DELHI PUBLIC SCHOOL BANGALORE EAST

COMPUTER SCIENCE PROJECT



BY PRAKHAR SONI

CLASS :12A

ROLL NUMBER :09

**ACKNOWLEDGEMENT**

The successful completion of any task would be incomplete without mentioning the names of those people without who helped to make it possible. I take this opportunity to express my gratitude I few words and respect to all those who helped me in the completion of this project. It is my humble pleasure to acknowledge my deep senses of gratitude to my computer teacher Vikranti ma’am for their valuable support and determination, without which this project wouldn’t be possible.

I also register my sense of gratitude to our principle, Mrs. Manila Carvalho, for her immense encouragement that has made this project successful. I would like to also thank my friends and family for encouraging me during this project.

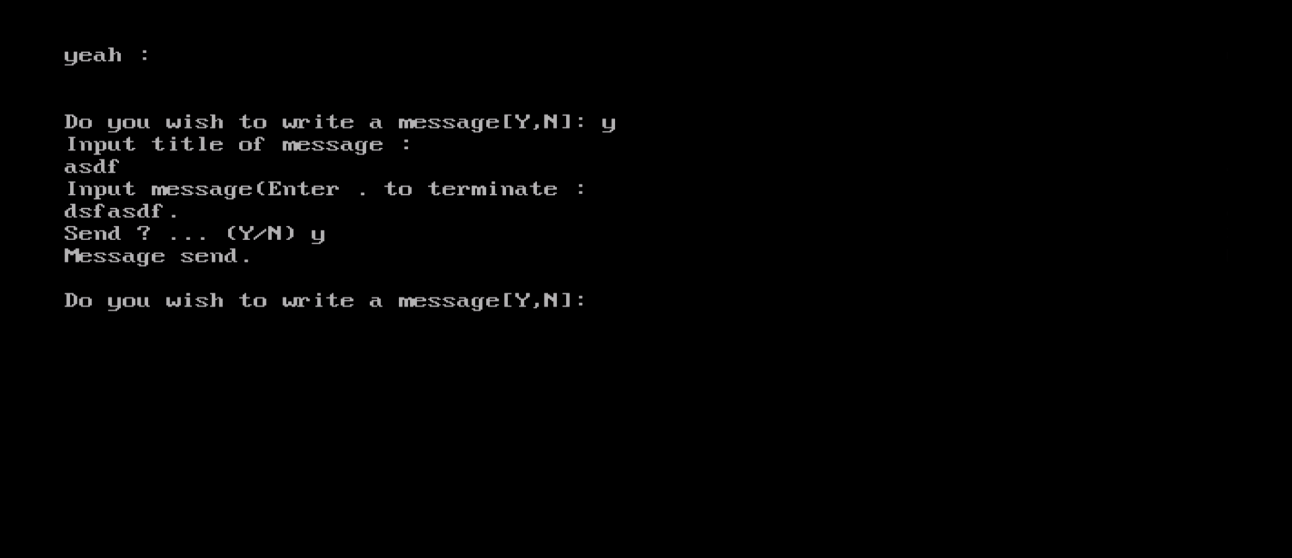
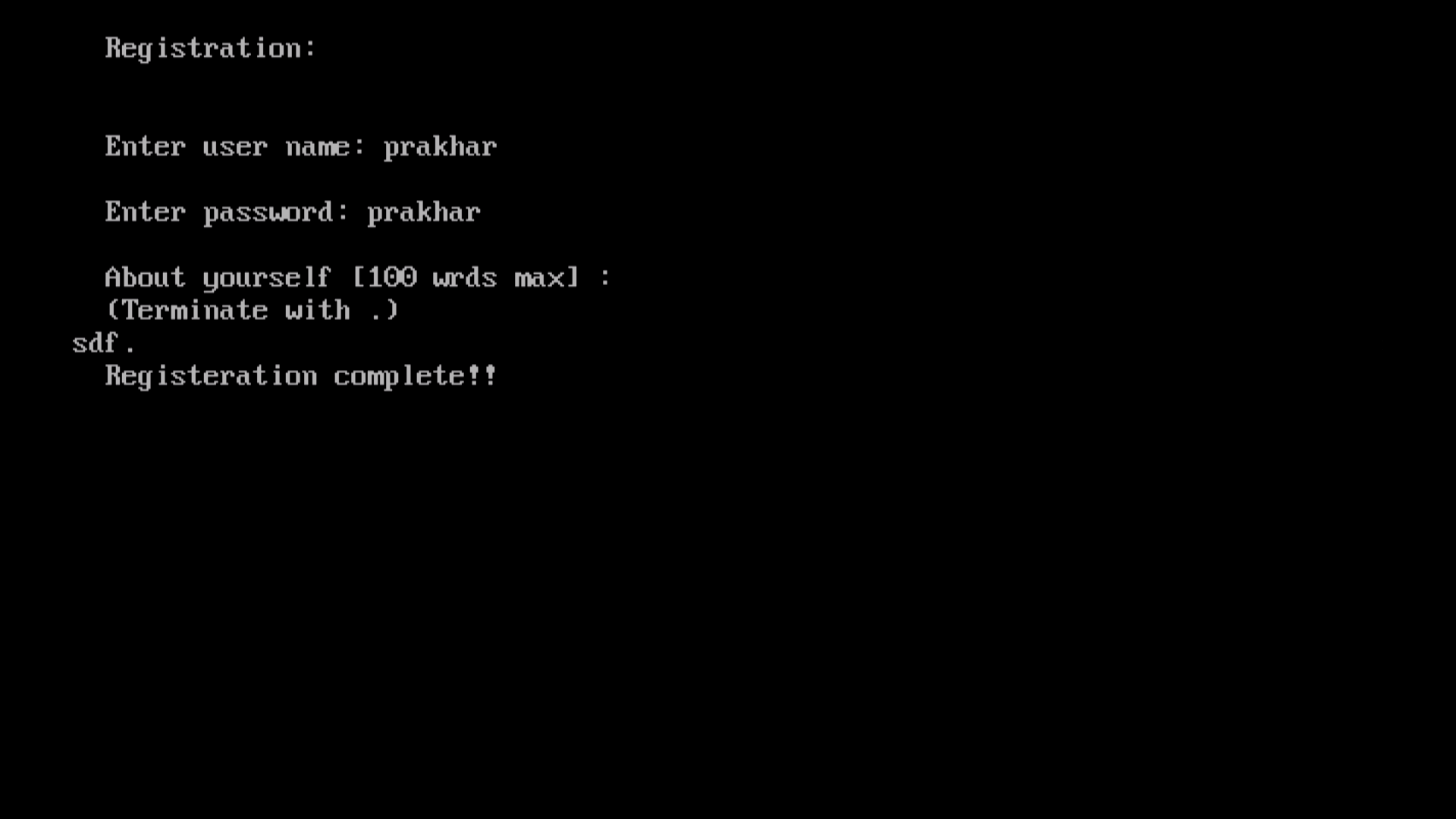
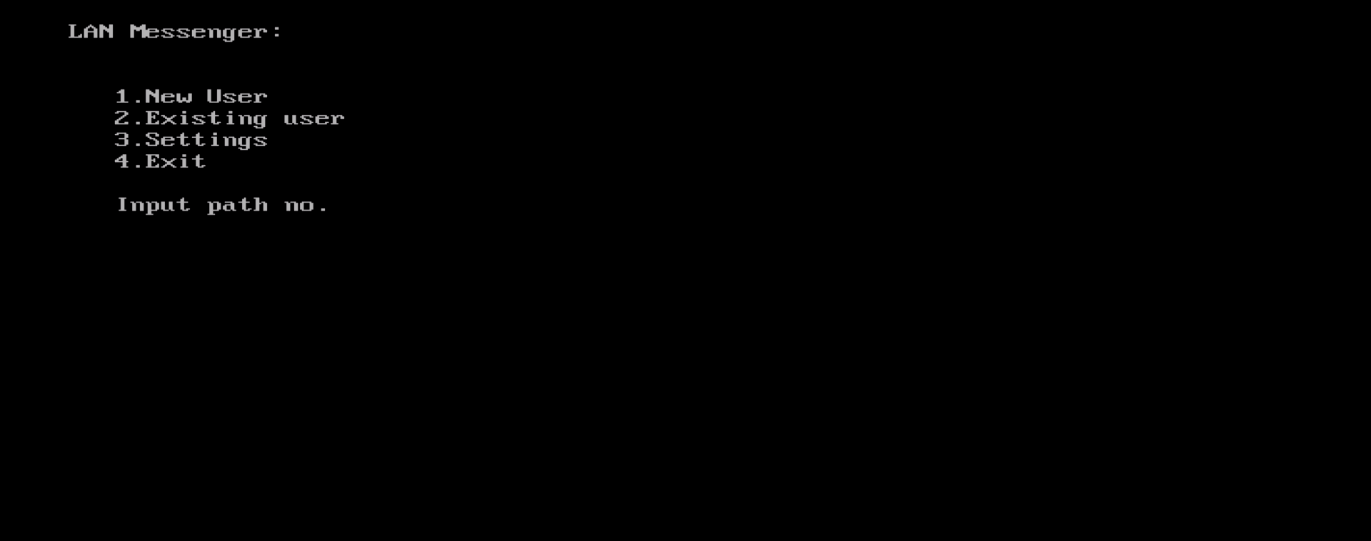
Last, but not the least, I would like to thank CBSE and my school Delhi Public School East for giving us the opportunity to undertake this project.

INDEX

* CODE
* OUTPUT
* BIBLEOGRAPHY

|  |
| --- |
| #include<iostream> |
|  | #include <conio.h> |
|  | #include <time.h> |
|  | #include<string.h> |
|  | #include<stdlib.h> |
|  | #include<stdio.h> |
|  | #include<fstream.h> |
|  | #include<ctype.h> |
|  | //v with pwd |
|  | void sp(int down,int left); |
|  | class user; |
|  | class chatroom; |
|  | void messageloop(int temp,char chatroom[30]); |
|  | void main(); |
|  | void newUser(); |
|  | void existingUser(); |
|  | void chat(char chatname[30]); |
|  | void userControlPanel(char \*uname); |
|  | char current\_user[30]; //current user |
|  | //end:Public function prototypes and global |
|  |  |
|  | //start: global functions |
|  | //ui designing function |
|  | void sp(int down, int left) { |
|  | for(int i =0;i<down;i++) { |
|  | cout << "\n"; |
|  | } |
|  | for(int l =0;l<left;l++) { |
|  | cout << " "; |
|  | } |
|  | } |
|  |  |
|  | //callBatchfile |
|  | void callBatchFile(char batchp[30]) { |
|  | strcat(batchp,".bat"); |
|  | system(batchp); |
|  | } |
|  | //end: global functions |
|  |  |
|  |  |
