3.1接口设计

3.1.1设计概述

在本次的JavaEE实训项目中，我们前端使用了Vue框架，后端主要使用SpringBoot和Mybatis框架。由于本项目分为用户中心模块、文章管理模块、创作者中心模块、管理员模块，因此想让这些模块之间互相协调工作，就必须在每个模块上开放以便其他模块调用的接口。从而使得这些模块能相对独立但又有条不紊的担任自己的职责。接口的数据传递基本都是由前端->Controller层->Service层->Dao层。

3.1.2用户中心模块

该模块是对用户的信息进行保存以及修改的模块，是整个项目的重要组成部分。几乎整个项目的功能模块都会与该模块存在依赖关系，例如在用户登录的时候就需要调用该模块去查询有没有存在此用户，该用户输入的账号和密码是否正确。对于注册功能，也需要调用该模块去添加用户。该模块提供对用户的多种查询接口，例如按用户昵称查询用户信息（getUserByName），按用户的在数据库中的id（唯一标识主键）来查询用户信息（getUserById）等，当然还向前端提供对于用户信息进行修改的接口，开发人员可根据项目内的Controller而文件夹下的各个Controller类中所写的接口进行调用便可完成相应的功能而无需担心底层的实现。

3.1.3文章管理模块

该模块与数据库之间的交流频繁，且每一次与数据库之间交互的数据量较大。该模块主要包括对文章的内容的管理，对于发帖操作，我们在前端设置了一个发帖的界面，当用户提交文章的时候，用户的基本信息、文章内容甚至是图片文件都会被传到服务器端，服务器会将一条文章的各个组成部分分开并行处理，对于上传的图片文件，我们则是通过阿里的云存储服务将图片存至服务器并范围唯一的资源路径名。对于文章的文字信息内容，我们采用html源码的格式存储，首先会根据发帖用户的昵称调用上述用户管理模块所提供的查询接口对用户的Id进行查询，得到id之后，便会将id与文章文字数据一同打包并通过“ArticleController”提供的添加文章接口与部署在服务器上的数据库进行交互，并将文章内容、发帖人id等信息写入数据库，并返回操作结果。对于文章的删除，与添加文章的过程基本相同，在ArticleController层中提供deleteArticle接口，系统便会根据request对象来对相应文章进行删除。当然最为重要的接口是对文章的查询，我们在此提供了多个查询接口，可以利用getArticlebByLabel接口来取得指定板块下的所有文章，也可以无条件式的利用getAllArticle接口来取得所有的文章。

此外我们还设计了草稿箱功能，其本质就是将作者写好的文章采用与发布文章类似的方法存到云端服务器中，但是设置标识与已发布的文章区别开。因此，这一功能在已有接口的基础上只增加了一个editDraft接口，用来继续编辑草稿箱的文章内容，通过此接口将服务器的数据渲染到前端的md编辑器中，供作者继续编辑。

3.1.4创作者中心模块

这一模块的功能与用户中心模块和文章管理模块有着很大的联系。它主要用来展示用户作为创作者，所发布文章的数据情况以及粉丝情况。在创作者中心的首页有一个网格展示了6类数据，这其实是通过CreateController中的6个接口实现的，它们分别查询数据库各类表，提供了这6类的当日数据（我们在设计数据库时，增加了时间字段便于使用）和累计数据。近期发布则采用了ArticleController中的getAllArticle接口，并按时间排序。文章管理和草稿箱管理，也同样调用了ArticleController中的几个接口提供增删改查的操作。内容数据与关注数据下的各类数据，则是由CreateController中的多个接口实现的，因为涉及到统计文章的点赞评论收藏回复等数据，以及用户的关注和粉丝数据，功能相对独立，所以单独设计了接口通信数据库返回此模块需要的数据，和提供关注的删除与查看（与用户中心模块联系，通过getUserByName实现）操作。

3.1.5管理员模块

该模块的功能较为简单，更像是作为一个后台管理系统，直接进行数据库的底层操作，与论坛的主要功能独立。正是因为这个特性，我们采用SpringSecurity给AdminController下的所有接口都设置了权限，只有角色为管理员的用户才能访问此界面并且操作。因此，这一模块的所有功能都不涉及其他模块的Controller层，仅仅能通过AdminController实现，但因为它的独立性，也不存在代码的冗余。这一模块又能够分为三个小模块：人员管理（User）、标签管理（Label）、投诉管理（Report），它们各自拥有接口，实现用户、标签、被投诉文章的增删改查功能。