# **MOBILE RECHARGE SYSTEM**

*A*

*Mini Project Report*

*Submitted in partial fulfilment of the*

*Requirements for the award of the Degree of*

**BACHELOR OF ENGINEERING**

IN

**INFORMATION TECHNOLOGY**

By

MUMMOJU HARSHIT- 1602-19-737-132

SANJITH REDDY – 1602-19-737-162



**Department of Information Technology**

**Vasavi College of Engineering (Autonomous)**

**(Affiliated to Osmania University)**

**Ibrahimbagh, Hyderabad-31**

**2020**

**Vasavi College of Engineering (Autonomous)**

**(Affiliated to Osmania University)**

**Hyderabad-500 031**

**Department of Information Technology**



**DECLARATION BY THE CANDIDATE**

We, MUMMOJU HARSHIT and SANITH REDDY bearing hall ticket numbers, 1602-19-737-132 and1602-19-737-162, hereby declare that the project report entitled “MOBILE RECHARGE SYSTEM” is submitted in partial fulfilment of the requirement for the award of the degree of Bachelor of Engineering in Information Technology.

This is a record of bonafide work carried out by us and the results embodied in this project report have not been submitted to any other university or institute for the award of any other degree or diploma.

MUMMOJU HARSHIT

1602-19-737-132

SANJITH REDDY

1602-19-737-162

(Faculty In-Charge) (Head.Dept IT)

**Acknowledgements**

The satisfaction that accompanies the successful completion of this project would not be in completewithout the mention of the people who made it possible, without whose constant guidance andencouragement would have made efforts go in vain. We consider ourselves privileged to expressgratitude and respect towards all those who guided us through the completion of this project.

We convey thanks to my project guide Mrs.Divya of Information technology Department for providing encouragement, constant support and guidance which was of a greathelp to complete this project successfully.

Last but not the least, we wish to thank our parents for financing our studies in this privileged Vasavi College of Engineering aswell as for constantly encouraging us to learn engineering. Their personal sacrifice in providingthis opportunity to learn engineering is gratefully acknowledged.

**Abstract**

**Objective**:

1. Recharge mobile phones of different service providers.

2. Service Providers: JIO.

3. Recharge Plans: Plans will be showed for the customer of given SIM based on talk-time, data package and top-up.

4. The customer should make his payment giving his service provider account details of his mobile number.

**AD-ONS**:

#Cash-back

**Table of contents**

|  |  |
| --- | --- |
| **Topic** | **Page Number** |
| **Introduction** | **6** |
| **Technology** | **10** |
| **Proposed Work** | **11** |
| **Results** | **52** |
| **Additional Knowledge Acquired** | **55** |
| **Conclusions and Future work** | **56** |
| **References** | **57** |

**INTRODUCTION**

The proposed project for recharging mobiles developed to automate the mobile recharging process. Because your talk time will not exhaust by the end of the month, you can make use for those minutes that you have paid already. In this kind of services, budget of the talk time is under control and can make changes as how to invest on the mobile phone.

Our project allows only registered users to recharge their mobiles. The project includes the features:

1. Recharge on data package, talktime (ex: 1.5GB/D, 2GB/D)

2. Top-up

3. Cashback

Register users have to register with an account if they are new users and continue with the login option, the following are the facilities provided for the registered user.

1. **Register**
2. **Login**
3. **Search Plan and Select**
4. **Check Status**
5. **Change Password**
6. **Unregister**

Service Provider updates and maintains the records of the customers.

An account is created on the mobile number of the customer. The Login ID will be the mobile number itself and user should login with ID to recharge hos mobile.

According to the recharge plan date , the expiry date will be calculated and if two or more plans are recharged on the same date then the days of plans will be carried forward.

**1.5GB/Day**

PLAN VALIDITY BENEFITS

-------------------------------------------------------------------

Rs.2121 336days 1) 1.5GB/D

2) Unlimited Calls

3) 100SMS/D

4) Jio Apps: 336days

-------------------------------------------------------------------

Rs.777 84days 1) 1.5GB/D

2) Unlimited Calls

3) 100SMS/D

4) Jio Apps: 84days

5) Hotstar: 1 year

--------------------------------------------------------------------

Rs.555 84days 1) 1.5GB/D

2) Unlimited Calls

3) 100SMS/D

4) Jio Apps: 84days

--------------------------------------------------------------------

Rs.399 28days 1) 1.5GB/D

2) Unlimited Calls

3) 100SMS/D

4) Jio Apps: 56days

---------------------------------------------------------------------

Rs.199 28days 1) 1.5GB/D

2) Unlimited Calls

3) 50SMS/D

4) Jio Apps: 28days

**2BG/Day**

PLAN VALIDITY BENEFITS

-------------------------------------------------------------------

Rs.2399 365days 1) 2GB/D

2) Unlimited Calls

3) 100SMS/D

4) Jio Apps: 365days

-------------------------------------------------------------------

Rs.599 84days 1) 2GB/D

2) Unlimited Calls

3) 100SMS/D

4) Jio Apps: 84days

--------------------------------------------------------------------

Rs.598 56days 1) 2GB/D

2) Unlimited Calls

3) 100SMS/D

4) Jio Apps: 56days

5) Hotstar: 1year

--------------------------------------------------------------------

Rs.249 28days 1) 2GB/D

2) Unlimited Calls

3) 100SMS/D

4) Jio Apps: 28days

**Top-Up**

PLAN VALIDITY BENEFITS

---------------------------------------------------------------

Rs.10 Unlimited Rs.7.47 Talktime

---------------------------------------------------------------

Rs.20 Unlimited Rs.14.95 Talktime

---------------------------------------------------------------

Rs.50 Unlimited Rs.39.37 Talktime

---------------------------------------------------------------

Rs.100 Unlimited Rs.81.75 Talktime

---------------------------------------------------------------

Rs.500 Unlimited Rs.420.73 Talktime

**#Cashback** will be given on every 5 plans carried out randomnly between Rs.50 to Rs.100/-

**TECHNOLOGY:**

All compute software needs certain hardware components or other software components resources to be present. In order for computesr to used efficiently these are the primary requestisites. There are two different categories of this section.

1. **Software Requirements:**

Software requirements mainly share out with defining the software resource requirements that need to be installed on a computer to provide optimal functioning of a particular application. Therse conditions are not included in the software installation package and need to be installed separately.

In order to use Mobile Recharge System the following are the prerequisties,

1. Operating System – Windows 7
2. C Compiler – GNU Compiler
3. Editor – Any basic editor is preferable (Example: Visual Studio)
4. **Hardware Requirements:**

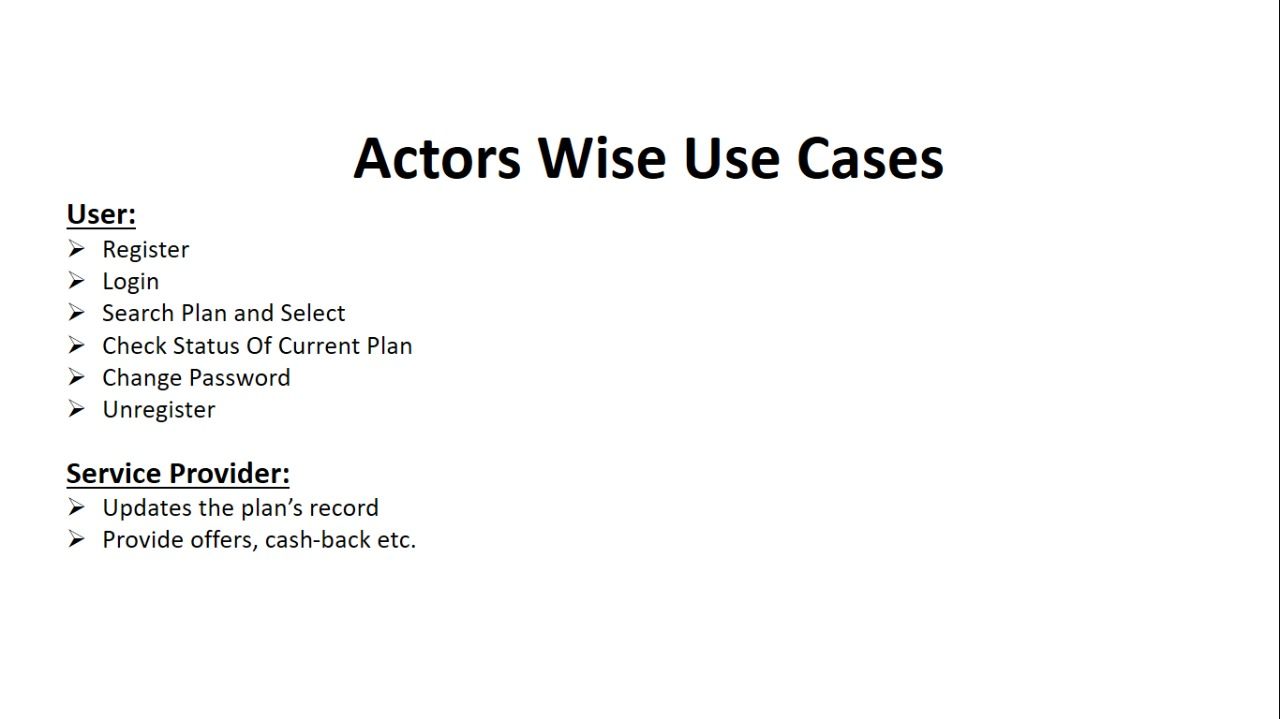
Hardware requirements refer to the common set requirements defined by any operating system or software application and are usually the physical computer resources. In this section we basically deal with primary memory, secondary memory, processing power.

