

PROJECT REPORT

Course Code: CSEG1032

Course Title: Programming in C

Project Title: Contact Management System

Student Name: Prince Raj Rawat

SAP ID: 590022507

Semester: 1st

1. ABSTRACT

This project implements a menu-driven Contact Management System using the C programming language. The program allows the user to add, view, search, and delete contact records stored persistently in a file. It demonstrates the use of structures, file handling, and modular programming techniques to build a scalable and maintainable software system.

2. OBJECTIVE

The objective of this project is to apply fundamental and intermediate concepts of the C language to design a working application. It aims to help store contact information efficiently.

- Use structures to represent complex data.
- Employ functions and modular design for code organization.
- Perform file operations for persistent data storage.

3. PROBLEM DEFINITION

Almost every people require simple tools to maintain contact . Manually handling these records is error-prone and inefficient. This project provides a basic command-line system for managing contact information such as Name, Phone Number, and Email Address in a text file.

4. SYSTEM DESIGN AND ALGORITHM

The system is modular, consisting of multiple source files:

- main.c controls program flow and user interaction.
- call.c handles CRUD (Create, Read, Update/Delete) operations.

5. IMPLEMENTATION DETAILS

Key Language Features Used:

- Structures to represent contact info.
- File Handling functions (fopen, fwrite, fread, fclose) to store and retrieve data.
- Functions for modularity and clarity.

Example Code Snippet

6. OUTPUT (Sample)

Menu:

1. 1.Add Contact
 2. View Contacts
 3. Search Contact
 4. Delete Contact
 5. Exit
- 2.INPUT : 1

Sample Output

```
Enter Name: John Doe
Enter Phone: 9876543210
Enter Email: john@example.com
Contact added successfully!
```

7. CONCLUSION

The Contact Management System demonstrates the practical application of C programming concepts such as structures, file handling, and modular design. It provides a simple yet effective solution for managing contact records in small organizations.

8. FUTURE ENHANCEMENTS

- Add Update Contact feature.
- add grouping in contact like family, friends , classmate etc.
- Create a GUI-based interface for better usability.

9. APPENDIX:

main.c (Provided)

```
c:\project_contact_manager>src> main.c > ...
1 #include "contact.h"
2
3
4 int main(){
5
6     struct contact phone_info[max_contact];
7     int count=0;
8     int opt=0;
9
10    printf("starting contact manager..\n");
11
12    count = upload_from_file(phone_info);
13    printf("Loaded %d contacts from file.\n", count);
14
15
16
17    while(opt != 5)
18    {
19        menu();
20
21        printf("\nEnter :");
22        scanf("%d",&opt);
23        clear_buffer();
24
25        switch(opt){
26
27            case 1 :
28                {clear_screen();
29                 add_contact(phone_info, &count);
30                 sort_contacts(phone_info, count);
31                 break;}
32            case 2 :
33                {clear_screen();
34                 show_contacts(phone_info, count);
35                 break;}
36            case 3 :
37                {clear_screen();}
38
39
40
41            case 4 :
42                {clear_screen();
43                 delete_contact(phone_info, &count);
44                 break;}
45            case 5 :
46                {save_in_file(phone_info, count);
47                 break;}
48            default:
49                {printf("invalid choice please choose no. from 1-5.\n");}
50
51
52
53    }
54
55
56
57    return 0;
58
59 }
60
```

