

راهنمای استفاده از سمپل پروژه مایکروسرویس

استک توسعه :

Language: Java 13

Framework: Spring Boot 2.3.0

Authentication: Spring Security + OAuth2

Data: Spring Data + JPA + Hibernate-ORM + Oracle Database

Caching: Spring Data + Redis

Indexing: Spring Data + Hibernate Search

Microservice: gRpc + ProtoBuffer3 + Eureka Server

Front-end: ReactJs 16.8

لطفا مراحل زیر را انجام بدهید:

1. جاوا 13 jdk نصب کنید
2. در قسمت system Environment ویندوز متغیر JAVA_HOME را روی مسیر نصب جاوا اضافه کنید و در Path آن را با پوشه bin اضافه کنید. برای مثال:
JAVA_HOME = C:\Program Files\Java\jdk-13.0.2
Add to path : %JAVA_HOME%\bin
3. اوراکل اکسپرس را نصب کنید(در این راهنما برای یوزر sys و system در مراحل نصب رمز 123456 ست میکنیم) از مسیر زیر میتوانید آن را دانلود کنید:
<https://drive.google.com/file/d/1sdMKUH9eXfYfYRRYMcYG2-rDc2Nav5Yz/view>
4. در قسمت system Environment ویندوز متغیر ORACLE_HOME را روی مسیر نصب اوراکل و متغیر ORACLE_SID را روی کلمه XE اضافه کنید. برای مثال:
ORACLE_HOME = C:\app\MyUser\product\18.0.0\dbhomeXE
ORACLE_SID = XE
5. فایل sqlplus را از مسیر زیر اجرا کنید:
C:\app\MyUser\product\18.0.0\dbhomeXE\bin\sqlplus.exe
6. با یوزر system و رمز 123456 که در نصب وارد کردید وارد شوید و با دستورات زیر دو یوزر/اسکیما برای دو دیتابیس دو مایکروسرویس با رمز 123456 ایجاد کنید(در فایل properties پروفایل dev این رمز ست شده است):
 - تنظیم session روی دیتابیس embedded اوراکل xe :
ALTER SESSION SET CONTAINER = XEPDB1;

- ایجاد table space :

```
CREATE BIGFILE TABLESPACE tbsmot_perm_01
DATAFILE 'tbsmot_perm_01.dat'
SIZE 20M
AUTOEXTEND ON;
```

```
CREATE TEMPORARY TABLESPACE tbsmot_temp_01
TEMPFILE 'tbsmot_temp_01.dbf'
SIZE 20M
AUTOEXTEND ON;
```

- ایجاد یوزر/اسکیما دیتابیس mslogin :

```
CREATE USER mslogin
IDENTIFIED BY 123456
DEFAULT TABLESPACE tbsmot_perm_01
TEMPORARY TABLESPACE tbsmot_temp_01
QUOTA 20M on tbsmot_perm_01;
```

- تنظیم دسترسی ها برای یوزر/اسکیما دیتابیس mslogin :

```
GRANT create session TO mslogin;
GRANT create table TO mslogin;
GRANT create view TO mslogin;
GRANT create any trigger TO mslogin;
GRANT create any procedure TO mslogin;
GRANT create sequence TO mslogin;
GRANT create synonym TO mslogin;
GRANT connect TO mslogin;
alter user mslogin default role all;
```

- ایجاد یوزر/اسکیما دیتابیس msgeo:

```
CREATE USER msgeo
IDENTIFIED BY 123456
DEFAULT TABLESPACE tbsmot_perm_01
TEMPORARY TABLESPACE tbsmot_temp_01
QUOTA 20M on tbsmot_perm_01;
```

- تنظیم دسترسی ها برای یوزر/اسکیما دیتابیس msgeo:

```
GRANT create session TO msgeo;
GRANT create table TO msgeo;
GRANT create view TO msgeo;
GRANT create any trigger TO msgeo;
GRANT create any procedure TO msgeo;
GRANT create sequence TO msgeo;
GRANT create synonym TO msgeo;
GRANT connect TO msgeo;
alter user msgeo default role all;
```

7. تنظیمات عمومی IntelliJ Idea :

1.InteljiIDEA: Help -> Edit Custom Vm Options -> add these two line:

-Dfile.encoding=UTF-8

-Dconsole.encoding=UTF-8

2.InteljiIDEA: File -> Settings -> Editor -> File Encodings-> Project Encoding: form "System default" to UTF-8. May be it affected somehow.

