VISVESVARAYA TECHNOLOGICAL UNIVERSITY
JNANASANGAMA, BELAGAVI – 590018

Mini Project Report
On

Phonebook

Submitted in partial fulfillment for the award of degree of

Bachelor of Engineering
In
Computer Science and Engineering

Submitted by
Meghana R – 1RF19CS031
Pallavi K J – 1RF19CS037
Shubham Luharuka – 1RF19CS050

RV Institute of Technology and Management®

(Affiliated to VTU, Belagavi)

JP Nagar, Bengaluru - 560076

Department of Computer Science and Engineering

RV Institute of Technology and Management®

(Affiliated to VTU, Belagavi)

JP Nagar, Bengaluru - 560076

Department of Computer Science and Engineering

CERTIFICATE

Certified that the Mini project entitled "**Phonebook**" carried out by

Meghana R – 1RF19CS031
Pallavi K J – 1RF19CS037
Shubham Luharuka – 1RF19CS050

are bonafied students of III Semester B.E, **RV Institute of Technology and Management** in partial fulfilment for the Bachelor of Engineering in COMPUTER SCIENCE AND ENGINEERING, of the **Visvesvaraya Technological University,** Belagavi, during the academic year 2020 - 2021. The Mini project report has been approved as it satisfies the academic requirements in respect of Data Structures and Applications.

Dr. Deepak N A
Associate Professor
Dept. CSE
RVITM, Bengaluru – 560076

Dr. Savitha G
Associate Professor
Dept. CSE
RVITM, Bengaluru - 560076

ABSTRACT

Phonebook is a GUI (Graphical User Interface) application written in python programming language. It performs a set of operations which include adding a new record, searching for an existing record by using the details of one of the three fields (i.e name, USN or phone number) provided by the user, updating, deleting records and displaying the existing records based on the branch specified by the user. It makes use of arrays and lists. It makes it easier for the user to operate the application as it's faster and efficient in performing the operations required by the user. An sorting algorithm is used to sort all the contact saved the database server. Phonebook is used by various organisation to keep the essential contact of their staff members. Some big organisation where there are a lot of human resources want such type of application which can do sorting and searching in less time. Asymptotic notation will help us to understand the time and space complexity of any algorithm.

Table of Contents

CONTENTS **PageNo.**

Chapter 1
INTRODUCTION

1.1 Phonebook

Phonebook is a directory where you can perform simple operations such as adding new records, listing them, updating them and searching for the contacts saved, and deleting them. Phonebook allows you to store contact information and retrieve and update them when required. It provides GUI, making it easier for the users to interact with the application. After adding of new contact the application automatically sort their data, and while searching application perform binary search if user search with the ID and if it is a other argument then it perform linear search.

1.2 Introduction to the application

The phonebook application allows the user to store names, phone numbers and many other details. It's a GUI (Graphical User Interface) application written in python programming language, which allows the user to add, update, search, and delete records.

Information such as first name, last name, USN, gender, Email ID, phone number and branch name are asked while adding a record into the Phonebook. These records are then displayed in the existing contacts list. The user can search a record in the Phonebook by entering the name, USN or phone number. When the search for a record is successful, the user has options to update or delete the record searched.

The user can view the list of records present in a particular branch by clicking on the Display button and entering the branch name. User can also export the data in csv file. This feature is in the Display frame.

1.3 Relevance of the application

The phonebook application lets the user to store complete information of a contact. It's easier for the user to search a record by typing the information of the selected field (name, USN or phone number). It allows the user to modify the entered details, to add new contacts or records, and to delete the existing records. A phonebook directory is a useful resource when looking for phone numbers and other important details of an individual or organization.

Chapter 2:

DESIGN-ALGORITHM

Step 1: Start
Step 2: Print Menu
Step 3: While Close Button not clicked
Step 4: If Add Button Clicked
Input First Name, Last name, USN, DOB, Mobile No., Gender, Branch, Email
Update the data server
Sort Data according to USN (Quick Sort)
Save Data in Existing Contacts
(end if)
Step 5: If Display Button clicked
Input Branch
If Show Button clicked
Display Existing Contacts of Branch
Else if Export button clicked
Save all data to a csv file
(end if)
(end if)
Step 6: If Search Button clicked
Input USN/Name/Mobile No.
Display contact from Existing Contacts (Binary Search)
Step 7: If Update Button clicked
Input First Name, Last name, USN, DOB, Mobile No., Gender, Branch, Email
Update the data server
Sort Data according to USN (Quick Sort)
Save data in Existing contacts
(end if)
Step 8: If Delete Button clicked
Delete contact from database
Sort Data according to USN (Quick Sort)
(end if)
(end if)
(end while)
Step 9: Stop

Quick Sort Algorithm:

Step 1: Start

Step 2: Make the right-most index value pivot

Step 3: Partition the array using pivot value

Step 4: Quick sort left partition recursively

Step 5: Quick sort right partition recursively

Step 6: Stop

FLOW-CHART

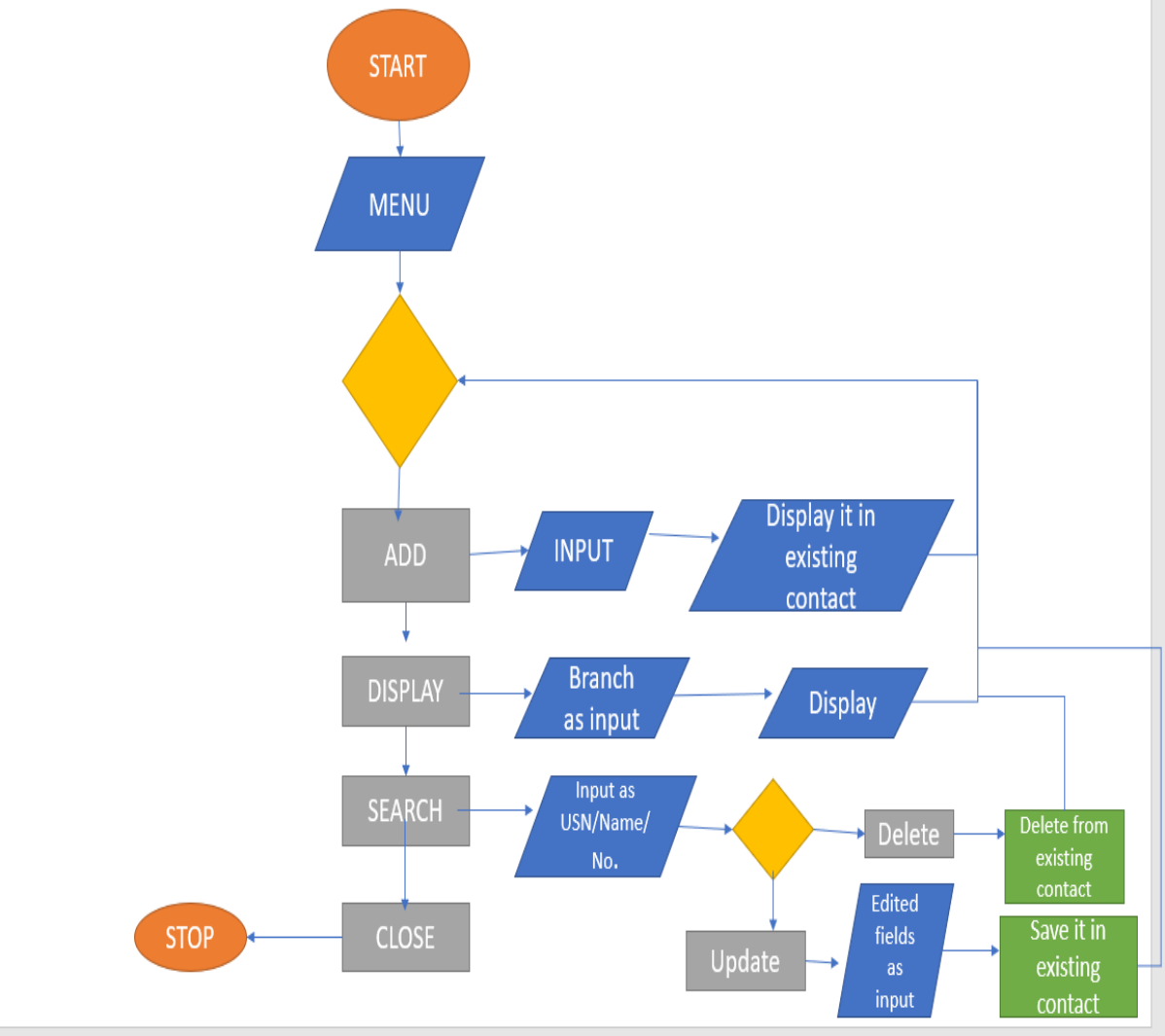


Figure no. 2.1- Flow Chart of Application

Chapter-3:
IMPLEMENTATION-DETAILS / CODE

```
#import all essential libraries
import tkinter as tk
from tkinter import ttk
from tkinter import *
from tkinter.ttk import *
from tkinter.filedialog import asksaveasfile
import os
from PIL import ImageTk, Image
import random
from tkinter import messagebox
from csv import writer
import pandas as pd
#make a small dataset
Student_data={"USN":["1RF19CS050","1RF19CS037","1RF19CS031"],
            "FIRST_NAME":["SHUBHAM","PALLAVI","MEGHANA"],
            "LAST_NAME":["LUHARUKA","K J","R"],
             "MOBILE":['9608757928','9874563210','9632587410'],
              "GENDER":["M","F","F"],
              "DOB":["18-11-2000","25-04-2001","18-05-2000"],
              "BRANCH":["Computer Science and Engineering","Computer Science and
          Engineering","Computer Science and Engineering"],
                    "EMAIL":['shubhaml_cs19.rvitm@rvei.edu.in','pallavkj.rvitm@rvei.edu.in','megh
                    nar_cs19.rvitm@rvei.edu.in'],}
Student_data=pd.DataFrame(Student_data)

#functions to raise a frame according to each button clicked
def add_frame():
   show_frame(frame1)
def search_frame():
   show_frame(frame2)
def update_frame():
   show_frame(frame3)
   display_in_frame3()
def delete_frame():
   show_frame(frame4)
def showall_frame():
   show_frame(frame5)
```

```
def sort_dataframe(data):
    data=data.sort_values(["USN"],kind='quicksort')
    data=data.reset_index(drop=True)
    return data
#function to do sorting after adding new contact
def quicksort(array):
    Sorted_data=pd.DataFrame(columns=Student_data.columns)
    def partition(array, start, end):
        pivot = array[start]
        low = start + 1
        high = end
        while True:
            while low <= high and array[high] >= pivot:
                high = high - 1
            while low <= high and array[low] <= pivot:
                low = low + 1
            if low <= high:
                array[low],array[high]=array[high],array[low]
            else:
                break
        array[start], array[high] = array[high], array[start]
        return high
    def quick_sort(array, start, end):
        if start >= end:
            return
        p = partition(array, start, end)
        quick_sort(array, start, p-1)
        quick_sort(array, p+1, end)
    quick_sort(array, 0, len(array) - 1)
    for i in range(len(array)):
        a=Student_data[Student_data["USN"]==array[i]]
        a=pd.DataFrame(a.values,columns=Student_data.columns)
        Sorted_data=Sorted_data.append(a,ignore_index=True)
    return Sorted_data
# configure a frame
window = tk.Tk()
window.geometry('1200x600')
window.resizable(0,0)
window.rowconfigure(0,weight=1)
window.columnconfigure(0,weight=1)
frame1 = tk.Frame(window)
```

```
frame2 = tk.Frame(window)
frame3 = tk.Frame(window)
frame4 = tk.Frame(window)
frame5 = tk.Frame(window)
for frame in (frame1,frame2,frame3,frame4,frame5):
    frame.grid(row=0,column=0,sticky="nsew")
def show_frame(frame):
    frame.tkraise()
# declaration of variables used while making GUI
select=tk.StringVar()
selected_branch=tk.StringVar()
first_name1=tk.StringVar()
last_name1=tk.StringVar()
usn1=tk.StringVar()
dob1=tk.StringVar()
mobile1=tk.StringVar()
gender1=tk.StringVar()
branch1=tk.StringVar()
email1=tk.StringVar()
updated_dob=tk.StringVar()
updated_mobile=tk.StringVar()
updated_gender=tk.StringVar()
updated_branch=tk.StringVar()
updated_email=tk.StringVar()
# function to display after frame 3 raise
def display_in_frame3():
    selection=select.get()
    search_element=frame21_entry.get()
    if selection=="name":
        a=Student_data[Student_data["FIRST_NAME"]==search_element.upper()]
    elif selection=="usn":
        a=Student_data[Student_data["USN"]==search_element.upper()]
    else:
        a=Student_data[Student_data["MOBILE"]==(str(search_element)).upper()]
    if a.empty:
        messagebox.showinfo("Information","No Such record found")
        frame31_label.config(text="NAME:-")
        frame32_label.config(text="USN:-")
        frame33_label.config(text="MOBILE:-")
        frame34_label.config(text="DOB:-")
        frame35_label.config(text="GENDER:-")
```

```
        frame36_label.config(text="BRANCH:-")
        frame37_label.config(text="EMAIL:-")
    else:
        a=a.reset_index(drop=True)
            frame31_label.config(text="NAME:-"+"\t"+a["FIRST_NAME"][0]+"
            "+a["LAST_NAME"][0])
        frame32_label.config(text="USN:-"+"\t"+a["USN"][0])
        frame33_label.config(text="MOBILE:-"+"\t"+str(a["MOBILE"][0]))
        frame34_label.config(text="DOB:-"+"\t"+a["DOB"][0])
        frame35_label.config(text="GENDER:-"+"\t"+a["GENDER"][0])
        frame36_label.config(text="BRANCH:-"+"\t"+a["BRANCH"][0])
        frame37_label.config(text="EMAIL:-"+"\t"+a["EMAIL"][0])
        frame38_label.config(text=a["FIRST_NAME"][0])
        frame39_label.config(text=a["LAST_NAME"][0])
        frame310_label.config(text=a["USN"][0])
```