contact.c

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure with files like C_Programming_Project_Guide.pdf, Call_Management_System_Project..., contact.c, main.c, and contact.h.
- Editor:** Displays the content of contact.c. The code defines a menu function that prints a menu of options (1.add contact, 2.show all contact, 3.search contact, 4.delete contact, 5.save and quit) and handles user input to clear the screen or exit.
- Bottom Status Bar:** Shows the current line (Ln 1, Col 19), spaces (Spaces: 4), encoding (UTF-8), and file type (Win32).

```
#include "contact.h"

void clear_buffer(){
    int c;
    while (c=(getchar()) != '\n' && c != EOF);
}

void stripNewline(char *str){
    str[strcspn(str, "\n")] = 0;
}

void clear_screen() {
    #ifndef _WIN32
        system("cls");
    #else
        printf("\033[H\033[J");
    #endif
}

void menu(){
    printf("\n*****\n*****contact manager*****\n*****\n");
    printf("1.add contact\n");
    printf("2.show all contact\n");
    printf("3.search contact\n");
    printf("4.delete contact\n");
    printf("5.save and quit\n");
    printf("*****\n");
}
```

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure with files like C_Programming_Project_Guide.pdf, Call_Management_System_Project..., contact.c, main.c, and contact.h.
- Editor:** Displays the content of contact.c. The code includes a menu function and an add_contact function. The add_contact function prompts the user to enter contact information (first name, last name, phone number, email) and adds it to an array of contacts.
- Bottom Status Bar:** Shows the current line (Ln 1, Col 19), spaces (Spaces: 4), encoding (UTF-8), and file type (Win32).

```
void menu(){

void add_contact(struct contact phone_info[],int *count){
    system("clear");

    if (*count >= max_contact){
        printf("-----contact list is full-----");
        return;
    }

    struct contact *new_contact=&phone_info[*count];
    printf("Enter first name: ");
    fgets(new_contact->first_name,sizeof(new_contact->first_name),stdin);
    stripnewline(new_contact->first_name);

    printf("Enter last name: ");
    fgets(new_contact->last_name,sizeof(new_contact->last_name),stdin);
    stripnewline(new_contact->last_name);

    printf("Enter phone no: ");
    fgets(new_contact->phone_no,sizeof(new_contact->phone_no),stdin );
    stripnewline(new_contact->phone_no);

    printf("Enter email: ");
    fgets(new_contact->email,sizeof(new_contact->email),stdin);
    stripnewline(new_contact->email);

    printf("-----new contact added successfully-----\n");
    (*count)++;
}
```

```
c_Programming_Project_Guide.pdf U Call_Management_System_Project_Report_Prince_Raj_Rawat.docx U contact.c M main.c M contact.h D README.md web-development

File Edit Selection View Go Run Terminal Help < > git demo

c.project_contact_manager > src > contact.c ...
void add_contact(struct contact phone_info[],int *count)
{
    ...
}

void show_contacts(struct contact phone_info[],int count)
{
    system("clear");
    printf("\n-----contact list-----\n");

    if(count == 0){
        printf("-----empty list-----\n");
        return;
    }

    for (int i = 0; i<count;i++){

        printf("contact : %d\n", i+1);
        printf("name : %s %s\n", phone_info[i].first_name,phone_info[i].last_name);
        printf("phone no : %s\n", phone_info[i].phone_no);
        printf("email address : %s\n", phone_info[i].email);
    }
}

void find_contact(struct contact phone_info[],int count){
    system("clear");

    char search_term[50];
    int found_contact=0;

    printf("-----search contact-----\n");
    printf("enter the search term: ");
    fgets(search_term,sizeof(search_term),stdin);
    stripNewline(search_term);

    if (strlen(search_term)==0){
        printf("search term can not be empty\n");
    }
}

Ln 1, Col 19 Spaces:4 UTF-8 CRLF () C ⚡ Go Live Win32 ENG IN 21:53 25-11-2025
```

```
c_Programming_Project_Guide.pdf U Call_Management_System_Project_Report_Prince_Raj_Rawat.docx U contact.c M main.c M contact.h D README.md web-development

File Edit Selection View Go Run Terminal Help < > git demo

c.project_contact_manager > src > contact.c ...
void find_contact(struct contact phone_info[],int count){
    stripNewline(search_term);

    if (strlen(search_term)==0){
        printf("search term can not be empty\n");
        return;
    }

    printf("search result...\n");

    for(int i=0;i< count; i++){
        if(strcasecmp(search_term,phone_info[i].email)==0||strcasecmp(search_term,phone_info[i].first_name)==0||strcasecmp(search_term,phone_info[i].last_name)==0){
            printf("Name: %s %s\n",phone_info[i].first_name,phone_info[i].last_name);
            printf("Phone No.: %s\n",phone_info[i].phone_no);
            printf("Email Address : %s\n",phone_info[i].email);
            found_contact++;
        }
    }

    if(found_contact == 0){
        printf("no contact found containing %s \n",search_term);
    }
    else{
        printf("%d contact containing %s \n",found_contact,search_term);
    }
}

void delete_contact(struct contact phone_info[],int *count){
    system("clear");
    int delete_index=0;

    printf("\n-----delete contact-----\n");
}

Ln 1, Col 19 Spaces:4 UTF-8 CRLF () C ⚡ Go Live Win32 ENG IN 21:53 25-11-2025
```

