PROJECT REPORT

On

“ ***Contact Management in ‘C' Language*** ”

Submitted By

Shivani Dhakate

(**CS23109**)

Guided By:-

Mrs.Roshni Talmale

DEPARTMENT OF FIRST YEAR ENGINEERING

S. B. JAIN INSTITUTE OF TECHNOLOGY MANAGEMENT AND RESEARCH, NAGPUR.

2023-2024

CONTENTS

Introduction (Project Details):

In today's interconnected world, efficient management of contacts is crucial for individuals and organizations alike. Whether it's maintaining a directory of business contacts, managing personal connections, or organizing customer information, a reliable Contact Management System (CMS) streamlines communication and enhances productivity. In this project, we embark on the development of a Contact Management System using the C programming language.

The Contact Management System will serve as a robust platform for storing, retrieving, updating, and deleting contact information. It will offer functionalities such as adding new contacts, searching for specific contacts based on various criteria, and modifying existing contact details. Additionally, the system will prioritize data security and integrity, ensuring that sensitive information remains protected.

Key Objectives:

1. **Efficient Data Organization:** The CMS will employ data structures and algorithms to efficiently organize contact information, enabling quick retrieval and manipulation of data.

2. **User-Friendly Interface**: A user-friendly interface will be developed to facilitate easy interaction with the system. Users will be able to perform operations intuitively, enhancing usability.

3. **Search and Filter Functionality:** The system will incorporate robust search and filter functionalities, allowing users to locate specific contacts based on criteria such as name, organization, or contact details.

4. **Data Security:** Implementation of secure data handling mechanisms will be a priority to safeguard sensitive contact information against unauthorized access or manipulation.

5. **Scalability:** The CMS will be designed with scalability in mind, allowing it to accommodate a growing database of contacts without compromising performance.

6. **Error Handling:** Comprehensive error handling mechanisms will be implemented to detect and handle errors effectively, ensuring the stability and reliability of the system.

7.**Documentation and Testing:** Thorough documentation will accompany the system, providing insights into its architecture, functionalities, and usage. Rigorous testing procedures will be employed to identify and rectify any bugs or inconsistencies

Major Library and Functions:

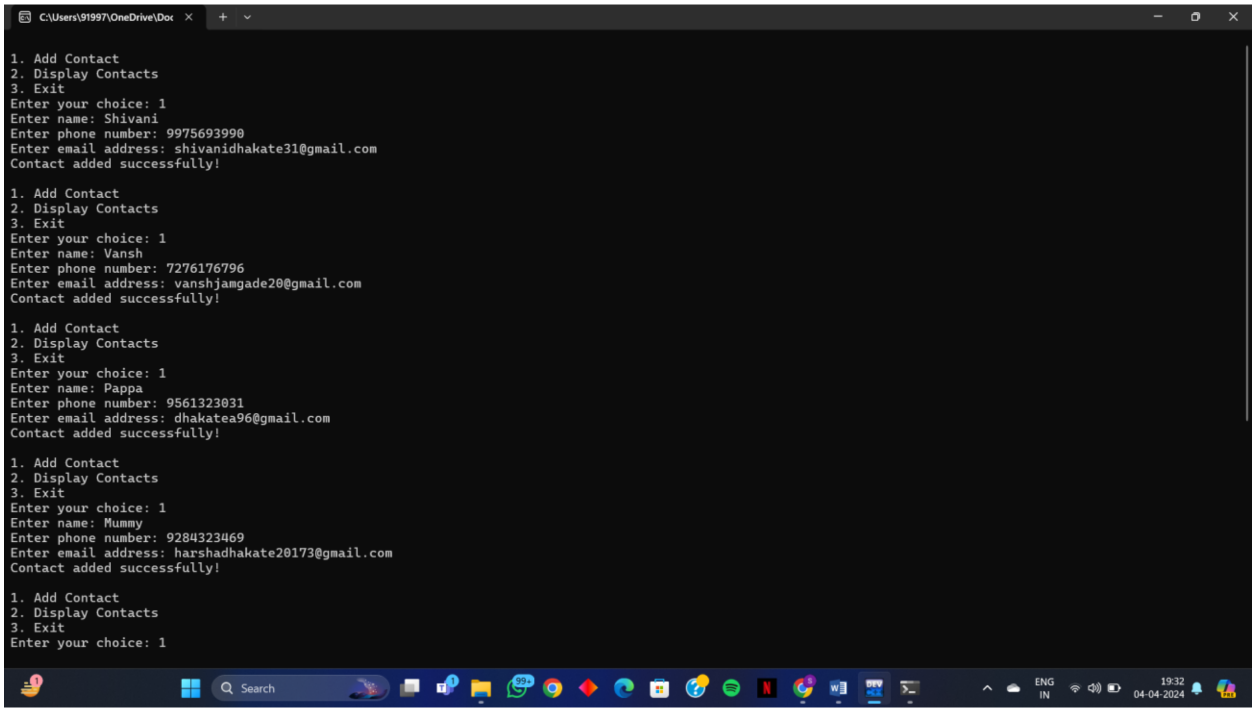
In C, there isn't a built-in library specifically for contact management like in some higher-level languages. However, you can use standard libraries and functions to manage contacts. Here are some major libraries and functions commonly used for this purpose:

1. stdio.h: This library provides functions for standard input and output, which are essential for interacting with the user. Functions like printf and scanf are used for this purpose.
2. string.h: This library provides functions for string manipulation, which can be useful for managing contact information. Functions like strcpy, strcat, and strcmp are commonly used.
3. stdlib.h: This library provides functions for memory allocation and other general utilities. Functions like malloc and free can be used for dynamic memory management if needed.
4. File I/O functions: To save contacts to a file or load contacts from a file, you can use functions like fopen, fclose, fwrite, and fread from stdio.h.
5. Data structures: While not a library per se, C programmers often use data structures like arrays, structs, and linked lists to organize and manage contact information efficiently.

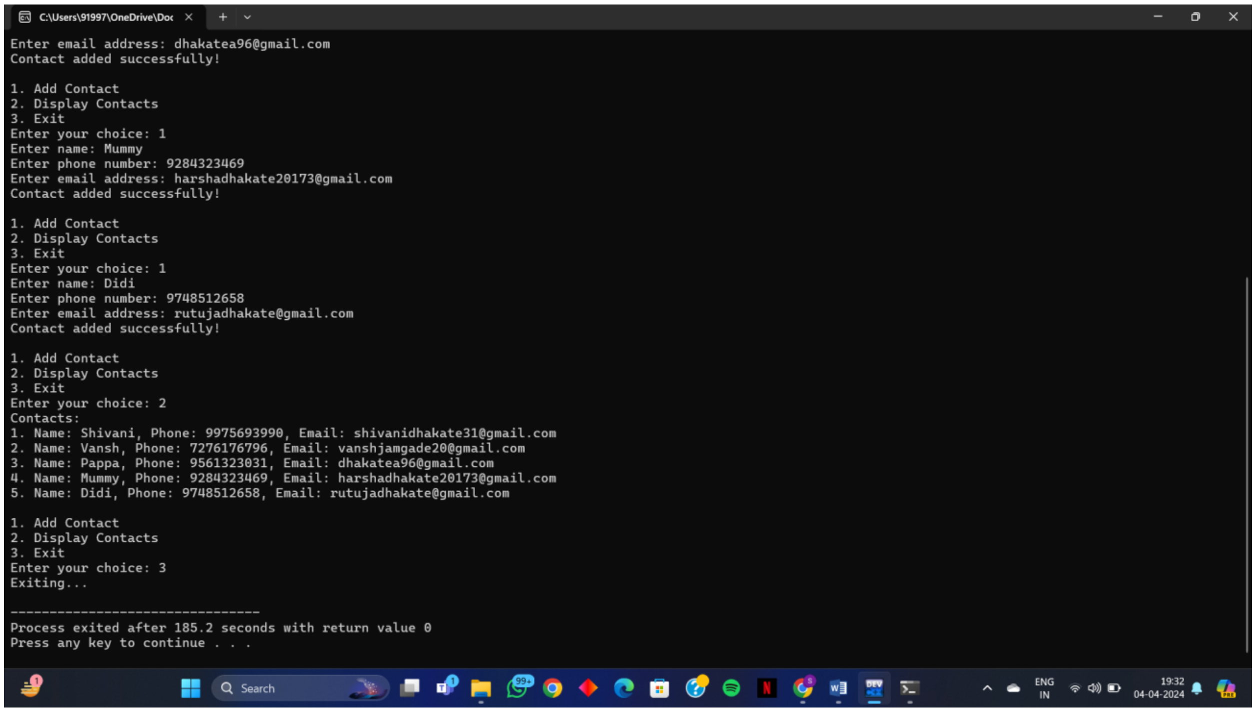
Source Code (Program):

```
#include <stdio.h>
#include <string.h>

#define MAX_CONTACTS 100

struct Contact {
    char name[50];
    char phone[20];
    char email[50];
};

struct Contact contacts[MAX_CONTACTS];
int numContacts = 0;

void addContact() {
    if (numContacts < MAX_CONTACTS) {
        struct Contact newContact;
        printf("Enter name: ");
        scanf("%s", newContact.name);
        printf("Enter phone number: ");
        scanf("%s", newContact.phone);
        printf("Enter email address: ");
        scanf("%s", newContact.email);

        contacts[numContacts] = newContact;
        numContacts++;
        printf("Contact added successfully!\n");
    } else {
        printf("Contact list is full!\n");
    }
}

void displayContacts() {
    printf("Contacts:\n");
    for (int i = 0; i < numContacts; i++) {
        printf("%d. Name: %s, Phone: %s, Email: %s\n", i+1, contacts[i].name, contacts[i].phone,
contacts[i].email);
    }
}

int main() {
    int choice;
    do {
        printf("\n1. Add Contact\n2. Display Contacts\n3. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                addContact();
                break;
            case 2:
                displayContacts();
                break;
            case 3:
                printf("Exiting...\n");
                break;
            default:
                printf("Invalid choice!\n");
        }
    } while (choice != 3);

    return 0;
}
```

