Source Code

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <ctype.h>
#include <conio.h> //windows specific
#ifdef _WIN32
#include <direct.h>
#else
#include <sys/stat.h>
#define _mkdir(x) mkdir(x, 0755)
#endif
// Console colors
#define RED "\x1b[31m"
#define GREEN "\x1b[32m"
#define YELLOW "\x1b[33m"
#define BLUE "\x1b[34m"
#define MAGENTA "\x1b[35m"
#define CYAN "\x1b[36m"
#define RESET "\x1b[0m"
// Global constants
int totalTransactions = 0;
int trans40 = 0;
int trans60 = 0;
int trans100 = 0;
int unsuccessful_attempts = 0;
int _CARD_SIZE = 32;
int _LINE_SIZE = 42;
int _PHONE_SIZE = 12;
int _TIME_SIZE = 20;
// To create all files and needed directories
void _SETUP()
{
    _mkdir("admin");
    _mkdir("admin/sold");
    _mkdir("user");
    _mkdir("user/pack");
    _mkdir("user/pass");
    _mkdir("FreshCards");
    _mkdir("UsedCards");

    const char *files[] = {
        "user/number.txt",
        "admin/balance.txt",
        "admin/blocked.txt",
        "admin/sold/40.txt",
        "admin/sold/60.txt",
        "admin/sold/100.txt",
        "FreshCards/40.txt",
        "FreshCards/60.txt",
        "FreshCards/100.txt",
        "UsedCards/40.txt",
        "UsedCards/60.txt",
        "UsedCards/100.txt"};
    for (int i = 0; i < 12; i++)
    {
        FILE *fptr = fopen(files[i], "a");
        if (fptr)
        {
            if (i < 6)
            {
                fseek(fptr, 0, SEEK_END);
                if (ftell(fptr) == 0)
                {
                    fprintf(fptr, "0");
                }
            }
            fclose(fptr);
        }
    }
}
//*** ADMIN PANEL FUNCTIONS ***
void generateCardAndAppend(int type)
{
    int card_number[20];
    int firstNum = (rand() % (9 - 2 + 1)) + 2;
    card_number[0] = firstNum;
    for (int i = 1; i < 20; i++)
    {
```

```
        card_number[i] = (rand() % (9 - 0 + 1)) + 0;
    }

    char filepath[100];
    sprintf(filepath, "FreshCards/%d.txt", type);

    FILE *fptr = fopen(filepath, "a");
    for (int i = 0; i < 20; i++)
    {
        fprintf(fptr, "%d", card_number[i]);
    }
    fprintf(fptr, "\n");
    fclose(fptr);
}
void cardDialog()
{
    printf("Enter the amount of cards: ");
    int amount;
    int typop1 = scanf("%d", &amount);
    if (typop1 == 0)
    {
        printf("\n" RED "Sorry please enter an integer type data" RESET "\n");
        char ch;
        while ((ch = getchar()) != '\n' && ch != EOF)
            ;
        amount = 0;
    }
    printf("\n Enter the type of card (40/60/100): ");
    int type;
    int typop2 = scanf("%d", &type);
    if (typop2 == 0)
    {
        printf("\n" RED "Sorry please enter an integer type data" RESET "\n");
        char ch;
        while ((ch = getchar()) != '\n' && ch != EOF)
            ;
        type = 40;
    }
    for (int i = 0; i < amount; i++)
    {
        generateCardAndAppend(type);
    }
    printf("\n\n\t" GREEN "Successfully Created %d cards of %d mins" RESET "\n\n", amount, type);
}
void removeCard(int type, int lineNum)
{
    char filepath[100];
    sprintf(filepath, "FreshCards/%d.txt", type);
    FILE *realfile = fopen(filepath, "r");
    FILE *tmpfile = fopen("FreshCards/temp.txt", "w");
    char real_card[_CARD_SIZE];
    int check_line = 1;
    while (fgets(real_card, _CARD_SIZE, realfile) != NULL)
    {
        if (strlen(real_card) <= 1)
            continue;

        if (check_line != lineNum)
        {
            fprintf(tmpfile, "%s\n", real_card);
        }
        check_line++;
    }
    fclose(realfile);
    fclose(tmpfile);

    if (remove(filepath) != 0)
    {
        perror("Error deleting original file");
        return;
    }

    if (rename("FreshCards/temp.txt", filepath) != 0)
    {
        perror("Error renaming temp file");
        return;
    }
}
void deleteCard(int min)
{
```