 # function to add data and sort it

```
def add_data():
    def remove(string):
        return ("".join(string.split())).upper()
    no_of_empty_count=0
    first_name=first_name1.get()
    last_name=last_name1.get()
    usn=usn1.get()
    dob=str(dob1.get())
    mobile=mobile1.get()
    gender=gender1.get()
    branch=branch1.get()
    email=email1.get()
elements=[remove(usn),remove(first_name),remove(last_name),remove(mobile),remove(gender)
,dob,remove(branch),remove(email).lower()]
    for x in elements:
        if len(x)==0:
            messagebox.showinfo("Information","Some boxes are still Empty")
            no_of_empty_count+=1
            break
    if no_of_empty_count==0:
        Student_data.loc[len(Student_data.index)] = elements
        messagebox.showinfo("Information","Contact successfully added")
        first_name1.set("")
        last_name1.set("")
```

```
        usn1.set("")
        dob1.set("")
        mobile1.set("")
        frame11_combobox.current(0)
        frame12_combobox.current(0)
        email1.set("")
        sort_dataframe(Student_data)
# function to search for contact
def search_func():
    selection=select.get()
    search_element=frame21_entry.get()
    if selection=="name":
        a=Student_data[Student_data["FIRST_NAME"]==search_element.upper()]
    elif selection=="usn":
        a=Student_data[Student_data["USN"]==search_element.upper()]
    else:
        a=Student_data[Student_data["MOBILE"]==(str(search_element)).upper()]
    if a.empty:
        messagebox.showinfo("Information","No Such record found")
        frame21_label.config(text="NAME:-")
        frame22_label.config(text="USN:-")
        frame23_label.config(text="MOBILE:-")
        frame24_label.config(text="DOB:-")
        frame25_label.config(text="GENDER:-")
        frame26_label.config(text="BRANCH:-")
        frame27_label.config(text="EMAIL:-")
    else:
        a=a.reset_index(drop=True)
        frame21_label.config(text="NAME:-"+"\t"+a["FIRST_NAME"][0]+"
"+a["LAST_NAME"][0])
        frame22_label.config(text="USN:-"+"\t"+a["USN"][0])
        frame23_label.config(text="MOBILE:-"+"\t"+str(a["MOBILE"][0]))
        frame24_label.config(text="DOB:-"+"\t"+a["DOB"][0])
        frame25_label.config(text="GENDER:-"+"\t"+a["GENDER"][0])
        frame26_label.config(text="BRANCH:-"+"\t"+a["BRANCH"][0])
        frame27_label.config(text="EMAIL:-"+"\t"+a["EMAIL"][0])
# function to display contact according department wise
def Showall():
    class A(Frame):
        def __init__(self, parent):
            Frame.__init__(self, parent)
```

```
        self.CreateUI()
        self.LoadTable()
        self.grid(sticky=(N, S, W, E))
        parent.grid_rowconfigure(0, weight=1)
        parent.grid_columnconfigure(0, weight=1)
    def CreateUI(self):
        tv= Treeview(self)
        tv['columns']=('USN','NAME','MOBILE','GENDER','DOB','BRANCH','EMAIL')
        tv.heading('#0',text='USN',anchor='center')
        tv.column('#0',anchor='center')
        tv.heading('#1', text='NAME', anchor='center')
        tv.column('#1', anchor='center')
        tv.heading('#2', text='MOBILE', anchor='center')
        tv.column('#2', anchor='center')
        tv.heading('#3', text='GENDER', anchor='center')
        tv.column('#3', anchor='center')
        tv.heading('#4', text='DOB', anchor='center')
        tv.column('#4', anchor='center')
        tv.heading('#5', text='BRANCH', anchor='center')
        tv.column('#5', anchor='center')
        tv.heading('#6', text='EMAIL', anchor='center')
        tv.column('#6', anchor='center')
        tv.grid(sticky=(N,S,W,E))
        self.treeview = tv
        self.grid_rowconfigure(0,weight=1)
        self.grid_columnconfigure(0,weight=1)
    def LoadTable(self):
        selected=selected_branch.get()
        a=Student_data[Student_data["BRANCH"]==selected]
        USN=""
        NAME=""
        MOBILE=""
        GENDER=""
        DOB=""
        BRANCH=""
        EMAIL=""
        for ind in a.index:
            USN=a['USN'][ind]
            NAME=a['FIRST_NAME'][ind]+" "+a['LAST_NAME'][ind]
            MOBILE=a['MOBILE'][ind]
            GENDER=a['GENDER'][ind]
```

```
            DOB=a['DOB'][ind]
            BRANCH=a['BRANCH'][ind]
            EMAIL=a['EMAIL'][ind]
self.treeview.insert("",'end',text=USN,values=(NAME,MOBILE,GENDER,DOB,BRANCH,EM
AIL))
    frame6=Tk()
    frame6.title("Overview Page")
    A(frame6)
# function to exit programme
def ExitApplication():
    MsgBox = tk.messagebox.askquestion ('Delete Contact','Are you sure to delete the
contact',icon = 'warning')
    if MsgBox == 'yes':
       delete_data()
    else:
        search_frame()

#function to export data to csv
def save():
    branch=selected_branch.get()

data=pd.DataFrame(Student_data[Student_data["BRANCH"]==branch],columns=Student_data.c
olumns)
    files = [('CSV', '*.csv'),
            ('Text Document', '*.txt')]
    file = asksaveasfile(filetypes = files, defaultextension = files)
    a=list(data["USN"].array)
    b=list(data["FIRST_NAME"].array)
    c=list(data["LAST_NAME"].array)
    d=list(data["MOBILE"].array)
    e=list(data["DOB"].array)
    f=list(data["GENDER"].array)
    g=list(data["BRANCH"].array)
    i=list(data["EMAIL"].array)

df=pd.DataFrame({"USN":a,"FIRST_NAME":b,"LAST_NAME":c,"MOBILE":d,"DOB":e,"GE
NDER":f,"BRANCH":g,"EMAIL":i})
    df.to_csv(r'{}'.format(file.name))
# function to delete data
def delete_data():
    selection=select.get()
```

```
    search_element=frame21_entry.get()
    if selection=="name":
        Student_data.drop(Student_data.index[Student_data["FIRST_NAME"]==
search_element],axis=0,inplace=True)
    elif selection=="usn":

Student_data.drop(Student_data.index[Student_data["USN"]==search_element],axis=0,inplace=
True)
    else:

Student_data.drop(Student_data.index[Student_data["MOBILE"]==search_element],axis=0,inpl
ace=True)
#function to update data
def update_data():
    selection=select.get()
    search_element=frame21_entry.get()
    if selection=="name":
        a=Student_data[Student_data["FIRST_NAME"]==search_element.upper()].index
    elif selection=="usn":
        a=Student_data[Student_data["USN"]==search_element.upper()].index
    else:
        a=Student_data[Student_data["MOBILE"]==(str(search_element)).upper()].index
    Student_data.at[a,"MOBILE"]=updated_mobile.get()
    Student_data.at[a,"DOB"]=updated_dob.get()
    Student_data.at[a,"GENDER"]=updated_gender.get()
    Student_data.at[a,"BRANCH"]=updated_branch.get()
    Student_data.at[a,"EMAIL"]=updated_email.get()
    tk.messagebox.showinfo("Information","Contact Updated")
    updated_mobile.set("")
    updated_dob.set("")
    updated_gender.set("")
    updated_branch.set("")
