

The Code For Our Activity Tracker Project

```
import os
import time
import tkinter as tk
from tkinter import messagebox
from datetime import datetime
from tkinter import Menu, ttk, filedialog
import string
from cv2 import cv2
import face_recognition
import numpy as np
import shutil
import smtplib as smtp
import random

window1 = tk.Tk()
window1.geometry("1280x720")
window1.minsize(1080,720)
window1.title("Employee Activity Tracker")
window1.configure(background = 'gray')

##### Registration Window #####
#####
class Registration:
    def __init__(self):
        self.img_new = []
        self.browse_pic_label = False

    def Take_Picture(self):
        path = 'images'
        mylist = os.listdir(path)
        if len(mylist)>0: #####
### Checking path folder have already an image or not #####
#####
```

```

        self.button1_label.config(text = "* your are already register!!!!")
        return 0
    self.entry3_label.config(text = "")
    cap = cv2.VideoCapture(0)
    if not cap.isOpened():
        raise Exception(" Could not open camera ")
    while True:
        success, img = cap.read()
        cv2.imshow('WebCam',img)
        key = cv2.waitKey(1)
        ##### Validating NAME #####
        NAME = self.entry2.get()
        if len(NAME) == 0 :
            self.entry2_label.config(text = "* Please Your Name!!!!!!")
            cap.release()
            cv2.destroyAllWindows()
            break
        else:
            self.entry2_label.config(text = "")
            if key == ord('s') or key == ord('S') :
                self.img_new = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
                cv2.imshow(' Showing Clicked Picture ',self.img_new)
                cv2.waitKey(2000)
                cap.release()
                cv2.destroyAllWindows()
                self.button1_label.config(text = ".....Picture Taken....",fg = 'green')
                if self.browse_pic_label:
                    self.browse_pic_label = False
                    self.var1.set("")
                    self.entry3_label.config(text = "* browsed Picture is clear!!!!",fg = 'blue')
                    break

```

```

        elif key == ord('q') or key == ord('Q'):
            cap.release()
            cv2.destroyAllWindows()
            break

    def Save_Profile(self):
        ##### Validating ID #####
        ID = self.entry1.get()
        if len(ID) == 0 :
            self.entry1_label.config(text = "* Please Your Employee ID!!!!!!")
        else:
            self.entry1_label.config(text = "")
            ##### Validating NAME #####
            NAME = self.entry2.get()
            if len(NAME) == 0 :
                self.entry2_label.config(text = "* Please Your Name!!!!!!")
            else:
                self.entry2_label.config(text = "")
                ##### Validating DURATION #####
                DURATION = self.var.get()
                if DURATION == "HH:MM" :
                    self.drop1_label.config(text = "* Please select duration!!!!!!")
                else:
                    self.drop1_label.config(text = "")
                    ##### Validating Browse Picture and Take picture #####
                    file_loc = self.entry3.get()
                    if (not os.path.isfile(file_loc)) and (len(self.image_new) == 0):
                        self.entry3_label.config(text = "* Please select file or Take Picture")
                        self.button1_label.config(text = "* Please select file or Take Picture ")

```

```

else :
    ##### Saving Picture #####
    #
    self.pic_save = False
    if os.path.isfile(file_loc):
        ##### Checking file_loc exist or not #####
        path = 'images'
        mylist = os.listdir(path)
        self.button1_label.config(text = "")
        if len(mylist)>0:
            ##### Checking path folder have already an image or not #####
            self.entry3_label.config(text = "* your pic is already register!!!")
            elif len(self.img_new) > 0:
                self.entry3_label.config(text = "* Either you can Browse pic or Take Picture.",fg = 'blue')
                self.img_new = []
                self.button1_label.config(text = "* Take n picture clear. if you want to register you taken picture, Please Click on <<Take Picture>>",fg = 'blue')
                self.browse_pic_label = True
            else:
                if messagebox.askquestion("Comfirm","Are You Sure?") == 'yes':
                    with open('Details.txt','w') as f:
                        detail =ID + "\n" + NAME + "\n"
+ DURATION
                        f.writelines(detail)
                        shutil.copy(file_loc, path)
                    ##### Copying photo to file_loc to path #####
                    self.entry3_label.config(text = "* your pic is register.....",fg = 'green')
                    messagebox.showinfo("Done","Successfully Register \n Now please go to Home page..")

