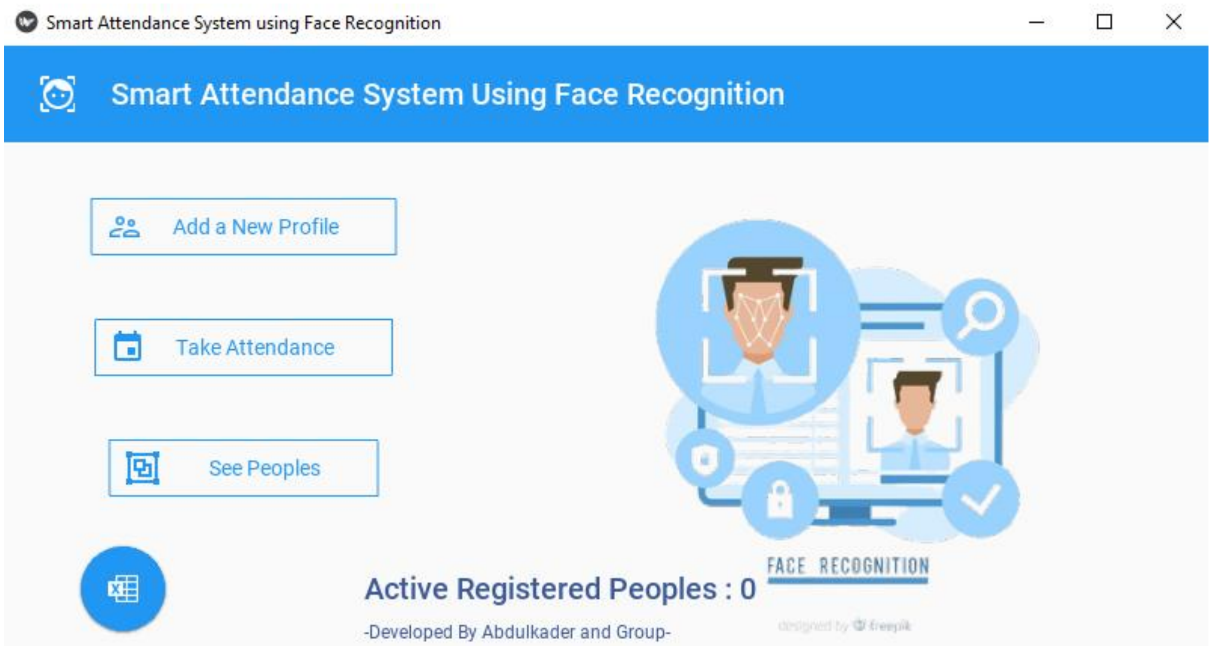
CHAPTER 6

6.1Snapshots

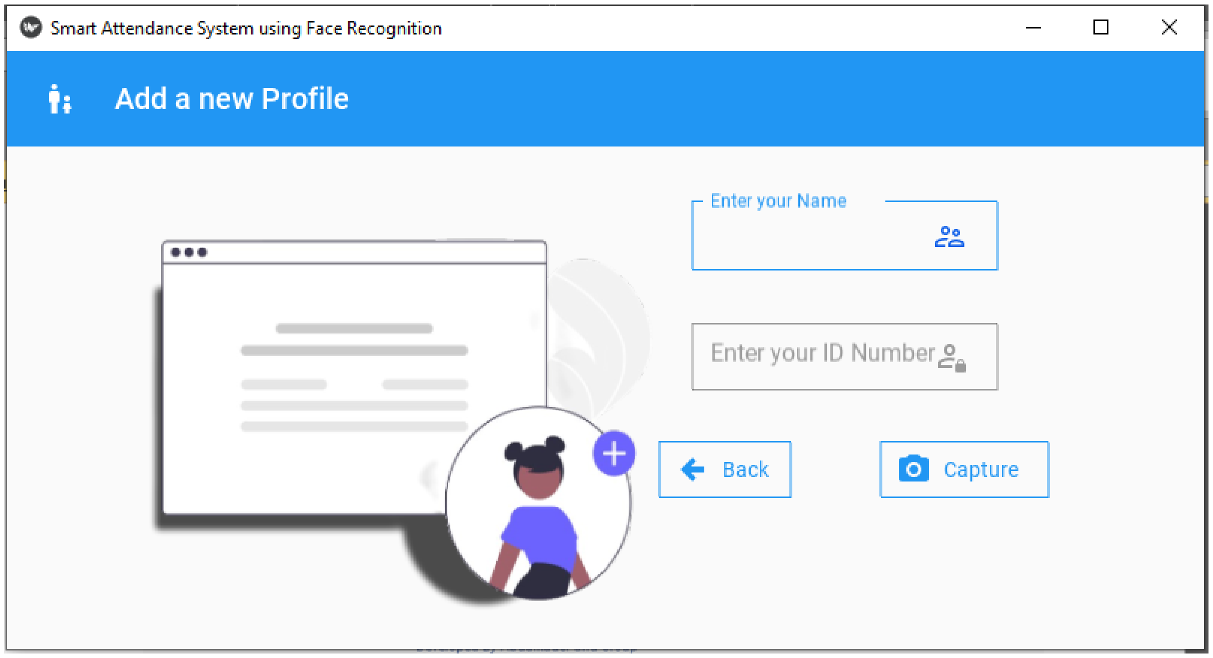
Home page:


Smart Attendance System using Face Recognition
Smart Attendance System Using Face Recognition
Add a New Profile
Take Attendance
See Peoples
FACE RECOGNITION
designed by freepik

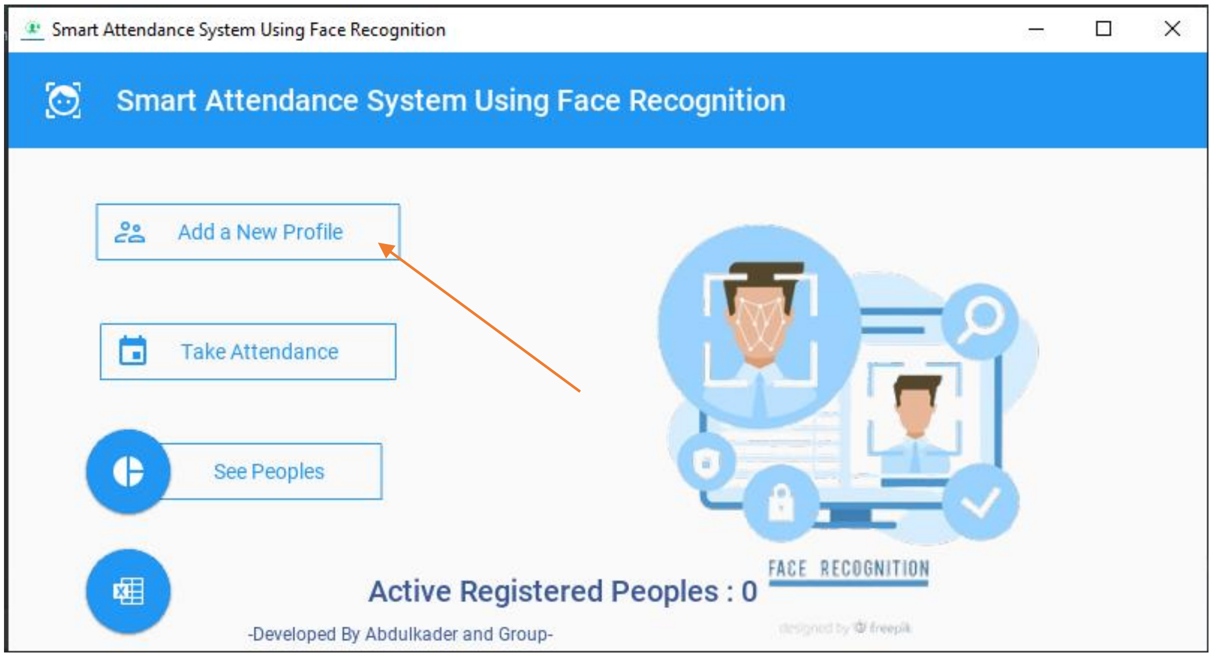
Active Registered Peoples : 0

-Developed By Abdulkader and Group-

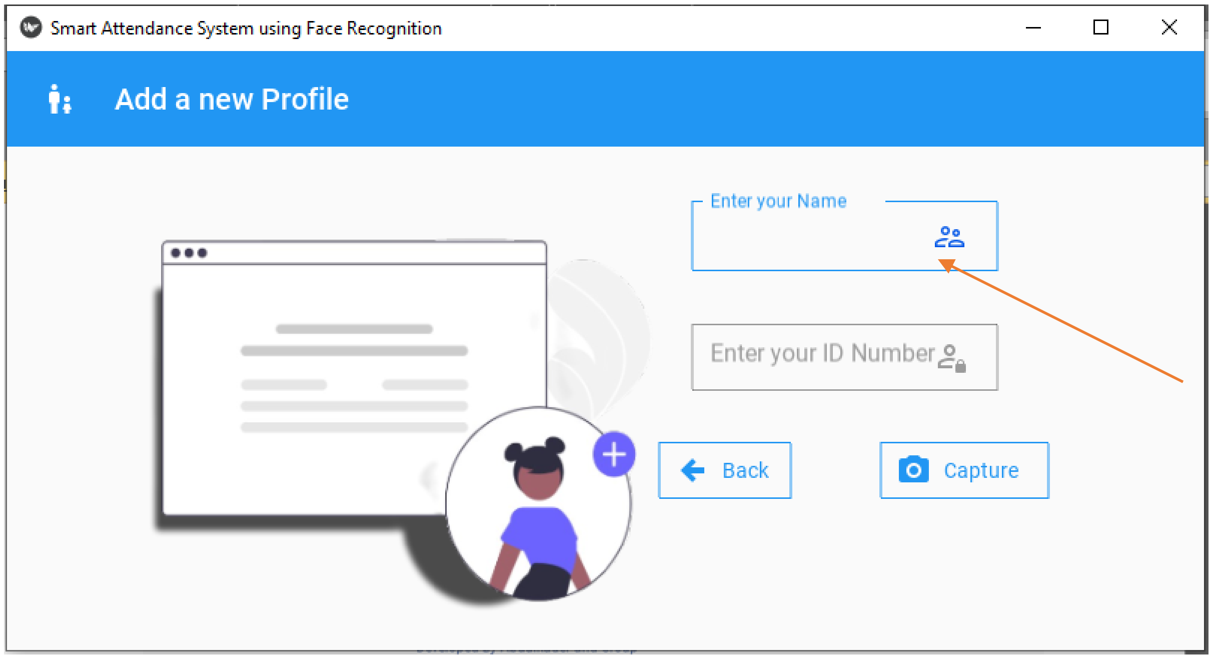
Registration page:



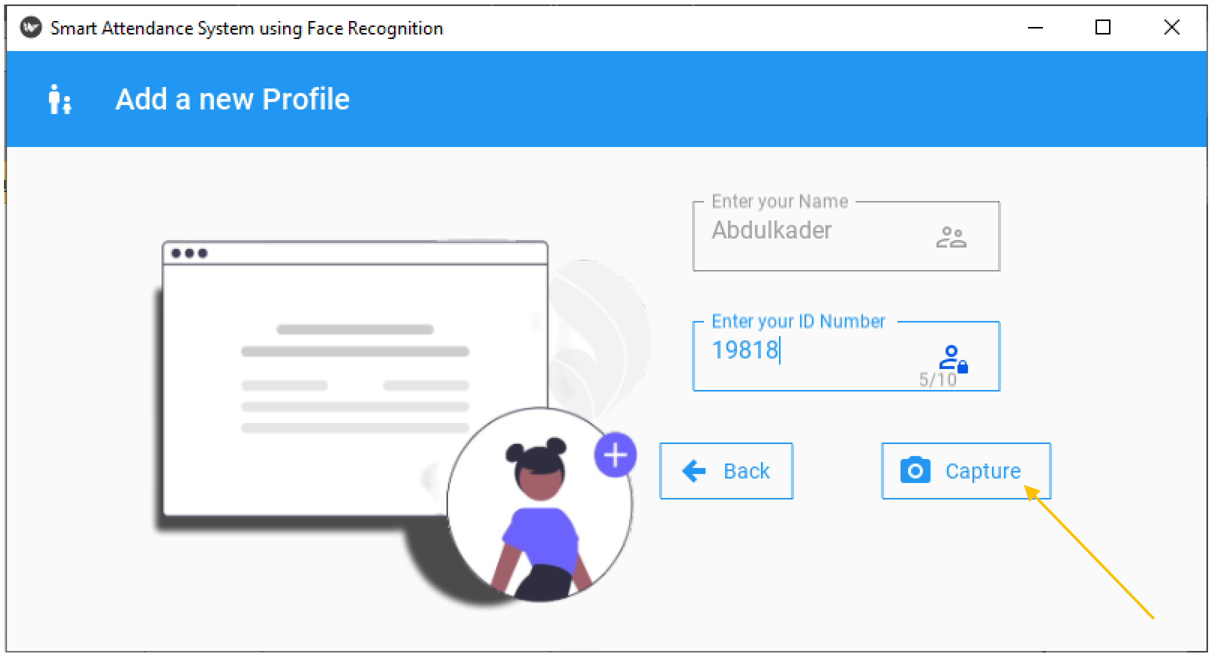
Home Screen:


Smart Attendance System Using Face Recognition
Smart Attendance System Using Face Recognition
Add a New Profile
Take Attendance
See Peoples
FACE RECOGNITION
Active Registered Peoples : 0
-Developed By Abdulkader and Group-

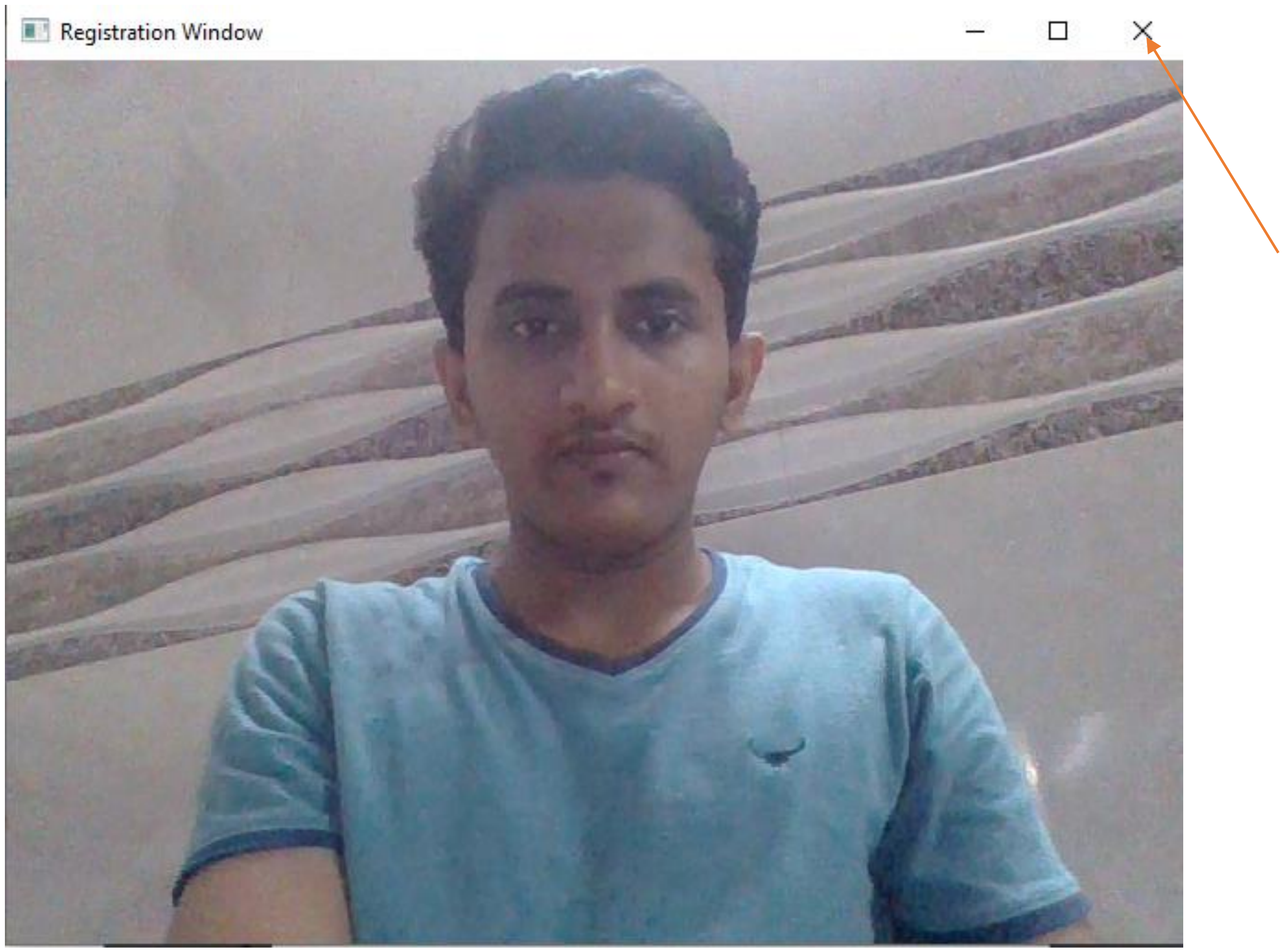
Registerattion screen:



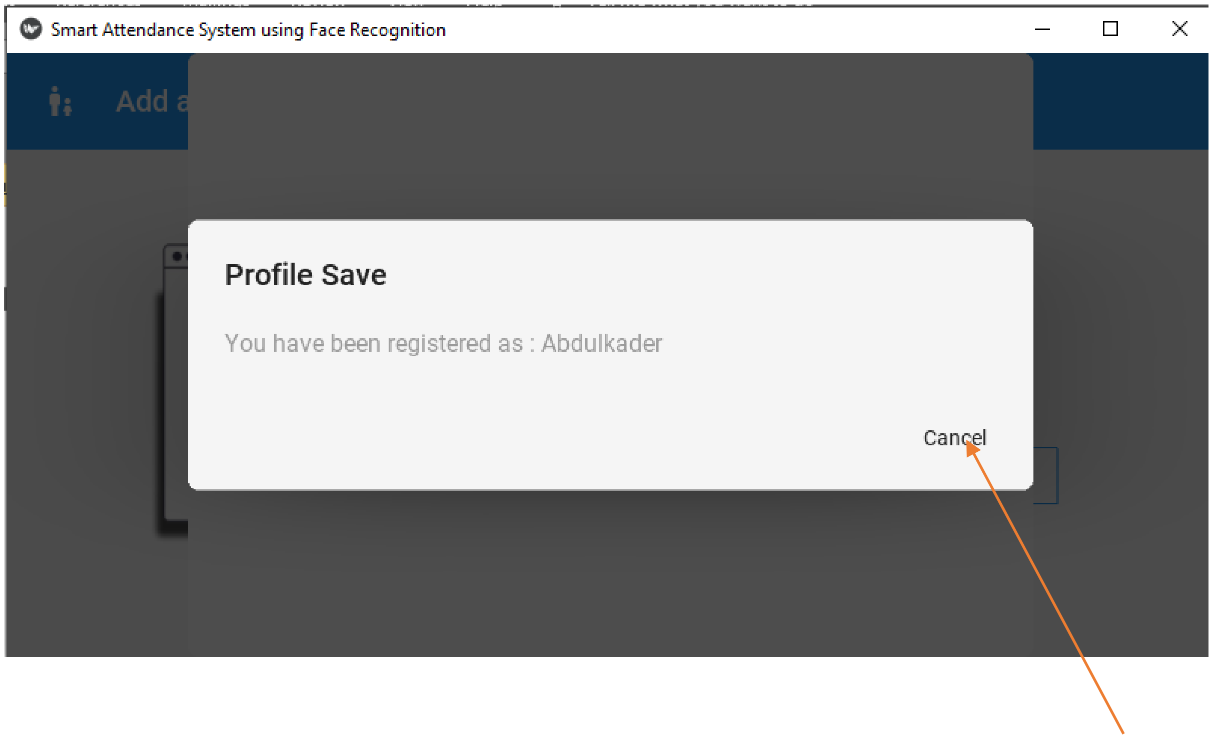
Registeration screen after inputing some values:



After clicking the capture button camera window will open



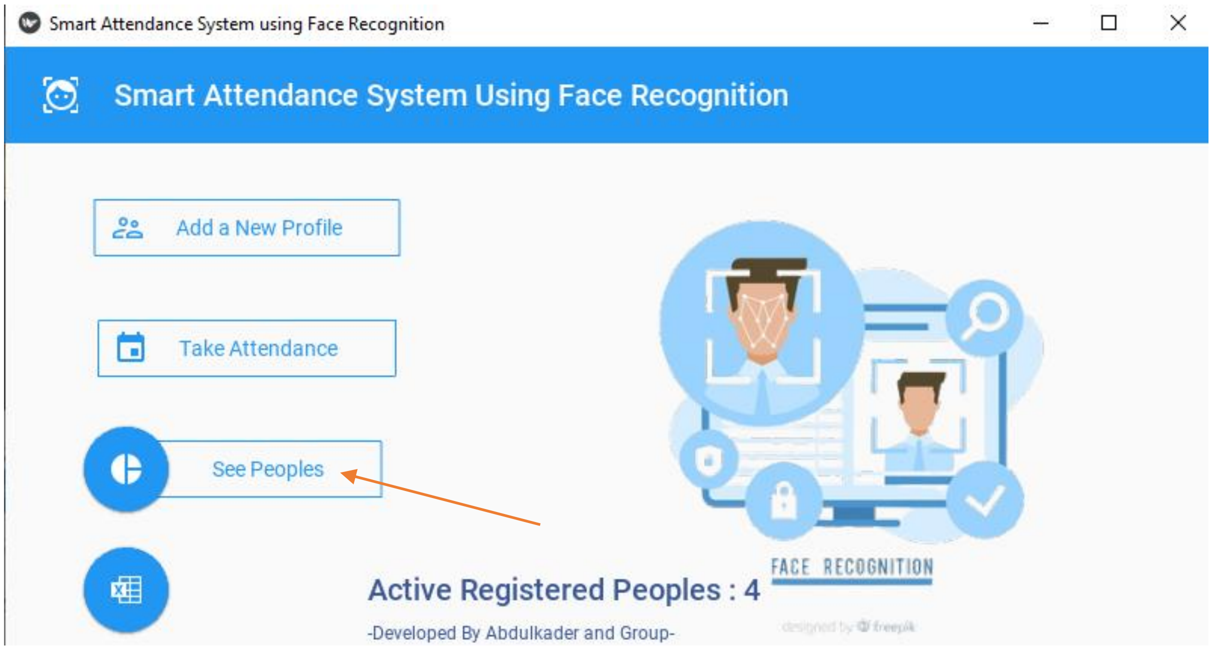
Profile saving popup:



Profile Save

You have been registered as : Abdulkader

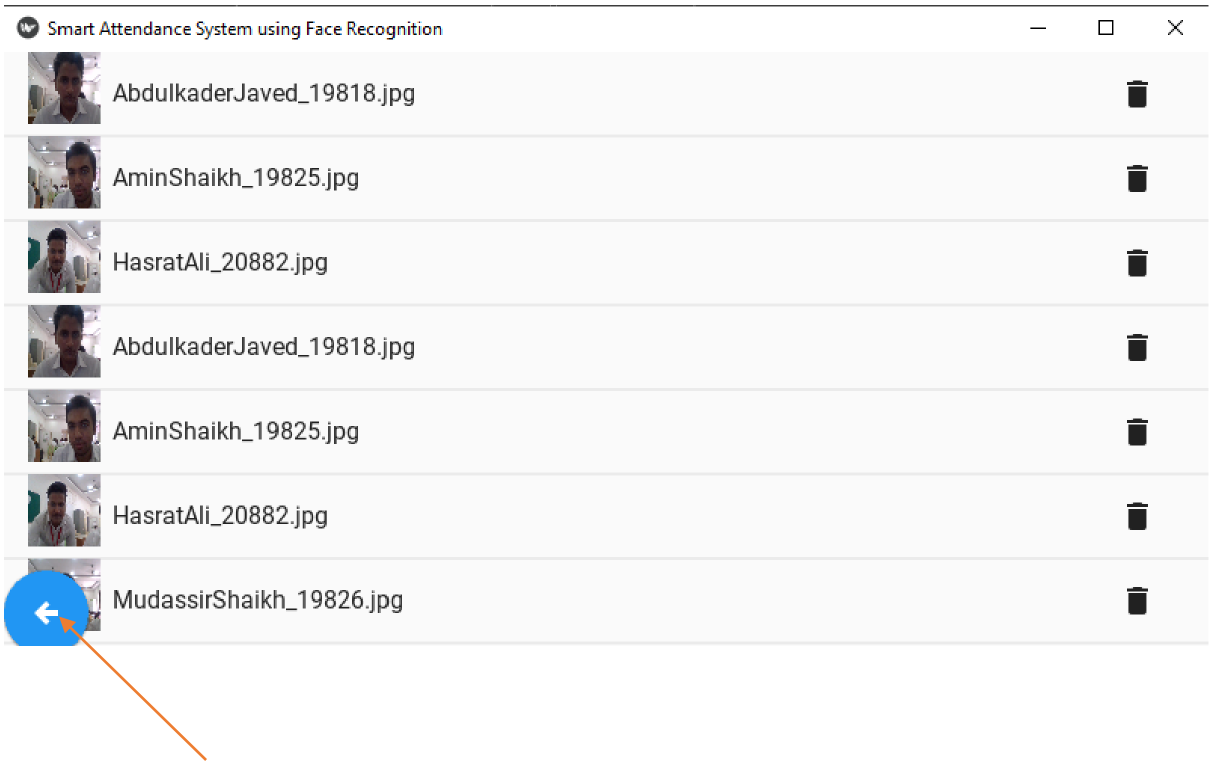
Home Screen after saving 4 profiles


Smart Attendance System using Face Recognition
Smart Attendance System Using Face Recognition
Add a New Profile
Take Attendance
See Peoples
FACE RECOGNITION

Active Registered Peoples : 4

-Developed By Abdulkader and Group-

After clicking on the See Peoples button Active Registered peoples list window will open


