# ACKNOWLEDGEMENT

## ACKNOWLEDGEMENT

I would like to express my gratitude towards the department of Information Technology of Shankar Narayan College of Arts & Commerce

I would like to extend my hearty thanks to my internal guide **Prof. Vaibhav Shinde** who’s guidance me to take right decisions with regards to my project. I thank him for sharing their immaculate knowledge and experience, which helped me to achieve my goal.

Their valuable inputs have helped me shape this project better.

A special thanks to Department of Information Technology for providing the lab facilities for completion of the project.

# DECLARATION

**DECLARATION**

Wehereby declare that I myself have completed the project under the guidance of **Prof. Vaibhav Shinde.** I on my own have designed the software and have done all the programming required.

It may require some modifications in the future as per the user’s requirements. From practical implementation, point of view flexibility in the changes have incorporated.

I am sure that I can do any kind of modification suggested while practical implementation by modifying file design or the program code if necessary.

**ARBAZ KHAN & SACHIN SINGH**

# INDEX

|  |  |  |
| --- | --- | --- |
| Sr.No. | Topic | Page No. |
| I. | Introduction | **7** |
| 1. | Objective | **9** |
| 2. | Scope of Study | **10** |
| 3. | Proposed System | **11** |
| II. | Literature Survey | **12** |
| 1. | Tools and Technology | **13** |
| 2. | Technical Specification | **14** |
| III. | System Design | **15** |
| 1. | Block Diagram | **16** |
| 2. | Use-case Diagram | **17** |
| 3. | Process Diagram | **18** |
| IV. | System Coding | **19** |
| 1. | Coding | **20** |
| 2. | Screen Layouts | **66** |
| 3. | Report Layouts | **74** |
| V. | Future Enhancements | **75** |
| VI. | Reference and Bibliography | **77** |

INTRODUCTION

## Attendance using face detection:

Applying machine learning techniques to biometric security solutions is one of the emerging AI trends. Today I would like to share some ideas about how to develop a face recognition-based biometric identification system using OpenCV library, DLib and real-time streaming via video camera. In order for the system to function, it’s necessary to implement three steps. First, it must detect a face. Then, it must recognize that face nearly instantaneously. Finally, it must take whatever further action is required, such as allowing access for an approved user. Face recognition-based attendance system is a process of recognizing the students face for taking attendance by using face biometrics based on high - definition monitor video and other information technology.

**OBJECTIVES:**

The aim is to detect, recognize and mark attendance by face recognition but the project has a lot more objectives:

* **Detection**
* **Recognition**

#### Updating record in Excel & managing student data through excel

##### Detection

*Detection is done by the help of OpenCV and Haar cascades*

*Face detection using Haar cascades is a machine learning based approach where a cascade function is trained with a set of input data. OpenCV already contains many pre-trained classifiers for face, eyes, smiles, etc.. Today we will be using the face classifier. You can experiment with other classifiers as well.*

##### - Recognition

*Recognition is done by LBPH recogniser*

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

*LBPH is one of the easiest face recognition algorithms. It can represent local features in the images. It is possible to get great results (mainly in a controlled environment). It is robust against monotonic gray scale transformations. It is provided by the OpenCV library (Open Source Computer Vision Library).*

##### Manage record in Excel files by GUI

*With the help of GUI, recording the students data on excel.*

### SCOPE OF STUDY:

* + Automating attendance using Face Recognition via Neural Networks.
  + The entire process of marking attendance in educational institutions, workplaces, when automated is the best and most cost-effective way of making it fool-proof and better.
  + This makes proxy attendance impossible and workplace ethics- trustworthy.
  + It creates Class-wise Excel.
  + It is fast as compared to paper work.
  + Data maintained in CSV format with date-wise.
  + Student Details maintains according to Class-wise
  + As the next day arises, it is automatically stored in new tab in the .xlxs sheet so files aren’t over-written.
  + Prevent loss of productivity, saves time, accurate, increases security & automated.

### PROPOSED SYSTEM:

* + Automating attendance using Face Recognition via Neural Networks.
  + The entire process of marking attendance in educational institutions, workplaces, when automised is the best and most cost-effective way of making it fool-proof and better.
  + This makes proxy attendance impossible and workplace ethics- trustworthy.
  + It creates Class-wise Excel.
  + It is fast as compared to paper work.
  + Data maintained in CSV format with date-wise.
  + Student Details maintains according to Class-wise Additional Highlights:
* As the next day arises, it is automatically stored in

new tab in the xlxs sheet so files arent over-written

* Seeing attendance or editing requires an master face print which can be set earlier so students cant change their records.
* Multiple times and mutiple faces is taken in consideration

# LITERATURE SURVEY

### TOOLS AND TECHNOLOGY:

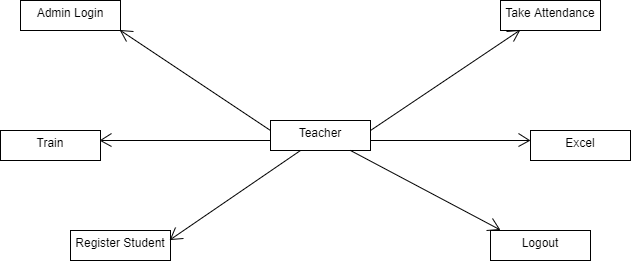
* Python
* Tensorflow
* Keras
* SqlLite3
* Tkinter
* OpenCV

### TECHNICAL SPECIFICATION

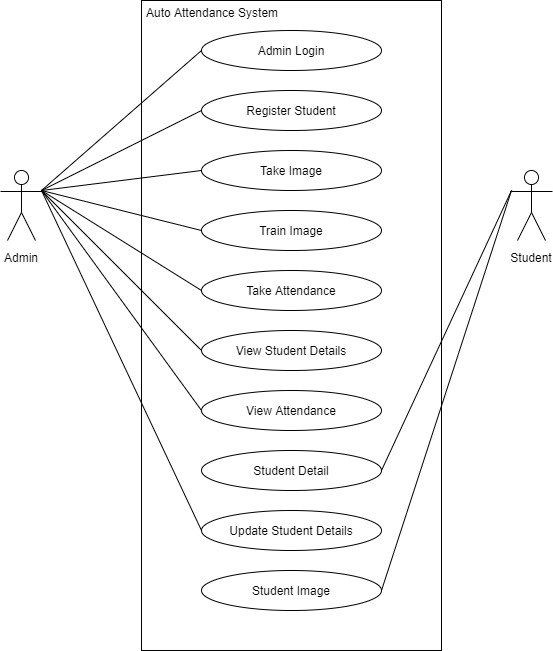
* **OpenCV-python**
* **Pandas**
* **Numpy**
* **csv**
* **Pilow**
* **PIL**
* **smtplib**
* **calender**
* **holidays**
* **datetime**
* **openpyxl**
* **tkinter**
* **xlrd**

# SYSTEM DESIGN

### BLOCK DIAGRAM



**USE-CASE DIAGRAM**



# SYSTEM CODING

### CODING:

###### Splash\_screen.ui:

<?xml version="1.0" encoding="UTF-8"?>

<ui version="4.0">

<class>SplashScreen</class>

<widget class="QMainWindow" name="SplashScreen">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>680</width>

<height>400</height>

</rect>

</property>

<property name="windowTitle">

<string>MainWindow</string>

</property>

<widget class="QWidget" name="centralwidget">

<layout class="QVBoxLayout" name="verticalLayout">

<property name="spacing">

<number>0</number>

</property>

<property name="leftMargin">

<number>10</number>

</property>

<property name="topMargin">

<number>10</number>

</property>

<property name="rightMargin">

<number>10</number>

</property>

<property name="bottomMargin">

<number>10</number>

</property>

<item>

<widget class="QFrame" name="dropShadowFrame">

<property name="font">

<font>

<weight>75</weight>

<bold>true</bold>

</font>

</property>

<property name="styleSheet">

<string notr="true">QFrame { background-color: rgb(0, 0, 0);

color: rgb(57, 255, 20); border-radius: 10px;

}</string>

</property>

<property name="frameShape">

<enum>QFrame::StyledPanel</enum>

</property>

<property name="frameShadow">

<enum>QFrame::Raised</enum>

</property>

<widget class="QLabel" name="label\_title">

<property name="geometry">

<rect>

<x>0</x>

<y>60</y>

<width>661</width>

<height>60</height>

</rect>

</property>

<property name="font">

<font>

<family>Century Gothic</family>

<pointsize>40</pointsize>

<weight>75</weight>

<bold>true</bold>

</font>

</property>

<property name="styleSheet">

<string notr="true">color:rgb(6, 142, 200)</string>

</property>

<property name="text">

<string>&lt;html&gt;&lt;head/&gt;&lt;body&gt;&lt;p&gt;ATTENDANCE&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>

</property>

<property name="alignment">

<set>Qt::AlignCenter</set>

</property>

</widget>

<widget class="QLabel" name="label\_description">

<property name="geometry">

<rect>

<x>0</x>

<y>150</y>

<width>661</width>

<height>31</height>

</rect>

</property>

<property name="font">

<font>

<family>Segoe UI</family>

<pointsize>14</pointsize>

<weight>75</weight>

<bold>true</bold>

</font>

</property>

<property name="styleSheet">

<string notr="true">color: rgb(96, 196, 218);</string>

</property>

<property name="text">

<string>&lt;strong&gt;Your Attendance Matters&lt;/strong&gt;</string>

</property>

<property name="alignment">

<set>Qt::AlignCenter</set>

</property>

</widget>

<widget class="QProgressBar" name="progressBar">

<property name="geometry">

<rect>

<x>50</x>

<y>280</y>

<width>561</width>

<height>23</height>

</rect>

</property>

<property name="styleSheet">

<string notr="true">QProgressBar {

background-color: rgb(98, 114, 164);

color: rgb(200, 200, 200); border-style: none; border-radius: 10px;

text-align: center;

}

QProgressBar::chunk{

border-radius: 10px;

background-color: qlineargradient(spread:pad, x1:0, y1:0.511364, x2:1, y2:0.523, stop:0 rgba(0, 241, 102), stop:1 rgba(170, 85, 255, 255));

}</string>

</property>

<property name="value">

<number>24</number>

</property>

</widget>

<widget class="QLabel" name="label\_loading">

<property name="geometry">

<rect>

<x>0</x>

<y>320</y>

<width>661</width>

<height>21</height>

</rect>

</property>

<property name="font">

<font>

<family>Segoe UI</family>

<pointsize>12</pointsize>

</font>

</property>

<property name="styleSheet">

<string notr="true">color: rgb(98, 114, 164);</string>

</property>

<property name="text">

<string>loading...</string>

</property>

<property name="alignment">

<set>Qt::AlignCenter</set>

</property>

</widget>

<widget class="QLabel" name="label\_credits">

<property name="geometry">

<rect>

<x>20</x>

<y>350</y>

<width>621</width>

<height>21</height>

</rect>

</property>

<property name="font">

<font>

<family>Segoe UI</family>

<pointsize>10</pointsize>

</font>

</property>

<property name="styleSheet">

<string notr="true">color: rgb(98, 114, 164);</string>

</property>

<property name="text">

<string>&lt;strong&gt;Created&lt;/strong&gt;: Wanderson M. Pimenta</string>

</property>

<property name="alignment">

<set>Qt::AlignRight|Qt::AlignTrailing|Qt::AlignVCenter</set>

</property>

</widget>

</widget>

</item>

</layout>

</widget>

</widget>

<resources/>

<connections/>

</ui>

# -\*- coding: utf-8 -\*-

###### Ui\_splash\_screen.py:

############################################################################## ##