```
    printf("Enter card number: ");
    char entered_card[_CARD_SIZE];
    getchar();
    fgets(entered_card, _CARD_SIZE, stdin);
    entered_card[strcspn(entered_card, "\n")] = 0;

    char filepath[100];
    sprintf(filepath, "FreshCards/%d.txt", min);

    FILE *fptr = fopen(filepath, "r");
    char real_card[_CARD_SIZE];
    int found = 0;
    int lineNum = 0;

    while (fgets(real_card, _CARD_SIZE, fptr) != NULL)
    {
        if (strlen(real_card) <= 1)
            continue;
        real_card[strcspn(real_card, "\n")] = 0;

        if (strcmp(entered_card, real_card) == 0)
        {
            printf(GREEN "Card Found! at line %d" RESET "\n", lineNum + 1);
            found = 1;
            break;
        }
        lineNum++;
    }

    if (!found)
    {
        printf(RED "Card does not exist!" RESET "\n");
        return;
    }

    fclose(fptr);
    removeCard(min, lineNum + 1);
    printf(GREEN "Card Deleted successfully" RESET "\n");
}
void deleteDialog()
{
    printf("What type of card do you want to delete?\n\t1. 40min\n\t2. 60min\n\t3. 100min\n\nEnter an option: ");
    int op;
    int typop = scanf("%d", &op);
    if (typop == 0)
    {
        printf("\n" RED "Sorry please enter an integer type data" RESET "\n");
        char ch;
        while ((ch = getchar()) != '\n' && ch != EOF)
            ;
    }
    if (op == 1)
    {
        deleteCard(40);
    }
    else if (op == 2)
    {
        deleteCard(60);
    }
    else if (op == 3)
    {
        deleteCard(100);
    }
    else
    {
        printf(RED "Invalid Selection" RESET "\n");
    }
}
void unblock(int lineNum)
{
    FILE *realfile = fopen("admin/blocked.txt", "r");
    FILE *tmpfile = fopen("admin/temp.txt", "w");

    char phn[_PHONE_SIZE];
    int check_line = 1;

    while (fgets(phn, _PHONE_SIZE, realfile) != NULL)
    {
        if (strlen(phn) <= 1)
            continue;
```

```
        phn[strcspn(phn, "\n")] = 0;

        if (check_line != lineNum)
        {
            fprintf(tmpfile, "%s\n", phn);
        }
        check_line++;
    }
    fclose(realfile);
    fclose(tmpfile);

    if (remove("admin/blocked.txt") != 0)
    {
        perror("Error deleting original file");
        return;
    }

    if (rename("admin/temp.txt", "admin/blocked.txt") != 0)
    {
        perror("Error renaming temp file");
        return;
    }
}
void showBlockList()
{
    printf(YELLOW "Block List: " RESET "\n\n");
    int count = 0;
    FILE *fptr = fopen("admin/blocked.txt", "r");
    char phn[_PHONE_SIZE];
    while (fgets(phn, _PHONE_SIZE, fptr) != NULL)
    {
        if (strlen(phn) <= 1)
            continue;
        phn[strcspn(phn, "\n")] = 0;
        printf("%s\n", phn);
        count++;
    }
    if (count > 0)
        printf(GREEN "Total %d numbers found in the list." RESET "\n", count);
    else
        printf(GREEN "List is Empty" RESET "\n");

    fclose(fptr);
}
void unlockAccount()
{
    showBlockList();

    printf("Enter user Number: ");
    char user_phn[_PHONE_SIZE];
    getchar();
    fgets(user_phn, _PHONE_SIZE, stdin);
    user_phn[strcspn(user_phn, "\n")] = 0;

    FILE *fptr = fopen("admin/blocked.txt", "r");
    char phn[_PHONE_SIZE];
    int found = 0;
    int lineNum = 0;

    while (fgets(phn, _PHONE_SIZE, fptr) != NULL)
    {
        if (strlen(phn) <= 1)
            continue;
        phn[strcspn(phn, "\n")] = 0;

        if (strcmp(user_phn, phn) == 0)
        {
            printf("User Found! at line %d\n", lineNum + 1);
            found = 1;
            break;
        }
        lineNum++;
    }
    fclose(fptr);

    if (found)
    {
        unblock(lineNum + 1);
        printf(GREEN "User unblocked successfully" RESET "\n");
    }
```

