



Baloot¹

مقدمه

هدف این تمرین کسب آشنایی بیشتر با ابزارهای Maven، Git و Unit Testing است که در تمامی تمرین‌های بعدی به آن‌ها نیاز خواهید داشت. همچنین شما در این تمرین بخشی از منطق دامنه پروژه اصلی را نیز پیاده‌سازی می‌کنید. توجه داشته باشید که در این تمرین نیازی به پیاده‌سازی مفاهیم وب ندارید و در تمرین‌های بعدی به این مفاهیم پرداخته خواهد شد.

کلیت پروژه

سیستمی که در طول درس به توسعه‌ی آن خواهید پرداخت، یک سیستم خرید لوازم مورد نیاز است. در این سیستم کاربر می‌تواند لیستی از کالاهای ارائه‌شده در دسته‌های مختلف را مشاهده کنند، به فیلتر کردن کالاها بر اساس ویژگی‌های آن‌ها پردازند، لیست خرید ایجاد کنند، لیست علاقه‌مندی از کالاها را مد نظر خود را ایجاد کنند و به کالاها امتیاز بدهند.

گام‌های پیاده‌سازی:

ایجاد پروژه Maven²

ابتدا یک پروژه‌ی Maven بسازید و با ساختار ایجاد شده توسط آن آشنا شوید. همچنین، فایل pom.xml و اطلاعات داخل آن را مشاهده کنید. پیشنهاد ما برای انجام پروژه‌های این درس، استفاده از ابزار IntelliJ IDEA می‌باشد. برای ایجاد پروژه Maven در محیط توسعه IntelliJ IDEA، می‌توانید از این [لینک](#) استفاده کنید.

¹ جنگل‌های بلوط زاگرس

² برای اطلاعات بیشتر در مورد Maven می‌توانید به لینک‌های زیر مراجعه کنید.

اضافه کردن وابستگی³ های مورد نیاز

داده های مورد نیاز برای انجام این تمرین، در قالب JSON به شما داده می شود. برای خواندن و تجزیه JSON و تبدیل آن به فرمت مورد نیاز از پکیج های مخصوص کار با JSON در جاوا استفاده کنید. در اینترنت جستجو کرده و بهترین پکیج موجود را به وابستگی های پروژه خود اضافه کنید.

پیاده سازی منطق برنامه

در این تمرین نیاز است تا تعدادی از عملیات های پایه ای برنامه خود را پیاده سازی کنید. این عملیات در قالب دستوراتی در خط فرمان⁴، به برنامه شما داده خواهند شد. برای سهولت پیاده سازی نیازی نیست که داده ای را در جایی ذخیره کنید و تمامی موجودیت های برنامه خود را در حافظه اصلی نگه داشته و متناسب با آن ها به دستورات پاسخ دهید. هر دستور، شکل کلی زیر را دارد:

```
command <JSONData>
```

command نشان دهنده نام دستور و JSONData داده ی مربوط به آن دستور به شکل `serialize`⁵ شده در قالب JSON است. برای استفاده از این داده، باید با استفاده از کتابخانه ای که در قسمت قبل به پروژه اضافه کردید، آن را `deserialize` کنید. همچنین، قابل ذکر است که JSONData برای همه دستورات وارد نمی شود.

ورودی ها به صورت JSON به شما داده می شوند که در هر بخش نمونه آن ها آمده است. پاسخ ها نیز باید به صورت JSON در خط فرمان چاپ شوند. فرمت پاسخ ها به دو حالت زیر می باشد:

```
{"success": true, "data": <ResponseData>}
```

```
{"success": false, "data": <ErrorMessage>}
```

³ Dependency

⁴ CLI (Command Line Interface)

⁵ [serialization - What is deserialize and serialize in JSON? - Stack Overflow](#)

دستوراتی که باید در این تمرین پیاده سازی کنید، در ادامه آمده اند.

۰۱. اضافه کردن کاربر

اطلاعات کاربر شامل نام کاربری (username)، رمز عبور (password)، ایمیل (email)، تاریخ تولد (birthDate)، آدرس (address) و اعتبار (credit) است. برای مثال با اجرای دستور زیر، user1 به مجموعه کاربران سامانه اضافه می شود. توجه کنید اگر کاربری با id مشابه در سامانه موجود بود، تنها اطلاعات آن کاربر به روز می شود. دقت کنید که نام کاربری نمیتواند شامل کاراکترهایی نظیر فاصله، نیم فاصله یا کاراکترهای دیگر نظیر ! @ # و ... باشد و نام کاربری برای هر کاربریگایک باشد.

در این تمرین، نیازی به بررسی درستی ساختار ایمیل نیست.