Smart Attendance System using Face Recognition

AbdulkaderJaved_19818.jpg

AminShaikh_19825.jpg

HasratAli_20882.jpg

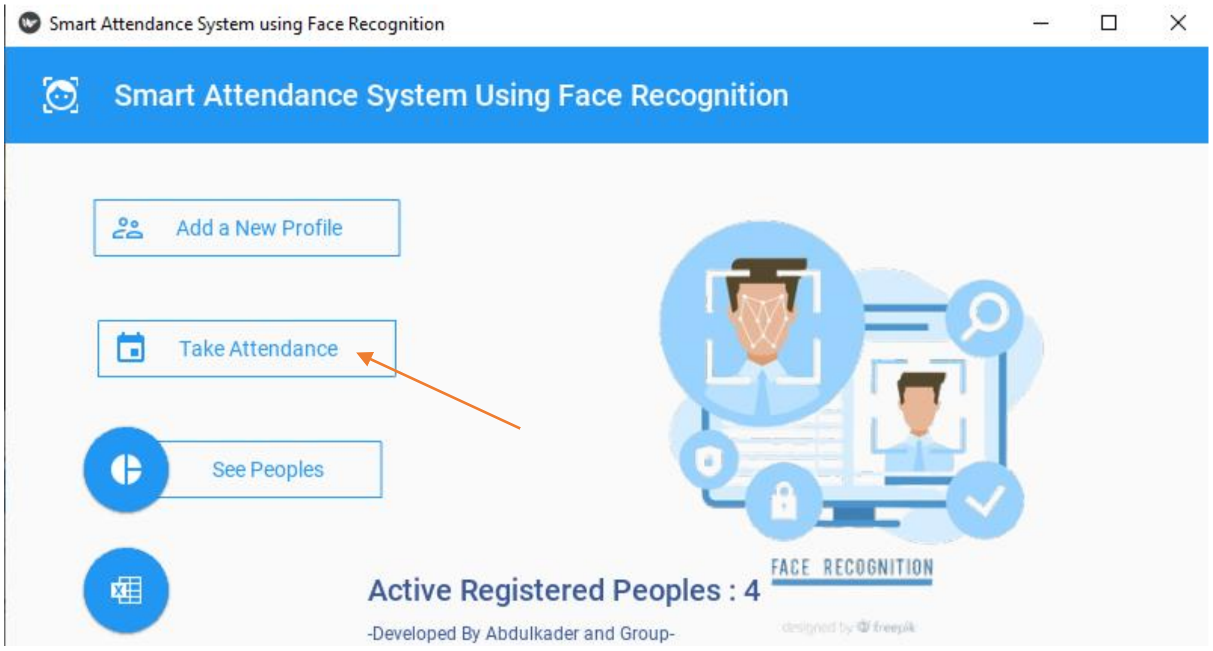
AbdulkaderJaved_19818.jpg

AminShaikh_19825.jpg

HasratAli_20882.jpg

MudassirShaikh_19826.jpg

After clicking the back button it will come back to Home Screen

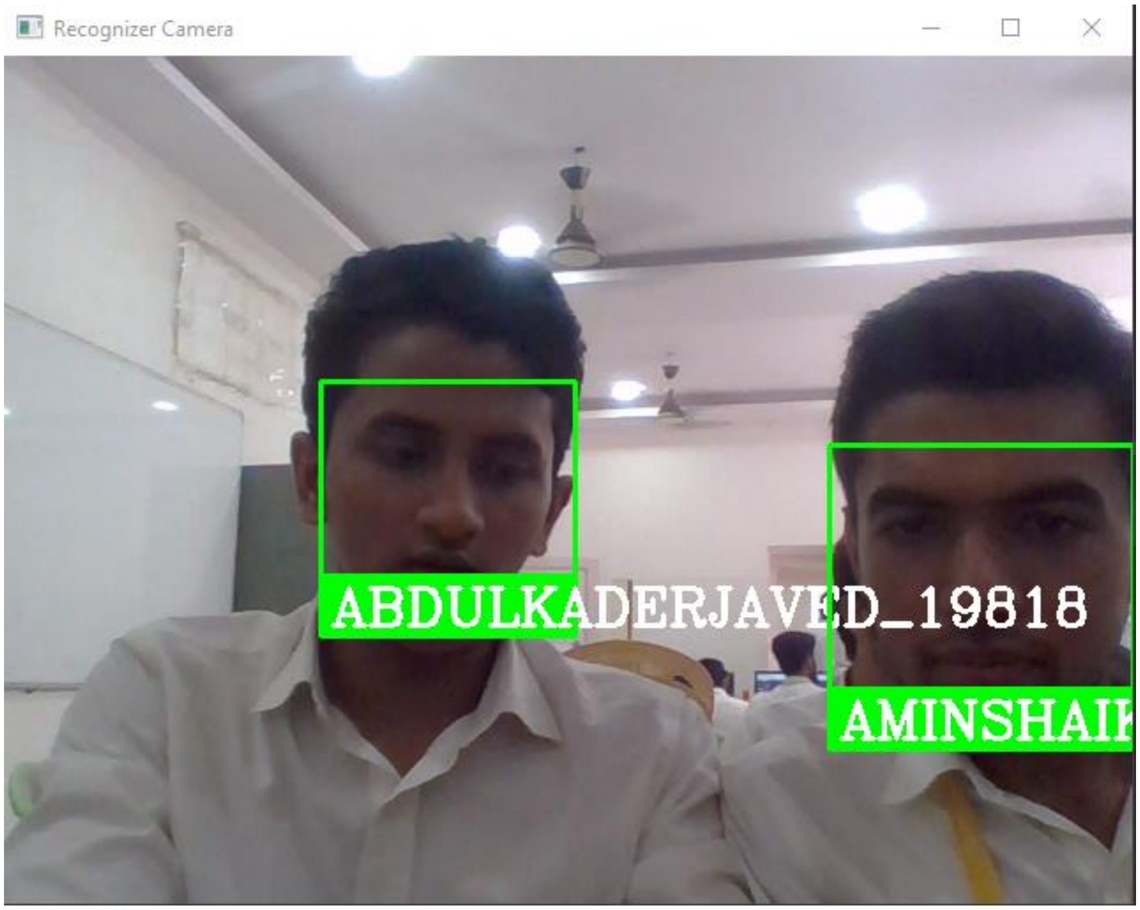

Smart Attendance System using Face Recognition

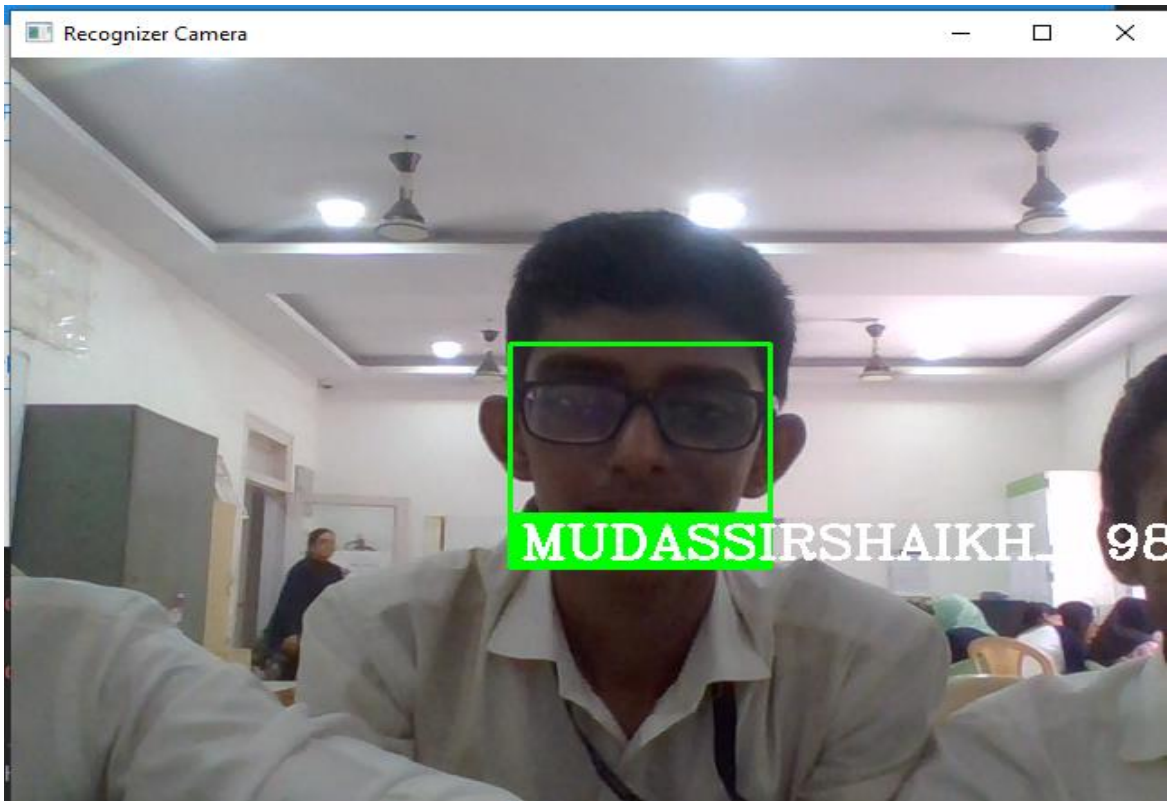
Smart Attendance System Using Face Recognition

