseph2.py
prims.py
bfs.py
Line 89, Column 103
Spaces: 4
Python

```python
                r += 1
        root.mainloop()
        print(newdf)

    subject = Tk()
    # windo.iconbitmap("AMS.ico")
    subject.title("Subject...")
    subject.geometry("580x320")
    subject.resizable(0, 0)
    subject.configure(background="black")
    # subject_logo = Image.open("UI_Image/0004.png")
    # subject_logo = subject_logo.resize((50, 47), Image.ANTIALIAS)
    # subject_logo1 = ImageTk.PhotoImage(subject_logo)
    titl = tk.Label(subject, bg="black", relief=RIDGE, bd=10, font=("arial", 30))
    titl.pack(fill=X)
    # l1 = tk.Label(subject, image=subject_logo1, bg="black",)
    # l1.place(x=100, y=10)
    titl = tk.Label(
        subject,
        text="Which Subject of Attendance?",
        bg="black",
        fg="green",
        font=("arial", 25),
    )
    titl.place(x=100, y=12)

    def Attf():
        sub = tx.get()
        if sub == "":
            t="Please enter the subject name!!!"
            text_to_speech(t)
        else:
            os.startfile(
            f"C:\\Users\\nxtbi\\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=360, y=170)

    sub = tk.Label(
        subject,
        text="Enter Subject",
        width=10,
```

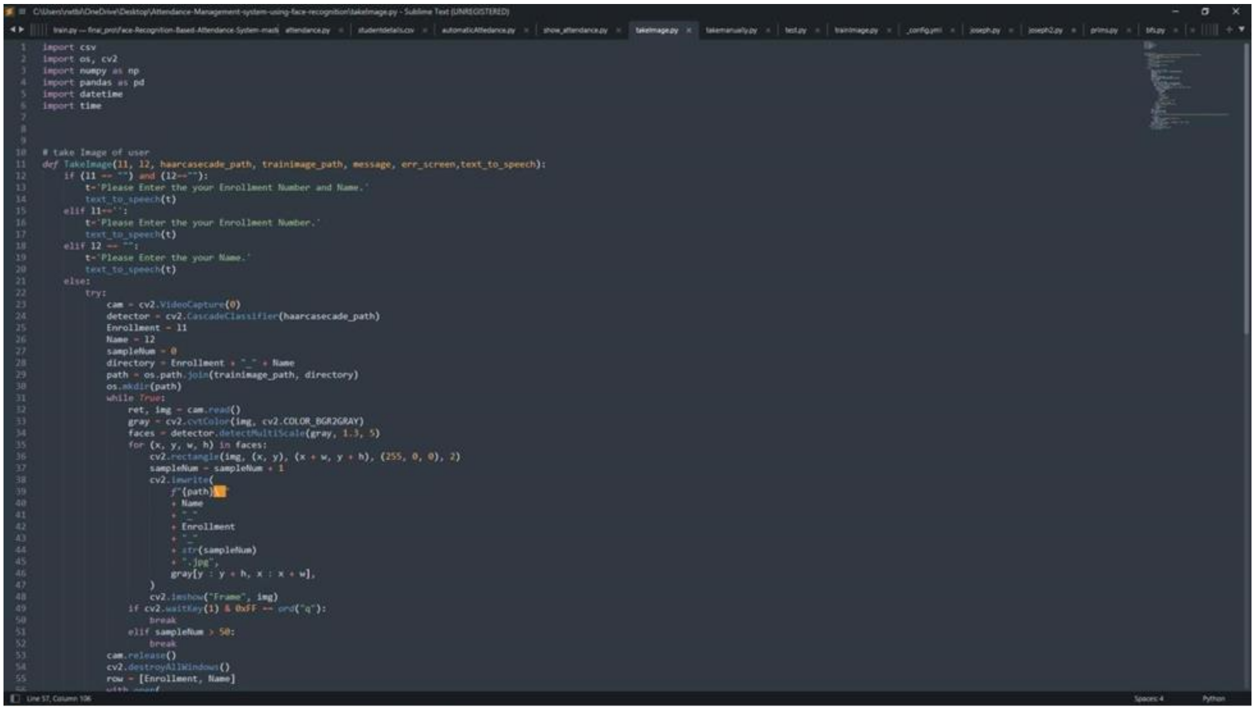

C:\Users\nxtbi\OneDrive\Desktop\Attendance-Management-system-using-face-recognition\show_attendance.py - Sublime Text (UNREGISTERED)
train.py — final_proj\Face-Recognition-Based-Attendance-System-mast
attendance.py
studentdetails.csv
automaticAttedance.py
show_attendance.py
takeImage.py
takemanually.py
test.py
trainImage.py
_config.yml
joseph.py
joseph2.py
prims.py
bfs.py
Line 89, Column 103
Spaces: 4
Python

```python
import pandas as pd
from glob import glob
import os
import tkinter
import csv
import tkinter as tk
from tkinter import *

def subjectchoose(text_to_speech):
    def calculate_attendance():
        Subject = tx.get()
        if Subject=="":
            t='Please enter the subject name.'
            text_to_speech(t)
        os.chdir(
            f"C:\\Users\\nxtbi\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\Attendance\\{Subject}"
        )
        filenames = glob(
            f"C:\\Users\\nxtbi\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\Attendance\\{Subject}\\{Subject}*.csv"
        )
        df = [pd.read_csv(f) for f in filenames]
        newdf = df[0]
        for i in range(1, len(df)):
            newdf = newdf.merge(df[i], how="outer")
        newdf.fillna(0, inplace=True)
        newdf["Attendance"] = 0
        for i in range(len(newdf)):
            newdf["Attendance"].iloc[i] = str(int(round(newdf.iloc[i, 2:-1].mean() * 100)))+'%'
            #newdf.sort_values(by=['Enrollment'],inplace=True)
        newdf.to_csv("attendance.csv", index=False)

        root = tkinter.Tk()
        root.title("Attendance of "+Subject)
        root.configure(background="black")
        cs = f"C:\\Users\\nxtbi\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\Attendance\\{Subject}\\attendance.csv"
        with open(cs) as file:
            reader = csv.reader(file)
            r = 0

            for col in reader:
                c = 0
                for row in col:

                    label = tkinter.Label(
                        root,
                        width=10,
                        height=1,
                        fg="yellow",
                        font=("times", 15, " bold "),
                        bg="black",
                        text=row,
                        relief=tkinter.RIDGE,
                    )
                    label.grid(row=r, column=c)
                    c += 1
```

