

本科毕业设计

论文题目： 在线组卷系统的设计与实现

学生姓名： 张迪

学号： 201511010123

专业： 计算机科学与技术

指导教师： 丁艳辉

学 院： 信息科学与工程学院

2019年 03 月 05 日

目 录

[摘要 - 1 -](#_Toc9334055)

[Abstract - 1 -](#_Toc9334056)

[1. 引 言 - 2 -](#_Toc9334057)

[1.1应用背景与价值 - 2 -](#_Toc9334058)

[1.2开发工具与技术 - 2 -](#_Toc9334059)

[2. 需求分析 - 4 -](#_Toc9334060)

[2.1可行性分析 - 4 -](#_Toc9334061)

[2.2功能需求 - 4 -](#_Toc9334062)

[2.3非功能性分析 - 6 -](#_Toc9334063)

[3. 概要设计 - 8 -](#_Toc9334064)

[3.1总体结构设计 - 8 -](#_Toc9334065)

[3.2基于MVC模式 - 9 -](#_Toc9334066)

[3.3系统业务流程 - 10 -](#_Toc9334067)

[4. 详细设计 - 12 -](#_Toc9334068)

[4.1数据库逻辑结构设计 - 12 -](#_Toc9334069)

[4.2数据库表结构设计 - 15 -](#_Toc9334070)

[5. 系统实现 - 17 -](#_Toc9334071)

[5.1题库管理 - 17 -](#_Toc9334072)

[5.1.1 题库查看 - 17 -](#_Toc9334073)

[5.1.2 试题录入 - 19 -](#_Toc9334074)

[5.2在线组卷 - 21 -](#_Toc9334075)

[5.2.1手动组卷 - 21 -](#_Toc9334076)

[4.2.2自动组卷 - 23 -](#_Toc9334077)

[5.3试卷下载 - 25 -](#_Toc9334078)

[6. 总结与展望 - 27 -](#_Toc9334079)

[参考文献： - 28 -](#_Toc9334080)

在线组卷系统的设计与实现

张迪

（山东师范大学 信息科学与工程学院 计算机科学与技术专业 计工本方向2015级1班）

摘要**：**为了满足在线组卷的需要，本文设计并实现了一款在线组卷系统。该系统主要包括题库管理，试卷管理，用户管理，手动组卷，自动组卷以及试卷下载等功能，为教师和管理员用户提供了相应的服务。

该系统基于SSM框架，以JAVA作为开发语言，以Mysql作为后台数据库，对在线组卷系统进行开发与实现。

**关键词**：在线组卷系统；SSM；Java；Mysql;

Design and Implementation of Online generating test paper System

Zhang-Di

（School of Information Science and Engineering, Shandong Normal University）

Abstract: In order to meet the needs of Online generating test paper, this paper designs and implementations an online genreating test paper system. The system mainly includes the function of question bank management, test paper management, user management, Manual test paper generation, auto-generating test paper and test paper download. Which provides corresponding services for teachers and administrator users.

The system is based on SSM framework, with Java as the development language, Mysql as the background database, and the system is developed and implemented.

**Key words:** Online generating test paper System; SSM; Java; Mysql.

# 引 言

## 1.1应用背景与价值

随着计算机技术和网络的高速发展，社会上的各行各业都受到了互联网的巨大影响[1]，比如“互联网+商品”产生网购，“互联网+社交”产生了各种网络社交软件。而“互联网+教育”也发生了剧烈的化学反应，像网上教学，在线考试，视频面试等都是“互联网+教育”的产物[2]。相比于传统的教育方式，互联网式的教育赋予给了它更加鲜活的生命力，传统的教育方式都只能局限在一间教室里，而互联网式的在线教育方式更有利于拓展学生们获取知识的途径，也更能使学生的碎片时间得到更加有效的利用[3]。

在上述教育背景下，线上的考试方式也逐渐在各大学校普及开来。一套试卷的结构、题目质量以及合理程度都是非常重要的，所以组卷就成为了整个考试流程中最为核心的一步。

不管是适合于线上的考试还是不适合于线上的考试，一套优秀的在线组卷系统都能为它们提供更多的便利。这种在线组卷可以通过教师手动选择题库中试题从而生成一套试卷，也可以通过教师设置的试卷结构来自动组成一套试卷，比传统的组卷方式更加的灵活，同样也让组成的试卷更加合理和客观[4]。

