In [ ]:

```
from tkinter import*
from tkinter import ttk
from PIL import Image,ImageTk
from tkinter import messagebox
import mysql.connector
import cv2
import os

class Person:
    def __init__(self,root):
        self.root=root
        self.root.geometry("1530x790+0+0")
        self.root.title("face recognition system")

        ##+++======variables
        self.var_Department=StringVar()
        self.var_Course=StringVar()
        self.var_Year=StringVar()
        self.var_Age=StringVar()
        self.var_Person_ID=StringVar()
        self.var_Name=StringVar()
        self.var_Gender=StringVar()
        self.var_Phone_no=StringVar()
        self.var_Address=StringVar()


#first image
        img = Image.open(r"C:\Users\HP\OneDrive\Desktop\Adrija\industrial internship\internship\costacloud\face recog\college
        img=img.resize((500,130),Image.ANTIALIAS)
        self.photoimg=ImageTk.PhotoImage(img)
        f_lbl=Label(self.root,image=self.photoimg)
        f_lbl.place(x=0,y=0,width=500,height=130)


#second image
        img1 = Image.open(r"C:\Users\HP\OneDrive\Desktop\Adrija\industrial internship\internship\costacloud\face recog\colleg
        img1=img1.resize((500,130),Image.ANTIALIAS)
        self.photoimg1=ImageTk.PhotoImage(img1)
        f_lbl=Label(self.root,image=self.photoimg1)
        f_lbl.place(x=500,y=0,width=500,height=130)

#third image
        img2 = Image.open(r"C:\Users\HP\OneDrive\Desktop\Adrija\industrial internship\internship\costacloud\face recog\colleg
        img2=img2.resize((500,130),Image.ANTIALIAS)
        self.photoimg2=ImageTk.PhotoImage(img2)
        f_lbl=Label(self.root,image=self.photoimg2)
        f_lbl.place(x=1000,y=0,width=500,height=130)

#bg image

        img3 = Image.open(r"C:\Users\HP\OneDrive\Desktop\Adrija\industrial internship\internship\costacloud\face recog\colleg
        img3=img3.resize((1530,710),Image.ANTIALIAS)
        self.photoimg3=ImageTk.PhotoImage(img3)
        bg_img=Label(self.root,image=self.photoimg3)
        bg_img.place(x=0,y=130,width=1530,height=710)


        title_lbl=Label(bg_img,text="Missing Management System ", font=("times new roman",30,"bold"),bg="white",fg="red")
        title_lbl.place(x=0,y=0,width=1350,height=45)


        main_frame=Frame(bg_img,bd=2,bg="white")
        main_frame.place(x=11,y=55,width=1250,height=455)

#Left label frame

        Left_frame= LabelFrame(main_frame,bd=2,bg="white",relief=RIDGE,text=" Missing Details",font=("times new roman",12,"bc
        Left_frame.place(x=21,y=10, width =650 , height=430)

        img_L = Image.open(r"C:\Users\HP\OneDrive\Desktop\Adrija\industrial internship\internship\costacloud\face recog\colle
        img_L=img_L.resize((720,130),Image.ANTIALIAS)
        self.photoimg_L=ImageTk.PhotoImage(img_L)
        f_lbl=Label(Left_frame,image=self.photoimg_L)
        f_lbl.place(x=0,y=0,width=600,height=80)
#current course
        cc_frame= LabelFrame(Left_frame,bd=2,bg="white",relief=RIDGE,text="Missing Person Info",font=("times new roman",12,"b
        cc_frame.place(x=5,y=80, width =580 , height=120)

        #department label
        dep_label = Label(cc_frame, text="Department",font=("times new roman",12,"bold"),bg="white")
        dep_label.grid(row=0,column=0,padx=2,pady=10,sticky=W)


        dep_combo=ttk.Combobox(cc_frame,textvariable=self.var_Department,font=("times new roman",12,"bold"),width=17,state="r
        dep_combo["values"]=("Select Department","A","B","C")
        dep_combo.current(0)
        dep_combo.grid(row=0,column=1,padx=2,pady=10,sticky=W)
```

```
    #Course label
        c_label = Label(cc_frame, text="Course",font=("times new roman",12,"bold"),bg="white")
        c_label.grid(row=0,column=2,padx=2,pady=10,sticky=W)


        c_combo=ttk.Combobox(cc_frame,textvariable=self.var_Course,font=("times new roman",12,"bold"),width=17,state="readonl
        c_combo["values"]=("Select Course","A","B","C")
        c_combo.current(0)
        c_combo.grid(row=0,column=3,padx=2,pady=10,sticky=W)

