**LOGIN PAGE**

from tkinter import\*

from tkinter import ttk

from tkinter import messagebox

from turtle import width

from PIL import Image,ImageTk

class login\_Window:

    def \_\_init\_\_(self,root):

        self.root=root

        self.root.title("Login")

        self.root.geometry("1300x800+0+0")

        self.bg=ImageTk.PhotoImage(file=r"C:/Users/HP/Desktop/minner project/nature image.jpg")

        lbl\_bg=Label(self.root,image=self.bg)

        lbl\_bg.place(x=0,y=0,relwidth=1,relheight=1)

        frame=Frame(self.root,bg="black")

        frame.place(x=410,y=120,width=400,height=450)

        img1=Image.open(r"C:/Users/HP\Desktop/minner project/logo\_transparent.png")

        img1=img1.resize((100,100),Image.ANTIALIAS)

        self.photoimage1=ImageTk.PhotoImage(img1)

        lblimg1=Label(image=self.photoimage1,bg="white",borderwidth=0)

        lblimg1.place(x=555,y=125,width=100,height=100)

        get\_str=Label(frame,text="Get started",font=("times new roman",20,"bold"),fg="white",bg="black")

        get\_str.place(x=120,y=110)

        #lable

        username=lbl=Label(frame,text="username",font=("times new roman",15,"bold"),fg="white",bg="black")

        username.place(x=70,y=155)

        self.txtuser=ttk.Entry(frame,font=("times new roman",15,"bold"))

        self.txtuser.place(x=40,y=190,width=270)

        password=lbl=Label(frame,text="Password",font=("times new roman",15,"bold"),fg="white",bg="black")

        password.place(x=70,y=230)

        self.txtpass=ttk.Entry(frame,font=("times new roman",15,"bold"))

        self.txtpass.place(x=40,y=265,width=270)

        #\*\*\*\*\*\*\*\*\*\*\*ICON Images\*\*\*\*\*\*\*\*\*\*\*

        img2=Image.open(r"C:/Users/HP\Desktop/minner project/app logo.jpeg")

        img2=img2.resize((25,25),Image.ANTIALIAS)

        self.photoimage2=ImageTk.PhotoImage(img2)

        lblimg1=Label(image=self.photoimage2,bg="black",borderwidth=0)

        lblimg1.place(x=450,y=275,width=25,height=25)

        img3=Image.open(r"C:/Users/HP\Desktop/minner project/app logo.jpeg")

        img3=img3.resize((25,25),Image.ANTIALIAS)

        self.photoimage3=ImageTk.PhotoImage(img3)

        lblimg3=Label(image=self.photoimage3,bg="black",borderwidth=0)

        lblimg3.place(x=450,y=350,width=25,height=25)

        #login button

        loginbtn=Button(frame,command=self.login, text="Login",font=("times new roman",15,"bold"),bd=3, relief=RIDGE, fg="white",bg="red",activeforeground="white",activebackground="red")

        loginbtn.place(x=110,y=320,width=120,height=30)

        #regester button

        regesterbtn=Button(frame,text="new user Regester",font=("times new roman",15,"bold"),borderwidth=0, fg="white",bg="black",activeforeground="white",activebackground="black")

        regesterbtn.place(x=20,y=400,width=160)

        #forget password

        forgetpasswordbtn=Button(frame,text="forget password",font=("times new roman",15,"bold"),borderwidth=0, fg="white",bg="black",activeforeground="white",activebackground="black")

        forgetpasswordbtn.place(x=10,y=370,width=160)

    def login(self):

        if self.txtuser.get()=="" and self.txtpass.get()=="":

            messagebox.showerror("error", "all field required")

        elif self.txtuser.get()=="shubham" and self.txtpass.get()=="shub":

            messagebox.showinfo("success","welcome to the company ")

        else:

            messagebox.showerror("Invalid","Invalid username and password")

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

    root=Tk()

    app=login\_Window(root)

    root.mainloop()

**register page**

from tkinter import\*

from tkinter import ttk

from tkinter import messagebox

from PIL import Image,ImageTk

class Register:

    def \_\_init\_\_(self,root):

        self.root=root

        self.root.title("regester")

        self.root.geometry("1300x800+0+0")

        #\*\*\*\*\*\*\*\*\*\*\*\*\*\*bg img\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

        self.bg=ImageTk.PhotoImage(file=r"C:/Users/HP/Desktop/minner project/nature image.jpg")

        lbl\_bg=Label(self.root,image=self.bg)

        lbl\_bg.place(x=0,y=0,relwidth=1,relheight=1)

        # \*\*\*\*\*\*\*\*\*\*\*\*\*left image\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

        self.bg1=ImageTk.PhotoImage(file=r"C:/Users/HP/Desktop/minner project/ss.jpeg")

        left\_bg1=Label(self.root,image=self.bg1)

        left\_bg1.place(x=50,y=100, width=400,height=500)

       #\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*main frame\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

        frame=Frame(self.root,bg="white")

        frame.place(x=450,y=150,width=750,height=450)

        register\_lbl=Label(frame,text="REGISTER HERE",font=("times new roman",20,"bold"),fg="dark green",bg="white")

        register\_lbl.place(x=20,y=20)

        #\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*lables and entry\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

        #\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*row1\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

        fname=Label(frame,text="First Name",font=("times new roman",15,"bold"),bg="white")

        fname.place(x=50,y=70)

        fname\_entry=ttk.Entry(frame,font=("times new roman",15,"bold"))

        fname\_entry.place(x=50,y=100,width=250)

        l\_name=Label(frame,text="Last Name",font=("times new roman",15,"bold"),bg="white",fg="black")

        l\_name.place(x=370,y=70)

        self.txt\_lname=ttk.Entry(frame,font=("times new roman",15))

        self.txt\_lname.place(x=370,y=100,width=250)

        #\*\*\*\*\*\*\*\*\*\*\*\*row2\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

        contact=Label(frame,text="contact no",font=("times new roman",15,"bold"),bg="white",fg="black")

        contact.place(x=50,y=140)

        self.txt\_contact=ttk.Entry(frame,font=("times new roman",15))

        self.txt\_contact.place(x=50,y=170,width=250)

        email=Label(frame,text="Email",font=("times new roman",15,"bold"),bg="white",fg="black")

        email.place(x=370,y=140)

        self.txt\_email=ttk.Entry(frame,font=("times new roman",15))

        self.txt\_email.place(x=370,y=170,width=250)

        #\*\*\*\*\*\*\*\*\*\*\*\*row3\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

        security\_Q=Label(frame,text="select security quetions",font=("times new roman",15,"bold"),bg="white",fg="black")

        security\_Q.place(x=50,y=210)

        self.txt\_combosecurity\_Q=ttk.Combobox(frame,font=("times new roman",15),state="ReadOnly")

        self.txt\_combosecurity\_Q["values"]=("select","your Birth place","your country")

        self.txt\_combosecurity\_Q.place(x=50,y=240,width=250)

        self.txt\_combosecurity\_Q.current(0)

        security\_A=Label(frame,text="security\_Answer",font=("times new roman",15,"bold"),bg="white",fg="black")

        security\_A.place(x=370,y=210)

        self.txt\_security=ttk.Entry(frame,font=("times new roman",15))

        self.txt\_security.place(x=370,y=240,width=250)

        #\*\*\*\*\*\*\*\*\*\*\*\*row4\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

        pswd=Label(frame,text="password ",font=("times new roman",15,"bold"),bg="white",fg="black")

        pswd.place(x=50,y=280)

        self.txt\_pswd=ttk.Entry(frame,font=("times new roman",15))

        self.txt\_pswd.place(x=50,y=310,width=250)

        confirm\_pswd=Label(frame,text="confirm password",font=("times new roman",15,"bold"),bg="white",fg="black")

        confirm\_pswd.place(x=370,y=280)

        self.txt\_confirm\_pswd=ttk.Entry(frame,font=("times new roman",15))

        self.txt\_confirm\_pswd.place(x=370,y=310,width=250)

        #\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*chechbutton\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

        Checkbtn=Checkbutton(frame,text="I agree the term and conditions",font=("times new roman",12,"bold"),onvalue=1,offvalue=0)

        Checkbtn.place(x=50,y=350)

        #\*\*\*\*\*\*\*\*\*\*\*\*\*\*buttons\*\*\*\*\*\*\*\*\*\*\*\*\*\*

        img4=Image.open("C:/Users/HP/Desktop/minner project/button regester.jpeg")

        img4=img4.resize((100,50),Image.ANTIALIAS)

        self.photoimage=ImageTk.PhotoImage(img4)

        b1=Button(frame,image=self.photoimage,borderwidth=0,cursor="hand2",font=("times new roman",15,"bold"),bg="white")

        b1.place(x=100,y=390,width=200)

        # img5=Image.open("C:/Users/HP/Desktop/minner project/b88.png")

        # img5=img5.resize((100,50),Image.ANTIALIAS)

        # self.photoimage=ImageTk.PhotoImage(img5)

        # b2=Button(frame,image=self.photoimage,borderwidth=0,cursor="hand2",font=("times new roman",15,"bold"),bg="white")

        # b2.place(x=300,y=390,width=200)

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

    root=Tk()

    app=Register(root)

    root.mainloop()

**Attendance management system**

from tkinter import\*

from tkinter import ttk

from tkinter.tix import TEXT

from turtle import left, st

from PIL import Image,ImageTk

class employee:

    def \_\_init\_\_(self,root):

        self.root=root

        self.root.geometry("1200x790+0+0")

        self.root.title("face recogniton system")

        #first image

        img1=Image.open(r"C:\Users\HP\Desktop\minner project\sss.jpeg")

        img1=img1.resize((500,130),Image.ANTIALIAS)

        self.photoimg1=ImageTk.PhotoImage(img1)

        f\_lbl=Label(self.root,image=self.photoimg1)

        f\_lbl.place(x=0,y=0,width=500,height=120)

        #2 image

        img2=Image.open(r"C:\Users\HP\Desktop\minner project\ss.jpeg")

        img2=img2.resize((500,130),Image.ANTIALIAS)

        self.photoimg2=ImageTk.PhotoImage(img2)

        f\_lbl=Label(self.root,image=self.photoimg2)

        f\_lbl.place(x=500,y=0,width=500,height=120)

        #3 image

        img3=Image.open(r"C:\Users\HP\Desktop\minner project\ssss.jpeg")

        img3=img3.resize((450,130),Image.ANTIALIAS)

        self.photoimg3=ImageTk.PhotoImage(img3)

        f\_lbl=Label(self.root,image=self.photoimg3)

        f\_lbl.place(x=1000,y=0,width=450,height=120)

        #bg image

        img4=Image.open(r"C:\Users\HP\Desktop\minner project\b77.jpeg")

        img4=img4.resize((1400,710),Image.ANTIALIAS)

        self.photoimg4=ImageTk.PhotoImage(img4)

        bg\_img=Label(self.root,image=self.photoimg4)

        bg\_img.place(x=0,y=120,width=1400,height=710)

        title\_lbl=Label(bg\_img,text="EMPLOYEE ATTENDANCE MANAGMENT SYSTEM ",font=("times new roman",28,"bold"),bg="white",fg="dark red")

        title\_lbl.place(x=0,y=0,width=1430,height=35)

        main\_frame=Frame(bg\_img,bd=2, bg="white")

        main\_frame.place(x=10,y=45,width=1250,height=470)

        #left lable frame

        left\_frame=LabelFrame(main\_frame,bd=2, bg="white",relief=RIDGE,text="employee Details",font=("times new roman ",12,"bold"))

        left\_frame.place(x=10,y=10,width=600,height=450)

        img\_left=Image.open(r"C:\Users\HP\Desktop\minner project\b66.jpeg")

        img\_left=img\_left.resize((600,100),Image.ANTIALIAS)

        self.photoimg\_left=ImageTk.PhotoImage(img\_left)

        f\_lbl=Label(self.root,image=self.photoimg\_left)

        f\_lbl.place(x=33,y=200,width=587,height=90)

        #current courses

        current\_course\_frame=LabelFrame(main\_frame,bd=2, bg="white",relief=RIDGE,text="current Work information",font=("times new roman ",12,"bold"))

        current\_course\_frame.place(x=20,y=120,width=580,height=110)

        #department

        dep\_label=Label(current\_course\_frame,text="department",font=("times new roman",12,"bold"),bg="white")

        dep\_label.grid(row=0,column=0,padx=10,sticky=W)

        dep\_combo=ttk.Combobox(current\_course\_frame,font=("times new roman",12,"bold"),state="read only",width=17)

        dep\_combo["values"]=("select Department","computer","it","civil","mechnical")

        dep\_combo.current(0)

        dep\_combo.grid(row=0,column=1,padx=2,pady=10,sticky=W)

        # course

        course\_label=Label(current\_course\_frame,text="courses",font=("times new roman",12,"bold"),bg="white")

        course\_label.grid(row=0,column=2,padx=10,sticky=W)

        course\_combo=ttk.Combobox(current\_course\_frame,font=("times new roman",12,"bold"),state="read only",width=17)

        course\_combo["values"]=("select Department","computer","it","civil","mechnical")

        course\_combo.current(0)

        course\_combo.grid(row=0,column=3,padx=2,pady=10,sticky=W)

        # year

        year\_label=Label(current\_course\_frame,text="year",font=("times new roman",12,"bold"),bg="white")

        year\_label.grid(row=1,column=0,padx=10,sticky=W)

        year\_combo=ttk.Combobox(current\_course\_frame,font=("times new roman",12,"bold"),state="read only",width=17)

        year\_combo["values"]=("select year","2020-21","2021-22","2022-23","2023-24")

        year\_combo.current(0)

        year\_combo.grid(row=1,column=1,padx=2,pady=10,sticky=W)

         #semester

        semester\_label=Label(current\_course\_frame,text="semester",font=("times new roman",12,"bold"),bg="white")

        semester\_label.grid(row=1,column=2,padx=10,sticky=W)

        semester\_combo=ttk.Combobox(current\_course\_frame,font=("times new roman",12,"bold"),state="read only",width=17)

        semester\_combo["values"]=("select sem","1st","2nd","3rd","4th","5th","6th")

        semester\_combo.current(0)

        semester\_combo.grid(row=1,column=3,padx=2,pady=10,sticky=W)

        #company employee information

        company\_employee\_frame=LabelFrame(main\_frame,bd=2, bg="white",relief=RIDGE,text="employee information",font=("times new roman ",12,"bold"))

        company\_employee\_frame.place(x=20,y=240,width=580,height=210)

        #employee ID

        employeeId\_label=Label(company\_employee\_frame,text="employeeId:",font=("times new roman",12,"bold"),bg="white")

        employeeId\_label.grid(row=0,column=0,padx=10,pady=7,sticky=W)

        employeeId\_entry=ttk.Entry(company\_employee\_frame,width=16,font=("times new roman",12,"bold"))

        employeeId\_entry.grid(row=0,column=1,padx=5,pady=2,sticky=W)

