logo_school

**毕业论文（设计）**

**基于SSH框架的学生信息管理系统的分析与实现**

**学 院 智能科学与工程学院**

**专 业 计算机科学与技术**

**班 级 计算机科学与技术1班**

**学 号 16612500025**

**姓 名 孙文阁**

**指 导 教 师 向金明**

**成 绩**

云南工商学院

20 年 月

**云南工商学院**

**毕业论文（设计）诚信声明**

本人声明，所呈交的论文（设计）是本人在导师的指导下独立完成的研究成果。除了文中特别加以标注引用的内容外，本论文（设计）不包含法律意义上已属于他人的任何形式的研究成果,也不包含本人已用于其他学位申请的论文（或成果。对本文的研究作出重要贡献的个人和集体，均已在文中以明确方式表明。本人完全意识到本声明的法律后果由本人承担。本毕业论文(设计)成果归云南工商学院所有。

作者签名：

日 期：

**基于SSH框架的学生信息管理系统的分析与实现**

摘 要

本次课题主要应用于高校学生管理，主要为解决高校学生管理工程量大，信息储存困难，流程复杂化而开发的一个管理系统，主要介绍开发工程师在开发系统过程中所经历的需求分析，系统的功能设计与数据库设计到代码的编写与完成，和对系统进行的各项测试，本人在本次课题内担任全栈开发工程师。

依照本次课题所设计的管理系统，为解决各项难题所需要用到Spring、Spring MVC 、Hibernate以及B/S框架技术和前端的Layui框架，数据库采用MySQL服务器。

该系统完成后，能够减轻高校管理人员的工作负担，高效率、规范化地处理、储存大量的学生信息，促进学校对于学生的规范管理。

本次课题能积累大量的开发经验与了解高校学生管理的复杂逻辑，熟悉系统开发的生命周期，锻炼独立解决问题能力，更多的了解系统各项错误的解决方法。

**关键词**：基于SSH框架的学生信息管理系统；全栈开发工程师；Spring+SpringMVC +Hibernate；B/S架构

**ABSTRACT**

This project is mainly applied to the management of college students. It is a management system developed to solve the problems of large quantities of College Students' management, difficult information storage and complicated process. It mainly introduces the requirements analysis experienced by the development engineer in the process of developing the system, the function design and database design of the system, the compilation and completion of the code, and various tests on the system, I am a full stack Development Engineer in this project.[[1]](#endnote-1)[1]

According to the management system designed in this project, in order to solve various problems, we need to use spring, spring MVC, hibernate, B / S framework technology and front-end layui framework. MySQL server is used as the database.[[2]](#endnote-2)[2]

After the completion of the system, it can reduce the workload of university administrators, process and store a large number of student information efficiently and normatively, and promote the standardized management of students.

This project can accumulate a lot of development experience and understand the complex logic of college student management, be familiar with the life cycle of system development, exercise the ability to solve problems independently, and learn more about the solutions to various errors in the system.

**Key Words**: Student information management system based on SSH framework； full stack development engineer；Spring+SpringMVC+Hibernate；B / S architecture

目 录

摘 要 I

ABSTRACT II

引言 1

第一章、绪 论 3

1.1. 开发背景 3

1.2. 开发意义 3

1.3. 可行性分析 3

1.4. 研究内容介绍 4

第二章、相关技术概论 5

2.1. B/S架构 5

2.2. JavaWeb网站技术 5

2.3. Spring容器 5

2.4. SpringMVC框架 5

2.5. Hibernate 6

2.6. Layui前端框架 6

2.7. 开发工具MyEclipse 6

2.8. 服务器MY SQL 6

2.9. Apache Tomcat8.0 6

第三章、需求分析 9

3.1. 编写目的 9

3.2. 需求描述 9

3.3. 运行环境需求分析 10

第四章、概要设计 11

**4.1** 11

4.1. 系统逻辑架构 11

4.2. 系统功能模块设计 12

4.3. 角色权限设计 16

4.4. 系统数据库概要设计 16

4.5. 数据库的实体设计 17

第五章、详细设计 29

5.1. 系统层次设计 29

5.2. 系统项目规范 29

5.3. 系统功能类图设计 30

5.4. 系统业务接口设计 30

5.5. 系统交互动作接口设计 36

5.6. 系统UI设计 41

第六章、系统的实现与测试 45

6.1. 环境搭建 45

6.2. 数据层的实现 45

6.3. 数据连接层的实现 45

6.4. 基于Hibernate的数据库 46

6.5. 数据操作层的实现 46

6.6. 数据操作层的调用实现 47

6.7. 实体交换层的实现 47

6.8. 系统业务层的实现 49

6.9. 信息业务层的调用实现 49

6.10. 控制层的实现 50

6.11. 控制层的调用实现 50

6.12. 表示层的实现 50

6.13. 软件测试用例说明 51

6.14. 测试报告 54

结 论 55

致 谢 56

参考文献 57

引言

学生信息化管理对于现在的各个高校至关重要，老式的高校学生管理存在着信息处理效率慢，管理人员工作负担大，信息储存不方便的而产生的各种问题，因此，现在的高校都急需改变学生管理方式。而采用学生信息系统，能够更加信息化、规范化的管理学生信息，能大大提高高校管理效率。

随着信息化时代的快速发展，信息化管理将会越发成熟，它能够提高各个行业的工作水平与工作效率，带动经济的快速发展。老式的传统信息管理方式必定会被计算机处理的信息化管理所取代。

该系统能降低管理成本，由于采用本系统的原因，使辅导员管理工作一改往日埋在各种表格中的被动局面，基本实现了无纸化办公，根除了很多日常管理中的浪费现象，体现出现代化学生管理的优势。

该项目可发布与高校网内，用于规范，合理的管理学生信息，减轻管理人员的工作负担，增加信息交互的效率，可节约纸张资源。对于促进高校学生管理有着重大意义。该项目使我增加大量开发经验，熟悉项目开发的完整流程，增强自己的开发能力，对于复杂的业务能够更好的理解。

学生管理系统中，管理员能够实现对学校、学院、专业、班级的增删改，以及对菜单的增删改，对用户权限的开启，禁用，对任课信息的管理和公告信息的管理。

学生用户登录后能发表自己的信息，展示在前端平台上，能够根据条件查询自己各个科目的成绩，可以根据学期统一查询成绩。

教师用户登录后对学生信息进行审核，修改。对自己授课的学生进行打分与评价。

### **第一章、绪 论**

#### 开发背景

现如今正是信息化和科技高速发展的时代，各行各业都离不开信息化处理，信息化为人们带来的好处随处可见，帮助人们减轻工作负担，减少工作流程，尤其是对于复杂的信息管理，信息化管理能充分发挥它的优势。

信息管理系统是进行对信息的进行储存、查询、修改合理操作的系统。它是科学管理和科学技术的发展而形成的。在各大高校中，一个完善的学生信息管理系统显得尤为重要。而它的功能对于学校的决策者和管理者也是至关重要的，完善的学生信息管理系统能为高校管理者提供更便捷且准确的学生信息管理方式。传统的学生信息管理方式为人工的处理方式，运用纸张储存学生信息与办公，人为的传递信息。在着工作效率低、工作流程复杂、信息保密性差等很多缺点，在信息的存储和学生信息的查询时都会出现不少问题。

与传统的学生信息管理相比，使用计算机对学生信息来进行管理，具有着比传统手工管理所无法比拟的优点，如，精准查询迅速、可靠性高、存储方便、安全性高、寿命长、成本低等。这些优点能极大提高管理者的工作效率，也是学校科学化，正规化管理的必要条件。因此开发这样一套管理软件是很有必要的。

#### 开发意义

开发学生信息管理系统，使用计算机对学生信息进行管理，可提高高校管理人员办公效率，减轻高校人员的工作流程，节省时间，实现更加规范的学生信息管理，更加方便管理人员的维护和储存学生的复杂信息。完整的系统可在学校的管理中发挥着巨大作用。能够实现工作流程的信息化，提高工作效率和减轻工作负担。

学生信息管理系统为高校管理着提供便捷的学生信息处理功能，方便管理者对学生信息的进行创建、查询、修改、删除、统计、维护等操作，可以减轻学校职工的工作压力，更多的对教务，教学的各个信息进行管理。对于高校规范管理有巨大意义。[[3]](#endnote-3)[1]

#### 可行性分析

##### 经济可行性分析

按照本系统的开发方案，按开发计划进行估算，办公用品的消耗，办公设备的消耗，开发人员的生活与维护（包括水、电、房、工作餐）等。项目总体预估耗费10000元。本系统非收益性质开发，重要的是为解决学校进行学生信息管理的需要。

##### 技术可行性分析

本系统的开发环境为Windoows10系统，主要使用了MyEclipse、Tomcat8.0和navicat等工具做开发。该系统主要使用的技术是JavaWeb技术、SpringMVC、Hibernate、JSP和Layui前端框架。所用到的技术都已掌握，该系统只是把所学技术合理应用即可开发，所以技术上是完全可行的。

##### 应用价值的可行性分析

开发学生信息管理系统，使用计算机对学生信息进行管理，可提高高校管理人员办公效率，减轻高校人员的工作流程，节省时间，实现更加规范的学生信息管理，更加方便管理人员的维护和储存学生的复杂信息。完整的系统可在学校的管理中发挥巨大作用。实现工作流程的信息化，提高工作效率和减轻工作负担。

学生信息系统能够有效地实现学生信息管理的信息化以及学生信息储存的规范化和高校管理人员的工作流程的简化，减轻高校管理人员的工作负担，高效率、规范化地处理、储存大量的学生信息，促进学校对于学生的规范管理。

#### 研究内容介绍

课题首先研究基于SSH的学生信息管理系统的设计与实现。该系统主要实现了学生信息管理中的基本信息管理、学校管理、课程管理、学生信息管理以及系统管理，系统的主流程为教师对学生信息的管理统计和对学生成绩的打分与对学生的评价，在主流程中，包含多个分流程及细节操作。[[4]](#endnote-4)[2]

本课题实现网站所涉及到的技术：SSH框架、日志管理、权限管理、JSP、Layui框架和MySQL数据库。其中艰难部分为学生信息的统计与采集，教师用户与课程信息和学生信息以及班级的映射关系。以及教师用户对课程的绑定以及对学生用户进行打分和点评的复杂逻辑关系。

最后本课题将对系统的开发进行系统文档的编写。1、项目的说明文档，将对系统的参数、规范进行描述，以便于今后对系统功能的更新和维护；2、系统的使用说明文档，对如何正确使用系统进行详细讲解，让用户能够更直观的了解系统的操作。

### **第二章、相关技术概论**



#### B/S架构

B/S（Brower/Server,浏览器/服务器）模式又称B/S结构，是Web兴起后的一种网络结构模式。工作模式是浏览器请求服务器响应，并返回给Web服务器，Web服务器又返回给用户。

#### JavaWeb网站技术

JavaWeb是一种技术总和，顾名思义就是后台处理用java代码编写的web项目， Java开发后端技术包含spring MVC框架、Spring容器、分布式session管理等技术，Web前端中可以使用JSP、html、jQuery、vue、layui等技术。

#### Spring容器

Spring拥有两个接口：ApplicationContext、BeanFactory， ApplicationContext接口用来完成Spring容器的配置和初始化，Spring容器又是一个bean工厂，对bean进行实例、获取、销毁。

#### SpringMVC框架

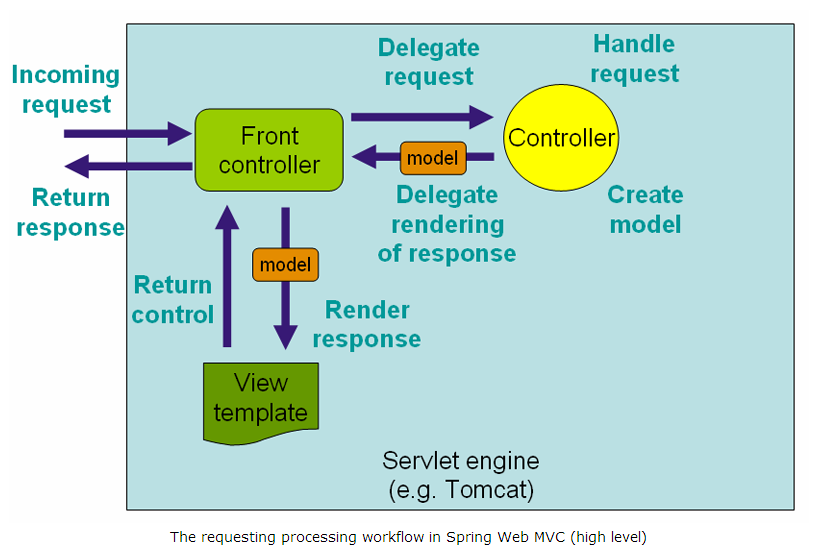
Spring MVC属于Spring内的一个模块，所以它享有 Spring IoC容器紧密结合Spring松耦合等特点，因此它有Spring的所有优点。Spring MVC也是基于mvc的一个WEB框架，此外它采用松散耦合可插拔式组件结构，Spring MVC的表示层更方便数据的传输，Spring MVC的控制器与Strurs类似，接受外部请求，解析参数并传给业务层处理逻辑。