نمونه دستور:

```
addUser {"username": "user1", "password": "1234", "email": "user@gmail.com", "birthDate":  
"1977-09-15", "address": "address1", "credit": 1500}
```

۰۲. اضافه کردن تهیه کننده

این دستور یک تهیه کننده را به مجموعه تهیه کنندگان اضافه می کند. اطلاعات تهیه کننده شامل شناسه (id)، نام تهیه کننده (name)، ...

```
addProvider {"id": 1, "name": "provider1", "registryDate": "2023-09-15"}
```

همچنین به ازای کالاهای جدید، لیست کالاهای موجود برای هر تهیه کننده به روزرسانی شده و برای هر تهیه کننده، میانگین نمرات کالاهای آن تهیه کننده قابل مشاهده است.

۰۳. اضافه کردن کالا

با این دستور، یک کالا به لیست کالاها اضافه خواهد شد. اطلاعات کالا شامل شناسه کالا (id)، نام کالا (name)، شناسه تهیه‌کننده (providerId)، قیمت کالا (price)، دسته‌بندی‌های کالا (categories)، امتیاز آن کالا (rating) و تعداد موجودی آن در انبار است. توجه داشته باشید که شناسه هر کالا، کلید یگای شناسایی آن کالا است. همچنین در صورتی که تهیه‌کننده با شناسه داده شده وجود نداشته باید خروجی خطا برگردانده شود. پس از هر خرید، یک عدد از تعداد کالای موجود در انبار کم می‌شود.
نمونه دستور:

```
addCommodity {"id": 1, "name": "Headphone", "providerId": 3, "price": 35000, "categories":  
"[Technology, Phone]", "rating": 8.8, "inStock": 50}
```

۰۴. دریافت لیست کالاهای موجود

با این دستور، اطلاعات تمامی کالاهای موجود را دریافت می‌کنیم.

نمونه دستور:

```
getCommoditiesList
```

نمونه اطلاعات خروجی:

```
"data": {"commoditiesList": [{"id": 6, "name": "Headphone", "providerId": 3, "price": 35000,  
"categories": "[Technology, Phone]", "rating": 8.3, "inStock": 30}, {"id": 7, "name": "broccoli",  
"providerId": 5, "price": 2000, "categories": "[Vegetables]", "rating": 8.8, "inStock": 3}]}
```

۵. امتیازدهی به کالا

با این دستور، کاربر می‌تواند به یک کالا امتیاز بدهد. امتیاز داده شده باید یک عدد صحیح از 1 تا 10 باشد و در غیر این صورت، خطای مناسب نمایش داده می‌شود. پس از آن باید با محاسبه مجدد میانگین امتیازات فیلم، این مقدار برای کالا به‌روزرسانی شود.

توجه داشته باشید که اگر کاربر قبلاً به کالای مورد نظر امتیاز داده بود، امتیاز او باید به‌روز شود (اضافه نشود!) و براساس آن، امتیاز کالا نیز تغییر کند. اگر کالا یا کاربر موجود نبود، باید خطای مناسب نمایش داده شود.

نمونه دستور:

```
rateCommodity {"username": "user1", "commodityId": 3, "score": 7}
```

۷. اضافه کردن کالا به لیست خرید (buyList)

پس از وارد کردن این دستور، کالای مورد نظر به لیست خرید کاربر اضافه می‌شود. البته ابتدا باید بررسی کنید که این کالا در انبار موجود باشد و در غیر این صورت پیغام مناسب نمایش دهید و کالا به لیست اضافه نشود. همچنین اگر چنین کالایی وجود ندارد، صرفاً یک خطا مبنی بر عدم وجود کالا به کاربر نمایش داده شود. اگر این کالا از قبل در لیست مشاهده کاربر وجود دارد نیز خطای مناسب نمایش داده شود. همچنین اگر کاربر موجود نبود، خطای مناسب نمایش داده شود.

نمونه دستور:

```
addToBuyList {"username": "user1", "commodityId": 4}
```

۸. حذف کالا از لیست خرید (buyList)

با وارد کردن این دستور، کالا با شناسه وارد شده از لیست خرید کاربر حذف می‌شود. اگر کالایی با این مشخصات در لیست خرید وجود نداشت، صرفاً یک خطا مبنی بر عدم وجود کالا به کاربر نمایش داده می‌شود. همچنین اگر کاربر وجود نداشت، خطای مناسب نمایش داده می‌شود.

نمونه دستور:

```
removeFromBuyList {"username": "user1", "commodityId": 7}
```

۹. جستجوی یک کالا بر اساس شناسه

```
getCommodityById {"id": 2}
```

در خروجی این دستور، اطلاعات کالای مورد نظر نمایش داده می‌شود. این اطلاعات شامل شناسه کالا، نام کالا، نام تهیه کننده، قیمت کالا، دسته‌بندی‌های کالا و امتیاز آن کالا می‌باشد. در صورتی که کالا با شناسه مورد نظر یافت نشد، خطای مناسب نمایش داده می‌شود.