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure with files like C_Programming_Project_Guide.pdf, Call_Management_System_Project..., contact.c, contact.h, main.c, README.md, and web-development.
- Open Editors:** Three tabs are open: C_Programming_Project_Guide.pdf, Call_Management_System.Project_Report_Prince_Raj_Rawat.docx, and contact.c (the active editor).
- Code Editor:** The contact.c file is displayed, containing code for a contact management system. The code includes functions for deleting contacts and saving them to a file.
- Bottom Status Bar:** Shows line 1, column 19, spaces: 4, UTF-8, CR LF, ENG, IN, Go Live, Win32, and the date/time 25-11-2025.

```
132 void delete_contact(struct contact phone_info[],int *count){  
133     system("clear");  
134     int delete_index=0;  
135  
136     printf("\n-----delete contact-----\n");  
137  
138     if(*count == 0){  
139         printf("Contact list is already empty\n");  
140         return;  
141     }  
142  
143     printf("Select a contact to delete.\n");  
144     for(int i=0;i< *count;i++){  
145         printf("%d. %s %s\n",i+1,phone_info[i].first_name,phone_info[i].last_name);  
146     }  
147  
148     printf("press 0 to cancel.\n");  
149     printf("Select index of the contact u want to delete : ");  
150     scanf("%d",&delete_index);  
151  
152  
153     if(delete_index == 0){  
154         printf("Deletion cancelled\n");  
155         return;  
156     }  
157  
158     if(delete_index<1 || delete_index>*count){  
159         printf("Invalid index selection,\n there is no contact on this index...\n");  
160         return;  
161     }  
162  
163     int real_index = delete_index - 1;  
164  
165     printf("Are you sure you want to delete %s (%s) %s (%s)\n",phone_info[real_index].first_name,phone_info[real_index].last_name);  
166  
167 }
```

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure with files like C_Programming_Project_Guide.pdf, Call_Management_System_Project..., contact.c, contact.h, main.c, README.md, and web-development.
- Open Editors:** Three tabs are open: C_Programming_Project_Guide.pdf, Call_Management_System.Project_Report_Prince_Raj_Rawat.docx, and contact.c (the active editor).
- Code Editor:** The contact.c file is displayed, containing code for a contact management system. The code includes functions for deleting contacts and saving them to a file.
- Bottom Status Bar:** Shows line 1, column 19, spaces: 4, UTF-8, CR LF, ENG, IN, Go Live, Win32, and the date/time 25-11-2025.

```
133 void delete_contact(struct contact phone_info[],int *count){  
134     clear_buffer();  
135     char confirm = getchar();  
136     clear_buffer();  
137  
138     if(confirm == 'y'||confirm == 'Y'){  
139         for(int i = real_index; i < *count-1; i++){  
140             phone_info[i]=phone_info[i+1];  
141         }  
142         (*count)--;  
143         printf("Contact deleted successfully.\n");  
144     }  
145     else{  
146         printf("Deletion cancelled.\n");  
147         return;  
148     }  
149  
150     void save_in_file(struct contact phone_info[],int count){  
151         FILE *file;  
152         file = fopen(filename,"wb");  
153         if(file == NULL){  
154             printf("Error: Could not open file '%s' for writing.\n", filename);  
155             return;  
156         }  
157  
158         size_t item_written = fwrite(phone_info,sizeof(struct contact),count,file);  
159         if(item_written != count){  
160             printf("Error: write %zu items instead of %d\n",item_written,count);  
161             return;  
162         }  
163     }  
164 }
```

