A

PROJECT REPORT

ON

PHONE BOOK

By

K . Priyanka

(EBEONO62139204)

Batch :- 2021-5735

Under the guidance of

Shalini Kumari

(Technical Trainer)

EduBridge Learning

(School of Coding)

EduBridge Learning

(School of Coding)

EduBridge

CERTIFICATE

This is to certify that this Project Work entitled “PHONE BOOK” is Bonafide work carried out by K. PRIYANKA(EBEONO62139204) in partial fulfilment of the requirements for the Java Full Stack Developer course from Edu bridge during the period 2021 under our guidance and supervision.

Guide

Shalini Kumari

(Technical Trainer)

CONTENTS

ABSTRACT

Phone Book: is primarily meant for keeping the records of the persons. Records consist of their name, office address, home address, email id, designation, phone numbers etc. In the project most of the functions are handled.

Phone Book will provide the basic set of features of adding a new contact, searching, list of all contacts, deleting a contact.

By successfully implementing the Phone Book: a substantial knowledge has been acquired regarding the implementation of the application-based system in order to store the records.

1. INTRODUCTION

It is designed for people to store the information about their contacts. Rather than going through the pages of their diaries and copies to search a person information, a person can simply use this software to view any of his stored contacts. A user can also add or update or delete the contact information according to his need. Big Organizations need to have this software in order to keep the records of thousands of his employees at one place without any paper work.

1.1 Background

The project Complete Phone Book provides a person to add its personal information like First name, Last name, Phone number, Email, etc. to the database. A user can search any other person or can add a new contact onto the software.

1.2 Project Objective

For easy and convenient processing, we need to develop software which works as per our requirements. While preparing this software we got to remember few things i.e., this software must have a friendly environment, in other words it should not be much complicated to handle, it should have options for future modifications in the society.

The major specifications of project are:

- Locate any Contact wanted by the user.
- Reduced written work and problem of storing the diaries as most of the work done by computer.
- Provide greater speed & reduced time consumption.

2. Requirement Analysis

The Phone Book requires computerizing its storage of contacts in order to provide fully service to automate the following activities. The requirements from the proposed software are as follows:

- Contact Information Handling
- Adding a new contact
- Deleting a contact
- Searching a contact

2.1 Problem Definition

The problem here is that storing the contact information on paper is little bulky and is difficult to manage and store these papers. They can be forgotten where they were placed or could be destroyed by the small animals like rat etc. For a personal purpose it could be managed but for organizations where there is a need to keep these information's for later use it will create a problem if such thing happens.

So, why don't we keep it in our computer as a record? You may say that we could have used WORD or NOTEPAD, yes, we could have used but in that case, it would be a little difficult to search a contact or delete one or add a contact after scrolling thousands of pages of word but the software PHONE BOOK provides a total and an easy interface to add, search, delete a contact without going through pages. All that work will be done by the software itself.

2.2 Software and Hardware Requirements

Software Requirements

For development, software's used are:

Operating System	: Windows 10
Platform	: Java
Technology	: Eclipse
Editor	: Microsoft word

Hardware Requirements

For Development, Hardware's used are:

1. Intel Core
2. 8 GB RAM

For Deployment, Hardware's used are :