```

```

        else:
            return 0
    else:
        self.entry3_label.config(text = "")

        if messagebox.askquestion("Comfirm","Are You
Sure?") == 'yes':
            with open('Details.txt','w') as f:
                detail =ID + "\n" + NAME + "\n" + DU
RATION

                f.writelines(detail)
                path = 'images\\'
                image_name = path + NAME + ".jpg"
                self.pic_save = cv2.imwrite(filename = i
mage_name, img = self.img_new)
                messagebox.showinfo("Done","Successfully
Register \n Now please go to Home page..")

            else:
                return 0

def register_new_employee(self,window1 = window1):
    window1.destroy()
    def Go_to_home_page():
        window1 = tk.Tk()
        window1.geometry("1280x720")
        window1.minsize(1080,720)
        window1.title("Employee Activity Tracker")
        window1.configure(background = 'gray')

        home_page(window1)
        self.registration_window.destroy()

    self.registration_window = tk.Tk()
    self.registration_window.geometry("1280x720")
    self.registration_window.minsize(1280,720)
    self.registration_window.title("Employee Activity Tr
acker")
    self.registration_window.iconbitmap("icon.ico")

```

```

        self.registration_window.configure(background = 'gray')

        frame1 = tk.Frame(self.registration_window, bg = 'gray')
        frame1.place(relx = 0.11, rely = 0, relwidth = 0.80,
relheight = 0.118)

        l1 = tk.Label(frame1, text = "Face Recognition Based
Employee Activity Tracker", bg = 'gray', fg = "black", font =
('verdana', 20, 'bold'))
        l1.pack(anchor = 'center', pady = 4 )
        l2 = tk.Label(frame1, text = "New Registration", bg =
'gray', fg = "green", font = ('verdana', 16, 'bold'))
        l2.pack(anchor = 'center', pady = 4 , side = 'bottom')
#####
Frame2 #####
        frame2 = tk.Frame(self.registration_window, bg= 'gray'
')
        frame2.place(relx = 0.2, rely = 0.12, relwidth = 0.75
, relheight = 0.5)

        l3 = tk.Label(frame2, text = "Employee ID", bg = 'gray'
, fg = "white", font = ('verdana', 16, 'bold'))
        l3.grid(pady=20, padx=30, row=0, column=0, sticky='w')
        self.entry1 = tk.Entry(frame2, bd = 4, font = ('verdan
a', 12, 'bold'))
        self.entry1.grid(pady=20, padx=20, row=0, column=1, stic
ky='w')
        self.entry1_label = tk.Label(frame2, text = "", bg = '
gray', fg = "red", font = ('verdana', 8, 'bold'))
        self.entry1_label.grid(pady=20, padx=30, row=0, column=
3, sticky='w')

        l4 = tk.Label(frame2, text = "Employee Name", bg = 'gr
ay', fg = "white", font = ('verdana', 16, 'bold'))
        l4.grid(pady=20, padx=30, row=1, column=0, sticky='w')
        self.entry2 = tk.Entry(frame2, bd = 4, font = ('verdan
a', 12, 'bold'))

```

```

        self.entry2.grid(pady=20,padx=20,row=1,column=1,sticky='w')
        self.entry2_label = tk.Label(frame2,text = "",bg = 'gray', fg = "red",font = ('verdana',8,'bold'))
        self.entry2_label.grid(pady=20,padx=30,row=1,column=3,sticky='w')

        l4 = tk.Label(frame2,text = "Set Working Duration",bg = 'gray', fg = "white",font = ('verdana',16,'bold'))
        l4.grid(pady=20,padx=30,row=2,column=0,sticky='w')

        self.var = tk.StringVar()
        self.var.set("HH:MM")
        self.drop1 = tk.OptionMenu(frame2,self.var,"01:00","02:00","03:00","04:00","05:00","06:00","07:00","08:00","09:00","10:00","11:00","12:00")
        self.drop1.grid(pady=20,padx=20,row=2,column=1,sticky='w')
        self.drop1_label = tk.Label(frame2,text = "",bg = 'gray', fg = "red",font = ('verdana',8,'bold'))
        self.drop1_label.grid(pady=20,padx=30,row=2,column=3,sticky='w')

        ##### Browsing Picture from file location #####

        self.var1 = tk.StringVar()
        self.var1.set("Select file location")
        self.entry3 = tk.Entry(frame2,bd = 4, font = ('verdana',12),textvariable = self.var1)
        self.entry3.grid(pady=20,padx=20,row=3,column=1,sticky='w')

        def Browse_pic():
            self.registration_window.filedialog1 = filedialog.askopenfilename(initialdir = "Desktop",title = "Select your photo",filetypes = (("jpg file","*.jpg"),("jpeg file","*.jpeg"),("png file","*.png")))