        #employee Name

        employeeName\_label=Label(company\_employee\_frame,text="employeeName:",font=("times new roman",12,"bold"),bg="white")

        employeeName\_label.grid(row=0,column=2,padx=10,pady=5,sticky=W)

        employeeName\_entry=ttk.Entry(company\_employee\_frame,width=16,font=("times new roman",12,"bold"))

        employeeName\_entry.grid(row=0,column=3,padx=5,pady=2,sticky=W)

        #employee Gender

        employeeGender\_label=Label(company\_employee\_frame,text="employeeGender:",font=("times new roman",12,"bold"),bg="white")

        employeeGender\_label.grid(row=1,column=0,padx=10,pady=5,sticky=W)

        employeeGender\_entry=ttk.Entry(company\_employee\_frame,width=16,font=("times new roman",12,"bold"))

        employeeGender\_entry.grid(row=1,column=1,padx=5,pady=2,sticky=W)

        #employee age

        employeeage\_label=Label(company\_employee\_frame,text="employeeage:",font=("times new roman",12,"bold"),bg="white")

        employeeage\_label.grid(row=1,column=2,padx=10,pady=5,sticky=W)

        employeeage\_entry=ttk.Entry(company\_employee\_frame,width=16,font=("times new roman",12,"bold"))

        employeeage\_entry.grid(row=1,column=3,padx=5,pady=2,sticky=W)

        #employee address

        employeeaddress\_label=Label(company\_employee\_frame,text="employeeaddress:",font=("times new roman",12,"bold"),bg="white")

        employeeaddress\_label.grid(row=1,column=0,padx=10,pady=7,sticky=W)

        employeeaddress\_entry=ttk.Entry(company\_employee\_frame,width=16,font=("times new roman",12,"bold"))

        employeeaddress\_entry.grid(row=1,column=1,padx=5,pady=2,sticky=W)

        #employee Email

        employeeEmail\_label=Label(company\_employee\_frame,text="employeeEmail:",font=("times new roman",12,"bold"),bg="white")

        employeeEmail\_label.grid(row=2,column=0,padx=10,pady=5,sticky=W)

        employeeEmail\_entry=ttk.Entry(company\_employee\_frame,width=16,font=("times new roman",12,"bold"))

        employeeEmail\_entry.grid(row=2,column=1,padx=5,pady=2,sticky=W)

        #employee PhNo

        employeePhNo\_label=Label(company\_employee\_frame,text="employeePhNo:",font=("times new roman",12,"bold"),bg="white")

        employeePhNo\_label.grid(row=2,column=2,padx=10,pady=5,sticky=W)

        employeePhNo\_entry=ttk.Entry(company\_employee\_frame,width=16,font=("times new roman",12,"bold"))

        employeePhNo\_entry.grid(row=2,column=3,padx=5,pady=2,sticky=W)

        #radio Button

        Radiobtn1=ttk.Radiobutton(company\_employee\_frame,text="take photo sample",value="yes")

        Radiobtn1.grid(row=6,column=0)

        Radiobtn2=ttk.Radiobutton(company\_employee\_frame,text="No photo sample",value="yes")

        Radiobtn2.grid(row=6,column=1)

        #bbuttons frame

        btn\_frame=Frame(company\_employee\_frame,bd=2,relief=RIDGE,bg="white")

        btn\_frame.place(x=0,y=135,width=575,height=50)

        save\_btn=Button(btn\_frame,text="save",width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        save\_btn.grid(row=0,column=0)

        update\_btn=Button(btn\_frame,text="update",width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        update\_btn.grid(row=0,column=1)

        delete\_btn=Button(btn\_frame,text="delete",width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        delete\_btn.grid(row=0,column=2)

        reset\_btn=Button(btn\_frame,text="reset",width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        reset\_btn.grid(row=0,column=3)

        take\_photo\_btn=Button(btn\_frame,text="Take photo sample",width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        take\_photo\_btn.grid(row=1,column=0)

        update\_photo\_btn=Button(btn\_frame,text="update photo sample",width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        update\_photo\_btn.grid(row=1,column=1)

        #Right lable frame

        Right\_frame=LabelFrame(main\_frame,bd=2, bg="white",relief=RIDGE,text="employee Details",font=("times new roman ",12,"bold"))

        Right\_frame.place(x=650,y=10,width=590,height=450)

        img\_right=Image.open(r"C:\Users\HP\Desktop\minner project\b.jpeg")

        img\_right=img\_right.resize((600,100),Image.ANTIALIAS)

        self.photoimg\_right=ImageTk.PhotoImage(img\_right)

        f\_lbl=Label(self.root,image=self.photoimg\_right)

        f\_lbl.place(x=670,y=200,width=580,height=90)

        #\*\*\*\*\*\*\*\*\*\*search system\*\*\*\*\*\*\*\*\*\*\*\*\*

        search\_frame=LabelFrame(Right\_frame,bd=2, bg="white",relief=RIDGE,text="search system",font=("times new roman ",12,"bold"))

        search\_frame.place(x=5,y=100,width=575,height=70)

        search\_label=Label(search\_frame,text="Search By:",font=("times new roman",13,"bold"),bg="white")

        search\_label.grid(row=0,column=0,padx=10,pady=2,sticky=W)

        search\_combo=ttk.Combobox(search\_frame,font=("times new roman",12,"bold"),state="read only",width=10)

        search\_combo["values"]=("select","1st","emp\_id","phone\_no")

        search\_combo.current(0)

        search\_combo.grid(row=0,column=1,padx=2,pady=10,sticky=W)

        search\_entry=ttk.Entry(search\_frame,width=14,font=("times new roman",12,"bold"))

        search\_entry.grid(row=0,column=2,padx=10,pady=5,sticky=W)

        search\_btn=Button(search\_frame,text="search",width=10,font=("times new roman",10,"bold"),bg="blue",fg="white")

        search\_btn.grid(row=0,column=3,padx=4)

        search\_btn=Button(search\_frame,text="show All",width=10,font=("times new roman",10,"bold"),bg="blue",fg="white")

        search\_btn.grid(row=0,column=4,padx=4)

        table\_frame=Frame(Right\_frame,bg="white",bd=2,relief=RIDGE)

        table\_frame.place(x=5,y=180,width=575,height=240)

        scroll\_x=ttk.Scrollbar(table\_frame,orient=HORIZONTAL)

        scroll\_y=ttk.Scrollbar(table\_frame,orient=VERTICAL)

        self.employee\_table=ttk.Treeview(table\_frame,column=("dep","course","year","sem","id","name","gender","age","email","phone","address","photo"),xscrollcommand=scroll\_x.set,yscrollcommand=scroll\_y.set)

        scroll\_x.pack(side=BOTTOM,fill=X)

        scroll\_y.pack(side=RIGHT,fill=Y)

        scroll\_x.config(command=self.employee\_table.xview)

        scroll\_y.config(command=self.employee\_table.yview)

        self.employee\_table.heading("dep",text="Department")

        self.employee\_table.heading("course",text="courses")

        self.employee\_table.heading("year",text="year of joining")

        self.employee\_table.heading("sem",text="semester")

        self.employee\_table.heading("id",text="id")

        self.employee\_table.heading("gender",text="gender")

        self.employee\_table.heading("age",text="age")

        self.employee\_table.heading("email",text="email")

        self.employee\_table.heading("phone",text="Phone no")

        self.employee\_table.heading("address",text="address")

        self.employee\_table.heading("photo",text="photo")

        self.employee\_table["show"]="headings"

        self.employee\_table.column("dep",width=100)

        self.employee\_table.column("course",width=100)

        self.employee\_table.column("year",width=100)

        self.employee\_table.column("sem",width=100)

        self.employee\_table.column("id",width=100)

        self.employee\_table.column("gender",width=100)

        self.employee\_table.column("age",width=100)

        self.employee\_table.column("email",width=100)

        self.employee\_table.column("phone",width=100)

        self.employee\_table.column("address",width=100)

        self.employee\_table.column("photo",width=150)

        self.employee\_table.pack(fill=BOTH,expand=1)

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

    root=Tk()

    obj=employee(root)

    root.mainloop()

**Home page**

from tkinter import\*

from tkinter import ttk

from PIL import Image,ImageTk

from employeedetail import Employee

class Face\_Recognition\_System:

   def \_\_init\_\_(self,root):

    self.root=root

    self.root.geometry("1400x660+0+0")

    self.root.title("face Recognition System")

    #first image

    img1=Image.open(r"C:\Users\HP\Desktop\minner project\sss.jpeg")

    img1=img1.resize((500,130),Image.ANTIALIAS)

    self.photoimg1=ImageTk.PhotoImage(img1)

    f\_lbl=Label(self.root,image=self.photoimg1)

    f\_lbl.place(x=0,y=0,width=500,height=130)

    #2 image

    img2=Image.open(r"C:\Users\HP\Desktop\minner project\ss.jpeg")

    img2=img2.resize((500,130),Image.ANTIALIAS)

    self.photoimg2=ImageTk.PhotoImage(img2)

    f\_lbl=Label(self.root,image=self.photoimg2)

    f\_lbl.place(x=500,y=0,width=500,height=130)

#3 image

    img3=Image.open(r"C:\Users\HP\Desktop\minner project\ssss.jpeg")

    img3=img3.resize((450,130),Image.ANTIALIAS)

    self.photoimg3=ImageTk.PhotoImage(img3)

    f\_lbl=Label(self.root,image=self.photoimg3)

    f\_lbl.place(x=1000,y=0,width=450,height=130)

#bg image

    img4=Image.open(r"C:\Users\HP\Desktop\minner project\bg.jpeg")

    img4=img4.resize((1400,710),Image.ANTIALIAS)

    self.photoimg4=ImageTk.PhotoImage(img4)

    bg\_img=Label(self.root,image=self.photoimg4)

    bg\_img.place(x=0,y=130,width=1400,height=710)

    title\_lbl=Label(bg\_img,text="FACE RECOGNITION ATTENDANCE SYSTEM SOFTWARE",font=("times new roman",35,"bold"),bg="yellow",fg="red")

    title\_lbl.place(x=0,y=0,width=1430,height=45)

#employee button

    img5=Image.open(r"C:\Users\HP\Desktop\minner project\b.jpeg")

    img5=img5.resize((180,180),Image.ANTIALIAS)

    self.photoimg5=ImageTk.PhotoImage(img5)

    b1=Button(bg\_img,image=self.photoimg5,command=self.employee2 ,cursor="hand2")

    b1.place(x=200,y=100,width=180,height=180)

    b1\_1=Button(bg\_img,text="Employee Details",command=self.employee2 ,cursor="hand2",font=("times new roman",15,"bold"),bg="yellow",fg="red")

    b1\_1.place(x=200,y=280,width=180,height=25)

#detect face button

    img7=Image.open(r"C:\Users\HP\Desktop\minner project\face.jpeg")

    img7=img7.resize((180,180),Image.ANTIALIAS)

    self.photoimg7=ImageTk.PhotoImage(img7)

    b1=Button(bg\_img,image=self.photoimg7,cursor="hand2")

    b1.place(x=420,y=100,width=180,height=180)

    b1\_1=Button(bg\_img,text="Face Recognition",cursor="hand2",font=("times new roman",15,"bold"),bg="yellow",fg="red")

    b1\_1.place(x=420,y=280,width=180,height=25)

#Attendance button

    img6=Image.open(r"C:\Users\HP\Desktop\minner project\bbb.jpeg")

    img6=img6.resize((180,180),Image.ANTIALIAS)

    self.photoimg6=ImageTk.PhotoImage(img6)

    b1=Button(bg\_img,image=self.photoimg6,cursor="hand2")

    b1.place(x=640,y=100,width=180,height=180)

    b1\_1=Button(bg\_img,text="Attendance",cursor="hand2",font=("times new roman",15,"bold"),bg="yellow",fg="red")

    b1\_1.place(x=640,y=280,width=180,height=25)

#help

    img8=Image.open(r"C:\Users\HP\Desktop\minner project\b44.jpeg")

    img8=img8.resize((180,180),Image.ANTIALIAS)

    self.photoimg8=ImageTk.PhotoImage(img8)

    b1=Button(bg\_img,image=self.photoimg8,cursor="hand2")

    b1.place(x=860,y=100,width=180,height=180)

    b1\_1=Button(bg\_img,text="Help",cursor="hand2",font=("times new roman",15,"bold"),bg="yellow",fg="red")

    b1\_1.place(x=860,y=280,width=180,height=25)

#train data button

    img9=Image.open(r"C:\Users\HP\Desktop\minner project\b55.jpeg")

    img9=img9.resize((180,180),Image.ANTIALIAS)

    self.photoimg9=ImageTk.PhotoImage(img9)

    b1=Button(bg\_img,image=self.photoimg9,cursor="hand2")

    b1.place(x=200,y=340,width=180,height=180)

    b1\_1=Button(bg\_img,text="Train Data",cursor="hand2",font=("times new roman",15,"bold"),bg="yellow",fg="red")

    b1\_1.place(x=200,y=500,width=180,height=25)

#photos button

    img10=Image.open(r"C:\Users\HP\Desktop\minner project\b66.jpeg")

    img10=img10.resize((180,180),Image.ANTIALIAS)

    self.photoimg10=ImageTk.PhotoImage(img10)

    b1=Button(bg\_img,image=self.photoimg10,cursor="hand2")

    b1.place(x=420,y=340,width=180,height=180)

    b1\_1=Button(bg\_img,text="Photos",cursor="hand2",font=("times new roman",15,"bold"),bg="yellow",fg="red")

    b1\_1.place(x=420,y=500,width=180,height=25)

#Developer button

    img11=Image.open(r"C:\Users\HP\Desktop\minner project\b77.jpeg")

    img11=img11.resize((180,180),Image.ANTIALIAS)

    self.photoimg11=ImageTk.PhotoImage(img11)

    b1=Button(bg\_img,image=self.photoimg11,cursor="hand2")

    b1.place(x=640,y=340,width=180,height=180)

    b1\_1=Button(bg\_img,text="Developer",cursor="hand2",font=("times new roman",15,"bold"),bg="yellow",fg="red")

    b1\_1.place(x=640,y=500,width=180,height=25)

#exit button

    img12=Image.open(r"C:\Users\HP\Desktop\minner project\b88.png")

    img12=img12.resize((180,180),Image.ANTIALIAS)

    self.photoimg12=ImageTk.PhotoImage(img12)

    b1=Button(bg\_img,image=self.photoimg12,cursor="hand2")

    b1.place(x=860,y=340,width=180,height=180)

    b1\_1=Button(bg\_img,text="Exit",cursor="hand2",font=("times new roman",15,"bold"),bg="yellow",fg="red")

    b1\_1.place(x=860,y=500,width=180,height=25)

      #\*\*\*\*\*\*\*\*\* function button\*\*\*\*\*\*\*\*\*\*\*

   def employee2(self):

    self.new\_window=Toplevel(self.root)

    self.app=Employee(self.new\_window)

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

    root=Tk()

    obj=Face\_Recognition\_System(root)

    root.mainloop()