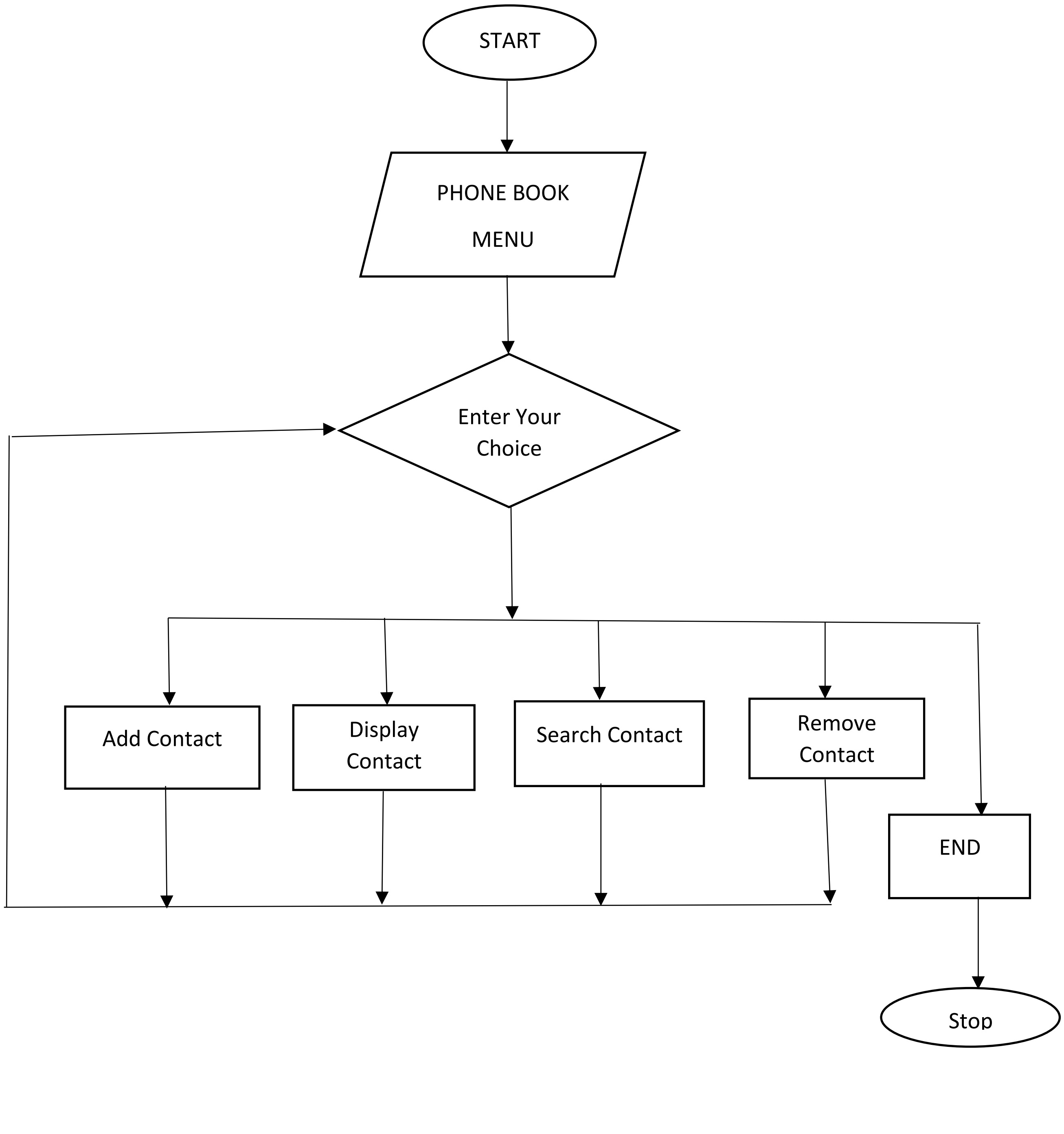
Minimum

1. Intel Core
2. 256MB RAM

Recommended

1. Intel Core
2. 1 GB RAM

3.System Design



4. Implementation

Modules

1.Add Contact

2.Display Contact

3.Search Contact

4.Remove Contact

4.1 Add Contact:

- This feature adds a new phone record to the file. It asks for the phone number, first name, last name and email address of the person whose record is to be created.

4.2 Display All Contacts:

- In phonebook management system, this feature allows user to list all the phone records stored in file. The information displayed here are phone record number, phone number and the person's name.

4.3 Search Contact:

- This function is very simple, and with-it users can search for a phone record by providing the phone number of a particular person whose record has already been added in the file.

