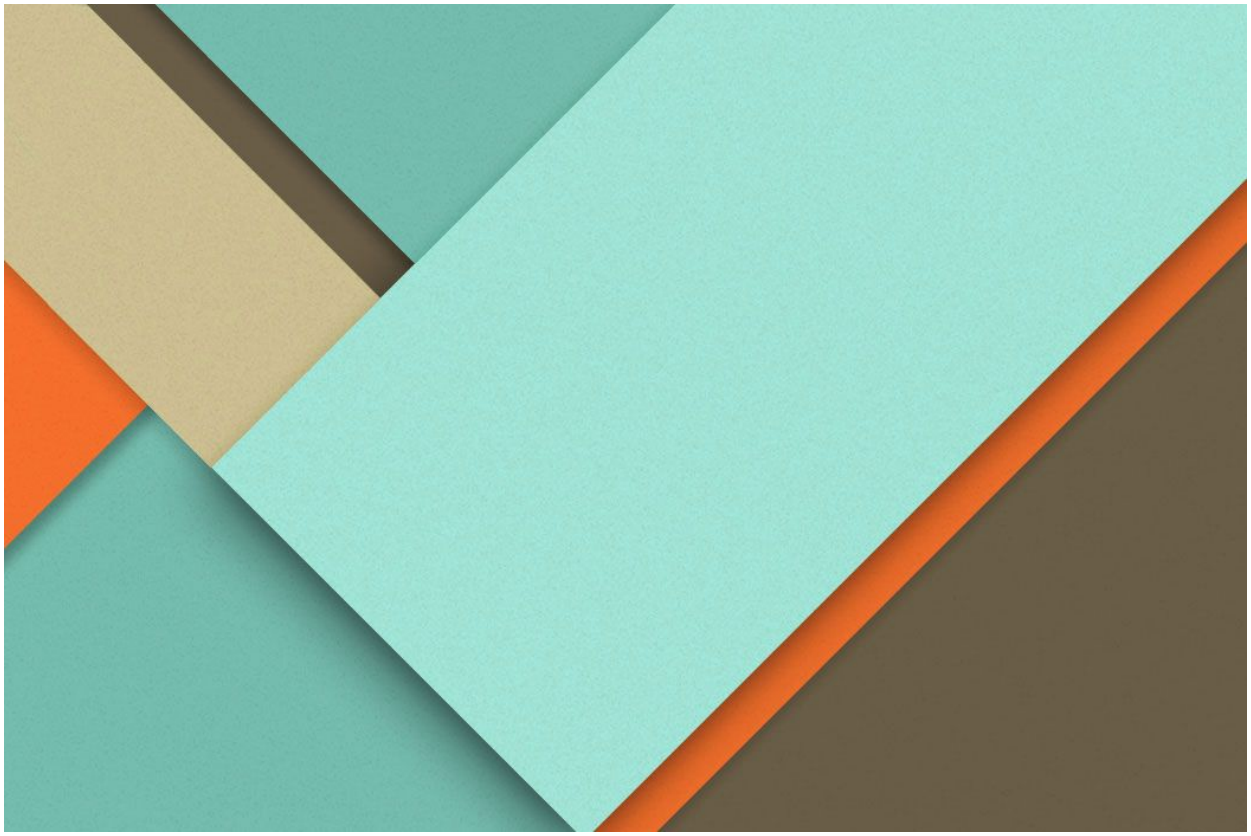

پروژه مبانی برنامه سازی

نیمسال اول 98-99

دانشکده مهندسی کامپیوتر دانشگاه صنعتی شریف

فاز 1 : چت اپلیکیشن (کلاینت)

فهرست

1. مقدمه:

چت اپلیکیشن ها برای تسهیل در برقراری ارتباط به وجود آمده اند.Internet Relay Chat یا به اختصار IRC از جمله اصطلاحات پرکاربرد دنیایی است که پا در آن گذاشته اید. این ارتباط بر بستر اینترنت ارائه می شود. در این‌جا می‌توانید نمونه ای از یک IRC را ببینید. هدف پروژه‌ی مبانی پیاده‌سازی چنین اپلیکیشنی به طور ابتدایی است تا گامی هدفمند و پایانی خوش برای این درس رقم بخورد.