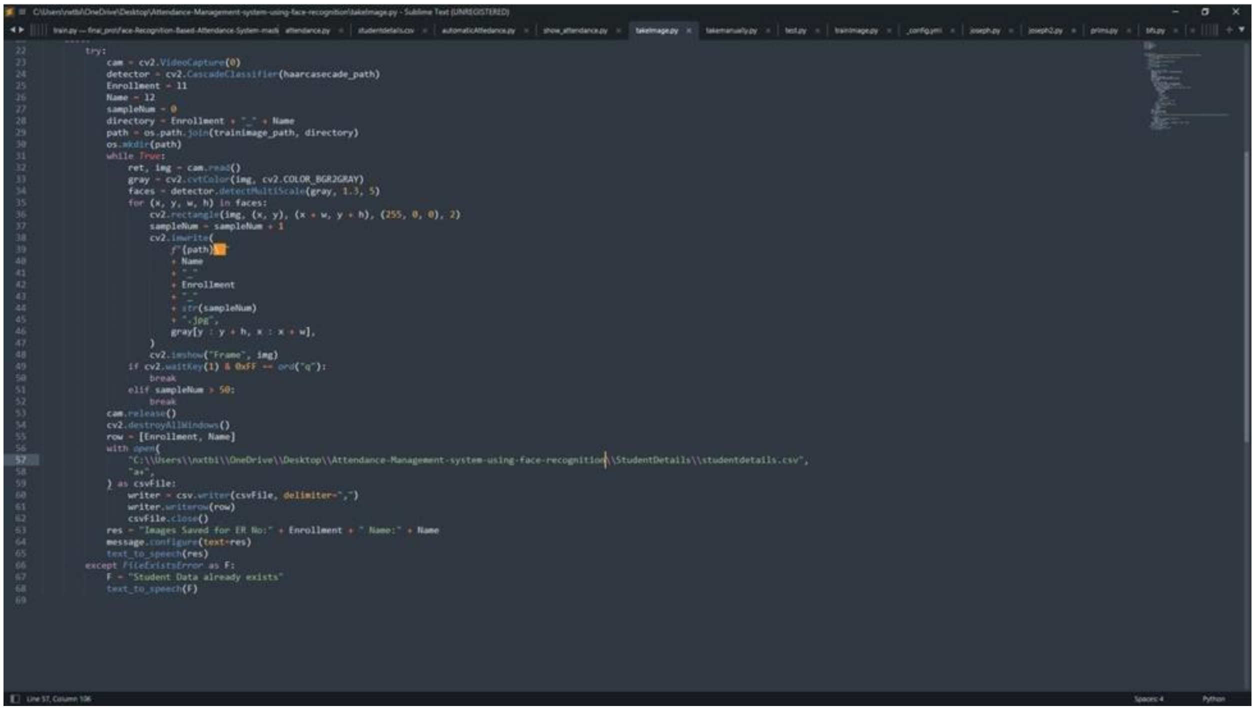

C:\Users\nebi\OneDrive\Desktop\Attendance-Management-system-using-face-recognition\show_attendance.py - Sublime Text (UNREGISTERED)
train.py — final_proj\Face-Recognition-Based-Attendance-System-mast
attendance.py
studentdetails.csv
automaticAttedance.py
show_attendance.py
takeImage.py
takemanually.py
test.py
trainImage.py
_config.yml
joseph.py
joseph2.py
prims.py
bfs.py
Line 89, Column 103
Spaces: 4
Python

```
        bd=7,
        font=("times new roman", 15),
        bg="black",
        fg="yellow",
        height=2,
        width=10,
        relief=RIDGE,
    )
    attf.place(x=360, 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="View Attendance",
        command=calculate_attendance,
        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()
```

## takeImage.py



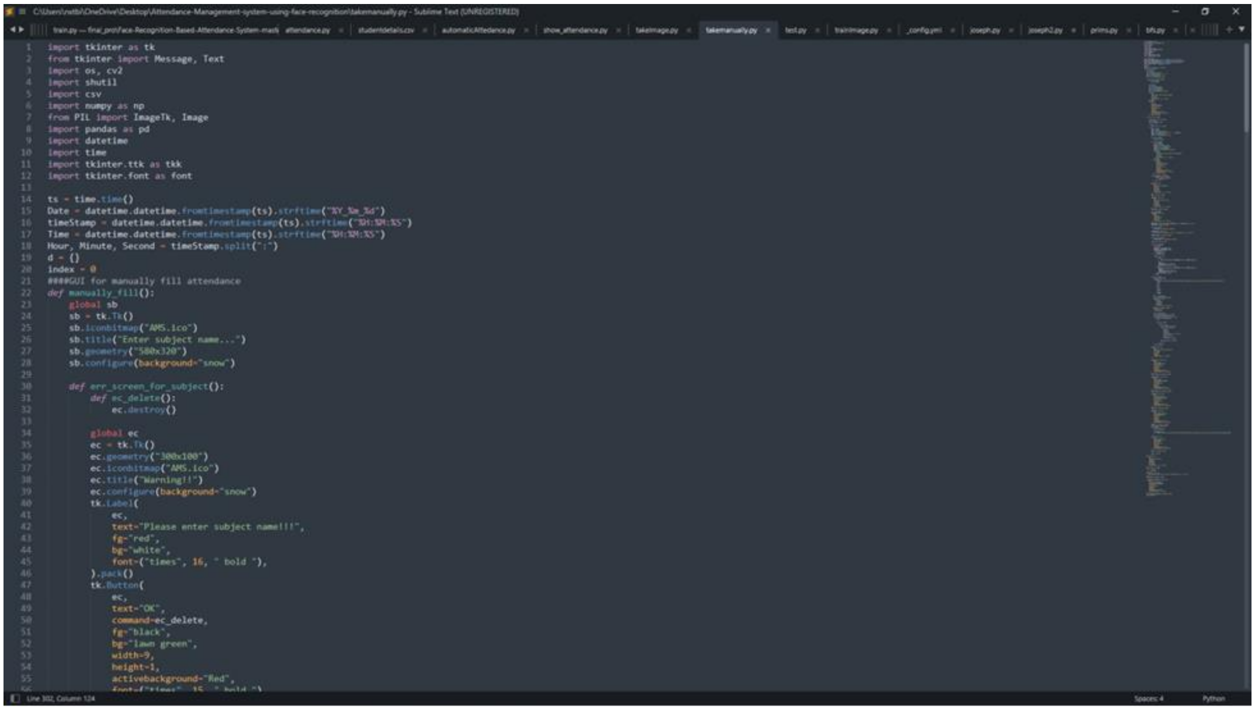
```python
import csv
import os, cv2
import numpy as np
import pandas as pd
import datetime
import time



# take Image of user
def TakeImage(l1, l2, haarcasecade_path, trainimage_path, message, err_screen,text_to_speech):
    if (l1 == "") and (l2==""):
        t='Please Enter the your Enrollment Number and Name.'
        text_to_speech(t)
    elif l1=='':
        t='Please Enter the your Enrollment Number.'
        text_to_speech(t)
    elif l2 == "":
        t='Please Enter the your Name.'
        text_to_speech(t)
    else:
        try:
            cam = cv2.VideoCapture(0)
            detector = cv2.CascadeClassifier(haarcasecade_path)
            Enrollment = l1
            Name = l2
            sampleNum = 0
            directory = Enrollment + "_" + Name
            path = os.path.join(trainimage_path, directory)
            os.mkdir(path)
            while True:
                ret, img = cam.read()
                gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
                faces = detector.detectMultiScale(gray, 1.3, 5)
                for (x, y, w, h) in faces:
                    cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 2)
                    sampleNum = sampleNum + 1
                    cv2.imwrite(
                        f"{path}\\"
                        + Name
                        + "_"
                        + Enrollment
                        + "_"
                        + str(sampleNum)
                        + ".jpg",
                        gray[y : y + h, x : x + w],
                    )
                    cv2.imshow("Frame", img)
                if cv2.waitKey(1) & 0xFF == ord("q"):
                    break
                elif sampleNum > 50:
                    break
            cam.release()
            cv2.destroyAllWindows()
            row = [Enrollment, Name]
```



