**Outpatient Management System**

***Project Code: PRJ 151***

A First Year Project

For the partial fulfillment of requirement for the degree of

Bachelor of Computer Application (BCA)



**Submitted By**

|  |  |  |
| --- | --- | --- |
| **Name** | **Registration No** | **Symbol No** |
| Sakar Aryal | 2019-1-53-0090 | 20530145 |
| Himal Aryal | 2019-1-53-0072 | 20530129 |
| Sandhya Khadka | 2019-1-53-0093 | 20530148 |
| Pawan Chaudhary | 2019-1-53-0081 | 20530138 |

**Submitted To**

Pokhara University

Faculty of Science and Technology,

Crimson College of Technology

Nov, 2021

**DECLARATION**

We hereby declare that, the project entitled **“Outpatient Management System”** is an outcome of our own efforts under the guidance of **Mr. Nabraj Koirala.** The project is submitted to **Pokhara University** for the partial fulfillment of the ***Bachelor of Computer Application (BCA) Second Semester Examination 2020.***

Any kinds of reproduction of this report and its parts for commercial use is strictly prohibited without the prior written permission of the author and the university. However, this report (excluding source code) is free to use for academic purpose.

**Team Members**

……………………………

…………………………. Mr. Nabraj Koirala

Sakar Aryal Supervisor,

Department of IT,

…………………………. Crimson College of Technology

Himal Aryal

………………………….

Sandhya Khadka

………………………….

Pawan Chaudhary

**BONAFIDE CERTIFICATE**

This is to certify that the project entitled “**Outpatient Management System**” has been carried out by the team member “**Sakar Aryal, Himal Aryal, Pawan Chaudhary and Sandhya Khadka**” under our guidance in partial fulfillment of the degree of Bachelor of Computer Application of Pokhara University, Nepal during the academic year 2020 (Semester II).

…………………………… ……………………………

Mr. Nabraj Koirala Mr. Abdul Hak

Supervisor, Coordinator,

Department of IT, Department of IT,

Crimson College of Technology Crimson College of Technology

**CERTIFICATE FROM THE EXTERNAL EXAMINER**

This is certified that the bonafide students, **“Sakar Aryal, Himal Aryal, Sandhya Khadka and Pawan Chaudhary”** from **Crimson College of Technology** have successfully completed, presented and demonstrated the project on “***Outpatient Management System***” in the partial fulfillment of the requirement of the degree of Bachelor of Computer Application (BCA) for the year 2020.

During the presentation, I’ve found that the students are bonafide on their work. They are enthusiastic, hardworking and ready to face any skillful work related to IT and Computer Application.

**………………………………..**

External Examiner

**ACKNOWLEDGEMENT**

We would like to express our gratitude and appreciation to all those who gave us the possibility to complete this project report. Special thanks are due to Mr. Nabraj Koirala, Supervisor of IT Department, Crimson College of Technology, whose stimulating suggestions and encouragement helped us in all time and in writing this report. We also sincerely thanks for the time spend proofreading and correcting out many mistakes.

Moreover, we would also like to acknowledge with must appreciation the crucial role of the staff of Library Management who gave us the valuable resources which were the most for undertaking our project work. Last, but not the least we are thankful to our friends for their direct and indirect help, co-operation and encouragement.

**Project Team Members**

Sakar Aryal

Himal Aryal

Sandhya Khadka

Pawan Chaudhary

**PREFACE**

With the change in the time and new discoveries in computer science, human life is no more a static existence. Computer’s invention is the greatest invention of mankind which made out life easier, comfortable and reliable.

We developed this system in DEV C++ in C Programming Language in Windows environment.

We tried to develop the software for the Outpatient Management System, which will provide the easy way to keep the record of Outpatient.

The system has been developed as a collection of several independent modules and any number of modules can be added later without affecting the existing modules.

We all members have tried our best to make this project errorless and if any errors found related to the project will be resolved heartily by us.

**TABLE OF CONTENTS**

**Contents Page No**

Abstract 9

List of Tables 10

List of Figures 10

List of Charts 10

**UNIT 1: Introduction of The Project 11-12**

* 1. Introduction 11
  2. Objectives 11
  3. Applications 12
  4. Features 12
  5. Limitations 12

**UNIT 2: Project Development Life Cycle 13-15**

2.1 Project Planning and Feasibility 13

2.2 Requirement Analysis 13

2.3 System Design 13

2.4 Methodology 13-14

2.5 Coding 14

2.6 Debugging and Testing 14-15

2.7 Implementing 15

2.8 Documentation 15

2.9 Project Development Tools 15

**Unit 3: System Requirements 16**

3.1 Hardware Configuration 16

3.2 Software Configuration 16

**UNIT 4: Time, Cost and Task Division 17**

4.1 Time and Cost 17

4.2 Task Division 17

**Unit 5: Appendixes 18-50**

5.1 Gantt Chart 18

5.2 Algorithm 19-22

5.3 Flowchart 23-35

5.4 Coding 36-50

**Unit 6: Snapshots 51-53**

**Unit 7: Conclusion and Bibliography 53-54**

7.1 Conclusion 53

7.2 Bibliography 54

**Abstract**

The proposal entitled “Outpatient Management System” is prepared as partial requirement for the completion of one credit hour course of Bachelor of Computer Application as the second semester project.

The use of computer has made the world smaller. It is quite impossible to think of any information system without the implementation of the computer technology. Computer technology has affected the entire field such as business field, scientific field, management field etc. Users like to access for different information of the Hospital and its patient.

The project mainly emphasizes on providing the information related to Outpatient Department. This helps administrative users to keep record of Outpatient. This project is made up of login-based system which provides secure, user friendly and clean multiple options for the administrator.

From the technical point of view, the C programming is used and the project can run even in Windows 98 and will low processor computer. For designing the system, we’ve used Window 10 OS with Intel Core i5 5th Gen processor with 8GB RAM and 500GB SSD.

**List of Tables**

|  |  |  |
| --- | --- | --- |
| **Table No** | **Description** | **Page No** |
| 1.1 | Cost Estimation | 10 |
| 1.2 | Task Division | 10 |

**List of Figures**

|  |  |  |
| --- | --- | --- |
| **Table No** | **Description** | **Page No** |
| 2.1 | Water Fall Model | 5 |

**List of Special Chart**

|  |  |  |
| --- | --- | --- |
| **Table No** | **Description** | **Page No** |
| 3.1 | Gantt Chart | 12 |

**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**

**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 basic of the required information the project should be planned. A feasibility report is prepared to resent in depth techno commercial analysis carried out on the project idea.

**2.2 Requirement Analysis**

This is the 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 record 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: 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 exits
* 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**

1. **Hardware**
2. Intel Core i5
3. 256 GB SSD
4. 8 GB RAM
5. Printer and Lamination Machine
6. **Software**
7. Windows 10 OS
8. Microsoft Word
9. DEV C++
10. VsCode
11. OneDrive

**Unit 3:**

**System Requirements**

**3.1 Hardware Configuration**

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

* Pentium i-55 MHz or higher
* VGA or other display compatible with Windows
* 16 MB of RAM
* 200 MB of Hard disks

**3.2 Software Configuration**

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

**Unit 4:**

**Time, Cost and Task Division**

**4.1 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 1.1 Cost Estimation**

**4.2 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 1.2: Task Division**

**Unit 5:**

**Appendixes**

**5.1 Gantt Chart**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **SN** | **Tasks** | **Oct**  **10-15** | **Oct**  **16-20** | **Oct**  **21-31** | **Nov**  **1-5** | **Nov**  **6-8** | **Nov**  **9-11** | **Nov**  **12-13** |
| 1 | Analysis |  |  |  |  |  |  |  |
| 2 | Design |  |  |  |  |  |  |  |
| 3 | Coding |  |  |  |  |  |  |  |
| 4 | Testing |  |  |  |  |  |  |  |
| 5 | Implementation |  |  |  |  |  |  |  |
| 6 | Maintenance |  |  |  |  |  |  |  |
| 7 | Documentation |  |  |  |  |  |  |  |

**Index**

** Less Work (Light Shade)**

 **More Work (Dark Shade)**

**5.2 Algorithm**

Step 1: Start

Step 2: Go to mainMenu

**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

**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.**

**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**

**Algorithm For *credit***

Step 1: Display all the contributor’s name

Step 2: Display “Enter any key to continue”

Step 3: Go to mainMenu

**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**

**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.**

**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

**Algorithm for *viewRecord***

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

**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**

**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**

**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

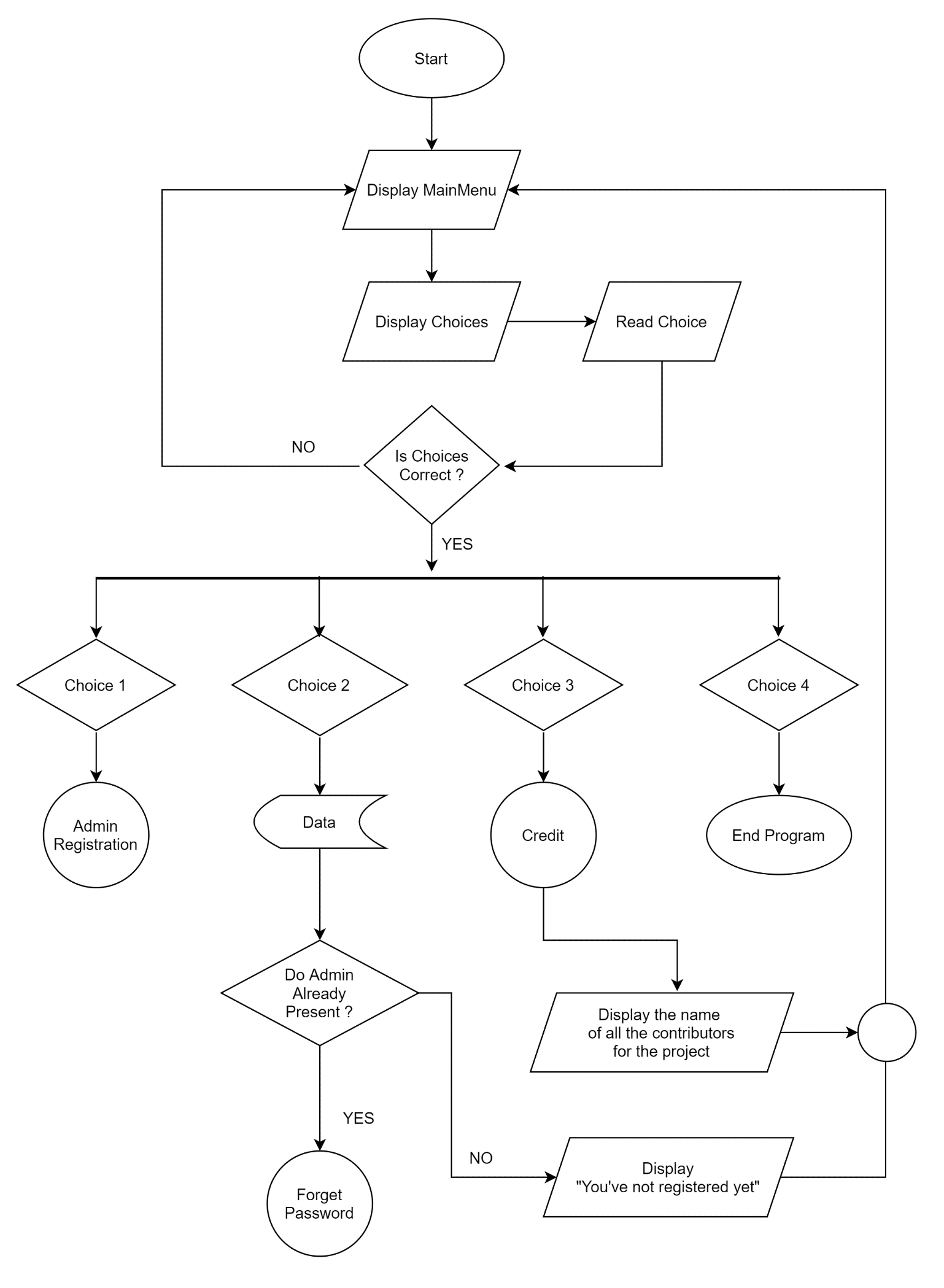
Step 6: Go to **adminPanel**

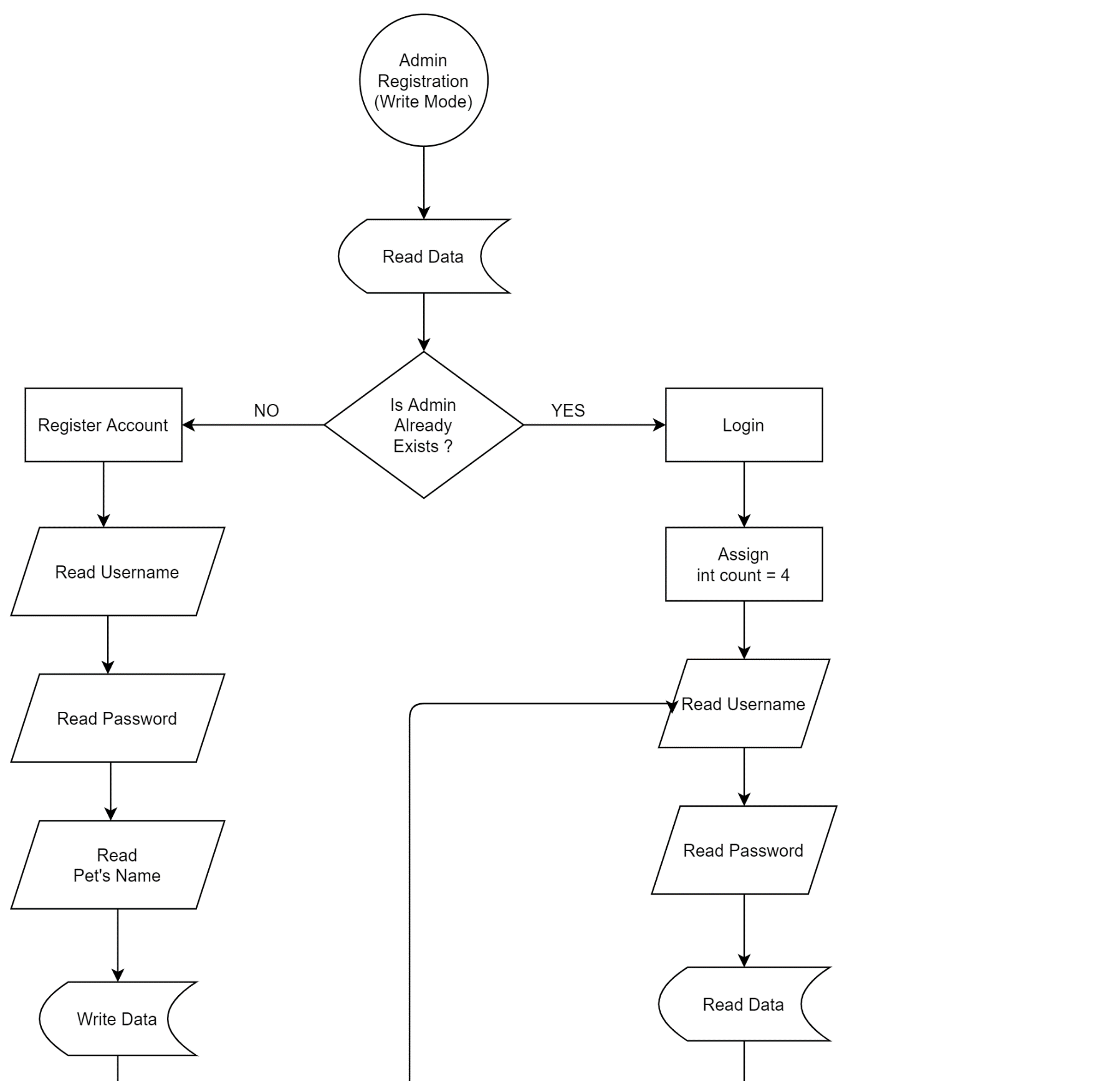
**Algorithm for *changePassword***

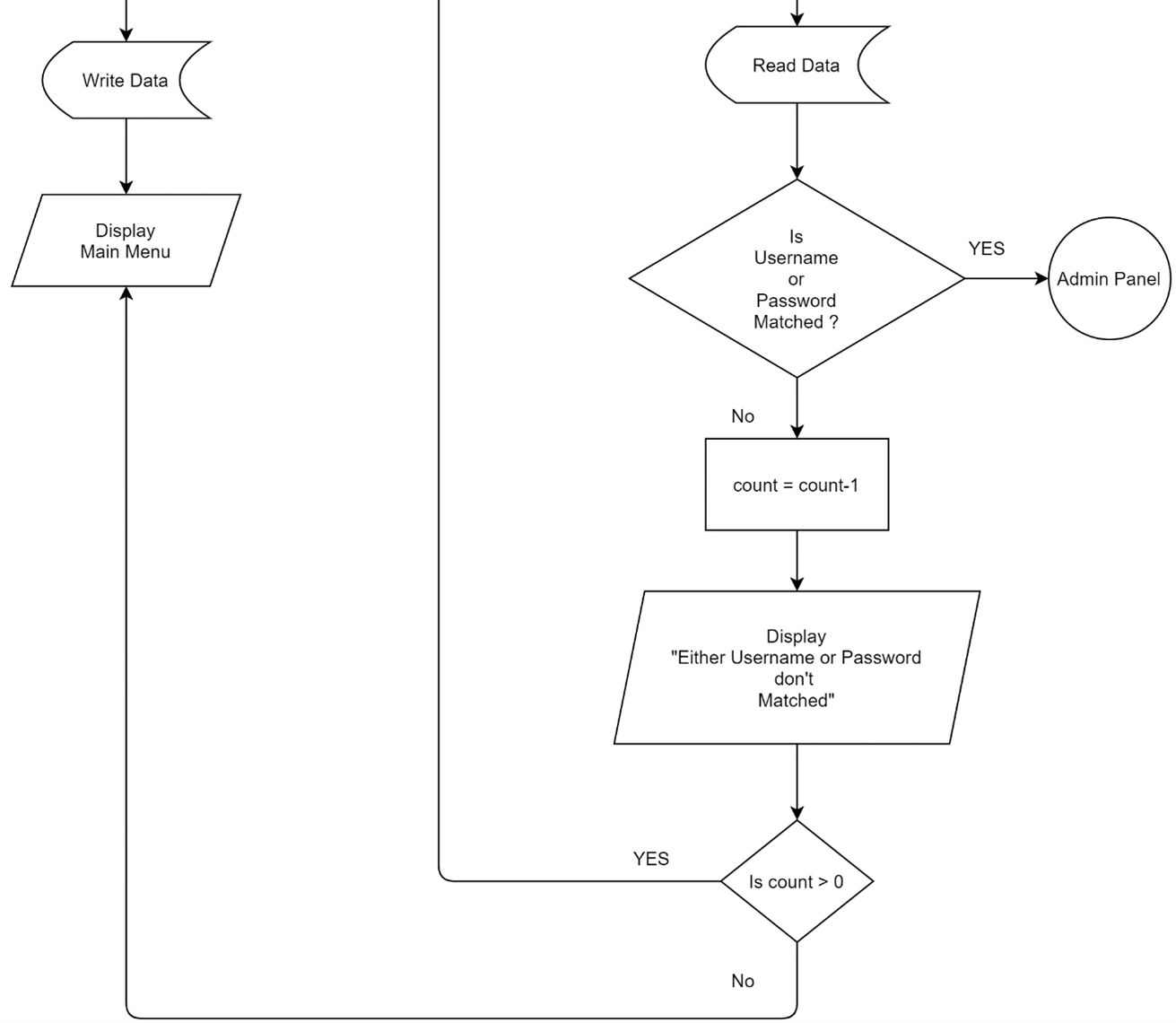
Step 1: Erase the existing record

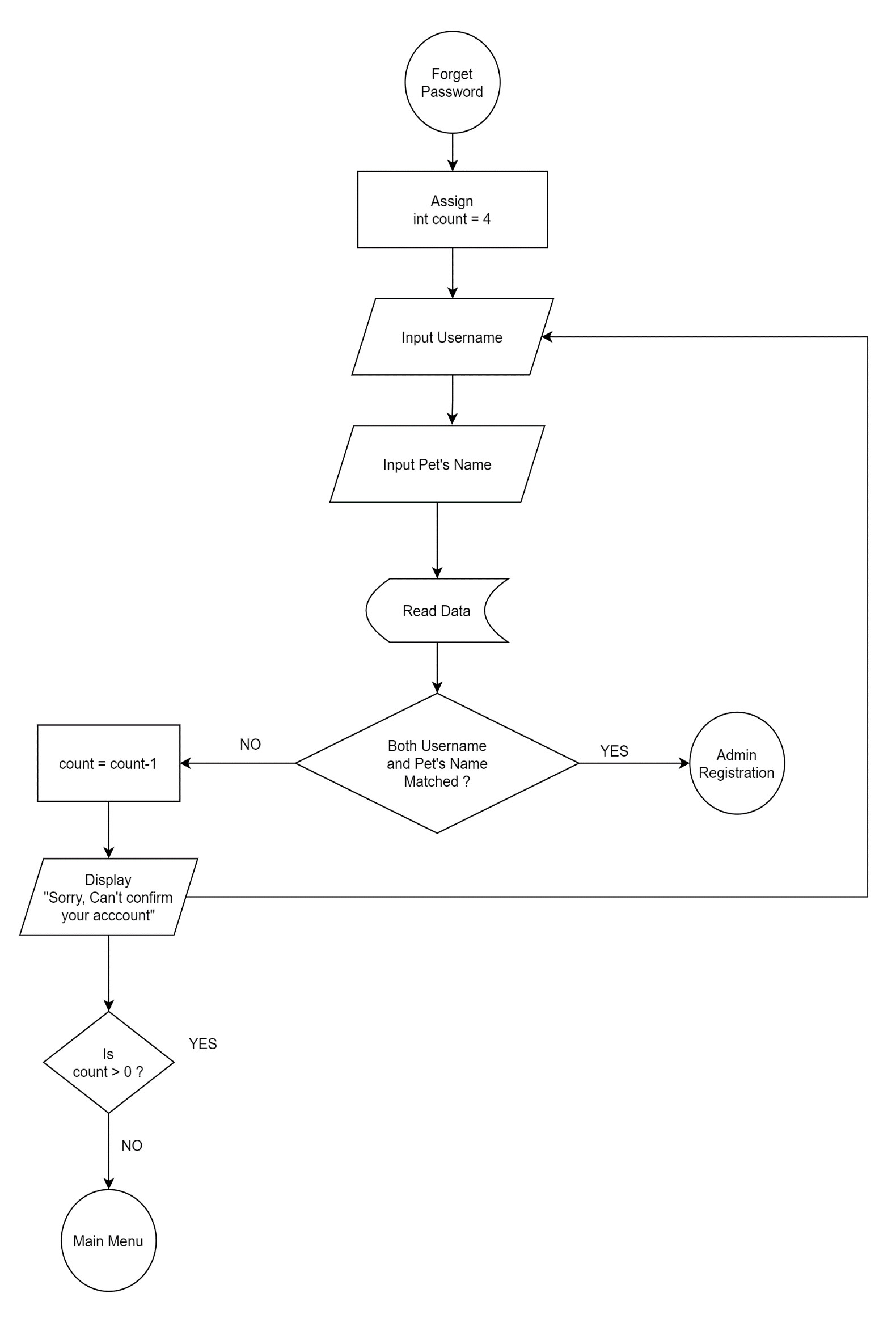
Step 2: Go to **adminRegistration**

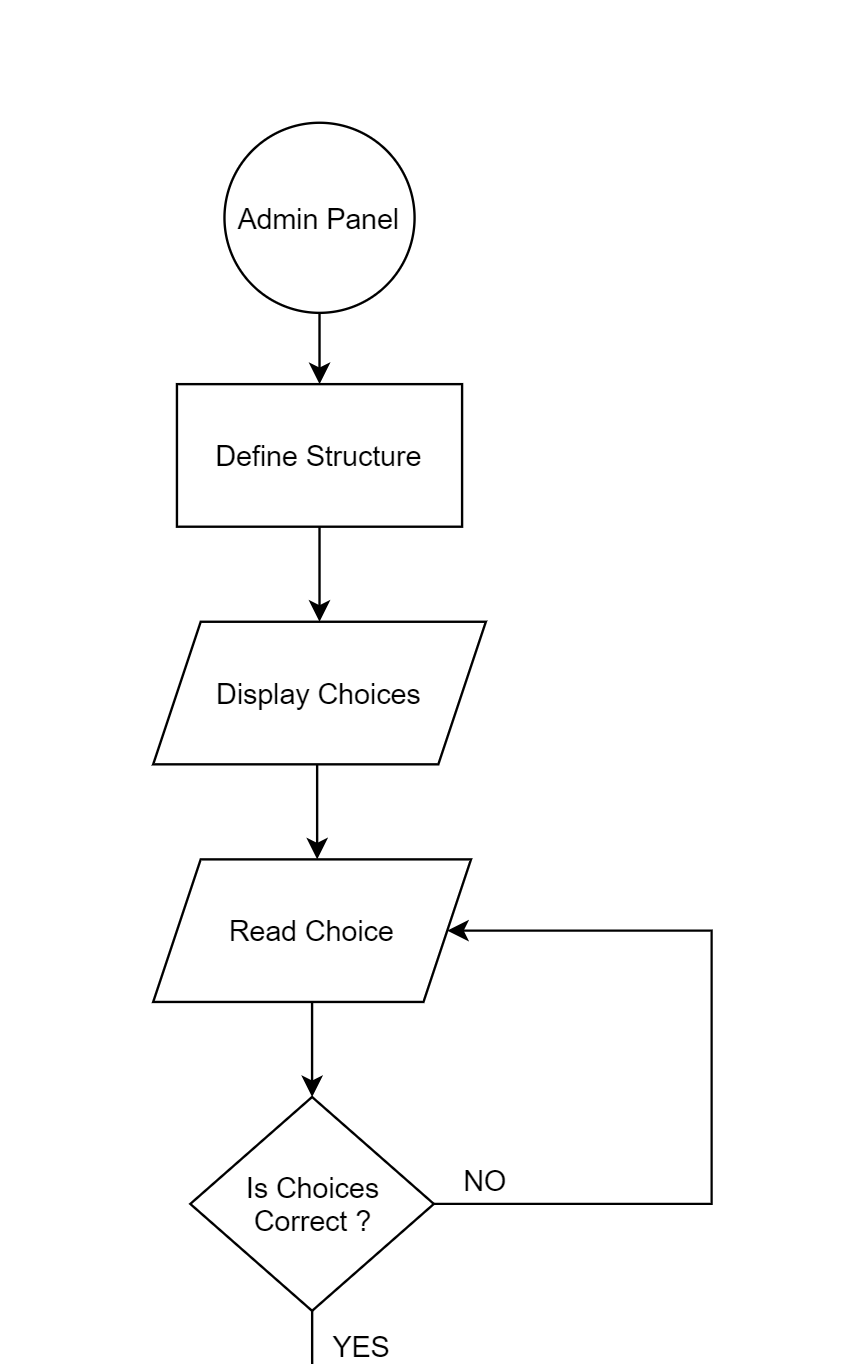
**5.2 FLOWCHART**

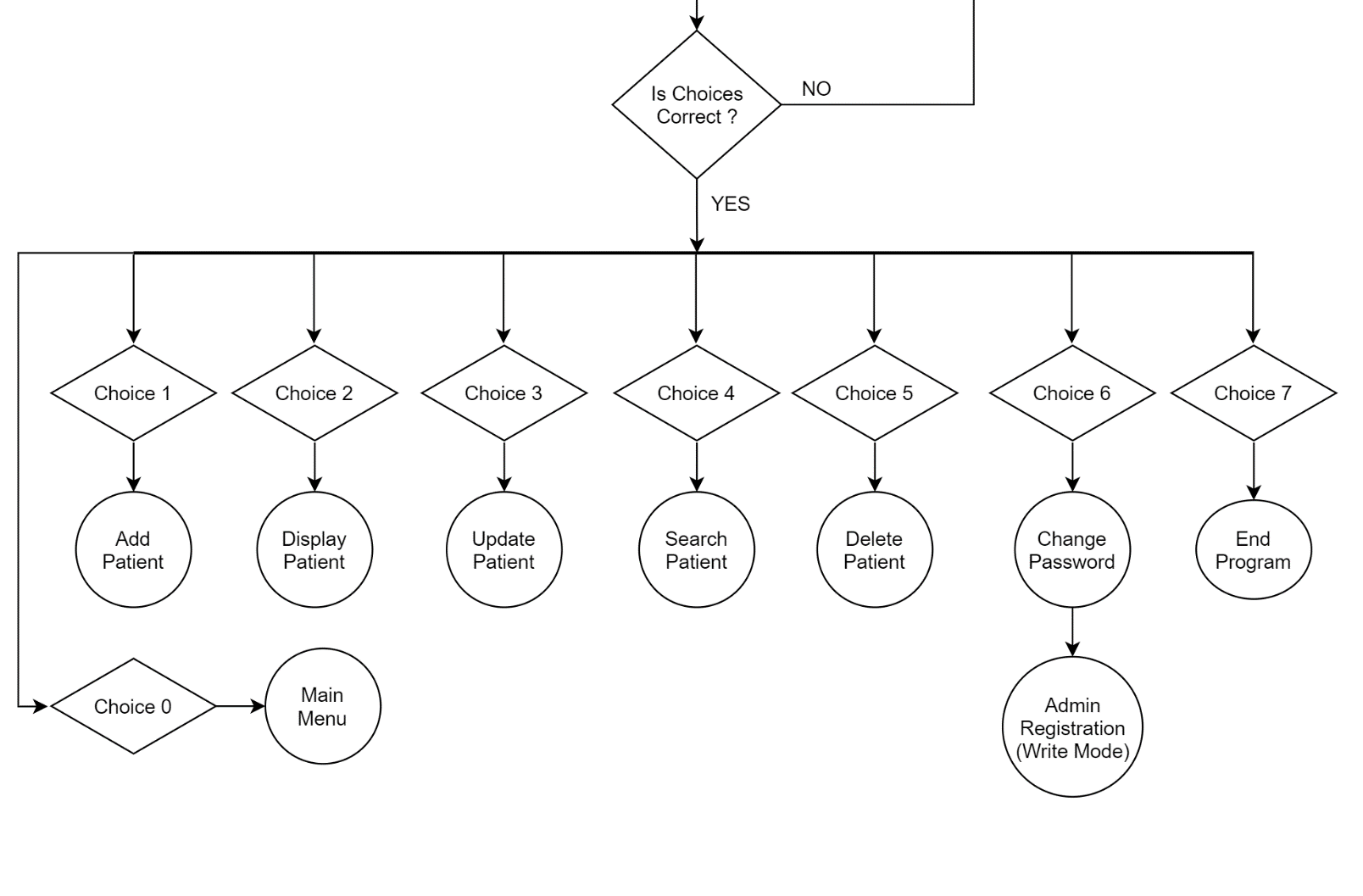
****

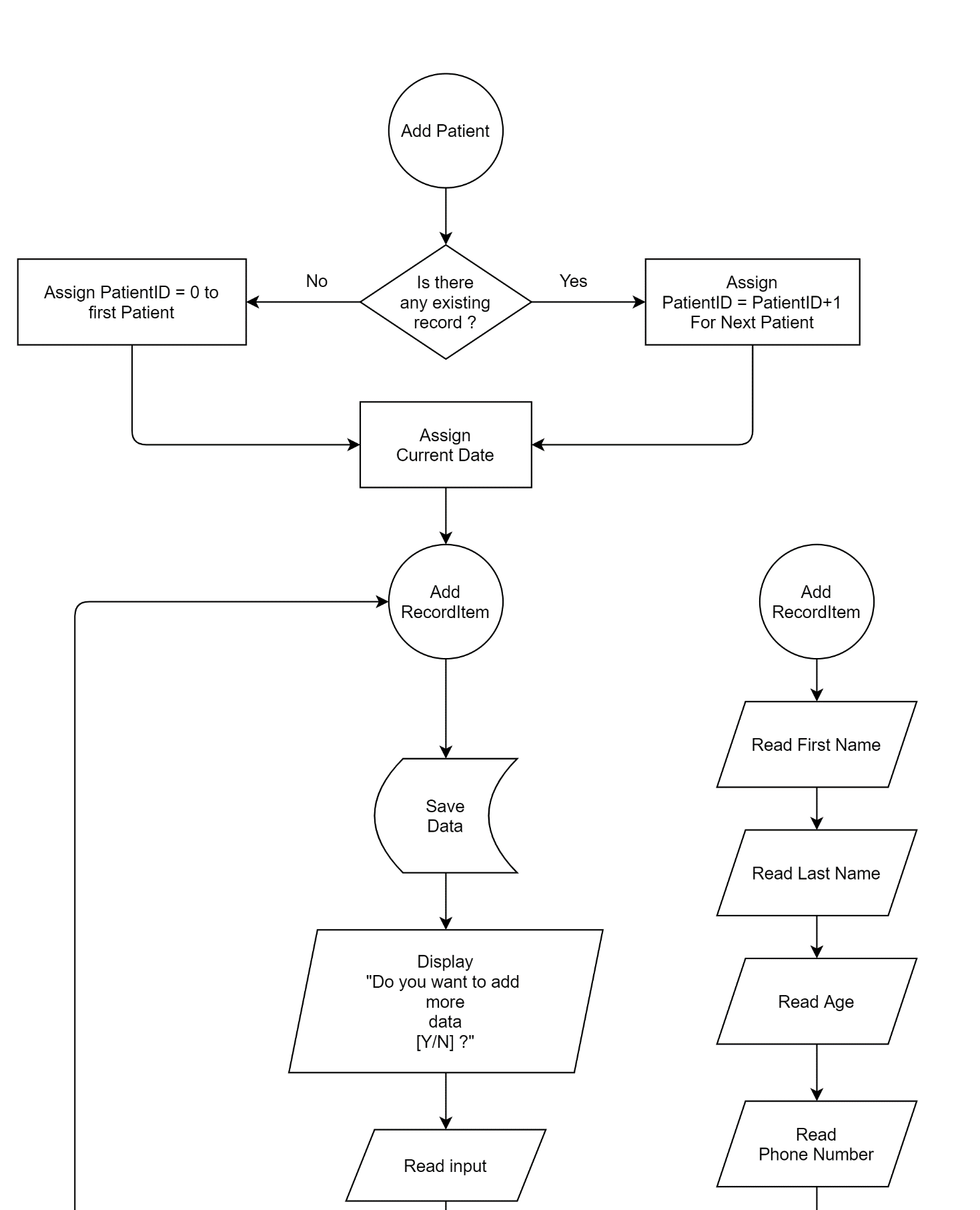
****

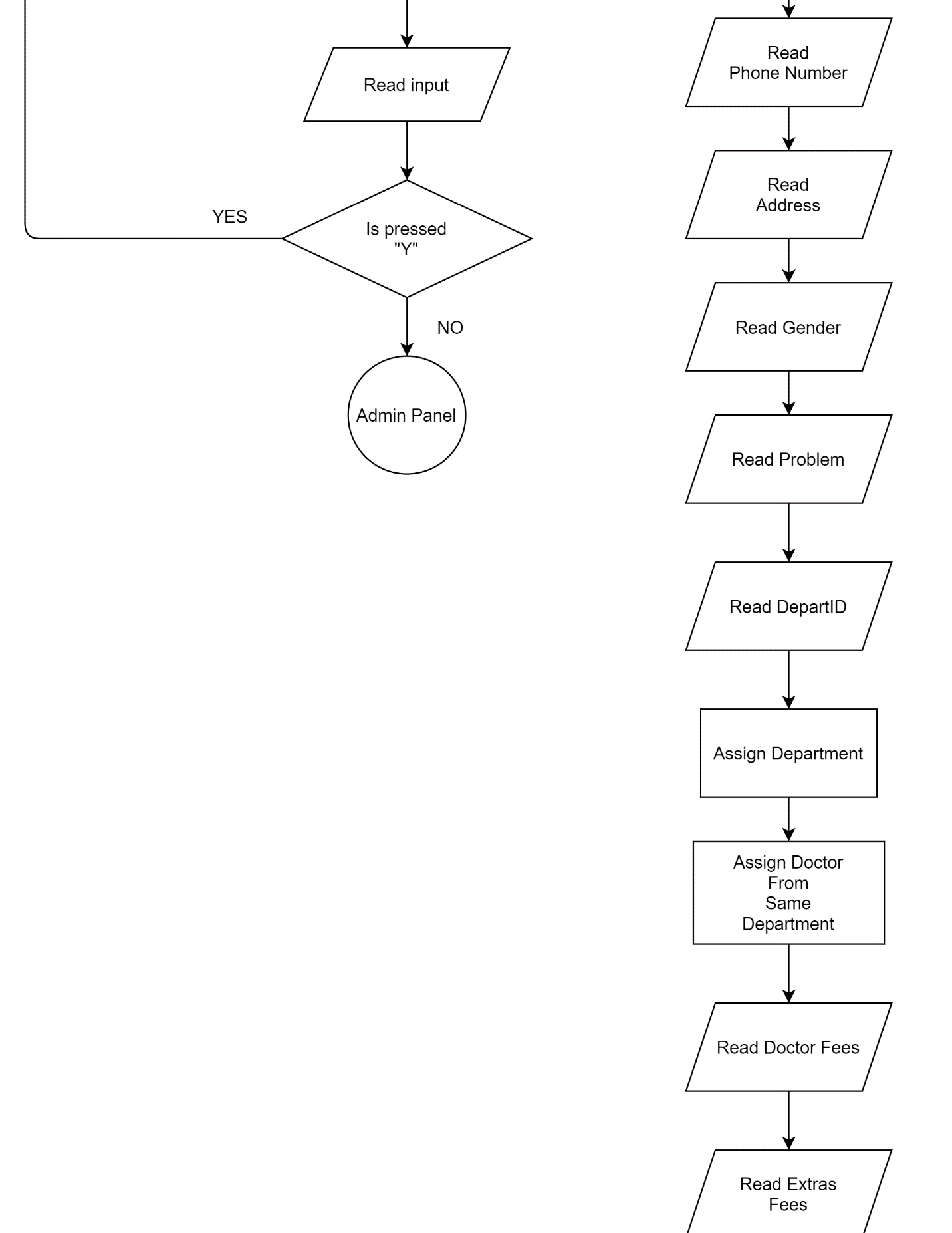
****

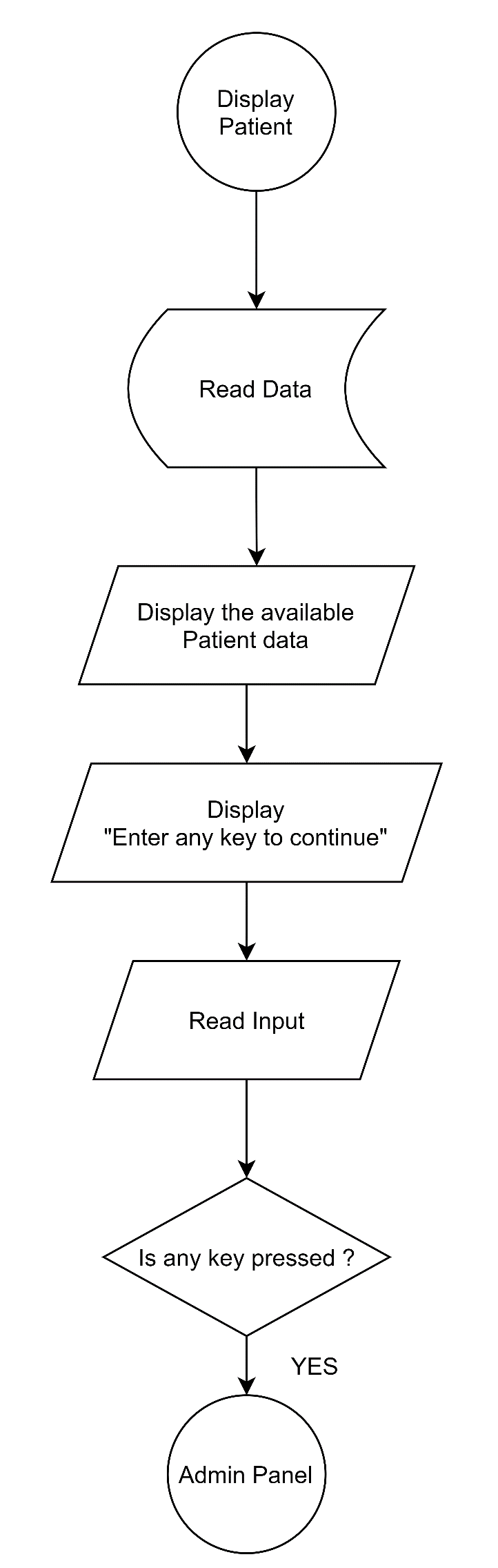
****

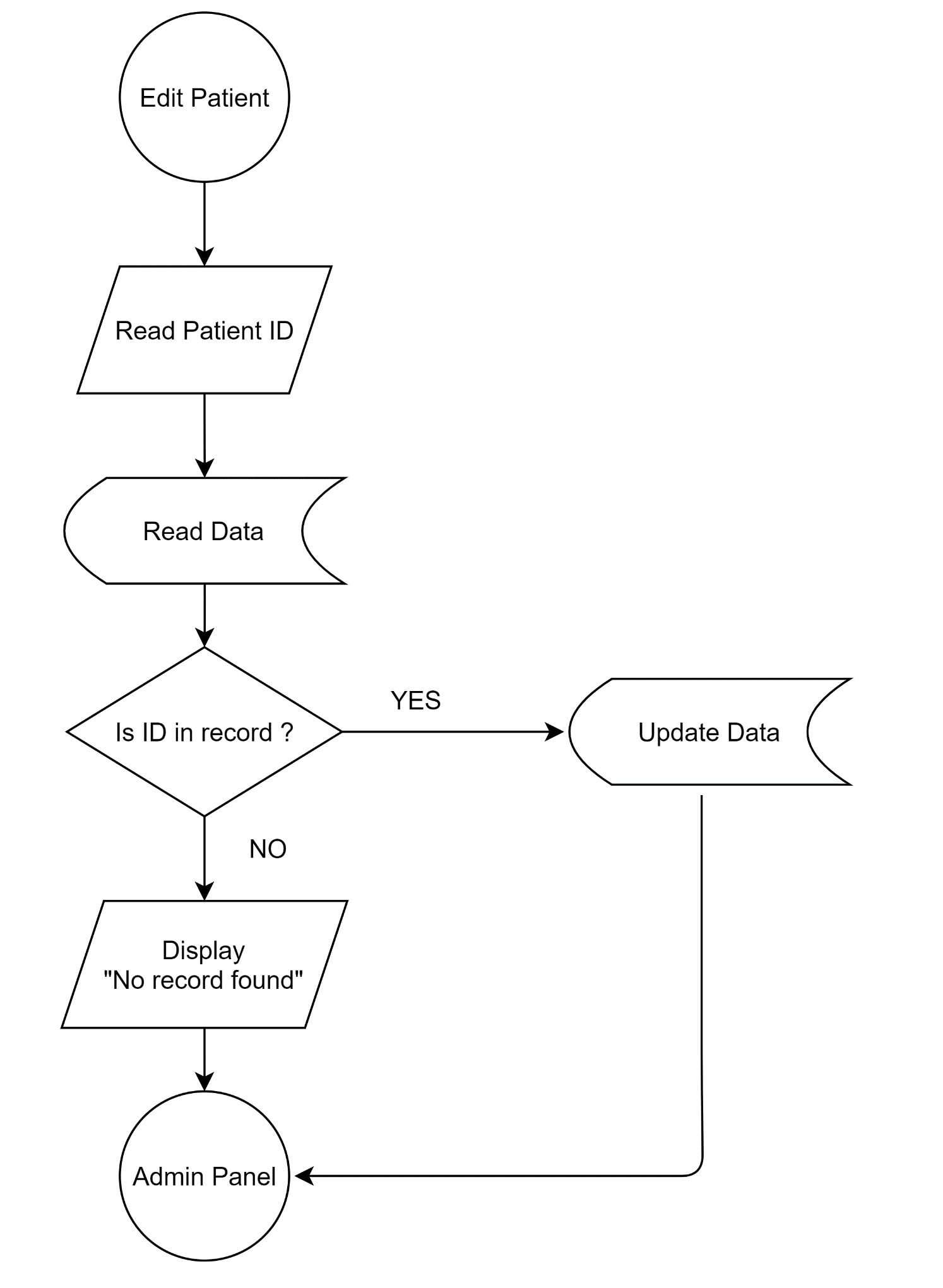
****

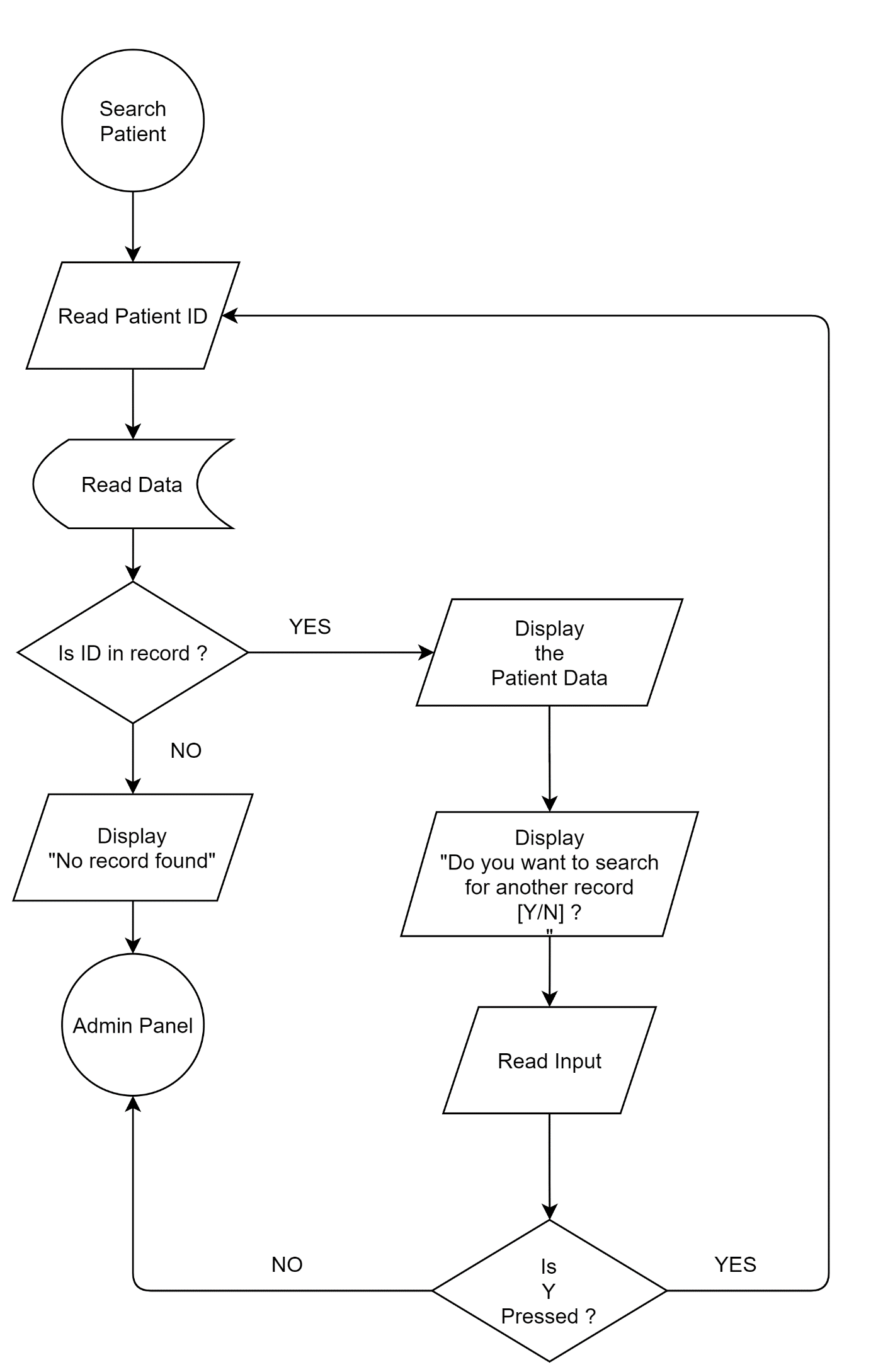
****

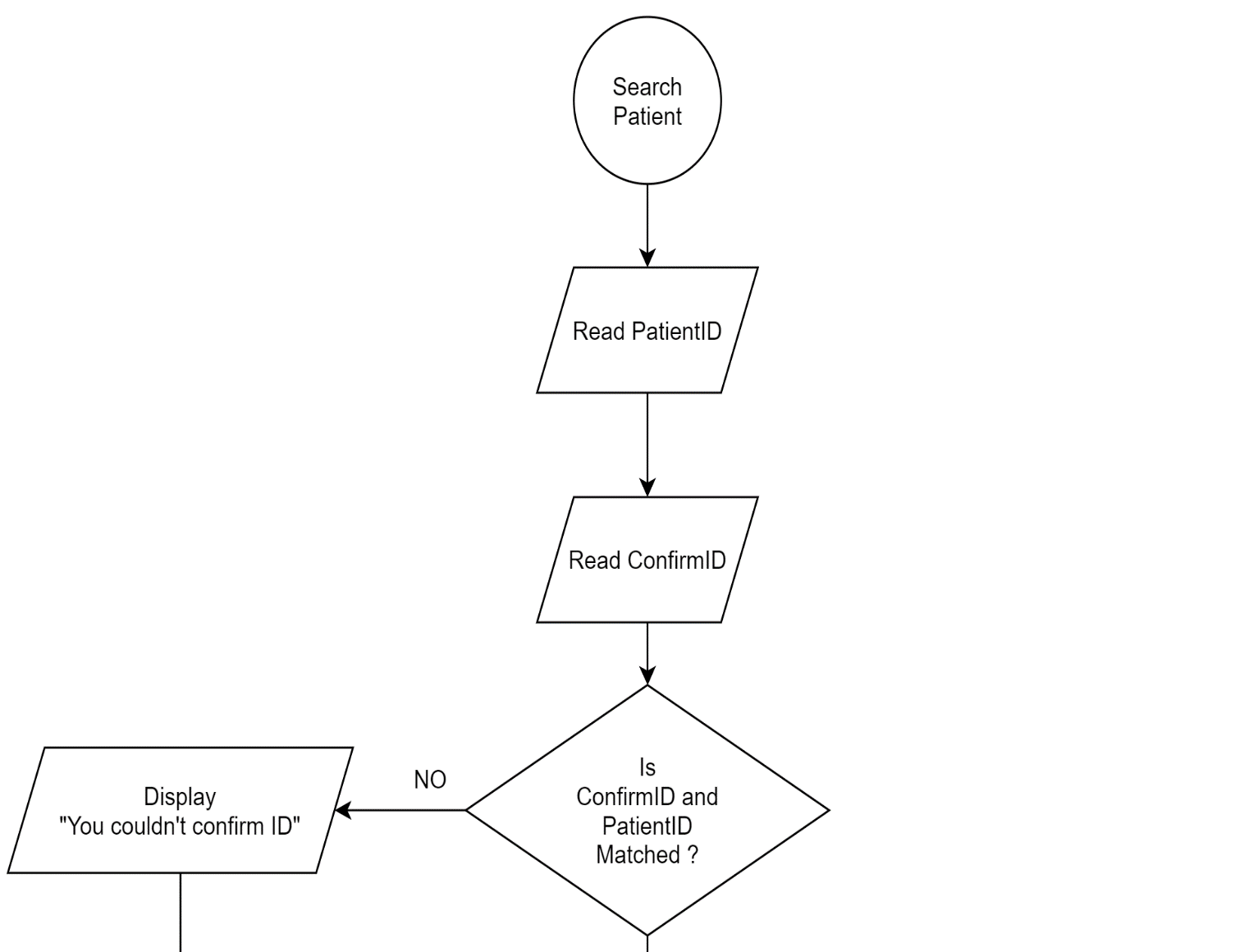


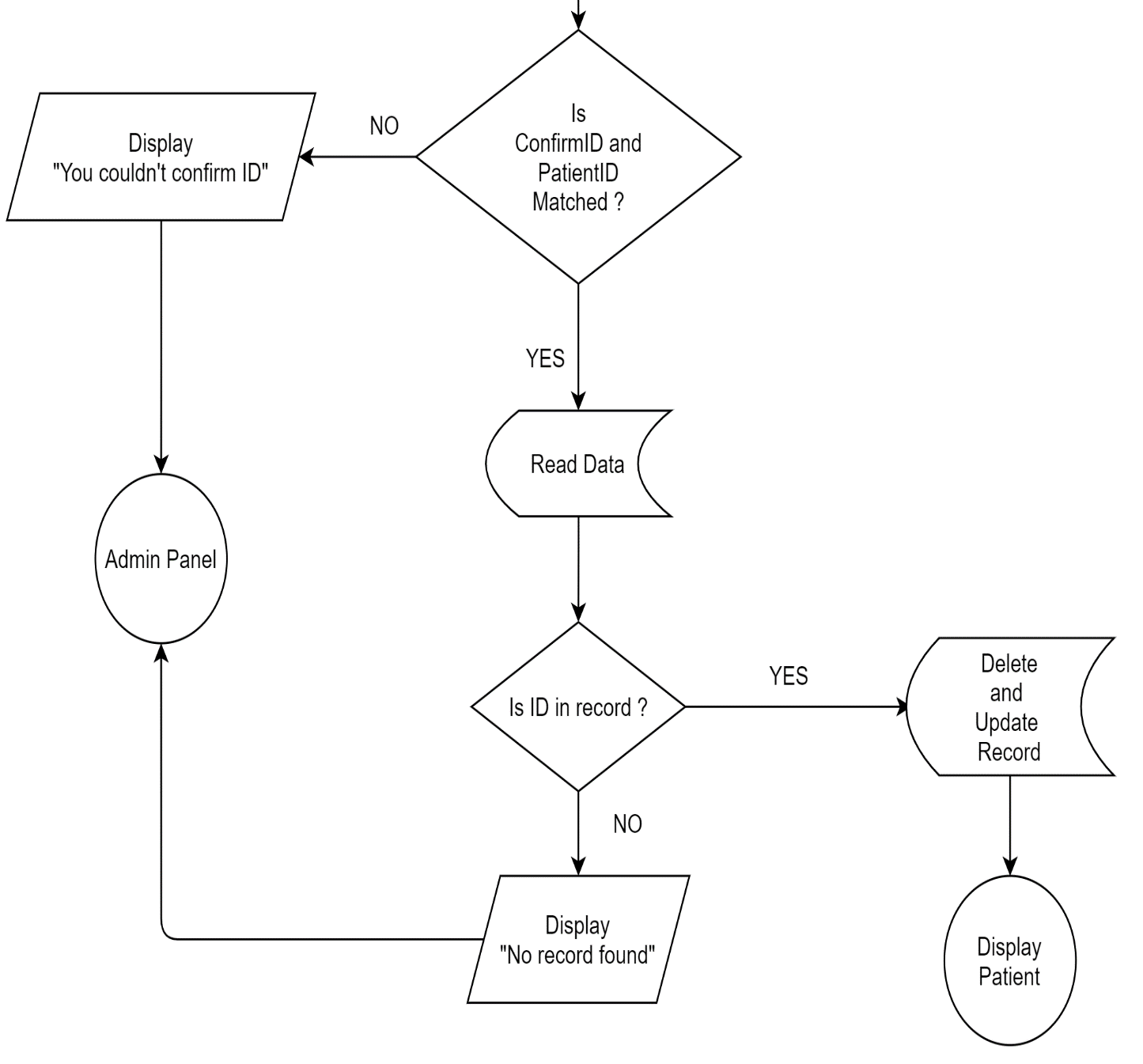












**5.4 Coding**

#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;

}

struct patient {

// defining structure for patient

char firstName[20], lastName[20];

char add[100],sex[10], problem[20],depart[20], consultant[10], registeredDate[10];

int patientno,age;

char phone[12];

float bed,doc,misc;

};

struct patient p;

int main () {

fullScreen();

mainMenu();

}

//Functions

int get\_size(const char\* file\_name) {

FILE \*file = fopen(file\_name, "r");

if(file == NULL)

return 0;

fseek(file, 0, SEEK\_END);

int size = ftell(file);

fclose(file);

return size;

}

//Password Encryption

void encrypt(char password[], int key) {

unsigned int i;

for(i=0; i<strlen(password); ++i) {

password[i] = password[i] - key;

}

}

void welcomeScreen() {

system("cls");

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

printf("\n\n\t\t\t\t\t\t\t\t WELCOME TO \t");

printf("\n\t\t\t\t\t\t\t\tPATIENT MANAGEMENT SYSTEM\t");

printf("\n\t\t\t\t\t\t\t\t ABC HOSPITAL \t\t");

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

}

void credit() {

welcomeScreen();

gotoxy(59,14);

printf("SPECIAL THANKS TO PROJECT'S CONTRIBUTORS");

gotoxy(66,16);

printf("Sakar Aryal - Team Leader");

gotoxy(69,18);

printf("Pawan Chaudhary");

gotoxy(69,20);

printf("Sandhya Khadka");

gotoxy(70,22);

printf("Himal Aryal");

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

printf("\n\n\t\t\t\t\t\t\tFor Source Code: Visit github.com/techiesakar");

printf("\n\n\t\t\t\t\t\t\t\tPress any key to continue...");

getch();

mainMenu();

}

void thankyou() {

system("cls");

printf("\n\n\n\n\n\n\n\n\n\t\t\t\t\t\t\t...THANK YOU FOR VISITING...");

printf("\n\n\t\t\t\t\t\tFor Source Code: Visit github.com/techiesakar");

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

printf("\n\n\t\t\t\t\t\t ...Press any key to close the program...");

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

getch();

}

void adminRegistration() {

// If user exist, then login if not then register user

char username[20], password[20], petName[20];

FILE \*pass;

if ((pass = fopen("password.txt", "r")) == NULL || get\_size("password.txt")==0 ) {

pass = fopen("password.txt", "w");

welcomeScreen();

printf("\n\n\t\t\t\t\t\t\tUpdating New Details");

printf("\n\n\t\t\t\t\t\t\tUsername : ");

scanf("%s", username);

// For Password

int i = 0;

char ch;

printf("\n\t\t\t\t\t\t\tPassword : ");

while(1) {

ch = getch();

if(ch==ENTER) {

password[i]='\0';

break;

} else if (ch==BKSP) {

if(i>0) {

i--;

printf("\b \b");

}

} else if (ch==TAB || ch==SPACE) {

continue;

} else {

password[i]=ch;

i++;

printf("\*");

}

}

encrypt(password,0xFACA);

printf("\n\n\t\t\t\t\t\t\tEnter Security Questions Answer");

printf("\n\n\t\t\t\t\t\t\tPet's Name : ");

scanf("%s", petName);

fprintf(pass, "%s\t%s\t%s",username,password,petName);

fclose(pass);

printf("\n\n\t\t\t\t\t\t\tACCOUNT CREATED SUCCESSFULLY");

printf("\n\n\t\t\t\t\t\t\t....Enter any key to continue... ");

getch();

mainMenu();

}

else {

int count = 4;

do {

int i = 0;

char ch;

char temp\_username[30];

char temp\_password[30];

welcomeScreen();

if(count<4) {

printf("\n\n\t\t\t\t\t\t\tYou've %d more attempts left : ", count);

}

printf("\n\n\t\t\t\t\t\t\tEnter Username : ");

scanf("%s", temp\_username);

printf("\n\t\t\t\t\t\t\tEnter Password: ");

while(1) {

ch = getch();

if(ch==ENTER) {

temp\_password[i]='\0';

break;

} else if (ch==BKSP) {

if(i>0) {

i--;

printf("\b \b");

}

} else if (ch==TAB || ch==SPACE) {

continue;

} else {

temp\_password[i]=ch;

i++;

printf("\*");

}

}

encrypt(temp\_password,0xFACA);

pass=fopen("password.txt","r");

fscanf(pass,"%s\t%s%s",username, password, petName);

if(strcmp(password, temp\_password)==0 && strcmp(username,temp\_username)==0) {

adminPanel();

break;

} else {

printf("\n\n\t\t\t\t\t\tOops...Either username or password may incorrect");

printf("\n\n\t\t\t\t\t\tEnter any key to continue... ");

getch();

}

count--;

if(count<=0) {

mainMenu();

}

} while(count>0);

}

fclose(pass);

} // Registration ends

void forgetPassword() {

// Ask username and pet's name to rest password

int count = 4;

do {

char username[20], password[20], petName[20], temp\_username[20], temp\_petName[20];

FILE \*pass;

pass=fopen("password.txt","r");

welcomeScreen();

fscanf(pass,"%s\t%s\t%s",username, password, petName);

if(count<4) {

printf("\n\n\t\t\t\t\t\t\tYou've %d more attempt left", count);

}

printf("\n\n\t\t\t\t\t\t\tConfirm Username : ");

scanf("%s", temp\_username);

printf("\n\t\t\t\t\t\t\tConfirm your pet's name : ");

scanf("%s", temp\_petName);

if(strcmp(temp\_petName, petName)==0&&strcmp(temp\_username,username)==0) {

fclose(pass);

pass=fopen("password.txt","w");

fclose(pass);

adminRegistration();

break;

} else {

printf("\n\n\t\t\t\t\t\t\tSorry !! Detailed not matched");

printf("\n\n\t\t\t\t\t\t\t...Enter any key to continue...");

getch();

}

count--;

if(count<=0) {

fclose(pass);

mainMenu();

break;

}

} while(count>0);

}

void changePassword() {

// Erase existing record and register account again

FILE \*pass;

pass=fopen("password.txt","w");

fclose(pass);

adminRegistration();

}

void adminPanel() {

// Display all the available options for admin

welcomeScreen();

int choice;

printf("\n\n\n\t\t\t\t1. Add Record");

printf("\n\n\t\t\t\t2. Display Record");

printf("\n\n\t\t\t\t3. Update Record");

printf("\n\n\t\t\t\t4. Search Record");

printf("\n\n\t\t\t\t5. Delete Record");

printf("\n\n\t\t\t\t6. Update Admin's Details");

printf("\n\n\t\t\t\t7. Terminate Program");

printf("\n\n\t\t\t\t0. Logout");

printf("\n\n\t\t\t\tEnter your choice... ");

int ch;

scanf("%d", &ch);

switch(ch) {

case 1:

addRecord();

break;

case 2:

viewRecord();

break;

case 3:

editRecord();

break;

case 4:

searchRecord();

break;

case 5:

deleteRecord();

break;

case 6:

changePassword();

break;

case 0:

mainMenu();

break;

default:

thankyou();

break;

}

}

void mainMenu() {

// This will show before getting logged in

welcomeScreen();

int choice;

printf("\n\n\n\t\t\t\t1. Register/Login");

printf("\n\n\t\t\t\t2. Forget Password");

printf("\n\n\t\t\t\t3. Credit");

printf("\n\n\t\t\t\t4. Exit Program");

printf("\n\n\t\t\t\tEnter your choice... ");

int ch;

scanf("%d", &ch);

switch(ch) {

case 1:

adminRegistration(); // Register or Login

break;

case 2:

if(get\_size("password.txt")==0) {

welcomeScreen();

printf("\n\n\n\t\t\t\t\t\t\tYou've not registered yet");

printf("\n\n\t\t\t\t\t\t ...Enter any key to continue... ");

getch();

mainMenu();

} else

forgetPassword();

break;

case 3:

credit();

break;

case 4:

thankyou();

break;

default:

fflush(stdin);

mainMenu();

break;

}

}

void addRecord() {

// Add patient - only available for logged in user

system("cls");

welcomeScreen();

char ans;

FILE \*fp;

fp = fopen("patient.txt","a+"); // open file in append mode

if (!fp) {

printf("\n\t Can not open file\n");

exit(0);

}

// Auto assign unique patient ID for new patient

int patientID;

int count = 0;

while(fscanf(fp,"%s%s%s%s%d%d%s%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;

// Auto assign date to new user

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 );

// Date Ends

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%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\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) {

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","Dermatalogy","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\t\tFirst Name: ");

scanf("%s",p.firstName);

p.firstName[0]=toupper(p.firstName[0]);

printf("\n\t\t\t\tLast Name: ");

scanf("%s",p.lastName);

p.lastName[0]=toupper(p.lastName[0]);

printf("\n\t\t\t\tAddress: ");

scanf("%s",p.add);

p.add[0]=toupper(p.add[0]);

printf("\n\t\t\t\tGender[M|F]:");

scanf("%s",p.sex);

p.sex[0]=toupper(p.sex[0]);

printf("\n\t\t\t\tAge: ");

scanf("%d",&p.age);

printf("\n\t\t\t\tPhone Number: ");

scanf("%s",p.phone);

printf("\n\t\t\t\tProblem: ");

scanf("%s",p.problem);

printf("\n\t\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\tDoctor's Charge:");

scanf("%f",&p.doc);

printf("\n\t\t\t\tMiscellaneous Charge:");

scanf("%f",&p.misc);

} // addRecordItem ends

void tableHead() {

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("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

} // Heading of records

int listLoopRow(int row) {

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() {

int totalMember;

int row = 20;

system("cls");

FILE \*fp;

if((fp=fopen("patient.txt","r"))==NULL) {

welcomeScreen();

printf("\n\n\t\t\t\t Enter any key to continue...");

getch();

welcomeScreen();

}

welcomeScreen();

tableHead();

while(fscanf(fp,"%s%s%s%s%d%d%s%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) {

listLoopRow(row);

row++;

}

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

fclose(fp);

printf("\n\n\t\t\t\t Enter any key to continue...");

getch();

welcomeScreen();

adminPanel();

} // ViewRecord Ends

void searchRecord(void) {

welcomeScreen();

int searchID;

FILE \*fp;

int row = 20;

char ans;

int count = 0; // It is used to show total available records

fp = fopen("patient.txt","r"); // open file in read mode

printf("\n\n\n\t\t\t\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ABC Hospital - Search Patient \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

gotoxy(12,16);

fflush(stdin);

printf("\n\t\t\t\t\t Enter Patient's ID : ");

scanf("%i",&searchID); // Takes user ID for search

welcomeScreen(); // Once data read, clearing screen again

printf("\n\n\n\t\t\t\t\t\t\t\tPatient Searched For : %i", searchID );

tableHead();

while(fscanf(fp,"%s%s%s%s%d%d%s%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(p.patientno == searchID) {

listLoopRow(row);

row++;

count++;

}

}

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

printf("\n\n\t\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() {

welcomeScreen();

int searchID;

int valid = 0;

FILE \*fp, \*tf;

int row = 20;

char ans;

printf("\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 File is empty...");

printf("\n\n\t\t\t\tEnter any key to continue...");

getch();

welcomeScreen();

adminPanel();

}

gotoxy(12,16);

fflush(stdin);

printf("\n\t\t\t\t\t Enter Patient's ID : ");

scanf("%i",&searchID);

welcomeScreen();

if (tf==NULL) {

printf("\n\n\t\t\t\t\t Cannot Open File !!");

adminPanel();

}

while(fscanf(fp,"%s%s%s%s%d%d%s%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(p.patientno == searchID) {

valid = 1;

welcomeScreen();

printf("\n\n\n\t\t\t\t\t\t\t\tPatient Searched For : %i", searchID );

tableHead();

listLoopRow(row);

row++;

printf("\n\n\t\t\t\tEnter any key to continue...");

getch();

welcomeScreen();

addRecordItem(); fprintf(tf,"%s\t%s\t%s\t%s\t%d\t%d\t%s\

t%s\t%s\t%s\t%f\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\t\tUpdated Successfully");

} else {

fprintf(tf,"%s\t%s\t%s\t%s\t%d\t%d\t%s\t%s\t%s\t%s\t%f\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);

}

}

if(!valid) {

printf("\n\n\t\t\t\tNo record found");

}

fclose(tf);

fclose(fp);

remove("patient.txt");

rename("temp\_patient.txt","patient.txt");

getch();

adminPanel();

}

// editRecord Ends

void deleteRecord() {

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\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ABC Hospital - Delete Patient's Record \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

if((fp=fopen("patient.txt","r"))==NULL) {

welcomeScreen();

printf("\n\n\t\t\t\t File is empty...");

printf("\n\n\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\t Enter Patient's ID : ");

scanf("%i",&searchID);

printf("\n\n\t\t\t\t\t Enter ID again to confirm : ");

scanf("%i", &confirmID);

if(searchID==confirmID) {

confirm = true;

while(fscanf(fp,"%s%s%s%s%d%d%s%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%s\t%s\t%s\t%s\t%f\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);

} 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\t\t Record not found ");

}

printf("\n\n\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 6**

**Snapshots**

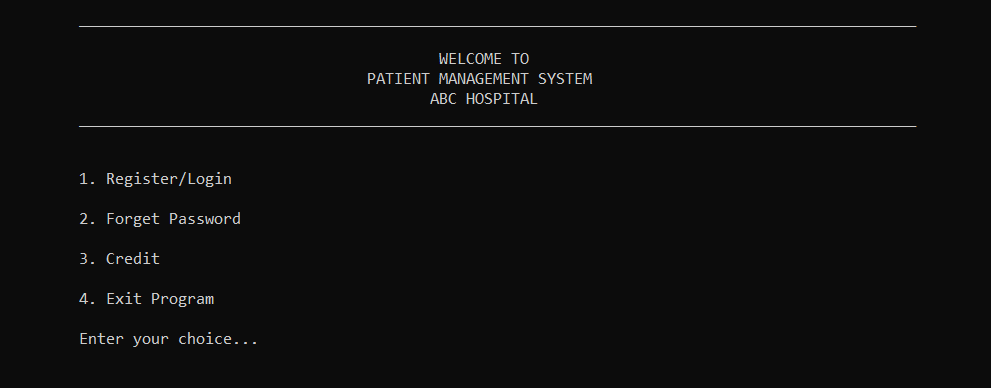


Fig: Main Menu



Fig: Admin Panel



Fig: Add Record

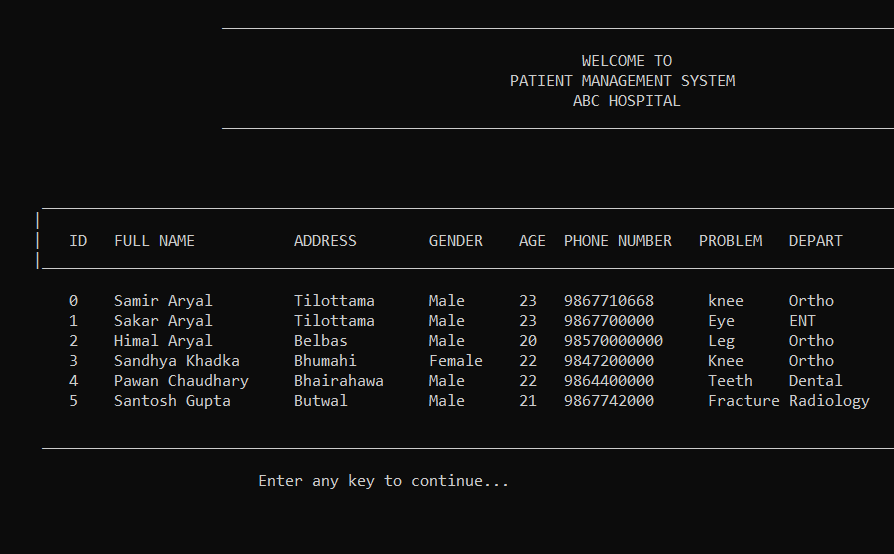
****

Fig: View Record



Fig: Credit

**7.1: Conclusion:**

A login-based Outpatient Management System using the C-Programming Language that enables a Hospital/Medical to maintain its records 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.

7.2: **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