Active Registered Peoples : 4

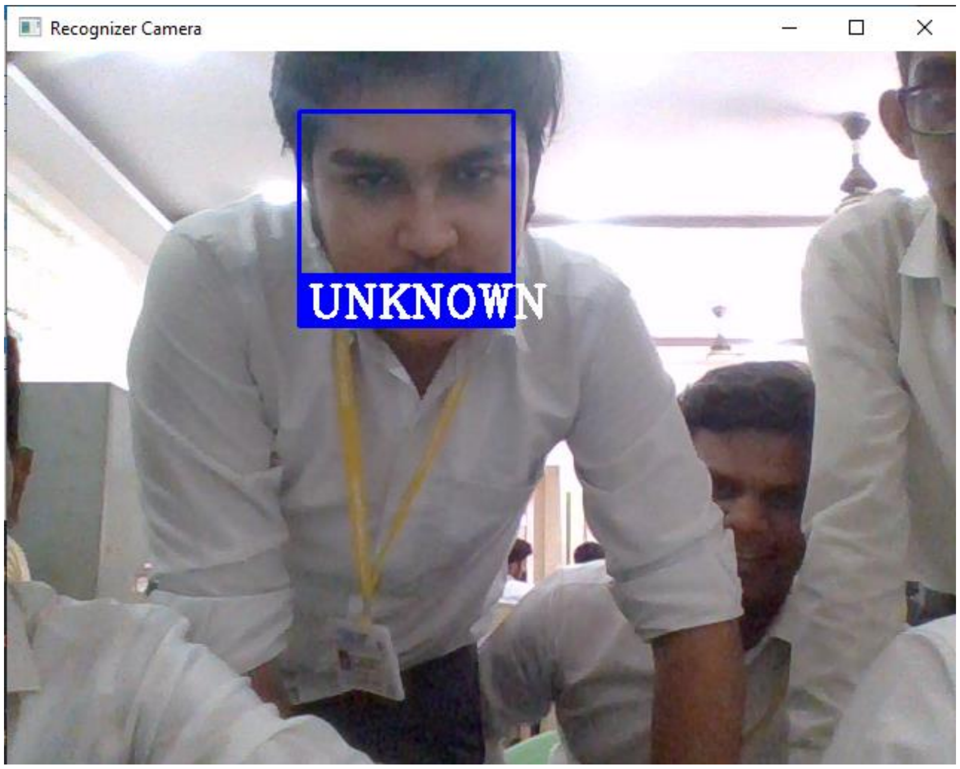
-Developed By Abdulkader and Group-

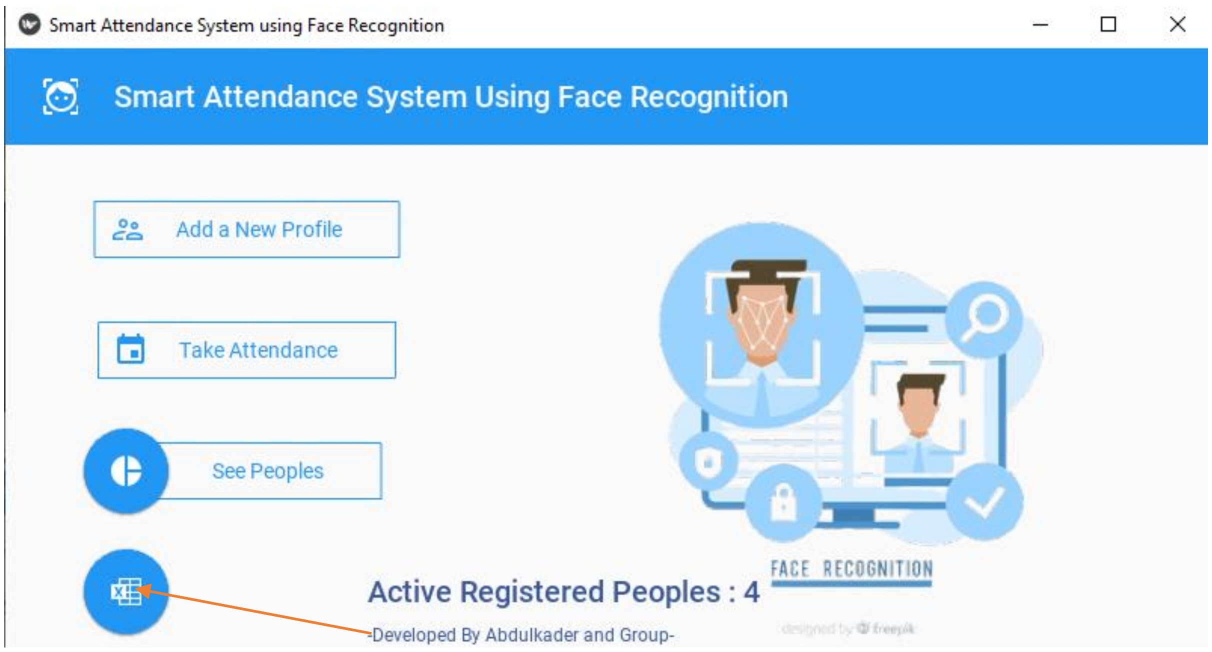
After clicking on the Take Attendance Button Recognizer Camera will Start





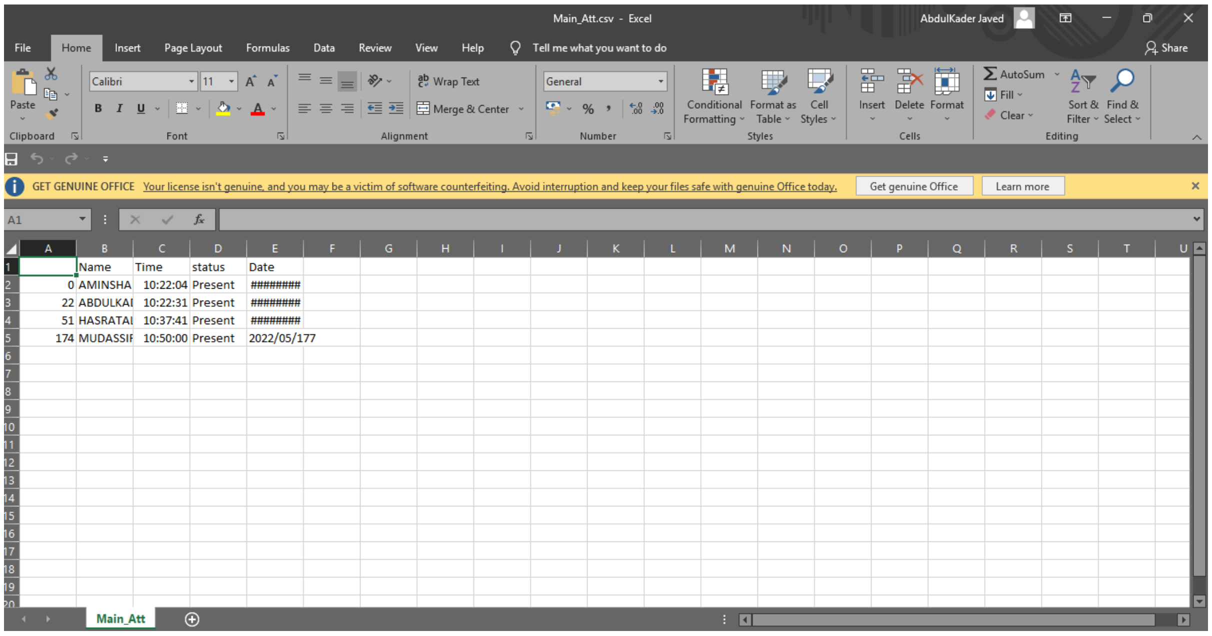
If software doesn’t recognises the face then it will show UNKNOWN and will not mark the attendance




Smart Attendance System using Face Recognition
Smart Attendance System Using Face Recognition
Add a New Profile
Take Attendance
See Peoples
FACE RECOGNITION

Active Registered Peoples : 4

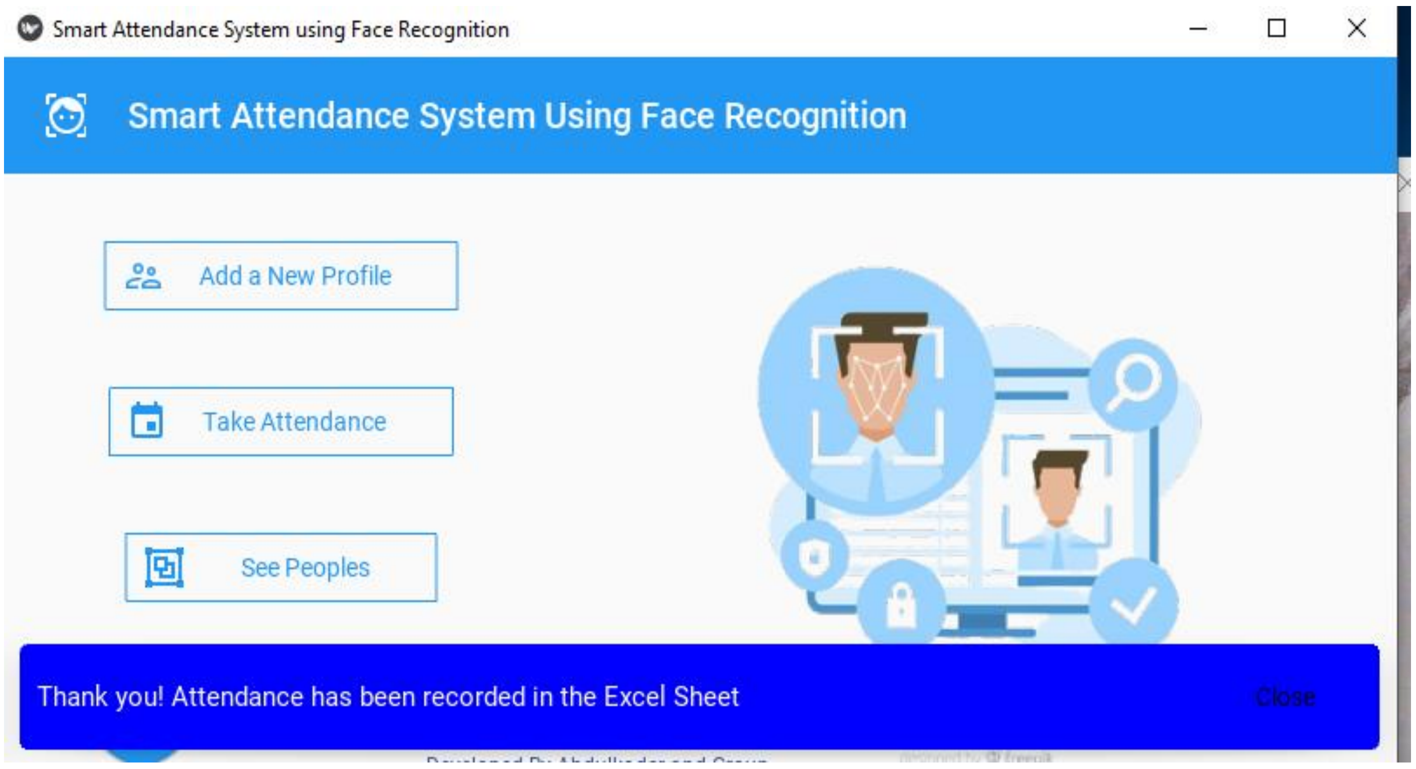
-Developed By Abdulkader and Group-



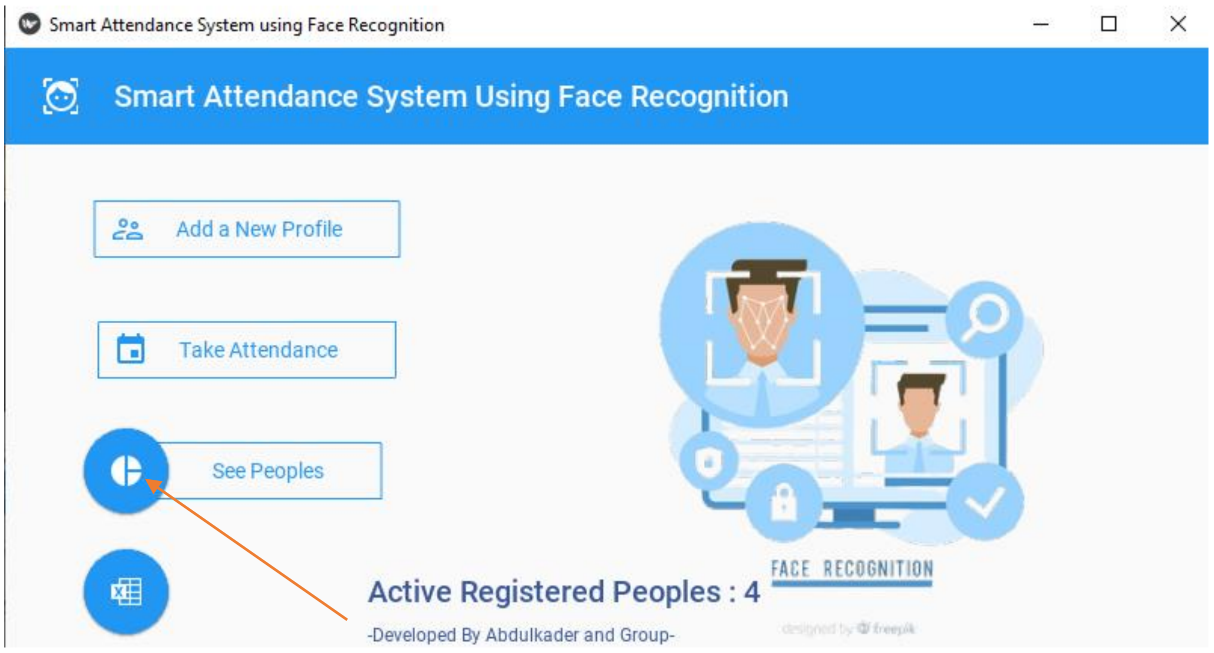
GET GENUINE OFFICE Your license isn't genuine, and you may be a victim of software counterfeiting. Avoid interruption and keep your files safe with genuine Office today.