```

```

        self.var1.set(self.registration_window.filedialo
g1)

        button2 = tk.Button(frame2,text = 'Browse Picture',
bd=2,bg = 'gray', fg = "black" ,command = Browse_pic,font =
('verdana',12,'bold'))
        button2.grid(pady=20,padx=20,row=3,column=0,sticky='
w')

        self.entry3_label = tk.Label(frame2,text = "",bg = '
gray', fg = "red",font = ('verdana',8,'bold'))
        self.entry3_label.grid(pady=20,padx=30,row=3,column=
3,sticky='w')
        #####
Frame3 #####
        self.frame3 = tk.Frame(self.registration_window, bg
= 'gray')
        self.frame3.place(relx = 0.15,rely = 0.5, relwidth =
0.8, relheight = 0.34)
        l5 = tk.Label(self.frame3, text = 'OR',font = ('verd
ana',12,'bold'),bg = '#262523',fg = 'white')
        l5.pack(anchor = 'center',pady =5 )

        button1 = tk.Button(self.frame3,text = 'Take Picture
', bd=4, command = self.Take_Picture,font = ('verdana',12,'b
old'),bg = '#262523',fg = 'white',height = 2, width = 35)
        button1.pack(anchor = 'center')
        self.button1_label = tk.Label(self.frame3,text = "Pr
ess 's' to take picture and 'q' to close WebCam",bg = 'gray'
, fg = "blue",font = ('verdana',8,'bold'))
        self.button1_label.pack(anchor = 'center',pady =10)

        button3 = tk.Button(self.frame3,text = 'Save Profile
', bd=4, command = self.Save_Profile,font = ('verdana',12,'b
old'),bg = '#262523',fg = 'white',height = 2, width = 35)
        button3.pack(anchor = 'center',pady =5)
        self.button3_label = tk.Label(self.frame3,text = "",
bg = 'gray', fg = "red",font = ('verdana',8,'bold'))
        self.button3_label.pack(anchor = 'center')

```



```

        frame4 = tk.Frame(self.registration_window,bg= 'gray
    ')
        frame4.place(relx = 0.11,rely = 0.9, relwidth = 0.80
, relheight = 0.14)
        button1 = tk.Button(frame4,text = 'Quit', bd=4, comm
and = self.registration_window.destroy,font = ('verdana',12,
'bold'),bg = '#262523',fg = 'white',height = 1, width = 10)
        button1.grid(pady=20,padx=20,row=0,column=1,sticky='
e')
        button2 = tk.Button(frame4,text = 'Go to home page',
        bd=4, command = Go_to_home_page,font = ('verdana',12,'bold'
),bg = '#262523',fg = 'white',height = 1, width = 15)
        button2.grid(pady=20,padx=20,row=0,column=0,sticky='
w')

        self.registration_window.mainloop()
##### End o
f Registration Window #####
#####

##### About
Window #####
#####

def About(window1, about_window):
    window1.destroy()
    def Go_to_home_page():
        window1 = tk.Tk()
        window1.geometry("1280x720")
        window1.minsize(1080,720)
        window1.title("Employee Activity Tracker")

        window1.configure(background = 'gray')
        about_window.destroy()
        home_page(window1)

##### Fra
me1 #####
        frame1 = tk.Frame(about_window, bg = 'gray')