    updated_email.set("")
# code for frame 1 (ADD data)
image1=Image.open(r"3RDSEMFRAME1.jpg")
image1=image1.resize((1200,600),Image.ANTIALIAS)
img1 = ImageTk.PhotoImage(image1)
panel1 = tk.Label(frame1, image = img1)
panel1.pack()
```

```
frame11_entry=tk.Entry(frame1,width=50,borderwidth=8,textvariable=first_name1)
frame11_entry.place(relx=0.5,rely=0.31)

frame12_entry=tk.Entry(frame1,width=50,borderwidth=8,textvariable=last_name1)
frame12_entry.place(relx=0.5,rely=0.3705)

frame13_entry=tk.Entry(frame1,width=50,borderwidth=8,textvariable=usn1)
frame13_entry.place(relx=0.5,rely=0.425)

frame14_entry=tk.Entry(frame1,width=50,borderwidth=8,textvariable=dob1)
frame14_entry.place(relx=0.5,rely=0.478)

frame15_entry=tk.Entry(frame1,width=50,borderwidth=8,textvariable=mobile1)
frame15_entry.place(relx=0.5,rely=0.54)

frame11_combobox = ttk.Combobox(frame1, width = 45,height=25, textvariable = gender1)
frame11_combobox['values'] = ('SELECT YOUR GENDER.',
                              'MALE',
                              'FEMALE',
                              'TRANSGENDER',
                              'OTHER')

frame11_combobox.place(relx=0.5,rely=0.605)
frame11_combobox.current(0)

frame12_combobox = ttk.Combobox(frame1, width = 45,height=25, textvariable = branch1)
frame12_combobox['values'] = ('SELECT A DEPT.',
                              'Computer Science and Engineering',
                              'Information Science and Engineering',
                              'Electronics and Communication Engineering',
                              'Mechanical Engineering')

frame12_combobox.place(relx=0.5,rely=0.66)
frame12_combobox.current(0)

frame18_entry=tk.Entry(frame1,width=50,borderwidth=8,textvariable=email1)
frame18_entry.place(relx=0.5,rely=0.725)

frame11_button=tk.Button(frame1,
text="ADD",background='grey',width=15,height=2,command=add_frame)
frame11_button.place(relx=0.2,rely=0.85)
```

```
frame12_button=tk.Button(frame1,
text="SEARCH",background='yellow',width=15,height=2,command=search_frame)
frame12_button.place(relx=0.4,rely=0.85)

frame13_button=tk.Button(frame1,
text="DISPLAY",background='yellow',width=15,height=2,command=showall_frame)
frame13_button.place(relx=0.6,rely=0.85)

frame14_button=tk.Button(frame1,
text="EXIT",background='yellow',width=15,height=2,command=window.destroy)
frame14_button.place(relx=0.8,rely=0.85)

frame14_button=tk.Button(frame1,
text="DONE",background='yellow',width=5,height=2,command=add_data)
frame14_button.place(relx=0.8,rely=0.5)
# code for frame 2 (SEARCH data and give option to delete or update)
image2=Image.open(r"3RDSEMFRAME2.jpg")
image2=image2.resize((1200,600),Image.ANTIALIAS)
img2 = ImageTk.PhotoImage(image2)
panel2 = tk.Label(frame2, image = img2)
panel2.pack()

frame21_entry=tk.Entry(frame2,width=50,borderwidth=8)
frame21_entry.place(relx=0.1,rely=0.50)

R21_button = tk.Radiobutton(frame2, variable=select, value='name',background='white')
R21_button.place(relx=0.165,rely=0.38)
R21_button.select()

R22_button = tk.Radiobutton(frame2, variable=select, value='usn',background='white')
R22_button.place(relx=0.26,rely=0.38)

R23_button =tk. Radiobutton(frame2, variable=select, value='mobile',background='white')
R23_button.place(relx=0.41,rely=0.38)

image21=Image.open(r"search_button.jpg")
image21=image21.resize((30,30))
img21 = ImageTk.PhotoImage(image21)
frame21_button=tk.Button(frame2,
image=img21,background='white',width=30,height=30,command=search_func)
frame21_button.place(relx=0.36,rely=0.4925)
```

```
frame22_button=tk.Button(frame2, text="UPDATE",background='light
blue',width=15,height=2,command=update_frame)
frame22_button.place(relx=0.1,rely=0.625)

frame23_button=tk.Button(frame2, text="DELETE",background='light
blue',width=15,height=2,command=ExitApplication)
frame23_button.place(relx=0.25,rely=0.625)

frame24_button=tk.Button(frame2,
text="DISPLAY",background='yellow',width=15,height=2,command=showall_frame)
frame24_button.place(relx=0.6,rely=0.85)

frame25_button=tk.Button(frame2,
text="EXIT",background='yellow',width=15,height=2,command=window.destroy)
frame25_button.place(relx=0.8,rely=0.85)

frame26_button=tk.Button(frame2,
text="ADD",background='yellow',width=15,height=2,command=add_frame)
frame26_button.place(relx=0.2,rely=0.85)

frame27_button=tk.Button(frame2,
text="SEARCH",background='grey',width=15,height=2,command=search_frame)
frame27_button.place(relx=0.4,rely=0.85)

frame21_label=tk.Label(frame2,text="NAME:-",background='white',font=20)
frame21_label.place(relx=0.49,rely=0.4)

frame22_label=tk.Label(frame2,text="USN:-",background='white',font=20)
frame22_label.place(relx=0.49,rely=0.45)

frame23_label=tk.Label(frame2,text="MOBILE:-",background='white',font=20)
frame23_label.place(relx=0.49,rely=0.5)

frame24_label=tk.Label(frame2,text="DOB:-",background='white',font=20)
frame24_label.place(relx=0.49,rely=0.55)

frame25_label=tk.Label(frame2,text="GENDER:-",background='white',font=20)
frame25_label.place(relx=0.49,rely=0.6)

frame26_label=tk.Label(frame2,text="BRANCH:-",background='white',font=20)
frame26_label.place(relx=0.49,rely=0.65)
```

```
frame27_label=tk.Label(frame2,text="EMAIL:-",background='white',font=20)
frame27_label.place(relx=0.49,rely=0.7)
# code for frame3 (UPDATE contact)
image3=Image.open(r"3RDSEMFRAME3.jpg")
image3=image3.resize((1200,600),Image.ANTIALIAS)
img3 = ImageTk.PhotoImage(image3)
pane31 = tk.Label(frame3, image = img3)
pane31.pack()

frame38_label=tk.Label(frame3,text="hii",font=0,background='white')
frame38_label.place(relx=0.6,rely=0.34)

frame39_label=tk.Label(frame3,text="hii",font=0,background='white')
frame39_label.place(relx=0.6,rely=0.40)

frame310_label=tk.Label(frame3,text="hii",font=0,background='white')
frame310_label.place(relx=0.6,rely=0.46)

frame33_entry=tk.Entry(frame3,width=50,borderwidth=8,textvariable=updated_dob)
frame33_entry.place(relx=0.6,rely=0.52)

frame34_entry=tk.Entry(frame3,width=50,borderwidth=8,textvariable=updated_mobile)
frame34_entry.place(relx=0.6,rely=0.58)

frame35_entry=tk.Entry(frame3,width=50,borderwidth=8,textvariable=updated_gender)
frame35_entry.place(relx=0.6,rely=0.65)

frame36_entry=tk.Entry(frame3,width=50,borderwidth=8,textvariable=updated_branch)
frame36_entry.place(relx=0.6,rely=0.71)

frame37_entry=tk.Entry(frame3,width=50,borderwidth=8,textvariable=updated_email)
frame37_entry.place(relx=0.6,rely=0.77)

frame31_button=tk.Button(frame3, text="CANCEL",background='light
blue',width=15,height=2,command=search_frame)
frame31_button.place(relx=0.1,rely=0.725)

frame32_button=tk.Button(frame3,
text="ADD",background='yellow',width=15,height=2,command=add_frame)
frame32_button.place(relx=0.2,rely=0.85)
```

