به نام خدا

پروژه پایان‌ترم

مبانی برنامه‌نویسی

اپلیکیشن چت

گردآورنده:محمد جواد سعیدی

شماره دانشجویی:98105802

زمستان 98

مقدمه

در مقدمه توضیحی کلی در مورد پروژه آن می‌دهم.

خب پروژه ما ساخت چت‌اپلیکیشن بود البته شاید با شنیدن این اسم فکر کنید خیلی پروژه خفن و سنگینی(البته سنگین یکم بود!!) بود ولی در واقع چت‌اپلیکیشن ما توانایی ارسال پیام فقط دارد و از ارسال تصویر و موسیقی و فیلم در آن خبری نیست.

پروژه ما در ۳ فاز جداگانه انجام شد که در فاز ۱ کلاینت برنامه‌نویسی شد و در فاز ۲ سرور و در فاز ۳ سرور و کلاینت یکم بیشتر ساخته دست خودمون شد و یک سری موارد مثل همین مستند سازی که در خدمت شما هست آماده شد.

راستی ما در دو فاز اول از مقوله‌ای به نام جیسون استفاده کردیم که در فاز سوم باید این جیسون رو خودمون برنامه‌نویسی می‌کردیم و لازم بود که کلاینت و سرور را با همین جیسونی که خودمون نوشتیم دوباره بازنویسی کنیم.

خب در ادامه فاز اول و دوم پروژه را بضورت مبسوط تری توضیح می‌دهیم...

فاز اول : کلاینت

خب در فاز اول ما باید با استفاده از سرور هوشمندی که به ما داده بودند کلاینت (یا در اصل محیط کاربری کلاینت)را برنامه نویسی می‌کردیم خب این یعنی چی:

ابتدا ما با استفاده از دستور printf یک منوی کاربری را برای کلاینت ظاهر می‌کردیم که که کلاینت باید در آن login یا register را انتخاب می‌کرد که در زیر شکل آن‌را مشاهده می‌کنید.

```
Account Menu:
1: Register
2: Login
```

اگر کاربر دستور register را انتخاب می‌کرد از کاربر نام کاربری و پسورد مورد نظر را می‌گرفتیم و آنرا در قالب یک رشته برای سرور می‌فرستیم تا بررسی کند که این نام کاربری قبلا وجود داشته است یا نه که اگر وجود داشته خطا برگرداند و در غیر اینصورت اطلاعات کاربر را ذخیره کند و پیغام موفقیت برای ما بفرستد که ما آنرا در قالب یک جیسون چاپ می‌کنیم.

در شکل زیر نمونه‌ای از کد کلاینت را که مربوط به قسمت ارسال رشته توسط کلاینت و دریافت رشته‌ای از سرور و تبدیل آن به جیسون و پرینت آن هست مشاهده می‌کنیم.

```
if (connect(client_socket, (SA*)&servaddr, sizeof(servaddr)) != 0)
{
    printf("Connection to the server failed...\n");
    exit(0);
}
send(client_socket, buffer, sizeof(buffer), 0);
memset(buffer, 0, sizeof(buffer));
recv(client_socket, buffer, sizeof(buffer), 0);
closesocket(client_socket);
cJSON * recieved=cJSON_Parse(buffer);
printf("%s\n",cJSON_GetObjectItemCaseSensitive(recieved,"type")->valuestring);
```

و پس از بررسی register نوبت قسمت لاگین می‌رسد که روندی مانند بالا دارد و فقط تفاوتش این است که اگر لاگین با موفقیت انجام شد باید سرور یک توکن بفرستد که این توکن به نوعی شناسه هر کاربر برای سرور می‌باشد که ما توکن را داخل یک رشته ریختیم و رشته را گلوبال تعریف کردیم که تا آخر از آن استفاده کنیم.

خب پس از اینکه کاربر با موفقیت لاگین کرد و موفق شد توکن خودش رو دریافت کنه وارد منوی اصلی می‌شود که مانند زیر است.

```
1: Create Channel
2: Join Channel
3: Logout
```

که کاربر پس از انتخاب creat channel به ما اسم کانالی را که می‌خواهد ایجاد کند به همراه توکن برای ما می‌فرستد را می‌دهد و ما اسم کانال را برای سرور به همراه توکن می‌فرستیم و سرور چک می‌کند که اولا توکن درست باشد و دوما نام کانال تکراری نباشد و اگر درست بود ما به صورت جیسون پیام موفق بودن را برای کاربر نمایش می‌دهیم و کاربر وارد منوی گفت و گو می‌شود.