    #Year
        y_label = Label(cc_frame, text="Year",font=("times new roman",12,"bold"),bg="white")
        y_label.grid(row=1,column=0,padx=2,pady=10,sticky=W)


        y_combo=ttk.Combobox(cc_frame,textvariable=self.var_Year,font=("times new roman",12,"bold"),width=17,state="readonly"
        y_combo["values"]=("Select Year","20-21","21-22","22-23")
        y_combo.current(0)
        y_combo.grid(row=1,column=1,padx=2,pady=10,sticky=W)

    #Age
        a_label = Label(cc_frame, text="Age",font=("times new roman",12,"bold"),bg="white")
        a_label.grid(row=1,column=2,padx=2,pady=10,sticky=W)


        a_combo=ttk.Combobox(cc_frame,textvariable=self.var_Age,font=("times new roman",12,"bold"),width=17,state="readonly")
        a_combo["values"]=("Select Year","10-20","20-30","30-40","40-50","50-60","60-70")
        a_combo.current(0)
        a_combo.grid(row=1,column=3,padx=2,pady=10,sticky=W)

#Personal Information
        p_frame= LabelFrame(Left_frame,bd=2,bg="white",relief=RIDGE,text="Missing Person Personal Info",font=("times new roma
        p_frame.place(x=5,y=200, width =580 , height=200)

        #Missing Person Id
        mi_label = Label(p_frame, text="Person Id",font=("times new roman",12,"bold"),bg="white")
        mi_label.grid(row=0,column=0,padx=1,pady=5,sticky=W)

        mi_entry= ttk.Entry(p_frame,textvariable= self.var_Person_ID,width=20,font=("times new roman",12,"bold"))
        mi_entry.grid(row=0,column=1,padx=1,pady=5,sticky=W)



         #Missing Person Name
        mn_label = Label(p_frame, text="Name",font=("times new roman",12,"bold"),bg="white")
        mn_label.grid(row=0,column=2,padx=1,pady=5,sticky=W)

        mn_entry= ttk.Entry(p_frame,textvariable=self.var_Name,width=20,font=("times new roman",12,"bold"))
        mn_entry.grid(row=0,column=3,padx=1,pady=5,sticky=W)


        #Missing Person gender
        g_label = Label(p_frame, text="Gender",font=("times new roman",12,"bold"),bg="white")
        g_label.grid(row=1,column=0,padx=1,pady=5,sticky=W)


        g_combo=ttk.Combobox(p_frame,textvariable=self.var_Gender,font=("times new roman",12,"bold"),width=17,state="readonly
        g_combo["values"]=("Select Gender","Male","Female","Others")
        g_combo.current(0)
        g_combo.grid(row=1,column=1,padx=2,pady=10,sticky=W)

        #Missing Person Phone no
        mp_label = Label(p_frame, text="Phone No",font=("times new roman",12,"bold"),bg="white")
        mp_label.grid(row=1,column=2,padx=1,pady=5,sticky=W)

        mp_entry= ttk.Entry(p_frame,textvariable=self.var_Phone_no,width=20,font=("times new roman",12,"bold"))
        mp_entry.grid(row=1,column=3,padx=1,pady=5,sticky=W)

        #Missing Person Address
        ma_label = Label(p_frame, text="Address",font=("times new roman",12,"bold"),bg="white")
        ma_label.grid(row=2,column=0,padx=1,pady=5,sticky=W)

        ma_entry= ttk.Entry(p_frame,textvariable=self.var_Address,width=20,font=("times new roman",12,"bold"))
        ma_entry.grid(row=2,column=1,padx=1,pady=5,sticky=W)




        #radio Buttons
        self.var_radio1=StringVar()
        radio1=ttk.Radiobutton(p_frame,variable=self.var_radio1,text="take a photo Sample",value="Yes")
        radio1.grid(row=5,column=0)


        radio2=ttk.Radiobutton(p_frame,variable=self.var_radio1,text=" donot take a photo Sample",value="No")
        radio2.grid(row=5,column=1)


        #savebutton
        save_btn=Button(p_frame,text="Save",command=self.add_data,width=10,font=("times new roman",12,"bold"),bg="blue",fg="w
        save_btn.grid(row=6, column=0)


        #take a photo sample
        tps_btn=Button(p_frame,command=self.generate_dataset,text="take photo",width=10,font=("times new roman",12,"bold"),bg
        tps_btn.grid(row=6, column=1)
        #update a photo sample
        ups_btn=Button(p_frame,text="update photo",width=10,font=("times new roman",12,"bold"),bg="blue",fg="white")
        ups_btn.grid(row=6, column=2)
```

```