## 1.2开发工具与技术

(1) IDEA：IDEA全称为IntelliJ IDEA

IntelliJ IDEA主要用于支持 Java、Scala、Groovy 等语言的开发工具，同时具备支持目前主流的技术和框架，擅长于企业应用、移动应用和 Web 应用的开发[5]。

(2) MySQL：本系统使用Mysql存储并管理后台数据，MySQL是一种开源的关系型数据库管理系统[6]，是当前应用主流之一的数据库[7]。因为MySQL是开放源代码并且免费的，所以任何人都可以根据自己的需求对其进行修改，并且它的速度、可靠性以及适应性都是极高的，而且它的使用简单易上手，几乎没有什么门槛，这也更加促使MySQL成为了当前主流的数据库管理系统之一。

(3) Java：Java是由Sun公司于1995年推出的一门面向对象的高级程序设计语言[8]。

在《Java编程思想》中提出Java语言是一种“Everything is object”的语言，它能够直接反映现实生活中的对象。并且Java具有许多的优点：平台无关性，提供很多内置的类库，提供了对WEB应用开发的支持，具有较高的安全性，跨平台，垃圾回收机制等等[9]。本系统即使用java最为后台开发语言。

(4) Spring：Spring是一个轻量级的Java开发框架，能够极大的简化Java的后台开发工作。Spring包括很多模块，我们可以根据自己的需求选择性的引入需要用到的模块，并且Spring是非侵入性的，这意味着业务逻辑代码可以不依赖于框架本身，我们可以只专注于业务功能的实现，而不必关心框架内部的处理逻辑。Spring的两大核心是控制反转（IOC）和面向切面（AOP）[10]。

# 需求分析

## 2.1可行性分析

传统的手动线下组卷的方式存在着许多缺点，如效率低、安全性不高、存储不方便等，所以目前许多学校的考试的试卷已经由人工线下组卷转换变成了采用计算机来进行题库管理和试卷的生成。

学校想要进行一次考试，那肯定需要先组成一套合理的试卷，这样对学生的检验和考查更能体现出客观性。而组卷的过程一般都是教师从庞大的题库中去挨个挑选出合适的试题，亦或是从相关的试题书中一道道的找出来从而组成一套试卷。这样做虽然能够从教师的专业程度上保证了试卷的质量，但是也会导致大量的时间和力气浪费在组卷上。一场考试流程下来，如果在组卷上消耗了如此多的师资力量，那么这场对学生们的检验考试无疑就失去了意义甚至是失败的。

而在线组卷系统能够帮助学校省去这繁杂的组卷过程，减少组卷时间，同时也能使教师空出更多的时间来对学生进行更有意义的教育[11]。

如今的计算机技术早已足够支撑这种在线组卷系统；在管理上，该系统主要是运行在每个学校内部，因此实现时做成单机版即可；在安全性上通过分成管理员和教师两种角色进行密码登录验证也能够得到保证。