```
    else
    {
        printf(RED "The number is not in the blocked list" RESET "\n");
    }
}
void showSoldCards(int min)
{
    char filepath[100];
    sprintf(filepath, "admin/sold/%d.txt", min);

    FILE *fptr = fopen(filepath, "r");
    char sold_count[6];
    fscanf(fptr, "%5s", sold_count);
    printf("\t %d Minutes: %s times\t", min, sold_count);
    fclose(fptr);
}
void showSoldCardsInTaka(int min)
{
    char filepath[100];
    sprintf(filepath, "admin/sold/%d.txt", min);

    FILE *fptr = fopen(filepath, "r");
    char sold_count[6];
    fscanf(fptr, "%5s", sold_count);
    int amount;
    if (min == 40)
        amount = 50;
    else if (min == 60)
        amount = 70;
    else
        amount = 120;

    amount *= atoi(sold_count);
    printf("\t %d Minutes: %d Taka\t", min, amount);
    fclose(fptr);
}
void showStock(int min)
{
    char filepath[100];
    sprintf(filepath, "FreshCards/%d.txt", min);

    FILE *fptr = fopen(filepath, "r");
    char real_card[_CARD_SIZE];
    int lineNum = 0;

    while (fgets(real_card, _CARD_SIZE, fptr) != NULL)
    {
        if (strlen(real_card) <= 1)
            continue;
        lineNum++;
    }
    fclose(fptr);
    printf("\t %d is available : %d\t", min, lineNum);
}
void showAdminBalance()
{
    FILE *fptr = fopen("admin/balance.txt", "r");
    char bal[12];
    fscanf(fptr, "%11s", bal);
    printf("\t " GREEN "Total balance %s Taka" RESET "\n", bal);
    fclose(fptr);
}
void showStatistics()
{
    printf("\n" YELLOW "Statistics: " RESET "\n");
    showSoldCards(40);
    showSoldCards(60);
    showSoldCards(100);
    printf("\n");
    showStock(40);
    showStock(60);
    showStock(100);
    printf("\n");
    showSoldCardsInTaka(40);
    showSoldCardsInTaka(60);
    showSoldCardsInTaka(100);
    printf("\n\n");
    showAdminBalance();
}
void formatDate(char stamp[])
```

```
{
    time_t rawtime = (time_t)atol(stamp);
    rawtime += 6 * 3600;
    struct tm *timeinfo = gmtime(&rawtime);

    if (timeinfo == NULL)
    {
        printf("Invalid time conversion\n");
        return;
    }

    printf("Date: %02d-%02d-%04d \nTime: %02d:%02d:%02d\n",
           timeinfo->tm_mday,
           timeinfo->tm_mon + 1,
           timeinfo->tm_year + 1900,
           timeinfo->tm_hour,
           timeinfo->tm_min,
           timeinfo->tm_sec);
}
void showTransaction(int min)
{
    char filepath[100];
    sprintf(filepath, "UsedCards/%d.txt", min);

    FILE *fptr = fopen(filepath, "r");
    char line[44];
    char card_num[22], timestamp_str[20], phone[12];
    int ifa = 1;
    while (fgets(line, 44, fptr) != NULL)
    {
        int n = sscanf(line, "%21s %19s %11s", card_num, timestamp_str, phone);
        if (n == 3)
        {
            if (min == 40)
                trans40++;
            else if (min == 60)
                trans60++;
            else
                trans100++;
            totalTransactions++;
            if (ifa)
            {
                printf("\n" CYAN "For %d minutes pack:" RESET " \n", min);
                ifa = 0;
            }
            printf("\n\nCard Number: ");
            for (int i = 0; i < 20; i++)
            {
                printf("%c", card_num[i]);
                if ((i + 1) % 4 == 0)
                {
                    printf(" ");
                }
            }
            printf("\n");
            formatDate(timestamp_str);
            printf("Phone: %s\n", phone);
        }
    }
    fclose(fptr);
}
void showHistory()
{
    printf(YELLOW "Transaction History: " RESET "\n");
    showTransaction(40);
    showTransaction(60);
    showTransaction(100);
    printf("\n");
    printf("\nTotal %02d transactions found for all packs.\t 40 mins: %02d\t 60 mins: %02d\t 100 mins: %02d\n",
totalTransactions, trans40, trans60, trans100);
    totalTransactions = 0;
    trans40 = 0;
    trans60 = 0;
    trans100 = 0;
}
void showUserTransaction(char phn[], int min)
{
    char filepath[100];
    sprintf(filepath, "UsedCards/%d.txt", min);
```