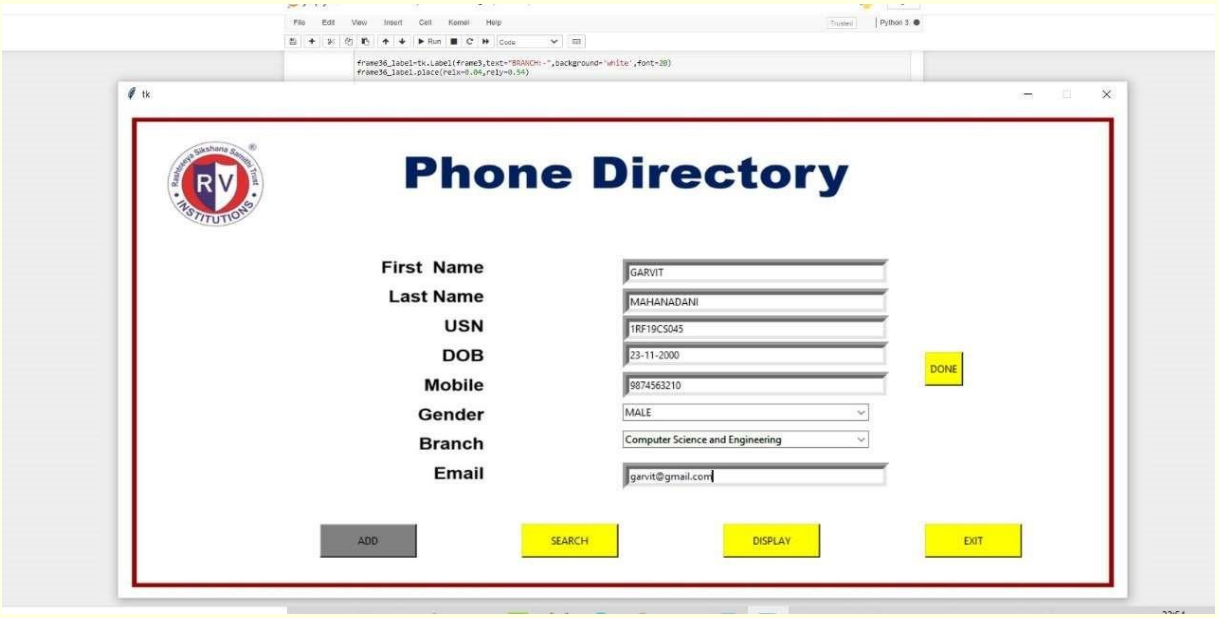
```
frame33_button=tk.Button(frame3,
text="SEARCH",background='grey',width=15,height=2,command=search_frame)
frame33_button.place(relx=0.4,rely=0.85)
frame34_button=tk.Button(frame3,
text="DISPLAY",background='yellow',width=15,height=2,command=showall_frame)
frame34_button.place(relx=0.6,rely=0.85)
frame35_button=tk.Button(frame3,
text="EXIT",background='yellow',width=15,height=2,command=window.destroy)
frame35_button.place(relx=0.8,rely=0.85)

frame36_button=tk.Button(frame3, text="CONFIRM",background='light
blue',width=15,height=2,command=update_data)
frame36_button.place(relx=0.25,rely=0.725)

frame31_label=tk.Label(frame3,text="NAME:-",background='white',font=20)
frame31_label.place(relx=0.04,rely=0.34)

frame32_label=tk.Label(frame3,text="USN:-",background='white',font=20)
frame32_label.place(relx=0.04,rely=0.38)

frame33_label=tk.Label(frame3,text="MOBILE:-",background='white',font=20)
frame33_label.place(relx=0.04,rely=0.42)

frame34_label=tk.Label(frame3,text="DOB:-",background='white',font=20)
frame34_label.place(relx=0.04,rely=0.46)

frame35_label=tk.Label(frame3,text="GENDER:-",background='white',font=20)
frame35_label.place(relx=0.04,rely=0.50)

frame36_label=tk.Label(frame3,text="BRANCH:-",background='white',font=20)
frame36_label.place(relx=0.04,rely=0.54)

frame37_label=tk.Label(frame3,text="EMAIL:-",background='white',font=20)
frame37_label.place(relx=0.04,rely=0.58)
```