4.4 Remove Contact:

- This feature deletes added phone record from the file. The user needs to provide the phone number to be deleted from phonebook. It will ask confirmation message – "Do you want to remove the contact(Y/N)",
- Upon successful deletion, it displays the message – "Contact is successfully deleted!". If the phone number provided is not found in file, the program displays the message – "Contact is not found".

5. Source Code

5.1. Phone Book Code:

```
package manage;
        import java.util.*;
        public class PhoneBook
        {
           private List<contact1> phoneBook=new ArrayList<contact1>();
           public void setPhoneBook(List<contact1>obj)
           {
              phoneBook=obj;
           }
           public List<contact1>getPhoneBook()
           {
              return phoneBook;
           }
           public void addContact(contact1 contactObj)
           {
              phoneBook.add(contactObj);
           }
           public List<contact1> viewAllContacts()
           {
              return phoneBook;
           }
           public contact1 viewContactGivenPhone(long phoneNumber)
           {
              contact1 obj=new contact1();
              for(contact1 obj1:phoneBook)
              {
                 if(obj1.getPhoneNumber()==phoneNumber)
                 {
                    obj=obj1;
                 }
        }
        return obj;
    }
    public boolean removeContact(long phoneNumber)
    {
        boolean f=false;
        for(contact1 obj:phoneBook)
        {
            if(obj.getPhoneNumber()==phoneNumber)
            {
                f=true;
                phoneBook.remove(obj);
                break;
            }
        }
        return f;
    }
}
```

5.2. <u>**Contact1.java**</u>:

```
package manage;
public class contact1 {
            private String firstName;
            private String lastName;
            private long  phoneNumber;
            private String emailId;
            public contact1(){}
            public String getFirstName() {
              return firstName;
            }
            public void setFirstName(String firstName) {
              this.firstName = firstName;
```

```
 }
 public String getLastName() {
   return lastName;
 }
 public void setLastName(String lastName) {
   this.lastName = lastName;
 }
 public long getPhoneNumber() {
   return phoneNumber;
 }
 public void setPhoneNumber(long phoneNumber) {
   this.phoneNumber = phoneNumber;
 }
 public String getEmailId() {
   return emailId;
 }
 public void setEmailId(String emailId) {
   this.emailId = emailId;
 }
 public contact1(String firstName, String lastName, long phoneNumber,
     String emailId) {
   super();
   this.firstName = firstName;
   this.lastName = lastName;
   this.phoneNumber = phoneNumber;
   this.emailId = emailId;
 }
}
```

