VISVESVARAYA TECHNOLOGY UNIVERSITY
BELAGAVI-590018

B.L.D.E.A's V.P. Dr.P.G.Halakatti College of Engineering and Technology

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

A MINI PROJECT REPORT ON :

"PHONEBOOK MANAGEMENT SYSTEM"

UNDER THE GUIDANCE OF:

Prof. Pradeep Deshpande

Prof. Supriya Patil

SUBMITTED BY:

Sajeed Malagi (2BL20IS031)

Azeem Bekinalkar(2BL20IS018)

VISVESVARAYA TECHNOLOGY UNIVERSITY BELAGAVI - 590018

B.L.D.E.A's V.P. Dr.P.G.Halakatti College of Engineering and Technology

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

This is to certify that the mini project titled " **PHONEBOOK MANAGENT SYSTEM** " under **File Structure laboratory[18CSL58]** work carried out by SAJEED MALAGI(2BL20IS031), AZEEM BEKINALKAR(2BL20IS018) submitted in partial fulfillment for the VI Semester of Bachelor of Engineering degree of the Visvesvaraya Technological University, Belgaum during the year 2021-2022. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report. The mini project report has been approved as it satisfies the academic requirements in respect of mini project work prescribed for File Structure laboratory with Mini Project prescribed in Information Science & Engineering of VI semester.

Guide	H.O.D	Principal
Prof. Pradeep Deshpande. **Prof. Supriya Patil.**	**Dr Prakash Unki**	**DR V G SANGAM**
SIGNATURE	**SIGNATURE**	**SIGNATURE**

Examiner 1:

Examiner 2:

Table of Contents

VISVESVARAYA TECHNOLOGY UNIVERSITY 1
ACKNOWLEDGEMENT 4
ABSTRACT 5
INTRODUCTION 6
OBJECTIVES 7
FEATURES 7
HARDWARE/SOFTWARE REQUIREMENTS 8
DATA FLOW DIAGRAM 9
SYSTEM DESIGN 10
Logical Design 10
TECHNOLOGY USED 11
APPLICATIONS FOR PHONEBOOK MANAGEMENT SYSTEM 12
Code Implementation 13
SNAPSHOTS 14

ACKNOWLEDGEMENT

We would like to express deep sense of gratitude to our beloved principal **Dr. V. G. Sangam** providing all facilities in the college. We would like to thank who heartedly to our Head of Department **Dr. Prakash Unki** for providing facilities and fostering congenial academic environment in the college. We feel deeply indebted to our esteemed project guides **Prof. Pradeep Deshpande & Prof. Ssupriya Patil** for their impactful help right from the conception and visualization to the very presentation of the mini project. They have been our guiding light. We would take this opportunity to thank all the faculty members and supporting staff for helping us in this endeavor.

Project Associates

1.Sajeed Malagi

2.Azeem Bekinalkar

ABSTRACT

The Phonebook Management System is a software application designed to efficiently manage and organize contact information. It employs file structures to store and retrieve contact details, providing users with an intuitive interface to add, update, search, and delete entries in the phonebook.The system utilizes various file structures to optimize data storage and retrieval. One such structure is the B-tree, which offers balanced search and insertion operations. It ensures efficient access to contact records, particularly when the phonebook contains a large number of entries. Additionally, the B-tree enables fast range searches, allowing users to retrieve contacts based on specific criteria, such as name, phone number, or address.

To enhance search performance further, the system incorporates indexes. Indexing improves the speed of retrieval operations by creating a separate structure that maps key values to their corresponding locations in the phonebook file. This enables rapid access to specific records, minimizing the time required to locate desired contacts.The Phonebook Management System employs file compression techniques to optimize storage utilization. By compressing the data, the system reduces the required storage space, resulting in more efficient disk usage. This not only improves performance but also enables the management of larger phonebooks within limited storage capacities.The system provides a user-friendly interface that enables users to perform various operations easily. Users can add new contacts by entering relevant details, including name, phone number, email address, and other pertinent information. The system validates the input and ensures data integrity before storing it in the appropriate file structure.

Furthermore, the system supports efficient update and deletion operations. Users can modify contact information by searching for a specific entry and updating the relevant fields. Similarly, the system allows users to remove contacts from the phonebook, ensuring data consistency by updating the file structure accordingly. In summary, the Phonebook Management System employing file structures offers an efficient and user-friendly solution for organizing and managing contact information. By utilizing B-trees, indexes, and compression techniques, the system optimizes storage utilization and enhances search and retrieval performance. It provides a reliable platform for individuals and organizations to maintain and access their phonebook efficiently.