## Form generated from reading UI file 'splash\_screenXBSmkq.ui' ##

## Created by: Qt User Interface Compiler version 5.14.1 ##

## WARNING! All changes made in this file will be lost when recompiling UI file!

############################################################################## ##

from PySide2.QtCore import (QCoreApplication, QMetaObject, QObject, QPoint, QRect, QSize, QUrl, Qt)

from PySide2.QtGui import (QBrush, QColor, QConicalGradient, QCursor, QFont, QFontDatabase, QIcon, QLinearGradient, QPalette, QPainter, QPixmap, QRadialGradient)

from PySide2.QtWidgets import \*

class Ui\_SplashScreen(object):

def setupUi(self, SplashScreen):

if SplashScreen.objectName(): SplashScreen.setObjectName(u"SplashScreen")

SplashScreen.resize(680, 400) self.centralwidget = QWidget(SplashScreen)

self.centralwidget.setObjectName(u"centralwidget") self.verticalLayout = QVBoxLayout(self.centralwidget) self.verticalLayout.setSpacing(0) self.verticalLayout.setObjectName(u"verticalLayout") self.verticalLayout.setContentsMargins(10, 10, 10, 10) self.dropShadowFrame = QFrame(self.centralwidget) self.dropShadowFrame.setObjectName(u"dropShadowFrame") self.dropShadowFrame.setStyleSheet(u"QFrame { \n"

" background-color: rgb(0, 0, 0); \n" " color: rgb(220, 220, 220);\n"

" border-radius: 10px;\n" "}")

self.dropShadowFrame.setFrameShape(QFrame.StyledPanel)

self.dropShadowFrame.setFrameShadow(QFrame.Raised) self.label\_title = QLabel(self.dropShadowFrame) self.label\_title.setObjectName(u"label\_title") self.label\_title.setGeometry(QRect(0, 90, 661, 61))

font = QFont() font.setFamily(u"Century Gothic") font.setPointSize(40) self.label\_title.setFont(font)

self.label\_title.setStyleSheet(u"color:rgb(6, 142, 200);") self.label\_title.setAlignment(Qt.AlignCenter) self.label\_description = QLabel(self.dropShadowFrame) self.label\_description.setObjectName(u"label\_description") self.label\_description.setGeometry(QRect(0, 150, 661, 31)) font1 = QFont()

font1.setFamily(u"Century Gothic") font1.setPointSize(14) self.label\_description.setFont(font1)

self.label\_description.setStyleSheet(u"color: rgb(96, 196, 218);") self.label\_description.setAlignment(Qt.AlignCenter) self.progressBar = QProgressBar(self.dropShadowFrame) self.progressBar.setObjectName(u"progressBar") self.progressBar.setGeometry(QRect(50, 280, 561, 23)) self.progressBar.setStyleSheet(u"QProgressBar {\n"

" \n"

" background-color: rgb(0, 0, 0);\n"

" color: rgb(200, 200, 200);\n"

" border-style: none;\n"

" border-radius: 10px;\n"

" text-align: center;\n" "}\n" "QProgressBar::chunk{\n"

" border-radius: 10px;\n"

" background-color: qlineargradient(spread:pad, x1:0, y1:0.511364, x2:1, y2:0.523, stop:0 rgba(0, 241, 102), stop:1 rgba(170, 85, 255, 255));\n"

"}")

self.progressBar.setValue(24)

self.label\_loading = QLabel(self.dropShadowFrame) self.label\_loading.setObjectName(u"label\_loading") self.label\_loading.setGeometry(QRect(0, 320, 661, 21)) font2 = QFont()

font2.setFamily(u"Segoe UI") font2.setPointSize(12) self.label\_loading.setFont(font2)

self.label\_loading.setStyleSheet(u"color: rgb(98, 114, 164);") self.label\_loading.setAlignment(Qt.AlignCenter) self.label\_credits = QLabel(self.dropShadowFrame) self.label\_credits.setObjectName(u"label\_credits") self.label\_credits.setGeometry(QRect(20, 350, 621, 21)) font3 = QFont()

font3.setFamily(u"Segoe UI") font3.setPointSize(10) self.label\_credits.setFont(font3)

self.label\_credits.setStyleSheet(u"color: rgb(98, 114, 164);") self.label\_credits.setAlignment(Qt.AlignRight|Qt.AlignTrailing|Qt.AlignVCenter)

self.verticalLayout.addWidget(self.dropShadowFrame)

SplashScreen.setCentralWidget(self.centralwidget)

self.retranslateUi(SplashScreen)

QMetaObject.connectSlotsByName(SplashScreen) # setupUi

def retranslateUi(self, SplashScreen): SplashScreen.setWindowTitle(QCoreApplication.translate("SplashScreen", u"MainWindow", None))

self.label\_title.setText(QCoreApplication.translate("SplashScreen", u"<strong>ATTENDANCE</strong>", None))

self.label\_description.setText(QCoreApplication.translate("SplashScreen", u"<strong>Your Attendance Matters</strong>", None))

self.label\_loading.setText(QCoreApplication.translate("SplashScreen", u"loading...", None)) self.label\_credits.setText(QCoreApplication.translate("SplashScreen", u"<strong>Created</strong>: Sachin&Arbaz",

None))

# retranslateUi

###### Ui\_main.py:

# -\*- coding: utf-8 -\*-

############################################################################## ##

## Form generated from reading UI file 'mainzhbIGI.ui' ##

## Created by: Qt User Interface Compiler version 5.14.1 ##

## WARNING! All changes made in this file will be lost when recompiling UI file!

############################################################################## ##

from PySide2.QtCore import (QCoreApplication, QMetaObject, QObject, QPoint, QRect, QSize, QUrl, Qt)

from PySide2.QtGui import (QBrush, QColor, QConicalGradient, QCursor, QFont, QFontDatabase, QIcon, QLinearGradient, QPalette, QPainter, QPixmap, QRadialGradient)

from PySide2.QtWidgets import \*

class Ui\_MainWindow(object):

def setupUi(self, MainWindow):

if MainWindow.objectName(): MainWindow.setObjectName(u"MainWindow")

MainWindow.resize(640, 480) self.centralwidget = QWidget(MainWindow)

self.centralwidget.setObjectName(u"centralwidget") self.verticalLayout = QVBoxLayout(self.centralwidget) self.verticalLayout.setObjectName(u"verticalLayout") self.label = QLabel(self.centralwidget) self.label.setObjectName(u"label")

font = QFont() font.setFamily(u"Roboto Thin") font.setPointSize(25) self.label.setFont(font)

self.label.setAlignment(Qt.AlignCenter)

self.verticalLayout.addWidget(self.label)

MainWindow.setCentralWidget(self.centralwidget) self.menubar = QMenuBar(MainWindow) self.menubar.setObjectName(u"menubar") self.menubar.setGeometry(QRect(0, 0, 640, 21)) MainWindow.setMenuBar(self.menubar) self.statusbar = QStatusBar(MainWindow) self.statusbar.setObjectName(u"statusbar") MainWindow.setStatusBar(self.statusbar)

self.retranslateUi(MainWindow)

QMetaObject.connectSlotsByName(MainWindow) # setupUi

def retranslateUi(self, MainWindow): MainWindow.setWindowTitle(QCoreApplication.translate("MainWindow", u"MainWindow", None)) self.label.setText(QCoreApplication.translate("MainWindow", u"MY APP - ADD HERE YOUR WIDGETS", None))

# retranslateUi

# This will import all the widgets

###### Login.py:

# and modules which are available in # tkinter and ttk module

from tkinter import \* import tkinter as tk

from tkinter import font as tkFont from tkinter.ttk import \*

import os

from PIL import Image, ImageTk import os.path

# creates a Tk() object master = Tk()

# sets the geometry of main # root window master.geometry("752x500") master.title("Login")

# photo=PhotoImage(file="C:/Users/sabnam choudhari/Desktop/rps3.png") # l=Label(master,image=photo)

# l.image=photo #just keeping a reference # # l.grid()

bg = PhotoImage(file = "GUI/img2.png")

canvas1 = Canvas( master, width = 400, height = 400)

canvas1.pack(fill = "both", expand = True)

# Display image

canvas1.create\_image( 0, 0, image = bg, anchor = "nw")

# label1 = Label( master, image = bg) # label1.place(x = 0, y = 0)

def startPlaying():

print("play.py") master.destroy() os.system('play.py')

def goToSettings():

os.system('settings.py')

def Login():

lid1=lid.get() lpass1=lpass.get()

if (lid1.isnumeric() == False):

print("isnumeric",lid1.isnumeric())

mess.\_show(title='Input Error', message="Please Enter Valid ID")

elif (lpass1.isnumeric() == False):

mess.\_show(title='Input Error', message="Please Enter Valid Password") elif ((lid1.isnumeric() == True) and (lpass1.isnumeric() == True)) :

master.destroy() os.system('attendance.py')

# def onExit():

# tkMessageBox.showinfo( "Hello Python", "Hello World")

# a button widget which will open a # new window on button click

# photo = PhotoImage(file = "settings\_icon.png") # photo2 = PhotoImage(file = "exiticon.png")

# exiticon

helv36 = tkFont.Font(family='Times New Roman', size=10, weight='bold')

lb0 =tk.Label(master,text="Enter ID :",background="deepskyblue",foreground="navyblue",font="lucida 10 bold",width=9,height=2)

label0\_canvas=canvas1.create\_window(465, 100,

anchor = "nw", window = lb0)

lb1 =tk.Label(master,text="Enter Password :",background="deepskyblue",foreground="navyblue",font="lucida 10 bold",width=15,height=2)

label1\_canvas=canvas1.create\_window(440, 200,

anchor = "nw", window = lb1)

large\_font = ('Verdana',20) lid =StringVar()

e1 =tk.Entry(master,textvariable=lid,width=12,font=large\_font) entry1\_canvas=canvas1.create\_window(401, 150,

anchor = "nw", window = e1)

lpass=StringVar() e2=Entry(master,textvariable=lpass,width=12,font=large\_font) entry2\_canvas=canvas1.create\_window(401, 250,

anchor = "nw", window = e2)

btn = tk.Button(master,

text ="LOGIN",

command = Login,font="lucida 12 bold",bg="black",activebackground='gold',fg="white", height = 2, width =