```

```

    frame1.place(relx = 0.11, rely = 0, relwidth = 0.80, relheight = 0.118)

    l1 = tk.Label(frame1, text = "Face Recognition Based Employee Activity Tracker", bg = 'gray', fg = "black", font = ('verdana', 20, 'bold'))
    l1.pack(anchor = 'center', pady = 4 )

    ##### Fra
me2 #####
    frame2 = tk.Frame(about_window, bg= 'gray')
    frame2.place(relx = 0.11, rely = 0.12, relwidth = 0.80, relheight = 0.75)

    l2 = tk.Label(frame2, text = '''This software aims to provide a realtime continuous attendance system using the concept of machine Learning.\n
    The software is designed and built by Prince , Tirthoraj , Shubham and Binay of BCA final year session 2018-2021. \n\n
    The software notes the user's presence continuously after certain interval of time making it an efficient way to\n
    ensure Employees availability through out the working hours and hence ensures organisation's productivity.\n\n
    ''', bg = 'gray', fg = "black", font = ('verdana', 12))
    l2.pack(anchor = 'center', pady = 15 , side = 'top')
    tk.Label(frame2, text = '''Contact us :-
    ''', bg = 'gray', fg = "black", font = ('verdana', 12)).pack(anchor = 'w', pady = 15, padx = 50)
    tk.Label(frame2, text = '''•Prince :: pk03215@gmail.com : 9504008839''', bg = 'gray', fg = "black", font = ('verdana', 12)).pack(anchor = 'w', pady = 15, padx = 150)
    tk.Label(frame2, text = '''•Tirthoraj :: tirthorajdasgupta@gmail.com :: 8434116014''', bg = 'gray', fg = "black", font = ('verdana', 12)).pack(anchor = 'w', pady = 15, padx = 150)

```

```

        tk.Label(frame2,text = '''•Shubham :: shubhamdutta5694@gmail.com:: 6202561126''',bg = 'gray', fg = "black",font = ('verdana',12)).pack(anchor = 'w',pady = 15,padx = 150)
        tk.Label(frame2,text = '''•Binay :: binaypurty22@gmail.com:: 6204998628''',bg = 'gray', fg = "black",font = ('verdana',12)).pack(anchor = 'w',pady = 15,padx = 150)

        frame3 = tk.Frame(about_window,bg= 'gray')
        frame3.place(relx = 0.11,rely = 0.85, relwidth = 0.80, relheight = 0.13)

        button1 = tk.Button(frame3,text = 'Quit', bd=4, command = about_window.destroy,font = ('verdana',12,'bold'),bg = '#262523',fg = 'white',height = 1, width = 10)
        button1.grid(pady=20,padx=20,row=0,column=1,sticky='e')
        button2 = tk.Button(frame3,text = 'Go to home page', bd=4, command = Go_to_home_page,font = ('verdana',12,'bold'),bg = '#262523',fg = 'white',height = 1, width = 15)
        button2.grid(pady=20,padx=20,row=0,column=0,sticky='w')
        about_window.mainloop()
##### End of About Window #####
#####

##### Help Index Window #####

def help_index(window1, help_index_window):
    window1.destroy()
    def Go_to_home_page():
        window1 = tk.Tk()
        window1.geometry("1280x720")
        window1.minsize(1080,720)
        window1.title("Employee Activity Tracker")
        window1.configure(background = 'gray')
        help_index_window.destroy()
        home_page(window1)

##### Heading Frame #####

```

```

frame1 = tk.Frame(help_index_window, bg = 'gray')
frame1.place(relx = 0.11, rely = 0, relwidth = 0.80, relheight = 0.15)

l1 = tk.Label(frame1, text = "Face Recognition Based Employee Activity Tracker", bg = 'gray', fg = "black", font = ('verdana', 20, 'bold'))
l1.pack(anchor = 'center', pady = 4)
l2 = tk.Label(frame1, text = ''' If you need any help feel free to contact us. Let us know about problem that you are facing.''' , bg = 'gray', fg = "white", font = ('verdana', 12))
l2.pack(anchor = 'center', pady = 15, side = 'top')
##### Bod
y Frame #####
#####
frame2 = tk.Frame(help_index_window, bg= 'black')
frame2.place(relx = 0.11, rely = 0.151, relwidth = 0.80, relheight = 0.74)

l1 = tk.Label(frame2, text = "Send To :", bg = "black", fg = "white", font = ('verdana', 16, 'bold'),)
l1.grid(pady=10, padx=30, row=0, column=0, sticky='w')
frame3 = tk.Frame(help_index_window, bg="black")
frame3.place(relx = 0.37, rely = 0.17, relwidth = 0.45, relheight = 0.08)
email_id = tk.Text(frame3, width = 100, height = 1, bd = 4, bg = "black", fg = "white", font = ('verdana', 14, "bold"))
email_id.pack()
email_id_label = tk.Label(frame3, text = "You can send email to more than one by seperating comma(,)", fg = "green", bg = "black", font = ('verdana', 10))
email_id_label.pack(anchor = "center")

l2 = tk.Label(frame2, text = "Subject :", bg = "black", fg = "white", font = ('verdana', 16, 'bold'))
l2.grid(pady=40, padx=30, row=1, column=0, sticky='w')
frame4 = tk.Frame(help_index_window, bg="black")
frame4.place(relx = 0.37, rely = 0.27, relwidth = 0.45, relheight = 0.08)

