**SILIGURI INSTITUTE OF TECHNOLOGY**
**DEPARTMENT OF ELECTRICAL ENGINEERING**
**SILIGURI -734009**

# Project On:

# Contact Book

**By**

**Arjun Singh**

**Subhankar Roy**

**Sudip Chanda**

**Suvajit Sarkar**

**Saroj Kumar Gouda**

Under the guidance of Ripum Kundu(Trainer)

**Siliguri Institute Of Technology**

**(Maulana Abul Kalam Azad University of Technology (WBUT)**

**FACULTY OF EE DEPARTMENT**

## Certificate of Recommendation

This is to certify that Group 3 has completed his project work titled " Project on: "Contact Book", under the direct supervision and guidance of Ripam Kundu. We are satisfied with their work, which is being presented for the partial fulfillment of the degree of Bachelor of Technology (BTech), West Bengal University of technology (WBUT), Kolkata– 700032.

Signature Of TPO

Date:

(Name of Teacher in charge of Project)

Date:

Signature Of Director

Siliguri Institute of Technology

Date

Arup Das

HOD EE Department (Siliguri Institute of Technology)

**Siliguri Institute Of Technology**

**FACULTY OF EE DEPARTMENT**

## Certificate of Approval

The foregoing project is hereby approved as a creditable study of Bachelor of Technology (BTech) and presented in a manner satisfactory to warrant its acceptance as a pre-requisite to the degree for which it has been submitted. It is understood that by this approval the undersigned do not necessarily endorse or any statement made, opinion expressed or conclusion therein but approve this Minor project only for the purpose for which it is submitted.

Arjun Singh

Subhankar Roy

Shuvajit Sarkar

Sudip Chanda

Saroj Kumar Gauda

Signature of the Members

## Table of Contents

## Introduction

Our task is to implement a smartphone directory that collects contact data from the user until the user prompts the program to. Contact data refers to the contact's name, phone number, date-of-birth, a category that contact belongs to (Friends, Family, Work, Other), e-mail address. The user may enter as much data as he can in the mentioned data labels. If some labels remain void of data, store it as None. A name & the number is mandatory to create contact. Implement the following operations on the directory: Insert, Delete, Search, Display.

## Project Description

To build your contact book application, you need to organize the code into modules and packages and give your project a coherent structure. In this project, we'll use the following directories and files structure:

```
rpcontacts_project/
│
├── rpcontacts/
│   ├── __init__.py
│   ├── views.py
│   ├── database.py
│   ├── main.py
│   └── model.py
│
├── requirements.txt
├── README.md
└── rpcontacts.py
```

Here's a brief summary of the contents of your project directory:

- **rpcontacts project/** is the project's root directory. It'll contain the following files:
  - **requirements.txt** provides the project's requirements list.
  - **README.md** provides general information about the project.
  - **rpcontacts.py** provides the entry-point script to run the application.
- **rpcontacts/** is a subdirectory that provides the application's main package. It provides the following modules:
  - __init__.py
  - views.py
  - database.py
  - main.py
  - model.py

You'll cover each of these files step by step in this tutorial. The name of each file gives an idea of its role in the application. For example, views.py will contain the code to generate the GUI of windows and dialogs, database.py will contain code to work with the database, and main.py will host the application itself. Finally, model.py will implement the model to manage the data in the application's database.

In general, the application will have a main window to display, add, remove, and update contacts. It'll also have a dialog to add new contacts to the database.

## Thinking

We come across lots of people daily. We make acquaintances and friends. We get their contacts to keep in touch later on. Sadly, keeping the received contact details can be hard. One way to do this is to write the contact details down. But this is not secure as the physical book can easily be lost.

This is where the Contact Book project comes in. A contact book is a tool for saving a contact's details, such as name, address, phone number, and email address. With this contact book project, you can build a software tool that people can use to save and find contact details.

With the contact book project idea, users can save their contacts with less risk of losing the saved contact details. It'll always be accessible from their computer, through the command-line.

### Examples of Contact Book Tools

There are Contact Book applications, but it's rare to find command-line Contact Book products, as most are web, mobile, or GUI applications.

**Note:** For an in-depth explanation of how to build a GUI-based contact book, check out Build a Contact Book With Python, PyQt, and SQLite.