|  | /\*start: Message class definition |
|  | //Sample implementation code |
|  | void main() { |
|  | clrscr(); |
|  | message msg1; |
|  | msg1.inputMessage(); |
|  | msg1.output(); |
|  | getch(); |
|  | } |
|  | //Status: Fully working and operational |
|  | \*/ |
|  |  |
|  | class message { |
|  | char title[30]; |
|  | char from[30]; |
|  | char to[30]; |
|  | char msg[200]; |
|  | public: |
|  | int msg\_status; |
|  | /\* msg status of 1 indicates user confirmed success delivery of message |
|  | msg status of 0 indicates a error \*/ |
|  | //prototypes |
|  | void inputMessage(); |
|  | void writetoDb(char filename[30],message m); |
|  | void checkInput(char x[300],int max); |
|  | void output(); |
|  | //default constructor |
|  | message() { |
|  | strcpy(title,"error") ; |
|  | strcpy(from,"error"); |
|  | strcpy(to,"error"); |
|  | strcpy(msg,"error"); |
|  | msg\_status = 0; |
|  | } |
|  | }newm,om[50],tempm[50]; //end of message class |
|  |  |
|  |  |
|  | //validator for inputs to the message objects |
|  | void message::checkInput(char x[300],int max) { |
|  | if(!(strlen(x) <= max)) { |
|  | cout << "Input must be less than 30 characters"; |
|  | msg\_status = 0; |
|  | //add input function here. |
|  | } |
|  | } |
|  | //Output message |
|  | void message :: output() { |
|  | cout<< endl <<endl; |
|  | if(strcmp(title,"error") != 0) { |
|  | cout << "Message from "; |
|  | puts(from) ; |
|  | cout << endl << "Title: "; |
|  | puts(title); cout << endl; |
|  | puts(msg); cout << endl; |
|  | cout << endl; |
|  | } |
|  | } |
|  | //Input message |
|  | void message :: inputMessage() { |
|  | char ch; |
|  | cin.ignore(); |
|  | strcpy(from,current\_user); //Chage crt user with the logged in user |
|  | cout << "Input title of message : " << endl; |
|  | cin.getline(title,30); |
|  | cout << "Input message(Enter . to terminate : " << endl; |
|  | cin.getline(msg,200,'.'); |
|  | cout << "Send ? ... (Y/N) "; |
|  | cin >> ch; |
|  | if(ch == 'Y' || ch == 'y') { |
|  | msg\_status = 1; |
|  | cout << "Message send."; |
|  | } else { |
|  | cout << endl; |
|  | cout << "NewDraft: " << endl; |
|  | inputMessage(); |
|  | } |
|  | cout << endl; |
|  | } |
|  |  |