```

```

    Subject_text = tk.Text(frame4,width = 100,height = 1,bd
= 4, bg = "black",fg = "white",font = ('verdana',14,"bold"))
    Subject_text.pack()
    Subject_text_label = tk.Label(frame4,text = "",fg = "green",bg = "black",font = ('verdana',10))
    Subject_text_label.pack(anchor = "center")

    l3 = tk.Label(frame2,text = "Compose Message :",bg = "black", fg = "white",font = ('verdana',16,'bold'))
    l3.grid(pady=30,padx=30,row=2,column=0,sticky='w')
    frame5 = tk.Frame(help_index_window,bg="black")
    frame5.place(relx = 0.37,rely = 0.38,relwidth = 0.45,relheight = 0.32)
    email_body = tk.Text(frame5,width = 100,height = 25,bd = 6, bg = "black",fg = "white",font = ('verdana',10))
    email_body.pack()

    tk.Label(frame2,text = "",bg = "black", fg = "white",font = ('verdana',16,'bold')).grid(pady=50,padx=30,row=3,column=0,sticky='w')
    tk.Label(frame2,text = "",bg = "black", fg = "white",font = ('verdana',16,'bold')).grid(pady=20,padx=30,row=4,column=0,sticky='w')

    def send_mail():
        id_to_send = email_id.get(1.0, "end")
        email_id_to_send_mail = id_to_send.split(",")
        sub = Subject_text.get(1.0,"end")
        body = email_body.get(1.0, "end")
        for id in email_id_to_send_mail:
            if len(id) < 10 :
                email_id_label.configure(text = "Please enter correct Email id !!!!!!!!!!!!!",fg = "red")
                break
            else:
                email_id_label.configure(text = "",fg = "red")

        if len(sub) < 2 :

```

```

        Subject_text_label.configure(text = "Please fill
subject field !!!!!!!!!!! ",fg="red")
    else:
        Subject_text_label.configure(text = "")
    if len(body) < 2 :
        body_label.configure(text = "Please fill all fie
ld !!!!!!!!!!! ")
    else:
        body_label.configure(text = "")
    try:
        with open("email_details.txt","r") as f:
            id_and_pwd_of_emp= f.readlines()
            emp_id = id_and_pwd_of_emp[0].replace("\n","")
            emp_pwd = id_and_pwd_of_emp[1]
            ob = smtp.SMTP("smtp.gmail.com",587)
            ob.starttls()#this initiat tls encryption
            ob.login(emp_id,emp_pwd)
            message = "Subject :{}\n\n{}".format(sub,body)
            ob.sendmail(id,[email_id_to_send_mail],message)
            messagebox.showinfo("Done","Mail send successful
ly....")
            ob.quit()
            email_id.delete(1.0,"end")
            Subject_text.delete(1.0,"end")
            email_body.delete(1.0,"end")
        except Exception as e:
            messagebox.showinfo("Error in connection",e)

    send_button = tk.Button(frame2,text = "SEND",bg = "black
",command = send_mail, fg = "white",font = ('verdana',16,'bo
ld'),width= 8)
    send_button.grid(pady=70,padx=20,row=5,column=0,sticky='
w')
    body_label = tk.Label(frame2,text = "",bg = "black", fg
= "red",font = ('verdana',10))
    body_label.grid(pady=30,padx=30,row=5,column=1,sticky='e
')
```

```

##### Footer Frame #####
#####
frame3 = tk.Frame(help_index_window,bg= 'gray')
frame3.place(relx = 0.11,rely = 0.895, relwidth = 0.80,
relheight = 0.09)

button1 = tk.Button(frame3,text = 'Quit', bd=4, command
= help_index_window.destroy,font = ('verdana',12,'bold'),bg
= '#262523',fg = 'white',height = 1, width = 10)
button1.grid(pady=20,padx=20,row=0,column=1,sticky='w')
button2 = tk.Button(frame3,text = 'Go to home page', bd=
4, command = Go_to_home_page,font = ('verdana',12,'bold'),bg
= '#262523',fg = 'white',height = 1, width = 15)
button2.grid(pady=20,padx=20,row=0,column=0,sticky='w')
help_index_window.mainloop()
##### End o
f Help Window #####
#####

##### Home p
age window #####
#####
def home_page(window1 = window1):

##### Framing Of Window #####
#####
frame1 = tk.Frame(window1, bg = 'gray')
frame1.place(relx = 0.11,rely = 0, relwidth = 0.80, relh
eight = 0.118)
frame2 = tk.Frame(window1,bg= 'gray')
frame2.place(relx = 0.11,rely = 0.12, relwidth = 0.80, r
elheight = 0.118)
frame3 = tk.Frame(window1, bg = 'gray')
frame3.place(relx = 0.11,rely = 0.24, relwidth = 0.80, r
elheight = 0.643)
frame4 = tk.Frame(window1, bg = 'gray')
frame4.place(relx = 0.11,rely = 0.88, relwidth = 0.80, r
elheight = 0.1)