图2..**错误!使用“开始”选项卡将 0 应用于要在此处显示的文字。**.1 Spring MVC 结构图

#### Hibernate

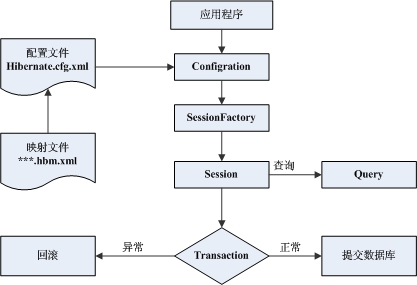
Hibernate是一种ORM框架，它对JDBC进行轻量化处理。可以自动生成数据库语句，自动执行。Hibernate继承JDBC的属性，所以JDBC可使用的场所，Hibernate皆可以使用。使程序开发工作者更方便。Hibernate的运行过程如图所示：

图2.2 hibernate结构图

#### Layui前端框架

layui是一款开源的模块化前端UI框架，它提供了丰富的内置模块，每个模块都可以通过模块化的方式按照调用者的需要加载。layui由布局、字体图标、按钮、表单、导航条、选项卡、表格等元素组成。[[5]](#endnote-5)[3]

#### 开发工具MyEclipse

MyEclipse是 Genuitec公司开发出来的一款企业级开发环境，它是以eclipse为基础，然后再加上一些属于它自己的插件，再进一步的进行开发，才形成了这个功能强大的开发环境。

#### 服务器MY SQL

MySQL 是一种关联数据库管理系统，比之SQL服务器具有更强大的性能，并且MySQL服务器的可扩展性与开放的数据引擎更让SQL黯然失色，相比与SQL减少了额外的复杂操作、磁盘处理、内存损耗等等。[[6]](#endnote-6)[4]

#### Apache Tomcat8.0

Apache Tomcat是由Apache开发的免费且开源的轻量级WEB应用服务器，8.0更是相比其他版本具有稳定性，是初学者开发和调试JSP 程序的首选,它的主要优点是技术先进并且性能稳定。[[7]](#endnote-7)[5]

### **第三章、需求分析**



#### 编写目的

该项目可发布与高校网内，用于规范，合理的管理学生信息，减轻管理人员的工作负担，增加信息交互的效率，可节约纸张资源。对于促进高校学生管理有着重大意义。该项目使我增加大量开发经验，熟悉项目开发的完整流程，增强自己的开发能力，对于复杂的业务能够更好的理解。[[8]](#endnote-8)[6]

#### 需求描述

##### 学生用户需求

图3.1 用户用例图

学生用户登录可查询，修改，提交自己的基本信息，提交后登录页面可查看自己个人信息并对个人进行编辑操作。并可以根据条件查看自己各个学期，各个科目的成绩及教师评价。

##### 教师用户需求

教师登录后能够按条件查询学生信息，显示学生信息列表，并能对学生按班级查询自己授课的科目进行学生的评价与打分，还能够对学期信息查询、修改对课程信息进行查询、修改、对已授权教师进行课程的绑定。

##### 管理员用户需求

管理员能够实现对学院、专业、班级的增删改，对课程信息，任课信息的添加，修改，删除，以及对教师身份真实性的校验和用户状态的启用与禁用，对公告信息的管理。对用户进行添加、修改、删除操作，对角色进行添加、修改、删除操作，对角色的各个菜单权限进行授权与禁用，对菜单进行添加、修改、删除，还可查看系统的登录与异常日志信息。

#### 运行环境需求分析

该系统的数据库采用Mysql，服务器在阿里云中。

使用该系统的计算机所需最低配置：

CPU： P4 3.0GHz

内存： 4G以及4G以上

硬盘： 160G以上

网络： 有线/无线网卡

该系统一般兼容现市场上的主流浏览器，较推荐使用Google浏览器。

该系统网站端主要采用B/S模式。运行原理大致是，用户在浏览器上操作，浏览器发送请求给服务器；服务器端接收并处理请求，应用服务器端通常使用服务器端技术，如JSP等，对请求进行数据处理，并产生响应；服务器端把用户请求得到的数据返回给浏览器，html的形式展现给用户。

### **第四章、概要设计**

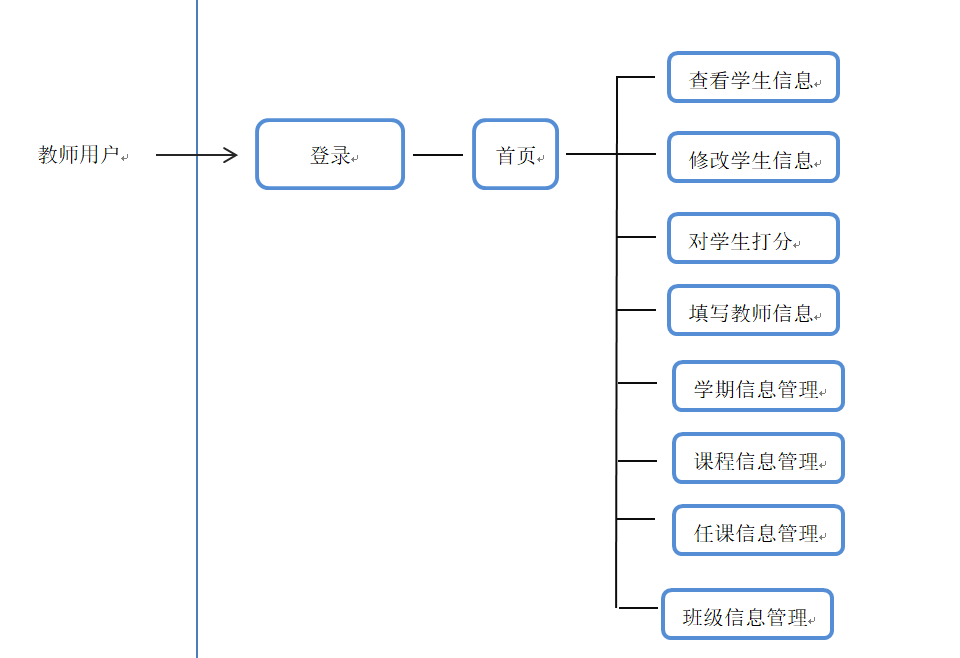
图4.1 学生登录用例图

图4.1 学生用户功能分析图



#### 系统逻辑架构

学生用户登录/注册，判断成功后进入首页，学生可填写，修改自己的个人信息，查看自己每个科目成绩，查看每个学期的成绩。



教师用户登录/注册，判断成功后进入首页，教师用户可填写自己基本信息，根据条件查询学生信息，列出学生列表，选中学生，可修改学生信息。可根据学院、专业、班级查询，可根据自己所授课进行查询。查询后对学生成绩进行评分，对学生进行评价。教师登录后进行权限判断，可管理学期信息，课程信息，任课信息以及学院、专业、班级信息。

图4.2 教师用户功能分析图

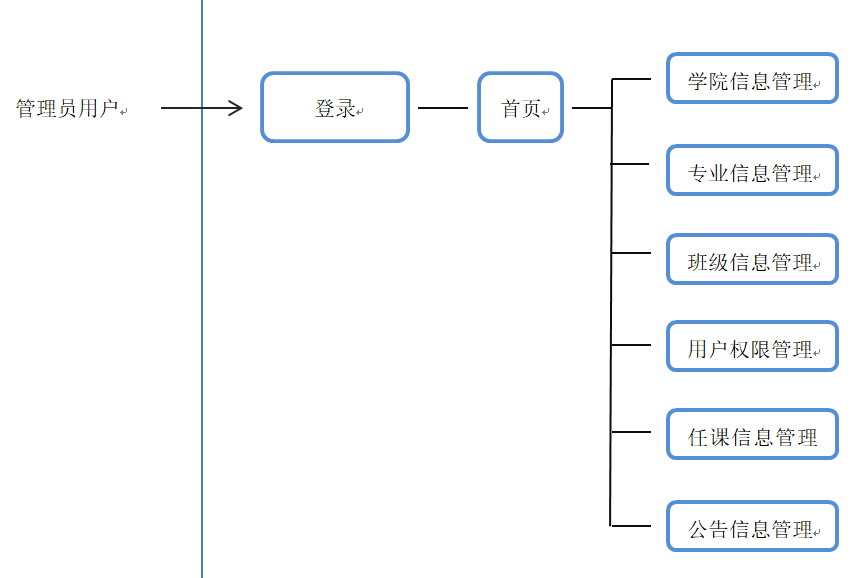
管理员用户登录，判断成功后进入首页，可对学院信息进行增删，查改以及对学院院长的绑定，对专业信息的增删，查改，对班级信息的增删查改，对班级辅导的绑定，对各个用户菜单权限的开启，禁用，对教师用户任课信息的增删，查改，对学校公告信息的发布，删除，修改。[[9]](#endnote-9)[7]

图4.3 管理员用户功能分析图

#### 系统功能模块设计

图4.3 学生注册用例图

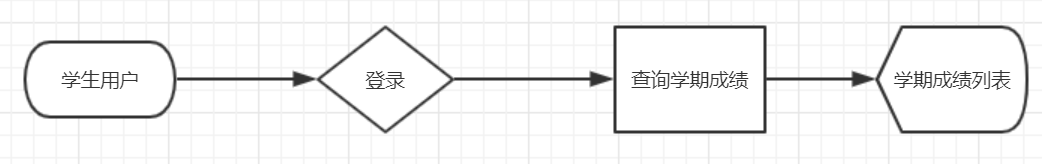
学生用户初次进行注册，注册成功后填写完善个人信息，完成提交并可登录进入首页查看信息。

图4.4 学生查询用例图

学生用户进行登录，登录成功后可按条件查询每个学期成绩，成功显示查询到的学期成绩列表。

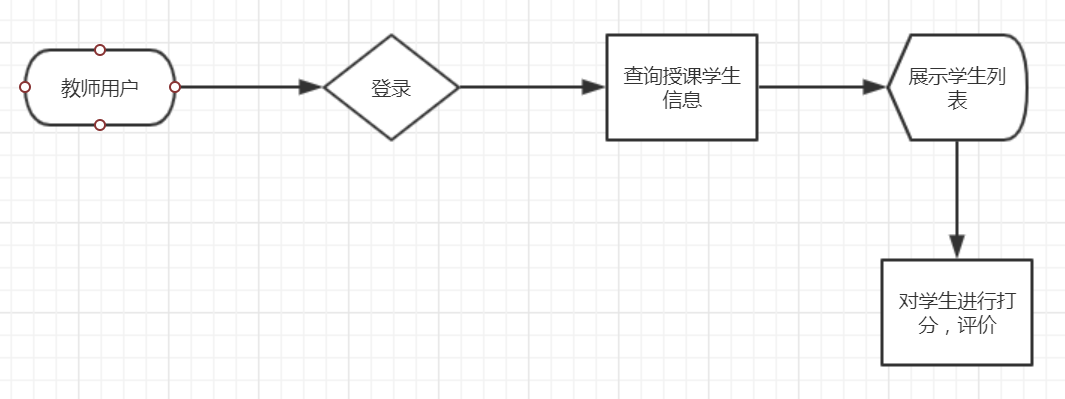
教师用户登录，判断成功后可按条件查询自己授课学生，查询后展示学生列表，可对学生成绩进行打分，对学生进行评价。