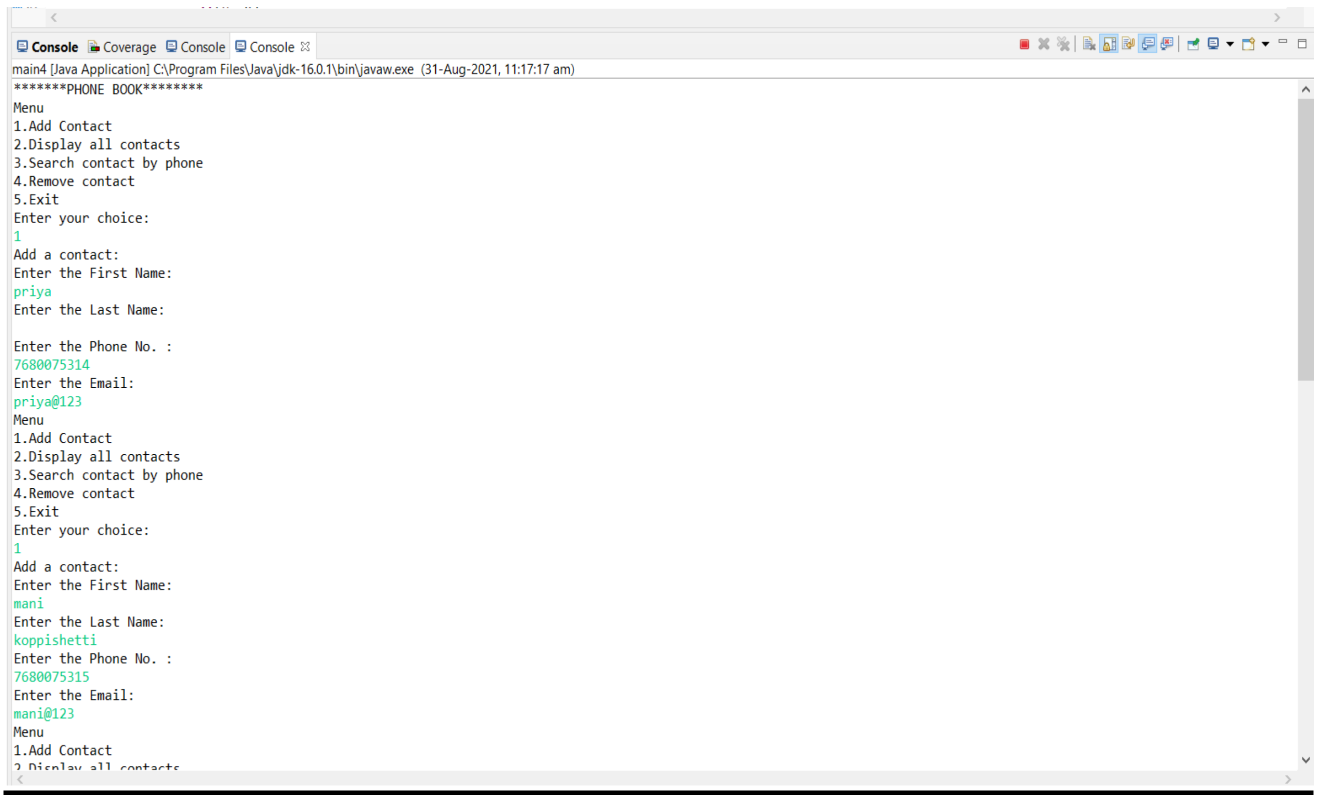
5.3. Main.java:

```
package manage;
import java.util.*;
public class main4 {
            public static void main(String args[])
          {
            Scanner sc=new Scanner(System.in);
            int i=0;
            PhoneBook objmain=new PhoneBook();
            System.out.println("*******PHONE BOOK********");
            while(i==0)
            {
              System.out.println("Menu\n1.Add Contact\n2.Display all contacts\n3.Search
contact by phone\n4.Remove contact\n5.Exit");
              System.out.println("Enter your choice: ");
              int n=Integer.parseInt(sc.nextLine());
              if(n==1)
              {
                contact1 obj=new contact1();
                System.out.println("Add a contact: ");
                System.out.println("Enter the First Name: ");
                obj.setFirstName(sc.nextLine());
                System.out.println("Enter the Last Name: ");
                obj.setLastName(sc.nextLine());
                System.out.println("Enter the Phone No. : ");
                obj.setPhoneNumber(Long.parseLong(sc.nextLine()));
                System.out.println("Enter the Email: ");
                obj.setEmailId(sc.nextLine());
                objmain.addContact(obj);
              }
              if(n==2)
              {
                System.out.println("The contacts in the List are:");
```