## 2.2功能需求

设计在线组卷系统需要考虑到以下几点：

（1）要组成一套试卷必须要有一个题库能够对试题进行增添删除修改等操作

（2）需要管理员和教师两种角色才能更加合理的对组卷系统进行使用

（3）组卷方面包括手动组卷和自动组卷

根据上述考虑，将系统分为三大模块来进行说明：用户模块，组卷模块和题库模块。

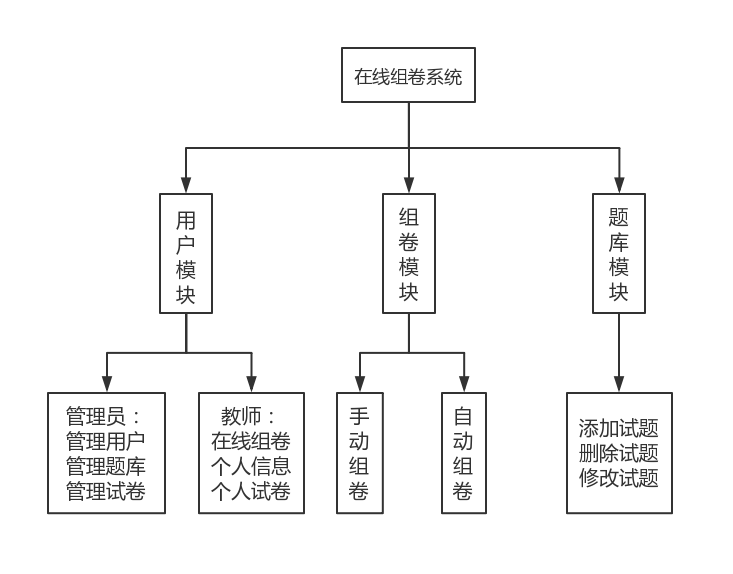


图1. 系统功能模块图

1. 用户模块：
2. 管理员：功能说明及用例图如下：
   1. 管理用户：添加或者删除教师用户，修改教师信息比如姓名，密码等。
   2. 管理题库：进行试题的添加、删除和修改操作。
   3. 管理试卷：可以查看所以已组试卷，并对齐进行修改删除等操作。

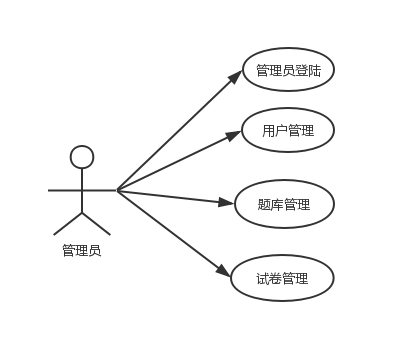


图2. 管理员用例图

1. 教师：功能说明及用例图如下：
   1. 在线组卷：通过手动组卷或者自动组卷两种方式组成一套试卷。
   2. 个人信息：教师登录系统可以对自己的个人信息进行修改，比如修改个人姓名，或者修改密码。
   3. 个人试卷：教师可以查看自己已组成的试卷，并且对其进行重新编辑。

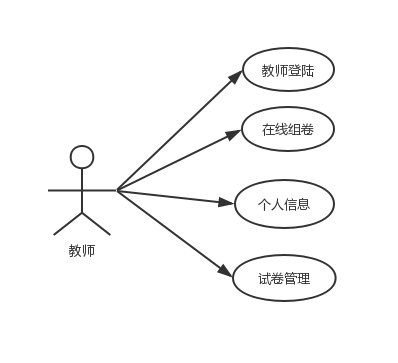


图3. 教师用例图

1. 组卷模块：
2. 手动组卷：教师可以从题库中挑选合适的试题组成一套试卷。
3. 自动组卷：教师可以通过设置试卷的题目类型，数目等各类属性然后自动生成一套试卷。
4. 题库模块：
5. 添加试题：管理员可以将各类试题录入到试题库，供组卷使用。
6. 删除试题：管理员可以对一些不完整或者没有存在价值的试题进行删除操作。
7. 修改试题：当输入的试题发现有错误时，管理员可以进行及时的修改。

## 2.3非功能性分析

该系统属于校园内部使用，所以只需要部署在学校的服务器上即可，不需要连接外网，制作为单机版，这样也能够保证信息的安全性和服务性能。而且该系统开发过程中模块划分的功能明确单一，充分实现了前后端分离，可扩展性和可维护性要强于复杂模块。网页上也使用ajax技术，通过后台与服务器进行少量的数据交换，ajax可以使网页实现异步更新，简单说就是网页局部更新的一种技术，这意味可以不用重新加载整个全部的网页，而对网页上的部分信息进行更新交互，极大的减少了服务器的压力[12]。

# 概要设计

## 3.1总体结构设计

该阶段主要是为了对整个系统的做一个全局的结构设计和划分[13]，为整个系统的开发指定一个总体的方向。将系统划分为两个大模块，一是属于管理员用户的后台管理模块，这一模块需要管理员账号登录，可以管理整个在线组卷系统的试题，试卷以及用户，二是属于教师用户的前台组卷模块，这一模块通过教师账号登录后可以浏览试题库，进行组卷操作等。