3.InteljiIDEA: File -> Settings -> Editor -> General -> Code Completion -> check "show the documentation popup in 500 ms"

4.InteljiIDEA: File -> Settings -> Editor -> General -> Auto Import -> check "Optimize imports on the fly (for current project)"

5.IntelijlIDEA: File -> Settings -> Editor -> Color Scheme -> Color Scheme Font -> Scheme: Default -> uncheck "Show only monospaced fonts" and set font to "Tahoma"

8. اگر پوشه ای با مسیر `C:\Users\MyUser\.m2\repository\com\motaharinia\MsUtility` (که در این مسیر `MyUser` نام اکانت ویندوز شما هست) در ویندوز شما وجود دارد `MUtility` را حذف نمایید. پروژه `msutility` را در `IntelliJ Idea` باز کنید (در تمام پروژه ها اگر در اولین باز شدن پروژه در `IntelliJ idea` پاپ آپ `enable auto import` و `Windows defender automatic fix` باز شد آن را تایید کنید و در `project structure` چک کنید که `jdk` پروژه ها روی 13 باشد) و طبق راهنمایی که در انتهای فایل `pom` نوشته شده آن را `install` کنید که در پوشه `m2` پکیج آن برای استفاده به عنوان وابستگی در تمام پروژه های مایکروسرویس دیگر ساخته بشود. این کار فقط برای یک بار و یا در زمانی نیاز است که `msutility` تغییرات داشته باشد. پروژه `msutility` به عنوان وابستگی در `pom` تمامی پروژه های مایکروسرویس اضافه میگردد تا تکرار کد بویلر در مایکروسرویس ها نداشته باشیم.

9. پروژه `msdiscovery` را در `IntelliJ idea` باز کنید و سپس پروژه را از بالا سمت راست `IntelliJ idea` به صورت معمولی اجرا کنید. این پروژه قابلیت `Eureka Server` برای مانیتور مایکروسرویس ها و ارتباط راحت تر ویی آنها را فراهم مینماید. بعد از اجرای این پروژه مایکروسرویس های دیگری که اجرا شوند خود را در این سرور رجیستر میکنند و میتوان وضعیت سلامت آنها را توسط آدرس <http://localhost:8761> مشاهده نمود.

10. پروژه `msgeo` را در `IntelliJ idea` باز کنید(در تمام پروژه ها اگر در اولین باز شدن پروژه در `IntelliJ idea` پاپ آپ `enable auto import` و `Windows defender automatic fix` باز شد آن را تایید کنید و در `project structure` چک کنید که `jdk` پروژه ها روی 13 باشد) و توسط متد `create` در کلاس تست `CityControllerTest` این پروژه یک شهر جدید در دیتابیس `msgeo` اضافه کنید. حالا یک اننتیتی شهر در دیتابیس `msgeo` با آی دی 1 جهت تست مایکروسرویس دارید.

11. در پروژه `msgeo` تنظیم زیر را انجام دهید تا اجرای پروژه به صورت عادی با پروفایل `dev` انجام شود و سپس پروژه را از بالا سمت راست `IntelliJ idea` به صورت معمولی اجرا کنید. (تمام پروژه های سمپل با قابلیت اکتیو پروفایل اسپرینگ بوت آماده شده اند به این صورت که برای اجرای عادی پروژه ها باید یک `profile` معرفی گردد و پروژه تنظیمات داخل `application.properties` را میخواند و سپس تنظیمات `application-profile.properties` را میخواند و پروژه را تنظیم میکند. ما برای نمونه دو پروفایل `dev` و `com` را در `resources` به صورت فایل های `application-dev.properties` و `application-com.properties` آماده کرده ایم و از پروفایل `dev` که در آن آی پی دیتابیس لوکال دارد استفاده میکنیم. برای اجرای کلاسهای تست نیز از انوتیشن پروفایل بالای کلاسهای تست استفاده شده است)

IntelijlIDEA: Run -> Edit Configuration -> Spring Boot -> XXXApplication -> Environment -> VM Options: -Dspring.profiles.active=dev

