

微服务项目：十次方

第一章

1. 十次方简介

《十次方》是程序员的专属社交平台，包括头条、问答、活动、交友、吐槽、招聘六大 频道。



《十次方》是程序员的专属社交平台，包括头条、问答、活动、交友、吐槽、招聘六大 频道。

十次方名称的由来：2的10次方为1024，程序员都懂的。

如果你是一位技术大咖，那么赶快发布文章，增加知名度吧。

如果你是一名技术小白，那么赶快到问答频道寻求帮助的，这里高手如云哦！

如果你不想错过各种技术交流会，那么请经常关注活动频道吧~

如果你还是单身，那么赶快到交友频道找到你心仪的另一半。 如果你有太多的苦恼，那么赶快吐个槽吧~

如果你正在找工作或是想跳槽拿高薪，那么来招聘频道淘金吧~



















2.1 系统架构

十次方采用的是前后端分离开发模式，系统架构为：

SpringBoot+SpringCloud+SpringMVC+SpringData 我们把这种架构称为
spring全家桶

2.2 模板划分

十次方项目分为18个子模块其中17个是微服务：

 jjwt_demo
 mongodb_demo
 rabbitmq_client
 rabbitmq_demo
 tensquare_article
 tensquare_base
 tensquare_common
 tensquare_config
 tensquare_eureka
 tensquare_friend
 tensquare_gathering
 tensquare_manager
 tensquare_qa
 tensquare_recruit
 tensquare_search
 tensquare_spit
 tensquare_user
 tensquare_web

第一章的内容是搭建父工程、公共子模块（tensquare_commons）、以及基础微服务（tensquare_base）；

模块名称以及对应的中文如下表

模块名称	模块中文名称
tensquare_common	公共模块
tensquare_article	文章微服务
tensquare_base	基础微服务
tensquare_friend	交友微服务
tensquare_gathering	活动微服务
tensquare_qa	问答微服务
tensquare_recruit	招聘微服务
tensquare_user	用户微服务
tensquare_spit	吐槽微服务
tensquare_search	搜索微服务
tensquare_web	前台微服务网关
tensquare_manager	后台微服务网关
tensquare_eureka	注册中心
tensquare_config	配置中心
tensquare_sms	短信微服务
tensquare_article_crawler	文章爬虫微服务
tensquare_user_crawler	用户爬虫微服务
tensquare_ai	人工智能微服务

2.3 表结构分析

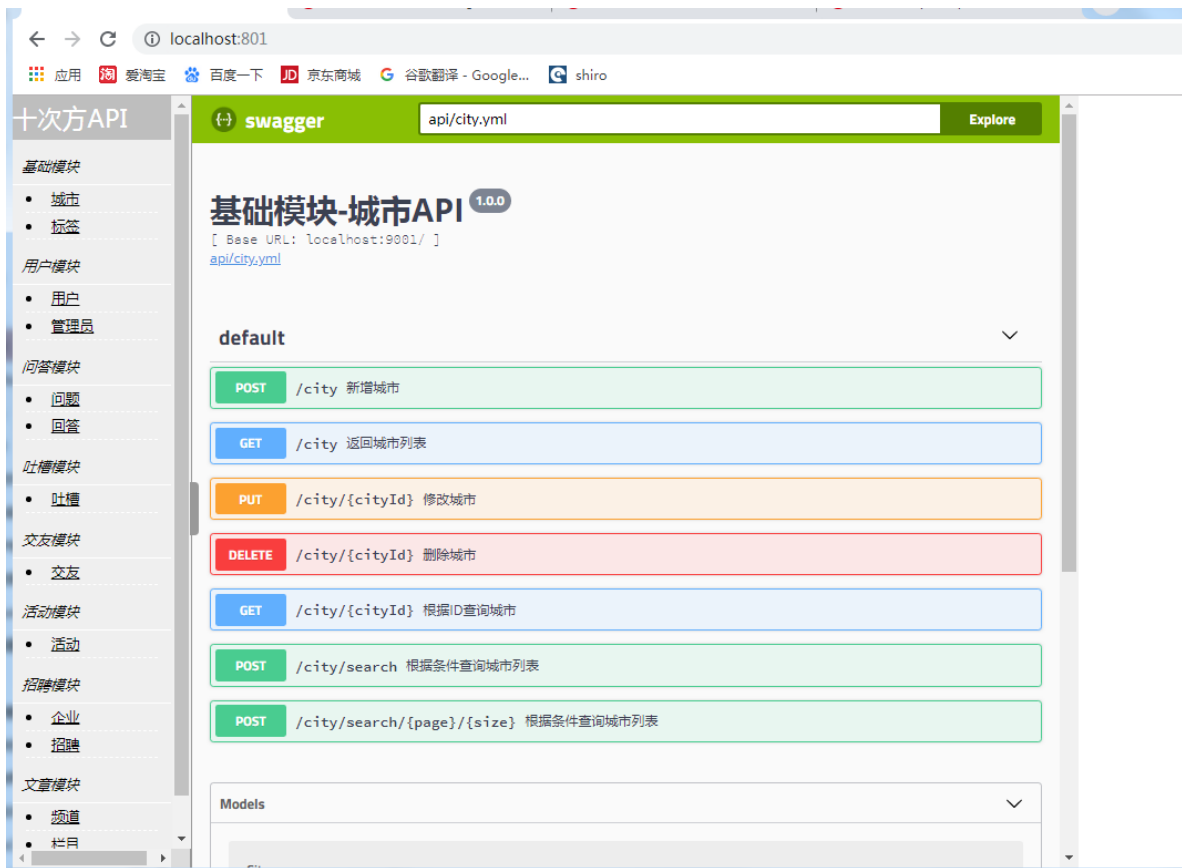
```

1  我们这里采用的分库分表设计，每个业务模块为1个独立的数据库。
2  1. tensquare_article 文章
3  2. tensquare_base 基础
4  3. tensquare_friend 交友
5  4. tensquare_gathering 活动
6  5. tensquare_qa 问答
7  6. tensquare_recruit 招聘
8  7. tensquare_user 用户
9  8. tensquare_spit 吐槽

