توضیحاتی راجع به فایل myclient.c

توابعی که در این فایل تعریف شده اند عبارتند از:

```
void socketmaker();
void Register();
void menu();
void login();
void loggedinmenu();
void createchannel();
void joinchannel();
void logout();
void chatmenu();
void sendmessage();
void refresh();
void listpeople();
void quitchannel();
void clear();
```

تمام این توابع بدون ورودی و خروجی هستند.

تابع socketmaker جهت ارتباط بین کلاینت و سرور ایجاد شده. در این تابع شمارنده ای نیز وجود دارد که مانع از چاپ شدن بیش از اندازه اعلان کانکت شدن میشود.

تابع register ورودی ها را از کاربر گرفته و به صورت register username, password به سرور میفرستد.

تابع clear صفحه ترمینال را پاک میکند.

تابع login پس از فرستادن یوزر و پسوردی که از کاربر گرفته شده اگر درخواست موفقیت آمیز باشد توکن را در رشته ای که به صورت گلوبال تعریف شده نگهداری میکند.

توابع کلاینت پیچیدگی خاصی ندارند و نیازی به توضیح نخواهند داشت.

توضیحاتی راجع به فایل server.c

توابعی که در سرور به کار برده شده اند :

```
void look();
void registerm();
void login();
void create();
void join();
void logout();
void sendm();
void refresh();
void listpeople();
void quitchannel();
int authtoken_finder();
int channel_finder();
void successfulsender();
void errorsender(char *p);
```

تابع look تابعیست که رشته ی فرستاده شده توسط کاربر را بررسی میکند تا تابع درست صدا زده شود. در این تابع هر بار سوکتی ایجاد شده و در پایان آن سوکت قطع میشود تا اتصال چند کاربر به صورت همزمان ممکن شود.

دقت شود در تابع main سوکت ایجاد و bind میشود و سرور شروع به گوش دادن میکند. در تابع look صرفا کاربر پذیرش میشود و اگر تمام تابع main در look قرار میگرفت سوکت برای استفاده مجدد bind نمیشد.

در تابع registerm ابتدا مطمئن میشویم فایل members.txt وجود داشته باشد.سپس نام کاربری دریافت شده از کاربر را در آن جست و جو میکنیم. درصورتی که موجود نبود آن را به فایل اضافه میکنیم.

در تابع login با استفاده از تابع زمان و توابع رندوم توکن را میسازیم که صرفا حروف کوچک و بزرگ انگلیسی و به طول ۳۲ کاراکتر است. سپس در آرایه member که به صورت گلوبال تعریف شده یوزر و توکن کاربر را ذخیره میکنیم که نشان دهنده آنلاین بودن آن کاربر است.

در تابع join نیز در آرایه member اسم کانال را برای همان کاربر ذخیره میکنیم.

در تابع logout کاربر را از آرایه member پاک میکنیم و تمام کاربران دیگر را یک واحد به عقب میبریم تا آرایه مرتب و بدون حفره باشد.

تابع sendm درون فایل messages.txt پیام کاربر را به علاوه نام کانال و فرستنده آن ذخیره میکند.

تابع refresh برای نشان دادن پیام های فرستاده شده است. به این صورت که در فایل messages.txt نام کانال را جست و جو کرده و هرکدام که مطابقت داشت را پرینت میکند. برای اینکه هربار پیام های قبلی پرینت نشود، کاربران آنلاین در آرایه counters مطابق با مکانشان در آرایه member عددی خواهند داشت که نشان دهنده تعداد پیام

های متفاوت خوانده شده تا آنجا است. به ازای این تعداد در حلقه while عملیات چاپ اسکیپ میشود و پس از آن نیز با تعداد پیام هایی که به تازگی خوانده شده جمع میشود.

در تابع quitchannel اسم کانال از جلوی کاربر در آرایه member پاک میشود.

توابعی کمک کننده وجود دارند که مانع از نوشتن تکه کدی در هر یک از توابع اصلی شده اند.

یکی از آنها که تنها تابعیست که ورودی دارد errorsender است. ورودی این تابع رشته ایست که میخواهیم به صورت جیسون و با تایپ ارور به کاربر فرستاده شود.

تابع successfulsender جیسونی را به کاربر میفرستد که پس از انجام عملیاتی موفقیت آمیز به کلاینت فرستاده میشد .
تابع authtoken_finder در آرایه member به دنبال توکنی که در رشته بافر ذخیره شده است میگردد. درصورتی که موجود بود اندیس j کاربر در آرایه member را برمیگرداند و در صورتی که وجود نداشت 1- برمیگرداند.
تابع channel_finder در فایل channels.txt به دنبال اسم کانال که در بافر ذخیره شده است میگردد. اگر موجود بود 0 و اگر ناموجود بود -۱ برمیگرداند.

راجع به فایل cJSON.h

این فایل توابعی از جیسون را دارد که به آن در ساخت server احتیاج داشتیم.

در این تابع استراکتی به صورت گلوبال تعریف شده که دارای یک پوینتر به کاراکتر است.

برای استفاده از این استراکت به دلیل گلوبال بودن آن باید در برخی توابع حافظه ای را برای آن الوکیت کرد.

در تابع cJSON_CreateObject استراکتی از همین جنس ساخته میشود و به عنوان خروجی بازگردانده میشود.

در تابع cJSON_PrintUnformatted خود استراکت به عنوان ورودی گفته میشود و رشته ی داخل آن بازگردانده میشود.

در تابع cJSON_CreateArray برای رشته داخل استراکت مقداری حافظه الوکیت میشود و استراکت بازگردانده میشود.

تابع cJSON_AddItemToObject که کار cJSON_AddStringToObject را نیز میکند استراکتی دیگر را در استراکت خواسته شده قرار میدهد. به صورتی که بین } و { قرار میگیرد.
تابع cJSON_AddItemToArray استراکتی را به عنوان ورودی گرفته ، در استراکت خواسته شده میریزد به صورتی که استراکت داده شده در [و] قرار میگیرد.
تابع cJSON_CreateString رشته ای را به عنوان ورودی میگیرد و استراکتی که همان رشته را در خودش دارد را به عنوان خروجی میدهد.

برای استفاده از این توابع کافی است این فایل را در هدر وارد کنیم.

توضیحی اضافه جهت دیباگ کردن

از آنجایی که ممکن است سرور و کلاینت خطا های ناجوری دهند دو کار برای راحت تر دیباگ کردن وجود دارد:

۱- در کلاینت پیش از فرستاده شدن رشته به سرور میتوان آن را پرینت کرد. کافی است کد مربوطه را از حالت کامنت در بیاورید.

۲- در سرور لوپی در سرور قرار داده شده که با خارج کردن آن از حالت کامنت میتوان زمان بسته شدن سرور را مشخص کرد. این مانع از اشغال شدن پورت توسط سوکت قبلی که ایراد پیدا کرده میشود و میتواند زمان خوبی را برای شما بخرد.