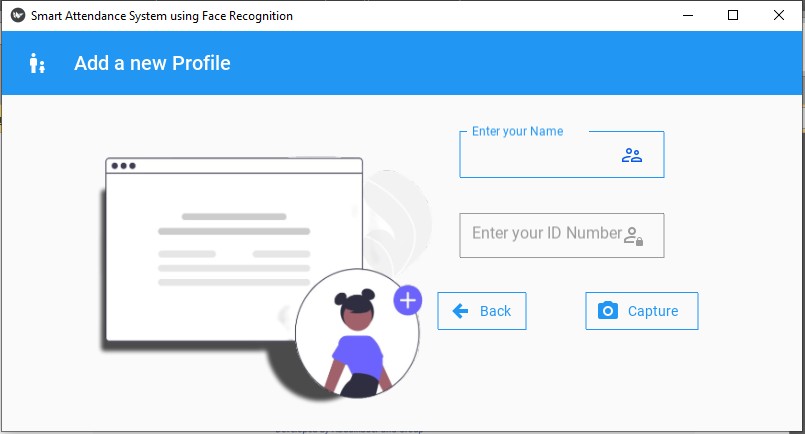
CHAPTER 6

# 6.1Snapshots

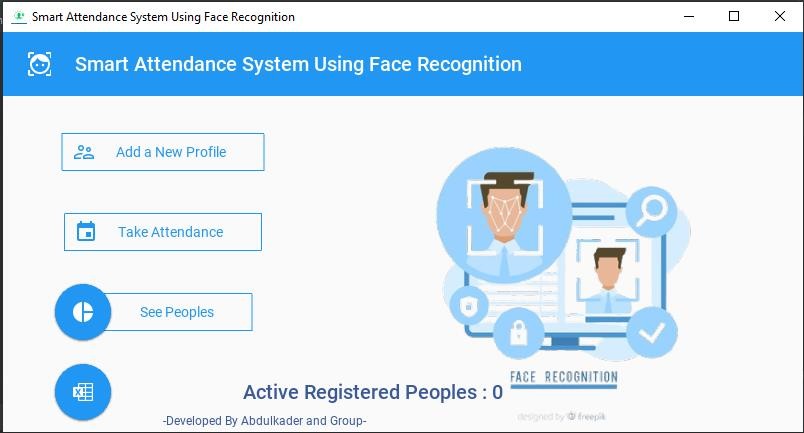
**Home page:**



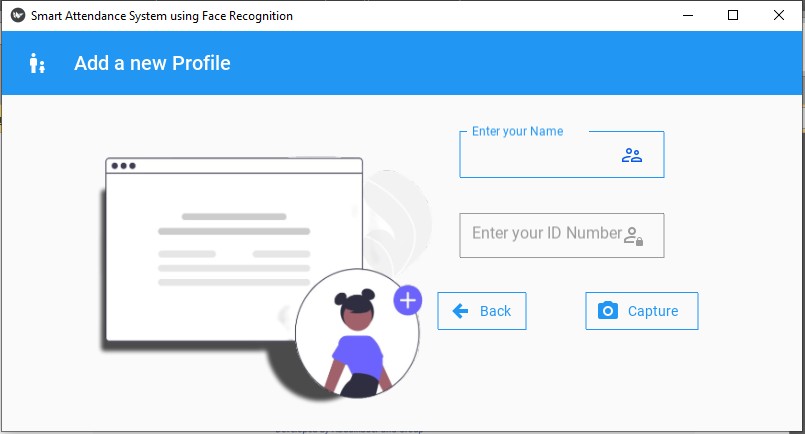
# Registration page:



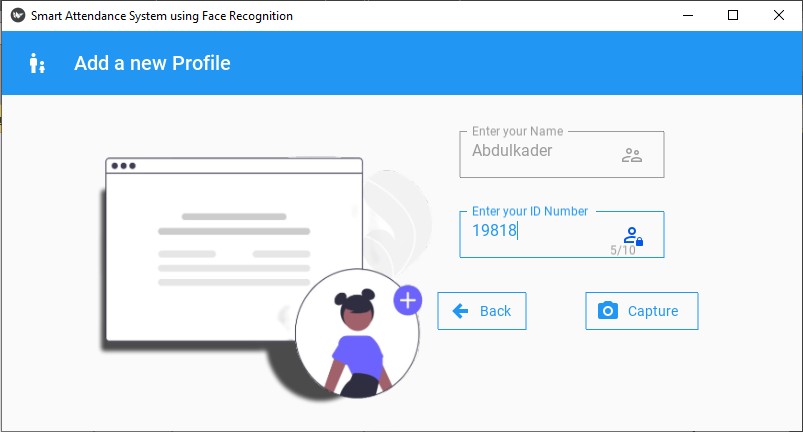
## Home Screen:



Registerattion screen:



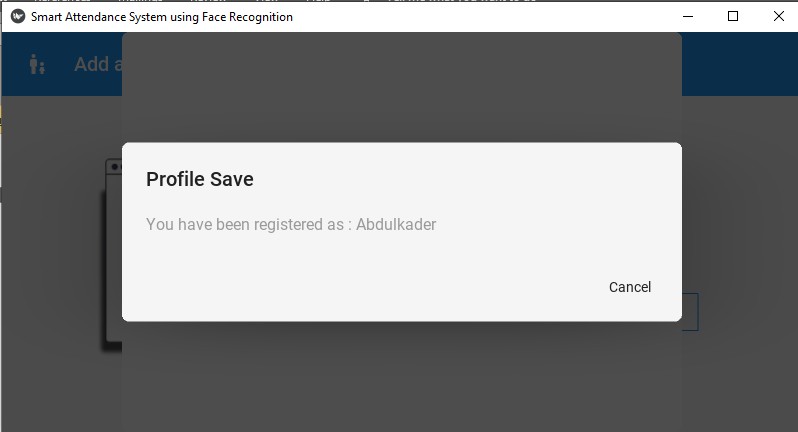
Registeration screen after inputing some values:



After clicking the capture button camera window will open



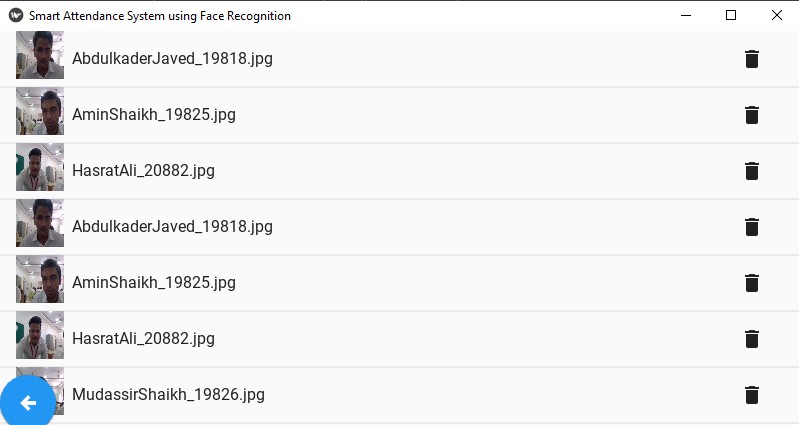
Profile saving popup:



Home Screen after saving 4 profiles



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



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

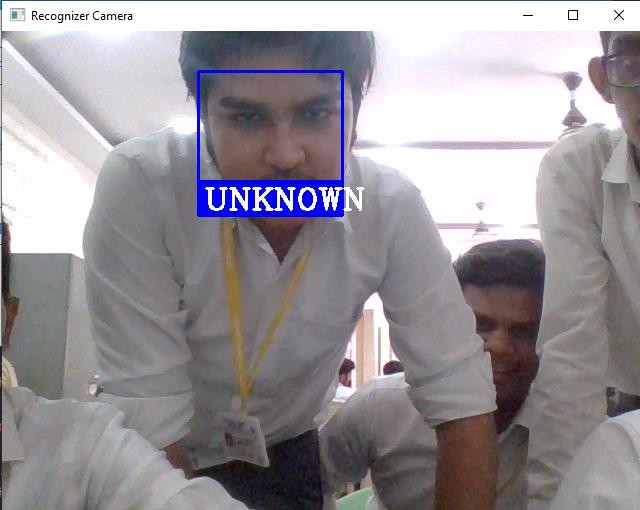


After clicking on the Take Attendance Button Recognizer Camera will Start

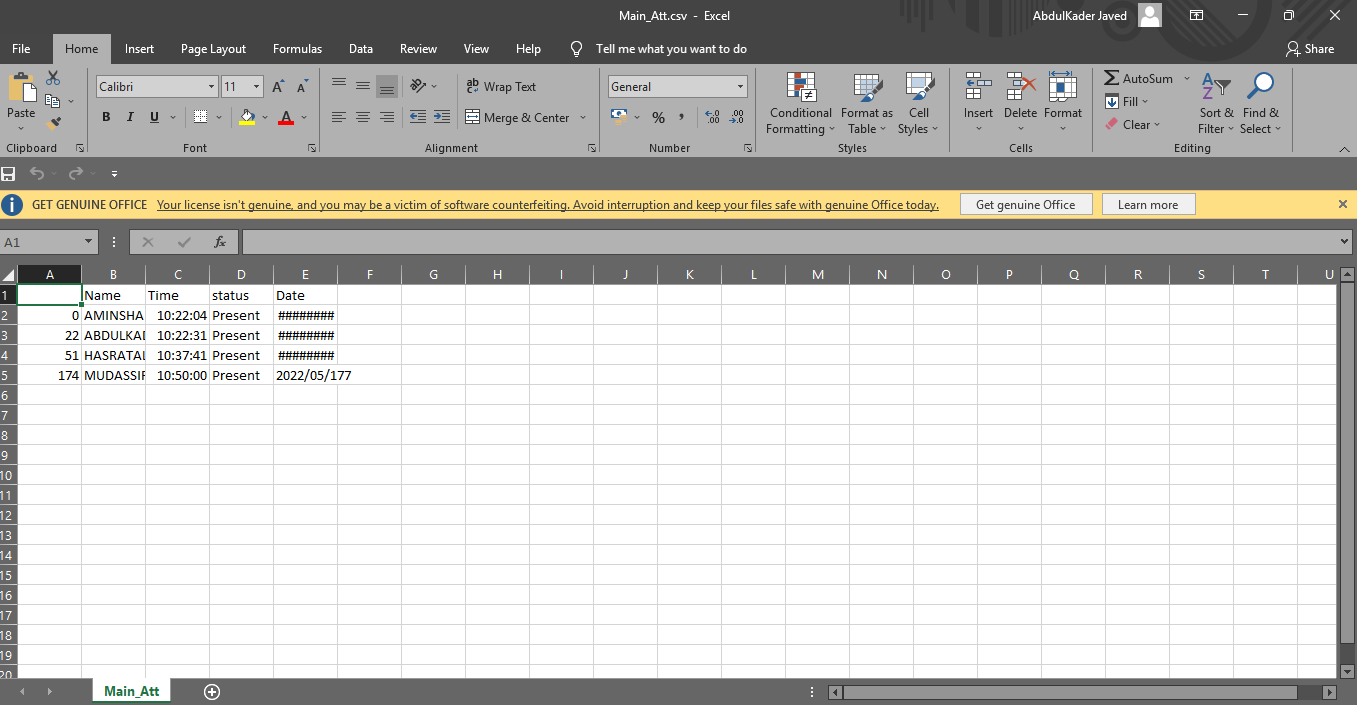




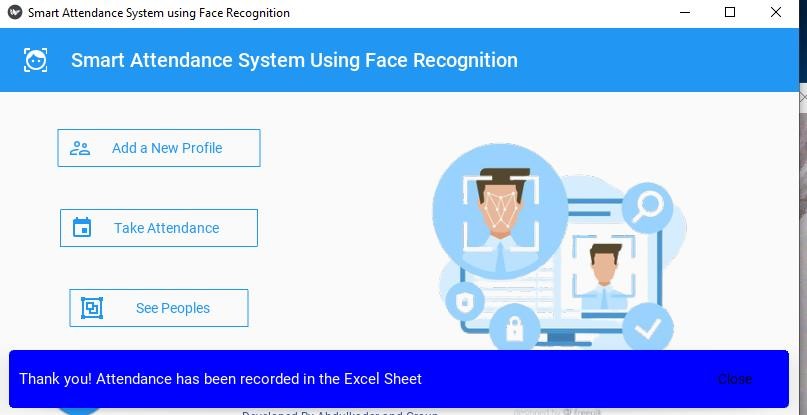
## If software doesn’t recognises the face then it will show UNKNOWN and will not mark the attendance