```

```

##### Loading Register Image
and Performing operation for start_face_reading function ###
#####
def Register_page():
    Registration_page = Registration()
    Registration_page.register_new_employee(window1)

path = 'images'
images = []
mylist = os.listdir(path)
def find_Encoding(images):
    encodedList = []
    for img in images:
        img = cv2.resize(img,(0,0),None,0.25,0.25)
        img = cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
        encode = face_recognition.face_encodings(img)[0]
        encodedList.append(encode)
    return encodedList
if (len(mylist)>0) and os.path.isdir(path):
    registered_img = cv2.imread("{}\\{}".format(path,myl
ist[0]))
    images.append(registered_img)
    encode_of_registered_img = find_Encoding(images)
    with open('Details.txt','r') as f :
        detail = f.readlines()
        id = detail[0].replace('\n','')
        name = detail[1].replace('\n','')
        duration = detail[2]+ ":00"
else:
    Register_page()
##### Showing
details of Employee #####
#####
tree = ttk.Treeview(frame3,height = 25)
style = ttk.Style(tree)

style.configure(".",font = ('Helvetica',10))
style.theme_use("clam")

```



```

    style.configure("Treeview.Heading",foreground = 'red',font = ('Helvetica',10,"bold"))
    style.configure("Treeview",rowheight=25,fieldbackground="silver")
    style.map('Treeview',background=[('selected','red')])
    tree.tag_configure('evenrow',background = "lightblue")
    tree.tag_configure('oddrow',background = "white")
    tree['show'] = 'headings'

    tree["columns"] = ("one","two","three","four","five")
    tree.column("one",width = 125,anchor = 'center')
    tree.column("two",width = 250,anchor = 'center')
    tree.column("three",width = 150,anchor = 'center')
    tree.column("four",width = 100,anchor='center')
    tree.column("five",width = 150,anchor='center')
    tree.heading("one", text='ID')
    tree.heading("two", text='Name')
    tree.heading("three",text='Working Hours')
    tree.heading("four",text='Login time')
    tree.heading("five",text='Working Hours Left')
    tree.pack()

    login_time = (datetime.now()).strftime('%H:%M:%S')
    def insert_details_to_home_page():
        tree.insert('','end',text = "0", value = (id,name,duration,login_time,duration),tags = ('evenrow',))

    class set_working_hour_left():
        def __init__(self):
            self.work_hour_left=""
        def set_working_hour(self,whl):
            self.work_hour_left = whl
        def get_whl(self):
            return self.work_hour_left
    obj = set_working_hour_left()
    class tracking_working_time():
        def __init__(self):
            self.date = (datetime.now()).strftime("20%y-%m-%d")

```

```

        self.id = id
        self.name = name
        self.duration = duration
        self.working_hour_left = obj.get_whl()
        self.i = 1
    def get_data(self):
        lst = [self.date,self.id,self.name,self.duration
,login_time,self.working_hour_left]
        return lst

    def tracking_working_hour(self):
        self.working_hour_left = obj.get_whl()
        self.working_hour_left = str(self.working_hour_l
eft)

        if self.i % 2 ==0 :
            tree.insert('', 'end', value = (self.id,self.
name,self.duration,login_time,self.working_hour_left),tags =
('evenrow',))
        else:
            tree.insert('', 'end', value = (self.id,self.
name,self.duration,login_time,self.working_hour_left),tags =
('oddrow',))
        self.i +=1

class Data_insert():
    def insert_details(self):
        lst = calculate_working_time.get_data()
        if lst[5] == "":
            return 0
        with open('Data.csv','a') as f1:
            data = ("\n{},{},{},{},{},{},{}".format(lst[0]
,lst[1],lst[2],lst[3],lst[4],lst[5])
            f1.writelines(data)

##### Heading Of Window #####
#####
l1 = tk.Label(master= frame1,text = "Face Recognition Ba
sed Employee Activity Tracker",bg = 'gray', fg = "black",fon
t = ('verdana',20,'bold'))