 # Code for frame 4 (DISPLAY data branchwise and give option to export file to csv)

```
image5=Image.open(r"3RDSEMFRAME5.jpg")
image5=image5.resize((1200,600),Image.ANTIALIAS)
img5 = ImageTk.PhotoImage(image5)
pane51 = tk.Label(frame5, image = img5)
pane51.place(relx=0,rely=0)
```

```
branchchoose = ttk.Combobox(frame5, width = 100,height=25, textvariable = selected_branch)
branchchoose['values'] = ('SELECT A DEPT.',
                          'Computer Science and Engineering',
                        'Information Science and Engineering',
                        'Electronics and Communication Engineering',
                        'Mechanical Engineering')
branchchoose.place(relx=0.2,rely=0.4)
branchchoose.current(0)

frame51_button=tk.Button(frame5, text="SHOW",background='light
blue',width=15,height=2,command=Showall)
frame51_button.place(relx=0.4,rely=0.55)

frame50_button=tk.Button(frame5,text="EXPORT TO CSV",background='light
blue',width=15,height=2,command=save)
frame50_button.place(relx=0.8,rely=0.5)

frame52_button=tk.Button(frame5,
text="ADD",background='yellow',width=15,height=2,command=add_frame)
frame52_button.place(relx=0.2,rely=0.85)

frame53_button=tk.Button(frame5,
text="SEARCH",background='yellow',width=15,height=2,command=search_frame)
frame53_button.place(relx=0.4,rely=0.85)

frame54_button=tk.Button(frame5,
text="DISPLAY",background='grey',width=15,height=2,command=showall_frame)
frame54_button.place(relx=0.6,rely=0.85)

frame55_button=tk.Button(frame5,
text="EXIT",background='yellow',width=15,height=2,command=window.destroy)
frame55_button.place(relx=0.8,rely=0.85)
# start application
show_frame(frame1)
window.mainloop()
```

