HKBK COLLEGE OF ENGINEERING

Department of Electronics and Communication Engineering

Bangalore – 560045

BPLCK205B

INTRODUCTION TO PYTHON PROGRAMMING

PYTHON PROJECT REPORT

📘 Project Title:
Contact Book Database storage Project

👨‍🎓 Submitted by:
Sunanda (1HK24EC139)
(First Year – 2nd Semester)

👩‍🏫 Under the guidance of:
Dr. Sanjana Prasad
Associate Professor, Dept. of ECE
HKBK College of Engineering

🗓️ Academic Year: 2024 – 2025

H.K.B.K COLLEGE OF ENGINEERING

S.NO.22/1, NAGAWARA, BENGALURU-560045

DEPARTMENT OF ELECTRONICS AND COMMUNICATION

CERTIFICATE

Certified that the Python Project entitled “Contact Book Database storage Project” is a bonafide work carried out by the following students of First Year, 2nd Semester, Department of Electronics and Communication Engineering, HKBK College of Engineering, Bengaluru, during the academic year 2024–25.

This project work has been completed under the guidance of Dr. Sanjana Prasad, Associate Professor, Department of ECE, as part of their academic curriculum. The project has been reviewed and approved as it satisfies the academic requirements of the Department.

1. Sunanda (1HK24EC139)

Date: __23-06-2025_____

Place: Bengaluru

_____	_____
DR. SANJANA PRASAD	**DR. MANJUNATH R KOUNTE**
Project Guide	**Head of the Department**

Python Contact Book Project Details

Introduction:

Contact Book is a GUI based project using Tkinter and a message module used for storing information about some person like name and contact number. In this project we have some functionality like add, edit, delete, view, and reset contacts.

We can add new contacts by clicking on the add button and filling in the information of the new contact. We can update or edit existing contacts by selecting and clicking on the edit button. Similarly, we can delete and view the contact. Like this, we can store data.

Source Code:

```
#import library

from tkinter import *

from tkinter import messagebox#PythonGeeks address book - Initialize window

import webbrowser


def open_link():

    webbrowser.open("https://uidai.gov.in/en/my-aadhaar/update-aadhaar.html")

    webbrowser.open("
https://www.onlineservices.nsdl.com/paam/endUserRegisterContact.html")
```

```
root = Tk()
root.geometry('800x600')
root.config(bg = '#d3f3f5')
root.title('Python Contact Book')
root.resizable(0,0)
contactlist = [
    ['Siddharth Nigam','369854712', 'PAN0', '240'],
    ['Gaurav Patil', '521155222', 'PAN1', '321'],
    ['Abhishek Nikam', '78945614', 'PAN2', '123'],
    ]

Name = StringVar()
Number = StringVar()
PAN = StringVar()
Adhar = StringVar()

#Python - create frame
frame = Frame(root)
frame.pack(side = RIGHT)

scroll = Scrollbar(frame, orient=VERTICAL)
select = Listbox(frame, yscrollcommand=scroll.set, font=('Times new roman'
,16), bg="#f0fffc", width=20, height=20, borderwidth=3, relief= "groove")
scroll.config (command=select.yview)
scroll.pack(side=RIGHT, fill=Y)
```

```
select.pack(side=LEFT,  fill=BOTH, expand=1)
#Python - function to get select value


def Selected():
    print("hello",len(select.curselection()))
    if len(select.curselection())==0:
        messagebox.showerror("Error", "Please Select the Name")
    else:
        return int(select.curselection()[0])


#Python -fun to add new contact
def AddContact():
    if Name.get()!="" and Number.get()!="" and PAN.get()!="" and
Adhar.get()!="":
        contactlist.append([Name.get() ,Number.get() ,PAN.get() ,Adhar.get()])
        print(contactlist)
        Select_set()
        EntryReset()
        messagebox.showinfo("Confirmation", "Successfully Add New
Contact")


    else:
        messagebox.showerror("Error","Please fill the information")
def EntryReset():
    Name.set('')
```