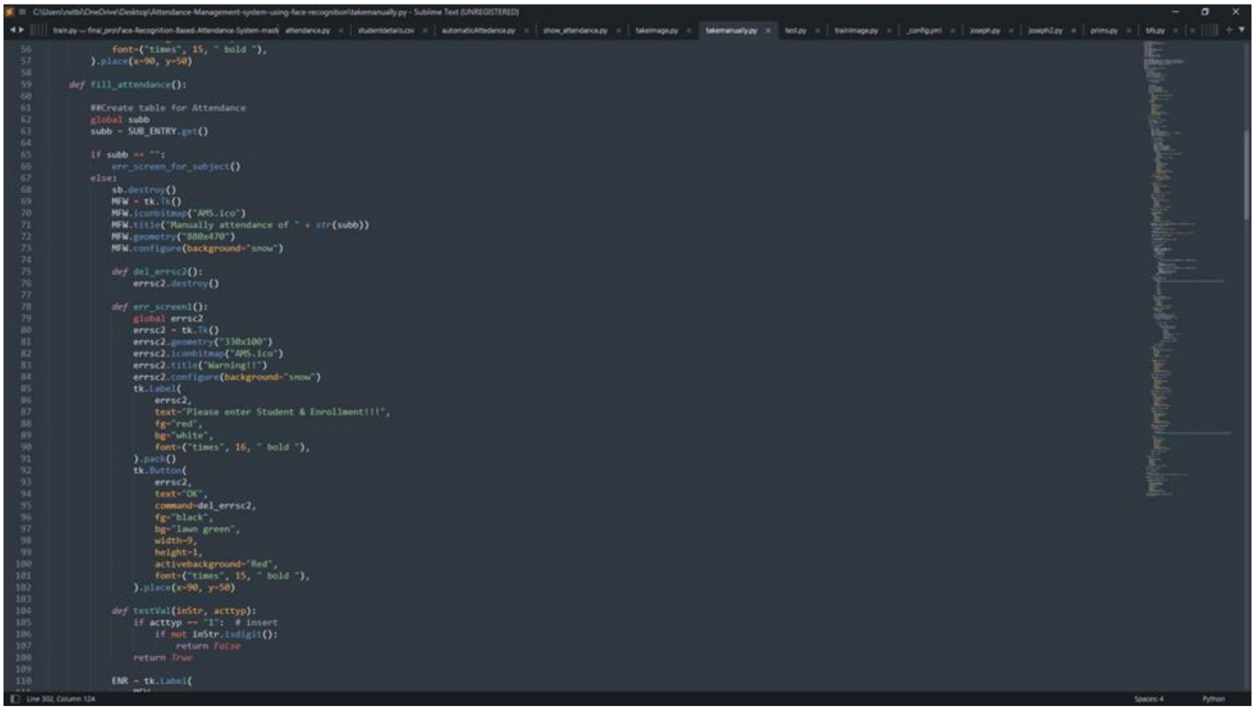
```python
        try:
            cam = cv2.VideoCapture(0)
            detector = cv2.CascadeClassifier(haarcasecade_path)
            Enrollment = l1
            Name = l2
            sampleNum = 0
            directory = Enrollment + "_" + Name
            path = os.path.join(trainimage_path, directory)
            os.mkdir(path)
            while True:
                ret, img = cam.read()
                gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
                faces = detector.detectMultiScale(gray, 1.3, 5)
                for (x, y, w, h) in faces:
                    cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 2)
                    sampleNum = sampleNum + 1
                    cv2.imwrite(
                        f"{path}\\"
                        + Name
                        + "_"
                        + Enrollment
                        + "_"
                        + str(sampleNum)
                        + ".jpg",
                        gray[y : y + h, x : x + w],
                    )
                    cv2.imshow("Frame", img)
                if cv2.waitKey(1) & 0xFF == ord("q"):
                    break
                elif sampleNum > 50:
                    break
            cam.release()
            cv2.destroyAllWindows()
            row = [Enrollment, Name]
```

```python
            with open(
                "C:\\Users\\nxtbi\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\StudentDetails\\studentdetails.csv",
                "a+",
            ) as csvFile:
                writer = csv.writer(csvFile, delimiter=",")
                writer.writerow(row)
                csvFile.close()
            res = "Images Saved for ER No:" + Enrollment + " Name:" + Name
            message.configure(text=res)
            text_to_speech(res)
        except FileExistsError as F:
            F = "Student Data already exists"
            text_to_speech(F)
```

