**Hospital Management System**



Session: 2021 – 2024

**Submitted by:**

Muhammad Nazir 2021-CS-8

**Supervised by:**

Sir Awais

Department of Computer Science

**University of Engineering and Technology**

**Lahore Pakistan**

**Short Description of my Project:**

There are four options in my project main menu.

1. Admin
2. Patient
3. Receptionist
4. Exit

This Hospital Management System is beneficial for the admin and patients. It contains some important features as**; sorting, recommendation and take appointments. I also use validity feature at some places as gender, age, blood group etc.**

**Patients** can enter their personal details as well as medical history. This can be seen by the receptionist in his menu. The patient can also view his personal and medical history. He can see the available doctors in hospitals and also view their timing of presence in hospital. He can also take appointments. My project will be beneficial for taking appointments. Program will tell the patient that when he will take the appointments. My program will recommend the patient that which doctor is specific for which disease. The patient can **give feedback** which admin can see. If two patients give feedback, then two feedback shows to admin.

In this management system, **admin** can add new patient and view them. If he selects view option before adding the patient, then there shows clear screen without any garbage value. The main thing about my project is that, there is sorting feature in it. Admin can see staff’s record, death records, birth records and room availability.

In this project, receptionist can see the entered personal and medical details of patients.

My program will **terminate** only when user select only the exit option.

**Users of Application**

There are three users in my application.

* Admin
* User
* Receptionist

**Functional Requirements of Hospital Management System:**

**For User(Patient):**

1. As a user, I can enter my **personal details** which receptionist can see.
2. As a user, I can see my entered personal details.
3. As a user, I can add my medical history.
4. As a user, I can see my **medical history which** receptionist can see.
5. As a user, I can see the list of **available doctor** in hospital.
6. As a user, I can **take appointments** from doctors of specific disease.
7. As a user, I can **see recommendation** about doctors.
8. As a user, I can **give feedback** which admin can see.
9. As a user, I can **exit the program**.

**For Admin:**

1. As an admin, I can add new patient.
2. As an admin, I can see all **patient’s record**.
3. As an admin, I can see all patient’s record in **sorted order**.
4. As an admin, I can see **staff record**.
5. As an admin, I can see birth records.
6. As an admin, I can see death records.
7. As an admin, I can give information about **room availability**.
8. As an admin, I can **see feedback** given by user.
9. As an admin, I can **exit the program**.

**For Receptionist:**

1. As a receptionist, I can see the **personal entered data** of patient.
2. As a receptionist, I can see the **medical history** of patient.
3. As an admin, I can **exit the program**.

**Functions working flow**

**Login menu**

**Receptionist menu**

**Patient menu**

**Exit**

**Admin menu**

c

**Function prototypes:**

**Mutually used Functions’ Prototypes:**

1. int different\_users\_login();
2. void logo();
3. void escape ();
4. void thanks();
5. void congratulation();
6. void incorrect\_role();
7. void invalid();

**For Admin Menu:**

1. string login\_manu();
2. int menu();
3. void addpatient();
4. void viewpatient( );
5. void viewstaff();
6. void birthreport();
7. void deathreport();
8. void roomrecord();
9. void feedback();
10. void sorting();
11. void datareader();
12. void patient\_count\_reader();

**For User(Patient) Menu:**

1. int menu1();
2. string login\_manu2();
3. void patient\_personal\_detail();
4. void patient\_medical\_detail();
5. void view\_my\_detail();
6. void view\_my\_medical\_detail();
7. void viewdoctors\_details();
8. void recommendation(string dis);
9. void get\_appointments(string dname , string day);
10. void datareader1();
11. void user\_count\_reader();
12. void datareader2();
13. void comment\_count\_reader();
14. void datareader3();
15. void count\_reader();

**For Receptionist Menu:**

1. string login\_manu3();
2. int menu2();
3. void receptionist\_view();
4. void receptionist\_view2();

**Data Structures:**

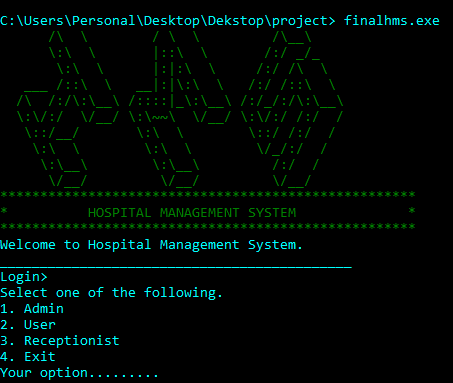
**Arrays:**

1. string patient\_name[max\_patient];
2. string patientf\_name[max\_patient];
3. string patient\_disease[max\_patient];
4. string patient\_gender[max\_patient];
5. string patient\_date[max\_patient];
6. float patient\_age[max\_patient];
7. int patient\_room[max\_patient];
8. string patient\_group[max\_patient];
9. string nameA[max\_patient];
10. string f\_nameA[max\_patient];
11. int ageA[max\_patient];
12. string genderA[max\_patient];
13. string diseaseA[max\_patient];
14. string blood\_groupA[max\_patient];
15. string dateA[max\_patient];
16. string comment[max\_patient];

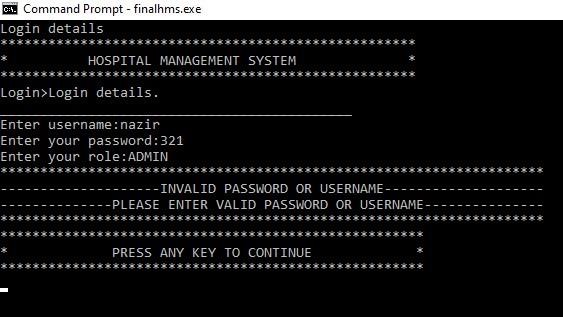
**Wireframes:**

**Admin:**

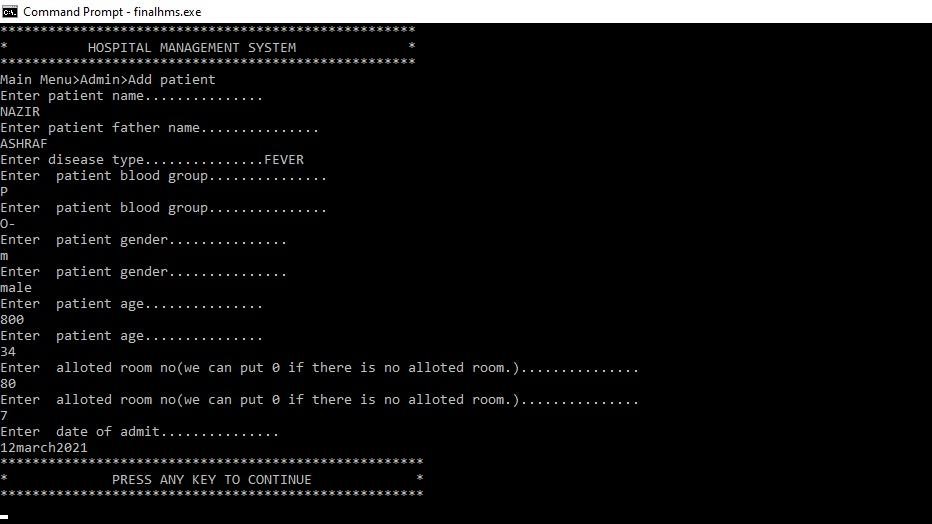
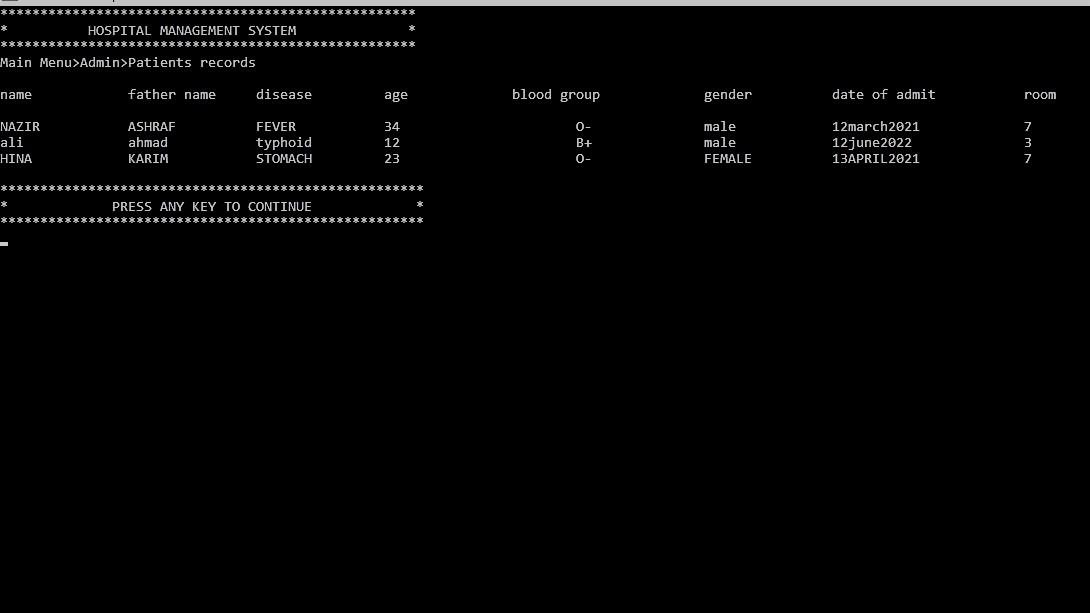
* Here you can select any option. Program will terminate only at option 4.

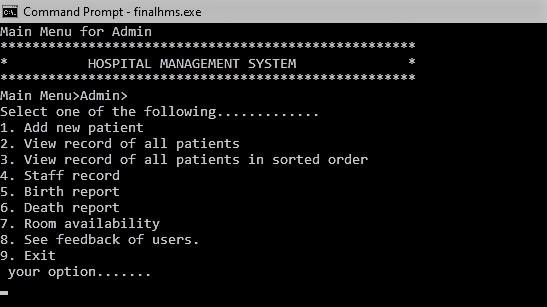
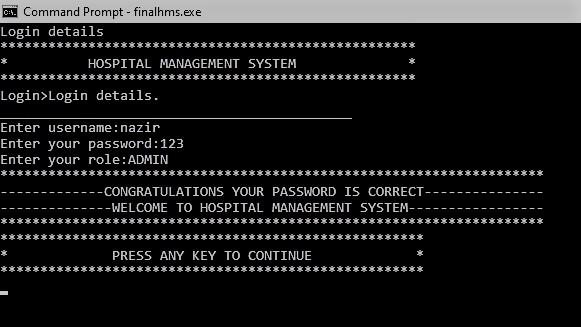


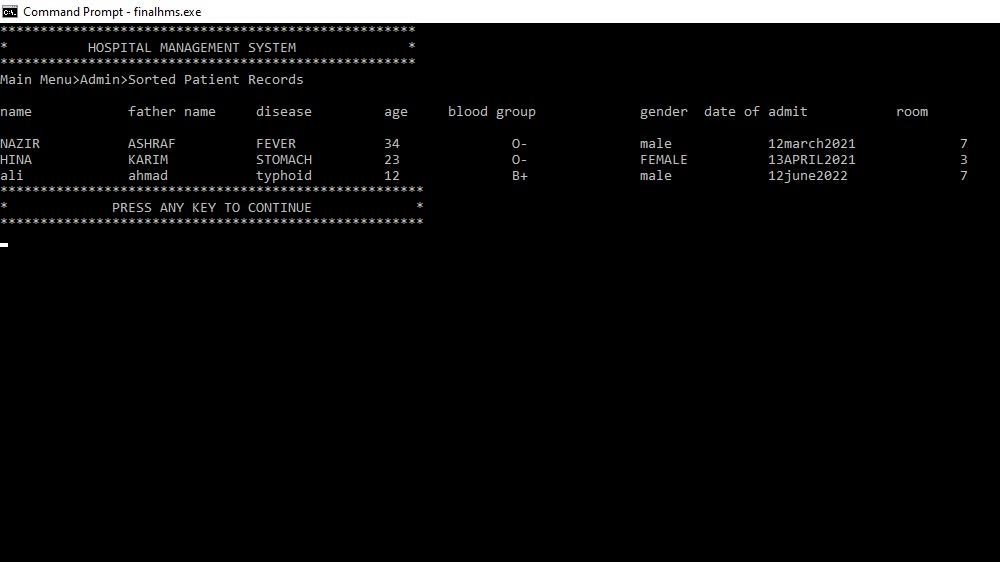
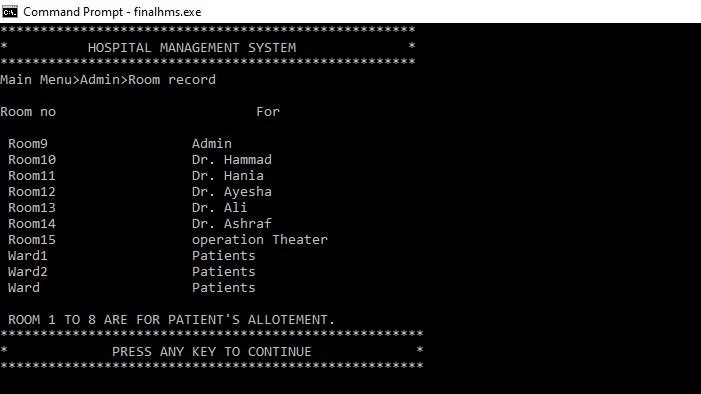
* If you put wrong password, username and role.

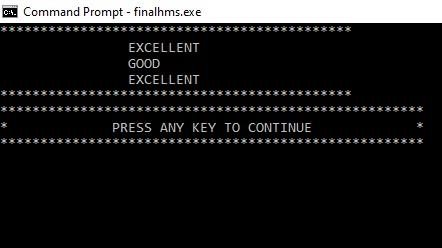


**Admin Login and Main Menu**

**Patient Personal Data**

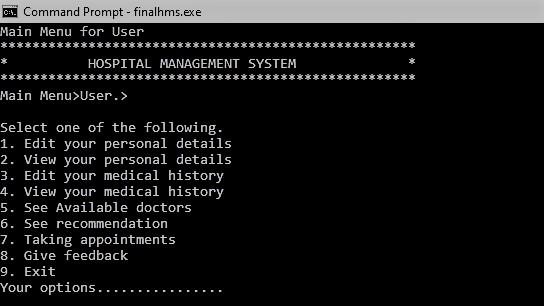


* **Sorted Data**
* **Birth Record:**
* **Death Record**
* **Room Availability**
* **See Screen**

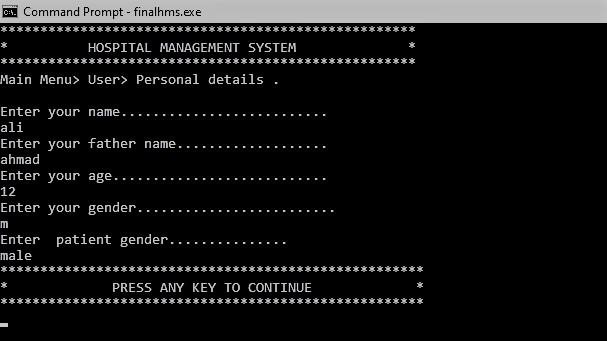


**Patient:**

* **Main Menu:**

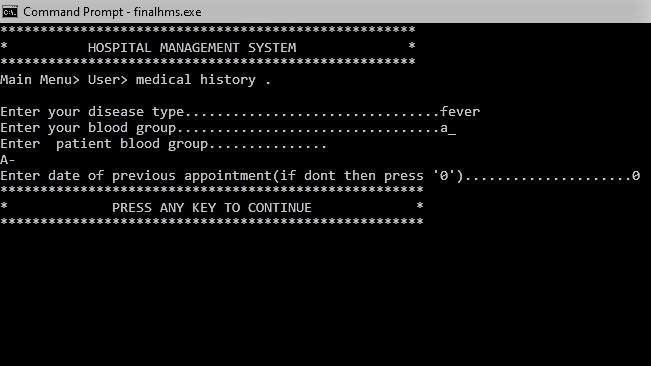


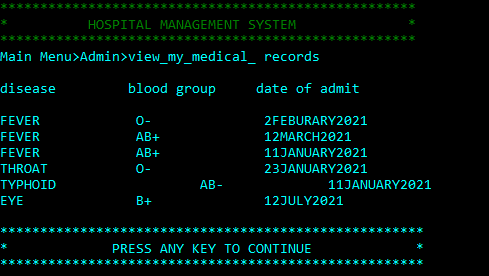
* **Personal Details:**



* **View Personal Details**

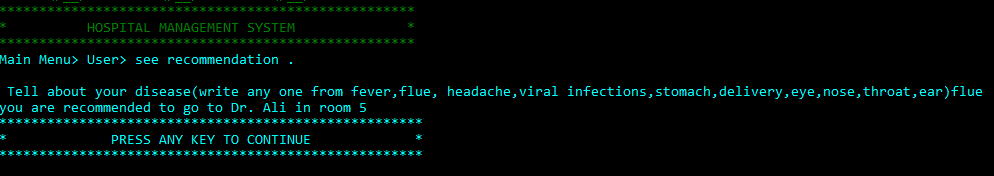


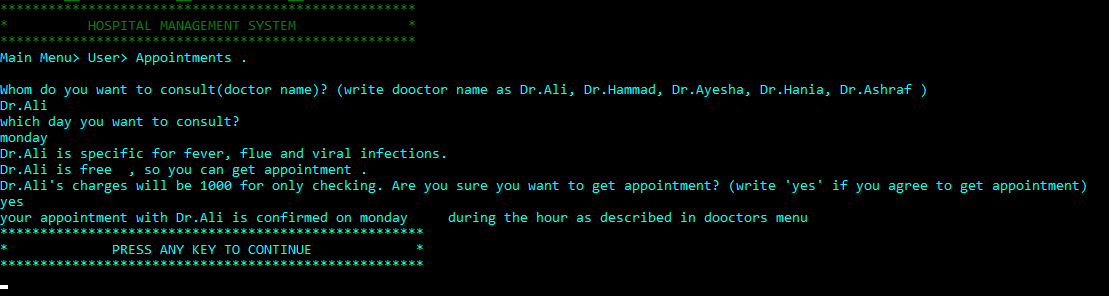
* **Medical History**

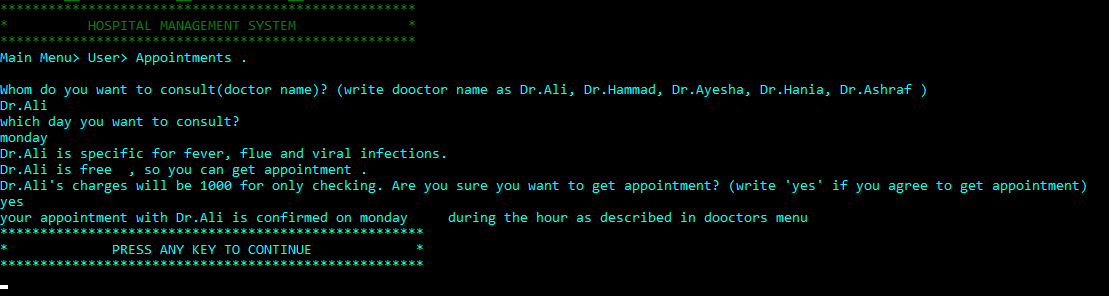
** View Medical History**

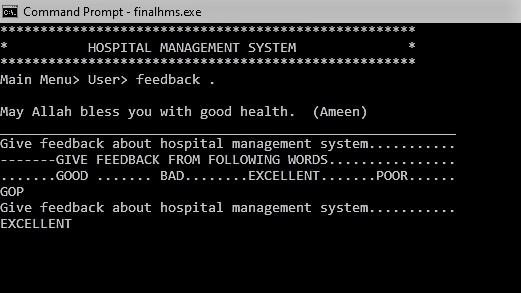
 **View Doctors Details**

**See Recommendation**



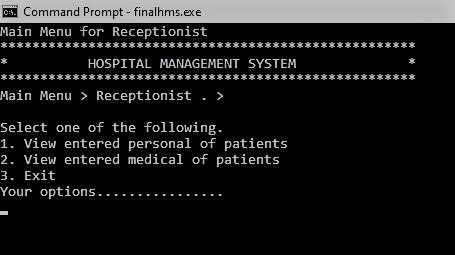
**Taking Appointments**

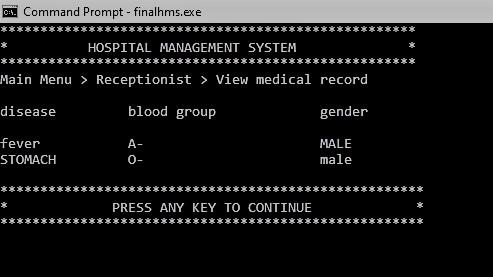
**Give Feedback**



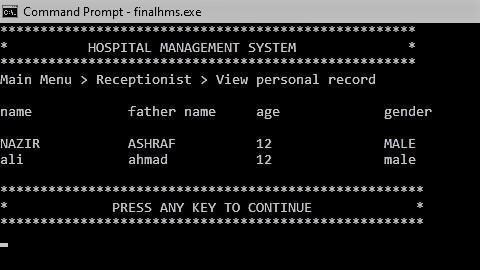
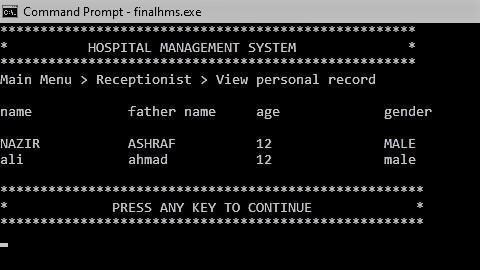
**Receptionist:**

**Main Menu**



**Medical Detail**

**Personal Record**



**COMPLETE CODE OF MY PROJECT:**

#include <iostream>

#include <fstream>

#include <conio.h>

#include <windows.h>

**using** **namespace** std;

/\*PROTOTYPES OF MY HOSPITAL MANAGEMENT SYSTEM ARE ...............................................................................\*/

/\* MUTUALLY USED PROTOTYPES OF FUNCTIONS.........................................................................................\*/

string **different\_users\_login**();

**void** **logo**();

**void** **escape**();

**void** **thanks**();

**void** **congratulation**();

**void** **incorrect\_role**();

**void** **invalid**();

/\* PROTOTYPES FOR ADMIN FUNCTIONS...............................................................................................\*/

string **login\_manu**();

string **menu**();

**void** **addpatient**();

**void** **viewpatient**( );

**void** **viewstaff**();

**void** **birthreport**();

**void** **deathreport**();

**void** **roomrecord**();

**void** **feedback**();

**void** **sorting**();

/\* PROTOTYPES FOR PATIENT FUNCTIONS..............................................................................................\*/

string **menu1**();

string **login\_manu2**();

**void** **patient\_personal\_detail**();

**void** **patient\_medical\_detail**();

**void** **view\_my\_detail**();

**void** **view\_my\_medical\_detail**();

**void** **viewdoctors\_details**();

**void** **recommendation**(string dis);

**void** **get\_appointments**(string dname , string day);

**void** **datareader1**();

/\* PROTOTYPES FOR RECEPTIONIST FUNCTIONS...............................................................................................\*/

string **login\_manu3**();

string **menu2**();

**void** **receptionist\_view**();

**void** **receptionist\_view2**();

/\*GLOBLY DECLARED VARIABLES OF HOSPITAL MANAGEMENT SYSTEM ARE........................................................................\*/

**const** **int** max\_patient = **20**;

**int** patient\_count ;

**int** user\_count ;

**int** count ;

**int** comment\_count ;

**int** start = **0**;

/\*ARREYS FOR MY HOSPITAL MANAGEMENT SYSTEM ARE.......................................................................................\*/

string patient\_name[max\_patient];

string patientf\_name[max\_patient];

string patient\_disease[max\_patient];

string patient\_gender[max\_patient];

string patient\_date[max\_patient];

**float** patient\_age[max\_patient];

**int** patient\_room[max\_patient];

string patient\_group[max\_patient];

string nameA[max\_patient];

string f\_nameA[max\_patient];

**float** ageA[max\_patient];

string genderA[max\_patient];

string diseaseA[max\_patient];

string blood\_groupA[max\_patient];

string dateA[max\_patient];

string comment[max\_patient];

// START OF FILE HANDLING............................

**void** **datareader**();

**void** **patient\_count\_reader**();

**void** **user\_count\_reader**();

**void** **datareader2**();

**void** **comment\_count\_reader**();

**void** **datareader3**();

**void** **count\_reader**();

// END OF FILE HANDLING .............................

//FOR COLORING......................................................................................................................

HANDLE h = GetStdHandle(STD\_OUTPUT\_HANDLE);

/\*MAIN FUNCTIONS OF MY HOSPITAL MANAGEMENT SYSTEM IS.................................................................................\*/

main(){

datareader();

patient\_count\_reader();

datareader1();

user\_count\_reader();

datareader2();

comment\_count\_reader();

datareader3();

count\_reader();

/\* VARIABLES OF HOSPITAL MANAGEMENT SYSTEM ARE........................................................................................\*/

string option1,option2,option3;

string option;

string role;

option = different\_users\_login();

**while**( true ){

**if** (option == "1" )

{

/\*TO REMOVE THE SCREEN OF PREVIOUS OPTION....................\*/

system ("cls");

role = login\_manu();

/\*START OF ADMIN OPTIONS...........................................................................................................................\*/

**if** ( role == "ADMIN" )

{

congratulation();

escape();

// to remove the previous screen

system ("cls");

option1= menu();

// VALIDATION................................................

**while**( option1 != "1" && option1 != "2" && option1 != "3" && option1 != "4" && option1 != "5" && option1 != "6"

&& option1 != "7" && option1 != "8" && option1 != "9" ){

system ("cls");

logo();

cout << "YOU SELECTED WRONG OPTION." << endl;

escape();

system ("cls");

option1 = menu();

**if** ( option1 == "1" || option1 == "2" || option1 == "3" || option1 == "4" || option1 == "5" || option1 == "6"

|| option1 == "7" || option1 == "8" || option1 == "9" )

{

**continue**;

}

**break**;

}

// END OF VALIDATION................................................

//USE OF WHILE LOOP;

**while** (option1 == "1" || option1 == "2" || option1 == "3" || option1 == "4" || option1 == "5" || option1 == "6"

|| option1 == "7" || option1 == "8" || option1 == "9" ){

**if** (option1 == "1")

{

system ("cls");

addpatient();

option1= menu();

}

**if** (option1 == "2")

{

system("cls");

viewpatient();

system("cls");

option1= menu();

}

**if** (option1 == "3")

{

system("cls");

sorting();

system("cls");

option1= menu();

}

**if** (option1 == "4")

{

system("cls");

viewstaff();

system("cls");

option1= menu( );

}

**if** (option1 == "5")

{

system ("cls");

birthreport();

system ("cls");

option1= menu( );

}

**if** (option1 == "6")

{

system ("cls");

deathreport();

system ("cls");

option1= menu( );

}

**if** (option1 == "7")

{

system ("cls");

roomrecord();

system ("cls");

option1= menu( );

}

**if** (option1 == "8")

{

system ("cls");

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

**for**(**int** i = **0** ; i <= comment\_count ; i++ ){

cout << "**\t\t**" << comment[i] << "**\t\t**" << endl;

}

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

escape();

system ("cls");

option1= menu( );

}

**if** (option1 == "9")

{

system ("cls");

thanks();

escape();

system ("cls");

option = different\_users\_login();

**break**;

}

}

}

// IF USER DOES NOT ENTERED CORRECT ROLE

**else** **if**(role!="ADMIN"){

incorrect\_role();

escape();

system ("cls");

option = different\_users\_login();

}

}

/\*END OF ADMIN OPTIONS...........................................................................................................................\*/

/\*START OF PATIENT OPTIONS...........................................................................................................................\*/

**if** (option == "2" )

{

system ("cls");

role = login\_manu2();

**if** (role == "PATIENT")

{

congratulation();

escape();

system ("cls");

//VALIDATION................................................

option2= menu1( );

**while**( option2 != "1" && option2 != "2" && option2 != "3" && option2 != "4" && option2 != "5" && option2 != "6"

&& option2 != "7" && option2 != "8" && option2 != "9" ){

system ("cls");

logo();

cout << "YOU SELECTED WRONG OPTION." << endl;

escape();

system ("cls");

option2 = menu1();

**if** ( option2 == "1" || option2 == "2" || option2 == "3" || option2 == "4" || option2 == "5" || option2 == "6"

|| option2 == "7" || option2 == "8" || option2 == "9" )

{

**continue**;

}

**break**;

}

// END OF VALIDATION................................................

**while** (option2 == "1" || option2 == "2" || option2 == "3" || option2 == "4" || option2 == "5" || option2 == "6"

|| option2 == "7" || option2 == "8" || option2 == "9" ){

**if** ( option2 == "1" )

{

system ("cls");

patient\_personal\_detail();

system("cls");

option2= menu1( );

}

**if** (option2 == "2" )

{

system ("cls");

view\_my\_detail();

system ("cls");

option2= menu1( );

}

**if** ( option2 == "3" )

{

system ("cls");

patient\_medical\_detail();

system ("cls");

option2= menu1( );

}

**if** (option2 == "4" )

{

system ("cls");

view\_my\_medical\_detail();

system ("cls");

option2= menu1();

}

**if** ( option2 == "5" )

{

system ("cls");

viewdoctors\_details();

system ("cls");

option2= menu1( );

}

**if** ( option2 == "6" )

{

system ("cls");

logo();

cout << "Main Menu> User> see recommendation ." << endl;

cout << " " << endl;

string disease;

cout << " Tell about your disease(write any one from fever,flue, headache,viral infections,stomach,delivery,eye,nose,throat,ear)";

cin >> disease;

recommendation( disease) ;

system ("cls");

option2= menu1( );

}

**if** ( option2 == "7" )

{

system ("cls");

logo();

cout << "Main Menu> User> Appointments ." << endl;

cout << " " << endl;

string doc\_name;

string c\_day;

cout << "Whom do you want to consult(doctor name)? (write dooctor name as Dr.Ali, Dr.Hammad, Dr.Ayesha, Dr.Hania, Dr.Ashraf )" << endl;

cin >> doc\_name;

**while** (doc\_name != "Dr.Ali" && doc\_name != "Dr.Hammad" && doc\_name != "Dr.Ayesha" && doc\_name != "Dr.Hania"

&& doc\_name != "Dr.Ashraf" ){

cout << "YOU ENTERED WRONG INPUT." << endl;

cout << "Whom do you want to consult(doctor name)? (write dooctor name as Dr.Ali, Dr.Hammad, Dr.Ayesha, Dr.Hania, Dr.Ashraf )" << endl;

cin >> doc\_name;

}

cout << "which day you want to consult?" << endl;

cin >> c\_day;

**while**(c\_day != "sunday"&& c\_day != "monday"&& c\_day != "tuesday"&& c\_day != "wednesday"&& c\_day != "friday"&&

c\_day != "thursday" ){

cout << "YOU ENTERED WRONG INPUT." << endl;

cout << "which day you want to consult?" << endl;

cin >> c\_day;

}

get\_appointments( doc\_name , c\_day);

system ("cls");

option2= menu1( );

}

**if**( option2 == "8" )

{

system ("cls");

feedback();

system ("cls");

option2= menu1( );

}

**if**( option2 == "9" )

{

system ("cls");

thanks();

escape();

system ("cls");

option = different\_users\_login();

**break**;

}

}

}

**else**{

incorrect\_role();

escape();

system ("cls");

option = different\_users\_login();

}

}

/\*END OF PATIENT OPTIONS...........................................................................................................................\*/

/\*START OF RECEPTIONIST OPTIONS...........................................................................................................................\*/

**if** (option == "3" )

{

system ("cls");

role = login\_manu3();

**if** (role == "RECEPTIONIST")

{

congratulation();

escape();

system ("cls");

option3= menu2( );

// VALIDATION................................................

**while**( option3 != "1" && option3 != "2" && option3 != "3" ){

system ("cls");

cout << "YOU SELECTED WRONG OPTION." << endl;

escape();

system ("cls");

option3 = menu2();

**if** ( option3 == "1" || option3 == "2" || option3 == "3" )

{

**continue**;

}

**break**;

}

// END OF VALIDATION................................................

**while** (option3 == "1" || option3 == "2" || option3 == "3" ){

**if** (option3 == "1"){

system("cls");

receptionist\_view();

system("cls");

option3= menu2( );

}

**if** (option3 == "2"){

system("cls");

receptionist\_view2();

system("cls");

option3= menu2( );

}

**if** (option3 == "3"){

system ("cls");

thanks();

escape();

system ("cls");

option = different\_users\_login();

**break**;

}

}

}

**else** **if**(role!="RECEPTIONIST"){

incorrect\_role();

escape();

system ("cls");

option = different\_users\_login();

}

}

/\*END OF RECEPTIONIST OPTIONS...........................................................................................................................\*/

/\*START OF EXIT OPTION...........................................................................................................................\*/

**if** (option == "4"){

thanks();

**break**;

}

/\*END OF EXIT OPTION...........................................................................................................................\*/

//IN CASE IF USER ENTERED OPTION OTHER THAN 1,2,3,4.

**while**( option != "1" && option != "2" && option != "3" && option != "4" ){

cout << "YOU SELECTED WRONG OPTION." << endl;

escape();

system ("cls");

option = different\_users\_login();

**if** (option == "1" || option == "2" || option == "3" || option == "4")

{

**continue**;

}

**break**;

}

}

}

/\*DEFINITIONS OF FUNCTIONS FOR MY HOSPITAL MANAGEMENT SYSTEM ARE.....................................................................\*/

/\* DEFINITIONS OF MUTUALLY USED FUNCTIONS...........................................................................................\*/

string different\_users\_login(){

logo();

string choice;

cout << "Welcome to Hospital Management System." << endl;

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

cout << "Login>" << endl;

cout << "Select one of the following." << endl;

cout << "1. Admin" << endl;

cout << "2. User " << endl;

cout << "3. Receptionist " << endl;

cout << "4. Exit " << endl;

cout << "Your option........." << endl;

cin >> choice;

**return** choice;

}

**void** logo(){

SetConsoleTextAttribute( h , **2**);

//THIS LOGO WILL BE CLEAR ON COMMAND PROMPT................

cout <<" /**\\** **\\** / **\\** **\\** /**\\**\_\_**\\** " << endl;

cout <<" **\\**:**\\** **\\** |::**\\** **\\** /:/ \_/\_ "<< endl;

cout <<" **\\**:**\\** **\\** |:|:**\\** **\\** /:/ /**\\** **\\** "<< endl;

cout <<" \_\_\_ /::**\\** **\\** \_\_|:|**\\**:**\\** **\\** /:/ /::**\\** **\\** "<< endl;

cout <<" /**\\** /:/**\\**:**\\**\_\_**\\** /::::|\_**\\**:**\\**\_\_**\\** /:/\_/:/**\\**:**\\**\_\_**\\** "<< endl;

cout <<" **\\**:**\\**/:/ **\\**/\_\_/ **\\**:**\\**~~**\\** **\\**/\_\_/ **\\**:**\\**/:/ /:/ / "<< endl;

cout <<" **\\**::/\_\_/ **\\**:**\\** **\\** **\\**::/ /:/ / "<< endl;

cout <<" **\\**:**\\** **\\** **\\**:**\\** **\\** **\\**/\_/:/ / "<< endl;

cout <<" **\\**:**\\**\_\_**\\** **\\**:**\\**\_\_**\\** /:/ / "<< endl;

cout <<" **\\**/\_\_/ **\\**/\_\_/ **\\**/\_\_/ "<< endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\* HOSPITAL MANAGEMENT SYSTEM \*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

SetConsoleTextAttribute( h , **11** );

}

**void** escape(){

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\* PRESS ANY KEY TO CONTINUE \*" << endl;

cout <<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

getch();

}

**void** thanks(){

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\* THANKS FOR USING HOSPITAL MANAGEMENT SYSTEM \*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

}

**void** congratulation(){

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "-------------CONGRATULATIONS YOUR PASSWORD IS CORRECT---------------" << endl;

cout << "--------------WELCOME TO HOSPITAL MANAGEMENT SYSTEM-----------------" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

}

**void** incorrect\_role(){

SetConsoleTextAttribute( h , **4**);

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "-------------------YOU DO NOT ENTER CORRECT ROLE--------------------" << endl;

cout << "---------------------PLEASE ENTER CORRECT ROLE----------------------" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

SetConsoleTextAttribute( h , **11**);

}

**void** invalid(){

SetConsoleTextAttribute( h , **4**);

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "--------------------INVALID PASSWORD OR USERNAME--------------------" << endl;

cout << "--------------PLEASE ENTER VALID PASSWORD OR USERNAME---------------" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

SetConsoleTextAttribute( h , **11**);

}

**void** datareader2(){

**int** number = **0** ;

string line ;

fstream file4 ;

file4.open("feedback.txt" , ios :: in ) ;

**while**(!file4.eof()){

getline(file4 , line) ;

comment[number] = line ;

number++ ;

}

file4.close() ;

}

**void** comment\_count\_reader(){

fstream file ;

file.open ("comment\_count.txt" , ios :: in ) ;

file >> comment\_count ;

file.close();

}

/\* DEFINITIONS OF ADMIN FUNCTIONS..................................................................................................\*/

string login\_manu(){

string admin;

string password;

string role;

cout << "Login details " << endl;

logo();

cout << "Login>Login details." << endl;

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

cout << "Enter username:";

cin >> admin;

cout << "Enter your password:";

cin >> password;

cout << "Enter your role:";

cin >> role;

**if** ( admin == "nazir" && password == "123" ){

**return** role;

}

**else**{

invalid();

escape();

system ("cls");

cout << "Login details " << endl;

logo();

cout << "Login>Login details." << endl;

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

cout << "Enter username:";

cin >> admin;

cout << "Enter your password:";

cin >> password;

cout << "Enter your role:";

cin >> role;

}

}

string menu(){

string choice1 ;

cout << "Main Menu for Admin" << endl;

logo();

cout << "Main Menu>Admin>" << endl;

cout << "Select one of the following............." << endl;

cout << "1. Add new patient" << endl;

cout << "2. View record of all patients" << endl;

cout << "3. View record of all patients in sorted order" << endl;

cout << "4. Staff record" << endl;

cout << "5. Birth report" << endl;

cout << "6. Death report" << endl;

cout << "7. Room availability" << endl;

cout << "8. See feedback of users." << endl;

cout << "9. Exit" << endl;

cout << " your option......." << endl;

cin >> choice1;

**return** choice1;

}

**void** addpatient(){

logo();

patient\_count ++;

fstream file ;

file.open ("patient\_count.txt" , ios ::out) ;

file << patient\_count ;

file <<endl ;

file.close() ;

cout << "Main Menu>Admin>Add patient" << endl;

cout << "Enter patient name..............." << endl;

cin >>patient\_name[patient\_count];

cout << "Enter patient father name..............." << endl;

cin >>patientf\_name[patient\_count];

cout << "Enter disease type...............";

cin >> patient\_disease[patient\_count];

//USE OF VALIDATIONS

cout << "Enter patient blood group..............." << endl;

cin >> patient\_group[patient\_count];

**while**( patient\_group[patient\_count] != "O+" && patient\_group[patient\_count] != "O-" && patient\_group[patient\_count] != "A+" && patient\_group[patient\_count] != "A-"

&& patient\_group[patient\_count] != "B+" && patient\_group[patient\_count] != "B-" && patient\_group[patient\_count] != "AB+" && patient\_group[patient\_count] != "AB-"){

cout << "Enter patient blood group..............." << endl;

cin >> patient\_group[patient\_count];

}

cout << "Enter patient gender..............." << endl;

cin >> patient\_gender[patient\_count];

**while**( patient\_gender[patient\_count] !="male" && patient\_gender[patient\_count] != "MALE" &&

patient\_gender[patient\_count] != "FEMALE" && patient\_gender[patient\_count] != "female" ){

cout << "Enter patient gender..............." << endl;

cin >> patient\_gender[patient\_count];

}

cout << "Enter patient age..............." << endl;

cin >> patient\_age[patient\_count];

**while**( patient\_age[patient\_count] <**1** || patient\_age[patient\_count] > **130**){

cout << "Enter patient age..............." << endl;

cin >> patient\_age[patient\_count];

}

cout << "Enter alloted room no(we can put 0 if there is no alloted room.)..............." << endl;

cin >> patient\_room[patient\_count];

**while**( patient\_room[patient\_count] < **0** || patient\_room[patient\_count] >**9**){

cout << "Enter alloted room no(we can put 0 if there is no alloted room.)..............." << endl;

cin >> patient\_room[patient\_count];

}

cout << "Enter date of admit..............." << endl;

cin >> patient\_date[patient\_count];

// file handling.

fstream file1;

file1.open ( "patientdata.txt" , ios :: app ) ;

file1 << patient\_name[patient\_count] << "," ;

file1 << patientf\_name[patient\_count] << "," ;

file1 << patient\_disease[patient\_count] << "," ;

file1 << patient\_group[patient\_count] << "," ;

file1 << patient\_gender[patient\_count] << "," ;

file1 << patient\_age[patient\_count] << "," ;

file1 << patient\_room[patient\_count] << "," ;

file1 << patient\_date[patient\_count] ;

file1 << endl;

file1.close ();

escape();

system ("cls");

}

**void** datareader(){

**int** number = **0** ;

string line ;

fstream file1 ;

file1.open("patientdata.txt" , ios :: in ) ;

**while**(!file1.eof()){

getline(file1 , line) ;

string temp = "" ;

**int** commaas = **1**;

**for** (**int** i = **0** ; line[i] != '\0' ; i++){

**if** (line[i] == ','){

commaas ++ ;

**continue** ;

}

**if** (commaas == **1**){

patient\_name[number] = patient\_name[number] + line[i] ;

}

**if** (commaas == **2**){

patientf\_name[number] = patientf\_name[number] + line[i] ;

}

**if** (commaas == **3**){

patient\_disease[number] = patient\_disease[number] + line[i] ;

}

**if** (commaas == **4**){

patient\_group[number] = patient\_group[number] + line[i] ;

}

**if** (commaas == **5**){

patient\_gender[number] = patient\_gender[number] + line[i] ;

}

**if** (commaas == **6**){

temp = temp + line[i] ;

patient\_age[number] = stoi(temp);

}

**if** (commaas == **7**){

temp = temp + line[i] ;

patient\_room[number] = stoi(temp) ;

}

**if** (commaas == **8**){

patient\_date[number] = patient\_date[number] + line[i] ;

}

}

number ++ ;

}

file1.close() ;

}

**void** patient\_count\_reader(){

fstream file4 ;

file4.open ("patient\_count.txt" , ios :: in ) ;

file4 >> patient\_count ;

file4.close();

}

**void** viewpatient( ){

logo();

cout << "Main Menu>Admin>Patients records" << endl;

cout << " " << endl;

cout << "name" << "**\t\t**" << "father name" << "**\t**" << "disease"<< "**\t\t**" << "age" << "**\t\t**" << "blood group" << "**\t\t**" << "gender" << "**\t\t\t**" << "date of admit"<< "**\t\t**" << "room" << endl;

cout << " " << endl;

**for** (**int** j = **0** ; j <=patient\_count ; j++ ){

cout << patient\_name[j] << "**\t\t**" << patientf\_name[j] << "**\t\t**" <<patient\_disease[j] << "**\t\t**" << patient\_age[j] << "**\t\t\t**" << patient\_group[j] << "**\t\t**" << patient\_gender[j] << "**\t\t**" << patient\_date[j]<< "**\t\t**" << patient\_room[j]<< endl;

}

cout << endl;

escape();

system ("cls");

}

**void** viewstaff(){

logo();

cout << "Main Menu>Admin>Staff record" << endl;

cout << "--------------------------------------------------DOCTOR'S RECORD----------------------------------------------------" << endl;

cout << " " << endl;

cout << "Name" << "**\t\t**" << "Age" << "**\t\t**" << "CNIC" << "**\t\t**" << "Specialist" << "**\t\t**" << "Phone no" << endl;

cout << " " << endl;

cout << "Dr.Ali" << "**\t\t**" << "40" << "**\t**" << "3310366032355" << " **\t**" << "flue,fever,viral diseases" << " **\t**" << "03106550274" << endl;

cout << "Dr.Hammad" << "**\t**" << "30" << "**\t**" << "3310366672356" << "**\t\t**" << "surgeon" << "**\t\t\t**" << "03106550273" << endl;

cout << "Dr.Ayesha" << "**\t**" << "29" << "**\t**" << "3310034567890" << "**\t\t**" << "eye specialist" << "**\t\t**" << "03230870971" << endl;

cout << "Dr.Hania" << "**\t**" << "31" << "**\t**" << "3310234567895" << "**\t\t**" << "ENT specialist" << "**\t\t**" << "03230870970" << endl;

cout << "Dr.Hania" << "**\t**" << "41" << "**\t**" << "3310892847848" << "**\t\t**" << "in emergency" << "**\t\t**" << "03245678999" << endl;

cout << " " << endl;

cout << "--------------------------------------------------NURSE'S RECORD----------------------------------------------------" << endl;

cout << " " << endl;

cout << "Name" << "**\t\t**" << "Phone no" << "**\t\t**" << "Name" << "**\t\t**" << "Phone no" << endl;

cout << " " << endl;

cout << "Hina" << "**\t\t**" << "03123456782" << "**\t\t**" << "Maria" << "**\t\t**" << "03245678998" << endl;

cout << "Saba" << "**\t\t**" << "03248978678" << "**\t\t**" << "Sonia" << "**\t\t**" << "03234567889" << endl;

cout << " " << endl;

cout << "--------------------------------------------------OTHER STAFF RECORD----------------------------------------------------" << endl;

cout << " " << endl;

cout << "Name" << "**\t\t**" << "**\t\t**" << "As" << endl;

cout << " " << endl;

cout << "Ali Ahmad" << "**\t\t**" << "**\t**" << "receptionist" << endl;

cout << "Saeed" << "**\t\t**" << "**\t\t**" << "sweeper" << endl;

cout << "Nasir" << "**\t\t**" << "**\t\t**" << "sweeper" << endl;

cout << "Rasheed" << "**\t\t**" << "**\t\t**" << "security guard" << endl;

escape();

}

**void** birthreport(){

logo();

cout << "Main Menu>Admin>Birth report" << endl;

cout << " " << endl;

cout << "Gender of Baby" << " " << "Condition" << "**\t** " << "Father Name" << " **\t**" << "Mother Name" << "**\t** " << "Phone no" << endl;

cout << " " << endl;

cout << "Boy" << "**\t\t**" << "normal" << "**\t\t**" << "Sufiyan" << "**\t\t**" << "Madiha" << "**\t\t**" << "03246667789" << endl;

cout << "Girl" << "**\t\t**" << "normal" << "**\t\t**" << "Ashraf" << "**\t\t**" << "Mahim" << "**\t\t**" << "03124567890 " << endl;

cout << "Boy" << "**\t\t**" << "normal" << "**\t\t**" << "Faisal" << "**\t\t**" << "Aliza" << "**\t\t**" << "03144567899" << endl;

cout << "Girl" << "**\t\t**" << "normal" << "**\t\t**" << "Rehan" << "**\t\t**" << "Nasira" << "**\t\t**" << "03245678909" << endl;

cout << "Girl" << "**\t\t**" << "normal" << "**\t\t**" << "Ahmad" << "**\t\t**" << "Adan" << "**\t\t**" << "03064264748" << endl;

escape();

}

**void** deathreport(){

logo();

cout << "Main Menu>Admin>Death report" << endl;

cout << " " << endl;

cout << "Death person" << "**\t\t\t** " << "Father Name" << " **\t**" << "Mother Name" << "**\t** " << "Phone no" << endl;

cout << " " << endl;

cout << "Boy" << "**\t\t**" << "**\t\t**" << "Sufiyan" << "**\t\t**" << "Madiha" << "**\t\t**" << "03246667789" << endl;

cout << "Girl" << "**\t\t**" << "**\t\t**" << "Ashraf" << "**\t\t**" << "Mahim" << "**\t\t**" << "03124567890 " << endl;

cout << "Boy" << "**\t\t**" << "**\t\t**" << "Faisal" << "**\t\t**" << "Aliza" << "**\t\t**" << "03144567899" << endl;

cout << "Girl" << "**\t\t**" << "**\t\t**" << "Rehan" << "**\t\t**" << "Nasira" << "**\t\t**" << "03245678909" << endl;

cout << "Girl" << "**\t\t**" << "**\t\t**" << "Ahmad" << "**\t\t**" << "Adan" << "**\t\t**" << "03064264748" << endl;

escape();

}

**void** roomrecord(){

logo();

cout << "Main Menu>Admin>Room record" << endl;

cout << " " << endl;

cout << "Room no" << "**\t\t\t\t**" << "For" << endl;

cout << " " << endl;

cout << " Room9" << "**\t\t\t**" << "Admin" << endl;

cout << " Room10" << "**\t\t\t**" << "Dr. Hammad" << endl;

cout << " Room11" << "**\t\t\t**" << "Dr. Hania" << endl;

cout << " Room12" << "**\t\t\t**" << "Dr. Ayesha" << endl;

cout << " Room13" << "**\t\t\t**" << "Dr. Ali" << endl;

cout << " Room14" << "**\t\t\t**" << "Dr. Ashraf" << endl;

cout << " Room15" << "**\t\t\t**" << "operation Theater" << endl;

cout << " Ward1" << "**\t\t\t**" << "Patients" << endl;

cout << " Ward2" << "**\t\t\t**" << "Patients" << endl;

cout << " Ward" << "**\t\t\t**" << "Patients" << endl;

cout << " " << endl;

cout << " ROOM 1 TO 8 ARE FOR PATIENT'S ALLOTEMENT. " << endl;

escape();

}

/\* DEFINITIONS OF PATIENT FUNCTIONS................................................................................................\*/

string login\_manu2(){

string user;

string pass;

string role;

cout << "Login details " << endl;

logo();

cout << "Login>Login details." << endl;

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

cout << "Enter username:";

cin >> user;

cout << "Enter your password:";

cin >> pass;

cout << "Enter your role:";

cin >> role;

**if** ( user == "ali" && pass == "321" ){

**return** role;

}

**else**{

invalid();

escape();

system ("cls");

cout << "Login details " << endl;

logo();

cout << "Login>Login details." << endl;

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

cout << "Enter username:";

cin >> user;

cout << "Enter your password:";

cin >> pass;

cout << "Enter your role:";

cin >> role;

}

}

string menu1(){

string choice2 ;

cout << "Main Menu for User " << endl;

logo();

cout << "Main Menu>User.>" << endl;

cout << " " << endl;

cout << "Select one of the following." << endl;

cout << "1. Edit your personal details" << endl;

cout << "2. View your personal details" << endl;

cout << "3. Edit your medical history" << endl;

cout << "4. View your medical history" << endl;

cout << "5. See Available doctors" << endl;

cout << "6. See recommendation" << endl;

cout << "7. Taking appointments" << endl;

cout << "8. Give feedback" << endl;

cout <<"9. Exit" << endl;

cout << "Your options................" << endl;

cin >> choice2;

**return** choice2;

}

**void** patient\_personal\_detail(){

user\_count++ ;

fstream file ;

file.open ("user\_count.txt" , ios ::out) ;

file << user\_count ;

file <<endl ;

file.close() ;

logo();

cout << "Main Menu> User> Personal details ." << endl;

cout << " " << endl;

cout << "Enter your name.........................." << endl;

cin >> nameA[user\_count];

cout << "Enter your father name..................." << endl;

cin >> f\_nameA[user\_count];

cout << "Enter your gender........................." << endl;

cin >> genderA[user\_count];

**while**( genderA[user\_count] !="male" && genderA[user\_count] != "MALE" &&

genderA[user\_count] != "FEMALE" && genderA[user\_count] != "female" ){

cout << "Enter patient gender..............." << endl;

cin >> genderA[user\_count];

}

fstream file2;

file2.open("personaldata.txt" , ios :: app) ;

file2 << nameA[user\_count] << " , " ;

file2 << f\_nameA[user\_count] << " , " ;

file2 << genderA[user\_count] ;

file2 << endl;

file2.close();

escape();

}

**void** view\_my\_detail(){

logo();

cout << "Main Menu> User> View personal details history ." << endl;

cout << " " << endl;

cout << "name" << "**\t\t\t**" << "father name" << "**\t\t**" << "gender" << endl;

cout << " " << endl;

**for** (**int** i = **0** ; i <= user\_count ; i++){

cout << nameA[i] << "**\t\t**" << " " << f\_nameA[i] << "**\t\t**" << genderA[i] << endl;

}

escape();

}

**void** patient\_medical\_detail(){

count ++;

fstream file ;

file.open ("count.txt" , ios ::out) ;

file << count ;

file <<endl ;

file.close() ;

logo();

cout << "Main Menu> User> medical history ." << endl;

cout << " " << endl;

cout << "Enter your disease type................................";

cin >> diseaseA[count];

cout << "Enter your blood group.................................";

cin >> blood\_groupA[count];

**while**( blood\_groupA[count] != "O+" && blood\_groupA[count] != "O-" && blood\_groupA[count] != "A+" && blood\_groupA[count] != "A-"

&& blood\_groupA[count] != "B+" && blood\_groupA[count] != "B-" && blood\_groupA[count] != "AB+" && blood\_groupA[count] != "AB-"){

cout << "Enter patient blood group..............." << endl;

cin >> blood\_groupA[count];

}

cout << "Enter date of previous appointment(if dont then press '0').....................";

cin >> dateA[count];

fstream file3;

file3.open("medicaldata.txt" , ios :: app) ;

file3 << diseaseA[count] << " , " ;

file3 << blood\_groupA[count] << " , " ;

file3 << dateA[count] ;

file3 << endl;

file3.close();

escape();

}

**void** view\_my\_medical\_detail(){

logo();

cout << "Main Menu>Admin>view\_my\_medical\_ records" << endl;

cout << " " << endl;

cout << "disease"<< "**\t\t**" << "blood group" << "**\t**" << "date of admit" << endl;

cout << " " << endl;

**for** (**int** i = **0** ; i <= count ; i++){

cout << diseaseA[i] << "**\t\t**" << blood\_groupA[i] << "**\t\t**" << dateA[i] << endl;

}

escape();

}

**void** viewdoctors\_details(){

logo();

cout << "Main Menu> User> Available doctors ." << endl;

cout << " " << endl;

cout << "Name of Doctors " << "**\t**" << "Specialist" << "**\t\t\t**" << "Timing" << endl;

cout << " " << endl;

cout << "Dr. Hammad" << "**\t\t**" << "surgeon" << "**\t\t\t\t**" << "12pm\_\_8pm" << endl;

cout << "Dr. Hania" << "**\t\t**" << "ENT specialist" << "**\t\t\t**" << "9am\_\_1pm" << endl;

cout << "Dr. Ali" << "**\t\t**" << "fever, flue and viral infections" << "**\t**" << "9am\_\_1pm, 3pm-6pm" << endl;

cout << "Dr. Ayesha" << "**\t\t**" << "eye specialist" << "**\t\t\t**" << "9am\_\_1pm" << endl;

cout << "Dr. Ashraf" << "**\t\t**" << "in emergency" << "**\t\t\t**" << "9pm\_\_4am" << endl;

escape();

}

**void** recommendation(string dis){

**if** (dis == "fever" || dis == "flue" || dis == "headache" || dis == "viral infections"|| dis == "stomach" )

{

cout <<"you are recommended to go to Dr. Ali in room 5" << endl;

}

**else** **if** (dis == "delivery" )

{

cout <<"you are recommended to go to Dr. Ashraf in room 7 or Dr.Hammad in room 2." << endl;

}

**else** **if** (dis == "eye" )

{

cout <<"you are recommended to go to Dr. Ayesha in room 4 ." << endl;

}

**else** **if** (dis == "ear" || dis == "nose" || dis == "throat" )

{

cout <<"you are recommended to go to Dr. Ayesha in room 3 ." << endl;

}

escape();

}

//FOR TAKING APPOINTMENTS

**void** get\_appointments(string dname , string day){

string agree;

**if** ( dname == "Dr.Ali" && day == "monday" || day == "tuesday" || day == "wednesday" || day == "thursday" || day == "friday" || day == "saturday" )

{

cout << "Dr.Ali is specific for fever, flue and viral infections." << endl;

cout << "Dr.Ali is free , so you can get appointment . " << endl;

cout << "Dr.Ali's charges will be 1000 for only checking. Are you sure you want to get appointment? (write 'yes' if you agree to get appointment) " << endl;

cin >> agree;

**if** ( agree == "yes" )

{

cout << "your appointment with Dr.Ali is confirmed on " << day << "**\t**" << "during the hour as described in dooctors menu" << endl;

}

**else**

{

cout << "you can't get appointmment." << endl;

}

}

**else** **if** (dname == "Dr.Ali" && day == "sunday")

{

cout << "Dr.Ali is not present on sunday in hospital.You cannot get appointment" << endl;

}

**else** **if** ( dname == "Dr.Ayesha" && day == "monday" || day == "tuesday" || day == "wednesday" || day == "thursday" )

{

cout << "Dr.Ayesha is specific for eye related problems" << endl;

cout << "Dr.Ayesha is free , so you can get appointment . " << endl;

string agree;

cout << "Dr.Ayesha's charges will be 2000 for only checking. Are you sure you want to get appointment? (write 'yes' if you agree to get appointment) " << endl;

cin >> agree;

**if** ( agree == "yes" )

{

cout << "your appointment with Dr.Ayesha is confirmed on " << day << "**\t**" << "during the hour as described in doctors menu" << endl;

}

**else**

{

cout << "you can't get appointmment." << endl;

}

}

**else** **if** (dname == "Dr.Ayesha" && day == "sunday" || day == "saturday" || day == "friday" )

{

cout << "Dr.Ayesha is not present on friday , saturday and sunday in hospital.You cannot get appointment." << endl;

}

**else** **if** ( dname == "Dr.Hania" && day == "monday" || day == "tuesday" || day == "wednesday" || day == "thursday" || day == "friday" )

{

cout << "Dr.Hania is specific for ENT related problems" << endl;

cout << "Dr.Hania is free , so you can get appointment . " << endl;

string agree;

cout << "Dr.Hania's charges will be 2000 for only checking. Are you sure you want to get appointment? (write 'yes' if you agree to get appointment) " << endl;

cin >> agree;

**if** ( agree == "yes" )

{

cout << "your appointment with Dr.Hania is confirmed on " << day << "**\t**" << "during the hour as described in doctors menu" << endl;

}

**else**

{

cout << "you can't get appointmment." << endl;

}

}

**else** **if** (dname == "Dr.Hania" && day == "sunday" || day == "saturday" )

{

cout << "Dr.Hania is not present on saturday and sunday in hospital.You cannot get appointment." << endl;

}

**else** **if** ( dname == "Dr.Hammad" && day == "monday" || day == "tuesday" || day == "wednesday" || day == "thursday" || day == "friday" )

{

cout << "Dr.Hammad is surgeon , specific for delivery . " << endl;

cout << " you can get appointment . " << endl;

string agree;

cout << "Dr.Hammad's charges will be 40000 for delivery. Are you sure you want to get appointment? (write 'yes' if you agree to get appointment) " << endl;

cin >> agree;

**if** ( agree == "yes" )

{

cout << "your appointment with Dr.Hammad is confirmed on " << day << "during the hour as described in doctors menu" << endl;

}

**else** {

cout << "you can't get appointmment." << endl;

}

}

**else** **if** (dname == "Dr.Hammad" && day == "sunday" || day == "saturday" )

{

cout << "Dr.Hammad is not present on saturday and sunday in hospital but in replacement of Dr. Hammad , Dr.Ashraf is present for delivery cases." << endl;

cout << "Are you want to consult with Dr.Ashraf?(press 'y' for confirming)" << endl;

**char** agree1;

cin >> agree1;

**if** (agree1 == 'y')

{

cout << "your appointment with Dr.Ashraf is confirmed on " << day << "**\t**" << "during the hour as described in doctors menu" << endl;

}

**else**

{

cout << "you can't get appointmment." << endl;

}

}

**else** **if** ( dname == "Dr.Ashraf" && day == "tuesday" || day == "wednesday" || day == "thursday" || day == "friday" || day == "sunday" || day == "saturday" )

{

cout << "Dr.Ashraf is specific for delivery" << endl;

cout << "Dr.Ashraf is free , so you can get appointment . " << endl;

string agree;

cout << "Dr.Ashraf's charges will be 40000 for delivery. Are you sure you want to get appointment? (write 'yes' if you agree to get appointment) " << endl;

cin >> agree;

**if** ( agree == "yes" )

{

cout << "your appointment with Dr.Ashraf is confirmed on " << day << "**\t**" << "during the hour as described in doctors menu" << endl;

}

**else**

{

cout << "you can't get appointmment." << endl;

}

}

**else** **if** (dname == "Dr.Ashraf" && day == "monday" )

{

cout << "Dr.Ashraf is not present on monday in hospital.You cannot get appointment." << endl;

}

escape();

}

**void** feedback(){

comment\_count++ ;

fstream file ;

file.open ("comment\_count.txt" , ios ::out) ;

file << comment\_count ;

file <<endl ;

file.close() ;

logo();

cout << "Main Menu> User> feedback ." << endl;

cout << " " << endl;

cout << "May Allah bless you with good health. (Ameen)" << endl;

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

cout << "Give feedback about hospital management system..........." << endl;

cout << "-------GIVE FEEDBACK FROM FOLLOWING WORDS................" << endl;

cout << ".......GOOD ....... BAD........EXCELLENT.......POOR......" << endl;

cin >> comment[comment\_count];

**while** (comment[comment\_count] != "GOOD" && comment[comment\_count] != "POOR" &&

comment[comment\_count] != "EXCELLENT" && comment[comment\_count] != "BAD" ){

cout << "Give feedback about hospital management system..........." << endl;

cin >> comment[comment\_count];

}

fstream file4;

file4 . open ("feedback.txt" , ios :: app ) ;

file4 << comment[comment\_count] ;

file4 << endl;

file4 .close() ;

}

//USE OF SORTING

**void** sorting(){

**float** max = -**10**;

string show\_disease , show\_name , show\_fname , show\_gender , show\_date , show\_group ;

**int** show\_room ;

**int** idx;

**for**(**int** i = start ; i <= patient\_count ; i++){

**if** ( patient\_age[i] > max ) {

max = patient\_age[i] ;

show\_name = patient\_name[i];

show\_fname = patientf\_name[i];

show\_disease = patient\_disease[i];

show\_gender = patient\_gender[i];

show\_date = patient\_date[i];

show\_group = patient\_group[i];

idx = i ;

}

}

patient\_age[idx] = patient\_age[start] ;

patient\_age[start] = max ;

patient\_name[idx] = patient\_name[start];

patient\_name[start] = show\_name ;

patientf\_name[idx] = patientf\_name[start] ;

patientf\_name[start] = show\_fname ;

patient\_disease[idx] = patient\_disease[start] ;

patient\_disease[start] = show\_disease ;

patient\_gender[idx] = patient\_gender[start] ;

patient\_gender[start] = show\_gender ;

patient\_date[idx] = patient\_date[start] ;

patient\_date[start] = show\_date ;

patient\_group[idx] = patient\_group[start] ;

patient\_group[start] = show\_group ;

logo();

cout << "Main Menu>Admin>Sorted Patient Records" << endl;

cout << " " << endl;

cout << "name" << "**\t\t**" << "father name" << "**\t**" << "disease"<< "**\t\t**" << "age" << "**\t**" << "blood group" << "**\t\t**" << "gender" << "**\t\t**" << "date of admit"<< "**\t\t**" << "room" << endl;

cout << " " << endl;

**for** (**int** i=**0**;i<=patient\_count;i++){

cout << patient\_name[i] << "**\t\t**" << patientf\_name[i] << "**\t\t**" <<patient\_disease[i] << "**\t\t**" << patient\_age[i] << "**\t\t**" << patient\_group[i] << "**\t\t**" << patient\_gender[i] << "**\t\t**" << patient\_date[i]<< "**\t\t**" << patient\_room[i]<< endl;

}

escape();

start++ ;

}

/\* DEFINITIONS OF RECEPTIONIST FUNCTIONS..................................................................................................\*/

string login\_manu3(){

string user;

string pass;

string role;

cout << "Login details " << endl;

logo();

cout << "Login>Login details." << endl;

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

cout << "Enter username:";

cin >> user;

cout << "Enter your password:";

cin >> pass;

cout << "Enter your role:";

cin >> role;

**if** ( user == "nasir" && pass == "456" ){

**return** role;

}

**else**{

invalid();

escape();

system ("cls");

cout << "Login details " << endl;

logo();

cout << "Login>Login details." << endl;

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

cout << "Enter username:";

cin >> user;

cout << "Enter your password:";

cin >> pass;

cout << "Enter your role:";

cin >> role;

}

}

string menu2(){

string choice3 ;

cout << "Main Menu for Receptionist " << endl;

logo();

cout << "Main Menu > Receptionist . >" << endl;

cout << " " << endl;

cout << "Select one of the following." << endl;

cout << "1. View entered personal of patients" << endl;

cout << "2. View entered medical of patients" << endl;

cout << "3. Exit" << endl;

cout << "Your options................" << endl;

cin >> choice3;

**return** choice3;

}

**void** receptionist\_view( ){

logo();

cout << "Main Menu > Receptionist > View personal record" << endl;

cout << " " << endl;

cout << "name" << "**\t\t**" << "father name" << "**\t\t**" << "gender" << endl;

cout << " " << endl;

**for** (**int** j = **0** ; j<=user\_count ; j++ ){

cout << nameA[j] << "**\t\t**" << f\_nameA[j] << "**\t\t**" << genderA[j] << endl;

}

cout << endl;

escape();

system ("cls");

}

**void** receptionist\_view2( ){

logo();

cout << "Main Menu > Receptionist > View medical record" << endl;

cout << " " << endl;

cout << "disease"<< "**\t\t**" << "blood group" << "**\t\t**" << "date of admit" << endl;

cout << " " << endl;

**for** (**int** j = **0** ; j<=count ; j++ ){

cout << diseaseA[j] << "**\t\t**" << blood\_groupA[j] << "**\t\t\t**" << dateA[j] << endl;

}

cout << endl;

escape();

system ("cls");

}

//....................................................................................................................................

**void** datareader1(){

**int** number = **0** ;

string line ;

fstream file1 ;

file1.open("personaldata.txt" , ios :: in ) ;

**while**(!file1.eof()){

getline(file1 , line) ;

string temp = "" ;

**int** commaas = **1**;

**for** (**int** i = **0** ; line[i] != '\0' ; i++){

**if** (line[i] == ','){

commaas ++ ;

**continue** ;

}

**if** (commaas == **1**){

nameA[number] = nameA[number] + line[i] ;

}

**if** (commaas == **2**){

f\_nameA[number] = f\_nameA[number] + line[i] ;

}

**if** (commaas == **3**){

genderA[number] = genderA[number] + line[i] ;

}

}

number ++ ;

}

file1.close() ;

}

**void** user\_count\_reader(){

fstream file4 ;

file4.open ("user\_count.txt" , ios :: in ) ;

file4 >> user\_count ;

file4.close();

}

//............................................................................................................