|  | void message :: writetoDb(char filename[30],message m) { |
|  | ofstream chat; |
|  | strcat(filename,".txt"); |
|  | chat.open(filename,ios::app | ios::binary); |
|  | chat.write((char\*)&m,sizeof(m)); |
|  | chat.close(); |
|  | } |
|  | //end: Message class definition |
|  |  |
|  | //start: Chat function start comes here after the join a chatrrom function |
|  | void chat(char chatname[30]) { |
|  | char filename[30]; |
|  | int tempmsgcount =0 ; |
|  | strcpy(filename,chatname); |
|  | strcat(filename,".txt"); |
|  | clrscr(); |
|  | sp(2,0); |
|  | cout << chatname << " : "; |
|  | fstream chat; |
|  | chat.open(filename,ios::in,ios::noreplace); |
|  | chat.seekg(0,ios::end); |
|  | // if file is empty |
|  | if(chat.tellg() == -1) { |
|  | chat.close(); |
|  | sp(3,10); |
|  | newm.inputMessage(); |
|  | newm.writetoDb(chatname,newm); |
|  | } else { |
|  | ifstream chat; |
|  | chat.open(filename,ios::in | ios::noreplace); |
|  | int count = 0; |
|  | //Lists existing messages |
|  | om[count].msg\_status =1; |
|  | while(!chat.eof()) { |
|  |  |
|  | chat.read((char\*)&om[count],sizeof(om[count])); |
|  | if(om[count].msg\_status== 1){ |
|  | om[count].output(); |
|  | om[count].msg\_status =0; |
|  | } |
|  | getch(); |
|  |  |
|  |  |
|  | }//end of while |
|  | chat.close(); |
|  | messageloop(tempmsgcount,chatname); |
|  | } //end of else |
|  | } //end of function |
|  |  |
|  | void messageloop(int tempmsgcount,char chatname[30]) { |
|  | char user\_ch; |
|  | ifstream chat; |
|  | cout << endl << "Do you wish to write a message[Y,N]: "; |
|  | cin >> user\_ch; |
|  | if(user\_ch == 'Y' || user\_ch == 'y') { |
|  | chat.read((char\*)&tempm[tempmsgcount],sizeof(tempm[tempmsgcount])); |
|  | tempm[tempmsgcount].inputMessage(); |
|  | tempm[tempmsgcount].writetoDb(chatname,tempm[tempmsgcount]); |
|  | tempmsgcount++; |
|  | messageloop(tempmsgcount,chatname); |
|  | } else { |
|  | clrscr(); |
|  | chat.close(); |
|  | userControlPanel(current\_user); |
|  | } |
|  | } |
|  | //end: Chat function end |
|  |  |
|  | //1. |
|  |  |
|  | //start: User class definition |
|  |  |
|  | //Users start |
|  | class user { |
|  | char username[30]; |
|  | char pw[30]; |
|  | char about[100]; |