Here are some implementations of the Contact Book idea:

- Simple Contacts
- Pobuca Connect

### Technical Details

The main objective of this project is to save contact details. It's important that you set up the commands users can use to enter the contact details. You can use the argparse or click command-line frameworks. They abstract a lot of complex stuff, so you only have to focus on the logic to be run when executing commands.

Some features you should implement include the commands to delete contacts, update contact information, and list saved contacts. You can also allow users to list contacts using different parameters, such as alphabetical order or contact creation date.

Since it's a command-line project, the SQLite database will be fine for saving contacts. SQLite is user-friendly to set up. You may save the contact details in a file, but a file will not offer the benefits you can gain from using SQLite, such as performance and security.

To use the SQLite database in this project, the Python sqlite3 module will be very useful.

**Extra Challenge**

Remember how the database is stored on the user's computer? What if something happens, like the user losing their files? It means they'll also lose the contact details.

You can challenge yourself further and backup the database to an online storage platform. To do this, you can upload the database files to the cloud at certain intervals.

You can also add a command that allows users to backup the database themselves. This way, the user can still have access to the contacts if the database file is lost.

You should note that you may need some form of identification, so the contact book can tell which database file belongs to which user. Implementing a user authentication feature is one way to go about it.

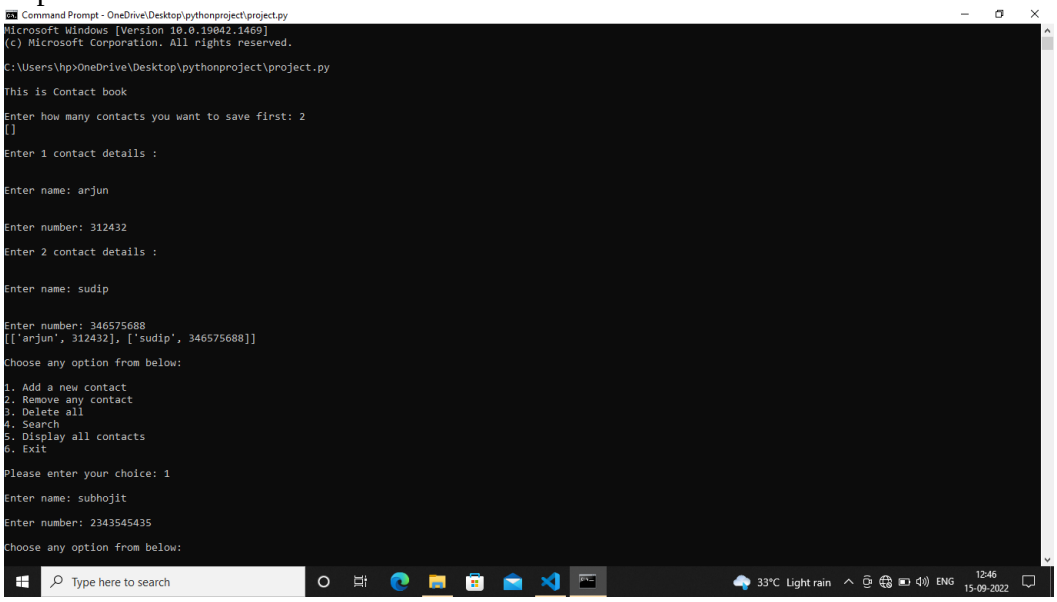
**Snapshot**

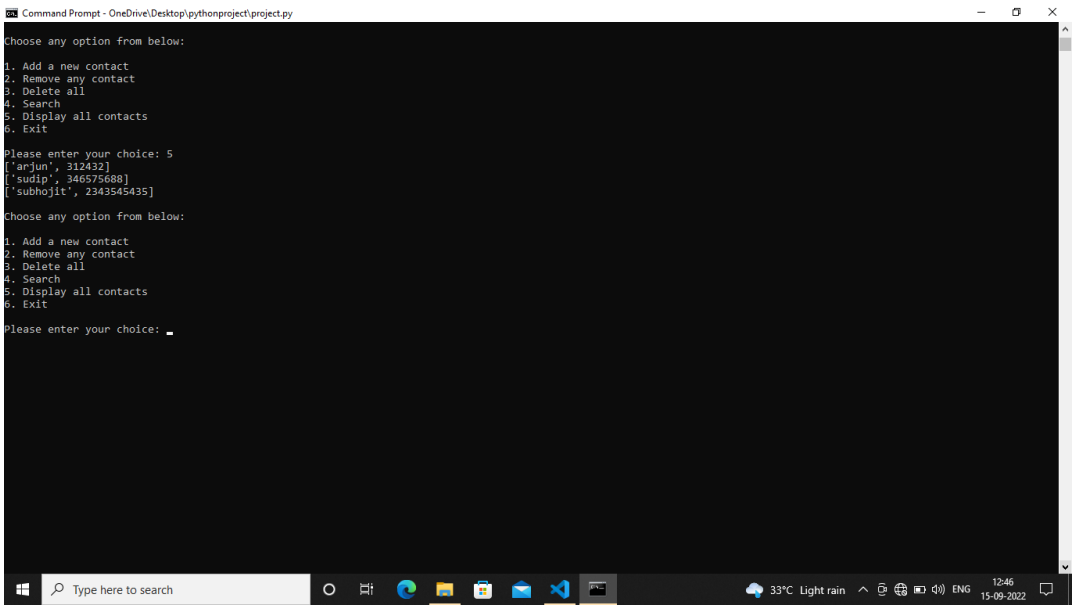
```
import sys

def save():
    rows, columns = int(input("\nEnter how many contacts you want to save first: ")), 2
    contacts = []
    print(contacts)
    for i in range(rows):
        print("\nEnter %d contact details : " % (i+1))
        var1 = []
        for j in range(columns):
            if j == 0:
                var1.append(str(input("\n\nEnter name: ")))
            if j == 1:
                var1.append(int(input("\n\nEnter number: ")))
        contacts.append(var1)
    print(contacts)
    x="".join(str(e) for e in contacts)
    f=open("project.txt","w")
    f.write(x)
```

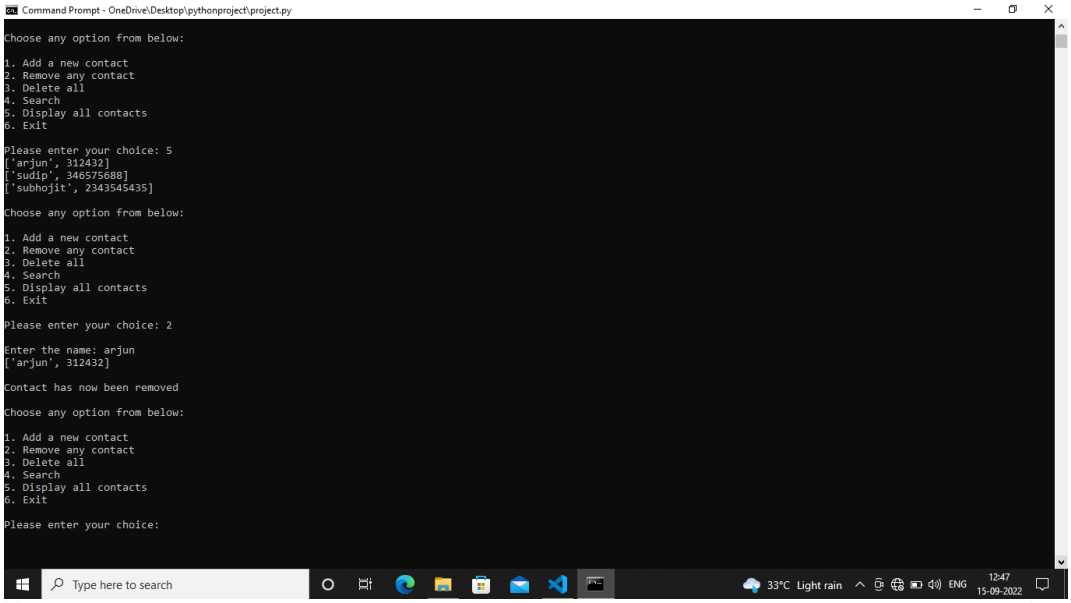
```
    f.close()
    return contacts


def add(s):
    var2 = []
    for i in range(len(s[0])):
        if i == 0:
            var2.append(str(input("\nEnter name: ")))
        if i == 1:
            var2.append(int(input("\nEnter number: ")))
    s.append(var2)
    x=" ".join(str(e) for e in var2)
    f=open("project.txt","a")
    f.write(x)
    f.close()
    return s

def remove(s):
    var = str(input("\nEnter the name: "))
    var4 = 0
    for i in range(len(s)):
        if var == s[i][0]:
            var4 += 1
            print(s.pop(i))
            print("\nContact has now been removed")
            return s
    if var4 == 0:
        print("\nThere's no contact saved by this name")
        return s

def delete(pb):
    return pb.clear()

def find(s):
    var5 = []
    check = -1
    var = str(input("\nEnter the name: "))
    for i in range(len(s)):
        if var == s[i][0]:
            check = i
            var5.append(s[i])
    if check == -1:
        return -1
    else:
        show(var5)
        return check

def show(s):
    if not s:
```