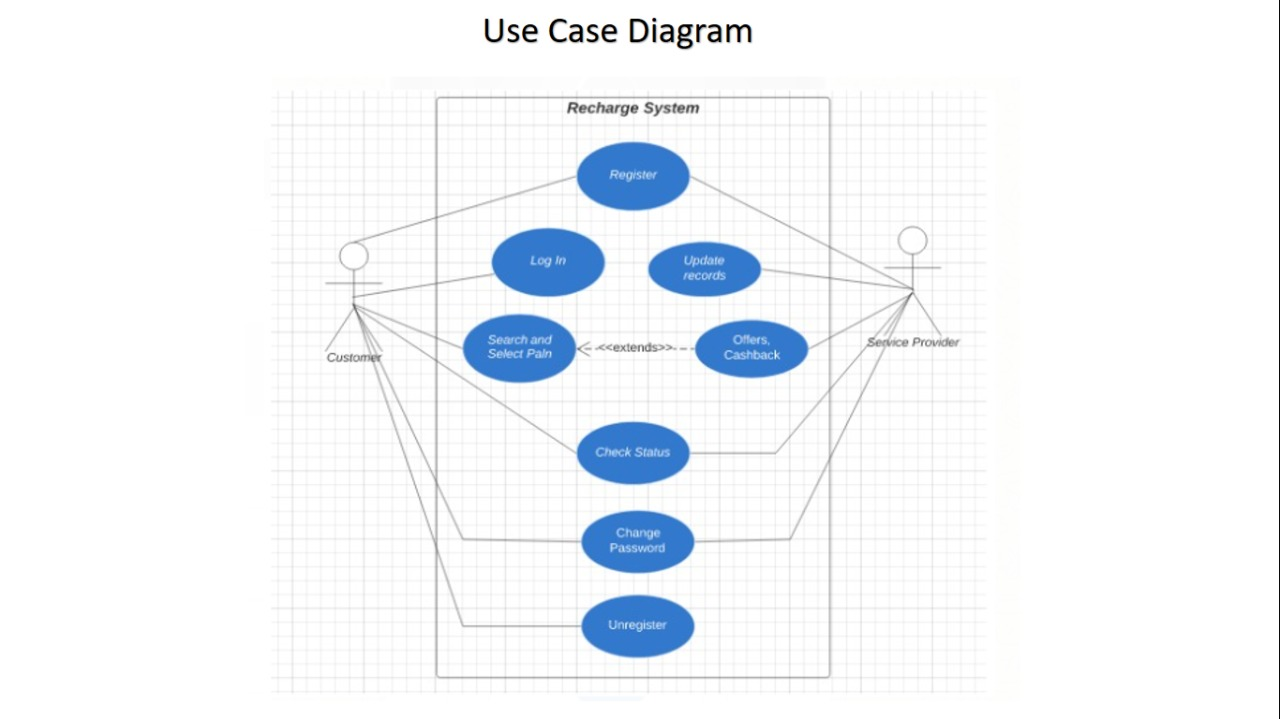
In order to use Mobile Recharge System one should have the following,

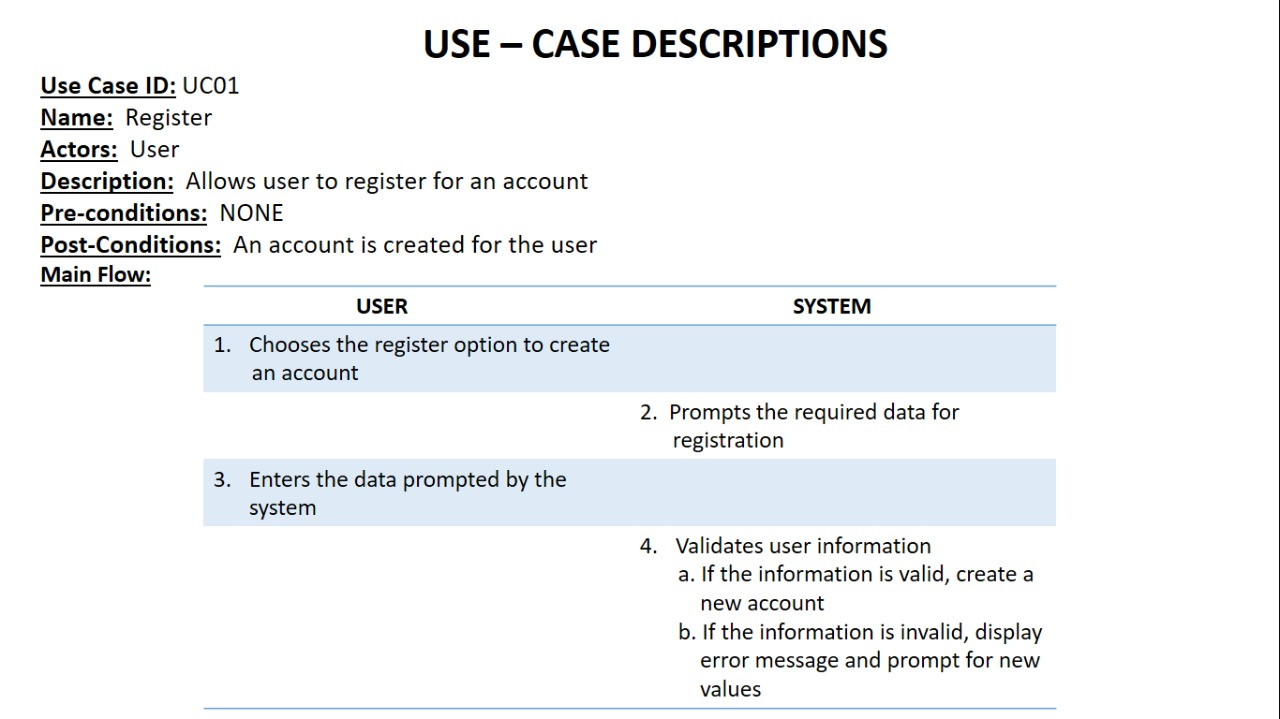
1. Processor - Intel Core i3 and above
2. Memory – 4GB Ram

