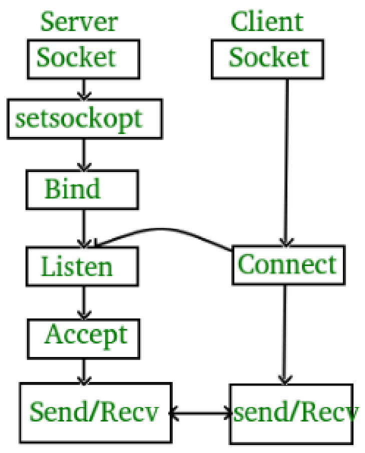
FINAL PROJECT – CYBER MANAGEMENT SYSTEM
BY GROUP 4

*“ **Cyber Management System***

The system consists of 2 modules: Client and Server. The server module is responsible for managing settings and client requests. On the other hand, the client component provides clients with internet access to the location's services. Both these parts have separate source code and run in sync with one another. The cyber management system can be programmed by utilizing the advanced features of C, such as socket programming and multithreading.

Topics covered: Socket Programming (basics), multithreading.”

In this project, we used socket programming to connect the server and the client. After the server and the client components connected successfully, the client sent the text file “cyber.txt” generated by the “cyber_system.c” program to the server; the server sync the text file and converted the received file to a new text file named “cyber_received.txt”.



Therefore, this project includes the following files:

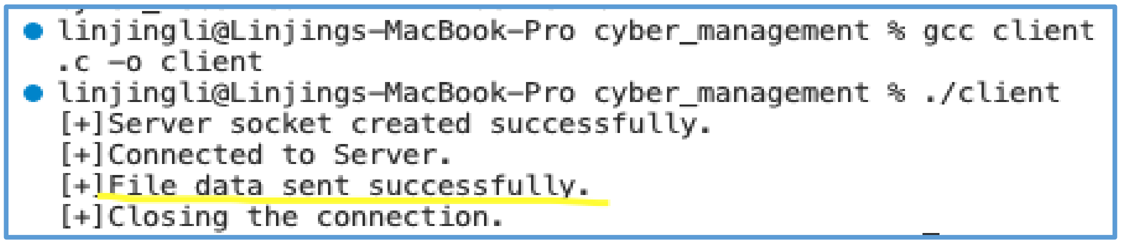
1) cyber_system.c file: provided for the user to generate, manipulate and store the student data if they log in with correct username and password. The final student data was saved in cyber.txt.

2) client.c and server.c: contains codes for connection with socket programing and methods to send file and receive file.

3) the server component accepts the connect request from the client and syncs the file sent by its connected client, and the received file is named "cyber_received.txt".

" client.c":

includes main() function and send_file() function



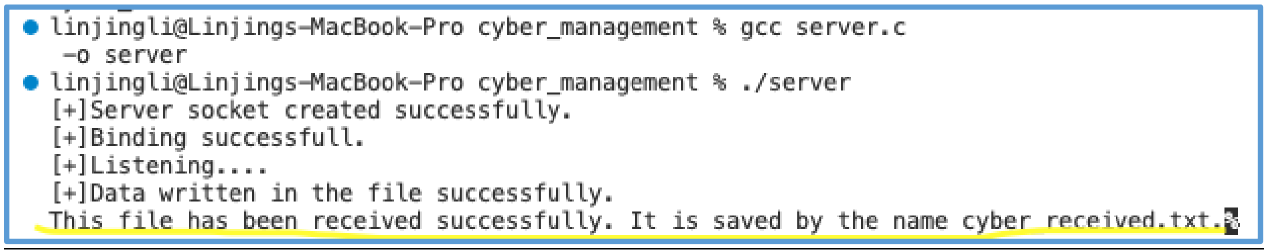
```
● linjingli@Linjings-MacBook-Pro cyber_management % gcc client
  .c -o client
● linjingli@Linjings-MacBook-Pro cyber_management % ./client
  [+]Server socket created successfully.
  [+]Connected to Server.
  [+]File data sent successfully.
  [+]Closing the connection.
```

```
int main(){
  char *ip = "127.0.0.1"; //the ip address for local device
  int port = 8080; //port number provided by the client
  int e; //connect status

  //create a socket
  int sockfd;
  struct sockaddr_in server_addr;

  FILE *fp;
  char *filename = "cyber.txt";

  //int socket(int domain, int type, int protocol);
  //socket creates an endpoint and returns its file descriptor
  //Domain AF_INET means: Internet Protocol IPv4
  //SOCK_STREM means: sequential and reliable 2 way communication.

  sockfd = socket(AF_INET, SOCK_STREAM, 0);
  if(sockfd < 0) {
    perror("[-]Error in socket");
    exit(1);
  }
  printf("[+]Server socket created successfully.\n");

  server_addr.sin_family = AF_INET;
  server_addr.sin_port = port;
  server_addr.sin_addr.s_addr = inet_addr(ip);

  //connect a socket
  e = connect(sockfd, (struct sockaddr*)&server_addr, sizeof(server_addr));
  if(e == -1) {
    perror("[-]Error in socket");
    exit(1);
  }
  printf("[+]Connected to Server.\n");

  // fopen()-opens the file for reading;
  //"r": Opens a file for reading. The file must exist.
  fp = fopen(filename, "r");
  if (fp == NULL) {
    perror("[-]Error in reading file.");
    exit(1);
  }

  //send "cyber.txt" by send_file() method
  send_file(fp, sockfd);

  printf("[+]File data sent successfully.\n");

  printf("[+]Closing the connection.\n");

  //close socket
  close(sockfd);

  return 0;
}
```

```
send_file() method:
void send_file(FILE *fp, int sockfd){
  int n;
  char data[SIZE] = {0};
  while(fgets(data, SIZE, fp) != NULL) {
    if (send(sockfd, data, sizeof(data), 0) == -1) {
      perror("[-]Error in sending file.");
      exit(1);
    }
    // bzero() places size null bytes in the char data, is used to set all the socket structures with null values
    bzero(data, SIZE);
  }
}
```

“ server.c”:

includes main() function and write_file() function

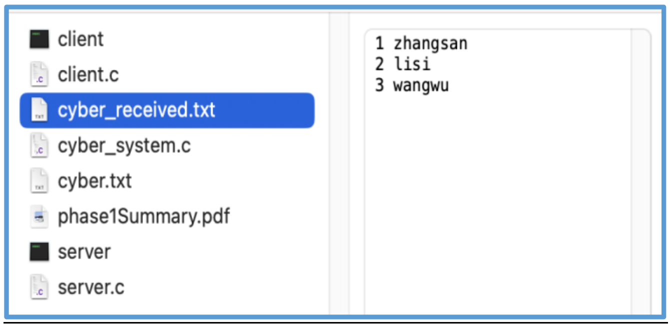


```
linjingli@Linjings-MacBook-Pro cyber_management % gcc server.c
 -o server
linjingli@Linjings-MacBook-Pro cyber_management % ./server
[+]Server socket created successfully.
[+]Binding successfull.
[+]Listening....
[+]Data written in the file successfully.
This file has been received successfully. It is saved by the name cyber_received.txt.
```

```
int main(){
  char *ip = "127.0.0.1";  //the ip address for local device
  int port = 8080;  //port number provided by the client
  int e;  //connect status

  //create sockets
  int sockfd, new_sock;
  struct sockaddr_in server_addr, new_addr;
  socklen_t addr_size;
  char buffer[SIZE];

  sockfd = socket(AF_INET, SOCK_STREAM, 0);
  if(sockfd < 0) {
    perror("[-]Error in socket");
    exit(1);
  }
  printf("[+]Server socket created successfully.\n");

  server_addr.sin_family = AF_INET;
  server_addr.sin_port = port;
  server_addr.sin_addr.s_addr = inet_addr(ip);

  //bind:
  e = bind(sockfd, (struct sockaddr*)&server_addr, sizeof(server_addr));
  if(e < 0) {
    perror("[-]Error in bind");
    exit(1);
  }
  printf("[+]Binding successfull.\n");

  // listen:
  if(listen(sockfd, 10) == 0){
    printf("[+]Listening....\n");
  }else{
    perror("[-]Error in listening");
    exit(1);
  }

   // accept:
  addr_size = sizeof(new_addr);
  new_sock = accept(sockfd, (struct sockaddr*)&new_addr, &addr_size);

   // write file to server
  write_file(new_sock);
  printf("[+]Data written in the file successfully.\n");
  printf("This file has been received successfully. It is saved by the name cyber_received.txt.");

  return 0;
}
```

