**Title/Problem Statement: Face Attendance System**

**RollNo: MCA033**

**Name: MUDALIYAR JAYASUDHA TAMILVANAN**

**RollNo: MCA036**

**Name: JIGNESH BHAVANBHAI NAKUM**

**Title/Problem description/explanation:**

* Face Attendance System is basically created to make the attendance of people easier by just scanning their face. Only once we have to create an account after that we just need to login using our system by their registered face to make their attendance.

**Expected outcomes:**

1. For the first time we have to register our face in the system along with the name.
2. It may store the data of the user in our system so after that when user makes his/her attendance then he/she may have to just login in the system with their face.
3. If the face is detected and it is already saved in the system it will display "Welcome username" kind of message. Otherwise it won't allow the user to login in the system.
4. Each detected face is stored in the system along with the date and time in it for attendance purpose.

**Tools:**

1. User Interface Tools like **Tkinter** for Graphical user Interface.
2. **OpenCV (Open Source Computer Vision Library)** or **CV2** for face detection and face recognition.
3. **Dlib** for image processing and facial recognition.
4. **PIL (Python Imaging Library) ImageTk** supports different image file formats.
5. Hardware Requirement such as **webcams** for capturing images.

* **Code:**
* util.py:

import tkinter as tk

from tkinter import messagebox

def get\_button(window, text, color, command, fg='white'):

button = tk.Button(window,

text=text,

activebackground='black',

activeforeground='white',

fg=fg,

bg=color,

command=command,

height=2,

width=20,

font=('Helvetica bold',20)

)

return button

def get\_img\_label(window):

label=tk.Label(window)

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

return label

def get\_text\_label(window,text):

label=tk.Label(window,text=text)

label.config(font=("sans-serif",21),justify="left")

return label

def get\_entry\_text(window):

inputtxt=tk.Text(window,

height=2,

width=15,

font=("Arial",32))

return inputtxt

def msg\_box(title,description):

messagebox.showinfo(title,description)

* main.py:

import tkinter as tk

import datetime

import subprocess

import os.path

import cv2

from PIL import Image, ImageTk

import util

class App:

    def \_\_init\_\_(self):

        self.main\_window=tk.Tk()

        self.main\_window.geometry("1200x520+370+120")

        self.login\_button\_main\_window = util.get\_button(self.main\_window, 'login', 'green', self.login)

        self.login\_button\_main\_window.place(x=750,y=300)

        self.register\_new\_user\_button\_main\_window = util.get\_button(self.main\_window, 'register new user',

        'gray', self.register\_new\_user, fg='black')

        self.register\_new\_user\_button\_main\_window.place(x=

750,y=400)

        self.webcam\_label = util.get\_img\_label(self.main\_window)

        self.webcam\_label.place(x=10, y=0, width=700, height=500)

        self.add\_webcam(self.webcam\_label)

self.db\_dir = 'E:\MCA\_1ST\_YEAR\SEM2\PYTHON\Termwork\db'

        if not os.path.exists(self.db\_dir):

            os.mkdir(self.db\_dir)

self.log\_path = 'E:\MCA\_1ST\_YEAR\SEM2\PYTHON\Termwork\

Attendance.xls'

    def add\_webcam(self,label):

        if 'cap' not in self.\_\_dict\_\_:

            self.cap = cv2.VideoCapture(0)

        self.\_label = label

        self.process\_webcam()

def process\_webcam(self):

        ret, frame = self.cap.read()

        self.most\_recent\_capture\_arr = frame

        img\_ = cv2.cvtColor(self.most\_recent\_capture\_arr, cv2.COLOR\_BGR2RGB)

        self.most\_recent\_capture\_arr\_pil = Image.fromarray(img\_)

        imgtk = ImageTk.PhotoImage(image =self.most\_recent\_capture\_arr\_pil)

        self.\_label.imgtk = imgtk

        self.\_label.configure(image=imgtk)

        self.\_label.after(20, self.process\_webcam)

    def login(self):

        unknown\_img\_path = './.tmp.jpg'

        cv2.imwrite(unknown\_img\_path, self.most\_recent\_capture\_arr)

        output= str(subprocess.check\_output(['face\_recognition', self.db\_dir, unknown\_img\_path]))

        name = output.split(',')[1][:-5]

if name in ['unknown\_person','no\_persons\_found']:

          util.msg\_box('Oops...','Unknown User. Please register new user or try again.')

else:

            util.msg\_box('Welcome Back!','Welcome, {}.'.format(name))

            with open(self.log\_path, 'a') as f:

              f.write('{},{}\n'.format(name,datetime.datetime.now()))

            f.close()

        os.remove(unknown\_img\_path)

    def register\_new\_user(self):

        self.register\_new\_user\_window = tk.Toplevel(self.main\_window)

        self.register\_new\_user\_window.geometry("1200x520+370+

120")

        self.accept\_button\_register\_new\_user\_window = util.get\_button(self.register\_new\_user\_window, 'Accept', 'green', self.accept\_register\_new\_user)

        self.accept\_button\_register\_new\_user\_window.place(x=750,y =300)

        self.try\_again\_button\_register\_new\_user\_window = util.get\_button(self.register\_new\_user\_window, 'Try Again', 'red', self.try\_again\_register\_new\_user)

        self.try\_again\_button\_register\_new\_user\_window.place(

x=750,y=400)

        self.capture\_label = util.get\_img\_label(self.register\_new\_user\_window)

        self.capture\_label.place(x=10, y=0, width=700, height=500)

        self.add\_img\_to\_label(self.capture\_label)

        self.entry\_text\_register\_new\_user = util.get\_entry\_text(self.register\_new\_user\_window)

        self.entry\_text\_register\_new\_user.place(x=750, y=150)

        self.text\_label\_register\_new\_user = util.get\_text\_label(self.register\_new\_user\_window,"Please,\n Input username:")

        self.text\_label\_register\_new\_user.place(x=750, y=70)

    def try\_again\_register\_new\_user(self):

        self.register\_new\_user\_window.destroy()

    def add\_img\_to\_label(self,label):

        imgtk = ImageTk.PhotoImage(image =self.most\_recent\_capture\_arr\_pil)

        label.imgtk = imgtk

        label.configure(image=imgtk)

        self.register\_new\_user\_capture = self.most\_recent\_capture\_arr.copy()

    def start(self):

        self.main\_window.mainloop()

    def accept\_register\_new\_user(self):

        name = self.entry\_text\_register\_new\_user.get(1.0, "end-1c")

        cv2.imwrite(os.path.join(self.db\_dir, '{}.jpg'.format(name)), self.register\_new\_user\_capture)

        util.msg\_box('Success!','User was registered Successfully!!!')

        self.register\_new\_user\_window.destroy()

if \_\_name\_\_ == "\_\_main\_\_":

    app=App()

    app.start()

* **Output:**















