CONTACT MANAGEMENT SYSTEM

AAKRISHI TIWARI- 21BEC0245

MAY 3, 2024

EMBEDDED C DA

Aim:

The aim of the project is to develop a Phonebook Management System using C programming language.

Objectives:

1.To create a system that allows users to store, organize, and manage contact records efficiently.

2.To implement functionalities such as adding new contacts, updating existing contacts, deleting contacts, searching contacts by name, number, or city, and displaying all contacts.

3.To provide separate login panels for both administrators and users with appropriate permissions.

4.To ensure data validation and error handling to prevent storing invalid data or encountering runtime errors.

5.To utilize file handling for storing and retrieving contact information from a text file.

Description of the Project:

1.The project is a Phonebook Management System implemented in C programming language.

2.It allows users to perform various operations related to managing contacts, such as adding, updating, deleting, and searching contacts.

3.There are two types of users: administrators and regular users. Administrators have additional privileges to perform administrative tasks like adding, updating, and deleting contacts, while regular users can only view and search for contacts.

4.The system stores contact information in a text file named "ContactList.txt".

5.Users can log in using their credentials and navigate through different functionalities based on their role.

6.Data validation is implemented to ensure that only valid data is stored in the system, such as valid phone numbers and gender inputs.

7.Error handling mechanisms are in place to handle invalid inputs and display appropriate error messages to users.

Explanation of C Programming Concepts Applied in the Project:

1.File Handling: File handling is extensively used in this project to store and retrieve contact information. The project uses functions like fopen(), fclose(), fscanf(), and fprintf() to interact with text files where contact details are stored. This enables the program to read from and write to the contact list file, allowing functionalities like adding, updating, and deleting contacts.

2.Structures: The project utilizes structures to define the format of contact records. The DETAILS structure contains members like firstName, lastName, gender, findNum, and cityName, representing different attributes of a contact. These structures provide a convenient way to organize related data into a single unit.

3.Pointers and Strings: Pointers and strings are used extensively for tasks like input validation, manipulation, and comparison. Functions like isValidGender(), isValidNumeric(), removeSpaces(), and isValidName() make use of pointers to strings to perform operations like checking gender validity, validating phone numbers, removing spaces from input strings, and validating names.

4.User Input Handling: The project implements user input handling techniques to ensure robustness and user-friendliness. Functions like adminLogin() and userPanel() utilize techniques like reading input character by character with getch(), handling special keys like ENTER, TAB, and BACKSPACE, and clearing input buffer with fflush(stdin) to manage user input effectively.

5.Control Structures: Control structures like if-else statements and switch-case statements are used throughout the project to implement conditional logic and control program flow. These structures help in making decisions based on user input, validating inputs, and executing appropriate actions accordingly.

6.Recursion: Recursion is utilized in the project for functions like loginPage() and invalidInput(), where these functions call themselves based on certain conditions. This helps in implementing repetitive tasks like displaying menu options, handling invalid inputs, and navigating through different screens of the program.

7.Modular Programming: The project follows a modular approach, breaking down functionalities into smaller, manageable functions. Each function is responsible for a specific task, promoting code readability, reusability, and maintainability. Modular programming facilitates easier debugging, testing, and future enhancements of the project.

C source code with appropriate comments:

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<conio.h>
#include<windows.h>
#include<unistd.h>
#include<ctype.h>

#define MAX_LENGTH 255
#define ENTER 13
#define TAB 9
#define BKSP 8
#define USER_CODE 101
#define ADMIN_CODE 102



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



char adminPassword[MAX_LENGTH] = "Admin";  // login password for ADMINISTRATOR
char inputPassword[MAX_LENGTH];
char findNumber[MAX_LENGTH];
int choice = 0, len = 0;
DETAILS contact;
DETAILS copy;
FILE *pF = NULL;
```

```
// Function Prototypes or Function Declaration
void infoScreen();  // introduction screen
void loginPage();  // choice login type
void endScreen();  // closing screen

// login password & permission functions
void adminLogin();  // secret password
void userPanel();
void adminPanel();

// validation functions
int isValidGender(char *gender);  // checks for the gender input
int isValidNumeric(char num[]);  // checks for the phone number input
int isValidName(char *name);  // checks string for searchByCity & searchByName
int checkNumberExistence(char *num);  // checks for the input number if its exists in the database or not
char *removeSpaces(char *str);  // removes empty spaces from name inputs
void invalidInput(int);  // shows error on invalid inputs and returns them back to the panel
void clearBuffer();  // clears the output screen & input buffer

// operational functions
void addNewContact();
void updateContact();
void deleteContact();
void deleteAllContact();
void displayContact(int);  // int entryCode -> used for checking the type of user (Admin/User)
void searchByName(int);
void searchByNumber(int);
void searchByCity(int);

int main(){

   infoScreen();  // starts with introduction screen & moves on to login Page
   return 0;
}
```

```
void infoScreen()

{

   clearBuffer();

   system("title Phonebook Management System");

   system("color 60");

   printf("\t\t\t\t    Phonebook Management System   \n");

   printf("\t\t\t\t---------------------------------------- \n\n");

   printf("\t\tIt is a software application that allows users to store, organize\n\t and manage large numbers of contact records efficiently. The system is designed\n\t    to have functions such as adding, updating, viewing and deleting contacts. \n\n");

   printf("\t\t\t\t---------------------------------- \n");

   printf("\t\t\t\t   Made by AAKRISHI TIWARI-21BEC0245 \n");

   printf("\t\t\t\t---------------------------------- \n\n");

   system("pause");

   loginPage();

}


void loginPage()