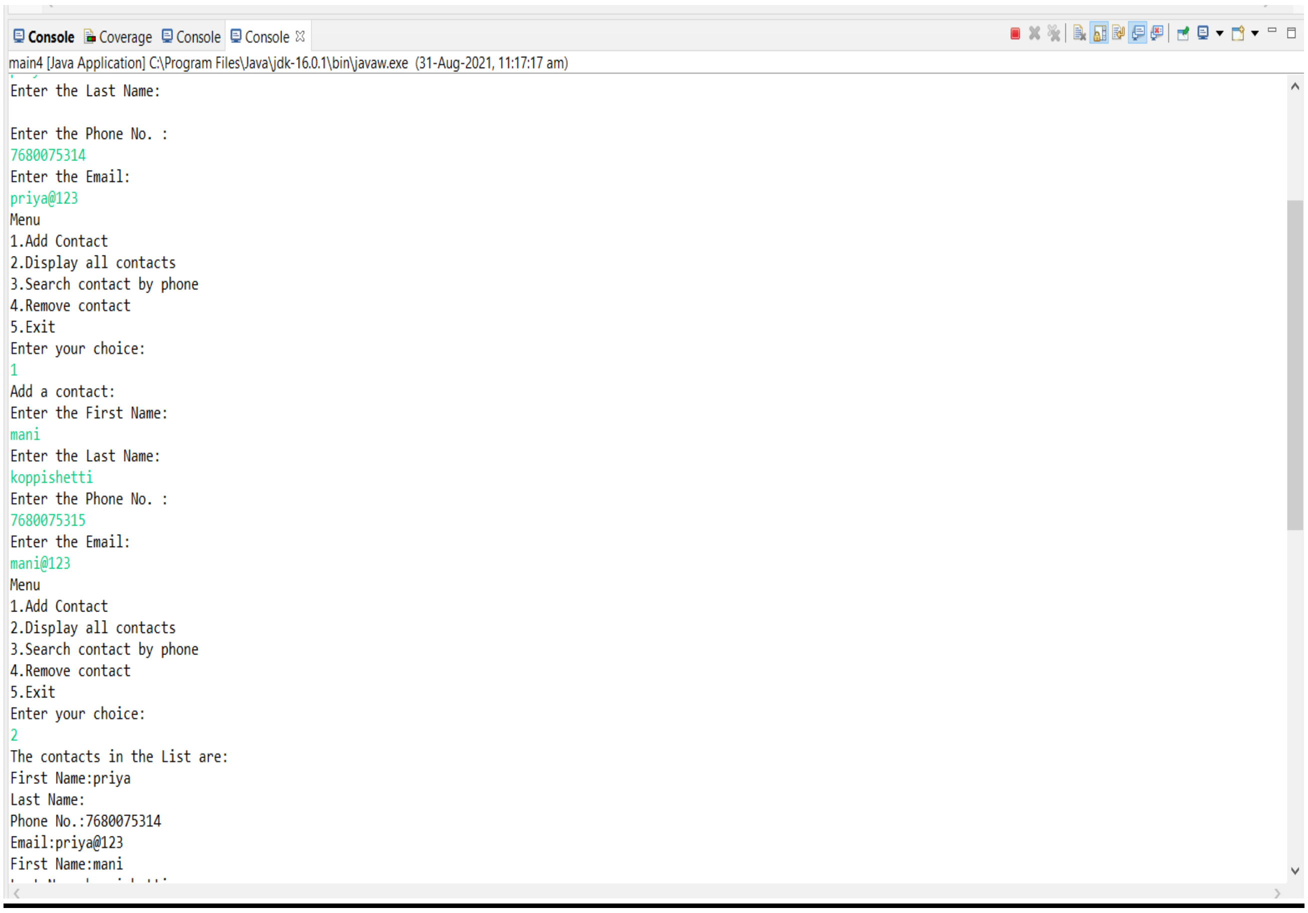
```
    List<contact1>obj=objmain.viewAllContacts();
    for(contact1 temp:obj)
    {
        System.out.println("First Name:"+temp.getFirstName());
        System.out.println("Last Name:"+temp.getLastName());
        System.out.println("Phone No.:"+temp.getPhoneNumber());
        System.out.println("Email:"+temp.getEmailId());
    }
}
if(n==3)
{
    System.out.println("Enter the Phone number to search contact:");
    Long n1=Long.parseLong(sc.nextLine());
    contact1 obj1=objmain.viewContactGivenPhone(n1);
    System.out.println("The contact is:");
    System.out.println("First Name:"+obj1.getFirstName());
    System.out.println("Last Name:"+obj1.getLastName());
    System.out.println("Phone No.:"+obj1.getPhoneNumber());
    System.out.println("Email:"+obj1.getEmailId());
}
if(n==4)
{
    System.out.println("Enter the Phone number to remove:");
    Long n1=Long.parseLong(sc.nextLine());
    System.out.println("Do you want to remove the contact(Y/N):");
    char ch=sc.nextLine().charAt(0);
    if(ch=='Y')
    {
        boolean f1=objmain.removeContact(n1);
        if(f1)
        System.out.println("The contact is successfully deleted");

        else
        System.out.println("Contact is not found");
```

```
                }
                if(ch=='N')
                {
                    System.out.println("ok");
                }

            }
            if(n==5)
            {
                System.exit(0);
            }
        }
    }
}
```