12)

button1\_canvas = canvas1.create\_window( 430, 320,

anchor = "nw", window = btn)

# btn2 = tk.Button(master,

# text ="",image=photo2,

# command = onExit,font=helv36,bg="black",activebackground='gold',fg='#FFFFFF',relief='groove', height = 40, width = 45)

# bg="darkviolet"

# btn2.place(pady = 10)

# button2\_canvas = canvas1.create\_window( 5, 60, # anchor = "nw",

# window = btn2)

# btn3= tk.Button(master,

# text =" ",image=photo,

# command = goToSettings,font=helv36,bg="black",activebackground='lightslategrey',fg='#FFFFFF', height = 40, width = 40)

# # btn3.setToolTip("Settings")

# btn.pack(pady = 10)

# button3\_canvas = canvas1.create\_window( 575, 30, # anchor = "nw",

# window = btn3)

# mainloop, runs infinitely mainloop()

###### Register.py:

########################## IMPORTING #####################################

import tkinter as tk from tkinter import ttk

from tkinter import messagebox as mess import tkinter.simpledialog as tsd

import cv2,os import csv

import numpy as np from PIL import Image import pandas as pd import datetime import time

################################## FUNCTIONS ##################################

def assure\_path\_exists(path):

dir = os.path.dirname(path) if not os.path.exists(dir):

os.makedirs(dir)

##############################################################################

def tick():

time\_string = time.strftime('%H:%M:%S') clock.config(text=time\_string) clock.after(200,tick)

##############################################################################

def contact():

mess.\_show(title='Contact us', message="Please contact us on : ‘saachu20@gmail.com' ")

############################################################################## #####

def check\_haarcascadefile():

exists = os.path.isfile("haarcascade\_frontalface\_default.xml") if exists:

pass else:

mess.\_show(title='Some file missing', message='Please contact us for help') window.destroy()

##############################################################################

def save\_pass():

assure\_path\_exists("TrainingImageLabel/")

exists1 = os.path.isfile("TrainingImageLabel\psd.txt") if exists1:

tf = open("TrainingImageLabel\psd.txt", "r") key = tf.read()

else:

master.destroy()

new\_pas = tsd.askstring('Old Password not found', 'Please enter a new password below', show='\*') if new\_pas == None:

mess.\_show(title='No Password Entered', message='Password not set!! Please try again') else:

tf = open("TrainingImageLabel\psd.txt", "w") tf.write(new\_pas)

mess.\_show(title='Password Registered', message='New password was registered successfully!!') return

op = (old.get())

newp= (new.get()) nnewp = (nnew.get()) if (op == key):

if(newp == nnewp):

txf = open("TrainingImageLabel\psd.txt", "w") txf.write(newp)

else:

mess.\_show(title='Error', message='Confirm new password again!!!') return

else:

mess.\_show(title='Wrong Password', message='Please enter correct old password.') return

mess.\_show(title='Password Changed', message='Password changed successfully!!') master.destroy()

##############################################################################

def change\_pass():

global master master = tk.Tk()

master.geometry("400x160") master.resizable(False,False) master.title("Change Password") master.configure(background="white")

lbl4 = tk.Label(master,text=' Enter Old Password',bg='white',font=('times', 12, ' bold ')) lbl4.place(x=10,y=10)

global old

old=tk.Entry(master,width=25 ,fg="black",relief='solid',font=('times', 12, ' bold '),show='\*') old.place(x=180,y=10)

lbl5 = tk.Label(master, text=' Enter New Password', bg='white', font=('times', 12, ' bold ')) lbl5.place(x=10, y=45)

global new

new = tk.Entry(master, width=25, fg="black",relief='solid', font=('times', 12, ' bold '),show='\*') new.place(x=180, y=45)

lbl6 = tk.Label(master, text='Confirm New Password', bg='white', font=('times', 12, ' bold ')) lbl6.place(x=10, y=80)

global nnew

nnew = tk.Entry(master, width=25, fg="black", relief='solid',font=('times', 12, ' bold '),show='\*') nnew.place(x=180, y=80)

cancel=tk.Button(master,text="Cancel", command=master.destroy ,fg="black" ,bg="red" ,height=1,width=25 , activebackground = "white" ,font=('times', 10, ' bold '))

cancel.place(x=200, y=120)

save1 = tk.Button(master, text="Save", command=save\_pass, fg="black", bg="#3ece48", height = 1,width=25, activebackground="white", font=('times', 10, ' bold '))

save1.place(x=10, y=120) master.mainloop()

##############################################################################

def psw():

Id = (txt.get()) name = (txt2.get())

if (((Id.isnumeric()) or (' ' in Id)) and ((name.isalpha()) or (' ' in name))): assure\_path\_exists("TrainingImageLabel/")

exists1 = os.path.isfile("TrainingImageLabel\psd.txt") if exists1:

tf = open("TrainingImageLabel\psd.txt", "r") key = tf.read()

else:

new\_pas = tsd.askstring('Old Password not found', 'Please enter a new password below', show='\*') if new\_pas == None:

mess.\_show(title='No Password Entered', message='Password not set!! Please try again') else:

tf = open("TrainingImageLabel\psd.txt", "w") tf.write(new\_pas)

mess.\_show(title='Password Registered', message='New password was registered successfully!!') return

password = tsd.askstring('Password', 'Enter Password', show='\*') if (password == key):

TrainImages()

elif (password == None):

pass else:

mess.\_show(title='Wrong Password', message='You have entered wrong password') ## else:

## if (Id.isnumeric() == False):

## print("isnumeric",Id.isnumeric())

## mess.\_show(title='Input Error', message="Please Enter ID") ## res = "Please Enter Id"

## message.configure(text=res)

## elif (name.isalpha() == False):

## mess.\_show(title='Input Error', message="Please Enter Name") ## res = "Please Enter Name"

## message.configure(text=res)

## elif (name.isalpha() == False):

## mess.\_show(title='Input Error', message="Please Enter Class") ## res = "Please Enter Name"

## message.configure(text=res)

##############################################################################

def clear():

txt.delete(0, 'end')

## res = "1)Take Images >>> 2)Save Profile" ## message1.configure(text=res)

def clear2():

txt2.delete(0, 'end')

## res = "1)Take Images >>> 2)Save Profile" ## message1.configure(text=res)

def clear3():

txt3a.delete(0, 'end')

## res = "1)Take Images >>> 2)Save Profile" ## message1.configure(text=res)

##############################################################################

def Logout():

window.destroy() os.system('login\_new.py')

##############################################################################

def Attendance():

window.destroy() os.system('attendance.py')

##############################################################################

counter=0

def TakeImages():

global counter check\_haarcascadefile()

columns = ['SERIAL NO.', '', 'ID', '', 'NAME','','CLASS','','YEAR','','SECTION','','REGISTRATION DATE','','E-MAIL','','CONTACT']

Id = (txt.get()) name = (txt2.get())

sclass = (vyear.get()) selyear= (vyear3.get()) selsec= (sec.get())

email = (txta.get()) contct = (txt2a.get())

print("====",Id, name,sclass,selyear,selsec,email,contct) assure\_path\_exists("StudentDetails/"+sclass+"/"+selyear+"/"+selsec+"/") assure\_path\_exists("TrainingImage/")

serial = 0

exists = os.path.isfile("StudentDetails/" +sclass+"/"+selyear+"/"+selsec+"/"+"StudentDetails.csv") if exists:

print("====if exists","StudentDetails/"+sclass+"/"+selyear+"/"+selsec+"/"+"StudentDetails.csv")

with open("StudentDetails/"+sclass+"/"+selyear+"/"+selsec+"/"+"StudentDetails.csv", 'r') as csvFile1: print("====if exists csvFile1")

reader1 = csv.reader(csvFile1) for l in reader1:

serial = serial + 1 serial = (serial // 2) csvFile1.close()

else:

print("====else not exists","StudentDetails/"+sclass+"/"+selyear+"/"+selsec+"/"+"StudentDetails.csv") with open("StudentDetails/"+sclass+"/"+selyear+"/"+selsec+"/"+"StudentDetails.csv", 'a+') as csvFile1:

writer = csv.writer(csvFile1) writer.writerow(columns) serial = 1

csvFile1.close()

ts = time.time()

date = datetime.datetime.fromtimestamp(ts).strftime('%d-%m-%Y')

if (((Id.isnumeric()) or (' ' in Id)) and ((name.isalpha()) or (' ' in name))): print("if Loop")

cam = cv2.VideoCapture(0)

harcascadePath = "haarcascade\_frontalface\_default.xml" detector = cv2.CascadeClassifier(harcascadePath) sampleNum = 0

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)

# incrementing sample number sampleNum = sampleNum + 1

# saving the captured face in the dataset folder TrainingImage cv2.imwrite("TrainingImage\ " + name + "." + str(serial) + "." + Id + '.' + str(sampleNum) + ".jpg",

gray[y:y + h, x:x + w]) # display the frame

cv2.imshow('Taking Images', img) # wait for 100 miliseconds

if cv2.waitKey(10) & 0xFF == ord('q'):

break

# break if the sample number is morethan 100 elif sampleNum > 10:

break cam.release() cv2.destroyAllWindows()

res = "Images Taken for ID : " + Id

## columns = ['SERIAL NO.', '', 'ID', '', 'NAME','','CLASS','','YEAR','','SECTION','','DATE OF REGISTRATION','','E- MAIL','','CONTACT']

row = [serial, '', Id, '', name,'',sclass,'',selyear,'',selsec,'',date,'',email,'',contct] counter=0

with open("StudentDetails/"+sclass+"/"+selyear+"/"+selsec+"/"+"StudentDetails.csv", 'a+') as csvFile: writer = csv.writer(csvFile)

writer.writerow(row) counter=counter+1 print("counter",counter)

csvFile.close() message1.configure(text=res)

## else:

## print("Else Loop")

## if (Id.isnumeric() == False):

## print("isnumeric",Id.isnumeric())

## mess.\_show(title='Input Error', message="Please Enter ID") ## res = "Please Enter Id"

## message.configure(text=res)

## elif (name.isalpha() == False):

## mess.\_show(title='Input Error', message="Please Enter Name") ## res = "Please Enter Name"

## message.configure(text=res)

## elif (sclass.isalpha() == False):

## mess.\_show(title='Input Error', message="Please Enter Class") ## res = "Please Enter Name"

## message.configure(text=res)

##############################################################################

def TrainImages():

global counter

print("counter ",counter)

if(counter == 0):

res="No Trained Image Of Student." message1.configure(text=res)

else:

check\_haarcascadefile() assure\_path\_exists("TrainingImageLabel/")

recognizer = cv2.face\_LBPHFaceRecognizer.create() harcascadePath = "haarcascade\_frontalface\_default.xml" detector = cv2.CascadeClassifier(harcascadePath)

faces, ID = getImagesAndLabels("TrainingImage") try:

recognizer.train(faces, np.array(ID)) except:

mess.\_show(title='No Registrations', message='Please Register someone first!!!') return

recognizer.save("TrainingImageLabel\Trainner.yml") counter=0

res = "Profile Saved Successfully" message1.configure(text=res)

message.configure(text='Total Registrations till now : ' + str(ID[0]))

