به نام خدا

محمدرضا مفیضی

۹۸۱۰۶۰۵۹

پروژه مبانی برنامه سازی

cyko messenger

فاز اول:

کلاینت:

فایل پروژه از دو بخش main **و** functions **تشکیل شده است که** main **تابع اتصل سوکت و منوي اصلي برنامه را صدا مي زند.**

توابع استفاده شده:

```
void socket_connect();
void welcome_screen();
void Mainmenu();
void create_account();
void signin();
void usermenu();
void create_channel();
void chat_menu();
void join_channel();
void logout();
void send_message();
void refresh();
void channel_members();
void leave_channel();
void search_members();
void search_messages();
```

1- socket_connect

براي اتصال به سوکت و ایجاد ارتباط بین کلاینت و سرور. پیغام خطا یا موفقیت مناسب را به کلاینت میفرستد.

2- welcome_screen

صفحه معرفي اوليه كه چند رشته را چاپ مي كند.

3- Mainmenu

چاپ کردن منوي اصلي و گرفتن گزينهي مورد نظر براي انجام ورود یا عضویت.

4- create_account

تابع عضویت است. در ابتداي همه ي توابع با دستور system("cls") **صفحه پاك ميشود.**

استراکچر عضو نام و پسورد آن را ذخیره مي‌کند. بعد پیام به سرور فرستاده مي‌شود.سپس پاسخ دریافتي به صورت جیسون را پارس مي‌کنیم.با توابع cJSON . **اگر خطا نداشت براي ورود وارد منوي اصلي مي‌شویم.**

5- signin

تابع ورود کاربر است.اطلاعات را دریافت و وارد استراکت کاربر مي‌کنیم.درخواست را میفرستیم، جواب را پارس مي‌کنیم اگر خطا نداشت وارد منوي کاربر مي‌شویم.

6- create_channel

تابع ساخت کانال است.نام کانال را مي‌گیریم، درخواست را میفرستیم ، اگر خطا نداشت وارد صفحه چت میشود.

7- join_channel

براي وارد شدن به یك كانال درخواست ارسال و اگر خطا نداشت وارد صفحه چت میشود.

8- logout

از منوي كاربر خارج مي شود.

9- send_message

پيام را دريافت و ارسال مي‌كند.

10- refresh

با گرفتن جواب بدون خطاي سرور كه آرايه اي از پيام هاست آن را با تابع هاي Array پارس مي‌كنيم.

11- channel_members

مثل refresh اين بار اسامي را مي‌گيريم.

12- search_members

درخواست را ارسال ،سپس با توجه به جواب بله لا خير نتيجه را چاپ مي‌كند.

13- search_messages

مثل refresh آرايه اي از پيام ها را مي‌گيرد.

فاز دوم:

سرور:

باز هم فایل پروژه از دو بخش main **و** functions **تشکیل شده است که** main **ایجاد سوکت و گرفتن درخواست هاي کلاینت را بر عهده دارد.**

در کل دو استراکت براي کاربر و کانال داریم:

- **استراکت کاربر:**

  نام ، پسورد، توکن و ایدي کانال فعلي را نگه مي‌دارد.
- **استراکت کانال:**

  نام کانال ، ایدي اعضاي آنلاین ، تعداد اعضا و اخرین پیام خوانده شده را نگه مي‌دارد.

با درخواست کلاینت کاربر توسط سرور accept **مي‌شود:**

```
// Accept the data packet from client and verify
int len = sizeof(client);
client_socket = accept(server_socket, (SA *)&client, &len);

if (client_socket < 0)
{
    printf("Server accceptance failed...\n");
    return 0;
}
else
    if (!flag) {
        flag = 1;
        printf("Server acccepted the client..\n");
    }
```

توابع استفاده شده:

```
void recieve();
void order_finder(char* buffer);
void create_account();
void signin();
int online(char*);
char* token_generator();
int id_finder(char*);
void create_channel();
int channel_finder(char*);
void join_channel();
void send_message();
void refresh();
void channel_members();
void leave_channel();
void logout();
void search_members();
void search_message();
```

1- receive

تابع این تابع درخواست ها را دریافت میکند و به تابع order_finder **میدهد تا نوع درخواست را پیدا کند.**

2- order_finder

همانطور که گفته شد این تابع با دستور strcmp **نوع درخواست را پید میکند، اطلاعات داده شده را بیرون میکشد و تابع موردنظر را صدا میزند.**

3- create_account

اگر کاربري با این نام (فايلي با این نام) قبلا وجود داشت خطا میدهد در غیر اینصورت با نام کاربر فايلي میسازدو اطلاعات آن را به صورت جیسون در فایل ذخیره میکند.