图4. 系统总体结构图

如图4所示，首先是用户上的结构划分，由于整个系统需要有后台管理和前台组卷两个子系统，所以需要定义管理员和教师两个角色。

1、由管理员管理后台，后台总体可以分为三个部分：

（1）题库管理：进入题库管理可以查看题库并对其进行修改和删除，同样也可以向系统中录入新的试题[14]。

（2）用户管理：进入用户管理可以查看用户并对用户进行修改删除，当然也可以添加新用户。

（3）试卷管理：管理所有用户已完成的试卷。

1. 由教师用户使用前台系统，前台总体划分为五个部分：

（1）手动组卷：手动操作试题完成组卷。

（2）自动组卷：自动操作试题完成组卷。

（3）当前试卷：查看当前正在组合的试卷信息。

（4）试卷库：个人已完成的试卷，可以在此部分对试卷进行重新编辑，下载和删

操作。

（5）个人中心：个人信息。

## 3.2基于MVC模式

本系统将基于MVC的编程模式进行开发。MVC是Model View Controller，是模型(model)－视图(view)－控制器(controller)的缩写[16]，它是用一种业务逻辑、数据与界面显示分离的方法来组织代码，将众多的业务逻辑聚集到一个部件里面。使用MVC有利于前后端分离开发，实现高内聚低耦合，也更益于维护和修改[17]。

Spring中配备了构建Web应用的全功能MVC框架Spring MVC，两者可以进行无缝整合。Spring MVC可以通过一套注解，将一个Java类变成一个前端控制器(Controller)，不需要实现任何接口，除此之外，Spring MVC在在数据绑定、视图解析、本地化处理及静态资源处理上都有不错的表现。

 Spring MVC围绕[DispatcherServlet](https://www.baidu.com/s?wd=DispatcherServlet&tn=24004469_oem_dg&rsv_dl=gh_pl_sl_csd)(前端控制器)为中心展开，DispatcherServlet(前端控制器)是Spring MVC的中枢，和MVC的思想一样，它负责从视图获取用户请求并且分派给相应的处理器处理，并决定用哪个视图去把数据呈现给给用户。

图5是Spring MVC的工作流程：

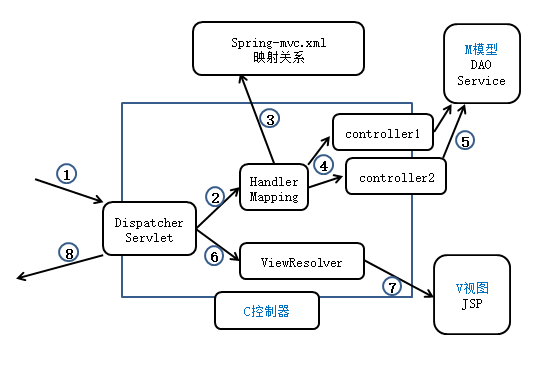


图5. SpringMVC的工作流程

用文字概括SpringMVC的工作流程：

1、浏览器发送一个HTTP请求

2、请求首先进入DispatcherServlet主控制器(通过web.xml引导进入)

3、主控制器调用HandlerMapping组件，根据请求找到映射的Controller处理

4、执行Controller处理方法，将返回结果给ViewResolver组件

5、ViewResolver组件根据Controller返回结果定位视图JSP，将模型数据传递给JSP

6、由JSP生成响应结果，给浏览器输出展示。

## 3.3系统业务流程

根据在线组卷系统的需求分析，本系统设计了两种用户，分别为后台管理员以及教师。

后台管理员通过管理员的账号密码登录进入本系统的后台管理页面，管理员可以通过导航栏菜单进入题库查看页面，试题录入页面，用户查看页面，添加用户页面以及试卷查看页面[18]。

在题库查看页面，管理员可以查看题库中已存在的试题，并且可以对试题进行修改和删除的操作。

在试题录入页面，管理员可以向题库中添加新的试题。

在用户查看页面，管理员可以查看当前系统中的教师用户信息，并且有权限去修

改或删除教师用户。

在添加用户页面，管理员可以向系统中添加新的教师用户。

最后的试卷查看页面，管理员可以在此页面中查看到当前系统中已经组好的试卷

列表，进一步可以查看每套试卷的详细内容，管理员也具有下载或删除试卷的权限。

教师通过教师账号和密码登录前台，前台页面的菜单包括五个子页面：手动组卷页面，自动组卷页面，当前试卷页面，试卷库页面，个人中心页面。

教师登录成功的主页面为手动组卷页面，可以查看试题并且挑选适合的题目组成试卷。而且在当前试卷页面可以进行试卷标题等属性的设置并可以查看已选试题。

进入自动组卷页面，可以通过设置试卷的相关属性来自动生成一套试卷，节省了教师挑选试题的时间，提高了效率。

进入试卷库页面，可以查看已保存的试卷，并且可以对其进行重新编辑，下载和删除操作。

个人中心页面，用户可以编辑自己的个人信息，也可在此页面注销登录。

# 详细设计

根据系统的需求分析以及用户的需求，分析出需要记录的数据，这些数据需要存储到数据库中，而如何将需记录的数据完整无误的存储到数据库中是至关重要的，这就需要对数据库的结构进行分析和设计。

本系统将采用E-R图的方法进行数据库的结构设计。E-R图也称实体图，提供了表示实体类型、属性和联系的方法，用来描述现实世界的[概念模型](https://baike.baidu.com/item/%E6%A6%82%E5%BF%B5%E6%A8%A1%E5%9E%8B/3187025)，它可以直观的描述数据的实体关系，介绍一下E-R图中各个形状所表示的意义[15]。

1. 矩形框：表示实体型，框内写明实体名称。
2. 椭圆形框：表示实体的属性。
3. 菱形框：表示实体型之间的联系。

## 4.1数据库逻辑结构设计

这里对该系统所需要存储的数据加以分析并且给出相对应的E-R图。

根据上述的总体结构来分析，从上向下，首先是角色，分为两个：管理员角色以及教师角色。

管理员信息实体属性图如下：

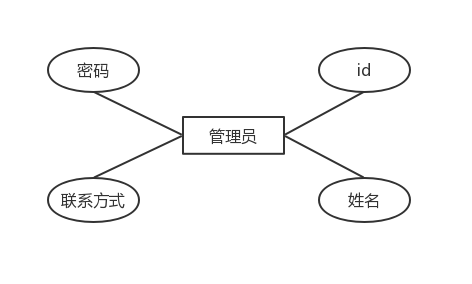


图6. 管理员实体属性图

教师信息实体属性图如下：

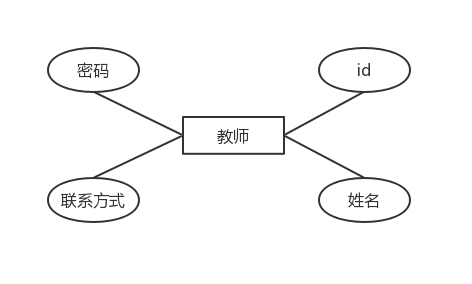


图7. 教师实体属性图

既然是组卷系统，一定要有试题，因为有各类各种各样的试题才能组成一套试卷。

试题实体属性图如下：

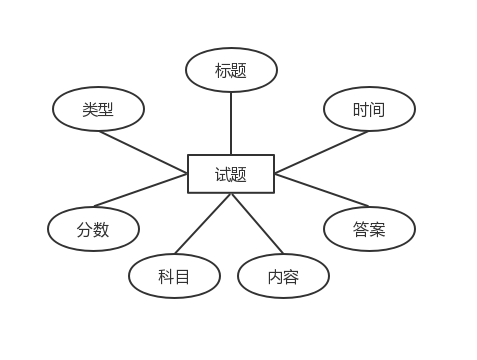


图8. 试题实体属性图

最后数据库需要存储一套试卷的信息，所以需要有试卷实体，试卷信息实体属性图如下：

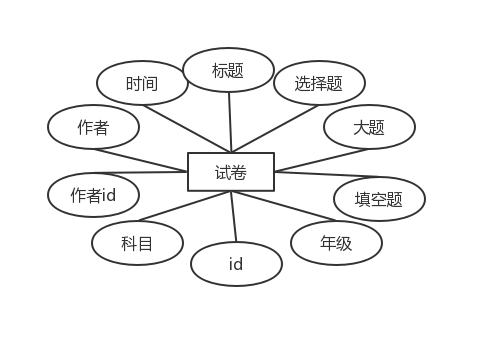


图9. 试卷实体属性图

整个系统的系统E-R图：

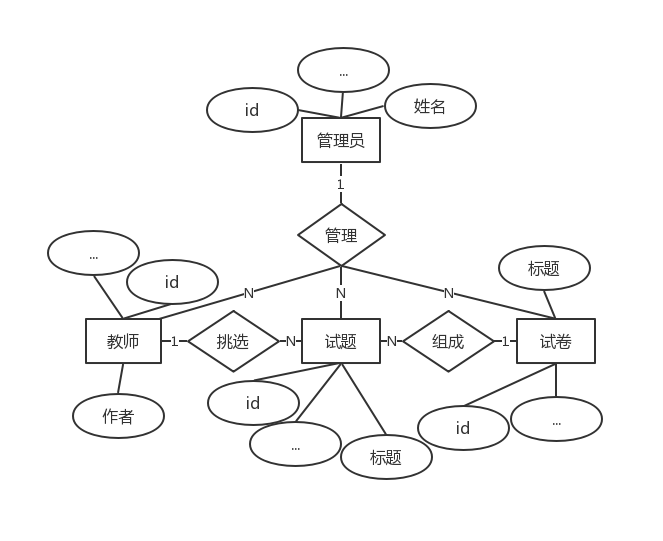


图10. 系统总的E-R图

## 4.2数据库表结构设计

该系统使用的是MySQL数据库。根据3.2.1中的分析以及E-R图本系统建立了Graduation数据库，并且建立了四张表，分别是：管理员信息表，教师信息表，试题信息表以及试卷信息表。

表1. 管理员信息表(admin)：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 列名 | 字段类型 | 长度 | 是否为空 | 默认 | 主键 | 备注 |
| id | varchar | 20 | NO | (Null) | 是 | 管理员登录ID |
| password | varchar | 20 | NO | (Null) | 否 | 管理员登录密码 |
| name | varchar | 10 | NO | (Null) | 否 | 管理员姓名 |
| contact | varchar | 30 | NO | (Null) | 否 | 管理员联系方式 |

表2. 教师信息表(teacher)：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 列名 | 字段类型 | 长度 | 是否为空 | 默认值 | 主键 | 备注 |
| id | varchar | 20 | NO | (Null) | 是 | 教师登录id |
| password | varchar | 50 | NO | (Null) | 否 | 教师登录密码 |
| name | varchar | 10 | NO | (Null) | 否 | 教师姓名 |
| contact | varchar | 30 | NO | (Null) | 否 | 教师联系方式 |