{

   clearBuffer();

   system("title Login Page");

   printf("---------------------------------- \n");

   printf("   >>> Login into system as <<< \n");

   printf("---------------------------------- \n");

   printf("[1] Administrator. \n");

   printf("[2] User profile. \n");

   printf("[3] EXIT. \n");

   printf("---------------------------------- \n\n");

   printf("Enter the number & Hit ENTER: ");

   scanf("%d",&choice);


   switch (choice)

   {

      case 1:

         adminLogin();

         break;
```

```
        case 2:
            userPanel();
            break;
        case 3:
            endScreen();
            break;
        default:
            printf("---------------------------------------- \n");
            printf("ERROR: Invalid input please try again.  \n");
            printf("---------------------------------------- \n");
            system("pause");
            loginPage();
    }
}

void endScreen()
{
    clearBuffer();
    system("title Credits Page");
    printf("\n\t\t\t\t\t---------------------------------------- \n");
    printf("\t\t\t\t   >>> Phonebook Management System <<<     \n");
    printf("\t\t\t\t\t---------------------------------------- \n\n");
    printf("\t\t\t\t\t\t\tThank You. \n");
    printf("\t\t\t\t\t---------------------------------------- \n\n");
    Sleep(1500);  // pause the screen for 3 seconds
    exit(0);  // ends the program safely
}

void adminLogin()
{
    clearBuffer();
    system("title Admin Login");
    char ch;
    int i = 0;
    printf("---------------------------------- \n");
```

```
printf("   >>> Administrator Login  <<< \n");
printf("---------------------------------- \n\n");

printf("Enter your password & Hit ENTER: ");
while(1)  // runs infinite times until breaked
{
   ch = getch(); // get single character at one time & validate it
   if(ch == ENTER || ch == TAB){  // if ENTER or TAB Key is pressed save & break the loop
      inputPassword[i] = '\0';  // add NULL at the end
      break;
   }
   else if(ch == BKSP){  // if Backspace Key is pressed erase the last input
      if(i > 0){
         i--;
         printf("\b \b"); // for backspace
      }
   }
   else{
      inputPassword[i++] = ch;  // store it at (i)th of variable
      printf("* \b");              // to replace password character with *
   }
}

fflush(stdin);
printf("\n");
// verifies the password
if(strcmp(adminPassword, inputPassword) == 0)  // if both are equal then 0 is return
   adminPanel();
else
{
   printf("---------------------------------------- \n");
   printf("ERROR: Invalid password please try again.  \n");
   printf("---------------------------------------- \n");
   system("pause");
   loginPage();
```

```
    }
}

void userPanel()
{
    clearBuffer();
    system("title User Panel");
    printf("---------------------------------- \n");
    printf("\t>>> User Panel <<< \n");
    printf("---------------------------------- \n");
    printf("[1] Display All Contact. \n");
    printf("[2] Search By Name. \n");
    printf("[3] Search By Number. \n");
    printf("[4] Search By City. \n");
    printf("[5] Back to Login. \n");
    printf("[6] EXIT. \n");
    printf("---------------------------------- \n\n");
    printf("Enter the number & Hit ENTER: ");
    scanf("%d",&choice);

    switch (choice)
    {
        case 1:
            displayContact(USER_CODE);
            break;
        case 2:
            searchByName(USER_CODE);
            break;
        case 3:
            searchByNumber(USER_CODE);
            break;
        case 4:
            searchByCity(USER_CODE);
            break;
        case 5:
```

```
            loginPage();
            break;
        case 6:
            endScreen();
            break;
        default:
            printf("---------------------------------------\n");
            printf("ERROR: Invalid input please try again. \n");
            printf("---------------------------------------\n");
            system("pause");
            userPanel();
    }
}

void adminPanel()
{
    clearBuffer();
    system("title Admin Panel");
    printf("------------------------------------ \n");
    printf("\t>>> Admin Panel <<< \n");
    printf("------------------------------------ \n");
    printf(" [1] Add New Contact. \n");
    printf(" [2] Update Contact. \n");
    printf(" [3] Display All Contact. \n");
    printf(" [4] Search By Name. \n");
    printf(" [5] Search By Number. \n");
    printf(" [6] Search By City. \n");
    printf(" [7] Delete Contact. \n");
    printf(" [8] Delete All Contact. \n");
    printf(" [9] Back to Login. \n");
    printf("[10] EXIT. \n");
    printf("------------------------------------ \n\n");
    printf("Enter the number & Hit ENTER: ");
    scanf("%d",&choice);
```

```
switch (choice)
{
   case 1:
      addNewContact();
      break;
   case 2:
      updateContact();
      break;
   case 3:
      displayContact(ADMIN_CODE);
      break;
   case 4:
      searchByName(ADMIN_CODE);
      break;
   case 5:
      searchByNumber(ADMIN_CODE);
      break;
   case 6:
      searchByCity(ADMIN_CODE);
      break;
   case 7:
      deleteContact();
      break;
   case 8:
      deleteAllContact();
      break;
   case 9:
      loginPage();
      break;
   case 10:
      endScreen();
      break;
   default:
      printf("---------------------------------------\n");
      printf("ERROR: Invalid input please try again. \n");
```

