**广州大学华软软件学院**

**本科毕业论文**

**论文题目** 毕业设计管理系统的设计与实现

**专 业** 计算机系

**班 级** 计算机科学与技术（WEB应用软件开发）

**姓 名**  范韩习

**学 号**  1540406137

**指导教师**  董少英

广州大学华软软件学院计算机系

2019年 4 月

摘要 随着现代信息技术的迅猛发展，各行业依托互联网发展的信息化趋势日益明显。为了提高行政管理工作方面的管理效率与办事效率，各大高校投资引入各类信息化管理系统，如：财政信息管理系统、教务管理系统、自动化办公系统等，通过这些信息化管理系统，不但提高行政效率，而且还简化了办事流程，更是节约了资源。但各方面都在提高自身行政效率的同时，部分高校的毕业设计管理模式仍是采用传统的人工管理模式，而这种管理模式又存在工作效率低、资料文件难管理等诸多缺陷。为解决这个问题，该毕业设计致力于建立一个高效简洁的毕业设计管理系统，以提高相关的行政效率、降低管理者的工作强度与难度。

系统所使用的编程语言为java，整体使用Spring Boot框架，基于MVC应用模式进行开发，数据库使用MySQL；关键技术为Spring、MyBatis，页面使用Thymeleaf取代jsp进行渲染。

关键字 信息管理系统；信息化；SSM框架；Thymeleaf 模板引擎

**ABSTRACT** With the rapid development of information technology in China, the trend of modern information technology relying on the Internet in all walks of life has become increasingly obvious. In order to improve the management efficiency and efficiency of administrative work, universities have invested in the introduction of office automation system, financial information management system, educational management system and other information management systems. Through these information management systems, not only simplifies the work process, improves administrative efficiency, but also saves resources. However, while improving their own administrative efficiency, some universities still adopt the traditional management mode based on manual work, while the traditional management mode has many shortcomings, such as low work efficiency and difficult management of data and documents. Unsolved this problem, the graduation project is committed to establishing an efficient and concise management system for graduation design, in order to improve the relevant administrative efficiency and reduce the work intensity and difficulty of managers.

The programming language used in the system is java, using Spring Boot framework as a whole, developing based on MVC application mode, using MySQL as the database; the key technologies are Spring and MyBatis, and the page is rendered using thymeleaf.

**KEY WORDS** information manage system；informationization；SSM framework；Thymeleaf template engine

目 录

[一、前言 1](#_Toc5990145)

[1.1 研究背景 1](#_Toc5990146)

[1.2 研究的意义与目的 1](#_Toc5990147)

[二、系统需求分析 2](#_Toc5990148)

[2.1 功能需求分析 2](#_Toc5990149)

[2.1.1 角色分析 2](#_Toc5990150)

[2.1.2 用例建模 4](#_Toc5990151)

[2.2 数据需求分析 12](#_Toc5990152)

[三、系统设计 14](#_Toc5990153)

[3.1 系统开发环境与开发语言 14](#_Toc5990154)

[3.2 系统体系结构 14](#_Toc5990155)

[3.3 系统功能设计 14](#_Toc5990156)

[3.3.1 系统子功能设计 15](#_Toc5990157)

[3.3.2 系统总体类图 23](#_Toc5990158)

[3.4 系统界面设计 24](#_Toc5990159)

[3.5 数据设计 27](#_Toc5990160)

[四、系统实现 33](#_Toc5990161)

[4.1 系统功能的实现 33](#_Toc5990162)

[4.1.1 管理员数据导入模板文件下载功能 33](#_Toc5990163)

[4.1.2 教务员课题信息导出功能 36](#_Toc5990164)

[4.1.3 管理员批量导入教师信息功能 40](#_Toc5990165)

[4.1.4 分页功能 42](#_Toc5990166)

[4.2 系统实现效果 47](#_Toc5990167)

[4.2.1 管理员数据模板文件下载 47](#_Toc5990168)

[4.2.2 教务员课题信息导出 48](#_Toc5990169)

[4.2.3 管理员批量导入教师信息 48](#_Toc5990170)

[4.2.4 分页 50](#_Toc5990171)

[五、系统测试 51](#_Toc5990172)

[5.1 测试方法 51](#_Toc5990173)

[5.2 测试用例设计 51](#_Toc5990174)

[5.3 测试结果 52](#_Toc5990175)

[六、总结与展望 53](#_Toc5990176)

[致谢 54](#_Toc5990177)

[参考文献 55](#_Toc5990178)

一、前言

1.1 研究背景

随着现代信息技术的快速发展，依托互联网的各行业信息化趋势日益明显。近年来，各大高校投资引入各类信息化管理系统，如：财政信息管理系统、教务管理系统、自动化办公系统等，通过这些信息化管理系统，不仅简化了办事流程，提高行政效率，更是节约了资源。但各方面都在提高自身行政效率的同时，部分高校的毕业设计管理模式仍是采用基于人工的传统管理模式，而传统的管理模式，又存在工作效率低、难管理等诸多缺陷。

1.2 研究的意义与目的

针对现今毕业设计的人工管理模式存在的诸多缺陷，建立一个毕业设计管理系统，能够提高对学生毕业设计任务流程管理的工作效率，降低管理难度。

二、系统需求分析

2.1 功能需求分析

系统的整体功能设计如图2-1所示：

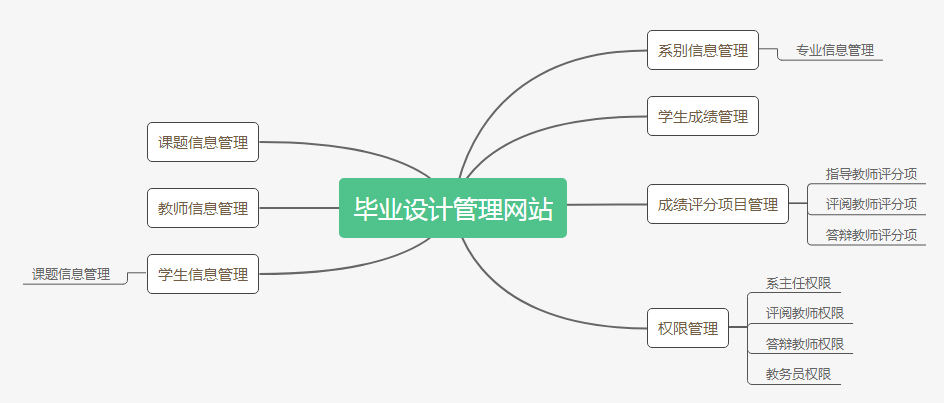


图2-1 系统功能设计图

2.1.1 角色分析

系统的参与者分为教师、系统管理员、教务员、学生四个用户，以及三个辅助角色权限：系主任角色权限、评阅教师角色权限以及答辩组组长角色权限。

1. 系统管理员的功能如图2-2所示：

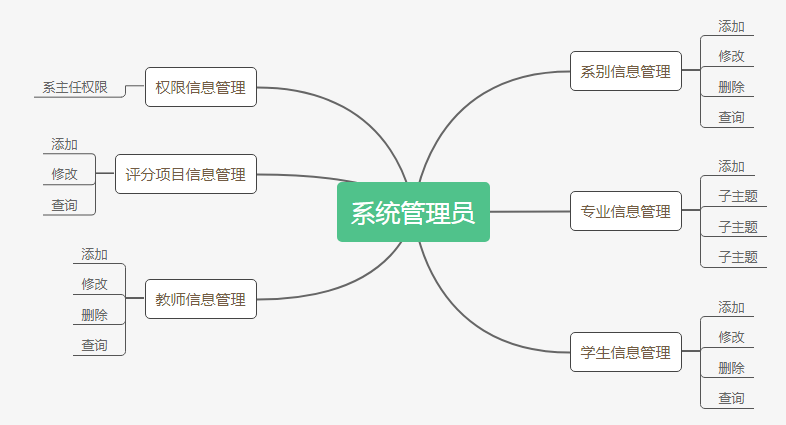


图2-2 系统管理员功能分析

学生和教师信息可进行批量导入，以免大量重复性的操作，得以节省时间。

1. 教务员功能如图2-3：

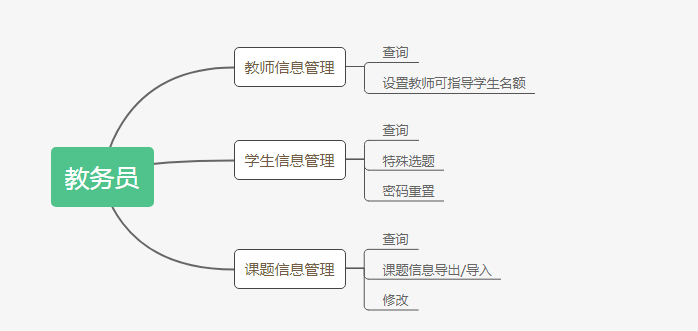


图2-3 教务员功能分析

1. 教师功能如图2-4：

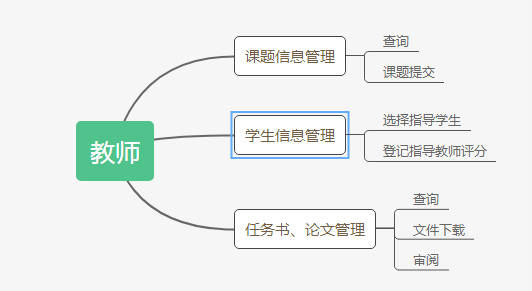


图2-4 教师功能分析

1. 系主任角色权限功能如图2-5

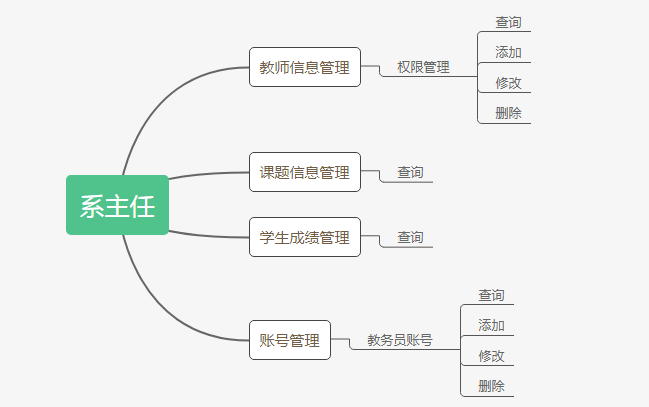


图2-5 系主任角色权限功能分析

1. 评阅教师角色权限功能如图2-6：



图2-6 评阅教师角色权限功能分析

1. 答辩组组长角色权限功能如图2-7

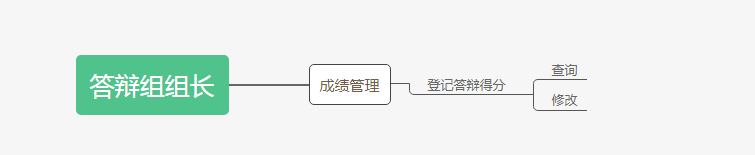


图2-7 答辩组组长角色权限功能分析

2.1.2 用例建模

1. 系统管理员用例图如图2-8：

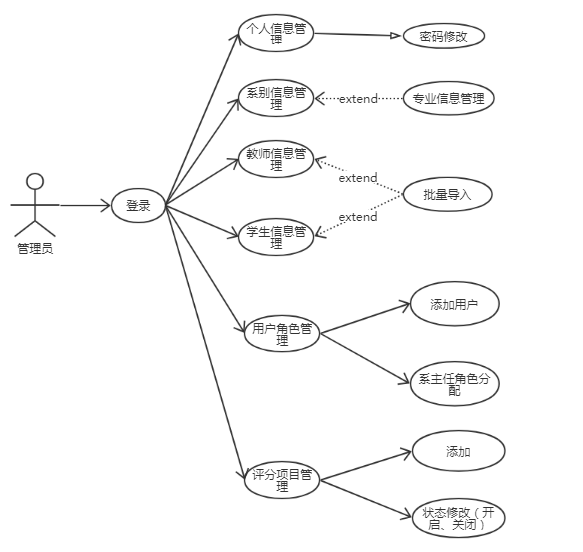


图2-8 管理员用例图

用户添加用例，如表2-1：

表2-1 用户添加-用例表

|  |  |
| --- | --- |
| 用例编号 | AM\_01 |
| 用例名称 | 用户角色管理-添加用户 |
| 功能执行者 | 系统管理员 |
| 功能描述 | 为用户添加角色身份时，需要先将该用户加入用户角色管理列表 |
| 操作前置条件 | 用户为系统管理员用户，并且成功登陆 |
| 基本操作流程 | 1. 点击角色分配按钮，进入用户角色权限管理界面 2. 点击添加按钮，弹出教师用户信息选择界面 3. 选择对应教师信息，点击添加按钮 |
| 结束状况 | 用户角色管理界面显示最新添加的用户信息 |
| 异常流程 | 添加用户的身份非教师或用户已添加入管理列表时，提示错误信息 |
| 说明 | 登录时会因用户所具有的角色不同而显示不同的功能界面 |

1. 学生用例图

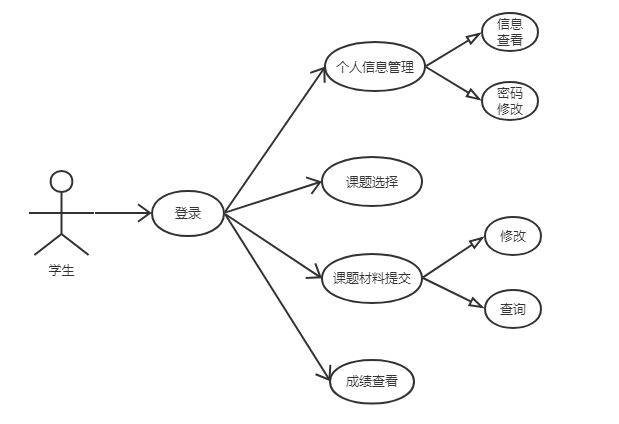


图2-9 学生用例图

课题选择用例，如表2-2：

表2-2 课题选择-用例表

|  |  |
| --- | --- |
| 用例编号 | ST\_02 |
| 用例名称 | 选题 |
| 功能执行者 | 学生 |
| 功能描述 | 浏览提供的课题信息，提交选题请求给相应指导教师 |
| 操作前置条件 | 用户为学生用户，并且成功登陆 |
| 基本操作流程 | 1. 单击选题按钮跳转至选题页面 2. 浏览可供选题的课题信息 3. 点击选择按钮，提交选题请求给对应的教师。 |
| 结束状况 | 提交成功后页面显示提交成功的提示信息信息 |
| 异常流程 | 学生已选课题数量为5时，单击选择按钮，提示可选课题数量已满信息 |
| 说明 | 无 |

1. 教师用例图

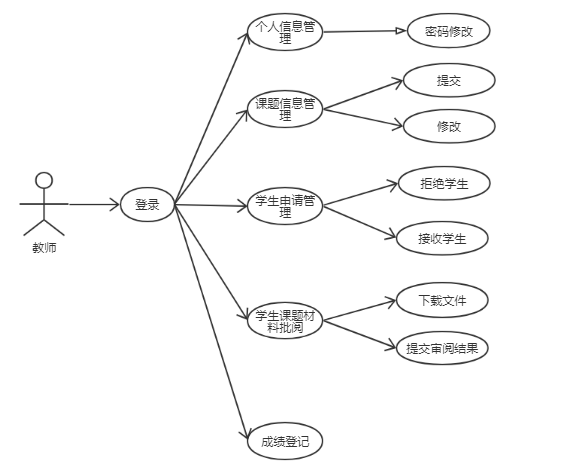


图2-10 教师用例图

课题提交用例，如表2-3：

表2-3 课题提交-用例表

|  |  |
| --- | --- |
| 用例编号 | TE\_01 |
| 用例名称 | 提交课题 |
| 功能执行者 | 教师 |
| 功能描述 | 提交课题 |
| 操作前置条件 | 用户为教师用户，并且成功登陆 |
| 基本操作流程 | 1. 点击课题管理按钮，进入课题管理界面 2. 点击提交课题按钮，弹出课题信息填写框 3. 确认好填写的信息后，点击确定按钮，提交信息 |
| 结束状况 | 提交成功后，显示最新提交的课题信息及审核状态 |
| 异常流程 | 当课题信息未填写完整时，提示错误信息 |
| 说明 | 提交完课题后，初始状态为未审核，审核通过后，学生才能选择该课题 |

成绩登记用例表，如表2-4：

表2-4 成绩登记-用例表

|  |  |
| --- | --- |
| 用例编号 | TE\_02 |
| 用例名称 | 成绩登记 |
| 功能执行者 | 教师 |
| 功能描述 | 登记学生成绩 |
| 操作前置条件 | 用户为教师用户，并且成功登陆 |
| 基本操作流程 | 1. 点击成绩登记按钮，进入学生选择界面 2. 选择要登记成绩的学生，点击对应的成绩登记按钮，进入成绩登记界面 3. 确定好填写的成绩信息后，点击提交按钮 |
| 结束状况 | 提示提交成功信息 |
| 异常流程 | 当填写的成绩信息超过合法范围时，或学生相关材料审核未通过时，提示错误信息 |
| 说明 | 教师身份只能填写指导教师评分项，且学生相关材料都审核通过时，才允许进行成绩登记 |

1. 教务员用例图

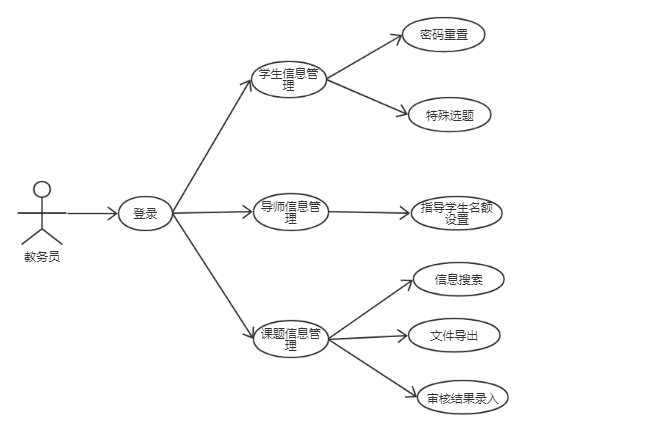


图2-11 教务员用例图

特殊选题用例，如表2-5：

表2-4 特殊选题-用例表

|  |  |
| --- | --- |
| 用例编号 | HE\_01 |
| 用例名称 | 特殊选题 |
| 功能执行者 | 教务员 |
| 功能描述 | 为学生进行特殊选题操作 |
| 操作前置条件 | 用户为教务员用户，并且成功登陆 |
| 基本操作流程 | 1. 点击学生信息管理，进入学生信息管理界面 2. 点击特殊选题按钮，弹出信息填写框 3. 确定后填写信息后，点击提交按钮，关闭 |
| 结束状况 | 学生界面信息刷新，该生选题状态发生改变 |
| 异常流程 | 2.a 在搜索框输入指定信息搜索学生  3.a 选定学生，点击对应的选题修改按钮，弹出信息填写框  继续基本流程步骤3 |
| 说明 | 填写信息不正确或信息不存在时，提示错误信息 |
|  | 无 |

1. 系主任角色用例图

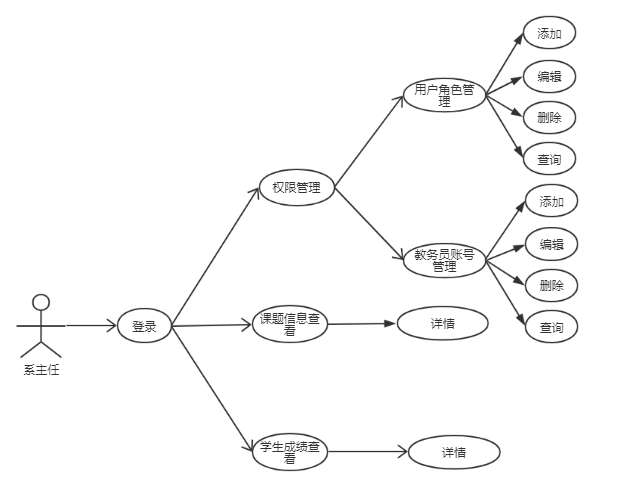


图2-12 系主任角色用例图

教务员账号添加用例，如表2-6：

表2-6 教务员账号添加-用例表

|  |  |
| --- | --- |
| 用例编号 | LD\_01 |
| 用例名称 | 添加教务员账号 |
| 功能执行者 | 系主任角色 |
| 功能描述 | 填写账号信息，添加教务员账号 |
| 操作前置条件 | 用户具有系主任角色权限，并且成功登陆 |
| 基本操作流程 | 1. 点击权限分配按钮，进入管理界面 2. 点击添加教务员按钮，弹出信息填写框 3. 确认填写的信息后，点击确认按钮，提交信息 |
| 结束状况 | 教务员信息列表刷新显示新添加的信息 |
| 异常流程 | 当信息填写不完整时或工号已占用时，提示错误信息 |
| 说明 | 无 |

1. 评阅教师用例图

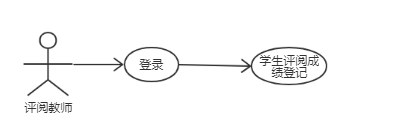


图2-13 评阅教师用例图

学生评阅成绩登记用例，如表2-7：

表2-7 学生评阅成绩登记-用例表

|  |  |
| --- | --- |
| 用例编号 | RT\_01 |
| 用例名称 | 学生评阅成绩登记 |
| 功能执行者 | 评阅教师角色 |
| 功能描述 | 登记学生评阅成绩 |
| 操作前置条件 | 用户具有评阅教师角色权限，并且成功登陆 |
| 基本操作流程 | 1. 点击评阅教师功能下的成绩登记按钮，进入初始界面 2. 输入要登记成绩的学生学号，点击提交，接入成绩登记界面 3. 填写完成绩信息，确认无误后，点击提交 |
| 结束状况 | 提示提交成功信息 |
| 异常流程 | 当填写的成绩信息超过合法范围时，或学生相关材料审核未通过时，提示错误信息 |
| 说明 | 评阅教师角色只能填写评阅教师评分项，且学生相关材料都审核通过时，才允许进行成绩登记 |

1. 答辩组组长用例图

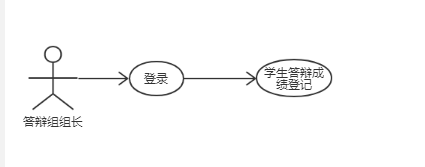


图2-14 答辩组组长用例图

学生答辩成绩登记用例，如表2-8：

表2-8 学生答辩成绩登记-用例表

|  |  |
| --- | --- |
| 用例编号 | RTL\_01 |
| 用例名称 | 学生答辩成绩登记 |
| 功能执行者 | 答辩组组长角色 |
| 功能描述 | 登记学生答辩成绩 |
| 操作前置条件 | 用户具有答辩组组长角色权限，并且成功登陆 |
| 基本操作流程 | 1. 点击答辩组组长功能下的成绩登记按钮，进入初始界面 2. 输入要登记成绩的学生学号，点击提交，接入成绩登记界面 3. 填写完成绩信息，确认无误后，点击提交 |
| 结束状况 | 提示提交成功信息 |
| 异常流程 | 当填写的成绩信息超过合法范围时，或学生相关材料审核未通过时，提示错误信息 |
| 说明 | 答辩组组长角色只能填写答辩评分项，且学生相关材料都审核通过时，才允许进行成绩登记 |

2.2 数据需求分析

系统使用的数据库为MySQL数据库，数据实体设计以功能需求为主，主要有用户数据、课题数据、权限数据、成绩评分项数据、学生成绩数据、提交材料数据、系别数据、专业数据共八大数据实体，各个实体无强外键关联，依靠代码逻辑进行相互关联，具体E-R图如2-15图、2-16图所示：

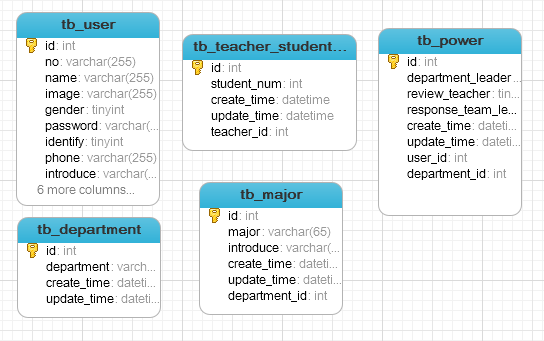


图2-15 数据库信息基本表

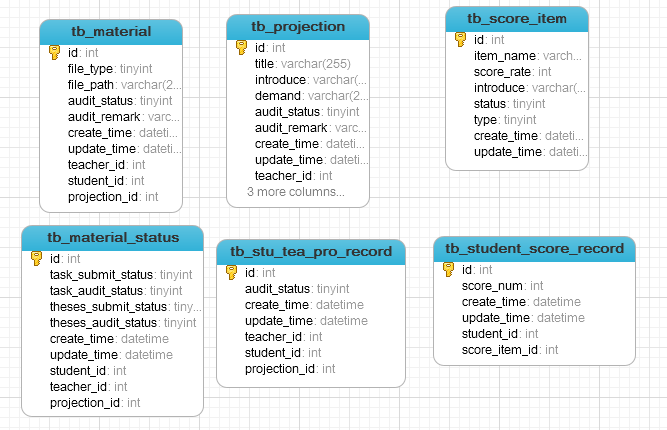


图2-16 数据库信息关联表

三、系统设计

3.1 系统开发环境与开发语言

该系统的的开发环境为Windows，所使用的开发工具为IntelliJ IDEA，IDEA所提倡的是智能编码，可以减少程序员的非代码编写方面的繁杂工作，使用的开发语言为Java，版本要求为java8及以上，使用的服务器为Tomcat，要求Tomcat8及以上。

* 1. 系统体系结构

该系统的体系结构是基于B/S体系机构，即浏览器-服务器结构，进行开发。这是基于C/S结构的所演化出来的网络软件系统结构，但随着科技的迅猛发展，凭借其自身的简易性及优越性，已逐渐取代C/S结构，成为最流行的网络软件系统结构。其体系结构图如图3-1：

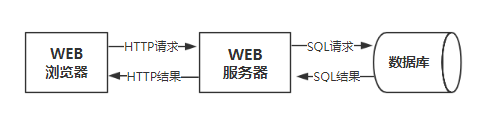


图3-1 B/S典型三层体系结构图

* 1. 系统功能设计

系统功能的结构示意，如3-2图:

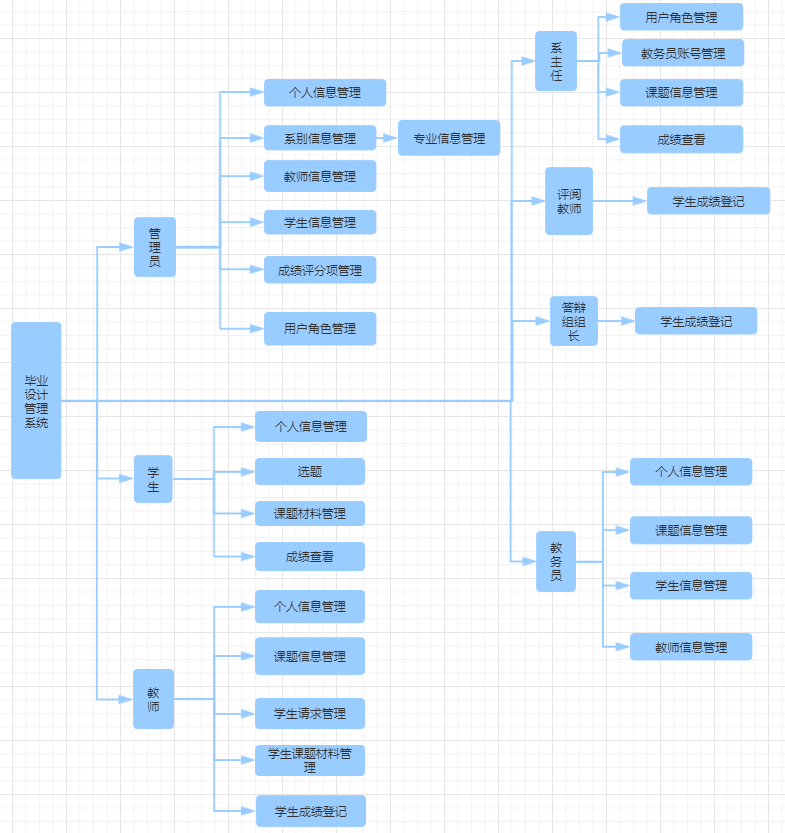


图3-2 系统整体功能结构图

* + 1. 系统子功能设计

1. 个人信息管理功能

用户个人信息管理的顺序图，如图3-3：

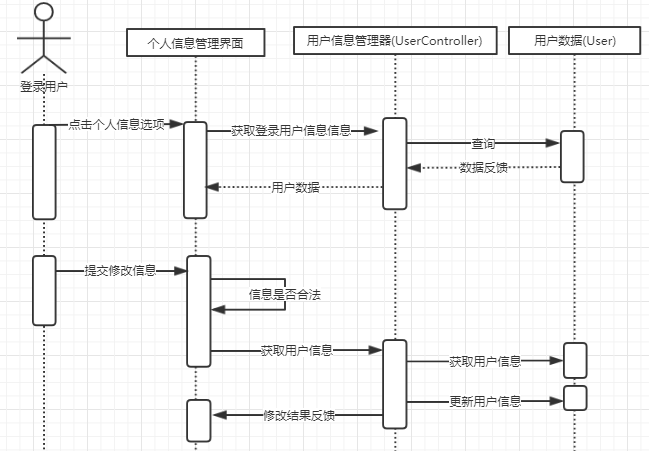


图3-3 个人信息管理-顺序图

1. 系别信息管理功能

可对系别信息进行增添、查询、删减以及修改操作，系别管理的添加功能顺序图如图3-4所示：

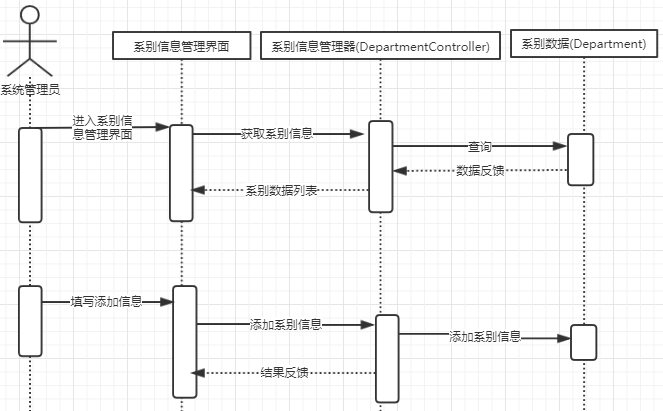


图3-4 系别信息管理-添加功能顺序图

1. 专业信息管理功能

可对专业信息进行增添、查询、删减以及修改操作，专业管理的修改功能顺序图如3-5图：

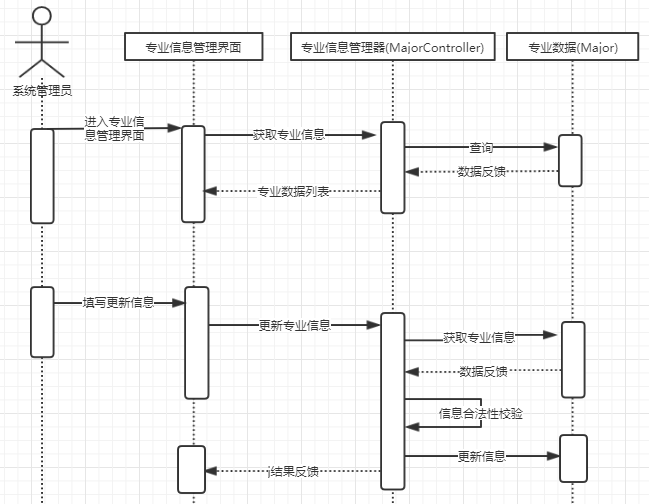


图3-5 专业信息管理-修改功能顺序图

1. 教师信息管理功能

管理员可以对教师信息进行增添、查询、删减以及修改操作，教务员可以对教师的指导学生名额进行设置。

以教务员修改教师指导学生名额为例，其顺序图如图3-6：

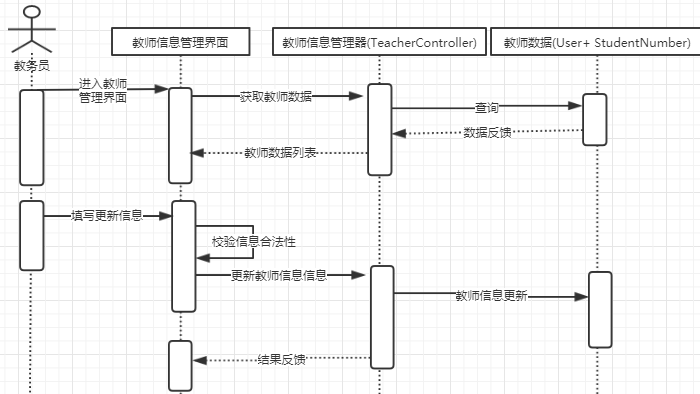


图3-6 教师信息管理-指导学生名额修改顺序图

1. 学生信息管理功能

管理员可对学生信息进行增添、查询、删减以及修改操作，教务员可查看学生选题情况，进行特殊选题及密码重置操作。

学生信息管理的主要顺序图如3-7图：

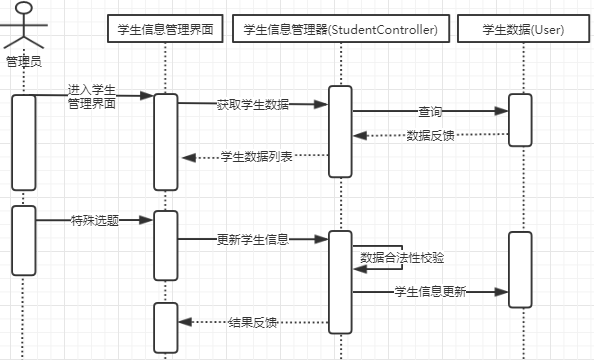


图3-7 学生信息管理-特殊选题顺序图

1. 成绩评分项管理功能

管理员可对成绩评分项目进行增添、状态修改、编辑、查询操作。其主要顺序图如3-8图：

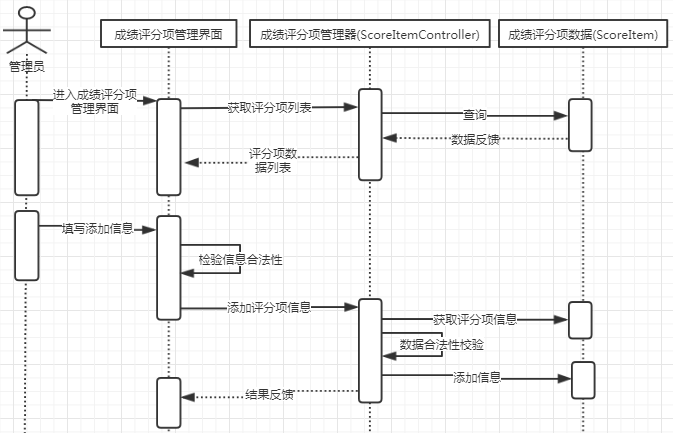


图3-8 成绩评分项管理-添加顺序图

1. 用户角色管理功能

管理员可对用户角色信息进行用户添加系主任角色分配操作，系主任角色可进行用户添加评阅教师角色及答辩组组长角色分配操作。

用户角色管理的主要顺序图如图3-9：

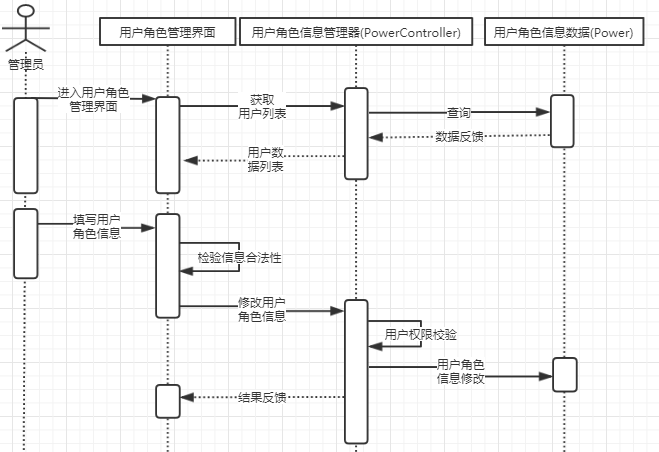


图3-9 用户角色管理-管理员分配角色顺序图

1. 课题材料管理功能

学生可进行课题材料提交操作及历史文件下载操作，教师可进行课题材料下载及审核操作。课题材料管理的主要顺序图如3-10图：

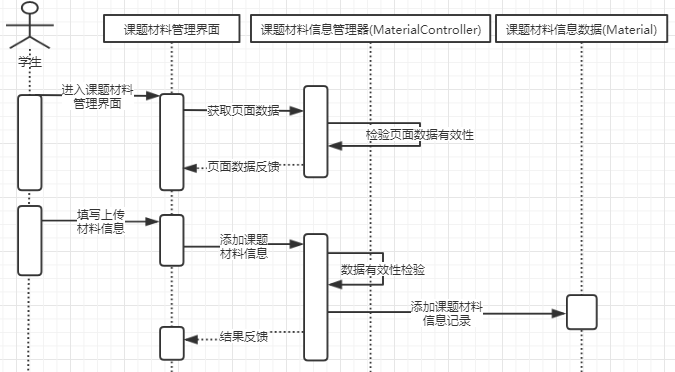


图3-10 课题材料管理-上传材料顺序图

1. 学生选题功能

学生选题功能的顺序图如3-11图：

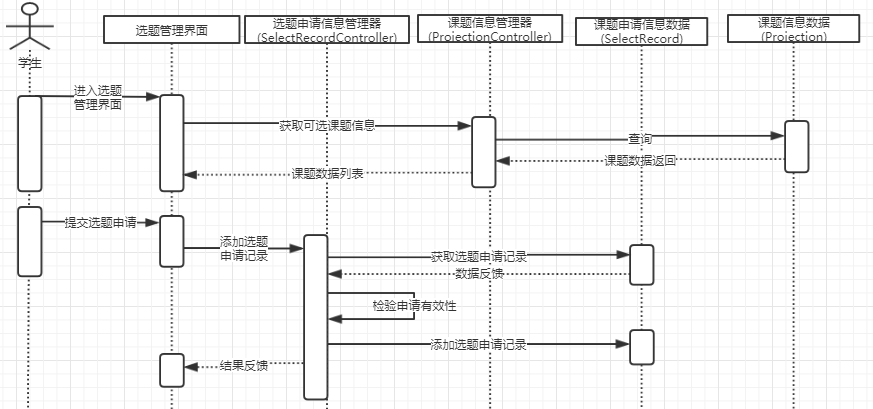


图3-11 学生选题功能顺序图

1. 学生选题请求管理功能

学生选题请求管理顺序图如3-12图：

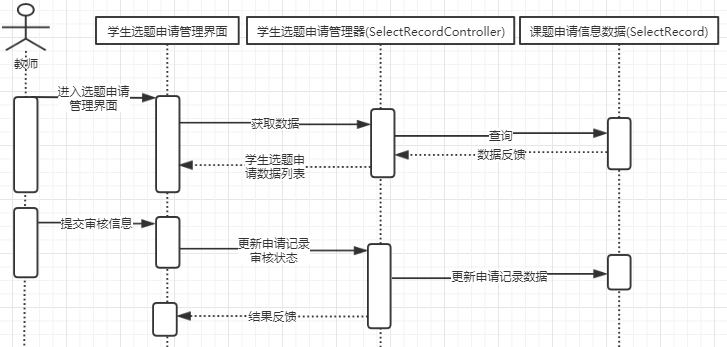


图3-12 学生选题请求管理-顺序图

1. 课题信息管理功能

教师用户进行课题的提交以及修改操作，教务员可以进行查询、信息导出以及课题审核结果的导入，系主任可以查看系内已审核通过的课题信息。

课题信息管理的主要顺序图如3-13图：

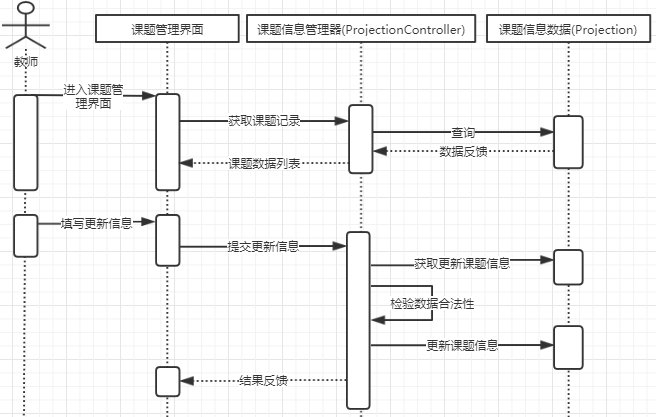


图3-13 课题信息管理-课题修改顺序图

1. 教务员账号管理功能

系主任角色可对系内教务员账号进行管理，可进行增删改查操作。

教务员账号管理的顺序图如3-14图：

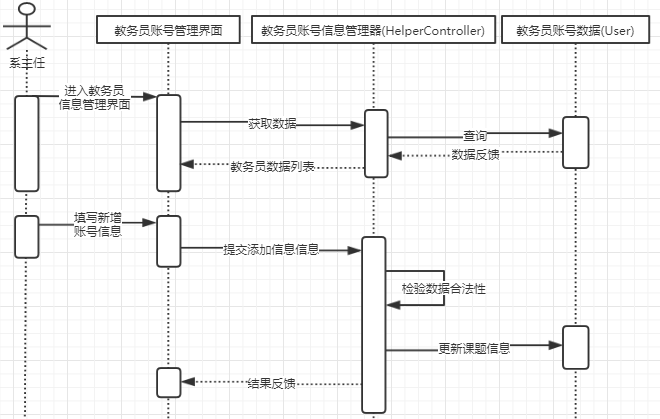


图3-14 教务员账号管理-添加顺序图

1. 学生成绩登记功能

教师用户可登记学生指导教师评分项成绩，评阅教师角色可登记学生评阅教师评分项成绩，答辩组组长角色可登记学生答辩成绩。

学生成绩登记功能的流程如3-15图：

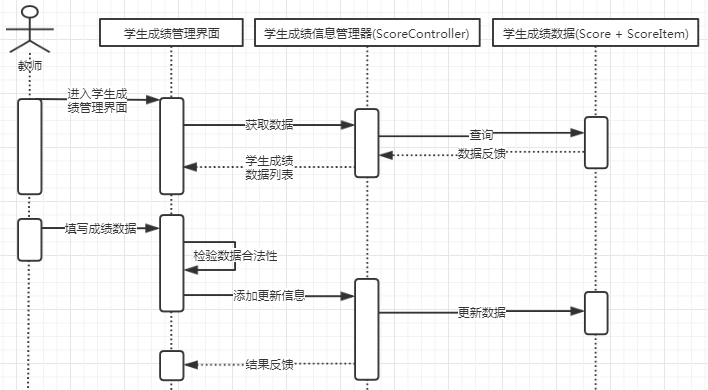


图3-15 学生成绩登记顺序图

* + 1. 系统总体类图

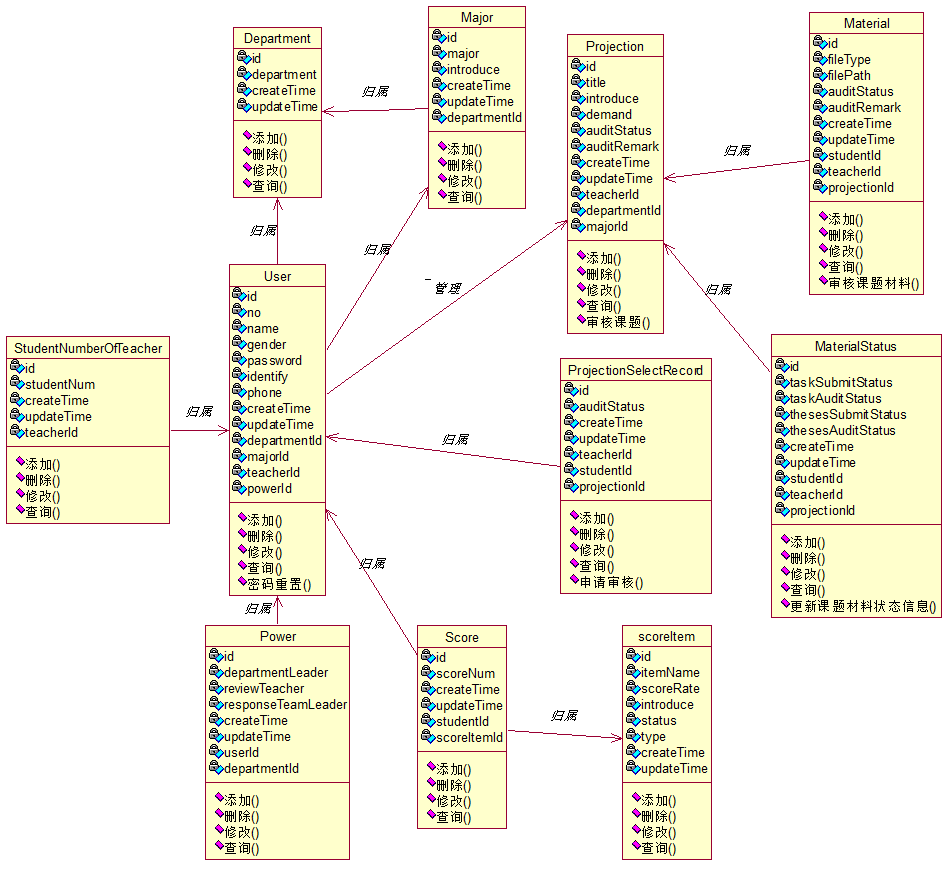


图3-16 系统类图

3.4 系统界面设计

系统用户界面主要是采用传统的HTML+CSS+JQuery进行设计编写，有系统管理员、教务员、学生和教师四类用户界面，各个用户的登录界面如下:

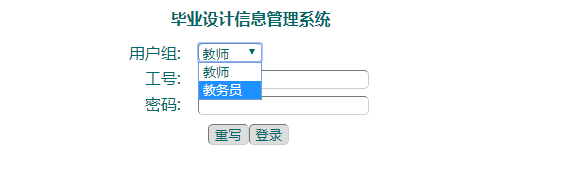


图3-17 教务员及教师登录界面



图3-18 系统管理员登录界面



图3-19 学生登录界面

管理员界面的主要功能有成绩评分项管理、权限管理和学生及教师信息管理，其界面如3-20图：



图3-20 管理员界面-评分项管理

教师界面会根据教师的角色权限不同而显示出不同的功能界面，如图3-21、图3-22：



图3-21 教师界面（无角色权限）

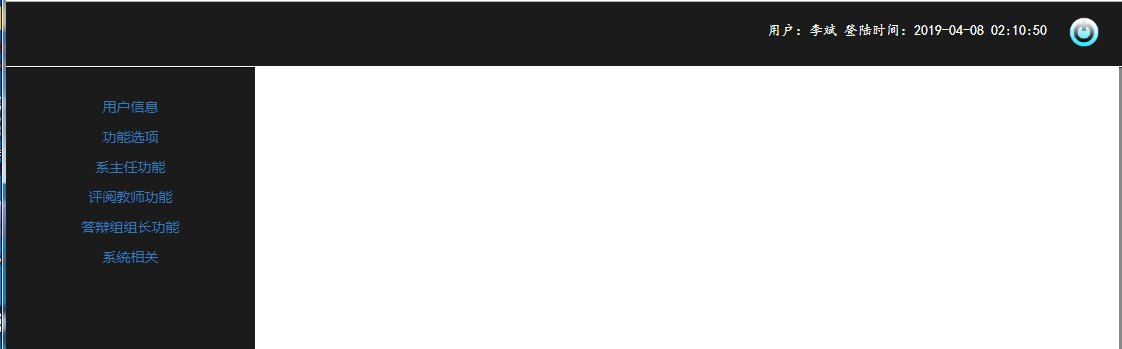


图3-22 教师界面（系主任、批阅教师以及答辩组组长角色权限）

学生界面主要是进行与课题相关的操作，如选题、任务书与论文的提交以及成绩查询。其界面如3-23图：



图3-23 学生界面-选题

教务员界面主要是进行毕业设计流程的相关协调工作，如课题相关信息的导入导出，学生课题的特殊安排以及导师指导名额的设置。其界面如3-24图：



图3-24 教务员界面-课题信息管理

3.5 数据设计

1、tb\_department系别表。如表3-1：

表3-1 系别表（tb\_department）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 数据字段 | 数据类型 | 是否为空 | 注释 |
| 1 | id | int | 否 | 主键 |
| 2 | department | varchar | 否 | 系名 |
| 3 | create\_time | varchar | 是 | 创建时间 |
| 4 | update\_time | tinyint | 是 | 更新时间 |

2、tb\_major专业表。如表3-2：

表3-2 专业表（tb\_major）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 数据字段 | 数据类型 | 是否为空 | 注释 |
| 1 | id | int | 否 | 主键 |
| 2 | major | varchar | 否 | 专业名称 |
| 3 | introduce | varchar | 是 | 简介 |
| 4 | create\_time | varchar | 是 | 创建时间 |
| 5 | update\_time | tinyint | 是 | 更新时间 |
| 6 | department\_id | int | 否 | 系别id |

3、tb\_user用户表。主要用于储存用户信息：ID、学工号、性别、密码、用户类型（1教师 2学生 3教务员 4管理员）以及联系号码。如表3-3：

表3-3 用户表（tb\_user）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 数据字段 | 数据类型 | 是否为空 | 注释 |
| 1 | id | int | 否 | 主键 |
| 2 | no | varchar | 否 | 学工号 |
| 3 | name | varchar | 否 | 姓名 |
| 4 | gender | tinyint | 否 | 性别 |
| 5 | password | varchar | 否 | 密码 |
| 6 | identify | tinyint | 否 | 用户类型 |
| 7 | phone | varchar | 是 | 联系电话 |
| 8 | create\_time | datetime | 是 | 创建时间 |
| 9 | update\_time | datetime | 是 | 更新时间 |
| 10 | department\_id | int | 是 | 系别id |
| 11 | major\_id | int | 是 | 专业id |
| 12 | teacher\_id | int | 是 | 指导教师id |
| 13 | power\_id | int | 是 | 权限记录id |

4、tb\_projection课题信息表。主要用户存储课题信息，如标题、简介、要求。如表3-4：

表3-4 课题信息表（tb\_projection）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 数据字段 | 数据类型 | 是否为空 | 注释 |
| 1 | id | int | 否 | 主键 |
| 2 | title | varchar | 否 | 标题 |
| 3 | introduce | varchar | 否 | 简介 |
| 4 | demand | varchar | 否 | 要求 |
| 5 | audit\_status | tinyint | 否 | 审核状态 |
| 6 | audit\_remark | varchar | 是 | 审核反馈 |
| 7 | create\_time | datetime | 是 | 创建时间 |
| 8 | update\_time | datetime | 是 | 更新时间 |
| 9 | teacher\_id | int | 否 | 指导教师id |

续 表3-4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 数据字段 | 数据类型 | 是否为空 | 注释 |
| 10 | major\_id | int | 否 | 专业id |
| 11 | department\_id | int | 否 | 系别id |

5、tb\_material课题材料表。主要储存学生提交的课题材料信息，如：文件存储地址，审核状态，文件类型（1任务书 2论文）。如表3-5：

表3-5 课题材料表（tb\_material）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 数据字段 | 数据类型 | 是否为空 | 注释 |
| 1 | id | int | 否 | 主键 |
| 2 | file\_type | varchar | 否 | 文件类型 |
| 3 | file\_path | varchar | 否 | 文件地址 |
| 4 | audit\_status | varchar | 否 | 审核状态 |
| 5 | audit\_remark | tinyint | 否 | 审核反馈 |
| 6 | create\_time | datetime | 是 | 创建时间 |
| 7 | update\_time | datetime | 是 | 更新时间 |
| 8 | teacher\_id | int | 否 | 指导教师id |
| 9 | student\_id | int | 否 | 学生id |
| 10 | projection\_id | int | 否 | 课题id |

6、tb\_power用户角色信息表。主要储存用户所具有的角色信息（0不具有，1具有）。如表3-6：

表3-6 用户角色信息表（tb\_power）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 数据字段 | 数据类型 | 是否为空 | 注释 |
| 1 | Id | int | 否 | 主键 |
| 2 | department\_leader | tinyint | 否 | 系主任 |
| 3 | review\_teacher | tinyint | 否 | 批阅教师 |
| 4 | response\_team\_leader | tinyint | 否 | 答辩组长 |
| 5 | create\_time | datetime | 是 | 创建时间 |

续 表3-6

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 数据字段 | 数据类型 | 是否为空 | 注释 |
| 6 | update\_time | datetime | 是 | 更新时间 |
| 7 | user\_id | int | 否 | 用户id |
| 8 | department\_id | int | 否 | 系别id |

7、tb\_score\_item评分项信息表，主要储存评分项信息，如：名称、介绍、总分占比、状态（0未启用，1启用）以及评分类型（1指导教师，2批阅教师，3答辩教师）。如表3-7：

表3-7 评分项信息表（tb\_score\_item）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 数据字段 | 数据类型 | 是否为空 | 注释 |
| 1 | id | int | 否 | 主键 |
| 2 | item\_name | tinyint | 否 | 名称 |
| 3 | score\_rate | tinyint | 否 | 总分占比 |
| 4 | introduce | tinyint | 否 | 介绍 |
| 5 | status | tinyint | 否 | 状态 |
| 6 | type | tinyint | 否 | 评分类型 |
| 7 | create\_time | datetime | 是 | 创建时间 |
| 8 | update\_time | datetime | 是 | 更新时间 |

8、tb\_material\_status学生课题材料状态信息表。主要用于储存学生相关课题材料的状态信息如：提交状态（0未提交，1已提交）、审阅状态（0未审阅，1未通过，2未通过）。如表3-8：

表3-8 学生课题材料状态信息表（tb\_material\_status）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 数据字段 | 数据类型 | 是否为空 | 注释 |
| 1 | id | int | 否 | 主键 |
| 2 | task\_submit\_status | tinyint | 是 | 任务书提交状态 |
| 3 | task\_audit\_status | tinyint | 是 | 任务书审阅状态 |
| 4 | theses\_submit\_status | tinyint | 是 | 论文提交状态 |

续 表3-8

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 数据字段 | 数据类型 | 是否为空 | 注释 |
| 5 | theses\_audit\_status | tinyint | 是 | 论文审阅状态 |
| 6 | create\_time | datetime | 是 | 创建时间 |
| 7 | update\_time | datetime | 是 | 更新时间 |
| 8 | student\_id | int | 否 | 学生id |
| 9 | teacher\_id | int | 否 | 教师id |
| 10 | projection\_id | int | 否 | 课题id |

9、tb\_stu\_tea\_pro\_record学生选题申请信息表。主要用于储存学生选题申请信息，如申请状态（0未提交，1已提交）。如表3-9：

表3-9 学生选题申请信息表（tb\_stu\_tea\_pro\_record）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 数据字段 | 数据类型 | 是否为空 | 注释 |
| 1 | id | int | 否 | 主键 |
| 2 | audit\_status | tinyint | 否 | 申请状态 |
| 3 | create\_time | datetime | 是 | 创建时间 |
| 4 | update\_time | datetime | 是 | 更新时间 |
| 5 | student\_id | int | 否 | 学生id |
| 6 | teacher\_id | int | 否 | 教师id |
| 7 | projection\_id | int | 否 | 课题id |

10、tb\_teacher\_student\_num教师指导名额信息表。主要用于储存教师指导名额信息。如表3-10：

表3-10 教师指导名额信息表（tb\_teacher\_student\_num）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 数据字段 | 数据类型 | 是否为空 | 注释 |
| 1 | id | int | 否 | 主键 |
| 2 | student\_num | tinyint | 否 | 指导名额 |
| 3 | create\_time | datetime | 是 | 创建时间 |
| 4 | update\_time | datetime | 是 | 更新时间 |
| 5 | teacher\_id | int | 否 | 教师id |

11、tb\_student\_score\_record学生成绩信息表。主要用于储存学生成绩信息信息。如表3-11：

表3-11 学生成绩信息表（tb\_student\_score\_record）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 数据字段 | 数据类型 | 是否为空 | 注释 |
| 1 | id | int | 否 | 主键 |
| 2 | score\_num | tinyint | 否 | 评分 |
| 3 | create\_time | datetime | 是 | 创建时间 |
| 4 | update\_time | datetime | 是 | 更新时间 |
| 5 | student\_id | int | 否 | 学生id |
| 6 | score\_item\_id | int | 否 | 评分项目id |

四、系统实现

4.1 系统功能的实现

系统功能的相关实现，以下几个关键功能点作为范例说明：

4.1.1 系统管理员数据导入模板文件下载功能

系统管理员登录成功并跳转至功能界面后，选择学生或者教师信息管理选项，进入信息管理界面后，点击批量导入按钮，选择弹出框下方的模板导出即可下载信息导入模板文件。关键代码如下：

@Controller  
@RequestMapping(**"/admin:file"**)  
**public class** ExcelOfStudentController {  
 @Autowired  
 **private** StudentService **studentService**;  
 @RequestMapping(value = **"/getExcelOfStudent"**, method = RequestMethod.***GET***)  
 **void** record(HttpServletResponse response) **throws** UnsupportedEncodingException {  
 ApiResult apiResult = **new** ApiResult();  
 *// 文件名* String filename = **"学生信息导入模板.xls"**;  
 *// 写到服务器上* String path = FileUtil.*getBaseFileDir*() + filename;  
 File file = **new** File(path);  
 **try** {  
 **if** (!file.exists()) {  
 **if** (!file.getParentFile().exists()) {  
 file.getParentFile().mkdirs();  
 }  
 file.createNewFile();  
 *// 创建写工作簿对象* WritableWorkbook workbook = Workbook.*createWorkbook*(file);  
 *// 工作表* WritableSheet sheet = workbook.createSheet(**"学生信息"**, 0);  
 *// 设置字体;* WritableFont font = **new** WritableFont(WritableFont.***ARIAL***, 12, WritableFont.***BOLD***, **false**, UnderlineStyle.***NO\_UNDERLINE***, Colour.***BLACK***);  
 WritableCellFormat cellFormat = **new** WritableCellFormat(font);  
  
 *// 设置背景颜色;* cellFormat.setBackground(Colour.***WHITE***);  
 *// 设置边框;* cellFormat.setBorder(Border.***ALL***, BorderLineStyle.***DASH\_DOT***);  
 *// 设置文字居中对齐方式;* cellFormat.setAlignment(Alignment.*CENTRE*);  
 *// 设置垂直居中;* cellFormat.setVerticalAlignment(VerticalAlignment.*CENTRE*);  
 *// 给sheet电子版中所有的列设置默认的列的宽度;* sheet.getSettings().setDefaultColumnWidth(15);  
 *// 设置自动换行;* cellFormat.setWrap(**true**);  
  
 *// 单元格* Label label0 = **new** Label(0, 0, **"学号"**, cellFormat);  
 Label label1 = **new** Label(1, 0, **"姓名"**, cellFormat);  
 Label label2 = **new** Label(2, 0, **"性别"**, cellFormat);  
 Label label3 = **new** Label(3, 0, **"联系电话"**, cellFormat);  
  
 sheet.addCell(label0);  
 sheet.addCell(label1);  
 sheet.addCell(label2);  
 sheet.addCell(label3);  
  
 *// 给第二行设置背景、字体颜色、对齐方式等等;* WritableFont font2 = **new** WritableFont(WritableFont.***ARIAL***, 14, WritableFont.***NO\_BOLD***, **false**, UnderlineStyle.***NO\_UNDERLINE***, Colour.***BLACK***);  
 WritableCellFormat cellFormat2 = **new** WritableCellFormat(font2);  
 *// 设置文字居中对齐方式;* cellFormat2.setAlignment(Alignment.*CENTRE*);  
 *// 设置垂直居中;* cellFormat2.setVerticalAlignment(VerticalAlignment.*CENTRE*);  
 cellFormat2.setBackground(Colour.***WHITE***);  
 cellFormat2.setBorder(Border.***ALL***, BorderLineStyle.***THIN***);  
 cellFormat2.setWrap(**true**);  
  
 *//开始执行写入操作* workbook.write();  
 *//关闭流* workbook.close();  
 }  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 *// 第六步，下载excel* OutputStream out = **null**;  
 response.setHeader(**"content-type"**, **"application/octet-stream"**);  
 response.setContentType(**"application/octet-stream"**);  
 *// 下载文件能正常显示中文* response.setHeader(**"Content-Disposition"**,  
 **"attachment;filename="** + **new** String(FileUtil.*getFileName*(file).getBytes(**"UTF-8"**), **"ISO-8859-1"**));  
 *// 实现文件下载* **byte**[] buffer = **new byte**[1024];  
 FileInputStream fis = **null**;  
 BufferedInputStream bis = **null**;  
 **try** {  
 fis = **new** FileInputStream(file);  
 bis = **new** BufferedInputStream(fis);  
 OutputStream os = response.getOutputStream();  
 **int** i = bis.read(buffer);  
 **while** (i != -1) {  
 os.write(buffer, 0, i);  
 i = bis.read(buffer);  
 }  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 **return**;  
 } **finally** {  
 **if** (bis != **null**) {  
 **try** {  
 bis.close();  
 } **catch** (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 **if** (fis != **null**) {  
 **try** {  
 fis.close();  
 } **catch** (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
 }  
………  
}

4.1.2 教务员课题信息导出功能

教务员登录跳转到功能界面后，选择课题信息管理选项，进入课题信息管理界面后，按需求筛选出信息后点击信息导出按钮，即可导出所筛选的信息。以学生模板为例， 关键代码如下：

Controller层：

@Controller  
@RequestMapping(**"/helper:projection"**)  
**public class** ProjectionOfHelperController {  
 @Autowired  
 **private** ProjectionService **projectionService**;  
 @Autowired  
 **private** PowerService **powerService**;  
 @Autowired  
 **private** HttpSession **session**;  
……  
 */\*\*\*\*\*\*\*\*\*\*\*\* 课题信息导出 \*\*\*\*\*\*\*\*\*\*\*\*\*/* @RequestMapping(value = **"/down"**, method = RequestMethod.***GET***)  
 **void** record(ProjectionModel model, HttpServletResponse response) **throws** UnsupportedEncodingException {  
 List<ProjectionModel> projectionModelList = **projectionService**.findList(model);  
 **if** (projectionModelList.size() == 0){  
 **return**;  
 }  
 *// 文件名* String filename = **"课题信息.xls"**;  
 *// 写到服务器上* String path = FileUtil.*getBaseFileDir*() + filename;  
 File file = **new** File(path);

**try** {  
 **if** (!file.exists()) {  
 **if** (!file.getParentFile().exists()) {  
 file.getParentFile().mkdirs();  
 }  
 file.createNewFile();  
 *// 创建写工作簿对象* WritableWorkbook workbook = Workbook.*createWorkbook*(file);  
 *// 工作表* WritableSheet sheet = workbook.createSheet(**"课题信息"**, 0);  
 sheet.setColumnView(0, 11);  
 sheet.setColumnView(1, 17);  
  
 sheet.setColumnView(2, 40);  
 sheet.setColumnView(3, 50);  
 sheet.setColumnView(4, 50);  
 sheet.setColumnView(5, 12);  
 sheet.setColumnView(6, 55);  
 sheet.setColumnView(7, 45);  
  
 CellView cellView= **new** CellView();  
 cellView.setHidden(**true**);  
 sheet.setColumnView(9, cellView);  
  
 *// 设置字体;* WritableFont font = **new** WritableFont(WritableFont.***ARIAL***, 12, WritableFont.***BOLD***, **false**, UnderlineStyle.***NO\_UNDERLINE***, Colour.***BLACK***);  
 WritableCellFormat cellFormat = **new** WritableCellFormat(font);  
  
 *// 设置背景颜色;* cellFormat.setBackground(Colour.***WHITE***);  
 *//设置边框* cellFormat.setBorder(Border.***ALL***, BorderLineStyle.***DASH\_DOT***);  
 *// 设置文字居中对齐方式;* cellFormat.setAlignment(Alignment.*CENTRE*);  
 *// 设置垂直居中;* cellFormat.setVerticalAlignment(VerticalAlignment.*CENTRE*);  
 *// 设置自动换行;* cellFormat.setWrap(**true**);  
  
 *// 单元格* Label label0 = **new** Label(0, 0, **"指导教师"**, cellFormat);  
 Label label1 = **new** Label(1, 0, **"专业需求"**, cellFormat);  
 Label label2 = **new** Label(2, 0, **"课题标题"**, cellFormat);  
 Label label3 = **new** Label(3, 0, **"课题简介"**, cellFormat);  
 Label label4 = **new** Label(4, 0, **"课题要求"**, cellFormat);  
 Label label5 = **new** Label(5, 0, **"审核状态"**, cellFormat);  
 Label label6 = **new** Label(6, 0, **"审核意见"**, cellFormat);  
 Label tip = **new** Label(7, 0, **"审核状态可选值（0未审核 1通过 2未通过）"**, cellFormat);  
  
 sheet.addCell(label0);  
 sheet.addCell(label1);  
 sheet.addCell(label2);  
 sheet.addCell(label3);  
 sheet.addCell(label4);  
 sheet.addCell(label5);  
 sheet.addCell(label6);  
 sheet.addCell(tip);  
 *// 给第二行设置背景、字体颜色、对齐方式等等;* WritableFont font2 = **new** WritableFont(WritableFont.***ARIAL***, 12, WritableFont.***NO\_BOLD***, **false**, UnderlineStyle.***NO\_UNDERLINE***, Colour.***BLACK***);

WritableCellFormat cellFormat2 = **new** WritableCellFormat(font2);  
 *// 设置文字居中对齐方式;* cellFormat2.setAlignment(Alignment.*CENTRE*);  
 *//设置边框* cellFormat2.setBorder(Border.***ALL***, BorderLineStyle.***DASH\_DOT***);  
 *// 设置垂直居中;* cellFormat2.setVerticalAlignment(VerticalAlignment.*CENTRE*);  
 cellFormat2.setAlignment(Alignment.*CENTRE*);  
 cellFormat2.setBackground(Colour.***WHITE***);  
 cellFormat2.setBorder(Border.***ALL***, BorderLineStyle.***THIN***);  
 cellFormat2.setWrap(**true**);  
  