表3. 试题信息表(question)，该表所有字段都不能为空：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 字段类型 | 长度 | 默认值 | 主键 | 备注 |
| id | varchar | 20 | (Null) | 是 | 题目唯一编号 |
| type | varchar | 10 | (Null) | 否 | 题目类型 |
| score | int |  | (Null) | 否 | 题目分值 |
| sub | varchar | 10 | (Null) | 否 | 题目适合科目 |
| content | text | 65535 | (Null) | 否 | 题目内容 |
| answer | text | 65535 | (Null) | 否 | 题目答案 |
| time | timestamp |  | CURRENT\_TIMESTAMP | 否 | 插入时间 |

表4. 试卷信息表(exam\_paper)：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 字段类型 | 长度 | 默认值 | 主键 | 备注 |
| title | varchar | 100 | (Null) | 否 | 试卷标题 |
| id | varchar | 50 | (Null) | 是 | 试卷唯一编号 |
| author | varchar | 10 | (Null) | 否 | 创作者姓名 |
| author\_id | varchar | 20 | (Null) | 否 | 创作者id |
| sub | varchar | 10 | (Null) | 否 | 试卷适应科目 |
| grade | varchar | 10 | (Null) | 否 | 试卷适应年级 |
| q\_choice | text | 65535 | (Null) | 否 | 单选题 |
| q\_fill | text | 65535 | (Null) | 否 | 填空题 |
| q\_big | text | 65535 | (Null) | 否 | 大题 |
| time | timestamp |  | CURRENT\_TIMESTAMP | 否 | 更新时间 |

表中出现了一个类型为timestamp(时间戳)的字段time，它的默认值为CURRENT\_TIMESTAMP，这个属性的特点是对该表中数据进行插入或修改操作时，它都会自动更新，插入当前的系统时间。

# 系统实现

通过了对在线组卷系统的总体结构设计和数据库结构设计，整个开发的地基已经打好，接下来要做的就是对系统各个功能以及模块的实现和开发。

该部分会对系统中的主要功能的实现进行描述说明，并且会附上对应的界面截图以及相关的主要代码。

## 5.1题库管理

### 5.1.1 题库查看

管理员登录后台后的主页面就是题库查看。因为试题有不同科目和不同类型的，需要将各个试题按照科目和类型分类，因此在上方添加一个分类导航栏，在加载页面时展示的是全部的试题，如果选择科目和题型，后台会重新整理试题在前台展示相对应类型的试题；同时在每一个试题框上都有一个修改和删除按钮，能够对试题进行修改和删除操作，删除运用了Ajax网页异步刷新技术，所以删除试题时不会刷新整个页面，从而提高了操作效率；由于题目数量众多，所以底部有一个分页栏，一页有10道题目，这样使浏览题库的时候更加灵活，同样使页面加载更加快速[19]。

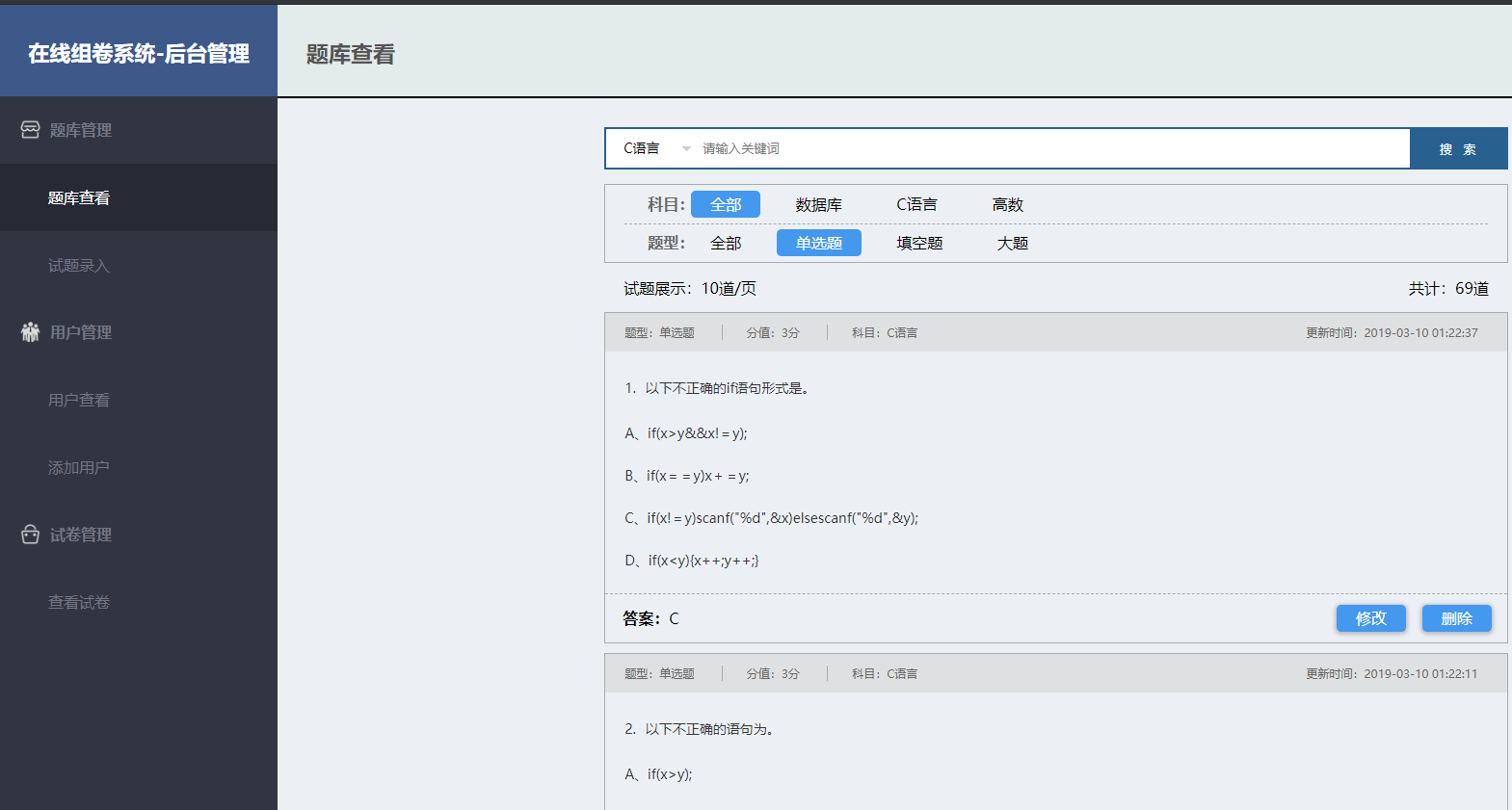


图11. 后台题库查看界面

1、DAO层中QuestionMapper.xml中的按类型搜索题目并按时间分页的数据库代码：

<select id="findAllQuestionLimit" resultType="Question">

select \* from question

<where>

<if test="sub!=null">

and sub=#{sub}

</if>

<if test="type!=null">

and type=#{type}

</if>

</where>

ORDER BY time DESC limit #{begin},10;

