CONTACT MANAGEMENT SYSTEM

Mini Project in C

Name : Athul George

Roll No : 21

Course Name : Programming in C

Date : 16 – 07 – 2024

INTRODUCTION

Project Overview:

The purpose of this project is developing a system for managing your contacts. The system is designed to enable users to keep a record of their contacts by storing name, phone number, email and category of a contact.

Problem Statement:

In the current digital age, individuals and businesses alike face the challenge of efficiently managing an ever-growing list of contacts that are essential for communication and networking. The lack of a centralized, accessible, and organized system leads to inefficiencies, missed opportunities, and decreased productivity. There is a need for a robust contact management system that can store, categorize, and retrieve contact information seamlessly across various platforms, ensuring that users can maintain and leverage their network effectively

Objective:

Develop a Contact Management System (CMS) that allows users to store, retrieve, update, and delete contact information efficiently. The CMS should provide a user-friendly interface for managing contacts, ensuring data integrity and security.

System Requirements

Minimum Requirements for C Programming Code to Run:

Hardware Requirement:

- A computer with at least 4GB RAM
- 500MB of free disk space

Software Requirement:

- **Operating System:** Windows/Linux/MacOS
- **Compiler:** GCC or any C compiler
- **IDE: Code:** Blocks, Dev-C++, or any C IDE

Design and Development

PROGRAM LOGIC:

The system's functions include processing modules for adding and saving new contacts, display all saved contacts, search for particular contacts, delete one or all contacts and edit an existing contact.

1. Data Structures:

- **Contact :** Store all the contacts by storing their name, phone number, email and category.

2. Global Variables:

- **MAX_CONTACTS :** To store the maximum number of contacts that can be saved.
- **MAX_LINE :** To store the maximum size of one contact.

3. Functions :

- **addContacts() :** To add a new contact to the file.
- **displayContacts() :** To display all saved contacts.
- **searchName() :** To search and display contacts using their names.
- **searchCategory() :** To search and display contacts using their category.
- **editContact() :** To edit an existing contact.
- **deleteContact() :** To delete a saved contact.
- **deleteAllContacts() :** To delete all saved contacts.

Pseudocode

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

struct Contact{
    char name[50];
    char phone[15];
    char email[50];
    char category[50];
};

#define MAX_CONTACTS 1000
#define MAX_LINE 500

void addContact(){

    struct Contact contact;
    printf("Enter name: ");
    fgets(contact.name,sizeof(contact.name),stdin);
    contact.name[strlen(contact.name)-1] = '\0';

    FILE *fp = fopen("contacts.txt","r");
    if (fp != NULL) {
        char line[MAX_LINE];
        while (fgets(line,sizeof(line),fp)) {
            char *token = strtok(line,",");
            if (token != NULL && strcmp(token,contact.name) == 0){
                printf("Contact already exists.\n");
                fclose(fp);
                return;
            }
        }
        fclose(fp);
    }


    printf("Enter phone: ");
    fgets(contact.phone,sizeof(contact.phone),stdin);
    contact.phone[strlen(contact.phone)-1] = '\0';

    printf("Enter email: ");
    fgets(contact.email,sizeof(contact.email),stdin);
    contact.email[strlen(contact.email)-1] = '\0';

    printf("Enter category: ");
```

```
    fgets(contact.category,sizeof(contact.category),stdin);
    contact.category[strlen(contact.category)-1] = '\0';

    fp = fopen("contacts.txt","a");
    if(fp == NULL){
        printf("Error opening file!\n");
        return;
    }
    fprintf(fp,"%s,%s,%s,%s,\n",contact.name,contact.phone,contact.e
mail,contact.category);
    fclose(fp);

    printf("Contact added successfully!\n");
}

void displayContacts(){

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

    FILE *fp = fopen("contacts.txt","r");
    if (fp == NULL){
        printf("Error opening file!!!");
        return;
    }

    char line[MAX_LINE];
    while(fgets(line,sizeof(line),fp)) {
        char *token = strtok(line,",");
        if(token != NULL) {
            strcpy(contacts[numContacts].name,token);
            token = strtok(NULL,",");
            if(token != NULL) {
                strcpy(contacts[numContacts].phone,token);
                token = strtok(NULL,",");
                if (token != NULL) {
                    strcpy(contacts[numContacts].email,token);
                    token = strtok(NULL,",");
                    if (token != NULL) {
                    strcpy(contacts[numContacts].category,token);
                    }
                }
            }
        }
        numContacts++;
    }
    fclose(fp);
```