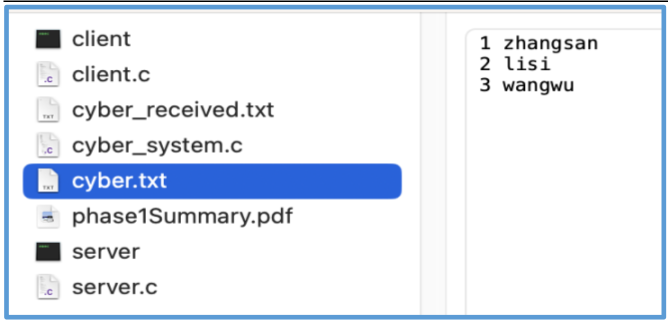
```
write_file() method:

void write_file(int sockfd){
  int n;
  FILE *fp;
  char *filename = "cyber_received.txt";
  char buffer[SIZE];

  fp = fopen(filename, "w");
  while (1) {
    n = recv(sockfd, buffer, SIZE, 0);
    if (n <= 0){
      break;
      return;
    }
    fprintf(fp, "%s", buffer);
    bzero(buffer, SIZE);
  }
  return;
}
```

In the local folder where the programs locate, we can see a new txt file named “cyber_received.txt” was created and its content is the same as “cyber.txt”.



client
client.c
cyber_received.txt
cyber_system.c
cyber.txt
phase1Summary.pdf
server
server.c

```
1 zhangsan
2 lisi
3 wangwu
```



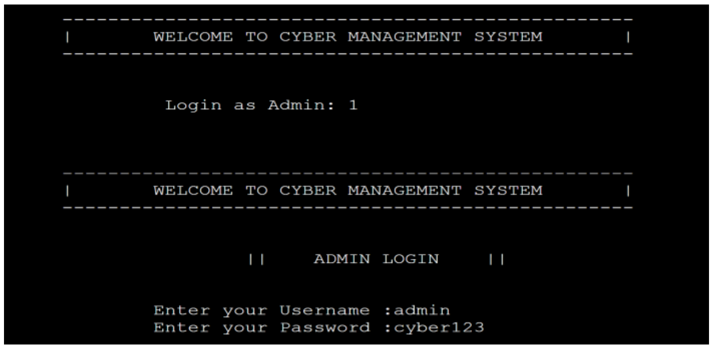
client
client.c
cyber_received.txt
cyber_system.c
cyber.txt
phase1Summary.pdf
server
server.c

```
1 zhangsan
2 lisi
3 wangwu
```

“cyber_system.c”

includes main() function, admin login method, insert_student(), display(), search_student(), delete_student() methods.

1) Users are allowed to enter the administrator system by entering “1”. When the instructions “Enter your username” and “Enter your password” appear, the corresponding input “admin” and “cyber123” are supposed to be given.



```
 ---------------------------------------------------
 |        WELCOME TO CYBER MANAGEMENT SYSTEM       |
 ---------------------------------------------------


          Login as Admin: 1


 ---------------------------------------------------
 |        WELCOME TO CYBER MANAGEMENT SYSTEM       |
 ---------------------------------------------------


                 ||    ADMIN LOGIN    ||


         Enter your Username :admin
         Enter your Password :cyber123
```

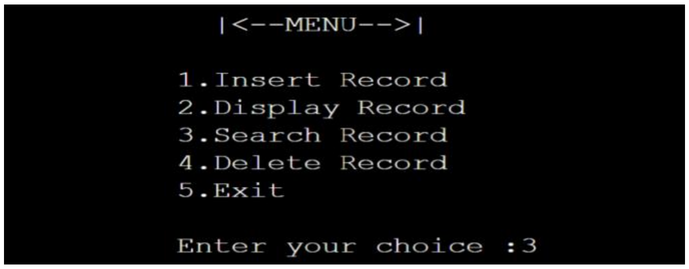
Codes for creating the above:

```
void main()
{
   char name[30];
   int id;
```

```
    int ch;
    char username[20];
    char password[20];

    system("CLS");
    system("color 8F");
    heading();
    printf("\n\n\t\t\t\t\t Login as Admin: ");
    scanf("%d",&ch);
    system("CLS");
```

```
        case 1:system("CLS");
              heading();
              printf("\n\n\t\t\t\t\t\t||    ADMIN LOGIN    ||\t\t\t\t");
              printf("\n\n\n\t\t\t\t\tEnter your Username :");
              scanf("%s",&username);
              printf("\t\t\t\t\tEnter your Password :");
              scanf("%s",&password);

              if(strcmp(username, "admin")==0)
              {
                 if(strcmp(password ,"cyber123")==0)
                 {
                     system("CLS");
                     main_heading();
                     printf("\n\n\t\t\t\t\tWELCOME ADMIN!!! LOGIN SUCCESSFULL");
                     menu();
                 }
                 else
                 {
                     printf("\t\t\t\t\tIncorrect Password !!! Failed to Login");
                 }
              }
              else
              {
                  printf("\t\t\t\t\tUsername is invalid !!! Failed to Login");
              }

               break;
```

2) After successfully login, 5 operation options will be provided: insert(1), display(2), search(3), delete(4) and exit(5). People need to input different numbers to call different functions.