```
    FILE *fptr = fopen(filepath, "r");
    char line[44];
    char file_num[11];
    char card_num[21];
    char timestamp_str[19];
    int ifa = 1;
    while (fgets(line, 44, fptr) != NULL)
    {
        if (strlen(line) <= 1)
            continue;
        if (ifa)
        {
            printf("\n\t" CYAN "Transactions of %s for %d minutes pack:" RESET " \n", phn, min);
            ifa = 0;
        }
        line[strcspn(line, "\n")] = 0;
        sscanf(line, "%21s %19s %11s", card_num, timestamp_str, file_num);

        if (strcmp(file_num, phn) == 0)
        {
            totalTransactions++;
            printf("\nCard Number: ");
            for (int i = 0; i < 20; i++)
            {
                printf("%c", card_num[i]);
                if ((i + 1) % 4 == 0)
                {
                    printf(" ");
                }
            }
            printf("\n");
            formatDate(timestamp_str);
            printf("\n");
            if (min == 40)
                trans40++;
            else if (min == 60)
                trans60++;
            else
                trans100++;
        }
    }
    fclose(fptr);
}
void searchUserTransaction()
{
    printf("Enter user number: ");
    char phn[_PHONE_SIZE];
    getchar();
    fgets(phn, _PHONE_SIZE, stdin);
    showUserTransaction(phn, 40);
    showUserTransaction(phn, 60);
    showUserTransaction(phn, 100);
    printf("\nTotal %02d transactions found for " GREEN "%s" RESET " for all packs\n\t 40 mins: %02d\n\t 60 mins: %02d\n\t 100
mins: %02d\n", totalTransactions, phn, trans40, trans60, trans100);
    totalTransactions = 0;
    trans40 = 0;
    trans60 = 0;
    trans100 = 0;
}
void adminPanel()
{
    printf("\nWelcome to Admin Panel!\n\n" YELLOW "The Admin Menu:" RESET "\n\t0. Menu\n\t1. New Card\n\t2. Delete Card\n\t3.
Unlock Account\n\t4. History\n\t5. Statistics\n\t6. Search\n\t7. Exit\n\n");
    while (1)
    {
        printf("[" GREEN "ADMIN" RESET "] Enter your choice: ");
        int op;
        int typop = scanf("%d", &op);
        if (typop == 0)
        {
            printf("\n" RED "Sorry please enter an integer type data!" RESET "\n");
            char ch;
            while ((ch = getchar()) != '\n' && ch != EOF)
                ;
        }
        if (op == 0)
        {
            printf("\n" YELLOW "The Admin Menu:" RESET "\n\t0. Menu\n\t1. New Card\n\t2. Delete Card\n\t3. Unlock Account\n\t4.
History\n\t5. Statistics\n\t6. Search\n\t7. Exit\n\n");
        }
```

```
        else if (op == 1)
        {
            cardDialog();
        }
        else if (op == 2)
        {
            deleteDialog();
        }
        else if (op == 3)
        {
            unlockAccount();
        }
        else if (op == 4)
        {
            showHistory();
        }
        else if (op == 5)
        {
            showStatistics();
        }
        else if (op == 6)
        {
            searchUserTransaction();
        }
        else if (op == 7)
        {
            printf(GREEN "Returning to main menu\n" RESET);
            break;
        }

        else
        {
            printf(RED "Invalid Selection!" RESET "\n");
        }
    }
}
//*** USER PANEL FUNCTIONS ***
void resetCurrentNumber()
{
    FILE *fptr2 = fopen("user/number.txt", "w");
    fprintf(fptr2, "%d", 0);
    fclose(fptr2);
}
void updateSold(int min)
{
    char filepath[100];
    sprintf(filepath, "admin/sold/%d.txt", min);