چت‌اپلیکیشن ها در ساختاری ارائه می شوند که به آن **معماری کلاینت/سرور** گفته می شود. برای توضیح این معماری به مثال جالب زیر دقت کنید:

یک مشتری را در نظر بگیرید که به رستورانی رفته است. این مشتری یک سری درخواست دارد که پاسخ به این درخواست‌ها از طرف آشپزخانه‌ی رستوران به فرد داده می‌شود. لذا مشتری این درخواست‌ها را به آشپزخانه می‌رساند و پاسخ را نیز دریافت می‌کند. در این صورت مشتری را که درخواست می دهد، **کلاینت** و آشپزخانه را که پاسخگوی نیاز مشتری‌ست، **سرور** می نامیم.

حال به مثال جالب تر زیر توجه فرمایید:

در گوگل، وقتی شما به عنوان **کلاینت** چیزی را در کادر مربوطه می نویسید و دکمه search را می‌فشارید، در واقع یک درخواست برای **سرور** شرکت گوگل ارسال می‌کنید و گوگل نیز بعد از پیدا کردن پاسخ شما، نتایج را برایتان ارسال می کند. در واقع یک ارتباط بین شما (کلاینت) و گوگل (سرور) برقرار می شود.

توجهتان را به آخرین مثال نیز جلب می کنیم:

در اپلیکیشنی مانند تلگرام، زمانی که شما(کلاینت) در یک گروه هستید و پیغامی را می‌نویسید و دکمه‌ی ارسال را می زنید،شما یک درخواست به سرور تلگرام می‌فرستید که این پیام را در این گروه قرار بده و تلگرام هم پس از بررسی درخواست، پیام را در گروه مورد نظر قرار می دهد.

مثال های بالا مواردی برای درک مفاهیم کلاینت و سرور بود. برای درک بهتر به لینک های زیر مراجعه کنید.اگر باز هم کم و کاستی در درک این موضوعات داشتید، **سرچ** کنید!!!

لینک 1

لینک 2

حال، پیاده‌سازی کلاینت چت‌اپلیکشین در این فاز بر عهده‌ی شماست! در فازهای بعدی قسمت‌های دیگر چت‌اپلیکیشن را پیاده‌سازی خواهید کرد...

2. کتابخانه ها

برای قسمت شبکه، بسته به سیستم عاملی که استفاده می‌کنید، باید از یکی از کتابخانه‌های زیر استفاده کنید.

اگر ویندوز دارید، با اضافه کردن< #include <winsock2.h به ابتدای کد، می‌توانید از توابع مربوط به شبکه استفاده کنید. توضیحاتی در مورد توابع کتابخانه‌ی winsock2 را می‌توانید در این‌جا پیدا کنید.

اگر از mac یا linux استفاده می کنید، می توانید #<include <sys/socket.h > را به ابتدای کد اضافه کرده و از توابع آن استفاده کنید. لیست توابع این کتابخانه نیز از طریق این لینک قابل دسترسی است.

برای استفاده از JSON، می توانید از کتابخانه‌ی cJSON که از طریق این و این لینک قابل دریافت هستند، استفاده کنید. بعد از دانلود کتابخانه از یکی از دو لینک داده شده، باید دو فایل cJSON.c و cJSON.h را از فایل zip در پوشه‌ی پروژه‌ی خود قرار دهید. فراموش نکنید که باید cJSON.h را بالای کد include کنید (#include <cJSON.h>)

توضیحاتی در مورد توابع کتابخانه‌ی JSON در فایل README لینک اول داده شده است. برای اطلاعات بیشتر، از google استفاده کنید!

3. منوی برنامه

اولین چیزی که باید در مورد پروژه بدانید این است که : این پروژه ، پروژه شماست! :)

یعنی می توانید این پروژه را برای خودتان شخصی سازی (customize) کنید، اما باید فرم ارتباط سرور و کلاینت و فیچرها را حفظ کنید. به طور مثال، پیامی که به سرور می‌فرستید، باید به شکلی که در ادامه توضیح داده شده‌است باشد.

و همچنین ما یک نمونه‌ی پیشنهادی برای شما داریم.

در ابتدا به ساختار کلی می‌پردازیم سپس در نمونه‌ی پیشنهادی دستورات و کارهایی را که باید برنامه شما انجام دهد گفته می‌شود.

1. منوی کاربری:

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

1.1. ساخت حساب جدید و ورود به حساب کاربری:

هنگامی که کاربر می‌خواهد با استفاده از برنامه شما یک حساب کاربری جدید در سرور ایجاد کند از دستور register استفاده می‌کند.

به عنوان مثال، در ساختار پیشنهادی ما ایجاد حساب کاربری جدید و ورود مطابق مراحل زیر صورت می‌گیرد.

ابتدا گزینه ی مربوطه انتخاب شده:

```
1
```

سپس مراحل وارد کردن username و password طی می شوند:

```
Enter Username
Alireza
Enter Password
1234
```

اگر کاربری با این نام از قبل وجود داشته باشد، سرور یک پیغام خطا برمی گرداند در غیر اینصورت یک پیام اجرای موفق بر می گرداند. سپس اگر registeration با موفقیت انجام شود، حال کاربر باید با account ساخته شده login کند. اگر registeration با خطا روبه‌رو می گردد، حساب کاربری جدید ایجاد نمی‌شود.

برای وارد شدن به حساب کاربری، باید همانند فرایند register، این بار به قسمت login رفته و با وارد کردن username و password حساب کاربری، وارد آن شوید.

اگر login با موفقیت انجام شود، سرور یک auth token می سازد (درباره auth token پایین تر توضیح داده شده) و به کلاینت بر می گرداند و سپس کلاینت وارد منوی اصلی شود. اگر حساب

کاربری وجود نداشت یا رمز اشتباه بود خطای مربوطه بر گردانده می‌شود و کلاینت باید در account menu بماند.

2. منوی اصلی:

در منوی اصلی، سه گزینه‌ای که در تصویر زیر می‌بینید روی میز است.ساخت کانال جدید، اضافه شدن به یک کانال و خروج.

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

2.1. ساخت کانال جدید:

با دستور create channel یک کانال جدید ایجاد می شود. هر کاربر تنها می تواند در هر لحظه در یک کانال باشد. با اجرای دستور ساخت کانال و وارد کردن نام کانال و ارسال ریکوئست مربوطه به سرور (در مورد ریکوئست ها در قسمت بعد توضیح داده شده) توسط کلاینت، یک کانال تازه توسط سرور ساخته و کاربر ایجاد کننده‌ی آن به طور خودکار در آن عضو می‌شود. در صورت موفقیت آمیز بودن ساخت کانال توسط سرور، یک پیام اجرای موفق برگردانده می‌شود و کلاینت باید وارد منوی گفت و گو شود. در صورت موفق نبودن درخواست خطا داده شده و کلاینت باید در منوی اصلی بماند.

2.2. اضافه شدن به کانال:

با دانستن نام یکی از کانالهای موجود در سرور، کاربر می تواند عضو یک کانال شود و پس از ارسال ریکوئست مربوطه، سرور، کاربر را عضو این کانال می‌کند و کاربر وارد صفحه‌ی گفت و گو می‌شود.

در صورت موفق بودن درخواست یک پیام اجرای موفق از سرور برگردانده شده و کلاینت وارد منوی گفت و گو می شود. در صورت موفق نبودن کلاینت در منوی اصلی می ماند.