 **for** (**int** r = 1, lenth = projectionModelList.size(); r <= lenth; r++){  
 Label teacherName = **new** Label(0, r, projectionModelList.get(r-1).getTeacherModel().getName(), cellFormat2);  
 Label major = **new** Label(1, r, projectionModelList.get(r-1).getMajorModel().getMajor(), cellFormat2);  
 Label title = **new** Label(2, r, projectionModelList.get(r-1).getTitle(), cellFormat2);  
 Label introduce = **new** Label(3, r, projectionModelList.get(r-1).getIntroduce(), cellFormat2);  
 Label demand = **new** Label(4, r, projectionModelList.get(r-1).getDemand(), cellFormat2);  
 Label status = **new** Label(5, r, String.*valueOf*(projectionModelList.get(r-1).getAuditStatus()), cellFormat2);  
 Label remark = **new** Label(6, r, projectionModelList.get(r-1).getAuditRemark(), cellFormat2);  
 Label id = **new** Label(9, r, String.*valueOf*(projectionModelList.get(r-1).getId()), cellFormat2);  
  
 sheet.addCell(teacherName);  
 sheet.addCell(major);  
 sheet.addCell(title);  
 sheet.addCell(introduce);  
 sheet.addCell(demand);  
 sheet.addCell(status);  
 sheet.addCell(remark);  
 sheet.addCell(id);  
 }  
  
 *//开始执行写入操作* workbook.write();  
 *//关闭流* workbook.close();  
 }  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 *// 第六步，excel文件输出* OutputStream out = **null**;  
 response.setHeader(**"content-type"**, **"application/octet-stream"**);  
 response.setContentType(**"application/octet-stream"**);  
 *// 下载文件能正常显示中文* response.setHeader(**"Content-Disposition"**,  
 **"attachment;filename="** + **new** String(FileUtil.*getFileName*(file).getBytes(**"UTF-8"**), **"ISO-8859-1"**));  
 *// 实现文件输出* **byte**[] buffer = **new byte**[1024];  
 FileInputStream fis = **null**;  
 BufferedInputStream bis = **null**;  
 **try** {  
 fis = **new** FileInputStream(file);  
 bis = **new** BufferedInputStream(fis);  
 OutputStream os = response.getOutputStream();  
 **int** i = bis.read(buffer);  
 **while** (i != -1) {  
 os.write(buffer, 0, i);  
 i = bis.read(buffer);  
 }  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 **return**;  
 } **finally** {  
 **if** (bis != **null**) {  
 **try** {  
 bis.close();  
 } **catch** (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 **if** (fis != **null**) {  
 **try** {  
 fis.close();  
 } **catch** (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 **if**(file.exists()){  
 file.delete();  
 }  
 }  
 }

……  
}

Service层：

@Service  
**public class** ProjectionServiceImpl **implements** ProjectionService {  
 @Autowired  
 **private** ProjectionRepository **projectionRepository**;  
 @Autowired  
 **private** TeacherService **teacherService**;  
 @Autowired  
 **private** MajorService **majorService**;  
 @Autowired  
 **private** SelectRecordService **selectRecordService**;  
 ……

@Override  
 **public** List<ProjectionModel> findList(ProjectionModel model) {  
 List<ProjectionModel> modelList = **projectionRepository**.findList(model);  
 modelList.stream().forEach(data->{  
 data.setTeacherModel(**teacherService**.findById(data.getTeacherId()));  
 data.setMajorModel(**majorService**.findById(data.getMajorId()));  
 });  
 **return** modelList;  
 }

……

}

4.1.3 系统管理员批量导入教师信息功能

系统管理员成功登录并跳转至功能界面后，选择教师信息管理选项，跳转到教师信息管理界面后，选择批量导入功能，选择目标文件及所指定导入的系别后，点击确定即可导入信息。关键代码如下：

Controller层：

@Controller  
@RequestMapping(**"/admin:file"**)  
**public class** ExcelOfTeacherController {  
 @Autowired  
 **private** TeacherService **teacherService**;  
……  
 @RequestMapping(value = **"/batchImportOfTeacher"**, method = RequestMethod.***POST***)  
 @ResponseBody  
 ApiResult batchImportOfStudent(

@RequestParam(**"departmentId"**) Integer departmentId,

@RequestParam(**"file"**) MultipartFile file)

{  
 ApiResult apiResult = **new** ApiResult();  
 Boolean isExcel = **false**;  
 **if** (file.isEmpty()) {  
 apiResult.setCode(-1);  
 apiResult.setMsg(**"请选择导入文件！"**);  
 **return** apiResult;

} **else if** (!file.getOriginalFilename().endsWith(**"xls"**)

&& !file.getOriginalFilename().endsWith(**"xlsx"**)) {  
 apiResult.setCode(-1);  
 apiResult.setMsg(**"文件格式错误！"**);  
 **return** apiResult;

}**else**{  
 **try** {  
 Workbook workbook = Workbook.*getWorkbook*(file.getInputStream());  
 Sheet sheet = workbook.getSheet(0);  
 System.***out***.println(sheet.getColumns()+**" "**+sheet.getRows());  
 UserModel teacher = **new** UserModel();  
 teacher.setDepartmentId(departmentId);  
 **for** (**int** i = 1, j= sheet.getRows(); i< j; i++){  
 **if** (**""**.equals(sheet.getCell(0,i).getContents())){  
 **break**;  
 }  
 teacher.setNo(sheet.getCell(0,i).getContents());  
 teacher.setName(sheet.getCell(1,i).getContents());  
 **if** (**"男"**.equals(sheet.getCell(2,i).getContents())) {  
 teacher.setGender(1);  
 }**else if** (**"女"**.equals(sheet.getCell(2,i).getContents())){

teacher.setGender(2);

} teacher.setPhone(sheet.getCell(3,i).getContents()); **teacherService**.saveTeacher(teacher);

}

} **catch** (IOException e) {  
 e.printStackTrace();  
} **catch** (BiffException e) {  
 e.printStackTrace();  
}

apiResult.setCode(0);  
apiResult.setMsg(**"数据导入成功！"**);  
**return** apiResult;

}

}

Service层：

@Service

**public class** TeacherServiceImpl **implements** TeacherService {

@Autowired  
 **private** TeacherRepository **teacherRepository**;  
 @Autowired  
 **private** DepartmentService **departmentService**;  
……  
 @Override  
 **public** UserModel saveTeacher(UserModel model) {  
 UserModel existModel = **this**.findByNo(model.getNo());  
 **if** (existModel != **null**){  
 **return null**;  
 }  
 model.setIdentify(1);  
 model = **this**.save(model);  
 model = getMoreInfo(model);  
 **return** model;  
 }  
……  
}

4.1.4 分页功能

实现数据的分页显示，每次查询只查询指定数量的数据而非整体查询出来，可进行上下翻页及首页、尾页跳转功能。关键代码如下：

前后端数据交互对象 ApiPageResult：

@Data  
**public class** ApiPageResult **extends** ApiResult {  
 */\*\*  
 \* 一页显示数量  
 \*/* **private** Integer **size**;  
 */\*\*  
 \* 页数  
 \*/* **private** Integer **page**;  
 */\*\*  
 \* 总页数  
 \*/* **private** Integer **totalPage**;  
 */\*\*  
 \* 上一页  
 \*/* **private** Boolean **pageUp**;  
 */\*\*  
 \* 下一页  
 \*/* **private** Boolean **pageDown**;  
 **public** ApiPageResult(Object data, Integer total, Integer page, Integer size) {  
 **this**.**totalPage** = total / size;  
 **if** (total % size > 0) {  
 **this**.**totalPage**++;  
 }  
 **if** (page != 0 && page <= **this**.**totalPage** - 1) {  
 **this**.**pageUp** = **true**;  
 } **else** {  
 **this**.**pageUp** = **false**;  
 }  
 **if** (page < **this**.**totalPage** - 1 && page >= 0) {  
 **this**.**pageDown** = **true**;  
 } **else** {  
 **this**.**pageDown** = **false**;  
 }  
 **this**.**totalPage** = total / size;  
 **if** (total % size > 0) {  
 **this**.**totalPage**++;  
 }  
 **this**.**page** = page;  
 **this**.**size** = size;  
 **this**.setData(data);  
 }  
}

Controller层：

@Controller  
@RequestMapping(**"/admin:student"**)  
**public class** StudentOfAdminController {  
 @Autowired  
 **private** StudentService **studentService**;  
 @Autowired  
 **protected** PowerService **powerService**;  
**……**

@RequestMapping(value = **"/findStudent"**, method = RequestMethod.***GET***, produces = MediaType.***APPLICATION\_JSON\_UTF8\_VALUE***)  
 @ResponseBody  
 ApiResult findStudent(UserModel model) {  
 ApiResult apiResult = **null**;  
 model.setName((model.getName() != **null**) ? **"%"** + model.getName() + **"%"** : **null**);  
 model.setNo((model.getName() != **null**) ? **"%"** + model.getNo() + **"%"** : **null**);  
 List<UserModel> studentModelList = **studentService**.findStudent(model);  
 Integer total = **studentService**.findTotal(model);  
 apiResult = **new** ApiPageResult(studentModelList, total, model.getPage(), model.getSize());  
 **return** apiResult;  
 }  
……

}

Service层：

@Service  
**public class** StudentServiceImpl **implements** StudentService {  
 @Autowired  
 **private** StudentRepository **studentRepository**;  
……  
 @Override  
 **public** Integer findTotal(UserModel model) {  
 **return studentRepository**.findTotal(model);  
 }  
 @Override  
 **public** List<UserModel> findStudent(UserModel model) {  
 List<UserModel> results = **studentRepository**.findList(model);  
 results.stream().forEach(data ->{  
 data = getMoreInfo(data);  
 });  
 **return** results;  
 }  
……

**private** UserModel getMoreInfo(UserModel model){  
 **if** (model.getTeacherId() != **null**) {  
 model.setTeacherModel(**teacherService**.findById(model.getTeacherId()));  
 }  
 model.setDepartmentModel(**departmentService**.findById(model.getDepartmentId()));  
 model.setMajorModel(**majorService**.findById(model.getMajorId()));  
 SelectRecordModel selectRecordModel = **new** SelectRecordModel();  
 selectRecordModel.setStudentId(model.getId());  
 selectRecordModel.setAuditStatus(1);  
 List<SelectRecordModel> results = **selectRecordService**.findList(selectRecordModel);  
 **if** (results.size() == 1){  
 model.setProjectionSelectModel(results.get(0));  
 model.setProjectionModel(

**projectionService**.findById(results.get(0).getProjectionId())

);  
 }  
 **return** model;  
 }  
}

持久（repository）层：

@Mapper  
@Component  
**public interface** StudentRepository {

……  
 @SelectProvider(type = StudentProvider.**class**, method = **"findListByModel"**)  
 @ResultMap(value = **"studentMap"**)  
 List<UserModel> findList(UserModel model);

@SelectProvider(type = StudentProvider.**class**, method = **"findTotal"**)  
 Integer findTotal(UserModel model);  
 */\*\*\*\*\*\*\*\*\*\* 内部类 \*\*\*\*\*\*\*\*\*/* **class** StudentProvider {  
 **public** String findListByModel(UserModel model) {  
 SQL sql = **new** SQL();  
 sql.SELECT(**"\*"**);  
 sql.FROM(**"tb\_user"**);  
 **if** (model.getNo() != **null** && !**""**.equals(model.getNo())) {  
 sql.WHERE(**"no like #{no}"**);  
 }  
 **if** (model.getName() != **null** && !**""**.equals(model.getName())) {  
 sql.WHERE(**"name like #{name}"**);  
 }  
 **if** (model.getGender() != **null** && !**""**.equals(model.getGender())) {  
 sql.WHERE(**"gender = #{gender}"**);  
 }  
 **if** (model.getPassword() != **null** && !**""**.equals(model.getPassword())) {  
 sql.WHERE(**"password = #{password}"**);  
 }  
 **if** (model.getDepartmentId() != **null** && !**""**.equals(model.getDepartmentId())) {  
 sql.WHERE(**"department\_id = #{departmentId}"**);  
 }  
 **if** (model.getMajorId() != **null** && !**""**.equals(model.getMajorId())) {  
 sql.WHERE(**"major\_id = #{majorId}"**);  
 }  
 **if** (model.getTeacherId() != **null** && !**""**.equals(model.getTeacherId())) {  
 sql.WHERE(**"teacher\_id = #{teacherId}"**);  
 }  
 sql.WHERE(**"identify = 2"**);  
 sql.ORDER\_BY(**"no DESC"**);  
 StringBuffer end = **new** StringBuffer();  
 **if** (model.getPage() != **null** && !**""**.equals(model.getPage())) {  
 **if** (model.getSize() != **null** && !**""**.equals(model.getSize()))  
 end.append(**" LIMIT "** + model.getPage()\*model.getSize()+**","**+model.getSize());  
 }  
 **return** sql.toString()+end.toString();  
 }  
 **public** String findTotal(UserModel model) {  
 SQL sql = **new** SQL();  
 sql.SELECT(**"count(1)"**);  
 sql.FROM(**"tb\_user"**);  
 **if** (model.getNo() != **null** && !**""**.equals(model.getNo())) {  
 sql.WHERE(**"no like #{no}"**);  
 }  
 **if** (model.getName() != **null** && !**""**.equals(model.getName())) {  
 sql.WHERE(**"name like #{name}"**);  
 }  
 **if** (model.getGender() != **null** && !**""**.equals(model.getGender())) {  
 sql.WHERE(**"gender = #{gender}"**);  
 }  
 **if** (model.getPassword() != **null** && !**""**.equals(model.getPassword())) {  
 sql.WHERE(**"password = #{password}"**);  
 }  
 **if** (model.getDepartmentId() != **null** && !**""**.equals(model.getDepartmentId())) {  
 sql.WHERE(**"department\_id = #{departmentId}"**);  
 }  
 **if** (model.getMajorId() != **null** && !**""**.equals(model.getMajorId())) {  
 sql.WHERE(**"major\_id = #{majorId}"**);  
 }  
 **if** (model.getTeacherId() != **null** && !**""**.equals(model.getTeacherId())) {  
 sql.WHERE(**"teacher\_id = #{teacherId}"**);  
 }  
 sql.WHERE(**"identify = 2"**);  
 sql.ORDER\_BY(**"update\_time DESC"**);  
 **return** sql.toString();  
 }  
 **……**  
 }  
}

4.2 系统实现效果

4.2.1 管理员数据模板文件下载

数据模板文件下载效果图如4-1图：

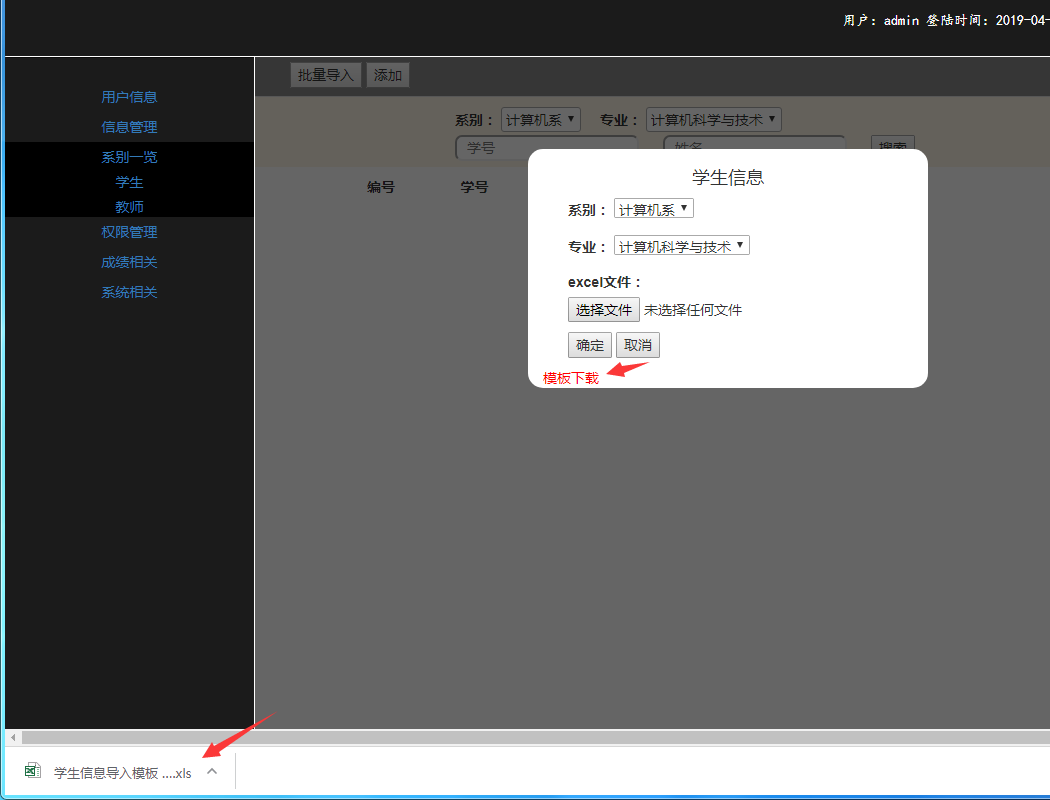


图4-1 数据模板文件下载功能效果图

4.2.2 教务员课题信息导出

教务员课题信息导出，可先对信息进行筛选，根据需要来导出数据，其效果图如4-2图：



图4-2 教务员课题信息导出功能效果图

4.2.3 管理员批量导入教师信息

教师信息批量导入效果图以下图：



图4-3 管理员教师信息批量导入功能效果图1



图4-4 管理员教师信息批量导入功能效果图2



图4-5 管理员教师信息批量导入功能效果图3

4.2.4 分页



图4-6 分页效果图1



图4-7 分页效果图2

五、系统测试

5.1 测试方法

该系统的主要是以黑盒测试为主要测试方法进行系统测试，测试目标是确保系统能够正常并且正确运行。

5.2 测试用例设计

下列用例为例，以此说明测试用例的相关设计：

（1）登录：

表5-1 登陆功能测试用例设计

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例编号 | 001 | | |
| 测试功能的相关描述 | 用户登录功能 | | |
| 测试的目的 | 验证登录功能是否正常运行 | | |
| 测试的前置条件 | 系统可以正常运行 | | |
| 输入的测试数据 | 期望的输出结果 | 实际情况 | 是否通过 |
| 正常值：输入用户学工号及密码 | 登陆成功，跳转至功能界面 | 与期望相符 | 是 |
| 异常值：输入学工号及错误的密码 | 提示错误信息 | 与期望相符 | 是 |
| 异常值：不输入信息 | 提示错误信息 | 与期望相符 | 是 |

（2）教师提交课题信息：

表5-2 教师提交课题信息功能测试

|  |  |
| --- | --- |
| 测试用例编号 | 002 |
| 测试功能的相关描述 | 教师提交课题功能功能 |
| 测试的目的 | 验证课题提交功能是否正常运行 |
| 测试的前置条件 | 系统正常运行且教师用户成功登陆 |

续 表5-2

|  |  |  |  |
| --- | --- | --- | --- |
| 输入的测试数据 | 期望输出 | 实际情况 | 是否通过 |
| 正常值：输入标题、简介、要求 | 提交成功吗，返回已提交未审核的课题列表 | 与期望相符 | 是 |
| 异常值：信息留空 | 提示错误信息 | 与期望相符 | 是 |

（3）教师登记学生成绩功能

表5-3 教师登记学生成绩功能测试

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例编号 | 003 | | |
| 测试功能的相关描述 | 教师用户登记学生成绩 | | |
| 测试的目的 | 验证学生成绩登记功能是否正常运行 | | |
| 测试的前置条件 | 系统正常运行且教师用户成功登陆 | | |
| 输入的测试数据 | 期望输出 | 实际情况 | 是否通过 |
| 正常值：各项评分均在0-100合理范围 | 登记成功，提示提交成功 | 与期望相符 | 是 |
| 异常值：信息留空 | 提示错误信息 | 与期望相符 | 是 |
| 异常值：输入非数字字符 | 提示错误信息 | 与期望相符 | 是 |

5.3 测试结果

在经过相关的系统测试之后，测试结果表明系统可正常并正确运行，测试情况与期望结果相符，无发现明显bug，些许瑕疵经排查后已经修复完毕。

六、总结与展望

这几个月的毕业设计，期间由于现实的些许原因及自身的懒惰，多次拖慢了毕设进程，使得时间从充裕变得紧迫，不禁让人悔不该当初，期间也有因为接口调用路径的问题，而去排查一整天才发现根由的略微有趣的体验，但还好，总算在答辩前完成了所有的准备工作。

回望该系统，虽然基本功能均已实现，能达到项目所需，但是仍有较大需要改进拓展的地方，如：

（1）重构系统架构，采用spring cloud的相关技术，将该系统构建成微服务，提高系统稳定性。

（2）美化界面，提高用户体验感。

（3）简化代码，降低整体代码赘余度，以达到良好代码阅读感。

虽然毕设业已结束，但对该系统我会持之以恒的改进下去。

致谢

这次的毕设已近尾声，在此衷心感谢大学四年间与我共同度过的老师与同学们，谢谢他们对我的帮助与鼓励，也特别要感谢董少英董老师的教导，自外语系转系过来后，在她的课程上我更加深刻的体会到了编程之美与编程的乐趣，谢谢她在论文工作上各方面的指导与帮助，同时也为我不能及时提交作品而拖慢了些许毕设进程而道歉，谢谢董老师宽容与体谅。

最后，感谢我的家人，谢谢他们一直的鼓励与支持。

参考文献

[1] 郭芳芳.基于Web技术的毕业设计管理系统研究与实现[D].解放军信息工程大学,2013

[2]林阳灿.高校行政管理信息化建设研究[J].长江大学学报：社会科学版，2014（12）.

[3]马燕霞.关于深入推进高职院校行政管理信息化建设的思考[J].湖北函授大学学报，2014（5）

[4] 李洋. SSM框架在Web应用开发中的设计与实现[J]. 计算机技术与发展，2016(12)

[5] 荣艳东. 关于Mybatis持久层框架的应用研究[J]. 信息安全与技术, 2015(12)

[6] 王永和,张劲松,邓安明,周智勋. Spring Boot研究和应用[D]. 信息通信,2016(10)

[7] 孙铀.学生毕业论文管理系统的设计与实现[D].大连：大连理工大学,2007

[8] 李玲.基于B/S的毕业设计管理系统研究与实现[D].青岛理工大学,2014