

毕 业 论 文

**题 目 基于BS的OA系统的设计与实现**

**英文题目 Design and Implementation of OA System Based on BS**

**学生姓名**： 程炜 **申请学位门类**： 工学

**学 号**： 201520182123

**专 业：** 软件工程(国际合作办学)

**学 院**： 软件学院

**指导教师：** 李丽华 **职称：** 讲师

**二0 一九 年 六 月 二十 日**

**作 者 声 明**

本人以信誉郑重声明：所呈交的学位毕业设计（论文），是本人在指导教师指导下由本人独立撰写完成的，没有剽窃、抄袭、造假等违反道德、学术规范和其他侵权行为。文中引用他人的文献、数据、图件、资料均已明确标注出，不包含他人成果及为获得东华理工大学或其他教育机构的学位或证书而使用过的材料。对本设计（论文）的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本毕业设计（论文）引起的法律结果完全由本人承担。

本毕业设计（论文）成果归东华理工大学所有。

特此声明。

毕业设计（论文）作者（签字）：

签字日期： 年 月 日

本人声明：该学位论文是本人指导学生完成的研究成果，已经审阅过论文的全部内容，并能够保证题目、关键词、摘要部分中英文内容的一致性和准确性。

学位论文指导教师签名：

年 月 日

**基于BS的OA系统的设计与实现**

**程炜**

*Design and implementation of Teaching Monitoring System*

*Cheng wei*

摘 要

随着信息技术的迅速发展，计算机网络已经普及到各个行业。而在如何充分利用计算机网络配合校园对教育活动的管理并做出更好的决策先得尤为重要。随着教育的不断扩展和深入，对教学过程中的出现的问题能够的及时反应并监控的平台需求越来越大。在日新月异的信息时代，提供一个教学监控的管理平台会大大提高教学质量的监管水平。

针对东华理工大学的需求，为此设计一个教学监控系统。系统采用Spring cloud微服务架构+Springboot+Mybatis+Shiro,前端页面使用HTML展示，使用Springboot框架技术完成业务逻辑层的业务逻辑，采用Mybatis框架技术实现数据访问[2]，构建了用户界面层、业务逻辑层、数据访问层的三层模式。本系统由消息管理、出勤管理、请假管理、评价管理、问卷调查以及系统管理六个模块组成。

本文首先从Springcloud微服务架构特点和教学监控系统现状出发，然后重点介绍了本教学监控系统的各个模块的设计和实现方法，并展示了运行结果，最后对整个系统进行了分析和总结。

关键字：教学监控系统；微服务；Springcloud

**Abstract**

With the rapid development of information technology, computer network has been widely used in various industries. How to make full use of computer network to cooperate with campus management of educational activities and make better decisions is particularly important. With the continuous expansion and deepening of education, there is a growing demand for timely response and monitoring of the problems in the teaching process. In the information age, providing a management

platform for teaching monitoring will greatly improve teaching results.

According to the demand of Donghua University of Technology, a teaching monitoring system is designed for this purpose. The system uses Springcloud micro-service architecture Springboot Mybatis Shiro, front-end page display using HTML, uses Springboot framework technology to complete business logic layer business logic, uses Mybatis framework technology to realize data access, constructs user interface layer, business logic layer. The three-tier pattern of the data access layer. The system consists of six modules: message management, attendance management, leave management, evaluation management, questionnaire survey and

system management.

This paper starts from the characteristics of Springcloud micro-service architecture and the current situation of teaching monitoring system, then introduces the design and implementation of each module of the teaching monitoring system, and shows the running results. Finally, the whole system is analyzed and summarized.

**Key words**:Teaching Monitoring System; Micro-service; SpringCloud

目 录

[第1章 绪论 1](#_Toc15626)

[1.1 项目开发背景以及意义 1](#_Toc22401)

[1.2 教学监控的研究现状 1](#_Toc25715)

[1.3 主要研究内容 2](#_Toc29580)

[1.4 论文章节安排 2](#_Toc935)

[第2章 系统相关技术概述 3](#_Toc15765)

[2.1 微服务架构 3](#_Toc10687)

[2.1.1 微服务介绍 3](#_Toc19794)

[2.1.2 微服务具备的特性 3](#_Toc18580)

[2.1.3 微服务的优点 3](#_Toc2637)

[2.1.4 微服务带来的挑战 4](#_Toc14932)

[2.2 springcloud介绍 4](#_Toc23456)

[2.3 springboot介绍 4](#_Toc28870)

[2.4 mybatis 4](#_Toc24228)

[2.5 shiro 安全架构 5](#_Toc975)

[2.6 消息中间件：activemq 5](#_Toc9193)

[2.6.1 消息中间件介绍 6](#_Toc16628)

[2.6.2 activemq 6](#_Toc11580)

[2.7 docker容器技术 6](#_Toc11228)

[2.8 其他技术 6](#_Toc31872)

[2.9 本章小结 7](#_Toc8246)

[第3章 系统需求分析 8](#_Toc17059)

[3.1 业务需求分析 8](#_Toc1890)

[3.1.1 总体需求分析 8](#_Toc30462)

[3.1.2 角色定义 8](#_Toc26833)

[3.1.3 业务流程 10](#_Toc8918)

[3.2 功能需求分析 10](#_Toc4603)

[3.3 非功能性需求分析 13](#_Toc18915)

[3.3.1 系统的性能需求 13](#_Toc15185)

[3.3.2 系统安全性要求 14](#_Toc24579)

[3.4 本章小结 14](#_Toc31143)

[第4章 系统总体设计 15](#_Toc32128)

[4.1 软件架构设计 15](#_Toc2009)

[4.2 总体功能模块设计 15](#_Toc20115)

[4.3 数据库设计 16](#_Toc4801)

[4.3.1 逻辑结构设计 16](#_Toc24596)

[4.3.2 消息管理数据库设计 17](#_Toc16824)

[4.3.3 请假管理数据库的设计 18](#_Toc14952)

[4.3.4 出勤管理数据库的设计 19](#_Toc18482)

[4.3.5 问卷调查数据库设计 21](#_Toc1333)

[4.3.6 评价管理数据库设计 23](#_Toc28988)

[4.3.7 系统管理数据库设计 25](#_Toc355)

[4.4 本章小结 27](#_Toc23524)

[第5章 系统详细设计与实现 28](#_Toc21617)

[5.1 系统的开发环境 28](#_Toc6091)

[5.2 系统各模块的详细设计与实现 28](#_Toc14122)

[5.2.1 消息管理模块的详细设计与实现 28](#_Toc2938)

[5.2.2 请假管理模块的详细设计与实现 37](#_Toc27446)

[5.2.3 出勤管理的详细设计与实现 42](#_Toc6703)

[5.2.4 评价管理、问卷调查的设计与实现 44](#_Toc13864)

[5.2.5 系统管理的详细设计与实现 48](#_Toc27966)

[5.4 本章小结 51](#_Toc21999)

[第6章 总结 52](#_Toc9522)

[参考文献 53](#_Toc16853)

[致 谢 54](#_Toc6490)

# 

# 

# 第1章 绪论

## 1.1 项目开发背景以及意义

随着信息技术的高速发展，计算机网络已经成为各个行业的至关重要的手段，在这种浪潮下，如何搭建计算机网络并将其恰如其分地运用在教学活动中显得尤为重要。如今有些学校已经建立起校园网络平台，通过提供校园网络平台提高教学质量水平和管理水平日益彰显其重要性。

教学水平是用来衡量一个学校好坏的标准，而教学监控为教学水平的提高提供了重要保证，因此对教学活动进行系统的监控意义重大。教学监控其实就是对教学状态进行监控，对学生学校状态进行监控，对教学效果进行评价，即学生、教师、校领导等多层次人员从各个角度对教学过程的教学管理、任课老师教学质量、学生学习状态及效果进行监控，从而实现对全校整体教风、学风、教学水平的监控，为预防可预见的问题作出及时调整提供较为客观、全面性的依据和分析意见[3]。

现今，很多学校对于教学质量的监控还不是很全面，在问卷调查上仍然采用传统的问卷调查，严重的浪费人力、物力，因此设计基于网络平台的教学监控系统可以大大地提高效率和监控力度。此外，教学监控系统可以充分利用网络资源并提供实施有效的统计数据和结果，方便学校管理者轻松掌握学校的教学状态，并作出有效的决策，从而提高学校教学水平和教学管理水平。

## 1.2 教学监控的研究现状

教学监控系统是指对教学质量进行评价，即学生、授课老师、辅导员、领导等多层次人员从多个角度对学生的学习情况及效果进行监控、对任课老师各课程的教学状态及效果。通过对数据的收集及对不同维度进行分类处理，以不同形式的数据展示给学生、老师、辅导员等不同角色，实现对全校整体教风、学风、教学质量的监控。

教学监控系统包含以下模块：消息管理、出勤管理、请假管理、问卷调查、评价管理以及系统管理等模块。在教学过程中出现的问题反应在学生的出勤率、学生的请假情况以及教学质量评价上。此外，教学监控系统提供的消息管理提供了学生与老师、学生与辅导员、领导跟教师等多类角色之间的交流机制，它能够实时收发消息，以及消息提醒。

教学监控系统是一个参与角色较多、复杂的动态系统，对于不同角色在业务上处理的差异性增加了系统的复杂性，此外，在系统的组织结构上、构成系

统环节的连接上也体现了系统的复杂性。

目前，国内校园在进行问卷调查时还是传统的进行纸质问卷调查，严重浪费人力物力，还容易造成信息的丢失和不完整性，对问卷回答的统计过度浪费人力而且麻烦。虽然有些对教学水平通过网上进行统计，但是针对性差，对教学水平的监控效果差。所以，开发一个具有对教学水平有针对性的、操作简单的信息交流平台十分必要。

## 1.3 主要研究内容

此次开发的教学监控系统，基于东华理工大学目前的教学状况，结合学校的教学流程、业务来分析并进行设计。本次项目选用Springcloud构建微服务，整合Springboot实现的B/S架构的分布式应用系统[4]，最后由docker容器对应用进行移植管理。

本系统实现了如下一些教学监控系统的相关功能，比如出勤管理，请假管理，问卷调查，评价管理，消息管理以及系统管理。

## 1.4论文章节安排

本文分为七章，论文首先简单介绍了教学监控系统的定义，以及教学监控系统的发展与现状。随后对现今学校的教学监控系统进行简单分析并指出对教学监控系统的开发的必要性，并对平台的架构的选择做出了初步的定位，引出了论文所做的主要工作内容。紧接着论文简单概述了毕业设计项目中用到的各项技术，并针对教学监控系统提出的解决方案。然后针对各个模块进行详细的设计阐述，以及运行结果展示。

论文具体安排如下：

第一章 简单介绍了教学监控系统的目前状况和项目的背景。

第二章 概述系统中所使用的软件体系架构和框架以及各项技术。

第三章 介绍教学监控系统的需求分析。

第四章 介绍教学监控系统的概要设计与数据库设计。

第五章 详细介绍了教学监控系统的各个模块的详细设计与实现。

第六章 最后论文总结了该设计所做的工作。

# 第2章 系统相关技术概述

## 教学监控系统采用Springcloud架构构建微服务，让每个服务都独立的运行在不同的主机上，在数据库层面使用Mybatis进行数据的管理，为了保证不同角色各司其职，本系统使用Shiro来进行权限管理,由redis来共享认证用户会话信息，实现分布式系统的单点登入[5]。由消息中间件ActiveMq来实时接收消息通知。

## 2.1 微服务架构

### 2.1.1 微服务介绍

微服务，顾名思义，微服务得从两个方面去理解，什么是"微"、什么是"服务"， 微 狭义来讲就是体积小。 而所谓服务，一定要区别于系统，服务一个或者一组相对较小且独立的功能单元，是用户可以感知最小功能集。

微服务最早由Martin Fowler与James Lewis于2014年共同提出，微服务架构风格是一种使用一套小服务来开发单个应用的方式途径，每个服务运行在自己的进程中，并使用轻量级机制通信，通常是HTTP API，这些服务基于业务能力构建，并能够通过自动化部署机制来独立部署，这些服务使用不同的编程语言实现，以及不同数据存储技术，并保持最低限度的集中式管理[6]。

简而言之，微服务架构风格这种开发方法，是以开发一组小型服务的方式来开发一个独立的应用系统的。

### 2.1.2 微服务具备的特性

(1)每个微服务可独立运行在自己的进程里；

(2)一系列独立运行的微服务共同构建了整个系统；

(3)每个服务为独立的业务开发，一个微服务一般完成某个特定的功能，比如：消息管理，用户管理等；

(4)微服务之间 通过一些轻量的通信机制进行通信，例如通过REST API或者RPC的方式进行调用。

### 2.1.3 微服务的优点

(1)易于开发和维护；

(2)启动较快；

(3)局部修改容易部署；

(4)按需伸缩。

### 2.1.4 微服务带来的挑战

(1)运维要求较高；

(2)分布式的复杂性；

(3)接口调整成本高；

(4)重复劳动。

## 2.2 Springcloud介绍

Springcloud是Spring家族的一个产品，它是一系列框架的有序集合。Spring Cloud是一个基于Spring Boot实现的云应用开发工具，它为基于JVM的云应用开发中的配置管理、服务发现、断路器、智能路由、微代理、控制总线、全局锁、决策竞选、分布式会话和集群状态管理等操作提供了一种简单的开发方式[7]。它利用Spring Boot的开发便利性巧妙地简化了[分布式系统](https://baike.sogou.com/v5697460.htm" \t "https://baike.sogou.com/_blank)基础设施的开发，如服务发现注册、配置中心、消息总线、负载均衡、断路器、数据监控等，都可以用Spring Boot的开发风格做到[一键启动](https://baike.sogou.com/v75063267.htm" \t "https://baike.sogou.com/_blank)和部署[8]。Spring并没有重复制造轮子，它只是将目前各家公司开发的比较成熟、经得起实际考验的服务框架组合起来，通过Spring Boot风格进行再封装屏蔽掉了复杂的配置和实现原理，最终给开发者留出了一套简单易懂、易部署和易维护的分布式系统开发[工具包](https://baike.sogou.com/v57704855.htm" \t "https://baike.sogou.com/_blank)[9]。

## 2.3 Springboot介绍

Springboot同样作为Spirng家族下的一个产品，用来简化新Spring应用的初始搭建以及开发过程。该框架使用了特定的方式来进行配置，从而使开发人员不再需要定义样板化的配置[10]。在以前的spring项目中，都会面对大量繁琐的配置，使用的时候基本上都是大量的复制粘贴。Springboot为开发笨重的J2EE提供一站式解决方案，约定大于配置，去繁从简，能够快速的创建一个独立的，产品级别的应用[11]。

## 2.4 Mybatis

Mybatis是一个可以自定义SQL、存储过程和高级映射的半自动化的ORM框架，它原是Apache的一个开源项目Ibatis, 2010年6月这个项目由Apache Software Foundation 迁移到了Google Code，随着开发团队转投Google Code旗下， iBatis3.x正式更名为MyBatis[12]。

## 2.5 Shiro 安全架构

Shiro是Apache下的一个开源项目，是一个强大的简单易用的Java安全框架，主要用来更便捷的认证，授权，加密，会话管理。它可以非常容易的开发出足够好的应用，其不仅可以用在JavaSE 环境，也可以用在 JavaEE 环境 Shiro首要的和最重要的目标就是容易使用并且容易理解[13]。

Shiro是一个有许多特性的全面的安全框架，下面这幅图可以了解Shiro的功能特性:

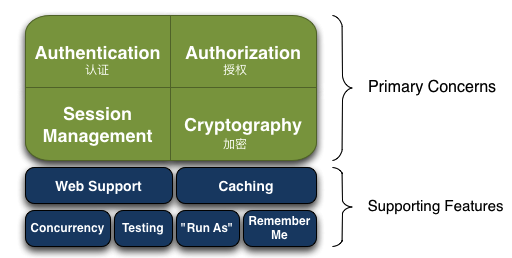


图2-1 Shiro框架图

Authentication：身份认证/登录，验证用户是不是拥有相应的身份；

Authorization：授权，即权限验证，验证某个已认证的用户是否拥有某个权限；即判断用户是否能进行什么操作，如：验证某个用户是否拥有某个角色。或者细粒度的验证某个用户对某个资源是否具有某个权限；  
 Session Manager：会话管理，即用户登录后就是一次会话，在没有退出之前，它的所有信息都在会话中；会话可以是普通 JavaSE 环境，也可以是 Web 环境的；  
 Cryptography：加密，保护数据的安全性，如密码加密存储到数据库，而不是明文存储；  
 Web Support：Web 支持，可以非常容易的集成到Web 环境；  
 Caching：缓存，比如用户登录后，其用户信息、拥有的角色/权限不必每次去查，这样可以提高效率；  
 Concurrency：Shiro 支持多线程应用的并发验证，即如在一个线程中开启另一个线程，能把权限自动传播过去；  
 Testing：提供测试支持；  
 Run As：允许一个用户假装为另一个用户（如果他们允许）的身份进行访问；  
 Remember Me：记住我，这个是非常常见的功能，即一次登录后，下次再来的话不用登录了。

## 2.6 消息中间件：Activemq

### 2.6.1 消息中间件介绍

消息中间件（Message Orient middleware）主要的用途就是将数据从一个应用程序传送到另一个应用程序，或者从软件的一个模块传送到另一个模块。负责建立网络通信的通道，进行数据的可靠传送，保证数据的不重发，不丢失，此外，它能够实现跨平台操作。

### 2.6.2 Activemq

Activemq 是Apache下的一个易于使用的消息中间件，作为最流行的，能力强劲的开源消息总线，从设计上保证了高性能的集群，客户端-服务器，点对点。它支持JMS2.1（全称是Java Message Service）和J2EE1.4规范[14]。支持多种语言和协议编写客户端，例如:Java,C,C++等等。

## 2.7 Docker容器技术

Docker是一个开源的引擎，可以轻松的为任何应用创建一个轻量级的、可移植的、自给自足的容器。让开发者可以打包他们的应用以及依赖包到一个可移植的容器中，然后发布到任何流行的Linux机器上，也可以实现虚拟化，容器是完全使用沙箱机制，互相之间不会有任何接口[15]。

## 2.8 其他技术

(1)JavaScript技术简介

JavaScript是用于浏览器的第一种具有通用目的、动态的客户端脚本语言。JavaScript是一门基于原型、函数先行的语言，是一门多范式的语言，它支持[面向对象编程](https://baike.sogou.com/v458310.htm" \t "https://baike.sogou.com/_blank)，[命令式](https://baike.sogou.com/v57737969.htm" \t "https://baike.sogou.com/_blank)编程，以及函数式编程。由Netscape于1995年首先提出。JavaScript不仅是一种基于对象和事件驱动并具有相对安全性的客户端脚本语言，而且也是一种广泛用于客户端Web开发的脚本语言。JavaScript语言在一定程度上弥补了HTML固有的缺陷，使得版本平淡的静态页面添加了许多动感。JavaScript虽然与java有很多相似性，但这两门编程语言从设计之初就有很大的不同。JavaScript的语言设计主要受到了Self（一种基于原型的编程语言）和Scheme（一门函数式编程语言）的影响。在语法结构上它又与C语言有很多相似（例如if条件语句、while循环、switch语句、do-while循环等）。JavaScript语言和Java、C语言等最大的一个区别就是编译方式不一样。Java语言以及C语言等高级语言的源代码在执行之前，必须要经过编译。而JavaScript是一种解释型的脚本语言，其源代码无须编译即可直接解释执行。

(2)Jquery技术简介

Jquery是一个快速、简洁的[JavaScript框架](https://baike.sogou.com/v143216894.htm" \t "https://baike.sogou.com/_blank)，也是一个优秀的JavaScript库，是由John Resig 创建于200612月的开源项目，Jquery设计的宗旨是”write Less,Do More”，即倡导写更少的代码，做更多的事。Jquery凭借简洁的语法和跨平台的兼容性，极大地简化了JavaScript开发人员遍历HTML文档操作DOM处理事件、执行动画和开发Ajax的操作，其独特而又优雅的代码风格改变了JavaScript程序员的设计思路和编写程序的方式。Jquery的核心特征可以总结为：具有独特的链式语法和短小清晰的多功能接口；具有高效灵活的[css选择器](https://baike.sogou.com/v54106580.htm" \t "https://baike.sogou.com/_blank)进行扩展；拥有便捷的插件扩展机制和丰富的插件[16]。

## 2.9 本章小结

本章节简单介绍系统所使用的Springcloud微服务架构，对分布式中应用与应用间的交流提供高并发支持的消息中间以及Shiro安全框架进行了简单介绍。以及对应用中数据持久化的Orm框架Mybatis,应用移植引擎Docker等技术展开介绍，并概述了本系统运用到的其他技术。

# 第3章 系统需求分析

在软件工程中，需求分析指的是在一个项目或者工程开始前对工程中所需要完成的工作、目的，工程的耗时以及投入的人力、物力进行定义工作。需求分析是软件工程中的一个关键过程。在这个过程中，系统分析师和软件工程师确定顾客的需要。只有在确定了这些需要后才能开始分析和寻求解决办法。

## 3.1 业务需求分析

### 3.1.1 总体需求分析

教学监控系统，是针对东华理工大学教学二开发的系统，主要用来监控学生的出勤、请假，然后通过教学质量的评价以及问卷调查来追踪教学过程中存在的问题。对收集的数据从不同维度进行处理，最后呈现出结果，能够让学校领导跟任课老师以及辅导员及时了解教学状况并迅速作出调整。本教学监控系统从以下几方面进行数据的收集：

(1)通过学生的出勤状况可以了解学生的上课情况，随时追踪学生的出勤状况。

(2)通过学生请假的状况可以得知学生的精神状态、学习面貌

(3)通过学生对任课老师教学质量的评价可以了解到老师的教学状态并及时对存在的教学问题作出调整

(4)通过对学生调查问卷的回答情况可以获取学生的思想情况，以及对学校的教学的建议从而指定更适合学生发展的教学方案。

### 3.1.2 角色定义

本系统涉及到学生的出勤、请假、评价等。参与的角色有普通用户(user),学生(student),老师(teacher),辅导员(assistant),领导(leader),管理员(admin)。

(1)普通用户(user)

描述：普通用户可以进行查看消息、发送消息，管理自己的消息。

(2)学生(student)

描述：学生可以对消息的收发进行管理，可以进行请假的发起以及查看请假的审批情况，可以及时查看自己的出勤状况，以及对教学质量的评价、回答调查问卷。

(3)老师(teacher)

描述：老师可以通过消息管理查看消息，也可以跟学生、老师进行通信，可以对上课班级进行考勤登记以及查看考勤，对缺勤次数达到预警的同学进行按时上课提醒并通知辅导员该同学的出勤情况。老师可以查看各班级对自己各课程的教学质量评价，同时查收辅导员发过来的批准的学生请假条。

(4)辅导员(assistant)

描述：辅导员可以管理自己的消息，查看管理的班级的出勤情况，审核学生的请假请求，查看自己已受理以及待受理的请假请求。可以创建调查问卷并查看回答情况，一键生成教学评价问卷并查看监控学生评价完成情况。

(5)领导(leader)

描述：领导可以管理消息的收发，查看全校的出勤情况并通知缺勤率较高的班级所在的辅导员，可以查看全校的请假情况以及教师的教学质量评价情况。

(6)管理员(admin)

描述：管理员可以查看用户，为用户添加或删除角色，可以创建角色并为角色进行职责的分配，以及职责的创建与查看、修改职责。

### 3.1.3 业务流程

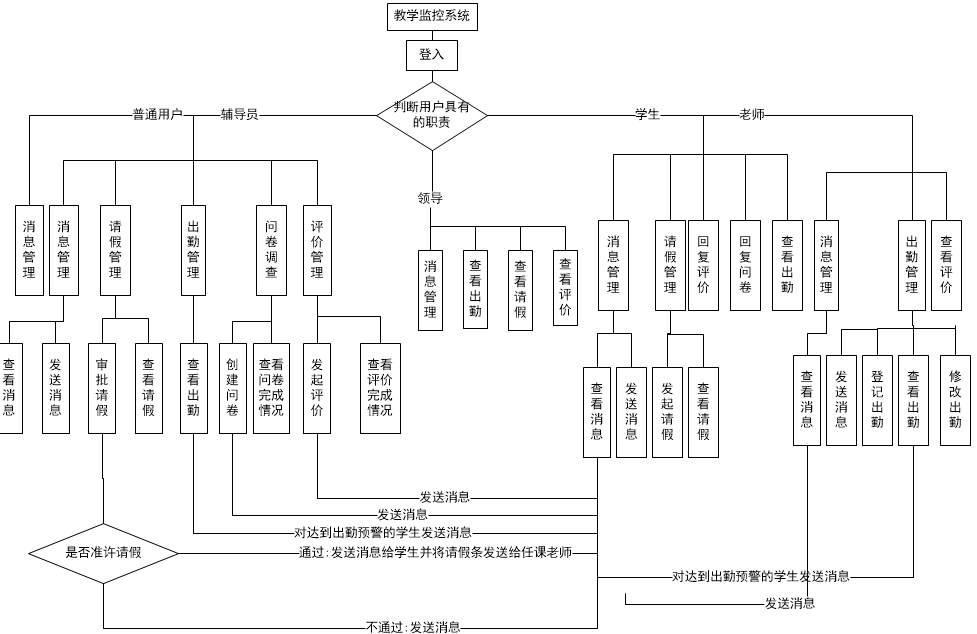


图 3-1 系统业务流程图

用户登录教学监控系统的时候，系统会获取用户所具有的角色再通过角色获取拥有的职责，之后加载对应的模块呈现在登入首页中，一般学生用户拥有学生角色，学生角色拥有消息管理、出勤管理、请假管理等，而授课老师用户在只有授课老师角色的情况下拥有消息、考勤记录、评价管理，没有请假管理。具体流程如图3-1所示。

## 3.2 功能需求分析

消息管理：用户可以发送私人消息或者群发消息、接收实时消息并对消息进行删除管理，有利于老师跟学生、学生跟辅导员、领导跟老师之间的交流。在出勤、评价、请假都设有消息提醒功能，用来及时通知学生或老师。

出勤管理：系统允许登记出勤并统计学生的出勤情况，并支持老师角色对出勤信息的修改，通过出勤预警提醒及时汇报辅导员、学生出勤情况，学生也能及时查看自己的出勤情况。

请假管理：系统会登记学生的请假记录，然后通知对应的辅导员，辅导员对学生的请假进行批准回复，对通过的请假请求系统会自动计算在这请假区间出现的上课信息，并将请假条以及辅导员批准会发一同发送给任课老师。学生可以实时接收请假更新动态及时查看自己的请假结果。

问卷调查：系统会记录辅导员创建的问卷调查，并发送至对应班级的学生，学生可以回答问卷调查，系统会统计好学生的回答情况供老师查看。

评价管理：系统允许辅导员对自己管理的班级发起教学评价，系统会自动计算出班级该学期所有的课程并生成评价问卷发送给学生，并将学生的回答情况统计呈现给老师、领导、辅导员查看。

图3-2至3-7是上述模块功能用例图。

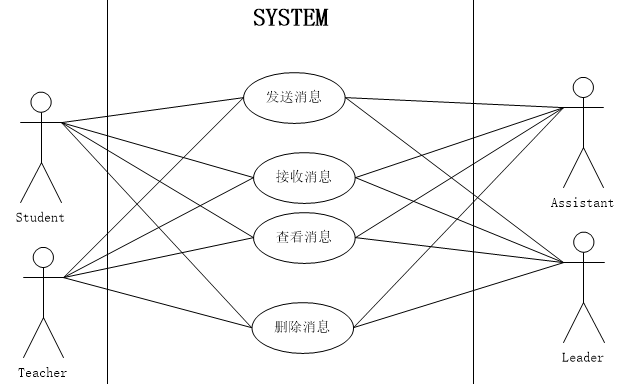


图3-2 消息管理用例图

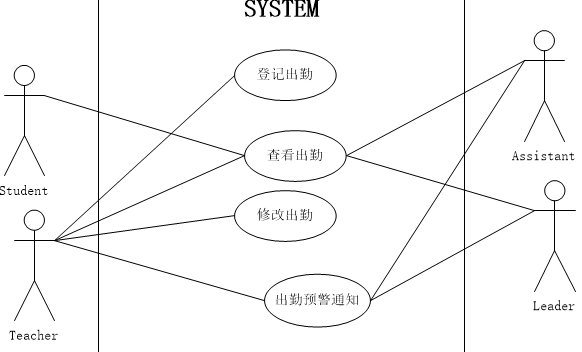


图3-3 出勤管理用例图

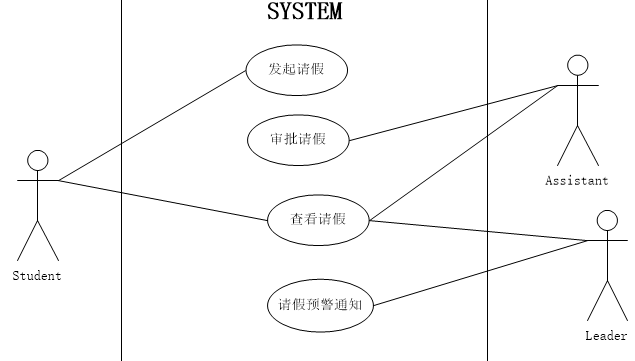


图3-4 请假管理用例图

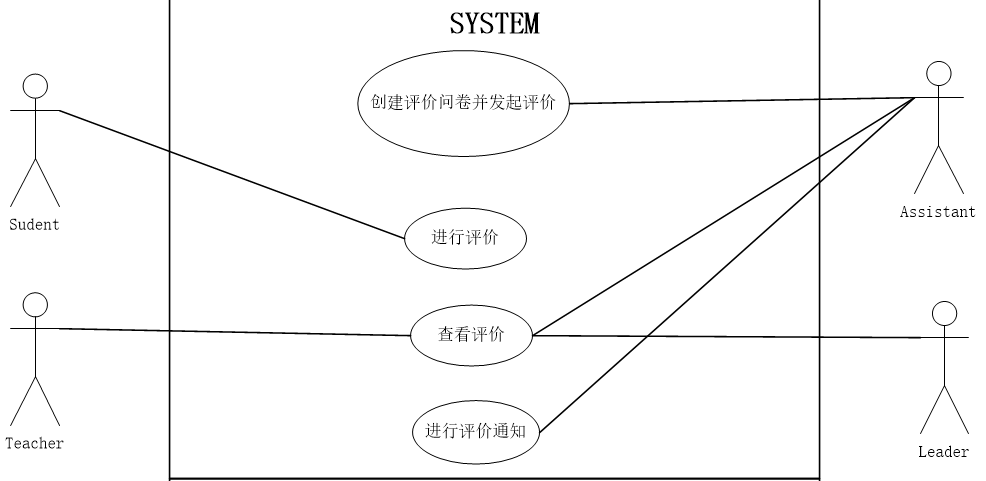


图3-5 评价管理用例图

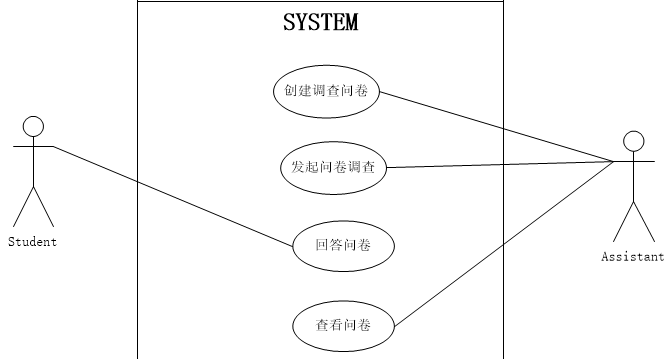


图3-6 问卷调查用例图

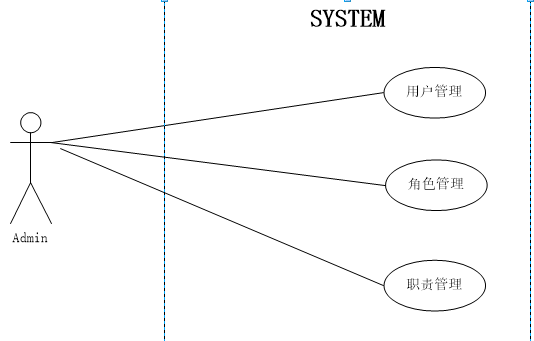


图3-7 系统管理用例图

## 3.3 非功能性需求分析

### 3.3.1 系统的性能需求

(1)要求系统页面风格美观，融合学校办公室和网站的元素，操作起来简单、实用。

(2)要求页面数据加载及保存更新的时间短，响应时间快。

(3)要求数据的精确性，完整性以及不重复性。

(4)系统的容错性要强，当系统出现故障，要求数据能完好无损的保存下载，并且系统恢复速度快。

(5)支持高并发，多人同时在线使用

(6)系统上手起来快，简单易学，用户不需要懂技术方面的知识，只需要懂得基本的计算机操作。

(7)可维护性；出现故障时，可提供及时的系统修复

(8)教学监控系统要求有较强的适应性，可以进行拓展功能，并能适应开发计划变化。要求系统能有效恢复和再启动等。

### 3.3.2 系统安全性要求

本系统要求对用户的个人信息严格保护；对用户的操作进行权限的检查，对未授予的权限系统采取丢弃操作。

## 3.4 本章小结

本章主要介绍了教学监控系统的功能性需求以及非功能性需求。以及系统整体需求，系统主要分为6个功能模块：消息管理、请假管理、出勤管理、问卷调查、评价管理以及系统管理，详细的介绍了系统的各参与角色以及角色所拥有的功能，并着重介绍了消息管理、出勤管理、请假管理、评价管理以及问卷调查等5个模块的需求并且使用了用例来描述。

# 第4章 系统总体设计

## 4.1 软件架构设计

本系统是基于Springcloud微服务架构系统,作为分布式系统，本系统由一个个独立的子系统组成，采用B/S架构，每个用户直接使用浏览器可直接访问使用。为了降低耦合度，提高系统可移植性、提高开发效率，每个子系统都采用前后端分离，系统与系统间通过HTTP协议来进行通信，每个系统最终是运行在docker容器中，每个docker容器互不影响。

## 4.2 总体功能模块设计

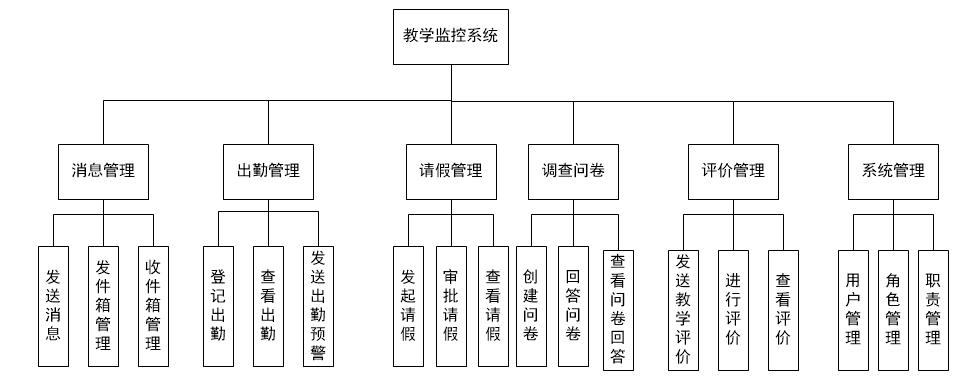


图4-1 系统总体模块功能图

如图4-1所示，教学监控系统主要由六个模块组成，分别是：消息管理，出勤管理，请假管理，调查问卷，评价管理以及系统管理。下面对每个的模块的功能做具体介绍：

(1)消息管理：消息管理提供了学生与学生、老师、辅导员之间的交流机制，通过消息管理可以及时掌握学生的状况，实现教学质量的监控，消息管理的收发都是实时的，学生发起了请假会立刻通知辅导员，请假动态的更新、辅导员问卷调查的发起、评价的发起、缺勤预警通知都会发送消息给学生。该模块与请假管理、出勤管理、问卷调查、请假管理密切相关。此外，对于辅导员，可以进行班级群发消息通知管理的班级来通知学校下达的通知，比如端午放假通知。

(2)请假管理：由学生填写请假单发起请假，系统会将学生的请假请求及请假通知发送到他们的辅导员，辅导员收到请假通知并审批请假请求，审批通过的系统会将请假单以及辅导员的审批回复发送给请假区间上课的任课老师，收到请假条的任课老师将学生的出勤状态修改为请假状态，方便教学考勤的进行。学生可以查看自己请假的审批结果，老师也可查看所有的请假表，领导查看全校的最近半个月的请假情况，请假过多的班级，领导可以发送预警通知给对应的辅导员，方便及时监控学生学习面貌。

(3)出勤管理：由任课老师登记考勤，系统会统计每个班级每个学生的考勤，任课老师可以查看每门课每个班级学生的出勤情况，缺勤次数达到预警的学生可发送缺勤预警消息通知学生跟辅导员，方便辅导员及时了解学生的上课情况。

(4)问卷调查：由辅导员创建调查问卷，针对不同的人群开放不同的问卷，系统自动统计学生的问卷回答以及完成情况。

(5)评价管理：辅导员对管理的班级一键生成请假问卷，系统统计学生的问卷回答情况及完成情况，任课老师可以查看自己课程的教学质量评价，领导可以查看所有老师的教学质量的满意度情况。

(5)系统管理：主要实现用户到角色，角色到职责的管理，可以为用户分配角色，创建角色，为角色分配职责以及职责的管理。

## 4.3 数据库设计

### 4.3.1 逻辑结构设计

问卷调查：一张问卷调查由问卷基本信息跟多个问题组成，而每个问题由问题跟多个答案组成，一份问卷可供多个学生回答，而一个学生可以回答多份问卷。

评价管理：每张评价问卷由多个课程评价问题组成，每个评价问题有多个答案选项，一张评价问卷可以供多个学生回答，每个学生只能回答一份评价问卷。

出勤管理：一门课程对应多个班级，而一个班级对应多张出勤列表，出勤列表有多条出勤记录。

系统管理：一个用户可以有多个角色，一个角色可以授予给多个用户，一个角色有多个职责，一个职责可以分配给多个角色。

请假管理：每个学生可以请假多次

消息管理：每个用户可以发送或接收多条消息，一条发送消息可以有一条接收消息，也可以有多条接收消息，一条接收消息只有一条发送消息。

具体E-R图如图4-2：

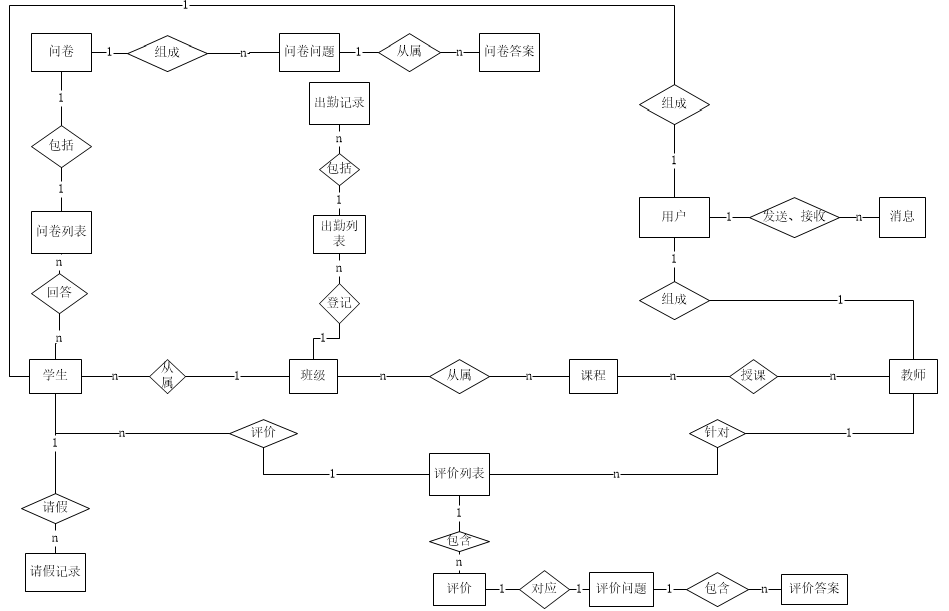


图4-2 系统E-R图

### 4.3.2 消息管理数据库设计

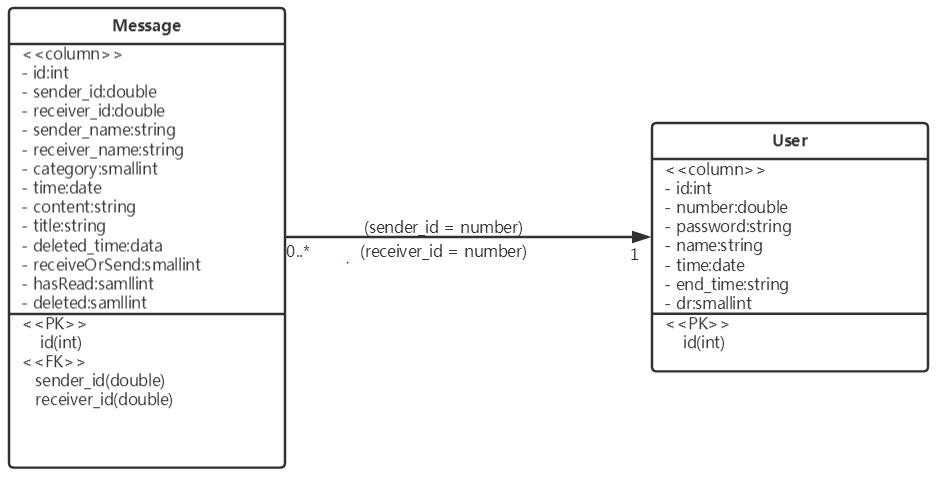


图4-4 消息管理数据模型

如图4-4所示，消息管理由一张消息表(Message)记录消息，消息表(Message)通过sender\_id跟receiver\_id作为外键关联用户表(User)获取发送人跟接收人信息，从而实现消息的发送与接收。具体设计如表4-1。

表4-1 Message表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 约束条件 | 描述 |
| Id | int | 主键 | 消息id |
| Sender\_id | double |  | 发送人id |
| Receiver\_id | double | 外键 | 接收人id |
| senderName | Varchar(30) | 外键 | 发送者姓名 |
| receiverName | Varchar(30) |  | 接收者姓名 |
| category | Varchar(10) |  | 分类 |
| Time | Timestamp |  | 发送或接收时间 |
| Content | text |  | 消息内容 |
| Title | Varchar(30) |  | 标题 |
| deletedTime | datetime |  | 删除时间 |
| receiveOrSend | Char(1) |  | 收发标志 |
| hasRead | Char(1) |  | 已读标志 |
| Deleted | Tinyint |  | 删除标志 |

消息表(Message)包括13个字段，其中id为主键，字段sendOrReceive（0、1）来区别消息是发送的还是接收的，hasRead(0、1)来表示消息是否已读，如果是1表示已读，反之则为未读，deleted(0、1)表示消息的删除与否。Category(0、1)表示消息的类型，0私发消息，1为群发消息。

### 4.3.3 请假管理数据库的设计

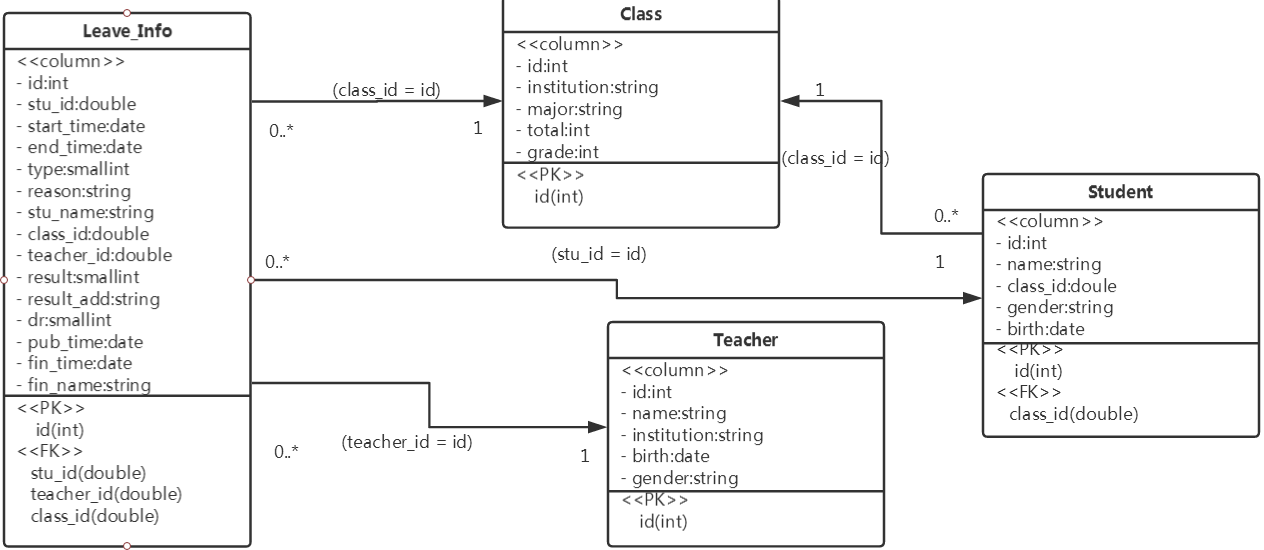


图4-5 请假管理数据模型

如图 4-5所示，每个学生可以对应多个请假记录，每个老师可以受理多个请假请求，每个班级可以有多个人请假，请假表(Leave\_Info) 中有15个，其中stu\_id,teacher\_id,class\_id为外键，分别关联到学生表，老师表，班级表，字段start\_time跟end\_time 分别为请假开始时间跟请假结束时间，pub\_time跟fin\_time为请假发起时间跟请求审批完成时间，字段type(0、1)为请假类型，0为病假，1为时间，字段result(0、1)为审批结果，0为不通过，1为通过，字段fin\_name为受理人姓名。具体设计如表4-2：

表4-2 Leave\_Info表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 约束条件 | 描述 |
| Id | int | 主键 | 请假记录id |
| Stu\_id | double | 外键 | 学生id |
| Class\_id | double | 外键 | 班级id |
| Teacher\_id | double | 外键 | 教师id |
| Stu\_name | Varchar(20) |  | 学生姓名 |
| Type | Varchar(5) |  | 请假类型 |
| Reason | Varchar(200) |  | 请假原因 |
| Start\_time | Datetime |  | 请假开始时间 |
| End\_time | Datetime |  | 请假结束时间 |
| Pub\_time | Timestamp |  | 请求请假时间 |
| Result | Char(1) |  | 是否准许请假 |
| Result\_add | Varchar(50) |  | 回复 |
| Fin\_time | Timestamp |  | 处理时间 |
| Fin\_name | Varchar(30) |  | 处理人 |
| Dr | Char(1) |  | 是否已完成 |

### 4.3.4 出勤管理数据库的设计

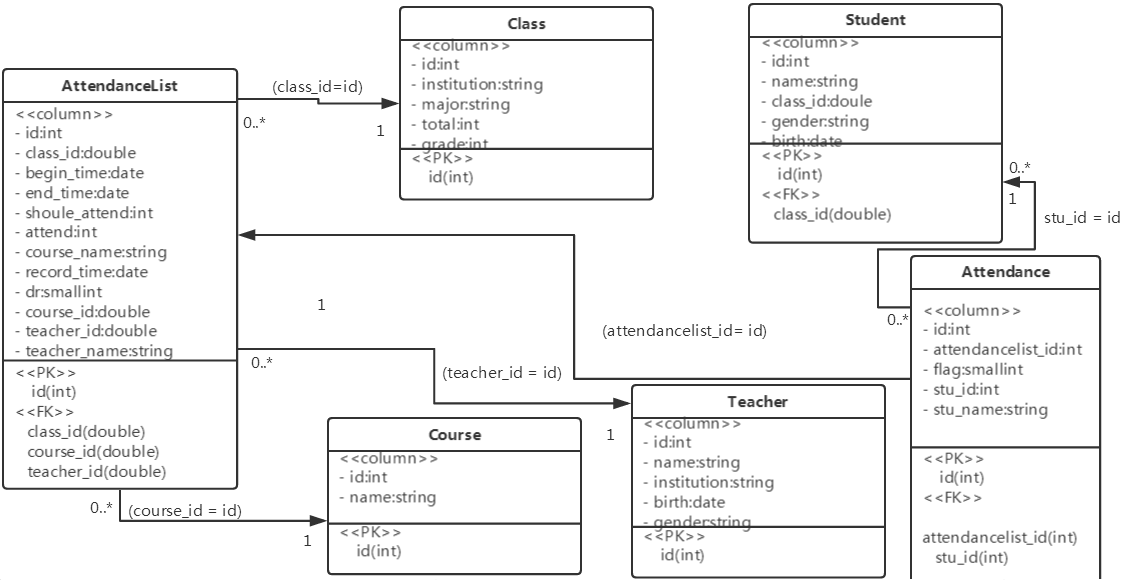


图4-6 出勤管理数据模型

如图4-6所示，出勤管理由两张表组成，一张是出勤表(Attendance),一张是出勤表(AttendanceList),出勤表中有12个字段，class\_id,course\_id,teacher\_id为外键，分别关联班级表，课程表，教师表，实现对每个课程上课的每个班级进行出勤登记，而出勤记录表通过attendancelist\_id关联到出勤表，实现从属关系。具体设计如表 4-3跟表4-4：

表4-3 AttendanceList表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 约束条件 | 描述 |
| Id | int | 主键 | 出勤总表Id |
| Class\_id | double | 外键 | 班级id |
| Teacher\_id | double | 外键 | 教师id |
| Course\_name | Varchar(50) |  | 教师姓名 |
| Teacher\_name | Varchar(30) |  | 出勤时间 |
| Should\_attend | int |  | 应到人数 |
| Attend | int |  | 实到人数 |
| Record\_time | Timestamp |  | 记录时间 |
| begin\_time | datetime |  | 上课时间 |
| End\_time | Datetime |  | 下课时间 |
| dr | tinyint |  | 是否已完成 |
| Course\_id | Double | 外键 | 课程id |

表4-4 Attendance表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 约束条件 | 描述 |
| Id | int | 主键 | 出勤记录id |
| attendancelist\_id | int | 外键 | 出勤总表Id |
| Stu\_id | double | 外键 | 学生id |
| Stu\_name | Varchar(30) |  | 学生姓名 |
| flag | Tinyint |  | 出勤状况 |

如表4-3所示，字段begin\_time跟end\_time表示考勤记录的时间段，should\_attend为对应班级的应到人数，attend为实到人数。

出勤记录表(attendance)中有五个字段，attendancelist\_id为外键，关联到出勤表，字段flag(0、1、2、3)分别表示到勤、缺勤、请假跟早退；stu\_id为外键，关联到学生表。

### 4.3.5 问卷调查数据库设计

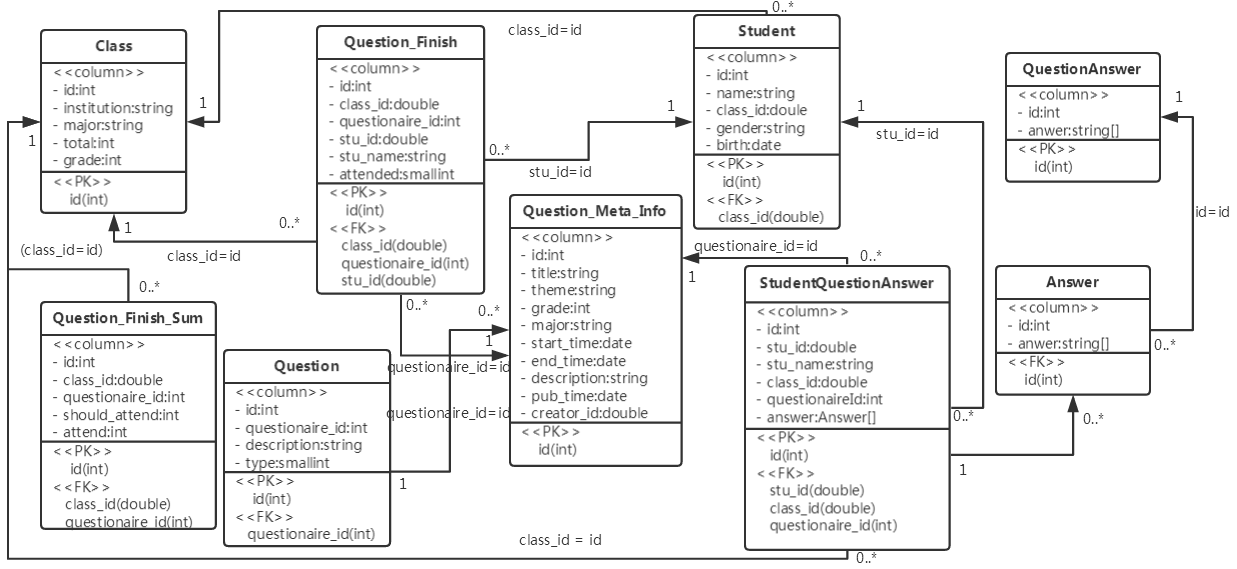


图4-7 问卷调查数据模型

如图4-7所示，一份问卷调查问卷包括问卷基本信息及问卷题目，其中基本信息由问卷元信息表(Questionaire\_Meta\_Info)存储，包括标题，主题，针对年级、专业，问卷开始时间跟结束时间，问卷详细描述以及问卷的创建者id。问题表(Question)中有四个字段，questionaire\_id为外键，关联到问卷元信息表(Questionaire\_Meta\_Info)，字段type(0、1、2)表示问题的分类，分别表示单选题、多选题以及问答题。问卷完成表(Question\_Finish)用来记录学生的完成情况，共有六个字段，questionaire\_id,stu\_id,class\_id为外键，分别关联到问卷元信息表(Questionaire\_Meta\_Info)，学生表(Student)，班级表(Class)，字段attended(0、1)表示是否参与了，0表示未参与，1表示已参与。问卷完成汇总表(Question\_Finish\_Sum)用来统计每个班级问卷回答的情况，共有五个字段，其中class\_id,questioniare\_id为外键，用来关联到班级表(Class)跟问卷元信息表(Questionaire\_Meta\_Info)。问题答案(QuestionAnswer)在数据上离散，因此作为一条条文档放入MongoDB数据库,文档格式为:{“\_id”:1,”answer”:[“A.非常了解”,”基本了解”,”不了解”]}.其中字段\_id为主键参照，参照问题表(Question)的主键。学生问卷回答(StudentQuestionAnswer)同样存入MongoDB。具体设计如表：

表4-5 Questionnaire\_Meta\_Info表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 约束条件 | 描述 |
| Id | int | 主键 | 问卷标识 |
| Title | Varchar(30) |  | 问卷标题 |
| Theme | Varchar(30) |  | 问卷主题 |
| Grade | int |  | 调查的年级 |
| Major | Varchar(60) |  | 调查的专业 |
| Start\_time | Datetime |  | 开始时间 |
| End\_time | Datetime |  | 结束时间 |
| Description | Text |  | 描述 |
| Pub\_time | Timestamp |  | 发布时间 |
| Creator\_id | double |  | 创建者id |

表4-6 Question表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 约束条件 | 描述 |
| Id | int | 主键 | 问卷问题标识 |
| Questionaire\_id | int | 外键 | 问卷标识 |
| description | Text |  | 问卷描述 |
| Type | Tinyint |  | 问题类型（单选、多选、问答） |

表4-7 Question\_Finish表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 约束条件 | 描述 |
| Id | int | 主键 | 学生参与问卷id |
| Stu\_id | double | 外键 | 学生id |
| Class\_id | double | 外键 | 班级Id |
| Questionaire\_id | int | 外键 | 问卷标识 |
| Attended | Tinyint |  | 是否已参与 |
| Stu\_name | Varchar(30) |  | 姓名 |

表4-8 Question\_Finish\_Sum表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 约束条件 | 描述 |
| Id | int | 主键 | 学生参与问卷汇总id |
| Class\_id | double | 外键 | 班级id |
| Questionaire\_id | int | 外键 | 问卷标识 |
| Should\_attend | int |  | 应参与人数 |
| Attend | int |  | 实际参与人数 |

### 4.3.6 评价管理数据库设计

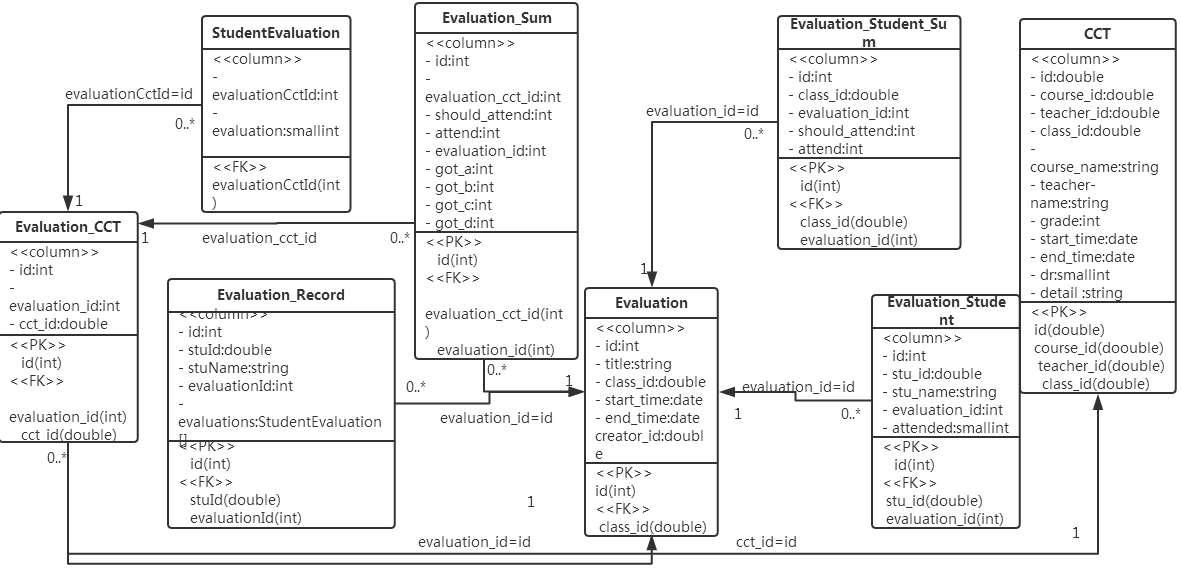


图4-8 评价管理数据模型

如图4-8所示，教学质量评价由教学基本信息跟评价问题组成，其中评价信息表(Evaluation)存储基本信息，包括标题，时间范围，参与评价的班级。表Evaluation\_CCT(id,evaluation\_id,cct\_id)用于存储评价的老师授课信息。表Evaluation\_Student用于记录学生参与评价的情况，每个学生完成了评价之后会更新里面的记录。表Evaluation\_Student\_Sum用来统计每个班级完成参与评价的情况。表Evaluation\_Sum用来统计每个老师各个课程的评价情况。学生的评价记录存入MongoDB中的evaluationRecord集合中，具体设计如4-9 到 4-14表：

表4-9 Evaluation表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 约束条件 | 描述 |
| Id | int | 主键 | 评价基础信息id |
| Title | Varchar(30) |  | 标题 |
| Class\_id | double | 外键 | 参与评价班级id |
| Start\_time | Datetime |  | 开始时间 |
| End\_time | Datetime |  | 结束时间 |
| Creator\_id | Double |  | 创建者id |

表4-10 Evaluation\_CCT表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 约束条件 | 描述 |
| Id | int | 主键 | 评价内容id |
| Evaluation\_id | int | 外键 | 评价基础信息id |
| Cct\_id | double | 外键 | 课程\_班级\_教师标识 |

表4-11 Evaluation\_Student表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 约束条件 | 描述 |
| Id | int | 主键 | 学生参与评估id |
| Stu\_id | double | 外键 | 学生id |
| Stu\_name | Varchar(30) |  | 学生姓名 |
| Evaluation\_id | int | 外键 | 评价基础信息id |
| Flag | tinyint |  | 是否参与评价 |

表4-12 Evaluation\_Student\_Sum表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 约束条件 | 描述 |
| Id | Int | 主键 | 学生参与评价汇总id |
| Class\_id | double | 外键 | 班级id |
| Evaluation\_id | int | 外键 | 评价基础信息id |
| Should\_attend | int |  | 应参与评价人数 |
| Attend | int |  | 实际参与评价人数 |

表4-13 Evaluation\_Record表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 约束条件 | 描述 |
| Id | int | 主键 | 评估id |
| Stu\_id | double | 外键 | 学号 |
| Stu\_name | Varchar(30) |  | 姓名 |
| EvaluationId | int | 外键 | 评价基础信息id |
| Evaluation\_cct\_id | int | 外键 | 评价内容标识 |
| Evaluation | tinyint |  | 评价 |

表4-14 Evaluation\_Sum表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 约束条件 | 描述 |
| Id | int | 主键 | 评价汇总id |
| Evaluation\_cct\_id | int | 外键 | 评价内容id |
| Class\_id | double | 外键 | 班级Id |
| Should\_attend | int |  | 应参与人数 |
| Attend | int |  | 实际参与人数 |
| Got\_a | int |  | 评价满意的人数 |
| got\_B | int |  | 评价基本满意的人数 |
| got\_C | int |  | 评价一般的人数 |
| got\_D | int |  | 评价不满意的人数 |

Evaluation\_Student表中，flag(0、1)表示是否参与评价，0表示未参与评价，1表示已参与。Evaluation\_Record中，字段evaluation(1,2,3,4)为评价，1表示非常满意，2表示基本满意，3为一般，4为不满意。

### 4.3.7 系统管理数据库设计

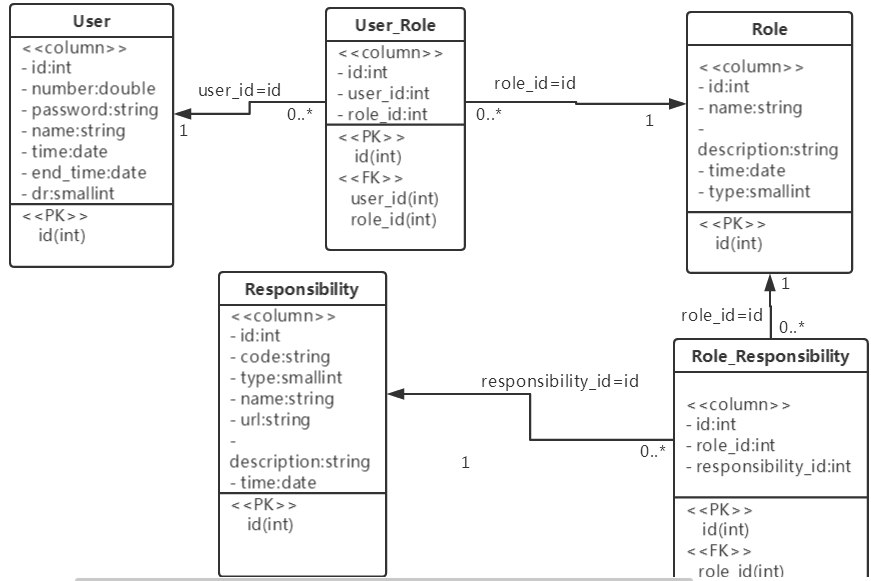


图4-9 系统管理数据模型

如图4-9所示，系统管理由用户表(User)，用户-角色表(User\_Role)，角色表(Role)，角色-职责表(Role\_Responsibility)，职责表(Responsibility)组成，这无噶表形成RBAC(Role-Based Access Control，基于角色的访问控制)的基础，即用户通过角色与权限进行关联。

表4-15 User表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 约束条件 | 描述 |
| Id | int | 主键 | 用户Id |
| Number | double |  | 学号或教职工工号 |
| Password | Varchar(14) |  | 密码 |
| Name | Varchar(30) |  | 姓名 |
| Time | Timestamp |  | 创建时间 |
| End\_time | Date |  | 到期日期 |
| Dr | tinyint |  | 是否失效 |

表4-16 Role表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 约束条件 | 描述 |
| Id | int | 主键 | 角色id |
| Name | Varcahr(50) |  | 角色名称 |
| Description | Varchar(100) |  | 描述 |
| Time | Timestamp |  | 创建时间 |
| Type | Tinyint |  | 角色类型 |

表4-17 Responsibility表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 约束条件 | 描述 |
| Id | int | 主键 | 职责id |
| Code | Varchar(30) |  | 职责编码 |
| Type | tinyint |  | 职责类型 |
| Name | Varchar(50) |  | 职责名称 |
| Url | Varchar(100) |  | 职责链接 |
| Description | Varchar(100) |  | 描述 |
| Time | Timestamp |  | 职责创建时间 |

4-18 User\_Role表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 约束条件 | 描述 |
| Id | Int | 主键 | id |
| User\_id | int | 外键 | 用户Id |
| Role\_id | int | 外键 | 角色id |

表4-19 Role\_Responsibility表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 约束条件 | 描述 |
| Id | int | 主键 | Id |
| Role\_id | int | 外键 | 角色id |
| responsibility\_id | int | 外键 | 职责id |

角色表(Role)中，字段type(0,1,2,3,4,6,7)为角色类型，分别对应角色：学生，老师，辅导员，授课老师，管理员，领导，普通用户。职责表(Responsibility)中，字段type(0,1,2,3)表示职责的类型，分别是：显示，查看，更新，保存。

## 4.4 本章小结

本章首先介绍了系统采用的系统架构，然后介绍了系统的总体功能模块划分和各部分的功能描述，最后对系统的数据库设计做了重点介绍。

# 第5章 系统详细设计与实现

## 5.1 系统的开发环境

本系统采用Springcloud微服务架构，具体采用了以下技术进行开发：

(1)用户界面技术:HTML,CSS,JavaScript

(2)用户界面使用的插件:Bootstrap,Bootstrap-table,Layui,Ztree,select2,grumble, Datetime-picker,Unslider

(3)软件体系架构:B/S架构

(4)开发环境：Window，Linux

(5)数据库：Mysql,Mongodb,Redis

(6)数据访问技术:Mybatis

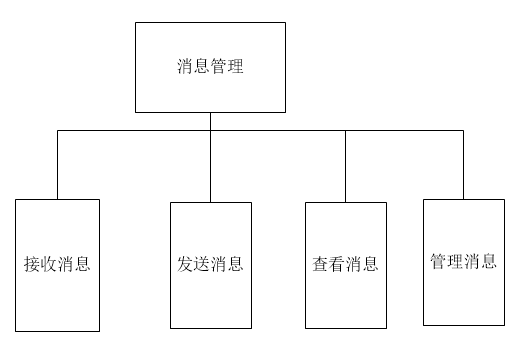
(7)开发工具:Intellj idea

## 5.2 系统各模块的详细设计与实现

教学监控系统是一个参与角色较多、复杂、多模块的动态系统，它由消息管理、出勤管理、请假管理、问卷调查、评价管理以及系统管理等模块组成。在教学过程中出现的问题反应在学生的出勤率、学生的请假情况以及教学质量评价上。此外，教学监控系统提供的消息管理提供了学生与老师、学生与辅导员、领导跟教师等多类角色之间的交流机制，它能够实时收发消息，以及消息提醒。

### 5.2.1 消息管理模块的详细设计与实现

在教学监控系统中，消息管理在各个模块中都起到了很重要的角色，保证了各个用户角色间的交流，让教学情况能够及时追踪反馈。消息管理模块包括发送消息，接收消息，查看消息以及消息的管理。本小节将介绍消息管理的详细设计与实现。



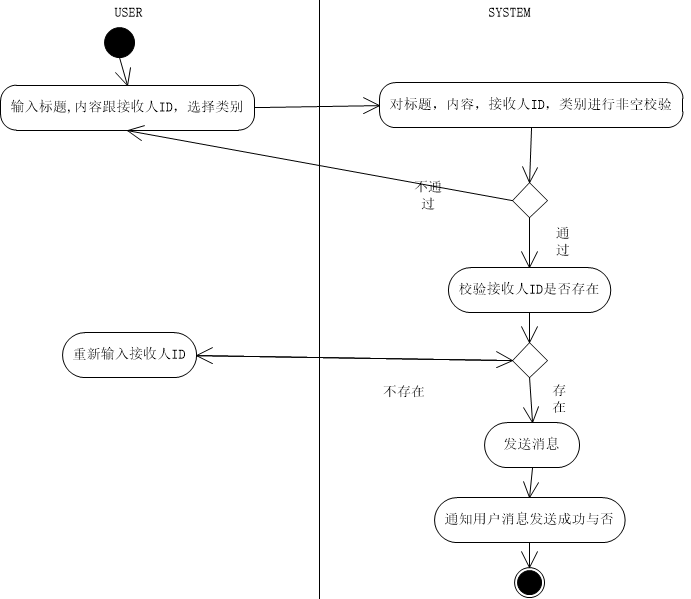
5-1 消息管理模块图

消息管理模块主要有如下接口：MessageService,MessageOperateService,DurableTopicSenderService,DurableTopicSubscriberService,QueueConsumberService,QueueConsumberService。接口的实现分别是:MessageServiceImpl,MessageOperateServiceImpl,DurableTopicSenderServiceImpl,DurableTopicSubscriberServiceImpl,QueueConsumerServiceImpl。

5.2.1.1 发送消息

(1)业务逻辑

发送消息分为私发消息跟群发消息，用户发送消息时，输入消息的标题、内容、接收者ID以及选择消息类别，系统会对发送的消息进行非空校验，校验通过后对接收者ID进行查看，存在则发送消息，否则通知用户从新输入接收人ID。



5-2 发送消息活动图

(2)流程逻辑设计

如图5-3所示，在发送个人消息的时候，会调用MessageController中的sendMessage,在方法中首先调用MesssageProducerService的iinit方法进行初始化，然后在调用MessageService对消息封装，返回一个实体类Message，最后通过调用MessageProducerService的saveMessage方法，在saveMessage方法中会先调用MessageMapper的saveMessage保存消息记录，在调用自身的send方法将消息发送到Activemq消息队列中。

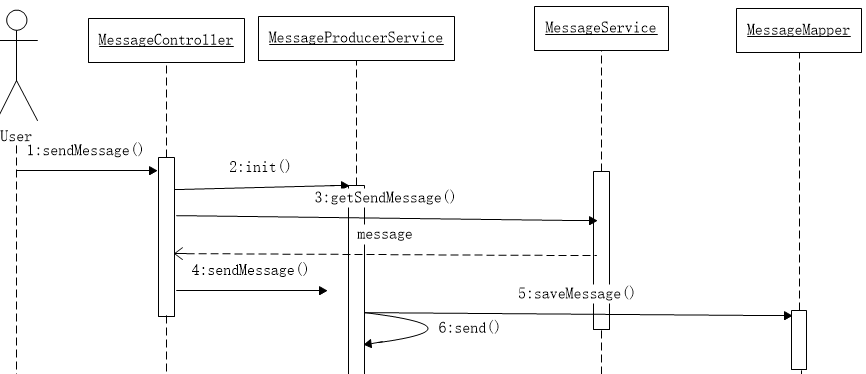


图5-3发送消息顺序图

(3)运行界面:

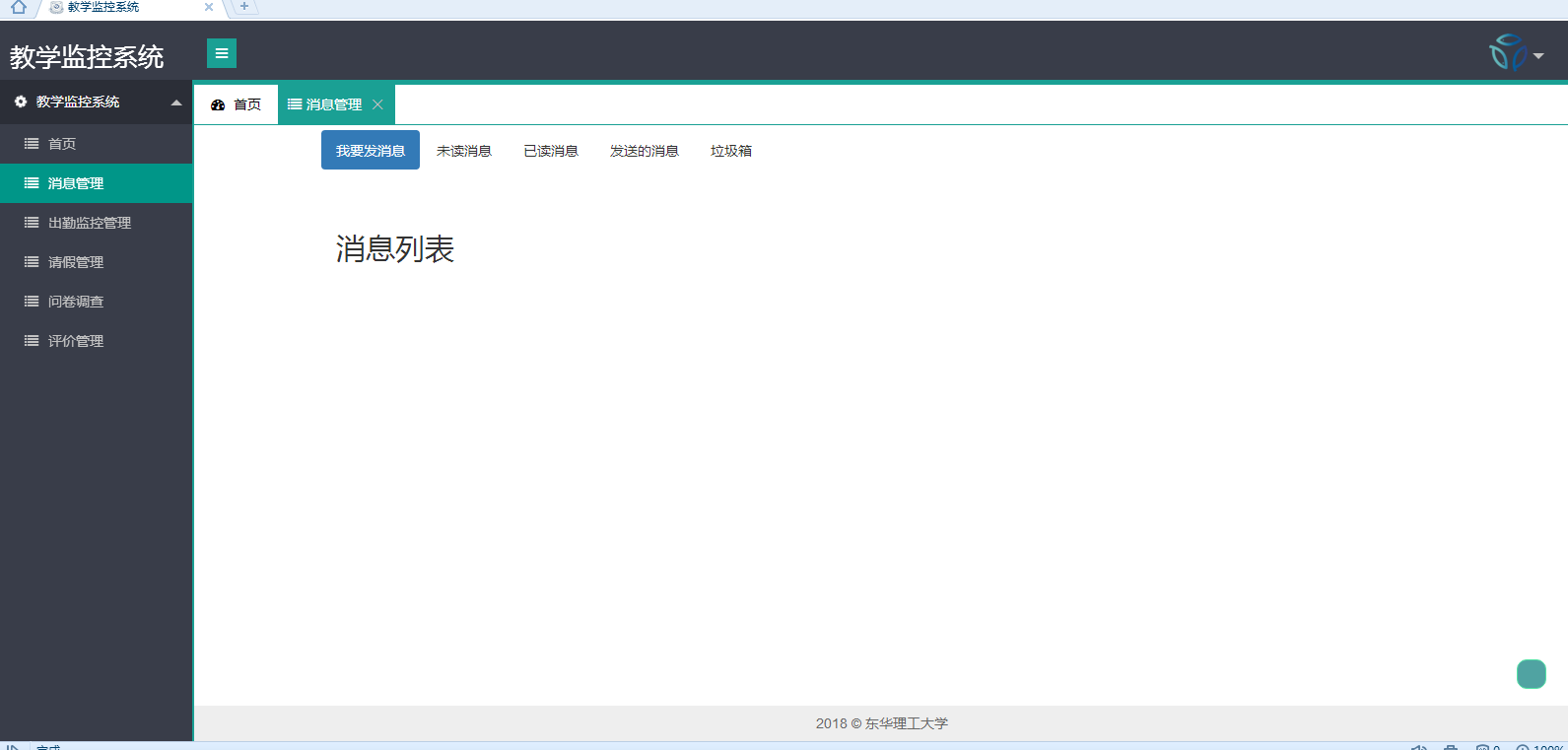


图5-4 消息管理主界面图

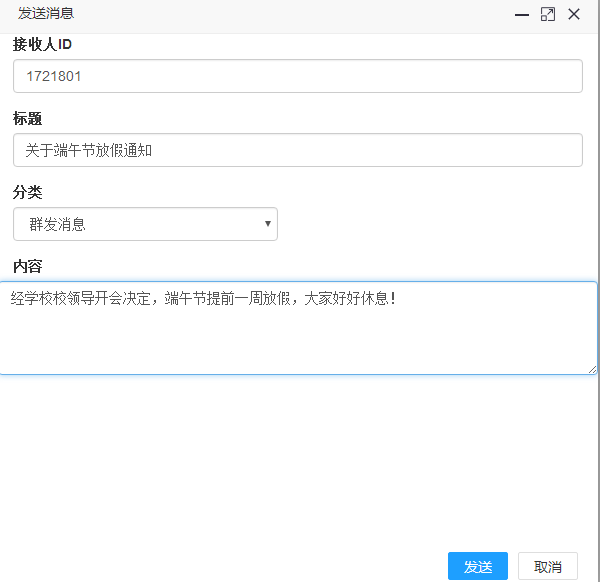


图5-5 发送消息界面图

5.2.1.2 接收消息

(1)业务逻辑

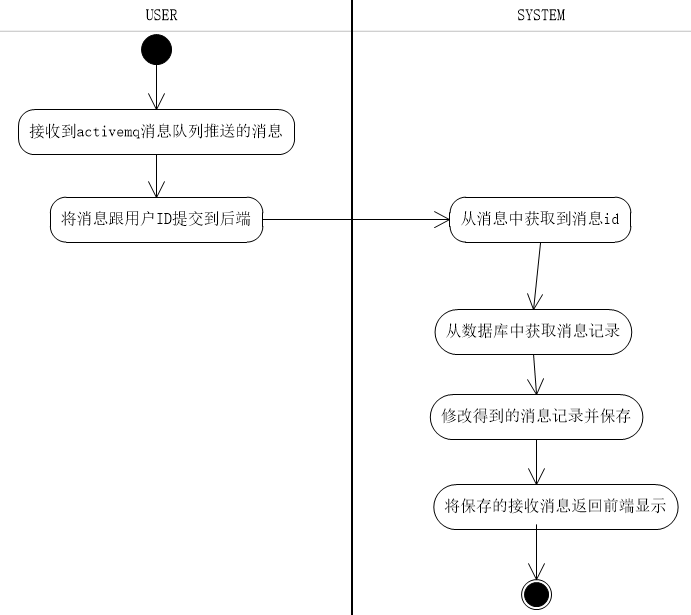
用户在前端界面接收到activemq消息队列里推送的消息，将消息发送到后台，后台系统从消息从得到messageId跟用户Id,从数据找到该消息记录拷贝一份修改对应的值之后进行保存，并将存入的接收数据。

图5-6 接收消息活动图

(2)流程逻辑

在接收消息时，会调用MessageOperateController的saveReceiveMessage方法，saveReceiveMessage方法主要通过调用MessageOperateService的saveReceiveMessage来实现的，在MessageOperateService的saveReceiveMessage中首先调用messageMapper获取接收的消息对应的发送消息对象，紧接着对其修改后再次调用messagMapper保存。

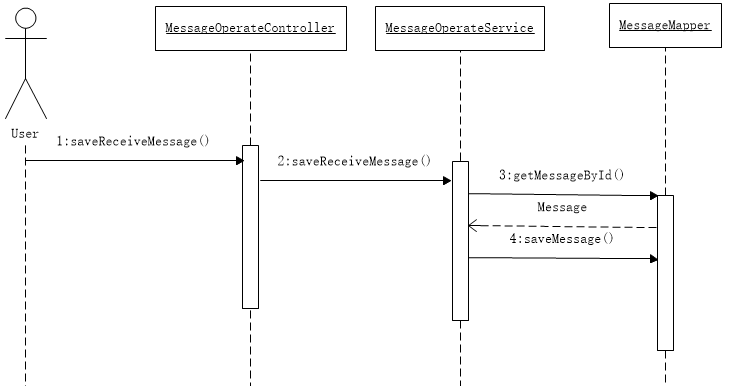


图5-7 接收消息顺序图

(3)运行界面:



图5-8 消息通知界面图



图5-9 消息列表界面图

5.2.1.3 查看消息跟删除消息

(1)业务逻辑

用户在未读消息，已读消息，或者发送的消息页面，点击消息查看对应消息，系统根据选中的消息id找到对应消息记录返回到页面展示。用户点击删除按钮，将对应的消息传到后台，进行消息更新操作，将消息的状态置为1，从逻辑上实现删除操作。

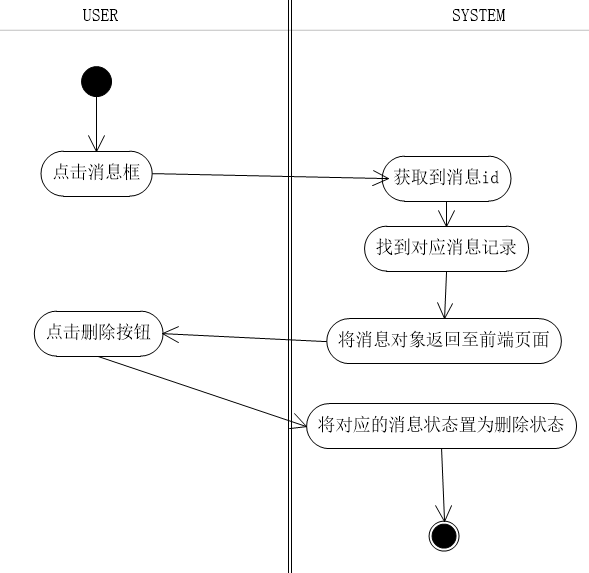


图5-10 查看消息与删除活动图

(2)流程逻辑

查看消息的时候会调用MessageOperateController的viewMessage方法，在此方法中先调用messageOperateService的viewMessage获取消息，紧接着调用messageOperateService的readMessage方法更新该消息的状态。

删除消息的时候会调用MessageOperateController的delMessage方法，方法中直接调用messageOperateService的deleteMessage删除消息记录，底层并未删除消息记录，而是更新消息的状态。

(3)运行界面:

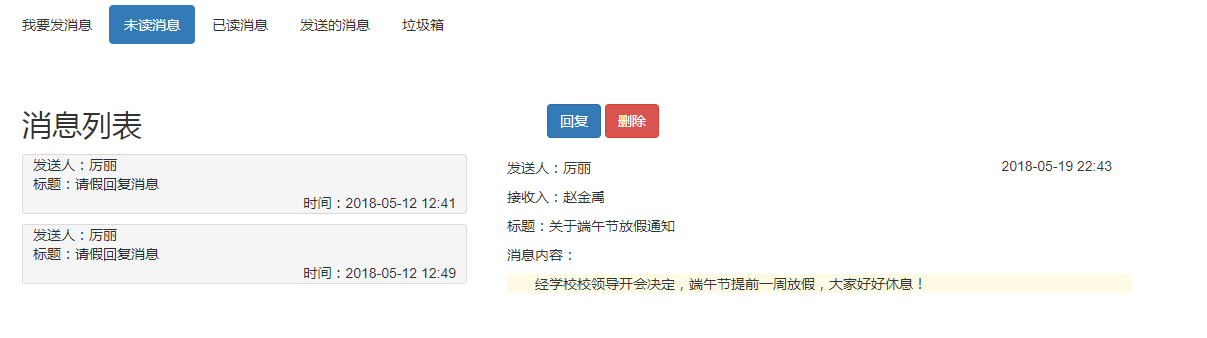


图5-11 查看消息界面图

### 5.2.2 请假管理模块的详细设计与实现

请假管理是评定教学质量的一个重要模块，同时它也是最能反映学生学习的精神面貌。请假管理包含以下几个子模块：登记请假、请假审批、查看请假、预警通知。

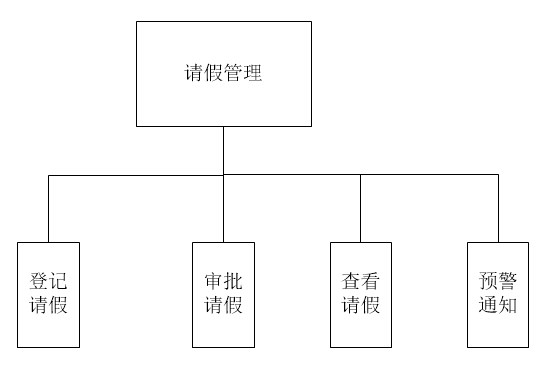


图5-12 请假管理模块图

5.2.2.1 登记请假

(1)业务逻辑

学生角色用户进行请假登记，填写请假原因，选择请假开始时间跟结束时间，请假类型之后提交发起请假请求，系统会将记录学生请假请求并通知学生对应的辅导员。

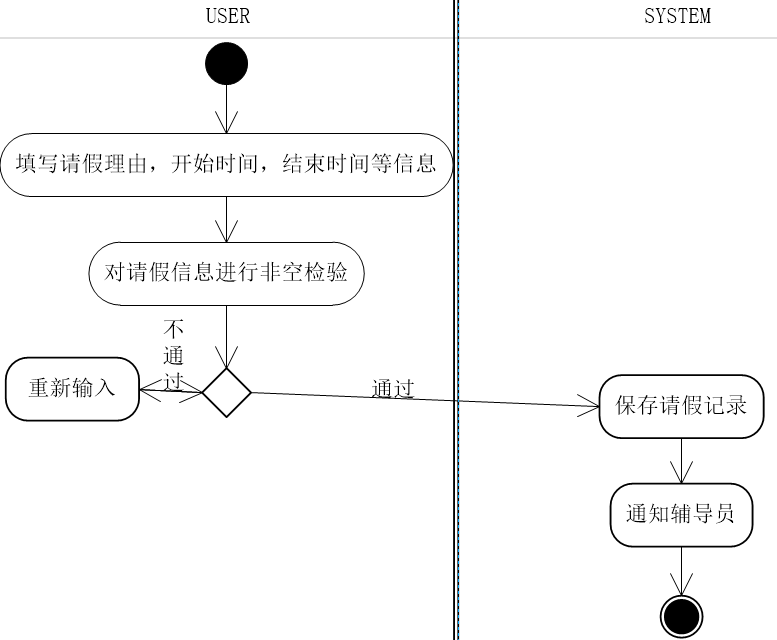


图5-13登记请假活动图

(2)流程逻辑

登记请假的时候，调用LeaveController的recordLeaveInfo方法，在recordLeaveInfo方法中，调用LeaveService的recordLeaveInfo,通过调用leaveInfoMapper的recordLeaveInfo来将请假保存到数据库，然后通过restTemplate调用消息管理模块的sendMessage服务，通知学生的请假动态。

(3)运行界面:

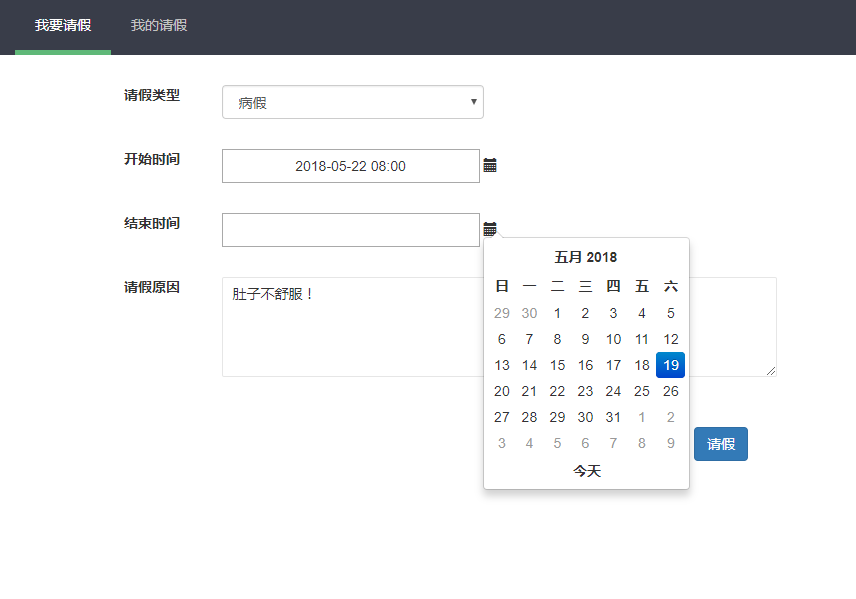


图5-14 登记请假界面图

5.2.2.2 审批请假

(1)业务逻辑

辅导员角色用户收到消息提醒查看学生请假，选择通过或者不通过，并附上审批回复，选择通过后，系统将更新请假记录的审批结果，发送请假更新动态消息给学生，之后计算出在请假区间内的上课信息，将学生的请假条跟审批回复发送到任课老师。不通过则直接更新请假记录的审批结果并发送消息给学生。

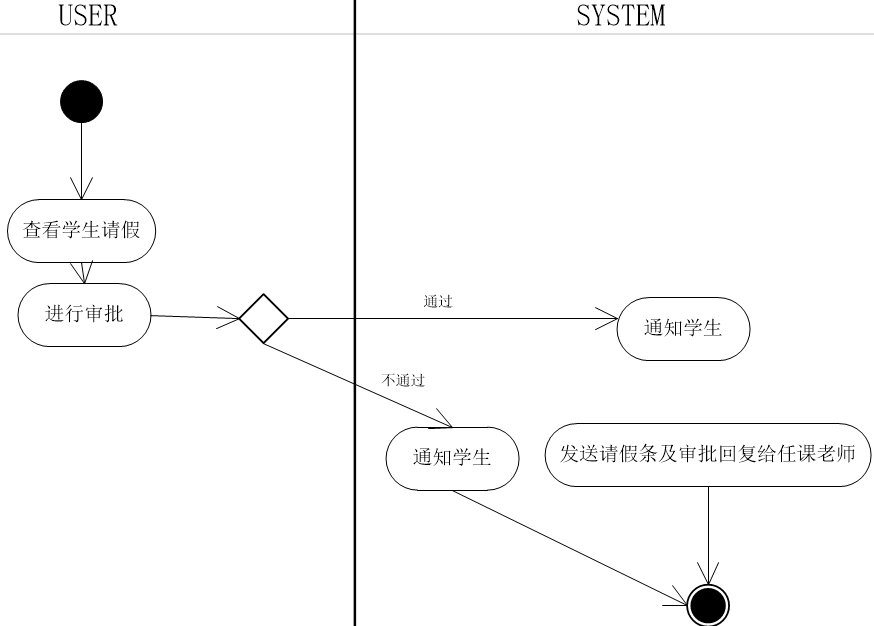


图5-15审批请假活动图

(2)流程逻辑

辅导员查看学生的请假请求：

选择通过调用LeaveController的agreeLeave方法，在此方法中，通过调用了leaveService的agreeLeave的方法，在leaveService的agreeLeave方法中，先调用leaveInfoMapper获取请假记录，然后进行更新审批结果，之后调用工具类CourseScheduleTimeSelectUtil的isTaken判断在请假区间的课程并进行收集，之后调用消息管理的sendMessage服务通知任课老师并发送请假条给任课老师。

选择不通过调用LeaveController的disagreeLeave方法，里面调用leaveService的disagreeLeave方法更新审批结果为不通过并通知学生。

(3)运行界面:

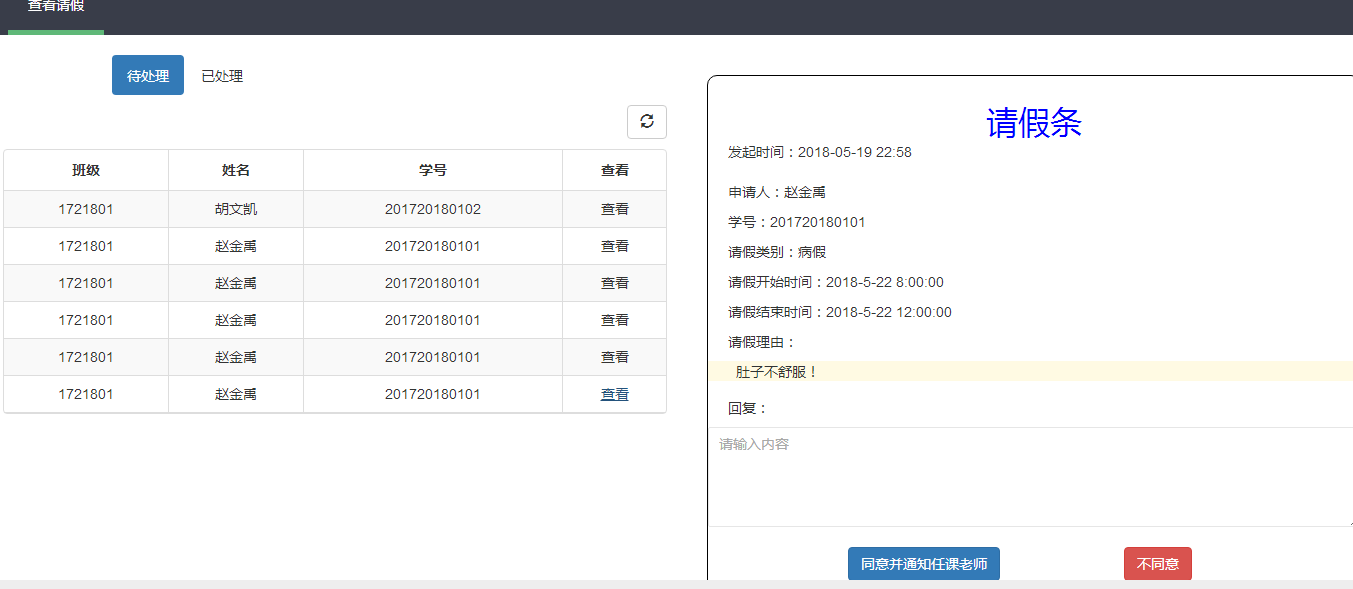


图5-16 审批请假界面图

实现要点:

LeaveController调用LeaveService的agreeLeave方法，在该方法中获取学生信息，学生本学期所有上课信息，对课程进行区间筛选，将出现在请假区间内的课程放入isTaken集合中，然后将学生请假条逐一发送给任课老师，具体代码如下：

FinishedLeaveInfoEntity leaveInfoEntity = leaveInfoMapper.getFinishedLeaveInfoById(id);  
String teacherName = **this**.**restTemplate**.getForObject(**"http://teach-monitor-data-provider/data/getUserNameByNumber?number="** + Integer.*parseInt*(clientId), String.**class**);  
List<Map<String, Object>> list = **this**.**restTemplate**.getForObject(**"http://teach-monitor-data-provider/data/getCCTEntityByClassId?classId="** + leaveInfoEntity.getClassId().intValue(), List.**class**);  
List<CCTEntity> entities = **new** ArrayList<>();  
List<CCTEntity> isTaken = **new** ArrayList<>();  
**for** (Map<String, Object> map : list) {  
 CCTEntity entity = **new** CCTEntity();  
 entity.setClassId(leaveInfoEntity.getClassId());  
 对课程-班级-老师实体进行封装

}  
entities.stream().forEach((e) -> System.***out***.println(e.getStartTime()));  
System.***out***.println(leaveInfoEntity.getEndTime());  
entities.stream().forEach((e) -> {  
 **if** (isTaken(e, leaveInfoEntity.getStartTime(), leaveInfoEntity.getEndTime())) {  
 isTaken.add(e);  
 }  
});  
MultiValueMap<String, Object> tempParamMap = **new** LinkedMultiValueMap<>();  
tempParamMap.add(**"category"**, 0);

对学生消息内容、标题、接收人等信息进行设置  
Integer result = Integer.*parseInt*(Constants.*resultType*.get(**"pass"**).toString());  
**leaveInfoMapper**.updateLeaveInfo(result\_add, id, **new** Date(), result, teacherName);  
Map<String, Object> tempMap = **this**.**restTemplate**.postForObject(**"http://teach-monitor-message-manage/message/sendMessage"**, tempParamMap, Map.**class**);  
**if** (isTaken.size() > 0) {  
 isTaken.stream().forEach((e) -> {  
 String title = **"请假消息通知"**;

编辑通知上课老师消息内容  
 Integer category = 0;  
 MultiValueMap<String, Object> paramMap = **new** LinkedMultiValueMap<>();  
 paramMap.add(**"category"**, category);

对消息参数进行设置  
 Map<String, Object> message1Map = **this**.**restTemplate**.postForObject(**"http://teach-monitor-message-manage/message/sendMessage"**, paramMap, Map.**class**);  
 });  
}

### 5.2.3 出勤管理的详细设计与实现

出勤管理分为出勤登记、查看出勤、修改出勤、出勤预警通知等模块。由任课老师登记出勤，学生、老师、领导、任课老师等角色都可查看出勤，领导、辅导员、任课老师可以发送出勤预警通知。

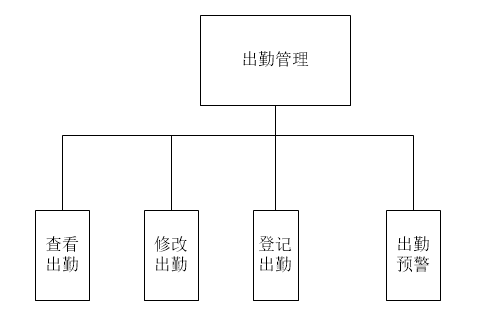


图5-17 出勤管理模块图

5.2.3.1 登记出勤

(1)业务逻辑

任课老师选择课程，班级，考勤开始时间、结束时间，实到人数，为每个学生进行出勤记录后提交，系统将每条记录插入对应数据库表中并同时进行统计。

(2)运行界面:



图5-18 登记出勤界面图

5.2.3.2 查看出勤

(1)业务逻辑

任课老师选择具体课程、具体班级查看具体日期的出勤，系统自动查找出当天的出勤记录以及统计缺勤的情况返回给前端页面展示。

(2)运行界面:



图5-19 查看出勤界面图

### 5.2.4 评价管理、问卷调查的设计与实现

评价管理作为衡量教学质量的一个重要模块，包含发起评价、进行评价、查看评价。

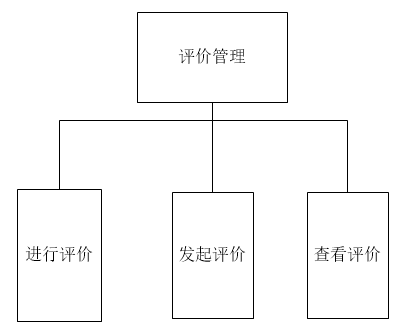


图5-20 评价管理模块图

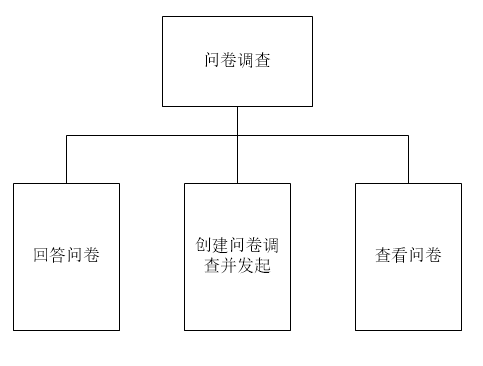


图5-21评价管理模块图

学生评价实现要点:

EvaluationController调用EvaluationFinishService的saveStudentEvaluation方法，对学生的回答进行统计并更新记录,具体代码如下:

EvaluationRecord evaluationRecord = **this**.getEvaluationRecord(stuId, stuName, evaluationId, evaluationItemList);  
**evaluationRecordRepository**.save(evaluationRecord);  
evaluationRecord.getAnswers().stream().forEach((e) -> {  
 **if** (**evaluationSumMapper**.checkIfExist(e.getEvaluationCctId()) > 0) {  
 **if** (e.getEvaluation() == Constants.*evaluationMark*.get(**"A"**))  
 **evaluationSumMapper**.updateA(e.getEvaluationCctId());  
 **else if** (e.getEvaluation() == Constants.*evaluationMark*.get(**"B"**))  
 **evaluationSumMapper**.updateB(e.getEvaluationCctId());  
 **else if** (e.getEvaluation() == Constants.*evaluationMark*.get(**"C"**))  
 **evaluationSumMapper**.updateC(e.getEvaluationCctId());  
 **else  
 evaluationSumMapper**.updateD(e.getEvaluationCctId());  
 } **else** {  
 **if** (e.getEvaluation() == Constants.*evaluationMark*.get(**"A"**))  
 **evaluationSumMapper**.save(evaluationRecord.getEvaluationId(), e.getEvaluationCctId(), totalNumber, 1, 1, 0, 0, 0);  
 **else if** (e.getEvaluation() == Constants.*evaluationMark*.get(**"B"**))  
 **evaluationSumMapper**.save(evaluationRecord.getEvaluationId(), e.getEvaluationCctId(), totalNumber, 1, 0, 1, 0, 0);  
 **else if** (e.getEvaluation() == Constants.*evaluationMark*.get(**"C"**))  
 **evaluationSumMapper**.save(evaluationRecord.getEvaluationId(), e.getEvaluationCctId(), totalNumber, 1, 0, 0, 1, 0);  
 **else  
 evaluationSumMapper**.save(evaluationRecord.getEvaluationId(), e.getEvaluationCctId(), totalNumber, 1, 0, 0, 0, 1);  
 }  
});  
**evaluationStudentMapper**.update(evaluationId, stuId, Integer.*parseInt*(Constants.*attended*.get(**"yes"**).toString()));  
**evaluationStudentSumMapper**.update(evaluationId, classId);

运行界面:

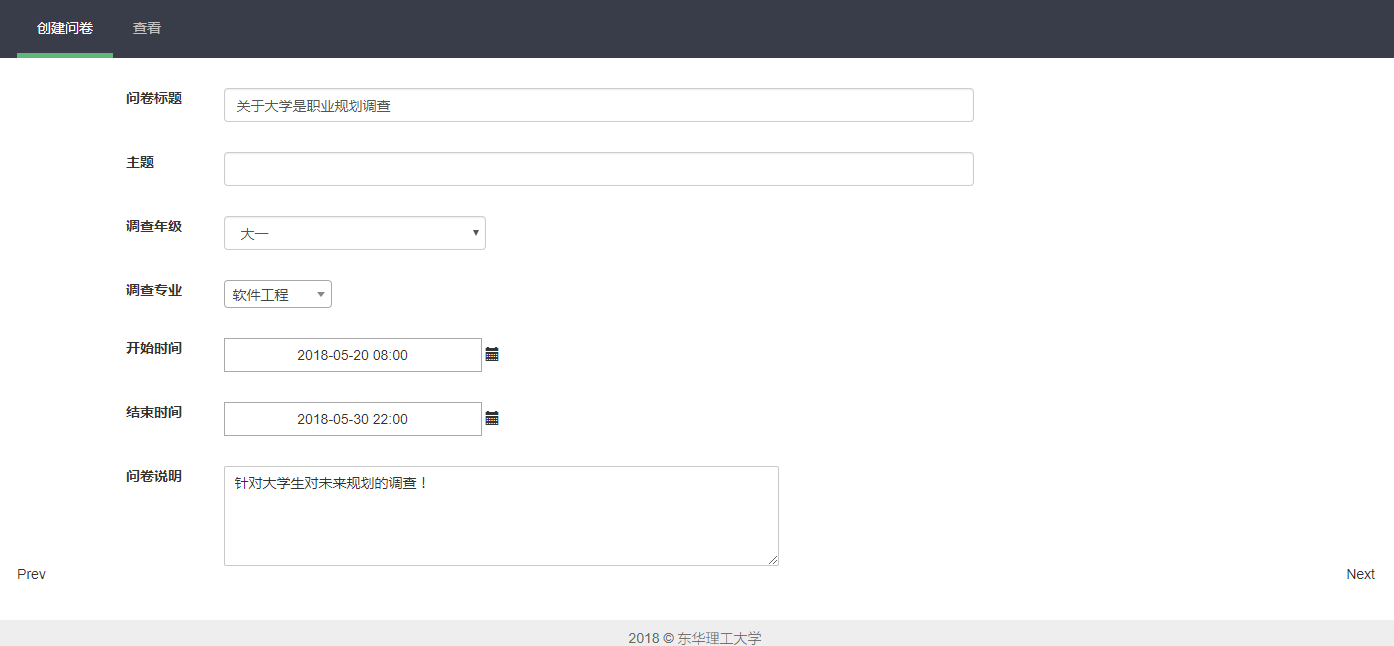


图5-22 问卷调查编辑界面图



图5-23 查看问卷回答情况界面图



图5-24 创建评价问卷界面图



图5-25 查看评价情况界面图



图5-26 评价满意度排行图

### 5.2.5 系统管理的详细设计与实现

系统管理分为用户管理、角色管理以及职责管理三个子模块。实现用户到角色的分配，角色到职责的分配。系统管理权限一般是管理管拥有，由管理员来进行管理。

(1)用户管理

业务逻辑：管理员根据用户ID查找用户，对用户的角色进行添加或者删除。

运行界面：

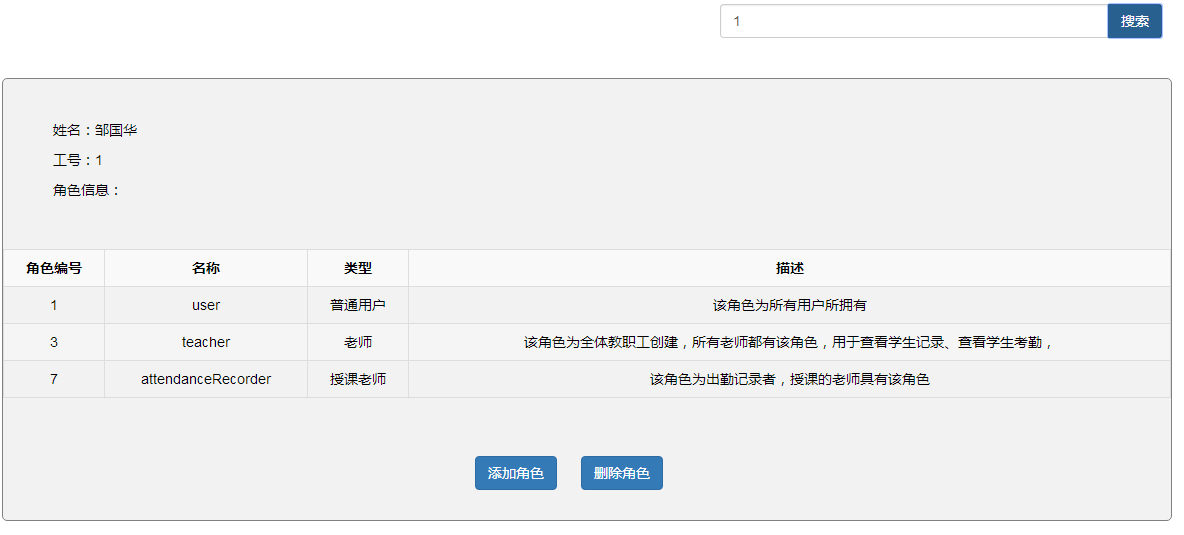


图5-27 查看用户界面图

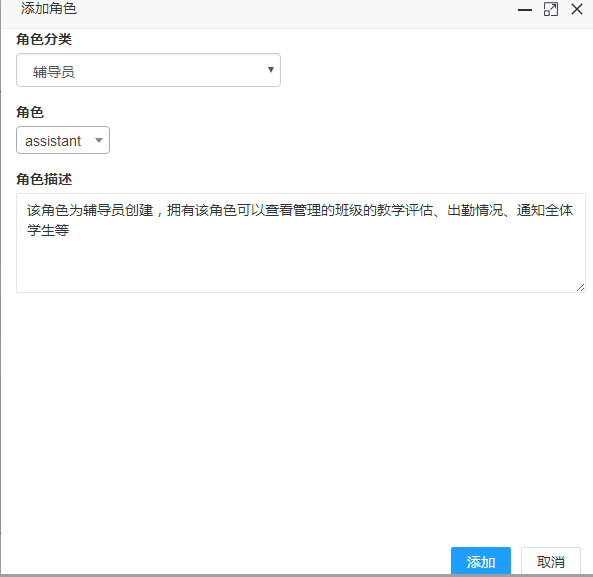


图5-28 添加用户角色界面图

(2)角色管理

业务逻辑：管理员通过角色名称查询到角色信息，为角色重新分配职责，另外，管理员可以创建、修改角色。

运行界面:



图5-29 查看角色界面图

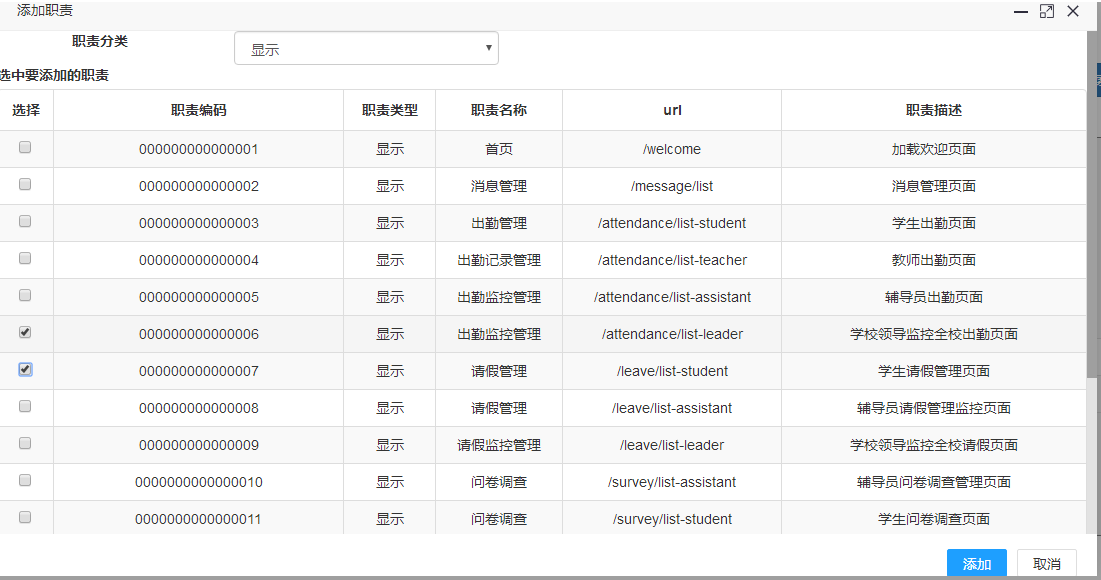


图5-30 添加角色职责界面图

(3)职责管理

业务逻辑：管理员根据职责编码查询职责信息，修改或者删除职责，还可以创建职责。

运行界面:



图5-31 查看职责界面图

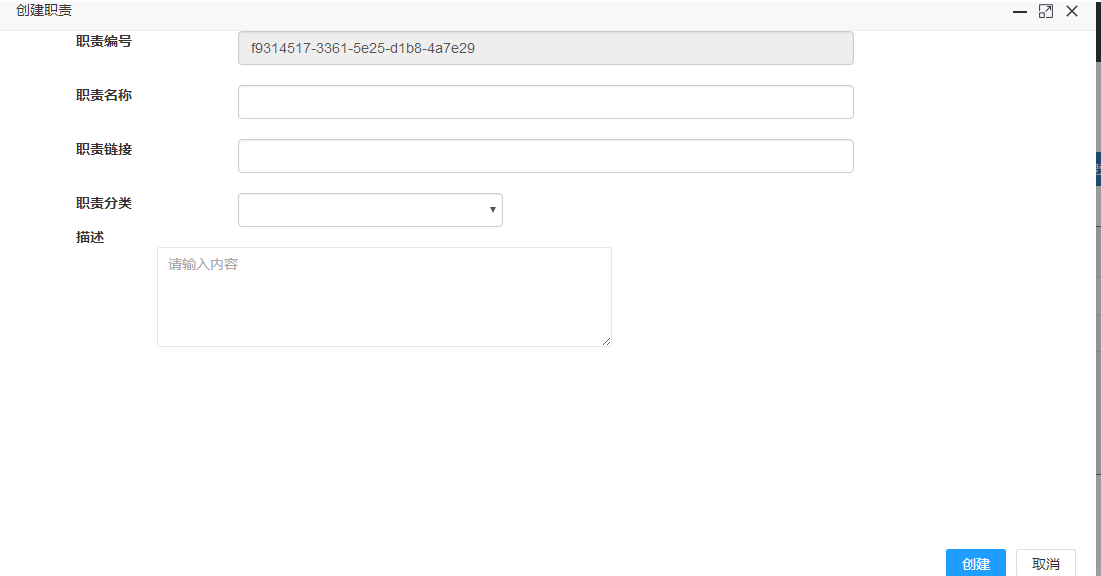


图5-32 创建职责界面图

## 5.4 本章小结

本章主要介绍了系统开发的相关主题，介绍了系统运行、开发环境，框架技术，结合活动图、系统模块图、顺序图以及运行截图介绍了系统各主要模块的开发过程。

# 第6章 总结

本文从当今学校的教学监控现状出发，对其进行分析与调查，得知如今的学校的教学监控系统还不是很实用，在问卷调查方面仍然采用传统的调查问卷，这样的人工统计教学数据造成了信息的滞后和遗漏，并且严重浪费人力、物力，不利于教学质量水平的提高。为此，结合我校的具体需求，本文提出了构建本教学监控系统的必要性和解决方案。

本文结合学校以及教学监控系统的特点，采用springcloud微服务的B/S架构进行设计与开发。在对程序的移植与部署上，采用docker容器技术，实现单个程序多个版本的运行。在客户端使用Ribbon来负载均衡，削弱访问高峰期。本系统采用表现层、业务逻辑层、数据访问层的三层架构实现了拓展性和维护性请的系统。此外，本系统采用前后端分离，降低耦合度，提高系统容错性。本文概述了教学监控系统的研究背景并着重对教学监控系统的请假管理、消息管理、出勤管理等6个功能模块的设计和实现分析和介绍。

本文遵从软件工程过程规范，生命周期从需求分析、概要设计、详细设计阐述系统的设计开发过程，并通过系统运行结果展示本系统已经实现的全部功能。

# 参考文献

1. 梁燕来，刘超. 一种教学质量监控系统的设计[J]. 教学与管理,2007.
2. 高建华,徐雯．基于Spring MVC及MyBatis的Web应用框架研究[J]．微型电脑应用，2012,28(07): 15.
3. 任文涛，张茂仁,耿立明.教学质量监控系统的研究与建设[J]. 2005.
4. 王富强. SpringBoot揭秘：快速构建微服务体系[M]. 机械工业出版社，2016.
5. 黄健宏. Redis设计与实现[M]. 机械工业出版社，2014.
6. 龚鹏. 微服务分布式架构开发[M]. 人民邮电出版社,2018

[7] 方志明. 深入理解Spring Cloud与微服务架构[M]. 人民邮电出版社,2018.

[8] 杨恩雄. 疯狂Spring Cloud微服务架构实战[M]. 电子工业出版社,2017.

[9] 柳伟卫. Spring Boot2.0企业级应用开发实战[M]. 北京大学出版社，2018.

[10] 汪云飞. JavaEE开发的颠覆者：Spring Boot实战[M]. 电子工业出版社,2016.

[11] Craig Walls. Spring Boot in Action[M]. 人民邮电出版社. 2016.

[12] 杨开振. 深入浅出Mybatis技术原理与实践[M]. 电子工业出版社，2016.

[13] Bullock．Effective Java (4th Edition) [M]．People's Posts and TelecommunicationsPress．2009.

[14] Bruce Synder. AciveMQ in Action[M]. Manning Publications;Pap/Psc. 2011.

[15] 廖煜，宴东. Docker容器实战:原理、架构与应用[M]. 电子工业出版社，2016.

[16] 单东林，张晓菲，魏然. 锋利的jQuery[M]. 北京:人民邮电出版社，2009:35-42.

# 致 谢

大学本科四年的生活也即将结束了，在这四年最要感谢的人首先要是我的父母，是他们的辛勤劳动和无私奉献，才使得我能走进大学，在大学之中学习到这么多知识并且认识到很多的朋友，接触到很多值得学习的榜样，也是因为他们让我在每次遇到困难时候也能够勇往直前。对此我唯有更加努力的学习和工作才能报答他们的付出和对我的期望。

同时论文的完成，也主要得力于我的导师李丽华老师的悉心指导和帮助，李丽华老师为人亲切、睿智而又负责任，在论文写作过程中，她十分关心我的进展，并且一直督促和指点着我们完成这次毕业设计，也为这次设计提供了很多宝贵的意见和建议，帮助完善了这次论文和设计。

同时在东华理工大学求学的这四年时间里，也得感谢这四年为我们传授知识和经验的老师们。感谢这四年在身边给予我的许多鼓励和关怀的同学和朋友们，如果没有你们，就没有大学生活的丰富多彩。另外，也对参考文献中所列的各文献的作者表示感谢。最后感谢各位专家和评委审阅我的论文。