برای جوین شدن در کانال نیز به ترتیب بالا کاربر نام کانال مورد علاقه و توکن خود را می‌فرستد و اگر چنین کانالی وجود داشت و کاربر قبلا در آن عضو نبود پیام موفقیت آمیز بودن را برای کاربر می‌فرستیم.

قسمت لاگ اوت هم برای این است که سرور نام کااربر را از بین کاربران آنلاین خارج کند و در اینجا هم کاربر باید توکن را بفرستد و در صورت تایید شدن توکن کاربر خارج می‌شود و وارد منوی کاربری می‌شود.

خب به منوی چت می‌رسیم که ظاهر آن مانند زیر است:

```
1: Send Message
2: Refresh
3: Channel Members
4: Leave Channel
```

که کاربر اگر بخواهد پیام بفرستد ما پیام ارسالی را به همراه توکن برای سرور می‌فرستیم که اگر توکن درست بود سرور پیام موفقیت آمیز می‌فرستد و ما آنرا به صورت جیسون برای کاربر پرینت می کنیم.

اما سراغ رفرش کردن می‌رویم که شاید هم در قسمت سرور و هم در قسمت کلاینت چالش برانگیزترین قسمت بود که دراین بخش قرار است آخرین پیام های دیده نشده ارسالی در آن گروه موردنظر را برای کاربر نمایش دهیم برای چاپ کردن این پیام ها به فرمت مورد نظر باید از cJSON_print استفاده می‌کردیم که نمونه کد آن قسمت را در زیر می‌بینیم:

```
send(client_socket, buffer, sizeof(buffer), 0);
memset(buffer, 0, sizeof(buffer));
recv(client_socket, buffer, sizeof(buffer), 0);
closesocket(client_socket);
cJSON * recieved=cJSON_Parse(buffer);
printf("%s\n",cJSON_Print(cJSON_GetObjectItemCaseSensitive(recieved, "content")));
```

در قسمت اعضای کانال قرار است لیست افراد حاظر در کانال را نمایش می‌دهیم که فقط توکن را برای سرور می‌فرستیم و اگه درست بود که سرور لیست اعضا را برای ما نمایش می‌داد.

در قسمت خارج شدن کانال هم مانند بالا عمل می‌کند و سرور نام کاربر را از لیست اعضای کانال خارج می‌سازد و کاربر وارد منوی اصلی می‌شود.

در آخر این بخش اگر بخواهم توضیح مختصری در مورد الگوریتم خودم بدهم می‌توانم این گونه بگویم که من یک حلقه کلی while داشتم که در ۴ تا state مختلف داشتم و که state = 0 برای ریجستر کردن و state = 1 برای لاگین کردن بود و state = 2 برای منوی اصلی بود که در آن ۳ تا if داشتم که هر کدام برای یکی از ۳ قسمت منوی اصلی بود و بعد اینکه هرکدام از کاراها انجام می‌شد یا نمی‌شد و سرور خطا بر می‌گرداند state تغییر می‌کرد تا در دور بعدی حلقه وارد قسمت مربوطه بشود. و اگر عملیات ایجاد کانال یا جوین شدن به کانال با موفقیت انجام می‌شد state = 3 می‌شد و کاربر وارد منوی گفت‌وگو می‌شد که رد آن هم ۴تا حالت مختلف در نظر گرفته شد بود.

در تابع main هم چیز خاصی ننوشتم و تمامی اتفاقات در تابع chat افتاد.

خب امیدوارم از بخش کلاینت یا همون فاز اول پروژه استفاده کرده باشید و توضیحات من برای‌تان کاربردی و قابل فهم بوده باشد حالا وارد فاز دوم پروژه یا همون سرور چت اپلیکیشن می‌شویم.

فاز دوم : سرور

خب رسیدیم به فاز دوم پروژه یا همون سرور چت‌اپلیکیشن.

فاز دوم پروژه از ما می‌خواست که سرور را برنامه نویسی کنیم که این فاز برخلاف فاز اول بیشتر به کار با فایل و استراکت نیاز داشت.

من در ابتدای برنامه اومدم و دوتا آرایه‌ای از استراکت به اسم هایkarbar و channels ایجاد کردم که استراکت دوم خیلی کاربردی نبود و استراکت اول ۳ تا بخش داشت که اولیش نام کاربری کاربر و دومیش توکن آن کاربر و سومیش نام کانالی که کاربر در آن حضور دارد(این نکته را هم باید بدانید که هرکاربر دریک لحظه تنها می‌تواند در یک کانال عضو باشد.) در استراکت channels نیز هم دوتا رشته وجود دارد که یکی نام کاربرانی است که در آن کانال هستند و دیگری نام خود کانال است.