|  | float activity; |
|  | float start,mins,end,seconds; |
|  | public: |
|  | //default constuctor |
|  | user() { |
|  | strcpy(username,"empty"); |
|  | strcpy(pw,"empty"); |
|  | strcpy(about,"empty"); |
|  | activity = 0; |
|  | } |
|  | //start timer on login |
|  | void timestart() { |
|  | start= time(0); |
|  | } |
|  |  |
|  | //Create new user |
|  | void create() { |
|  | sp(1,2); |
|  | cout << "Registration: "; |
|  | cin.ignore(); |
|  | sp(3,2); |
|  | cout<<"Enter user name: "; |
|  | cin.getline(username,30); |
|  | cin.ignore(0,'\n'); |
|  | sp(1,2); |
|  | cout<<"Enter password: "; |
|  | cin.getline(pw,30); |
|  | sp(1,2); |
|  | cout << "About yourself [100 wrds max] : " ; |
|  | sp(1,2); |
|  | cout << "(Terminate with .)" ; |
|  | sp(1,0); |
|  | cin.getline(about,100,'.'); |
|  | } |
|  | //Print out user details |
|  | void printUserData() { |
|  | cout << username << endl; |
|  | cout << pw << endl; |
|  | cout << about << endl; |
|  | } |
|  |  |
|  |  |
|  |  |
|  | //stop and log time on user logout |
|  | void logout() { |
|  | end=time(0); |
|  | seconds=difftime(end,start); |
|  | mins=seconds/60; |
|  | } |
|  |  |
|  | //write user data to users.txt |
|  | void writeUserData(user obj) { |
|  | //obj must be intialized with values at this point |
|  | ofstream db("users.txt",ios::app|ios::binary); |
|  | db.write((char\*)&obj,sizeof(obj)); |
|  | db.close(); |
|  | sp(0,2); |
|  | cout<<"Registeration complete!!"; |
|  | } |
|  |  |
|  | //intialises a temp variable with username value |
|  | void uname(char input[30]) { |
|  | for(int i =0;i<30;i++) { |
|  | input[i] = username[i]; |
|  | } |
|  | } |
|  |  |
|  | //initialises a temp variable with pwd value |
|  | void pwd(char input[30]) { |
|  | for(int i =0;i<30;i++) { |
|  | input[i] = pw[i]; |
|  | } |
|  | } |
|  |  |
|  |  |
|  | //read All User from db |
|  | void readUserFromDb() { |
|  | user ts[100]; |
|  | ifstream db; |
|  | db.open("users.txt",ios::in|ios::binary); |
|  | clrscr(); |
|  | db.seekg(0); |
|  | int i =0; //user object array counter |
|  | cout << "The Users Databbase: " << endl; |
|  | while(!db.eof()) { |
|  | db.read((char\*)&ts[i],sizeof(ts[i])); |
|  | ts[i].printUserData(); |
