基于SpringBoot的会议系统平台设计

作者姓名：谢强 专业班级：通信工程4班 指导教师：赵义红

摘要

SpringBoot是一些框架的集合，由Pivotal团队提供的全新框架，它能够被任意的构建系统所使用，设计目是为了简化Spring框架的搭建、配置、开发、调试和最后部署到生产环境上进行生产，配置只需要在库包管理中添加少量的配置信息，就可以把开发者从Spring框架的xml配置文件中解放出来。目前SpringBoot几乎集成了大部分的主流框架，例如：日志框架，持久化框架，JMS框架,缓存框架等。此外，SpringBoot在很大程度上简化了Spring各个应用方面的配置，其目标致力于在蓬勃发展的快速应用开发领域成为领导者。

本文采用SpringBoot作为后台设计框架，vue技术作为前端设计框架，选用mysql数据库、redis缓存技术作为数据存储以及邮箱验证机制实现会议系统平台设计，通过浏览器（browser）访问前端服务器（node），接收到请求之后，由前端服务器作为代理（proxy）对后台服务器进行数据请求，中间需要进行安全认证（token），安全认证的密钥保存在redis缓存中,通过之后进行逻辑处理，从数据库获取数据。平台将传统会议的部分流程简化，使会议的形式变得更加人性化，减少人工准备会议的工作量，更加丰富了会议的内容。

**关键字：**Spring；SpringBoot；框架；vue框架；会议系统平台

Design of conference system platform based on SpringBoot

**Abstract**: Spring Boot is a collection of libraries. The new framework provided by the Pivotal team can be used by any project's build system. The design goal is to simplify the construction, configuration, development, debugging, and final deployment of the Spring framework to the production environment. Production, configuration only need to add a small amount of configuration information in the library package management tool, you can free up the developer from the Spring framework's xml configuration file. Currently, SpringBoot integrates almost all of the mainstream frameworks such as: logging framework, persistence framework, JMS framework, cache framework, etc. In addition, SpringBoot greatly simplifies the configuration of Spring's various applications, and its goal is to become a leader in the rapid development of rapid application development.  
This paper uses SpringBoot as a background design framework, vue technology as a front-end design framework, using mysql database, redis cache technology as a data storage and mailbox authentication mechanism to achieve the design of the conference system platform, through the browser (browser) to access the front-end server (node), received After the request, the front-end server acts as a proxy to request data from the back-end server. A security authentication (token) is required in the middle. The security authentication key is stored in the redis cache, and logic processing is performed to obtain data from the database. The platform simplifies part of the processes of traditional meetings, making the format of the meetings more user-friendly, reducing the amount of manual preparation for meetings, and enriching the content of the meetings.

**Key words**: Spring; SpringBoot; Framework;Vue Framework;Conference System Platform

目录

[第1章 前言 1](#_Toc515644882)

[1.1 背景及研究意义 1](#_Toc515644883)

[1.2 本文的主要内容 1](#_Toc515644884)

[1.3 系统平台的预期目标 2](#_Toc515644885)

[第2章 技术介绍及方案选择 3](#_Toc515644886)

[2.1 SpringBoot介绍 3](#_Toc515644887)

[2.2 SpringBoot应用开发架构设计 3](#_Toc515644888)

[2.3 方案选择 4](#_Toc515644889)

[2.3.1 方案一 4](#_Toc515644890)

[2.3.2 方案二 5](#_Toc515644891)

[2.3.3 方案三 6](#_Toc515644892)

[第3章 系统平台框架及软件 8](#_Toc515644893)

[3.1 整体架构 8](#_Toc515644894)

[3.1.1 物理模型 8](#_Toc515644895)

[3.1.2 逻辑模型 8](#_Toc515644896)

[3.2 前端框架vue 9](#_Toc515644897)

[3.2.1 vue初始化 10](#_Toc515644898)

[3.2.2 组件componets 11](#_Toc515644899)

[3.2.3 node框架 11](#_Toc515644900)

[3.2.4 nuxt框架 11](#_Toc515644901)

[3.3 后端框架 12](#_Toc515644902)

[3.3.1 SpringMvc 12](#_Toc515644903)

[3.3.2 Swagger2 13](#_Toc515644904)

[3.3.3 JWT token 13](#_Toc515644905)

[3.3.4 JPA数据库操作 13](#_Toc515644906)

[3.4 开发软件 14](#_Toc515644907)

[3.4.1 IEDA开发软件 14](#_Toc515644908)

[3.4.2 SQLyog 14](#_Toc515644909)

[3.4.3 redis Server 15](#_Toc515644910)

[3.4.4 Sublime 15](#_Toc515644911)

[第4章 系统设计与实现 17](#_Toc515644912)

[4.1 前端项目开发 17](#_Toc515644913)

[4.1.1 组件开发 17](#_Toc515644914)

[4.1.2 页面开发 17](#_Toc515644915)

[4.2 数据库表设计 18](#_Toc515644916)

[4.2.1 mysql数据库设计 18](#_Toc515644917)

[4.2.2 redis数据库设计 19](#_Toc515644918)

[4.3数据存储 20](#_Toc515644919)

[4.3.1 mysql数据库存储 20](#_Toc515644920)

[4.3.2 redis数据库 21](#_Toc515644921)

[4.4 后台功能开发 21](#_Toc515644922)

[4.4.1 安全认证 21](#_Toc515644923)

[4.4.2 文件上传 22](#_Toc515644924)

[4.4.3 文章存储 22](#_Toc515644925)

[4.4.4 图片保存 23](#_Toc515644926)

[第5章 系统部署与测试 25](#_Toc515644927)

[5.1 系统平台部署 25](#_Toc515644928)

[5.1.1 后端部署 25](#_Toc515644929)

[5.1.2 前端配置 25](#_Toc515644930)

[5.2 平台测试 26](#_Toc515644931)

[5.2.1 服务端测试 26](#_Toc515644932)

[5.2.2 前端测试 26](#_Toc515644933)

[结论 28](#_Toc515644934)

[致谢 29](#_Toc515644935)

[参考文献 30](#_Toc515644936)

# 第1章 前言

## 1.1 背景及研究意义

在当今这个互联网时代格局下，各大互联网公司，如：阿里巴巴、腾讯、百度等都从其中收获了巨大的效益。通过各自的平台，推出新的技术，影响着整个互联网的发展。新的技术不断地被推出，对旧版本进行改进的同时，又在不断地进行创新。正是由于技术的不断革新，我国的科技水平在这几年得到了飞速的发展，超级计算机，航天事业，大数据，军用设备，达到甚至超越国际领先水平，进入国际前列。当然不仅仅在这些领域，其他领域也在互联网时代下受到猛烈的冲击，web企业级应用就是其中之一。近年来使用Spring进行开发越来越流行，很多开发者，甚至是企业都使用Spring架构进行开发，它不仅简化了开发的模式，而且提高了开发的效率，受到了很多人的青睐。随着开源社区的不断壮大，技术的不断更新，为了最大程度的简化开发的流程，加快应用软件的部署，SpringBoot在这样的趋势下出现了。它使得企业级应用系统开发、维护变得更加的简便。只需要简单的几步就可以启动一个web应用。当然最大的优点在于它集成了许多开发普遍采用的框架，解决了项目之间文件包依赖和稳定性的问题。除此之外，在其内部默认设置了Tomcat服务器和Jetty服务器，或者直接打包成jar包，我们可以不用在外部服务器就可以启动应用，此外，通过使用starterPOMs的配置方式简化了Maven的配置。基于这些诸多的优势，SpringBoot给Spring框架以及web开发带来了巨大的改变和影响，正在悄悄的改变着互联网企业的开发和架构模式。

## 1.2 本文的主要内容

本文主要介绍SpringBoot技术，mysql数据库，redis缓存技术，vue前端技术，邮箱验证机制，设计一个会议系统平台，以互联网的形式展现会议的精彩回顾、重要日期通知、会议举办地点、文章管理、资料下载、事情的通知，将传统会议举办中某些流程通过平台的形式进行呈现，既达到了会议的简单化，又使得会议的举办不像传统形式那样枯燥。不仅解决目前在会议中部分繁杂的事物，而且减少会议工作人员在会议开始之前的部分准备工作，实现会议的智能化。

## 1.3 系统平台的预期目标

利用计算机技术，通过互联网形式，将会议从传统意义上的人工服务转变为互联网服务，提高会议前期准备工作效率，会议的举办紧跟互联网时代的步伐，让互联网的便利使得会议的举行更加的简便。系统平台设计开发的预期目标实现以下功能：会议系统平台应用的展示、用户的注册与登录、系统平台的管理、文章的编写与发布、系统平台图片的上传与更改、邮箱的认证。

# 第2章 技术介绍及方案选择

SpringBoot框架由Pivotai团队于2013提供。在Spring框架的基础上进行改进，摒弃了Spring框架繁杂的配置。同时由于开发社区的不断壮大，SpringBoot的出现使得应用程序的开发和维护变得更加的简单方便，SpringBoot正在给web开发带来巨大的改变和影响。

## 2.1 SpringBoot介绍

SpringBoot在继承Spring有点的同时，也在对Spring进行改进，它简化了应用程序开发基本框架的搭建，部署和维护，部分特点如下所示：

1. 在Spring框架的基础上进行集成，将开发中普遍用到的框架进行整合，实现开箱即用机制；
2. 帮助开发人员在较短时间内构建一个项目，只需使用SpringBoot的引导页面，在较短的时间内就可以完成一个项目的搭建；
3. SpringBoot支持各种各样的服务，如：JPA、RESTFUL API、Docker、Web等。这些对进行微服务设计起到了很好的支持，将整体架构服务的复杂性进行拆分，表示为一个个独立的子模块，这些独立的子模块可以进行单独的实现和部署，当需要对功能进行更改的时候，只需对单独的子模块进行改动即可，不影响整体的功能；
4. 其内部内置了tomcat、jetty服务器，或者将项目打包成jar包，通过java –jar xxx.jar就可以运行项目，实现启动的简单化；
5. 使用了starterPOMs的配置，简化库依赖的配置，对大多数的开发人员，经常使用Spring和SpringMvc，使用Spring或者SpringMvc时需要添加大量的依赖，而大多数的依赖都是固定的，SpringBoot通过starter的形式帮助我们大幅度的简化Maven的配置；
6. 集成了支持大中型项目的功能特点：系统状态监测、自动配置、系统安全等。

## 2.2 SpringBoot应用开发架构设计

从项目的前台到后端的设计中，页面的设计已经摒弃了jsp技术作为页面的开发。前端常用到的开发框架是模板引擎，流行的有Freemarker，它是用Java 语言实现的模板引擎，通过模板的形式来生成文本进行输出。Thymeleaf也是模板引擎，它的原理和Freemarker的实现原理类似，可以到达将前端的页面设计和后端的应用逻辑处理进行分离。假若将前端的页面和后端的逻辑处理都设计在一个项目中的话，推荐使用模板引擎Thymeleaf进行前端页面开发，原因在于Thymeleaf可以很好的和SpringMvc框架进行结合，SpringMvc是后端逻辑中必不可少的框架之一。而且与其他的模板引擎相比，Thymeleaf模板引擎能够在浏览器中正确的显示。除了常用的模板引擎之外，在前端的页面开发中还常用到bootstrap、AngularJS、JQuery等技术。

后端的数据传输中，一般通过浏览器进行系统平台的访问，采用json格式的数据进行传输，这种形式的数据在处理的时候非常方便，而且在数据进行传输的时候，可以很好地保证数据的安全性。数据在到达服务器之后，通过SpringMvc框架拦截所有的http请求，目前几乎所有的企业级应用开发中，SpringMvc已经替代了struts框架，成为控制层的主流。经过业务逻辑处理（Service）到达数据持久层（Dao），目前流行的数据持久层框架有Hibernate、mybatis、JPA，它们之间各有优缺点，SpringBoot中建议采用SpringData与JPA的形式，引擎则采用Hibernate，数据库采用Mysql。

## 2.3 方案选择

本文在设计的初期阶段，查阅资料，结合自身学习的知识，选定三种技术方案作为本次设计的架构。在三种技术方案都能达到最终设计目的基础上，对比三者的优势与不足，最后选定一个作为设计方案。各方案的技术描述如下：

### 2.3.1 方案一

方案一考虑的是传统的架构体系SSH来进行搭建，全称struts、spring、hibernate。作为一个集成的框架，在过去的几年中，这套开发框架在企业的应用程序开发中是比较流行的。

使用SSH框架进行搭建的系统，设计主要分为四层进行开发：视图层(View)、业务逻辑层(Service)、数据持久层(Dao)、域模块层(Domain)。这样的好处在于各个部分独立开发，程序之间的耦合性比较小，代码的可复用性比较高，即使在后期遇到系统出现问题或者需要给系统增加新的功能的时候，可以很好的进行维护与开发。此外，采用这样的方式可以帮助开发者在很短的时间内搭建结构清晰的开发框架。在SSH框架用struts作为应用系统整体架构，分离视图层、业务逻辑层、数据持久化层。模型部分，控制业务(Service)的跳转，利用Hibernate的框架对数据持久化(Persistent)提供支持；Spring是整个应用程序的管理模块，它管理着Struts、Hibernate之间的调用。架构体系图如图2-1所示：

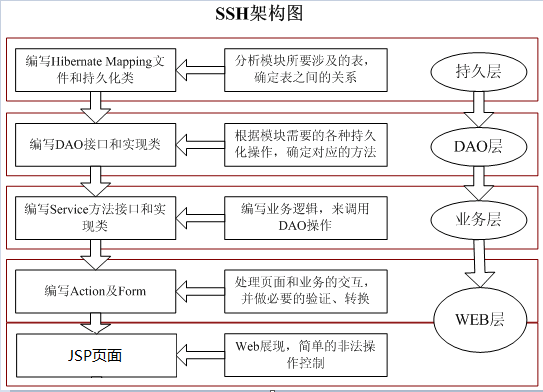


图2-1 SSH架构图

但是这个体系框架也存在着一些不足之处，以至于现在许多的企业都放弃了使用其作为架构体系。当页面向后台传送很多的参数的时候，在控制层就会添加很多的属性，实现他们的getter和setter方法，这样就会导致控制层的方法很臃肿，给人的感觉很乱。不仅如此其过度依赖servlet，在处理action的时候必须要依赖HttpRequest和HttpResponse对象。所以我否定了这个方案作为我的开发方案。

### 2.3.2 方案二

SSM框架，目前在许多大型企业级应用系统开发中普遍采用。SSM全称SpringMvc、Spring、mybatis。是标准的MVC模式，将整个系统分为视图层(View)、控制层(Controller)、业务逻辑层(Service)，数据持久层(Dao)。采用这样的架构体系，各个模块独立开发，提高开发的效率，缩小模块与模块之间的耦合性，复用性较高，后期的维护也非常的方便。

SSM使用SpringMvc来负责请求转发，客户端通过浏览器发送请求，由分发器(DispatcherServlet)统一拦截，拦截之后分发器查询处理器映射器,找到处理请求的处理器(Controller),处理器调用Service处理进行业务逻辑处理，处理完之后返回模型视图ModelAndView。分发器(DispatcherServlet)查询视图解析器，找到ModelAndView指定的页面，将页面返回给客户端。架构图如图2-2所示：

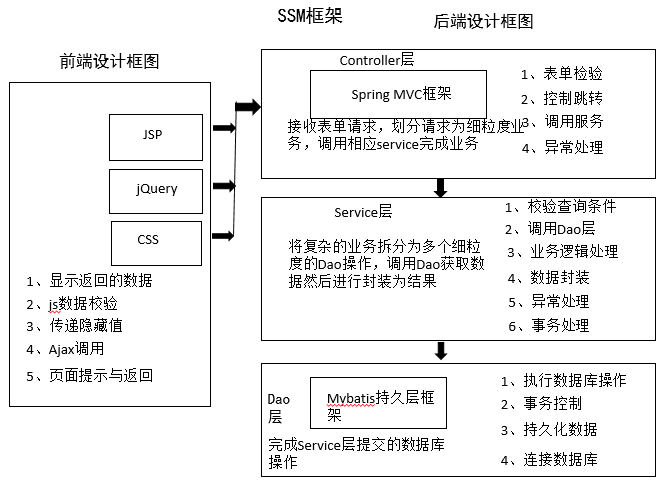


图2-2 SSM框架架构图

SSM有这些优点的同时，也存在着一定的难度，其SQL写在xml。便于统一管理的优化，提供映射的标签。但是由于编写的SQL的工作量大，除此之外mybatis的可移植性不好，不支持标语表之间的级联，在功能上可能达不到我们要求，因此我否定了这个方案。

### 2.3.3 方案三

SpringBoot是2013年提出的全新项目，其设计的目是为了简化Spring框架的搭建、配置、开发、调试和最后部署到生产环境上进行生产。配置只需要在库包管理中添加少量的配置信息即可，使得开发者从Spring框架的xml配置文件中解放出来，快速的搭建开发架构。目前SpringBoot几乎集成了大部分的主流框架，例如：日志框架，持久化框架，JMS框架,缓存框架等。此外SpringBoot的一大优点就是很大程度上简化了Spring各个应用方面的配置，还解决了项目之间包的版本依赖和稳定性问题。

SpringBoot有很多特点：任意的选配、组装、生成一个初始化的项目架构，其内部配置了tomcat服务器和jetty服务器，除此之外还可以将项目打包成jar包的形式，这是为Docker服务器准备的。内部的健康状态监测，外部配置，指标，安全的非业务的功能也是其一大特点。

各方案的优缺点比较如下表所示：

表2-1 各方案比较

|  |  |  |
| --- | --- | --- |
| 方案 优缺点 | 优点 | 缺点 |
| 方案一 | 架构简单，层次分明，开发实现起来比较容易 | 框架安全性能不是太好，会很容易受到网络的攻击 |
| 方案二 | 企业开发常用框架，团队同步进行开发，开发周期缩短，安全性能比较高 | 配置文件过于繁杂，不必要的xml配置文件占据很大的时间 |
| 方案三 | 集成常用的开发框架，项目的搭建，部署、测试简单，配置文件简单，统一编写，开箱即用 | 框架的搭建，系统的集成比较复杂，复杂性较高 |

基于这些优势，紧跟技术的发展，简化开发的周期，我最终选择了SpringBoot作为平台的开发架构。

# 第3章 系统平台框架及软件

根据对系统平台进行需求分析，进行了初步的概要设计，系统平台主体功能主要分为系统平台主界面的展示和系统平台功能管理配置两个部分，然后在这两个主体功能中，将它们进行细分，拆分为一个个小功能模块分别进行独立的开发，然后进行集成呈现为一个完整的页面。

为了能够实现概要设计中对系统平台的功能划分要求，以及页面的呈现，本毕业设计采用前后端分离的技术对系统平台进行架构。将整个系统平台的设计划分成两个部分，前端系统平台展示部分和后端逻辑功能实现部分。前端的系统平台展示部分采用vue技术进行开发，后台采用SpringBoot进行开发。

## 3.1 整体架构

### 3.1.1 物理模型

物理架构从理论上对整个系统平台进行架构描述，对系统平台功能进行分析。从理论上假设能够实现，平台物理模型如图3-1所示：

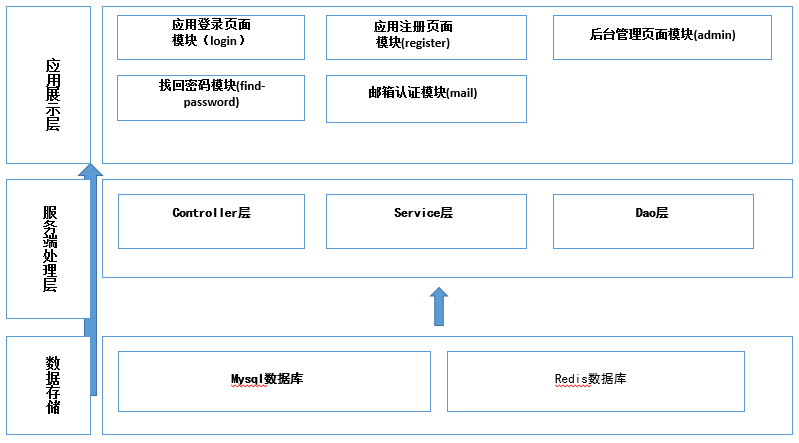


图3-1 物理模型架构

### 3.1.2 逻辑模型

逻辑架构从实现功能的角度上进行阐述，将所需要的框架进行整合，从数据持久层到控制层的数据请求响应，每一层进行分离，各层之间的调用不受到其他层影响，逐步实现内部之间的逻辑功能，实现功能确保系统功能的稳定性和健壮性。架构图如图3-2所示：

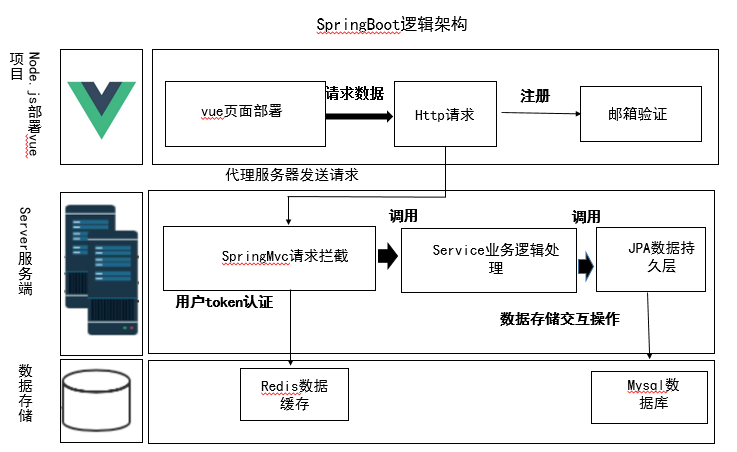


图 3-2 系统逻辑架构图

## 3.2 前端框架vue

前端的设计为了能够适应未来业务和功能的扩展，采用vue作为整体架构进行设计，这样可以达到一次编写重复使用。最主要的是，vue是数据驱动形式的，响应式的数据绑定，只要数据发生改变，页面的渲染就会立刻发生改变，相比以往的页面渲染有很大的差别。vue生命周期图如图3-3下所示：

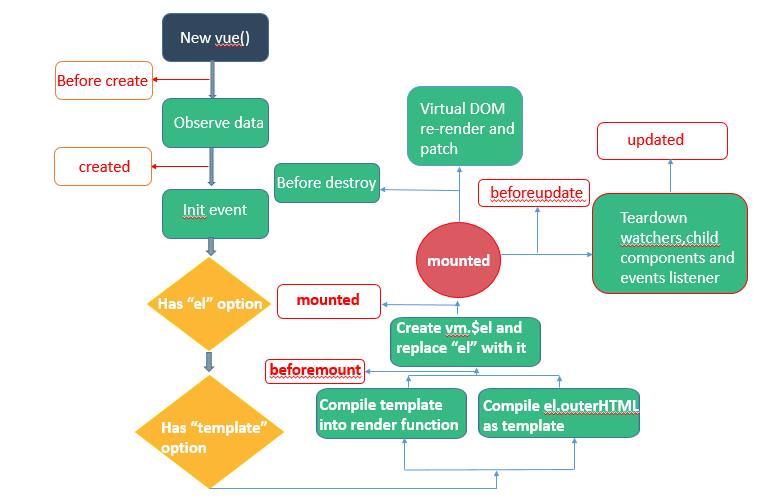


图 3-3 vue生命周期图

### 3.2.1 vue初始化

本设计中我们使用vue作为前端页面的设计框架，得到初始化的vue项目，需要安装一系列的软件，成功之后使用vue init webpack指令新建一个vue的项目，然后进入初始化的vue项目中使用npm install 安装项目所有的依赖，当所有的依赖文件安装完毕之后，控制台提示没有报错的情况下，我们就可以通过指令npm run dev 启动项目，成功启动的初始化项目如图3-4所示：

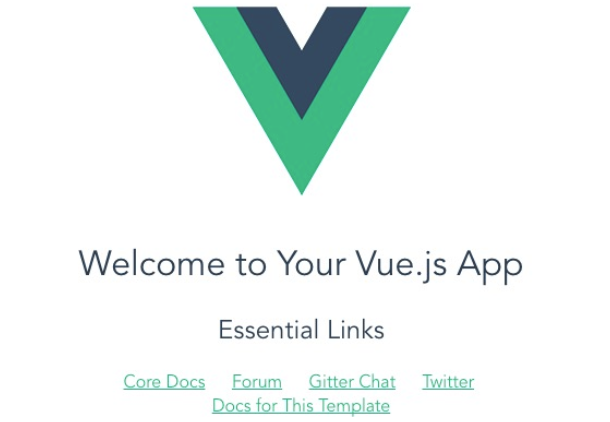


图 3-4 vue初始化界面

成功启动页面说明项目搭建没有错误，之后就可以在这个项目的基础上进行设计开发。开发工作一般都放在项目的src目录下，在src目录下开发页面，添加图片，设计样式，开发组件。下图是vue项目的结构目录如图3-5所示：



图3-5 vue项目目录图

### 3.2.2 组件componets

组件化开发，简单来说就是将页面分成不同的小模块，每个模块分别实现不同的功能，相互之间逻辑独立。同时相互之间又根据接口的规则进行互相融合，然后根据实现不同的功能将其进行组合，最后形成一个完整的应用。例如：导航栏、列表、弹窗、下拉菜单等。这些在开发的工程中就可以直接设计成一个个的组件分别进行开发，在将其进行自由的组合，当不需要某个组件时或者想要替换某个组件时，可以随时进行替换和删除，而不影响整个应用的运行。

设计中我们也采用这样的形式来进行开发，将一些通用的功能模块设计成组件，如应用的顶部菜单栏(AppHeader)，侧部导航栏(AppSideBar)，底部展现栏(AppFooter)，后台管理顶部菜单栏(AdminHeader)，后台管理侧部导航栏(AdminSideBar)。

### 3.2.3 node框架

使用vue框架进行页面的开发node是必不可少的。因为node是一个运行在服务端的javascript运行环境，可以帮助我们快速的搭建开发环境，而且可以建立响应速度快，便于扩展的大型应用。其次node是基于事件驱动I/O型的前端运行环境，这样可以在分布式的设备上进行一些大量实时数据的交互。

本设计中对前端和后端进行了分离，将前端的设计通过node环境进行运行，vue初始化项目的启动就是在node进行的。不需要有后端的协作，边开发边调试，很大程度的提高了开发的进度。

### 3.2.4 nuxt框架

vue框架作为前端设计三大开发框架之一，衍生出许多的基础开发框架，nuxt框架就是其中之一。作为一个通用型的前端应用框架，其目标在于创建一个灵活的前端应用框架，对客户端和服务端的框架进行抽象化，主要关注的重点是对应用界面的UI图形渲染。它提供了各种配置，为vue.js开发服务端渲染提供了便利。

设计中集成nuxt框架目的在于，nuxt可以作为代理服务器，本地开发的前端应用可以通过代理的机制进行跨域请求。只需在nuxt.config.js中设置proxy即可，如图3-6配置所示：

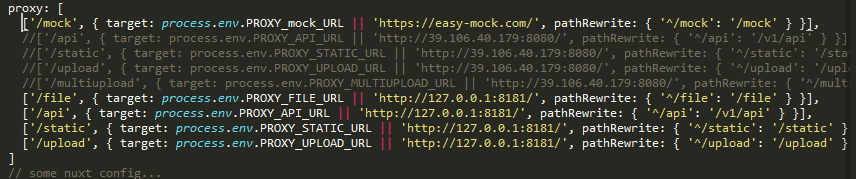


图 3-6 nuxt代理配置

## 3.3 后端框架

系统平台的服务端采用RESTFUL API的形式向外部暴露接口。前端只需要调用相应的服务就可以请求到所需要的数据，开发中我们使用SpringBoot集成的一系列基础框架进行搭建，与其他的架构相比，SpringBoot极大地加快了我们开发效率。其架构图如图3-7所示：

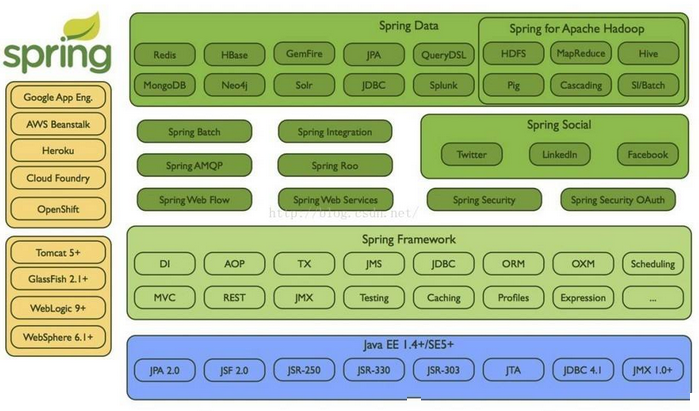


图 3-7 spring系统架构图

### 3.3.1 SpringMvc

SpringMvc目前已经替代struts成为主流的控制器，目前大型的企业都摒弃了struts，普遍采用SpringMvc，在后端的设计中它负责请求转发。客户端通过浏览器发送请求，内部的分发器(DispatcherServlet)统一拦截，拦截之后分发器查询处理器映射器,找到处理请求的处理器(Controller)，处理器调用Service进行业务逻辑处理，处理完之后返回模型视图ModelAndView。分发器(DispatcherServlet)查询视图解析器，找到ModelAndView指定的页面，将页面返回给客户端。

### 3.3.2 Swagger2

在后端的框架中，集成了Swagger2框架的目的是为了方便以后对系统平台进行扩展和维护。通过Swageer2我们可以在看不到后端源代码的条件下，自动的生成系统平台的API在线文档，API文档中详细记录了系统平台提供的所有功能，前端程序只需要调用相应的API即可。

在本次的设计中，我们加入swagger的目的主要是为了检测应用中的功能是不是正常的启动，如果没有启动起来的话，我们在API的文档中是没有办法看到接口的，这样我们就可以很好的维护我们的应用程序。

### 3.3.3 JWT token

JWT全称 Java Web Token。作为前后端数据传输时的一个标识，客户端在第一次访问系统平台的时候，为用户生成一个唯一的签名：token返回给客户端，客户端在收到token之后进行保存，在以后的每次请求中都带上token，后端在接到请求之后，解析token，从中取到用户的信息进行验证，通过之后才能进行后续的操作，否则请求无法通过。

设计中使用JWT token进行安全认证，防止其他用户违法操作，将恶意的数据上传到系统平台，从一定程度来说，采用这样的方式，提高了系统平台的安全性。

### 3.3.4 JPA数据库操作

设计中使用JPA操作数据库，减少了大量的工作量。对于数据库的操作，不像其他框架还要去实现对应的接口函数，JPA只需继承相应的接口，定义操作数据库的函数即可，不需要编写对应的实现方法，就可以达到操作数据库，框架内部会自动去实现接口函数，许多开发者都选择其作为框架。操作如图3-8所示：

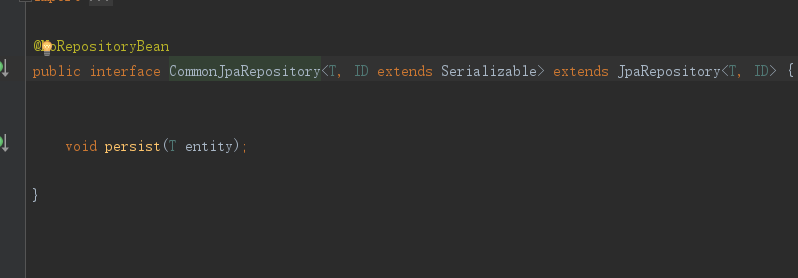


图 3-8 JPA操作数据库接口

## 3.4 开发软件

### 3.4.1 IEDA开发软件

IEDA全称IntelliJ IEDA，是java语言开发的集成环境。其在智能代码助手，代码自动提示，重构，J2EE支持以及版本管理工具都是超常的。不仅如此，在IEDA中还集成了许多的开发框架，我们只需要简单的几步就可以搭建一个基于这些框架项目。如今，许多的开发人员都不在使用eclipse进行开发，而是使用IEDA开发项目，因为其简便的开发模式，快速的构建，堪称开发人员的青睐。

基于许多的优势与一身，使得我们选择IEDA作为我们的开发工具，当然也有其他的一些开发工具，可以进行选择，软件视图如图3-9所示：

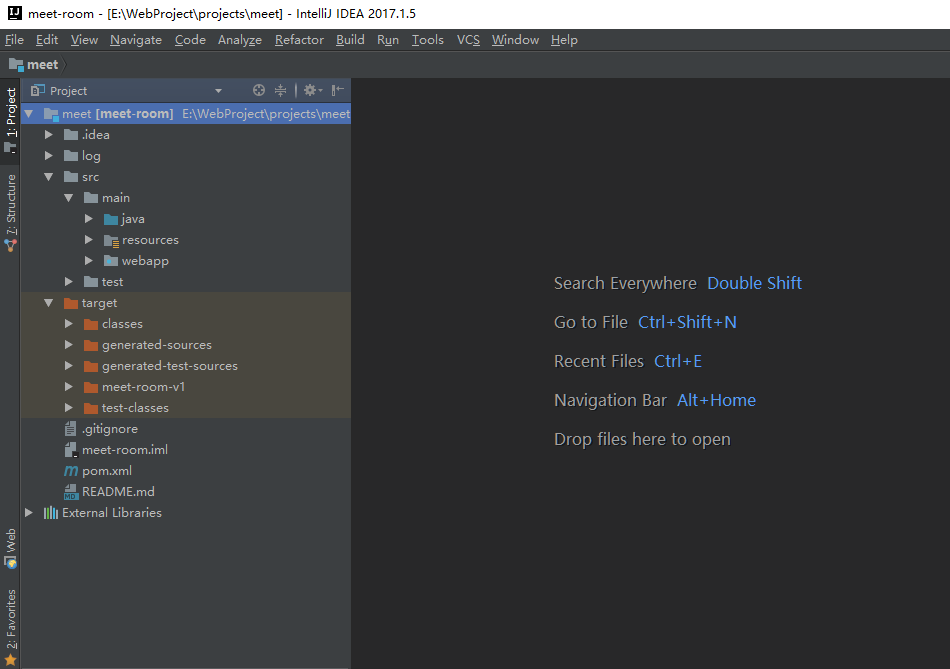


图 3-9 IEDA视图界面

### 3.4.2 SQLyog

SQLyog，一个数据库客户端软件。启动mysql服务，就可以使用其进行连接。SQLyog中我们可以设计数据库的表，查询数据库中的数据，最重要的一个功能，SQLyog可以在线将数据从一个数据库复制到另一个数据库。同时连接本地数据库，和其他的网络数据库。这是它与其他数据库客户端软件最大的区别，也是最大的优势，界面如图3-10所示：

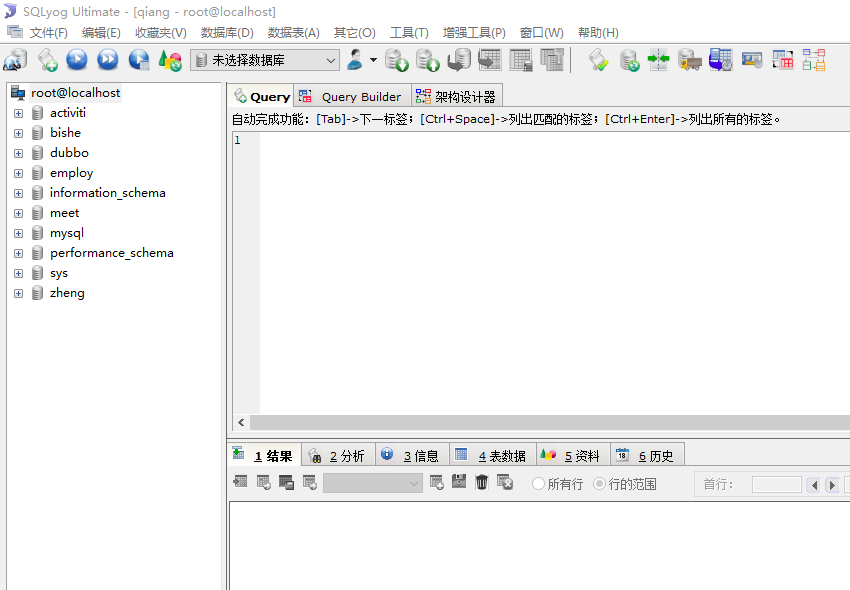


图 3-10 SQLyog界面视图

### 3.4.3 redis Server

redis数据库，作为一个Nosql技术，也有服务端，它的数据存储和关系型数据库存储有很大的不同，redis数据库将数据存储在内存当中，服务停止下次启动时，数据会自动加载，不会丢失，相比关系型数据库更加能保证数据的安全

设计中使用redis数据库存储用户的token,实时更新数据，响应的速度快。很大程度上提高系统平台的性能。

### 3.4.4 Sublime

Sublime文本编辑工具，可以编写几乎所有的编程语言，功能非常的强大。如果要编写某一门语言的代码，仅需在软件的有下角的功能切换去进行切换即可。设计中我们使用Sublime对我们的前端的设计进行开发，虽然Sublime中没有集成调试的工具，弄起来稍微复杂一点，我们需要在node中启动应用进行查看，然后根据页面的显示进行调试。然后在Sublime中进行修改，保存之后可以实时的更新页面，开发界面如图3-11所示：

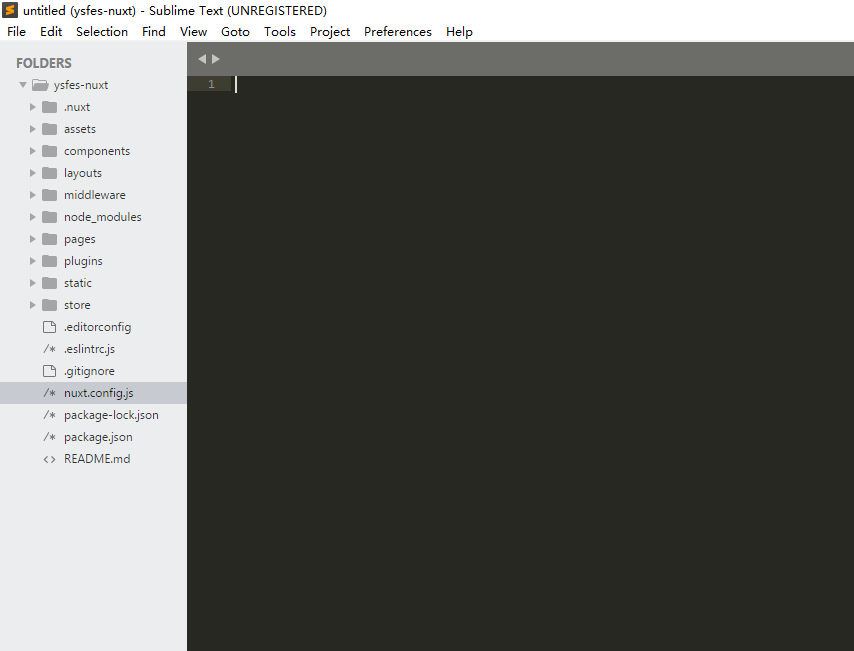


图 3-11 sublime界面视图

# 第4章 系统设计与实现

通过对系统平台框架的介绍之后，本章对系统平台进行设计开发。根据对框架的介绍搭建开发环境，分别搭建前端开发环境和后端开发环境，进行系统平台的开发。

## 4.1 前端项目开发

前端项目的开发中，根据设计的要求，我们将页面划分成不同的层次：组件开发、页面设计，应用展示。

### 4.1.1 组件开发

设计中我们采用组件化的形式进行开发，将一些通用的功能模块设计成组件，如应用的顶部菜单栏(AppHeader)，侧部导航栏(AppSideBar)，底部展现栏(AppFooter)，后台管理顶部菜单栏(AdminHeader)，后台管理侧部导航栏(AdminSideBar)。组件开发视图如图4-1所示：

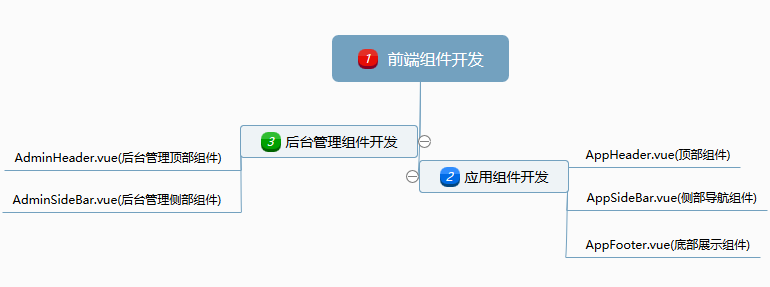


图4-1前端页面组件设计目录

### 4.1.2 页面开发

vue页面的开发有别于传统的页面开发。在vue中将一个复杂的、请求延时较高的应用拆分为粒度合理的模块即单个的组件，在形成一个完整的应用时，再组合各个组件，每个模组件独立完成相应的功能，传统的页面设计针对每一个应用都要进行页面的设计，应用需要更换时，就得从头开始就行设计。vue很大程度上解决了这个问题，不仅提高了效率，而且可以一次编写，重复使用；不仅如此，使用vue还提升了整个项目的可维护性，在大型的开发中便于团队之间的协同开发。

设计中我们按照vue的核心思想，针对每一个功能就设计为一个组件，将页面看成是一个容器，在容器中添加不同的组3件，就可以形成不同的页面，完成不同的功能，可以根据需要随时的添加删除任意的组件，不会影响整个应用的运行。

页面设计目录如图4-2所示：

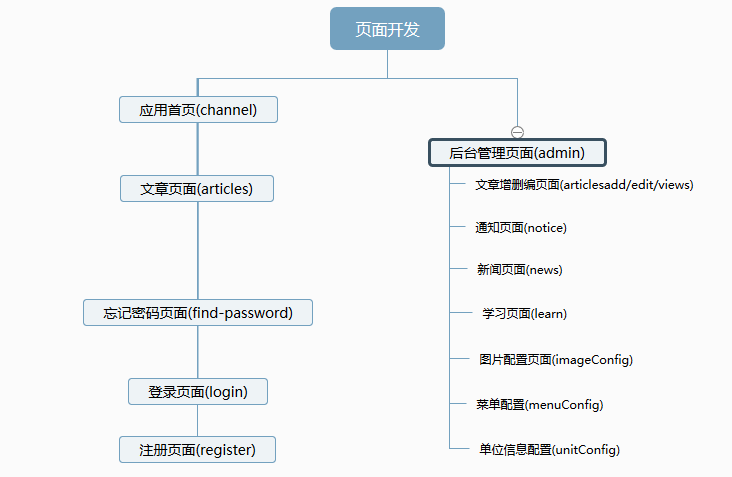


图4-2 页面开发设计图

## 4.2 数据库表设计

根据页面需要展示的数据，我们抽象出实体类，然后根据实体类的属性，我们采用框架JPA的注解形式自动的生成数据库的表，在redis数据库中，因为它是存储在内存中的，所以我们不需要生成任何的表。直接以key-value的形式将数据存储在内存中即可。

### 4.2.1 mysql数据库设计

在mysql数据中我们创建的数据库名为meet，根据后端的实体类映射，在数据库中生成相应的表进行存储，生成的表视图如图4-3所示：

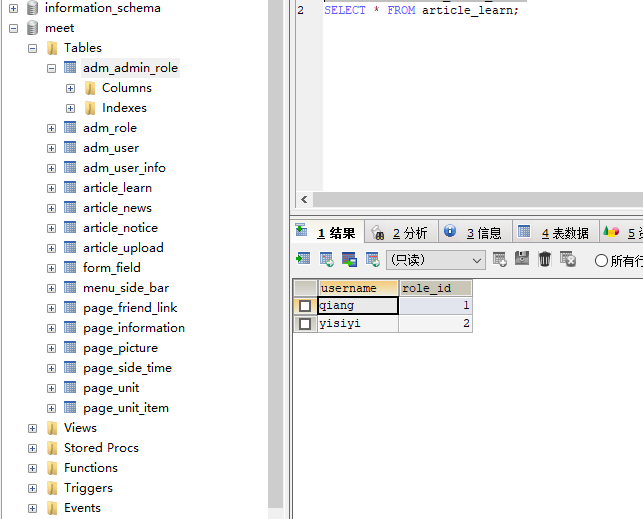


图4-3 mysql数据库表展示图

### 4.2.2 redis数据库设计

redis数据库我们不需要进行表设计，redis很好的为我们集成了各种字符类型，如：字符串(String)，哈希(hash)，列表(list)，集合(set)，有序集合(zset)。redis数据库中存储key-value的值，需要时我们通过key就可以取到存储在redis数据库中的值。这样的存储方式，会大大的提升读取写性能。不仅如此，redis数据库支持主从复制，主机自动将数据同步到从机，可以进行数据库的读写分离。redis数据库操作如图4-4，4-5所示：

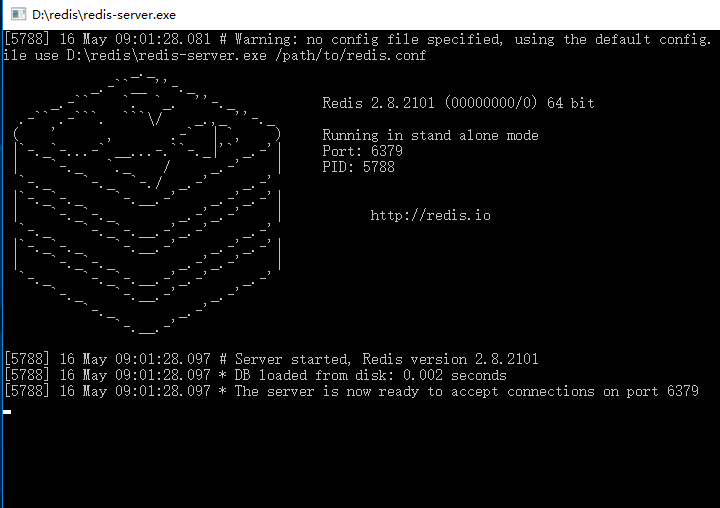


图4-4 redis数据库服务端界面

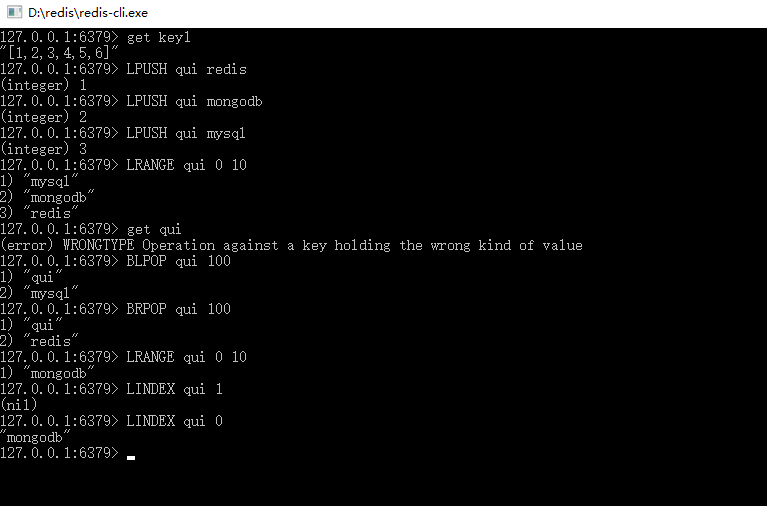


图4-5 redis数据库客户端连接操作

## 4.3数据存储

我们都知道应用程序只要部署上生产线之后就会产生非常大量的数据，而这些数据都必须要进行保存，用户的基本信息，用户的日常的浏览，用户的订单或者支付的情况等。每天都会有海量的数据产生，而且这些海量的数据必须一条不漏的保存下来，保存这些海量的数据也是一件不容易的事情，存储不得当的话就会导致数据缺失，甚至数据的损坏，因此在设计的时候对数据的存储也要非常的谨慎。

### 4.3.1 mysql数据库存储

在我们设计中，为了使用的简便，数据便于存储和适应大部分的数据存储，我们使用mysql数据库来存储我们的数据，用户的基本信息，以及用户上传的文章，系统公告，发布的一些学习资料之类的都保存在数据库。当然有的文章特别的长，数据量特别的大，如果用一个字段进行存储的话，会造成时间的相应非常的长，为了对其进行改进，我们可以对数据的存储进行优化，在数据库中保存文章的路径，当我们需要的时候直接通过路径去加载应用程序某个目下的内容就可以，这样也是解决大文件存储的一种方式。

为了加快相应的速度，我们只将应用程序部分数据存储在了mysql数据库，而将一部分数据存储在Redis数据库。

### 4.3.2 redis数据库

设计中我们没有采用redis的数据库存储，用它来缓存我们的token。这样的用户在每次登录验证成功之后都会生成一个token,然后就将其存在redis中,并设置过期时间，每当用户验证没有通过或者用户登录时间过期，那么token就失效，需要重新登录才能进行下一步的操作，不然就无法进行操作。

## 4.4 后台功能开发

后台功能开发是实现整个系统平台的所有功能：文件上传、安全认证、文章存储、图片保存、数据存储。为了方便以后进行功能的迭代，我们采用全新的开发模式SpringBoot架构，它颠覆了web应用开发，不需要过多的配置文件，如xml文件。就像maven中所说的“约定大于配置”，SpringBoot为我们省去了繁杂的配置，用户需要的个性配置采用 properties 文件或 yaml 文件的格式写在统一的文件中。其他在开发中的配置信息要么采用 Annotation 要么采用配置类形式。

### 4.4.1 安全认证

系统平台的安全认证中，采用JWT Token的方式来验证客户端数据传输的安全性。客户端在第一次通过用户名和密码登录系统平台的时候，后台生成一个token返回给客户端，客户端在接收到token之后将其进行保存，以后每次和后台进行通信的时候，都会带上这个token，后台在处理数据之前都会首先解析这个token，拿到认证通过的用户的信息，和redis数据库中的值进行比对，认证通过，那么后端的程序才会进行数据的操作，否则用户的本次操作视作违法的，不能进行后续的操作。后台认证逻辑实现如图4-6所示：

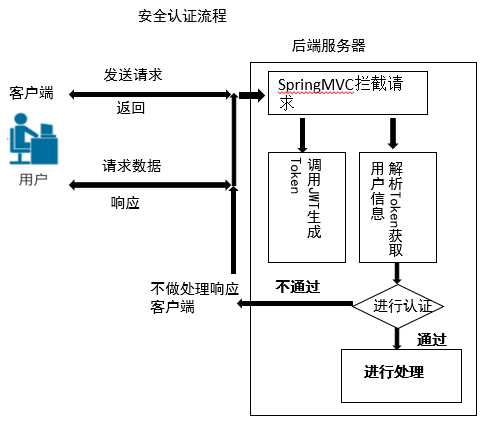


图 4-6 安全认证流程图

### 4.4.2 文件上传

文件上传部分，我们采用了Apache推出的架构commons-pool2,commons-lang3框架，它们适应大多数的应用进行文章的上传，而且能够很好的和其他的框架进行整合，不会因为版本的不兼容，而导致应用无法正常运行，在目前的流行开发框架中很受欢迎。文件上传配置如图4-7所示：

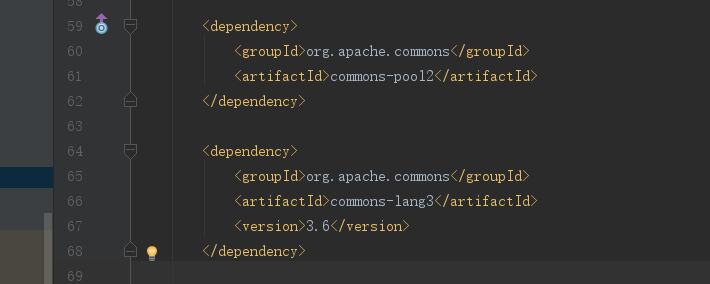


图 4-7文件上传配置

### 4.4.3 文章存储

文章存储，我们直接将文章的所有的内容存储在mysql数据中。文章编辑可以通过内嵌的编辑器进行编辑，它实现和word大部分的功能，可以编写出格式，要求规范的文章供别人浏览。不仅如此，一些精美的论文，我们也可以通过此途径放到我们的系统平台上发表出去。文章存储、编写如图4-8，4-9所示：

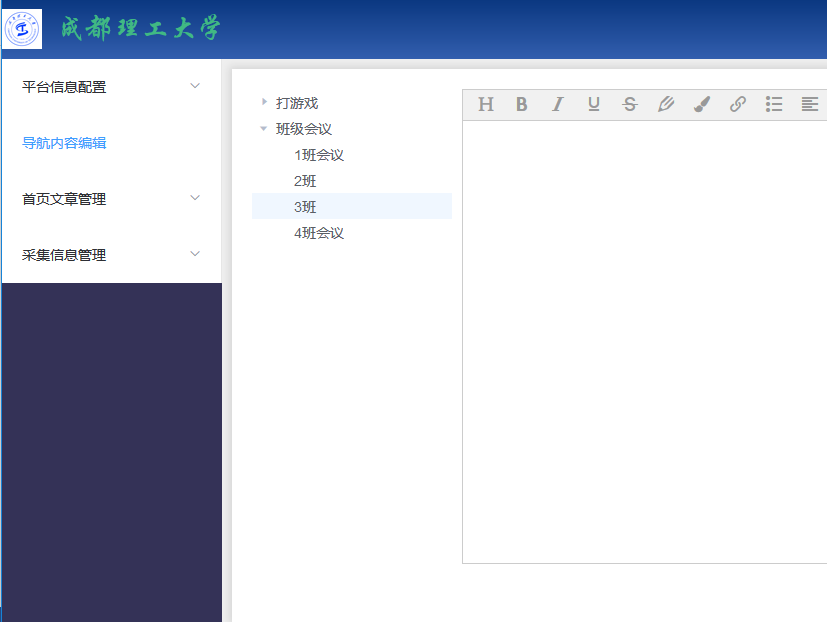


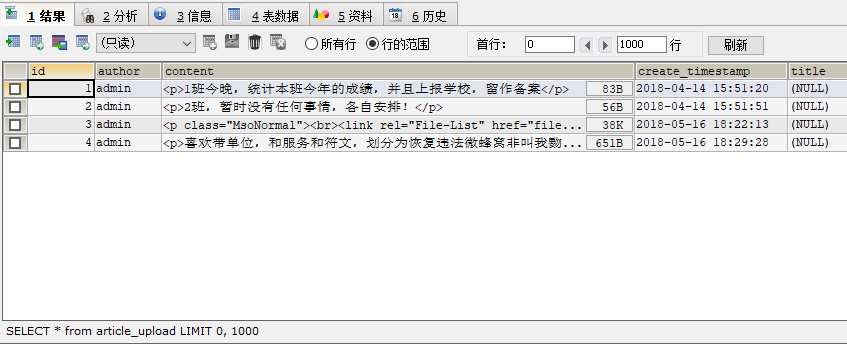
图 4-8 文章在线编辑

图 4-9 文章存储视图

### 4.4.4 图片保存

图片是一个很大的文件，在应用程序中，我们一般在数据库中只存储图片的路径，然后通过I/O流的形式将图片数据保存在系统平台的某个路径下，当我们需要访问某张图片的时候，直接通过数据库中存储的路径。发送请求就可以获取到图片的数据。图片的上传、保存如图4-10，4-11所示：



图 4-10 图片上传

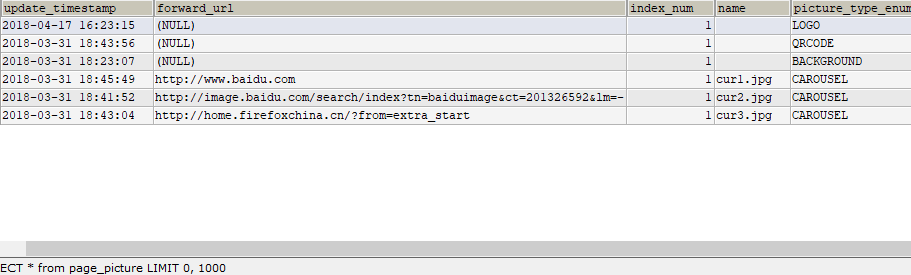


图 4-11 后台路径存储

# 第5章 系统部署与测试

## 5.1 系统平台部署

对于大多数开发者来说，在应用程序开发完毕之后都需要部署在服务器上，才能被别人访问。首先想到的就是选用tomcat作为应用的服务器，众所周知，tomcat是一个免费开发源代码的web应用服务器，而且属于轻量级的应用服务器，在中小型的企业和用户访问量不是很多的场景下被普遍的采用。

### 5.1.1 后端部署

在设计中虽然SpringBoot中内嵌了tomcat服务器，执行mian方法就可以启动服务，以此来达到简单的部署，但我们放弃了这样的部署方式，采用传统的打包部署，将应用打包成一个war包的形式，然后将其添加到tomcat容器，在启动tomcat。系统平台部署配置如图5-1所示：

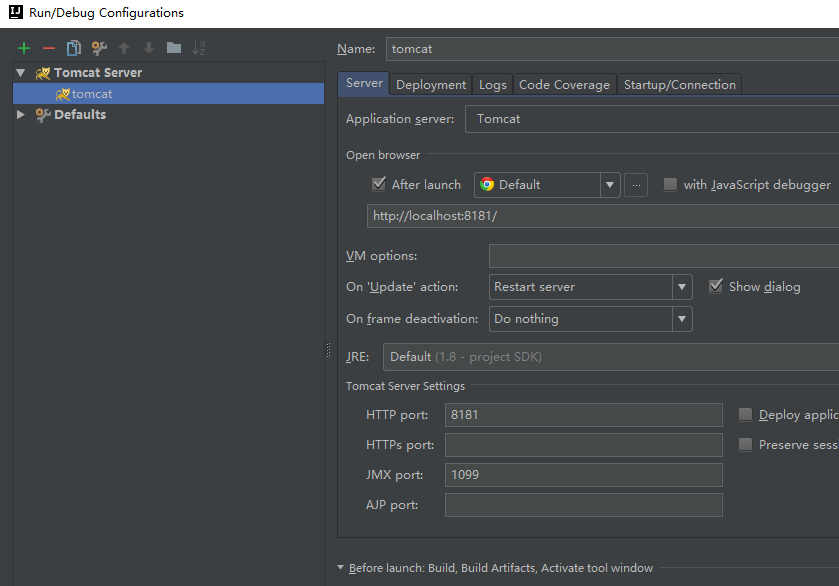


图 5-1 系统部署配置

### 5.1.2 前端配置

由于系统平台是基于前后端分离的技术进行设计的，在进行部署的时候，我们需要对前后端的项目进行单独的部署。因为前后端分离，在发起请求的时候，可能会被拦截，所以我们需要设置一个代理服务器，这样才能够实现跨域发送请求，不会被拦截。前端配置如图5-2所示：

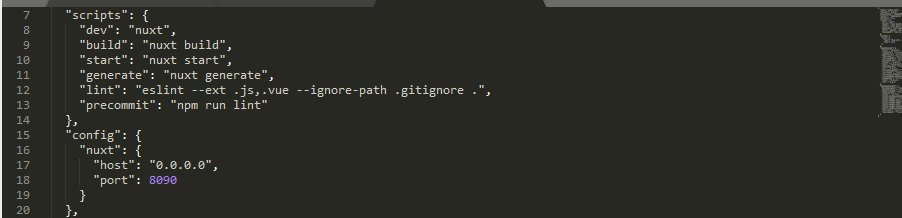


图 5-2 系统前端IP配置

## 5.2 平台测试

平台在部署到服务器之后，为了保证应用能够按照我们预想的那样执行，需要对其进行测试，这是必不可免的。通过测试我们可以发现系统的问题和缺陷，然后不断的对其进行改进，对功能进行不断的优化，达到需求预期的效果。

### 5.2.1 服务端测试

系统的测试有很多种方法，通过专业的工具可以更精确的知道系统问题的所在，如：黑盒测试，白盒测试，单元的是，集成测试，系统测试……。设计中，我们直接通过对平台的使用来进行测试。首先我们测试的是服务端提供的接口是否正常的启动，访问地址<http://localhost:8181/swagger-ui.html>查看是否所有的接口正常其，如图5-3所示：



图 5-3 服务端接口示图

### 5.2.2 前端测试

对于前端的测试，主要检测的是开发的页面是否和后端的功能处理能够衔接的上。对于前端的程序可以在线进行调试，出现问题可以及时的进行调整，页面可以直接运行在浏览器中，不像后端程序需要部署在应用服务器中才能进行访问测试。前端程序部署测试如图5-4所示：

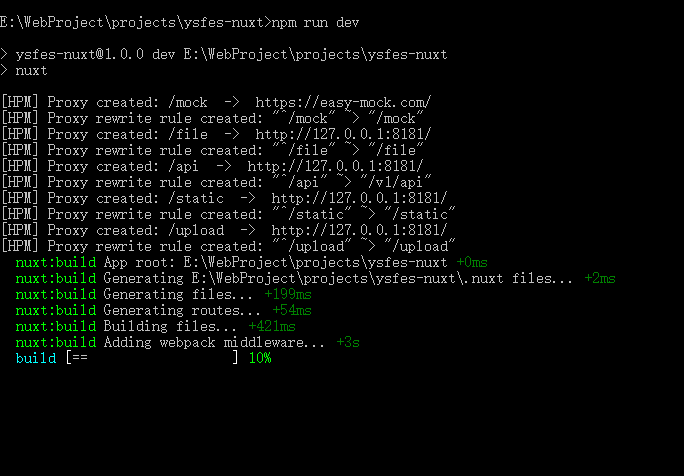


图 5-4 前端程序测试

# 结论

本文的毕业设计，采用软件开发领域最前沿的技术和框架，数据存储全新的技术。本着对新技术的学习，提升自己动手的能力以及对于网络的热爱，最终克服许许多多的困难，设计出了基于SpringBoot的会议系统平台，达到了毕业设计的预期目标。

本设计采用一种全新的设计理念，前后端分离的形式。前端采用三大流行框架的vue进行设计，将相同部分的页面设计为一个个小的组件，这样可以达到一次编写，多次使用的目的。不仅如此vue强大的前端页面开发框架集成了路由功能，使得页面的跳转不需要通过服务端程序的处理，直接通过前端就可以跳转到其他的页面，很大程度上减轻了服务端程序的压力。其次，vue中还集成了http协议，这样我们不仅不用添加额外的插件就可以发送http请求，请求服务端的数据，而且不用因为插件的不兼容，而导致应用不能正常的运行，很大程度上解决了我们开发遇到的烦恼。

服务端程序的设计，采用最前沿的技术SpringBoot。里面集成了许多在开发中经常用到框架，只需要稍微的配置就可以轻松的使用。我们不用去担心系统中所要使用的某个功能的框架是否与整个系统相兼容。使用SpringBoot作为整个系统平台的开发框架，为我们省去了以往许多繁重的配置，我们只需要关注逻辑部分的程序编写，实现预定功能。对于所要使用的到的其他框架，不像以前的开发框架那样，需要编写一个xml文件将其引入到我们的平台中，而是采用“约定大于配置”的思想，使用Anntation的形式或者java Bean的形式将其引入即可。

本设计完成了预定的目标，在有限的时间内是不能够对整个系统的功能进行完善的，初步实现了文章的推送，图片的上传，通告的发布，文章的下载。由于时间短暂，没有将所有的功能都开发出完成，只能进行一个小小的演示。在本次毕业设计中感觉到了许多的不足，遇到了许多的问题，在以后的工作学习中，我将不断地完善此设计，做出一个完整的会议系统平台。

# 致谢

感谢大学生涯学校四年来对我的培养、关心与支持！

感谢通信工程系的所有老师，正是他们的谆谆教诲让我学会了很多有趣的知识。首先感谢赵义红老师，每一次遇到困难时，赵老师的指导使我瞬间豁然开朗，在赵老师的细心指导和关怀之下， 我的论文得以顺利地完成， 老师渊博的专业  
知识，严谨的学术作风，精益求精的科学态度给我指明了方向，使我受益匪浅。尤其是老师刻苦的学习精神和对工作忘我的投入和高瞻远瞩，是我永远的学习榜样。老师在学业上的对我的谆谆教导是我永远的精神财富。论文完成之际，向尊敬的赵老师致以最衷心的感谢。

感谢陪我度过了大学四年美好时光的所有同学，正是因为有了他们，我们的大学时光过得那样的充实而又快乐，我们一起参加学校组织的，校级的，省级的，国级的比赛，做各种的电子作品，设计自己喜欢的东西。在这里我结交了一群非常非常好的朋友，遇到问题我们一起讨论，一起想解决的办法，从中不仅学到知识，而且还收获了看到结果的那一刻的喜悦！

感谢本文所列参考文献的所有作者们！

感谢所有关心、支持和帮助过的其他朋友！

# 参考文献

陈韶键著.2016.深入实践Spring Boot.北京:机械工业出版社.

董健康,王洪波,李阳阳.2014.ＩａａＳ环境下改进能源效率和网络性能的虚拟机放置方法 [J]. 通信学报.35（1）.72-81.

葛宝龙,邓欣,秦嘉罗.2017.SpringBoot集成Redis缓存在电信运营系统中的应用研究.中国联合网络通信有限公司哈尔滨软件研究院.

王永和,张劲松,邓安明等.2016.Spring Boot 研究和应用[J].信息通信.(10)：91-94

汪云飞.2016.JavaEE 开发的颠覆者 Spring Boot 实战[Z].

王福强著.2016.SprongBoot揭秘快速构建微服务体系.北京:机械工业出版社.

Craig Walls著. 2016.SpringBoot 实战(美).北京:人民邮电出版社.

温晓丽，苏浩伟，陈欢.邹大毕.2017.基于SpringBoot微服务架构的城市一卡通手机充值支撑系统研究广州羊城通有限公司（广东 广州 510080）

张峰.2017.应用SpringBoot该Web应用开发模式[A].23.

Craig Walls.2016.Spring Boot in Action[M].Manning publications.［2012-09 06］

G. Sibiy, H. Venter, T. Fogwill .2012. Digital forensic framework for a cloud environment.Proceedings of the 2012 Africa Conference*.* 1-8

Lucas Krause,Microservices.2016.Theory and Application[J].

Applicative.2017.SpringBoot Reference Guide1.5.3.RELEASE[OL].

R. Marty .2011. Cloud Application Logging for Forensics. Proceedings of the 2011 ACM Symposium on Applied Computing*.* 178-184.

S. Zawoad, A.K. Dutta and R. Hasan .2013. SecLaaS; Secure Logging-as-a-Service for Cloud Forensics. in *ACM Symposium on Information. Computer and Communications Security*.DOI: 10.1145/2484313.2484342, 219-230.