图4.5 教师查询用例图

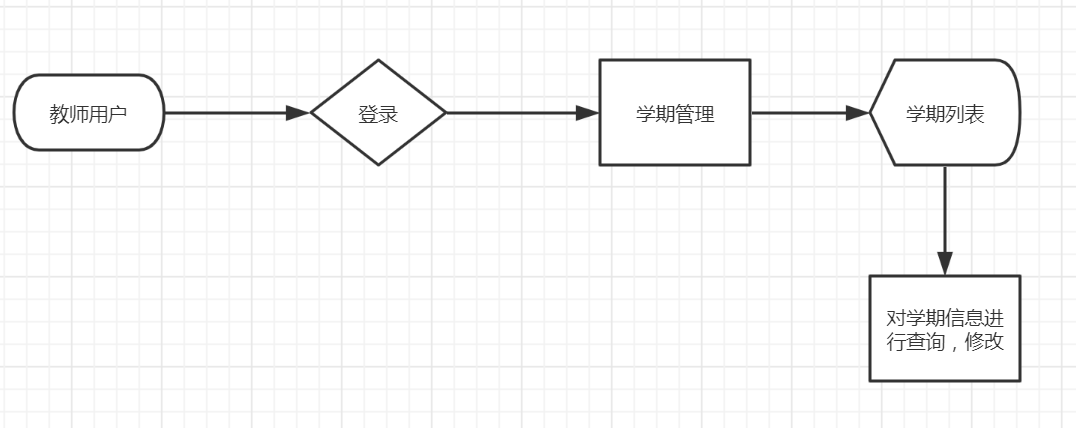
教师用户登录，判断成功后可按条件查询学期信息，查询后展示学期列表，可对学期信息进行查询，修改。

图4.6 教师学期管理用例

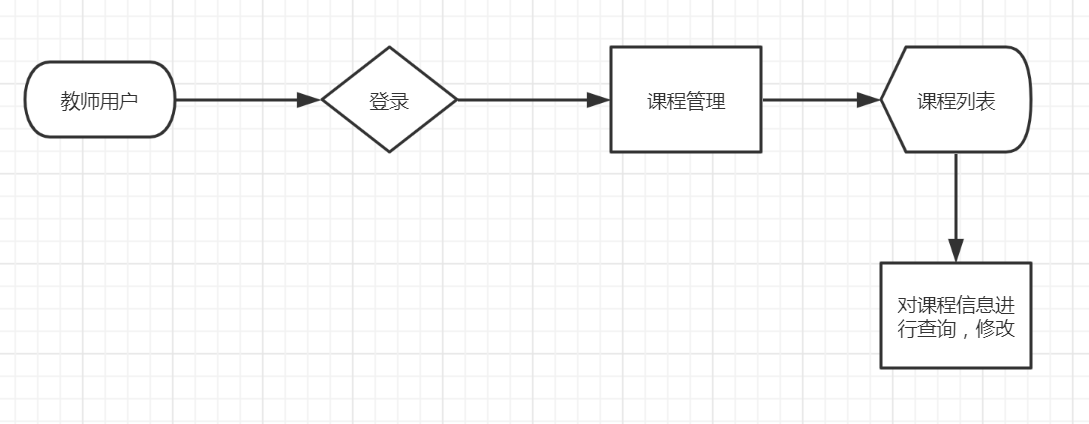
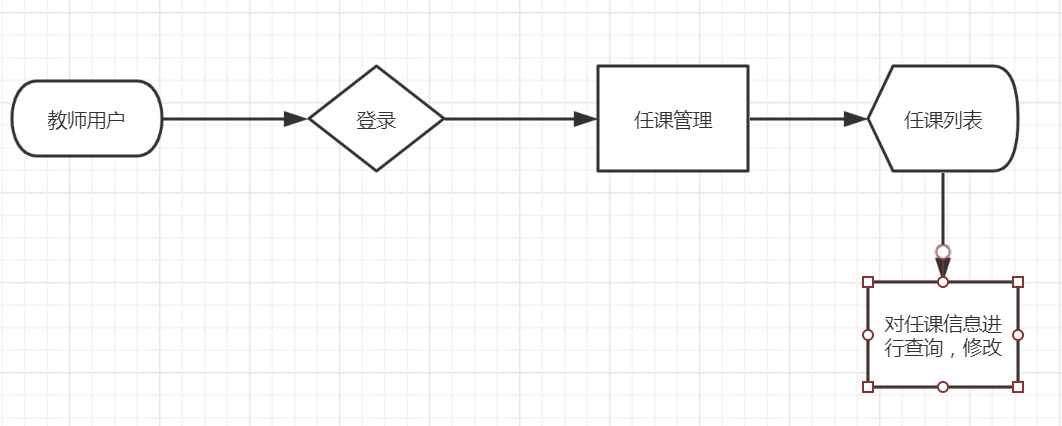
教师用户登录，判断成功后可按条件查询课程信息，查询后展示课程信息列表，可对课程信息进行添加，修改，删除。

图4.7 教师课程管理用例图

教师用户登录，判断成功后可按条件查询任课信息，查询后展示任课信息列表，可对任课信息进行添加，修改，删除。

图4.8 教师任课管理用例

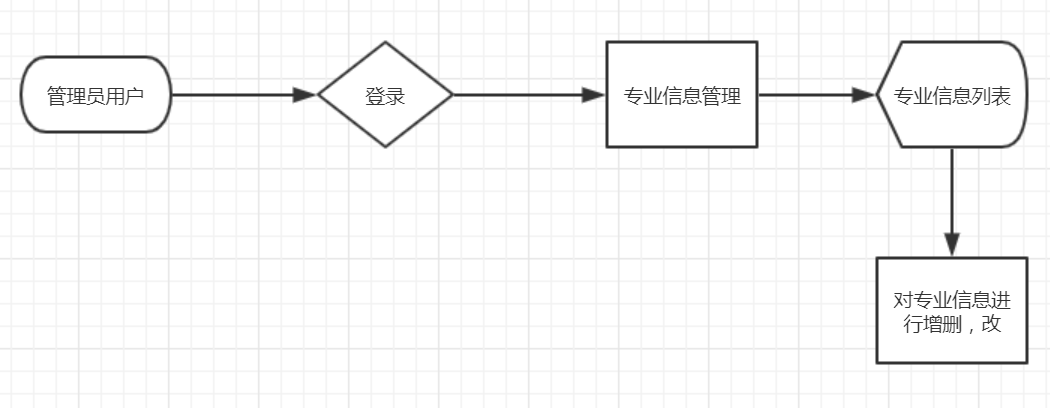
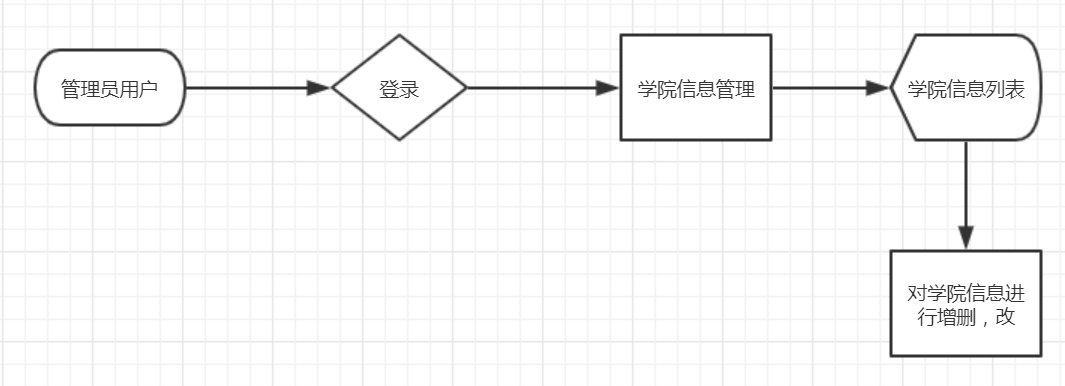
管理员用后登录，判断成功后可进入学院信息管理，获得学院信息列表，可对学院信息进行添加，修改，删除，对学院院长进行修改。

图4.9 管理员学院管理用例

管理员用后登录，判断成功后可进入专业信息管理，获得专业信息列表，可对专业信息进行添加，修改，删除。对专业科长进行修改。

图4.10 管理员专业管理用例

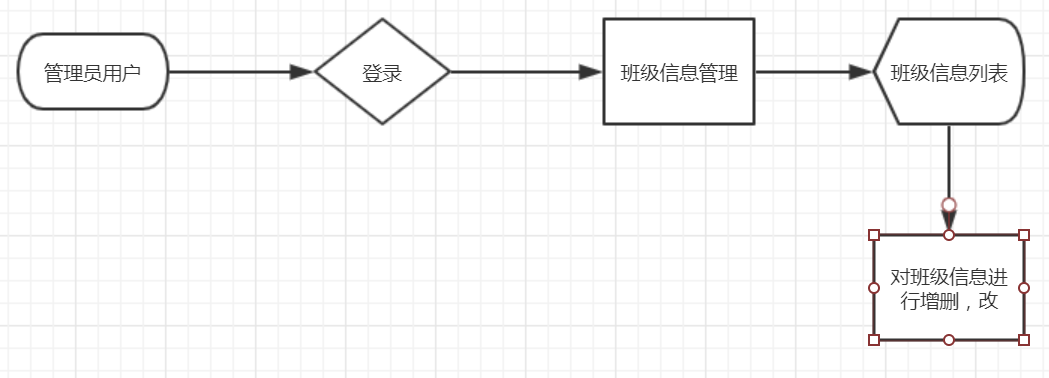
管理员用后登录，判断成功后可进入班级信息管理，获得班级信息列表，可对班级信息进行添加，修改，删除，对班级进行辅导员的修改。

图4.11 管理员班级管理用例

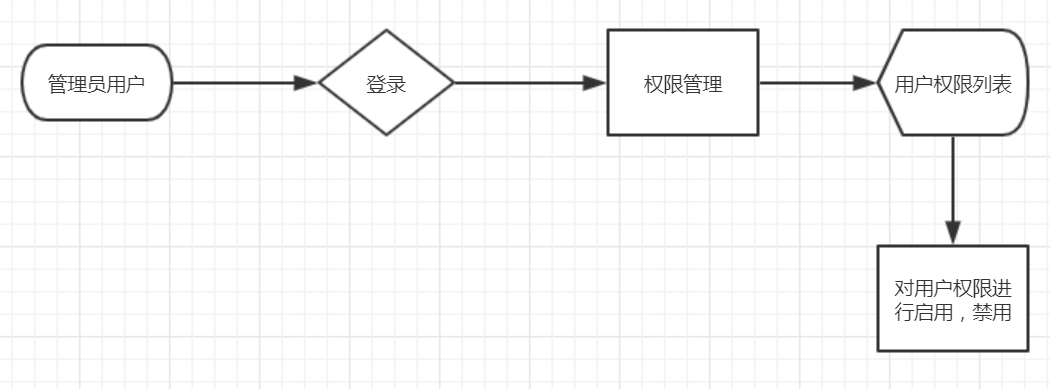
管理员用后登录，判断成功后可进入权限管理，获得用户权限列表，可按条件查询用户权限，对用户权限进行启用，禁用。