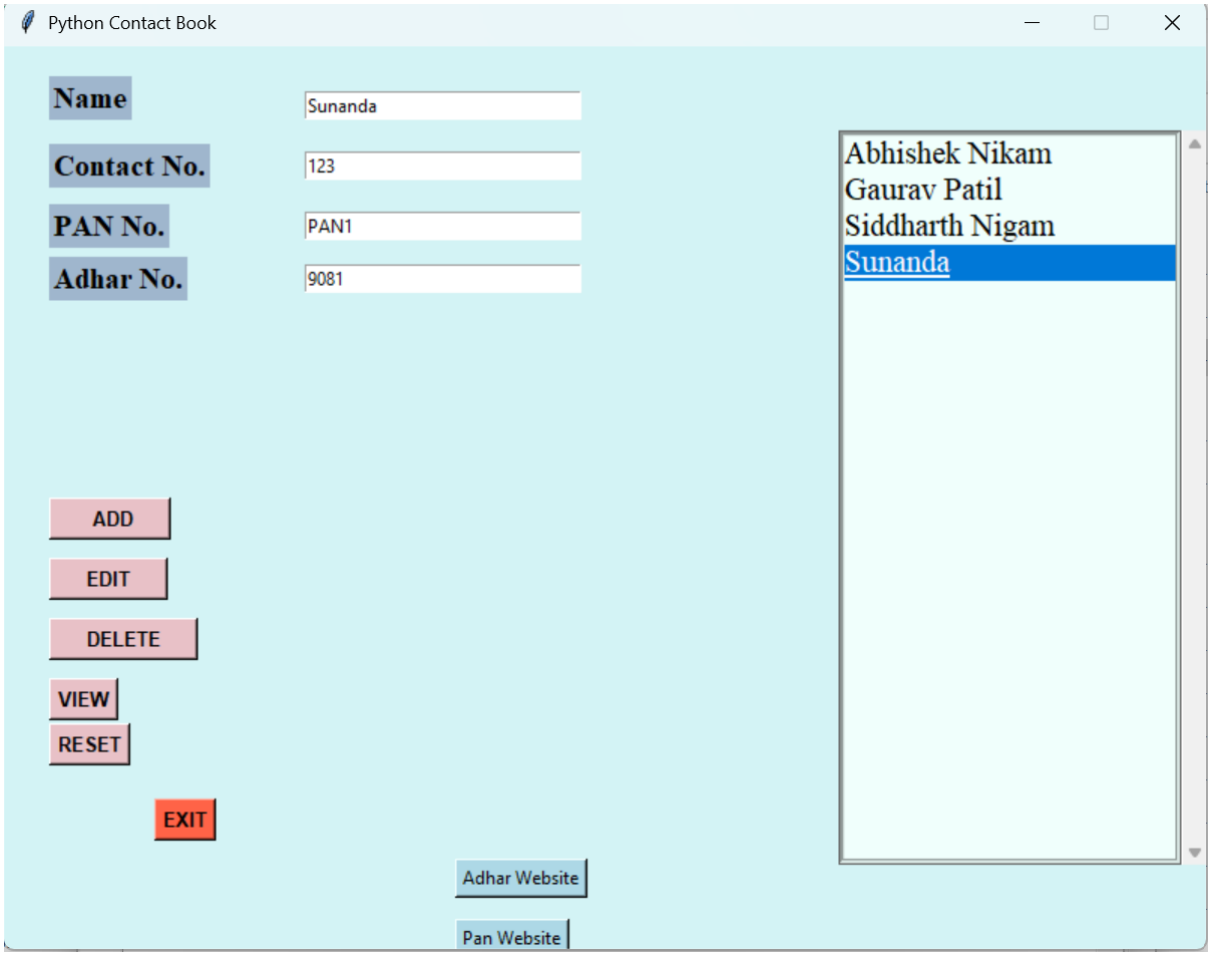
```
    Number.set('')
    PAN.set('')
    Adhar.set('')


def UpdateDetail():
    if Name.get() and Number.get() and PAN.get() and Adhar.get():
        contactlist[Selected()] = [Name.get(), Number.get(),
PAN.get(),Adhar.get()]


        messagebox.showinfo("Confirmation", "Successfully Update Contact")
        EntryReset()
        Select_set()



    elif not(Name.get()) and not(Number.get()) and not(PAN.get())  and
not(Adhar.get()) and(len(select.curselection())==0):
        messagebox.showerror("Error", "Please fill the information")


    else:
        if len(select.curselection())==0:
            messagebox.showerror("Error", "Please Select the Name and \n press
Load button")
        else:
            message1 = """To Load the all information of \n
                    selected row press Load button\n.
                    """
```

```
            messagebox.showerror("Error", message1)
def Delete_Entry():
    if len(select.curselection())!=0:
        result=messagebox.askyesno('Confirmation','You Want to Delete Contact\n Which you selected')
        if result==True:
            del contactlist[Selected()]
            Select_set()
    else:
        messagebox.showerror("Error", 'Please select the Contact')


def VIEW():
    NAME, PHONE, PAN_, ADHAR_ = contactlist[Selected()]
    Name.set(NAME)
    Number.set(PHONE)
    PAN.set(PAN_)
    Adhar.set(ADHAR_)


# exit game window
def EXIT():
    root.destroy()


def Select_set() :
    contactlist.sort()
    select.delete(0,END)
```

```
    for name,phone,pan,Adhar in contactlist :
        # contact_summary = f"{name} | {phone} | {pan} | {adhar}"
         select.insert(END, name)
Select_set()
#Python - define buttons labels and entry widget
Label(root, text = 'Name', font=("Times new roman",15,"bold"), bg =
'SlateGray3').place(x= 30, y=20)
Entry(root, textvariable = Name, width=30).place(x= 200, y=30)
Label(root, text = 'Contact No.', font=("Times new roman",15,"bold"),bg =
'SlateGray3').place(x= 30, y=65)
Entry(root, textvariable = Number, width=30).place(x= 200, y=70)
Label(root, text = 'PAN No.', font=("Times new roman",15,"bold"), bg =
'SlateGray3').place(x= 30, y=105)
Entry(root, textvariable = PAN, width=30).place(x= 200, y=110)
Label(root, text = 'Adhar No.', font=("Times new roman",15,"bold"), bg =
'SlateGray3').place(x= 30, y=140)