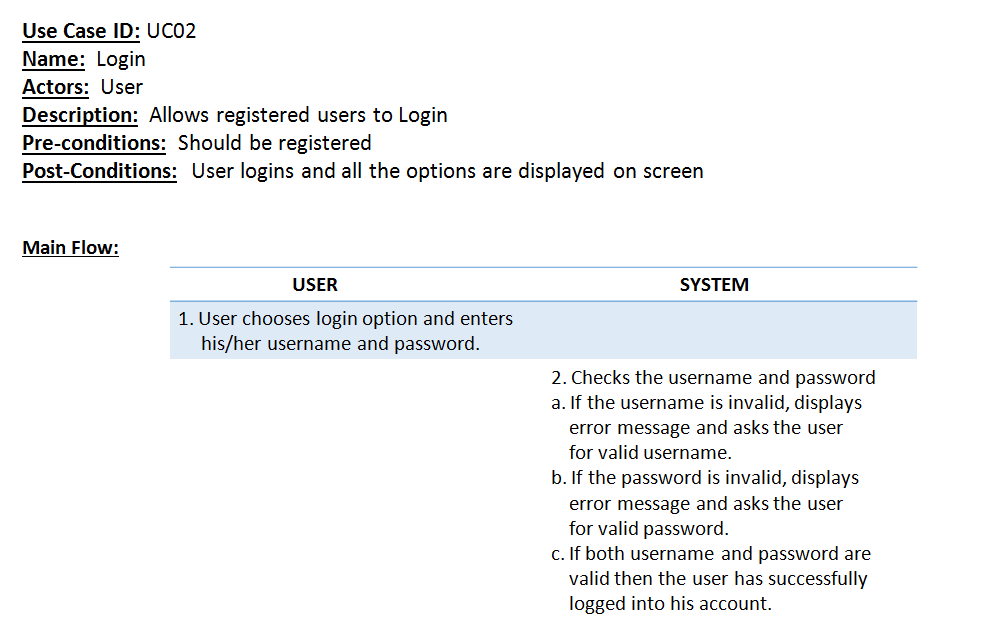
**PROPOSED WORK**

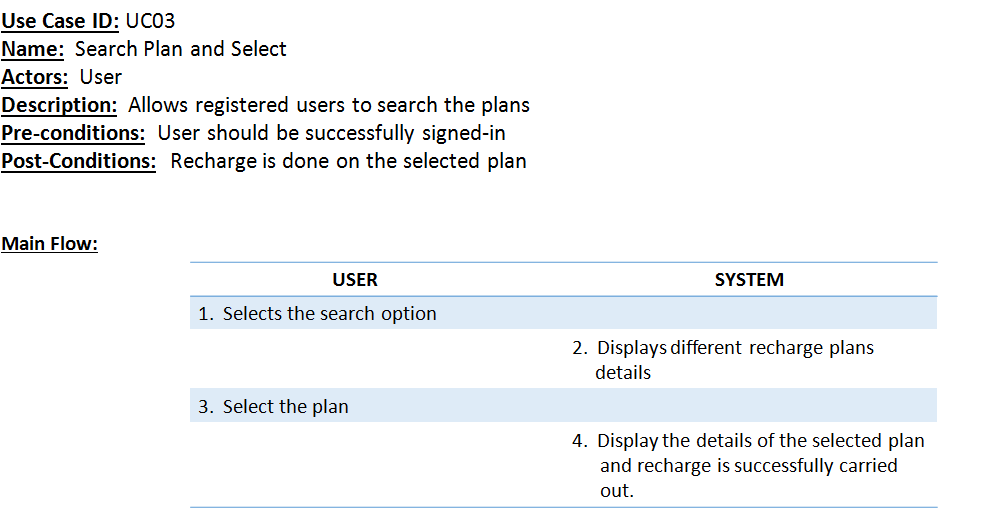
1. **Design**
2. **User Case Diagram and descriptions for all the use cases**