        #Right label frame

        Right_frame= LabelFrame(main_frame,bd=2,bg="white",relief=RIDGE,text="Person Details",font=("times new roman",12,"bol
        Right_frame.place(x=630,y=10, width =600 , height=430)



        #image
        img_R = Image.open(r"C:\Users\HP\OneDrive\Desktop\Adrija\industrial internship\internship\costacloud\face recog\colle
        img_R=img_R.resize((720,130),Image.ANTIALIAS)
        self.photoimg_R=ImageTk.PhotoImage(img_R)
        f_lbl=Label(Right_frame,image=self.photoimg_R)
        f_lbl.place(x=0,y=0,width=600,height=80)

        #Searching Sytem

        s_frame= LabelFrame(Right_frame,bd=2,bg="white",relief=RIDGE,text="Searching System",font=("times new roman",12,"bold
        s_frame.place(x=5,y=80, width =585, height=80)

        s_label = Label(s_frame, text="Search BY:",font=("times new roman",12,"bold"),bg="red",fg="white")
        s_label.grid(row=0,column=0,padx=1,pady=5,sticky=W)

        s_combo=ttk.Combobox(s_frame,font=("times new roman",12,"bold"),width=15,state="readonly")
        s_combo["values"]=("Select","Person ID ","Name ")
        s_combo.current(0)
        s_combo.grid(row=0,column=1,padx=2,pady=10,sticky=W)

        s_entry= ttk.Entry(s_frame,width=15,font=("times new roman",12,"bold"))
        s_entry.grid(row=0,column=2,padx=1,pady=5,sticky=W)


        s_btn=Button(s_frame,text="Search",width=10,font=("times new roman",12,"bold"),bg="blue",fg="white")
        s_btn.grid(row=0, column=3 ,padx=4)

        show_btn=Button(s_frame,text="Show",width=10,font=("times new roman",12,"bold"),bg="blue",fg="white")
        show_btn.grid(row=0, column=4)


        #table
        t_frame= Frame(Right_frame,bd=2,bg="white",relief=RIDGE)
        t_frame.place(x=5,y=150, width =585, height=250)

        scroll_x=ttk.Scrollbar(t_frame,orient=HORIZONTAL)
        scroll_y=ttk.Scrollbar(t_frame,orient=VERTICAL)
        self.person_table=ttk.Treeview(t_frame,column=("Department","Course","Year","Age","Person ID","Name","Gender","Phone

        scroll_x.pack(side=BOTTOM,fill=X)
        scroll_y.pack(side=RIGHT,fill=Y)
        scroll_x.config(command=self.person_table.xview)
        scroll_y.config(command=self.person_table.yview)


        self.person_table.heading("Department",text="Department")
        self.person_table.heading("Course",text="Course")
        self.person_table.heading("Year",text="Year")
        self.person_table.heading("Age",text="Age")
        self.person_table.heading("Person ID",text="Person ID")
        self.person_table.heading("Name",text="Name")
        self.person_table.heading("Gender",text="Gender")
        self.person_table.heading("Phone No",text="Phone No")
        self.person_table.heading("Address",text="Address")
        self.person_table.heading("photo",text="PhotoSample Status")

        self.person_table["show"]="headings"


        self.person_table.column("Department",width=100)
        self.person_table.column("Course",width=100)
        self.person_table.column("Year",width=100)
        self.person_table.column("Age",width=100)
        self.person_table.column("Person ID",width=100)
        self.person_table.column("Name",width=100)
        self.person_table.column("Gender",width=100)
        self.person_table.column("Phone No",width=100)
        self.person_table.column("Address",width=100)
        self.person_table.column("photo",width=100)


        self.person_table.pack(fill=BOTH,expand=1)
        self.person_table.bind("<ButtonRelease>",self.get_cursor)
        self.fetch_data()

    ###function declaration

    def add_data(self):
        if self.var_Department.get()=="Select Department" or self.var_Name.get()=="" or self.var_Person_ID.get()=="":
            messagebox.showerror("Error","All Fields are required",parent=self.root)
        else:
            try:
                conn=mysql.connector.connect(host="localhost",username="root",password="Dimpy#1609*",database="face_recognize
                my_cursor=conn.cursor()
