# 设计报告

### 实体设计

Base：

其他实体的基础，包含id（主键），createTime（创建时间），updateTime（更新时间）

User：

继承Base，是网站的用户实体。

包含username（用户名），password（密码）

Gym：

继承Base，是健身房实体。

包含name（健身房名称），info（健身房信息，如位置），trainers（关联实体Trainer列表）

Trainer：

继承Base，是教练实体

包含name（教练名称），info（教练信息），gymId（外键，指向Gym的id）

Gym和Trainer之间是多对一关系

### 数据源设计

采用了多数据源，可以同时连接两个数据库

一个是app\_user,一个是gym，数据库自己创建，表是spring自动创建的

app\_user里面有user表，



Gym里面有gym表



和trainer表



application.properties中的相关配置：

spring.datasource.user.url=jdbc:mysql://127.0.0.1:3306/app\_user?useSSL=false  
spring.datasource.user.username=root  
spring.datasource.user.password=123456  
spring.datasource.user.driver-class-name=com.mysql.cj.jdbc.Driver  
spring.datasource.user.max-idle=10  
spring.datasource.user.max-wait=10000  
spring.datasource.user.min-idle=5  
spring.datasource.user.initial-size=5  
spring.datasource.gym.url=jdbc:mysql://127.0.0.1:3306/gym?useSSL=false  
spring.datasource.gym.username=root  
spring.datasource.gym.password=123456  
spring.datasource.gym.driver-class-name=com.mysql.cj.jdbc.Driver  
spring.datasource.gym.max-idle=10  
spring.datasource.gym.max-wait=10000  
spring.datasource.gym.min-idle=5  
spring.jpa.properties.hibernate.hbm2ddl.auto=update  
spring.jpa.show-sql=true

### Repository设计

User表对应UserRepository

public interface UserRepository extends JpaRepository<User, Integer> {  
 User findByUsernameAndPassword(String username, String password);  
 User findByUsername(String username);  
}

findByUsernameAndPassword：验证用户名和密码

findByUsername：根据用户名查询数据库

Gym表对应GymRepository

public interface GymRepository extends JpaRepository<Gym, Integer> {  
 Gym findByName(String name);  
}

findByName：根据健身房名称查询数据库

Trainer表对应TrainerRepository

public interface TrainerRepository extends JpaRepository<Trainer, Integer> {  
 List<Trainer> findByName(String name);  
}

findByName：根据教练姓名查询数据库

### Service设计

##### UserService：

登录功能：

*/\*\*  
 \** ***@desc*** *：验证用户账号和密码  
 \** ***@param*** *：username：用户名  
 \* password：密码  
 \** ***@return*** *：Object[0]是查询状态  
 \* Object[1]是查询到的User  
 \*/*

public Object[] login(String username, String password)

注册功能：

*/\*\*  
 \** ***@author*** *：Yimyl  
 \** ***@date*** *：Created in 2019.4.27  
 \** ***@desc*** *：注册用户  
 \** ***@param*** *：username：用户名  
 \* password：密码  
 \** ***@return*** *：注册的状态  
 \*/*public String register(String username, String password)

判断用户名是否已注册：

*\*\*  
 \** ***@desc*** *：用户名是否可以注册  
 \** ***@param*** *：username：用户名  
 \** ***@return*** *：false：已使用，不可以注册  
 \* true：未使用，可以注册  
 \*/*public boolean isUsernameCanRegister(String username)

#### GymService:

查询健身馆：

*/\*\*  
 \** ***@param*** *：queryBy：若为Integer，根据体育馆id查询；若为String，根据体育馆名称查询  
 \** ***@return*** *：查询到的体育馆  
 \** ***@desc*** *：查询体育馆信息  
 \*/*public Gym query(Object queryBy)

pageQuery：query的分页查询版

*/\*\*  
 \** ***@desc*** *：分页查询体育馆，按id排序  
 \** ***@param*** *：  
 \** ***@return*** *：查询页码的数据  
 \*/*public List<Gym> pageQuery(int page, int size)

pageQuery：query的分页查询版，可选择按Id或name排序

*/\*\*  
 \** ***@desc*** *：分页查询，按照sortBy排序  
 \** ***@param*** *：page：页码  
 \* size：每页大小  
 \* sortBy：排序的依据，若为“id”则按照id排序，若为“name”则按照name排序  
 \** ***@return*** *：查询页码的数据  
 \*/*public List<Gym> pageQuery(int page, int size, String sortBy)