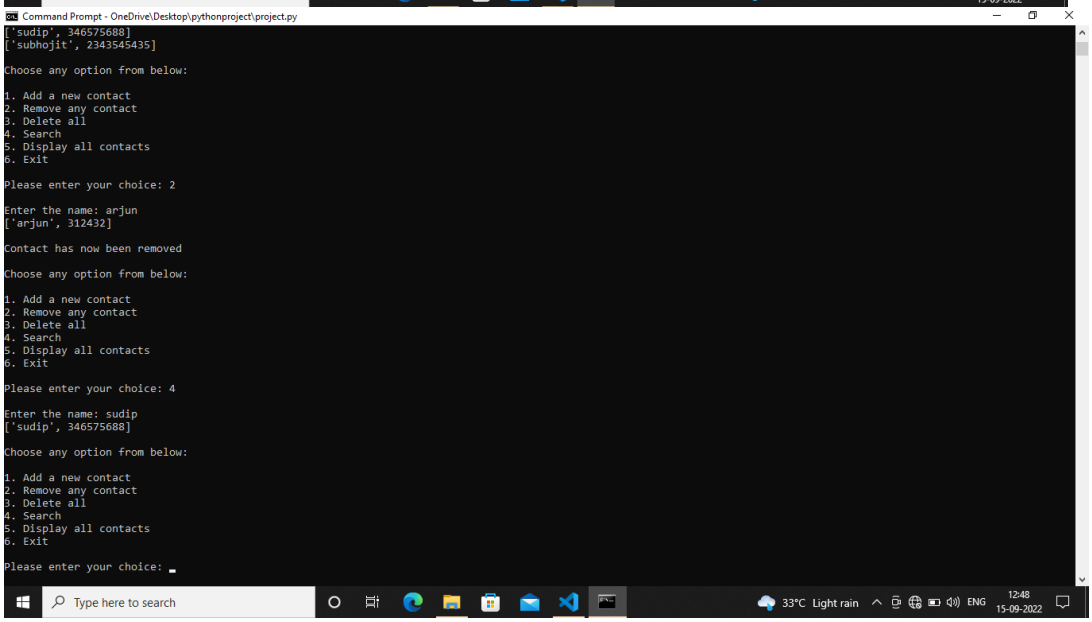
```
        print("\nList is empty: []")
    else:
        for i in range(len(s)):
            print(s[i])

print("\nThis is Contact book")
ch = 1
s= save()
while ch in (1, 2, 3, 4, 5):
    print("\nChoose any option from below:\n")
    print("1. Add a new contact")
    print("2. Remove any contact")
    print("3. Delete all ")
    print("4. Search ")
    print("5. Display all contacts")
    print("6. Exit ")

    ch = int(input("\nPlease enter your choice: "))
    if ch == 1:
        s = add(s)
    elif ch == 2:
        s = remove(s)
    elif ch == 3:
        s = delete(s)
    elif ch == 4:
        d = find(s)
        if d == -1:
            print("\nThe contact does not exist.")
    elif ch == 5:
        show(s)
    else:
        sys.exit()
```