**employee2**

from tkinter import\*

from tkinter import ttk

from tkinter.tix import TEXT

from turtle import left, st

from PIL import Image,ImageTk

from tkinter import messagebox

class Employee:

    def \_\_init\_\_(self,root):

        self.root=root

        self.root.geometry("1200x790+0+0")

        self.root.title("face recogniton system")

        # \*\*\*\*\*\*\*\*\*\*\*\*\*\*variables\*\*\*\*\*\*\*\*\*\*\*

        self.var\_dep=StringVar()

        self.var\_Position=StringVar()

        self.var\_year=StringVar()

        self.var\_semester=StringVar()

        self.var\_id=StringVar()

        self.var\_name=StringVar()

        self.var\_gender=StringVar()

        self.var\_age=StringVar()

        self.var\_email=StringVar()

        self.var\_phone=StringVar()

        self.var\_address=StringVar()

        #first image

        img1=Image.open(r"C:\Users\HP\Desktop\minner project\sss.jpeg")

        img1=img1.resize((500,130),Image.ANTIALIAS)

        self.photoimg1=ImageTk.PhotoImage(img1)

        f\_lbl=Label(self.root,image=self.photoimg1)

        f\_lbl.place(x=0,y=0,width=500,height=120)

        #2 image

        img2=Image.open(r"C:\Users\HP\Desktop\minner project\ss.jpeg")

        img2=img2.resize((500,130),Image.ANTIALIAS)

        self.photoimg2=ImageTk.PhotoImage(img2)

        f\_lbl=Label(self.root,image=self.photoimg2)

        f\_lbl.place(x=500,y=0,width=500,height=120)

        #3 image

        img3=Image.open(r"C:\Users\HP\Desktop\minner project\ssss.jpeg")

        img3=img3.resize((450,130),Image.ANTIALIAS)

        self.photoimg3=ImageTk.PhotoImage(img3)

        f\_lbl=Label(self.root,image=self.photoimg3)

        f\_lbl.place(x=1000,y=0,width=450,height=120)

        #bg image

        img4=Image.open(r"C:\Users\HP\Desktop\minner project\b77.jpeg")

        img4=img4.resize((1400,710),Image.ANTIALIAS)

        self.photoimg4=ImageTk.PhotoImage(img4)

        bg\_img=Label(self.root,image=self.photoimg4)

        bg\_img.place(x=0,y=120,width=1400,height=710)

        title\_lbl=Label(bg\_img,text="EMPLOYEE ATTENDANCE MANAGMENT SYSTEM ",font=("times new roman",28,"bold"),bg="white",fg="dark red")

        title\_lbl.place(x=0,y=0,width=1430,height=35)

        main\_frame=Frame(bg\_img,bd=2, bg="white")

        main\_frame.place(x=10,y=45,width=1250,height=470)

        #left lable frame

        left\_frame=LabelFrame(main\_frame,bd=2, bg="white",relief=RIDGE,text="employee Details",font=("times new roman ",12,"bold"))

        left\_frame.place(x=10,y=10,width=600,height=450)

        img\_left=Image.open(r"C:\Users\HP\Desktop\minner project\b66.jpeg")

        img\_left=img\_left.resize((600,100),Image.ANTIALIAS)

        self.photoimg\_left=ImageTk.PhotoImage(img\_left)

        f\_lbl=Label(self.root,image=self.photoimg\_left)

        f\_lbl.place(x=33,y=200,width=587,height=90)

        #current courses

        current\_position\_frame=LabelFrame(main\_frame,bd=2, bg="white",relief=RIDGE,text="current Work information",font=("times new roman ",12,"bold"))

        current\_position\_frame.place(x=20,y=120,width=580,height=110)

        #department

        dep\_label=Label(current\_position\_frame,text="department",font=("times new roman",12,"bold"),bg="white")

        dep\_label.grid(row=0,column=0,padx=10,sticky=W)

        dep\_combo=ttk.Combobox(current\_position\_frame, textvariable=self.var\_dep, font=("times new roman",12,"bold"),state="read only",width=17)

        dep\_combo["values"]=("select Department","computer","it","civil","mechnical")

        dep\_combo.current(0)

        dep\_combo.grid(row=0,column=1,padx=2,pady=10,sticky=W)

        # course

        position\_label=Label(current\_position\_frame,text="courses",font=("times new roman",12,"bold"),bg="white")

        position\_label.grid(row=0,column=2,padx=10,sticky=W)

        position\_combo=ttk.Combobox(current\_position\_frame,textvariable=self.var\_Position ,font=("times new roman",12,"bold"),state="read only",width=17)

        position\_combo["values"]=("select Department","computer","it","civil","mechnical")

        position\_combo.current(0)

        position\_combo.grid(row=0,column=3,padx=2,pady=10,sticky=W)

        # year

        year\_label=Label(current\_position\_frame,text="year",font=("times new roman",12,"bold"),bg="white")

        year\_label.grid(row=1,column=0,padx=10,sticky=W)

        year\_combo=ttk.Combobox(current\_position\_frame,textvariable=self.var\_year ,font=("times new roman",12,"bold"),state="read only",width=17)

        year\_combo["values"]=("select year","2020-21","2021-22","2022-23","2023-24")

        year\_combo.current(0)

        year\_combo.grid(row=1,column=1,padx=2,pady=10,sticky=W)

         #semester

        semester\_label=Label(current\_position\_frame,text="semester",font=("times new roman",12,"bold"),bg="white")

        semester\_label.grid(row=1,column=2,padx=10,sticky=W)

        semester\_combo=ttk.Combobox(current\_position\_frame,textvariable=self.var\_semester, font=("times new roman",12,"bold"),state="read only",width=17)

        semester\_combo["values"]=("select sem","1st","2nd","3rd","4th","5th","6th")

        semester\_combo.current(0)

        semester\_combo.grid(row=1,column=3,padx=2,pady=10,sticky=W)

        #company employee information

        company\_employee\_frame=LabelFrame(main\_frame,bd=2, bg="white",relief=RIDGE,text="employee information",font=("times new roman ",12,"bold"))

        company\_employee\_frame.place(x=20,y=240,width=580,height=210)

        #employee ID

        employeeId\_label=Label(company\_employee\_frame,text="employeeId:",font=("times new roman",12,"bold"),bg="white")

        employeeId\_label.grid(row=0,column=0,padx=10,pady=7,sticky=W)

        employeeId\_entry=ttk.Entry(company\_employee\_frame,textvariable=self.var\_id ,width=16,font=("times new roman",12,"bold"))

        employeeId\_entry.grid(row=0,column=1,padx=5,pady=2,sticky=W)

        #employee Name

        employeeName\_label=Label(company\_employee\_frame,text="employeeName:",font=("times new roman",12,"bold"),bg="white")

        employeeName\_label.grid(row=0,column=2,padx=10,pady=5,sticky=W)

        employeeName\_entry=ttk.Entry(company\_employee\_frame,textvariable=self.var\_name ,width=16,font=("times new roman",12,"bold"))

        employeeName\_entry.grid(row=0,column=3,padx=5,pady=2,sticky=W)

        #employee Gender

        employeeGender\_label=Label(company\_employee\_frame,text="employeeGender:",font=("times new roman",12,"bold"),bg="white")

        employeeGender\_label.grid(row=1,column=0,padx=10,pady=5,sticky=W)

        employeeGender\_entry=ttk.Entry(company\_employee\_frame, textvariable=self.var\_gender ,width=16,font=("times new roman",12,"bold"))

        employeeGender\_entry.grid(row=1,column=1,padx=5,pady=2,sticky=W)

        #employee age

        employeeage\_label=Label(company\_employee\_frame,text="employeeage:",font=("times new roman",12,"bold"),bg="white")

        employeeage\_label.grid(row=1,column=2,padx=10,pady=5,sticky=W)

        employeeage\_entry=ttk.Entry(company\_employee\_frame,textvariable=self.var\_age ,width=16,font=("times new roman",12,"bold"))

        employeeage\_entry.grid(row=1,column=3,padx=5,pady=2,sticky=W)

        #employee address

        employeeaddress\_label=Label(company\_employee\_frame,text="employeeaddress:",font=("times new roman",12,"bold"),bg="white")

        employeeaddress\_label.grid(row=1,column=0,padx=10,pady=7,sticky=W)

        employeeaddress\_entry=ttk.Entry(company\_employee\_frame,textvariable=self.var\_address ,width=16,font=("times new roman",12,"bold"))

        employeeaddress\_entry.grid(row=1,column=1,padx=5,pady=2,sticky=W)

        #employee Email

        employeeEmail\_label=Label(company\_employee\_frame,text="employeeEmail:",font=("times new roman",12,"bold"),bg="white")

        employeeEmail\_label.grid(row=2,column=0,padx=10,pady=5,sticky=W)

        employeeEmail\_entry=ttk.Entry(company\_employee\_frame,textvariable=self.var\_email ,width=16,font=("times new roman",12,"bold"))

        employeeEmail\_entry.grid(row=2,column=1,padx=5,pady=2,sticky=W)

        #employee PhNo

        employeePhNo\_label=Label(company\_employee\_frame,text="employeePhNo:",font=("times new roman",12,"bold"),bg="white")

        employeePhNo\_label.grid(row=2,column=2,padx=10,pady=5,sticky=W)

        employeePhNo\_entry=ttk.Entry(company\_employee\_frame,textvariable=self.var\_phone ,width=16,font=("times new roman",12,"bold"))

        employeePhNo\_entry.grid(row=2,column=3,padx=5,pady=2,sticky=W)

        #radio Button

        self.var\_radio1=StringVar()

        Radiobtn1=ttk.Radiobutton(company\_employee\_frame,variable=self.var\_radio1, text="take photo sample",value="yes")

        Radiobtn1.grid(row=6,column=0)

        self.var\_radio2=StringVar()

        Radiobtn2=ttk.Radiobutton(company\_employee\_frame,variable=self.var\_radio2,text="No photo sample",value="No")

        Radiobtn2.grid(row=6,column=1)

        #bbuttons frame

        btn\_frame=Frame(company\_employee\_frame,bd=2,relief=RIDGE,bg="white")

        btn\_frame.place(x=0,y=135,width=575,height=50)

        save\_btn=Button(btn\_frame,text="save",command=self.add\_data ,width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        save\_btn.grid(row=0,column=0)

        update\_btn=Button(btn\_frame,text="update",width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        update\_btn.grid(row=0,column=1)

        delete\_btn=Button(btn\_frame,text="delete",width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        delete\_btn.grid(row=0,column=2)

        reset\_btn=Button(btn\_frame,text="reset",width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        reset\_btn.grid(row=0,column=3)

        take\_photo\_btn=Button(btn\_frame,text="Take photo sample",width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        take\_photo\_btn.grid(row=1,column=0)

        update\_photo\_btn=Button(btn\_frame,text="update photo sample",width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        update\_photo\_btn.grid(row=1,column=1)

        #Right lable frame

        Right\_frame=LabelFrame(main\_frame,bd=2, bg="white",relief=RIDGE,text="employee Details",font=("times new roman ",12,"bold"))

        Right\_frame.place(x=650,y=10,width=590,height=450)

        img\_right=Image.open(r"C:\Users\HP\Desktop\minner project\b.jpeg")

        img\_right=img\_right.resize((600,100),Image.ANTIALIAS)

        self.photoimg\_right=ImageTk.PhotoImage(img\_right)

        f\_lbl=Label(self.root,image=self.photoimg\_right)

        f\_lbl.place(x=670,y=200,width=580,height=90)

        #\*\*\*\*\*\*\*\*\*\*search system\*\*\*\*\*\*\*\*\*\*\*\*\*

        search\_frame=LabelFrame(Right\_frame,bd=2, bg="white",relief=RIDGE,text="search system",font=("times new roman ",12,"bold"))

        search\_frame.place(x=5,y=100,width=575,height=70)

        search\_label=Label(search\_frame,text="Search By:",font=("times new roman",13,"bold"),bg="white")

        search\_label.grid(row=0,column=0,padx=10,pady=2,sticky=W)

        search\_combo=ttk.Combobox(search\_frame,font=("times new roman",12,"bold"),state="read only",width=10)

        search\_combo["values"]=("select","1st","emp\_id","phone\_no")

        search\_combo.current(0)

        search\_combo.grid(row=0,column=1,padx=2,pady=10,sticky=W)

        search\_entry=ttk.Entry(search\_frame,width=14,font=("times new roman",12,"bold"))

        search\_entry.grid(row=0,column=2,padx=10,pady=5,sticky=W)

        search\_btn=Button(search\_frame,text="search",width=10,font=("times new roman",10,"bold"),bg="blue",fg="white")

        search\_btn.grid(row=0,column=3,padx=4)

        search\_btn=Button(search\_frame,text="show All",width=10,font=("times new roman",10,"bold"),bg="blue",fg="white")

        search\_btn.grid(row=0,column=4,padx=4)

        table\_frame=Frame(Right\_frame,bg="white",bd=2,relief=RIDGE)

        table\_frame.place(x=5,y=180,width=575,height=240)

        scroll\_x=ttk.Scrollbar(table\_frame,orient=HORIZONTAL)

        scroll\_y=ttk.Scrollbar(table\_frame,orient=VERTICAL)

        self.employee\_table=ttk.Treeview(table\_frame,column=("dep","course","year","sem","id","name","gender","age","email","phone","address","photo"),xscrollcommand=scroll\_x.set,yscrollcommand=scroll\_y.set)

        scroll\_x.pack(side=BOTTOM,fill=X)

        scroll\_y.pack(side=RIGHT,fill=Y)

        scroll\_x.config(command=self.employee\_table.xview)

        scroll\_y.config(command=self.employee\_table.yview)

        self.employee\_table.heading("dep",text="Department")

        self.employee\_table.heading("course",text="courses")

        self.employee\_table.heading("year",text="year of joining")

        self.employee\_table.heading("sem",text="semester")

        self.employee\_table.heading("id",text="id")

        self.employee\_table.heading("gender",text="gender")

        self.employee\_table.heading("age",text="age")

        self.employee\_table.heading("email",text="email")

        self.employee\_table.heading("phone",text="Phone no")

        self.employee\_table.heading("address",text="address")

        self.employee\_table.heading("photo",text="photo")

        self.employee\_table["show"]="headings"

        self.employee\_table.column("dep",width=100)

        self.employee\_table.column("course",width=100)

        self.employee\_table.column("year",width=100)

        self.employee\_table.column("sem",width=100)

        self.employee\_table.column("id",width=100)

        self.employee\_table.column("gender",width=100)

        self.employee\_table.column("age",width=100)

        self.employee\_table.column("email",width=100)

        self.employee\_table.column("phone",width=100)

        self.employee\_table.column("address",width=100)

        self.employee\_table.column("photo",width=150)

        self.employee\_table.pack(fill=BOTH,expand=1)

    def add\_data(self):

        if self.var\_dep.get()=="select Department" or self.var\_name.get()=="" or self.var\_id.get():

            messagebox.showerror("Error ", "All the fields are required",parent=self.root)

        else:

            pass

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

    root=Tk()

    obj=Employee(root)

    root.mainloop()