```

```

l1.pack(anchor = 'center',pady = 4 )
l2 = tk.Label(master=frame1,text="",font = ('verdana',16
,'bold'),bg = "gray", fg = "black")
l2.pack(anchor = 'e',side = 'right',pady = 10)

##### Operation on loaded register
image completed #####
calculate_working_time = tracking_working_time()
def start_face_reading(already_login=True,name=name,calculate_working_time=calculate_working_time):
    count = 0
    cap = cv2.VideoCapture(0)
    while True:
        success, img = cap.read()
        if success:
            imgs = cv2.resize(img,(0,0),None,0.25,0.25)
            imgs = cv2.cvtColor(imgs,cv2.COLOR_BGR2RGB)
            facesInFrame = face_recognition.face_locations(imgs)

            encodefacesInFrame = face_recognition.face_encodings(imgs,facesInFrame)
            for encodeface, faceloc_Cap in zip(encodefacesInFrame,facesInFrame):
                matches = face_recognition.compare_faces(encode_of_registered_img,encodeface)
                facedis = face_recognition.face_distance(encode_of_registered_img,encodeface)
                matchIndex = np.argmin(facedis)
                if matches[matchIndex]:
                    y1, x2, y2, x1 = faceloc_Cap
                    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),2,cv2.FILLED)
                    cv2.putText(img,name,(x1,y2+30),cv2.FONT_HERSHEY_COMPLEX,1,(255,255,255),2)
                    count +=1

```

```

        if count > 3 and already_login:
            cap.release()
            cv2.destroyAllWindows()
            insert_details_to_home_page()
            timer1 = timer()
            timer1.countdown()
            count = 0
            return 0
        elif count > 3:
            cap.release()
            cv2.destroyAllWindows()
            calculate_working_time.tracking_
working_hour()

            count = 0
            return 0
        cv2.imshow("Detecting Face ",img)
        cv2.waitKey(1)
    else:
        messagebox.showinfo("Camera Opening Error",
"Can't able open webCam... ")
    def exit():
        obj = Data_insert()
        obj.insert_details()
        window1.destroy()

    ##### TIMER #####
    #####
    class timer():
        def __init__(self,duration = duration):
            self.t = duration.split(":")[0]
            self.t = int(self.t)*60*60
            self.count = 0
            self.rand_time = random.randint(10,15)
        def countdown(self):
            mins, sec = divmod(self.t,60)
            hour, mins = divmod(mins,60)
            timer = '{:02d}:{:02d}:{:02d}'.format(hour,mins,
sec)

            obj.set_working_hour(timer)
            l2.config(text = timer)

```

```

        12.after(1000,self.countdown)
        self.t -= 1
        if self.count >= self.rand_time:
            start_face_reading(False)
            self.count = 0
        self.count += 1
        if self.t <= 0 :
            exit()

def go_to_help_index():
    help_index_window = tk.Tk()
    help_index_window.geometry("1280x780")
    help_index_window.minsize(1080,720)
    help_index_window.title("Employee Activity Tracker")
    help_index_window.iconbitmap("icon.ico")
    help_index_window.configure(background = 'gray')
    help_index(window1 , help_index_window)

def go_to_about():
    about_window = tk.Tk()
    about_window.geometry("1280x720")
    about_window.minsize(1080,720)
    about_window.title("Employee Activity Tracker")
    about_window.iconbitmap("icon.ico")
    about_window.configure(background = 'gray')
    About(window1 , about_window)

def Delete_employee():
    msg = messagebox.askquestion("Comfirm","Are you sure
!!!!\nYou want to Delete Employee details")
    if msg == "yes":
        path = "images"
        image = os.listdir(path)
        for img in image:
            path = path + "\\ " + img
            if os.path.isfile(path):
                os.remove(path)
            else:
                messagebox.showinfo("Invalid","Employee
not registered")
        if os.path.isfile("Details.txt"):