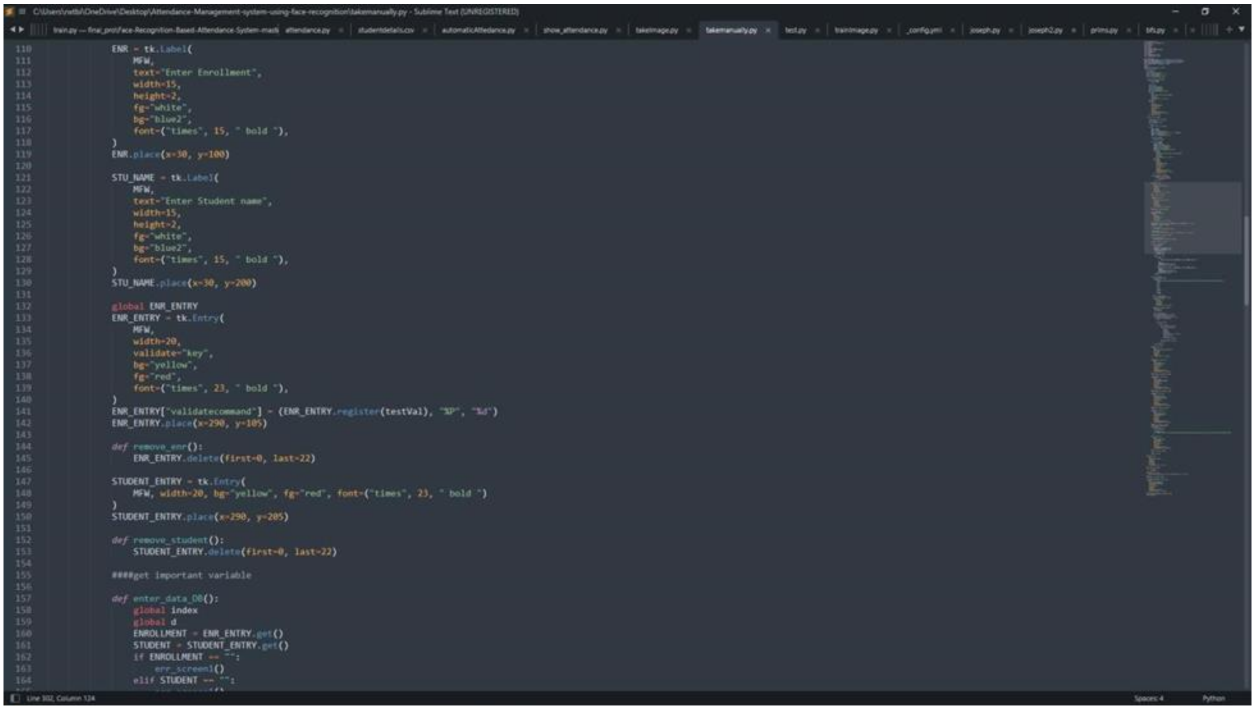
## takemanually.py


C:\Users\netbi\OneDrive\Desktop\Attendance-Management-system-using-face-recognition\takemanually.py - Sublime Text (UNREGISTERED)
Line 302, Column 124
Spaces: 4
Python

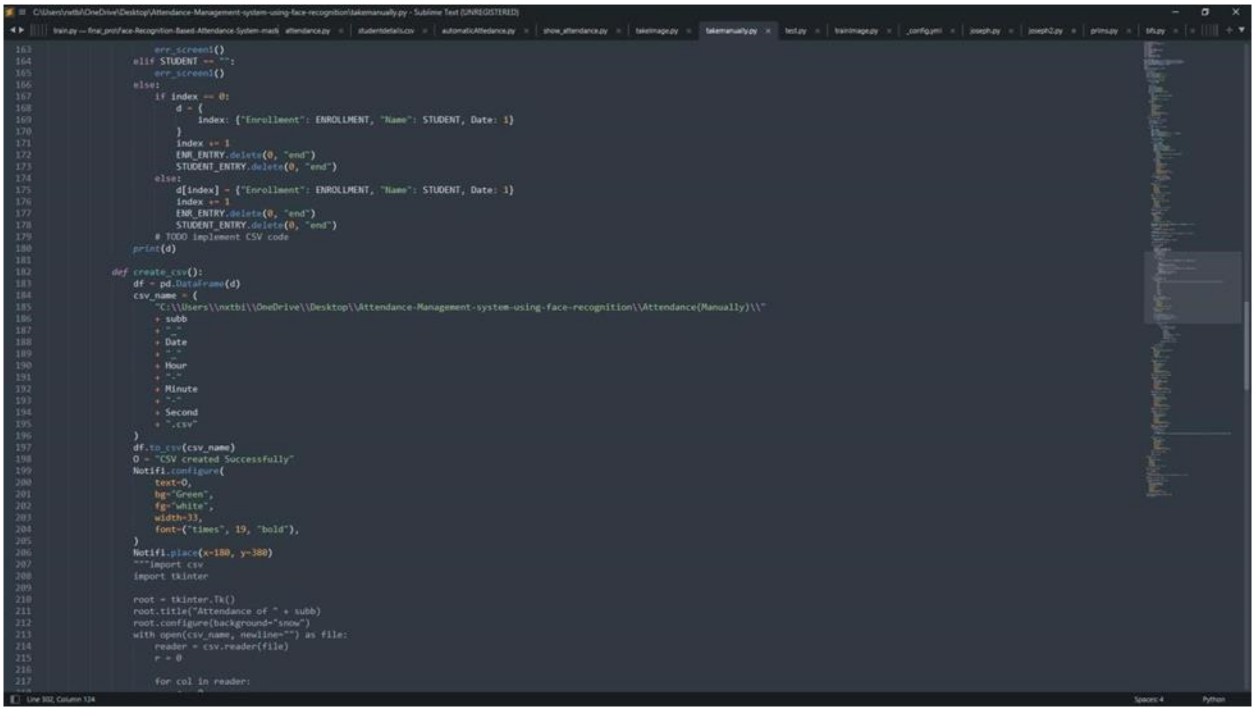
```python
import tkinter as tk
from tkinter import Message, Text
import os, cv2
import shutil
import csv
import numpy as np
from PIL import ImageTk, Image
import pandas as pd
import datetime
import time
import tkinter.ttk as tkk
import tkinter.font as font

ts = time.time()
Date = datetime.datetime.fromtimestamp(ts).strftime("%Y_%m_%d")
timeStamp = datetime.datetime.fromtimestamp(ts).strftime("%H:%M:%S")
Time = datetime.datetime.fromtimestamp(ts).strftime("%H:%M:%S")
Hour, Minute, Second = timeStamp.split(":")
d = {}
index = 0
####GUI for manually fill attendance
def manually_fill():
    global sb
    sb = tk.Tk()
    sb.iconbitmap("AMS.ico")
    sb.title("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.iconbitmap("AMS.ico")
        ec.title("Warning!!")
        ec.configure(background="snow")
        tk.Label(
            ec,
            text="Please enter subject name!!!",
            fg="red",
            bg="white",
            font=("times", 16, " bold "),
        ).pack()
        tk.Button(
            ec,
            text="OK",
            command=ec_delete,
            fg="black",
            bg="lawn green",
            width=9,
            height=1,
            activebackground="Red",
```