图4.12 管理员权限管理用例

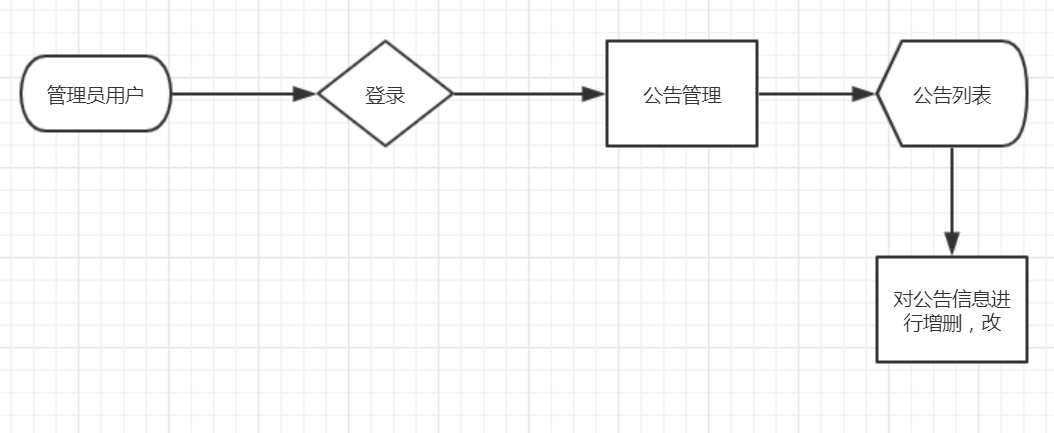
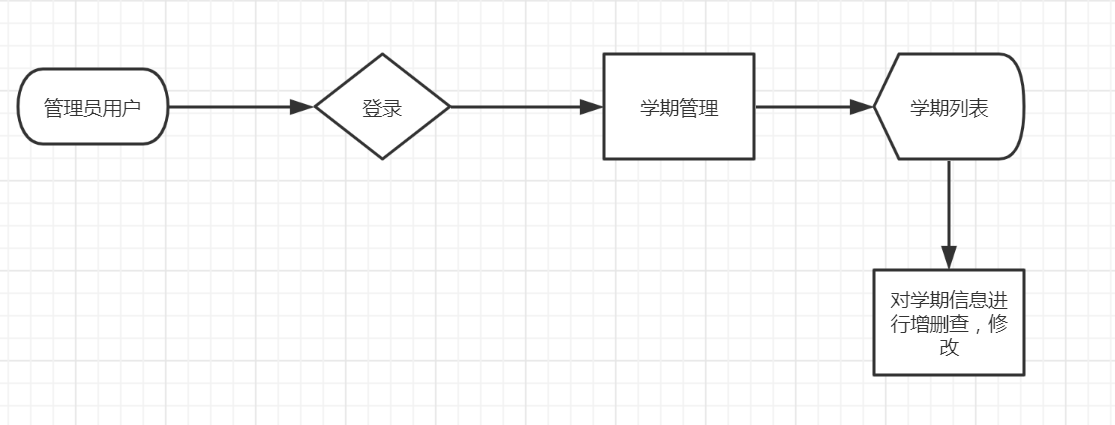
管理员用后登录，判断成功后可进入公告管理，获得公告列表，可按公告标题查询公告信息，对公告信息进行添加，修改，删除。

图4.14 管理员学期管理用例图

图4.13 管理员公告管理用例

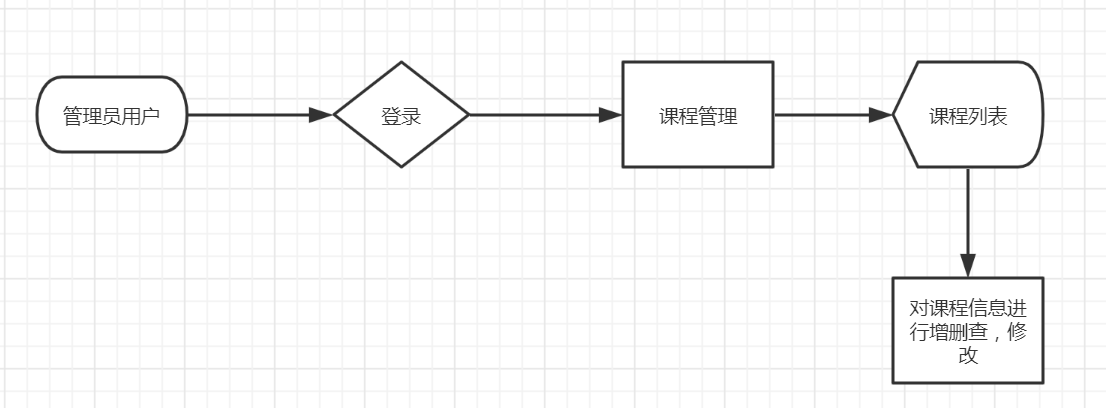
管理员用后登录，判断成功后可进入学期管理，获得学期列表，可按学期名称查询学期信息，对学期信息进行添加，修改，删除

图4.15 管理员课程管理用例图

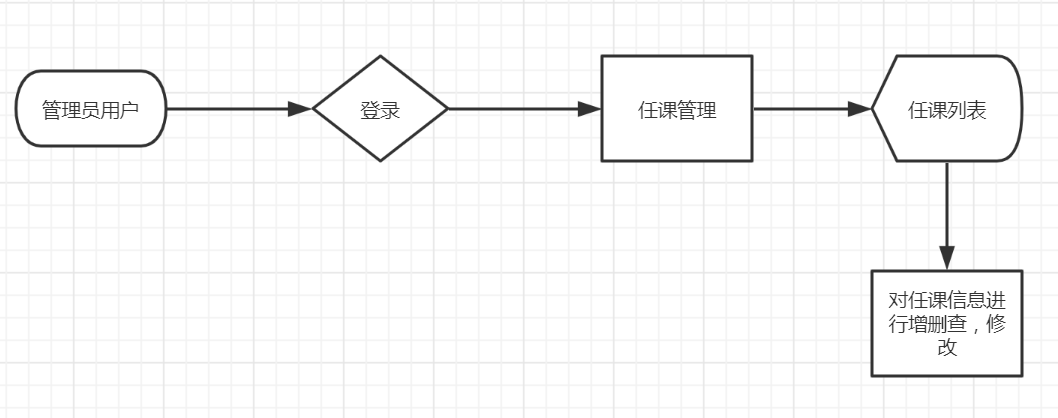
管理员用后登录，判断成功后可进入课程管理，获得课程列表，可按课程名称查询课程信息，对课程信息进行添加，修改，删除。

图4.16 管理员任课管理用例图

管理员用后登录，判断成功后可进入任课管理，获得任课列表，可按课程名称以及教师查询任课信息，对任课信息进行添加，修改，删除。

#### 角色权限设计

权限分配系统是每个系统中必不可少的最基本系统。每个角色都能拥有操作系统不同功能模块的权限，更方便于系统的管理员授权，如高级管理员可以管理和操作整个系统的所有功能。

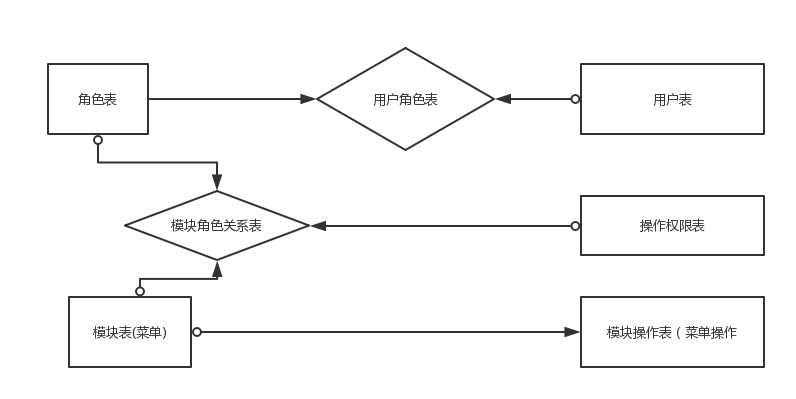
在权限设计中，有三大核心表即用户表、角色表、权限表，另外还有角色与用户，角色与权限的映射表。根据用户名查询是否拥有某个操作权限。在用户注册的时候根据其角色授予系统各个模块的权限，在用户登录后就可以给出用户所对应的权限。权限结构设计如图所示：

图4.17 角色权限结构图

该系统的角色设置更人性化，管理员用户可自行添加，角色的权限由管理员进行启用。禁用。

#### 系统数据库概要设计

图4.15 数据库ER图

#### 数据库的实体设计

图4.17 数据库ER图

##### 术语定义

表格“备注”列表说明：“\*”表示唯一主键，要做索引，“1++”表示值为系统自增，步长为1。

表名前缀，表示table\_

V\_ : 视图前缀，表示view\_

up\_ : 存储过程前缀，表示userprocedure\_

utri\_ : 触发器前缀，表示usertrigger

pk\_ : 表列的主键约束前缀

fk\_ : 表列的外键约束前缀

ck\_ : 表列的check约束前缀

def\_ : 表列的默认约束前缀

##### 表详细设计

用户信息表T\_User:

表4.1 用户信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 字段标识 | 类型 | 长度 | 备注 |
| 用户信息ID | userID | int |  | \* |
| 用户名 | account | nvarchar | 80 |  |
| 密码 | pwd | nvarchar | 40 |  |
| 用户状态 | usertype | bit | 1 |  |
| 角色id | roleIID | int |  | fk |
| 是否删除 | isdel | int |  |  |

角色信息表T\_RoleB:

表4.2 角色信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 字段标识 | 类型 | 长度 | 备注 |
| 角色ID | roleID | int |  | \* 1++ |
| 角色名称 | rolename | nvarchar | 80 |  |
| 备注 | remark | ntext |  |  |
| 是否删除 | isdel | int |  |  |

角色权限表T\_RoleSystemModel:

表 4.3 角色权限表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 字段标识 | 类型 | 长度 | 备注 |
| ID | id | int |  | \* 1++ |
| 角色ID | roleID | int |  | fk |
| 菜单id | sysid | int |  | fk |
| 是否删除 | isdel | int |  |  |

菜单表T\_SystemModel:

表4.4 菜单表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 字段标识 | 类型 | 长度 | 备注 |
| ID | id | int |  | \* 1++ |
| 名称 | name | nvarchar | 80 |  |
| 中文名称 | chinesename | nvarchar | 80 |  |
| 链接 | navurl | nvarchar | 80 |  |
| 等级 | deepth | int |  |  |
| 父级 | parentid | int |  |  |
| 序号 | displayorder | int |  |  |
| 图标链接 | imageurl | nvarchar | 80 |  |
| 是否删除 | isdelete | int |  |  |

日志信息表T\_SystemLog:

表4.5 日志信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 字段标识 | 类型 | 长度 | 备注 |
| ID | id | int |  | \* 1++ |
| 类型 | opertype | nvarchar | 80 |  |
| 操作 | description | nvarchar | 80 |  |
| 操作地址 | opermethod | nvarchar | 80 |  |
| 请求IP | requestip | nvarchar | 80 |  |
| 异常代码 | exceptioncole | nvarchar | 80 |  |
| 异常详细信息 | exceptiondetail | nvarchar | 80 |  |
| 操作用户 | params | nvarchar | 80 |  |
| 创建时间 | createdate | datetime |  |  |
| 创建人 | createby | nvarchar | 80 |  |

学生信息表T\_Student:

表4.6 学生信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 字段标识 | 类型 | 长度 | 备注 |
| ID | studentid | int |  | \* 1++ |
| 用户id | userID | nvarchar |  | fk |
| 学生姓名 | stuName | nvarchar | 80 |  |
| 学生性别 | stusex | nvarchar | 10 |  |
| 学生学号 | stunumber | nvarchar | 80 |  |
| 出生日期 | stubirthday | datetime | 80 |  |
| 身份证号 | stucard | nvarchar | 80 |  |
| 邮箱 | email | nvarchar | 80 |  |
| 联系方式 | phone | nvarchar | 80 |  |
| 班级 | classid |  |  |  |
| 是否删除 | isdelete | int |  |  |

教师信息表T\_ Teacher:

表4.7 教师信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 字段标识 | 类型 | 长度 | 备注 |
| ID | teacherid | int |  | \* 1++ |
| 用户id | userID | nvarchar |  | fk |
| 教师姓名 | teaName | nvarchar | 80 |  |
| 教师性别 | teasex | nvarchar | 10 |  |
| 联系方式 | teaphone | nvarchar | 80 |  |
| 确认状态 | auditstatus | bit | 1 |  |
| 是否删除 | isdelete | int |  |  |

公告表T\_ Notice:

表4.8 公告表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 字段标识 | 类型 | 长度 | 备注 |
| ID | noticeid | int |  | \* 1++ |
| 公告主题 | noticeName | nvarchar | 80 |  |
| 公告内容 | noticecontent | text |  |  |
| 备注 | remarks | text |  |  |
| 创建人 | userID | nvarchar |  | fk |
| 创建时间 | createdate | datetime |  |  |
| 是否删除 | isdelete | int |  |  |

学院表T\_ College:

表4.9 学院表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 字段标识 | 类型 | 长度 | 备注 |
| ID | collegeid | int |  | \* 1++ |
| 学院名称 | collegeName | nvarchar | 80 |  |
| 院长 | userID | nvarchar |  | fk |
| 是否删除 | isdelete | int |  |  |

专业表T\_ major:

表4.10 专业表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 字段标识 | 类型 | 长度 | 备注 |
| ID | majorid | int |  | \* 1++ |
| 班级名称 | majorName | nvarchar | 80 |  |
| 所属学院 | collegeid | int |  | fk |
| 科长 | userID | nvarchar |  | fk |
| 是否删除 | isdelete | int |  |  |

班级表T\_ Class:

表4.11 班级表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 字段标识 | 类型 | 长度 | 备注 |
| ID | classid | int |  | \* 1++ |
| 班级名称 | className | nvarchar | 80 |  |
| 所属专业 | majorid | int |  | fk |
| 辅导员 | userID | nvarchar |  | fk |
| 是否删除 | isdelete | int |  |  |

学期表T\_ Semester:

表4.12 学期表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 字段标识 | 类型 | 长度 | 备注 |
| ID | semesterid | int |  | \* 1++ |
| 学期名称 | semesterName | nvarchar | 80 |  |
| 开始时间 | startdate | datetime |  |  |
| 结束时间 | enddate | datetime |  |  |
| 是否删除 | isdelete | int |  |  |

课程表T\_ Curriculum:

表4.13 课程表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 字段标识 | 类型 | 长度 | 备注 |
| ID | curriculumid | int |  | \* |
| 课程名称 | curriculumName | nvarchar | 80 |  |
| 所属学期 | semesterid | int |  | fk |
| 课程介绍 | introduce | Text |  |  |
| 备注 | remarks | text |  |  |
| 是否删除 | isdelete | int |  |  |

任课表T\_ Course:

表4.14 任课表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 字段标识 | 类型 | 长度 | 备注 |
| ID | courseid | int |  | \* |
| 课程id | curriculumid | int |  | fk |
| 教师id | teacherid | int |  | fk |
| 班级id | classid | int |  | fk |
| 备注 | remakes | text |  |  |
| 是否删除 | isdelete | int |  |  |

成绩表T\_ Sheet:

表4.15 成绩表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 字段标识 | 类型 | 长度 | 备注 |
| ID | sheetid | int |  | \* |
| 任课id | courseid | int |  | fk |
| 学生id | studentid | int |  | fk |
| 分数 | fraction | doeble |  |  |
| 评价 | curriculumName | nvarchar | 80 |  |
| 是否删除 | isdelete | int |  |  |

##### 视图详细设计

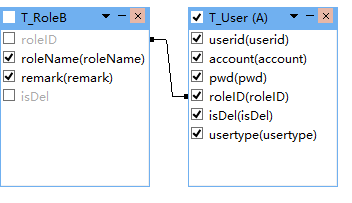
用户视图V\_ AdminUser: 

图4.16 用户视图关系图

表4.16 用户视图表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 字段标识 | 类型 | 长度 | 备注 |
| 用户信息ID | userID | int |  | \* |
| 用户名 | account | nvarchar | 80 |  |
| 密码 | pwd | nvarchar | 40 |  |
| 用户状态 | usertype | bit | 1 |  |
| 角色id | roleIID | int |  |  |
| 是否删除 | isdel | int |  |  |
| 角色名称 | rolename | nvarchar | 80 |  |
| 备注 | remark | ntext |  |  |

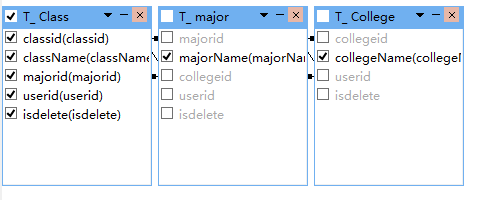
班级视图V\_ class:

图4.17 班级视图关系图

表4.17 班级视图表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 字段标识 | 类型 | 长度 | 备注 |
| ID | classid | int |  | \* 1++ |
| 班级名称 | className | nvarchar | 80 |  |
| 所属专业 | majorid | int |  |  |
| 辅导员 | userID | nvarchar |  |  |
| 专业名称 | majorName | nvarchar | 80 |  |
| 学院名称 | collegeName | nvarchar | 80 |  |

任课视图V\_Course:

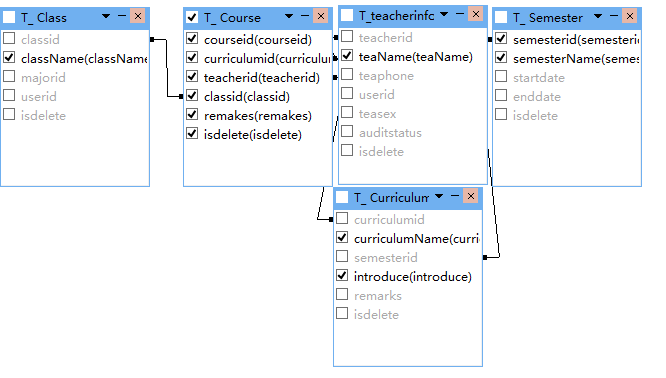


图4.18 任课视图关系图

表4.18 任课视图表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 字段标识 | 类型 | 长度 | 备注 |
| ID | courseid | int |  | \* |
| 课程id | curriculumid | int |  |  |
| 教师id | teacherid | int |  |  |
| 班级id | classid | int |  |  |
| 备注 | remakes | text |  |  |
| 是否删除 | isdelete | int |  |  |
| 班级名称 | className | nvarchar | 80 |  |
| 教师姓名 | teaName | nvarchar | 80 |  |
| ID | semesterid | int |  | \* 1++ |
| 学期名称 | semesterName | nvarchar | 80 |  |
| 课程名称 | curriculumName | nvarchar | 80 |  |
| 课程介绍 | introduce | Text |  |  |

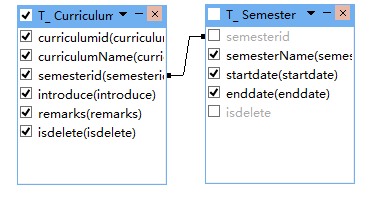
课程视图V\_Curriculum:

图4.19 课程视图关系图

表4.19 课程视图表

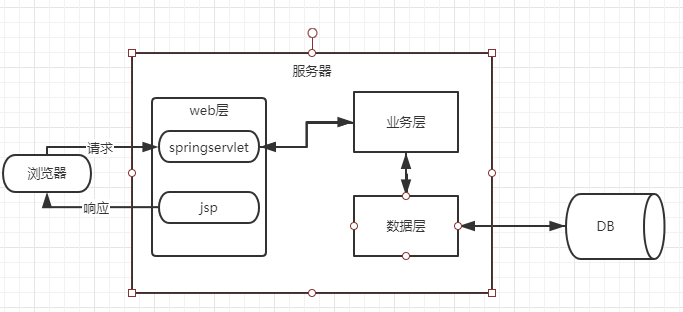
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 字段标识 | 类型 | 长度 | 备注 |
| ID | curriculumid | int |  | \* |
| 课程名称 | curriculumName | nvarchar | 80 |  |
| 所属学期 | semesterid | int |  |  |
| 课程介绍 | introduce | Text |  |  |
| 备注 | remarks | text |  |  |
| 是否删除 | isdelete | int |  |  |
| 学期名称 | semesterName | nvarchar | 80 |  |
| 开始时间 | startdate | datetime |  |  |
| 结束时间 | enddate | datetime |  |  |

### 第五章、详细设计



#### 系统层次设计

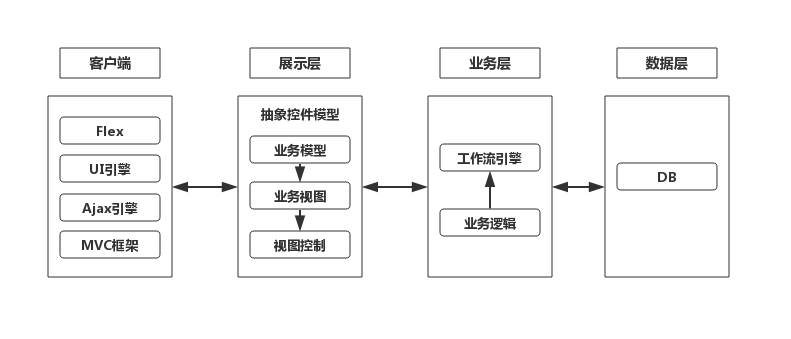
系统分层：是对系统中的类进行分组的一种特定方式。其优点在于，限制子系统间依赖关系，让开发人员能够更好的去维护系统。

系统分层的三种常用方法：

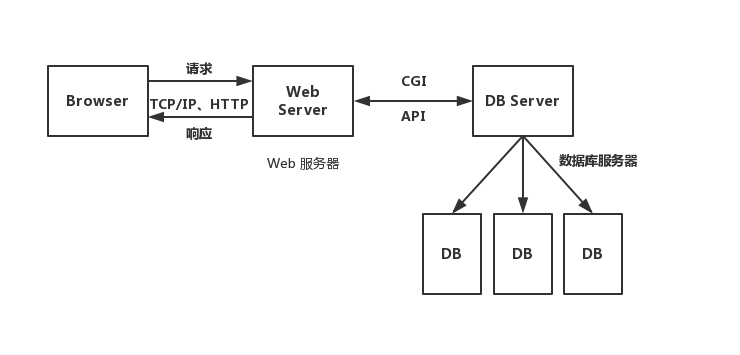
##### 三层服务：用户层、业务层、数据层；

图5.1 三层服务结构图

##### 多层结构的技术组成模型：展示层、业务层、数据层；



##### B/S架构的模式为：表示层、业务层、服务接口层、数据链接层；



#### 系统项目规范

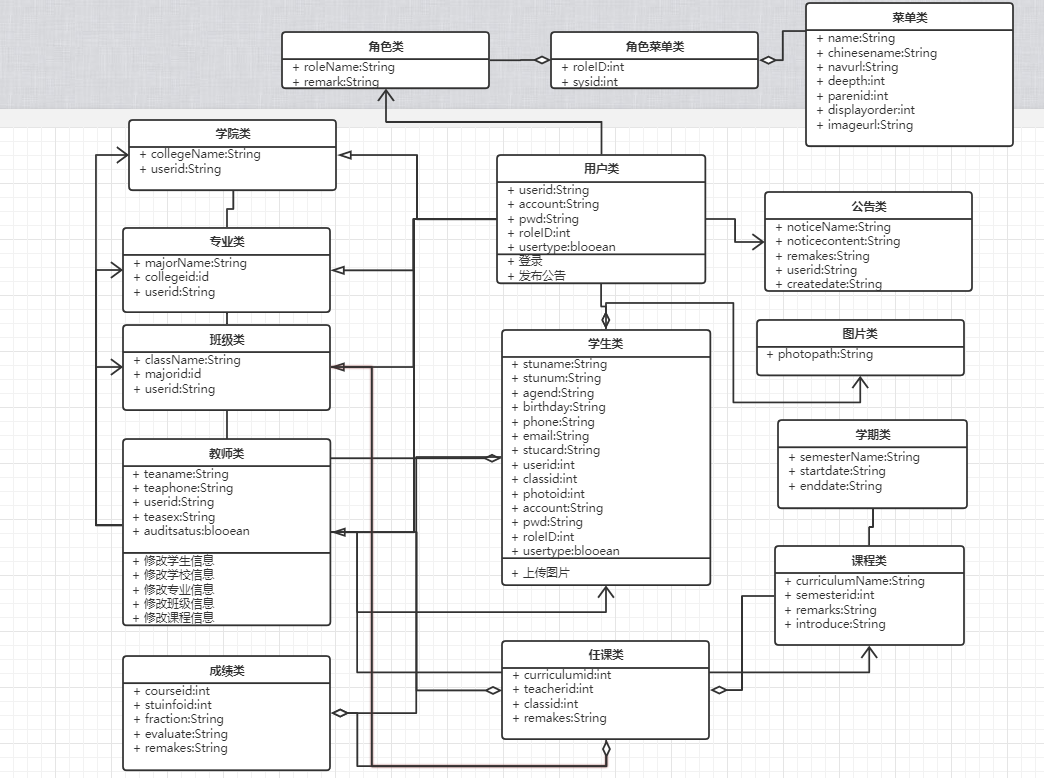
##### 项目文件命名规范

项目文件命名规范如下表所示：

表5.1 项目文件命名规范表

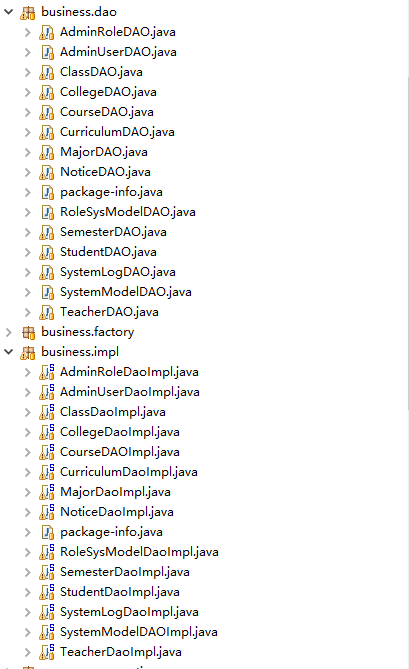
|  |  |
| --- | --- |
| 标识符 | 说明 |
| T实体类名 | 表实体类，如T\_AdminUser表的实体类为TAdminUserModel |
| V实体类名 | 视图实体类，如V\_AdminUser视图的实体类为VAdminUserModel |
| 业务逻辑类名DAO | 业务逻辑接口类,如AdminUserDao |
| 业务逻辑类名Impl | 业务逻辑实现类，如AdminUserDaoImpl |
| 控制类类名controller | 业务控制类，如后台用户管理的控制类AdminUserController |

#### 系统功能类图设计

类图主要用于描述系统的结构化设计。它也是最常用的UML图，用类图可以显示出类、接口以及它们之间的静态结构和关系。泛化、实现、关联、聚合、组合、依赖是它较常见的几种关系。

#### 系统业务接口设计

图5.2 系统UML图

系统业务接口设计如下所示，表中需要实现的方法例举了出来，在接口说明中需要实现的方法会详细说明。详细说明如下：

AdminUserDAO（后台用户信息管理业务类接口）:

图5.3 业务接口图

表5.2后台用户信息管理业务类接口表

|  |  |  |  |
| --- | --- | --- | --- |
| 接口名 | 参数 | 返回值 | 描述 |
| getaAdminUserList | wherecondition、page、pageSize（查询条件、当前页、每页条数） | 成功返回List<VAdminUser>对象，失败返回null | 用户信息列表(带分页) |
| getaAdminUserList | roleID（角色id） | 成功返回List<TroleB>对象，失败返回null | 根据角色查询用户列表 |
| getaAdminUserList |  | 返回List<VAdminUser> | 返回所有用户列表 |
| getAdaminUserAmount | wherecondition（查询条件） | 成功返回int，失败返回0 | 返回查询到的信息数量 |
| addAdminUser | Tuser（用户列表实体类） | 成功返回true对象，失败返回false | 添加用户 |
| login | VAdminUser（查询条件） | 成功返回VAdminUser对象，失败则null | 用户登录 |
| delAdminUser | Tuser（用户表实体类） | 成功返回true对象，失败返回false | 删除用户信息 |
| updatePwd | userid,pwd（用户名，密码） | 成功返回true对象，失败返回false | 修改用户密码 |
| changeState | userid(用户名) | 成功返回true对象，失败返回false | 修改用户状态 |
| getuser | userid(用户名) | 成功返回Tuser对象，失败返回null | 根据用户名查询信息 |
| update | Tuser（用户表实体类） | 成功返回true对象，失败返回false | 修改用户信息 |

AdminRoleDAO（后台角色信息管理业务类接口）:

表5.3后台角色信息管理业务类接口表

|  |  |  |  |
| --- | --- | --- | --- |
| 接口名 | 参数 | 返回值 | 描述 |
| getRolelist | Opreation（查询条件） | 成功返回List<TroleB>，失败返回null | 查询用户角色信息 |
| delAdminRole | TroleB（角色实体类） | 成功返回true，失败返回false | 删除角色信息 |
| addAdminRole | TroleB（角色实体类） | 成功返回true，失败返回false | 添加角色信息 |
| edlAdminRole | TroleB（角色实体类） | 成功返回true，失败返回false | 修改角色信息 |

CollegeDAO（后台学院信息管理业务类接口）:

表5.4 后台学院信息管理业务类接口表

|  |  |  |  |
| --- | --- | --- | --- |
| 接口名 | 参数 | 返回值 | 描述 |
| getCollegeList | wherecondition、page、pageSize（查询条件、当前页、每页条数） | 成功返回List<TCollege>对象，失败返回null | 学院信息列表(带分页) |
| getCollegeList | wherecondition（查询条件） | 成功返回int，失败返回0 | 返回查询到的信息数量 |
| getCollegeList |  | 成功返回List<TCollege>对象，失败返回null | 返回所有学院信息列表 |
| addCollege | TCollege（学院实体类） | 成功返回true，失败返回null | 添加学院信息 |
| delCollege | TCollege（学院实体类） | 成功返回true，失败返回null | 删除学院信息 |
| update | TCollege（学院实体类） | 成功返回true，失败返回null | 修改学院信息 |

NoticeDAO（后台公告信息管理业务类接口）:

表5.5后台公告信息管理业务类接口表

|  |  |  |  |
| --- | --- | --- | --- |
| 接口名 | 参数 | 返回值 | 描述 |
| getNoticeList | wherecondition、page、pageSize（查询条件、当前页、每页条数） | 成功返回List< TNotice>对象，失败返回null | 公告信息列表(带分页) |
| getNoticeList | wherecondition（查询条件） | 成功返回int，失败返回0 | 返回查询到的信息数量 |
| addNotice | TNotice（公告实体类） | 成功返回true，失败返回null | 添加公告信息 |
| delNotice | TNotice（公告实体类） | 成功返回true，失败返回null | 删除公告信息 |
| update | TNotice（公告实体类） | 成功返回true，失败返回null | 修改公告信息 |

CurriculumDAO（后台课程信息管理业务类接口）:

表5.6后台课程信息管理业务类接口表

|  |  |  |  |
| --- | --- | --- | --- |
| 接口名 | 参数 | 返回值 | 描述 |
| getVcurrList | wherecondition、page、pageSize（查询条件、当前页、每页条数） | 成功返回List< VCurriculum >对象，失败返回null | 课程信息列表(带分页) |
| getVcurrList | wherecondition（查询条件） | 成功返回int，失败返回0 | 返回查询到的信息数量 |
| getVcurrList |  | 成功返回List< VCurriculum >对象，失败返回null | 返回所有课程信息列表 |
| addTCurr | TCurriculum（课程实体类） | 成功返回true，失败返回null | 添加课程信息 |
| delTCurr | TCurriculum（课程实体类） | 成功返回true，失败返回null | 删除课程信息 |
| update | TCurriculum（课程实体类） | 成功返回true，失败返回null | 修改课程信息 |

CourseDAO（后台任课信息管理业务类接口）:

表5.7 后台任课信息管理业务类接口表

|  |  |  |  |
| --- | --- | --- | --- |
| 接口名 | 参数 | 返回值 | 描述 |
| getCourseList | wherecondition、page、pageSize（查询条件、当前页、每页条数） | 成功返回List< VCourse >对象，失败返回null | 任课信息列表(带分页) |
| getCourseList | wherecondition（查询条件） | 成功返回int，失败返回0 | 返回查询到的信息数量 |
| getCourseList |  | 成功返回List< VCourse >对象，失败返回null | 返回所有任课信息列表 |
| getCourseList | teacherid,classid(教师id,班级id) | 成功返回List< VCourse >对象，失败返回null | 根据条件返回任课信息列表 |
| getVsheetList | stuinfoid,ourseid(教师id,班级id) | 成功返回List< Vsheet >对象，失败返回null | 根据条件返回成绩信息列表 |
| addTCourse | TCourse（任课实体类） | 成功返回true，失败返回null | 添加任课信息 |
| delTCourse | TCourse（任课实体类） | 成功返回true，失败返回null | 删除任课信息 |
| update | TCourse（任课实体类） | 成功返回true，失败返回null | 修改任课信息 |

StudentDAO（后台学生信息管理业务类接口）:

表5.8 后台学生信息管理业务类接口表

|  |  |  |  |
| --- | --- | --- | --- |
| 接口名 | 参数 | 返回值 | 描述 |
| getVstuList | wherecondition、page、pageSize（查询条件、当前页、每页条数） | 成功返回List< Vstudent >对象，失败返回null | 学生信息列表(带分页) |
| getVstuList | wherecondition（查询条件） | 成功返回int，失败返回0 | 返回查询到的信息数量 |
| getVstuList | wherecondition（查询条件） | 成功返回List< Vstudent >，失败返回null | 返回查询到的信息数量 |
| getsheetList | wherecondition、page、pageSize（查询条件、当前页、每页条数） | 成功返回List< Vsheet >对象，失败返回null | 学生成绩信息列表(带分页) |
| getsheetList | wherecondition（查询条件） | 成功返回int，失败返回0 | 返回查询到的信息数量 |
| getVstudentList | classid(班级id) | 成功返回List< Vstudent >对象，失败返回null | 根据班级查询学生信息 |
| addsheet | TSheet（学生成绩实体类） | 成功返回true，失败返回null | 添加学生成绩信息 |
| upsheet | TSheet（学生成绩实体类） | 成功返回true，失败返回null | 修改学生成绩信息 |
| getSheet | id（学生成绩id） | 成功返回TSheet，失败返回null | 获取学生成绩信息 |
| getstu | id（学生id） | 成功返回TStuinfo，失败返回null | 获取学生成绩信息 |
| addStu | TStuinfo（学生实体类） | 成功返回true，失败返回null | 添加学生信息 |
| delStu | TStuinfo（学生实体类） | 成功返回true，失败返回null | 删除学生信息 |
| update | TStuinfo（学生实体类） | 成功返回true，失败返回null | 修改学生信息 |

TeacherDAO（后台教师信息管理业务类接口）:

表5.9 后台教师信息管理业务类接口表

|  |  |  |  |
| --- | --- | --- | --- |
| 接口名 | 参数 | 返回值 | 描述 |
| getteacherList | wherecondition、page、pageSize（查询条件、当前页、每页条数） | 成功返回List< VTeacher >对象，失败返回null | 教师信息列表(带分页) |
| getteacherList | wherecondition（查询条件） | 成功返回int，失败返回0 | 返回查询到的信息数量 |
| getteacherList |  | 成功返回List< VTeacher >对象，失败返回null | 返回所有教师信息列表 |
| getVteacherList | userid（用户名） | 成功返回List< VTeacher >对象，失败返回null | 根据用户名查询教师 |
| addTteacher | Tteacher（教师实体类） | 成功返回true，失败返回null | 添加教师信息 |
| delTteacher | Tteacher（教师实体类） | 成功返回true，失败返回null | 删除教师信息 |
| changeState | Teacherid（教师id） | 成功返回true，失败返回null | 修改教师状态 |
| update | TCurriculum（教师实体类） | 成功返回true，失败返回null | 修改教师信息 |

#### 系统交互动作接口设计

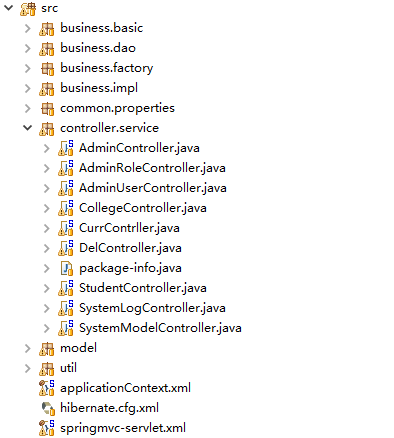
系统交互动作接口设计信息如下所示，在控制类说明、方法名和方法功能中都有详细的文字描述。