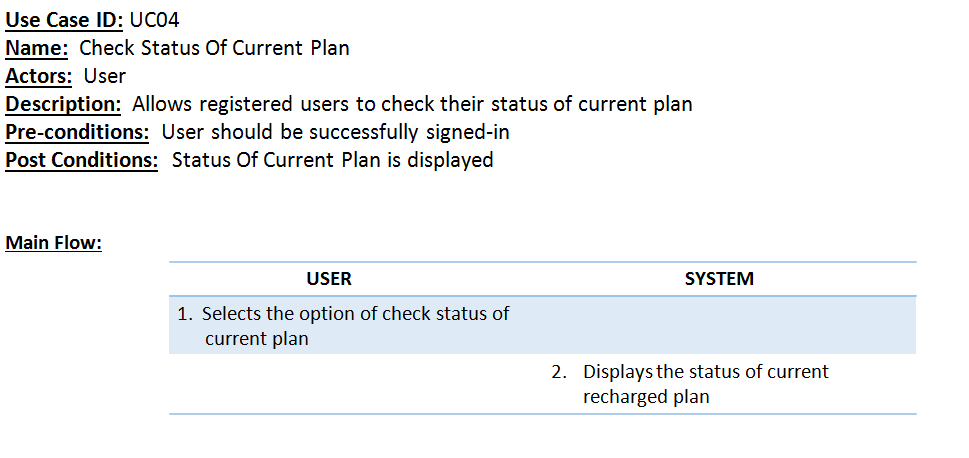
****

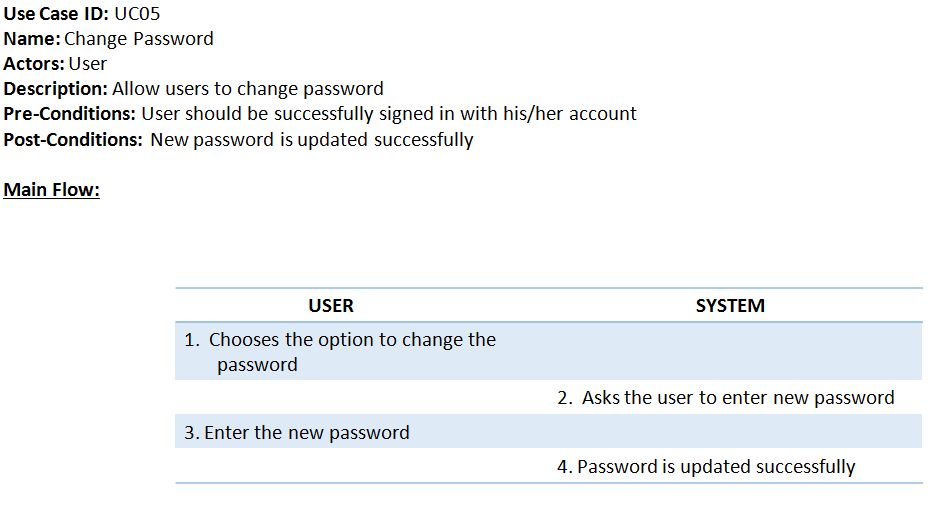
****

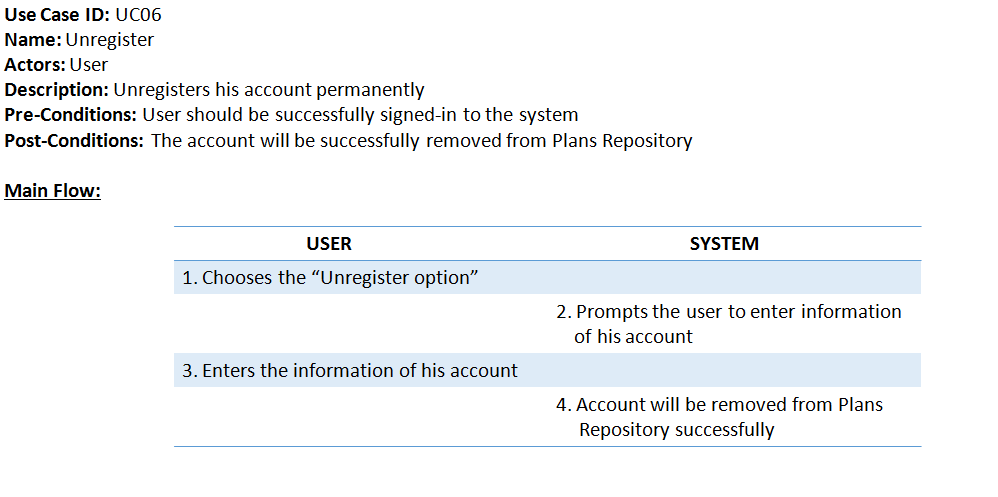
****

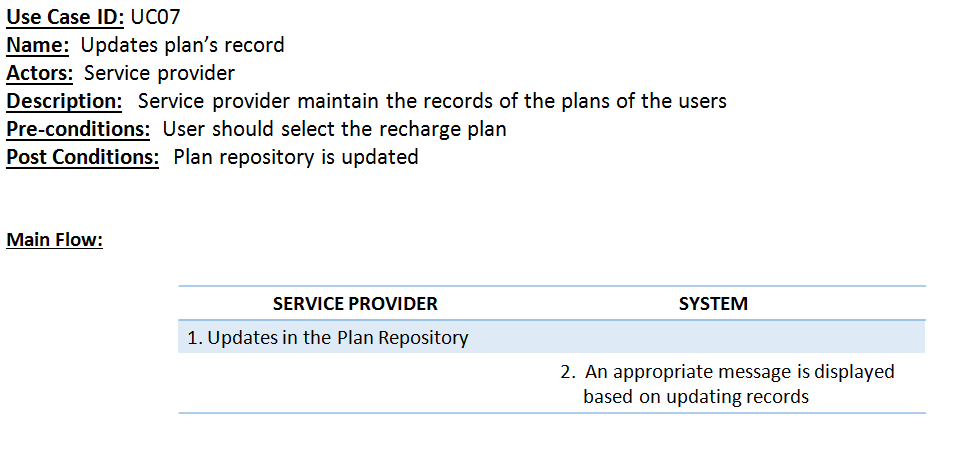
****

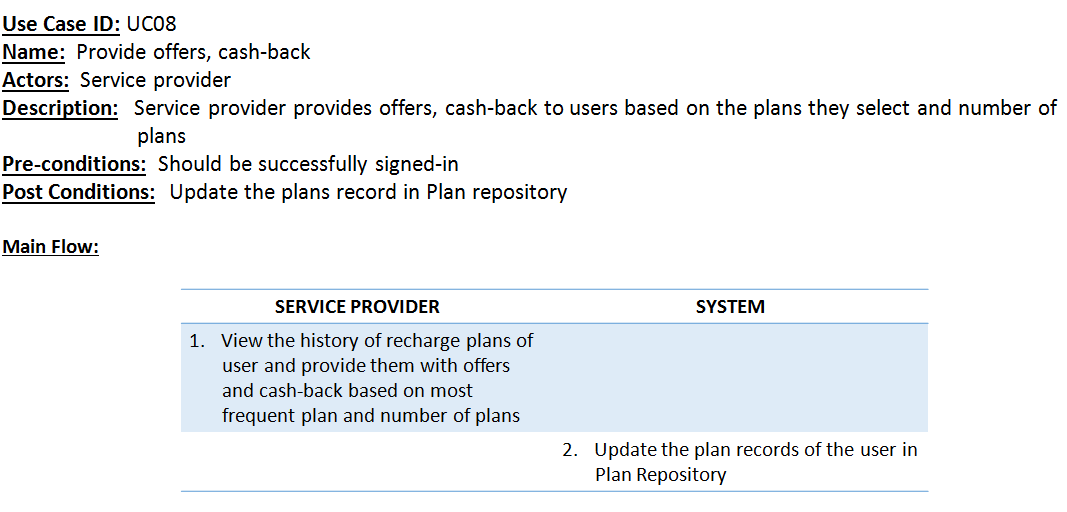
****

****

****

****





**b . IMPLEMENTATION**

**Name of C-File** : Main.c

#include <stdio.h>

#include <windows.h>

#include <conio.h>

**//Including Main.h for the function prototypes**

#include "Main.h"

COORD coord = {0, 0};

void Login(int a, int b);

void CreateAccount(void);

int main() {

system("COLOR F1");

int choice, flag;

flag = 0;

while (!flag) {

system("cls");

gotoxy(60, 10);

printf("MOBILE RECHARGE SYSTEM\n");

Sleep(2000);

gotoxy(65, 15);

printf("1. REGISTER");

gotoxy(65, 20);

printf("2. LOGIN");

gotoxy(65, 25);

printf("3. EXIT");

gotoxy(55, 30);

printf("Enter your choice: ");

scanf("%d", &choice);

if (choice == 1) {

CreateAccount();

}

else if (choice == 2) {

Login(0, 0);

}

else if (choice == 3) {

flag = 1;

}

}

system("cls");

return 0;

}

**//Pre-Defined Function Implementation**

void gotoxy(short x, short y) {

coord.X = x;

coord.Y = y;

SetConsoleCursorPosition(GetStdHandle(STD\_OUTPUT\_HANDLE), coord);

}

**//Function Implementation**

void rectangularBox(int d1, int d2, int d3, int d4) {

int i, j;

**//For Horizantal Lines**

for (i = d1; i < d3; i++) {

gotoxy(i, d2); printf("\xcd");

gotoxy(i, d4); printf("\xcd");

}

**//For vertical Lines**

for (j = d2 + 1; j < d4; j++) {

gotoxy(d1, j); printf("\xba");

gotoxy(d3, j); printf("\xba");

}

gotoxy(d1, d2); printf("\xc9");

gotoxy(d3, d2); printf("\xbb");

gotoxy(d1, d4); printf("\xc8");

gotoxy(d3, d4); printf("\xbc");

}

**Name of C-File** : CreateAccount.c

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <conio.h>

#include <time.h>

#include <windows.h>

#define ENTER 13

**//Using Main.h for gotoxy() and Rectangular box functions**

#include "Main.h"

struct Account

{

char FirstName[50];

char LastName[50];

char password[30];

char PN[11];

};

**//Functions Prototype**

int checkMobileNumber(char mobileNumber[30]);

int validateMobileNumber(char mobileNumber[30]);

void CreateAccount(void)

{

int r1, r2;

struct Account\* AC = (struct Account\*)malloc(sizeof(struct Account));

system("cls");

gotoxy(60, 8);

printf(" Account Create\n");

gotoxy(60, 10);

printf("Enter FisrtName: ");

scanf("%s",AC->FirstName);

gotoxy(60, 12);

printf("Enter LastName: ");

scanf("%s",AC->LastName);

gotoxy(60, 14);

printf("Enter PhoneNumber: ");

scanf("%s", AC->PN);

r2 = validateMobileNumber(AC->PN);

if (r2 == 0) {

gotoxy(60, 20);

printf("Invalid Phone number\n");

Sleep(3000);

return;

}

r1 = checkMobileNumber(AC->PN);

if (r1 == 0) {

return;

}

gotoxy(60, 16);

printf("Enter password: ");

int j = 0, l = 0, result = 0, res = 1;

char ch, check[20];

while (1) {

ch = getch();

if (ch == ENTER)

break;

AC->password[j] = ch;

putchar('\*');

j++;

}

AC->password[j] = '\0';

do {

j = 0;

gotoxy(60, 18 + l);

printf("Enter to re-confirm password: ");

while (1) {

ch = getch();

if (ch == ENTER)

break;

check[j] = ch;

putchar('\*');

j++;

}

check[j] = '\0';

result = strcmp(AC->password, check);

if (result != 0) {

l += 2;

gotoxy(60, 18 + l);

printf("Password didn't match");

l += 2;

res = 0;

}

}while (result != 0);

char number[20];

strcpy(number, AC->PN);

strcat(number, ".txt");

if (res == 0) {

gotoxy(50, 27 + l);

printf("Account has created SUCCESSFULLY with user name: %s\n", AC->PN);

}

else {

gotoxy(50, 27 + l);

printf("Account has created SUCCESSFULLY with user name: %s\n", AC->PN);

}

Sleep(5000);

FILE \*fp1, \*fp2;

fp1 = fopen(number, "w");

fp2 = fopen("JioCustomers.csv", "a");

if (fp1 == NULL) {

printf("Unable to access the file\n");

return;

}

if (fp2 == NULL) {

printf("Unable to access the file\n");

return;

}

fprintf(fp1, "Name of the customer: %s %s\n", AC->FirstName, AC->LastName);

fprintf(fp1, "Phone number of the customer: %s\n", AC->PN);

int plans = 0;

time\_t currentTime;

time(&currentTime); //Obtains Current Time

struct tm \*localTime = localtime(&currentTime);

int date, month, year, ad;

date = localTime->tm\_mday;

month = localTime->tm\_mon + 1;

year = localTime->tm\_year + 1900;

fprintf(fp2, "%s,%s,%d,%d/%d/%d,\n", AC->PN, AC->password, plans, date, month, year);

fclose(fp1);

fclose(fp2);

}

**//Functions Implementation**

int checkMobileNumber(char mobileNumber[30]) {

FILE \*fp;

fp = fopen("JioCustomers.csv", "r");

if (fp == NULL) {

return 1;

}

char totalLine[100];

while (fgets(totalLine, sizeof(totalLine), fp)) {

char\* t;

t = strtok(totalLine, ",");

if ((strcmp(t, mobileNumber)) == 0) {

fclose(fp);

gotoxy(55, 20);

printf("Account already exists on given phone number: %s\n", mobileNumber);

Sleep(3000);

return 0;

}

}

fclose(fp);

return 1;

}

int validateMobileNumber(char mobileNumber[30]) {

int l = 0;

if (strlen(mobileNumber) != 10) {

return 0;

}

if ((mobileNumber[0] == '9') || (mobileNumber[0] == '8')) {

for (int i = 0; i < 10; i++) {

if ((mobileNumber[i] >= 48) && (mobileNumber[i] <= 57)) {

l++;

}

}

if (l == 10) {

return 1;

}

else {

return 0;

}

}

else {

return 0;

}

}

**Name of C-File** : Login.c

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <conio.h>

#include <math.h>

#include <time.h>

#include <windows.h>

#include "Main.h"

#define ENTER 13

struct Account {

char userName[20];

char password[20];

int numberOfPlans;

int date, month, year;

};

struct Date {

int date;

int month;

int year;

};

//Global Variables

int WrongUserName, WrongPassword;

int days\_1GB[] = {336, 84, 84, 28, 28};

int days\_2GB[] = {365, 84, 56, 28};

int numberOfDaysInMonth[12] = { 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };

int m2, d2;

int remDays;

struct Account acc[100];

//Functions Prototype

int totalUsers();

void start(int users, char\* mobileNumber);

void checkStatus(int users, char\* mobileNumber);

void unregister(int users, char\* mobileNumber);

void changePassword(int users, char\* mobileNumber);

void plans(char\* mobileNumber);

void Plan1(char\* fileName, char\* mobileNumber, int choice);

void Topup(char\* fileName, char\* mobileNumber);

void update(char\* mobileNumber, int plan, int choice);

void cashBack(char\* mobileNumber, int plans);

int isLeap(int y);

int offsetDays(int d, int m, int y);

void revoffsetDays(int offset, int y);

struct Date addDays(int d1, int m1, int y1, int x);

int countLeapYears(struct Date date);

int getDifference(struct Date date1, struct Date date2);

void Login(int a, int b) {

WrongUserName = a;

WrongPassword = b;

int day, month, year;

int hours, minutes, seconds;

system("cls");

char mobileNumber[20];

char password[20];

char ch;

int r1, r2, i, j;

int users = totalUsers();

if (users != 0) {

FILE \*fp;

fp = fopen("JioCustomers.csv", "r");

char line[100];

i = 0;

while (fgets(line, sizeof(line), fp)) {

char\* s;

s = strtok(line, ",");

strcpy(acc[i].userName, s);

s = strtok(NULL, ",");

strcpy(acc[i].password, s);

s = strtok(NULL, ",");

int value;

sscanf(s, "%d", &value);

Sleep(3000);

acc[i].numberOfPlans = value;

s = strtok(NULL, ",");

char\* tk;

tk = strtok(s, "/");

sscanf(tk, "%d", &acc[i].date);

tk = strtok(NULL, "/");

sscanf(tk, "%d", &acc[i].month);

tk = strtok(NULL, "/");

sscanf(tk, "%d", &acc[i].year);

i += 1;

}

fclose(fp);

}

gotoxy(60, 8);

printf("INFORMATION TO LOGIN INTO ACCOUNT");

//Implementation of checking username

gotoxy(65, 10);

printf(" USER NAME");

gotoxy(65, 12);

rectangularBox(65, 12, 80, 14);

gotoxy(66, 13);

scanf("%s", mobileNumber);

int k = 0;

for (i = 0; i < users; i++) {

r1 = strcmp(acc[i].userName, mobileNumber);

if (r1 == 0) {

break;

}

}

if (r1 != 0) {

WrongUserName += 1;

if (WrongUserName > 2) {

gotoxy(55, 30);

printf("You have entered wrong user name more than 2 times");

Sleep(3000);

return;

}

gotoxy(65, 17);

printf("Invalid Username");

Sleep(2000);

Login(WrongUserName, WrongPassword);

}

if (r1 == 0) {

gotoxy(65, 20);

printf(" PASSWORD");

gotoxy(65, 22);

rectangularBox(65, 22, 80, 24);

gotoxy(66, 23);

j = 0;

while (1) {

ch = getch();

if (ch == ENTER)

break;

password[j] = ch;

putchar('\*');

j++;

}

password[j] = '\0';

for (i = 0; i < users; i++) {

r2 = strcmp(acc[i].password, password);

if (r2 == 0) {

gotoxy(55, 30);

printf(" Login Successfull\n");

Sleep(3000);

start(users, mobileNumber);

break;

}

}

if (r2 != 0) {

WrongPassword += 1;

if (WrongPassword > 2) {

gotoxy(55, 30);

printf("You have entered wrong password name more than 2 times");

Sleep(3000);

return;

}

gotoxy(65, 25);

printf("Invalid Password");

Sleep(2000);

Login(WrongUserName, WrongPassword);

}

}

}

//Functions Implementation

void start(int users, char\* mobileNumber) {

int choice = 1;

while (choice) {

system("cls");

int option;

rectangularBox(60, 10, 92, 35);

gotoxy(67, 13);

printf("1. Recharge Plans");

gotoxy(67, 16);

printf("2. Check Status");

gotoxy(67, 19);

printf("3. Change Password");

gotoxy(67, 22);

printf("4. Unregister");

gotoxy(67, 25);

printf("5. Go Back");

gotoxy(62, 30);

printf(" Choose your option: ");

scanf("%d", &option);

switch(option) {

case 1:

plans(mobileNumber);

break;

case 2:

checkStatus(users, mobileNumber);

break;

case 3:

changePassword(users, mobileNumber);

break;

case 4:

unregister(users, mobileNumber);

choice = 0;

break;

case 5:

choice = 0;

break;

default:

gotoxy(62, 30);

printf("\nInvalid option entered\n");

Sleep(3000);

break;

}

}

}

int totalUsers() {

FILE \*fp;

fp = fopen("JioCustomers.csv", "r");

if (fp == NULL) {

return 0;

}

int count = 0;

char line[101];

while (fgets(line, sizeof(line), fp)) {

count++;

}

fclose(fp);

return count;

}

void checkStatus(int users, char\* mobileNumber) {

system("cls");

int d, m, y;

int i;

for (i = 0; i < users; i++) {

if ((strcmp(acc[i].userName, mobileNumber)) == 0) {

d = acc[i].date;

m = acc[i].month;

y = acc[i].year;

break;

}

}

time\_t currentTime;

time(&currentTime); //Obtains Current Time

struct tm \*localTime = localtime(&currentTime);

int date, month, year, ad;

date = localTime->tm\_mday;

month = localTime->tm\_mon + 1;

year = localTime->tm\_year + 1900;

struct Date d1 = {date, month, year};

struct Date d2 = {d, m, y};

remDays = getDifference(d1, d2);

if (remDays > 0) {

gotoxy(60, 20);

printf("Mobile Number: %s",mobileNumber);

gotoxy(60, 22);

printf("There are %d days for the expiry of plan", remDays);

Sleep(3000);

}

else if (remDays == 0) {

gotoxy(60, 20);

printf("Mobile Number: %s",mobileNumber);

gotoxy(60, 22);

if (acc[i].numberOfPlans != 0)

printf("Your plan has been expired today");

else

printf("NO PLANS");

Sleep(3000);

}

else if (remDays < 0) {

gotoxy(60, 20);

printf("Mobile Number: %s",mobileNumber);

gotoxy(60, 22);

printf("Your plan has expired %d days ago", (-1 \* remDays));

Sleep(3000);

}

}

void unregister(int users, char\* mobileNumber) {

system("cls");

FILE \*fp;

int i;

char fileName[20];

strcpy(fileName, mobileNumber);

strcat(fileName, ".txt");

remove(fileName);

remove("JioCustomers.csv");

fp = fopen("JioCustomers.csv", "w");

for (i = 0; i < users; i++) {

if ((strcmp(acc[i].userName, mobileNumber)) != 0) {

fprintf(fp, "%s,%s,%d,%d/%d/%d,\n", acc[i].userName, acc[i].password, acc[i].numberOfPlans, acc[i].date, acc[i].month, acc[i].year);

}

}

fclose(fp);

gotoxy(56, 30);

printf("You account has been succesfully unregistered\n");

Sleep(5000);

}

void changePassword(int users, char\* mobileNumber) {

FILE \*fp;

int i, j = 0;

FILE \*fp1;

fp1 = fopen("JioCustomers.csv", "r");

char totalLine[100];

while (fgets(totalLine, sizeof(totalLine), fp1)) {

char\* token;

token = strtok(totalLine, ",");

int r = strcmp(mobileNumber, token);

if (r == 0) {

token = strtok(NULL, ",");

token = strtok(NULL, ",");

token = strtok(NULL, ",");

char\* tk;

tk = strtok(token, "/");

sscanf(tk, "%d", &acc[j].date);

tk = strtok(NULL, "/");

sscanf(tk, "%d", &acc[j].month);

tk = strtok(NULL, "/");

sscanf(tk, "%d", &acc[j].year);

}

j += 1;

}

fclose(fp1);

remove("JioCustomers.csv");

fp = fopen("JioCustomers.csv", "w");

for (i = 0; i < users; i++) {

if ((strcmp(acc[i].userName, mobileNumber)) != 0) {

fprintf(fp, "%s,%s,%d,%d/%d/%d,\n", acc[i].userName, acc[i].password, acc[i].numberOfPlans, acc[i].date, acc[i].month, acc[i].year);

}

else {

system("cls");

char pass[20], check[20];

char ch;

int j = 0, l = 0, result = 0, res = 1;

gotoxy(60, 16);

printf("Enter new password: ");

while (1) {

ch = getch();

if (ch == ENTER)

break;

pass[j] = ch;

putchar('\*');

j++;

}

pass[j] = '\0';

do {

j = 0;

gotoxy(60, 18 + l);

printf("Enter to re-confirm password: ");

while (1) {

ch = getch();

if (ch == ENTER)

break;

check[j] = ch;

putchar('\*');

j++;

}

check[j] = '\0';

result = strcmp(pass, check);

if (result != 0) {

l += 2;

gotoxy(60, 18 + l);

printf("Password didn't match");

l += 2;

res = 0;

}

else {

gotoxy(60, 27);

printf("Password has been updated\n");

Sleep(5000);

strcpy(acc[i].password, pass);

fprintf(fp, "%s,%s,%d,%d/%d/%d,\n", acc[i].userName, acc[i].password, acc[i].numberOfPlans, acc[i].date, acc[i].month, acc[i].year);

}

}while (result != 0);

}

}

fclose(fp);