12. بگذارید پروژه msgeo در حال اجرا بماند. در این حالت پروژه msgeo با پورت gRpc شماره 9091 به عنوان سرور در حال اجرا است. و پروژه mslogin می خواهد در هنگام ثبت یک AdminUser از آی دی 1 شهر بعنوان شهر اطلاعات تماس AdminUser استفاده نماید. در application.properties پروژه msgeo با تنظیمات زیر پورت مایکروسرویس این پروژه به عنوان یک سرور را مشخص کرده ایم و تا کلاشتهایی که می خواهند به آن وصل شوند از آن پورت استفاده کنند و برای فراخوانی مایکروسرویس authorizationStub.checkAccess از مایکروسرویس mslogin چه پروتکل ارتباطی و آدرس و پورتی را استفاده نماید:

```
grpc.server.port=9091
grpc.client.grpcClientAuthorization.negotiationType=PLAINTEXT
grpc.client.grpcClientAuthorization.address=static://localhost:9092
```

13. پروژه mslogin را در IntelliJ idea باز کنید(در تمام پروژه ها اگر در اولین باز شدن پروژه در IntelliJ idea پاپ آپ project structure و enable auto import و Windows defender automatic fix باز شد آن را تایید کنید و در jdk که پروژه ها روی 13 باشد) و توسط متد create در کلاس تست AdminUserControllerTest این پروژه یک AdminUser جدید در دیتابیس mslogin اضافه کنید. در متد AdminUserServiceImpl.create دقت کنید که از مایکروسرویس cityStub.grpcReadOne استفاده شده است و آی دی شهر دریافتی از کلاینت با مایکروسرویس msgeo چک شده که فیک نباشد و سپس در انتیتی ذخیره شده است. در application.properties پروژه mslogin با تنظیمات زیر پورت مایکروسرویس این پروژه به عنوان یک سرور را مشخص کرده ایم و تا کلاشتهایی که می خواهند به آن وصل شوند از آن پورت استفاده کنند و برای فراخوانی مایکروسرویس cityStub.grpcReadOne از مایکروسرویس msgeo چه پروتکل ارتباطی و آدرس و پورتی را استفاده نماید:

```
grpc.server.port=9092
grpc.client.grpcClientCity.negotiationType=PLAINTEXT
grpc.client.grpcClientCity.address=static://localhost:9091
```

14. دقت کنید که یک فایل ms standard.xlsx به عنوان قوانین دستخط ارسال می گردد که کمک میکند همکاران توسعه پیاده سازی یکسان در تولید سامانه داشته باشند.

15. تغییراتی که باید در کلاینت React برای استفاده از مایکروسرویس انجام شود را در پوشه ms react change همراه با نمونه کد برای دریافت توکن لاگین قرار داده ایم. این تغییرات به صورتی است که بجای متد UtilAjax.sendRequest از متد جدید UtilAjax.msRequest استفاده میشود و در کد صفحات همکاران توسعه دهنده دیگر آدرسها را به صورت مستقیم به این متد نمیدهند و ابتدا هر آدرس را در Microservice.js در enum مربوطه وارد میکنند و آن enum را که شامل href و method و جواب و فرمت بازگشتی است را به متد UtilAjax.msRequest ورودی میدهند. با این تغییر میتوان مایکروسرویس ها را در کنار پروژه monolithic فعلی اجرا نمود و کلاینت میتواند به صورت همزمان با monolithic و مایکروسرویس ها به صورت هم زمان ارتباط برقرار کند.

16. قابلیت actuator به مایکروسرویسهای mslogin و msgeo اضافه شده و بعد از اجرای این دو پروژه با رفرش صفحه مانیتورینگ Eureka و با کلیک روی نام مایکروسرویس مورد نظر میتونید اطلاعات actuator اون مایکروسرویس رو مشاهده کنید

17. برای تست دریافت اکسپشن در فراخوانی متد مایکروسرویس دیگر میتوانید بعد از اجرای msgeo آدرس <http://localhost:8081/authorizationClient> را اجرا نمایید. در این آدرس متدی از mslogin فراخوانی شده است و در صورت خالی بودن url در متد AuthorizationStub.checkAccess یک اکسپشن بیزینس پرتاب شده و در AuthorizationClientService.checkAccess در msgeo این اکسپشن دریافت و به CustomException تبدیل میشود. برای فعال کردن قابلیت دریافت اکسپشنها بر بروی انوتیشن @GrpcServer ورودی Interceptors را با مقدار GrpcExceptionHandler.class به صورت زیر پر کنید:

```
@GrpcService(interceptors = { GrpcExceptionHandler.class })
```

18. ردیس یکی از رایجترین دیتابیس های Nosql است که اطلاعات در آن بصورت Key و Value، بر روی معماری داخلی سیستم ذخیره سازی میشود. برای دیدن اطلاعات کش شده میتوان از نرم افزار redisDesktopManager استفاده کرد که اطلاعات کش شده را بصورت جیسون نمایش میدهد.