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure with files like `C_Programming_Project_Guide.pdf`, `Call_Management_System_Project_Report_Prince_Raj_Rawat.docx`, `contact.c`, `main.c`, and `contact.h`.
- Editor:** Displays the C code for the `save_in_file` function.
- Status Bar:** Shows the line number (Ln 1, Col 19), spaces (Spaces: 4), encoding (UTF-8), and file type (CR LF).

```
c_Programming_Project_Guide.pdf U Call_Management_System_Project_Report_Prince_Raj_Rawat.docx U contact.c U main.c M contact.h D contact.h U README.md web-development

void save_in_file(struct contact phone_info[], int count) {
    FILE *file;
    file = fopen(filename, "w");
    if(file == NULL) {
        printf("No existing contact file ('%s') found. Starting fresh.\n", filename);
        return 0;
    }

    size_t itemsRead = fread(phone_info, sizeof(struct contact), max_contact, file);
    if (ferror(file)) {
        fprintf(stderr, "Error reading from file '%s'.\n", file);
    }

    fclose(file);
}

int upload_from_file(struct contact phone_info[]) {
    FILE *file;
    file = fopen(filename, "rb");
    if(file == NULL) {
        printf("No existing contact file ('%s') found. Starting fresh.\n", filename);
        return 0;
    }

    size_t itemsRead = fread(phone_info, sizeof(struct contact), max_contact, file);
    if (ferror(file)) {
        fprintf(stderr, "Error reading from file '%s'.\n", file);
    }

    fclose(file);
}

void sort_contacts(struct contact phone_info[], int count) {
    if (count < 2) {
        printf("Not enough contacts to sort.\n");
        return;
    }

    for (int i = 0; i < count - 1; i++) {
        for (int j = 0; j < count - i - 1; j++) {
            int cmp = strcasecmp(phone_info[j].first_name, phone_info[j + 1].first_name);

            if (cmp > 0) {
                struct contact temp = phone_info[j];
                phone_info[j] = phone_info[j + 1];
                phone_info[j + 1] = temp;
            }
        }
    }

    printf("Contacts sorted by name successfully.\n");
}
```

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure with files like `C_Programming_Project_Guide.pdf`, `Call_Management_System_Project_Report_Prince_Raj_Rawat.docx`, `contact.c`, `main.c`, and `contact.h`.
- Editor:** Displays the C code for the `sort_contacts` function.
- Status Bar:** Shows the line number (Ln 1, Col 19), spaces (Spaces: 4), encoding (UTF-8), and file type (CR LF).

```
c_Programming_Project_Guide.pdf U Call_Management_System_Project_Report_Prince_Raj_Rawat.docx U contact.c U main.c M contact.h D contact.h U README.md web-development

void sort_contacts(struct contact phone_info[], int count) {
    if (count < 2) {
        printf("Not enough contacts to sort.\n");
        return;
    }

    for (int i = 0; i < count - 1; i++) {
        for (int j = 0; j < count - i - 1; j++) {
            int cmp = strcasecmp(phone_info[j].first_name, phone_info[j + 1].first_name);

            if (cmp > 0) {
                struct contact temp = phone_info[j];
                phone_info[j] = phone_info[j + 1];
                phone_info[j + 1] = temp;
            }
        }
    }

    printf("Contacts sorted by name successfully.\n");
}
```