```
    if(numContacts == 0){
        printf("No contacts found.\n");
    }
    else{
        printf("\nAll Contacts:\n\n");
        for(int i=0; i < numContacts; i++){
            printf("Name: %s\n",contacts[i].name);
            printf("Phone: %s\n",contacts[i].phone);
            printf("Email: %s\n",contacts[i].email);
            printf("Category: %s\n\n",contacts[i].category);
        }
    }
}

void searchName(){

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

    char name[50];
    printf("Enter name of contact to search: ");
    fgets(name,sizeof(name),stdin);
    name[strlen(name)-1] = '\0';

    FILE *fp = fopen("contacts.txt","r");
    if(fp == NULL){
        printf("Error opening file!!!");
        return;
    }

    char line[MAX_LINE];
    while(fgets(line,sizeof(line),fp)){
        char *token = strtok(line,",");
        if(token != NULL){
            strcpy(contacts[numContacts].name,token);
            token = strtok(NULL,",");
            if(token != NULL){
                strcpy(contacts[numContacts].phone,token);
                token = strtok(NULL,",");
                if(token != NULL){
                    strcpy(contacts[numContacts].email,token);
                    token = strtok(NULL,",");
                    if(token != NULL){
                        strcpy(contacts[numContacts].category,token)
;
                        numContacts++;
                    }
                }
```

```
            }
        }
    }
    fclose(fp);

    for(int i=0; i < numContacts; i++) {
        if(strcmp(contacts[i].name,name)==0){
            printf("\nName: %s\n",contacts[i].name);
            printf("Phone: %s\n",contacts[i].phone);
            printf("Email: %s\n",contacts[i].email);
            printf("Category: %s\n\n",contacts[i].category);
            found++;
        }
    }

    if(found==0)
        printf("No contacts found!");
    else
        printf("%d contact(s) found.",found);

}

void searchCategory(){

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

    char category[50];
    printf("Enter category of contact to search: ");
    fgets(category,sizeof(category),stdin);
    category[strlen(category)-1] = '\0';

    FILE *fp = fopen("contacts.txt","r");
    if(fp == NULL){
        printf("Error opening file!!!");
        return;
    }

    char line[MAX_LINE];
    while(fgets(line,sizeof(line),fp)){
        char *token = strtok(line, ",");
        if(token != NULL){
            strcpy(contacts[numContacts].name,token);
            token = strtok(NULL,",");
            if(token != NULL){
                strcpy(contacts[numContacts].phone,token);
                token = strtok(NULL,",");
                if(token != NULL){
```

```
                    strcpy(contacts[numContacts].email,token);
                    token = strtok(NULL,",");
                    if(token != NULL){
                        strcpy(contacts[numContacts].category,token)
;
                        numContacts++;
                    }
                }
            }
        }
    }
    fclose(fp);

    for(int i=0; i < numContacts; i++){
        if(strcmp(contacts[i].category,category)==0){
            printf("\nName: %s\n",contacts[i].name);
            printf("Phone: %s\n",contacts[i].phone);
            printf("Email: %s\n",contacts[i].email);
            printf("Category: %s\n\n",contacts[i].category);
            found++;
        }
    }

    if(found==0)
        printf("No contacts found!");
    else
        printf("%d contact(s) found.",found);

}

void editContact(){

    displayContacts();

    char name[50];
    printf("Enter name of contact to edit: ");
    fgets(name,sizeof(name),stdin);
    name[strlen(name)-1] = '\0';

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

    FILE *fp = fopen("contacts.txt","r");
    if(fp == NULL){
        printf("Error opening file!!!\n");
        return;
    }
```

```
    char line[MAX_LINE];
    while(fgets(line,sizeof(line),fp)){
        char *token = strtok(line,",");
        if(token != NULL && strcmp(token,name) == 0){
            found = 1;
            strcpy(contacts[numContacts].name,token);
            token = strtok(NULL,",");
            if(token != NULL){
                strcpy(contacts[numContacts].phone,token);
                printf("\nCurrent phone:
%s\n",contacts[numContacts].phone);
                printf("Enter new phone: ");
                fgets(contacts[numContacts].phone,sizeof(contacts[nu
mContacts].phone),stdin);
                contacts[numContacts].phone[strlen(contacts[numConta
cts].phone)-1] = '\0';

                token = strtok(NULL,",");
                if(token != NULL){
                    strcpy(contacts[numContacts].email,token);
                    printf("\nCurrent email: %s",
contacts[numContacts].email);
                    printf("\nEnter new email: ");
                    fgets(contacts[numContacts].email,sizeof(contact
s[numContacts].email),stdin);
                    contacts[numContacts].email[strlen(contacts[numC
ontacts].email)-1] = '\0';

                    token = strtok(NULL,",");
                    if(token != NULL){
                        strcpy(contacts[numContacts].category,token)
;
                        printf("\nCurrent category:
%s",contacts[numContacts].category);
                        printf("Enter new category: ");
                        fgets(contacts[numContacts].category,sizeof(
contacts[numContacts].category),stdin);
                        contacts[numContacts].category[strlen(contac
ts[numContacts].category)-1] = '\0';
                        numContacts++;
                    }
                }
            }
        }
        else{
            strcpy(contacts[numContacts].name,token);
            token = strtok(NULL,",");
            if(token != NULL){
```