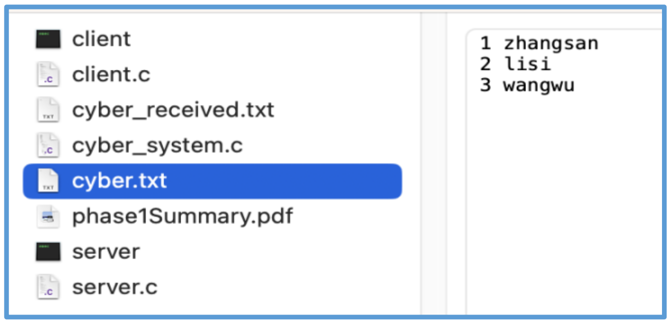


```
|<--MENU-->|

1.Insert Record
2.Display Record
3.Search Record
4.Delete Record
5.Exit

Enter your choice :3
```

Codes for creating the above:

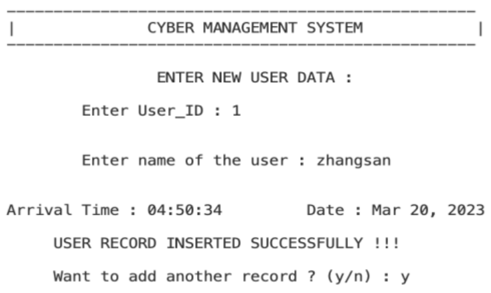
```
void menu()
{
int choice;
printf("\n\n\n\t\t\t\t\t\t  |<--MENU-->|");
printf("\n\n\t\t\t\t\t\t1.Insert Record");
printf("\n\t\t\t\t\t\t2.Display Record");
printf("\n\t\t\t\t\t\t3.Search Record");
printf("\n\t\t\t\t\t\t4.Delete Record");
printf("\n\t\t\t\t\t\t5.Exit");
printf("\n\n\t\t\t\t\t\tEnter your choice :");
scanf("%d",&choice);

switch(choice)
{
case 1: system("CLS");
insert();
break;

case 2: system("CLS");
main_heading();
display();
printf("\n\n\t\t\t\t\tPress any key to continue ");
fflush(stdin);
getchar();
menu();

case 3: system("CLS");
search();
printf("\n\n\t\t\t\t\tPress any key to continue ");
fflush(stdin);
getchar();
menu();
break;

case 4: system("CLS");
delete();
break;
case 5: system("CLS");
exit(0);
break;

default: printf("INVALID CHOICE !!!");
}
}
```

3) If option "1" is input, then the admin is allowed to input the student ID and student name.
By implementing the insert() function, our code is able to inform the administrator that the student records are successfully created by printing "*USER RECORD INSERTED SUCCESSFULLY !!!"*
And a text file named "cyber.txt" was generated in the local folder storing this student information.



```
1 zhangsan
2 lisi
3 wangwu
```

If more student records needed to add, the admin can continue by hitting "y" key.



```
-------------------------------------------------------
|                CYBER MANAGEMENT SYSTEM               |
-------------------------------------------------------

              ENTER NEW USER DATA :
         Enter User_ID : 1

         Enter name of the user : zhangsan

Arrival Time : 04:50:34          Date : Mar 20, 2023
      USER RECORD INSERTED SUCCESSFULLY !!!
      Want to add another record ? (y/n) : y
```