```
            printf("---------------------------------------\n");
            system("pause");
            adminPanel();
    }
}

int isValidGender(char *gender)  // checks gender value through pointer
{
    switch (*gender)
    {
        case 'M':
        case 'm':
            *gender = 'm';
            return 1;
        case 'F':
        case 'f':
            *gender = 'f';
            return 1;
        default:
            printf("---------------------------------------- \n");
            printf("ERROR: Invalid gender try again later.  \n");
            printf("---------------------------------------- \n");
            return 0;
    }
    return 0;
}

int isValidName(char *name)  // checks name through pointer
{
    for(int cnt=0; cnt < 255; cnt++)
    {
        if(name[cnt] == '\0')
            break;
        if(isalpha(name[cnt]) == 0)
            return 4;
```

```
    }
    return 0;
}

int isValidNumeric(char num[])
{   len = 0;
    for(int cnt = 0; cnt < 255; cnt++)
    {
        if(num[cnt] == '\0')
            break;
        len++;
        if(isdigit(num[cnt]) == 0)
        {
            len = 0;
            return 4;
        }
    }

    if(len != 10)
    {
        printf("---------------------------------------- \n");
        printf("ERROR: phone number must be of 10 digits. \n");
        printf("---------------------------------------- \n");
        return 0;
    }
    else
        return 1;

    return 0;
}

int checkNumberExistence(char *num)  // checks for the input number if its exists in the database or not
{
    FILE *pT = fopen("ContactList.txt","r");
```

```
   while(fscanf(pT, "%s %s %c %s %s\n",copy.firstName, copy.lastName, &copy.gender, copy.findNum, copy.cityName) != EOF)
   {
      if(strcmp(num, copy.findNum) == 0)
      {
         fclose(pT);
         return 0;
      }
   }
   fclose(pT);
   return 1;
}

void invalidInput(int entryCode)
{
   printf("-------------------------------------------- \n");
   printf("ERROR: Invalid input please try again later.  \n");
   printf("-------------------------------------------- \n");
   system("pause");
   (entryCode == 101) ? userPanel() : adminPanel();
}

char *removeSpaces(char *str) // functions return type is (char pointer) and accepts argument as pointer
{
	int i = 0, j = 0;
	while (str[i])
	{
		if (str[i] != ' ')
		str[j++] = str[i];
		i++;
	}
	str[j] = '\0';
   strlwr(str);
	return str;
}
```

```
void clearBuffer()
{
   system("cls");  // clears the output screen
   fflush(stdin);  // clears the input buffers like '\n'
}

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);
    // validates gender & phoneNumber inputs to prevent storing invalid data into files
    if(isValidGender(&contact.gender))
    {
      if(isValidNumeric(contact.findNum) == 1)
      {
        if(checkNumberExistence(contact.findNum))
        {
          removeSpaces(contact.firstName);
          removeSpaces(contact.lastName);
          removeSpaces(contact.cityName);
          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
        {
          printf("---------------------------------------------------- \n");
          printf("ERROR: Phone number already exists in the database. \n");
          printf("---------------------------------------------------- \n");
        }
      }
      else
        flag = 1;
    }
  }
  else
    printf("ERROR: Unable to locate or open the file. \n");


  fclose(pF);
  if(flag == 1 && len == 0)  // if the phoneNumber entered is invalid it wil execute
```

```
    {
        printf("------------------------------------------------------- \n");
        printf("ERROR: Invalid input for phone number try again later.  \n");
        printf("------------------------------------------------------- \n");
    }
    system("pause");
    adminPanel();
}

void updateContact()
{
    clearBuffer();
    system("title Update Contact");
    char confirmDelete[MAX_LENGTH];
    int flag = 0, counter = 1;

    printf("------------------------------ \n");
    printf("   >>> Update Contact <<< \n");
    printf("------------------------------ \n\n");
    printf("Enter the phone number to update [+91]: ");
    gets(findNumber);
    printf("\n");

    if(isValidNumeric(findNumber) == 4)  // checks for each character of the number
        invalidInput(ADMIN_CODE);

    pF = fopen("ContactList.txt", "r");
    FILE *pT = fopen("temporary.txt", "w");

    while(fscanf(pF, "%s %s %c %s %s\n",contact.firstName, contact.lastName, &contact.gender, contact.findNum,
contact.cityName) != EOF)
    {
        if(strcmp(findNumber, contact.findNum) == 0)
        {
            // if there two contacts will same phoneNumber it will pop alert message
```

```
        if(counter >= 2)
           printf("\n\t\tALERT: We found another contact with the same phone number.\n\t\t\tPlease review it too.\n\n");

        printf("\t\t|=================================================================| \n");
        printf("\t\t|ID| \tName\t\t| Gender | Phone Number\t| City Name\t| \n");
        printf("\t\t|=================================================================| \n");
        printf("\t\t| %d| %s %s \t| %c |\t %s \t| %s \t| \n\n", counter++, contact.firstName, contact.lastName,
contact.gender, contact.findNum, contact.cityName);
        printf("------------------------------------------------- \n");
        printf("> Contact Found in records. Enter the new details.  \n");
        printf("------------------------------------------------- \n");
        fflush(stdin);
        printf("Enter the first name: ");
        gets(copy.firstName);

        printf("Enter the last name: ");
        gets(copy.lastName);

        printf("Enter the city name: ");
        gets(copy.cityName);

        fflush(stdin);
        printf("\n");

        printf("Type `CONFIRM` to update this details: ");
        gets(confirmDelete);
        if(strcmp(confirmDelete, "CONFIRM") == 0)
        {
           // validates everything and fetchs into file
           removeSpaces(copy.firstName);
           removeSpaces(copy.lastName);
           removeSpaces(copy.cityName);
           printf("\nProcessing . . . \n\n");
           fprintf(pT, "%s %s %c %s %s\n",copy.firstName, copy.lastName, contact.gender, contact.findNum, copy.cityName);
           flag = 2;
```