C:\Users\netbi\OneDrive\Desktop\Attendance-Management-system-using-face-recognition\takemanually.py - Sublime Text (UNREGISTERED)
Line 302, Column 124
Spaces: 4
Python

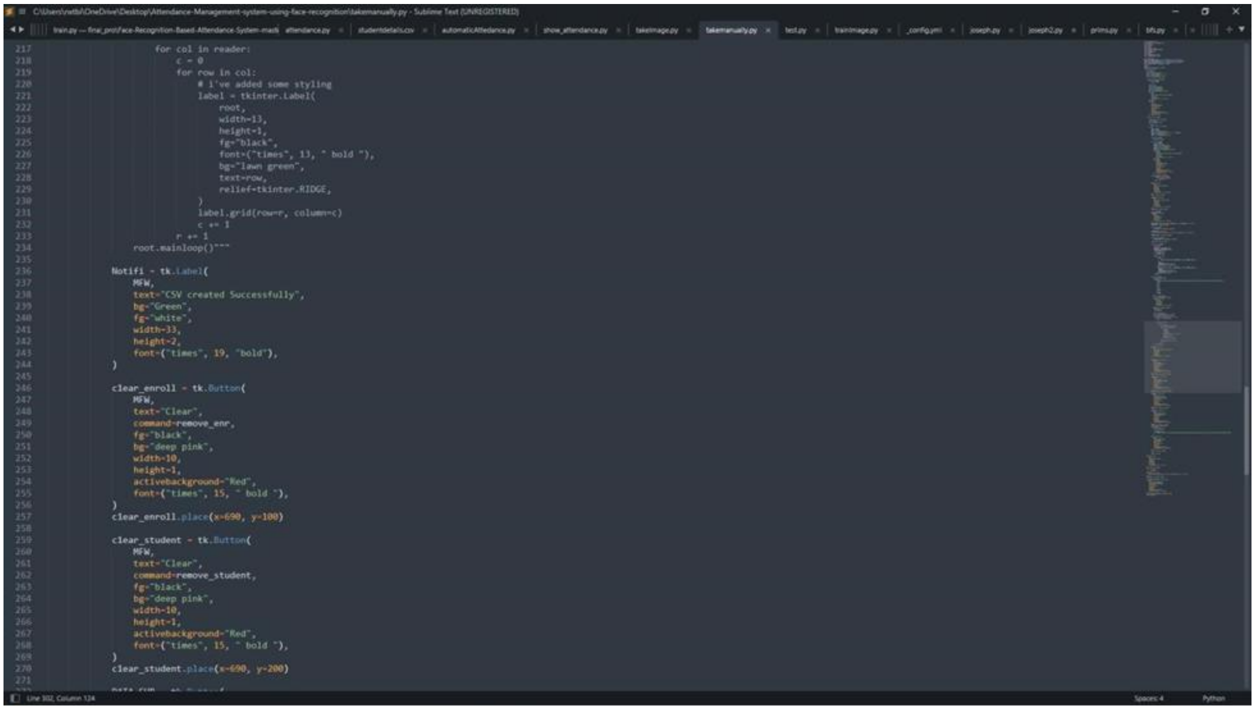
```python
            font=("times", 15, " bold "),
        ).place(x=90, y=50)

    def fill_attendance():

        ##Create table for Attendance
        global subb
        subb = SUB_ENTRY.get()

        if subb == "":
            err_screen_for_subject()
        else:
            sb.destroy()
            MFW = tk.Tk()
            MFW.iconbitmap("AMS.ico")
            MFW.title("Manually attendance of " + str(subb))
            MFW.geometry("880x470")
            MFW.configure(background="snow")

            def del_errsc2():
                errsc2.destroy()

            def err_screen1():
                global errsc2
                errsc2 = tk.Tk()
                errsc2.geometry("330x100")
                errsc2.iconbitmap("AMS.ico")
                errsc2.title("Warning!!")
                errsc2.configure(background="snow")
                tk.Label(
                    errsc2,
                    text="Please enter Student & Enrollment!!!",
                    fg="red",
                    bg="white",
                    font=("times", 16, " bold "),
                ).pack()
                tk.Button(
                    errsc2,
                    text="OK",
                    command=del_errsc2,
                    fg="black",
                    bg="lawn green",
                    width=9,
                    height=1,
                    activebackground="Red",
                    font=("times", 15, " bold "),
                ).place(x=90, y=50)

            def testVal(inStr, acttyp):
                if acttyp == "1":  # insert
                    if not inStr.isdigit():
                        return False
                return True

            ENR = tk.Label(
```