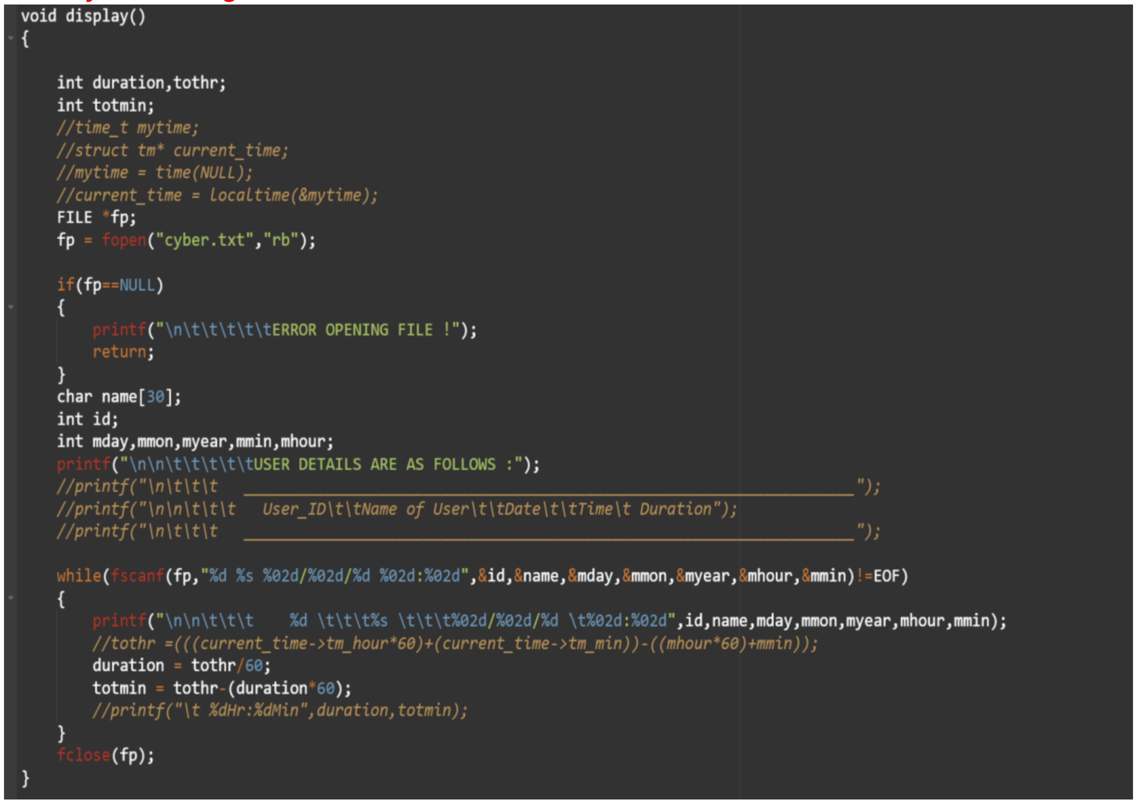
Codes for creating the above:

```
void insert()
{
   time_t mytime;
   struct tm* current_time;
   mytime = time(NULL);
   current_time = localtime(&mytime);
```

```
    FILE *fp;
    char name[30];
    int id;
    char choice='y';
    system("CLS");
    fp = fopen("cyber.txt","ab+");

    if(fp==NULL)
    {
        printf("\n\t\t\t\tERROR OPENING FILE !!!");
        return;
    }
    fflush(stdin);
    while(choice=='y')
    {
        main_heading();
        printf("\n\n\t\t\t\t\t\tENTER NEW USER DATA :");
        printf("\n\n\t\t\t\t\tEnter User_ID : ");
        scanf("%d",&id);
        fflush(stdin);
        printf("\n\n\t\t\t\t\tEnter name of the user : ");
        scanf("%s",&name);
        fflush(stdin);
        fprintf(fp,"%d %s\t",id,name);

fprintf(fp,"%02d/%02d/%d %02d:%02d",current_time->tm_mday,current_time->tm_mon+1,current_time->tm
_year+1900,current_time->tm_hour,current_time->tm_min);
        fprintf(fp,"\n");
        arrival_time();
        printf("\n\n\t\t\t\t     USER RECORD INSERTED SUCCESSFULLY !!!");

        printf("\n\n\t\t\t\t     Want to add another record ? (y/n) : ");
        scanf("%c",&choice);
        fflush(stdin);
        system("CLS");
        fflush(stdin);
    }
    fclose(fp);
    printf("\n\n\t\t\t\t\t     Press any key to continue ");
    getchar();
    menu();
}
```

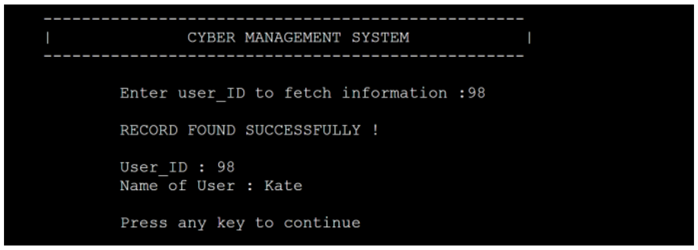
4) If option “2” is input, then current student information created by administrators will be displayed.