INTRODUCTION

You can learn the fundamentals of functions, file handling, and data structure by working on the Phonebook Management System mini-project.This system will show you how to add, list, change or edit, search for, and remove data from/to a file. The functions that make up this project's menu include adding new records, listing them, editing and updating them, looking for saved contacts, and deleting phonebook entries.When you add a record to your phonebook, you are asked for personal information like name, gender, phone number, and address. Then, these records can be modified, added to, searched for, and deleted. In this project, I've used a lot of functions. These functions are easy to understand because their names only describe their corresponding operations. This system is developed using the general need required by the user while using the phone directory book. In order to keep updated the phone book directory, the admin will have the authority to add and delete as well as modify the existing records within the phone book directory. The users of the directory will only have the authority to search any particular record and listing details of all available records. To provide the search result within short interval of time optimized search algorithm code have been used that will able to provide the results within seconds. For searching operation, users will able to get any particular record using their contact name or phone number but the only condition is that,customers record must be available within the file system. If no such record will be available, proper error message will be displayed as per user input provided to the system.

OBJECTIVES

The objectives of a Phone Book Management System in the C programming language include the following:

1. Storing and managing a large number of contact records in an efficient organized manner.
2. Providing a user-friendly interface for adding, updating, viewing, and deleting records.
3. Allowing users to easily search for a specific contact by name or phone number.
4. Implementing error handling and input validation to ensure the system is robust and secure.
5. Making it easy to maintain and update the system.
6. Optimizing the performance by using efficient data structures and algorithms.
7. Providing the feature of searching for a contact in the fastest possible time. Overall, the goal of a Phone Book Management System in C programming language is to provide an efficient way of storing, managing, and searching large amounts of contact information, while also being user-friendly and secure.

FEATURES

1. Adding new contacts: Allows admins to add new contact records to the system, including information such as name, gender, phone number, and address.
2. Updating existing contacts: Allows admins to edit or update existing contact records.
3. Viewing specific contacts: Allows users and administrators to view the details of a specific contact, including all the stored information.
4. Listing all contacts: Allows users and administrators to view a list of all contacts stored in the system.
5. Deleting contacts: Allows admins to delete existing contact records.
6. Searching for contacts: Allows users and administrators to search for a specific contact by name or phone number.
7. Error handling and input validation: These ensures the system is robust and secure by handling errors and validating user input.
8. Search optimization: It implements efficient data structures and algorithms for optimized and faster search results. This is a very basic and brief description of the Phone Book Management System, depending on the requirement and the actual implementation, there could be many more features or complexity involved.

HARDWARE/SOFTWARE REQUIREMENTS

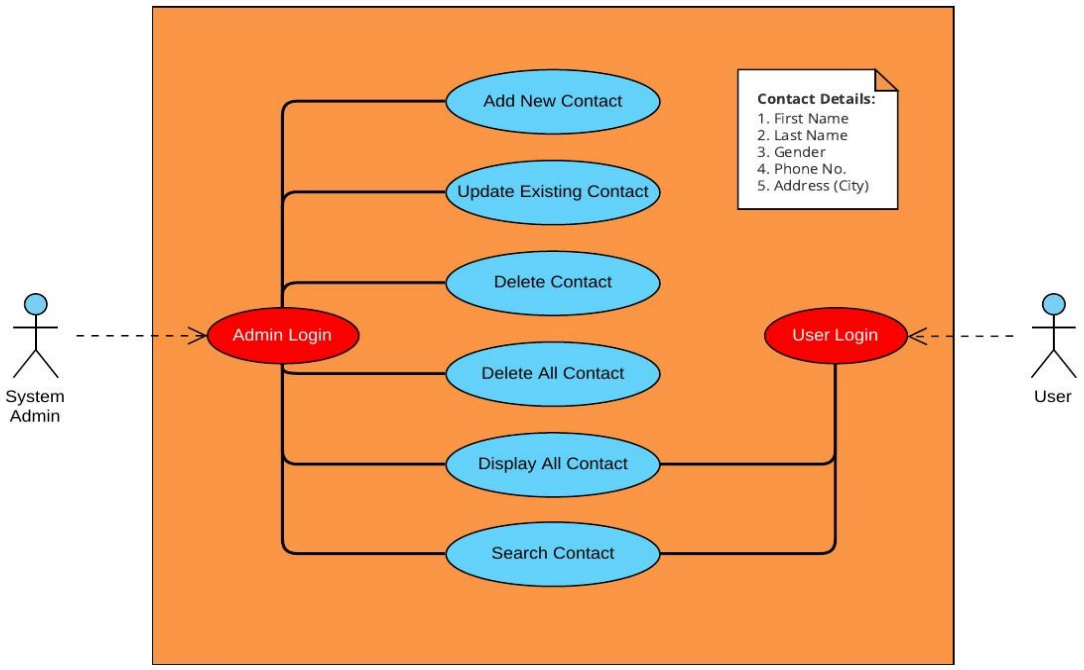
Hardware Requirements:

- A computer with a suitable amount of storage space to store the phonebook data.
- A sufficient amount of RAM to run the system efficiently.
- A monitor with a mini mum resolution of 1024 x 768 pixels. A keyboard and mouse.
- A printer, if the system needs to print out the contact records.

Software Requirements:

- An operating system such as Windows, Linux, or MacOS.
- A C compiler such as GCC or Microsoft Visual C++.
- A code editor or IDE for reviewing the code.

DATA FLOW DIAGRAM:



- **Main Menu:** The system presents a menu of options for the user to choose from, such as adding a contact, searching for a contact, updating a contact, deleting a contact, or exiting the system.
- **Add Contact:** If the user selects the "Add Contact" option, the system prompts the user to enter the contact details, such as name, phone number, email address, and other relevant information. Once the user provides the details, the system validates the input and stores the contact in the phonebook.
- **Search Contact**: If the user selects the "Search Contact" option, the system prompts the user to enter search criteria, such as name, phone number, or any other relevant information
- **Update Contact**: If the user selects the "Update Contact" option, the system prompts the user to enter the contact details to be updated. The user can search for the desired contact using search criteria (e.g., name or phone number.
- **Delete Contact:** If the user selects the "Delete Contact" option, the system prompts the user to enter search criteria to locate the contact to be deleted. The system then searches the phonebook for the specified criteria and displays the matching contacts.

SYSTEM DESIGN:

The system design for a phonebook management system typically involves breaking down the functionality into different modules or components. These components may include:

- **User Interface:** Handles user interactions, displays menus, and prompts for user input.
- **File Management:** Manages reading from and writing to files that store the phonebook data.
- **Data Structures:** Defines the data structures used to represent contacts and the phonebook itself. This may include structures or classes to hold contact information.
- **Search and Manipulation:** Contains functions to search for contacts, add new contacts, update existing contacts, and delete contacts from the phonebook.

Logical Design:

- The logical design focuses on the organization and structure of data within the phonebook management system. Here are some key elements:
- **Phonebook Data Structure:** You can use an array, linked list, or other appropriate data structure to store the contacts in the phonebook. Each contact entry may consist of fields like name, phone number, email address, etc.
- **File Format:** Determine the file format for storing the phonebook data. You can choose a simple text-based format where each line represents a contact with its respective fields separated by delimiters. Alternatively, you can use a structured format like CSV (Comma-Separated Values) or JSON (JavaScript Object Notation) for easier data manipulation.
- **Operations and Functions:** Define functions for performing various operations on the phonebook, such as adding a contact, searching for a contact, updating contact details, and deleting a contact.
- **Error Handling:** Implement error handling mechanisms to handle invalid user inputs, file read/write errors, and other exceptional situations that may arise during the execution of the system.

TECHNOLOGY USED:

- Structs: Structs are used to define the structure of a contact entry in the phonebook. Each struct may include fields like name, phone number, email address, and other relevant information.
- File I/O: C provides file I/O functions like fopen(), fclose(), fread(), and fwrite() that allow reading from and writing to files. These functions are used to store the phonebook data in files and retrieve it when needed.
- Text File: A simple text file can be used to store the phonebook data. Each contact entry can be represented as a line in the file, with the fields separated by delimiters like commas or tabs.
- CSV (Comma-Separated Values): CSV file format is commonly used for storing structured data. Each contact entry is represented as a line, and the fields are separated by commas. This format simplifies parsing and manipulation of data using standard file I/O functions.
- Indexing: To improve search performance, you can implement indexing techniques. For example, you can create an index file that maps key values (such as names or phone numbers) to the corresponding file positions of the contact entries. This allows for faster access to specific records.
- B-trees: B-trees are balanced tree structures that can be used for efficient storage and retrieval of data. They are commonly employed in file systems and databases. You can utilize B-trees to organize the phonebook data, allowing for quick search and insertion operations.
- Compression Techniques: To optimize storage utilization, you can apply compression techniques to the phonebook data. Various algorithms, such as Huffman coding or Lempel-Ziv-Welch (LZW) compression, can be used to reduce the size of the stored data, resulting in efficient disk usage.
- These concepts and file structures form the foundation for designing and implementing a phonebook management system in C.