                my_cursor.execute("insert into person values(%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)",(
                                                                                    self.var_Department.get(),
                                                                                    self.var_Course.get(),
                                                                                    self.var_Year.get(),
                                                                                    self.var_Age.get(),
                                                                                    self.var_Person_ID.get()
```

```
                                                                                                    self.var_Name.get(),
                                                                                                    self.var_Gender.get(),
                                                                                                    self.var_Phone_no.get(),
                                                                                                    self.var_Address.get(),
                                                                                                    self.var_radio1.get()

                                                                                                ))
                conn.commit()
                self.fetch_data()
                conn.close()
                messagebox.showinfo("Success","Person details has been added",parent=self.root)

            except Exception as es:
                messagebox.showerror("Error",f"Due To:{str(es)}",parent=self.root)


 #### fetch data
    def fetch_data(self):
        conn=mysql.connector.connect(host="localhost",username="root",password="Dimpy#1609*",database="face_recognizer")
        my_cursor=conn.cursor()
        my_cursor.execute("select * from person")
        data=my_cursor.fetchall()

        if len(data)!=0:
            self.person_table.delete(*self.person_table.get_children())
            for i in data:
                self.person_table.insert("", END,values=i)
            conn.commit()
        conn.close()



#####get_cursor
    def get_cursor(self,event=""):
        cursor_focus=self.person_table.focus()
        content=self.person_table.item(cursor_focus)
        data=content["values"]

        self.var_Department.set(data[0]),
        self.var_Course.set(data[1]),
        self.var_Year.set(data[2]),
        self.var_Age.set(data[3]),
        self.var_Person_ID.set(data[4]),
        self.var_Name.set(data[5]),
        self.var_Gender.set(data[6]),
        self.var_Phone_no.set(data[7]),
        self.var_Address.set(data[8]),
        self.var_radio1.set(data[9])



####Generate data set and take a photo sample #####
    def generate_dataset(self):
        if self.var_Department.get()=="Select Department" or self.var_Name.get()=="" or self.var_Person_ID.get()=="":
            messagebox.showerror("Error","All Fields are required",parent=self.root)
        else:
            try:
                conn=mysql.connector.connect(host="localhost",username="root",password="Dimpy#1609*",database="face_recognize
                my_cursor=conn.cursor()
                my_cursor.execute("Select * from person")
                myresult=my_cursor.fetchall()
                id=0
                for x in myresult:
                    id+=1
                my_cursor.execute("update person set Department=%s,Course=%s,Year=%s,Age=%s,Name=%s,Gender=%s,Phone_no=%s,Add



                conn.commit()
                self.fetch_data()
                conn.close()

                ######load predefined data
                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)
                    #scalinf 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:
```

```
                    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=Person(root)
    root.mainloop()
```

In [ ]:

In [ ]:

In [ ]: