

UNIT 1.

INTRODUCTION

UNIT 1

INTRODUCTION

1.1 INTRODUCTION

The project aims at developing a login-based Outpatient Management System using the C Programming Language that enables a hospital to maintain its records.

The software consists of two sections i.e., Login Section and Admin Section. In login section, admin have to login their account with the previously set credentials or have to register account if they haven't registered before. In the same section, if admin forget the password, there is an option to reset password.

When user get logged in, they will be promoted to Admin Section, where they get multiple options like to Add Patient, Update Patient, Search Patient, Delete Patient and Change Password too.

The software demonstrates the creation of a user interfaces of a system without the use of C Graphics Library. The application uses the basic C features to generate menus and print text on the screen. To display text according to the application requirements functions have been generated in the application. The application also implements the concepts of structures to define the Patients records. It also effectively applies the various C concepts such as FILE operations, Looping, Branching, Constants, Data and String manipulation functions.

1.2 OBJECTIVES

- To develop secure login-based Outpatient Management System.
- To provides a clean, user-friendly menu options for the user.
- To demonstrate the creation of a user interface of a system without the use of C graphics library.
- To make use of the basic C features to generate menus and display the text on the screen.
- To display text according to the application requirements, functions have been generated in the application.
- To effectively applies the various C concepts such as File Operation, Looping, String manipulation, Time functions.
- To gain good knowledge about file processing, looping, structures, functions, pointers etc.

1.3 APPLICATIONS

This is login-based application that provides secure and password protected system to the user. User can't access the record nor manipulate the record until they logged in to the account.

Options Available

1. Register/Login
2. Forget Password
3. Credits
4. Exit Program

Once User Logged In

1. Add Patient
2. Update Patient
3. Search Patient
4. Delete Patient
5. Change Password
6. Exit Program
7. Logout

1.4 FEATURES

1. Encrypted Password
2. Secure
3. Easy to use
4. Reliable and accurate
5. Clean and User-Friendly User Interface

1.5 LIMITATIONS

1. No online mode.
2. Advance validation is not done.
3. Unexpected input may crash the program.

UNIT 2.

PROJECT DEVELOPMENT LIFE
CYCLE

Unit 2:

Project Development Life Cycle

2.1 Project Planning and Feasibility Study:

Before buying any software, we have to go through the system that is beneficial for our daily activities. On the basis of the required information the project should be planned. A feasibility report is prepared to present in depth techno commercial analysis carried out on the project idea.

2.2 Requirement Analysis

This is one of the important phases of the PDLC. In this we come up with a detail report from the different field for finding the business. We need to get the entire requirement regarding the problem otherwise it creates a lot of problem in future. We need to be clear about the way to stock used to be recorded and how the owner wants those goods to be recorded in this system.

2.3 System Design

Design refers to the modules used to build in the software development. There are different modules such as E-R diagram, DFD'S Flowchart etc. These modules help in the design of the program. According to the need of the programmer such modules are used.

2.4 Methodology

We have used Water Fall Model to develop this project. This model consists of the following phases:

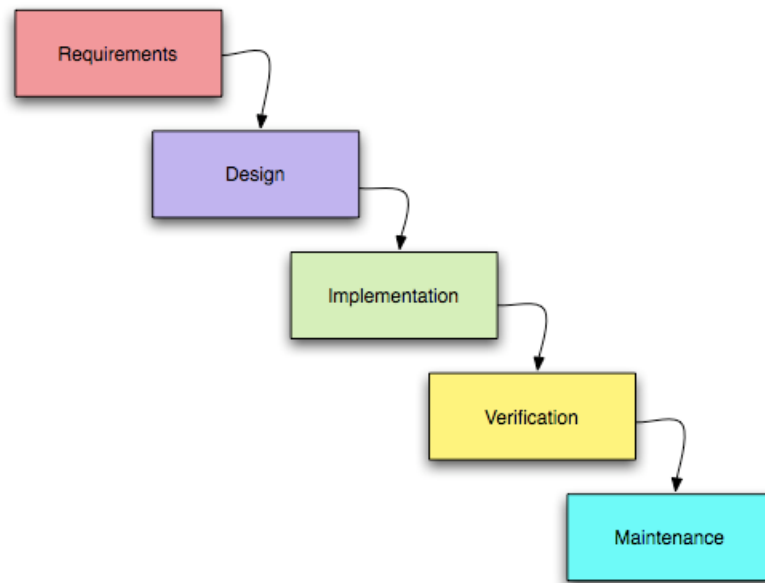


Fig no 1.0: Water Fall Model

2.5 Coding

On the basis of the system design, we did the coding of the system. The coding is done using the programming language. The technology used for coding in this project is programming language C. It will develop in window platform. C is a general language and used as general purpose. It was first used as the system language for UNIX operating system.

2.6 Debugging and Testing

We have debugged our program for finding the reducing the number of bugs or defect in a computer or a piece of electronic hardware thus making it behave as expected. The basic steps that we have used in our program are:

- Recognize that at bugs exists
- Isolate the source of the bug
- Identify the cause of the bugs
- Determine a fix for the bugs
- Apply the fix and test it.

Testing is the process of debugging of the software i.e., discovering the errors of the bugs and removing them. Actually, we have tested a program to work correctly, to discover the causes of these errors, and to revise the program code to eliminate the errors. Our tested program has a final measure of quality assurance for software product during the later phase of the system development life cycle.

Thus, as we have tested our program using various techniques (testing processes), our software is free of errors and can serve best.

2.7 Implementation

System implementation generally focuses on the coding and installing of the system. Our system implementation is composed of activities, which are coding, testing and installation. The purpose of these steps is to convert the physical system specification into the working and reliable software.

2.8 Documentation and Evaluation

We have all documents all the activity performed during the development of the system, which will be very useful in the future modifications and changes.

As per time being, if the vendor wants to make some amendments in the existing programs of his system the developer should edit as per his requirements.

2.9 Project Development Tools

A. Hardware

1. Intel Core i5
2. 256 GB SSD
3. 8 GB RAM
4. Printer and Lamination Machine

B. Software

1. Windows 10 OS
2. Microsoft Word
3. DEV C++
4. VsCode
5. OneDrive

UNIT 3.
TIME, COST AND TASK
DIVISION

Unit 3:

Time, Cost and Task Division

3.1 Time and Cost

Time, Cost and Task Division plays a vital role in the software development. So, the above factors should be properly considered while developing the software.

The estimated time duration of this application is 1 month. We should develop such type of software that could be prepared within 1 month.

SN	Activities	Price (Nrs)
1	Internet Usage	300
2	Transportation	100
3	Electricity	300
4	Designing	100
5	Testing	100
6	Miscellaneous Expenses	600
	Total	1500

Table No 1.0 Cost Estimation

3.2 Task Division

This project is prepared in a group. The Name of the members of the and their task division are given below:

SN	Name of Students	Task Performed
1	Sakar Aryal	Coding and Testing
2	Himal Aryal	Requirement Analysis, Design and Testing
3	Sandhya Khadka	Data Collection, Rough Sketching and Testing
4	Pawan Chaudhary	Feasibility Study, Preparing Report and Testing

Table No 1.1 Task Division

UNIT 4. APPENDIXES

Unit 4:

Appendixes

4.1 Gantt Chart

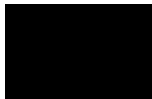
SN	Tasks	Oct 10- 15	Oct 16-20	Oct 21-31	Nov 1-5	Nov 6-8	Nov 9-11
1	Analysis						
2	Design						
3	Coding						
4	Testing						
5	Implementation						
6	Maintenance						
7	Documentation						

Chart No 1.0: Gantt Chart

Index



Less Work (Light Shade)



More Work (Dark Shade)

4.2 Algorithm

Step 1: Start

Step 2: Go to mainMenu

I. Algorithm For *mainMenu*

Step 1: Read choices

Step 2: If the choice isn't either 1 or 2 or 3 or 4 go to step 1

Step 3: If choice is 1 then go to **adminRegistration**

Step 4: If choice is 2 then, check if there is existing admin or not

If there isn't admin, then display "***You've not registered yet***". Then go to step 1, otherwise go to **forgetPassword**

Step 5: If choice is 3 then go to **credit**

Step 6: If choice is 4 then end program

II. Algorithm For *adminRegistration*

Step 1: If there is existing member then go to step 5, if not then go to step 2

Step 2: Read Username, Password, Pet's name

Step 3: Save the data

Step 4: Go to **mainMenu**

Step 5: Read Username and Password

Step 6: If Username and Password matched then go to Step 0, otherwise go to Step 7

Step 7: Give user 3 more chance. If user can't able to enter the correct Username and Password then go to **mainMenu**. Of if user entered the correct Username and Password then go to **adminPanel**.

III. Algorithm for *forgetPassword*

Step 1: Read Username and Pet's name

Step 2: If the input is matched with the record, then go to **adminRegistration**. Otherwise, give user 3 more chance. If user entered correct Username and Pet's name then go to **adminRegistration**, otherwise go to **mainMenu**

IV. Algorithm For *credit*

- Step 1: Display all the contributor's name
- Step 2: Display "Enter any key to continue"
- Step 3: Go to mainMenu

V. Algorithm For *adminPanel*

- Step 1: Read choice
- Step 2: If choice isn't 1 or 2 or 3 or 4 or 5 or 6 or 0 then end the program
- Step 3: If choice is 0 go to **mainMenu**
- Step 4: If choice is 1 go to **addRecord**
- Step 5: If choice is 2 go to **viewRecord**
- Step 6: If choice is 3 go to **editRecord**
- Step 7: If choice is 4 go to **searchRecord**
- Step 8: If choice is 5 go to **deleteRecord**
- Step 9: If choice is 6 go to **changePassword**

VI. Algorithm for *addRecord*

- Step 1: If there isn't any record present, then assign 0 to the patient ID of new patient. And if there already exist patient, then increase the value of the patient ID of previous patient and assign the value to the new patient ID of new Patient
- Step 2: Assign current date to the registration date of patient.
- Step 3: Go to **addRecordItem**
- Step 4: Save the data
- Step 5: Ask user if they want to add more data. If they want then go to Step 1 otherwise go to **adminPanel**.

VII. Algorithm for *addRecordItem*

- Step 1: Read first name, last name, sex, phone number, address, problem, depart ID, doctor fees and other fees.
- Step 2: Assign department and doctor according to the depart ID

VIII. Algorithm for *viewRecord*

Step 1: Read total number of patients from the file and display the record of all patients available in the record

IX. Algorithm for *searchRecord*

Step 1: Read patient ID

Step 2: Check if there is patient with same patient ID in the record or not. If there is patient, then display the patient data. Otherwise go to step 3

Step 3: Display an empty record

Step 4: Ask user if they want to search for again. If yes then go to Step 1 otherwise go to **adminRegistration**

X. Algorithm for *editRecord*

Step 1: Read patientID

Step 2: If patientID matched with record, then go to **addRecordItem**, otherwise display **"No record Found"** then go to **Step 4**

Step 3: Save the data

Step 4: Go to **adminPanel**

XI. Algorithm for *deleteRecord*

Step 1: Read patientID

Step 2: Read confirmID

Step 3: If both patientID and confirmID matched then go to Step 4 otherwise go to Step 5

Step 4: Delete the record and go to Step 6

Step 5: If User didn't confirm ID then Display **"You can't confirm the ID"** or display **"Record not found"** if there is no record present. Then go to step 6

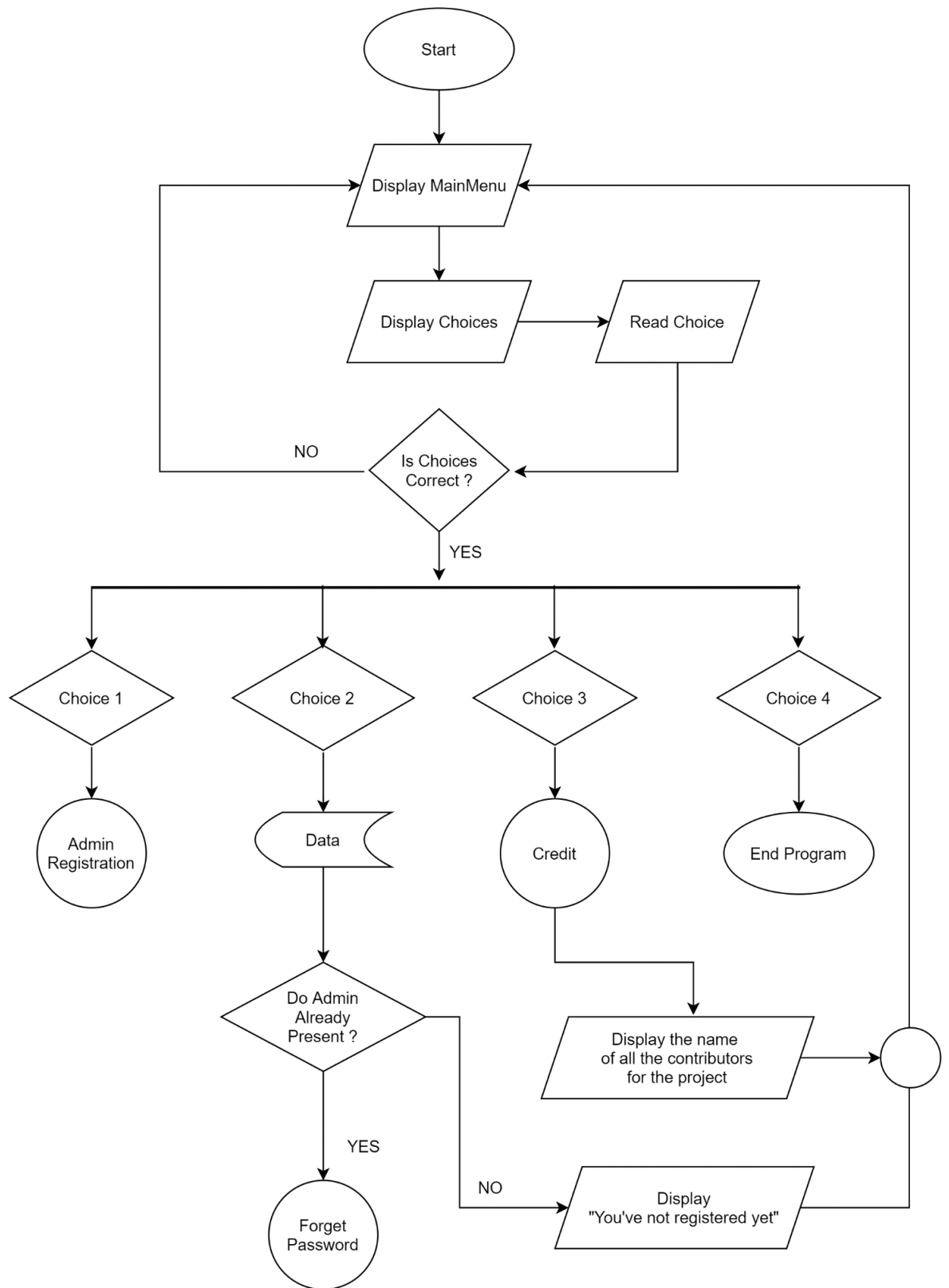
Step 6: Go to **adminPanel**

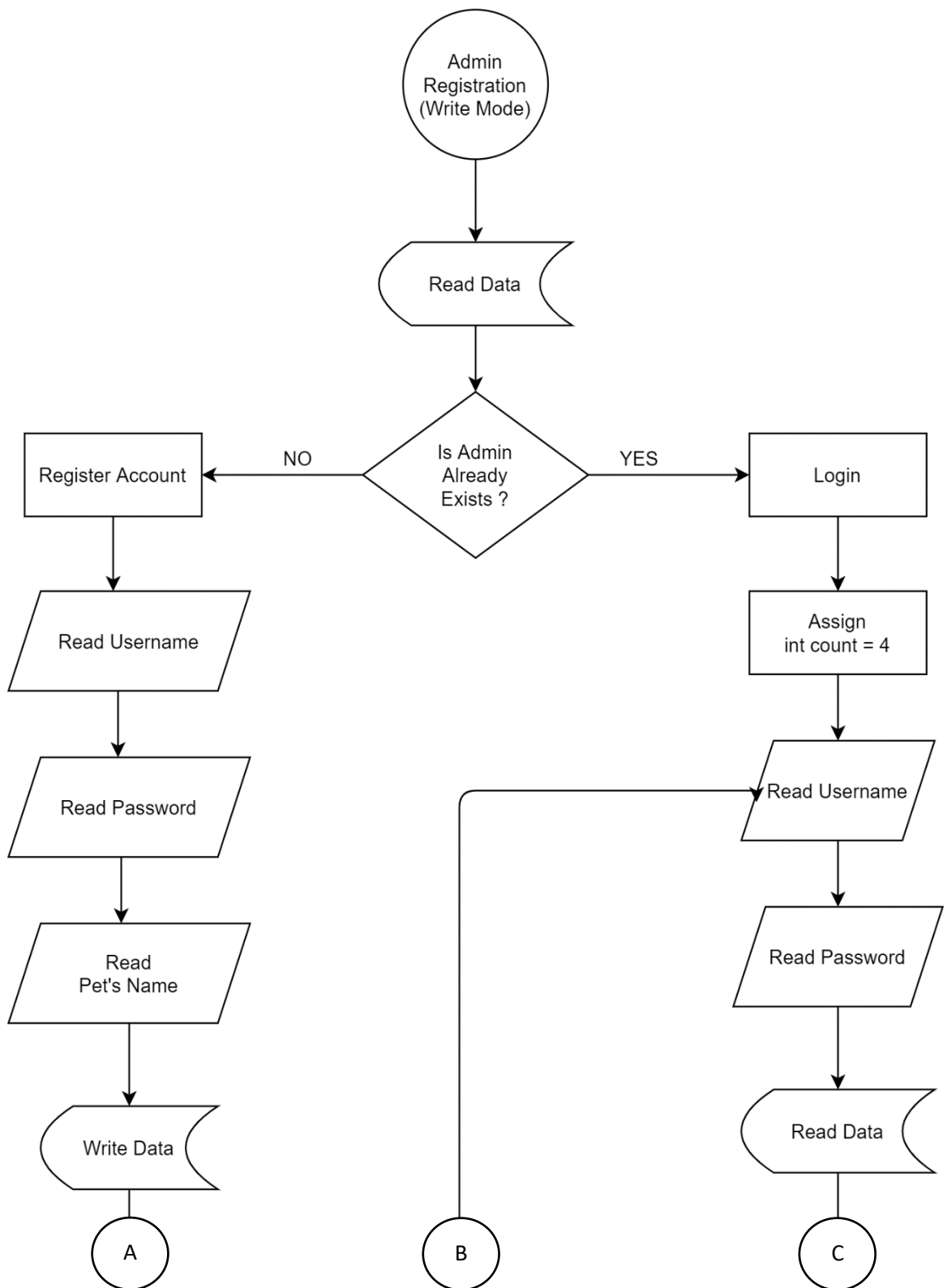
XII. Algorithm for *changePassword*

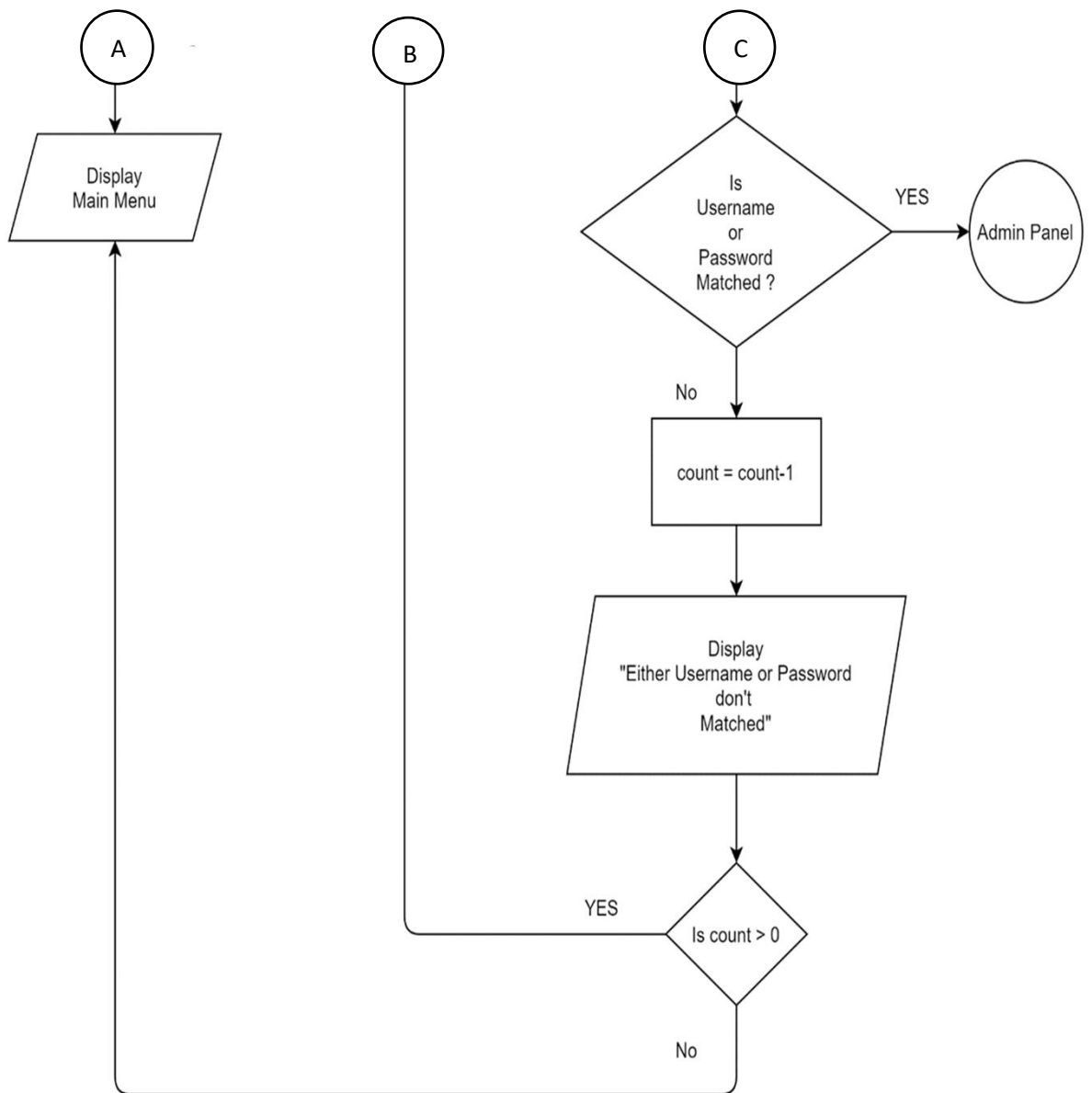
Step 1: Erase the existing record

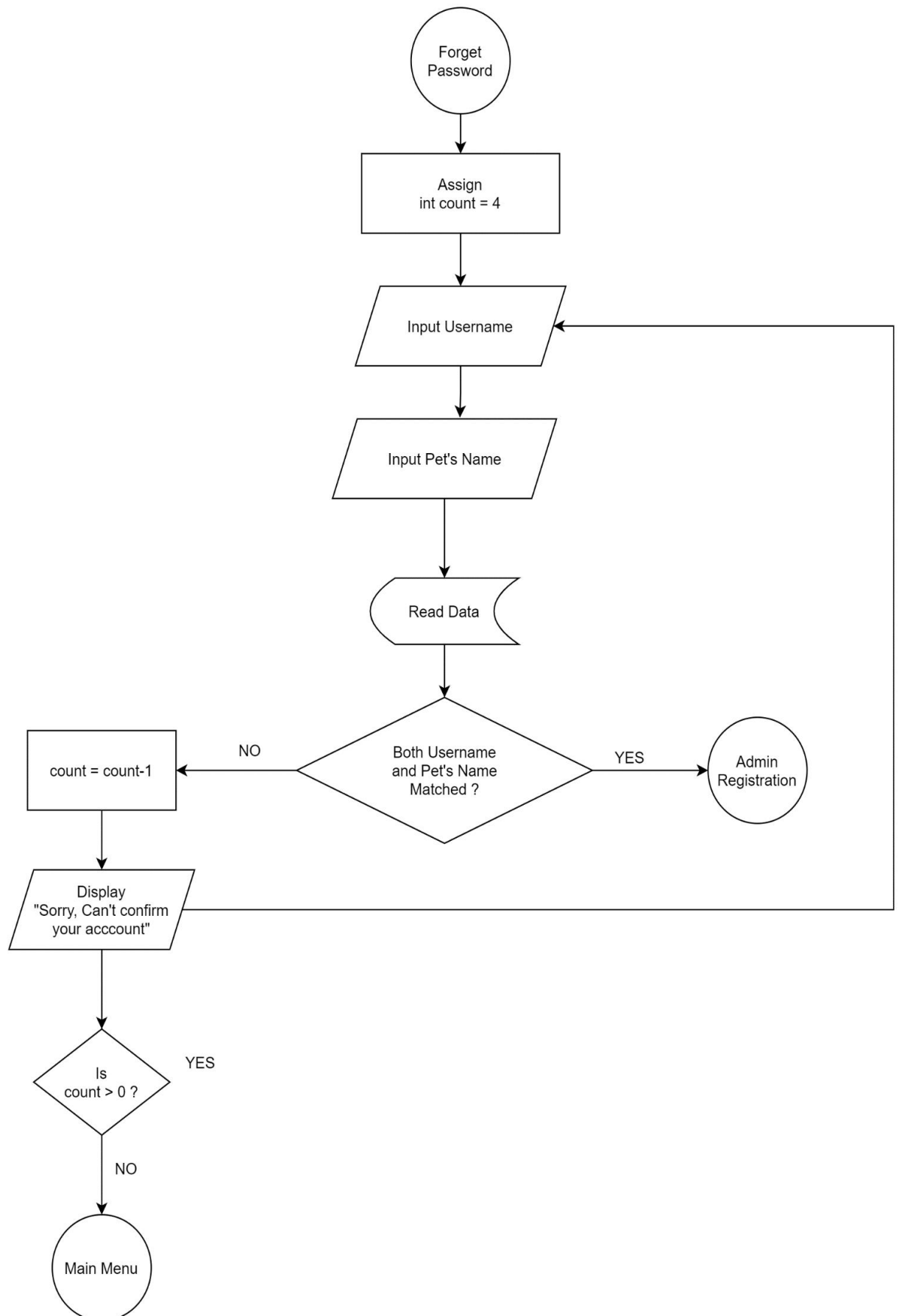
Step 2: Go to **adminRegistration**

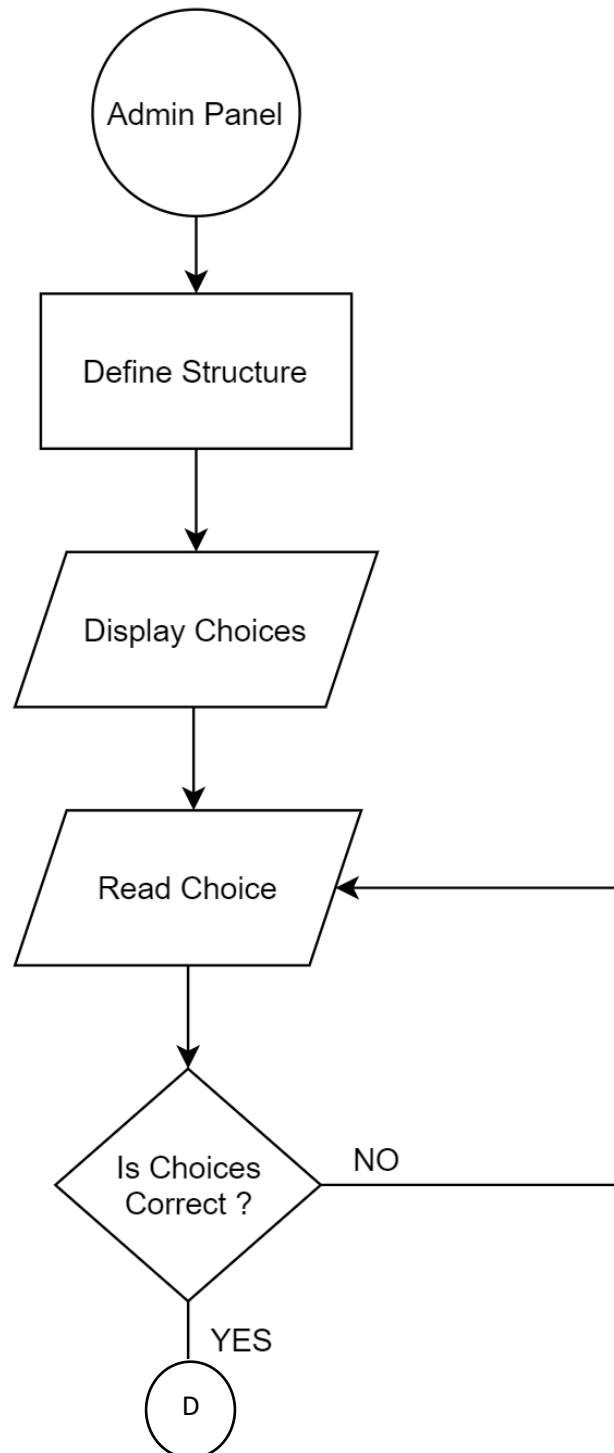
4.3 FLOWCHART

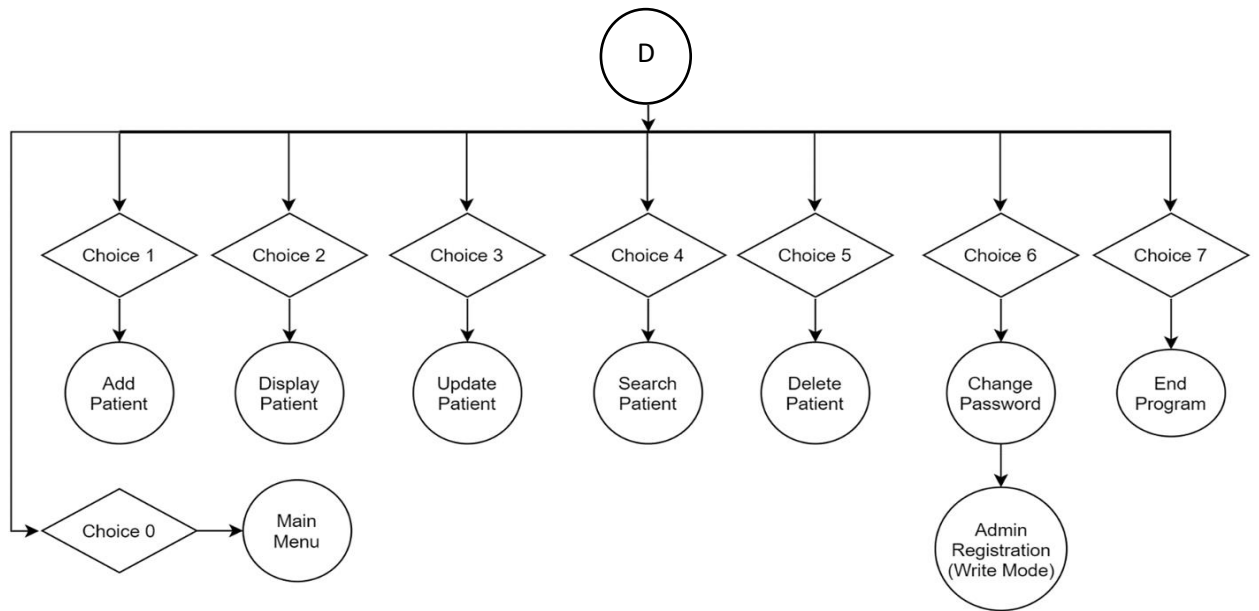


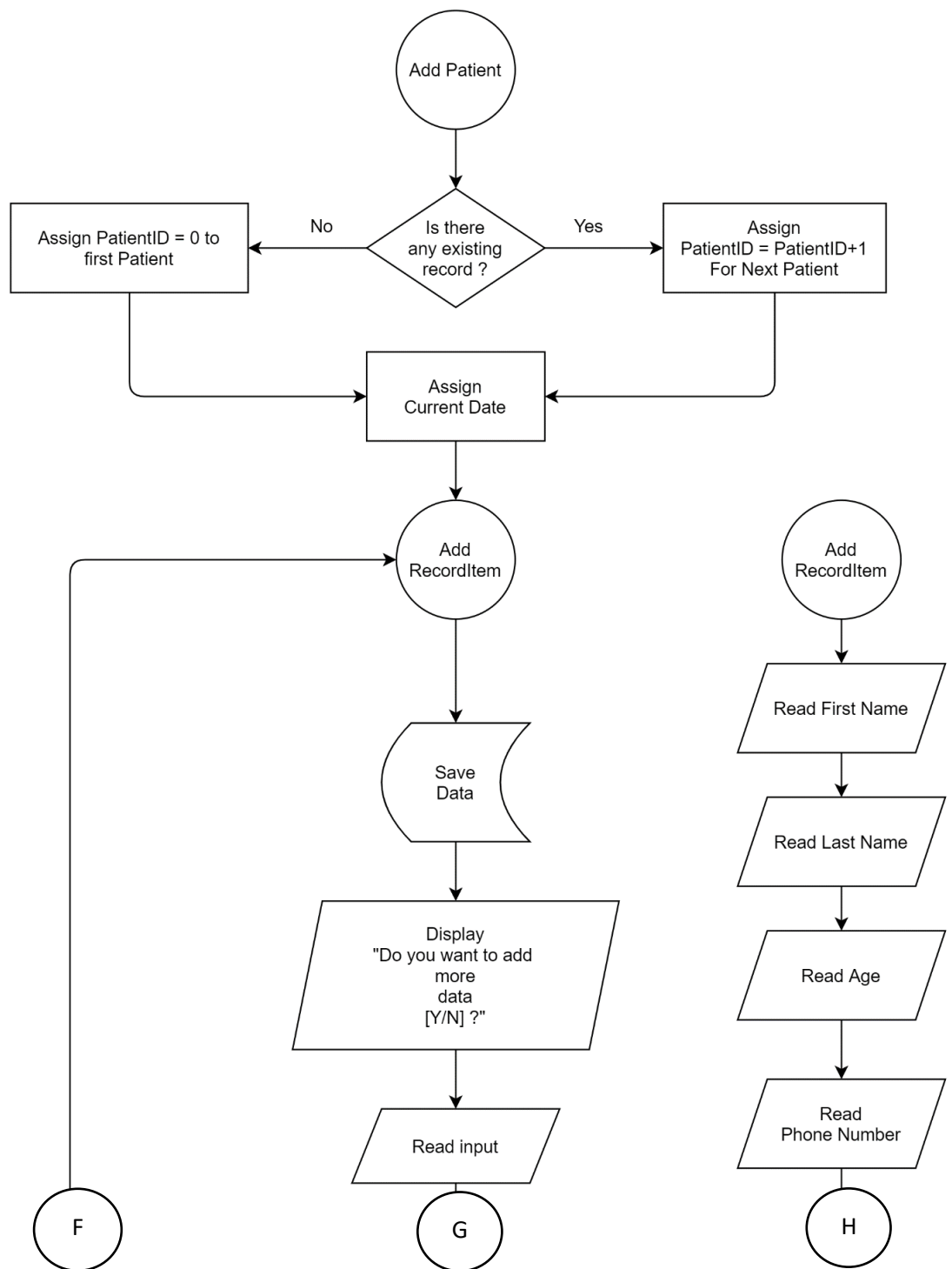


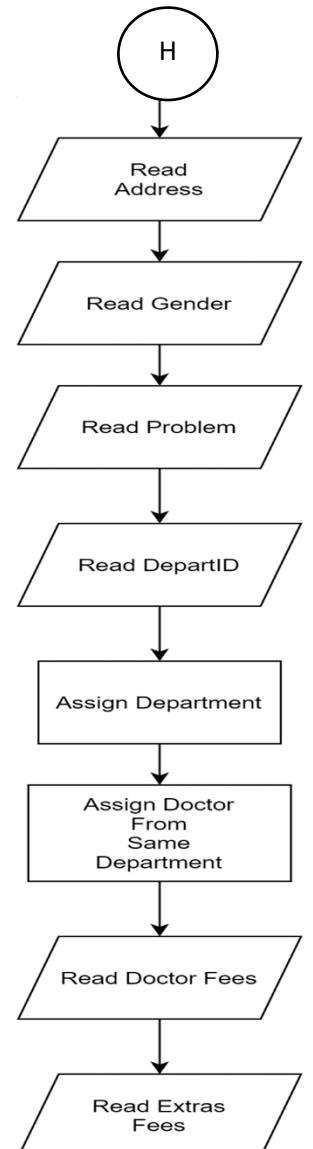
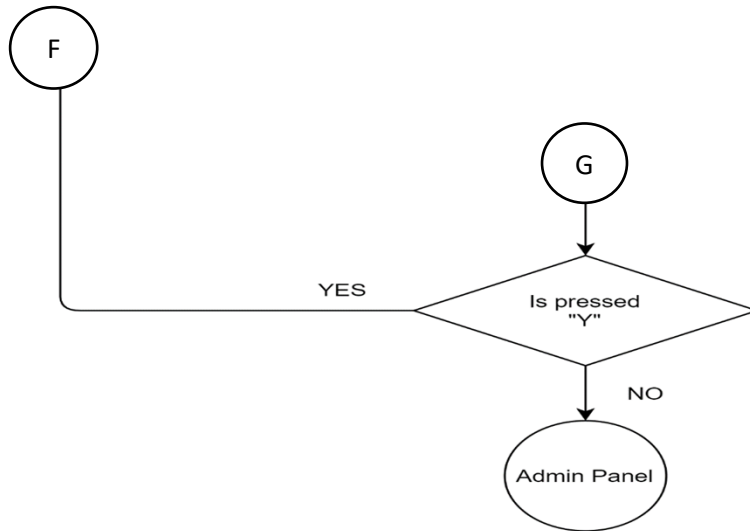


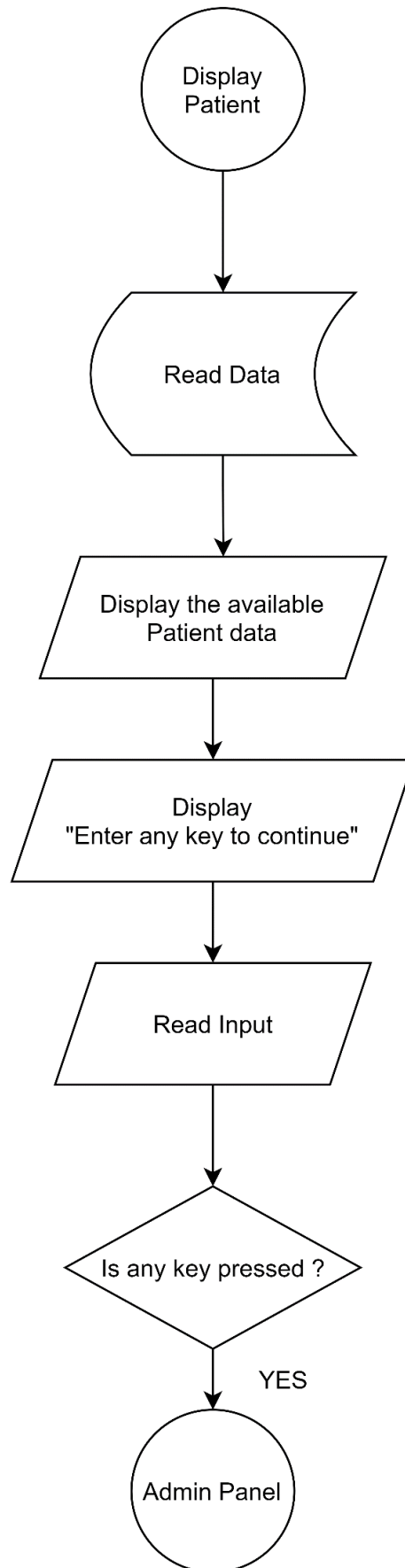


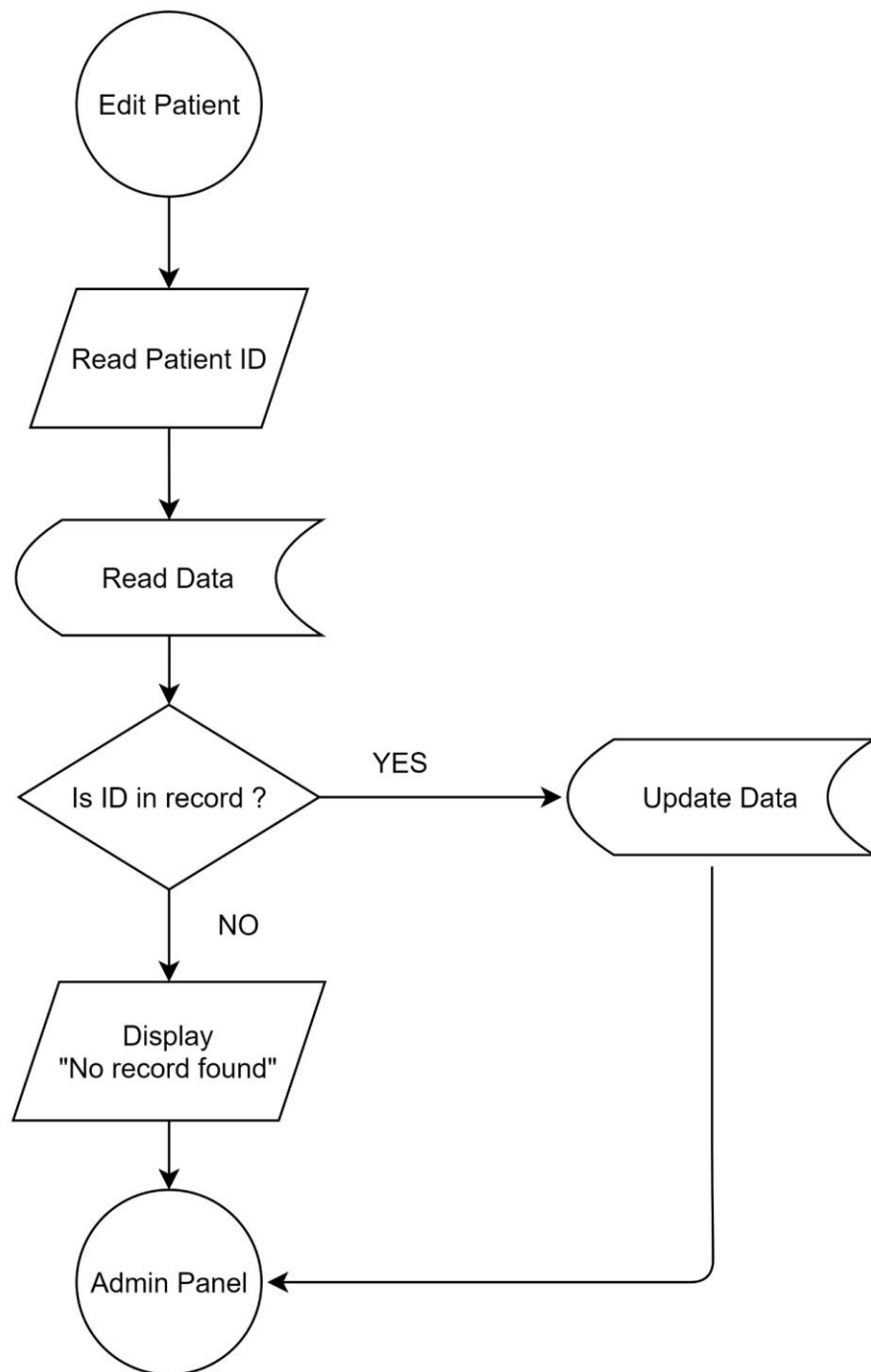


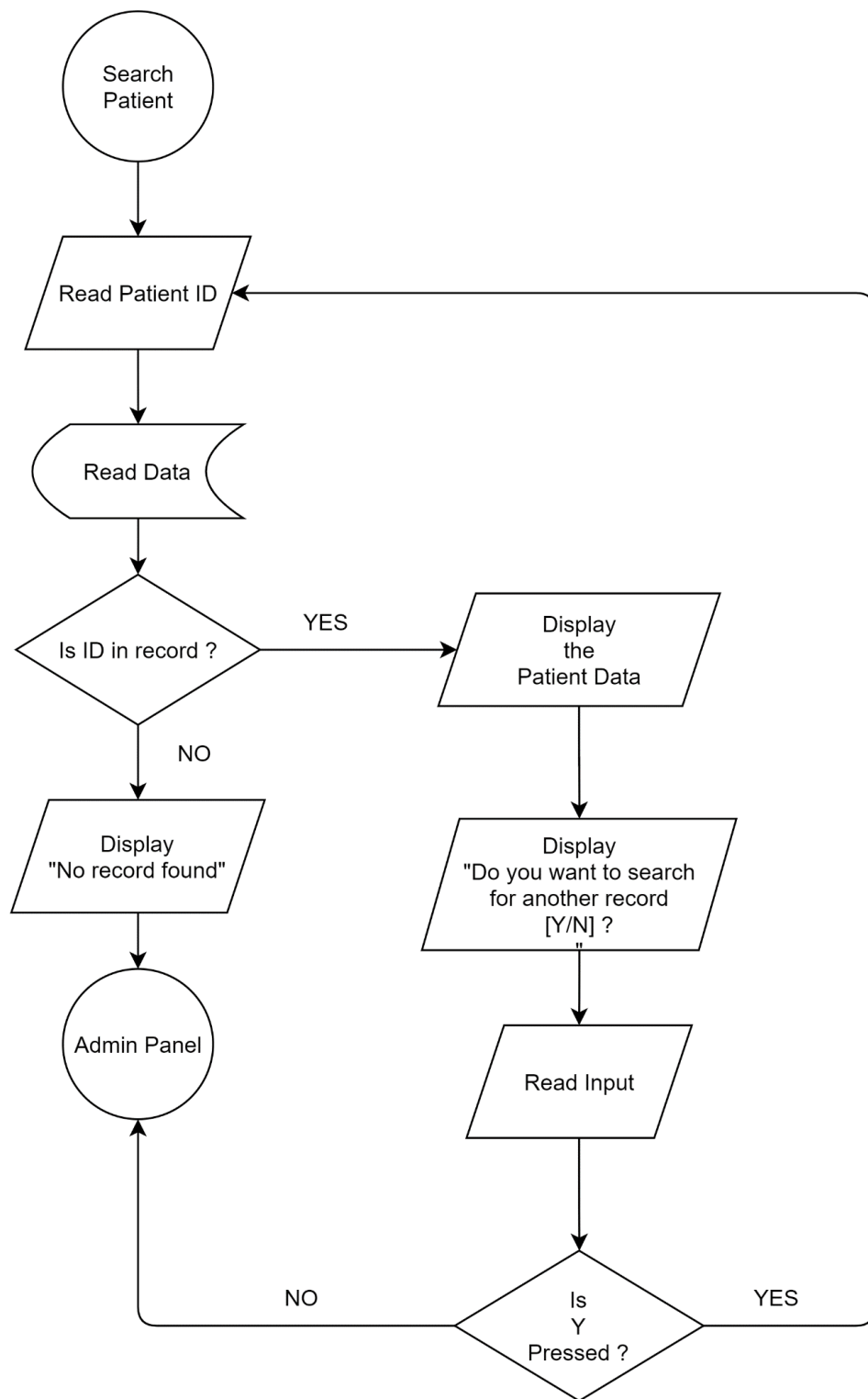


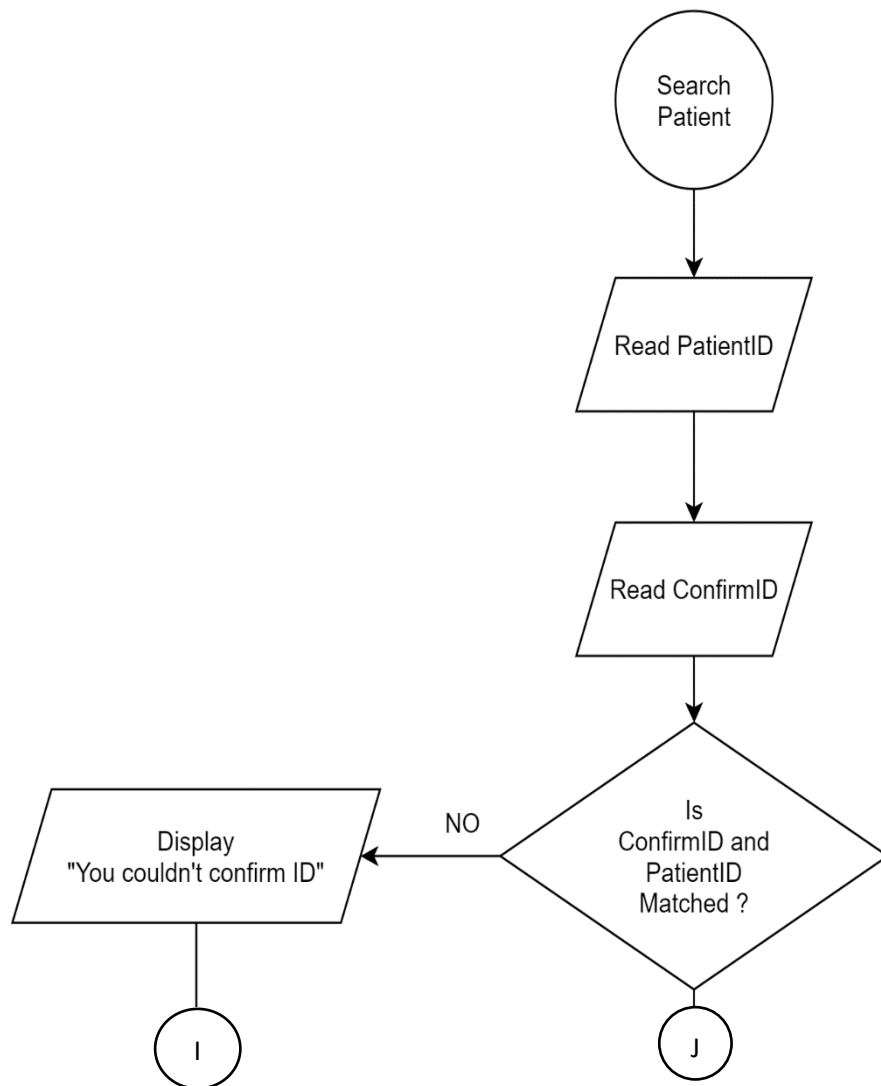


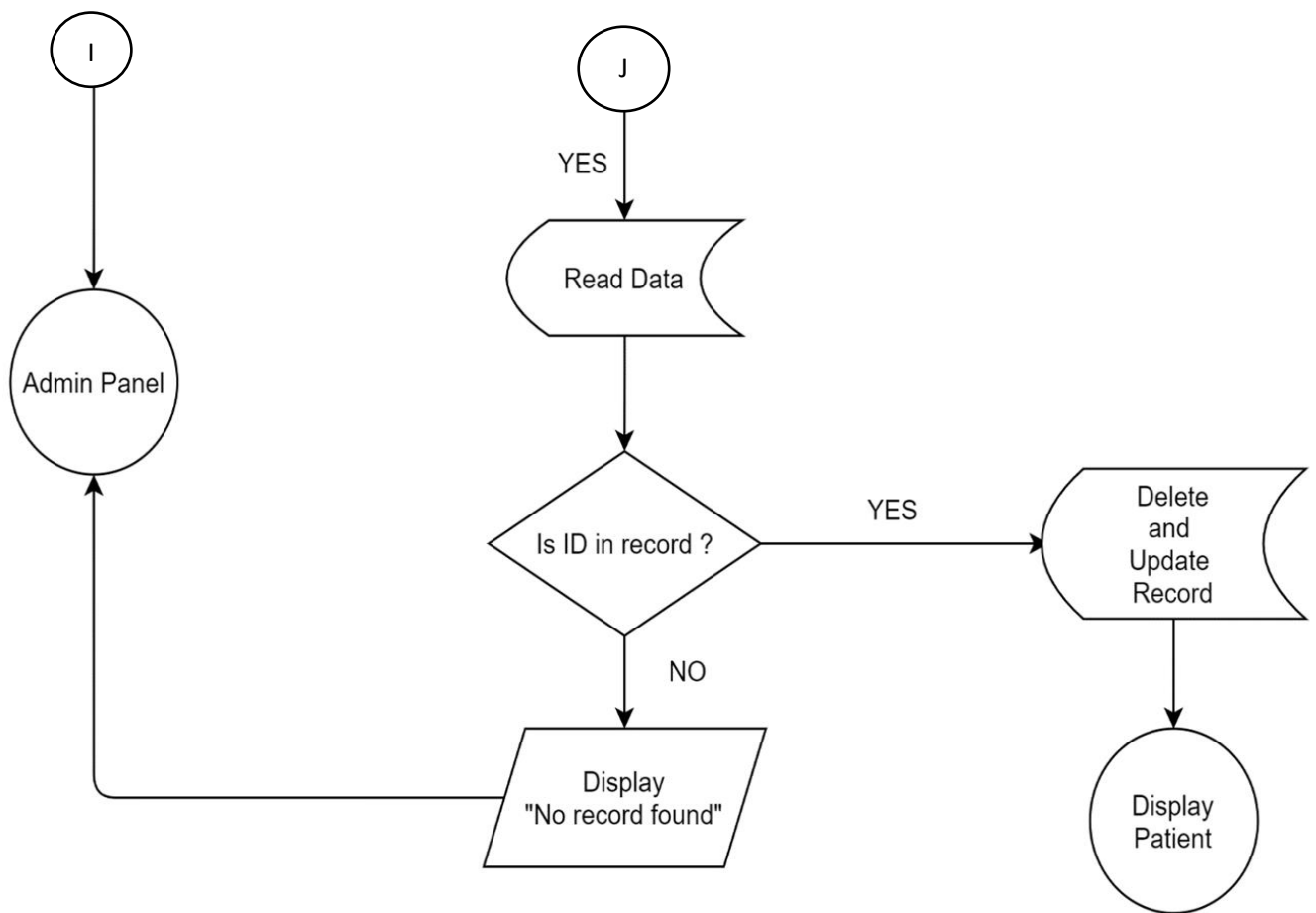












4.4 Coding

```
#include<stdio.h>
#include <time.h>
#include <conio.h>
#include <ctype.h>
#include <string.h>
#include <stdlib.h>
#include <windows.h>
#define true 1
#define false 0
#define ENTER 13
#define BKSP 8
#define TAB 9
#define SPACE 32

int get_size();
void adminRegistration();
void encrypt();
void forgetPassword();
void adminPanel();
void mainMenu();
void welcomeScreen();
void credit();
void changePassword();
void thankyou();
void fullScreen();

void fullScreen() {
    // This makes the application open in fullscreen
    keybd_event(VK_MENU,0x38,0,0);
    keybd_event(VK_RETURN,0x1c,0,0);
    keybd_event(VK_RETURN,0x1c,KEYEVENTF_KEYUP,0);
    keybd_event(VK_MENU,0x38,KEYEVENTF_KEYUP,0);
}

// For AdminPanel Function
void addRecord();
void addRecordItem();
void viewRecord();
int listLoopRow(int row);
void tableHead();
void searchRecord();
void editRecord();
void deleteRecord();

void gotoxy(short x, short y) {
    // For placing (x,y) co-ordinate
    COORD pos = {x,y};
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE)
    ),pos;
}
```



```

        break;
    case 3:
        credit();
        break;
    case 4:
        thankyou();
        break;
    default:
        fflush(stdin);
        mainMenu();
        break;
    }
}

void addRecord() {
// Function to add record of new patient
    system("cls");
    welcomeScreen();
    char ans;
    FILE *fp;
    fp = fopen("patient.txt","a+");
    if (!fp) {
        printf("\n\t Can not open file\n");
        exit(0);
    }

    int patientID;
    int count = 0;
    while(fscanf(fp,"%s%s%s%d%d%s%s%s%f%f%s",
    p.firstName, p.lastName,p.add,p.sex,&p.age,
    &p.patientno,&p.phone,p.problem
    ,p.depart,p.consultant,&p.doc,
    &p.misc,p.registeredDate)!=EOF) {
        count++;
        if (count!=0) {
            patientID = p.patientno+1;
        } else {
            patientID = 0;
        }
    }
    p.patientno = patientID;

    char currentDate[10];
    time_t t = time(NULL);
    struct tm tm = *localtime(&t);
    sprintf(currentDate, "%d-%d-%d",
    tm.tm_year+1900,tm.tm_mon + 1,tm.tm_mday);
    strcpy(p.registeredDate, currentDate );

    addRecordItem(); // Add patient's all data
    fprintf(fp,"%s\t%s\t%s\t%s\t%d\t%d\t%s\t%s\t%s\t%s\t%f\t
    t%f\t% s\n",p.firstName, p.lastName,p.add, p.sex, p.age,
    p.patientno, p.phone,p.problem,p.depart,p.consultant,
    p.doc, p.misc,p.registeredDate);
}

```

```

printf("\n\n\t\t\ Information Record Successful ...");
fclose(fp);
getch();
printf("\n\n\t\t\tDo you want to add more [Y/N]?? ");
fflush(stdin);
scanf("%c", &ans);
if (toupper(ans)=='Y')
    addRecord();
else if (toupper(ans)=='N') {
    adminPanel();
}
}
void addRecordItem(void) {
    // Add record of new patient, this is called inside
    //addRecord function
    int i, departID;

    char doctor[10][10] = {
        "Siddharth", "Dipson", "Simpal", "Madhav", "Sima",
        "Sia", "Poonam", "Anuj", "Yadav", "Bhuwan"
    };
    char department[20][20]= {

        "Ortho", "ENT", "Radiology", "Dental", "Dermatology",
        "Physiology", "Neurology", "Therapy", "Internal-
        Medicine", "Cardiology"};
    fflush(stdin);
    gotoxy(85,15);
    printf("Available Department List\n");
    for(i=0; i<10; i++) {
        gotoxy(85,17+i);
        printf("%d   %s", i, department[i]);
    }
    gotoxy(20,15);
    printf("\t\t\t***** Add Patients Record *****");
    printf("\n\n\t\t\tFirst Name: ");
    scanf("%s",p.firstName);
    p.firstName[0]=toupper(p.firstName[0]);
    printf("\n\n\t\t\tLast Name: ");
    scanf("%s",p.lastName);
    p.lastName[0]=toupper(p.lastName[0]);
    printf("\n\n\t\t\tAddress: ");
    scanf("%s",p.add);
    p.add[0]=toupper(p.add[0]);
    printf("\n\n\t\t\tGender[M|F]:");
    scanf("%s",p.sex);
    p.sex[0]=toupper(p.sex[0]);
    printf("\n\n\t\t\tAge: ");
    scanf("%d", &p.age);
    printf("\n\n\t\t\tPhone Number: ");
    scanf("%s",p.phone);
    printf("\n\n\t\t\tProblem: ");
    scanf("%s",p.problem);
    printf("\n\n\t\t\tDepart ID: ");

```

```
scanf("%d",&departID);

for(i=0; i<10; i++) {
    if(departID==i) {
        strcpy(p.depart, department[i]);
        strcpy(p.consultant, doctor[i]);
        break;
    } else {
        strcpy(p.depart, "Invalid");
        strcpy(p.consultant, "Invalid");
    }
}

printf("\n\t\t\t\t\tDoctor's Charge:");
scanf("%f",&p.doc);
printf("\n\t\t\t\t\tMiscellaneous Charge:");
scanf("%f",&p.misc);

} // addRecordItem ends

void tableHead() {
    // To display table heading
    gotoxy(11,15);
    printf("
_____
_____
_____") ;
    gotoxy(11,16);
    printf("|");
    gotoxy(15,17);
    printf("ID");
    gotoxy(20,17);
    printf("FULL NAME");
    gotoxy(40,17);
    printf("ADDRESS");
    gotoxy(55,17);
    printf("GENDER");
    gotoxy(65,17);
    printf("AGE");
    gotoxy(70,17);
    printf("PHONE NUMBER");
    gotoxy(85,17);
    printf("PROBLEM");
    gotoxy(95,17);
    printf("DEPART");
    gotoxy(110,17);
    printf("CONSULTANT");
    gotoxy(125,17);
    printf("DOC FEES");
    gotoxy(135,17);
    printf("OTHER FEES");
    gotoxy(150,17);
    printf("TOTAL FEES");
    gotoxy(165, 17);
```

```

        printf("Registered Date");
        gotoxy(183,16);
        printf("|");
        gotoxy(183,17);
        printf("|");
        gotoxy(183,18);
        printf("|");
        gotoxy(11,17);
        printf("|");
        gotoxy(11,18);
        printf("|");
        gotoxy(12,18);
        printf("_____
_____
_____
_____
");
}

```

```

int listLoopRow(int row) {
    // To display all the available record (Inside While Loop)
    gotoxy(15,row);
    printf("%i", p.patientno);
    gotoxy(20,row);
    printf("%s %s", p.firstName, p.lastName);
    gotoxy(40,row);
    printf("%s", p.add);
    gotoxy(55,row);
    printf("%s", p.sex);
    gotoxy(65,row);
    printf("%d", p.age);
    gotoxy(70,row);
    printf("%s", p.phone);
    gotoxy(85,row);
    printf(" %s", p.problem);
    gotoxy(95,row);
    printf("%s", p.depart);
    gotoxy(110,row);
    printf("Dr. %s", p.consultant);
    gotoxy(125,row);
    printf("%.2f", p.doc);
    gotoxy(135,row);
    printf("%.2f", p.misc);
    gotoxy(150,row);
    printf("%.2f", p.misc+p.doc);
    gotoxy(165,row);
    printf("%s", p.registeredDate);
} // listLoopRow Ends

```

```

void viewRecord() {
    // View record of all available patient
    int totalMember;
    int row = 20;

```



```

p.registeredDate) !=EOF)
{
    if(p.patientno == searchID) {
        listLoopRow(row);
        row++;
        count++;
    }
}
printf("\n\n\t
_____
_____");

printf("\n\n\t\t\t\t\tTotal Patients found : %d",
count);
fclose(fp); // closing file pointer

searchAgain:
printf("\n\n\t\t\t\t\t Enter any key to
continue...");
getch();
printf("\n\n\t\t\t\t\t Do you want to view more
[Y/N]?? ");
scanf(" %c", &ans);
if (toupper(ans)=='Y')
    searchRecord();
else if (toupper(ans)=='N') {
    adminPanel();
} else {
    printf("\n\tInvalid Input. \n");
    goto searchAgain;
}
} // searchRecord ends

void editRecord() {
// Edit record of patient through ID
welcomeScreen();
int searchID;
int valid = 0;
FILE *fp, *tf;
int row = 20;
char ans;
printf("\n\n\n\n\t\t\t\t***** ABC
Hospital      - Edit Patient's Record
*****\n");
tf=fopen("temp_patient.txt", "w+");
if((fp=fopen("patient.txt", "r"))==NULL) {
    welcomeScreen();
    printf("\n\n\t\t\t\t\t File is empty...");
    printf("\n\n\t\t\t\t\t Enter any key to
continue...");
    getch();
    welcomeScreen();
    adminPanel();
}
}

```



```

        adminPanel();
    }
    // editRecord Ends

void deleteRecord() {
    // delete patient record using ID
    welcomeScreen();
    int searchID;
    int confirmID, found = false;
    int confirm;
    int valid = 0;
    FILE *fp, *tf;
    int row = 20;
    char ans;
    printf("\n\n\n\t\t\t\t\t***** ABC
Hospital - Delete Patient's Record
*****\n");
    if((fp=fopen("patient.txt","r"))==NULL) {
        welcomeScreen();
        printf("\n\n\t\t\t\t\tFile is empty...");
        printf("\n\n\t\t\t\t\tEnter any key to
continue...");
        getch();
        welcomeScreen();
        adminPanel();
    }
    tf=fopen("temp_patient.txt","w+");
    gotoxy(12,16);
    fflush(stdin);
    printf("\n\t\t\t\t\tEnter Patient's ID : ");
    scanf("%i",&searchID);
    printf("\n\n\t\t\t\t\tEnter ID again to confirm : ");
    scanf("%i", &confirmID);
    if(searchID==confirmID) {
        confirm = true;
        while(fscanf(fp,"%s%s%s%d%d%s%s%s%f%f%s",
p.firstName,p.lastName,p.add,p.sex,&p.age,
&p.patientno,&p.phone,p.problem,p.depart,
p.consultant,&p.doc,&p.misc,
p.registeredDate)!=EOF) {
            if(confirmID!=p.patientno) {
                fprintf(tf,"%s\t%s\t%s\t%s\t%d\t%d\t\t
%s\t%s\t%s\t%s\t%f\t%f\t%\n",p.firstName,
p.lastName, p.add, p.sex, p.age,
p.patientno,p.phone,p.problem,p.depart,
p.consultant,p.doc,p.misc,
p.registeredDate);
            }
            else {
                found=true; // record found
            }
        }
    } else {

```

```

        confirm = false;
    }
    if(!confirm || !found) {
        fclose(fp);
        fclose(tf);
        remove("temp_patient.txt");
        if(!confirm) {
            printf("\n\n\t\t\t\t\t You couldn't
            confirm ID ");
        } else if (!found) {
            printf("\n\n\t\t\t Record not found ");
        }
        printf("\n\n\t\t\t\t\tEnter any key to
        continue...");
        getch();
        welcomeScreen();
        adminPanel();
    }
    if(found) {
        printf("\n\n\t\t\t\t\t Record Deleted
        Successfully !!");
        fclose(fp);
        fclose(tf);
        remove("patient.txt");
        rename("temp_patient.txt","patient.txt");
        printf("\n\n\t\t\t\t\t Enter any key to continue
        ");
        :
        getch();
        welcomeScreen();
        viewRecord();
    }
} // Delete record ends

```

UNIT 5.

USER MANUAL

Unit 5

User Manual

5.1 Minimum Hardware Requirements

For smooth running of the project the following minimum hardware is needed.

- Core 2 Duo or Later
- VGA or other display compatible with Windows
- 200 MB of RAM
- 4GB of Hard disks

5.2 Minimum Software Requirements

Choosing a programming language appropriate to the system is equally important as designing a system. Now, there are many languages and packages available and all have their own importance in their respective fields. We developed our project in Windows environment using C language for smooth running of the project, the following minimum software is needed.

- Windows 98 or higher

5.3 Installation and Launch

There is ready made portable compiled console-based program file with name **OutPatientManagement.exe** which works directly by double clicking it. It doesn't need any installation. User can simply copy the file and paste wherever they like and run the program.

The best way is to place the program inside the own new folder. For this, simply create a folder and place the program inside it.

5.4 User Manual

There are multiple features available in this program. Amin will get login/register menu and when they logged in, they will migrate to Admin Panel. To properly run and use all the features, kindly follow all the necessary manual given.

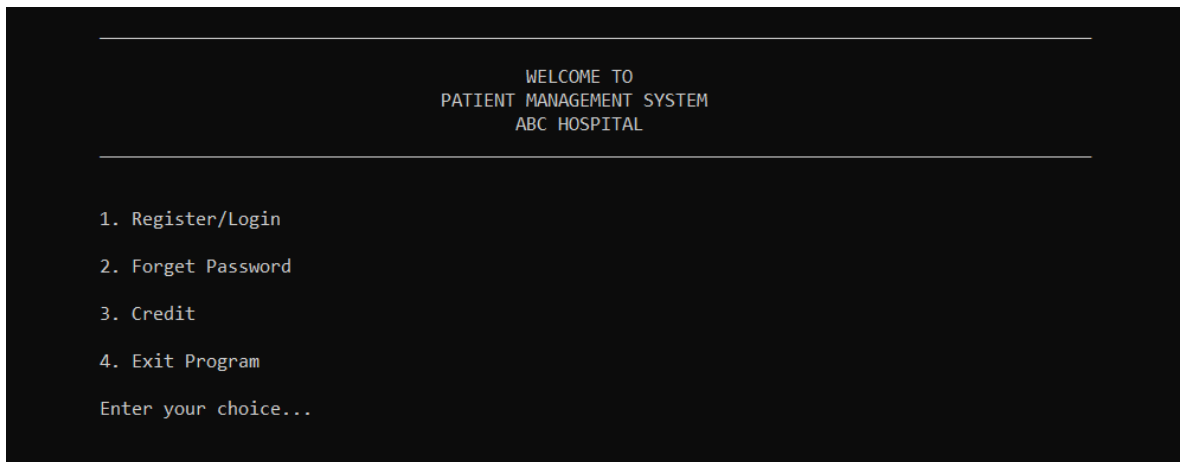


Fig 1.1: Main Menu

5.4.1 Main Menu

Double click the program to open it. Then the Main Menu will get displayed. Here are 4 options available for user. User can choose according to their need.

1 for either to register or login, 2 if user forget the password, 3 to display the name of all the contributors of the project, and 4 to terminate the program.

5.4.2 Register/Login Interface

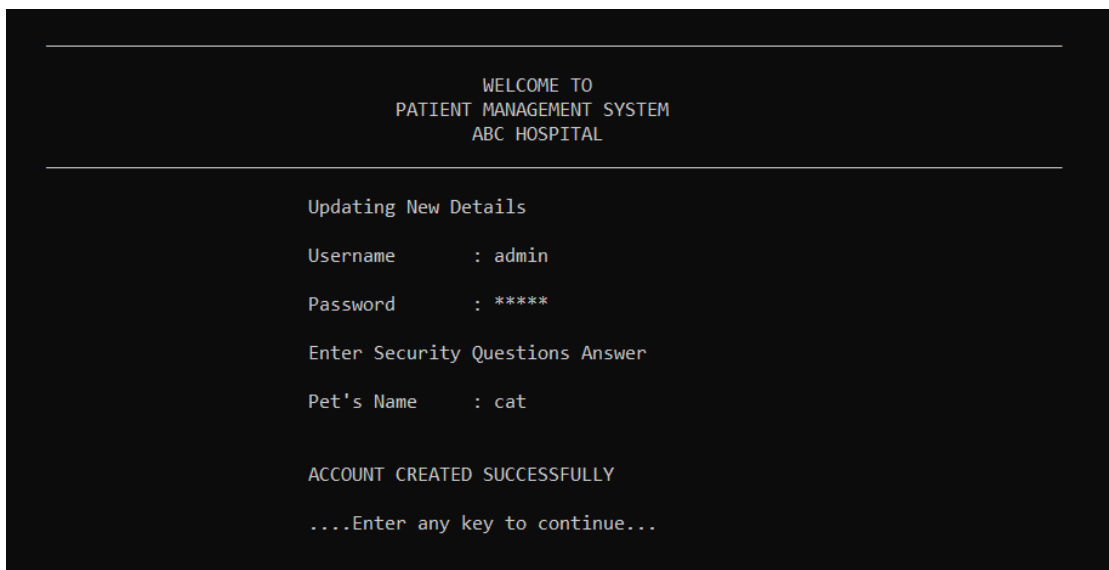


Fig 1.2: Register Interface

By entering all the details asked, user can register to their account. After register user need to login in and then they can visit admin panel. Both Register and Login option is pretty had same interface, where login lacks Pet's name. User have only need to enter Username and Password.

5.4.3 Forget Password



Fig 1.3: Forget Password

By any chance if user forget their password, then there is option to reset their password, for that, user need to verify their account with username and pet's name. If the details get matched, then they will have the option to overwrite their existing account with new details.

5.4.4 Credit

When user enter 3 in Main Menu, then the screen shows the name of all the contributors.

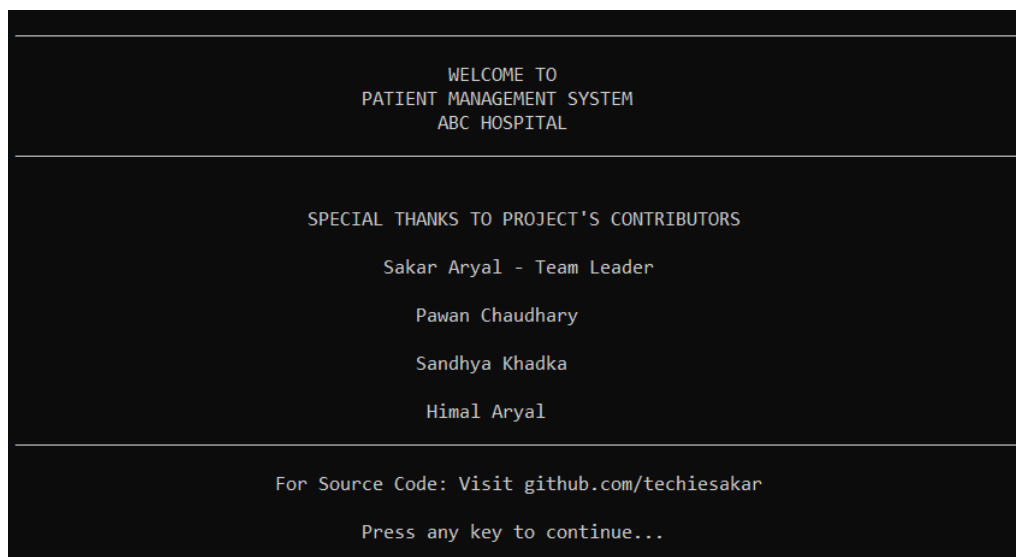


Fig 1.4: Credit

5.4.5 Add Patient

In add record, user have to add the record of name, age, gender, phone number, of the patient along with referred depart ID and charges.

User don't have to assign ID for patient, patient ID will be automatically assigned when there is new patient added on the record.

```

WELCOME TO
PATIENT MANAGEMENT SYSTEM
ABC HOSPITAL

***** Add Patients Record *****

First Name: Samir
Last Name: Aryal
Address: Manigram
Gender[M|F]:M
Age: 21
Phone Number: 9867700000
Problem: Leg
Depart ID: 0
Ticket Charge:100
Miscellaneous Charge:0

Available Department List
0 Ortho
1 ENT
2 Radiology
3 Dental
4 Dermatology
5 Physiology
6 Neurology
7 Therapy
8 Urology
9 Cardiology

.Information Record Successful ...
```

Fig 1.5: Add Patient

5.4.6 View Patient

All the listed record of patient will show here in ordered list. ID and date of registration will auto assign and department and doctor will get auto assigned according to the depart ID as entered before during adding the patient record.

ID	FULL NAME	ADDRESS	GENDER	AGE	PHONE NUMBER	PROBLEM
0	Samir Aryal	Manigram	M	21	9867700000	Leg
1	Samar Kharel	Mangalapur	M	22	9867700556	Gastric
2	Priya Tamang	Drivertole	F	18	9867723100	Hand
3	Suman Rana	Murgiya	M	26	9867703256	Teeth
4	Eelena Karki	Manigram	F	22	9867723875	Skin

Enter any key to continue...

Fig 1.6: Patient Record 1

DEPART	CONSULTANT	TICKET FEES	OTHER FEES	TOTAL FEES	Registered Date
Ortho	Dr. Siddharth	100.00	0.00	100.00	2021-11-17
Medicine	Dr. Sangam	100.00	0.00	100.00	2021-11-17
Ortho	Dr. Siddharth	100.00	0.00	100.00	2021-11-18
Dental	Dr. Malika	100.00	0.00	100.00	2021-11-18
Dermatology	Dr. Nabeen	100.00	0.00	100.00	2021-11-18

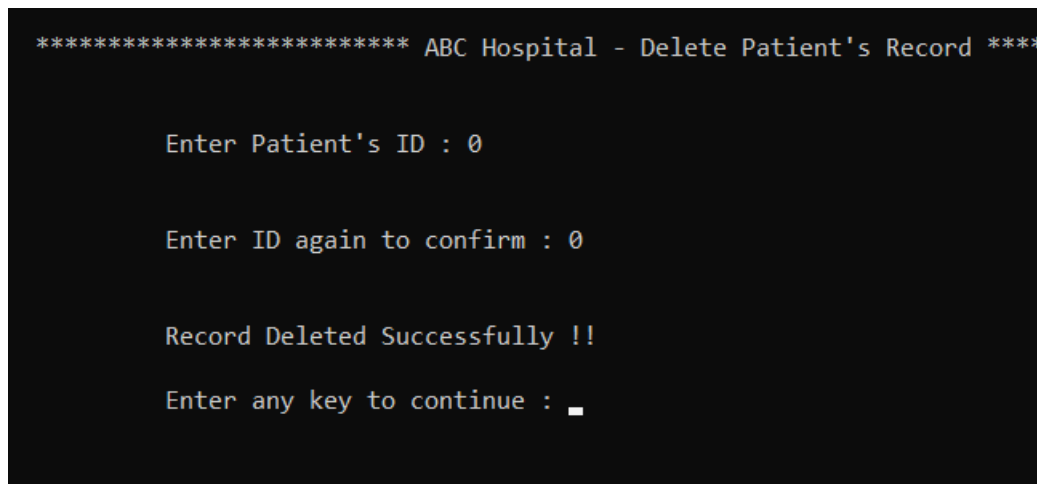
Fig 1.7: Patient Record 2

5.4.7 Search Patient:

User have to search patient with their ID, if there exists a patient, then the detail of the patient will show similar like in above Patient Record figure. If not then, user will get message "Patient not found"

5.4.8 Delete Patient:

To delete patient from the record, user need to enter ID of the respective patient for twice. One for entering the user ID and another for confirming the ID. This is used to prevent the user from deleting the record mistakenly.



```
***** ABC Hospital - Delete Patient's Record *****

Enter Patient's ID : 0

Enter ID again to confirm : 0

Record Deleted Successfully !!

Enter any key to continue : _
```

Fig 1.8: Delete Patient

5.4.9 Update Admin's Details

If user select to update the admin's details, then all the admin's existing record will be overwritten with the new details. Then admin can use the new details to access their account.

5.5.0 Logout:

If user logout then, they will move to the Main Menu, where they have to login again to reach to the Admin Panel.

Conclusion

A login-based Outpatient Management System using the C-Programming Language has been successfully completed.

The project demonstrated the creation of a user interface of a system without the use of C Graphics library. We have made use of the basic features to generate menus and prints texts on the screen, to display text according to the application requirements.

We have also implemented the concept of structures to keep record of patients. It also effectively applies the various C concepts such as File operations, Looping and branching, constants and string manipulation functions.

All the necessary outputs along with flowcharts, coding, snapshots for the project have been presented. Hence, the project has been completed successfully.

Bibliography

System Analysis and Design Methods:

- Jeffrey L. Whitten
- Lonnie D. Bentley
- Kevin C. Dittman
- draw.io (To draw Flowchart)

Programming in C:

- Yashwant Kanetkar
- Ram Datta Bhatta

Online references:

- stackoverflow.com
- youtube.com
- github.com