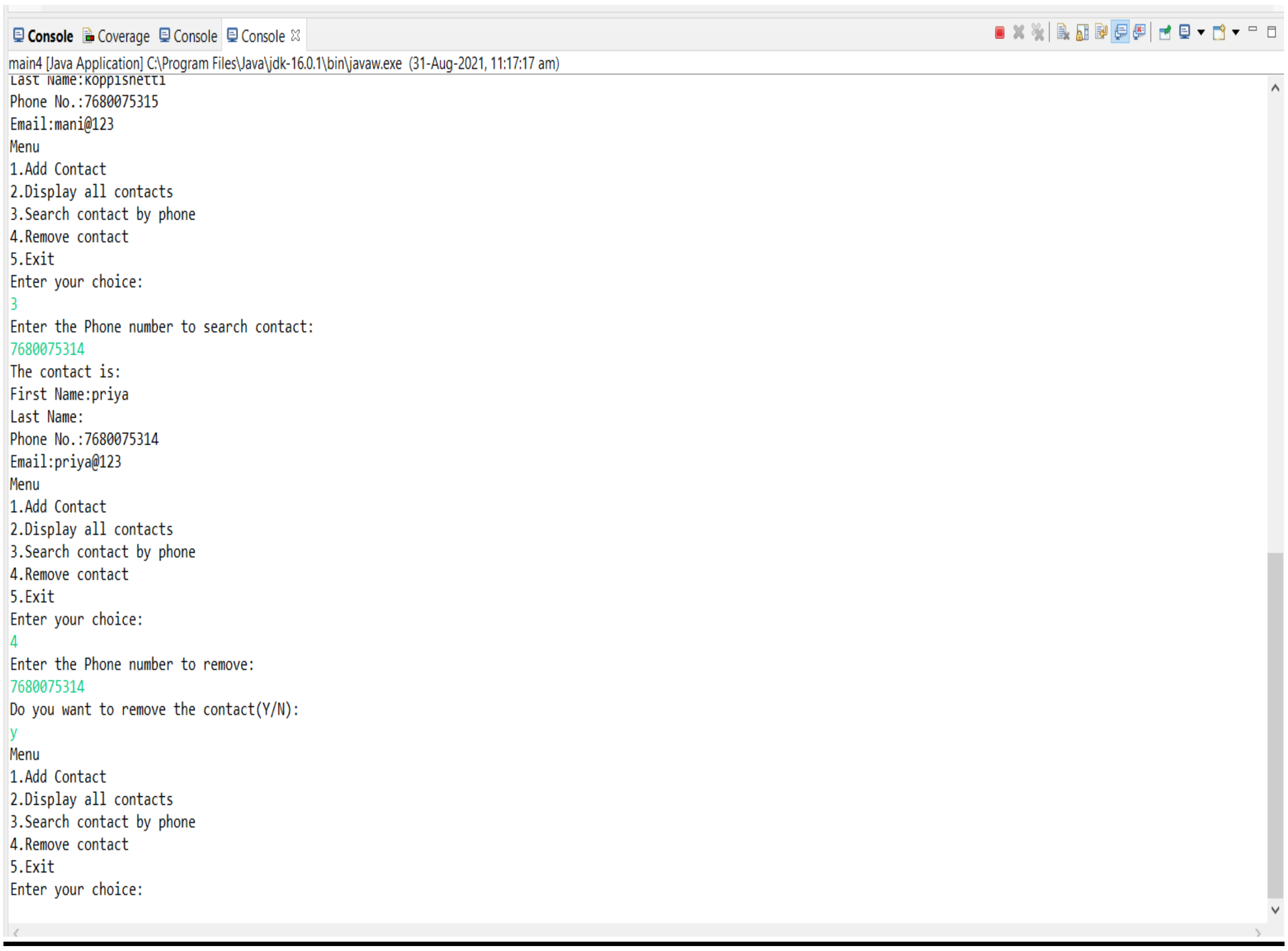
6.Output



```
main4 [Java Application] C:\Program Files\Java\jdk-16.0.1\bin\javaw.exe  (31-Aug-2021, 11:17:17 am)
*******PHONE BOOK********
Menu
1.Add Contact
2.Display all contacts
3.Search contact by phone
4.Remove contact
5.Exit
Enter your choice:
1
Add a contact:
Enter the First Name:
priya
Enter the Last Name:

Enter the Phone No. :
7680075314
Enter the Email:
priya@123
Menu
1.Add Contact
2.Display all contacts
3.Search contact by phone
4.Remove contact
5.Exit
Enter your choice:
1
Add a contact:
Enter the First Name:
mani
Enter the Last Name:
koppishetti
Enter the Phone No. :
7680075315
Enter the Email:
mani@123
Menu
1.Add Contact
2.Display all contacts
```



```
main4 [Java Application] C:\Program Files\Java\jdk-16.0.1\bin\javaw.exe  (31-Aug-2021, 11:17:17 am)
Enter the Last Name:

Enter the Phone No. :
7680075314
Enter the Email:
priya@123
Menu
1.Add Contact
2.Display all contacts
3.Search contact by phone
4.Remove contact
5.Exit
Enter your choice:
1
Add a contact:
Enter the First Name:
mani
Enter the Last Name:
koppishetti
Enter the Phone No. :
7680075315
Enter the Email:
mani@123
Menu
1.Add Contact
2.Display all contacts
3.Search contact by phone
4.Remove contact
5.Exit
Enter your choice:
2
The contacts in the List are:
First Name:priya
Last Name:
Phone No.:7680075314
Email:priya@123
First Name:mani
```


