پروژه ی شاره ۲ پراکنش پای ما مکی

#### ۱. مقدمه

یک ساختار ساده از شبکه ی بانکی را در نظر بگیرید. در این ساختار تعدادی پایانه (Terminal) بانکی با اتصال به سرور مرکزی درخواستهای واریز و یا برداشت را برای سرور ارسال می کنند. این درخواستها پس از اعتبارسنجی توسط سرور اجرا شده و نتیجه ی انجام آنها برای پایانه ی مورد نظر ارسال می گردد.

گستردگی پایانهها و همچنین حجم بالای تراکنشها باعث افزایش احتمال ارسال همزمان چند درخواست برای سرور می شود که باید به نحو صحیحی مدیریت شود. به این صورت که از بروز خطا در اثر اجرای همزمان چند درخواست بر روی یک سپرده جلوگیری شود.

همچنین جهت پیگیری خطا و رفع مغایرتهای احتمالی، لازم است که تمام وقایعی که در سرور و پایانه ها اتفاق میافتد در فایلهای ثبت وقایع (Log File) نوشته شود.

#### ۲. تعریف یروژه:

نحوه ی اجرای برنامه به این شکل است که ابتدا شماره پورت سرور در فایل تنظیمات آن به صورت دستی مشخص می شود. سپس سرور اجرا شده و تنظیمات و اطلاعات سپرده ها را از فایل خوانده و منتظر می ماند تا پایانه ها از طریق شبکه به آن متصل شوند.

در سمت پایانه، ابتدا آدرس و شماره پورت سرور در فایل تنظیمات به صورت دستی مشخص شده و سپس ترمینال اجرا می شود. ترمینال بر اساس تنظیمات به سرور متصل شده و درخواستها را ارسال می کند.

سرور با دریافت هر درخواست، پس از اعتبارسنجی (صحت مقادیر داده شده، کفایت مقدار موجودی، عدم تخطی از سقف موجودی و ...)، آن را اجرا کرده و نتیجه را برای ترمینال ارسال می کند. نیازی به ثبت اطلاعات سپرده ها در پایگاه داده نیست و کافی است در حافظه نگهداری شوند. انتخاب و طراحی ساختار داده ی مورد استفاده برای نگهداری و تبادل اطلاعات به عهده ی برنامه نویس است.

پایانه با دریافت پاسخ درخواست خود، نتیجه را در فایلی با عنوان response.xml (با فرمت مناسب) ثبت می کند.

لازم است سرور و همهی پایانهها تمام وقایع مربوط به دریافت و ارسال پیام و عملیات داخلی خود را به میزان لازم و کافی در فایلهای وقایعنگاری ثبت کنند.

در سمت سرور در هر زمان که از طریق console، پیغام "sync" به سرور داده شد، کلیه اطلاعات سپردهها که بر روی حافظه موجود است در همان فایل core.json با همان فرمت اولیه به روز رسانی میشوند.

# ۳. فرمت ورودیها

به ازای هر پایانه فایلی با عنوان terminal.xml وجود دارد که اطلاعات مربوط به پایانه و همچنین تراکنشهای مالی مرتبط با آن پایانه را مشخص می کند. ساختار این فایل به شکل زیر است:

برای سرور نیز فایلی به نام core.json وجود دارد که حاوی اطلاعات زیر است:

### ۴. نكات قابل توجه

- √ برای parse کردن XML از parser های موجود در JDK استفاده کنید. اما برای parse کردن JSON لازم است از کتابخانههای متن باز استفاده کنید.
  - ✓ برای مدیریت کتابخانههای مورد نیاز همچنین مدیریت نحوهی تولید محصول نهایی از ابزار maven استفاده کنید.
- ✓ بدیهی است که برای انجام درست این پروژه نیاز به critical section handling دارید. شما باید پس از اتمام پروژه ی خود، یک سناریوی تست آماده کنید که نشان دهید بدون handling شما خروجی برنامه اشتباه خواهد بود.
  - ✓ برای انجام این تمرین مدت ده روز مهلت دارید.

## ۵. کلیدواژههای مرتبط

- Thread ✓
- Critical Section ✓
- Synchronization ✓
  - Socket ✓
    - JSON ✓
    - XML ✓
  - Maven ✓
  - Java.util ✓
    - Git ✓