**data fetching coad set delete update etc**

from tkinter import\*

from tkinter import ttk

from tkinter.tix import TEXT

from turtle import left, st, update

from PIL import Image,ImageTk

from tkinter import messagebox

from colorama import Cursor

import mysql.connector

class Employee:

    def \_\_init\_\_(self,root):

        self.root=root

        self.root.geometry("1200x790+0+0")

        self.root.title("face recogniton system")

        # \*\*\*\*\*\*\*\*\*\*\*\*\*\*variables\*\*\*\*\*\*\*\*\*\*\*

        self.var\_dep=StringVar()

        self.var\_City=StringVar()

        self.var\_year=StringVar()

        self.var\_Shift=StringVar()

        self.var\_employeeid=StringVar()

        self.var\_name=StringVar()

        #self.var\_gender=StringVar()

        self.var\_age=StringVar()

        self.var\_email=StringVar()

        self.var\_phone=StringVar()

        self.var\_address=StringVar()

        #first image

        img1=Image.open(r"C:\Users\HP\Desktop\minner project\sss.jpeg")

        img1=img1.resize((500,130),Image.ANTIALIAS)

        self.photoimg1=ImageTk.PhotoImage(img1)

        f\_lbl=Label(self.root,image=self.photoimg1)

        f\_lbl.place(x=0,y=0,width=500,height=120)

        #2 image

        img2=Image.open(r"C:\Users\HP\Desktop\minner project\ss.jpeg")

        img2=img2.resize((500,130),Image.ANTIALIAS)

        self.photoimg2=ImageTk.PhotoImage(img2)

        f\_lbl=Label(self.root,image=self.photoimg2)

        f\_lbl.place(x=500,y=0,width=500,height=120)

        #3 image

        img3=Image.open(r"C:\Users\HP\Desktop\minner project\ssss.jpeg")

        img3=img3.resize((450,130),Image.ANTIALIAS)

        self.photoimg3=ImageTk.PhotoImage(img3)

        f\_lbl=Label(self.root,image=self.photoimg3)

        f\_lbl.place(x=1000,y=0,width=450,height=120)

        #bg image

        img4=Image.open(r"C:\Users\HP\Desktop\minner project\b77.jpeg")

        img4=img4.resize((1400,710),Image.ANTIALIAS)

        self.photoimg4=ImageTk.PhotoImage(img4)

        bg\_img=Label(self.root,image=self.photoimg4)

        bg\_img.place(x=0,y=120,width=1400,height=710)

        title\_lbl=Label(bg\_img,text="EMPLOYEE ATTENDANCE MANAGMENT SYSTEM ",font=("times new roman",28,"bold"),bg="white",fg="dark red")

        title\_lbl.place(x=0,y=0,width=1430,height=35)

        main\_frame=Frame(bg\_img,bd=2, bg="white")

        main\_frame.place(x=10,y=45,width=1250,height=470)

        #left lable frame

        left\_frame=LabelFrame(main\_frame,bd=2, bg="white",relief=RIDGE,text="employee Details",font=("times new roman ",12,"bold"))

        left\_frame.place(x=10,y=10,width=600,height=450)

        img\_left=Image.open(r"C:\Users\HP\Desktop\minner project\b66.jpeg")

        img\_left=img\_left.resize((600,100),Image.ANTIALIAS)

        self.photoimg\_left=ImageTk.PhotoImage(img\_left)

        f\_lbl=Label(self.root,image=self.photoimg\_left)

        f\_lbl.place(x=33,y=200,width=587,height=90)

        #current courses

        current\_position\_frame=LabelFrame(main\_frame,bd=2, bg="white",relief=RIDGE,text="current Work information",font=("times new roman ",12,"bold"))

        current\_position\_frame.place(x=20,y=120,width=580,height=110)

        #department

        dep\_label=Label(current\_position\_frame,text="department",font=("times new roman",12,"bold"),bg="white")

        dep\_label.grid(row=0,column=0,padx=10,sticky=W)

        dep\_combo=ttk.Combobox(current\_position\_frame, textvariable=self.var\_dep, font=("times new roman",12,"bold"),state="read only",width=17)

        dep\_combo["values"]=("select Department","computer","it","civil","mechnical")

        dep\_combo.current(0)

        dep\_combo.grid(row=0,column=1,padx=2,pady=10,sticky=W)

        # City

        City\_label=Label(current\_position\_frame,text="City",font=("times new roman",12,"bold"),bg="white")

        City\_label.grid(row=0,column=2,padx=10,sticky=W)

        City\_combo=ttk.Combobox(current\_position\_frame,textvariable=self.var\_City ,font=("times new roman",12,"bold"),state="read only",width=17)

        City\_combo["values"]=("select City","Dewas","Indore","Ujjain","Delhi","Bhopal")

        City\_combo.current(0)

        City\_combo.grid(row=0,column=3,padx=2,pady=10,sticky=W)

        # year

        year\_label=Label(current\_position\_frame,text="year",font=("times new roman",12,"bold"),bg="white")

        year\_label.grid(row=1,column=0,padx=10,sticky=W)

        year\_combo=ttk.Combobox(current\_position\_frame,textvariable=self.var\_year ,font=("times new roman",12,"bold"),state="read only",width=17)

        year\_combo["values"]=("select year","2020-21","2021-22","2022-23","2023-24")

        year\_combo.current(0)

        year\_combo.grid(row=1,column=1,padx=2,pady=10,sticky=W)

         #semester

        Shift\_label=Label(current\_position\_frame,text="Shift",font=("times new roman",12,"bold"),bg="white")

        Shift\_label.grid(row=1,column=2,padx=10,sticky=W)

        Shift\_combo=ttk.Combobox(current\_position\_frame,textvariable=self.var\_Shift, font=("times new roman",12,"bold"),state="read only",width=17)

        Shift\_combo["values"]=("select Shift","1st","2nd")

        Shift\_combo.current(0)

        Shift\_combo.grid(row=1,column=3,padx=2,pady=10,sticky=W)

        #company employee information

        company\_employee\_frame=LabelFrame(main\_frame,bd=2, bg="white",relief=RIDGE,text="employee information",font=("times new roman ",12,"bold"))

        company\_employee\_frame.place(x=20,y=240,width=580,height=210)

        #employee ID

        employeeId\_label=Label(company\_employee\_frame,text="employeeId:",font=("times new roman",12,"bold"),bg="white")

        employeeId\_label.grid(row=0,column=0,padx=10,pady=7,sticky=W)

        employeeId\_entry=ttk.Entry(company\_employee\_frame,textvariable=self.var\_employeeid ,width=16,font=("times new roman",12,"bold"))

        employeeId\_entry.grid(row=0,column=1,padx=5,pady=2,sticky=W)

        #employee Name

        employeeName\_label=Label(company\_employee\_frame,text="employeeName:",font=("times new roman",12,"bold"),bg="white")

        employeeName\_label.grid(row=0,column=2,padx=10,pady=5,sticky=W)

        employeeName\_entry=ttk.Entry(company\_employee\_frame,textvariable=self.var\_name ,width=16,font=("times new roman",12,"bold"))

        employeeName\_entry.grid(row=0,column=3,padx=5,pady=2,sticky=W)

        # #employee Gender

        # employeeGender\_label=Label(company\_employee\_frame,text="employeeGender:",font=("times new roman",12,"bold"),bg="white")

        # employeeGender\_label.grid(row=1,column=0,padx=10,pady=5,sticky=W)

        # employeeGender\_entry=ttk.Entry(company\_employee\_frame, textvariable=self.var\_gender ,width=16,font=("times new roman",12,"bold"))

        # employeeGender\_entry.grid(row=1,column=1,padx=5,pady=2,sticky=W)

        #employee age

        employeeage\_label=Label(company\_employee\_frame,text="employeeage:",font=("times new roman",12,"bold"),bg="white")

        employeeage\_label.grid(row=1,column=2,padx=10,pady=5,sticky=W)

        employeeage\_entry=ttk.Entry(company\_employee\_frame,textvariable=self.var\_age ,width=16,font=("times new roman",12,"bold"))

        employeeage\_entry.grid(row=1,column=3,padx=5,pady=2,sticky=W)

        #employee address

        employeeaddress\_label=Label(company\_employee\_frame,text="employeeaddress:",font=("times new roman",12,"bold"),bg="white")

        employeeaddress\_label.grid(row=1,column=0,padx=10,pady=7,sticky=W)

        employeeaddress\_entry=ttk.Entry(company\_employee\_frame,textvariable=self.var\_address ,width=16,font=("times new roman",12,"bold"))

        employeeaddress\_entry.grid(row=1,column=1,padx=5,pady=2,sticky=W)

        #employee Email

        employeeEmail\_label=Label(company\_employee\_frame,text="employeeEmail:",font=("times new roman",12,"bold"),bg="white")

        employeeEmail\_label.grid(row=2,column=0,padx=10,pady=5,sticky=W)

        employeeEmail\_entry=ttk.Entry(company\_employee\_frame,textvariable=self.var\_email ,width=16,font=("times new roman",12,"bold"))

        employeeEmail\_entry.grid(row=2,column=1,padx=5,pady=2,sticky=W)

        #employee PhNo

        employeePhNo\_label=Label(company\_employee\_frame,text="employeePhNo:",font=("times new roman",12,"bold"),bg="white")

        employeePhNo\_label.grid(row=2,column=2,padx=10,pady=5,sticky=W)

        employeePhNo\_entry=ttk.Entry(company\_employee\_frame,textvariable=self.var\_phone ,width=16,font=("times new roman",12,"bold"))

        employeePhNo\_entry.grid(row=2,column=3,padx=5,pady=2,sticky=W)

        #radio Button

        self.var\_radio1=StringVar()

        Radiobtn1=ttk.Radiobutton(company\_employee\_frame,variable=self.var\_radio1, text="take photo sample",value="yes")

        Radiobtn1.grid(row=6,column=0)

        Radiobtn2=ttk.Radiobutton(company\_employee\_frame,variable=self.var\_radio1,text="No photo sample",value="No")

        Radiobtn2.grid(row=6,column=1)

        #bbuttons frame

        btn\_frame=Frame(company\_employee\_frame,bd=2,relief=RIDGE,bg="white")

        btn\_frame.place(x=0,y=135,width=575,height=50)

        save\_btn=Button(btn\_frame,text="save",command=self.add\_data ,width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        save\_btn.grid(row=0,column=0)

        update\_btn=Button(btn\_frame,text="update",width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        update\_btn.grid(row=0,column=1)

        delete\_btn=Button(btn\_frame,text="delete",command=self.delete\_data, width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        delete\_btn.grid(row=0,column=2)

        reset\_btn=Button(btn\_frame,text="reset",command=self.reset\_data, width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        reset\_btn.grid(row=0,column=3)

        take\_photo\_btn=Button(btn\_frame,text="Take photo sample",width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        take\_photo\_btn.grid(row=1,column=0)

        update\_photo\_btn=Button(btn\_frame,text="update photo sample",width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        update\_photo\_btn.grid(row=1,column=1)

        #Right lable frame

        Right\_frame=LabelFrame(main\_frame,bd=2, bg="white",relief=RIDGE,text="employee Details",font=("times new roman ",12,"bold"))

        Right\_frame.place(x=650,y=10,width=590,height=450)

        img\_right=Image.open(r"C:\Users\HP\Desktop\minner project\b.jpeg")

        img\_right=img\_right.resize((600,100),Image.ANTIALIAS)

        self.photoimg\_right=ImageTk.PhotoImage(img\_right)

        f\_lbl=Label(self.root,image=self.photoimg\_right)

        f\_lbl.place(x=670,y=200,width=580,height=90)

        #\*\*\*\*\*\*\*\*\*\*search system\*\*\*\*\*\*\*\*\*\*\*\*\*

        search\_frame=LabelFrame(Right\_frame,bd=2, bg="white",relief=RIDGE,text="search system",font=("times new roman ",12,"bold"))

        search\_frame.place(x=5,y=100,width=575,height=70)

        search\_label=Label(search\_frame,text="Search By:",font=("times new roman",13,"bold"),bg="white")

        search\_label.grid(row=0,column=0,padx=10,pady=2,sticky=W)

        search\_combo=ttk.Combobox(search\_frame,font=("times new roman",12,"bold"),state="read only",width=10)

        search\_combo["values"]=("select","1st","emp\_id","phone\_no")

        search\_combo.current(0)

        search\_combo.grid(row=0,column=1,padx=2,pady=10,sticky=W)

        search\_entry=ttk.Entry(search\_frame,width=14,font=("times new roman",12,"bold"))

        search\_entry.grid(row=0,column=2,padx=10,pady=5,sticky=W)

        search\_btn=Button(search\_frame,text="search",width=10,font=("times new roman",10,"bold"),bg="blue",fg="white")

        search\_btn.grid(row=0,column=3,padx=4)

        search\_btn=Button(search\_frame,text="show All",width=10,font=("times new roman",10,"bold"),bg="blue",fg="white")

        search\_btn.grid(row=0,column=4,padx=4)

        table\_frame=Frame(Right\_frame,bg="white",bd=2,relief=RIDGE)

        table\_frame.place(x=5,y=180,width=575,height=240)

        scroll\_x=ttk.Scrollbar(table\_frame,orient=HORIZONTAL)

        scroll\_y=ttk.Scrollbar(table\_frame,orient=VERTICAL)

        self.employee\_table=ttk.Treeview(table\_frame,column=("dep","City","year","Shift","employeeId","employeeName","address","age","email","phone"),xscrollcommand=scroll\_x.set,yscrollcommand=scroll\_y.set)

        scroll\_x.pack(side=BOTTOM,fill=X)

        scroll\_y.pack(side=RIGHT,fill=Y)

        scroll\_x.config(command=self.employee\_table.xview)

        scroll\_y.config(command=self.employee\_table.yview)

        self.employee\_table.heading("dep",text="Department")

        self.employee\_table.heading("City",text="City")

        self.employee\_table.heading("year",text="year")

        self.employee\_table.heading("Shift",text="Shift")

        self.employee\_table.heading("employeeId",text="employeeId")

        self.employee\_table.heading("employeeName",text="employeeName")

        self.employee\_table.heading("address",text="employeeaddress")

        self.employee\_table.heading("age",text="employeeage")

        self.employee\_table.heading("email",text="employeeEmail")

        self.employee\_table.heading("phone",text="employeePhNo")

        #self.employee\_table.heading("photo",text="photo")

        self.employee\_table["show"]="headings"

        self.employee\_table.column("dep",width=100)

        self.employee\_table.column("City",width=100)

        self.employee\_table.column("year",width=100)

        self.employee\_table.column("Shift",width=100)

        self.employee\_table.column("employeeId",width=100)

        self.employee\_table.column("employeeName",width=100)

        self.employee\_table.column("address",width=100)

        self.employee\_table.column("age",width=100)

        self.employee\_table.column("email",width=100)

        self.employee\_table.column("phone",width=100)