4- signin

با جستوجوي نام کاربر اگر کاربري با این نام (فايلي با این نام) قبلا وجود داشت و رمز عبور صحیح بود وارد میشود.

5- Online

با گرفتن نام کاربر چك میکند که آیا کاربر با این نام قبلا وارد شده است یا نه. خروجي به صورت 0 و 1 است.

6- token_generator

با استفاده از تابع ()rand **يك رشته تصادفي به عنوان توکن میسازد و خروجي میدهد.**

7- id_finder

با گرفتن توکن يك کاربر ايدي آن را خروجي میدهد.اگر کاربري وجود نداشت 1- میدهد.

8- create_channel

با جست و جوي در فایل ها اگر ین کانال قبلا وجود نداشت يك فایل با نام کانال میسازد بعد اولیان پیام ها را با توجه به دستورات جیسون در فایل پرینت میکند.

همچنین کاربر فعلي را به کاربران کانال اضافه میکند.

9- channel_finder

با گرفتن نام کانال ايدي ان آن را پیدا میکند.

10- join_channel

با باز کردن فایل کانال پیام هاي جدید را اضافه میکند سپس کاربر را به کاربران کانال اضافه میکند.

11- send_message

فایل کانال را باز میکند و پیام هاي جدید را درآن اضافه میکند.

12- Refresh

با باز کردن فایل کانال پیام هاي خوانده نشده را با توجه به متغیر آخرین پیام خوانده شده وارد جیسون میکند.

13- channel_members

با توجه به استراکت کانال ايدي اعضاي آن را بدست آورده و نام آنها را باتوجه به ايدي و استراکتشان در جیسون اضافه میکنیم.

14- leave_channel

پیام را در فایل کانال اضافه میکنیم سپس ايدي کاربر را از استراکت کانال حذف میکنیم.

15- Logout

پاسخ مناسب را براي کاربر میفرستد.

16- search_members

با جستجو در استراکت کانال و ايدي اعضاي آن نام کاربر را جست و جو میکنیم.

17- search_message

با باز کردن فایل کانال و استخراج آرایه ي پیامها پیام دلخواه را جستو جو و پیامها را به صورت جیسون میفرستد.

فاز سوم:

جيسون:

اين قسمت شامل دو فايل Cyko_JSON.h **و** Cyko_JSON.c **مي‌باشد.**

- **فايل هدر شامل استراكچر جيسون و پروتوتايپ توابع جيسون مي‌باشد.**

ابتدا انواع داده‌اي جيسون را تعريف مي‌كنيم. استراكت جيسون شامل گره هايي براي قسمت بعد و قبل و زير قسمت است.

همچنين شامل نوع جيسون و مقدار دروني به صورت رشته و عدد است. يك رشته نيز نام استراكت را نگه مي‌دارد.

توابع استفاده شده:

```
static unsigned char* DuplicateString_ckJSON(unsigned char* string);
ckJSON *NewItem_ckJSON();
ckJSON * Parse_ckJSON(const char *value);
void ParseArray_ckJSON(ckJSON *, const char *, const char *, int &, int );
void ParseString_ckJSON(ckJSON *json, const char *value, const char *str, int &start, int state);
void ParseObject_ckJSON(ckJSON *json, const char *value, const char *str, int &start, int state);
char *PrintUnformatted_ckJSON(ckJSON *json);
void PrintArray_ckJSON(ckJSON *array, char *print);
void PrintString_ckJSON(ckJSON *string, char *print);
void PrintObject_ckJSON(ckJSON *object, char *print);
void Delete_ckJSON(ckJSON *rmv);
int GetArraySize_ckJSON(const ckJSON *array);
ckJSON *GetArrayItem_ckJSON(const ckJSON *array, int index);
ckJSON *GetObjectItem_ckJSON(const ckJSON * const object, const char * const string);
ckJSON *CreateNumber_ckJSON(double num);
ckJSON *CreateString_ckJSON(const char *string);
ckJSON *CreateArray_ckJSON();
ckJSON *CreateObject_ckJSON();
void AddItemToArray_ckJSON(ckJSON *array, ckJSON *item);
void AddItemToObject_ckJSON(ckJSON * object, char * string, ckJSON * item);
ckJSON *cJSON_Duplicate(const ckJSON *item);
```

1- DuplicateString_ckJSON

اين تابع يك رشته را به عنوان ورودي دريافت مي‌كند سپس آن را دوبرابر مي‌كند (به انتهاي خود مي‌چسباند) و آدرس اولين خانه رشته را خروجي مي‌دهد.

2- NewItem_ckJSON

اين تابع يك جيسون جديد ميسازد. به اينصورت كه به اندازه سايز يك استراكت جيسون حافظه ميگيرد سپس پوينتر بدست آمده را به استراكت جيسون cast **مي‌كند.سپس پوينتر را خروجي مي‌دهد.**

3- Parse_ckJSON

تابع اصلي پارس كردن جيسون است.