**void** datareader3(){

**int** number = **0** ;

string line ;

fstream file3 ;

file3.open("medicaldata.txt" , ios :: in ) ;

**while**(!file3.eof()){

getline(file3 , line) ;

string temp = "" ;

**int** commaas = **1**;

**for** (**int** i = **0** ; line[i] != '\0' ; i++){

**if** (line[i] == ','){

commaas ++ ;

**continue** ;

}

**if** (commaas == **1**){

diseaseA[number] = diseaseA[number] + line[i] ;

}

**if** (commaas == **2**){

blood\_groupA[number] = blood\_groupA[number] + line[i] ;

}

**if** (commaas == **3**){

dateA[number] = dateA[number] + line[i] ;

}

}

number ++ ;

}

file3.close() ;

}

**void** count\_reader(){

fstream file ;

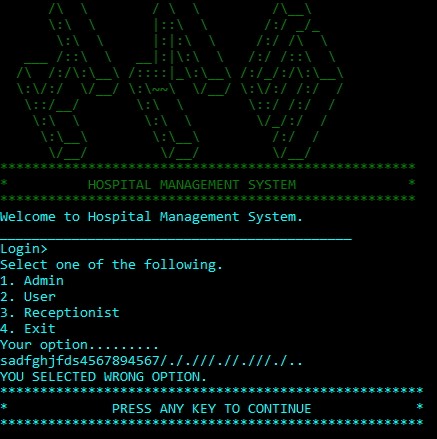
file.open ("count.txt" , ios :: in ) ;

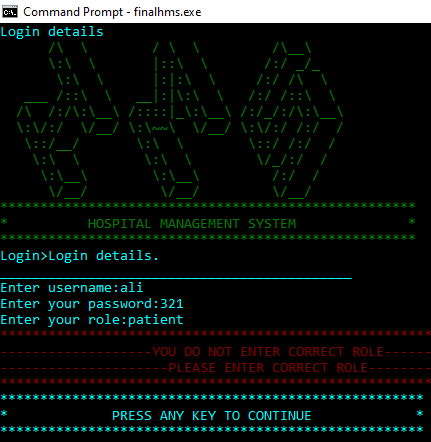
file >> count ;

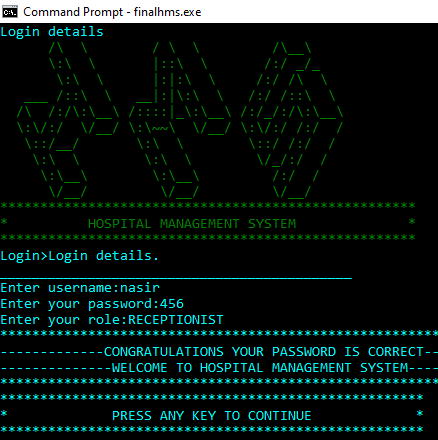
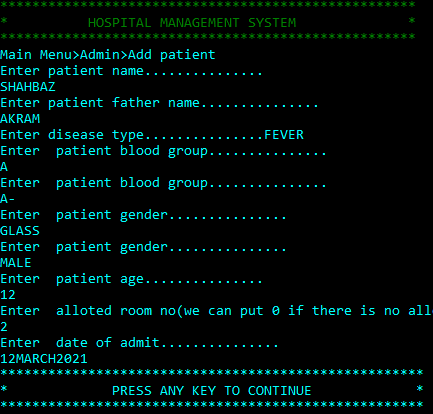
file.close();

**TEST CASES**

Here I describeed some test cases of my code which proves the validations of my code.

* **For invalid selection of options**
* **For invalid role**



* **For Selecting Right option**
* **Here is validation on some points as gender,bloodgroup , age etc.**

**FOR STUDENT:**

**Student Reg. No. :2021-CS-8**   **Student Name. Muhammad Nazir**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Checked by:** | **A-Extensive**  **Evidence** | **B-Convincing**  **Evidence** | **C-Limited**  **Evidence** | **D-No Evidence** |
| Documentation Formatting **Grade:** | All the documentation meets all the criteria. |  |  | Documentation is not  Available |
| Documentation is well formatted but some of the criteria is not fulfilled. | Documentation is required a lot of improvement. |
| **Documentation Formatting Criteria:** In **Binder**, **Title** Page, **Header**-Footers, Font **Style**, Font **Size** all are all consistence and according to given **guidelines**. Project **Poster** is professionally design and well presented | | | | |
| Documentation  Contents **Grade:** | Documentation  includes all of the criteria. | Documentation meet more than 80% of the criteria given. | Documentation meet more than 50% of the criteria. | When the  documentation meet less than 50% of the criteria. |
| **Documentation Contents Criteria:** **Title** Page - **Table** of Contents - Project **Abstract** - **Functional** Requirements - **Wire** Frames –**Data Flow**  Diagram-**Data** Structure (Arrays)-**Function** Headers and Description - **Algorithms** and Flow Charts of all functions- **Test Cases** are defined Project **Code.** - **Weakness** in the Project and **Future** Directions. **Conclusion** and What your **Learn** from the Project and Course and What is your **Future** Planning. | | | | |

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Project  Complexity **Grade:** | Project has at least 2 user’s types and each user has at least 5 functionalities. | Project complexity meet 80% criteria given in extensive evidence | Project complexity meet 50% criteria given in extensive evidence | Project complexity meet less than 50% criteria given in extensive evidence |
| Code Style **Grade:** | All Code style criteria is followed | All code style criteria followed but some  improvements required | lot of improvements required in coding style. | **Did not follow** code  style, |
| **Code Style Criteria:**  Consistent code style. Code is well indented. Variable and Function names are well defined. White Spaces are well used. Comments are added. | | | | |
| Code  Documentation  Mapping **Grade:** | Code and documentation is synchronized. | Code and documentation does not synchronized at **some** places | Code and documentation does not synchronized at **many** places | Code and documentation **does not** synchronized. |
| Data Structure (Arrays) **Grade:** | Data structure is sufficient for the project requirements | Data Structure is sufficient but require improvement to meet project requirements. | Data structure is not sufficient and need a lot of improvement | Data Structure is not properly identified and declared. |
| Sorting Features **Grade:** | Sort working 100% and generating useful report | Sorting Feature is working but sorted data is not useful for project. | Sorting feature is partial implemented | Project do not contain sorting |
| Modularity **Grade:** | Meet all Modularity  criteria | Meet all Modularity criteria but at some places it is missing | Do not sufficiently meet the modularity criteria. | No modularity or very minimum modularity. |
| **Modularity criteria:** Functions are defined for each major feature. Functions are independent (identify from parameter list and return types)- Demo Data Functionality AddedAt least Two Unit Tests are defined. | | | | |
| Validations **Grade:** | Validations on all number type inputs are applied | Validations are applied but at some places it is missing. | Validations are missing at lot of places | No Validations are used |
| Recommendatio n  Feature | Proper meaning full recommendation is present into system | Partial  Recommendation is implemented | Implemented but not meaning full. | Not implemented |
| Presentation and  Demo  **Grade:** | Presentation and Demo was 100%  working | Presentation and Demo require some improvements | Presentation and Demo require a lot of improvements | Presentation was not ok and Demo was not working |
| Student Understanding with the Code.  **Grade:** | Student has complete understanding how the code is working and knows the concept. | Student has good understand but some place he does not know the concepts | Student has a very little understand and lack the major concepts. | Student does not have any level of understanding of the code. |

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

**FOR PROFESSOR:**

**Student Reg. No. :2021-CS-8**   **Student Name. Muhammad Nazir**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Checked by:** | **A-Extensive**  **Evidence** | **B-Convincing**  **Evidence** | **C-Limited**  **Evidence** | **D-No Evidence** |
| Documentation Formatting **Grade:** | All the documentation meets all the criteria. |  |  | Documentation is not  Available |
| Documentation is well formatted but some of the criteria is not fulfilled. | Documentation is required a lot of improvement. |
| **Documentation Formatting Criteria:** In **Binder**, **Title** Page, **Header**-Footers, Font **Style**, Font **Size** all are all consistence and according to given **guidelines**. Project **Poster** is professionally design and well presented | | | | |
| Documentation  Contents **Grade:** | Documentation  includes all of the criteria. | Documentation meet more than 80% of the criteria given. | Documentation meet more than 50% of the criteria. | When the  documentation meet less than 50% of the criteria. |
| **Documentation Contents Criteria:** **Title** Page - **Table** of Contents - Project **Abstract** - **Functional** Requirements - **Wire** Frames –**Data Flow**  Diagram-**Data** Structure (Arrays)-**Function** Headers and Description - **Algorithms** and Flow Charts of all functions- **Test Cases** are defined Project **Code.** - **Weakness** in the Project and **Future** Directions. **Conclusion** and What your **Learn** from the Project and Course and What is your **Future** Planning. | | | | |

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Project  Complexity **Grade:** | Project has at least 2 user’s types and each user has at least 5 functionalities. | Project complexity meet 80% criteria given in extensive evidence | Project complexity meet 50% criteria given in extensive evidence | Project complexity meet less than 50% criteria given in extensive evidence |
| Code Style **Grade:** | All Code style criteria is followed | All code style criteria followed but some  improvements required | lot of improvements required in coding style. | **Did not follow** code  style, |
| **Code Style Criteria:**  Consistent code style. Code is well indented. Variable and Function names are well defined. White Spaces are well used. Comments are added. | | | | |
| Code  Documentation  Mapping **Grade:** | Code and documentation is synchronized. | Code and documentation does not synchronized at **some** places | Code and documentation does not synchronized at **many** places | Code and documentation **does not** synchronized. |
| Data Structure (Arrays) **Grade:** | Data structure is sufficient for the project requirements | Data Structure is sufficient but require improvement to meet project requirements. | Data structure is not sufficient and need a lot of improvement | Data Structure is not properly identified and declared. |
| Sorting Features **Grade:** | Sort working 100% and generating useful report | Sorting Feature is working but sorted data is not useful for project. | Sorting feature is partial implemented | Project do not contain sorting |
| Modularity **Grade:** | Meet all Modularity  criteria | Meet all Modularity criteria but at some places it is missing | Do not sufficiently meet the modularity criteria. | No modularity or very minimum modularity. |
| **Modularity criteria:** Functions are defined for each major feature. Functions are independent (identify from parameter list and return types)- Demo Data Functionality AddedAt least Two Unit Tests are defined. | | | | |
| Validations **Grade:** | Validations on all number type inputs are applied | Validations are applied but at some places it is missing. | Validations are missing at lot of places | No Validations are used |
| Recommendatio n  Feature | Proper meaning full recommendation is present into system | Partial  Recommendation is implemented | Implemented but not meaning full. | Not implemented |
| Presentation and  Demo  **Grade:** | Presentation and Demo was 100%  working | Presentation and Demo require some improvements | Presentation and Demo require a lot of improvements | Presentation was not ok and Demo was not working |
| Student Understanding with the Code.  **Grade:** | Student has complete understanding how the code is working and knows the concept. | Student has good understand but some place he does not know the concepts | Student has a very little understand and lack the major concepts. | Student does not have any level of understanding of the code. |

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