می‌دانیم کلاینت برای هر درخواست خودش یک رشته به اسم buffer برای ما می‌فرستاد که کلمه اول آن شامل کاری که از سرور می‌خواهد(مثل ریجستر کردن یا لاگین کردن و این ها) و ما این رشته را با استفاده از sscanf کلمه اولش را داخل یک رشته به نام type می‌ریختیم که در هر if یا else if ای که داشتیم که چک می‌کرد type چیست که این کار با استفاده از strcmp صورت می‌گرفت.

خب قسمت به قسمت جلو می‌رویم.

ریجستر کردن:

در این قسمت ابتدا با استفاده از دوتا for نام کاربری و پسورد را از رشته‌ای که کاربر فرستاده خارج می‌کردیم و و ابتدا بوسیله یک تابع به اسم compare چک می‌کنیم که در فایل users چنین نام کاربری‌ای وجود دارد یا نه. تابع compare را در زیر مشاهده می‌کنید. اگر ۰ را برگرداند یعنی چنین کاربری قبلا ریجستر کرده و باید پیغام مربوطه را برای کاربر با استفاده از جیسون می‌فرستادیم.

```
int compare(char a[])
{
    DIR *dir;
    struct dirent *ent;
    if ((dir = opendir ("resources\\users"))!= NULL)
    {
        while ((ent = readdir (dir)) != NULL)
        {
            if(strcmp(ent->d_name,a)==0)
            {
                closedir(dir);
                return 0;
            }
        }
        closedir (dir);
        return 1;
    }
}
```

سپس بعد از بررسی عدم تکرااری بودن نام کاربری با استفاده از جیسون برای کاربر پیام موفقیت آمیز می‌فرستادیم که فرمت ارسال پیام ها به صورت زیر است.

```
cJSON *response;
response = cJSON_CreateObject();
cJSON_AddItemToObject(response, "type", cJSON_CreateString("Successful"));
cJSON_AddItemToObject(response, "content", cJSON_CreateString(""));
memset(buffer, 0, sizeof(buffer));
strcpy(buffer, cJSON_PrintUnformatted(response));
printf("%s",buffer);
send(client_socket, buffer, sizeof(buffer), 0);
```

که جیسون در اینجا به ما چنین فرمتی را می‌دهد که برای کلاینت خوانا است.

```
{"type":"Successful","content":""}
```

و در ادامه ما فایلی به اسم نام کاربری در پوشه users ایجاد کردیم که محتوایات فایل رمز عبور کاربر می‌باشد.

لاگین کردن:

در این بخش مانند بخش بالا ابتدا توسط تابع compare چک می‌کنیم که چنین نام کاربری‌ای وجود دارد یا ندارد که اگر وجود نداشت خطا برمی‌گردانیم و بعد از آن شرط آنلاین بودن کاربر را چک می‌کنیم بگونه‌ای که اگر استراکت کاربری با چنین نام کاربری ای وجود داشت اجازه لاگین کردن مجدد به کاربر نمی‌دهیم. که این کار توسط تابع compare_online انجام می‌شود که این تابع شامل یک حلقه است که از بین تمام کاربران اگر کاربری با چنین نام کاربری‌ای وجود داشت به ما اعلام می‌کند.

پس از چک می‌کنیم که آیا پسورد درست است یا نه که می‌آییم محتویات فایل کاربر که شامل رمز عبور آن است را با استفاده از fscanf خارج می‌کنیم و با رمز عبور داده شده توسط کاربر مقایسه می‌کنیم.

خب وقتی تمام این موارد چک شد و کاربر به صورت موفقیت آمیز وارد شد برای کاربر یک توکن با استفاده از تابع fun_auth می‌سازیم که تابع را در قسمت زیر مشاهده می‌کنید.

```
void fun_auth(){
    time_t t;
    srand((unsigned) time(&t));
    for(int i = 0; i < 35; i++){
        int x = (rand() % 26);
        authtoken[i] = (x + 97);
    }
}
```

که در اینجا با استفاده کتابخانه time.h یک عدد رندوم بین ۰ تا ۲۶ پیدا می‌کنیم وبا ۹۷ که کد اسکی حرف a است جمع میکنیم که به ما توکنی شامل حروف می‌دهد.

و سپس نام کاربر و توکن کاربر را درون استراکت کاربر می‌ریزیم.

ایجاد کانال و عضو شدن در کانال:

برای این کار ابتدا مانند قسمت ریجستر در پوشه channels می‌رویم و چک می‌کنیم که آیا چنین کانالی وجود دارد یا نه که اگر وجود داشت خطا برگرداند و در غیراینصورت چنین پوشه‌ای ایجاد کند و داخل آن بنویسد که کانال توسط این نام کاربری ایجاد شده است.