```
        }
        else
        {
           flag = 1;
           fprintf(pT, "%s %s %c %s %s\n", contact.firstName, contact.lastName, contact.gender, contact.findNum,
contact.cityName);
        }
      }
      else
         fprintf(pT, "%s %s %c %s %s\n", contact.firstName, contact.lastName, contact.gender, contact.findNum,
contact.cityName);
   }
   fclose(pF);
   fclose(pT);

   // only perform update process only if the user is ready for it
   if(flag == 1 || flag == 2)
   {
      pF = fopen("ContactList.txt", "w");
      pT = fopen("temporary.txt", "r");

      while(fscanf(pT, "%s %s %c %s %s\n",contact.firstName, contact.lastName, &contact.gender, contact.findNum,
contact.cityName) != EOF)
         fprintf(pF, "%s %s %c %s %s\n", contact.firstName, contact.lastName, contact.gender, contact.findNum,
contact.cityName);

      fclose(pF);
      fclose(pT);
   }

   pT = fopen("temporary.txt", "w");
   fclose(pT);

   if(flag == 0 && len == 10)  // return different error messages upon the flag value
   {
      printf("---------------------------------------- \n");
      printf("> No contacts found matching your input.  \n");
```

```
        printf("---------------------------------------- \n");
    }
    else if(flag == 1)
    {
        printf("---------------------------------------- \n");
        printf("> ERROR: Invalid message try again later. \n");
        printf("---------------------------------------- \n");
    }
    else if(flag == 2)
    {
        printf("----------------------------- \n");
        printf("> Success: contact updated.   \n");
        printf("----------------------------- \n");
    }

    system("pause");
    adminPanel();
}

void deleteContact()
{
    clearBuffer();
    system("title Delete Contact");
    char confirmDelete[MAX_LENGTH];
    int flag = 0, counter = 1;

    printf("------------------------------ \n");
    printf("   >>> Delete Contact <<< \n");
    printf("------------------------------ \n\n");
    printf("Enter the phone number to delete [+91]: ");
    gets(findNumber);
    printf("\n");

    if(isValidNumeric(findNumber) == 4)
        invalidInput(ADMIN_CODE);
```

```
   pF = fopen("ContactList.txt", "r");
   FILE *pT = fopen("temporary.txt", "w");

   while(fscanf(pF, "%s %s %c %s %s\n",contact.firstName, contact.lastName, &contact.gender, contact.findNum,
contact.cityName) != EOF)
   {
      if(strcmp(findNumber, contact.findNum) == 0)
      {
         if(counter >= 2)
            printf("\n\t\tALERT: We found another contact with the same phone number.\n\t\t\tPlease review it too.\n\n");

         printf("\t\t|=================================================================| \n");
         printf("\t\t|ID| \tName\t\t| Gender | Phone Number\t| City Name\t| \n");
         printf("\t\t|=================================================================| \n");
         printf("\t\t| %d| %s %s \t| %c |\t %s \t| %s \t| \n", counter++, contact.firstName, contact.lastName, contact.gender,
contact.findNum, contact.cityName);

         fflush(stdin);
         flag = 1;
         printf("\nType `CONFIRM` to delete this contact: ");
         gets(confirmDelete);

         if(strcmp(confirmDelete, "CONFIRM") == 0)
         {
            printf("\n\t\tProcessing . . . \n\n");
            flag = 2;
         }
         else
         {
            printf("\n\t\tProcessing . . . \n\n");
            fprintf(pT, "%s %s %c %s %s\n", contact.firstName, contact.lastName, contact.gender, contact.findNum,
contact.cityName);
         }
      }
      else
```

```
        fprintf(pT, "%s %s %c %s %s\n", contact.firstName, contact.lastName, contact.gender, contact.findNum,
contact.cityName);
    }
    fclose(pF);
    fclose(pT);

    if(flag == 1 || flag == 2)
    {
        pF = fopen("ContactList.txt", "w");
        pT = fopen("temporary.txt", "r");

        while(fscanf(pT, "%s %s %c %s %s\n",contact.firstName, contact.lastName, &contact.gender, contact.findNum,
contact.cityName) != EOF)
            fprintf(pF, "%s %s %c %s %s\n", contact.firstName, contact.lastName, contact.gender, contact.findNum,
contact.cityName);

        fclose(pF);
        fclose(pT);
    }

    pT = fopen("temporary.txt", "w");
    fclose(pT);

    // show the messaage according to the operations perform
    if(flag == 0 && len == 10)
    {
        printf("---------------------------------------- \n");
        printf("> No contacts found matching your input.  \n");
        printf("---------------------------------------- \n");
    }
    else if(flag == 1)
    {
        printf("---------------------------------------- \n");
        printf("> ERROR: Invalid message try again later. \n");
        printf("---------------------------------------- \n");
    }
```

```
    else if(flag == 2)
    {
        printf("------------------------------ \n");
        printf("> Success: contact deleted.   \n");
        printf("------------------------------ \n");
    }

    system("pause");
    adminPanel();
}

void deleteAllContact()
{
    clearBuffer();
    system("title Delete All Contact");
    char confirmDelete[MAX_LENGTH];
    printf("---------------------------------- \n");
    printf("   >>> Delete All Contact <<< \n");
    printf("---------------------------------- \n\n");
    printf("Type `CONFIRM` to delete all the contact. \n");
    printf("Message: ");
    gets(confirmDelete);
    printf("\n");

