|  |  |  |
| --- | --- | --- |
|  | به نام خدا  شبکه های کامپیوتری | C:\Users\mohammad\AppData\Local\Microsoft\Windows\INetCache\Content.Word\ut_fanni_logo.png |
| تمرین برنامه نویسی شماره 01  (FTP-Server) | | |

**اعضای گروه :**

**محمد سعادتی 810198410**

**محمد عراقی 810198436**

**مقدمه**

هدف از این پروژه آشنایی با مفاهیم socket programming و ارتباط از طریق socket ها بوده

است . کلاینت ها با دانستن پورت سرور به سرور وصل شده و ارتباط برقرار می کنند و بسته به نوع command ها ، اطلاعات و پیام های متناسب میان سرور و هر کلاینت رد و بدل می شود.

از آنجایی که قرار اسـت بر روی دو پورت اطلاعات را منتقل كنيم ، دو ســـوكـت در نظر ميگيريم. يكي را روي پورت command channel ، bind می کنیم و دیگر پورت را روی data channel ، bind می کنیم.

**نحوه اجرا**

در دایرکتوری های Client و Server دستور make را برای ساخت پروژه توسط g++ اجرا کنید.

در دایرکتوری Server اجرا کنید:./server.out config.json

در دایرکتوری Client اجرا کنید./client.out config.json

لاگ های مربوط به اطلاعات اجرا در یک فایل با نام log.txtدر پوشه Serverنوشته می شوند.

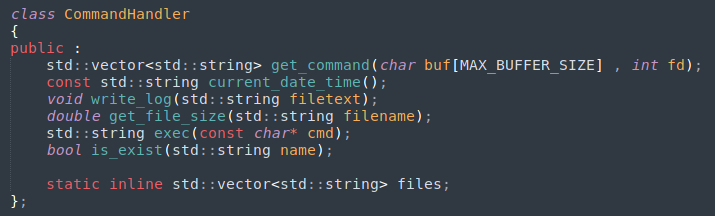
**کلاس ها و متد ها**

**1) CommandHandler**

وظیفه ی این کلاس تشخیص دستور ارسال شده برای سرور و پاسخ مناسب به آن است و بخش بیشتر پروژه به عهده ی آن است .

این کلاس دارای وکتوری ازنام فایل ها (فایل های مجاز کاربران ) و همچنین تابع get\_command است.

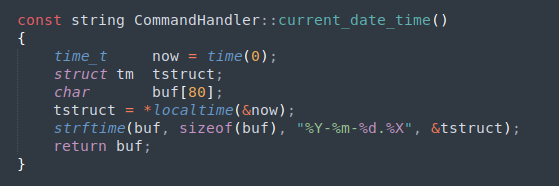
تابع get\_command با هربار دریافت داده توسط سرور صدا زده میشود و پیام ارسالی را تحلیل میکند و پاسخ مناسب را آماده ی ارسال میکند . ورودی این تابع ، بافر سرور (پیام ارسال شده توسط کلاینت) و file descriptor کلاینت فرستنده ی پیام است.



**دیگر توابع :**

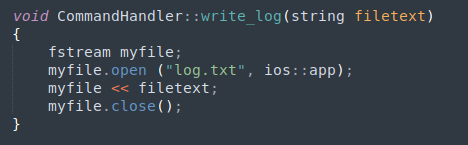
cuttent\_date\_time:

برای به دست آوردن زمان کنونی بر حسب سال و ماه و روز و ...



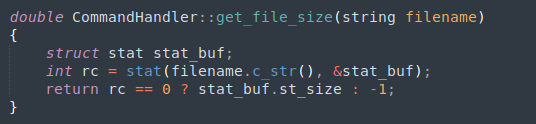
write\_log:

برای نوشتن لاگ در فایل مربوطه برای نگهداری اطلاعات ورود و خروج و ...



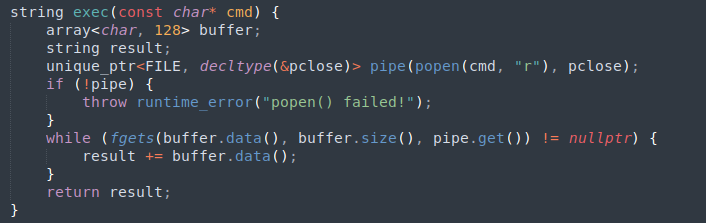
get\_file\_size:

برای پیدا کردن سایز فایل با اسم داده شده در مسیر کنونی برای استفاده در دانلود های یوزر



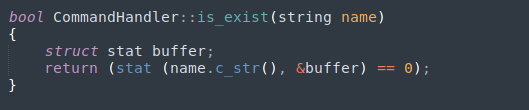
exec:

برای اجرای کامند ها در سطح کرنل به وسیله ی پایپ و بازگرداندن نتیجه ی به دست آمده به عنوان result



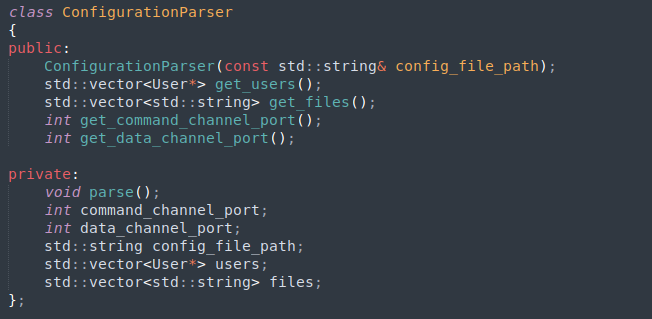
is\_exist:

چک کردن اینکه آیا فایل یا دایرکتوری وجود دارد یا خیر.