Chapter-4
OUTPUT SCREENSHOTS



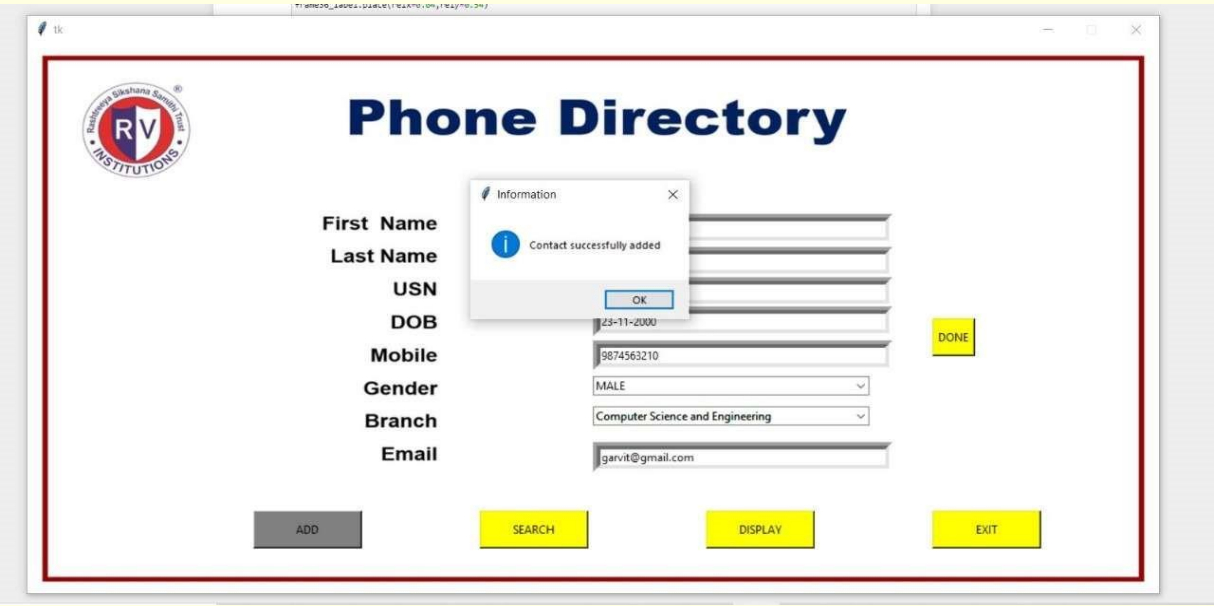
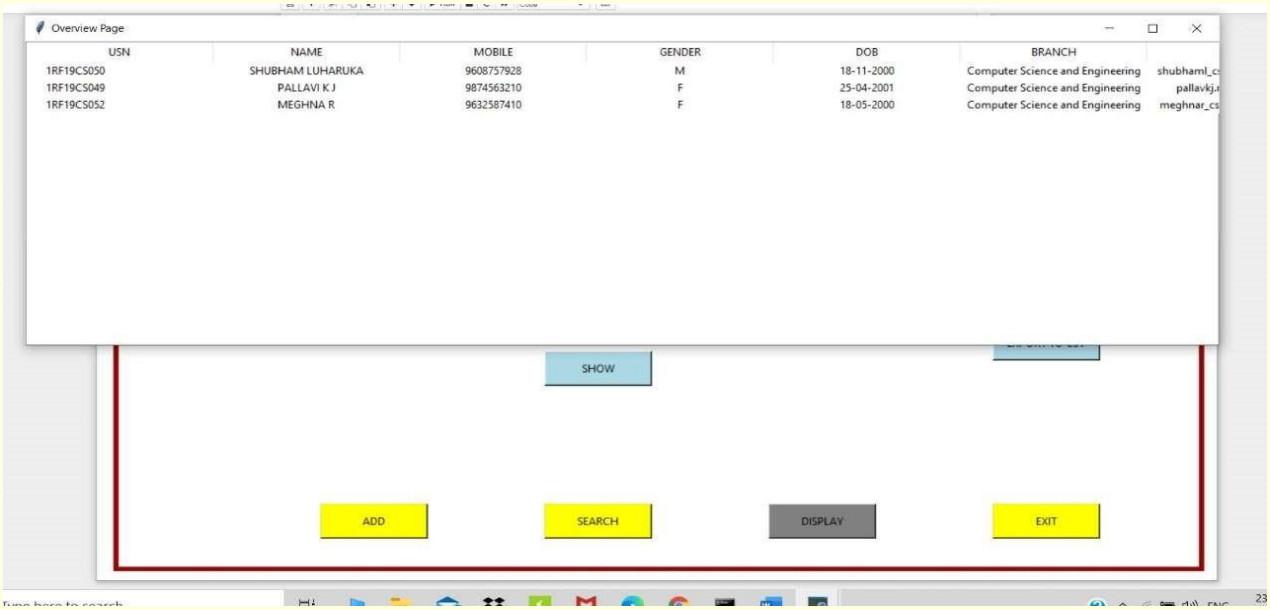