```
                strcpy(contacts[numContacts].phone,token);
                token = strtok(NULL,",");
                if(token != NULL){
                    strcpy(contacts[numContacts].email,token);
                    token = strtok(NULL,",");
                    if(token != NULL){
                        strcpy(contacts[numContacts].category,token)
;
                        numContacts++;
                    }
                }
            }
        }
    }
    fclose(fp);

    if(found==0){
        printf("Contact not found.\n");
        return;
    }

    fp = fopen("contacts.txt","w");
    if(fp == NULL){
        printf("Error opening file!\n");
        return;
    }

    for(int i=0; i < numContacts; i++){
        fprintf(fp,"%s,%s,%s,%s,\n",contacts[i].name,contacts[i].pho
ne,contacts[i].email,contacts[i].category);
    }

    fclose(fp);
    printf("Contact edited successfully!\n");
}

void deleteContact(){

    displayContacts();

    char name[50];
    printf("Enter name of contact to delete: ");
    fgets(name,sizeof(name),stdin);
    name[strlen(name)-1] = '\0';

    FILE *fp = fopen("contacts.txt","r");
    if(fp == NULL){
        printf("Error opening file!!!");
```

```
        return;
    }

    struct Contact contacts[MAX_CONTACTS];
    int numContacts = 0,found = 0;
    char line[MAX_LINE];

    while(fgets(line, sizeof(line),fp)){
        char *token = strtok(line,",");
        if(token != NULL && strcmp(token,name) == 0){
            found = 1;
        } else{
            strcpy(contacts[numContacts].name,token);
            token = strtok(NULL,",");
            if(token != NULL){
                strcpy(contacts[numContacts].phone,token);
                token = strtok(NULL,",");
                if(token != NULL){
                    strcpy(contacts[numContacts].email,token);
                    token = strtok(NULL,",");
                    if(token != NULL){
                        strcpy(contacts[numContacts].category,token)
;
                        numContacts++;
                    }
                }
            }
        }
    }
    fclose(fp);

    if(found == 0){
        printf("Contact not found.\n");
        return;
    }
    fp = fopen("contacts.txt", "w");
    if(fp == NULL){
        printf("Error opening file!\n");
        return;
    }

    for(int i=0; i < numContacts; i++){
        fprintf(fp,"%s,%s,%s,%s,\n",contacts[i].name,contacts[i].pho
ne,contacts[i].email,contacts[i].category);
    }

    fclose(fp);
    printf("Contact deleted successfully!\n");
```

```
}

void deleteAllContacts(){

    char check;
    printf("Are you sure you want to delete all contacts? (y/n):");
    scanf("%c",&check);
    if(check=='y' || check=='Y'){
        FILE *fp = fopen("contacts.txt", "w");
        fclose(fp);
        printf("All contacts have been deleted successfully!\n");
    }
}

void main(){
    printf("*** Welcome to CONTACT MANAGEMENT SYSTEM ***\n\n");
    int choice,search;
    char cont = 'y';

    while(cont == 'y' || cont == 'Y'){
        printf("\n\nACTION MENU:\n");
        printf("[1] Add contact\n");
        printf("[2] Display contacts\n");
        printf("[3] Search contacts\n");
        printf("[4] Edit contact\n");
        printf("[5] Delete a contact\n");
        printf("[6] Delete all contacts\n");
        printf("[7] Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);
        getchar();

        switch(choice){
            case 1:
                addContact();
                break;
            case 2:
                displayContacts();
                break;
            case 3:
                printf("\n\nSEARCH MENU:\n");
                printf("[1] Search using Name\n");
                printf("[2] Search using Category\n");
                printf("Enter your choice: ");
                scanf("%d", &search);
                getchar();
```