Codes for creating the above:



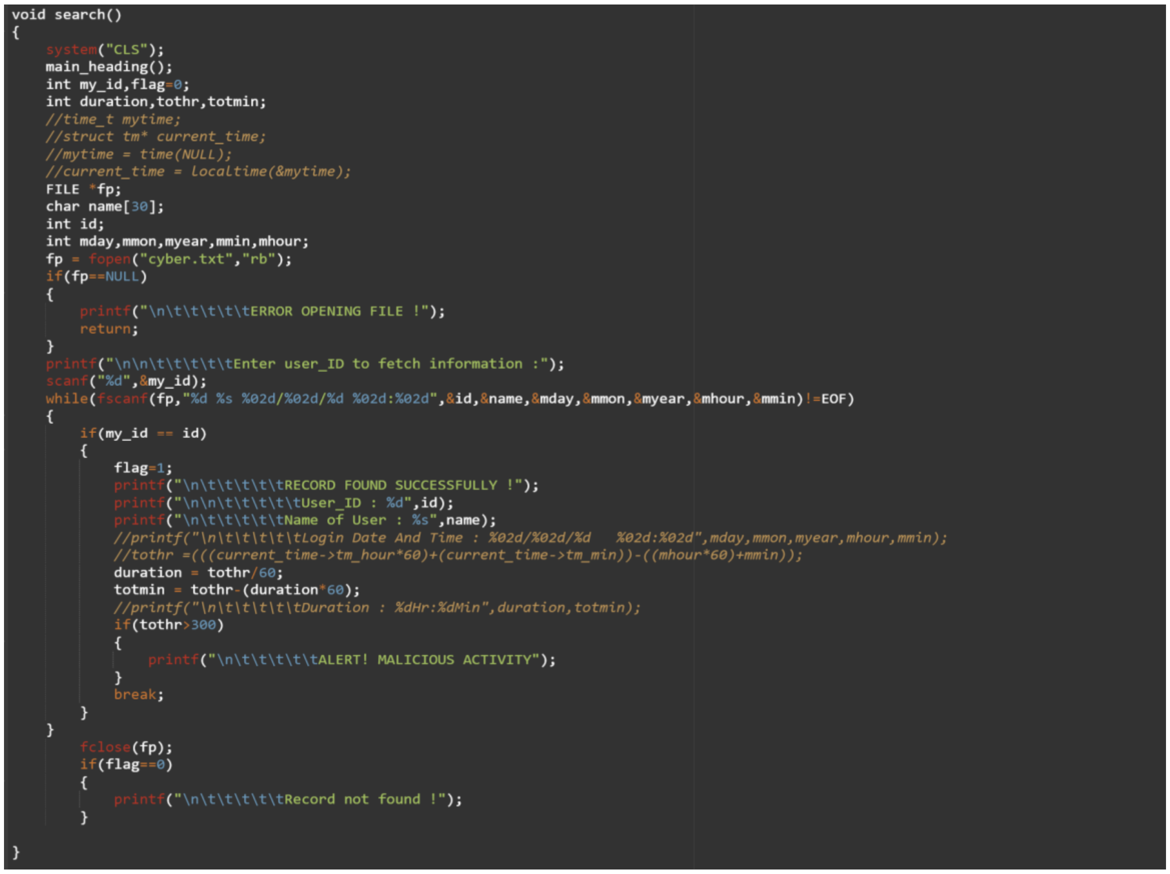
```c
void display()
{

    int duration,tothr;
    int totmin;
    //time_t mytime;
    //struct tm* current_time;
    //mytime = time(NULL);
    //current_time = localtime(&mytime);
    FILE *fp;
    fp = fopen("cyber.txt","rb");

    if(fp==NULL)
    {
        printf("\n\t\t\t\t\tERROR OPENING FILE !");
        return;
    }
    char name[30];
    int id;
    int mday,mmon,myear,mmin,mhour;
    printf("\n\n\t\t\t\t\tUSER DETAILS ARE AS FOLLOWS :");
    //printf("\n\t\t\t   ______________________________________________________________________");
    //printf("\n\n\t\t\t   User_ID\t\tName of User\t\tDate\t\tTime\t Duration");
    //printf("\n\t\t\t   ______________________________________________________________________");

    while(fscanf(fp,"%d %s %02d/%02d/%d %02d:%02d",&id,&name,&mday,&mmon,&myear,&mhour,&mmin)!=EOF)
    {
        printf("\n\n\t\t\t    %d \t\t\t%s \t\t\t%02d/%02d/%d \t%02d:%02d",id,name,mday,mmon,myear,mhour,mmin);
        //tothr =(((current_time->tm_hour*60)+(current_time->tm_min))-((mhour*60)+mmin));
        duration = tothr/60;
        totmin = tothr-(duration*60);
        //printf("\t %dHr:%dMin",duration,totmin);
    }
    fclose(fp);
}
```