        #self.employee\_table.column("photo",width=150)

        self.employee\_table.pack(fill=BOTH,expand=1)

        self.employee\_table.bind("ButtonRelease",self.get\_cursor)

        #self.fetch\_data()

    def add\_data(self):

        if self.var\_dep.get()=="select Department" or self.var\_name.get()=="" or self.var\_employeeid.get()=="":

            messagebox.showerror("Error ", "All the fields are required",parent=self.root)

        else:

            try:

                messagebox.showinfo("success","welcome")

                c=mysql.connector.connect(host="localhost",username="root",password="SHpatel@9893",database="face\_recognizer")

                my\_cursor=c.cursor()

                messagebox.showinfo("success","welcome2")

                my\_cursor.execute("insert into employee values(%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)",[    self.var\_dep.get(),

                                                                                                    self.var\_City.get(),

                                                                                                    self.var\_year.get(),

                                                                                                    self.var\_Shift.get(),

                                                                                                    self.var\_employeeid.get(),

                                                                                                    self.var\_name.get(),

                                                                                                    self.var\_address.get(),

                                                                                                    self.var\_age.get(),

                                                                                                    self.var\_email.get(),

                                                                                                    self.var\_phone.get()

                                                                                                                          ])

            #                                                                                        # self.var\_radio1.get()

            #

                # val=[self.var\_employeeid.get()]                                                                                # self.var\_radio1.get()

                # my\_cursor.execute("insert into employee(employeeid) values(%s)",val)

                messagebox.showinfo("success","welcome2")                                                                             #))

                c.commit()

                #self.fetch\_data()

                c.close()

                print("successfull")

                #messagebox.showinfo("success","employee details has been added successfully",parent=self.root)

            except Exception as es:

                # messagebox.showerror("Error",f"due To :{str(es)}",parent=self.root)

                print(es)

    #\*\*\*\*\*\*\*\*\*\*fetch data \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

    # def fetch\_data(self):

    #     conn=mysql.connector.connect(host="localhost",username="root",Password="SHpatel@9893",database="face\_recognizer")

    #     my\_cursor=conn.cursor()

    #     my\_cursor.execute("select \* from employee")

    #     data=my\_cursor.fetchall()

    #     if len(data)!=0:

    #         self.employee\_table.delete(\*self.employee\_table.get\_children())

    #         for i in data:

    #             self.employee\_table.insert("",END,values=i)

    #         conn.commit()

    #     conn.close()

    #=============get curser===============

    def get\_cursor(self,event=""):

        Cursor\_focus=self.employee\_table.focus()

        content=self.employee\_table.item(Cursor\_focus)

        data=content["values"]

        self.var\_dep.set(data[0]),

        self.var\_City.set(data[1]),

        self.var\_year.set(data[2]),

        self.var\_Shift.set(data[3]),

        self.var\_employeeid.set(data[4]),

        self.var\_name.set(data[5]),

        self.var\_address.set(data[6]),

        self.var\_age.set(data[7]),

        self.var\_email.set(data[8]),

        self.var\_phone.set(data[9])

    # #========update function======

    def update\_data(self):

        if self.var\_dep.get()=="select Department" or self.var\_name.get()=="" or self.var\_employeeid.get()=="":

            messagebox.showerror("Error ", "All the fields are required",parent=self.root)

        else:

            try:

                update=messagebox.askyesno("update","do you want to update this employee details",paeent=self.root)

                if update>0:

                    c=mysql.connector.connect(host="localhost",username="root",password="SHpatel@9893",database="face\_recognizer")

                    my\_cursor=c.cursor()

                    my\_cursor.execute("update employee set Department=%s,City=%s,year=%s,Shift=%s,employeename=%s,employeeaddres=%s,employeeage=%s,email=%s,phone=%s where ,employeeId=%s",[

                                                                                                    self.var\_dep.get(),

                                                                                                    self.var\_City.get(),

                                                                                                    self.var\_year.get(),

                                                                                                    self.var\_Shift.get(),

                                                                                                    self.var\_name.get(),

                                                                                                    self.var\_address.get(),

                                                                                                    self.var\_age.get(),

                                                                                                    self.var\_email.get(),

                                                                                                    self.var\_phone.get(),

                                                                                                    self.var\_employeeid.get()

                    ])

                else:

                    if not update:

                        return

                messagebox.showinfo("success","employee details successfully updated completely",parent=self.root)

                c.commit()

                #self.fetch\_data()

                c.close()

            except Exception as es:

                messagebox.showerror("Error",f"Due to:{str(es)}",parent=self.root)

    #delete function

    def delete\_data(self):

        if self.var\_employeeid.get()=="":

            messagebox.showerror("error","enployee id must be required",parent=self.root)

        else:

            try:

                delete=messagebox.askyesno("Employee delete page","do you you want to delete this employee",parent=self.root)

                if delete>0:

                     c=mysql.connector.connect(host="localhost",username="root",password="SHpatel@9893",database="face\_recognizer")

                     my\_cursor=c.cursor()

                     sql="delete from employee where employeeId=%s"

                     val=(self.var\_employeeid.get(),)

                     my\_cursor.execute(sql,val)

                else:

                    if not delete:

                        return

                c.commit()

                #self.fetch\_data()

                c.close()

                messagebox.showinfo("delete","succesfully delete details",parent=self.root)

            except Exception as es:

                messagebox.showerror("Error",f"Due to:{str(es)}",parent=self.root)

    #reset

    def reset\_data(self):

        self.var\_dep.set("select Department")

        self.var\_City.set("select City")

        self.var\_year.set("select year")

        self.var\_Shift.set("select Shift")

        self.var\_employeeid.set("")

        self.var\_name.set("")

        self.var\_address.set("")

        self.var\_age.set("")

        self.var\_email.set("")

        self.var\_phone.set("")

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

    root=Tk()

    obj=Employee(root)

    root.mainloop()

**take photo sample (4th  video)**

from tkinter import\*

from tkinter import ttk

from tkinter.tix import TEXT

from turtle import left, st, update

from PIL import Image,ImageTk

from tkinter import messagebox

from colorama import Cursor

import mysql.connector

import cv2

class Employee:

    def \_\_init\_\_(self,root):

        self.root=root

        self.root.geometry("1200x790+0+0")

        self.root.title("face recogniton system")

        # \*\*\*\*\*\*\*\*\*\*\*\*\*\*variables\*\*\*\*\*\*\*\*\*\*\*

        self.var\_dep=StringVar()

        self.var\_City=StringVar()

        self.var\_year=StringVar()

        self.var\_Shift=StringVar()

        self.var\_employeeid=StringVar()

        self.var\_name=StringVar()

        #self.var\_gender=StringVar()

        self.var\_age=StringVar()

        self.var\_email=StringVar()

        self.var\_phone=StringVar()

        self.var\_address=StringVar()

        #first image

        img1=Image.open(r"C:\Users\HP\Desktop\minner project\sss.jpeg")

        img1=img1.resize((500,130),Image.ANTIALIAS)

        self.photoimg1=ImageTk.PhotoImage(img1)

        f\_lbl=Label(self.root,image=self.photoimg1)

        f\_lbl.place(x=0,y=0,width=500,height=120)

        #2 image

        img2=Image.open(r"C:\Users\HP\Desktop\minner project\ss.jpeg")

        img2=img2.resize((500,130),Image.ANTIALIAS)

        self.photoimg2=ImageTk.PhotoImage(img2)

        f\_lbl=Label(self.root,image=self.photoimg2)

        f\_lbl.place(x=500,y=0,width=500,height=120)

        #3 image

        img3=Image.open(r"C:\Users\HP\Desktop\minner project\ssss.jpeg")

        img3=img3.resize((450,130),Image.ANTIALIAS)

        self.photoimg3=ImageTk.PhotoImage(img3)

        f\_lbl=Label(self.root,image=self.photoimg3)

        f\_lbl.place(x=1000,y=0,width=450,height=120)

        #bg image

        img4=Image.open(r"C:\Users\HP\Desktop\minner project\b77.jpeg")

        img4=img4.resize((1400,710),Image.ANTIALIAS)

        self.photoimg4=ImageTk.PhotoImage(img4)

        bg\_img=Label(self.root,image=self.photoimg4)

        bg\_img.place(x=0,y=120,width=1400,height=710)

        title\_lbl=Label(bg\_img,text="EMPLOYEE ATTENDANCE MANAGMENT SYSTEM ",font=("times new roman",28,"bold"),bg="white",fg="dark red")

        title\_lbl.place(x=0,y=0,width=1430,height=35)

        main\_frame=Frame(bg\_img,bd=2, bg="white")

        main\_frame.place(x=10,y=45,width=1250,height=470)

        #left lable frame

        left\_frame=LabelFrame(main\_frame,bd=2, bg="white",relief=RIDGE,text="employee Details",font=("times new roman ",12,"bold"))

        left\_frame.place(x=10,y=10,width=600,height=450)

        img\_left=Image.open(r"C:\Users\HP\Desktop\minner project\b66.jpeg")

        img\_left=img\_left.resize((600,100),Image.ANTIALIAS)

        self.photoimg\_left=ImageTk.PhotoImage(img\_left)

        f\_lbl=Label(self.root,image=self.photoimg\_left)

        f\_lbl.place(x=33,y=200,width=587,height=90)

        #current courses

        current\_position\_frame=LabelFrame(main\_frame,bd=2, bg="white",relief=RIDGE,text="current Work information",font=("times new roman ",12,"bold"))

        current\_position\_frame.place(x=20,y=120,width=580,height=110)

        #department

        dep\_label=Label(current\_position\_frame,text="department",font=("times new roman",12,"bold"),bg="white")

        dep\_label.grid(row=0,column=0,padx=10,sticky=W)

        dep\_combo=ttk.Combobox(current\_position\_frame, textvariable=self.var\_dep, font=("times new roman",12,"bold"),state="read only",width=17)

        dep\_combo["values"]=("select Department","computer","it","civil","mechnical")

        dep\_combo.current(0)

        dep\_combo.grid(row=0,column=1,padx=2,pady=10,sticky=W)

        # City

        City\_label=Label(current\_position\_frame,text="City",font=("times new roman",12,"bold"),bg="white")

        City\_label.grid(row=0,column=2,padx=10,sticky=W)

        City\_combo=ttk.Combobox(current\_position\_frame,textvariable=self.var\_City ,font=("times new roman",12,"bold"),state="read only",width=17)

        City\_combo["values"]=("select City","Dewas","Indore","Ujjain","Delhi","Bhopal")

        City\_combo.current(0)

        City\_combo.grid(row=0,column=3,padx=2,pady=10,sticky=W)

        # year

        year\_label=Label(current\_position\_frame,text="year",font=("times new roman",12,"bold"),bg="white")

        year\_label.grid(row=1,column=0,padx=10,sticky=W)

        year\_combo=ttk.Combobox(current\_position\_frame,textvariable=self.var\_year ,font=("times new roman",12,"bold"),state="read only",width=17)

        year\_combo["values"]=("select year","2020-21","2021-22","2022-23","2023-24")

        year\_combo.current(0)

        year\_combo.grid(row=1,column=1,padx=2,pady=10,sticky=W)

         #semester

        Shift\_label=Label(current\_position\_frame,text="Shift",font=("times new roman",12,"bold"),bg="white")

        Shift\_label.grid(row=1,column=2,padx=10,sticky=W)

        Shift\_combo=ttk.Combobox(current\_position\_frame,textvariable=self.var\_Shift, font=("times new roman",12,"bold"),state="read only",width=17)

        Shift\_combo["values"]=("select Shift","1st","2nd")

        Shift\_combo.current(0)

        Shift\_combo.grid(row=1,column=3,padx=2,pady=10,sticky=W)

        #company employee information

        company\_employee\_frame=LabelFrame(main\_frame,bd=2, bg="white",relief=RIDGE,text="employee information",font=("times new roman ",12,"bold"))

        company\_employee\_frame.place(x=20,y=240,width=580,height=210)

        #employee ID

        employeeId\_label=Label(company\_employee\_frame,text="employeeId:",font=("times new roman",12,"bold"),bg="white")

        employeeId\_label.grid(row=0,column=0,padx=10,pady=7,sticky=W)

        employeeId\_entry=ttk.Entry(company\_employee\_frame,textvariable=self.var\_employeeid ,width=16,font=("times new roman",12,"bold"))

        employeeId\_entry.grid(row=0,column=1,padx=5,pady=2,sticky=W)

        #employee Name

        employeeName\_label=Label(company\_employee\_frame,text="employeeName:",font=("times new roman",12,"bold"),bg="white")

        employeeName\_label.grid(row=0,column=2,padx=10,pady=5,sticky=W)

        employeeName\_entry=ttk.Entry(company\_employee\_frame,textvariable=self.var\_name ,width=16,font=("times new roman",12,"bold"))

        employeeName\_entry.grid(row=0,column=3,padx=5,pady=2,sticky=W)

        # #employee Gender

        # employeeGender\_label=Label(company\_employee\_frame,text="employeeGender:",font=("times new roman",12,"bold"),bg="white")

        # employeeGender\_label.grid(row=1,column=0,padx=10,pady=5,sticky=W)

        # employeeGender\_entry=ttk.Entry(company\_employee\_frame, textvariable=self.var\_gender ,width=16,font=("times new roman",12,"bold"))

        # employeeGender\_entry.grid(row=1,column=1,padx=5,pady=2,sticky=W)

        #employee age

        employeeage\_label=Label(company\_employee\_frame,text="employeeage:",font=("times new roman",12,"bold"),bg="white")

        employeeage\_label.grid(row=1,column=2,padx=10,pady=5,sticky=W)

        employeeage\_entry=ttk.Entry(company\_employee\_frame,textvariable=self.var\_age ,width=16,font=("times new roman",12,"bold"))

        employeeage\_entry.grid(row=1,column=3,padx=5,pady=2,sticky=W)

        #employee address

        employeeaddress\_label=Label(company\_employee\_frame,text="employeeaddress:",font=("times new roman",12,"bold"),bg="white")

        employeeaddress\_label.grid(row=1,column=0,padx=10,pady=7,sticky=W)

        employeeaddress\_entry=ttk.Entry(company\_employee\_frame,textvariable=self.var\_address ,width=16,font=("times new roman",12,"bold"))

        employeeaddress\_entry.grid(row=1,column=1,padx=5,pady=2,sticky=W)

        #employee Email

        employeeEmail\_label=Label(company\_employee\_frame,text="employeeEmail:",font=("times new roman",12,"bold"),bg="white")

        employeeEmail\_label.grid(row=2,column=0,padx=10,pady=5,sticky=W)

        employeeEmail\_entry=ttk.Entry(company\_employee\_frame,textvariable=self.var\_email ,width=16,font=("times new roman",12,"bold"))

        employeeEmail\_entry.grid(row=2,column=1,padx=5,pady=2,sticky=W)