}

void plans(char\* mobileNumber) {

system("cls");

int choice;

int finish = 0;

do {

gotoxy(65, 15);

printf("1>> 1.5GB per day");

gotoxy(65, 18);

printf("2>> 2GB per day");

gotoxy(65, 21);

printf("3>> TopUp plans");

gotoxy(65, 24);

printf("4>> GoBack");

gotoxy(61, 28);

printf(" Enter your choice: ");

scanf("%d", &choice);

switch(choice) {

case 1:

Plan1("1.5GB.txt", mobileNumber, choice);

finish = 1;

break;

case 2:

Plan1("2GB.txt", mobileNumber, choice);

break;

case 3:

Topup("Topup.txt", mobileNumber);

break;

case 4:

finish = 1;

break;

default:

printf("\nInvalid option entered\n");

break;

}

}while (finish != 1);

}

void Plan1(char\* fileName, char\* mobileNumber, int choice) {

system("cls");

FILE \*fp;

fp = fopen(fileName, "r");

char totalLine[300];

while (fgets(totalLine, sizeof(totalLine), fp)) {

puts(totalLine);

}

fclose(fp);

printf("\n\n");

int plan;

printf("Enter your plan: ");

scanf("%d", &plan);

char fname[20];

strcpy(fname, mobileNumber);

strcat(fname, ".txt");

FILE \*fp1, \*fp2;

fp1 = fopen(fileName, "r");

fp2 = fopen(fname, "a");

fprintf(fp2, "-------------------------------------------------------------------\n");

int count = 0;

while (fgets(totalLine, sizeof(totalLine), fp1)) {

count += 1;

if (plan == 1) {

if ((count >= 3) && (count <= 7)) {

fprintf(fp2, "%s", totalLine);

}

}

else if (plan == 2) {

if ((count >= 9) && (count <= 13)) {

fprintf(fp2, "%s", totalLine);

}

}

else if (plan == 3) {

if ((count >= 15) && (count <= 19)) {

fprintf(fp2, "%s", totalLine);

}

}

else if (plan == 4) {

if ((count >= 21) && (count <= 25)) {

fprintf(fp2, "%s", totalLine);

}

}

else if (plan == 5) {

int r;

r = strcmp(fileName, "1.5GB.txt");

if (r == 0) {

if ((count >= 27) && (count <= 31)) {

fprintf(fp2, "%s", totalLine);

}

}

else {

printf("You have entered wrong choice\n");

system("cls");

return;

}

}

else {

printf("You have entered wrong choice\n");

system("cls");

return;

}

}

fclose(fp1);

fclose(fp2);

printf("\nYour plan has been recharged succesfully\n");

Sleep(3000);

update(mobileNumber, plan, choice);

system("cls");

}

void Topup(char\* fileName, char\* mobileNumber) {

system("cls");

FILE \*fp;

fp = fopen(fileName, "r");

char totalLine[300];

while (fgets(totalLine, sizeof(totalLine), fp)) {

puts(totalLine);

}

fclose(fp);

printf("\n\n");

int topup;

printf("Enter your topup: ");

scanf("%d", &topup);

char fname[20];

strcpy(fname, mobileNumber);

strcat(fname, ".txt");

FILE \*fp1, \*fp2;

fp1 = fopen(fileName, "r");

fp2 = fopen(fname, "a");

fprintf(fp2, "-------------------------------------------------------------------\n");

int count = 0;

while (fgets(totalLine, sizeof(totalLine), fp1)) {

count += 1;

if (topup == 1) {

if (count == 3) {

fprintf(fp2, "%s", totalLine);

}

}

else if (topup == 2) {

if (count == 5) {

fprintf(fp2, "%s", totalLine);

}

}

else if (topup == 3) {

if (count == 7) {

fprintf(fp2, "%s", totalLine);

}

}

else if (topup == 4) {

if (count == 9) {

fprintf(fp2, "%s", totalLine);

}

}

else if (topup == 5) {

if (count == 11) {

fprintf(fp2, "%s", totalLine);

}

}

}

fclose(fp1);

fclose(fp2);

printf("\nYour topup has been recharged succesfully\n");

Sleep(3000);

system("cls");

}

void update(char\* mobileNumber, int plan, int choice) {

int users = totalUsers();

int i, j;

j = 0;

FILE \*fp1;

fp1 = fopen("JioCustomers.csv", "r");

char totalLine[100];

while (fgets(totalLine, sizeof(totalLine), fp1)) {

char\* token;

token = strtok(totalLine, ",");

int r = strcmp(mobileNumber, token);

if (r == 0) {

token = strtok(NULL, ",");

token = strtok(NULL, ",");

token = strtok(NULL, ",");

char\* tk;

tk = strtok(token, "/");

sscanf(tk, "%d", &acc[j].date);

tk = strtok(NULL, "/");

sscanf(tk, "%d", &acc[j].month);

tk = strtok(NULL, "/");

sscanf(tk, "%d", &acc[j].year);

}

j += 1;

}

fclose(fp1);

for (i = 0; i < users; i++) {

if ((strcmp(acc[i].userName, mobileNumber)) == 0) {

acc[i].numberOfPlans++;

break;

}

}

int ad;

if (choice == 1) {

ad = days\_1GB[plan - 1];

}

else if (choice == 2) {

ad = days\_2GB[plan - 1];

}

struct Date dt1;

if (remDays < 0) {

dt1 = addDays(acc[i].date, acc[i].month, acc[i].year, (ad + (-1 \* remDays)));

}

else {

dt1 = addDays(acc[i].date, acc[i].month, acc[i].year, ad);

}

FILE \*fp;

fp = fopen("JioCustomers.csv", "w");

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

cashBack(mobileNumber, acc[i].numberOfPlans);

if ((strcmp(acc[i].userName, mobileNumber)) == 0) {

acc[i].date = dt1.date;

acc[i].month = dt1.month;

acc[i].year = dt1.year;

fprintf(fp, "%s,%s,%d,%d/%d/%d,\n", acc[i].userName, acc[i].password, acc[i].numberOfPlans, dt1.date, dt1.month, dt1.year);

}

else if (acc[i].numberOfPlans == 0) {

time\_t currentTime;

time(&currentTime); //Obtains Current Time

struct tm \*localTime = localtime(&currentTime);

int date, month, year, ad;

date = localTime->tm\_mday;

month = localTime->tm\_mon + 1;

year = localTime->tm\_year + 1900;

acc[i].date = date;

acc[i].month = month;

acc[i].year = year;

fprintf(fp, "%s,%s,%d,%d/%d/%d,\n", acc[i].userName, acc[i].password, acc[i].numberOfPlans, date, month, year);

}

else {

fprintf(fp, "%s,%s,%d,%d/%d/%d,\n", acc[i].userName, acc[i].password, acc[i].numberOfPlans, acc[i].date, acc[i].month, acc[i].year);

}

}

fclose(fp);

}

void cashBack(char\* mobileNumber, int plans) {

int min = 50, max = 100;

int range;

range = (max - min) + 1;

int cashback = (rand() % range) + min;

FILE \*fp;

char fname[50];

strcpy(fname, mobileNumber);

strcat(fname, ".txt");

fp = fopen(fname, "a");

if (fp != NULL) {

if ((plans % 5) == 0) {

fprintf(fp, "Cash Back: Rs. %d\n", cashback);

}

}

fclose(fp);

}

int isLeap(int y)

{

if (y % 100 != 0 && y % 4 == 0 || y % 400 == 0)

return 1;

return 0;

}

int offsetDays(int d, int m, int y)

{

int offset = d;

if(m - 1 == 11)

offset += 335;

if(m - 1 == 10)

offset += 304;

if(m - 1 == 9)

offset += 273;

if(m - 1 == 8)

offset += 243;

if(m - 1 == 7)

offset += 212;

if(m - 1 == 6)

offset += 181;

if(m - 1 == 5)

offset += 151;

if(m - 1 == 4)

offset += 120;

if(m - 1 == 3)

offset += 90;

if(m - 1 == 2)

offset += 59;

if(m - 1 == 1)

offset += 31;

if (isLeap(y) && m > 2)

offset += 1;

return offset;

}

void revoffsetDays(int offset, int y)

{

int month[] = { 0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };

if (isLeap(y))

month[2] = 29;

int i;

for (i = 1; i <= 12; i++)

{

if (offset <= month[i])

break;

offset = offset - month[i];

}

d2 = offset;

m2 = i;

}

struct Date addDays(int d1, int m1, int y1, int x)

{

int offset1 = offsetDays(d1, m1, y1);

int remDays = isLeap(y1) ? (366 - offset1) : (365 - offset1);

int y2, offset2 = 0;

if (x <= remDays)

{

y2 = y1;

offset2 =offset1 + x;

}

else

{

x -= remDays;

y2 = y1 + 1;

int y2days = isLeap(y2) ? 366 : 365;

while (x >= y2days)

{

x -= y2days;

y2++;

y2days = isLeap(y2) ? 366 : 365;

}

offset2 = x;

}

revoffsetDays(offset2, y2);

//Creating Of Date variable

struct Date d;

d.date = d2;

d.month = m2;

d.year = y2;

return d;

}

int countLeapYears(struct Date d)

{

int years = d.year;

if (d.month <= 2)

years--;

return years / 4

- years / 100

+ years / 400;

}

int getDifference(struct Date dt1, struct Date dt2)

{

long int n1 = dt1.year \* 365 + dt1.date;

for (int i = 0; i < dt1.month - 1; i++)

n1 += numberOfDaysInMonth[i];

n1 += countLeapYears(dt1);

long int n2 = dt2.year \* 365 + dt2.date;

for (int i = 0; i < dt2.month - 1; i++)

n2 += numberOfDaysInMonth[i];

n2 += countLeapYears(dt2);

return (n2 - n1);

}

1. **Specific algorithms (or) logic**

**Writing and reading data by .csv file**

1. Define a structure for holding the values such as mobile number, firstName, lastName, password, number of plans etc

**Retreiving the following details:**

1. **Mobile Number**
2. **Passoword**
3. **Number of plans**
4. **Date of Expiry**

**Writing Into csv file**

1. In the functions:

void CreateAccount(void)

void update(char\* mobileNumber, int plan, int choice)

void changePassword(int users, char\* mobileNumber).

**GITHUB LINKS:**

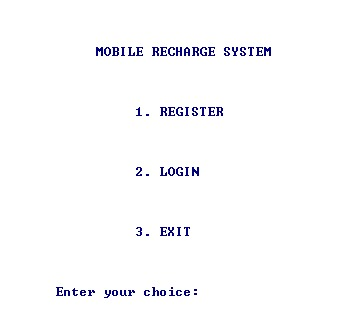
<https://github.com/Mummoju-Harshit>

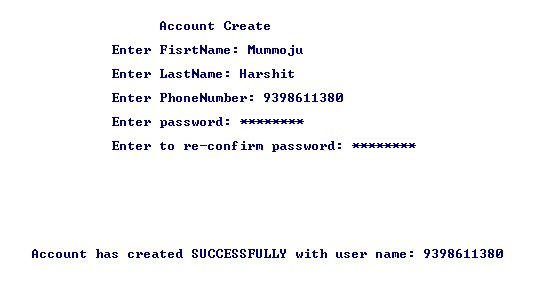
<https://github.com/SanjithReddy>

1. **Testing**

|  |  |  |  |
| --- | --- | --- | --- |
| TEST CASE TEMPLATE | | | |
| **Test case ID** : TC01 | | | Use case ID :  **UC-01** |
| **Test case title** : Register | | |
| **Test case description** : Allows new user to register to create an account | | |
| **Test steps** | **Expected result** | **Actual result** | |
| The system allows the user to enter the correct details to create an account with their user name and password. | If the user enters details in correct format then system will allow to create and account. | A message is displayed saying “Account has been successfully created with username XXX“ | |

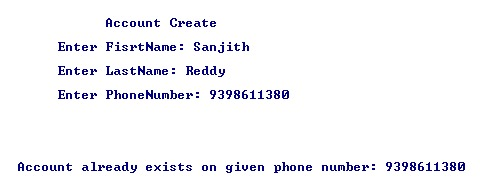
**OUTPUT SCREEN SHOTS:**





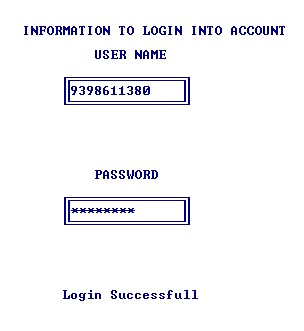
|  |  |  |  |
| --- | --- | --- | --- |
| TEST CASE TEMPLATE | | | |
| **Test case ID** : TC02 | | | Use case ID :  **UC-01** |
| **Test case title** : Register | | |
| **Test case description** : Allows new user to register to create an account | | |
| **Test steps** | **Expected result** | **Actual result** | |
| The system allows the user to enter the correct details to create an account with their user name and password. | If the entered mobile number already exists then won’t allow to create an account | A message is displayed saying user already exists. | |

**OUTPUT SCREEN SHOTS:**



|  |  |  |  |
| --- | --- | --- | --- |
| TEST CASE TEMPLATE | | | |
| **Test case ID** : TC01 | | | Use case ID :  **UC-02** |
| **Test case title** : Login | | |
| **Test case description** : Allows existing users to login | | |
| **Test steps** | **Expected result** | **Actual result** | |
| The system prompts the user to enter password and user name correctly . | If the user enters correct username and password an appropriate message is displayed. | A message is displayed saying “LOGGED IN SUCCESSFULLY “ | |

**OUTPUT SCREEN SHOTS:**



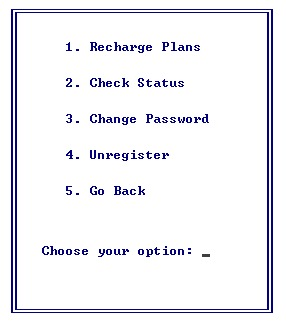
|  |  |  |  |
| --- | --- | --- | --- |
| **Test case ID** : TC02 | | | Use case ID :  **UC-02** |
| **Test case title** : Login | | |
| **Test case description** : Allows existing users to login | | |
| **Test steps** | **Expected result** | **Actual result** | |
| The system prompts the user to enter password and user name correctly . | If the username and password are invalid an appropriate message is displayed | A message is displayed saying username (or) password is invalid and also user will given only 3 chances to enter valid details | |

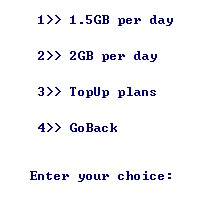
**OUTPUT SCREEN SHOTS:**

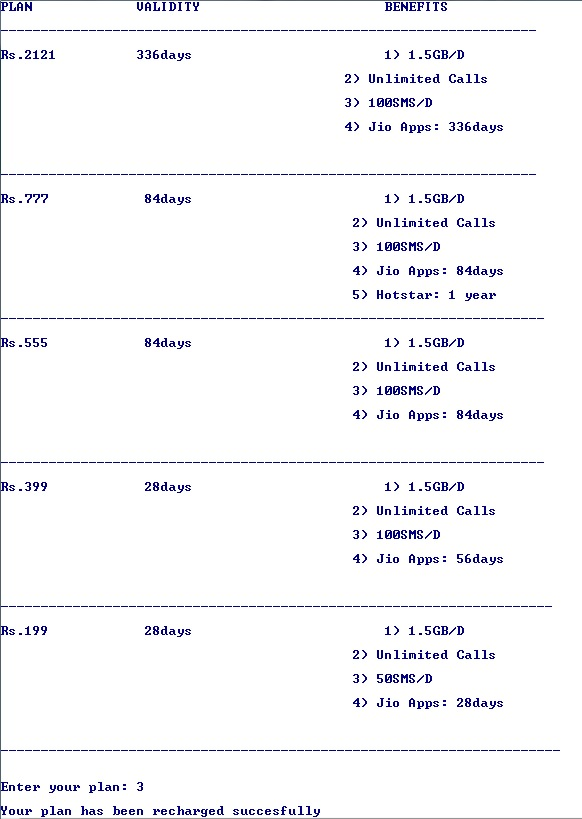
****

|  |  |  |  |
| --- | --- | --- | --- |
| **Test case ID** : TC01 | | | Use case ID :  **UC-03** |
| **Test case title** : Search Plan and Select | | |
| **Test case description** : Allows registered users to search the plans | | |
| **Test steps** | **Expected result** | **Actual result** | |
| After successful login the user is allowed to click on Recharge Plans . | Plans will be displayed based on data package and Top-up | Recharge is carried out on selected plan | |

**OUTPUT SCREEN SHOTS**







|  |  |  |  |
| --- | --- | --- | --- |
| **Test case ID** : TC01 | | | Use case ID :  **UC-04** |
| **Test case title** : Check Status | | |
| **Test case description** : Allows registered users to check their status of current plan | | |
| **Test steps** | **Expected result** | **Actual result** | |
| User is allowed to select on Check Staus. | The expiry days of the plans of the customer will be dispalyed. | A message is displayed saying how manydays are left for expiry. | |

**OUTPUT SCREEN SHOTS**



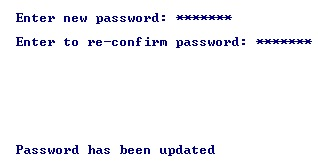
|  |  |  |  |
| --- | --- | --- | --- |
| **Test case ID** : TC02 | | | Use case ID :  **PC-04** |
| **Test case title** : Check Status | | |
| **Test case description** : Allows registered users to check their status of current plan | | |
| **Test steps** | **Expected result** | **Actual result** | |
| User is allowed to select on Check Staus | If no recharge is carried out then appropriate message will be displayed. | The message “No Plans” will be displayed. | |