|  | cout << endl; |
|  | i++; |
|  | getch(); |
|  | } // end of while |
|  | } //end of read db function |
|  |  |
|  | //search user database for user |
|  | void searchUser(char\* login\_name,char\* login\_code) { |
|  | user ts[100]; |
|  | ifstream db; |
|  | db.open("users.txt",ios::in|ios::binary); |
|  | clrscr(); |
|  | db.seekg(0); |
|  | int i =0; //user object array counter |
|  | int flag = 0; |
|  | char uname[30]; |
|  | char pwd[30]; |
|  | sp(2,0); |
|  | while(!db.eof()) { |
|  | db.read((char\*)&ts[i],sizeof(ts[i])); |
|  | ts[i].uname(uname); |
|  | ts[i].pwd(pwd); |
|  | //comparing credentails |
|  | if(!strcmp(uname,login\_name)) { |
|  | if(!strcmp(pwd,login\_code)) { |
|  | clrscr(); |
|  | flag=1; |
|  | strcpy(current\_user,login\_name); |
|  | userControlPanel(uname); |
|  | getch(); |
|  | break; |
|  | } else { |
|  | clrscr(); |
|  | cout << "User identified with Wrong password"; |
|  | getch(); |
|  | abort(); |
|  | } |
|  | } else { |
|  | continue; // user does not match , check next record |
|  | } |
|  | i++; |
|  | } //end of while |
|  | if(!flag) { |
|  | clrscr(); |
|  | cout << "No such user exists ."; |
|  | getch(); |
|  | abort(); |
|  | } |
|  | } //end of fucntion searchUser |
|  | }; |
|  | //end: User class definition |
|  |  |
|  | //2. |
|  |  |
|  | //start:Chatroom class definition |
|  | class chatroom { |
|  | //participants |
|  | char name[30]; |
|  | char pc[30]; //passcode |
|  | int max; |
|  | int pri; //1 indicates private chatroom,0 indicates public chtaroom |
|  | public: |
|  | int ppsize ; |
|  | chatroom() { |
|  | ppsize = 0; |
|  | max =20; |
|  | strcpy(name,"empty"); |
|  | strcpy(pc,"empty"); |
|  | } |
|  | //custom max users |
|  | chatroom(int i) { |
|  | max =i; |
|  | } |
|  | //prototypes |
|  | void create(char \*creator); |
|  |  |
|  | //inline functions |
|  |  |
|  | //compares the name with input |
|  | int searchname(char input[30]) { |
|  | if(strcmp(name,input) == 0) { |
|  | return 1; |
|  | } else { |
|  | return 0; |
|  | } |
|  | } |
|  | int searchpwd(char input[30]) { |
|  | if(strcmp(pc,input) == 0) { |
|  | return 1; |
|  | } else { |
|  | return 0; |
|  | } |
|  | } |
|  | int isPri() { |
|  | return pri; |
|  | } |
|  | //prints out the name |
|  | void printName() { |
|  | cout << name; |
|  | } |
|  | } newChatroom,tc[50],rc[50] ; |
|  |  |
|  | void chatroom :: create(char \*creator) { |
|  | clrscr(); |
|  | char ch; |
|  | char uch; |
|  | //intializing p\_input and pp size |
|  | cout << endl << "Create new chatroom ? (Y/N): "; |
|  | cin >>ch; |
|  | if(ch == 'Y' || ch == 'y') { |
|  | cout << endl << "Enter name of chatroom: "; |
|  | gets(name); |
|  | //cap on max users is 100 by defulat |
|  | cout << "Add password ? [Y/N]:"; |
|  | cin >> uch; |
|  | if(uch == 'Y' || uch == 'y') { |
|  | cout << "Enter passcode: " ; |
|  | gets(pc); |
|  | pri = 1; |
|  | }else { |
|  | pri = 0; |
|  | } |
|  | char temp[30]; |
|  | strcpy(temp,name); |
|  | strcat(temp,".txt"); |
|  | //creating on empty txt file |
|  | fstream file; |
|  | file.open(temp,ios::out | ios::binary); |
|  | file.close(); |
|  | } else { |
|  | userControlPanel(creator); |
|  | } |
|  | }; //endof create function; |
|  |  |
|  | //end: Chatroom class defintion |
|  |  |
|  |  |
|  | //start: Homescreen functions |
|  |  |
|  | //User Registration |
|  | void newUser() { |
|  | user newUser; |
|  | clrscr(); |
|  | newUser.create(); |
|  | newUser.writeUserData(newUser); |
|  | getch(); |
|  | abort(); |
|  | } |
|  |  |
|  | //to handle existing users |
|  | void existingUser() { |
|  | //login screen |
|  | clrscr(); |
|  | char login\_name[30]; |
|  | char login\_pwd[30]; |
|  | sp(1,2); |
|  | cout << "Login: "; |
|  | cin.ignore(); |
|  | sp(3,2); |
|  | cout<<"Enter user name: "; |
|  | cin.getline(login\_name,30); |
|  | cin.ignore(0,'\n'); |
|  | sp(1,2); |
|  | cout<<"Enter password: "; |
|  | cin.getline(login\_pwd,30); |
|  | sp(1,2); |
|  | user temp\_loginuser; |
|  | temp\_loginuser.searchUser(login\_name,login\_pwd); |
|  | } ; |
|  |  |
|  | //User Menu Screen |
|  | void userControlPanel(char \*uname) { |
|  | int ch ; |
|  | strcpy(current\_user,uname); |
|  | sp(1,1); |
|  | cout << "Welcome back, " << uname; |
|  | sp(3,3); |
|  | cout << "1.Create Chatroom" ; |
|  | sp(2,3); |
|  | cout << "2.Join Chatroom"; |
|  | sp(2,3); |
|  | cout << "3.Back "; |
|  | sp(3,3); |