#############################################################################

def getImagesAndLabels(path):

# get the path of all the files in the folder

imagePaths = [os.path.join(path, f) for f in os.listdir(path)] # create empth face list

faces = []

# create empty ID list Ids = []

# now looping through all the image paths and loading the Ids and the images for imagePath in imagePaths:

# loading the image and converting it to gray scale pilImage = Image.open(imagePath).convert('L')

# Now we are converting the PIL image into numpy array imageNp = np.array(pilImage, 'uint8')

# getting the Id from the image

ID = int(os.path.split(imagePath)[-1].split(".")[1])

# extract the face from the training image sample faces.append(imageNp)

Ids.append(ID) return faces, Ids

##############################################################################

def TrackImages():

check\_haarcascadefile() Id = (txt.get())

name = (txt2.get()) sclass = (vyear.get()) selyear= (vyear3.get()) selsec= (sec.get()) email = (txta.get()) contct = (txt2a.get()) ts = time.time()

date = datetime.datetime.fromtimestamp(ts).strftime('%d-%m-%Y') print("====",Id, name,sclass,selyear,selsec,email,contct)

assure\_path\_exists("Attendance/"+sclass+"/"+selyear+"/"+selsec+"/"+date+"/"+"Attendance\_" + date + ".csv") assure\_path\_exists("StudentDetails/"+sclass+"/"+selyear+"/"+selsec+"/")

for k in tv.get\_children():

tv.delete(k) msg = ''

i = 0

j = 0

recognizer = cv2.face.LBPHFaceRecognizer\_create() # cv2.createLBPHFaceRecognizer() exists3 = os.path.isfile("TrainingImageLabel\Trainner.yml")

if exists3:

recognizer.read("TrainingImageLabel\Trainner.yml") else:

mess.\_show(title='Data Missing', message='Please click on Save Profile to reset data!!') return

harcascadePath = "haarcascade\_frontalface\_default.xml" faceCascade = cv2.CascadeClassifier(harcascadePath);

cam = cv2.VideoCapture(0)

font = cv2.FONT\_HERSHEY\_SIMPLEX

col\_names = ['Id', '', 'Name', '', 'Date', '', 'Time','','Class']

exists1 = os.path.isfile("StudentDetails/"+sclass+"/"+selyear+"/"+selsec+"/"+"StudentDetails.csv") if exists1:

df = pd.read\_csv("StudentDetails/"+sclass+"/"+selyear+"/"+selsec+"/"+"StudentDetails.csv") else:

mess.\_show(title='Details Missing', message='Students details are missing, please check!') cam.release()

cv2.destroyAllWindows() window.destroy()

while True:

ret, im = cam.read()

gray = cv2.cvtColor(im, cv2.COLOR\_BGR2GRAY) faces = faceCascade.detectMultiScale(gray, 1.2, 5) for (x, y, w, h) in faces:

cv2.rectangle(im, (x, y), (x + w, y + h), (225, 0, 0), 2) serial, conf = recognizer.predict(gray[y:y + h, x:x + w]) if (conf < 50):

ts = time.time()

date = datetime.datetime.fromtimestamp(ts).strftime('%d-%m-%Y') timeStamp = datetime.datetime.fromtimestamp(ts).strftime('%H:%M:%S') aa = df.loc[df['SERIAL NO.'] == serial]['NAME'].values

ID = df.loc[df['SERIAL NO.'] == serial]['ID'].values

CLASS = df.loc[df['SERIAL NO.'] == serial]['CLASS'].values ID = str(ID)

ID = ID[1:-1]

bb = str(aa) bb = bb[2:-2] cc=str(CLASS) cc = cc[2:-2] print(cc)

attendance = [str(ID), '', bb, '', str(date), '', str(timeStamp),'',str(cc)] else:

Id = 'Unknown' bb = str(Id)

cv2.putText(im, str(bb), (x, y + h), font, 1, (255, 255, 255), 2) cv2.imshow('Taking Attendance', im)

if (cv2.waitKey(1) == ord('q')):

break

ts = time.time()

date = datetime.datetime.fromtimestamp(ts).strftime('%d-%m-%Y')

exists = os.path.isfile("Attendance/"+sclass+"/"+selyear+"/"+selsec+"/"+date+"/"+"Attendance\_" + date + ".csv") if exists:

with open("Attendance/"+sclass+"/"+selyear+"/"+selsec+"/"+date+"/"+"Attendance\_" + date + ".csv", 'a+') as csvFile1:

writer = csv.writer(csvFile1) writer.writerow(attendance)

csvFile1.close() else:

with open("Attendance/"+sclass+"/"+selyear+"/"+selsec+"/"+date+"/"+"Attendance\_" + date + ".csv", 'a+') as csvFile1:

writer = csv.writer(csvFile1) writer.writerow(col\_names) writer.writerow(attendance)

csvFile1.close()

with open("Attendance/"+sclass+"/"+selyear+"/"+selsec+"/"+date+"/"+"Attendance\_" + date + ".csv", 'r') as csvFile1: reader1 = csv.reader(csvFile1)

for lines in reader1:

i = i + 1 if (i > 1):

if (i % 2 != 0):

iidd = str(lines[0]) + ' '

tv.insert('', 0, text=iidd, values=(str(lines[2]), str(lines[4]), str(lines[6]))) csvFile1.close()

cam.release() cv2.destroyAllWindows()

############################# USED STUFFS #######################################

global key key = ''

ts = time.time()

date = datetime.datetime.fromtimestamp(ts).strftime('%d-%m-%Y') day,month,year=date.split("-")

month={'01':'Jan',

'02':'Feb',

'03':'Mar',

'04':'Apr',

'05':'May',

'06':'Jun',

'07':'Jul',

'08':'Aug',

'09':'Sept',

'10':'Oct',

'11':'Nov',

'12':'Dec'

}

########################## GUI FRONT-END ########################################

window = tk.Tk() window.geometry("800x700") window.resizable(True,False) window.title("Attendance System") window.configure(background='lavender')

message3 = tk.Label(window, text="Face Recognition Based Attendance System" ,fg="white",bg="#262523"

,width=33 ,height=1,font=('times', 29, ' bold ')) message3.place(x=20, y=10)

frame2 = tk.Frame(window, bg="SlateGray3") frame2.place(relx=0.11, rely=0.17, relwidth=0.70, relheight=0.80)

frame3 = tk.Frame(window, bg="#c4c6ce") frame3.place(relx=0.46, rely=0.10, relwidth=0.25, relheight=0.05)

frame4 = tk.Frame(window, bg="#c4c6ce") frame4.place(relx=0.26, rely=0.10, relwidth=0.25, relheight=0.05)

datef = tk.Label(frame4, text = day+"-"+mont[month]+"-"+year+" | ", fg="orange",bg="#262523" ,width=55

,height=1,font=('times', 20, ' bold ')) datef.pack(fill='both',expand=1)

clock = tk.Label(frame3,fg="orange",bg="#262523" ,width=55 ,height=1,font=('times', 20, ' bold ')) clock.pack(fill='both',expand=1)

tick()

head2 = tk.Label(frame2, text=" NEW REGISTRATION", fg="black",bg="SlateGray3" ,font=('times', 17, ' bold ') )

head2.grid(row=0,column=0)

lbl2 = tk.Label(frame2, text="Name : ",width=10 ,fg="black" ,bg="SlateGray3" ,font=('times', 14, ' bold ')) lbl2.place(x=0, y=60)

txt2 = tk.Entry(frame2,width=25 ,fg="black",font=('times', 14, ' bold ') ) txt2.place(x=130, y=60)

lbl = tk.Label(frame2, text="Roll No :",width=10 ,height=1 ,fg="black" ,bg="SlateGray3" ,font=('times', 14, ' bold ') ) lbl.place(x=0, y=100)

txt = tk.Entry(frame2,width=25 ,fg="black",font=('times', 15, ' bold ')) txt.place(x=130, y=100)

lbl3a = tk.Label(frame2, text="Course : ",width=10 ,fg="black" ,bg="SlateGray3" ,font=('times', 14, ' bold ')) lbl3a.place(x=0, y=140)

my\_list = ["MSC-IT", "M-COM","MA"]

vyear = tk.StringVar() vyear.set(my\_list[0]) # default value

txt3a = tk.OptionMenu(frame2,vyear,\*my\_list) txt3a.config(width=20)

txt3a.place(x=130, y=140)

lbl3aa = tk.Label(frame2, text="Year : ",width=10 ,fg="black" ,bg="SlateGray3" ,font=('times', 14, ' bold ')) lbl3aa.place(x=0, y=180)

my\_list3 = ["FY", "SY"] vyear3 = tk.StringVar()

vyear3.set(my\_list3[0]) # default value

txt3aa = tk.OptionMenu(frame2,vyear3,\*my\_list3) txt3aa.config(width=20)

txt3aa.place(x=130, y=180)

lbl3b = tk.Label(frame2, text="Section : ",width=10 ,fg="black" ,bg="SlateGray3" ,font=('times', 14, ' bold ')) lbl3b.place(x=0, y=220)

my\_list2 = ["1", "2"] sec = tk.StringVar()

sec.set(my\_list2[0]) # default value

txt3b = tk.OptionMenu(frame2,sec,\*my\_list2) txt3b.config(width=10)

txt3b.place(x=130, y=220)

lbl2a = tk.Label(frame2, text="Contact :",width=10 ,fg="black" ,bg="SlateGray3" ,font=('times', 14, ' bold ')) lbl2a.place(x=0, y=260)

txt2a = tk.Entry(frame2,width=25 ,fg="black",font=('times', 14, ' bold ') ) txt2a.place(x=130, y=260)

lbla = tk.Label(frame2, text="Email :",width=10 ,height=1 ,fg="black" ,bg="SlateGray3" ,font=('times', 14, ' bold ') ) lbla.place(x=0, y=300)

txta = tk.Entry(frame2,width=25 ,fg="black",font=('times', 15, ' bold ')) txta.place(x=130, y=300)

message1 = tk.Label(frame2, text="Step 1) Take Images\nStep 2) Save Profile" ,bg="rosy brown" ,fg="black"

,width=29 ,height=2, activebackground = "yellow" ,font=('times', 15, ' bold ')) message1.place(x=100, y=340)

message = tk.Label(frame2, text="" ,fg="black" ,width=29,height=1, activebackground = "yellow" ,font=('times', 16, ' bold '))

message.place(x=100, y=500)

Id = (txt.get()) name = (txt2.get())

sclass = (vyear.get()) selyear= (vyear3.get()) selsec= (sec.get()) email = (txta.get()) contct = (txt2a.get())

res=0

exists = os.path.isfile("StudentDetails/"+sclass+"/"+selyear+"/"+selsec+"/"+"StudentDetails.csv") if exists:

with open("StudentDetails/"+sclass+"/"+selyear+"/"+selsec+"/"+"StudentDetails.csv", 'r') as csvFile1: reader1 = csv.reader(csvFile1)

for l in reader1:

res = res + 1 res = (res // 2) - 1 csvFile1.close()

else:

res = 0

message.configure(text='Total Registrations till now : '+str(res))

##################### EXCEL VIEW #################################

def open\_student\_excel():

sclass = (vyear.get()) selyear= (vyear3.get()) selsec= (sec.get())

file="C:/Users/square/Desktop/Projects2/Attendance\_Systems/ /StudentDetails/"+sclass+"/"+selyear+"/"+selsec+"/"+"StudentDetails.csv"

print("StudentDetails/"+sclass+"/"+selyear+"/"+selsec+"/"+"StudentDetails.csv") os.startfile(file)

def open\_attendance\_excel():

ts = time.time() Id = (txt.get())

name = (txt2.get()) sclass = (vyear.get()) selyear= (vyear3.get()) selsec= (sec.get()) email = (txta.get()) contct = (txt2a.get())

date = datetime.datetime.fromtimestamp(ts).strftime('%d-%m-%Y')

exists = os.path.isfile("C:/Users/square/Desktop/Projects2/Attendance\_Systems/Attendance/"+sclass+"/"+selyear+"/"+selsec+"/"+date+"/"+"Attendance\_" + date + ".csv")

if exists:

file = " C:/Users/square/Desktop/Projects2/Attendance\_Systems/Attendance/"+sclass+"/"+selyear+"/"+selsec+"/"+date+"/"+"Attendance\_" + date + ".csv"

os.startfile(file)

else:

mess.\_show(title='No Record Found.', message="Attendance has not been taken yet.")

##################### MENUBAR #################################

menubar = tk.Menu(window,relief='ridge') filemenu = tk.Menu(menubar,tearoff=0)

filemenu.add\_command(label='Students Detail Excel', command = open\_student\_excel) filemenu.add\_command(label='Attendance Excel', command = open\_attendance\_excel) menubar.add\_cascade(label='View',font=('times', 29, ' bold '),menu=filemenu)

filemenu2 = tk.Menu(menubar,tearoff=0) filemenu2.add\_command(label='Take Attendance', command = Attendance)

menubar.add\_cascade(label='Attendance',font=('times', 29, ' bold '),menu=filemenu2)

filemenu1 = tk.Menu(menubar,tearoff=0) filemenu1.add\_command(label='Change Password', command = change\_pass) filemenu1.add\_command(label='Contact Us', command = contact) filemenu1.add\_command(label='Exit',command = window.destroy) menubar.add\_cascade(label='Help',font=('times', 29, ' bold '),menu=filemenu1)

###################### BUTTONS ##################################

##clearButton = tk.Button(frame2, text="Clear", command=clear ,fg="white" ,bg="black" ,width=11

,activebackground = "white" ,font=('times', 11, ' bold ')) ##clearButton.place(x=335, y=88)

##clearButton2 = tk.Button(frame2, text="Clear", command=clear2 ,fg="white" ,bg="black" ,width=11 , activebackground = "white" ,font=('times', 11, ' bold '))

##clearButton2.place(x=335, y=154)

##clearButton3 = tk.Button(frame2, text="Clear", command=clear3 ,fg="white" ,bg="black" ,width=11 , activebackground = "white" ,font=('times', 11, ' bold '))

##clearButton3.place(x=335, y=220)

takeImg = tk.Button(frame2, text="Take Images", command=TakeImages ,fg="white" ,bg="cadet blue" ,width=29

,height=1, activebackground = "white" ,font=('times', 15, ' bold ')) takeImg.place(x=100, y=400)

trainImg = tk.Button(frame2, text="Save Profile", command=psw ,fg="white" ,bg="cadet blue" ,width=29 ,height=1, activebackground = "white" ,font=('times', 15, ' bold '))

trainImg.place(x=100, y=450)

##trackImg = tk.Button(frame1, text="Take Attendance", command=TrackImages ,fg="white" ,bg="OliveDrab4"

,width=35 ,height=1, activebackground = "white" ,font=('times', 15, ' bold '))

##trackImg.place(x=30,y=50)

logoutWindow = tk.Button(window, text="Logout", command=Logout ,fg="white" ,bg="red3" ,width=6 ,height=1, activebackground = "white" ,font=('times', 15, ' bold '))

logoutWindow.place(x=650, y=70)

##################### END ######################################

window.configure(menu=menubar) window.mainloop()

##############################################################################

###### Attendance.py:

############################### IMPORTING ###################################

import tkinter as tk from tkinter import ttk

from tkinter import messagebox as mess import tkinter.simpledialog as tsd

import cv2,os import csv

import numpy as np from PIL import Image import pandas as pd import datetime import time

############################### FUNCTIONS ####################################

def assure\_path\_exists(path):

dir = os.path.dirname(path) if not os.path.exists(dir):

os.makedirs(dir)

##############################################################################

def tick():

time\_string = time.strftime('%H:%M:%S') clock.config(text=time\_string) clock.after(200,tick)

##############################################################################

def contact():

mess.\_show(title='Contact us', message="Please contact us on : ‘saachu20@gmail.com’")

##############################################################################

def check\_haarcascadefile():

exists = os.path.isfile("haarcascade\_frontalface\_default.xml") if exists:

pass else:

mess.\_show(title='Some file missing', message='Please contact us for help') window.destroy()

##############################################################################

def save\_pass():

assure\_path\_exists("TrainingImageLabel/")

exists1 = os.path.isfile("TrainingImageLabel\psd.txt") if exists1:

tf = open("TrainingImageLabel\psd.txt", "r") key = tf.read()

else:

master.destroy()

new\_pas = tsd.askstring('Old Password not found', 'Please enter a new password below', show='\*') if new\_pas == None:

mess.\_show(title='No Password Entered', message='Password not set!! Please try again') else:

tf = open("TrainingImageLabel\psd.txt", "w") tf.write(new\_pas)

mess.\_show(title='Password Registered', message='New password was registered successfully!!') return

op = (old.get())

newp= (new.get()) nnewp = (nnew.get()) if (op == key):

if(newp == nnewp):

txf = open("TrainingImageLabel\psd.txt", "w") txf.write(newp)

else:

mess.\_show(title='Error', message='Confirm new password again!!!') return

else:

mess.\_show(title='Wrong Password', message='Please enter correct old password.') return

mess.\_show(title='Password Changed', message='Password changed successfully!!') master.destroy()

##############################################################################

def change\_pass(): global master master = tk.Tk()

master.geometry("400x160") master.resizable(False,False) master.title("Change Password") master.configure(background="white")

lbl4 = tk.Label(master,text=' Enter Old Password',bg='white',font=('times', 12, ' bold ')) lbl4.place(x=10,y=10)

global old

old=tk.Entry(master,width=25 ,fg="black",relief='solid',font=('times', 12, ' bold '),show='\*') old.place(x=180,y=10)

lbl5 = tk.Label(master, text=' Enter New Password', bg='white', font=('times', 12, ' bold ')) lbl5.place(x=10, y=45)

global new

new = tk.Entry(master, width=25, fg="black",relief='solid', font=('times', 12, ' bold '),show='\*') new.place(x=180, y=45)

lbl6 = tk.Label(master, text='Confirm New Password', bg='white', font=('times', 12, ' bold ')) lbl6.place(x=10, y=80)

global nnew

nnew = tk.Entry(master, width=25, fg="black", relief='solid',font=('times', 12, ' bold '),show='\*') nnew.place(x=180, y=80)

cancel=tk.Button(master,text="Cancel", command=master.destroy ,fg="black" ,bg="red" ,height=1,width=25 , activebackground = "white" ,font=('times', 10, ' bold '))

cancel.place(x=200, y=120)

save1 = tk.Button(master, text="Save", command=save\_pass, fg="black", bg="#3ece48", height = 1,width=25, activebackground="white", font=('times', 10, ' bold '))

save1.place(x=10, y=120) master.mainloop()

##############################################################################

def psw():

Id = (txt.get()) name = (txt2.get())

if (((Id.isnumeric()) or (' ' in Id)) and ((name.isalpha()) or (' ' in name))): assure\_path\_exists("TrainingImageLabel/")

exists1 = os.path.isfile("TrainingImageLabel\psd.txt") if exists1:

tf = open("TrainingImageLabel\psd.txt", "r") key = tf.read()

else:

new\_pas = tsd.askstring('Old Password not found', 'Please enter a new password below', show='\*')

if new\_pas == None:

mess.\_show(title='No Password Entered', message='Password not set!! Please try again') else:

tf = open("TrainingImageLabel\psd.txt", "w") tf.write(new\_pas)

mess.\_show(title='Password Registered', message='New password was registered successfully!!') return

password = tsd.askstring('Password', 'Enter Password', show='\*') if (password == key):

TrainImages()

elif (password == None):

pass else:

mess.\_show(title='Wrong Password', message='You have entered wrong password')

##############################################################################

def clear():

txt.delete(0, 'end')

## res = "1)Take Images >>> 2)Save Profile" ## message1.configure(text=res)

def clear2():

txt2.delete(0, 'end')

## res = "1)Take Images >>> 2)Save Profile" ## message1.configure(text=res)

def clear3():

txt3a.delete(0, 'end')

## res = "1)Take Images >>> 2)Save Profile" ## message1.configure(text=res)

##############################################################################

def Register(): window.destroy() os.system('register.py')