APPLICATIONS FOR PHONEBOOK MANAGEMENT SYSTEM:

- Personal Phonebook: Individuals can use the system to maintain their personal contacts, including family, friends, colleagues, and acquaintances. It allows them to store contact details, update information as needed, and quickly search for specific contacts when required.
- Business Contacts: Professionals and businesses can utilize the phonebook management system to store and manage their business contacts. This includes maintaining a database of clients, suppliers, partners, and other relevant contacts. The system helps in organizing contact information, tracking communication history, and facilitating efficient business interactions.
- Customer Relationship Management (CRM): The system can serve as a simplified CRM tool for small businesses. It allows businesses to maintain a centralized repository of customer contact information, track customer interactions, and manage communication effectively. It assists in providing personalized customer service and streamlining sales and marketing efforts.
- Organizational Phonebook: Large organizations can employ the system as an internal phonebook to manage employee contact information. It enables employees to quickly find and contact colleagues within the organization, fostering collaboration and efficient communication.
- Emergency Services: Phonebook management systems can be utilized by emergency services, such as police departments, fire departments, and hospitals. These systems help store important contact details, such as emergency responders, hospitals, and other essential services. In critical situations, the system enables quick access to vital contact information, aiding in the prompt response and resolution of emergencies.
- Service Providers: Service-oriented businesses, such as plumbers, electricians, or technicians, can use the system to manage customer contacts and appointments. It allows service providers to maintain customer information, schedule appointments, and provide timely service by quickly accessing customer details.
- Community Directories: The system can be implemented as a community directory, providing a centralized platform for residents to store and access contact information of their neighbors, local businesses, community organizations, and emergency services.

Code Implementation:

```
typedef struct RECORD{
    char firstName[MAX_LENGTH];
    char lastName[MAX_LENGTH];
    char gender;
    char findNum[MAX_LENGTH];
    char cityName[MAX_LENGTH];
} DETAILS;

void addNewContact(){
    clearBuffer();
    system("title Add New Contact");
    int flag = 0;
    printf("----------------------------- \n");
    printf("   >>> Add New Contact <<<     \n");
    printf("----------------------------- \n\n");
    pF = fopen("ContactList.txt", "ab+");
    // checks if the txt file exists or not
    if (pF != NULL){
        printf("Enter the first name: ");
        gets(contact.firstName);
        printf("Enter the last name: ");
        gets(contact.lastName);
        printf("Enter the city name: ");
        gets(contact.cityName);
        printf("Enter the gender [M/F]: ");
        scanf(" %c", &contact.gender);
        fflush(stdin);
        printf("Enter the phone number [+91]: ");
        gets(contact.findNum);
        printf("\n");
        removeSpaces(contact.findNum);
        fprintf(pF, "%s %s %c %s %s\n", contact.firstName, contact.lastName,
contact.gender, contact.findNum, contact.cityName);
         printf("------------------------------------------- \n");
         printf("Success: Contact details added in the record.  \n");
          printf("------------------------------------------- \n");
       }else  flag = 1;
```

SNAPSHOTS:

Fig:1 Introduction page

Fig :2 Login Screen

Fig:3 Admin_login Screen

Fig:5 Admin Panel

Fig:6 User panel

Fig:7 Add new Contact

```
----------------------------
   >>> Update Contact <<<
----------------------------

Enter the phone number to update [+91]: 6360915569

               |============================================================|
               |ID|    Name            | Gender | Phone Number | City Name     |
               |============================================================|
               | 1| maaz rakkasagi     | m |    6360915569     | bijapur       |

------------------------------------------------
> Contact Found in records. Enter the new details.
------------------------------------------------
Enter the first name: maaz
Enter the last name: rakkasagi
Enter the city name: bijapur

Type `CONFIRM` to update this details: CONFIRM

Processing . . .

----------------------------
> Success: contact updated.
----------------------------
Press any key to continue . . .
```