    // checks for the confirmation message
    choice = strcmp(confirmDelete, "CONFIRM");
    if(choice == 0)
    {
        pF = fopen("ContactList.txt", "w");  // erases everything from file and then save it
        fclose(pF);
        printf("------------------------------------------- \n");
        printf("Success: All contact details are deleted. \n");
        printf("------------------------------------------- \n");
    }
    else
```

```
    {
        printf("---------------------------------------- \n");
        printf("ERROR: Invalid message try again later. \n");
        printf("---------------------------------------- \n");
    }

    system("pause");
    adminPanel();
}

void displayContact(int entryCode)
{
    clearBuffer();
    int counter = 1;
    system("title Display All Contact");
    printf("\n\n\t\t\t\t---------------------------- \n");
    printf("\t\t\t\t   >>> Contacts List <<< \n");
    printf("\t\t\t\t---------------------------- \n\n");
    pF = fopen("ContactList.txt", "r");

    if(fscanf(pF, "%s %s %c %s %s\n",contact.firstName, contact.lastName, &contact.gender, contact.findNum,
contact.cityName) != EOF)
    {
        printf("\t\t|=================================================================| \n");
        printf("\t\t|ID| \tName\t\t| Gender | Phone Number\t| City Name\t| \n");
        printf("\t\t|=================================================================| \n");

        do
        {
            printf("\t\t| %d| %s %s \t| %c |\t %s \t| %s \t| \n", counter++, contact.firstName, contact.lastName, contact.gender,
contact.findNum, contact.cityName);

        } while(fscanf(pF, "%s %s %c %s %s\n",contact.firstName, contact.lastName, &contact.gender, contact.findNum,
contact.cityName) != EOF);

        printf("\t\t|=================================================================| \n\n");
```

```
    }
    else
        printf(" ERROR: Either we are unable to locate the ContactList.txt file or\n\t there are no contacts saved in the file.
\n\n");

    fclose(pF);
    system("pause");
    (entryCode == 101) ? userPanel() : adminPanel();
}

void searchByName(int entryCode)
{
    clearBuffer();
    system("title Search By Name");
    char findName[MAX_LENGTH];
    int compare = 1, flag = 0, counter = 1;

    printf("------------------------------ \n");
    printf("   >>> Search By Name <<<    \n");
    printf("------------------------------ \n\n");

    printf("Enter the first or last name: ");
    gets(findName);

    // if the input has space at end remove it
    if(findName[strlen(findName) - 1] == ' ')
        findName[strlen(findName) - 1] = '\0';

    printf("\n");
    strlwr(findName);

    if(isValidName(findName) == 4)
        invalidInput(entryCode);

    pF = fopen("ContactList.txt", "r");
```

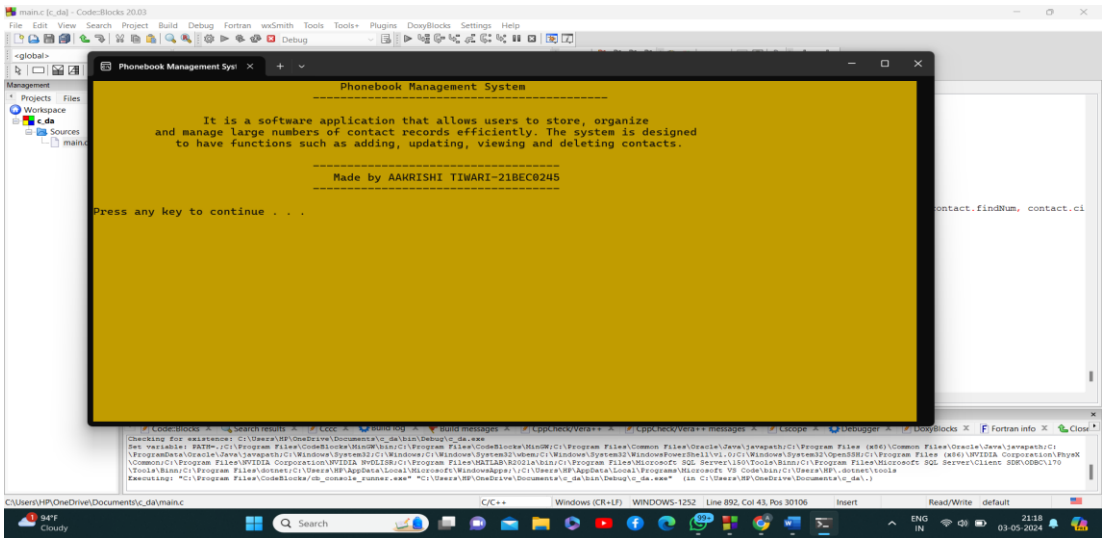
```
   while(fscanf(pF, "%s %s %c %s %s\n",contact.firstName, contact.lastName, &contact.gender, contact.findNum,
contact.cityName) != EOF)
   {
      compare = strcmp(findName, contact.firstName);

      if(compare != 0)
         compare = strcmp(findName, contact.lastName);

      if(compare == 0)
      {
         if(counter == 1)
         {
            printf("\t\t|=====================================================================| \n");
            printf("\t\t|ID| \tName\t\t| Gender | Phone Number\t| City Name\t| \n");
            printf("\t\t|=====================================================================| \n");
         }
         printf("\t\t| %d| %s %s \t| %c |\t %s \t| %s \t| \n", counter++, contact.firstName, contact.lastName, contact.gender,
contact.findNum, contact.cityName);
         flag++;
      }
   }
   fclose(pF);

   if(counter > 1)
      printf("\t\t|=====================================================================| \n\n");