Result & Discussion (Output):



```
1. Add Contact
2. Display Contacts
3. Exit
Enter your choice: 1
Enter name: Shivani
Enter phone number: 9975693990
Enter email address: shivanidhakate31@gmail.com
Contact added successfully!

1. Add Contact
2. Display Contacts
3. Exit
Enter your choice: 1
Enter name: Vansh
Enter phone number: 7276176796
Enter email address: vanshjamgade20@gmail.com
Contact added successfully!

1. Add Contact
2. Display Contacts
3. Exit
Enter your choice: 1
Enter name: Pappa
Enter phone number: 9561323031
Enter email address: dhakatea96@gmail.com
Contact added successfully!

1. Add Contact
2. Display Contacts
3. Exit
Enter your choice: 1
Enter name: Mummy
Enter phone number: 9284323469
Enter email address: harshadhakate20173@gmail.com
Contact added successfully!

1. Add Contact
2. Display Contacts
3. Exit
Enter your choice: 1
```



```
Enter email address: dhakatea96@gmail.com
Contact added successfully!

1. Add Contact
2. Display Contacts
3. Exit
Enter your choice: 1
Enter name: Mummy
Enter phone number: 9284323469
Enter email address: harshadhakate20173@gmail.com
Contact added successfully!

1. Add Contact
2. Display Contacts
3. Exit
Enter your choice: 1
Enter name: Didi
Enter phone number: 9748512658
Enter email address: rutujadhakate@gmail.com
Contact added successfully!

1. Add Contact
2. Display Contacts
3. Exit
Enter your choice: 2
Contacts:
1. Name: Shivani, Phone: 9975693990, Email: shivanidhakate31@gmail.com
2. Name: Vansh, Phone: 7276176796, Email: vanshjamgade20@gmail.com
3. Name: Pappa, Phone: 9561323031, Email: dhakatea96@gmail.com
4. Name: Mummy, Phone: 9284323469, Email: harshadhakate20173@gmail.com
5. Name: Didi, Phone: 9748512658, Email: rutujadhakate@gmail.com

1. Add Contact
2. Display Contacts
3. Exit
Enter your choice: 3
Exiting...

--------------------------------
Process exited after 185.2 seconds with return value 0
Press any key to continue . . .
```

Conclusion:

In conclusion, contact management in C can be implemented using standard libraries and functions such as `stdio.h` for input/output, `string.h` for string manipulation, and data structures like arrays or structs to store contact information. While C does not have built-in libraries specifically for contact management, you can create a simple contact management system using these standard libraries and functions.

References:

1. www.wikipedia.org/
2. www.Google.com
3. www.beginnersbook.com
4. https://youtube.com