**Program source Code:**

print("Kanderi Guruteja") #Student Name  
print("Open source Programming-Python") #Course name  
print("Lab-07") #Lab Number  
print("Working with tkinter") # Name of teh lAB  
print("A20526883") #Student ID  
  
# Import required modules  
from tkinter import \*  
from tkinter import messagebox  
import pickle  
import os  
import re  
import time  
import time  
from datetime import date  
  
# Check if the contacts.pickle file exists  
if os.path.exists('contacts.pickle'):  
 # contacts.pickle file for serialization and deserialization  
 with open('contacts.pickle', 'rb') as f:  
 contactlist1 = pickle.load(f)  
else:  
 # If contacts.pickle doesn't exist, load the contacts from contacts.py  
 from contacts import contactlist1  
  
def selection(): ## Function to get the selected index from the listbox  
 selected\_indices1 = select.curselection()  
 if selected\_indices1:  
 return int(selected\_indices1[0])  
 else:  
 return -1  
  
def addContact(): ## Function to add a new contact  
 nameofcontact = nameVariable1.get()  
 phoneofcontact = phone\_Variables1.get()  
  
 if nameofcontact and phoneofcontact:  
 for contact in contactlist1:  
 if contact[0] == nameofcontact:  
 messagebox.showerror("Error", "Contact with the same name already exists.") #Throws exception if the name is already existis  
 return  
  
 if not re.match("^[a-zA-Z0-9, -]+$", nameofcontact):  
 messagebox.showerror("Error", "Invalid characters in the name. Please use only letters, numbers, hyphen, comma, or space.")  
 #Make sure no speacial characters are inluded while entering name  
 elif not re.match("^[0-9-]+$", phoneofcontact) or len(phoneofcontact) < 10:  
 messagebox.showerror("Error", "Invalid phone number. Please enter at least 10 digits and use only numbers and hyphen.")  
 else:  
 contactlist1.append([nameofcontact, phoneofcontact])  
 setList()  
 saveContacts() # Save the updated contactlist  
 messagebox.showinfo("Success", "Contact added successfully.") #print contact added successfully in pop up screen  
 else:  
 messagebox.showerror("Error", "Please enter a name and phone number.")  
  
  
def updateContact():  
# Function to update a contact  
 selected\_index = selection()  
 if selected\_index >= 0:  
 name = nameVariable1.get()  
 phone = phone\_Variables1.get()  
 if name and phone:  
 if not re.match("^[a-zA-Z0-9,-]+$", name):  
 messagebox.showerror("Error", "Invalid characters in the name. Please use only letters, numbers, hyphen, or comma.")  
 else:  
 contactlist1[selected\_index] = [name, phone]  
 setList()  
 saveContacts() # Save the updated contactlist  
 messagebox.showinfo("Success", "Contact updated successfully.")  
 else:  
 messagebox.showerror("Error", "Please select a contact and load it before updating.")  
 else:  
 messagebox.showerror("Error", "No contact selected for update.")  
  
  
def deleteContact(): # Function to delete a contact  
 selected\_indexof = selection()  
 if selected\_indexof >= 0:  
 del contactlist1[selected\_indexof]  
 setList()  
 saveContacts() # Save the updated contactlist  
 messagebox.showinfo("Success", "Contact deleted successfully.")  
 else:  
 messagebox.showerror("Error", "No contact selected.")  
  
def loadContact(): # Function to load a contact  
 selected\_index2 = selection()  
 if selected\_index2 >= 0:  
 name, phone = contactlist1[selected\_index2]  
 nameVariable1.set(name)  
 phone\_Variables1.set(phone)  
  
def buildFrame(): # Function to build the GUI frame  
 global nameVariable1, phone\_Variables1, select  
 root = Tk()  
 root.title("My Contact List")  
  
 frame1 = Frame(root)  
 frame1.grid(row=0, column=0)  
 Label(frame1, text="Name:").grid(row=0, column=0, sticky=W)  
 nameVariable1 = StringVar()  
 name = Entry(frame1, textvariable=nameVariable1)  
 name.grid(row=0, column=1, sticky=W)  
 Label(frame1, text="Phone:").grid(row=1, column=0, sticky=W)  
 phone\_Variables1 = StringVar()  
 phone = Entry(frame1, textvariable=phone\_Variables1)  
 phone.grid(row=1, column=1, sticky=W)  
# Various buttons for different functionalities like adding contact, updating contact, delete contact, load contact, save contact  
 frame2 = Frame(root)  
 frame2.grid(row=1, column=0)  
 btn1 = Button(frame2, text=" Add ", command=addContact)  
 btn1.grid(row=0, column=0, padx=5)  
 btn2 = Button(frame2, text="Update", command=updateContact)  
 btn2.grid(row=0, column=1, padx=5)  
 btn3 = Button(frame2, text="Delete", command=deleteContact)  
 btn3.grid(row=0, column=2, padx=5)  
 btn4 = Button(frame2, text=" Load ", command=loadContact)  
 btn4.grid(row=0, column=3, padx=5)  
 btn5 = Button(frame2, text="Save", command=saveContacts)  
 btn5.grid(row=0, column=4, padx=5)  
  
 frame3 = Frame(root)  
 frame3.grid(row=2, column=0, pady=10)  
 scroll = Scrollbar(frame3, orient=VERTICAL)  
 select = Listbox(frame3, yscrollcommand=scroll.set, height=10)  
 scroll.config(command=select.yview)  
 scroll.pack(side=RIGHT, fill=Y)  
 select.pack(side=LEFT, fill=BOTH)  
  
 btn6 = Button(root, text="Exit", command=exitProgram) # Exit button  
 btn6.grid(row=3, column=0, pady=10) # Add the Exit button  
  
 return root  
  
def setList(): # Function to update the listbox with the contact names  
 contactlist1.sort()  
 select.delete(0, END)  
 for contact in contactlist1:  
 select.insert(END, contact[0])  
  
  
  
def saveContacts(): # Function to save the contact list to a file  
 with open('contacts.pickle', 'wb') as f:  
 pickle.dump(contactlist1, f)  
 current\_time = time.strftime("%H:%M:%S")  
 current\_date = date.today().strftime("%Y-%m-%d")  
 messagebox.showinfo("Save", f"Contacts saved successfully.\nDate: {current\_date}\nTime: {current\_time}")  
  
  
 # messagebox.showinfo("Save", "Contacts saved successfully.")  
  
def exitProgram(): # Function to exit the program  
 if messagebox.askokcancel(title="Exit", message="Are you sure you want to exit?"):  
 os.\_exit(1)  
  
root = buildFrame()  
setList()  
root.protocol("WM\_DELETE\_WINDOW", exitProgram)  
root.mainloop()

**OUTPUT:**

1. **Entire Interface including SAVE Button**

**A screenshot of a computer program

Description automatically generated**

**2. Snap of list box at run time:**

**A screenshot of a computer

Description automatically generated**

**3. Showing contact being added:**

**A screenshot of a computer

Description automatically generated**

**4. Contact being deleted:**

**A screenshot of a computer

Description automatically generated**

**5. Contact being Updated:**

**After update(Entry: Guruteja , Kanderi updated to Guruteja,K )**

**A screenshot of a computer

Description automatically generated**

**6. Modified contact list from contacts.py**

**A screenshot of a computer

Description automatically generated**

**Grad Requirements:**

**1.Completed message box**

**A screenshot of a computer

Description automatically generated**

**2. GUI added with EXIT button added**

**A screenshot of a computer

Description automatically generated**