با دريافت رشته جيسون به عنوان ورودي آغز ميشود.ابتدا يك شي جيسون ميسازد.(به عنوان خروجي اصلي). سپس تا زماني كه به انتهاي رشته برسيم يكي يكي زير رشته ها را دريفت ميكند و مشخص مي‌كند براي چه نوع جيسوني است.(رشته و آرايه و شيِ) و تابع مناسب را صدا مي‌زند.در نهايت پوينتر به جيسون را برمي‌گرداند.

4- ParseString_ckJSON

اين تابع به عنوان تابع كمكي پارس اصلي است كه وظيفه پارس جيسون هاي ازنوع رشته را دارد. به اينصورت كه جيسون اصلي و رشته اصلي و زير رشته(به عنوان نام جيسون رشته‌اي) و نقطه شروع در رشته اصلي و همچنين نوع اضافه شدن اين رشته به جيسون اصلي را به عنوان ورودي مي‌گيرد. سپس با گرفتن رشته بعدي (به عنوان valuestring **) با توجه به نوع اضافه شدن به جيسون اضافه مي‌شود.**

5- ParseArray_ckJSON

اين تابع جيسون هاي آرايه‌اي را پارس مي‌كند. به اينصورت كه با گرفتن جيسون اصلي و رشته اصلي و زير رشته(به عنوان نام جيسون رشته‌اي) و نقطه شروع در رشته اصلي و همچنين نوع اضافه شدن اين رشته به جيسون اصلي با پيمايش در رشته انواع جيسون را استخراج و با كمك توبع به جيسون اضافه ميكند تا به ته آرايه برسد.

6- ParseObject_ckJSON

باز هم يك تابع كمكي پارس جيسون براي شي هانحوه كار آن بسيار شبيه به تابع اصلي است.

7- PrintUnformatted_ckJSON

اين تابع عمل عكس پارس كردن را انجام مي‌دهد. يعني با گرفتن يك جيسون آن را به صورت يك رشته خروجي ميدهدبه اين صورت كه به پيمايش در جيسون در هر مرحله بجه ها فراخواني مي‌شود و با توجه به نوع جيسون تابع كمكي صدا زده مي‌شود.

8- PrintString_ckJSON

اين تابع به عنوان تابع كمكي رشته محتوا(valuestring) **را به رشته اصلي ميچسباند.**

9- PrintArray_ckJSON

اين تابع با پيمايش در اعضاي آرايه جيسوني در هر مرحله محتوا را به رشته اصلي ميچسباند.

10- PrintObject_ckJSON

اين تابع هم اعضاي شي را به رشته اصلي ميچسباند.

11- GetArraySize_ckJSON

اين تابع تعداد اعضاي يك آرايه جيسوني را بر مي‌گرداند. به اينصورت كه يكي يكي در آرايه جلو ميرود تا به به NULL **برسد.**

12- GetArrayItem_ckJSON

با گرفتن جيسون و شماره انديس عضو مشخصي از ارايه را به ما ميدهد. نحوه كاركرد به اين صورت است كه با پيمايش به عضو دلخواه ميرسد وآدرس آن را خروجي مي‌دهد.

13- GetObjectItem_ckJSON

با گرفتن جيسون و يك رشته اطلاعات موجود در بخش مورد نظر با نام رشته داده شده زا برميگرداند. نحوه كار جستجو در اعضاي يك شي و مقايسه رشته هاست.

14- CreateNumber_ckJSON

اين تابع يك جيسون از نوع عددي ميسازد. به اين صورت كه با تخصيص حافظه پويا و تغيير نوع درون استراكت ساخته شده كار انجام ميشود.

15- CreateString_ckJSON

با گرفتن نام جيسون رشته اي مثل تابع قبل يك جيسون از نوع رشته ميسازد.

16- CreateArray_ckJSON

يك جيسون با نوع آرايه ميسازد.

17- CreateObject_ckJSON

يك جيسون با نوع شي ميسازد.

18- AddItemToArray_ckJSON

با گرفتن يك جيسون آن را به انتهاي اعضاي يك آرايه جيسون اضافه مي كند.

19- AddItemToObject_ckJSON

با گرفتن يك جيسون آن را به انتهاي اعضاي يك شي جيسون اضافه مي كند.

20- cJSON_Duplicate

يك جيسون را دوتا ميكند.از طريق كپي تك تك اعضاي آن.

21- Delete_ckJSON

جيسون داده شده را پاك مي‌كند. درواقع حافظه داده شده به آن را آزاد مي‌كند.