```

2.3 API文档

提供的前后端分析开发的文档采用Swagger语言进行编写，与Nginx整合，双击Nginx.exe启动即可：



前后端约定的返回码列表

状态描述	返回码
成功	20000
失败	20001
用户名密码错误	20002
权限不足	20003
远程调用失败	20004
重复操作	20005

2.4 理解restful

2.5.1 何为restful

RESTful架构，就是目前最流行的一种互联网软件架构。它结构清晰、符合标准、易于理解、扩展方便，所以正得

到越来越多网站的采用。REST这个词，是Roy Thomas Fielding在他2000年的博士论文中提出的

REST 是Representational State Transfer的缩写，翻译是”表现层状态转化”。可以总结为一句话：REST是

所有Web应用都应该遵守的架构设计指导原则。

面向资源是REST最明显的特征，对于同一个资源的一组不同的操作。资源是服务器上一个可命名的抽象概

念，资源是以名词为核心来组织的，首先关注的是名词。REST要求，必须通过统一的接口来对资源执行各种操

作。对于每个资源只能执行一组有限的操作

7个http方法: GET / PUT / POST / DELETE / PATCH / HEAD / OPTIONS

2.5.2 接口规范

我们在项目中用到了GET/POST/PUT/DELETE四种方法，现在介绍一下这四种方法

GET

- 安全且幂等的获取表示
- 变更时获取表示
- 200 (ok) -表示已在响应中发出
- 204 (无内容) -资源有空表示
- 301 (Moved Permanently) -资源的url已经被更新
- 303 (see other) - 其他 (如, 负载均衡)
- 304 (not modified) -资源为更改 (缓存)
- 400 (bad request) -错误的请求 (如, 参数错误)
- 404 (not found) -资源不存在
- 406 (not acceptable) -服务端不支持所需表示
- 500 (internal server error) -通用错误响应
- 503 (Service Unavailable)-服务端当前无法处理请求

POST

- 不安全且不幂等
- 使用服务端管理的 (自动产生) 的实例号创建资源创建子资源
- 部分更新资源
- 如果没有被修改, 则不过更新资源 (乐观锁)

PUT

- 不安全单幂等
- 客户端管理的实例去创建一个资源
- 通过替换的方式更新资源

DELETE

- 不安全但幂等
- 删除资源

3 项目前的准备

3.1 开发环境需求

- JDK 1.8
- 数据库mysql 5.7
- idea
- maven3.3.9以上
- docker latest
- cent OS7
- VM12

注意:

给的资料里面已经有安装好docker 的centos7 镜像 并且已经下载好了mysql5.7的镜像

将centos7 挂载到VM12 修改内存为8G。用户名为root，密码itcast

3.2 使用docker容器加载mysql数据库

3.2.1 下载镜像

```
1 docker pull centos/mysql-57-centos7
```

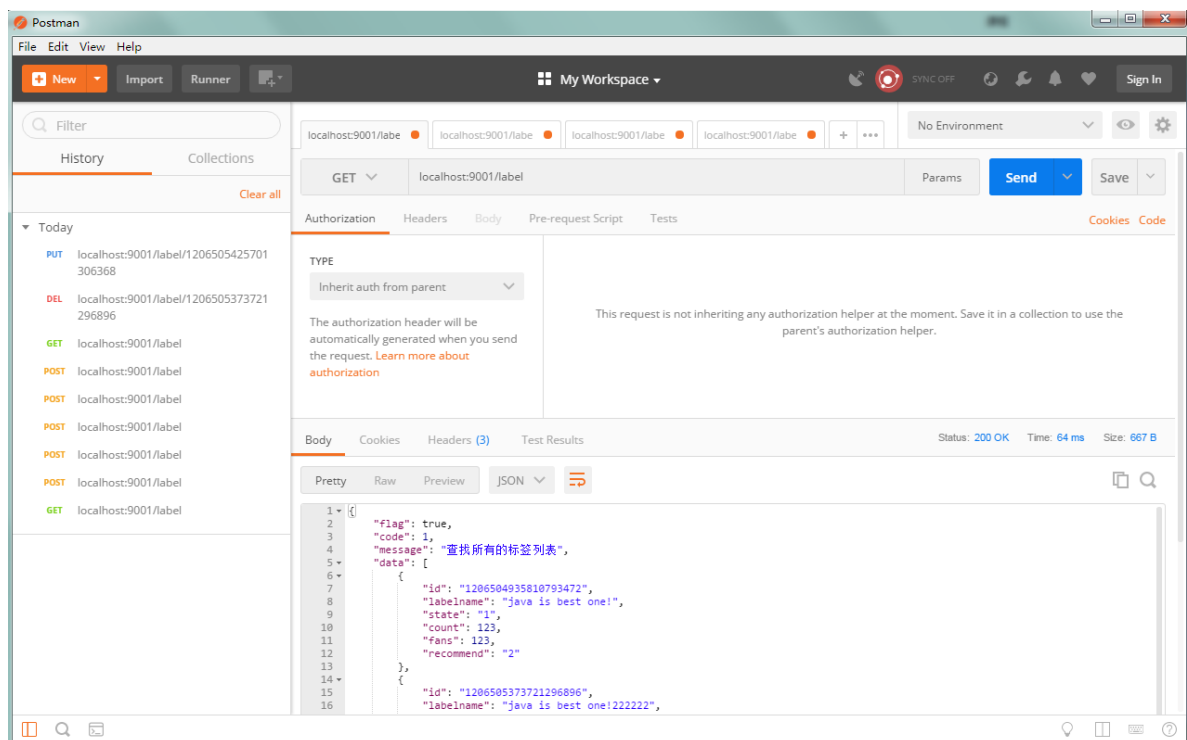
3.1.2 安装mysql数据库

```
1 docker run -it --name=mysql -p 3306:3306 -e MYSQL_ROOT_PASSWORD=123 (镜像名称: 版本号或者镜像id)
```

3.1.3 连接mysql

使用mysql的客户端工具sql yog连接远程docker中的mysql，并执行资料中的sql语句，创建数据库。

3.3 测试psotman工具



postman工具可以模拟http请求 并且能将请求中设置参数 以及查看返回的参数

4.工程搭建

4.1 搭建父工程

1)建一个普通的maven父工程 springboot项目是jar还是war看加入的配置

```
1 GroupId: com.tensquare
2 ArtifactId:tensquare_parent
3 Version:1.0-SNAPSHOT
```

2)修改pom文件:

主要是引入spring-boot-starter-web环境与spring-boot-starter-test 测试环境

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0"
3         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
5     <modelVersion>4.0.0</modelVersion>
6
7     <groupId>com.tensquare</groupId>
8     <artifactId>tensquare_parent</artifactId>
9     <packaging>pom</packaging>
10    <version>1.0-SNAPSHOT</version>
11    <modules>
12        <module>tensquare_commons</module>
13        <module>tensquare_base</module>
14    </modules>
15
16    <!--springboot父项目-->
17    <parent>
18        <groupId>org.springframework.boot</groupId>
19        <artifactId>spring-boot-starter-parent</artifactId>
20        <version>2.0.1.RELEASE</version>
21        <relativePath/>
22    </parent>
23
24    <!--相关属性值-->
25    <properties>
26        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
27        <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>
28        <java.version>1.8</java.version>
29    </properties>
30    <dependencies>
31        <!--spring-boot-starter-web 说明这是一个web项目-->
32        <dependency>
33            <groupId>org.springframework.boot</groupId>
34            <artifactId>spring-boot-starter-web</artifactId>
35        </dependency>
36        <!--spring-boot测试-->
37        <dependency>
38            <groupId>org.springframework.boot</groupId>
39            <artifactId>spring-boot-starter-test</artifactId>
40            <scope>test</scope>
41        </dependency>
42        <!--lombok小辣椒为Bean添加方法 注意要插件! -->
43        <dependency>
44            <groupId>org.projectlombok</groupId>
45            <artifactId>lombok</artifactId>
46        </dependency>
47        <!--lang3工具判断字符串null 或者空串-->
48        <dependency>
49            <groupId>org.apache.commons</groupId>
50            <artifactId>commons-lang3</artifactId>
51        </dependency>
52    </dependencies>
53
```