        #employee PhNo

        employeePhNo\_label=Label(company\_employee\_frame,text="employeePhNo:",font=("times new roman",12,"bold"),bg="white")

        employeePhNo\_label.grid(row=2,column=2,padx=10,pady=5,sticky=W)

        employeePhNo\_entry=ttk.Entry(company\_employee\_frame,textvariable=self.var\_phone ,width=16,font=("times new roman",12,"bold"))

        employeePhNo\_entry.grid(row=2,column=3,padx=5,pady=2,sticky=W)

        #radio Button

        self.var\_radio1=StringVar()

        Radiobtn1=ttk.Radiobutton(company\_employee\_frame,variable=self.var\_radio1, text="take photo sample",value="yes")

        Radiobtn1.grid(row=6,column=0)

        Radiobtn2=ttk.Radiobutton(company\_employee\_frame,variable=self.var\_radio1,text="No photo sample",value="No")

        Radiobtn2.grid(row=6,column=1)

        #bbuttons frame

        btn\_frame=Frame(company\_employee\_frame,bd=2,relief=RIDGE,bg="white")

        btn\_frame.place(x=0,y=135,width=575,height=50)

        save\_btn=Button(btn\_frame,text="save",command=self.add\_data ,width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        save\_btn.grid(row=0,column=0)

        update\_btn=Button(btn\_frame,text="update",width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        update\_btn.grid(row=0,column=1)

        delete\_btn=Button(btn\_frame,text="delete",command=self.delete\_data, width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        delete\_btn.grid(row=0,column=2)

        reset\_btn=Button(btn\_frame,text="reset",command=self.reset\_data, width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        reset\_btn.grid(row=0,column=3)

        take\_photo\_btn=Button(btn\_frame,command=self.generate\_dataset,text="Take photo sample",width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        take\_photo\_btn.grid(row=1,column=0)

        update\_photo\_btn=Button(btn\_frame,text="update photo sample",width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        update\_photo\_btn.grid(row=1,column=1)

        #Right lable frame

        Right\_frame=LabelFrame(main\_frame,bd=2, bg="white",relief=RIDGE,text="employee Details",font=("times new roman ",12,"bold"))

        Right\_frame.place(x=650,y=10,width=590,height=450)

        img\_right=Image.open(r"C:\Users\HP\Desktop\minner project\b.jpeg")

        img\_right=img\_right.resize((600,100),Image.ANTIALIAS)

        self.photoimg\_right=ImageTk.PhotoImage(img\_right)

        f\_lbl=Label(self.root,image=self.photoimg\_right)

        f\_lbl.place(x=670,y=200,width=580,height=90)

        #\*\*\*\*\*\*\*\*\*\*search system\*\*\*\*\*\*\*\*\*\*\*\*\*

        search\_frame=LabelFrame(Right\_frame,bd=2, bg="white",relief=RIDGE,text="search system",font=("times new roman ",12,"bold"))

        search\_frame.place(x=5,y=100,width=575,height=70)

        search\_label=Label(search\_frame,text="Search By:",font=("times new roman",13,"bold"),bg="white")

        search\_label.grid(row=0,column=0,padx=10,pady=2,sticky=W)

        search\_combo=ttk.Combobox(search\_frame,font=("times new roman",12,"bold"),state="read only",width=10)

        search\_combo["values"]=("select","1st","emp\_id","phone\_no")

        search\_combo.current(0)

        search\_combo.grid(row=0,column=1,padx=2,pady=10,sticky=W)

        search\_entry=ttk.Entry(search\_frame,width=14,font=("times new roman",12,"bold"))

        search\_entry.grid(row=0,column=2,padx=10,pady=5,sticky=W)

        search\_btn=Button(search\_frame,text="search",width=10,font=("times new roman",10,"bold"),bg="blue",fg="white")

        search\_btn.grid(row=0,column=3,padx=4)

        search\_btn=Button(search\_frame,text="show All",width=10,font=("times new roman",10,"bold"),bg="blue",fg="white")

        search\_btn.grid(row=0,column=4,padx=4)

        table\_frame=Frame(Right\_frame,bg="white",bd=2,relief=RIDGE)

        table\_frame.place(x=5,y=180,width=575,height=240)

        scroll\_x=ttk.Scrollbar(table\_frame,orient=HORIZONTAL)

        scroll\_y=ttk.Scrollbar(table\_frame,orient=VERTICAL)

        self.employee\_table=ttk.Treeview(table\_frame,column=("dep","City","year","Shift","employeeId","employeeName","address","age","email","phone"),xscrollcommand=scroll\_x.set,yscrollcommand=scroll\_y.set)

        scroll\_x.pack(side=BOTTOM,fill=X)

        scroll\_y.pack(side=RIGHT,fill=Y)

        scroll\_x.config(command=self.employee\_table.xview)

        scroll\_y.config(command=self.employee\_table.yview)

        self.employee\_table.heading("dep",text="Department")

        self.employee\_table.heading("City",text="City")

        self.employee\_table.heading("year",text="year")

        self.employee\_table.heading("Shift",text="Shift")

        self.employee\_table.heading("employeeId",text="employeeId")

        self.employee\_table.heading("employeeName",text="employeeName")

        self.employee\_table.heading("address",text="employeeaddress")

        self.employee\_table.heading("age",text="employeeage")

        self.employee\_table.heading("email",text="employeeEmail")

        self.employee\_table.heading("phone",text="employeePhNo")

        #self.employee\_table.heading("photo",text="photo")

        self.employee\_table["show"]="headings"

        self.employee\_table.column("dep",width=100)

        self.employee\_table.column("City",width=100)

        self.employee\_table.column("year",width=100)

        self.employee\_table.column("Shift",width=100)

        self.employee\_table.column("employeeId",width=100)

        self.employee\_table.column("employeeName",width=100)

        self.employee\_table.column("address",width=100)

        self.employee\_table.column("age",width=100)

        self.employee\_table.column("email",width=100)

        self.employee\_table.column("phone",width=100)

        #self.employee\_table.column("photo",width=150)

        self.employee\_table.pack(fill=BOTH,expand=1)

        self.employee\_table.bind("ButtonRelease",self.get\_cursor)

        #self.fetch\_data()

    def add\_data(self):

        if self.var\_dep.get()=="select Department" or self.var\_name.get()=="" or self.var\_employeeid.get()=="":

            messagebox.showerror("Error ", "All the fields are required",parent=self.root)

        else:

            try:

                messagebox.showinfo("success","welcome")

                c=mysql.connector.connect(host="localhost",username="root",password="SHpatel@9893",database="face\_recognizer")

                my\_cursor=c.cursor()

                messagebox.showinfo("success","welcome2")

                my\_cursor.execute("insert into employee values(%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)",[    self.var\_dep.get(),

                                                                                                    self.var\_City.get(),

                                                                                                    self.var\_year.get(),

                                                                                                    self.var\_Shift.get(),

                                                                                                    self.var\_employeeid.get(),

                                                                                                    self.var\_name.get(),

                                                                                                    self.var\_address.get(),

                                                                                                    self.var\_age.get(),

                                                                                                    self.var\_email.get(),

                                                                                                    self.var\_phone.get()

                                                                                                                          ])

            #                                                                                        # self.var\_radio1.get()

            #

                # val=[self.var\_employeeid.get()]                                                                                # self.var\_radio1.get()

                # my\_cursor.execute("insert into employee(employeeid) values(%s)",val)

                messagebox.showinfo("success","welcome2")                                                                             #))

                c.commit()

                #self.fetch\_data()

                c.close()

                print("successfull")

                #messagebox.showinfo("success","employee details has been added successfully",parent=self.root)

            except Exception as es:

                # messagebox.showerror("Error",f"due To :{str(es)}",parent=self.root)

                print(es)

    #\*\*\*\*\*\*\*\*\*\*fetch data \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

    # def fetch\_data(self):

    #     conn=mysql.connector.connect(host="localhost",username="root",Password="SHpatel@9893",database="face\_recognizer")

    #     my\_cursor=conn.cursor()

    #     my\_cursor.execute("select \* from employee")

    #     data=my\_cursor.fetchall()

    #     if len(data)!=0:

    #         self.employee\_table.delete(\*self.employee\_table.get\_children())

    #         for i in data:

    #             self.employee\_table.insert("",END,values=i)

    #         conn.commit()

    #     conn.close()

    #=============get curser===============

    def get\_cursor(self,event=""):

        Cursor\_focus=self.employee\_table.focus()

        content=self.employee\_table.item(Cursor\_focus)

        data=content["values"]

        self.var\_dep.set(data[0]),

        self.var\_City.set(data[1]),

        self.var\_year.set(data[2]),

        self.var\_Shift.set(data[3]),

        self.var\_employeeid.set(data[4]),

        self.var\_name.set(data[5]),

        self.var\_address.set(data[6]),

        self.var\_age.set(data[7]),

        self.var\_email.set(data[8]),

        self.var\_phone.set(data[9])

    # #========update function======

    def update\_data(self):

        if self.var\_dep.get()=="select Department" or self.var\_name.get()=="" or self.var\_employeeid.get()=="":

            messagebox.showerror("Error ", "All the fields are required",parent=self.root)

        else:

            try:

                update=messagebox.askyesno("update","do you want to update this employee details",paeent=self.root)

                if update>0:

                    c=mysql.connector.connect(host="localhost",username="root",password="SHpatel@9893",database="face\_recognizer")

                    my\_cursor=c.cursor()

                    my\_cursor.execute("update employee set Department=%s,City=%s,year=%s,Shift=%s,employeename=%s,employeeaddres=%s,employeeage=%s,email=%s,phone=%s where ,employeeId=%s",[

                                                                                                    self.var\_dep.get(),

                                                                                                    self.var\_City.get(),

                                                                                                    self.var\_year.get(),

                                                                                                    self.var\_Shift.get(),

                                                                                                    self.var\_name.get(),

                                                                                                    self.var\_address.get(),

                                                                                                    self.var\_age.get(),

                                                                                                    self.var\_email.get(),

                                                                                                    self.var\_phone.get(),

                                                                                                    self.var\_employeeid.get()

                                                                                                ])

                else:

                    if not update:

                        return

                messagebox.showinfo("success","employee details successfully updated completely",parent=self.root)

                c.commit()

                #self.fetch\_data()

                c.close()

            except Exception as es:

                messagebox.showerror("Error",f"Due to:{str(es)}",parent=self.root)

    #delete function

    def delete\_data(self):

        if self.var\_employeeid.get()=="":

            messagebox.showerror("error","enployee id must be required",parent=self.root)

        else:

            try:

                delete=messagebox.askyesno("Employee delete page","do you you want to delete this employee",parent=self.root)

                if delete>0:

                     c=mysql.connector.connect(host="localhost",username="root",password="SHpatel@9893",database="face\_recognizer")

                     my\_cursor=c.cursor()

                     sql="delete from employee where employeeId=%s"

                     val=(self.var\_employeeid.get(),)

                     my\_cursor.execute(sql,val)

                else:

                    if not delete:

                        return

                c.commit()

                #self.fetch\_data()

                c.close()

                messagebox.showinfo("delete","succesfully delete details",parent=self.root)

            except Exception as es:

                messagebox.showerror("Error",f"Due to:{str(es)}",parent=self.root)

    #reset

    def reset\_data(self):

        self.var\_dep.set("select Department")

        self.var\_City.set("select City")

        self.var\_year.set("select year")

        self.var\_Shift.set("select Shift")

        self.var\_employeeid.set("")

        self.var\_name.set("")

        self.var\_address.set("")

        self.var\_age.set("")

        self.var\_email.set("")

        self.var\_phone.set("")

    #============= Generate data set or take photo samples =================

    def generate\_dataset(self):

        if self.var\_dep.get()=="select Department" or self.var\_name.get()=="" or self.var\_employeeid.get()=="":

            messagebox.showerror("Error ", "All the fields are required",parent=self.root)

        else:

            try:

                c=mysql.connector.connect(host="localhost",username="root",password="SHpatel@9893",database="face\_recognizer")

                my\_cursor=c.cursor()

                my\_cursor.execute("select \* from employee")

                myresult=my\_cursor.fetchall()

                id=0

                for x in myresult:

                    id+=1

                my\_cursor.execute("update employee set Department=%s,City=%s,year=%s,Shift=%s,employeename=%s,employeeaddres=%s,employeeage=%s,email=%s,phone=%s where ,employeeId=%s",[

                                                                                                        self.var\_dep.get(),

                                                                                                        self.var\_City.get(),

                                                                                                        self.var\_year.get(),

                                                                                                        self.var\_Shift.get(),

                                                                                                        self.var\_name.get(),

                                                                                                        self.var\_address.get(),

                                                                                                        self.var\_age.get(),

                                                                                                        self.var\_email.get(),

                                                                                                        self.var\_phone.get(),

                                                                                                        self.var\_employeeid.get()==id+1

                                                                                                    ])

                c.commit()

                self.fetch\_data()

                self.reset\_data()

                c.close()

                #============== Load predifined data on face frontals from opencv ===================

                face\_classifier=cv2.CascadeClassifier(".xml")

                def face\_cropped(img):

                    gray=cv2.cvtColor(img,cv2.COLOR\_BGR2GRAY)

                    faces=face\_classifier.detectMultiScale(gray,1,3,5)

                    #scaling factor=1.3

                    #minimum Neighbor=5

                    for(x,y,w,h) in faces:

                        face\_cropped=img[y:y+h,x:x+w]

                        return face\_cropped

                cap=cv2.VideoCapture(0)

                img\_id=0

                while True:

                    ret,my\_frame=cap.read()

                    if face\_cropped(my\_frame) is not None:

                        img\_id+=1

                    face=cv2.resize(face\_cropped(my\_frame),(450,450))

                    face=cv2.cvtColor(face,cv2.COLOR\_BGR2GRAY)

                    file\_name\_path="data/user."+str(id)+"."+str(img\_id)+".jpg"

                    cv2.imwrite(file\_name\_path,face)

                    cv2.putText(face,str(img\_id),(50,50) ,cv2.FONT\_HERSHEY\_COMPLEX,2,(0,255,0),2)

                    cv2.imshow("Crooped Face",face)

                    if cv2.waitKey(1)==13 or int(img\_id)==100:

                       break

                cap.release()

                cv2.destroyAllWindows()

                messagebox.showinfo("Result","Generating data sets completed!!!!")

            except Exception as es:

                messagebox.showerror("Error",f"Due to:{str(es)}",parent=self.root)

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

    root=Tk()

    obj=Employee(root)

    root.mainloop()

**Train datasets(5th video)**

from tkinter import\*

from tkinter import ttk

from PIL import Image,ImageTk

from tkinter import messagebox

import mysql.connector

import cv2

import os

import numpy as np

class Train:

    def \_\_init\_\_(self,root):

        self.root=root

        self.root.geometry("1200x790+0+0")

        self.root.title("face recogniton system")

        title\_lbl=Label(self.root,text="TRAIN DATA SET ",font=("times new roman",28,"bold"),bg="white",fg="red")

        title\_lbl.place(x=0,y=0,width=1430,height=35)

        img\_top=Image.open(r"C:\Users\HP\Desktop\minner project\face.jpeg")

        img\_top=img\_top.resize((1300,300),Image.ANTIALIAS)

        self.photoimg\_top=ImageTk.PhotoImage(img\_top)

        f\_lbl=Label(self.root,image=self.photoimg\_top)

        f\_lbl.place(x=0,y=55,width=1300,height=300)

        #button

        b1\_1=Button(self.root,text="TRAIN DATA",command=self.train\_classifier,cursor="hand2",font=("times new roman",27,"bold"),bg="yellow",fg="red")

        b1\_1.place(x=0,y=350,width=1300,height=40)

        img\_bottom=Image.open(r"C:\Users\HP\Desktop\minner project\b66.jpeg")

        img\_bottom=img\_bottom.resize((1300,300),Image.ANTIALIAS)

        self.photoimg\_bottom=ImageTk.PhotoImage(img\_bottom)

        f\_lbl=Label(self.root,image=self.photoimg\_bottom)

        f\_lbl.place(x=0,y=390,width=1300,height=300)

    def train\_classifier(self):

        data\_dir=("data")

        path=[ os.path.join(data\_dir,file) for file in os.listdir(data\_dir)]

        faces=[]

        ids=[]

        for image in path:

            img=Image.open(image).convert('L') #gray scale image

            imageNp=np.array(img,'uint8')

            id=int(os.path.split(image)[1].split('.')[1])

            faces.append(imageNp)

            ids.append(id)

            cv2.imshow("Training",imageNp)

            cv2.waitKey(1)==13

        id=np.array(ids)

        #\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Train the classifier And save\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

        clf=cv2.faces.LBPHFaceRecognizer\_create()

        clf.train(faces,ids)

        clf.write("classifier.xml")

        cv2.destroyAllWindows()

        messagebox.showinfo("result","Training datasets completed!!!")