**OUTPUT SCREEN SHOTS:**

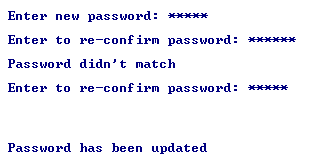


|  |  |  |  |
| --- | --- | --- | --- |
| **Test case ID** : TC01 | | | Use case ID :  **UC-05** |
| **Test case title** : Change Password | | |
| **Test case description** : **:** Allow users to change password | | |
| **Test steps** | **Expected result** | **Actual result** | |
| User is allowed to change his account password | New password will be updated. | New password entered will be re-confirmed. | |

**OUTPUT SCREEN SHOTS**

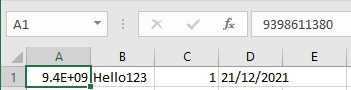


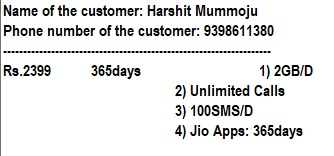
|  |  |  |  |
| --- | --- | --- | --- |
| **Test case ID** : TC02 | | | Use case ID :  **UC-05** |
| **Test case title** : Change Password | | |
| **Test case description** : **:** Allow users to change password | | |
| **Test steps** | **Expected result** | **Actual result** | |
| User is allowed to change his account password | When new password entered is incorrect then it prompts to enter again | The message “ Password didn’t match” will be displayed. | |



|  |  |  |  |
| --- | --- | --- | --- |
| **Test case ID** : TC01 | | | Use case ID :  **UC-06** |
| **Test case title** : Updates plan’s record | | |
| **Test case description** : Service provider maintain the records of the plans of the users | | |
| **Test steps** | **Expected result** | **Actual result** | |
| User should recharge a plan to update the plan details. | The plan details will be stored in csv and the account file. | The details of plans will be updated on every recharge plan. | |

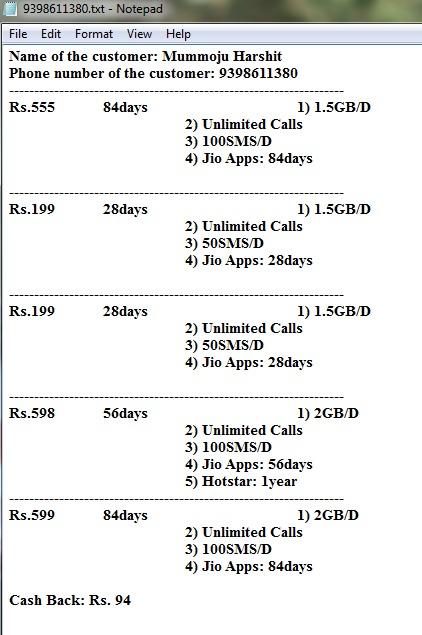
**OUTPUT SCREEN SHOTS**





|  |  |  |  |
| --- | --- | --- | --- |
| **Test case ID** : TC01 | | | Use case ID :  **PC-07** |
| **Test case title** : Cash-back | | |
| **Test case description** : Service provider provides cash-back to users based on the plans they select and number of plans. | | |
| **Test steps** | **Expected result** | **Actual result** | |
| User gets cash-back on every 5 plans. | Cah-back will be updated in the file. | Cash-back will be in between Rs.50 to Rs.100/- | |

**OUTPUT SCREEN SHOTS:**

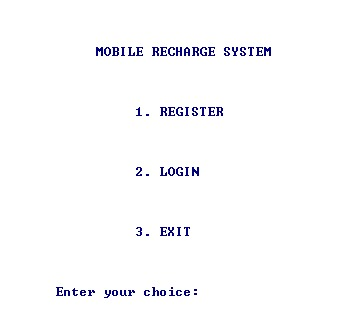


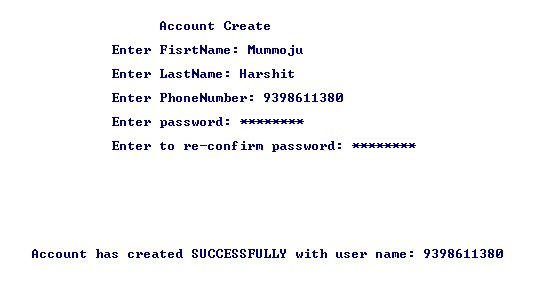
|  |  |  |  |
| --- | --- | --- | --- |
| **Test case ID** : TC01 | | | Use case ID :  **UC-08** |
| **Test case title** : Unregister | | |
| **Test case description** : Unregisters his account permanently | | |
| **Test steps** | **Expected result** | **Actual result** | |
| User is allowed to select Unregister option. | An appropriate message is displayed when he unregisteres the account. | The account will be unregistered permanently. | |

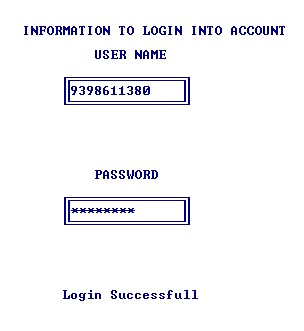
**OUTPUT SCREEN SHOTS:**

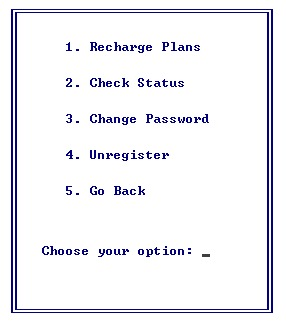


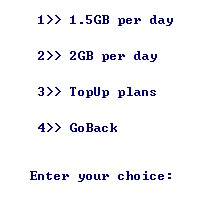
**RESULT**

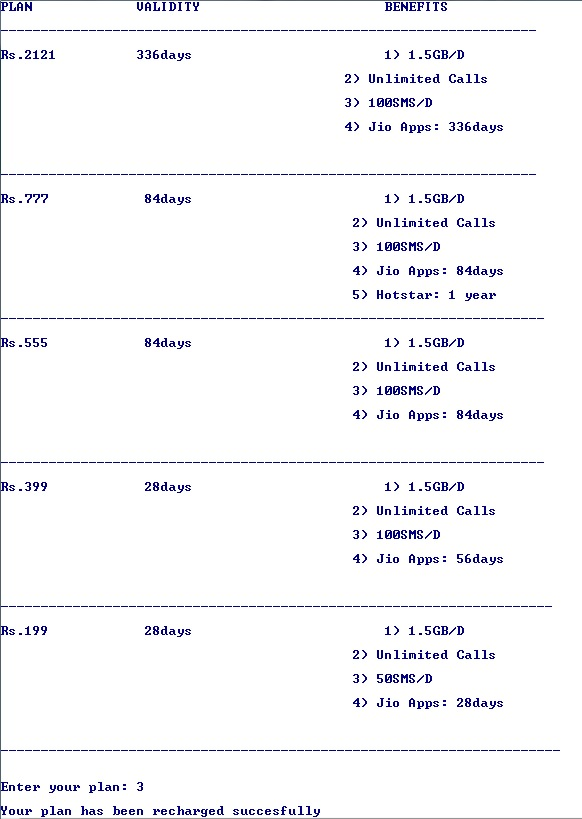












**What is the additional knowledge gained as a result of implementing this miniproject apart from the syllabus covered in the course programming for problem solving?**

Implementing this project in C Language has introduced us to different libraries such as: ‘conio.h’, ‘time.h’ and ‘windows.h’. We were able to use the knowledge we have on the Linked List Data Structure and execute it as a real-time application. We used the ‘windows.h’ library for controlling the display colours in a controlled manner. We explored the ‘time.h’ and ‘conio.h’ libraries for achieving a look-and-feel of an actual window application by constructing our own time delay function. Also, we have further improved in our knowledge in file-handling because of the vast amount of data manipulation we have done using text files.

One of the most important thing me and my team mate learnt about was reading and writing form and to **csv files**.

**Conclusion and Future work**

Therefore, conclusion of the proposed system is based on user’s need. The system is developed in considering all issues related to all user which are included in this system. Based on the result of this research, it can be concluded: It helps customer in recharging their mobile phones with selected plans easily; It gives information needed in recharging the mobile.With online mobile recharge system,the customers can easily recharge.It improves the mobile recharge system in the present life. The proposed system would attract customers because it is very user friendly and very efficient.

**References**

Reading and Writing data from and into csv file

[**https://www.geeksforgeeks.org/csv-file-management-using-c/**](https://www.geeksforgeeks.org/csv-file-management-using-c/)

Stack Overflow (for debugging errors)

**https://stackoverflow.com/**