	Name	Time	status	Date
0	AMINSHA	10:22:04	Present	#########
22	ABDULKAI	10:22:31	Present	#########
51	HASRATAI	10:37:41	Present	#########
174	MUDASSIF	10:50:00	Present	2022/05/177

HOME SCREEN WITH SNACK BAR


Smart Attendance System using Face Recognition
Smart Attendance System Using Face Recognition
Add a New Profile
Take Attendance
See Peoples
Close

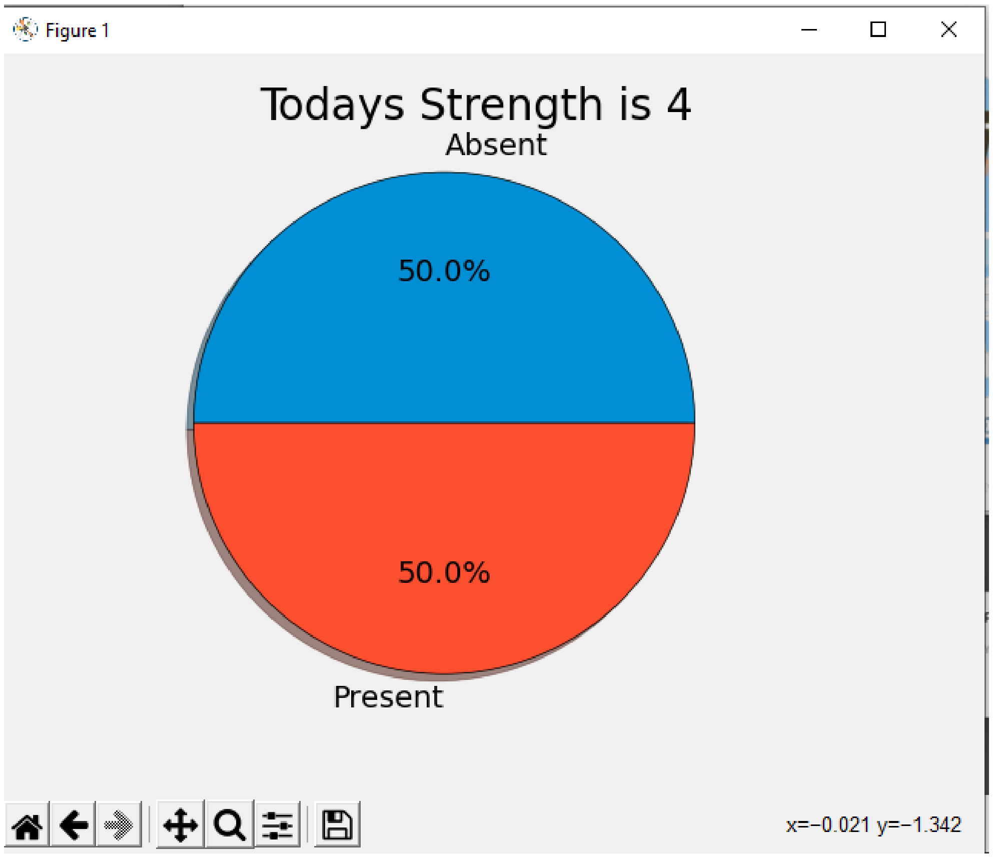
Thank you! Attendance has been recorded in the Excel Sheet


Smart Attendance System using Face Recognition
Smart Attendance System Using Face Recognition
Add a New Profile
Take Attendance
See Peoples
FACE RECOGNITION
designed by freepik