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

    root=Tk()

    obj=Train(root)

    root.mainloop()

**face recognition coad(video 6th)**

from itertools import \_Predicate

from tkinter import\*

from tkinter import ttk

from PIL import Image,ImageTk

from tkinter import messagebox

from cv2 import rectangle

import mysql.connector

import cv2

import os

import numpy as np

class Face\_Recognition:

    def \_\_init\_\_(self,root):

        self.root=root

        self.root.geometry("1200x790+0+0")

        self.root.title("face recogniton system")

        title\_lbl=Label(self.root,text="FACE RECOGNITION ",font=("times new roman",28,"bold"),bg="white",fg="green")

        title\_lbl.place(x=0,y=0,width=1330,height=35)

        img\_top=Image.open(r"C:\Users\HP\Desktop\minner project\face.jpeg")

        img\_top=img\_top.resize((600,600),Image.ANTIALIAS)

        self.photoimg\_top=ImageTk.PhotoImage(img\_top)

        f\_lbl=Label(self.root,image=self.photoimg\_top)

        f\_lbl.place(x=0,y=55,width=600,height=600)

        img\_bottom=Image.open(r"C:\Users\HP\Desktop\minner project\bbb.jpeg")

        img\_bottom=img\_bottom.resize((700,610),Image.ANTIALIAS)

        self.photoimg\_bottom=ImageTk.PhotoImage(img\_bottom)

        f\_lbl=Label(self.root,image=self.photoimg\_bottom)

        f\_lbl.place(x=610,y=50,width=700,height=610)

        #button

        b1\_1=Button(self.root,text="FACE RECOGNATION",cursor="hand2",font=("times new roman",20,"bold"),bg="red",fg="white")

        b1\_1.place(x=800,y=600,width=300,height=40)

    #====================== face recognition =====================

    def face\_recog(self):

        def draw\_boundary(img,classifier,scaleFactor,minNeighbours,color,text,clf):

            gray\_image=cv2.cvtColor(img,cv2.COLOR\_BGR2GRAY)

            feature=classifier.detectMultiscale(gray\_image,scaleFactor,minNeighbours)

            coord=[]

            for(x,y,w,h) in feature:

                cv2.rectangle(img,(x,y),(x+w,y+h),(0,255,0),3)

                id,Predict=clf.predict(gray\_image[y:y+h,x:x+w])

                confidence=int(100\*(1-Predict/300))

                c=mysql.connector.connect(host="localhost",username="root",password="SHpatel@9893",database="face\_recognizer")

                my\_cursor=c.cursor()

                my\_cursor.execute("select name from employee where employeeid="+str(id))

                n=my\_cursor.fetchone()

                n="+".join(n)

                my\_cursor.execute("select Shift from employee where employeeid="+str(id))

                r=my\_cursor.fetchone()

                r="+".join(r)

                my\_cursor.execute("select dep from employee where employeeid="+str(id))

                d=my\_cursor.fetchone()

                d="+".join(d)

                if confidence>77:

                    cv2.putText(img,f"employeeName:{n}",(x,y-55),cv2.FONT\_HERSHEY\_COMPLEX,0.8,(255,255,255),3)

                    cv2.putText(img,f"employeeid:{r}",(x,y-30),cv2.FONT\_HERSHEY\_COMPLEX,0.8,(255,255,255),3)

                    cv2.putText(img,f"Department:{d}",(x,y-5),cv2.FONT\_HERSHEY\_COMPLEX,0.8,(255,255,255),3)

                else:

                    cv2.rectangle(img(x,y),(x+w,y+h),(0,0,255),3)

                    cv2.putText(img,"unknown face",(x,y-5),cv2.FONT\_HERSHEY\_COMPLEX,0.8,(255,255,255),3)

                coord=[x,y,w,h]

            return coord

        def recognize(img,clf,faceCascade):

            coord=draw\_boundary(img,faceCascade,1.1,10,(255,25,255),"face",clf)

            return img

        faceCscade=cv2.CascadeClassifier(".xml")

        clf=cv2.face.LBPHFfaceRecognizer\_create()

        clf.read("classifier.xml")

        video\_cap=cv2.VideoCapture(0)

        while True:

            ret,img=video\_cap.read()

            img=recognize(img,clf,faceCscade)

            cv2.imshow("welcome To face recognition",img)

            if cv2.waitKey(1)==13:

                break

        video\_cap.release()

        cv2.destroyAllWindows()

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

    root=Tk()

    obj=Face\_Recognition(root)

    root.mainloop()

**Employee211111111111111111111111**

from tkinter import\*

from tkinter import ttk

from tkinter.tix import TEXT

from turtle import left, st, update

from PIL import Image,ImageTk

from tkinter import messagebox

from colorama import Cursor

import mysql.connector

import cv2

class Employee:

    def \_\_init\_\_(self,root):

        self.root=root

        self.root.geometry("1200x790+0+0")

        self.root.title("face recogniton system")

        # \*\*\*\*\*\*\*\*\*\*\*\*\*\*variables\*\*\*\*\*\*\*\*\*\*\*

        self.var\_dep=StringVar()

        self.var\_City=StringVar()

        self.var\_year=StringVar()

        self.var\_Shift=StringVar()

        self.var\_employeeid=StringVar()

        self.var\_name=StringVar()

        #self.var\_gender=StringVar()

        self.var\_age=StringVar()

        self.var\_email=StringVar()

        self.var\_phone=StringVar()

        self.var\_address=StringVar()

        #first image

        img1=Image.open(r"C:\Users\HP\Desktop\minner project\sss.jpeg")

        img1=img1.resize((500,130),Image.ANTIALIAS)

        self.photoimg1=ImageTk.PhotoImage(img1)

        f\_lbl=Label(self.root,image=self.photoimg1)

        f\_lbl.place(x=0,y=0,width=500,height=120)

        #2 image

        img2=Image.open(r"C:\Users\HP\Desktop\minner project\ss.jpeg")

        img2=img2.resize((500,130),Image.ANTIALIAS)

        self.photoimg2=ImageTk.PhotoImage(img2)

        f\_lbl=Label(self.root,image=self.photoimg2)

        f\_lbl.place(x=500,y=0,width=500,height=120)

        #3 image

        img3=Image.open(r"C:\Users\HP\Desktop\minner project\ssss.jpeg")

        img3=img3.resize((450,130),Image.ANTIALIAS)

        self.photoimg3=ImageTk.PhotoImage(img3)

        f\_lbl=Label(self.root,image=self.photoimg3)

        f\_lbl.place(x=1000,y=0,width=450,height=120)

        #bg image

        img4=Image.open(r"C:\Users\HP\Desktop\minner project\b77.jpeg")

        img4=img4.resize((1400,710),Image.ANTIALIAS)

        self.photoimg4=ImageTk.PhotoImage(img4)

        bg\_img=Label(self.root,image=self.photoimg4)

        bg\_img.place(x=0,y=120,width=1400,height=710)

        title\_lbl=Label(bg\_img,text="EMPLOYEE ATTENDANCE MANAGMENT SYSTEM ",font=("times new roman",28,"bold"),bg="white",fg="dark red")

        title\_lbl.place(x=0,y=0,width=1430,height=35)

        main\_frame=Frame(bg\_img,bd=2, bg="white")

        main\_frame.place(x=10,y=45,width=1250,height=470)

        #left lable frame

        left\_frame=LabelFrame(main\_frame,bd=2, bg="white",relief=RIDGE,text="employee Details",font=("times new roman ",12,"bold"))

        left\_frame.place(x=10,y=10,width=600,height=450)

        img\_left=Image.open(r"C:\Users\HP\Desktop\minner project\b66.jpeg")

        img\_left=img\_left.resize((600,100),Image.ANTIALIAS)

        self.photoimg\_left=ImageTk.PhotoImage(img\_left)

        f\_lbl=Label(self.root,image=self.photoimg\_left)

        f\_lbl.place(x=33,y=200,width=587,height=90)

        #current courses

        current\_position\_frame=LabelFrame(main\_frame,bd=2, bg="white",relief=RIDGE,text="current Work information",font=("times new roman ",12,"bold"))

        current\_position\_frame.place(x=20,y=120,width=580,height=110)

        #department

        dep\_label=Label(current\_position\_frame,text="department",font=("times new roman",12,"bold"),bg="white")

        dep\_label.grid(row=0,column=0,padx=10,sticky=W)

        dep\_combo=ttk.Combobox(current\_position\_frame, textvariable=self.var\_dep, font=("times new roman",12,"bold"),state="read only",width=17)

        dep\_combo["values"]=("select Department","computer","it","civil","mechnical")

        dep\_combo.current(0)

        dep\_combo.grid(row=0,column=1,padx=2,pady=10,sticky=W)

        # City

        City\_label=Label(current\_position\_frame,text="City",font=("times new roman",12,"bold"),bg="white")

        City\_label.grid(row=0,column=2,padx=10,sticky=W)

        City\_combo=ttk.Combobox(current\_position\_frame,textvariable=self.var\_City ,font=("times new roman",12,"bold"),state="read only",width=17)

        City\_combo["values"]=("select City","Dewas","Indore","Ujjain","Delhi","Bhopal")

        City\_combo.current(0)

        City\_combo.grid(row=0,column=3,padx=2,pady=10,sticky=W)

        # year

        year\_label=Label(current\_position\_frame,text="year",font=("times new roman",12,"bold"),bg="white")

        year\_label.grid(row=1,column=0,padx=10,sticky=W)

        year\_combo=ttk.Combobox(current\_position\_frame,textvariable=self.var\_year ,font=("times new roman",12,"bold"),state="read only",width=17)

        year\_combo["values"]=("select year","2020-21","2021-22","2022-23","2023-24")

        year\_combo.current(0)

        year\_combo.grid(row=1,column=1,padx=2,pady=10,sticky=W)

         #semester

        Shift\_label=Label(current\_position\_frame,text="Shift",font=("times new roman",12,"bold"),bg="white")

        Shift\_label.grid(row=1,column=2,padx=10,sticky=W)

        Shift\_combo=ttk.Combobox(current\_position\_frame,textvariable=self.var\_Shift, font=("times new roman",12,"bold"),state="read only",width=17)

        Shift\_combo["values"]=("select Shift","1st","2nd")

        Shift\_combo.current(0)

        Shift\_combo.grid(row=1,column=3,padx=2,pady=10,sticky=W)

        #company employee information

        company\_employee\_frame=LabelFrame(main\_frame,bd=2, bg="white",relief=RIDGE,text="employee information",font=("times new roman ",12,"bold"))

        company\_employee\_frame.place(x=20,y=240,width=580,height=210)

        #employee ID

        employeeId\_label=Label(company\_employee\_frame,text="employeeId:",font=("times new roman",12,"bold"),bg="white")

        employeeId\_label.grid(row=0,column=0,padx=10,pady=7,sticky=W)

        employeeId\_entry=ttk.Entry(company\_employee\_frame,textvariable=self.var\_employeeid ,width=16,font=("times new roman",12,"bold"))

        employeeId\_entry.grid(row=0,column=1,padx=5,pady=2,sticky=W)

        #employee Name

        employeeName\_label=Label(company\_employee\_frame,text="employeeName:",font=("times new roman",12,"bold"),bg="white")

        employeeName\_label.grid(row=0,column=2,padx=10,pady=5,sticky=W)

        employeeName\_entry=ttk.Entry(company\_employee\_frame,textvariable=self.var\_name ,width=16,font=("times new roman",12,"bold"))

        employeeName\_entry.grid(row=0,column=3,padx=5,pady=2,sticky=W)

        # #employee Gender

        # employeeGender\_label=Label(company\_employee\_frame,text="employeeGender:",font=("times new roman",12,"bold"),bg="white")

        # employeeGender\_label.grid(row=1,column=0,padx=10,pady=5,sticky=W)

        # employeeGender\_entry=ttk.Entry(company\_employee\_frame, textvariable=self.var\_gender ,width=16,font=("times new roman",12,"bold"))

        # employeeGender\_entry.grid(row=1,column=1,padx=5,pady=2,sticky=W)

        #employee age

        employeeage\_label=Label(company\_employee\_frame,text="employeeage:",font=("times new roman",12,"bold"),bg="white")

        employeeage\_label.grid(row=1,column=2,padx=10,pady=5,sticky=W)

        employeeage\_entry=ttk.Entry(company\_employee\_frame,textvariable=self.var\_age ,width=16,font=("times new roman",12,"bold"))

        employeeage\_entry.grid(row=1,column=3,padx=5,pady=2,sticky=W)

        #employee address

        employeeaddress\_label=Label(company\_employee\_frame,text="employeeaddress:",font=("times new roman",12,"bold"),bg="white")

        employeeaddress\_label.grid(row=1,column=0,padx=10,pady=7,sticky=W)

        employeeaddress\_entry=ttk.Entry(company\_employee\_frame,textvariable=self.var\_address ,width=16,font=("times new roman",12,"bold"))

        employeeaddress\_entry.grid(row=1,column=1,padx=5,pady=2,sticky=W)