2.3. خروج از حساب کاربری:

در صورت موفق بودن درخواست یک پیام اجرای موفق توسط سرور برگردانده شده و کلاینت باید وارد منوی کاربری شود.

3. منوی گفت و گو:

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

3.1. ارسال پیام:

پس از انتخاب این گزینه شما می توانید پیامتان را ارسال کنید. روند پیشنهادی به شرح زیر است:

ابتدا گزینه ی ارسال پیام را انتخاب می کنیم:

```
1
```

سپس پیغام مورد نظر را می نویسیم:

```
Hello!This is my first message
```

سپس دوباره منوی چت ظاهر می گردد:

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

3.2. تازه سازی:

با انتخاب این دستور و ارسال ریکوئست آن به سرور، سرور لیستی از پیام‌هایی که از زمان آخرین refresh شما تا الان توسط دیگر کاربران ارسال شده است را به عنوان پاسخ برمی‌گرداند.

3.3. لیست افراد:

با انتخاب این دستور و ارسال ریکوئست آن به سرور، سرور لیستی از افراد گروه را برایتان ارسال می کند.

این که چگونه با سرور ارتباط داشته باشید و از پیام های ارسالی سرور استفاده کنید، در ادامه توضیح داده شده است.

3.4. خارج شدن از کانال:

با انتخاب این گزینه و ارسال ریکوئست به سرور، شما از کانال خارج می‌شوید و وارد منوی اصلی خواهید شد.

4.ارسال پیام به سرور

هر درخواستی که شما از کاربر می‌گیرید، باید آن را به یک فرمت خاص تبدیل کنید. این فرمت خاص، یک ریکوئست قابل فهم برای سرور است که می‌توانید آن را برای سرور ارسال کنید.در واقع این ریکوئست ها، استاندارد هایی است که بین کلاینت و سرور قرارداد شده است. این ریکوئست ها در واقع یک رشته (string) هستند و شما این رشته ها را به سرور می‌فرستید. ریکوئست‌ها به فرمت‌های زیر تعریف شدند: (در خط اول هر بخش فرمت ریکوئست و در ادامه توضیح آن آمده است.)

4.1.ثبت کاربر جدید:

"register <username>, <password>"

این فرمت پیامی‌ست که باید برای ثبت کاربر جدید به سرور ارسال کنید. یعنی شما پس از دریافت username و password از کاربر، این رشته را به سرور ارسال می‌کنید.

مثال:

register Alireza, 1234

دقت کنید: بعد از register یک فاصله، بین Alireza و "," هیچ فاصله و بعد از "," یک فاصله وجود دارد.

4.2.ورود کاربر به حساب خود:

"login <username>, <password>"

اگر این ارسال خطایی نداشته باشد، سرور برای شما یک پیام ارسال می کند که حاوی یک توکن است (در ادامه توضیح آن آمده است). این توکن برای اطمینان سرور است که شما همانی هستید که باید باشید. پس لازم است که در تمام درخواست‌های بعدی، توکن را نیز باید به عنوان بخشی از ریکوئست طبق فرمت گفته شده برای سرور ارسال کنید.

مثال:

ارسال ریکوئست login به سرور:

```
login mohammadhosein, 12345mh
```

دقت کنید: بعد از login یک فاصله، بین mohammadhosein و "," هیچ فاصله و بعد از "," یک فاصله وجود دارد.

پاسخ سرور:

```
{"type":"AuthToken","content":"HsVMzQNa-w_fO5RRxbr7hz8qqN_2fpdB"}
```

عبارت `HsVMzQNa-w_fO5RRxbr7hz8qqN_2fpdB` توکن شماست...

4.3.ساخت کانال:

"create channel <channel name>, <AuthToken>"

دقت کنید: که باید همراه ریکوئست توکن را ارسال کنید.

مثال:

create channel fop98, HsVMzQNa-w_fO5RRxbr7hz8qqN_2fpdB

دقت کنید: بعد از create channel یک فاصله، بین fop98 و "," هیچ فاصله و بعد از "," یک فاصله وجود دارد.

4.4.اضافه شدن به کانال:

"join channel <channel name>, <AuthToken>"

مثال:

join channel fop98, HsVMzQNa-w_fO5RRxbr7hz8qqN_2fpdB

دقت کنید: بعد از join channel یک فاصله، بین fop98 و "," هیچ فاصله و بعد از "," یک فاصله وجود دارد.

4.5.خروج کاربر:

"logout <AuthToken>"

مثال:

logout HsVMzQNa-w_fO5RRxbr7hz8qqN_2fpdB

4.6.ارسال پیام در کانال فعلی:

"send <message>, <AuthToken>"

مثال:

send Hello!This is my first Message, HsVMzQNa-w_fO5RRxbr7hz8qqN_2fpdB

دقت کنید: بعد از send یک فاصله، بین Hello ! ... و "," هیچ فاصله و بعد از "," یک فاصله وجود دارد.

4.7.تازه سازی:

"refresh <AuthToken>"

مثال:

refresh HsVMzQNa-w_fO5RRxbr7hz8qqN_2fpdB

4.8.لیست افراد کانال فعلی:

"channel members HsVMzQNa-w_fO5RRxbr7hz8qqN_2fpdB"

مثال:

channel members HsVMzQNa-w_fO5RRxbr7hz8qqN_2fpdB

4.9.خروج از کانال:

"leave <AuthToken>"

مثال:

leave HsVMzQNa-w_fO5RRxbr7hz8qqN_2fpdB

دقت کنید: همانطور که گفته شد، ریکوئست‌ها باید به همان فرمت‌های بالا باشند.

5. جیسون (JSON)

جیسون معادل اختصاری عبارت JavaScript Object Notation به معنی «نمادگذاری اشیا در جاوا اسکریپت» است.

جیسون از نوع رشته است و چیز عجیبی نیست. یک رشته با یک فرمت خاص. لذا شما این فرمت خاص را دریافت و اطلاعات مورد نیاز خود را از آن استخراج می‌نمایید.

6. تازه سازی (Refresh)

این ویژگی شما را قادر می سازد که پیام هایی را که دیگران ارسال می کنند، ببینید.اگر در طول خواندن داک تا به اینجا دقت کرده باشید، متوجه می شوید که شما در این برنامه، باید برای هر کاری به سرور ریکوئست بزنید و اطلاعات را دریافت کنید.در نتیجه برای اینکه پیام هایی را که در کانال موجود است، ببینید،باید یک ریکوئست به سرور بزنید و این پیام ها را دریافت کنید.تازه سازی یا refresh همین ریکوئست مورد نظر است.دقت کنید که این پیام های دریافتی از سرور را باید به فرمتی دلخواه در کنسول چاپ کنید.یک نمونه در زیر آمده است:

پیام کلاینت به سرور:

```
refresh HsVMzQNa-w_fO5RRxbr7hz8qqN_2fpdB
```

پیام سرور به کلاینت:

```
{"type":"List","content":[{"sender":"mohammadhosein","content":"salam"},{"sender":"mohammadhosein","c
ontent":"be kanale fop98 khosh amadid"}]}
```

نگران عکس بالا نباشید.عکس زیر بیانگر یک پیام به همراه فرستنده ی آن می باشد که در عکس بالا آمده است:

```
{"sender":"mohammadhosein","content":"salam"}
```

7. توکن (Auth Token)

توکن چیست؟

توکن یک عبارت یا یک رمز برای انحصار ارتباط بین کلاینت و سرور است.

در واقع برای اینکه سرور بفهمد شما همانی هستید که باید باشید، از شما توکن می خواهد.