5) If option “3” is input, then search() function will be called. Given an ID, people can find a corresponding name.



```
    --------------------------------------------------
    |               CYBER MANAGEMENT SYSTEM           |
    --------------------------------------------------

            Enter user_ID to fetch information :98

            RECORD FOUND SUCCESSFULLY !

            User_ID : 98
            Name of User : Kate

            Press any key to continue
```

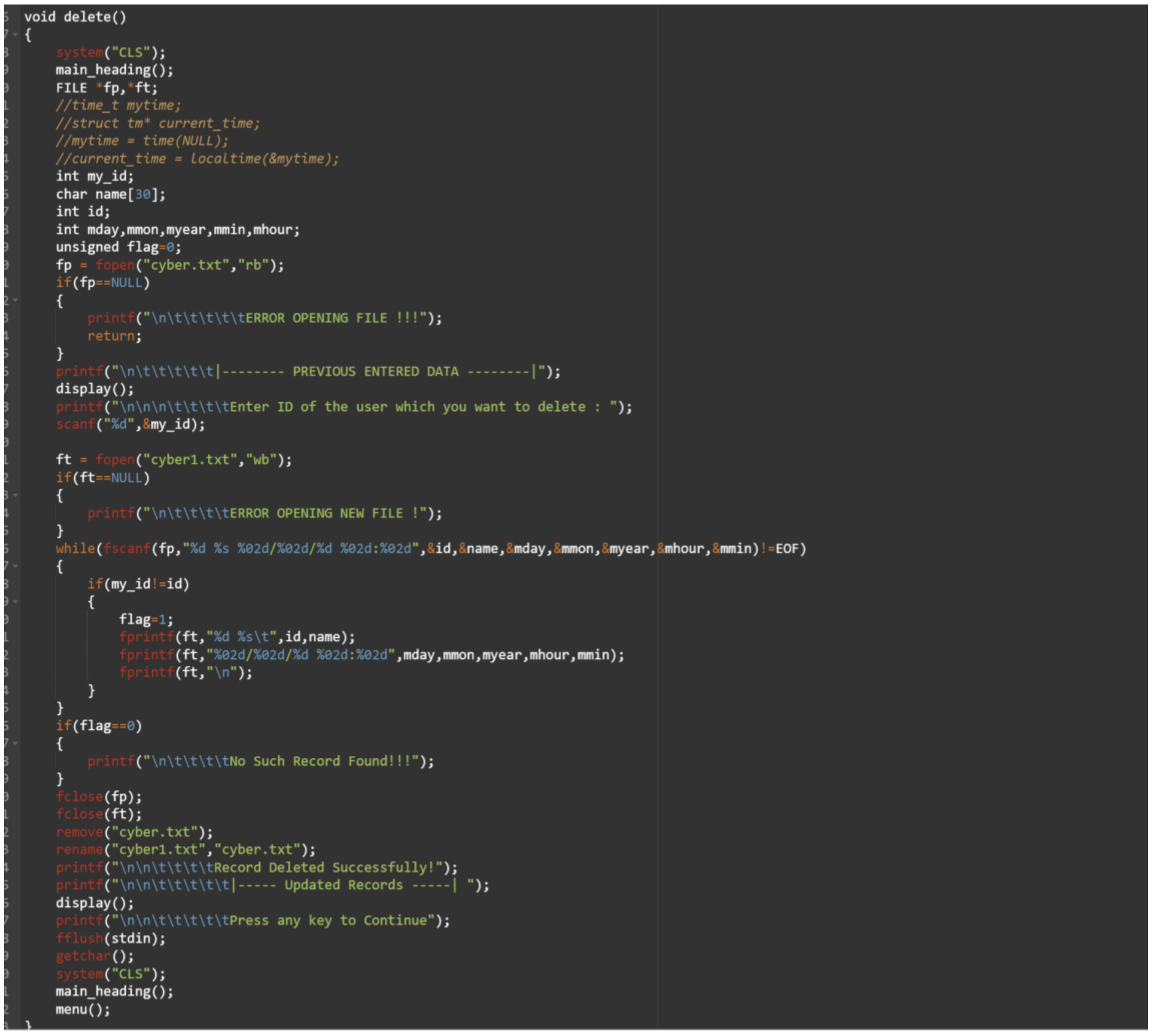
Codes for creating the above:



```c
void search()
{
    system("CLS");
    main_heading();
    int my_id,flag=0;
    int duration,tothr,totmin;
    //time_t mytime;
    //struct tm* current_time;
    //mytime = time(NULL);
    //current_time = localtime(&mytime);
    FILE *fp;
    char name[30];
    int id;
    int mday,mmon,myear,mmin,mhour;
    fp = fopen("cyber.txt","rb");
    if(fp==NULL)
    {
        printf("\n\t\t\t\t\tERROR OPENING FILE !");
        return;
    }
    printf("\n\n\t\t\t\t\tEnter user_ID to fetch information :");
    scanf("%d",&my_id);
    while(fscanf(fp,"%d %s %02d/%02d/%d %02d:%02d",&id,&name,&mday,&mmon,&myear,&mhour,&mmin)!=EOF)
    {
        if(my_id == id)
        {
            flag=1;
            printf("\n\t\t\t\t\tRECORD FOUND SUCCESSFULLY !");
            printf("\n\n\t\t\t\t\tUser_ID : %d",id);
            printf("\n\t\t\t\t\tName of User : %s",name);
            //printf("\n\t\t\t\t\tLogin Date And Time : %02d/%02d/%d   %02d:%02d",mday,mmon,myear,mhour,mmin);
            //tothr =(((current_time->tm_hour*60)+(current_time->tm_min))-((mhour*60)+mmin));
            duration = tothr/60;
            totmin = tothr-(duration*60);
            //printf("\n\t\t\t\t\tDuration : %dHr:%dMin",duration,totmin);
            if(tothr>300)
            {
                printf("\n\t\t\t\t\tALERT! MALICIOUS ACTIVITY");
            }
            break;
        }
    }
        fclose(fp);
        if(flag==0)
        {
            printf("\n\t\t\t\t\tRecord not found !");
        }

}
```