#### TrainerService:

query：

*/\*\*  
 \** ***@param*** *：queryBy：若为Integer，根据教练id查询；若为String，根据教练姓名查询  
 \** ***@return*** *：查询到的教练(可能多个)  
 \** ***@desc*** *：查询教练信息  
 \*/*public List<Trainer> query(Object queryBy)

pageQuery：query的分页查询版

*/\*\*  
 \** ***@param*** *：page：页码 size：每页大小  
 \** ***@return*** *：该页码中的数据  
 \** ***@author*** *：Yimyl  
 \** ***@date*** *：Created in 2019.4.27  
 \** ***@desc*** *：分页查询，按id排序  
 \*/*public List<Trainer> pageQuery(int page, int size)

pageQuery：query的分页查询版，并且选择排序方式为id或name

*/\*\*  
 \** ***@param*** *：page：页码 size：每页大小  
 \* sortBy：排序的依据，若为“id”则按照id排序，若为“name”则按照name排序  
 \** ***@return*** *：该页码中的数据  
 \** ***@desc*** *：分页查询，按id或name排序  
 \*/*public List<Trainer> pageQuery(int page, int size, String sortBy)

findTrainerOfGym：

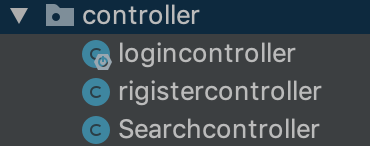
*/\*\*  
 \** ***@param*** *：queryBy：健身馆信息（id 或name ）  
 \** ***@return*** *：查到的教练  
 \** ***@desc*** *：分页查询健身馆中的教练  
 \*/*public List<Trainer> findTrainerOfGym(GymService gymService, Object queryBy,int page, int size)

page：实现对List进行分页

*/\*\*  
 \** ***@param*** *：list：要分页的链表  
 \** ***@return*** *：对应页码的链表  
 \** ***@author*** *：Yimyl  
 \** ***@date*** *：Created in 2019.4.27  
 \** ***@desc*** *：对链表分页  
 \*/*private List<Trainer> page(List<Trainer> list, int page, int size)

Controller设计

共有三个控制器，分别控制登陆，注册，查询



其中logincontroller,在mapping到”/”,””,”login”时返回登陆界面，

其中，邮箱，密码有校验，为空或是不符合规范会报错，账号密码不对时后端返回一个对象，登陆界面显示账号密码不对。有一个超链接跳转到注册页面

rigistercontroller控制注册，注册按钮也类似，同样有校验，点击注册按钮会在后端校验是否是合法的注册。

Searchcontroller控制搜索，什么也不输入，点击查询按钮可以搜索所有的俱乐部和教练信息，支持输入俱乐部id(int)或名称(string),以及教练id或名称来查询，根据不同的情况，controller返回不同的对象到前端，前端根据返回的对[文件]