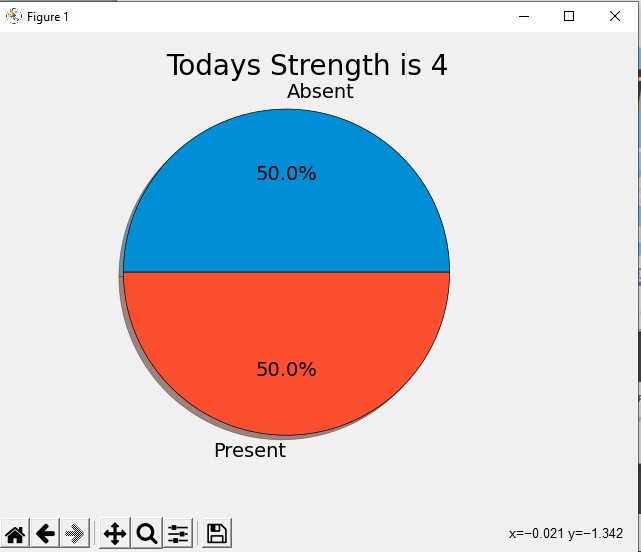


HOME SCREEN WITH SNACK BAR

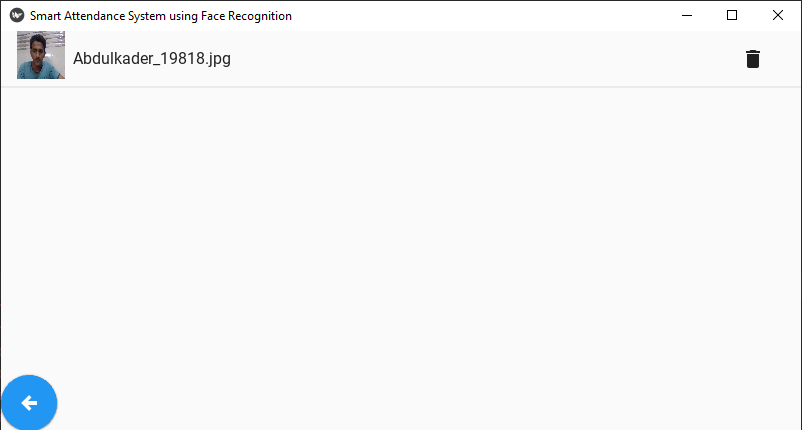


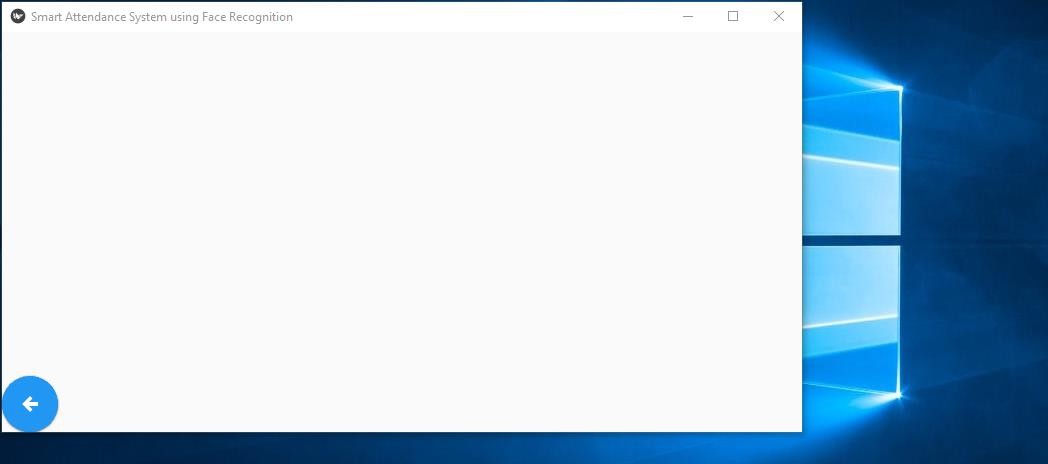


As we click on the pie button it will start the statistics(pie chart)



# Deleting Profile:







* 1. **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 = []

inplace=True)

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",

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 = (

2)

Window.width - (snackbar.snackbar\_x \*

) / 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)

encodeFace) encodeFace)

for encodeFace, faceloc in zip(encodeCur, faceCur):

matches = face\_recognition.compare\_faces(encodelistknown, faceDis = face\_recognition.face\_distance(encodelistknown,

# print(faceDis)

matchIndex = np.argmin(faceDis)

cv2.FILLED)

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.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"

cv2.FILLED)

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.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 = (

2)

Window.width - (snackbar.snackbar\_x \*

snackbar.buttons = [ MDFlatButton(

text="OK",

) / Window.width

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()

# 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.