**2) ConfigurationParser**

اطلاعات مربوط به فایل config در این کلاس نگهداری شده و در صورت نیاز با دسترسی به آن میتوان به اطلاعاتی اعم از اطلاعات یوزرها ، فایل های مجاز ، و همچنین پورت های چنل دیتا و کامند دسترسی یافت .



Attributes:

command\_channel\_port: پورت چنل کامند (مورد نیاز برای کلاینت ها)

data\_channel\_port: پورت چنل دیتا (مورد نیاز برای کلاینت ها)

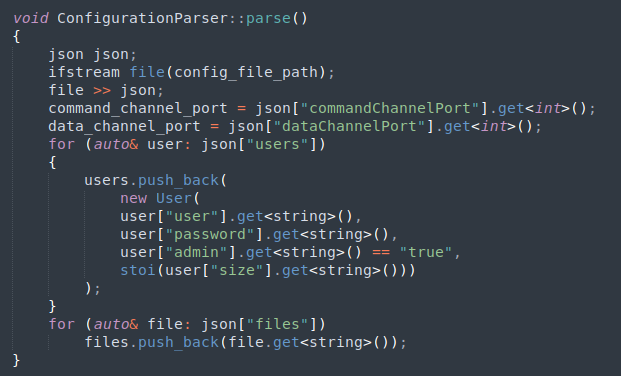
config\_file\_path: مسیر فایل config.json

users: وکتوری از یوزرهایمان که اطلاعات خوانده شده ی یوزر ها در آن نگه داری میشود

files: وکتوری از نام فایل های مجاز یوزرهایمان

Methods:

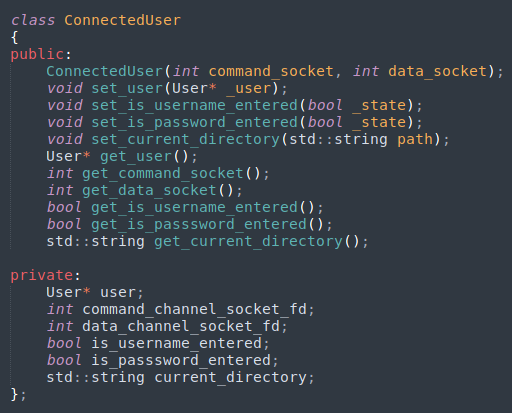
Parse: برای خواندن اطلاعات فایل config.json صدا زده میشود و به شکل زیر عمل میکند :



که همانگونه که مشخص است اطلاعات را خوانده و به ازای هر یوزر یک یوزر ساخته و به وکتور یوزرها اضافه میکند .

**3) ConnectedUser**

اطلاعات مربوط به یوزر هایی که به سرور متصل شده اند را نگه میدارد ، اعم از وضعیت login ، directory کنونی آن و ...



Attributes:

User: یوزر مربوطه در وکتور users در ConfigurationParser

fd: file descriptor یوزر وصل شده به سرور

command\_channel\_socket\_fd و data\_channel\_socket\_fd:

fd های کانال دیتا و کامند

is\_user\_name\_entered و is\_password\_entered:

برای بررسی وضعیت لاگین و اینکه در کدام مرحله ی لاگین است . (username و password وارد شده اند یا خیر).

current\_directory: directory کنونی کلاینت مان

Methods:

decrease\_available\_size: برای کاهش حجم کنونی یوزر پس از دانلود فایل

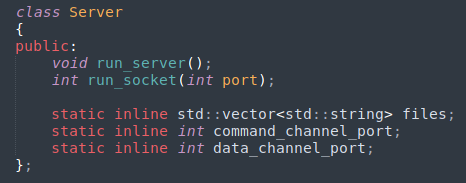
is\_able\_to\_download: برای بررسی ملزومات کافی برای دانلود فایل

is\_admin\_user: برای بررسی admin بودن یا یوزر عادی بودن

**4) Server**

برای ایجاد و بایند کردن سرورمان.

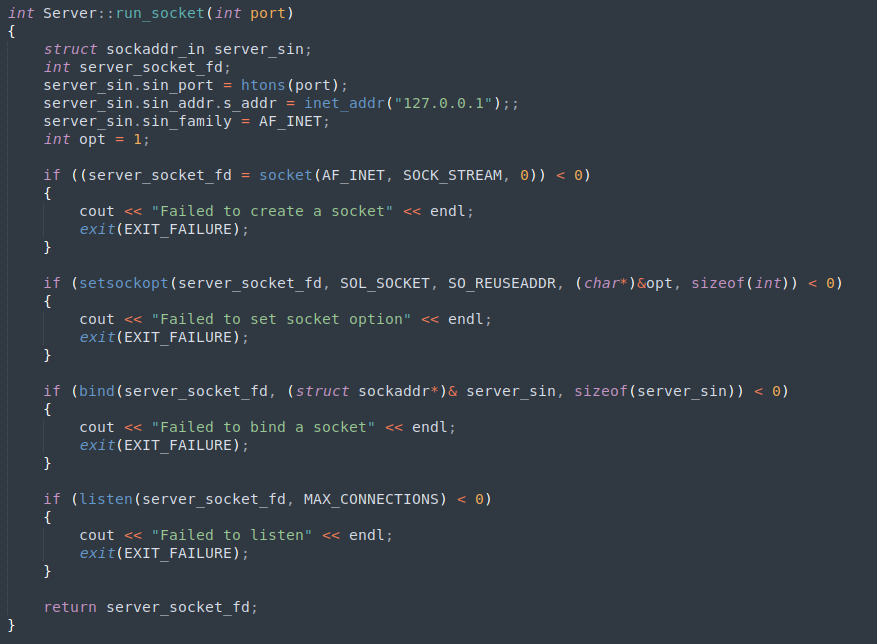
به دلیل بلاکینگ بودن سیستم کال ها و برای اینکه سرور بتواند همزمان به چند کلاینت پاسخ دهد ، از select استفاده شده است .



Methods:

run\_server: برای برقراری سرور

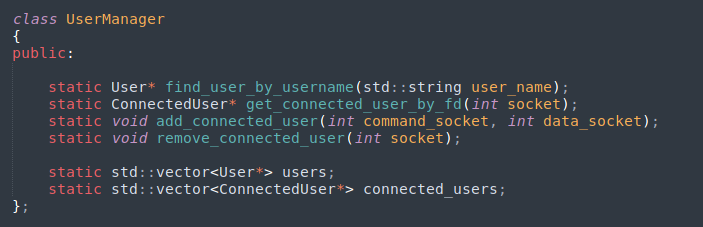
run\_socket: برای بایند کردن سرور به سوکت مناسب و بقیه کارهای اجرایی که به شکل زیر است:



و طبق گفته های کلاس پیاده سازی شده است .

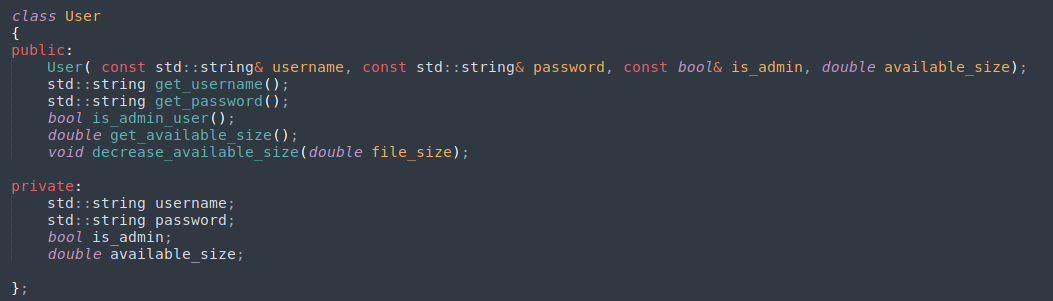
**5) UserManager**

تمام کارهای مربوط به مدیریت یوزرها ، اعم از اطلاعات اولیه یوزرها و وضعیت کنونی آن ها ، پیدا کردن یوزر ها بر اساس پارامتر های مناسب ، حذف و اضافه یوزرها و ... به عهده این کلاس است و به users و connected\_users دسترسی دارد .



**6) User**

همانند connected\_user ، با این تفاوت که اطلاعات اولیه یوزرها را که از فایل کانفیگ خوانده شده شامل است و به وضعیت کنونی یوزرها ارتباطی ندارد.

****

Attributes:

is\_admin : برای بررسی یوزر عادی یا ادمین بودن

available\_size: سایز دانلود باقی مانده برای یوزر

fd: file descriptor یوزر متصل شده

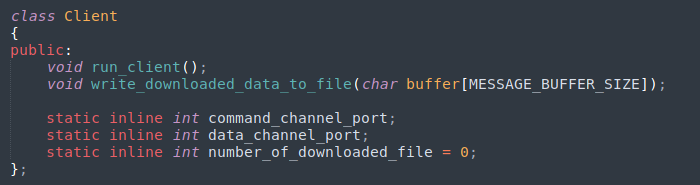
Methods:

Is\_matched\_with : برای بررسی تطابق یوزرنیم و پسورد

Decrease\_available\_size: برای کاهش سایز باقی مانده یوزر پس از دانلود

**7) Client**

کلاس کلاینت مان که اجرا شده و با داشتن پورت های لازم (پورت کانال کامند و دیتا) به سرور متصل شده و شروع به رد و بدل اطلاعات میکند .



run\_client: برای اجرای کلاینت و ساختن کلاینت جدید به شکل زیر:

