

文档编号	SC20191215
版本号	V1.0.0

四川科技职业学院学生工作室宣传官网设计 毕业设计论文（一稿）

2019-10-09

审签记录

版本号	修订人	修订内容	校队	审核	批准	签批日期
1.0.0	周巍	编写主体内容	周巍	周巍	周巍	2019.11.28



四川科技职业学院
UNIVERSITY FOR SCIENCE & TECHNOLOGY SICHUAN

毕业设计（论文）

题目：四川科技职业学院学生工作室宣传官网设计

院 系：互联网+学院

专 业：软件技术

年 级：2017 级

班级	学号	姓名
软件项目 17-1	201943040344	周 巍

二〇一九年十二月十五日

摘 要

在逐渐信息化的现代社会，人们早就习惯于在网络是获取自己所需要的信息，我们的想法来自于各种各样的工作室在校园内扩大自身知名度的宣传活动，工作室下了很大功夫，在校园内不管是宣传活动还是传单或是讲座，都不甚理想，因为很多的同学不喜欢参加这类活动，自然宣传不到他们的耳朵里，就算听到过，那也是左耳进右耳出，并不会留下深刻的印象，而如果我们将工作室的宣传工作以一个有趣的方式集成展现在一个官网中，那么应该是能给同学们留下印象的，无论是宣传力度还是宣传成本都会降低而效果却一定会上升，所以这样的一个有价值的课题值得我们去研究。

关键字： 工作室、 跨域、 前后段分离、 SpringBoot、 数据库

abstract

In the modern society of gradual informationization, people have long been used to obtaining the information they need on the Internet. Our ideas come from the propaganda activities of various studios to expand their popularity on campus. The studios have made great efforts. No matter in propaganda activities, leaflets or lectures on campus, they are not ideal, because many students do not like to participate in this activity. The natural publicity of such activities can't reach their ears. Even if we have heard them, it's from left ear to right ear, and it won't leave a deep impression. If we integrate the publicity work of the studio into an official website in an interesting way, it should impress the students. Both the publicity intensity and the publicity cost will be reduced, but the effect will be certain. It will rise, so such a valuable topic is worth studying.

Keywords: studio, cross domain, front and back segment separation, SpringBoot, database

目 录

摘 要.....	4
第一章 绪论.....	12
1.1 本论文课题来源.....	12
1.2 本论文研究目的背景及意义.....	12
1.2.1 研究目的及背景.....	12
1.2.2 研究意义.....	12
1.2.3 研究背景.....	13
1.3 国内外发展状态，发展水平存在的问题.....	13
1.3.1 国外研究现状.....	13
1.3.2 国内研究现状.....	13
1.4 本论文研究目标和主要内容.....	13
1.4.1 研究目标.....	13
1.4.2 主要内容.....	14
1.4.3 拟解决的关键问题.....	14
1.5 研究的方法.....	15
1.5.1 本论文研究方法.....	15
第二章 项目概述.....	15
2.1 项目术语解释.....	15
2.2 项目概述.....	15
2.3 项目需求定义.....	16
2.4 项目系统架构设计.....	16
2.4.1 系统功能架构设计.....	16
2.4.2 系统技术框架设计.....	17
第三章 前端功能详细设计.....	17
3.1 前端界面功能架构设计.....	17
3.1.1 前端功能框架设计.....	17
3.1.2 前端技术框架设计.....	18
3.2 学生工作室首页.....	19
3.2.1 功能描述.....	19
3.2.2 原型 UI 设计.....	20

3.2.3 业务流程.....	20
3.3 查询更多学生工作室页面.....	21
3.3.1 功能描述.....	21
3.3.2 原型 UI 设计.....	21
3.3.3 业务流程.....	21
3.4 工作室详情页面.....	21
3.4.1 功能描述.....	21
3.4.2 原型 UI 设计.....	22
3.4.3 业务流程.....	22
3.5 添加上传工作室页面.....	22
3.5.1 功能描述.....	22
3.5.2 原型 UI 设计.....	23
3.5.3 业务流程.....	23
3.6 表单未填写页面.....	23
3.6.1 功能描述.....	23
3.6.2 原型 UI 设计.....	24
3.6.3 业务流程.....	24
3.7 关于我们页面.....	24
3.7.1 功能描述.....	24
3.7.2 原型 UI 设计.....	25
3.7.3 业务流程.....	26
3.8 管理员登录页面（管理员）.....	26
3.8.1 功能描述.....	26
3.8.2 原型 UI 设计.....	26
3.8.3 业务流程.....	26
3.9 错误提示页面（管理员）.....	27
3.9.1 功能描述.....	27
3.9.2 原型 UI 设计.....	27
3.9.3 业务流程.....	28
3.10 后台管理页面（管理员）.....	28
3.10.1 功能描述.....	28
3.10.2 原型 UI 设计.....	28

3.10.3 业务流程.....	28
3.11 修改工作室（管理员）	29
3.11.1 功能描述.....	29
3.11.2 原型 UI 设计	29
3.11.3 业务流程.....	29
3.12 删除数据页面 （管理员）	29
3.12.1 功能描述.....	29
3.12.2 原型 UI 设计	30
3.12.3 业务流程.....	30
3.13 搜索分页页面 （管理员）	30
3.13.1 功能描述.....	30
3.13.2 原型 UI 设计	31
3.13.3 业务流程.....	31
第四章 服务端 API 详细设计	32
4.1 服务端功能技术框架设计.....	32
4.1.1 服务端功能框架设计	32
4.1.2 服务端技术框架设计	32
4.2 查询所有学生工作室 API 接口设计	33
4.2.1 API 接口功能描述	33
4.2.2 接口原形定义.....	33
4.2.3 接口流程.....	33
4.2.4 请求 URL 数据格式.....	33
4.2.5 接口返回数据格式.....	33
4.3 查询指定 id 的工作室 API 接口设计	34
4.3.1 API 接口功能描述	34
4.3.2 接口原形定义.....	34
4.3.3 接口流程.....	34
4.3.4 请求 URL 数据格式.....	35
4.3.5 接口返回数据格式.....	35
4.4 获取上传操作 token API 接口设计	36
4.4.1 接口功能描述.....	36
4.4.2 接口原形定义.....	36

4.4.3 接口流程.....	36
4.4.4 请求 URL 数据格式.....	36
4.4.5 服务器返回数据格式.....	36
4.5 上传工作室 API 接口设计.....	37
4.5.1 API 接口功能描述.....	37
4.5.2 接口原形定义.....	37
4.5.3 接口流程.....	37
4.5.4 请求 URL 数据格式.....	37
4.5.5 服务器返回数据格式.....	38
4.6 获取上传图片 Token API 接口设计.....	38
4.6.1 API 接口功能描述.....	38
4.6.2 接口原形定义.....	38
4.6.3 接口流程.....	39
4.6.4 请求 URL 数据格式.....	39
4.6.5 服务器返回数据格式.....	39
4.7 获取验证码 API 接口设计.....	39
4.7.1 API 接口功能描述.....	39
4.7.2 接口原形定义.....	39
4.7.3 接口流程.....	40
4.7.4 请求 URL 数据格式.....	40
4.7.5 服务器返回数据格式.....	40
4.8 管理员登录 API 接口设计.....	40
4.8.1 API 接口功能描述.....	40
4.8.2 接口原形定义.....	40
4.8.3 接口流程.....	41
4.8.4 请求 URL 数据格式.....	41
4.8.5 服务器返回数据格式.....	41
4.9 获取管理员 SESSION API 接口设计.....	42
4.9.1 API 接口功能描述.....	42
4.9.2 接口原形定义.....	42
4.9.3 接口流程.....	42
4.9.4 请求 URL 数据格式.....	42

4.9.5 服务器返回数据格式	43
4.10 管理员修改工作室 API 接口设计	43
4.10.1 API 接口功能描述	43
4.10.2 接口原形定义	43
4.10.3 接口流程	43
4.10.4 请求 URL 数据格式	43
4.10.5 服务器返回数据格式	44
4.11 管理员删除工作室 API 接口设计	44
4.11.1 API 接口功能描述	44
4.11.2 接口原形定义	44
4.11.3 接口流程	45
4.11.4 请求 URL 数据格式	45
4.11.5 服务器返回数据格式	45
第五章 数据库设计	46
5.1 数据库模型设计	46
5.2 学生工作室记录表设计	46
5.3 管理员表设计	46
第六章 系统性能优化设计	47
6.1 Web 前端开发技术优化方向	47
6.1.1 减小静态文件的大小	47
6.1.2 减少 HTTP OR HTTPS 请求	48
6.1.3 页面设计优化	48
6.2 后端技术开发优化方向	49
6.2.1 API 接口幂等性	49
6.2.2 前后端业务分离	49
6.2.3 操作数据缓存	50
6.2.4 文件上传优化	50
6.2.5 HTTPS 请求协议	51
第七章 系统安装部署	51
7.1 安装步骤	51
7.1.1 服务器安装环境	51
7.1.2 数据库安装	52

7.1.3 前端页面安装.....	52
7.2 数据配置.....	52
7.2.1 数据库配置	52
7.2.2 后端服务器配置	52
7.2.3 CDN 静态服务器配置.....	52
7.3 系统环境调试.....	52
第八章 系统功能测试报告	53
8.1 前端业务逻辑测试.....	53
8.1.1 页面渲染测试.....	53
8.2 API 接口高并发测试	53
8.3 系统综合测试.....	54
结 语.....	55
致 谢.....	56
参考文献.....	57

第一章 绪论

1.1 本论文课题来源

在逐渐信息化的现代社会，人们早就习惯于在网络是获取自己所需要的信息，我们的想法来自于各种各样的工作室在校园内扩大自身知名度的宣传活动，工作室下了很大功夫，在校园内不管是宣传活动还是传单或是讲座，都不甚理想，因为很多的同学不喜欢参加这类活动，自然宣传不到他们的耳朵里，就算听到过，那也是左耳进右耳出，并不会留下深刻的印象，而如果我们将工作室的宣传工作以一个有趣的方式集成展现在一个官网中，那么应该是能给同学们留下印象的，无论是宣传力度还是宣传成本都会降低而效果却一定会上升，所以这样的一个有价值的课题值得我们去研究。

1.2 本论文研究目的背景及意义

1.2.1 研究目的及背景

本次课题研究，对于学校各种工作室的宣传会起到很大的正面作用，减少经费开支人力使用或是加强宣传力度都有着很好的正面作用，就实际意义来说，这是一次校园宣传模式的大胆创新，课题的研究结论也会为同学们的各种工作宣传提供建议和帮助。

学校存在许多各色各样的工作室，但没有一个规范的展示平台。而 Web 网页是一种无需下载安装即可使用的应用，能以最低成本触达用户。而且 Web 网页还可以将图标生成到手机电脑桌面，不占内存。Web 端在手机和 PC 浏览器上都能展示，方便高效。基于这个背景，本项目采用 Web 网页端开发。用户可以通过浏览器访问平台。

1.2.2 研究意义

在我们所知道的，学校并没有和我们所研究的课题一样或差不多的官网，说明这是一次很有意义的尝试，本此课题使用后端 SpringBoot^[1]，MyBatis，Swagger，MySQL，Token，Redis，和前端小程序^[2]，BootStraop，交叉使用，对于丰富课题研究和对教学理论的支持具有重要的研究意义。

1.2.3 研究背景

1.3 国内外发展状态，发展水平存在的问题

1.3.1 国外研究现状

在国外各种宣传官网或是企业门户网站层出不穷，这些网站发展至今，现在的状况已经可以说达到了一个相当高的水平，无论是从宏观经济、企业内部、还是企业大链条上的各个方面，信息化已经成为互联网经济下的必备手段。

互联网信息化带来最为直观、也是最有说服力的经济效益是它对国民经济的发展起到了十分重要的推动作用，美国就是一个十分典型的例子，九十年代后期，美国约三分之一的经济增长来源于信息化的拉动。

1.3.2 国内研究现状

根据中国互联网发展中心调查数据显示：从 2001 年到 2006 年，中国的网站总数平均年增长约 27%，到了 2011 年 6 月 30 日我国网站数量约为 183 万家，而 2009 年之前，国内网站数量每年都是以高速递增的态势发展，纵观网站建设市场从 2003 至 2011 年由高转低的曲线发展趋势，反应的正式我国网站建设市场由“萌芽式”到“粗放式”再向“细分式”的转变。

虽然国内建站水平已经是比较成熟了，但国内网站最主要的是缺少很好的宣传推广工作，尚且还没有较为良好的用户体验。

1.4 本论文研究目标和主要内容

1.4.1 研究目标

论文预期实现以下目标：

（1）本论文将结合工作室的实际情况，分析同学们关注工作室的原动力，并利用大产业的概念来优化界定宣传的模式，构成工作室的宣传系统化，以使为各类工作室提供一条有效途径。

(2) 探究宣传网站的演化规律，论文将引入各类宣传理论的结构，对宣传途径进行分析，并对宣传官网运行过程中可能出现的问题进行解决研究，寻求宣传最好的规律和策略。

1.4.2 主要内容

本论文研究内容主要包括以下几个方面

(1) 工作室宣传官网宣传概念界定与理论基础研究

首先我们要对主流宣传手段进行辨析，对宣传的内涵进行阐述，对工作室宣传官网的宣传概念给予界定，即包含各类宣传手段的渠道、方式、方法、安全、效率、稳定的广义概念，指出宣传的主要视角。

(2) 宣传官网的建设手段分析

对官网主要使用技术进行剖析，在框架的基础上，结合各类前后端技术，并根据宣传手段和模式进行特殊改变，展现技术内涵，指出整个官网的功能定位。

1.4.3 拟解决的关键问题

(1) 系统的设计

传统 Web 应用开发架构经历了从视图和业务逻辑完全混在一起的单体应用发展到视图和业务逻辑分离的 MVC 结构。但也存在缺点如：用户每次想要看到最终页面必须要经过控制器、模型、视图三层。视图到达浏览器之前的所有渲染工作都在后端服务器进行，占用了服务器运算资源，同时页面性能无法得到很好的优化^[3]。

为了提高用户的流畅上网体验，同时防止高并发对服务器造成资源占用，应将页面渲染工作从后端服务器转移到前端进行，后台只负责提供数据，前端负责解析数据和页面渲染，这种架构就是前后端分离架构。

(2) 宣传问题

利用主流宣传手段 Web 端与小程序集合进行宣传，降低宣传成本，提高宣传力度和效果，相比与传统宣传手段更具典型性和实用性。

1.5 研究的方法

1.5.1 本论文研究方法

主要通过文献研究法进行研究

第二章 项目概述

2.1 项目术语解释

(一)SCSTC.S.S	【四川科技职业学院学生工作室】
(二)SpringBoot	【Java 平台上的一种开源应用框架】 ^[3]
(三)MyBatis	【优秀的持久层框架，它支持定制化 SQL】 ^[3]
(四)Swagger	【接口文档规范框架】
(五)MySQL	【免费开源关系型数据库】
(六)Redis	【免费 Nosql(非关系型)型数据库】
(七)BootStraop	【前端开发免费框架】
(八)Ajax	【用于创建更好更快以及交互性更强的 Web 应用程序的技术】
(九)JSON	【轻量级的数据交换格式】
(十)CORS	【"跨域资源共享"（Cross-origin resource sharing）】
(十一) UA	【提供浏览器类型、操作系统、浏览器内核的信息标识】

2.2 项目概述

本次课题研究，对于学校各种工作室的宣传会起到很大的正面作用，减少经费开支人力使用或是加强宣传力度都有着很好的正面作用，就实际意义来说，这是一次校园宣传模式的大胆创新，课题的研究结论也会为同学们的各种工作宣传提供建议和帮助。

普通用户浏览官网，页面将展示学生工作室列表及查看更多工作室列表入口，首页提供提交自己工作室接口，接口设置提交验证。管理员可以对工作室进行修改，删除等操作，登录权限验证，验证码等。

2.3 项目需求定义

表 2.3 需求定义分析

功能需求	主要功能点	优先级
提交工作室	上传图片	高
提交工作室	提交工作室数据	高
查看更多	展示工作室数据	高
工作室详情	工作室详细数据	高
工作室管理员	管理员登录	高
工作室管理员	管理员修改工作室	高
工作室管理员	管理员删除工作室	高
工作室管理员	验证码 & Session	高
性能要求	支持 3000/s 并发请求	中

2.4 项目系统架构设计

2.4.1 系统功能架构设计

是以图形和文字等形式形象得描述系统的整体物理架构模型，解释系统组成结构，重要节点，及其之间的物理联系方式如图 2.4.1

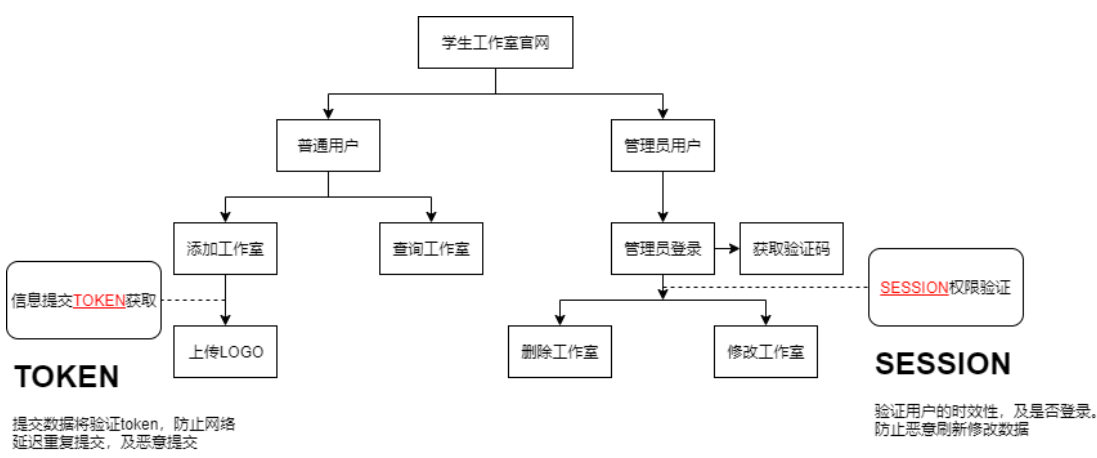


图 2.4.1 系统功能架构设计

2.4.2 系统技术框架设计

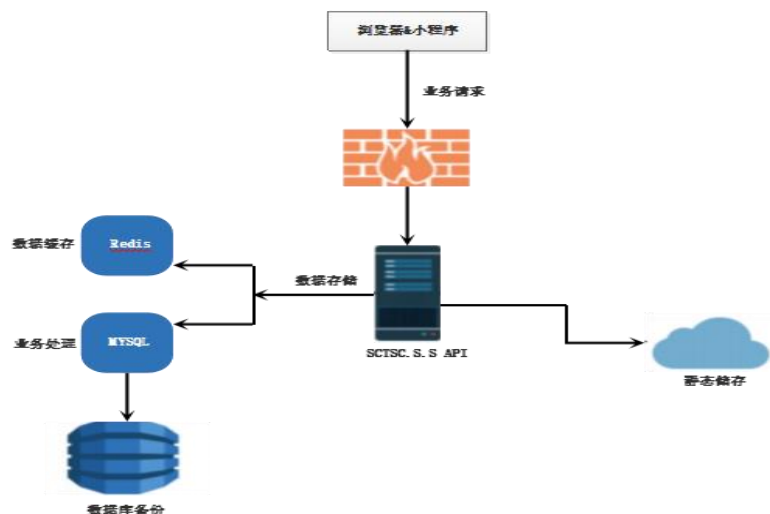


图 2.4.2 系统技术框架设计

- (一)浏览器网页与微信小程序提供用户操作界面
- (二)使用七牛云服务设置工作室 Logo 的上传功能
- (三)使用 Java SpringBoot, 提供安全稳定业务的 API 服务
- (四)使用 MySQL 关系型数据库, 提供高效、安全的数据持久化服务
- (五)使用 Redis 数据库, 提供高效, 快速, 安全的数据缓存
- (六)前端页面文件使用 CDN 加速页面的加载速度

第三章 前端功能详细设计

3.1 前端界面功能架构设计

Web 前端开发语言包括: HTML、CSS、JavaScript 这些语言都是前端开发中基本开发语言, 每种都有对应的任务。HTML 负责网页界面的整体结构与构图区域; CSS 主要负责网页界面的外观效果, 可以开发各式各样的网页外观样式, 如颜色、背景、阴影等等; JavaScript 语言是将网页客户端业务分离单独进行运行。客户端操作业务都用 JavaScript 语言进行编码。从而使开发语言相互之间独立互不影响^[6]。

3.1.1 前端功能框架设计

传统 Web 应用开发架构经历了从视图和业务逻辑完全混在一起的单体应用发展到视图和业务逻辑分离的 MVC 结构。但也存在缺点如：用户每次想要看到最终页面必须要经过控制器、模 型、视图三层。视图到达浏览器之前的所有渲染工作都在后端服务器进行，占用了服务器运算资源，同时页面性能无法得到很好的优化^{[4] [5]}。

为了提高用户的流畅上网体验，同时防止高并发对服务器造成资源占用，应将页面渲染工作从后端服务器转移到前端进行，后台只负责提供数据，前端负责解析数据和页面渲染，这种架构就是前后端分离架构如图 3.1.1 所示。

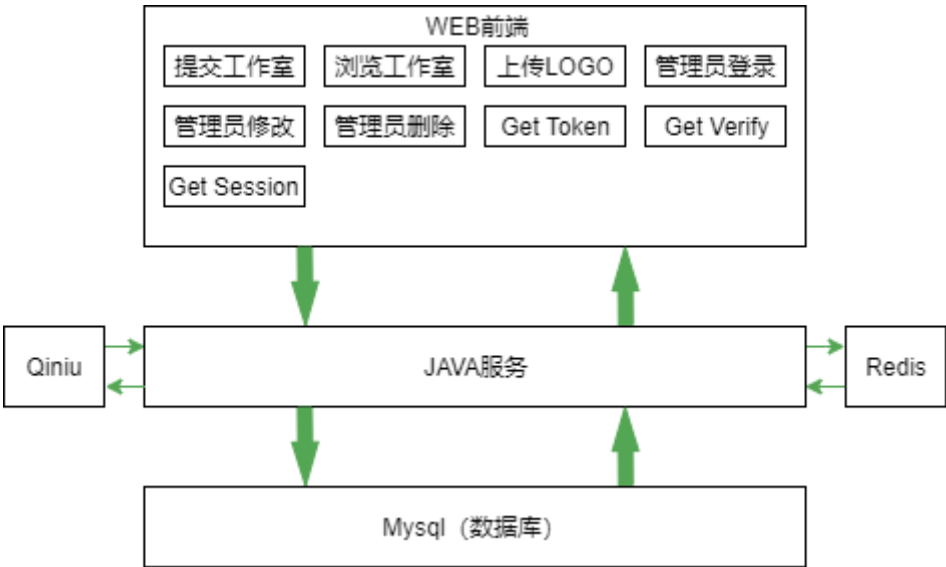


图 3.1.1 前端功能框架设计

3.1.2 前端技术框架设计

前后端之间通过 RESTFUL 风格的 URL 发出 Ajax 请求。REST(Representational State Transfer)即表述性状态传递，RESTFUL 的核心思想是将互联网资源对应到一个 URL 上，对资源的相关操作分别用 HTTP 协议里的 GET,POST,DELETE,PUT 表示。通过不同的请求协议，对指定资源进行对应操作。

RESTFUL 既简单又直观，通过 HTTP 的方法对资源进行操作，简化了客户端和服务 器之间交流及表达方式如图 3.1.2.1 所示。服务器将客户端对应请求的数据以 Json 数据格式返回，客户端将对 JSON 数据进行解析处理如图 3.1.2.2 所示

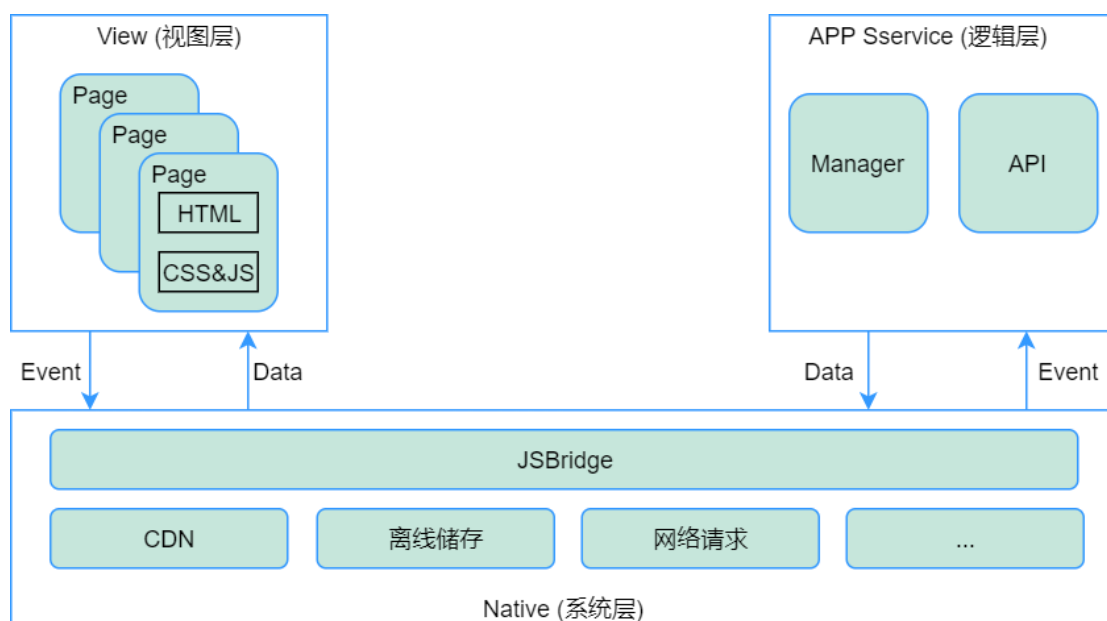


图 3.1.2.1 前端技术框架设计

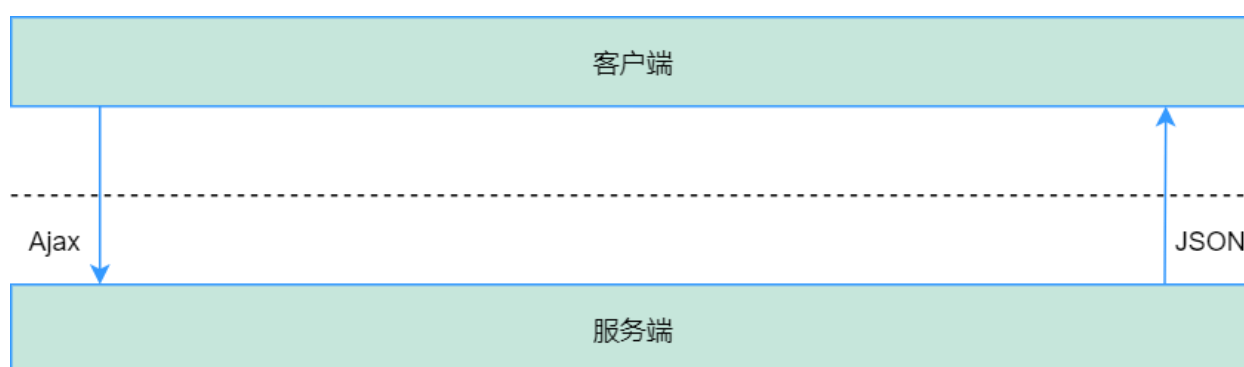


图 3.1.2.2 客户端与服务端交流表达方式

3.2 学生工作室首页

3.2.1 功能描述

当用户通过 URL、分享地址进入学生工作室首页时，先判断访问浏览器 UA 标识 (UserAgent)，通过 UA 标识显示对应比例的页面。页面加载成功将展示部分工作室数据。

系统首页提供上传工作室、查询更多工作室、查询工作室详细信息等接口，不同的接口对应不同的系统功能。

3.2.2 原型 UI 设计



图 3.2.2 系统首页 UI

3.2.3 业务流程

- (一)用户通过网址 URL、浏览器、分享连接访问官网, 将获取工作室列表(DATA)数据展示在首页
- (二)首页提供添加工作室入口、查询更多工作室、查询工作室详细数据等接口

3.3 查询更多学生工作室页面

3.3.1 功能描述

通过首页接口，获取所有学生工作室数据列表展示在页面中，为用户展示所有工作室。方便用户查询、搜索、了解、联系工作室负责人。

3.3.2 原型 UI 设计



图 3.3.2 工作室列表页面 UI

3.3.3 业务流程

- (一)请求访问所有工作室时，将服务器中所有的工作以一行六个展示(注意:手机浏览采用自适应展示)
- (二)每一个工作室提供查询详情的接口

3.4 工作室详情页面

3.4.1 功能描述

展示工作室的详情，详情信息包含 LOGO、人数、地址、简介、负责人联系方式

3.4.2 原型 UI 设计



图 3.4.2 工作室详情页面 UI

3.4.3 业务流程

- (一)用户点击某一个工作室将获取该工作室的详细信息，将数据展示在页面中
- (二)页面提供联系工作室负责人入口（腾讯开发者提供 QQ 接口）

3.5 添加上传工作室页面

3.5.1 .功能描述

提供上传工作室服务，工作室负责人将工作室上传到系统进行宣传展示。但上传学生工作室接口常常会遇到提交按钮可能会被点击多次，或者由于网络重发，或者 Nginx 重发等情况会导致数据被重复提交。

基于缺陷在系统设计的过程中，合理的使用乐观锁，通过 UploadKey 等其他条件，来做乐观锁的判断条件，这样保证更新操作即使在并发的情况下，不会有重复提交的问题。进入页面服务器将分配唯一的 UploadKey，提交时将会对 UploadKey 进行时效性，唯一性进行判断，存在提交成功反之提交失败，刷新重复提交。

3.5.2 原型 UI 设计



图 3.5.2 添加工作室页面 UI

3.5.3 业务流程

填写表单前，先获取服务器分配的唯一 UploadKey，用于上传验证，点击提交时将工作室负责人填写的表单信息保持在服务器与静态储存服务器中出现错误时，页面弹出错误提示

3.6 表单未填写页面

3.6.1 功能描述

负责人填写表单数据提交时，先判断表单是否有未填写，存在将未填写内容提示给填写者，当所有表单内容填写完成时，才能将数据提交保存在服务器上。表单验证采用 JavaScript 与 Jquery 进行验证，当表单未填写，将提示错误，提示错误信息包括如下：

Studio name not filled (工作室名称未填写)、 The number of studio not filled (工作室人数未填写)、 Address where the studio has not filled (工作室所在地址未填写)、 Contact QQ studio head not filled (工作室负责人联系 QQ 未填写)、 Introduction not filled studio (工作室简介未填写)、 LOGO studio did not upload (工作室 LOGO 未上传)、 LOGO upload failed (LOGO 上传失败)、 Image is not selected (未选择图片)

3.6.2 原型 UI 设计

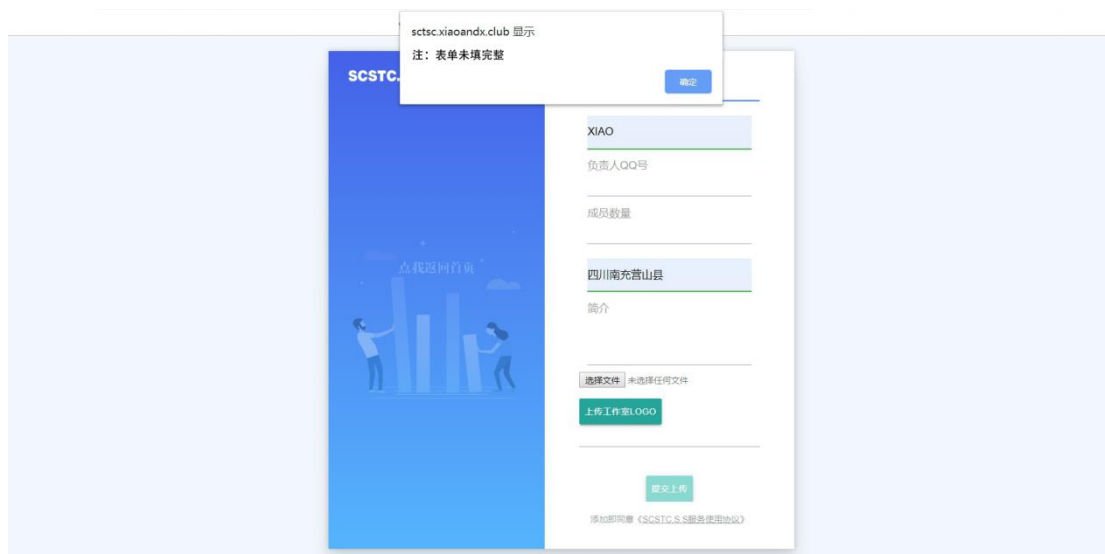


图 3.6.2 表单未填写 UI

3.6.3 业务流程

负责人填写表单数据提交时，先判断表单是否有未填写，存在未填写内容将错误信息提示填写者，并阻断请求，防止错误提交。

3.7 关于我们页面

3.7.1 功能描述

用户进入查看“关于我们”接口是，从 CDN 加速服务器上获取页面资源，展示在客户端上，页面采用滑动展示，最后一页为系统操作视频。

3.7.2 原型 UI 设计



图 3.7.2.1 关于我们页面 UI



图 3.7.2.2 关于我们页面 UI

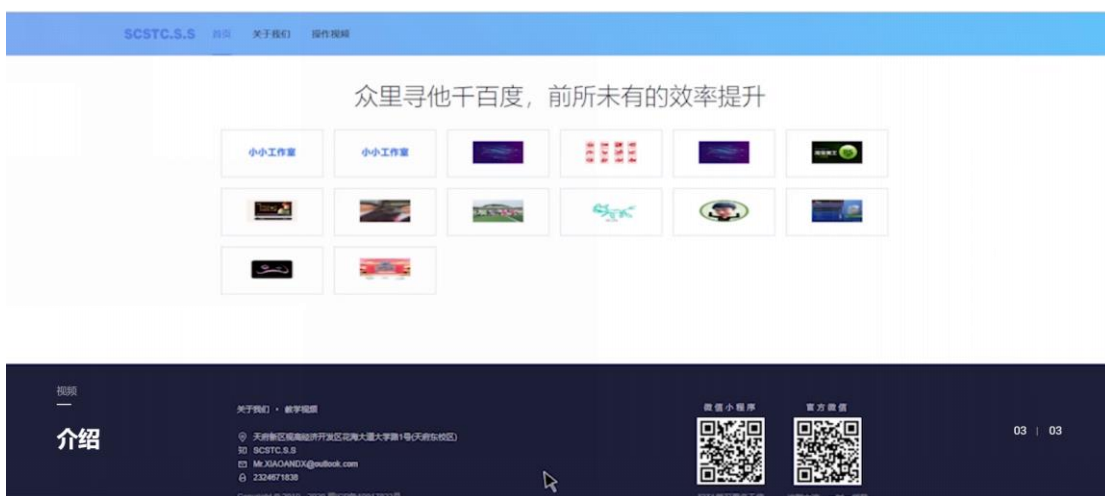


图 3.7.2.3 关于我们页面操作视频 UI

3.7.3 业务流程

展示官网介绍及官网操作视频(需要优质网络)，页面所有文件采用 CDN 加速，优化访问流畅度。

3.8 管理员登录页面（管理员）

3.8.1 功能描述

管理员操作系统需要通过 URL 后面加 admin 访问，进入系统需要身份验证。身份无误登录到系统，反之将提示错误信息刷新页面。

3.8.2 原型 UI 设计



3.8.3 业务流程

系统加载页面时，通过 API 获取验证码；管理员填写表单，当用户点击登录按钮时。客户端验证表单数据完整性，服务器将验证验证码是否正确，验证通过后服务器在验证用户名、密码是否正确，身份核实无误，服务器将自定义 UserSession 返回给客户端。客户端通过 Ajax 异步请求，判断用户身份及其身份的时效性。

3.9 错误提示页面（管理员）

3.9.1 功能描述

用户输入的验证码与服务器中 Redis 数据库中的验证码进行比对(区分大小写); 获取用户输入的用户名密码, 验证是否正确。当验证码或者用户名密码错误时, 页面将错误内容显示在页面中。

3.9.2 原型 UI 设计

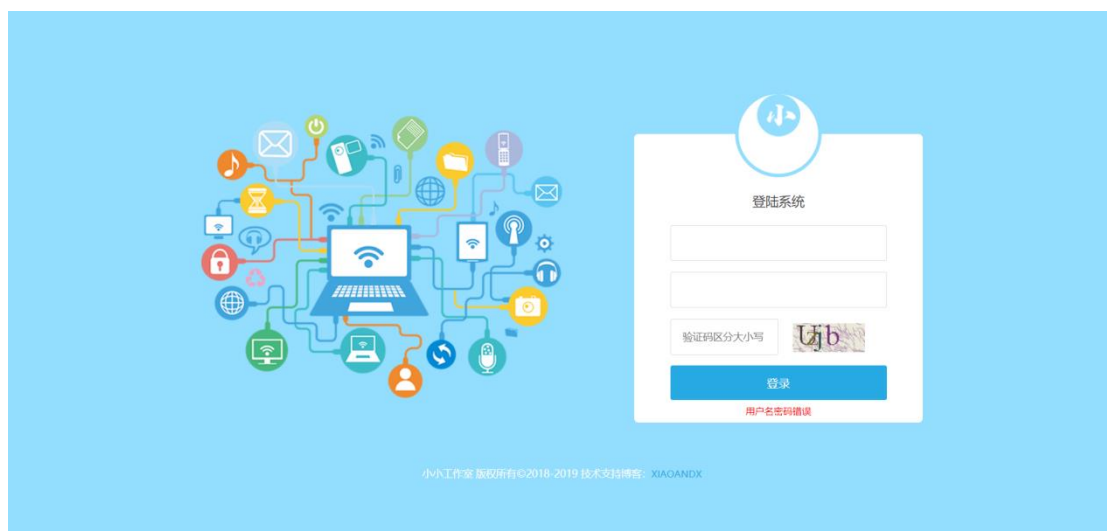


图 3.9.2.1 验证码错误提示 UI



图 3.9.2.2 用户名密码错误提示 UI

3.9.3 业务流程

登录失败将错误信息展示在页面上提醒用户确认填写信息正确再登录

3.10 后台管理页面（管理员）

3.10.1 功能描述

后台管理系统主页将显示所有工作室列表数据，并提供对每一个工作室的修改，删除；页面还包括模糊搜索查询，分页显示。

3.10.2 原型 UI 设计

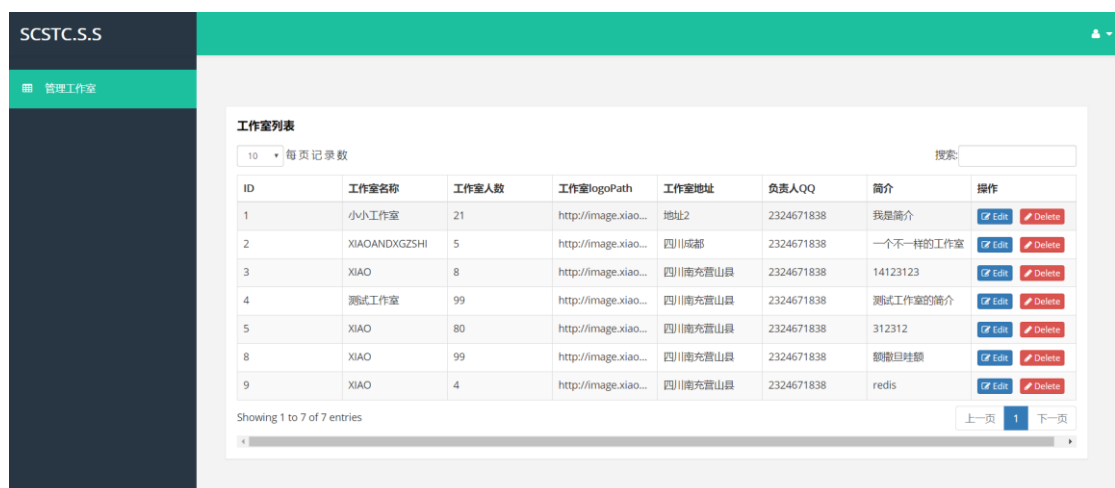


图 3.10.2 后台管理系统主页 UI

3.10.3 业务流程

管理员通过后台系统登录成功后，系统主页将获取用户名并判断服务端自定义 session 失效时间及身份验证，如有身份过期将提示重新登录，身份验证成功客户端将请求服务端获取返回所有的工作室数据，以分页方式展示在界面中。

3.11 修改工作室（管理员）

3.11.1 功能描述

点击对应工作室后面的修改按钮，弹出修改框，进行数据修改，修改内容不包括工作室名称，工作室 Logo 等。

3.11.2 原型 UI 设计

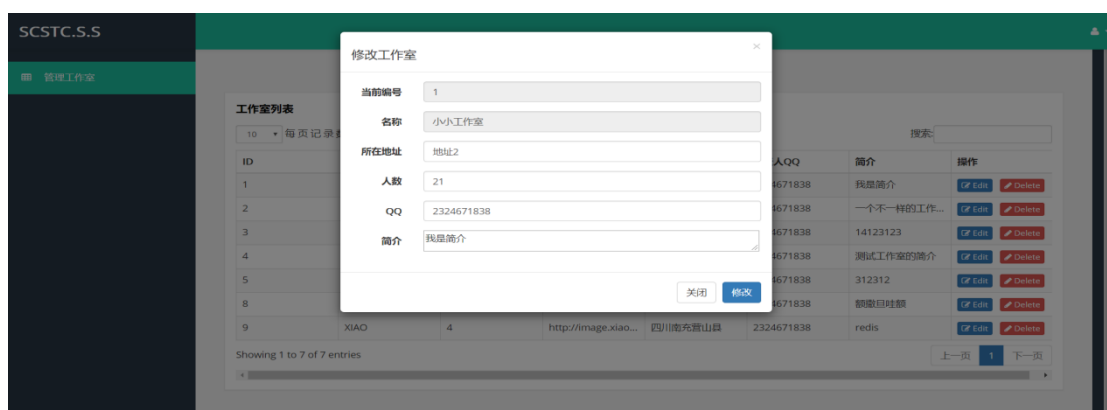


图 3.11.2 修改工作室接口 UI

3.11.3 业务流程

点击修改按钮，查询对应的工作室数据显示在文本框中，修改信息填写完成后点击修改按钮，服务器返回修改状态，修改成功返回页面自动显示更新后的数据。反之弹出错误提示。

3.12 删除数据页面（管理员）

3.12.1 功能描述

管理员可以对工作室进行删除处理，当出现违规，不符合要求，不存在的工作室，管理员有权进行删除数据处理。

3.12.2 原型 UI 设计

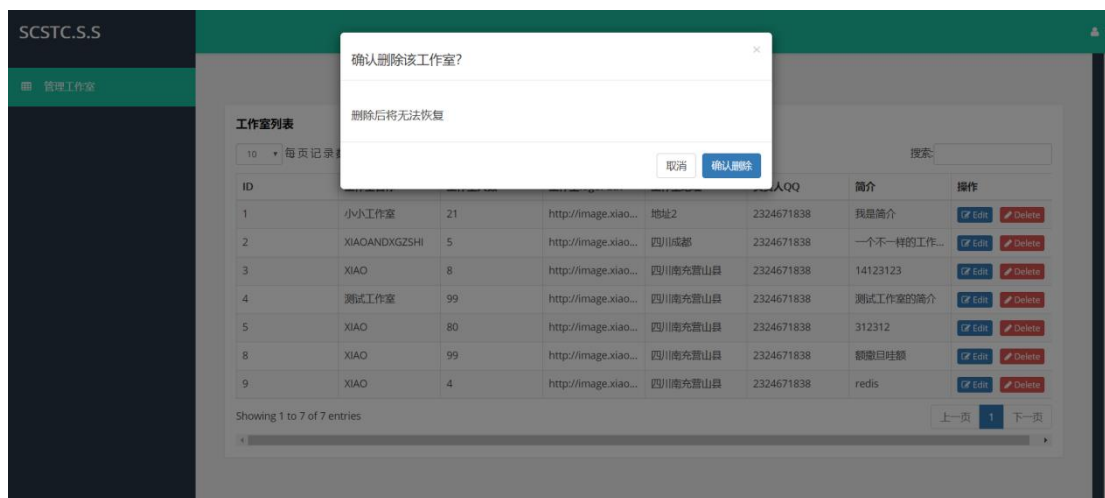


图 3.12.2 修改功能 UI

3.12.3 业务流程

管理员删除具体工作室时弹出确认提示框，避免操作错误删除数据。

3.13 搜索分页页面（管理员）

3.13.1 功能描述

管理员在主页搜索框中，输入搜索关键字，客户端会进行模糊查询，并将查询结果显示在页面中，查询与分页功能使用的 JQUERY 框架搭建，解决了在服务端进行模糊查询的延迟性。

搜索可以快速定位指定的数据，将数据进行分页展示，分页可以自定义显示数量；分页展示的数据也可以进行修改删除操作。

3.13.2 原型 UI 设计

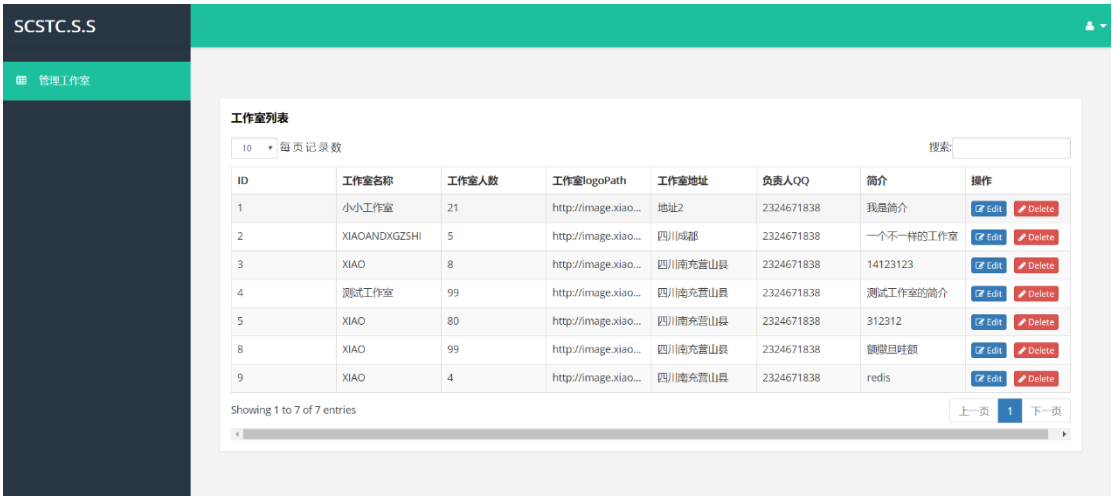


图 3.13.2.1 搜索界面 UI

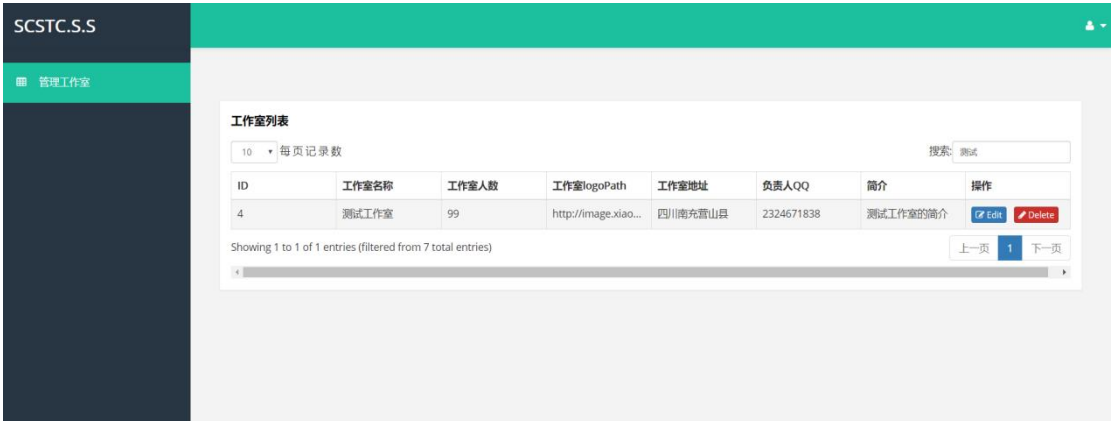


图 3.13.2.2 搜索结果 UI

3.13.3 业务流程

搜索采用模糊搜索，如果有对应的数据将直接显示匹配数据进行分页展示，分页及搜索都是建立在获取到服务端返回的数据基础上。

第四章 服务端 API 详细设计

4.1 服务端功能技术框架设计

4.1.1 服务端功能框架设计

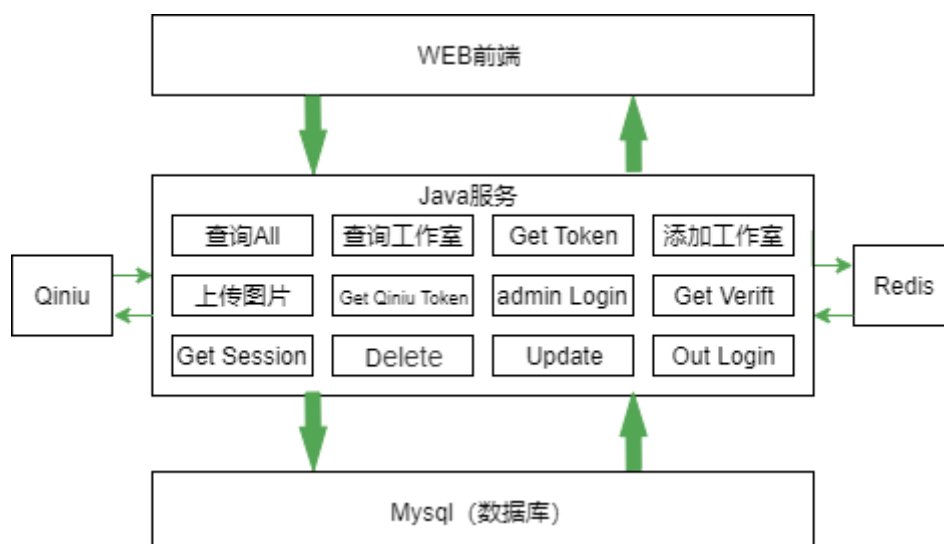


图 4.1.1 后端功能框架设计

4.1.2 服务端技术框架设计

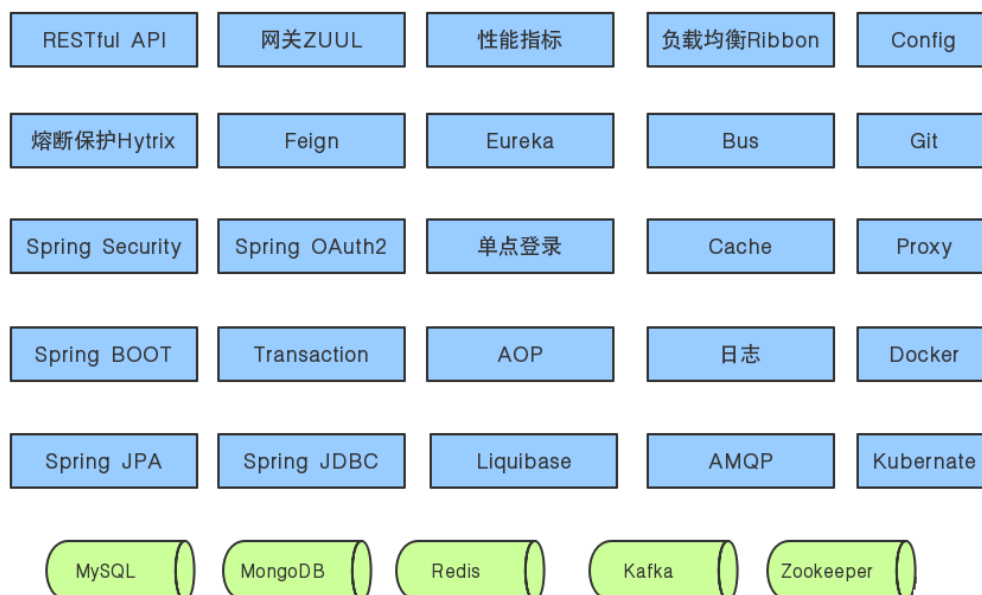


图 4.1.2 后端技术框架设计

4.2 查询所有学生工作室 API 接口设计

4.2.1 API 接口功能描述

接口简述：客户端请求该接口，服务端返回所有学生工作室的 JSONData

4.2.2 接口原形定义

/v1/open/studio/getStudioList

4.2.3 接口流程

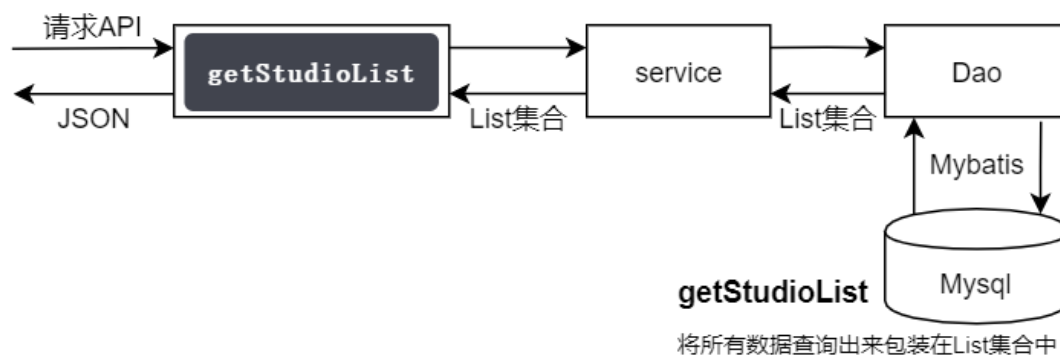


图 4.2.3 查询所有工作室接口流程图

4.2.4 请求 URL 数据格式

`http://api.xiaoandx.club/v1/open/studio/getStudioLis` (GET)

4.2.5 接口返回数据格式

```
[
  {
    "saddress": "string",
    "sid": 0, "slogoPath": "string",
    "sname": "string", "snumber": 0,
    "sqq": "string", "sstatus": 0,
```

```

        "ssummary": "string", "stime": "2019-11-28T01:49:24.054Z"
    }
]

```

表 4.2.5 返回数据字段说明

字段	字段类型	字段含义	备注
sid	int	工作室 ID	工作室唯一标识符
saddress	String	工作室所在地址	长度 100 字符
slogoPath	String	工作室 logo 地址	图片存储在对象储存服务器上地址
sname	String	工作室名称	工作室的全称
snumber	int	工作室人数	总人数
sqq	String	联系人 QQ	工作室负责人 QQ 联系方式
sstatus	int	工作室状态	0 审核通过，1 不通过
ssummary	String	工作室简介	工作室对应的简介介绍
stime	Data	工作室添加时间	上传平台时间

4.3 查询指定 id 的工作室 API 接口设计

4.3.1 API 接口功能描述

接口简述：查询具体某个学生工作室详情数据，注意先将 id 进行 Base64 转换在发起请求

4.3.2 接口原形定义

/v1/open/studio/findById/{studioId}

4.3.3 接口流程

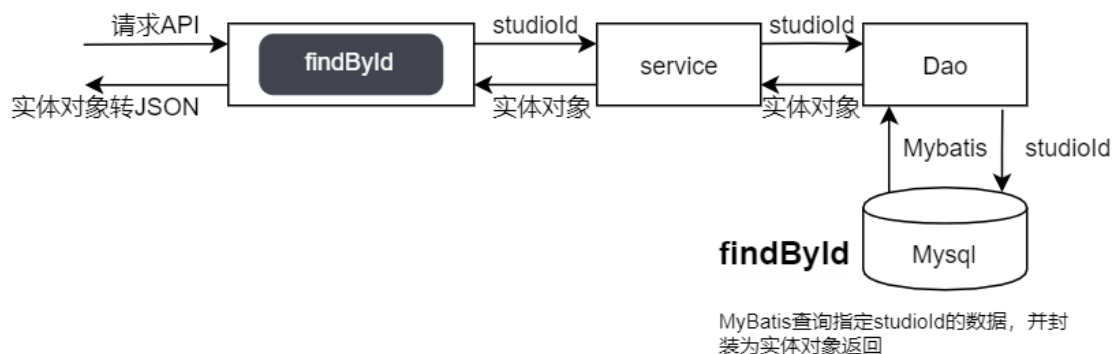


图 4.3.3 查询指定 id 的工作室 API 接口流程

4.3.4 请求 URL 数据格式

向服务地址 `http://api.xiaoandx.club/v1/open/studio/findById/{studioId}` （GET），
API 实例：`http://api.xiaoandx.club/v1/open/studio/findById/ Mw==` （注：前端将 id 为 2 的字符转换成 Base64 字符格式)

表 4.3.4 API 请求参数说明

参数名	参数类型	是否必传	含义
studioId	String	true	id 转换 Base64 字符

4.3.5 接口返回数据格式

数据格式定义：

```
{
  "saddress": "string",
  "sid": 0, "slogoPath": "string",
  "sname": "string", "snumber": 0,
  "sqq": "string", "sstatus": 0,
  "ssummary": "string", "stime": "2019-11-28T01:49:24.054Z"
}
```

表 4.3.5 返回数据字段说明

字段	字段类型	字段含义	备注
sid	int	工作室 ID	工作室唯一标识符
saddress	String	工作室所在地址	长度 100 字符
slogoPath	String	工作室 logo 地址	图片存储在对象储存服务器上地址
sname	String	工作室名称	工作室的全称
snumber	int	工作室人数	总人数
sqq	String	联系人 QQ	工作室负责人 QQ 联系方式
sstatus	int	工作室状态	0 审核通过，1 不通过
ssummary	String	工作室简介	工作室对应的简介介绍
stime	Data	工作室添加时间	上传平台时间

4.4 获取上传操作 token API 接口设计

4.4.1 接口功能描述

接口简述：上传工作室时，需要在请求头部中设置 Token，这个接口就是用于获取对应操作 Token 接口，避免重复上传 ，恶意上上传提交。

4.4.2 接口原形定义

/v1/open/studio/redisToken

4.4.3 接口流程

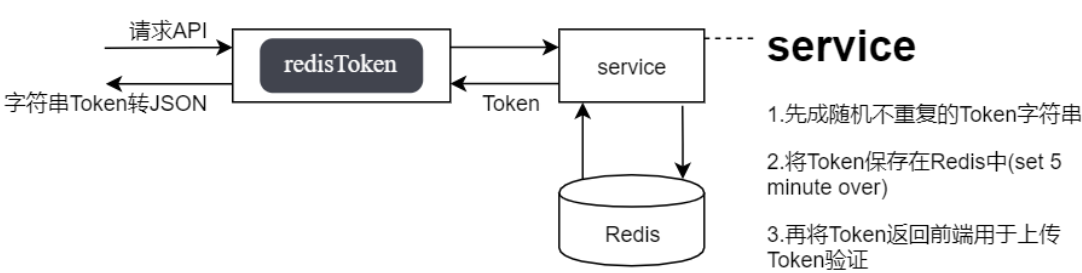


图 4.4.3 获取上传操作 token API 接口流程

4.4.4 请求 URL 数据格式

http://api.xiaoandx.club/v1/open/studio/redisToken （GET）

4.4.5 服务器返回数据格式

```
{
  "Token": string
}
```

表 4.4.5 返回数据字段说明

字段	字段类型	字段含义	备注
Token	String	上传验证 Token	用于验证表单提交

4.5 上传工作室 API 接口设计

4.5.1 API 接口功能描述

将用户填写的表单信息保存在数据库中，该接口需要 Head 传值 Token (用于验证) 和 Body 传值数据对象。该接口采用幂等性设计，有效防止重复提交，恶意提交。

4.5.2 接口原形定义

/v1/open/studio/addStudio

4.5.3 接口流程

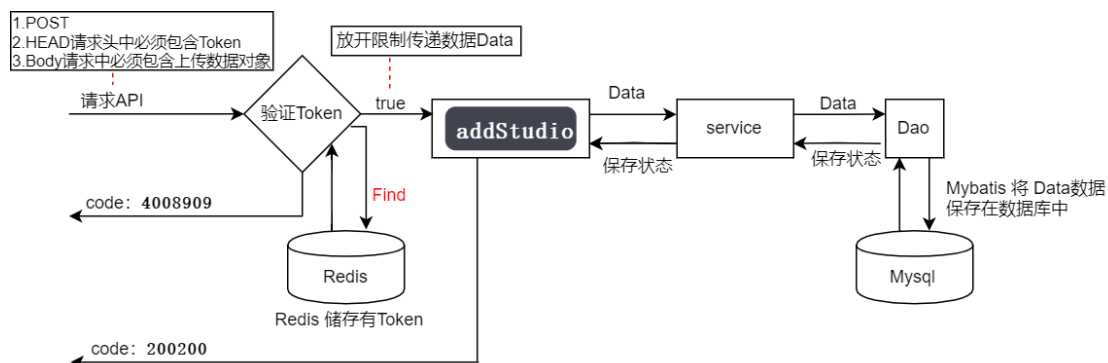


图 4.5.3 上传工作室 API 接口流程

4.5.4 请求 URL 数据格式

http://api.xiaoandx.club/v1/open/studio/addStudio (POST)

HEAD: Token 值（通过/v1/open/studio/redisToken 获取）

BODY:

```
{
  "sname": "string",
  "snumber": 0,
  "sqq": "string",
  "saddress": "string",
  "slogoPath": "string",
}
```

```
"ssummary": "string",  
}
```

表 4.5.4 API 请求参数说明

参数名	参数类型	是否必传	含义
saddress	String	true	工作室所在地址
slogoPath	String	true	工作室 logo 地址
sname	String	true	工作室名称
snumber	int	true	工作室人数
sqq	String	true	联系人 QQ
ssummary	String	true	工作室简介

4.5.5 服务器返回数据格式

```
{  
  "message": "string",  
  "statusCode": 0  
}
```

表 4.5.5 返回数据字段说明

字段	字段类型	字段含义	备注
statusCode	Int	保存数据状态码	200200 , 400400 OR 4008909
message	String	状态信息说明	说明操作状态或者异常原因

4.6 获取上传图片 Token API 接口设计

4.6.1 API 接口功能描述

接口简述：获取静态储存上传 Token，Token 用于上传图片权限验证，每次上传图片都需要获取 Token。

4.6.2 接口原形定义

/v1/open/studio/getToken

4.6.3 接口流程

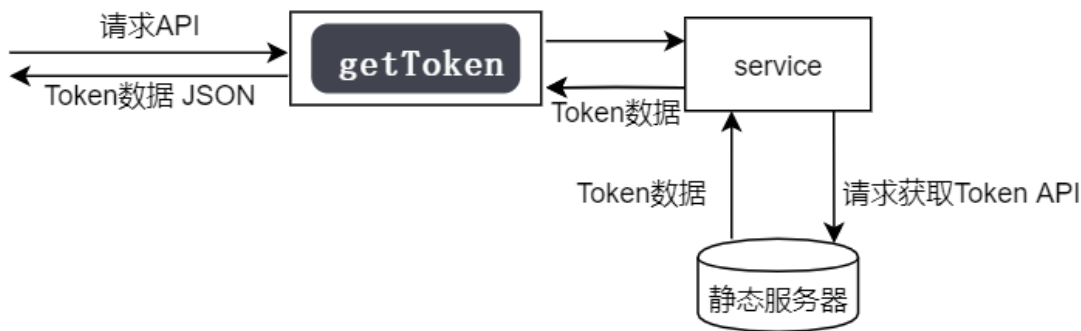


图 4.6.3 获取上传图片 Token API 接口流程

4.6.4 请求 URL 数据格式

http://api.xiaoandx.club/v1/open/studio/getToken （GET）

4.6.5 服务器返回数据格式

```
{
  "uptoken": "String"
}
```

表 4.6.5 返回数据字段说明

字段	字段类型	字段含义	备注
uptoken	String	上传 Token	用于上传权限验证

4.7 获取验证码 API 接口设计

4.7.1 API 接口功能描述

接口简述：客户端请求接口服务端返回验证码图片，验证码用于管理员登录验证

4.7.2 接口原形定义

/v1/open/studio/user/check.jpg

4.7.3 接口流程

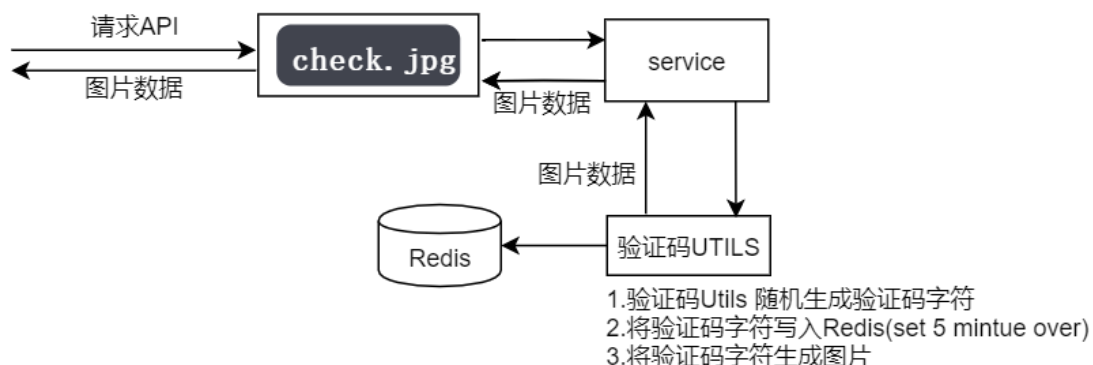


图 4.7.3 获取验证码 API 接口流程

4.7.4 请求 URL 数据格式

<http://api.xiaoandx.club/v1/open/studio/user/check.jpg> (GET)

4.7.5 服务器返回数据格式

服务器直接返回图片 IO 流

4.8 管理员登录 API 接口设计

4.8.1 API 接口功能描述

接口简述：服务端验证管理员身份，无误允许管理员登录系统

4.8.2 接口原形定义

[/ 1/open/studio/doLogin](#)

4.8.3 接口流程

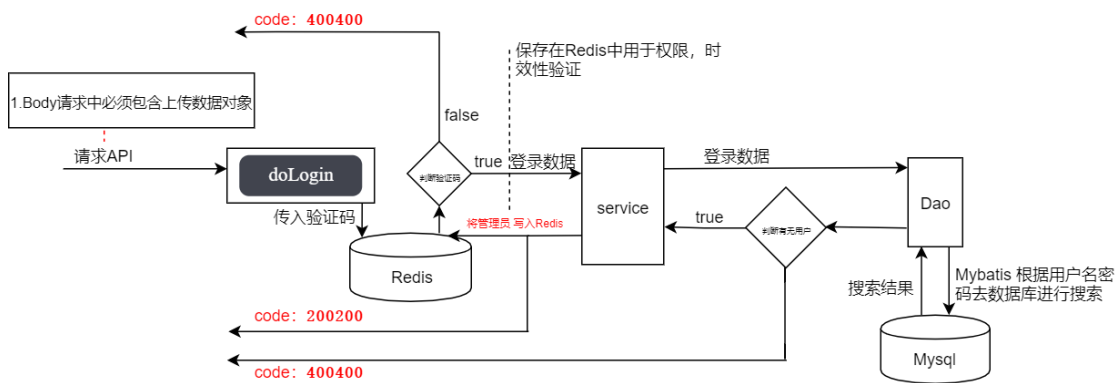


图 4.8.3 管理员登录 API 接口流程

4.8.4 请求 URL 数据格式

http://api.xiaoandx.club/v1/open/studio/doLogin （POST）

```
{
  "check": "string",
  "password": "string",
  "username": "string"
}
```

表 4.8.4 API 请求参数说明

参数名	参数类型	是否必传	含义
check	String	验证码	区分大小写
password	String	管理员密码	如：admin
username	String	管理员账号	如：admin

4.8.5 服务器返回数据格式

```
{ "message": "string","statusCode": 0 }
```

表 4.8.5 返回数据字段说明

字段	字段类型	字段含义	备注
statusCode	Int	保存数据状态码	200200 , 400400 OR 4008909
message	String	状态信息说明	说明操作状态或者异常原因

4.9 获取管理员 SESSION API 接口设计

4.9.1 API 接口功能描述

接口简述：获取管理员登录 session，判断权限及身份是否过期

4.9.2 接口原形定义

/v1/open/studio/getUserSession/{userName}

4.9.3 接口流程

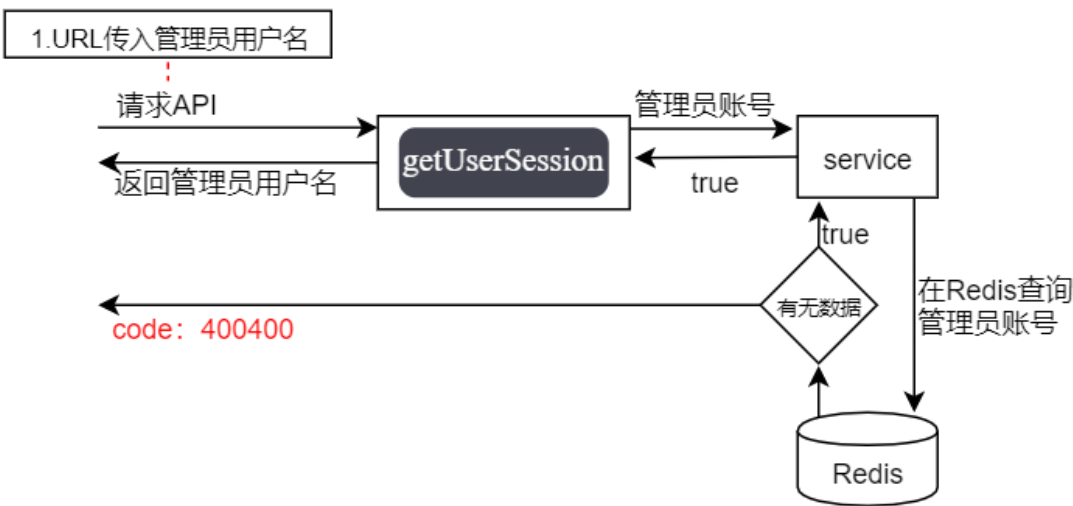


图 4.9.3 获取管理员 SESSION API 接口流程

4.9.4 请求 URL 数据格式

http://api.xiaoandx.club/v1/open/studio/getUserSession/{username} (GET)

API 示例：http://api.xiaoandx.club/v1/open/studio/getUserSession/admin

表 4.9.4 API 请求参数说明

参数名	参数类型	是否必传	含义
username	String	管理员用户名	如：admin

4.9.5 服务器返回数据格式

服务器直接返回客户端此时登录管理员用户名

4.10 管理员修改工作室 API 接口设计

4.10.1 API 接口功能描述

接口简述：客户端请求 API，向服务端传递工作室 ID 及修改数据内容，服务端将修个工作室数据

4.10.2 接口原形定义

/v1/open/studio/updateStudio

4.10.3 接口流程

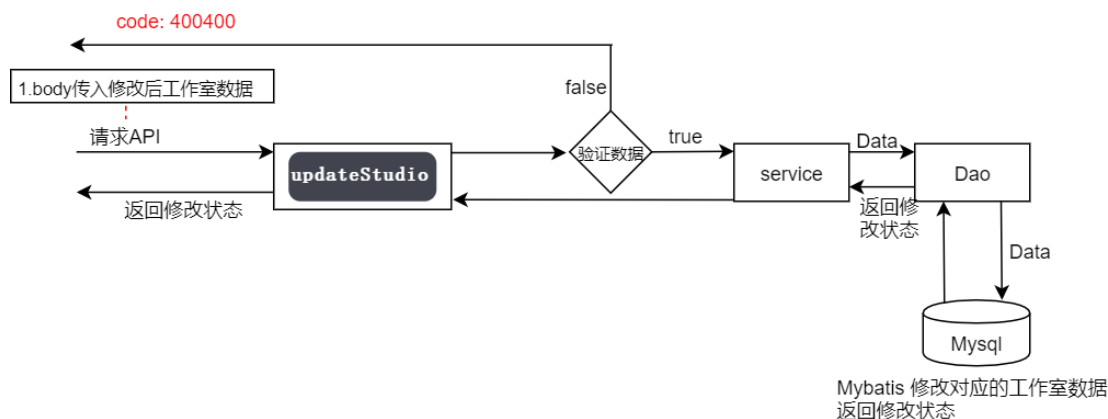


图 4.10.3 管理员修改工作室 API 接口

4.10.4 请求 URL 数据格式

http://api.xiaoandx.club/v1/open/studio/updateStudio (PUT)

Body 传值:

```
{ "saddress": "string",
  "sid": 0,
```

```
"snumber": 0,  
"sqq": "string",  
"ssummary": "string",  
}
```

表 4.10.4 API 请求参数说明

参数名	参数类型	是否必传	含义
saddress	String	工作室所在地址	长度 100 字符
sid	Int	工作室 ID	工作室唯一标识符
snumber	int	工作室人数	总人数
sqq	String	联系人 QQ	工作室负责人 QQ 联系方式
ssummary	String	工作室简介	工作室对应的简介介绍

4.10.5 服务器返回数据格式

```
{"message": "string", "statusCode": 0}
```

表 4.10.5 返回数据字段说明

字段	字段类型	字段含义	备注
statusCode	Int	保存数据状态码	200200 , 400400 OR 4008909
message	String	状态信息说明	说明操作状态或者异常原因

4.11 管理员删除工作室 API 接口设计

4.11.1 API 接口功能描述

接口简述：通过工作室 Id 删除数据库中对应工作室 Id 的所有数据

4.11.2 接口原形定义

/v1/open/studio/deleteStudio/{studioId}

4.11.3 接口流程

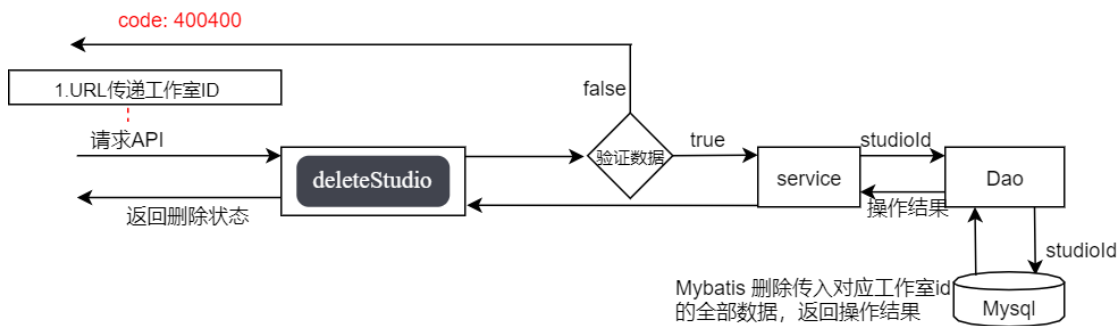


图 4.11.3 管理员删除工作室 API 接口流程

4.11.4 请求 URL 数据格式

http://api.xiaoandx.club/v1/open/studio/deleteStudio/{studioId}（DELETE）

API 示例：http://api.xiaoandx.club/v1/open/studio/getUserSession/98376

表 4.11.4 API 请求参数说明

参数名	参数类型	是否必传	含义
studioId	Int	工作室 ID	如：98376

4.11.5 服务器返回数据格式

{"message": "string", "statusCode": 0}

表 4.11.5 返回数据字段说明

字段	字段类型	字段含义	备注
statusCode	Int	保存数据状态码	200200 , 400400 OR 4008909
message	String	状态信息说明	说明操作状态或者异常原因

第五章 数据库设计

5.1 数据库模型设计

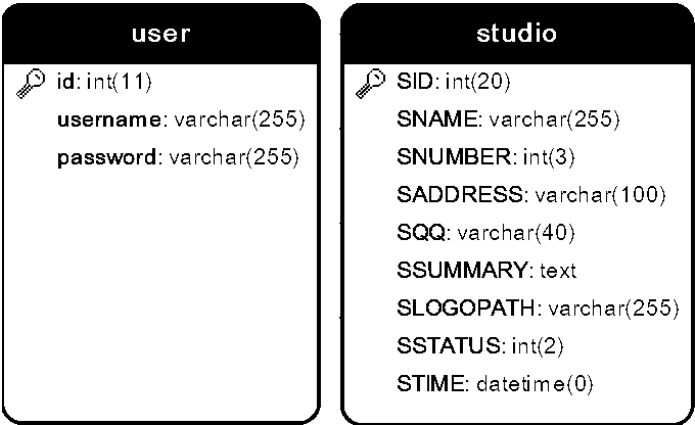


图 5.1 数据库模型设计

5.2 学生工作室记录表设计

表 5.2 studio

字段	主键键	类型	长度	是否自增	是否为空	描述
SID	主键	Int	20	是	否	工作室 id
SNAME		Varchar	255	否	否	工作室名称
SNUMBER		Int	3	否	否	工作室人数
SADDRESS		Varchar	100	否	否	所在地址
SQQ		Varchar	40	否	否	负责人 QQ
SSUMMARY		text	0	否	否	工作室简介
SLOGOPATH		Varchar	255	否	否	Logo地址
SSTATUS		Int	2	否	否	工作室状态
STIME		Datetime	0	否	否	上传时间

5.3 管理员表设计

表 5.3 user

字段	主外键	类型	长度	是否自增	是否为空	描述
ID	主键	Int	11	是	否	管理员 id
username		Varchar	255	否	否	管理员账号
password		Varchar	255	否	否	管理员密码

第六章 系统性能优化设计

现阶段随着人们对互联网使用体验要求的提升，网络技术员与软件开发人员对各个方向对现有的互联网技术、软件系统等进行功能完善和性能提升。从现有网络技术发展来看，现在 Web 前端开发技术的优化方向主要是优化页面加载速度，页面美观，便利操作给使用者提供更好的优质体验。后端开发技术的优化方向主要是优化 CPU 性能、服务器内存、功能逻辑、系统的安全性等，给使用者提供更快，更安全的使用体验。就目前而言的优化过程中，技术开发人员常常采用目前比较成熟的开发技术，而不是在原有技术上进行优化和技术创新，这样用户的使用体验很难得到有效的提升。那么，我们应该从那几个方向对 Web 前端开发技术和后端开发技术进行优化。可以参照以下几点^[7]：

6.1 Web 前端开发技术优化方向

中国目前网络用户接受并使用的浏览器包括 Chrome、IE、火狐、360 浏览器、百度浏览器等等，但是不同的浏览器采用的内核不一致，不同的浏览器都有自己的解析协议，所以浏览器间往往会出现互不兼容的问题，目前这个问题也是前端开发工程师在开发过程中必须要解决的问题之一。现在，常使用的浏览器当对同一段相同代码解析不同浏览器显示的效果和结果不一致。所以前端开发者如何将同一代码在不同浏览器中显示效果一致，也是开发中的难题。往往前端开发工程师将同一代码针对不同的浏览器进行单独开发调试，然后实现浏览兼容显示效果一致^[8]。

6.1.1 减小静态文件的大小

当用户访问网址网页时，最直观的感受就是页面内容出来的速度，网页加载速度直接影响用户使用体验。针对这个问题，我们需要优化网页文件的大小，提高加载速度。

(一)代码压缩

前端软件开发工程师，在编写代码开发时，都会遵循开发规范，会造成 JS，CSS 文件数据量较大。但是当系统发布上线时，客户端只是识别执行代码，但是不会去遵循代码规范。面对这样问题，我们将单元测试通过后的功能代码，进行混淆和压缩，减小静态文件的数据大小。代码混淆压缩后也会起到保护业务源码的效果。

(二)文件合并

当前前端软件开发工程师在进行项目开发时，会经常使用第三方插件及第三方代码库，如常用的 jQuery, Ajax, Axios, Bootstrap, weixin-js-sdk, lodash 等等，每个插件都有自己的脚本支持文件，在页面使用<script>标签引入脚本文件，但是当脚本多时，会加大客户端请求时间与数量，大大延迟页面加载渲染时间。

可以通过合并 JS 和 CSS 文件来减少脚本请求，从而减少请求数量，达到提高加载速度的效果。

(三)CDN 加速

什么是 CDN? 为什么使用 CDN?

CDN(Content Delivery Network)即内容分发网络，CDN 是构建在现有网络基础之上的智能虚拟网络，依靠部署在各地的边缘服务器，通过中心平台的负载均衡、内容分发、调度等功能模块，使用户就近获取所需内容，降低网络拥塞，提高用户访问响应速度和命中率。CDN 的关键技术主要有内容存储和分发技术^[9]。

CDN 它把网站内容更快地传递给服务范围内的一个具体位置，将网站静态文件通过 CDN 分发则会使用户更快的加载页面，提高访问体验。

6.1.2 减少 HTTP OR HTTPS 请求

网页加载过程中请求服务器次数，页面渲染过程中，大部分时间都是浪费在 HTTP/HTTPS 请求上，同一域名下的 HTTP/HTTPS 请求有限，过多的请求会造成严重的加载渲染耗时，我们采用配置多个 CDN 域名来进行静态文件提速^[10]。

配置多个域名能够最大限度的增加并发请求量。提高用户使用体验。减少 API 请求数量，如将系统中的关键字搜索，分页显示数据都采用前端开发。后端不做搜索，分页业务；从而提高使用体验。

6.1.3 页面设计优化

目前许多系统中的业务功能操作复杂，用户使用系统中的功能需要三步以上的操作，这样的设计降低了用户体验。为此我们将系统中每个功能的操作控制在三步一下，从而降低操作的繁琐，提高用户的使用体验。

6.2 后端技术开发优化方向

服务端常常因为性能，代码的高内聚等等原因造成服务器 CPU，内存被大量占用，使的服务端响应耗时高、性能低、使用体验差、安全性低、不稳定等种种原因。为此我们将从下面几个方面进行服务器优化，从而提高服务器访问效率等。

6.2.1 API 接口幂等性

系统中存在上传提交保存工作室功能，设计这个接口时考虑到，当用户多次提交或者当网络延迟时存在多次请求时，如果不考虑幂等性，数据库就会保存多条相同的数据，这样造成服务器资源浪费，从而影响服务器性能，影响使用体验。

当出现多版本并发请求的时候，系统我们采用 `update with condition` 【更新带条件来防止】来保证多次客户端请求调用对后端服务器系统的影响是一致的。在系统接口设计的过程中，合理的使用乐观锁，通过 `UploadKey`，来做为乐观锁的判断条件，这样保证更新操作即使在并发的情况下，也不会出现重复数据保存在数据库中。客户端调用接口的时候，需要在 API 请求头中，传递 `UploadKey` 令牌参数，每次的令牌（不管是单机还是微服务还是多并发都是唯一的）只能使用一次，每次令牌只用五分钟失效，一旦使用或者失效之后，`UploadKey` 就会被删除，如果令牌失效服务端将返回 `code` 和 `message`。这样可以有效防止重复提交。

6.2.2 前后端业务分离

现阶段随着互联网的高速发展，前端页面及客户端界面交互越来越灵活，响应式体验要求越来越好。而且后端也要，要求系统服务的高并发、高可用、高稳定、高性能、高安全等特征。随着前后端的技术要求越来越高，从而导致了前后端都专注于自己领域的开发及技术研究创新。前后端分离开发随着时代的进步也是大势所趋。

前后端分离开发，也常常伴随着出现 CORS 跨域请求资源的错误。常用解决跨域的方法如下^[2]：

(一)JSONP

基本原理就是通过动态创建 `script` 标签，然后利用 `src` 属性进行跨域，但是要注意 JSONP 只支持 GET 请求，不支持 POST 请求。

(二)CORS（跨域资源共享）

利用 Nginx 或者 php、Java 等后端语言设置允许跨域请求 `header('Access-Control-Allow-Origin:'); header('Access-Control-Allow-Method:POST,GET')`。

(三)Nginx 服务器

直接使用 Nginx 服务器来部署前端项目，前端页面 ajax 请求的 URL 直接指向 Nginx 服务器的地址，由 Nginx 反向代理指向后端服务器即可，相对来说这种方式是最佳解决跨域的方案。

6.2.3 操作数据缓存

在系统中存在重复读取、UploadKey、Verification code、UserSession 等反复操作数据。Wile 达到高可用、高性能、高并发的条件，在系统架构设计加入了 NoSQL Redis 数据库，利用 Redis 的读写快、高性能、支持各种不同方式的排序、操作都是原子性、数据都是缓存在内存中的特点，将部分缓存数据储存在 Redis 中，从而提高访问加载速度，使系统接口 API 请求更快。

6.2.4 文件上传优化

系统设计时考虑到如果将文件图片保存在当前服务中，会造成服务器资源浪费，影响服务器性能。基于这个特点，将系统中图片等文件保存在 CDN 静态服务器中，当前服务器只需要保存记录对应图片储存在静态服务器上的 URL 地址即可，这样使的当前服务的性能得到优化，也能更好的管理图片文件如图 6.2.4 所示。

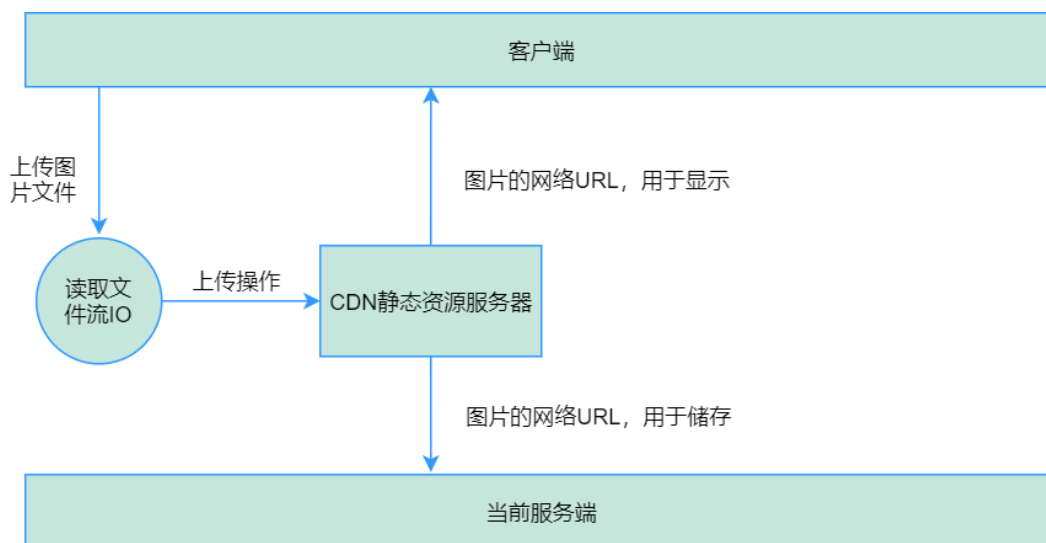


图 6.2.4 文件图片上传流程图

6.2.5 HTTPS 请求协议

HTTPS 是通过 HTTP 加上 TLS/SSL 协议共同构建的可进行加密传输、身份认证的网络协议，保护主要通过数字证书、加密算法、非对称密钥等技术完成对互联网数据传输加密，实现互联网传输安全保护^[10]。

四川科技职业学院学生工作室官网系统、服务器后端、CDN 静态资源服务器都采用 SSL 协议，为用户的访问提供网络传输安全保存。

第七章 系统安装部署

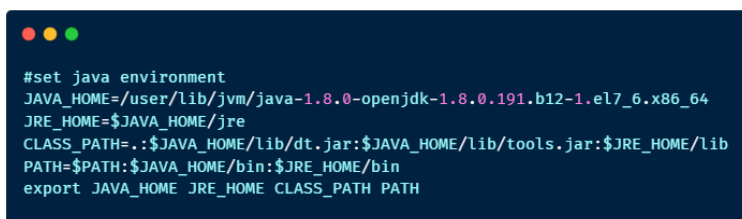
7.1 安装步骤

7.1.1 服务器安装环境

(一)服务器操作系统: Linux 操作系统

(二)服务运行环境安装: JVM;

- a) 启动 Linux 系统输入安装 jdk1.8 操作指令: `yum install java-1.8.0-openjdk java-1.8.0-openjdk-devel`
- b) 安装 Java 到指定文件夹: `ll /user/lib/jvm`
- c) 通过 vim 编辑模式设置 Java 环境变量: `vi /etc/profile`
- d) 在配置文件中添加如下配置如图 7.1.1.1 所示



```
#set java environment
JAVA_HOME=/user/lib/jvm/java-1.8.0-openjdk-1.8.0.191.b12-1.el7_6.x86_64
JRE_HOME=$JAVA_HOME/jre
CLASS_PATH=.:$JAVA_HOME/lib/dt.jar:$JAVA_HOME/lib/tools.jar:$JRE_HOME/lib
PATH=$PATH:$JAVA_HOME/bin:$JRE_HOME/bin
export JAVA_HOME JRE_HOME CLASS_PATH PATH
```

图 7.1.1.1 JDK 环境变量配置

- e) 输入测试是否安装成功指令: `java -version` 成功将显示对应的 JDK 版本号

(三)服务器还需要安装 MySQL、Redis 数据库

- a) 通过指令安装对应服务

(四)安装 Nginx 服务器用于配置访问域名

7.1.2 数据库安装

数据库是储存工作室数据的容器，在创建一个名为 `scstcss_data` 数据库并在库中创建两张对应的数据结构表如表 5.2 和表 5.3。

7.1.3 前端页面安装

前端采用静态安装模式，将系统网页源码打包上传到服务器指定目录下，并解压源码，源码内容包括 HTML、CSS、Image、Javascript 等内容，用于渲染系统界面。

7.2 数据配置

软件安装完成后将对软件进行关系配置

7.2.1 数据库配置

配置数据库的用户名及密码，通过服务器提供商，开启数据库端口，开放端口用于公网访问。

7.2.2 后端服务器配置

系统首页配置端口为 80、API 系统服务配置端口为 80，后端的配置将所有软件功能整合起来，形成一个完整的系统。

7.2.3 CDN 静态服务器配置

CDN 静态服务器用于储存上传的 Logo 与部分 JS 和 CSS 文件。CDN 服务器需要配置 SSL 协议，与默认首页。

7.3 系统环境调试

浏览器访问系统首页，通过首页展示出来的数据及内容判断系统的各个功能是否正常，系统采用前后端分离开发，这样有助于修改维护系统测试中出现的问题。

第八章 系统功能测试报告

8.1 前端业务逻辑测试

将前端程序运行环境部署在服务器上，通过模拟测试用户访问系统页面等功能请求，判断请求系统的流畅性及运行效果，将运行效果与设计效果进行比对。将突出的问题进行修改完善。我们将重页面配色优化，页面渲染时耗、文件加载耗时、功能模块衔接、业务的完整性等多方面进行模拟用户测试。

8.1.1 页面渲染测试

用户访问系统，浏览器将请求服务器，服务器将响应数据返回给浏览器，在这过程中会存在访问响应加载耗时，根据耗时长短判断用户使用体验如表 8.1.1 所示：

表 8.1.1 页面渲染时耗

页面	DOMContentLoaded	Load	Finish	Requests
/index.html	607ms	1.91s	2.06s	28
/sList.html	703ms	2.10s	2.29	23
/add.html	662ms	806ms	946ms	11
/dataS.html	547ms	808ms	907ms	18
/present.html	2.45s	2.94s	4.27s	299
/admin.html	623ms	603ms	900ms	9
/admin/conn.html	766ms	869ms	971ms	17

根据表中数据分析得出页面的 Requests Number 将会影响到页面数据的加载与渲染，我们通过 CDN 加速提高数据响应耗时及页面渲染耗时，使每个页面渲染加载都低于 3s 的使用体验。

8.2 API 接口高并发测试

一个完整的系统由前端模块和后端模块组成，后端主要包括后端业务逻辑代码、数据库、服务器、缓存等组成，前端的数据操作都会请求后端模块，所以怎么提供、维护、设计的后端接口是我们考虑的重点方向。好的接口设计会保障系统的高并发、高可用、高吞吐量等特点。测试将以一个线程以(1 second/1 request)的设定模拟用户使用系统接口，后端模拟请求结果如下表 8.2 所示：

表 8.2 接口吞吐量数据

Label	Samples	Average	Min	Max	Error	throughput	Received KB/S
getStudioList	1728	4279	189	100524	0.00%	16.37853	30.09
findById	1682	781	46	19574	0.00%	16.16298	5.79
redisToken	1655	551	45	21005	0.00%	16.00518	5.28
deleteStudio	1643	564	46	19360	0.00%	16.32673	3.68
total	6717	1561	45	100524	0.00%	63.66523	46.5

测试结果表中 Label (测试项目)、Sample (为测试样本)、Average (平均数)、Min (最小值)、Max (最大值)、Error (错误占比)、throughput (吞吐量)、Received KB/S (接收), 分析数据可以得到系统后端接口可以应对高并发等需求。

8.3 系统综合测试

系统综合测试提现了整个系统的流畅性和用户使用体验, 我们将通过访问问卷的形式得出测试报告。根据用户反映的实际问题进行系统的修改和完善。我们会根据用户访问系统的部分功能的响应时间来判断功能的体验是否满足需求, 如响应时间长, 将从前端模块与后端模块进行优化和修改, 以此提高用户使用体验。如图 8.3 系统部分功能响应时间图所示。

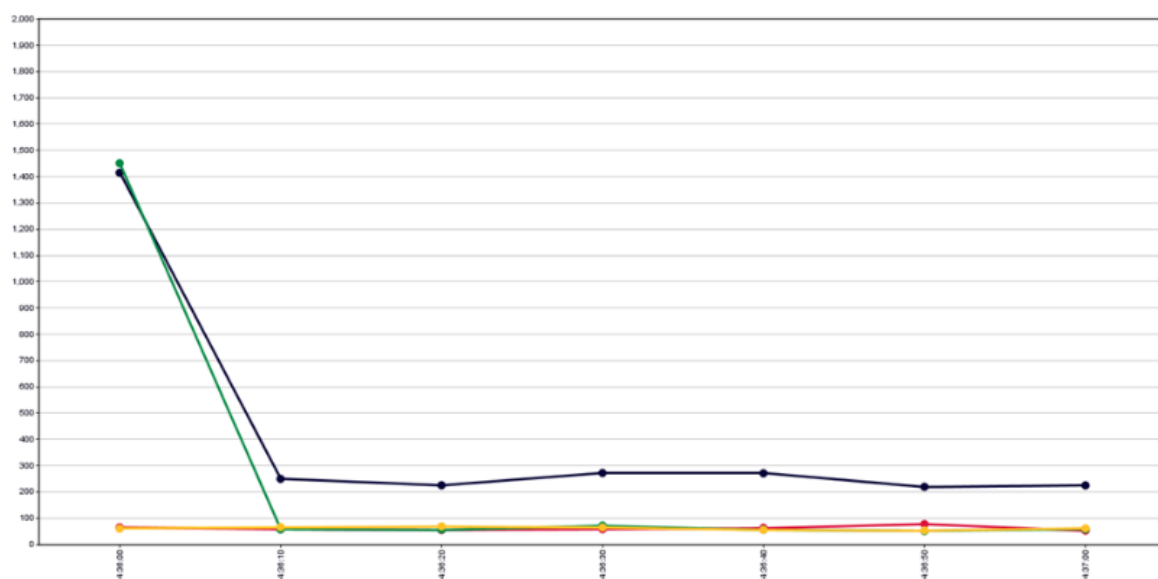


图 8.3 系统响应时间图

结 语

四川科技职业学院学生工作室官网系统通过前后端分离开发，提高了系统的维护性，以及业务逻辑的处理能力。分离开发将大大提高前端模块及后端模块的代码质量以及对应功能的技术优化和改进，这样使对应的模块更加稳定和更好的使用体验。随着计算机互联网技术的发展，Web 系统、手机软件、小程序、硬件系统都会慢慢采用前后端开发，这样使对应的系统能得到更好的维护和创新设计更好，更方便的开发技术等。

致 谢

在本课题的设计开发可写作中首先感谢导师的指导及修改建议，从我们最初的定题，到课题的资料收集，写作、修改、论文定稿，他都给了我耐心的指导和无私的帮助，在导师的指导下我们更好、更快、更优的完成了课题的设计和论文的编写。随着毕业设计论文答辩的谢幕，毕业已然来临。同时，感谢我大学所有学校学院领导、任课老师、同学在这三年来给自己的指导和帮助，是在你们的帮助下我学习了更多的专业知识和自身综合能力的提高，使自己有一个丰富的大学生活。正是由于他们，自己才能在各方面取得显著的进步，在此我将感谢你们。

参考文献

- [1] 陈瑞.基于 Spring Boot 高并发 Java Web 开发模式[J].电脑编程技巧与维护,2019(04):27-30.
- [2] 迟殿委.前后端分离的 Web 架构解决方案[J].智慧工厂,2019(06):37-38.
- [3] 何军,陈倩怡.Vue、Spring Boot、MyBatis 开发消费管理系统[J].电脑编程技巧与维护,2019(02):87-88+102.
- [4] 吕宇琛.SpringBoot 框架在 web 应用开发中的探讨[J].科技创新导报,2018,15(08):168+173.
- [5] 张峰.应用 SpringBoot 改变 web 应用开发模式[J].科技创新与应用,2017(23):193-194.
- [6] 龚电花.基于网站制作的 Web 前端开发技术与优化 [J]. 信息通信 ,2018(07):286-288.
- [7] 张德法. Web 前端开发技术与优化工作研究[J]. 科技与信息,2019(09).
- [8] 陈捷.基于网站制作的 Web 前端开发技术与优化 [J].现代信息技 ,2019(08):111-112.
- [9] 刘长建.企业防御 DDoS 攻击需要多管齐下[J].计算机与网络,2017,43(14):54-55.
- [10] 夏刚.互联网信息安全加固技术探讨——HTTPS 技术介绍与应用[J].中国金融电脑,2018(06):65-67.