   if(flag == 0)
   {
      printf("---------------------------------------- \n");
      printf("> No contacts found matching your input. \n");
      printf("---------------------------------------- \n");
   }
   system("pause");
   (entryCode == 101) ? userPanel() : adminPanel();
}
```

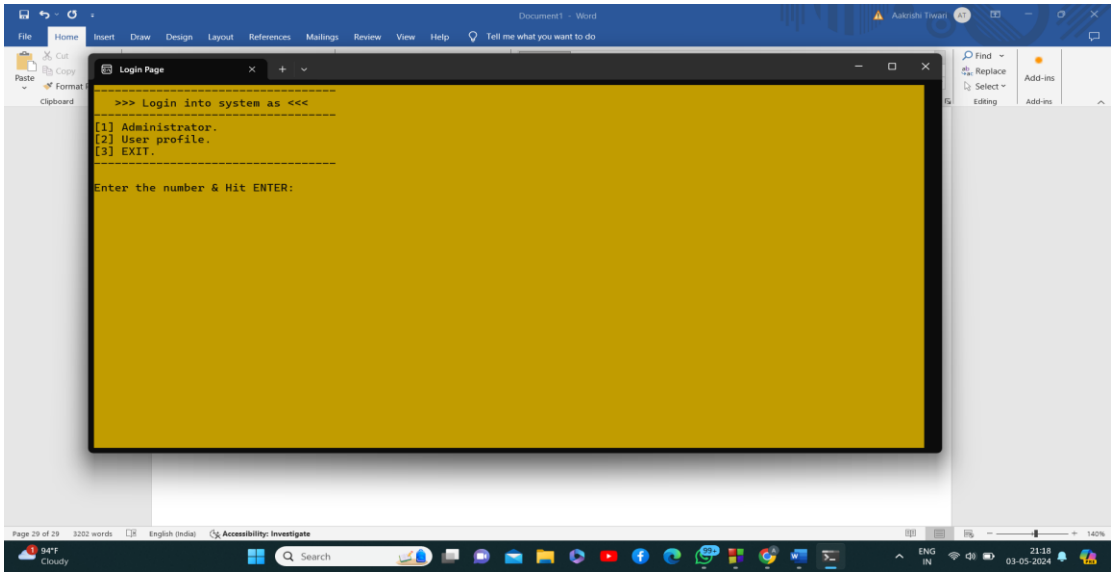
```
void searchByNumber(int entryCode)
{
   clearBuffer();
   system("title Search By Number");
   int flag = 0, counter = 1;

   printf("------------------------------- \n");
   printf("   >>> Search By Number <<< \n");
   printf("------------------------------- \n\n");

   printf("Enter the phone number [+91]: ");
   gets(findNumber);

   printf("\n");
   if(isValidNumeric(findNumber) == 4)
      invalidInput(entryCode);
   if(len != 10)
      flag++;

   pF = fopen("ContactList.txt", "r");
   while(fscanf(pF, "%s %s %c %s %s\n",contact.firstName, contact.lastName, &contact.gender, contact.findNum,
contact.cityName) != EOF)
   {
      if(strcmp(findNumber, contact.findNum) == 0)
      {
         if(counter == 1)
         {
            printf("\t\t|=================================================================| \n");
            printf("\t\t|ID| \tName\t\t| Gender | Phone Number\t| City Name\t| \n");
            printf("\t\t|=================================================================| \n");
         }
         printf("\t\t| %d| %s %s \t| %c |\t %s \t| %s \t| \n", counter++, contact.firstName, contact.lastName, contact.gender,
contact.findNum, contact.cityName);
         flag++;
      }
   }
```

```
    fclose(pF);

    if(counter > 1)
        printf("\t\t|================================================================| \n\n");

    if(flag == 0)
    {
        printf("---------------------------------------- \n");
        printf("> No contacts found matching your input. \n");
        printf("---------------------------------------- \n");
    }
    system("pause");
    (entryCode == 101) ? userPanel() : adminPanel();  // get them back to there panel according to entrycode
}

void searchByCity(int entryCode)
{
    clearBuffer();
    system("title Search By City Name");
    char findCity[MAX_LENGTH];
    int compare = 1, flag = 0, counter = 1;

    printf("----------------------------- \n");
    printf("   >>> Search By City <<<    \n");
    printf("----------------------------- \n\n");

    printf("Enter the city name: ");
    gets(findCity);

    // if the input has space at end remove it
    if(findCity[strlen(findCity) - 1] == ' ')
        findCity[strlen(findCity) - 1] = '\0';

    printf("\n");
    strlwr(findCity);
```

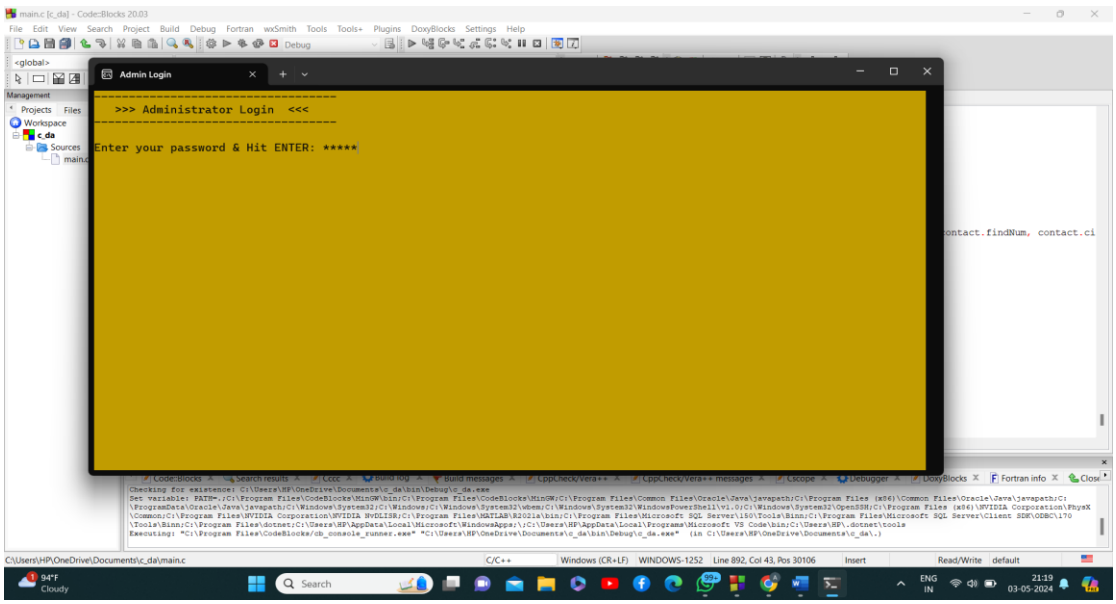
```
    if(isValidName(findCity) == 4)
        invalidInput(entryCode);