6) If option “4” is input, then delete() function will be called. Given an ID, people are able to remove a corresponding name.

Codes for creating the above:



```c
void delete()
{
    system("CLS");
    main_heading();
    FILE *fp,*ft;
    //time_t mytime;
    //struct tm* current_time;
    //mytime = time(NULL);
    //current_time = localtime(&mytime);
    int my_id;
    char name[30];
    int id;
    int mday,mmon,myear,mmin,mhour;
    unsigned flag=0;
    fp = fopen("cyber.txt","rb");
    if(fp==NULL)
    {
        printf("\n\t\t\t\t\tERROR OPENING FILE !!!");
        return;
    }
    printf("\n\t\t\t\t\t|-------- PREVIOUS ENTERED DATA --------|");
    display();
    printf("\n\n\n\t\t\t\tEnter ID of the user which you want to delete : ");
    scanf("%d",&my_id);

    ft = fopen("cyber1.txt","wb");
    if(ft==NULL)
    {
        printf("\n\t\t\t\tERROR OPENING NEW FILE !");
    }
    while(fscanf(fp,"%d %s %02d/%02d/%d %02d:%02d",&id,&name,&mday,&mmon,&myear,&mhour,&mmin)!=EOF)
    {
        if(my_id!=id)
        {
            flag=1;
            fprintf(ft,"%d %s\t",id,name);
            fprintf(ft,"%02d/%02d/%d %02d:%02d",mday,mmon,myear,mhour,mmin);
            fprintf(ft,"\n");
        }
    }
    if(flag==0)
    {
        printf("\n\t\t\t\tNo Such Record Found!!!");
    }
    fclose(fp);
    fclose(ft);
    remove("cyber.txt");
    rename("cyber1.txt","cyber.txt");
    printf("\n\n\t\t\t\tRecord Deleted Successfully!");
    printf("\n\n\t\t\t\t\t|----- Updated Records -----| ");
    display();
    printf("\n\n\t\t\t\t\tPress any key to Continue");
    fflush(stdin);
    getchar();
    system("CLS");
    main_heading();
    menu();
}
```

7) If option “5” is input, then the administrator will leave the operation system.

<u>End of this Summary.</u>