        #employee Email

        employeeEmail\_label=Label(company\_employee\_frame,text="employeeEmail:",font=("times new roman",12,"bold"),bg="white")

        employeeEmail\_label.grid(row=2,column=0,padx=10,pady=5,sticky=W)

        employeeEmail\_entry=ttk.Entry(company\_employee\_frame,textvariable=self.var\_email ,width=16,font=("times new roman",12,"bold"))

        employeeEmail\_entry.grid(row=2,column=1,padx=5,pady=2,sticky=W)

        #employee PhNo

        employeePhNo\_label=Label(company\_employee\_frame,text="employeePhNo:",font=("times new roman",12,"bold"),bg="white")

        employeePhNo\_label.grid(row=2,column=2,padx=10,pady=5,sticky=W)

        employeePhNo\_entry=ttk.Entry(company\_employee\_frame,textvariable=self.var\_phone ,width=16,font=("times new roman",12,"bold"))

        employeePhNo\_entry.grid(row=2,column=3,padx=5,pady=2,sticky=W)

        #radio Button

        self.var\_radio1=StringVar()

        Radiobtn1=ttk.Radiobutton(company\_employee\_frame,variable=self.var\_radio1, text="take photo sample",value="yes")

        Radiobtn1.grid(row=6,column=0)

        Radiobtn2=ttk.Radiobutton(company\_employee\_frame,variable=self.var\_radio1,text="No photo sample",value="No")

        Radiobtn2.grid(row=6,column=1)

        #bbuttons frame

        btn\_frame=Frame(company\_employee\_frame,bd=2,relief=RIDGE,bg="white")

        btn\_frame.place(x=0,y=135,width=575,height=45)

        save\_btn=Button(btn\_frame,text="save",command=self.add\_data ,width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        save\_btn.grid(row=0,column=0)

        update\_btn=Button(btn\_frame,command=self.update\_data,text="update",width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        update\_btn.grid(row=0,column=1)

        delete\_btn=Button(btn\_frame,text="delete",command=self.delete\_data, width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        delete\_btn.grid(row=0,column=2)

        reset\_btn=Button(btn\_frame,text="reset",command=self.reset\_data, width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        reset\_btn.grid(row=0,column=3)

        btn\_frame1=Frame(company\_employee\_frame,bd=2,relief=RIDGE,bg="white")

        btn\_frame1.place(x=0,y=157,width=575,height=30)

        take\_photo\_btn=Button(btn\_frame1,command=self.generate\_dataset,text="Take photo sample",width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        take\_photo\_btn.grid(row=0,column=0)

        update\_photo\_btn=Button(btn\_frame1,text="update photo sample",width=19,font=("times new roman",10,"bold"),bg="blue",fg="white")

        update\_photo\_btn.grid(row=0,column=1)

        #Right lable frame

        Right\_frame=LabelFrame(main\_frame,bd=2, bg="white",relief=RIDGE,text="employee Details",font=("times new roman ",12,"bold"))

        Right\_frame.place(x=650,y=10,width=590,height=450)

        img\_right=Image.open(r"C:\Users\HP\Desktop\minner project\b.jpeg")

        img\_right=img\_right.resize((600,100),Image.ANTIALIAS)

        self.photoimg\_right=ImageTk.PhotoImage(img\_right)

        f\_lbl=Label(self.root,image=self.photoimg\_right)

        f\_lbl.place(x=670,y=200,width=580,height=90)

        #\*\*\*\*\*\*\*\*\*\*search system\*\*\*\*\*\*\*\*\*\*\*\*\*

        search\_frame=LabelFrame(Right\_frame,bd=2, bg="white",relief=RIDGE,text="search system",font=("times new roman ",12,"bold"))

        search\_frame.place(x=5,y=100,width=575,height=70)

        search\_label=Label(search\_frame,text="Search By:",font=("times new roman",13,"bold"),bg="white")

        search\_label.grid(row=0,column=0,padx=10,pady=2,sticky=W)

        search\_combo=ttk.Combobox(search\_frame,font=("times new roman",12,"bold"),state="read only",width=10)

        search\_combo["values"]=("select","City","employee\_id","name")

        search\_combo.current(0)

        search\_combo.grid(row=0,column=1,padx=2,pady=10,sticky=W)

        search\_entry=ttk.Entry(search\_frame,width=14,font=("times new roman",12,"bold"))

        search\_entry.grid(row=0,column=2,padx=10,pady=5,sticky=W)

        search\_btn=Button(search\_frame,command=self.fetch\_data,text="search",width=10,font=("times new roman",10,"bold"),bg="blue",fg="white")

        search\_btn.grid(row=0,column=3,padx=4)

        search\_btn=Button(search\_frame,command=self.fetch\_data,text="show All",width=10,font=("times new roman",10,"bold"),bg="blue",fg="white")

        search\_btn.grid(row=0,column=4,padx=4)

        table\_frame=Frame(Right\_frame,bg="white",bd=2,relief=RIDGE)

        table\_frame.place(x=5,y=180,width=575,height=240)

        scroll\_x=ttk.Scrollbar(table\_frame,orient=HORIZONTAL)

        scroll\_y=ttk.Scrollbar(table\_frame,orient=VERTICAL)

        self.employee\_table=ttk.Treeview(table\_frame,column=("dep","City","year","Shift","employeeId","employeeName","address","age","email","phone"),xscrollcommand=scroll\_x.set,yscrollcommand=scroll\_y.set)

        scroll\_x.pack(side=BOTTOM,fill=X)

        scroll\_y.pack(side=RIGHT,fill=Y)

        scroll\_x.config(command=self.employee\_table.xview)

        scroll\_y.config(command=self.employee\_table.yview)

        self.employee\_table.heading("dep",text="Department")

        self.employee\_table.heading("City",text="City")

        self.employee\_table.heading("year",text="year")

        self.employee\_table.heading("Shift",text="Shift")

        self.employee\_table.heading("employeeId",text="employeeId")

        self.employee\_table.heading("employeeName",text="employeeName")

        self.employee\_table.heading("address",text="employeeaddress")

        self.employee\_table.heading("age",text="employeeage")

        self.employee\_table.heading("email",text="employeeEmail")

        self.employee\_table.heading("phone",text="employeePhNo")

        #self.employee\_table.heading("photo",text="photo")

        self.employee\_table["show"]="headings"

        self.employee\_table.column("dep",width=100)

        self.employee\_table.column("City",width=100)

        self.employee\_table.column("year",width=100)

        self.employee\_table.column("Shift",width=100)

        self.employee\_table.column("employeeId",width=100)

        self.employee\_table.column("employeeName",width=100)

        self.employee\_table.column("address",width=100)

        self.employee\_table.column("age",width=100)

        self.employee\_table.column("email",width=100)

        self.employee\_table.column("phone",width=100)

        #self.employee\_table.column("photo",width=150)

        self.employee\_table.pack(fill=BOTH,expand=1)

        self.employee\_table.bind("<ButtonRelease>",self.get\_cursor)

        self.fetch\_data()

#====================== function declearation=================

    def add\_data(self):

        if self.var\_dep.get()=="select Department" or self.var\_name.get()=="" or self.var\_employeeid.get()=="":

            messagebox.showerror("Error ", "All the fields are required",parent=self.root)

        else:

            try:

                messagebox.showinfo("success","welcome")

                c=mysql.connector.connect(host="localhost",username="root",password="SHpatel@9893",database="face\_recognizer")

                my\_cursor=c.cursor()

                messagebox.showinfo("success","welcome2")

                my\_cursor.execute("insert into employee values(%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)",[    self.var\_dep.get(),

                                                                                                    self.var\_City.get(),

                                                                                                    self.var\_year.get(),

                                                                                                    self.var\_Shift.get(),

                                                                                                    self.var\_employeeid.get(),

                                                                                                    self.var\_name.get(),

                                                                                                    self.var\_address.get(),

                                                                                                    self.var\_age.get(),

                                                                                                    self.var\_email.get(),

                                                                                                    self.var\_phone.get()

                                                                                                                          ])

            #                                                                                        # self.var\_radio1.get()

            #

                messagebox.showinfo("success","welcome2")

                c.commit()

                self.fetch\_data()

                c.close()

                print("successfull")

                #messagebox.showinfo("success","employee details has been added successfully",parent=self.root)

            except Exception as es:

                # messagebox.showerror("Error",f"due To :{str(es)}",parent=self.root)

                print(es)

    #\*\*\*\*\*\*\*\*\*\*fetch data \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

    def fetch\_data(self):

          conn=mysql.connector.connect(host="localhost",username="root",password="SHpatel@9893",database="face\_recognizer")

          my\_cursor=conn.cursor()

          my\_cursor.execute("select \* from employee")

          data=my\_cursor.fetchall()

          if len(data)!=0:

            self.employee\_table.delete(\*self.employee\_table.get\_children())

            for i in data:

                self.employee\_table.insert("",END,values=i)

            conn.commit()

          conn.close()

    #=============get curser===============

    def get\_cursor(self,event=""):

        Cursor\_focus=self.employee\_table.focus()

        content=self.employee\_table.item(Cursor\_focus)

        data=content["values"]

        self.var\_dep.set(data[0]),

        self.var\_City.set(data[1]),

        self.var\_year.set(data[2]),

        self.var\_Shift.set(data[3]),

        self.var\_employeeid.set(data[4]),

        self.var\_name.set(data[5]),

        self.var\_address.set(data[6]),

        self.var\_age.set(data[7]),

        self.var\_email.set(data[8]),

        self.var\_phone.set(data[9])

    # #========update function======

    def update\_data(self):

        if self.var\_dep.get()=="select Department" or self.var\_name.get()=="" or self.var\_employeeid.get()=="":

            messagebox.showerror("Error ", "All the fields are required",parent=self.root)

        else:

            try:

                update=messagebox.askyesno("update","do you want to update this employee details",parent=self.root)

                if update>0:

                    c=mysql.connector.connect(host="localhost",username="root",password="SHpatel@9893",database="face\_recognizer")

                    my\_cursor=c.cursor()

                    my\_cursor.execute("update employee set Department=%s,City=%s,year=%s,Shift=%s,employeename=%s,employeeaddres=%s,employeeage=%s,email=%s,phone=%s where ,employeeId=%s",[

                                                                                                                                                                self.var\_dep.get(),

                                                                                                                                                                self.var\_City.get(),

                                                                                                                                                                self.var\_year.get(),

                                                                                                                                                                self.var\_Shift.get(),

                                                                                                                                                                self.var\_name.get(),

                                                                                                                                                                self.var\_address.get(),

                                                                                                                                                                self.var\_age.get(),

                                                                                                                                                                self.var\_email.get(),

                                                                                                                                                                self.var\_phone.get(),

                                                                                                                                                                self.var\_employeeid.get()

                                                                                                                                                            ])

                else:

                    if not update:

                        return

                messagebox.showinfo("success","employee details successfully updated completely",parent=self.root)

                c.commit()

                self.fetch\_data()

                c.close()

            except Exception as es:

                messagebox.showerror("Error",f"Due to:{str(es)}",parent=self.root)

    #delete function

    def delete\_data(self):

        if self.var\_employeeid.get()=="":

            messagebox.showerror("error","enployee id must be required",parent=self.root)

        else:

            try:

                delete=messagebox.askyesno("Employee delete page","do you you want to delete this employee",parent=self.root)

                if delete>0:

                     c=mysql.connector.connect(host="localhost",username="root",password="SHpatel@9893",database="face\_recognizer")

                     my\_cursor=c.cursor()

                     sql="delete from employee where employeeId=%s"

                     val=(self.var\_employeeid.get(),)

                     my\_cursor.execute(sql,val)

                else:

                    if not delete:

                        return

                c.commit()

                #self.fetch\_data()

                c.close()

                messagebox.showinfo("delete","succesfully delete details",parent=self.root)

            except Exception as es:

                messagebox.showerror("Error",f"Due to:{str(es)}",parent=self.root)

    #reset

    def reset\_data(self):

        self.var\_dep.set("select Department")

        self.var\_City.set("select City")

        self.var\_year.set("select year")

        self.var\_Shift.set("select Shift")

        self.var\_employeeid.set("")

        self.var\_name.set("")

        self.var\_address.set("")

        self.var\_age.set("")

        self.var\_email.set("")

        self.var\_phone.set("")

    #============= Generate data set or take photo samples =================

    def generate\_dataset(self):

        if self.var\_dep.get()=="select Department" or self.var\_name.get()=="" or self.var\_employeeid.get()=="":

            messagebox.showerror("Error ", "All the fields are required",parent=self.root)

        else:

            try:

                c=mysql.connector.connect(host="localhost",username="root",password="SHpatel@9893",database="face\_recognizer")

                my\_cursor=c.cursor()

                my\_cursor.execute("select \* from employee")

                myresult=my\_cursor.fetchall()

                id=0

                for x in myresult:

                    id+=1

                my\_cursor.execute("update employee set Department=%s,City=%s,year=%s,Shift=%s,employeename=%s,employeeaddres=%s,employeeage=%s,email=%s,phone=%s where ,employeeId=%s",[

                                                                                                        self.var\_dep.get(),

                                                                                                        self.var\_City.get(),

                                                                                                        self.var\_year.get(),

                                                                                                        self.var\_Shift.get(),

                                                                                                        self.var\_name.get(),

                                                                                                        self.var\_address.get(),

                                                                                                        self.var\_age.get(),

                                                                                                        self.var\_email.get(),

                                                                                                        self.var\_phone.get(),

                                                                                                        self.var\_employeeid.get()==id+1

                                                                                                    ])

                c.commit()

                self.fetch\_data()

                self.reset\_data()

                c.close()

                #============== Load predifined data on face frontals from opencv ===================

                face\_classifier=cv2.CascadeClassifier("haarcascade\_frontalface\_default.xml")

                def face\_cropped(img):

                    gray=cv2.cvtColor(img,cv2.COLOR\_BGR2GRAY)

                    faces=face\_classifier.detectMultiScale(gray,1.3,5)

                    # scaling factor=1.3

                    # minimum Neighbor=5

                    for(x,y,w,h) in faces:

                        face\_cropped=img[y:y+h,x:x+w]

                        return face\_cropped

                cap=cv2.VideoCapture(0)

                img\_id=0

                while True:

                    ret,my\_frame=cap.read()

                    if face\_cropped(my\_frame) is not None:

                        img\_id+=1

                    face=cv2.resize(face\_cropped(my\_frame),(450,450))

                    face=cv2.cvtColor(face,cv2.COLOR\_BGR2GRAY)

                    file\_name\_path="data/user."+str(id)+"."+str(img\_id)+".jpg"

                    cv2.imwrite(file\_name\_path,face)

                    cv2.putText(face,str(img\_id),(50,50) ,cv2.FONT\_HERSHEY\_COMPLEX,2,(0,255,0),2)

                    cv2.imshow("Crooped Face",face)

                    if cv2.waitKey(1)==13 or int(img\_id)==100:

                       break

                cap.release()

                cv2.destroyAllWindows()

                messagebox.showinfo("Result","Generating data sets completed!!!!")

            except Exception as es:

                messagebox.showerror("Error",f"Due to:{str(es)}",parent=self.root)

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

    root=Tk()

    obj=Employee(root)

    root.mainloop()