    FILE *fptr = fopen(filepath, "r+");
    char count[11];
    fscanf(fptr, "%10s", count);

    int sold = atoi(count);
    int final = sold + 1;
    char finalsold[20];
    sprintf(finalsold, "%d", final);

    fseek(fptr, 0, SEEK_SET);
    fprintf(fptr, "%s", finalsold);
    fclose(fptr);
}
void updateAdminBalance(int val)
{
    FILE *fptr = fopen("admin/balance.txt", "r+");
    char bal[11];
    fscanf(fptr, "%10s", bal);

    int balance = atoi(bal);
    int final = balance + val;
    char finalBalance[20];
    sprintf(finalBalance, "%d", final);

    fseek(fptr, 0, SEEK_SET);
    fprintf(fptr, "%s", finalBalance);
    fclose(fptr);
}
void showMinutes()
{
    printf("\n\tAvailable Minutes: ");
    char filepathofuser[100] = "user/pack/";
```

```
    FILE *unp = fopen("user/number.txt", "r");
    char cur_user_num[25];
    fscanf(unp, "%11s", cur_user_num);
    strcat(filepathofuser, cur_user_num);
    strcat(filepathofuser, ".txt");
    fclose(unp);

    FILE *fptr2 = fopen(filepathofuser, "r");
    if (!fptr2)
    {
        printf("0 minutes\n\n");
        return;
    }
    char bal[20];
    if (fscanf(fptr2, "%19s", bal) != 1)
    {
        printf("0 minutes\n\n");
        fclose(fptr2);
        return;
    }
    fclose(fptr2);
    printf("%s minutes\n\n", bal);
}
void updateBalance(int val)
{
    char filepathofuser[100] = "user/pack/";
    FILE *unp = fopen("user/number.txt", "r");
    char cur_user_num[25];
    fscanf(unp, "%11s", cur_user_num);
    strcat(filepathofuser, cur_user_num);
    strcat(filepathofuser, ".txt");
    fclose(unp);

    FILE *fptr = fopen(filepathofuser, "r+");
    char bal[6];
    fscanf(fptr, "%5s", bal);

    int balance = atoi(bal);
    int final = balance + val;
    char finalBalance[20];
    sprintf(finalBalance, "%d", final);

    fseek(fptr, 0, SEEK_SET);
    fprintf(fptr, "%s", finalBalance);
    fclose(fptr);
}
void commitRecharge(int min, int charge)
{
    printf("Please enter the scratch card number: ");
    char user_card[_CARD_SIZE];
    getchar();
    fgets(user_card, _CARD_SIZE, stdin);
    user_card[strcspn(user_card, "\n")] = 0;

    char filepath[100];
    sprintf(filepath, "FreshCards/%d.txt", min);

    FILE *fptr = fopen(filepath, "r");
    char real_card[_CARD_SIZE];
    int found = 0;
    int lineNum = 0;

    while (fgets(real_card, _CARD_SIZE, fptr) != NULL)
    {
        if (strlen(real_card) <= 1)
            continue;
        real_card[strcspn(real_card, "\n")] = 0;

        if (strcmp(user_card, real_card) == 0)
        {
            printf(GREEN "Card is Valid! Found at line %d in package list" RED "\n", lineNum + 1);
            FILE *usrnum = fopen("user/number.txt", "r");
            char usrnumber[_PHONE_SIZE];
            fscanf(usrnum, "%11s", usrnumber);
            fclose(usrnum);

            time_t currentTime;
            time(&currentTime);

            char filename2[6];
```

```
            sprintf(filename2, "%d", min);
            char filepath2[100] = "UsedCards/";
            strcat(filepath2, filename2);
            strcat(filepath2, ".txt");

            FILE *fptr3 = fopen(filepath2, "a");
            fprintf(fptr3, "%s %ld %s\n", user_card, currentTime, usrnumber);
            fclose(fptr3);

            found = 1;
            break;
        }
        lineNum++;
    }

    if (!found)
    {
        unsuccessful_attempts++;
        if (unsuccessful_attempts >= 3)
        {
            FILE *fptr = fopen("admin/blocked.txt", "r");
            FILE *fptr2 = fopen("user/number.txt", "r");
            char number[_PHONE_SIZE];
            fscanf(fptr2, "%11s", number);
            fclose(fptr2);

            resetCurrentNumber();