|  | cout << "Input ch number: " ; |
|  | cin >> ch; |
|  | if(ch == 1 ) { |
|  | //:start chatroom database handling (inserting records) |
|  | newChatroom.create(uname); |
|  | ofstream db("chatrooms.txt",ios::app|ios::binary); |
|  | db.write((char\*)&newChatroom,sizeof(newChatroom)); |
|  | db.close(); |
|  | sp(2,0); |
|  | cout<<"Chatroom registered !"; |
|  | getch(); |
|  | clrscr(); |
|  | userControlPanel(uname); |
|  | //:end chatroom database handling (inserting records) |
|  | } else if(ch == 2) { |
|  | //:start listing out records in chatrooms.txt |
|  | ifstream wb; |
|  | wb.open("chatrooms.txt",ios::in|ios::binary); |
|  | clrscr(); |
|  | sp(1,3); |
|  | cout << "Active Chatrooms" << endl; |
|  | sp(1,3); |
|  | wb.seekg(0); |
|  | int c =0; //user object array counter |
|  | int temp\_c =1; |
|  | while(!wb.eof()) { |
|  | wb.read((char\*)&rc[c],sizeof(rc[c])); |
|  | //checks for empty object |
|  | if(rc[c].searchname("empty")) { |
|  | break; |
|  | } |
|  | cout << temp\_c << "." ; |
|  | rc[c].printName(); |
|  | sp(1,3); |
|  | c++; |
|  | temp\_c++; |
|  | getch(); |
|  | } // end of while |
|  | //end: listing out records in chatrooms.txt |
|  | //:start chatroom database handling (searching records) |
|  | char login\_chatname[30]; //user\_input |
|  | char temp\_name[30]; |
|  | char login\_pwd[30]; |
|  | sp(2,0); |
|  | cout << "Enter name of chatroom: " ; |
|  | cin.ignore(); |
|  | gets(login\_chatname); |
|  | // search chatroom db for input here |
|  | ifstream db; |
|  | db.open("chatrooms.txt",ios::in|ios::binary); |
|  | clrscr(); |
|  | db.seekg(0); |
|  | int i =0; //user object array counter |
|  | int flag = 0; |
|  | sp(2,0); |
|  | while(!db.eof()) { |
|  | db.read((char\*)&tc[i],sizeof(tc[i])); |
|  | int search\_result = tc[i].searchname(login\_chatname); |
|  | if(search\_result == 1) { |
|  | if(tc[i].isPri()) { |
|  | cout << "Enter pwd of chatroom: " ; |
|  | gets(login\_pwd); |
|  | int pwd\_search\_result = tc[i].searchpwd(login\_pwd); |
|  | if(pwd\_search\_result == 1) { |
|  | flag =1; |
|  | chat(login\_chatname); |
|  | break; |
|  | } |
|  | } else { |
|  | chat(login\_chatname); |
|  | break; |
|  | } |
|  | } |
|  | } |