از آنجایی که قرار است شما پیوسته به سرور ریکوئست دهید، لذا لازم است که سرور شما را به ازای هر ریکوئست تایید کند و بتواند برایتان پاسخ را ارسال کند. پس این ارتباط نیازمند این است که چیزی بین شما و سرور به صورت منحصر به فرد موجود باشد. چنین چیزی را توکن یا authentication token می نامیم.توکن پس از لاگین برای شما ارسال می شود و باید آن را جایی ذخیره کنید و به سرور ارسال کنید.پیاده سازی شما باید به گونه ای باشد که پس از هر بار login توکن را عوض کند؛ زیرا توکن شما در سرور عوض شده و سرور توکن جدیدی را برای شما ارسال می کند.در مثال های بالا توکن عبارت زیر بود:

```
HsVMzQNa-w_fO5RRxbr7hz8qqN_2fpdB
```

همانطور که پیش‌تر نیز به آن اشاره شد،در فاز اول فقط پیاده‌سازی کلاینت بر عهده‌ی شماست. حال برای آزمایش کلاینت خود، از اتصال به سرور تا ارسال درخواست به آن و دریافت پاسخ از آن، سروری متناسب با فرمت ریکوئست‌های این پروژه در اختیارتان قرار می‌دهیم. برای راه‌اندازی و استفاده از آن، این بخش را دنبال کنید.

8. راه‌اندازی سرور

سرور این چت‌اپلیکیشن، متشکل از دو فایل server.jar و configuration.json می‌باشد که برای راه‌اندازی سرور هر دوی آن‌ها ضروری‌اند. اکانت‌ها و کانال‌هایی که می‌سازید، در پوشه‌ای به

نام Resources ذخیره می‌شوند. این پوشه در ابتدا وجود ندارد. اما بعد از ساخت اولین کاربر، توسط سرور به وجود می‌آید.

برای اجرای سرور روی لپتاپ خود، نیاز به ابزارهای توسعه‌ی java دارید.

ابتدا از این‌جا و از جدول Java SE Development Kit 11.0.5 ، نسخه‌ی مخصوص به سیستم عامل خود را دریافت و سپس آن را نصب کنید. می‌توانید آموزش ویدیویی نصب java برای سیستم عامل‌های ویندوز و مک را به ترتیب از این‌جا و این‌جا ببینید. همچنین کاربران لینوکس می‌تواند با اجرای دستور sudo apt-get install openjdk-11-jre در ترمینال، بدون رفتن به سایت و دانلود فایل، ابزار توسعه‌ی java را روی لپتاپ خود داشته‌باشند. (اگر از توزیع‌هایی از لینوکس نظیر monjro استفاده می‌کنید که apt ندارند، باید از packaging tool مخصوص به آن توزیع استفاده کنید)

بعد از نصب java، کاربران لینوکس و مک در terminal و کاربران ویندوز در command prompt خود می‌توانند با اجرای دستور java -version، از موفقیت آمیز بودن مراحل نصب اطمینان حاصل کنند.

```
mohammadhosein@mohammadhosein-ZenBook-UX333FN-UX333FN:~$ java -version
openjdk version "11.0.5-ea" 2019-10-15
OpenJDK Runtime Environment (build 11.0.5-ea+10-post-Ubuntu-0ubuntu1)
OpenJDK 64-Bit Server VM (build 11.0.5-ea+10-post-Ubuntu-0ubuntu1, mixed mode, sharing)
mohammadhosein@mohammadhosein-ZenBook-UX333FN-UX333FN:~$
```

با نصب شدن java روی سیستم، حال می‌توانید از سرور استفاده کنید.

برای فعال‌سازی سرور، اول به پوشه‌ای که فایل سرور را در آن قرار داده‌اید بروید. کاربران ویندوز می‌توانند با کلیک بر روی فایل server.jar، سرور را بالا بیاورند.