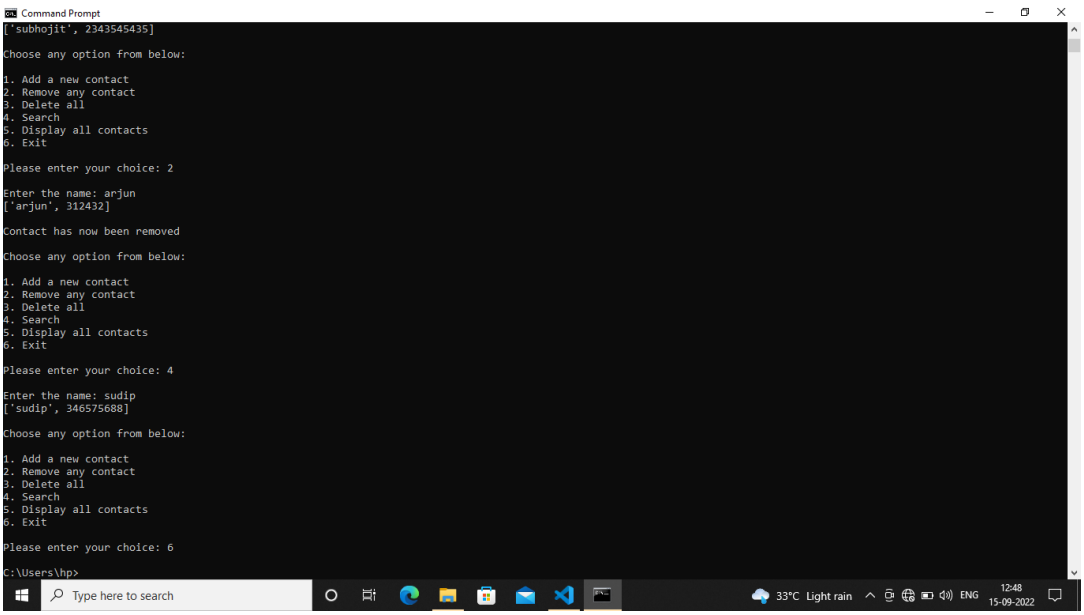
Output:




Command Prompt - OneDrive\Desktop\pythonproject\project.py
Choose any option from below:
1. Add a new contact
2. Remove any contact
3. Delete all
4. Search
5. Display all contacts
6. Exit
Please enter your choice: 5
['arjun', 312432]
['sudip', 346575688]
['subhojit', 2343545435]
Choose any option from below:
1. Add a new contact
2. Remove any contact
3. Delete all
4. Search
5. Display all contacts
6. Exit
Please enter your choice:
Type here to search
33°C Light rain
ENG
12:46
15-09-2022


Command Prompt - OneDrive\Desktop\pythonproject\project.py
Choose any option from below:
1. Add a new contact
2. Remove any contact
3. Delete all
4. Search
5. Display all contacts
6. Exit
Please enter your choice: 5
['arjun', 312432]
['sudip', 346575688]
['subhojit', 2343545435]
Choose any option from below:
1. Add a new contact
2. Remove any contact
3. Delete all
4. Search
5. Display all contacts
6. Exit
Please enter your choice: 2
Enter the name: arjun
['arjun', 312432]
Contact has now been removed
Choose any option from below:
1. Add a new contact
2. Remove any contact
3. Delete all
4. Search
5. Display all contacts
6. Exit
Please enter your choice:
Type here to search
33°C Light rain
ENG
12:47
15-09-2022


Command Prompt - OneDrive\Desktop\pythonproject\project.py
['sudip', 346575688]
['subhojit', 2343545435]
Choose any option from below:
1. Add a new contact
2. Remove any contact
3. Delete all
4. Search
5. Display all contacts
6. Exit
Please enter your choice: 2
Enter the name: arjun
['arjun', 312432]
Contact has now been removed
Choose any option from below:
1. Add a new contact
2. Remove any contact
3. Delete all
4. Search
5. Display all contacts
6. Exit
Please enter your choice: 4
Enter the name: sudip
['sudip', 346575688]
Choose any option from below:
1. Add a new contact
2. Remove any contact
3. Delete all
4. Search
5. Display all contacts
6. Exit
Please enter your choice:
Type here to search
33°C Light rain
ENG
12:48
15-09-2022




## Conclusion

Building a contact book GUI application with Python, PyQt, and SQLite is an excellent exercise for you to expand your skills with these tools and as a developer in general. Coding projects like this allows you to apply the knowledge and skills you already have and also pushes you to research and learn about new topics every time you encounter a new programming problem.

## References

https://www.geeksforgeeks.org/implementing-a-contacts-directory-in-python/

https://realpython.com/python-contact-book/