C:\Users\nxtbi\OneDrive\Desktop\Attendance-Management-system-using-face-recognition\takemanually.py - Sublime Text (UNREGISTERED)
train.py — final_proj\Face-Recognition-Based-Attendance-System-mast
attendance.py
studentdetails.csv
automaticAttedance.py
show_attendance.py
takeImage.py
takemanually.py
test.py
trainImage.py
_config.yml
joseph.py
joseph2.py
prims.py
bfs.py
Line 302, Column 124
Spaces: 4
Python

```python
        ENR = tk.Label(
            MFW,
            text="Enter Enrollment",
            width=15,
            height=2,
            fg="white",
            bg="blue2",
            font=("times", 15, " bold "),
        )
        ENR.place(x=30, y=100)

        STU_NAME = tk.Label(
            MFW,
            text="Enter Student name",
            width=15,
            height=2,
            fg="white",
            bg="blue2",
            font=("times", 15, " bold "),
        )
        STU_NAME.place(x=30, y=200)

        global ENR_ENTRY
        ENR_ENTRY = tk.Entry(
            MFW,
            width=20,
            validate="key",
            bg="yellow",
            fg="red",
            font=("times", 23, " bold "),
        )
        ENR_ENTRY["validatecommand"] = (ENR_ENTRY.register(testVal), "%P", "%d")
        ENR_ENTRY.place(x=290, y=105)

        def remove_enr():
            ENR_ENTRY.delete(first=0, last=22)

        STUDENT_ENTRY = tk.Entry(
            MFW, width=20, bg="yellow", fg="red", font=("times", 23, " bold ")
        )
        STUDENT_ENTRY.place(x=290, y=205)

        def remove_student():
            STUDENT_ENTRY.delete(first=0, last=22)

        ####get important variable

        def enter_data_DB():
            global index
            global d
            ENROLLMENT = ENR_ENTRY.get()
            STUDENT = STUDENT_ENTRY.get()
            if ENROLLMENT == "":
                err_screen1()
            elif STUDENT == "":
```

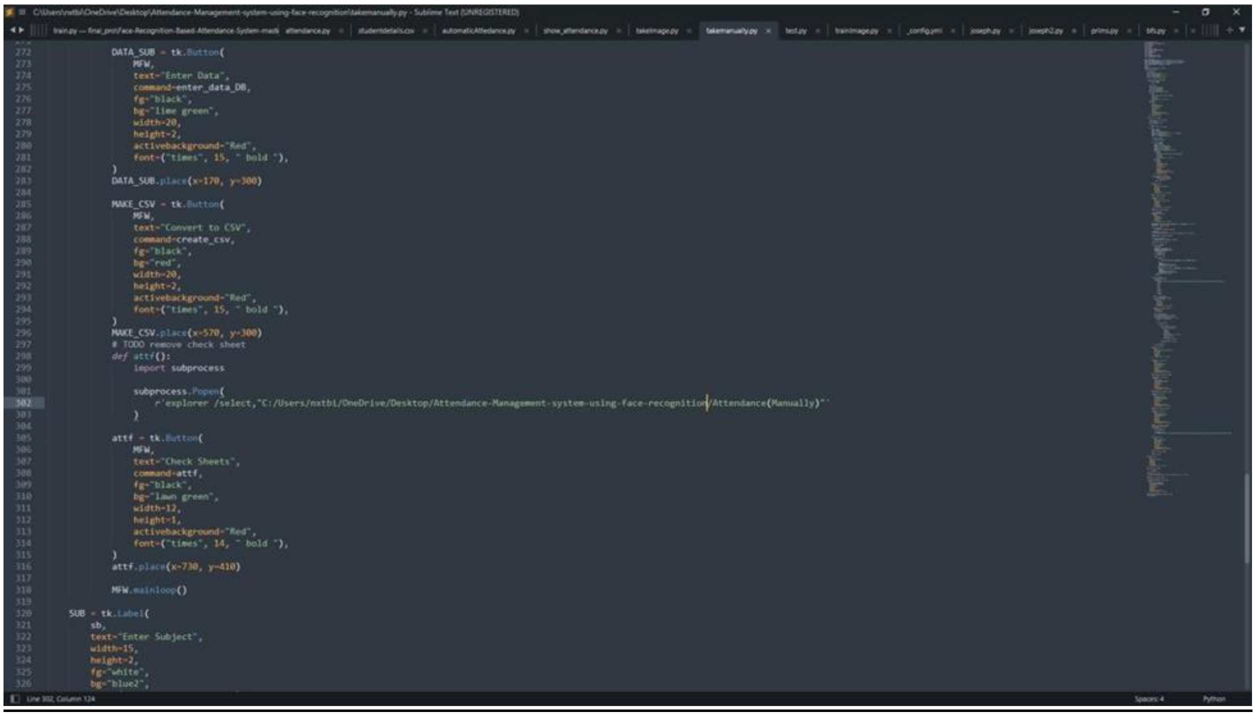

C:\Users\nxtbi\OneDrive\Desktop\Attendance-Management-system-using-face-recognition\takemanually.py - Sublime Text (UNREGISTERED)
train.py — final_proj\Face-Recognition-Based-Attendance-System-mast
attendance.py
studentdetails.csv
automaticAttedance.py
show_attendance.py
takeImage.py
takemanually.py
test.py
trainImage.py
_config.yml
joseph.py
joseph2.py
prims.py
bfs.py
Line 302, Column 124
Spaces: 4
Python

```python
                err_screen1()
            elif STUDENT == "":
                err_screen1()
            else:
                if index == 0:
                    d = {
                        index: {"Enrollment": ENROLLMENT, "Name": STUDENT, Date: 1}
                    }
                    index += 1
                    ENR_ENTRY.delete(0, "end")
                    STUDENT_ENTRY.delete(0, "end")
                else:
                    d[index] = {"Enrollment": ENROLLMENT, "Name": STUDENT, Date: 1}
                    index += 1
                    ENR_ENTRY.delete(0, "end")
                    STUDENT_ENTRY.delete(0, "end")
                # TODO implement CSV code
            print(d)

        def create_csv():
            df = pd.DataFrame(d)
            csv_name = (
                "C:\\Users\\nxtbi\\OneDrive\\Desktop\\Attendance-Management-system-using-face-recognition\\Attendance(Manually)\\"
                + subb
                + "_"
                + Date
                + "_"
                + Hour
                + "-"
                + Minute
                + "-"
                + Second
                + ".csv"
            )
            df.to_csv(csv_name)
            O = "CSV created Successfully"
            Notifi.configure(
                text=O,
                bg="Green",
                fg="white",
                width=33,
                font=("times", 19, "bold"),
            )
            Notifi.place(x=180, y=380)
            """import csv
            import tkinter

            root = tkinter.Tk()
            root.title("Attendance of " + subb)
            root.configure(background="snow")
            with open(csv_name, newline="") as file:
                reader = csv.reader(file)
                r = 0

                for col in reader:
```