Console | Coverage | Console | Console

main4 [Java Application] C:\Program Files\Java\jdk-16.0.1\bin\javaw.exe (31-Aug-2021, 11:17:17 am)

```
Last Name:koppisnetti
Phone No.:7680075315
Email:mani@123
Menu
1.Add Contact
2.Display all contacts
3.Search contact by phone
4.Remove contact
5.Exit
Enter your choice:
3
Enter the Phone number to search contact:
7680075314
The contact is:
First Name:priya
Last Name:
Phone No.:7680075314
Email:priya@123
Menu
1.Add Contact
2.Display all contacts
3.Search contact by phone
4.Remove contact
5.Exit
Enter your choice:
4
Enter the Phone number to remove:
7680075314
Do you want to remove the contact(Y/N):
y
Menu
1.Add Contact
2.Display all contacts
3.Search contact by phone
4.Remove contact
5.Exit
Enter your choice:
```

7. <u>**Advantages & Disadvantages**</u>

- It is simple for the user to store complete information (e.g., e-mail ID, address) about his or her contact.
- It's simple for the user to simply check his appropriate contact number by entering the contact number.
- It is difficult to store contacts with two or more contact numbers.

8. Future Scope

- In future, we can enhance this project by adding java frameworks and SQL server too.
- We can add Login password, Finger prints etc., for providing more security for user.
- And one more major change which can be done in the project is that to add the snaps of the person of which the address is entered.

9. Conclusion

This Project has been completed successfully . It is user friendly and has required options , which can be utilized by the user to perform the desired operations.

The goals that are achieved by the software are:

- Optimum utilization of resources.
- Efficient management of records.
- Simplification of the operations.
- Less processing time and getting required information.
- User friendly.
- Portable and flexible for further enhancement.