Figure no-4.1- Adding contact to Database



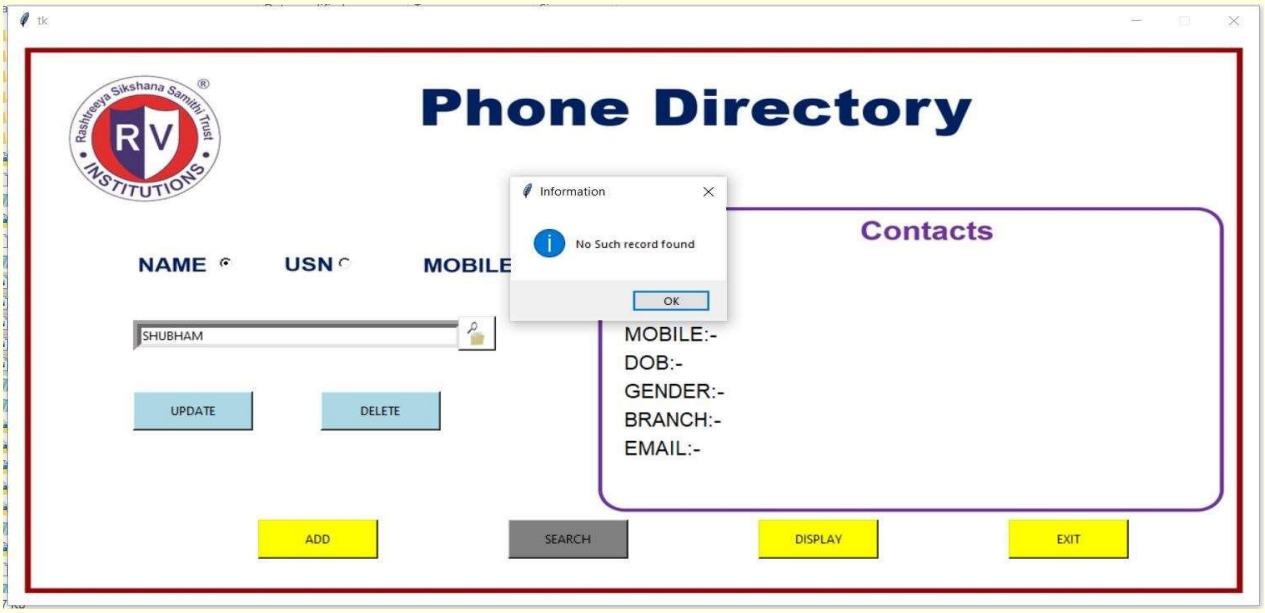


Figure no-4.2-Displaying contact branchwise

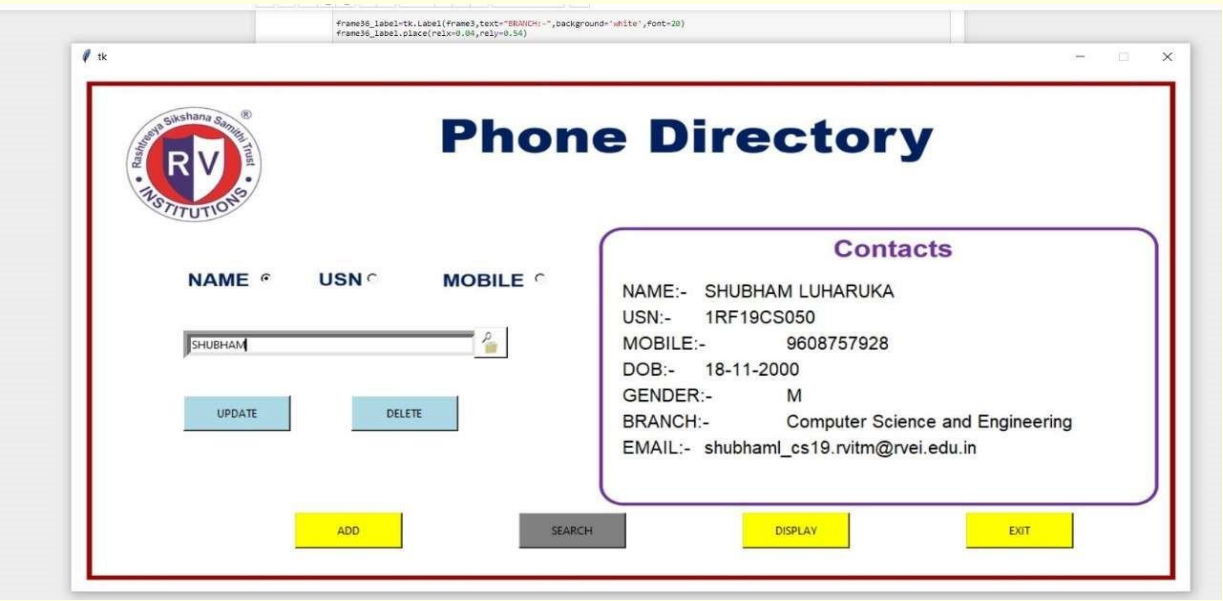


Figure-4.3-Searching and Updating in Database using (NAME/ USN/ MOBILE)

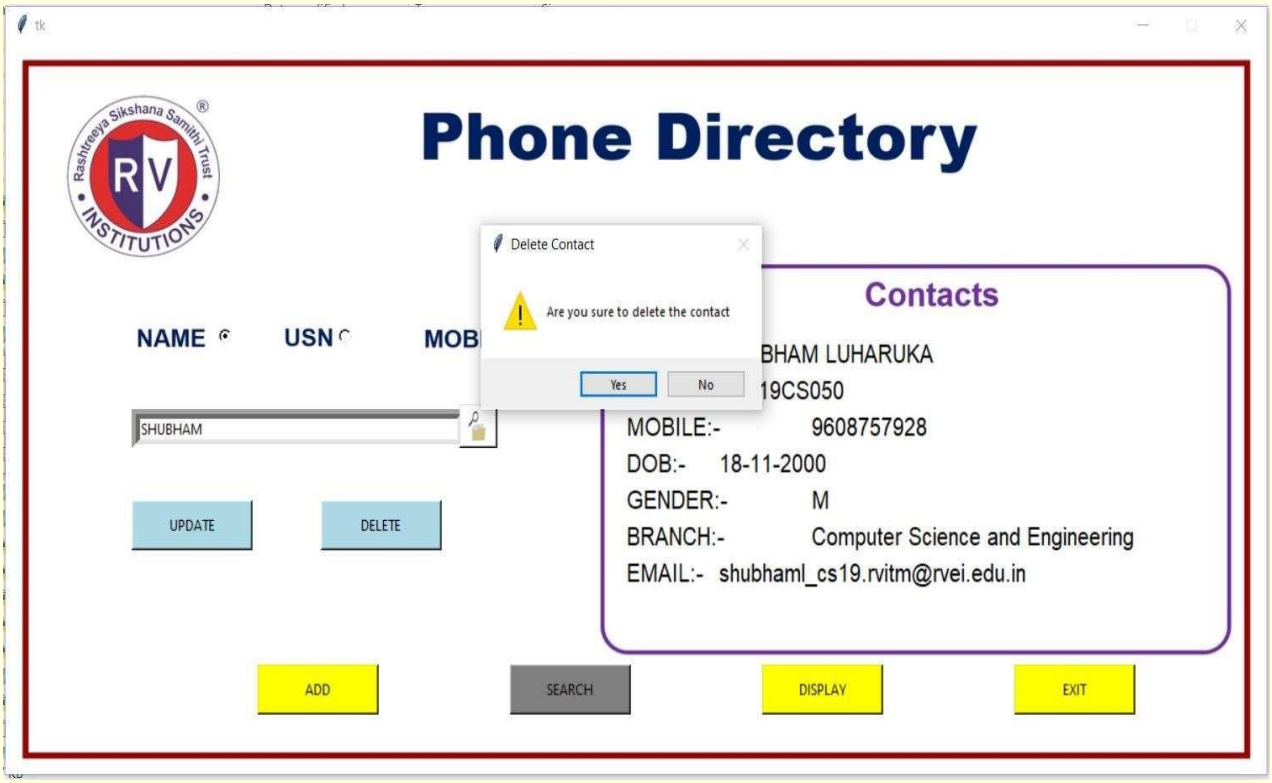


Figure-4.4-Deletion of Contact

Chapter-5:
CONCLUSION AND FUTURE EXTENSION

The phonebook application has been implemented using experimental cases and the programming language used is python. The application can be further improved by adding details like address, Aadhaar number, blood group, age, etc. The records can be sorted based on the age group. If we want to access data more easily, then we can use cloud storage instead of SQL data server. Further improvements can be made as per user requirement. We can use many other technologies like Blockchain and can implement many other Cryptographic method to make our application more immutable. As the research progress we can reduce its time as well as space complexity. We can also use Biometric authentication while updating the information on server.

Chapter-6:
REFERENCES:

1. By Kenneth Alfred Lambert," Fundamentals of Python - Data structures", CENGAGE Learning, 2013 edition
2. By Gayle Laakmann McDowell, "Cracking the Coding Interview",CareerCup,6th edition
3. https://www.learnpython.org/en/Dictionaries
4. https://medium.com/@mardiyyah/building-a-simple-phonebooklearnpythonthroughprojects-series-10-af56d527f463