نمونه اطلاعات خروجی:

```
"data": {"id": 6, "name": "Headphone", "provider": "headphoneProvider", "price": 3000, "categories":  
"[Technology, Phone]", "rating" : 8.3}
```

۱۰. مشاهده لیست کالاها بر اساس دسته‌بندی

```
getCommoditiesByCategory {"category": "Vegetables"}
```

در خروجی این دستور، لیستی از کالاهایی که در دسته‌بندی مورد نظر می‌باشند، در قالب JSON نمایش داده می‌شود. توجه کنید که اطلاعات چاپ شده شامل شناسه کالا، نام کالا، شناسه تهیه‌کننده، قیمت کالا، دسته‌بندی‌ها و امتیاز کالا می‌باشد. اگر کالایی با این دسته‌بندی یافت نشد، یک لیست خالی برگردانده می‌شود و خطایی داده نمی‌شود.

نمونه اطلاعات خروجی:

```
"data": {"commoditiesListByCategory": [{"id": 6, "name": "Onion", "providerId": 3, "price": 3000,  
"categories": "[Vegetables]", "rating" : 8.3}, {"id": 7, "name": "broccoli", "providerId": 5, "price": 2000,  
"categories": "[Vegetables]", "rating" : 8.8}]}
```

```
getBuyList {"username": "user1"}
```

در خروجی این دستور، لیست کالاهای موجود در لیست خرید کاربر، در قالب JSON نمایش داده می‌شود. توجه کنید که اطلاعات چاپ شده شامل شناسه کالا نام کالا، نام تهیه‌کننده، قیمت کالا، دسته‌بندی کالا و امتیاز کالا می‌باشد. اگر کالایی در لیست خرید کاربر موجود نبود، صرفاً یک لیست خالی نمایش داده می‌شود.

نمونه اطلاعات خروجی:

```
"data": {"buyList": [{"id": 6, "name": "Onion", "providerId": 3, "price": 3000, "categories": "[Vegetables]", "rating": 8.3}, {"id": 7, "name": "Broccoli", "providerId": 5, "price": 2000, "categories": "[Vegetables]", "rating": 8.8}]}
```

آزمون واحد^۶

در این قسمت باید با استفاده از چارچوب JUnit برای سناریوهای مختلف امتیازدهی به کالا، مشاهده لیست کالاها بر اساس دسته‌بندی، اضافه کردن کالا به لیست خرید و جستجوی یک کالا بر اساس شناسه آزمون واحد بنویسید. آزمون‌های شما باید ساختار مناسب Test، Setup و Teardown را رعایت کنند.

افزودن پروژه به گیت

ابتدا در سایت Github عضو شوید و یک مخزن خصوصی ایجاد کنید. سپس کاربر ieSpring1402 را با دسترسی **Maintainer** به پروژه خود اضافه کنید. تمامی تغییرات خود را به گیت اضافه کنید و در نهایت، در مخزن خود بارگذاری کنید. توجه کنید که پروژه شما پس از clone شدن باید به راحتی قابل اجرا باشد. برای تمرین نحوه کار با گیت توصیه می‌شود که [این لینک](#) و برای آشنایی با شیوهی مناسب کامیت توصیه می‌شود که [این لینک](#) را مطالعه کنید.

^۶ Unit Testing

نکات پایانی

- این تمرین در گروه‌های حداکثر دو نفری انجام می‌شود و کافی است که یکی از اعضای گروه Hash مربوط به آخرین کامیت پروژه را در سایت درس آپلود کند. در هنگام تحویل، پروژه روی این کامیت مورد ارزیابی قرار می‌گیرد.
- ساختار صحیح و تمیزی کد برنامه، بخشی از نمره‌ی این فاز پروژه‌ی شما خواهد بود. بنابراین در طراحی ساختار برنامه دقت به خرج دهید.
- هدف این تمرین یادگیری شماست. لطفاً تمرین را خودتان انجام دهید. در صورت مشاهده‌ی مشابهت بین کدهای دو گروه، از نمره هر دو گروه مطابق سیاستی که در کلاس گفته شده است کسر خواهد شد.
- سوالات خود را تا حد ممکن در فروم درس مطرح کنید تا سایر دانشجویان نیز از پاسخ آن‌ها بهره‌مند شوند. در صورتی که قصد مطرح کردن سوال خاص‌تری داشتید، از طریق ایمیل با طراحان این تمرین ارتباط برقرار کنید.
- ایمیل طراحان پروژه:

hrkh78@gmail.com

aalizad79@gmail.com