##############################################################################

counter=0

def TakeImages():

global counter check\_haarcascadefile()

columns = ['SERIAL NO.', '', 'ID', '', 'NAME','','CLASS','','YEAR','','SECTION','','REGISTRATION DATE','','E-MAIL','','CONTACT']

sclass = (vyear.get()) selyear= (vyear3.get()) selsec= (sec.get())

print("====",Id, name,sclass,selyear,selsec,email,contct) assure\_path\_exists("StudentDetails/"+sclass+"/"+selyear+"/"+selsec+"/") assure\_path\_exists("TrainingImage/")

serial = 0

exists = os.path.isfile("StudentDetails/" +sclass+"/"+selyear+"/"+selsec+"/"+"StudentDetails.csv") if exists:

print("====if exists","StudentDetails/"+sclass+"/"+selyear+"/"+selsec+"/"+"StudentDetails.csv")

with open("StudentDetails/"+sclass+"/"+selyear+"/"+selsec+"/"+"StudentDetails.csv", 'r') as csvFile1: print("====if exists csvFile1")

reader1 = csv.reader(csvFile1) for l in reader1:

serial = serial + 1 serial = (serial // 2) csvFile1.close()

else:

print("====else not exists","StudentDetails/"+sclass+"/"+selyear+"/"+selsec+"/"+"StudentDetails.csv") with open("StudentDetails/"+sclass+"/"+selyear+"/"+selsec+"/"+"StudentDetails.csv", 'a+') as csvFile1:

writer = csv.writer(csvFile1) writer.writerow(columns) serial = 1

csvFile1.close()

ts = time.time()

date = datetime.datetime.fromtimestamp(ts).strftime('%d-%m-%Y')

if (((Id.isnumeric()) or (' ' in Id)) and ((name.isalpha()) or (' ' in name))): print("if Loop")

cam = cv2.VideoCapture(0)

harcascadePath = "haarcascade\_frontalface\_default.xml" detector = cv2.CascadeClassifier(harcascadePath)

sampleNum = 0 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) # incrementing sample number

sampleNum = sampleNum + 1

# saving the captured face in the dataset folder TrainingImage cv2.imwrite("TrainingImage\ " + name + "." + str(serial) + "." + Id + '.' + str(sampleNum) + ".jpg",

gray[y:y + h, x:x + w]) # display the frame

cv2.imshow('Taking Images', img) # wait for 100 miliseconds

if cv2.waitKey(10) & 0xFF == ord('q'):

break

# break if the sample number is morethan 100 elif sampleNum > 10:

break cam.release() cv2.destroyAllWindows()

res = "Images Taken for ID : " + Id

## columns = ['SERIAL NO.', '', 'ID', '', 'NAME','','CLASS','','YEAR','','SECTION','','DATE OF REGISTRATION','','E- MAIL','','CONTACT']

row = [serial, '', Id, '', name,'',sclass,'',selyear,'',selsec,'',date,'',email,'',contct] counter=0

with open("StudentDetails/"+sclass+"/"+selyear+"/"+selsec+"/"+"StudentDetails.csv", 'a+') as csvFile: writer = csv.writer(csvFile)

writer.writerow(row) counter=counter+1 print("counter",counter)

csvFile.close() message1.configure(text=res)

##############################################################################

def TrainImages():

global counter

print("counter ",counter)

if(counter == 0):

res="No Trained Image Of Student." message1.configure(text=res)

else:

check\_haarcascadefile() assure\_path\_exists("TrainingImageLabel/")

recognizer = cv2.face\_LBPHFaceRecognizer.create() harcascadePath = "haarcascade\_frontalface\_default.xml" detector = cv2.CascadeClassifier(harcascadePath)

faces, ID = getImagesAndLabels("TrainingImage") try:

recognizer.train(faces, np.array(ID)) except:

mess.\_show(title='No Registrations', message='Please Register someone first!!!') return

recognizer.save("TrainingImageLabel\Trainner.yml") counter=0

res = "Profile Saved Successfully" message1.configure(text=res)

message.configure(text='Total Registrations till now : ' + str(ID[0]))

#############################################################################

def getImagesAndLabels(path):

# get the path of all the files in the folder

imagePaths = [os.path.join(path, f) for f in os.listdir(path)] # create empth face list

faces = []

# create empty ID list Ids = []

# now looping through all the image paths and loading the Ids and the images for imagePath in imagePaths:

# loading the image and converting it to gray scale pilImage = Image.open(imagePath).convert('L')

# Now we are converting the PIL image into numpy array imageNp = np.array(pilImage, 'uint8')

# getting the Id from the image

ID = int(os.path.split(imagePath)[-1].split(".")[1])

# extract the face from the training image sample

faces.append(imageNp) Ids.append(ID)

return faces, Ids

#############################################################################

def TrackImages():

check\_haarcascadefile() sclass = (vyear.get()) selyear= (vyear3.get()) selsec= (sec.get())

ts = time.time()

date = datetime.datetime.fromtimestamp(ts).strftime('%d-%m-%Y') assure\_path\_exists("Attendance/"+sclass+"/"+selyear+"/"+selsec+"/"+date+"/"+"Attendance\_" + date + ".csv") assure\_path\_exists("StudentDetails/"+sclass+"/"+selyear+"/"+selsec+"/")

for k in tv.get\_children():

tv.delete(k) msg = ''

i = 0

j = 0

recognizer = cv2.face.LBPHFaceRecognizer\_create() # cv2.createLBPHFaceRecognizer() exists3 = os.path.isfile("TrainingImageLabel\Trainner.yml")

if exists3: recognizer.read("TrainingImageLabel\Trainner.yml")

else:

mess.\_show(title='Data Missing', message='Please click on Save Profile to reset data!!') return

harcascadePath = "haarcascade\_frontalface\_default.xml" faceCascade = cv2.CascadeClassifier(harcascadePath);

cam = cv2.VideoCapture(0)

font = cv2.FONT\_HERSHEY\_SIMPLEX

col\_names = ['Id', '', 'Name', '', 'Date', '', 'Time','','Class']

exists1 = os.path.isfile("StudentDetails/"+sclass+"/"+selyear+"/"+selsec+"/"+"StudentDetails.csv") if exists1:

df = pd.read\_csv("StudentDetails/"+sclass+"/"+selyear+"/"+selsec+"/"+"StudentDetails.csv") else:

mess.\_show(title='Details Missing', message='Students details are missing, please check!') cam.release()

cv2.destroyAllWindows() ## window.destroy()

while True:

ret, im = cam.read()

gray = cv2.cvtColor(im, cv2.COLOR\_BGR2GRAY) faces = faceCascade.detectMultiScale(gray, 1.2, 5) for (x, y, w, h) in faces:

cv2.rectangle(im, (x, y), (x + w, y + h), (225, 0, 0), 2) serial, conf = recognizer.predict(gray[y:y + h, x:x + w]) if (conf < 50):

ts = time.time()

date = datetime.datetime.fromtimestamp(ts).strftime('%d-%m-%Y') timeStamp = datetime.datetime.fromtimestamp(ts).strftime('%H:%M:%S') aa = df.loc[df['SERIAL NO.'] == serial]['NAME'].values

ID = df.loc[df['SERIAL NO.'] == serial]['ID'].values

CLASS = df.loc[df['SERIAL NO.'] == serial]['CLASS'].values ID = str(ID)

ID = ID[1:-1]

bb = str(aa) bb = bb[2:-2] cc=str(CLASS) cc = cc[2:-2]

attendance = [str(ID), '', bb, '', str(date), '', str(timeStamp),'',str(cc)] else:

Id = 'Unknown' bb = str(Id)

cv2.putText(im, str(bb), (x, y + h), font, 1, (255, 255, 255), 2) cv2.imshow('Taking Attendance', im)

if (cv2.waitKey(1) == ord('q')):

break

ts = time.time()

date = datetime.datetime.fromtimestamp(ts).strftime('%d-%m-%Y')

exists = os.path.isfile("Attendance/"+sclass+"/"+selyear+"/"+selsec+"/"+date+"/"+"Attendance\_" + date + ".csv") if exists:

with open("Attendance/"+sclass+"/"+selyear+"/"+selsec+"/"+date+"/"+"Attendance\_" + date + ".csv", 'a+') as csvFile1:

writer = csv.writer(csvFile1) writer.writerow(attendance)

csvFile1.close() else:

with open("Attendance/"+sclass+"/"+selyear+"/"+selsec+"/"+date+"/"+"Attendance\_" + date + ".csv", 'a+') as csvFile1:

writer = csv.writer(csvFile1) writer.writerow(col\_names) writer.writerow(attendance)

csvFile1.close()

with open("Attendance/"+sclass+"/"+selyear+"/"+selsec+"/"+date+"/"+"Attendance\_" + date + ".csv", 'r') as csvFile1: reader1 = csv.reader(csvFile1)

for lines in reader1:

i = i + 1 if (i > 1):

if (i % 2 != 0):

iidd = str(lines[0]) + ' '

tv.insert('', 0, text=iidd, values=(str(lines[2]), str(lines[4]), str(lines[6]))) csvFile1.close()

cam.release() cv2.destroyAllWindows()

################################ USED STUFFS ###################################

global key key = ''

ts = time.time()

date = datetime.datetime.fromtimestamp(ts).strftime('%d-%m-%Y') day,month,year=date.split("-")

mont={'01':'Jan',

'02':'Feb',

'03':'Mar',

'04':'Apr',

'05':'May',

'06':'Jun',

'07':'Jul',

'08':'Aug',

'09':'Sept',

'10':'Oct',

'11':'Nov',

'12':'Dec'

}

################################# GUI FRONT-END #################################

window = tk.Tk() window.geometry("800x700") window.resizable(True,False) window.title("Attendance System") window.configure(background='lavender')

message3 = tk.Label(window, text="Face Recognition Based Attendance System" ,fg="white",bg="#262523"

,width=33 ,height=1,font=('times', 29, ' bold ')) message3.place(x=20, y=10)

frame1 = tk.Frame(window, bg="SlateGray3") frame1.place(relx=0.11, rely=0.17, relwidth=0.70, relheight=0.80)

frame3 = tk.Frame(window, bg="#c4c6ce") frame3.place(relx=0.46, rely=0.10, relwidth=0.25, relheight=0.05)

frame4 = tk.Frame(window, bg="#c4c6ce") frame4.place(relx=0.26, rely=0.10, relwidth=0.25, relheight=0.05)

datef = tk.Label(frame4, text = day+"-"+mont[month]+"-"+year+" | ", fg="orange",bg="#262523" ,width=55

,height=1,font=('times', 20, ' bold ')) datef.pack(fill='both',expand=1)

clock = tk.Label(frame3,fg="orange",bg="#262523" ,width=55 ,height=1,font=('times', 20, ' bold ')) clock.pack(fill='both',expand=1)

tick()

head1 = tk.Label(frame1, text=" REGISTERED", fg="black",bg="SlateGray3" ,font=('times', 17, ' bold ') )

head1.place(x=0,y=0)

lbl3a = tk.Label(frame1, text="Course : " ,fg="black" ,bg="SlateGray3" ,font=('times', 14, ' bold ')) lbl3a.place(x=10, y=40)

my\_list = ["MSC-IT", "M-COM","MA"]

vyear = tk.StringVar()

vyear.set(my\_list[0]) # default value

txt3a = tk.OptionMenu(frame1,vyear,\*my\_list) txt3a.place(x=100, y=40)

lbl3aa = tk.Label(frame1, text="Year : ",width=10 ,fg="black" ,bg="SlateGray3" ,font=('times', 14, ' bold ')) lbl3aa.place(x=190, y=40)

my\_list3 = ["FY", "SY"] vyear3 = tk.StringVar()

vyear3.set(my\_list3[0]) # default value

txt3aa = tk.OptionMenu(frame1,vyear3,\*my\_list3) txt3aa.place(x=300, y=40)

lbl3b = tk.Label(frame1, text="Section : ",width=10 ,fg="black" ,bg="SlateGray3" ,font=('times', 14, ' bold ')) lbl3b.place(x=380, y=40)

my\_list2 = ["1", "2"] sec = tk.StringVar()

sec.set(my\_list2[0]) # default value

txt3b = tk.OptionMenu(frame1,sec,\*my\_list2) txt3b.place(x=480, y=40)

message = tk.Label(frame1, text="" ,fg="black" ,width=29,height=1, activebackground = "yellow" ,font=('times', 16, ' bold '))

message.place(x=90, y=470)

lbl3 = tk.Label(frame1, text="Attendance List",width=20 ,fg="black" ,bg="SlateGray3" ,height=1 ,font=('times', 17, ' bold '))

lbl3.place(x=120,y=120)

sclass = (vyear.get()) selyear= (vyear3.get()) selsec= (sec.get())

res=0

exists = os.path.isfile("StudentDetails/"+sclass+"/"+selyear+"/"+selsec+"/"+"StudentDetails.csv") if exists:

with open("StudentDetails/"+sclass+"/"+selyear+"/"+selsec+"/"+"StudentDetails.csv", 'r') as csvFile1: reader1 = csv.reader(csvFile1)

for l in reader1:

res = res + 1 res = (res // 2) - 1 csvFile1.close()

else:

res = 0

message.configure(text='Total Registrations till now : '+str(res))