```
                switch(search){
                    case 1:
                        searchName();
                        break;
                    case 2:
                        searchCategory();
                        break;
                    default:
                        printf("Invalid choice.");
                        break;
                }
                break;
            case 4:
                editContact();
                break;
            case 5:
                deleteContact();
                break;
            case 6:
                deleteAllContacts();
                break;
            case 7:
                printf("Exiting...\n");
                cont = 'n';
                break;
            default:
                printf("Invalid choice. Please enter a number from 1
to 5.\n");
                break;
        }
        if(choice != 7){
            printf("\nDo you want to continue? (y/n): ");
            scanf(" %c", &cont);
            getchar();
            system("cls");
        }
    }
}
```

Testing and Results

- ## Test Cases:

1. Add a new contact :

```
ACTION MENU:
[1] Add contact
[2] Display contacts
[3] Search contacts
[4] Edit contact
[5] Delete a contact
[6] Delete all contacts
[7] Exit
Enter your choice: 1
Enter name: Athul
Enter phone: 6282584821
Enter email: athul123@gmail.com
Enter category: friend
Contact added successfully!

Do you want to continue? (y/n):
```

2. Display all contacts :

```
ACTION MENU:
[1] Add contact
[2] Display contacts
[3] Search contacts
[4] Edit contact
[5] Delete a contact
[6] Delete all contacts
[7] Exit
Enter your choice: 2

All Contacts:

Name: Athul
Phone: 6282584821
Email: athul123@gmail.com
Category: friend


Do you want to continue? (y/n):
```

3. Search using contact name :

```
ACTION MENU:
[1] Add contact
[2] Display contacts
[3] Search contacts
[4] Edit contact
[5] Delete a contact
[6] Delete all contacts
[7] Exit
Enter your choice: 3


SEARCH MENU:
[1] Search using Name
[2] Search using Category
Enter your choice: 1
Enter name of contact to search: Athul

Name: Athul
Phone: 6282584821
Email: athul123@gmail.com
Category: friend

1 contact(s) found.
Do you want to continue? (y/n):
```

4. Search using contact category :

```
ACTION MENU:
[1] Add contact
[2] Display contacts
[3] Search contacts
[4] Edit contact
[5] Delete a contact
[6] Delete all contacts
[7] Exit
Enter your choice: 3


SEARCH MENU:
[1] Search using Name
[2] Search using Category
Enter your choice: 2
Enter category of contact to search: friend

Name: Athul
Phone: 6282584821
Email: athul123@gmail.com
Category: friend

1 contact(s) found.
Do you want to continue? (y/n):
```

5. Edit a contact :

```
ACTION MENU:
[1] Add contact
[2] Display contacts
[3] Search contacts
[4] Edit contact
[5] Delete a contact
[6] Delete all contacts
[7] Exit
Enter your choice: 4

All Contacts:

Name: Athul
Phone: 6282584821
Email: athul123@gmail.com
Category: friend

Enter name of contact to edit: Athul

Current phone: 6282584821
Enter new phone: 8590969550

Current email: athul123@gmail.com
Enter new email: athul123@gmail.com

Current category: friend
Enter new category: friend
Contact edited successfully!

Do you want to continue? (y/n):
```

6. Delete a contact :

```
ACTION MENU:
[1] Add contact
[2] Display contacts
[3] Search contacts
[4] Edit contact
[5] Delete a contact
[6] Delete all contacts
[7] Exit
Enter your choice: 5

All Contacts:

Name: Athul
Phone: 8590969550
Email: athul123@gmail.com
Category: friend

Enter name of contact to delete: Athul
Contact deleted successfully!

Do you want to continue? (y/n):
```

7. Delete all contacts :

```
ACTION MENU:
[1] Add contact
[2] Display contacts
[3] Search contacts
[4] Edit contact
[5] Delete a contact
[6] Delete all contacts
[7] Exit
Enter your choice: 6
Are you sure you want to delete all contacts? (y/n):y
All contacts have been deleted successfully!

Do you want to continue? (y/n):
```

8. Exit :

```
ACTION MENU:
[1] Add contact
[2] Display contacts
[3] Search contacts
[4] Edit contact
[5] Delete a contact
[6] Delete all contacts
[7] Exit
Enter your choice: 7
Exiting...
PS C:\MinGW\c prog>
```

Discussion of Results

Managing of contacts works well. Adding, editing or deleting contacts as well as displaying all the contacts has been made easy for the users.

Conclusion

- ## Summary of the Project:

The Contacts Management System offers an effective platform through which users can manage and view their contacts in a simplified matter.

- ## Future Enhancements:

Features that might appear in forthcoming versions of the system include integration over various platforms and also beable to store call history and enable messaging services.

References

- Contact Management System Program in c language with source code (insidethediv.com)
- Contact Management System Project Using C Language - Studytonight