Fig:8 Update Contact

```
                          ---------------------------
                             >>> Contacts List <<<
                          ---------------------------

              |============================================================|
              |ID|    Name            | Gender | Phone Number | City Name     |
              |============================================================|
              | 1| azeem bekinalkar   | m |    9448440468     | bijapur       |
              | 2| sajeed malagi      | m |    6360915568     | bijapur       |
              | 3| maaz rakkasagi     | m |    6360915569     | bijapur       |
              |============================================================|

Press any key to continue . . .
```

Fig:9 Display All Contacts

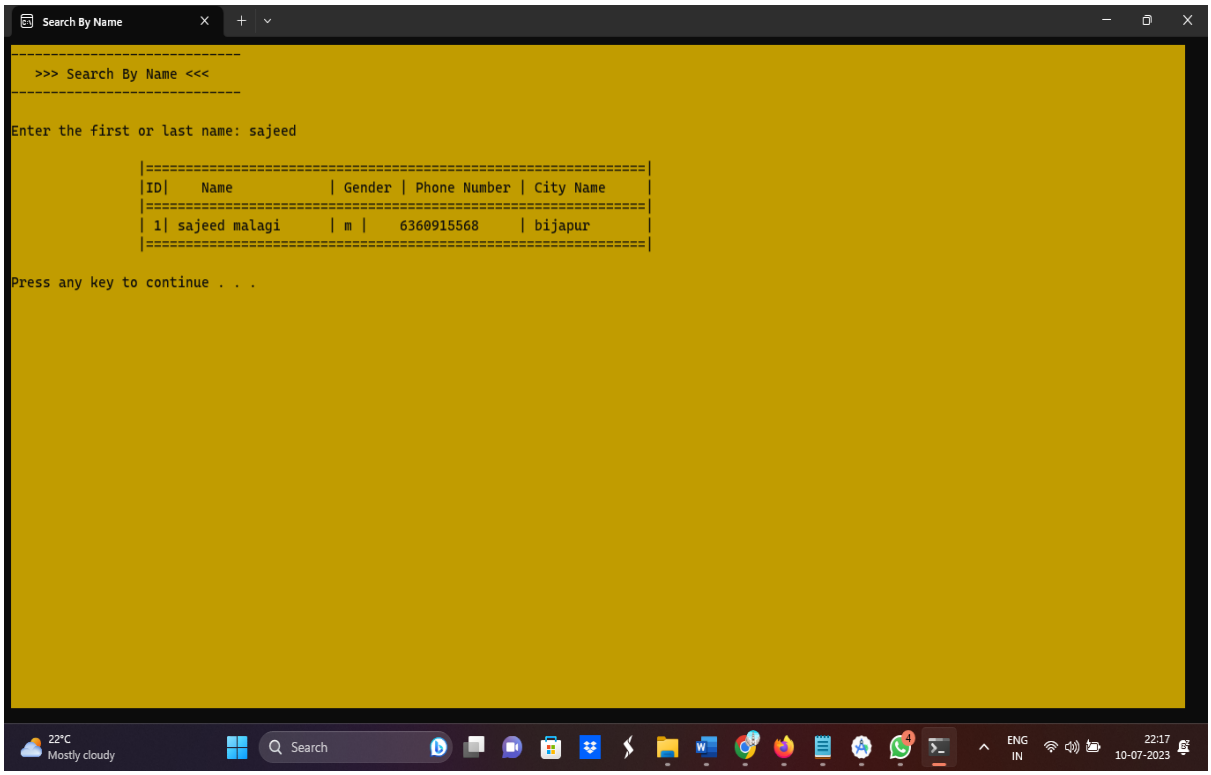


Fig:10 Search_By_name

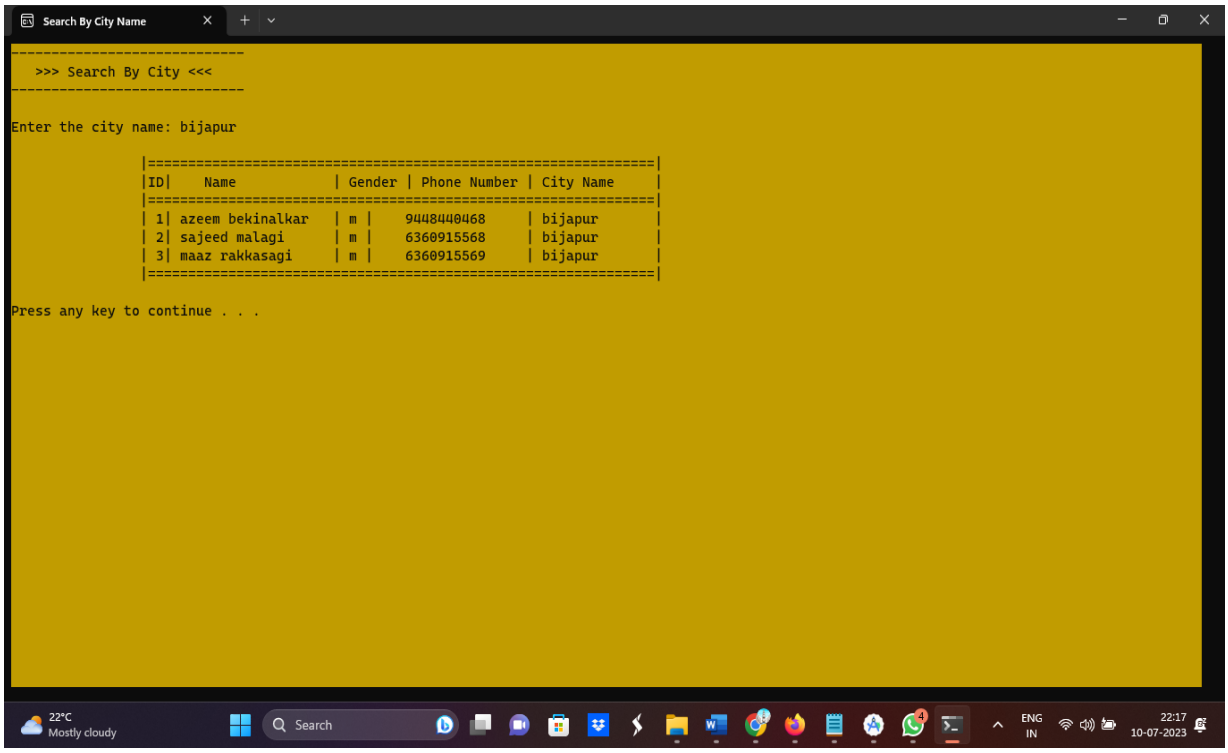


Fig:11 Search_by_City

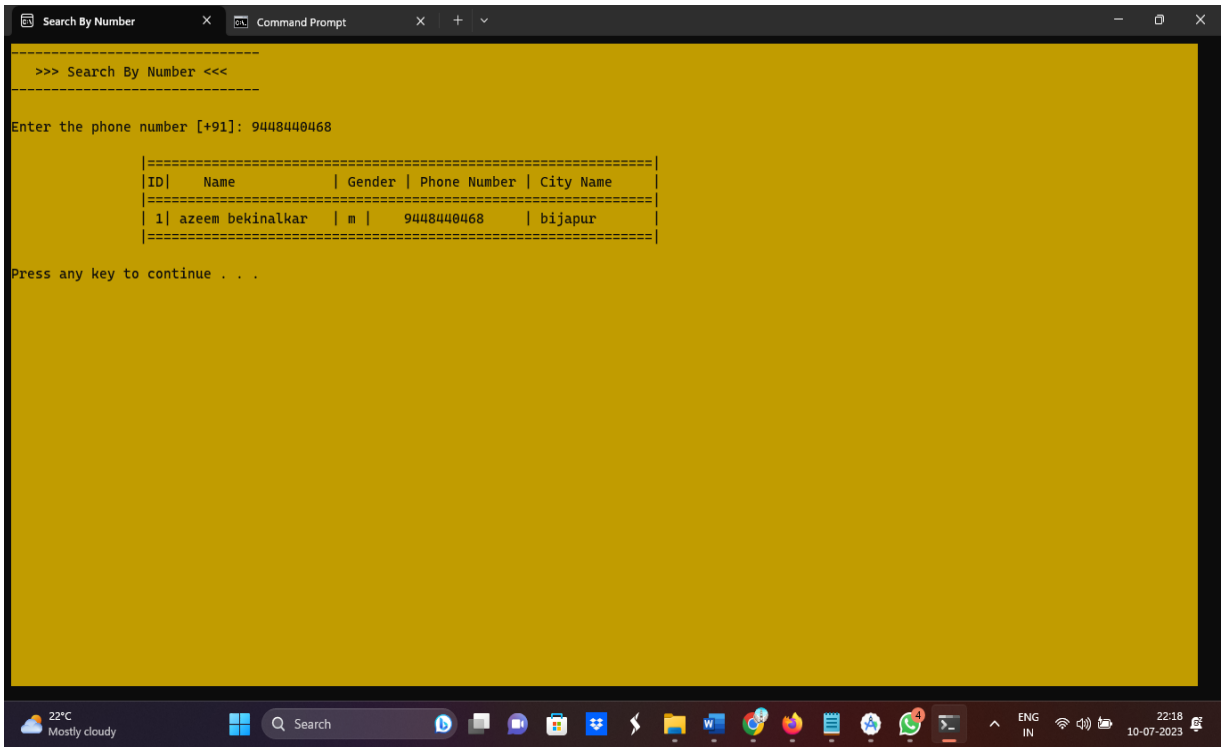


Fig:12 Search_by_number

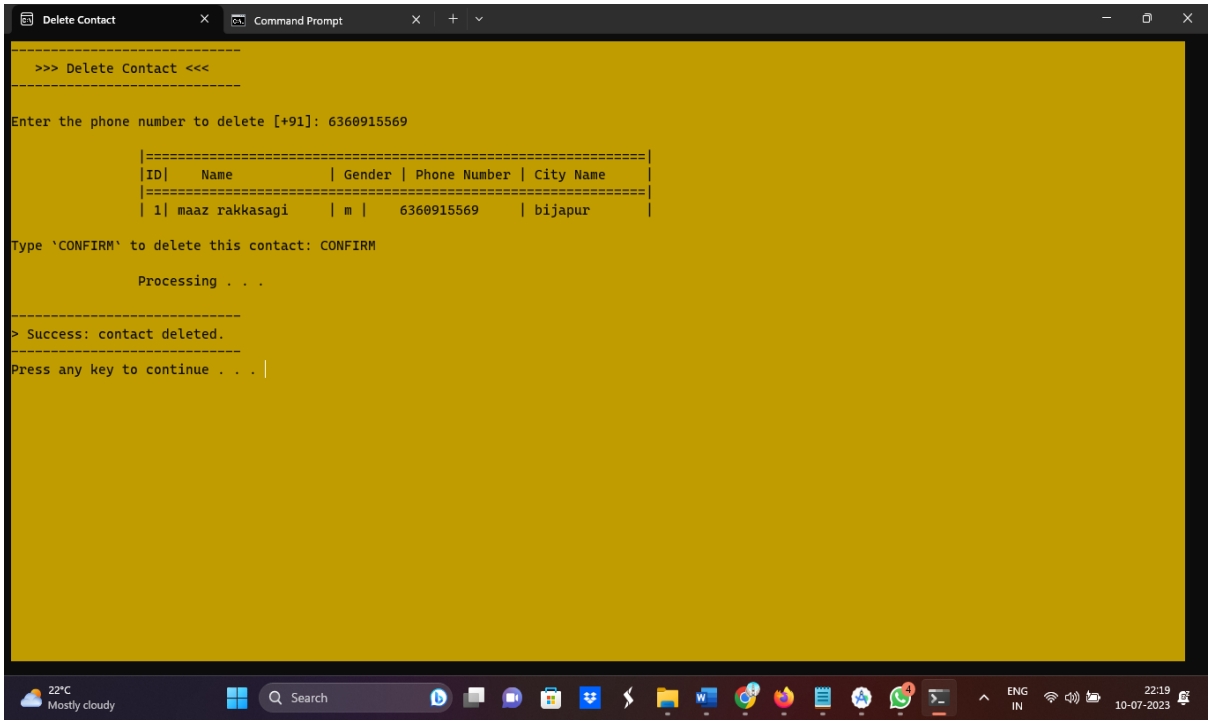


Fig:13 Delete Contact

CONCLUSION:

In conclusion, a Phone Book Management System in the C programming language is a software application that allows users to store, organize and manage large numbers of contact records efficiently. The system can be designed to have functions such as adding, updating, viewing, listing and deleting contacts. Overall, a Phone Book Management System in C programming language can provide an efficient and user-friendly solution for managing and searching large amounts of contact information.

References:

Michael J. Folk, Bill Zoellick, Greg Riccardi: File Structures-An Object Oriented Approach with C++, 3rd Edition