##################### EXCEL VIEW #################################

def open\_student\_excel():

sclass = (vyear.get()) selyear= (vyear3.get()) selsec= (sec.get())

file="C:/Users/square/Desktop/Projects2/Attendance\_Systems/StudentDetails/ "+sclass+"/"+selyear+"/"+selsec+"/"+"StudentDetails.csv"

print("StudentDetails/"+sclass+"/"+selyear+"/"+selsec+"/"+"StudentDetails.csv") os.startfile(file)

def open\_attendance\_excel():

ts = time.time() sclass = (vyear.get())

selyear= (vyear3.get()) selsec= (sec.get())

date = datetime.datetime.fromtimestamp(ts).strftime('%d-%m-%Y')

exists = os.path.isfile("C:/Users/square/Desktop/Projects2/Attendance\_Systems/StudentDetails/ "+sclass+"/"+selyear+"/"+selsec+"/"+date+"/"+"Attendance\_" + date + ".csv")

if exists:

file = "C:/Users/square/Desktop/Projects2/Attendance\_Systems/StudentDetails/"+sclass+"/"+selyear+"/"+selsec+"/"+date+"/"+"Attendance\_" + date + ".csv"

os.startfile(file) else:

mess.\_show(title='No Record Found.', message="Attendance has not been taken yet.")

##################### MENUBAR #################################

menubar = tk.Menu(window,relief='ridge') filemenu = tk.Menu(menubar,tearoff=0)

filemenu.add\_command(label='Students Detail Excel', command = open\_student\_excel) filemenu.add\_command(label='Attendance Excel', command = open\_attendance\_excel) menubar.add\_cascade(label='View',font=('times', 29, ' bold '),menu=filemenu)

filemenu1 = tk.Menu(menubar,tearoff=0) filemenu1.add\_command(label='Change Password', command = change\_pass) filemenu1.add\_command(label='Contact Us', command = contact) filemenu1.add\_command(label='Exit',command = window.destroy) menubar.add\_cascade(label='Help',font=('times', 29, ' bold '),menu=filemenu1)

################## TREEVIEW ATTENDANCE TABLE ####################

tv= ttk.Treeview(frame1,height =10,columns = ('name','date','time')) tv.column('#0',width=82)

tv.column('name',width=130) tv.column('date',width=133) tv.column('time',width=133)

tv.grid(row=2,column=0,padx=(40,0),pady=(150,0),columnspan=4) tv.heading('#0',text ='ID')

tv.heading('name',text ='NAME') tv.heading('date',text ='DATE') tv.heading('time',text ='TIME')

###################### SCROLLBAR ################################

scroll=ttk.Scrollbar(frame1,orient='vertical',command=tv.yview) scroll.grid(row=2,column=4,padx=(0,100),pady=(150,0),sticky='ns') tv.configure(yscrollcommand=scroll.set)

###################### BUTTONS ##################################

trackImg = tk.Button(frame1, text="Take Attendance", command=TrackImages ,fg="white" ,bg="OliveDrab4"

,width=30 ,height=1, activebackground = "white" ,font=('times', 15, ' bold ')) trackImg.place(x=100, y=80)

regWindow = tk.Button(frame1, text="REGISTER", command=Register ,fg="white" ,bg="DeepSkyBlue3" ,width=10

,height=1, activebackground = "white" ,font=('times', 15, ' bold ')) regWindow.place(x=100, y=400)

quitWindow = tk.Button(frame1, text="QUIT", command=window.destroy ,fg="white" ,bg="red3" ,width=10 ,height=1, activebackground = "white" ,font=('times', 15, ' bold '))

quitWindow.place(x=300, y=400)

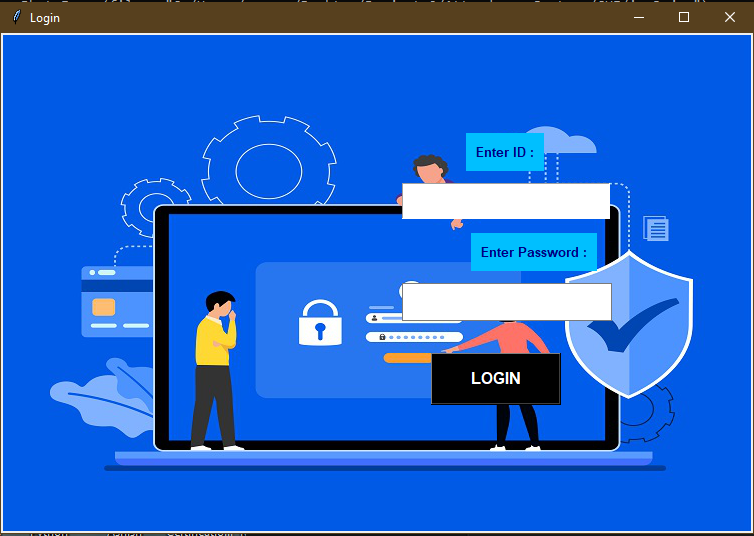
##################### END ######################################

window.configure(menu=menubar) window.mainloop()

#############################################################################

### 

### Admin Login

****

**Admin View**

1. **View Excels**

###### Student Details Excel

* 1. **Attendance Excel**



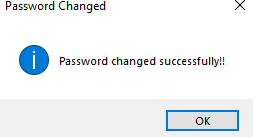
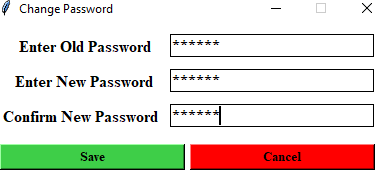
1. **Take Attendance**