19. برای ساخت کش از انوتیشن @Cacheable استفاده میشود. برای تست ایجاد کش میتوانید بعد از اجرای msLogin آدرس <http://localhost:8082/v1/adminUser/id> را اجرا نمایید. در این آدرس متدی از mslogin فراخوانی شده است. برای فعال کردن قابلیت کش از انوتیشن زیر استفاده میکنیم.

```
@Cacheable(value = "AdminUser", key = "#id")
```

20. برای آپدیت کش ایجاد شده از انوتیشن @CachePut استفاده میشود. برای تست ویرایش کش میتوانید بعد از اجرای msLogin آدرس <http://localhost:8082/v1/adminUser/id> را اجرا نمایید. در این آدرس متدی از mslogin فراخوانی شده است. برای فعال کردن قابلیت ویرایش کش از انوتیشن زیر استفاده میکنیم.

```
@CachePut(value = "AdminUser", key = "#adminUserModel.getId()")
```

21. برای پاک کردن کش ایجاد شده از انوتیشن @cacheEvict استفاده میشود. برای تست حذف کش میتوانید بعد از اجرای msLogin آدرس <http://localhost:8082/v1/adminUser/id> را اجرا نمایید. در این آدرس متدی از mslogin فراخوانی شده است. برای فعال کردن قابلیت حذف از انوتیشن زیر استفاده میکنیم.

```
@CacheEvict(value = "AdminUser", key = "#id")
```

22. برای تنظیم اولیه Hibernate Search شی FullTextEntityManager در کلاس IndexingConfiguration در پکیج com.motaharinia.base.config تعریف شده است.

23. برای پیاده سازی Hibernate search انتیتی AdminUser در mslogin ایندکس شده است. برای جستجوی ساده متد hchFindByName و برای جستجوی relational متد hchFindByGender در AdminUserController نوشته شده است. برای تست دو متد تست hchFindTest و hchFindByGenderTest در AdminUserControllerTest را اجرا کنید.

24. برای ایجاد rebuildIndex ، متد buildIndex در BuildIndexController نوشته شده است. برای تست، متدهای BuildIndexControllerTest را اجرا کنید.

25. برای فعال کردن ویژگی اسکال در پروژه باید انوتیشن @EnableScheduling را در فایل اصلی Application اضافه نماییم.

26. متدهایی که قرار است در زمانبندی خاصی در پروژه اجرا شوند باید با انوتیشن @Scheduled مشخص شوند.

مثال:

```
@Scheduled(cron = "0 * 9 * * ?")  
public void cronJobSch() {....}
```

27. برای فعال کردن ویژگی ناهمزمانی در پروژه باید انوتیشن @EnableAsync را در فایل اصلی Application اضافه نماییم.

28. تنظیمات Executor در پروژه و در در فایل اصلی Application اضافه میشود.
مثال:

```
@Bean(name = "asyncExecutor")  
public Executor asyncExecutor() {  
    ThreadPoolTaskExecutor executor = new ThreadPoolTaskExecutor();  
    executor.setCorePoolSize(3);  
    executor.setMaxPoolSize(3);  
    executor.setQueueCapacity(100);  
    executor.setThreadNamePrefix("AsynchThread-");  
    executor.initialize();  
    return executor;  
}
```

29. متدهایی که قرار است به صورت ناهمزمان و در Executor مشخصی اجرا شوند باید با انوتیشن @Async مشخص شوند.
مثال:

```
@Async("asyncExecutor")  
public CompletableFuture<نام کلاس موردنظر> getAddress() throws InterruptedException {...}
```

30. برای اینکه بتوانیم در متدهای Async خروجی داشته باشیم ، خروجی متد را از نوع CompletableFuture<> تعریف مینماییم.

با تشکر

مصطفی مطهری نیا

تیم CTO

1399-04-17