            int count = 0;
            int ch;
            while ((ch = fgetc(fptr)) != EOF)
            {
                count++;
            }
            if (count < 5)
            {
                fclose(fptr);
                FILE *xmp = fopen("admin/blocked.txt", "w");
                fclose(xmp);
            }
            FILE *fptr3 = fopen("admin/blocked.txt", "a");
            fprintf(fptr3, "%s\n", number);
            fclose(fptr3);

            printf(RED "Sorry Account blocked!" RESET "\n\n");
            exit(0);
        }
        printf(RED "Invalid card!" RESET "\n");
        return;
    }

    fclose(fptr);
    updateBalance(min);
    updateAdminBalance(charge);
    updateSold(min);
    removeCard(min, lineNum + 1);
    printf("\n\n\t" GREEN "[OK] Account Recharged Successfully" RESET "\n\n");
}
void makeRecharge()
{
    printf(YELLOW "Recharge Menu" RESET "\n\n\t1. 40 min at Tk 50\n\t2. 60 min at Tk. 70\n\t3. 100 min at Tk. 120\n\n");
    int rch;
    printf("Select an option: ");
    int typop = scanf("%d", &rch);
    if (typop == 0)
    {
        printf("\n" RED "Sorry please enter an integer type data" RESET "\n");
        char ch;
        while ((ch = getchar()) != '\n' && ch != EOF)
            ;
    }

    if (rch == 1)
    {
        commitRecharge(40, 50);
    }
    else if (rch == 2)
    {
        commitRecharge(60, 70);
    }
```

```
    else if (rch == 3)
    {
        commitRecharge(100, 120);
    }
    else
    {
        printf(RED "Invalid selection" RESET "\n");
    }
}
void showNumber()
{
    FILE *fptr = fopen("user/number.txt", "r+");
    char number[_PHONE_SIZE];
    fscanf(fptr, "%11s", number);
    printf("\n\nCurrent User: %s\n\n", number);
    fclose(fptr);
}
int isBlocked(char user_phn[])
{
    FILE *fptr = fopen("admin/blocked.txt", "r");
    char phn[_PHONE_SIZE];
    int found = 0;
    int lineNum = 0;

    while (fgets(phn, _PHONE_SIZE, fptr) != NULL)
    {
        if (strlen(phn) <= 1)
            continue;
        phn[strcspn(phn, "\n")] = 0;

        if (strcmp(user_phn, phn) == 0)
        {
            printf(RED "User is suspended. Found at line" RESET " %d\n", lineNum + 1);
            printf("\n" YELLOW "logging out..." RESET "\n");
            found = 1;
            break;
        }
        lineNum++;
    }
    fclose(fptr);
    return found;
}
int checkIfBlocked()
{
    FILE *fptr = fopen("user/number.txt", "r");
    char number[_PHONE_SIZE];
    fscanf(fptr, "%11s", number);
    fclose(fptr);
    if (isBlocked(number))
    {
        resetCurrentNumber();
        printf(RED "Sorry your account is blocked" RESET "\n\n");
        return 1;
    }
    return 0;
}
void logOut()
{
    resetCurrentNumber();
    printf("\n" GREEN "Logged out." RESET "\n");
}
void showDialog()
{
    showNumber();
    printf(YELLOW "User Menu:" RESET "\n\t1. Check Balance\n\t2. Recharge\n\t3. Log out\n\t4. Exit\n\n");
    while (1)
    {
        printf("[" MAGENTA "USER" RESET "] Enter your choice: ");
        int select;
        int typop = scanf("%d", &select);
        if (typop == 0)
        {
            printf("\n" RED "Sorry please enter an integer type data" RESET "\n");
            char ch;
            while ((ch = getchar()) != '\n' && ch != EOF)
                ;
        }
        if (select == 0)
        {
            printf(YELLOW "User Menu:" RESET "\n\t1. Check Balance\n\t2. Recharge\n\t3. Log out\n\t4. Exit\n\n");
```