```

```

        with open("Details.txt","w") as f:
            f.write("")
        messagebox.showinfo("Closing ","Restart the
application")
        window1.destroy()
    else:
        messagebox.showinfo("Invalid","Employee not
registered")

class Reset_Working_hour():
    def __init__(self):
        self.reset_duration = tk.Tk()
        self.reset_duration.geometry("360x180")
        self.reset_duration.resizable('false','false')
        self.reset_duration.title("Reset Working Duratio
n")

        self.reset_duration.iconbitmap("icon.ico")
        self.reset_duration.configure(background = 'blac
k')

        l1 = tk.Label(self.reset_duration,text = "Reset
Duration : ",bg = "black", fg = "white",font = ('Helvetica',
12,'bold'))
        l1.grid(pady=20,padx=20,row=0,column=0,sticky='w
')

        option = ["01:00","02:00","03:00","04:00","05:00
","06:00","07:00","08:00","09:00","10:00","11:00","12:00"]
        self.var = tk.StringVar()
        self.var.set("HH:MM")
        self.drop2 = tk.OptionMenu(self.reset_duration,s
elf.var,*option)
        self.drop2.grid(pady=20,padx=20,row=0,column=1,s
ticky='e')
        self.drop2.configure(bg="gray",width = 10)

        self.reset_button = tk.Button(self.reset_duratio
n,text = "Reset",command = self.reset ,font = ('Helvetica',
12,'bold'),bg = "gray",fg="white",bd = 4,width = 10)

```

```

        self.reset_button.grid(pady=20,padx=20,row=1,column=0,sticky='w')
        self.Quit = tk.Button(self.reset_duration,text =
            "Quit", command = self.exit,font = ('Helvetica',12,'bold'),
            bg = "gray",fg="white",bd = 4,width = 10)
        self.Quit.grid(pady=20,padx=20,row=1,column=1,sticky='e')

        self.l2 = tk.Label(self.reset_duration,text = ""
            ,bg = "black", fg = "red",font = ('Helvetica',12,'bold'))
        self.l2.grid(pady=0,padx=20,row=2,column=0,sticky='w',columnspan = 2)

        self.reset_duration.mainloop()
    def exit(self):
        self.reset_duration.destroy()
    def reset(self):
        if os.path.isfile("Details.txt"):
            with open("Details.txt",'r') as f:
                data = f.readlines()
                if (len(data) < 1):
                    self.l2.configure(text = "Please Register
First!!!!!!")
                elif (self.var.get() == "HH:MM"):
                    self.l2.configure(text = "Please select t
he duration!!!!!!")
                else:
                    with open("Details.txt",'w') as f:
                        data[2] = self.var.get()
                        f.writelines(data)
                        messagebox.showinfo("Done","Reset Du
ration Successfully.....")
                    self.reset_duration.destroy()

        else:
            self.l2.configure(text = "Please Register Fir
st!!!!!!")

    def reset_hour():

```

```

obj = Reset_Working_hour()

##### MenuBar #####
#####
menubar = Menu(window1)
filemenu = Menu(menubar, tearoff = 0)
filemenu.add_command(label = 'Register New Employee', co
mmand = Register_page,font = ('verdana',8,'bold'))
filemenu.add_command(label = 'Reset Working Duration', c
ommand = reset_hour,font = ('verdana',8,'bold'))
filemenu.add_command(label = 'Delete Employee Details',
command = Delete_employee,font = ('verdana',8,'bold'))
filemenu.add_command(label = 'Exit', command = exit,font
= ('verdana',8,'bold'))
menubar.add_cascade(label = 'Edit', menu = filemenu)

helpmenu = Menu(menubar, tearoff = 0)
helpmenu.add_command(label = 'Help index', command = go_
to_help_index, font = ('verdana',8,'bold'))
helpmenu.add_command(label = 'About...', command = go_to
_about, font = ('verdana',8,'bold'))
menubar.add_cascade(label = 'Help',menu = helpmenu)
window1.config(menu = menubar )

##### Buttons of home page #####
#####
button1 = tk.Button(frame2,text = 'Start Your Work', bd=
4, command = start_face_reading ,font = ('verdana',12,'bold'
),bg = '#262523',fg = 'white',height = 2, width = 35)
button1.pack(anchor = 'center',pady = 20)

button2 = tk.Button(frame4,text = 'Quit', bd=4, command
= exit,font = ('verdana',12,'bold'),bg = '#262523',fg = 'whi
te',height = 1, width = 10)
button2.pack(anchor = 'w',side = 'bottom',pady = 10, pad
x = 50 )

window1.iconbitmap("icon.ico")

```



```
##### End of
Home page window #####
#####
home_page()
window1. mainloop()
```