|  | if (flag==0) { |
|  | clrscr(); |
|  | cout << "No such chatroom exists or password entered is wrong."; |
|  | getch(); |
|  | db.close(); |
|  | abort(); |
|  | } |
|  | //:end chatroom database handling (searching records) |
|  | } else if(ch == 3) { |
|  | abort(); //back to homescreen |
|  | }else { |
|  | clrscr(); |
|  | cout << "invalid option" ; |
|  | getch(); |
|  | clrscr(); |
|  | userControlPanel(uname); |
|  | } |
|  | }; |
|  |  |
|  | //end: Homescreen functions |
|  |  |
|  | void main() { |
|  | clrscr(); |
|  | int user\_ch; |
|  | //Homescreen |
|  | sp(1,0); |
|  | cout << "LAN Messenger:"; |
|  | sp(3,3); |
|  | cout << "1.New User" ; |
|  | sp(1,3); |
|  | cout << "2.Existing user"; |
|  | sp(1,3); |
|  | cout << "3.Settings"; |
|  | sp(1,3); |
|  | cout << "4.Exit"; |
|  | sp(2,3); |
|  | cout << "Input path no. "; |
|  | cin >> user\_ch; |
|  | if(user\_ch == 1) { |
|  | //User Regstration path |
|  | newUser(); |
|  | }else if(user\_ch == 2) { |
|  | //Login path |
|  | existingUser(); |
|  | }else if(user\_ch == 3) { |
|  | //Admin Backend path |
|  | clrscr(); |
|  | cout << "Welcome to admin settings:" << endl; |
|  | cout << "Input pwd : "; |
|  | char pwd[10]; |
|  | gets(pwd); |
|  | if(!strcmp(pwd,"admin") || !strcmp(pwd,"ADMIN")){ |
|  | clrscr(); |
|  | user admin; |
|  | admin.readUserFromDb(); |
|  | }else if(!strcmp(pwd,"chat") || !strcmp(pwd,"CHAT")) { |
|  | ifstream wb; |
|  | wb.open("chatrooms.txt",ios::in|ios::binary); |
|  | clrscr(); |
|  | wb.seekg(0); |
|  | int c =0; //user object array counter |
|  | while(!wb.eof()) { |
|  | wb.read((char\*)&rc[c],sizeof(rc[c])); |
|  | //checks for empty object |
|  | if(rc[c].searchname("empty")) { |
|  | break; |
|  | } |
|  | cout << c << "." ; |
|  | rc[c].printName(); |
|  | cout << endl; |
|  | c++; |
|  | getch(); |
|  | } // end of while |
|  | }else { |
|  | cout << "Wrong password"; |
|  | } |
|  | } else if(user\_ch == 4) { |
|  | abort(); |
|  | }else { |
|  | clrscr(); |
|  | cout << "Invalid choice" << endl; |
|  | getch(); |
|  | clrscr(); |
|  | abort(); |
|  | } |
|  |  |
|  | getch(); |
|  | } |

OUTPUT SCREENS



BIBLEOGRAPHY

* GITHUB.COM
* GEEKSFORGEEKS.ORG
* STACKOVERFLOW.COM