برای عضو شدن نیز همین روند است و اگر نام کانال وجود داشت کاربر را وارد اعضای کانال می‌کنیم.

البته این را هم باید بگویم که در دو مرحله و سایر مراحل باقی‌مانده اول کار توکن مورد نظر نیز باید بررسی بشود که درست است یا نه و به قولی شناسه کاربر باید تایید بشود که این کار توسط تابع compare_auth صورت می‌گیرد که کد آن مطابق عکس زیر است.

```
int compare_auth(char a[]){
    for(int i = 0; i < 1000; i++){
        if(strcmp(karbar[i].auth, a) == 0){
            return 1;
        }
    }
    return 0;
}
```

لاگ اوت:

در این قسمت ابتدا توکن کاربر بررسی می‌شود که اگر درست بود نام و توکن کاربر را در استراکت کاربران از بین می‌بریم تا برای لاگین کردن مجدد مشکلی نداشته باشد.

منوی گفت و گو:

در این منو گفتیم ۴ بخش وجود دارد که به دلیل کمبود وقت باید خیلی خلاصه تر از بقیه توضیح بدهم.

در هریک از این ۴ بخش باید ابتدا شرط درست بودن توکن بررسی بشود.

برای ارسال پیام ما پیام فرستاده شده توسط کاربر را درون فایل کانال که در قسمت ایجاد کانال به آن اشاره کردیم به صورت جیسون می‌ریزیم و ذخیره می‌کنیم. که در شکل زیر تکه کد مربوط به آن قسمت را آوردیم.

```
content=cJSON_GetObjectItem(rec,"content");
cJSON_AddItemToArray(content, car = cJSON_CreateObject());
cJSON_AddItemToObject(car, "sender", cJSON_CreateString(karbar[number1].user));
cJSON_AddItemToObject(car, "content", cJSON_CreateString(messages));
char output[1000];
strcpy(output,cJSON_PrintUnformatted(rec));
fp=fopen(address,"w");
fprintf(fp,"%s",output);
fclose(fp);
```

برای رفرش کردن باید به این نکته تتوجه داشت که کلاینت هر دفعه که رفرش می‌زند پیام های ندیده شده در آن کانال را باید برایش بفرستیم که این باعث می‌شود دوتا حالت بوجود بیاید و مجبور شویم داخل پوشه رفرش نیز فایلی ایجاد کنیم و تعداد پیام های دیده شده توسط کاربر را در آن بریزیم که دفعه بعدی که کاربر درخواست رفرش داد ما فقط پیام های دیده نشده را برای آن نمایش دهیم.

برای لیست اعضای کانال هم ما با استفاده از توکن دریافت شده از کاربر با استفاده از استراکت کاربران نام کانالی را که در آن عضو هستند را بدست میاوریم و با استفاده از یک حلقه تمام کاربرانی که در آن کانال هستند را پیدا می‌کنیم و آنرا به صورت جیسون نمایش می‌دهیم.

تیکه کد زیر مربوط به قسمت اعضای کانال است.

```
for(int i = 0; i < 1000; i++){
    if(strcmp(karbar[i].auth, authtoken_vorodi) == 0){
        number1 = i;
    }
}
cJSON *response;
cJSON *ch_m;
response = cJSON_CreateObject();
ch_m = cJSON_CreateArray();
cJSON_AddItemToObject(response, "type", cJSON_CreateString("Successful"));
cJSON_AddItemToObject(response, "content", ch_m);
for(int i = 0; i < 1000; i++){
    if(strcmp(karbar[number1].name_channel, karbar[i].name_channel) == 0){
        cJSON_AddItemToObject(ch_m, "content", cJSON_CreateString(karbar[i].user));
    }
}
memset(buffer, 0, sizeof(buffer));
strcpy(buffer, cJSON_PrintUnformatted(response));
printf("%s",buffer);
send(client_socket, buffer, sizeof(buffer), 0);
```

در قسمت خارج شدن از کانال نیز ما با استفاده از توکن نام کانالی که در آن کاربر عضو است را پیدا می‌کنیم و آن قسمت را پاک می‌کنیم که به این معناست این کاربر در کانال دیگری نیست.

خب این بود پروژه چت اپلیکیشن ما...

در آخر هم لازمه از تمامی ta های عزیز که برای ما وقت گذاشتن تشکر کنم و واقعا به نظرم لازمه که بگم پروژه بسیار اموزنده و زیبایی بود.

با آرزوی موفقیت برای همه...

پایان...