    pF = fopen("ContactList.txt", "r");
    while(fscanf(pF, "%s %s %c %s %s\n",contact.firstName, contact.lastName, &contact.gender, contact.findNum,
contact.cityName) != EOF)
    {
        compare = strcmp(findCity, contact.cityName);

        if(compare == 0)
        {
            if(counter == 1)
            {
                printf("\t\t|================================================================| \n");
                printf("\t\t|ID| \tName\t\t| Gender | Phone Number\t| City Name\t| \n");
                printf("\t\t|================================================================| \n");
            }
            printf("\t\t| %d| %s %s \t| %c |\t %s \t| %s \t| \n", counter++, contact.firstName, contact.lastName, contact.gender,
contact.findNum, contact.cityName);
            flag++;
        }
    }
    fclose(pF);

    if(counter > 1)
        printf("\t\t|================================================================| \n\n");
    if(flag == 0)
    {
        printf("---------------------------------------- \n");
        printf("> No contacts found matching your input. \n");
        printf("---------------------------------------- \n");
    }
     system("pause");
    (entryCode == 101) ? userPanel() : adminPanel();
}
```

Start-

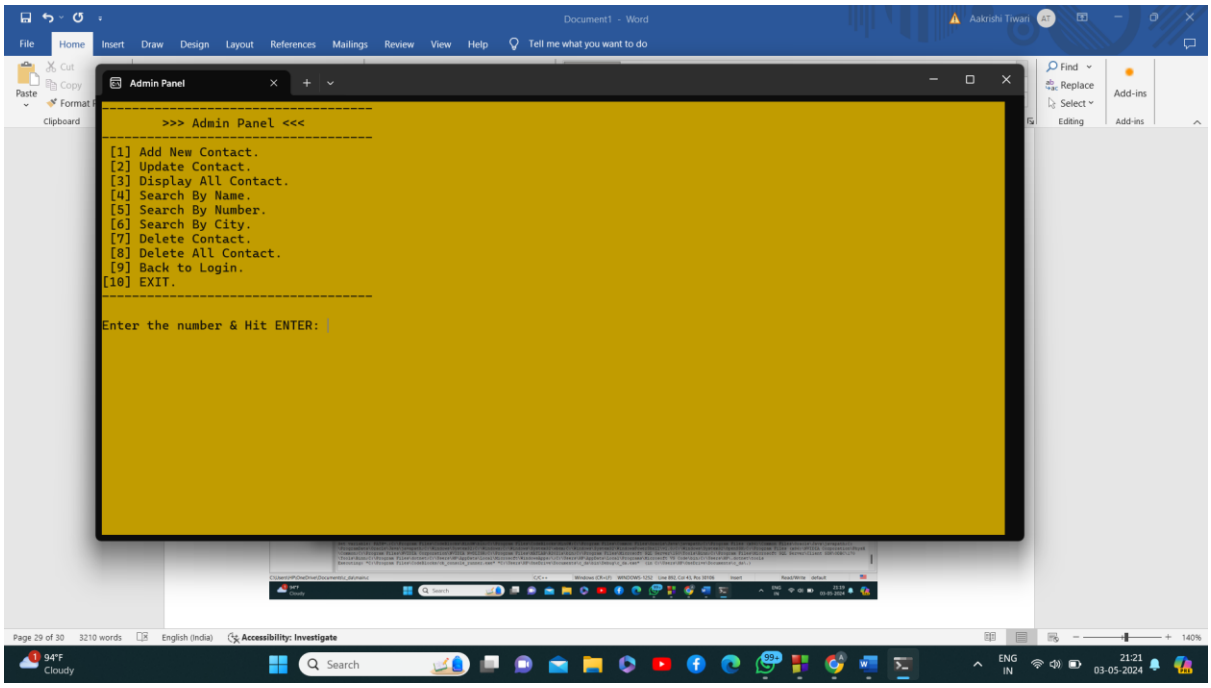




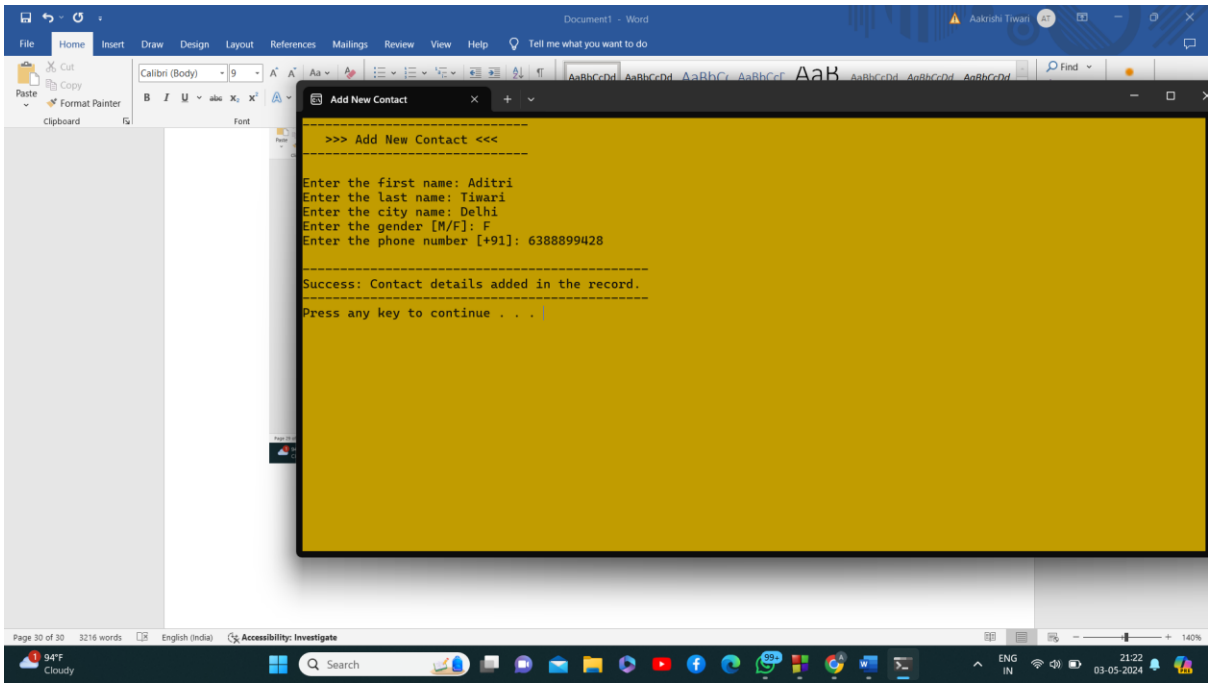
Press 1 for admin and type the password-Admin



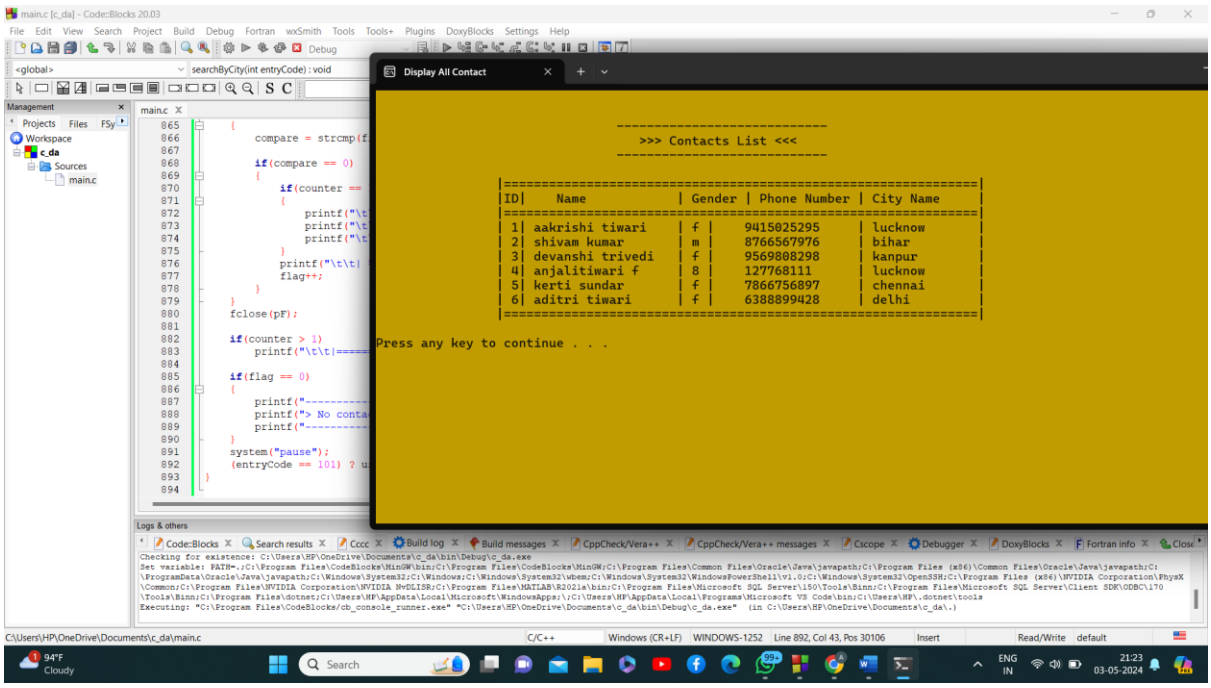
Press 1 to add new contact



Write all the information



Press 3 to view all saved contacts



Conclusion:

In conclusion, this project showcases the practical application of essential C programming concepts. By utilizing features like file handling, structures, pointers, strings, input handling, control structures, recursion, and modular programming, it efficiently manages a contact management system. Through these concepts, it achieves functionalities such as adding, updating, and deleting contacts, user authentication, input validation, and user-friendly interface navigation. Additionally, robust error handling and input validation enhance the reliability and user experience. Overall, this project demonstrates the effective application of C programming concepts in building real-world applications.