در همه‌ی سیستم عامل ها، از جمله ویندوز، می‌توان با باز کردن terminal یا command prompt در آدرسی که سرور در آن قرار دارد، با اجرای دستور java -jar server.jar، سرور را اجرا کرد.

```
mohammadhosein@mohammadhosein-ZenBook-UX333FN-UX333FN:~/Codes/Java/ChatApplication_Server$ java -jar server.jar
    Info | 2019/12/12 18:42:52 | Initializing...
    Info | 2019/12/12 18:42:52 | Users initialized from file successfully
    Info | 2019/12/12 18:42:52 | Channels initialized from file successfully
    Info | 2019/12/12 18:42:52 | Initialized.
    Info | 2019/12/12 18:42:52 | Listening on port 12345
```

برای باز کردن terminal در آدرس مورد نظر برای سیستم عامل‌های مک و لینوکس، با file explorer به پوشه‌ی سرور بروید و با راست کلیک روی پوشه و انتخاب گزینه‌ی open in terminal، ترمینال را در آن پوشه باز کنید. برای باز کردن command prompt در سیستم عامل ویندوز، ابتدا به پوشه‌ای که سرور در آن قرار دارد رفته، و سپس در نوار بالای پنجره که آدرس پوشه‌ی فعلی نوشته شده

است، عبارت cmd را تایپ و enter بزنید. همچنین با cd کردن از آدرس root در terminal و command prompt نیز می‌توانید به پوشه‌ی موردنظر دسترسی پیدا کنید.

توجه کنید برای اجرای سرور، حتما فایل configuration.json کنار فایل server.jar قرار داشته باشد. محتویات این فایل را به هیچ وجه تغییر ندهید.

توجه کنید بعد از ساختن یک account و channel در چت اپلیکیشن، پوشه‌ای به نام Resources در کنار server.jar به وجود می‌آید.این پوشه دیتابیس سرور است. آن را پاک نکنید!

9. نحوه‌ی اتصال و ارسال درخواست به سرور

سرورها از نظر نحوه‌ی اتصال و نحوه‌ی ارتباط با کلاینت‌ها به دو دسته‌ی stateless و stateful تقسیم می‌شوند. سروری که در اختیار شما قرار داده شده‌است، از نوع اول می‌باشد. در این نوع از سرورها، فرایند ارسال درخواست به کلاینت به این صورت است:

ابتدا کلاینت اتصال سوکت خود با سرور را برقرار می‌کند، سپس درخواست خود را برای سرور ارسال می‌کند، بعد اتصال قطع می‌شود. یعنی کلاینت برای ارسال درخواست دیگر، باید دوباره ارتباط را برقرار کند.

توجه کنید اگر کلاینتی به سرور متصل شود و درخواستی برای آن ارسال نکند، بعد از چند ثانیه سرور خود ارتباط را قطع می‌کند.

```
Error | 2019/12/14 12:19:03 | java.net.SocketTimeoutException: Read timed out
    at java.base/java.net.SocketInputStream.socketRead0(Native Method)
    at java.base/java.net.SocketInputStream.socketRead(SocketInputStream.java:115)
    at java.base/java.net.SocketInputStream.read(SocketInputStream.java:168)
    at java.base/java.net.SocketInputStream.read(SocketInputStream.java:140)
    at java.base/sun.nio.cs.StreamDecoder.readBytes(StreamDecoder.java:284)
    at java.base/sun.nio.cs.StreamDecoder.implRead(StreamDecoder.java:326)
    at java.base/sun.nio.cs.StreamDecoder.read(StreamDecoder.java:178)
    at java.base/java.io.InputStreamReader.read(InputStreamReader.java:185)
    at java.base/java.io.BufferedReader.fill(BufferedReader.java:161)
    at java.base/java.io.BufferedReader.readLine(BufferedReader.java:326)
    at java.base/java.io.BufferedReader.readLine(BufferedReader.java:392)
    at server.ServerHandler.Run(ServerHandler.java:45)
    at Main.main(Main.java:12)
```