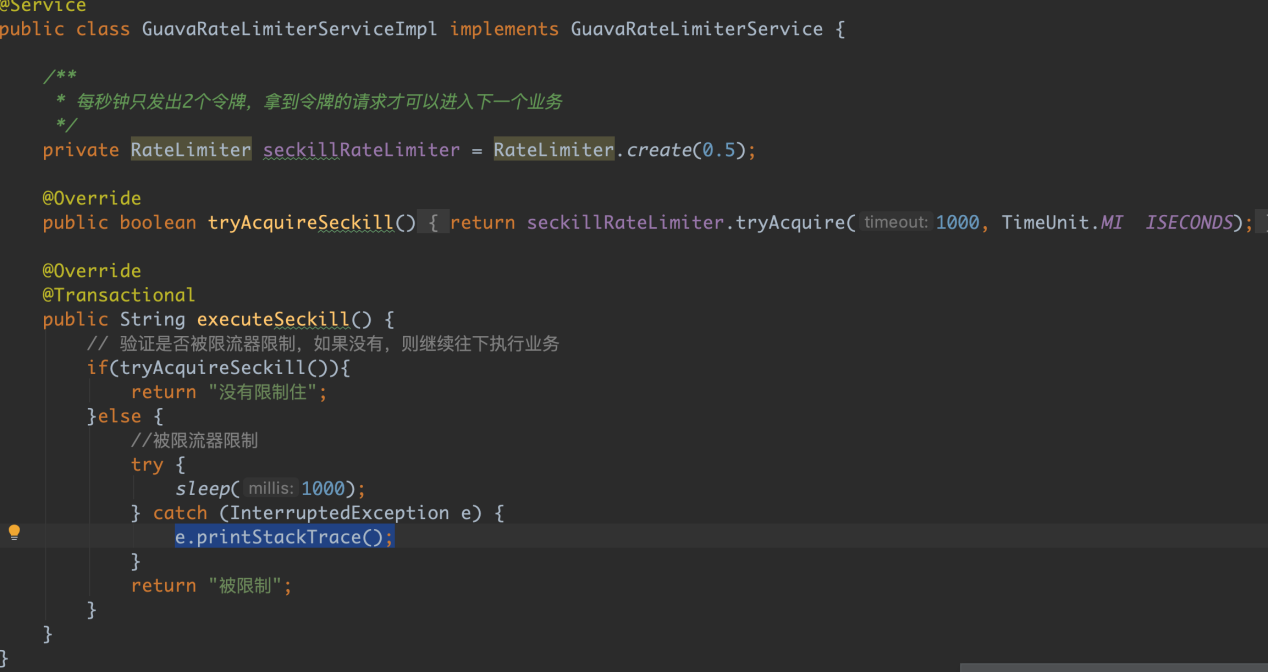
象显示对应的数据。

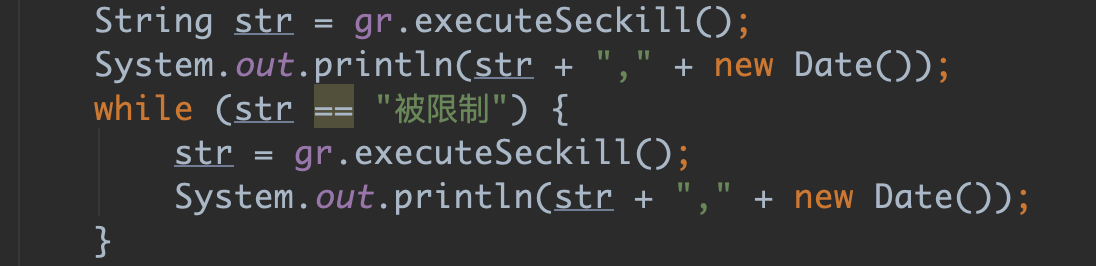
**RESTful Service**

资源有资源标识，controller设计符合restful规范

**RateLimiter**

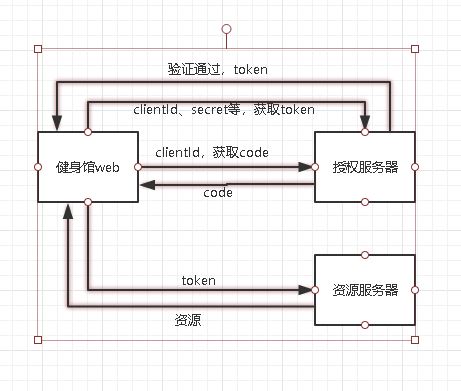
**使用Guava-RateLimiter，令牌桶算法，每秒钟生成一0.5个令牌，只有获得令牌的请求才可以继续，否则需要等待令牌的获取。**





#### Oauth2设计

我们的web程序，可以通过Oauth2来实现第三方登录注册。我们健身馆web作为第三方应用程序，可以通过授权服务器获取资源服务器上的用户资料，从而可以在我们的web上进行注册、登录。



这里的资源就是用户的信息，用来注册和登录。

#### 缓存设计

1.spring cache

通过注解缓存从数据库中得到的数据

2.http缓存：

设置http头部

Cache-Control：max-age

Last-Modified：

等属性，实现浏览器内部缓存，减少传输数据。

A4设计报告

1.数据层设计

将数据库改为MongoDB-react以适应Reactive编程。

2.Mono+thymeleaf

在Controller中新增方法，view层使用thymeleaf模版引擎，处理MogoDB返回的数据并显示在前端页面上