```
        }
        else if (select == 1)
        {
            showMinutes();
        }
        else if (select == 2)
        {
            makeRecharge();
        }
        else if (select == 3)
        {
            logOut();
            break;
        }
        else if (select == 4)
        {
            printf(GREEN "Program Closed" RESET "\n");
            exit(0);
        }
        else
        {
            printf(RED "Invalid selection! Please try again later." RESET "\n");
        }
    }
}
void clientPanel()
{
    FILE *fptr = fopen("user/number.txt", "r+");
    char number[_PHONE_SIZE];
    fscanf(fptr, "%11s", number);
    if (strcmp(number, "0") == 0)
    {
        printf(YELLOW "Client Side" RESET "\nYou have to register or log in first. Enter phone Number: ");
        char phn[_PHONE_SIZE];
        getchar();
        fgets(phn, _PHONE_SIZE, stdin);
        phn[strcspn(phn, "\n")] = 0;
        if (strlen(phn) != 11)
        {
            printf(RED "Invalid phone number. It must contain 11 digits. It has only %d digits." RESET "\n", strlen(phn));
            fclose(fptr);
            return;
        }
        fseek(fptr, 0, SEEK_SET);
        fprintf(fptr, "%s", phn);
        fclose(fptr);

        char filepathofuser[100];
        sprintf(filepathofuser, "user/pack/%s.txt", phn);

        FILE *user_pack = fopen(filepathofuser, "r");

        if (!user_pack)
        {
            FILE *tmpup = fopen(filepathofuser, "w");
            fprintf(tmpup, "%d", 0);
            fclose(tmpup);

            char filepathofuser2[100];
            sprintf(filepathofuser2, "user/pass/%s.txt", phn);

            printf("\nSelect a Strong password: ");
            char pass[500];
            getchar();
            fgets(pass, 500, stdin);
            FILE *upass = fopen(filepathofuser2, "w");
            fprintf(upass, "%s", pass);
            fclose(upass);

            printf("\n\n" GREEN " Registration finished. Please remember your password." RESET "\n\n");
        }
        else
        {
            fclose(user_pack);
            printf("\n" GREEN "You already have an existing account. Enter the password below." RESET "\n");

            printf("\nPassword: ");
            char pass[500];
            getchar();
            fgets(pass, 500, stdin);
```

```
            char fpath[100];
            sprintf(fpath, "user/pass/%s.txt", phn);

            FILE *eupass = fopen(fpath, "r");
            char savedpass[500];
            fscanf(eupass, "%s", savedpass);
            fclose(eupass);

            savedpass[strcspn(savedpass, "\n")] = 0;
            pass[strcspn(pass, "\n")] = 0;
            if (strcmp(savedpass, pass) == 0)
            {
                printf("\n" GREEN "Profile found! and password matched" RESET "\n");
                if (checkIfBlocked())
                {
                    return;
                }
                else
                {
                    printf("\n\n" GREEN "Welcome back! Logged in successfully." RESET "\n\n");
                }
            }
            else
            {
                printf("\n" RED "Sorry wrong password. Try again later" RESET "\n");
                resetCurrentNumber();
                printf("\n" YELLOW "All sessions are cleared and program exited." RESET "\n");
                return;
            }
        }
        showDialog();
    }
    else
    {
        fclose(fptr);
        if (checkIfBlocked())
        {
            return;
        }
        showDialog();
    }
}
// ***MAIN APP***
int main()
{
    srand(time(NULL));
    _SETUP(); // creating file+directories if not exists
    printf(YELLOW "[*] CSE Assignment by BITTO SAHA (2403142)" RESET "\n\nSELECT AN USER:\n\t1. Admin\n\t2. User\n\t3.
Exit\n\n");

    while (1)
    {
        printf("[APP] Enter your choice: ");
        int op;
        int typop = scanf("%d", &op);
        if (typop == 0)
        {
            printf("\n" RED "Sorry please enter an integer type data" RESET "\n");
            char ch;
            while ((ch = getchar()) != '\n' && ch != EOF)
                ;
            continue;
        }
        if (op == 1)
        {
            printf("Enter Admin password: ");
            int password;
            scanf("%d", &password);
            if (password == 61770)
            {
                printf("\n" GREEN "Success!" RESET "\n");
                adminPanel();
                printf("\n" GREEN "Getting back to menu..." RESET "\n");
            }
            else
            {
                printf(RED "Sorry authentication failed!\nRestart or (Press enter)\n" RESET);
            }
        }
```

```
        else if (op == 2)
        {
            clientPanel();
            printf("\n" GREEN "Getting back to menu..." RESET "\n");
        }
        else if (op == 3)
        {
            printf("\n\tThanks for using. Press any key to close :)\n\n");
            break;
        }
        else
        {
            printf(RED "Sorry That option does not exist. Try again later" RESET "\n\n");
        }
    }
    return 0;
}
```