图5.4 控制类图

表5.10 控制类表

|  |  |  |  |
| --- | --- | --- | --- |
| 控制类名称 | 控制类说明 | 请求方法名 | 方法功能 |
| AdminControllerr  url=”/ admin2  ” | 管理员用户处理类 | getNoticeList (HttpServletRequest request, intpage,intlimit,tringnoticename,HttpServletResponseresponse, Model model)  url=”/getnotice” | 公告信息分页 |
| addNotice(String noticeName,String noticecontent,String remarks,HttpServletRequest request,HttpServletResponse response, Model model)  url=” deleteAdminRole” | 添加公告信息 |
| getTeacherList (HttpServletRequest request, intpage,intlimit,tringnoticename,HttpServletResponseresponse, Model model) url=”/getnotice” | 教师信息分页 |
| Addteacher(String teaName,String teaphone,String userid,String teasex, HttpServletRequest request,  HttpServletResponse response, Model model)  url=” addteacher” | 添加教师信息 |
| changeUser(String userid, HttpServletRequest request,HttpServletResponse response, Model model)  url=” changeuser” | 修改用户状态 |
| GetLoacdtear(HttpServletRequestrequest,HttpServletResponse response, Model model)  url=” loacdtea” | 加载教师下拉框 |
| CollegeController  url=”/ college  ” | 学院信息处理类 | getCollegeList(HttpServletRequest request, intpage,intlimit,tringnoticename,HttpServletResponseresponse, Model model)  url=”/ getcollege” | 学院信息分页 |
| addCollege(String collegeName,String userid,  HttpServletRequest request,  HttpServletResponse response, Model model)  url=”/ getcollege” | 添加学院信息 |
| upCollege(Integer collegeid,String collegeName,String userid,  HttpServletRequest request,  HttpServletResponse response, Model model)  url=”/ upcollege” | 修改学院信息 |
| getMajorList(HttpServletRequest request, intpage,intlimit,tringnoticename,HttpServletResponseresponse, Model model)  url=”/ getmajor” | 专业信息分页 |
| addMajor(String majorName,String userid, Integer collegeid,  HttpServletRequest request,  HttpServletResponse response, Model model)  url=”/ addmajor” | 添加专业信息 |
| getclass(HttpServletRequest request, intpage,intlimit,tringnoticename,HttpServletResponseresponse, Model model)  url=”/ getclass” | 班级信息分页 |
| addcalss(String calssName,String userid, Integer majorid,  HttpServletRequest request,  HttpServletResponse response, Model model)  url=”/ addclass” | 添加班级信息 |
| GetLoacduser(HttpServletRequestrequest,HttpServletResponse response, Model model)  url=” loacduser” | 加载用户下拉框 |
| CurrContrller  url=”/ curr | 课程信息处理类 | getcurrList(HttpServletRequest request, intpage,intlimit,tringnoticename,HttpServletResponseresponse, Model model)  url=”/ getcurr” | 课程信息分页 |
| addcurr(String curriName,Integer semesterid,String introduce, String remarks, HttpServletRequest request,  HttpServletResponse response, Model model)  url=”/ addcurri” | 添加课程信息 |
| GetLoacdcurr(HttpServletRequestrequest,HttpServletResponse response, Model model)  url=”/ loacdcurr” | 加载课程下拉框 |
| getcourseList(HttpServletRequest request, intpage,intlimit,tringnoticename,HttpServletResponseresponse, Model model)  url=”/ getcourse” | 任课信息分页 |
| getSemesterList(HttpServletRequest request, intpage,intlimit,tringnoticename,HttpServletResponseresponse, Model model)  url=”/ getsemester” | 学期信息分页 |
| addcourseUser(Integer curriculumid,Integer teacherid,Integer classid, Stringremakes,HttpServletRequestrequest,HttpServletResponse response, Model model)  url=”/ addcourse” | 添加任课信息 |
| GetLoacdcurr(HttpServletRequestrequest,HttpServletResponse response, Model model)  url=”/ loacdcurr” | 学期信息下拉框 |
| GetLoacdsemester(HttpServletRequest request,Integer classid,HttpServletResponse response, Model model)  url=”/ loacdseme” | 根据班级获取课程 |

#### 服务器设计与实现

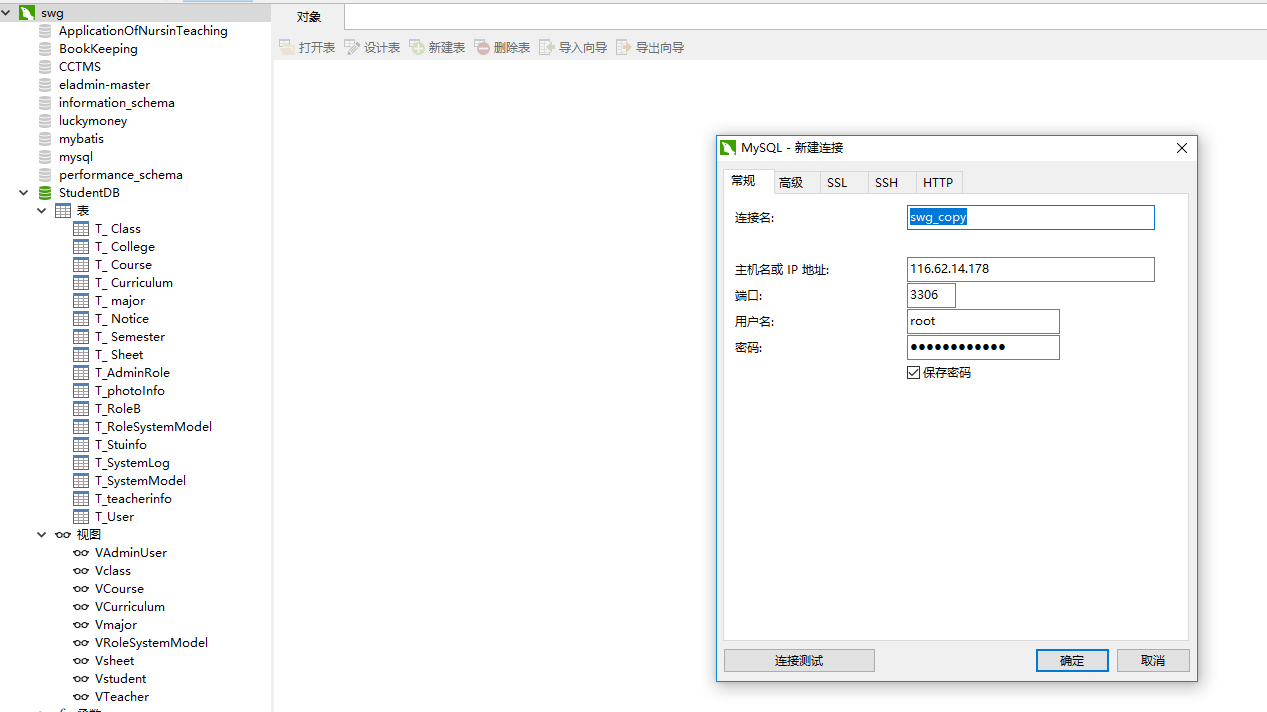
本系统服务器部署于阿里云内，再电脑内下载navicat,再navicat中新建MySql连接,在链接属性中输入连接名，IP地址，端口，用户名，密码。如图所示：

图5.3 数据库图

#### 系统UI设计

用户登录后，首页左边包含用户登录判断权限后所能操作的信息管理，页面内显示日期的备注与记录、最新发布的公告信息与待处理的信息。如下图所示：

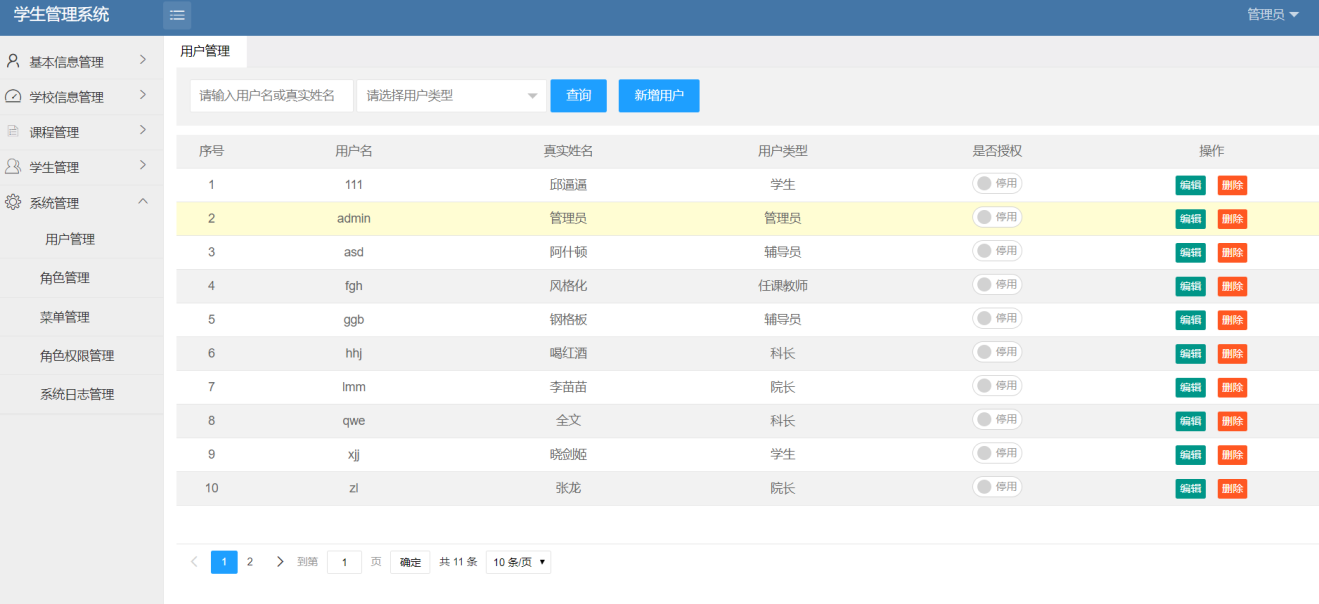
管理员功能模块只允许管理员角色操作，功能包括用户管理、角色管理、菜单管理、角色权限管理、系统日志管理，用户管理进行用户的添加、修改、删除和用户状态的授权，角色管理是对角色进行添加、修改、删除，菜单管理是对菜单进行添加、修改与删除，角色权限管理进行角色对菜单权限的授权与禁用，系统日志管理是查看系统各种日志信息。如下图所示：

图5.4 首页图

图5.5 系统管理图

基本信息管理模块允许管理员、院长、科长角色操作，功能包括公告管理、教师管理、公告管理发布修改公告信息，教师管理进行添加、修改、删除教师用户并且对教师用户进行开启与授权。如下图所示：

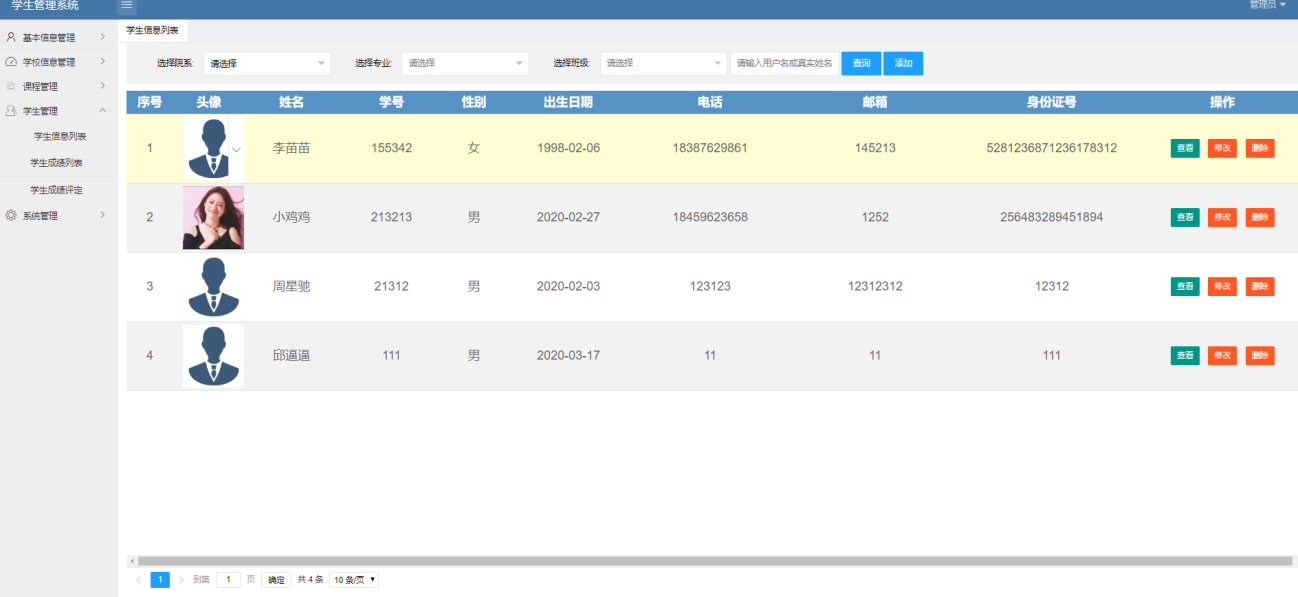
学校管理模块允许管理员、院长、科长角色操作，功能包括学院管理、专业管理、班级管理，学院管理添加学院和删除学院信息，专业管理添加专业和删除专业信息，班级管理添加班级和删除班级信息。如下图所示：

图5.6 基本管理图

课程管理模块允许管理员、院长、科长、辅导员角色操作，功能包括学期管理、课程管理、任课管理，学期管理添加学期和删除学期信息，课程管理添加课程和删除课程信息，任课管理是对授权教师与课程的绑定。如下图所示：

图5.7 课程管理图

图5.7 学校管理图

学生管理模块允许院长、科长、辅导员、任课教师角色操作，功能包括学生信息管理、学生成绩管理、学生成绩评定，学生信息管理是对学生个人信息的汇总与统计，学生成绩管理是对学生成绩信息的汇总与统计，学生成绩评定是任课教师对任课班级的学生进行打分与打评语。如下图所示：

学生登录系统后，可查看系统内管理员发出的公告信息，以及查看和修改自己的个人信息，能够按照学期查看个人的成绩信息。如下图所示：

图5.8 学生信息管理图



图5.9 个人信息管理图

### 第六章、系统的实现与测试



#### 环境搭建

1.下载JDK.网址:<http://www.oracle.com/technetwork/java/javase/downloads/index.html>,

2 安装jdk,

3.安装好jdk后配置环境变量, .安装完成后，右击"我的电脑"，点击"属性"，选择"高级系统设置"；选择高级选项卡，点击环境变量在 "系统变量" 中设置 3 项属性，JAVA\_HOME、PATH、CLASSPATH(大小写无所谓),若已存在则点击"编辑"，不存在则点击"新建"。

如果使用 1.5 以上版本的 JDK，不用设置 CLASSPATH 环境变量，也可以正常编译和运行 Java 程序。

设置3个系统变量，参数如下：

变量名：JAVA\_HOME

变量值：\*:\\*\*\*\*\*\*\*\*\*\* // 根据自己的实际路径配置

变量名：CLASSPATH

变量值：.;%JAVA\_HOME%\lib\dt.jar;%JAVA\_HOME%\lib\tools.jar; //记得前面有个"."

变量名：Path

变量值：%JAVA\_HOME%\bin;%JAVA\_HOME%\jre\bin;

#### 数据层的实现

本课题数据库采用navicat 服务器上的Mysql，数据库名称：StudentDB，链接名：swg，IP：116.62.14.178；

#### 数据连接层的实现

该系统数据连接技术采用Hibernate，hibernate.cfg.xml为数据连接层的配置文件。



图6.1 Hibernate 配置图

#### 基于Hibernate的数据库

本系统采用Hibernate框架技术来连接数据库，示例代码：

#### 数据操作层的实现

图6.2 Hibernate 业务图

数据操作层，采用hibernate框架技术，封装了对数据的增删查改的方法，便于业务逻辑层的调用。

类名：HibBaseDAO

示例代码：

图6.3 Hibernate操作图

#### 数据操作层的调用实现

业务层根据业务模块先定义DAO业务接口，再实现DAO以创建IMPL业务实现类。

类名：AdminuserDaoImpl

示例代码：

#### 实体交换层的实现

图6.4 数据操作图

实体交换层，即模型层，用于存储数据对象，进行数据的存储和传输，使用Hibernate可以直接逆向工程将表生成实体对象，实体对象一般放在model层。

生成的实体层的.xml文件Tuser.hbm.xml实例代码：

图6.5 实体xml图

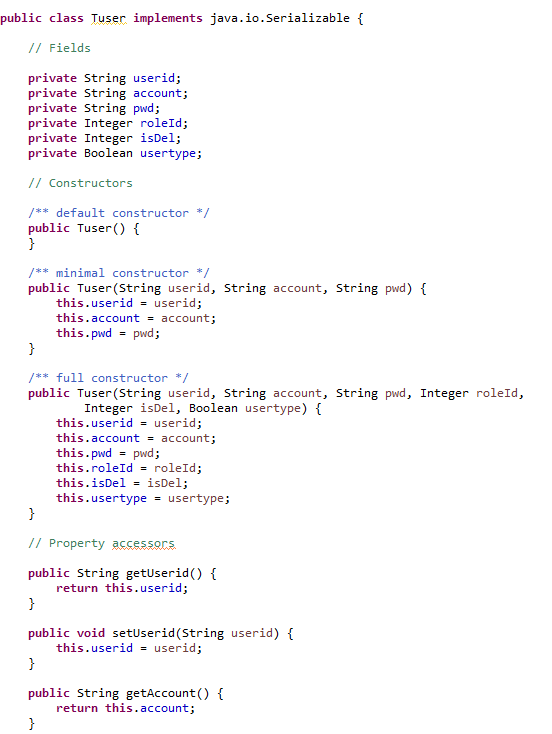
实体层Tuser类实例代码：

图6.6 实体实例图

#### 系统业务层的实现

系统业务层接口调用model层。

类名：AdminUserDAO

示例代码：

#### 信息业务层的调用实现

图6.7 业务接口图

类名：DAOFactory

业务层的的调用代码示例：

图6.8 业务工厂图

#### 控制层的实现

控制层采用springMVC框架技术，利用注解实现URL的定义。

类名：AdminUserController

示例代码：

#### 控制层的调用实现

图6.9 控制类图

用户输入想添加的数据，调用Ajax[[10]](#endnote-10)[8]，代码示例如下：

#### 表示层的实现

图6.10 前端Ajax图

##### 代码

表示层由html页面和JSP页面组成，主要采用LayUI框架和jQuery技术。

代码示例：

##### 实用结果

图6.11 前端html图

#### 软件测试用例说明

图6.12 实用图

##### 测试目的

测试检查系统是否能按照预期的效果运行；

检测系统运行的时候是不是具有良好的可操作性和较美观的界面效果，是不是能最大程度的满足用户的使用要求，通过测试检查和发现系统的不足之处，然后改进完善。

##### 网站端功能测试用例

网站登录功能测试如表：

表6.1 登录功能测试表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 前提条件 | 输入动作 | 期望输出 | 实际显示 | 结果 |
| 输入正确的用户名、密码、验证码 | 输入“admin”，密码“123456” | 跳转网站主界面 | 跳转成功 | 成功 |
| 输入错误的密码 | 输入“admin”，密码“123” | 提示用户名或者密码输入错误 | 提示用户名或密码错误 | 成功 |
| 输入未赋予权限的教师用户 | 输入“lmm”，密码“123456” | 提示用户登录失败 | 提示用户登录失败 | 成功 |

网站权限功能测试如表：

表6.2 权限功能测试表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 前提条件 | 输入动作 | 期望输出 | 实际显示 | 结果 |
| 输入管理员用户 | 输入“admin”，密码“123456” | 跳转网站主界面，所有权限 | 跳转网站主界面，所有权限 | 成功 |
| 输入教师用户 | 输入“lmm”，密码“123456” | 跳转网站主界面，教师权限 | 跳转网站主界面，教师权限 | 成功 |
| 输入学生用户 | 输入“xjj”，密码“123456” | 跳转网站主界面，学生权限 | 跳转网站主界面，学生权限 | 成功 |

任课功能测试如表：

表6.3任课功能测试表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 前提条件 | 输入动作 | 期望输出 | 实际显示 | 结果 |
| 输入正确的班级，教师，课程 | 班级“16级汉语言一班”，教师“lmm” ,课程“数学” | 添加成功 | 任课添加成功 | 成功 |
| 输入重复的班级，教师，课程 | 班级“16级汉语言一班”，教师“lmm” ,课程“数学” | 教师记录已存在 | 教师记录已存在 | 成功 |

修改教师状态功能测试如表：

表6.4 教师状态功能测试表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 前提条件 | 输入动作 | 期望输出 | 实际显示 | 结果 |
| 点击教师启用 | 点击教师启用 | 教师启用成功 | 教师启用成功 | 成功 |

学生成绩评定功能测试如表：

表6.5 成绩评定功能测试表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 前提条件 | 输入动作 | 期望输出 | 实际显示 | 结果 |
| 教师用户登录 | 点击成绩表格输入“80”,评定“优秀” | 数据保存数据库，无输出 | 数据保存数据库，无输出 | 成功 |

角色功能测试如表：

表6.6角色功能测试表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 前提条件 | 输入动作 | 期望输出 | 实际显示 | 结果 |
| 管理员登录进行添加 | 角色名称“测试”，描述“测试” | 角色添加成功 | 角色添加成功 | 成功 |

角色功能测试如表：

表6.7角色功能测试表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 前提条件 | 输入动作 | 期望输出 | 实际显示 | 结果 |
| 管理员登录进行修改 | 查询院长角色，授予“系统管理权限” | 授权成功，请刷新页面 | 授权成功，请刷新页面 | 成功 |
| 管理员登录进行修改 | 查询院长角色，取消授予“系统管理权限” | 取消授权成功，请刷新页面 | 取消授权成功，请刷新页面 | 成功 |

#### 测试报告

经过多次测试，证实本系统已达到预期目标，实现了功能设计目标，但还存在许多缺陷和不足，还需以后进一步完善。

结 论

经过长时间的代码编写与测试，终于完成了本次的毕业设计，从选题、开题，开始设计系统、对系统进行需求分析、概要设计，经历了无数次的报错，修改代码，再试错的过程。直到系统的测试成功，期间做了许多工作，对程序开发积累了很多经验。

在设计过程中深刻体会了系统开发的整个过程。首先前期的需求分析、总体设计非常重要，不要基于编码，前期工作的不完整，准确可能使后期工作变得更复杂。本次毕业设计是对大学四年学习的一次综合能力的锻炼和实践，而且培养了坚强的毅力、耐力，更需要持之以恒，要远超在学生时所需要的。

但由于时间和能力有限，在各个功能模块上并不完美。在后续的开发中应该打各个模块功能的扩展以及系统工作流程的完善作为主要方向。并且这是我工作前一次很好的练习和实践机会，使我培养独立思考问题和自我解决问题的能力，使我意识到必须掌握足够的知识才能在工作中得心应手，适应社会的需要。

致 谢

本文是在向金明指导老师的热情关心和指导下完成的，他渊博的知识和耐心的讲解使我受益匪浅，对顺利完成本课题起到了极大的作用。在此向他表示我最衷心的感谢！

在系统的测试完成过程中，本人在学校与家里得到学校和父母的支持与帮助，让我能够全身心的参与到系统的开发与论文的编写当中，在这里对学校，对父母表示崇高的谢意。

在此，我还要感谢帮助过我写完论文的所有同学，正是由于你们的帮助和支持，我才能克服一个一个的困难和疑感，我的论文才能顺利完成。在这里请大家接受我真诚的谢意！

参考文献

**一、中文部**分

[1] 秦朝明.学生工作管理信息系统在高校学生管理中的应用[J].学理论,2019(08):136-137.

[2]潘蕊.SSH框架的Web网站设计与实现研究[J].成才之路,2019(36):58-59.

[3]曹灿,刘志刚.基于SSH和Layui的工程科学前沿与实践系统[J].工业控制计算机,2019,32(02):91-92+96.

[4] 胡强.MySQL数据库常见问题分析与研究[J].电脑编程技巧与维护,2019(12):91-92.

[5]余炀,曲毅,孙亦乐.基于Apache Tomcat的一站式Java应用服务器解决方案[J].中国金融电脑,2018(01):59-63.

[6]陈华平.民办高职院校学生诚信档案管理信息系统的研建[J].电子世界,2018(24):16-17.

[7]刘巍峰. 基于框架模式的工作流程网站设计与实现[D].吉林大学,2015.

[8]何晶.以SSH框架与jQuery技术为基础的Java-Web开发应用探讨[J].计算机产品与流通,2019(11):104-105.

**二、英文**部分

[1] Privacy Act of 1974; System of Records--Migrant Student Information Exchange[J]. The Federal Register / FIND,2019,84(132).

[2] Hong ZHANG. Research on the Information Management System of University Student Status Archives[P]. Proceedings of the 2019 3rd International Conference on Education, Management Science and Economics (ICEMSE 2019),2019.

1. [↑](#endnote-ref-1)
2. [↑](#endnote-ref-2)
3. [↑](#endnote-ref-3)
4. [↑](#endnote-ref-4)
5. [↑](#endnote-ref-5)
6. [↑](#endnote-ref-6)
7. [↑](#endnote-ref-7)
8. [↑](#endnote-ref-8)
9. [↑](#endnote-ref-9)
10. [↑](#endnote-ref-10)