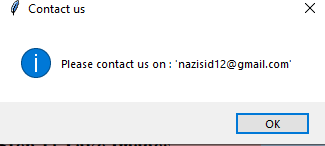
1. **Help**



###### Change Password

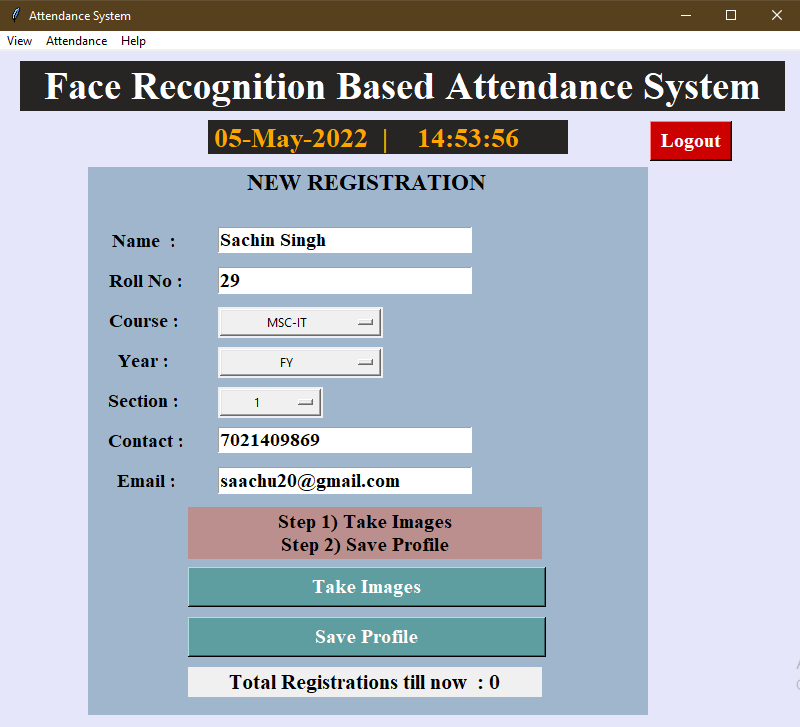


* 1. **Contact Us**

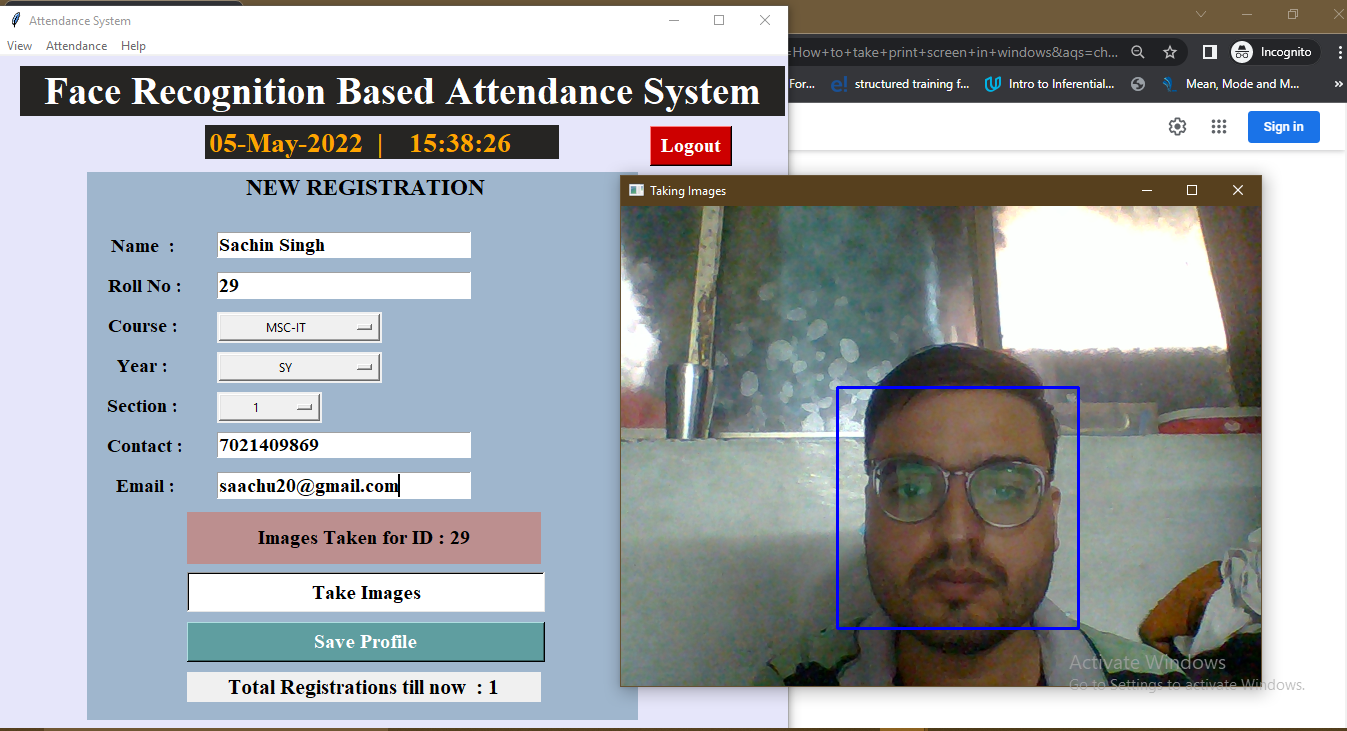


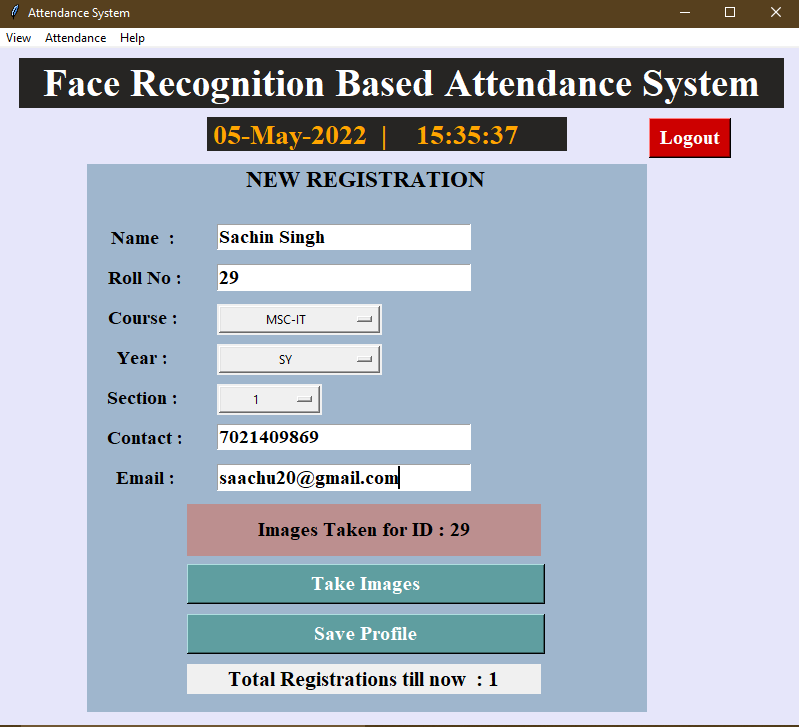
* 1. **Exit**

### Student Registration

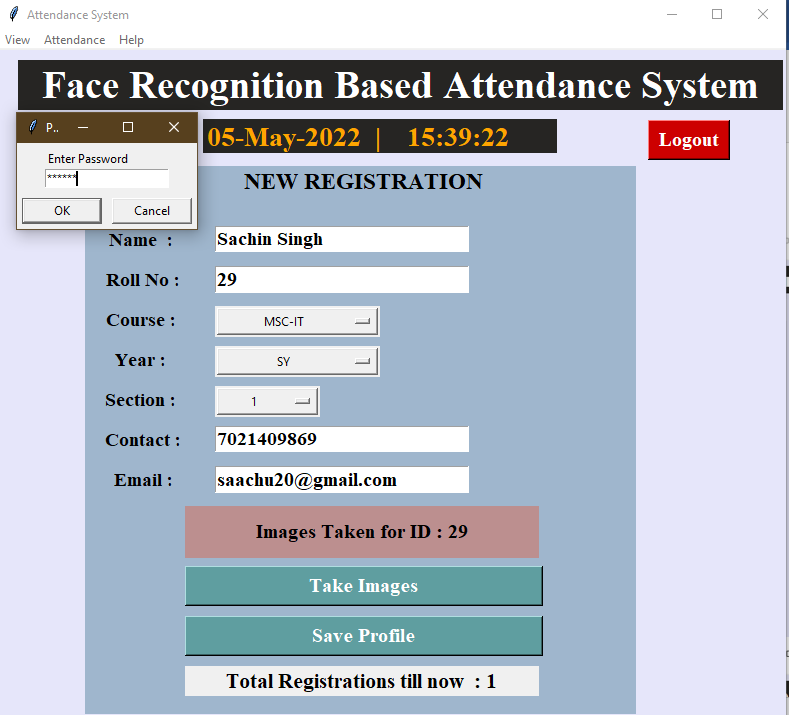
****

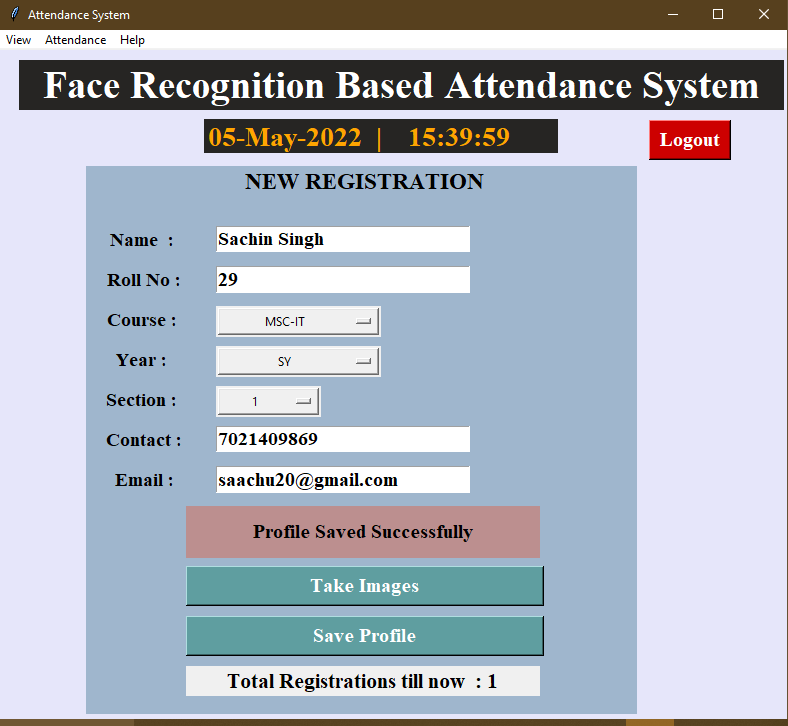
**TAKE IMAGE**

****

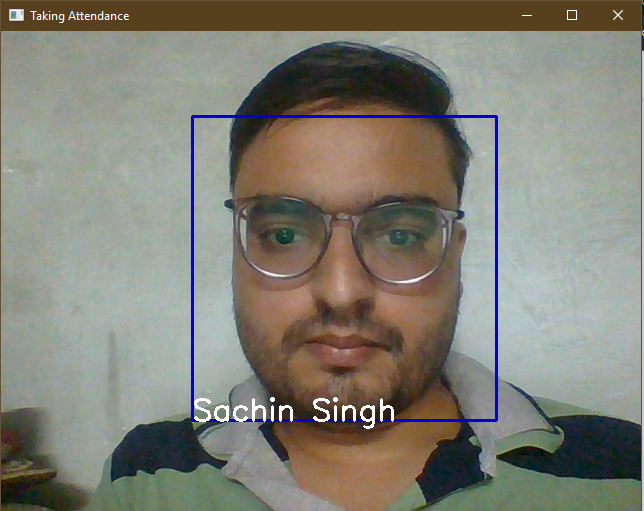


### SAVING PROFILE

****

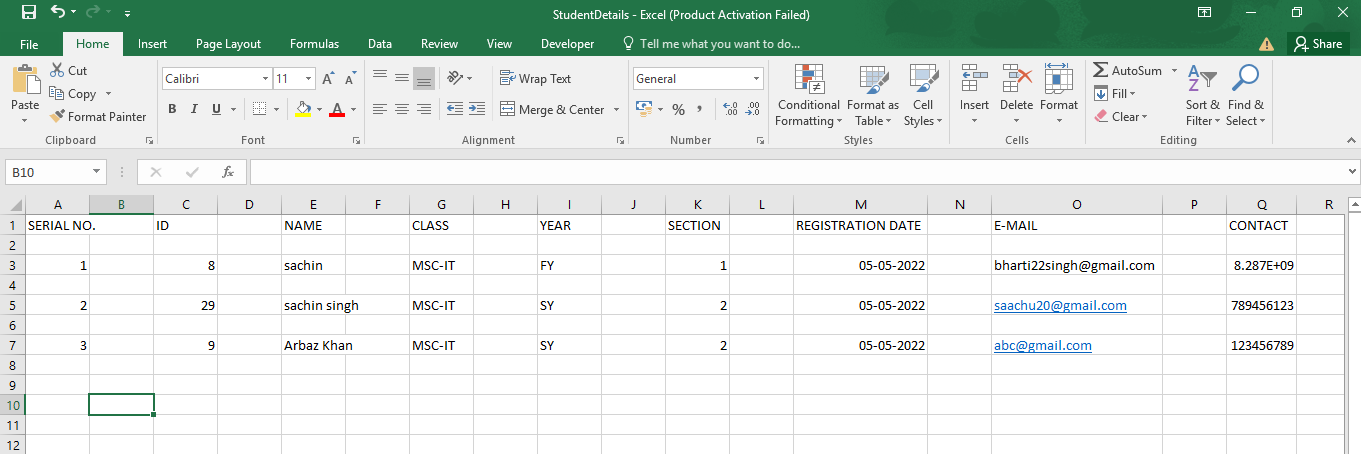


**TAKE ATTENDANCE**

****

**REPORT LAYOUT:**

**StudentDetails:**

****

# FUTURE ENHANCEMENT

**Future Enhancement:**

1. Seeing attendance or editing requires a master face print which can be set earlier so students can’t change their records.
2. Multiple times and multiple faces are not taken into consideration.
3. Defaulter list.
4. Report for student whose attendance is less than 50%.
5. Setting in and out timing as well so as to create a proxy payroll system as well.

# REFERENCE

AND BIBLIOGRAPHY

**I referred to the following Websites to give functionality to my project: Websites:-**

* stackoverflow.com
* tutorialspoint.com
* codeproject.com
* w3schools.com
* youtube.com
* stackblitz.com
* realpython.com
* towardsdatascience.com
* github.com
* [plus2net.com](http://www.plus2net.com/)
* geeksforgeeks.org