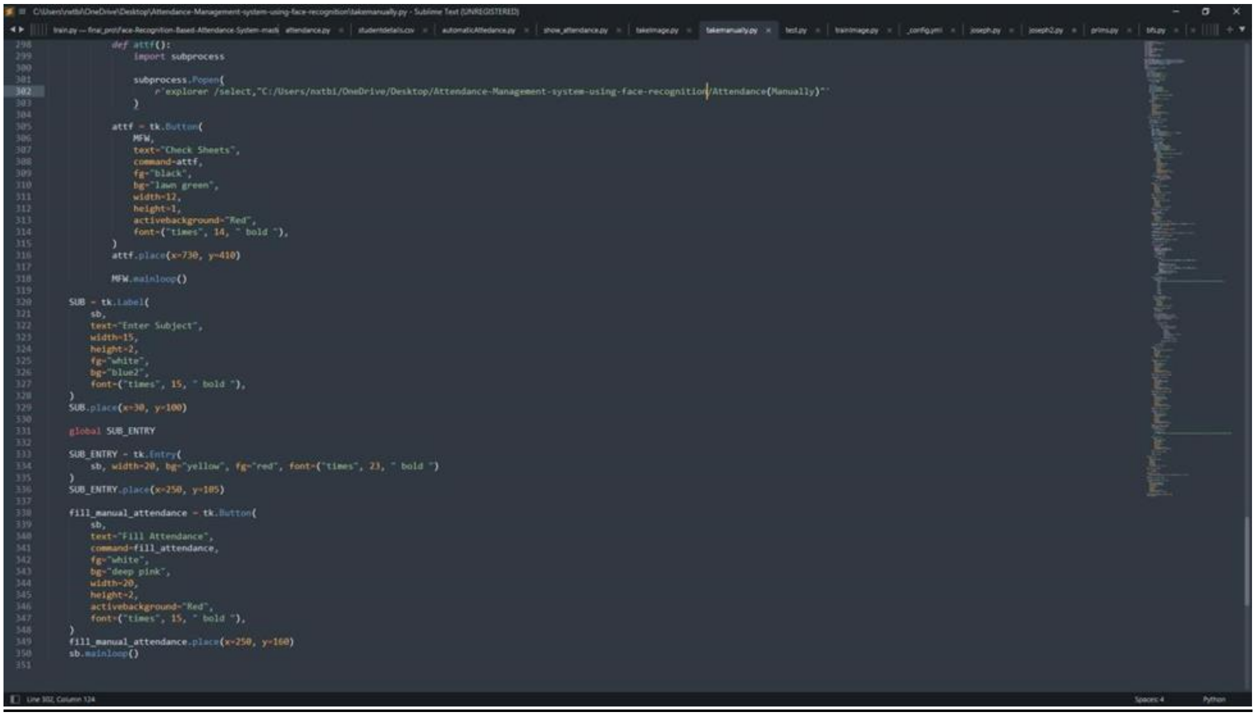
C:\Users\nxtbi\OneDrive\Desktop\Attendance-Management-system-using-face-recognition\takemanually.py - Sublime Text (UNREGISTERED)
train.py — final_proj\Face-Recognition-Based-Attendance-System-master
attendance.py
studentdetails.csv
automaticAttedance.py
show_attendance.py
takeImage.py
takemanually.py
test.py
trainImage.py
_config.yml
joseph.py
joseph2.py
prims.py
bfs.py
Line 302, Column 124
Spaces: 4
Python

```
                for col in reader:
                    c = 0
                    for row in col:
                        # i've added some styling
                        label = tkinter.Label(
                            root,
                            width=13,
                            height=1,
                            fg="black",
                            font=("times", 13, " bold "),
                            bg="lawn green",
                            text=row,
                            relief=tkinter.RIDGE,
                        )
                        label.grid(row=r, column=c)
                        c += 1
                    r += 1
            root.mainloop()"""

        Notifi = tk.Label(
            MFW,
            text="CSV created Successfully",
            bg="Green",
            fg="white",
            width=33,
            height=2,
            font=("times", 19, "bold"),
        )

        clear_enroll = tk.Button(
            MFW,
            text="Clear",
            command=remove_enr,
            fg="black",
            bg="deep pink",
            width=10,
            height=1,
            activebackground="Red",
            font=("times", 15, " bold "),
        )
        clear_enroll.place(x=690, y=100)

        clear_student = tk.Button(
            MFW,
            text="Clear",
            command=remove_student,
            fg="black",
            bg="deep pink",
            width=10,
            height=1,
            activebackground="Red",
            font=("times", 15, " bold "),
        )
        clear_student.place(x=690, y=200)
```