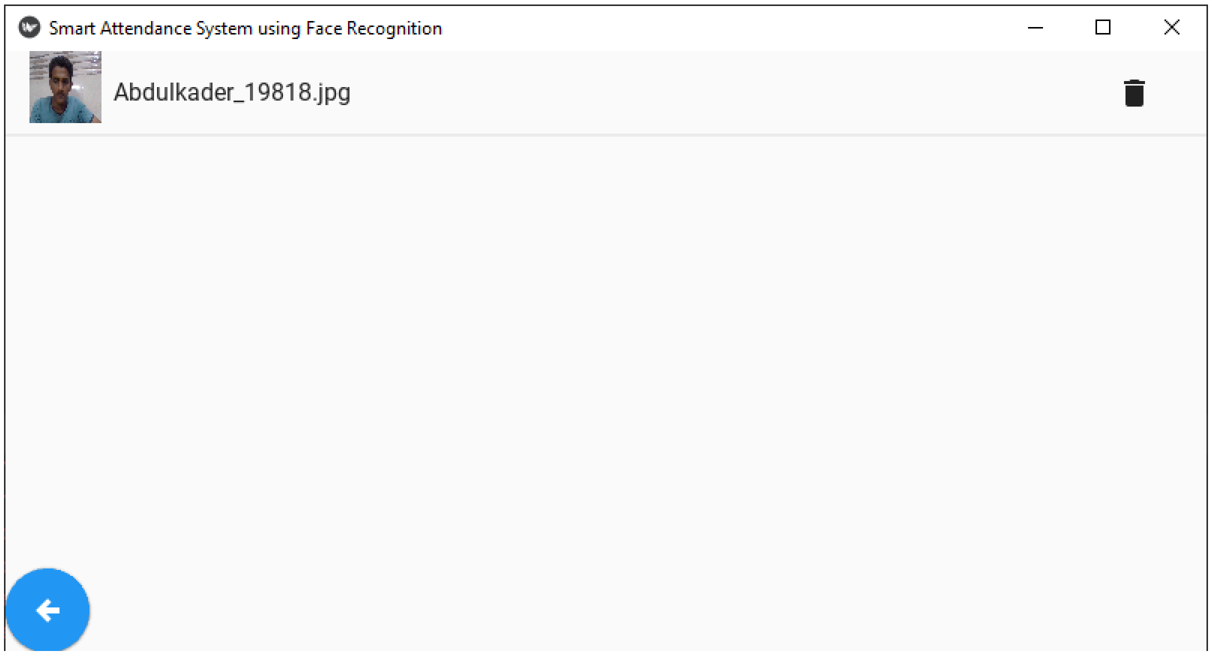
Active Registered Peoples : 4

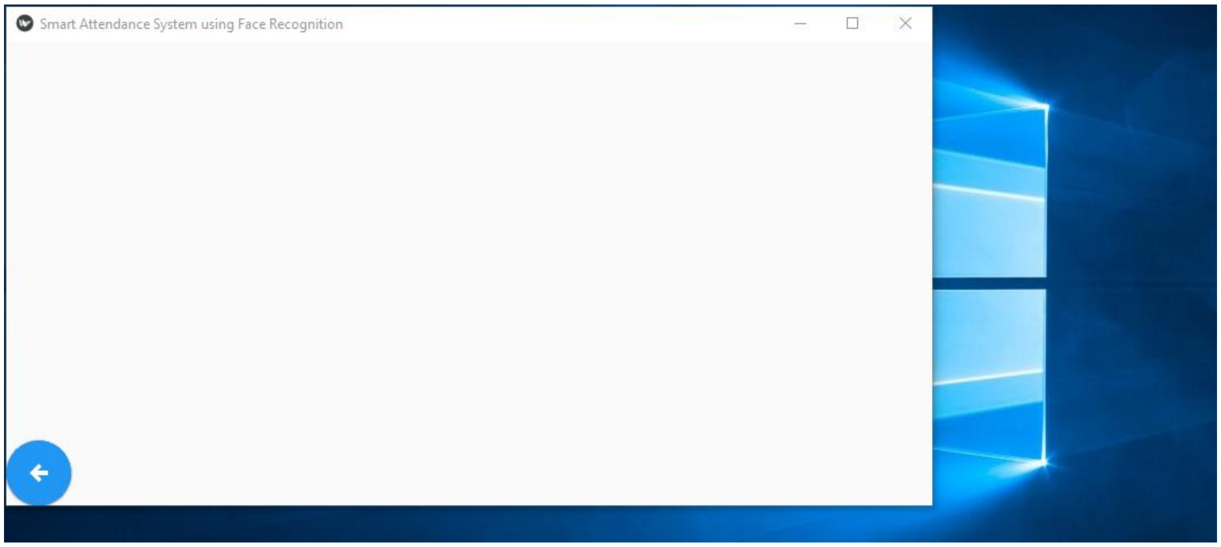
-Developed By Abdulkader and Group-

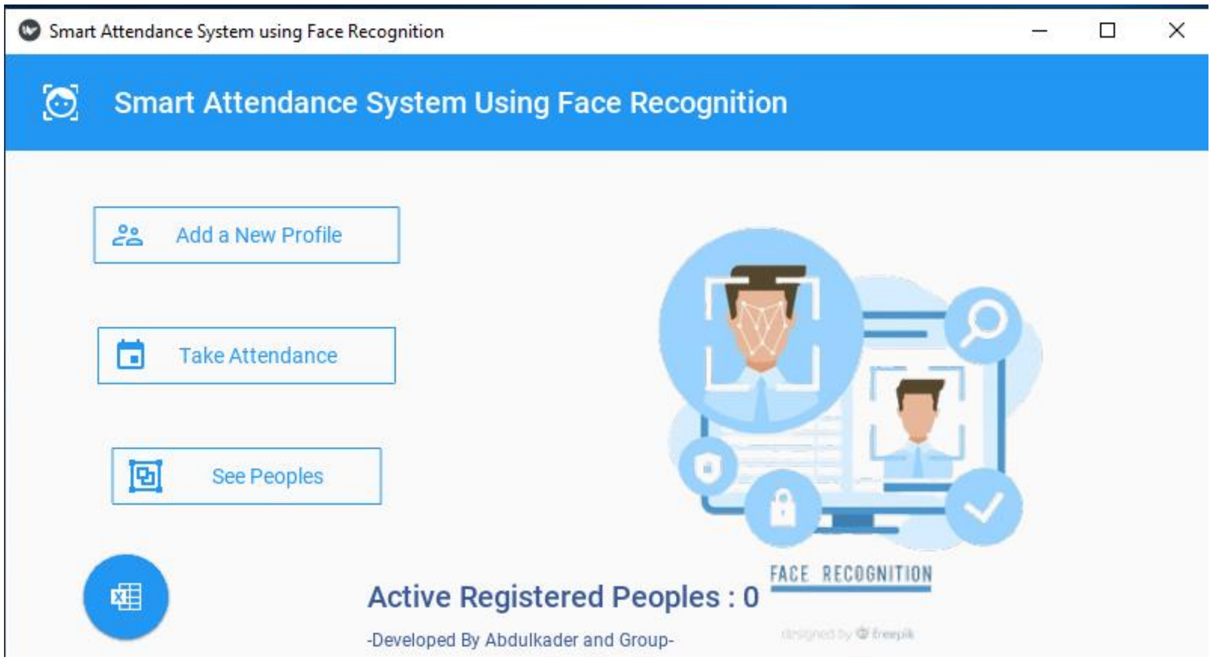
As we click on the pie button it will start the statistics(pie chart)



Deleting Profile:






Smart Attendance System using Face Recognition

Smart Attendance System Using Face Recognition

Active Registered Peoples : 0

-Developed By Abdulkader and Group-

6.3 Code

Main_kivy.py

```
from tkinter.constants import COMMAND
from typing import Text

import notification
import self as self
from kivy import Config

from kivy.properties import ListProperty, StringProperty
from kivy.uix.modalview import ModalView
from kivy.uix.scrollview import ScrollView
from kivymd.uix.snackbar import Snackbar
from kivymd.uix.behaviors import HoverBehavior
from kivymd.uix.boxlayout import MDBoxLayout
from kivymd.uix.expansionpanel import MDExpansionPanel,
MDExpansionPanelOneLine,MDExpansionPanelThreeLine
from kivymd.uix.list import MDList, ThreeLineListItem,
ThreeLineAvatarListItem, OneLineListItem, OneLineIconListItem, \
    TwoLineIconListItem, OneLineAvatarIconListItem
from kivymd.uix.list import
IconLeftWidget,ImageLeftWidget,IconRightWidget,IRightBodyTouch
from kivy.uix.button import Button
from kivymd.app import MDApp

from kivymd.uix.label import MDLabel
from kivymd.uix.button import MDFlatButton, MDFloatingActionButton,
MDFillRoundFlatIconButton,MDRectangleFlatButton, MDIconButton,MDRaisedButton
from kivy.uix.screenmanager import Screen,ScreenManager
from kivy.uix.scrollview import ScrollView
from kivy.uix.image import Image
from kivymd.uix.textfield import MDTextField,MDTextFieldRect
from matplotlib import pyplot as plt
from openpyxl import load_workbook
import pandas as pd
from kivy.lang import Builder
from kivymd.toast import toast
import cv2
from kivymd.utils import asynckivy
import numpy as np
import face_recognition
import os
from datetime import datetime
from tkinter import *
from tkinter import messagebox
from kivymd.uix.dialog import MDDialog
from kivy.core.window import Window
from playsound import playsound
from gtts import gTTS

from kivymd_extensions.sweetalert import SweetAlert
```

```
Window.size = (800,400)
Window.title = "Smart Attendance System Using Face Recognition"
Window.resizeable = False


newScreen = """



<ListwithCheckbox>:
    IconLeftWidget:
        icon : root.icon

    IconRightWidget:
        icon : "delete"
        on_release:app.dele(root)


ScreenManager:

    MenuScreen:
    ProfileScreen:
    Gallery:
<MenuScreen>:
    name : 'menu'


    BoxLayout:
        orientation : 'vertical'
        MDToolbar:
            left_action_items:[["face-recognition"]]

            title : "Smart Attendance System Using Face Recognition "

        MDLabel:
            text:""
            halign : 'center'
        Image:
            source  : "C:/Desktop/Face_Attend/fr123.png"
            pos_hint: {'center_x':0.7,'center_y':0.2}
            size_hint_x : 12
            height : "20dp"
            width : "80dp"
            size_hint_y : 12

    MDRectangleFlatIconButton:
        icon : 'account-supervisor-outline'
        text:'    Add a New Profile        '


        pos_hint:{'center_x':0.2 , 'center_y':0.7}
        elevation:20

        on_release :

            root.manager.current = 'Profile'
            root.manager.transition.direction = 'left'
    MDFloatingActionButton:
        icon :"microsoft-excel"
        pos_hint : {'center_x':0.1,'center_y':0.1}
        on_release : app.op()

    MDRectangleFlatIconButton:
        icon : 'calendar'
        text:'    Take Attendance        '
        pos_hint:{'center_x':0.2 , 'center_y':0.5}
        on_release : app.att(self)
    MDLabel:
        id:registered
        font_style:'H6'
        pos_hint : {'center_x':0.8,'center_y':0.1}
        text_color:(59/255,89/255,152/255)
        theme_text_color: 'Custom'
    MDLabel:
        text : "-Developed By Abdulkader and Group-"
        font_style:'Caption'
        pos_hint : {'center_x':0.8,'center_y':0.03}
        text_color:(59/255,89/255,152/255)
        theme_text_color: 'Custom'

    MDRectangleFlatIconButton:
        icon : 'group'
        text:'        See Peoples        '

        pos_hint:{'center_x':0.2 , 'center_y':0.3}
        on_release :
            root.manager.transition.direction = 'right'
            root.manager.current = 'Gall'
    MDFloatingActionButton:
        icon : 'chart-pie'
        pos_hint:{'center_x':0.1,'center_y':0.3}
        on_release:app.Attendance_Visualize()


<ProfileScreen>:
    name : 'Profile'

    MDTextField:
        mode: "rectangle"
        fill_color: 0, 0, 0, .4
        id : naam
        icon_right: "account-supervisor-outline"
        icon_right_color: app.theme_cls.primary_color
        hint_text : 'Enter your Name'
        pos_hint:{'center_x':0.7 , 'center_y':0.7}
        size_hint_x:None
        width : 180
    MDTextField:
        id : rollno
        required : True
        mode: "rectangle"
        fill_color: 0, 0, 0, .4
        hint_text : 'Enter your ID Number '
        icon_right: "account-lock-outline"
        icon_right_color: app.theme_cls.primary_color
        pos_hint:{'center_x':0.7 , 'center_y':0.5}
        required :True
        max_text_length:10
        helper_text_mode: "on_error"
        helper_text: "Required to put ID"
        size_hint_x:None
        width : 180



    MDRectangleFlatIconButton:
        widget_style:'desktop'
        icon : "camera"
        text : 'Capture  '
        tooltip_text : "Open the Camera"

        pos_hint:{'center_x':0.8 , 'center_y':0.3}
        on_release : app.click(self)

    MDRectangleFlatIconButton:
        icon : "arrow-left-thick"
        text : 'Back'
        pos_hint:{'center_x':0.6, 'center_y':0.3}
        on_release :
            root.manager.current = 'menu'
            root.manager.transition.direction = 'right'
    BoxLayout:
        orientation : 'vertical'


        MDToolbar:

            title:'Add a new Profile'
            left_action_items : [["human-male-girl"]]

        MDLabel:
            text : ''
            halign : 'center'
        Image :
            source : "adduser.png"
            pos_hint: {'center_x':0.3,'center_y':0.2}
            size_hint : (5,5)




<Gallery>
    name:'Gall'

    ScrollView:

        MDGridLayout:
            id:box
            cols:1
            adaptive_height : True
    MDFloatingActionButton:
        icon:'arrow-left-thick'
        on_press:root.manager.current = 'menu'















"""


class MenuScreen(Screen):
    pass
class Content(MDBoxLayout):
    pass



class ProfileScreen(Screen):
    pass
class Gallery(Screen):
    pass
class ListwithCheckbox(OneLineAvatarIconListItem):
    icon = StringProperty("android")

sm = ScreenManager()
sm.add_widget(MenuScreen(name  = 'menu'))
sm.add_widget(ProfileScreen(name  = 'Profile'))
sm.add_widget(Gallery(name = 'Gall'))


class DemoApp(MDApp):
    title = "Smart Attendance System using Face Recognition "
    #this is for opening the excel sheet


    def op(self):

        file_loc = "C:\Desktop\Face_Attend\Main_Att.csv"
        os.system('"%s"' % file_loc)

        # this for ScreenShot

    def click(self,obj):
        hello=self.root.get_screen('Profile').ids.naam.text

        roll = self.root.get_screen('Profile').ids.rollno.text



        cam = cv2.VideoCapture(0)
        # cv2.namedWindow("MHSSP Smart Attendance System")
        img_c = 0
        while True:

            ret, frame = cam.read()
            # cv2.imshow("Frame",frame)
            if not ret:
                print("Faolded")
            cv2.imshow("Registration Window", frame)

            key = cv2.waitKey(1)

            if key == ord("q"):
                break
            elif key % 256 == 32:
                img_res = str(hello) + "_" + str(roll) + ".jpg"
                cv2.resize(frame,(250,200))
                cv2.imwrite("./ImagesBasic/" + img_res, frame)
                self.dlog = MDDialog(title="Profile Save",text="You have been
registered as : "+ hello,size_hint=(0.7,1),buttons=[
                    MDFlatButton(text="Cancel",on_release=self.cl)
                ])
                self.soundplayer()
                self.dlog.open()
                self.on_start()


                img_c += 1
        cam.release()

    #This close fun for Dialog

    def cl(self,obj):
        self.dlog.dismiss()



    def soundplayer(self):
        username1 = self.root.get_screen('Profile').ids.naam.text
        username2 = self.root.get_screen('Profile').ids.rollno.text

        #audio = gTTS(" Your Profile  has been saved in our dataset    " +
username1,lang='en')
        #AudioSave = str(username1) + ".mp3"

        #audio.save(AudioSave)

        #playsound(AudioSave)
        #os.remove(AudioSave)
    def att(self,obj):

        path = 'ImagesBasic'
        images = []
        className = []
        myList = os.listdir(path)
        print(myList)

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

        print(className)


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

            return encodelist

        def markattend(name):
            with open('Attendance.csv', 'r+') as f:
                mydataList = f.readline()
                namelist = []


                for line in mydataList:
                    entry = line.split(',')
                    namelist.append(entry[0])
                if name not in namelist:
                    now = datetime.now()
                    dateString = now.strftime('%I:%M:%S')
                    taarik = now.strftime('%Y/%m/%d')
                    f.writelines(f'\n{name},{dateString},Present,{taarik}')
                    data = pd.read_csv("Attendance.csv")
                    data.drop_duplicates(subset="Name", keep="first",
inplace=True)
                    df = data
                    df.to_csv("Main_Att.csv")

        snackbar = Snackbar(
            text="Thank you! Attendance has been recorded in the Excel Sheet
",
            snackbar_x="10dp",
            bg_color=(0, 0, 1, 1),
            snackbar_y="20dp",
        )
        snackbar.size_hint_x = (
                                       Window.width - (snackbar.snackbar_x *
2)
                               ) / Window.width
        snackbar.buttons = [
            MDFlatButton(
                text="Close",
                text_color=(1, 1, 1, 1),
                on_release=snackbar.dismiss,
            ),

        ]

        snackbar.open()

        encodelistknown = findencodings(images)
        print("Encoding Completed!!!")


        #def Sounder(name):
           # date_string = datetime.now().strftime("%d%m%Y%H%M%S")

            #audio = gTTS(text=name,lang="en")
            #AudioSave = "voice" + date_string + ".mp3"
           # audio.save(AudioSave)

            #playsound(AudioSave)
            #os.remove(AudioSave)

        cap = cv2.VideoCapture(0)
        while True:
            success, img = cap.read()
            imgS = cv2.resize(img, (0, 0), None, 0.25, 0.25)
            imgS = cv2.cvtColor(imgS, cv2.COLOR_BGR2RGB)
            faceCur = face_recognition.face_locations(imgS)
            encodeCur = face_recognition.face_encodings(imgS, faceCur)

            for encodeFace, faceloc in zip(encodeCur, faceCur):
                matches = face_recognition.compare_faces(encodelistknown,
encodeFace)
                faceDis = face_recognition.face_distance(encodelistknown,
encodeFace)
                # print(faceDis)
                matchIndex = np.argmin(faceDis)

                if matches[matchIndex]:
                    name = className[matchIndex].upper()
                    # print(name)
                    y1, x2, y2, x1 = faceloc
                    y1, x2, y2, x1 = y1 * 4, x2 * 4, y2 * 4, x1 * 4
                    cv2.rectangle(img, (x1, y1), (x2, y2), (0, 255, 0), 2)
                    cv2.rectangle(img, (x1, y2 - 35), (x2, y2), (0, 255, 0),
cv2.FILLED)
                    cv2.putText(img, name, (x1 + 6, y2 - 6),
cv2.FONT_HERSHEY_COMPLEX, 1, (255, 255, 255), 2)
                    markattend(name)


                else:
                    name = className[matchIndex].upper()
                    # print(name)
                    unknown = "UNKNOWN"
                    y1, x2, y2, x1 = faceloc
                    y1, x2, y2, x1 = y1 * 4, x2 * 4, y2 * 4, x1 * 4
                    cv2.rectangle(img, (x1, y1), (x2, y2), (255,0,0), 2)
                    cv2.rectangle(img, (x1, y2 - 35), (x2, y2), (255,0,0),
cv2.FILLED)
                    cv2.putText(img, unknown, (x1 + 6, y2 - 6),
cv2.FONT_HERSHEY_COMPLEX, 1, (255, 255, 255), 2)

            cv2.imshow('Recognizer Camera', img)
            k = cv2.waitKey(1)
            if k == ord("q"):
                break




    # this is for main Screen
    def build(self):
        #scrn = Screen()
        # theme the color

        self.screen = Builder.load_string(newScreen)
        self.theme_cls.primary_palette = "Blue"
        self.theme_cls.theme_style = "Light"
        structure = len(os.listdir("ImagesBasic"))

        self.screen.get_screen('menu').ids.registered.text=f'Active
Registered Peoples : {structure}'




        # label

        #l1 = MDLabel(text="ML Based Smart Attendance System ",
halign="auto", theme_text_color="Custom",
                    # text_color=(0, 0, 1, 1), font_style="H5",
pos_hint={'center_x': 0.7, 'center_y': 0.9})
        #scrn.add_widget(l1)

        #self.Name = MDTextField(text="Enter the Name", pos_hint={'center_x':
0.3, 'center_y': 0.5}, size_hint_x=None,
                           #width=180)
        #scrn.add_widget(self.Name)
        #self.Id = MDTextField(text="Enter your Roll No ",
pos_hint={'center_x': 0.6, 'center_y': 0.5}, size_hint_x=None,
                  #       width=180)
        #scrn.add_widget(self.Id)

       # self.Attendance = MDRectangleFlatButton(text="Take Attendance",
pos_hint={'center_x': 0.3, 'center_y': 0.3}
                                                #,on_release=self.att)
        #scrn.add_widget(self.Attendance)
        #self.ScreenShot = MDRectangleFlatButton(text="Save My Profile",
pos_hint={'center_x': 0.7, 'center_y': 0.3}
                                                #,on_release=self.click)
        #scrn.add_widget(self.ScreenShot)
        return self.screen

    def dele(self, widget):

        pth = "ImagesBasic"
        for images in os.listdir(pth):
            joined = os.path.join(pth, images)

        self.screen.get_screen('Gall').ids.box.remove_widget(widget)

        print("Item Deleted Sucessfully")
        print(joined)
        os.remove(joined)
        self.on_start()
        structure = len(os.listdir("ImagesBasic"))
        self.screen.get_screen('menu').ids.registered.text = f'Active
Registered Peoples : {structure}'



        snackbar = Snackbar(
            text=f"Profile Deleted {images}  ",
            snackbar_x="20dp",
            bg_color=(0, 1, 0, 1),
            snackbar_y="20dp",
        )
        snackbar.size_hint_x = (
                                       Window.width - (snackbar.snackbar_x *
2)
                               ) / Window.width
        snackbar.buttons = [
            MDFlatButton(
                text="OK",
                text_color=(1, 1, 1, 1),
                on_release=snackbar.dismiss,
            ),

        ]

        snackbar.open()

    def Attendance_Visualize(self):
        df = pd.read_csv("Main_Att.csv")
        df_count = df['Name'].count()
        List_item = len(os.listdir("C:\\Desktop\\Face_Attend\\ImagesBasic"))
        plt.style.use("fivethirtyeight")
        slices = [List_item, df_count]
        labels = ["Absent", "Present"]
        plt.pie(slices, labels=labels, wedgeprops={"edgecolor": "black"},
shadow=True, autopct='%1.1f%%')
        plt.title(f"Todays Strength is {slices[1]}")
        plt.tight_layout()
        plt.show()
    def on_start(self):
        pth = "ImagesBasic"
        mobile = self.root.get_screen('Profile').ids.rollno.text
        print(mobile)

        for images in os.listdir(pth):
            joined = os.path.join(pth, images)
            text = images

            self.screen.get_screen('Gall').ids.box.add_widget(
                ListwithCheckbox(text=f"{text}",icon=f"{joined}")
            )


        structure = len(os.listdir("ImagesBasic"))
        self.screen.get_screen('menu').ids.registered.text = f'Active
Registered Peoples : {structure}'

if __name__ == '__main__':
    DemoApp().run()
```

6.4 System Testing

Introduction

Testing is the process of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not. Testing is executing a system in order to identify any gaps, errors, or missing requirements in contrary to the actual requirements. This tutorial will give you a basic understanding on software testing, its types, methods, levels, and other related terminologies.

Verification:

Verification is the process to make sure the product satisfies the conditions imposed at the start of the development phase. In other words, to make sure the product behaves the way we want it to.

Validation:

Validation is the process to make sure the product satisfies the specified requirements at the end of the development phase. In other words, to make sure the product is built as per

customer requirements.

Testing is done in different forms at every phase of SDLC:

- During the requirement gathering phase, the analysis and verification of requirements are also considered as testing.
- Reviewing the design in the design phase with the intent to improve the design is also considered as testing.
- Testing performed by a developer on completion of the code is also categorized as testing.

Testing Approaches

A test approach is the test strategy implementation of a project, defines how testing would be carried out. Test approach has two techniques:

Proactive:

An approach in which the test design process is initiated as early as possible in order to find and fix the defects before the build is created.

Reactive:

An approach in which the testing is not started until after design and coding are completed.

Factors to be considered:

- Risks of product or risk of failure or the environment and the company.
- Expertise and experience of the people in the proposed tools and techniques.
- Regulatory and legal aspects, such as external and internal regulations of the development process.
- The nature of the product and the domain.