Entry(root, textvariable = Adhar, width=30).place(x= 200, y=145)
Button(root,text=" ADD", font='Helvetica 10 bold',bg='#e8c1c7', command =
AddContact, padx=20). place(x= 30, y=300)
Button(root,text="EDIT", font='Helvetica 10 bold',bg='#e8c1c7',command =
UpdateDetail, padx=20).place(x= 30, y=340)
Button(root,text="DELETE", font='Helvetica 10
bold',bg='#e8c1c7',command = Delete_Entry, padx=20).place(x= 30, y=380)
Button(root,text="VIEW", font='Helvetica 10 bold',bg='#e8c1c7', command
= VIEW).place(x= 30, y=420)
```

```
Button(root,text="EXIT", font='Helvetica 10 bold',bg='tomato', command = EXIT).place(x=100, y=500)

Button(root,text="RESET", font='Helvetica 10 bold',bg='#e8c1c7', command = EntryReset).place(x=30, y=450)


Button(root, text="Adhar Website", command=open_link, bg="#add8e6").place(x=300, y=540)

Button(root, text="Pan Website", command=open_link, bg="#add8e6").place(x=300, y=580)


root.mainloop()
```

OUTPUT:



Applications of Contact book project:

A Python contact book can be surprisingly versatile! Here are some practical places and scenarios where it can be used:

1. **Personal Use**: Keep track of friends, family, and service contacts—especially if you want a lightweight, offline alternative to cloud-based apps.
2. **Small Businesses**: Store customer or client contact details without needing a full CRM system. It's great for freelancers or local shops.
3. **Educational Projects**: Perfect for students learning Python, file handling, databases, or GUI development. It's a classic beginner-to-intermediate project.
4. **Command-Line Tools**: Developers and sysadmins can use it to manage team or vendor contacts directly from the terminal.
5. **Internal Tools**: Companies can use it as a quick internal directory for teams, especially when integrated with a database or web interface.
6. **Embedded in Larger Apps**: It can be a module inside a bigger app—like a school management system or event planner.

Conclusion:

The Python contact book project is a practical and educational tool that combines essential programming concepts like data structures, file handling, and user interaction. It helps learners understand how to build real-world applications by implementing features such as adding, updating, deleting, and searching contacts. With options for data persistence and interface design, it lays the groundwork for more advanced software development.