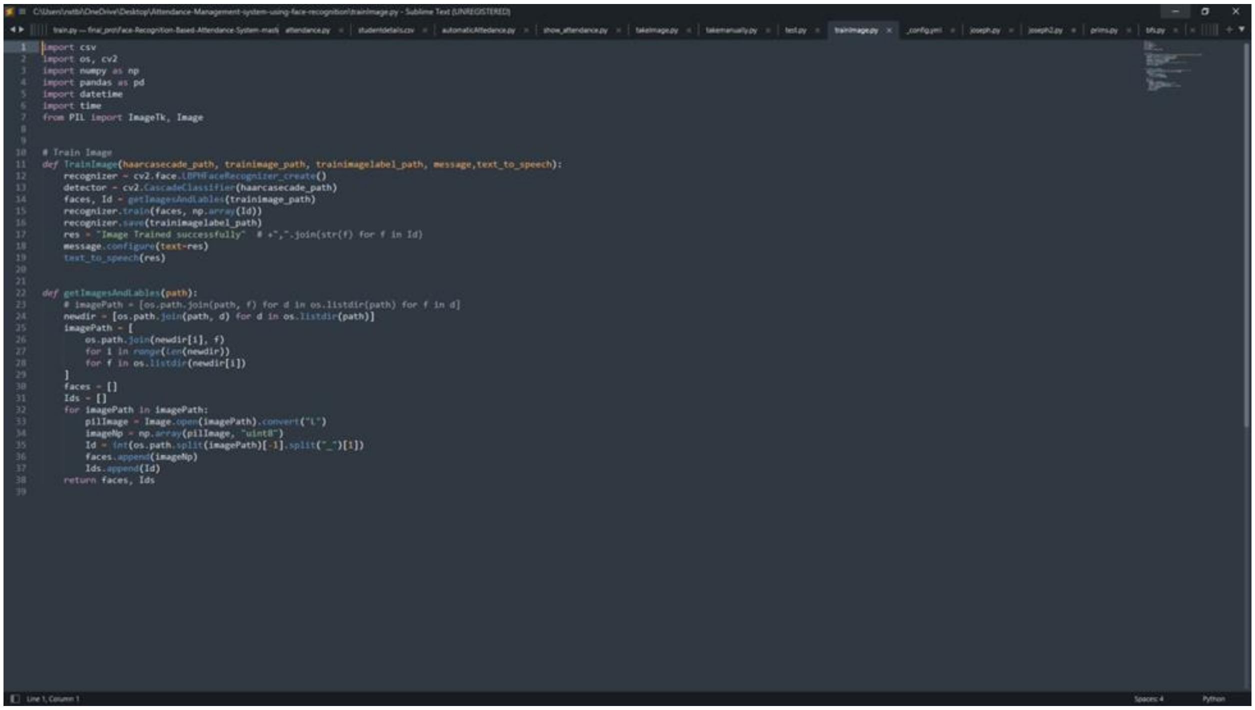
C:\Users\nxtbi\OneDrive\Desktop\Attendance-Management-system-using-face-recognition\takemanually.py - Sublime Text (UNREGISTERED)
train.py — final_proj\Face-Recognition-Based-Attendance-System-master
attendance.py
studentdetails.csv
automaticAttedance.py
show_attendance.py
takeImage.py
takemanually.py
test.py
trainImage.py
_config.yml
joseph.py
joseph2.py
prims.py
bfs.py
Line 302, Column 124
Spaces: 4
Python

```
        DATA_SUB = tk.Button(
            MFW,
            text="Enter Data",
            command=enter_data_DB,
            fg="black",
            bg="lime green",
            width=20,
            height=2,
            activebackground="Red",
            font=("times", 15, " bold "),
        )
        DATA_SUB.place(x=170, y=300)

        MAKE_CSV = tk.Button(
            MFW,
            text="Convert to CSV",
            command=create_csv,
            fg="black",
            bg="red",
            width=20,
            height=2,
            activebackground="Red",
            font=("times", 15, " bold "),
        )
        MAKE_CSV.place(x=570, y=300)
        # TODO remove check sheet
        def attf():
            import subprocess

            subprocess.Popen(
                r'explorer /select,"C:/Users/nxtbi/OneDrive/Desktop/Attendance-Management-system-using-face-recognition/Attendance(Manually)"'
            )

        attf = tk.Button(
            MFW,
            text="Check Sheets",
            command=attf,
            fg="black",
            bg="lawn green",
            width=12,
            height=1,
            activebackground="Red",
            font=("times", 14, " bold "),
        )
        attf.place(x=730, y=410)

        MFW.mainloop()

    SUB = tk.Label(
        sb,
        text="Enter Subject",
        width=15,
        height=2,
        fg="white",
        bg="blue2",
```


C:\Users\nxtbi\OneDrive\Desktop\Attendance-Management-system-using-face-recognition\takemanually.py - Sublime Text (UNREGISTERED)
train.py — final_proj\Face-Recognition-Based-Attendance-System-mast
attendance.py
studentdetails.csv
automaticAttedance.py
show_attendance.py
takeImage.py
takemanually.py
test.py
trainImage.py
_config.yml
joseph.py
joseph2.py
prims.py
bfs.py
Line 302, Column 124
Spaces: 4
Python

```
            def attf():
                import subprocess

                subprocess.Popen(
                    r'explorer /select,"C:/Users/nxtbi/OneDrive/Desktop/Attendance-Management-system-using-face-recognition/Attendance(Manually)"'
                )

            attf = tk.Button(
                MFW,
                text="Check Sheets",
                command=attf,
                fg="black",
                bg="lawn green",
                width=12,
                height=1,
                activebackground="Red",
                font=("times", 14, " bold "),
            )
            attf.place(x=730, y=410)

            MFW.mainloop()

    SUB = tk.Label(
        sb,
        text="Enter Subject",
        width=15,
        height=2,
        fg="white",
        bg="blue2",
        font=("times", 15, " bold "),
    )
    SUB.place(x=30, y=100)

    global SUB_ENTRY

    SUB_ENTRY = tk.Entry(
        sb, width=20, bg="yellow", fg="red", font=("times", 23, " bold ")
    )
    SUB_ENTRY.place(x=250, y=105)

    fill_manual_attendance = tk.Button(
        sb,
        text="Fill Attendance",
        command=fill_attendance,
        fg="white",
        bg="deep pink",
        width=20,
        height=2,
        activebackground="Red",
        font=("times", 15, " bold "),
    )
    fill_manual_attendance.place(x=250, y=160)
    sb.mainloop()
```

## trainImage.py


C:\Users\netb\OneDrive\Desktop\Attendance-Management-system-using-face-recognition\trainImage.py - Sublime Text (UNREGISTERED)
train.py — final_proj\Face-Recognition-Based-Attendance-System-mask
attendance.py
studentdetails.csv
automaticAttedance.py
show_attendance.py
takeImage.py
takemanually.py
test.py
trainImage.py
_config.yml
joseph.py
joseph2.py
prims.py
bfs.py
Line 1, Column 1
Spaces: 4
Python

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


# Train Image
def TrainImage(haarcasecade_path, trainimage_path, trainimagelabel_path, message,text_to_speech):
    recognizer = cv2.face.LBPHFaceRecognizer_create()
    detector = cv2.CascadeClassifier(haarcasecade_path)
    faces, Id = getImagesAndLables(trainimage_path)
    recognizer.train(faces, np.array(Id))
    recognizer.save(trainimagelabel_path)
    res = "Image Trained successfully"  # +",".join(str(f) for f in Id)
    message.configure(text=res)
    text_to_speech(res)


def getImagesAndLables(path):
    # imagePath = [os.path.join(path, f) for d in os.listdir(path) for f in d]
    newdir = [os.path.join(path, d) for d in os.listdir(path)]
    imagePath = [
        os.path.join(newdir[i], f)
        for i in range(len(newdir))
        for f in os.listdir(newdir[i])
    ]
    faces = []
    Ids = []
    for imagePath in imagePath:
        pilImage = Image.open(imagePath).convert("L")
        imageNp = np.array(pilImage, "uint8")
        Id = int(os.path.split(imagePath)[-1].split("_")[1])
        faces.append(imageNp)
        Ids.append(Id)
    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.