```

54     <repositories>
55         <repository>
56             <id>spring-snapshots</id>
57             <name>Spring Snapshots</name>
58             <url>https://repo.spring.io/snapshot</url>
59             <snapshots>
60                 <enabled>true</enabled>
61             </snapshots>
62         </repository>
63         <repository>
64             <id>spring-milestones</id>
65             <name>Spring Milestones</name>
66             <url>https://repo.spring.io/milestone</url>
67             <snapshots>
68                 <enabled>false</enabled>
69             </snapshots>
70         </repository>
71     </repositories>
72     <pluginRepositories>
73         <pluginRepository>
74             <id>spring-snapshots</id>
75             <name>Spring Snapshots</name>
76             <url>https://repo.spring.io/snapshot</url>
77             <snapshots>
78                 <enabled>true</enabled>
79             </snapshots>
80         </pluginRepository>
81         <pluginRepository>
82             <id>spring-milestones</id>
83             <name>Spring Milestones</name>
84             <url>https://repo.spring.io/milestone</url>
85             <snapshots>
86                 <enabled>false</enabled>
87             </snapshots>
88         </pluginRepository>
89     </pluginRepositories>
90 </project>

```

4.2 搭建公共子模块

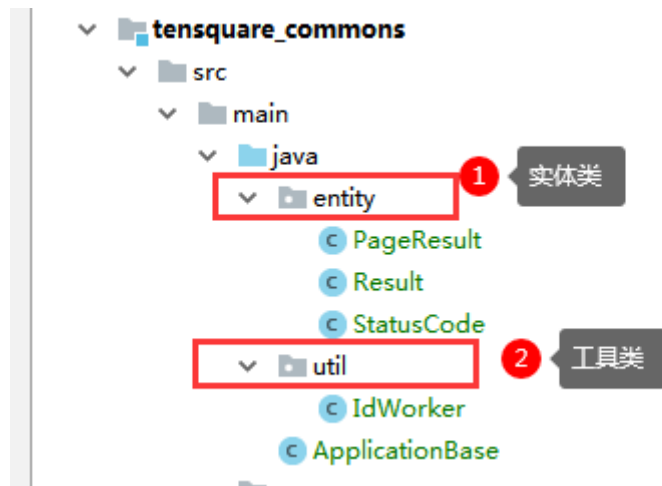
1) 新建普通maven项目

```

1  groupId: com.tensquare
2  artifactId: tensquare_commons
3  version: 1.0-SNAPSHOT

```

各种结构如下：



工具类暂时不需要依赖什么包

2)根据 [开发文档的返回形式](#) 来设计我们的festful返回类型：如下

```
1 package entity;
2
3 import lombok.AllArgsConstructor;
4 import lombok.Data;
5 import lombok.NoArgsConstructor;
6
7 /*
8  * @Author:wtp
9  * @Date:Create at 2019/12/16 0016
10  * @TIME:下午 3:32
11  * @Description:统一的返回结果类
12  * @Tips:
13  */
14 @Data //小辣椒的注解表示set get方法 等一些方法
15 @AllArgsConstructor //小辣椒的注解表示 全参构造
16 @NoArgsConstructor //小辣椒的注解表示 无参的构造
17 public class Result {
18     private boolean flag;//是否成功
19     private Integer code;// 返回码
20     private String message;//返回信息
21     private Object data;// 返回数据
22
23     //业务需要自己写的一个构造方法
24     public Result(boolean flag, Integer code, String message) {
25         this.flag = flag;
26         this.code = code;
27         this.message = message;
28     }
29 }
30
```

分页:

```
1 package entity;
2
3 /*
4  * @Author:wtp
5  * @Date:Create at 2019/12/16 0016
6  * @TIME:下午 3:36
```

```

7  * @Description:分页的实体结果类
8  * @Tips:
9  */
10 import lombok.AllArgsConstructor;
11 import lombok.Data;
12 import lombok.NoArgsConstructor;
13 import java.util.List;
14
15 @Data //小辣椒的注解表示set get方法 等一些方法
16 @AllArgsConstructor //小辣椒的注解表示 全参构造
17 @NoArgsConstructor //小辣椒的注解表示 无参的构造
18 public class PageResult<T> {
19     private long total; //总记录数
20     private List<T> rows; //每页的结果集
21 }
22

```

返回的状态码（静态常量类）

```

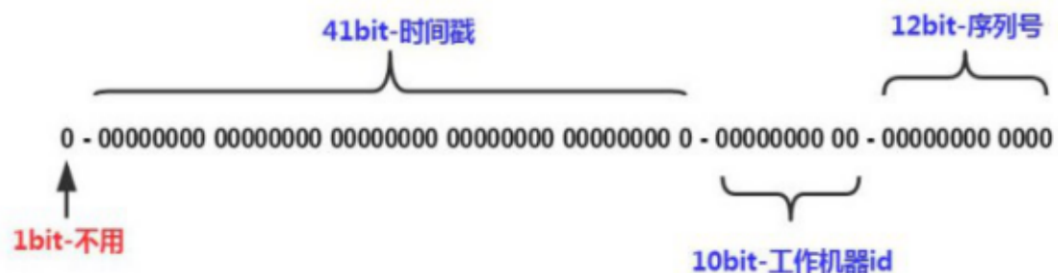
1  package entity;
2
3  /*
4   * @Author:wtp
5   * @Date:Create at 2019/12/16 0016
6   * @TIME:下午 3:38
7   * @Description:返回的状态码类 静态的常量类 直接类名.常量名就行了
8   * @Tips:
9   */
10 public class StatusCode {
11     public static final int OK=20000;//成功
12     public static final int ERROR =20001;//失败
13     public static final int LOGINERROR =20002;//用户名或密码错误
14     public static final int ACCESSERROR =20003;//权限不足
15     public static final int REMOTEERROR =20004;//远程调用失败
16     public static final int REPERROR =20005;//重复操作
17 }
18

```

工具类

idWorker用雪花算法随机生成的19位的id

由于我们的数据库在生产环境中要分片部署（MyCat），所以我们不能使用数据库本身的自增功能来产生主键值，只能由程序来生成唯一的主键值。我们采用的是开源的twitter（非官方中文惯称：推特 是国外的一个网站，是一个社交网络及微博客服务）的snowflake（雪花）算法。



5. 基础微服务CRUD

5.1 搭建基础微服务

1) 新建普通maven项目

```
1  groupId: com.tensquare
2  artifactId: tensquare_parent
3  version: 1.0-SNAPSHOT
```

2) 引入依赖

主要是引入数据库连接相关依赖 以及 jpa

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <project xmlns="http://maven.apache.org/POM/4.0.0"
3          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4          xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
5          http://maven.apache.org/xsd/maven-4.0.0.xsd">
6      <parent>
7          <artifactId>tensquare_parent</artifactId>
8          <groupId>com.tensquare</groupId>
9          <version>1.0-SNAPSHOT</version>
10     </parent>
11     <modelVersion>4.0.0</modelVersion>
12     <artifactId>tensquare_base</artifactId>
13
14     <dependencies>
15         <!-- jpa 与数据库连接相关 -->
16         <dependency>
17             <groupId>org.springframework.boot</groupId>
18             <artifactId>spring-boot-starter-data-jpa</artifactId>
19         </dependency>
20         <dependency>
21             <groupId>mysql</groupId>
22             <artifactId>mysql-connector-java</artifactId>
23         </dependency>
24
25         <!-- 本项目的commons -->
26         <dependency>
27             <groupId>com.tensquare</groupId>
28             <artifactId>tensquare_commons</artifactId>
29             <version>1.0-SNAPSHOT</version>
30         </dependency>
31     </dependencies>
32 </project>
```

3) 基础微服务项目结构（与mvc类似）



按照开发文档的要求

我们写控制器到修改数据库的操作

开发文档：

API documentation for the '标签' (Label) module. The left sidebar shows the project structure with '标签' highlighted. The main content area lists the following endpoints:

Method	URL	Description
POST	<code>/label</code>	增加标签
GET	<code>/label</code>	标签全部列表
GET	<code>/label/toplist</code>	推荐标签列表
GET	<code>/label/list</code>	有效标签列表
GET	<code>/label/{labelId}</code>	根据ID查询
PUT	<code>/label/{labelId}</code>	修改标签
DELETE	<code>/label/{labelId}</code>	根据ID删除
POST	<code>/label/search/{page}/{size}</code>	标签分页
POST	<code>/label/search</code>	标签分页

根据这里的url来写控制器