</select>

2、Controller中处理并跳转题库查看页面的部分代码：

int count = 0;

int page\_num = 0;

List<Question> list = this.qsl.doSelectAllQuestionLimit(10 \* (page - 1), sub, type);

count = this.qsl.doSelectCount(sub, type);

if (count <= 10) {

page\_num = 1;

} else if (count % 10 == 0) {

page\_num = count / 10;

} else {

page\_num = count / 10 + 1;

}

3、删除操作的Ajax代码以及后台代码：

function del(id){

if(confirm("确认删除该题么？！")){

$.ajax({

type:'POST',

url:'delTest',

data:{id:id,count:${count}},

success:function (data) {

//前端页面该题目消失，并且修改题目数量显示

$("#q-"+id).slideUp();

$("#q-count").text("共计：" + data + "道");

},

error:function () {

alert("删除失败！");

}

});

}

};

@RequestMapping({"delTest"})

@ResponseBody

public void toDelTest(HttpServletRequest request, HttpServletResponse response)

throws IOException{

String id = request.getParameter("id");

int count = Integer.parseInt(request.getParameter("count")) - 1;

//从后台删除该题目

if (this.qsl.doDeleteQuestion(id)){

response.setContentType("text/html;charset=utf-8");

PrintWriter out = response.getWriter();

//返回给前台当前题目数量

out.print(count);

out.flush();

out.close();

}

}

### 5.1.2 试题录入

根据之前的数据库表设计，试题共有7个属性，出了唯一编号id和自动填充系统时间的time之外，其它的五项：题型，科目，分数，内容，答案都需要管理员在添加试题时填写。而在设计开发试题录入功能时有许多细节需要注意。

因为科目和题型一般都是固定的几种，比如题型是单选题，填空题以及大题，因此这两项属性系统中设计为下拉框。而且当题型变化时，底部的题目内容和答案输入框也会随着变化，因为不同类型的题目答案属性不同。

由于题目内容的不可预见性，因此题目具体内容使用富文本编辑器录入，本系统使用的是Froala Editor。



图12. 后台试题录入界面

1、从富文本编辑器中获取图片并保存到本地的后台代码：

CommonsMultipartFile orginalFile = (CommonsMultipartFile) request.getFile("file");

Map<String, String> map = new HashMap<>();

String savePath = request.getServletContext().getRealPath("/WEB-INF/img/q\_img/");

String url = "img/q\_img/"; //一会回显图片的url

if(!orginalFile.isEmpty()) { //如果上传的文件不为空

String myFileName = orginalFile.getOriginalFilename();

if (myFileName.trim() != "") {

SimpleDateFormat sdf = new SimpleDateFormat("yyyyMMddHHmmss");

String filename = sdf.format(new Date()) + "upload" + orginalFile.getOriginalFilename();

savePath = savePath + filename;

File localFile = new File(savePath);

try {

orginalFile.transferTo(localFile);

} catch (IOException e) {

e.printStackTrace();

}

url = url + filename; //将url补充完整

}

}

/\*因为富文本编辑框内需要回显图片，所以需要将img的url传回前端，而且这里需要使用map存储，然后转换为json对象，这样前端接收到的才能识别\*/

map.put("link",url);

response.setContentType("json/html;charset=utf-8");

PrintWriter out = response.getWriter();

JSONObject json = new JSONObject(map);

out.print(json);

out.flush();

out.close();

## 5.2在线组卷

### 5.2.1手动组卷

手动组卷大体流程一般是用户从网页上的题库中挑选合适的试题，挑选完毕后，将试卷的标题等一些必需的属性设置好，然后将试卷保存或下载。

挑选试题时，有一个选择按钮，点击选择，题目被选上，同时按钮变成灰色的移除按钮，这样不会导致用户重复选题。

后台使用session[20]存储当前已选的试题的id，这样做的好处是：当用户切换页面时，后台仍然会记住你当前已选择的试题，切回选题页面，试题篮中不会清空你之前选择的题目。在当前试卷页面也是如此，用户随意切换页面，系统不会清空你已设置好的试卷标题等属性。

在挑选过程中，用户可能会忘记自己选择题目的类型和对应的数量，所以这时候需要一个试题篮，当用户选择或者移除试题时，试题篮都会有相应的变化，这样可以让用户挑选试题时对自己已选的试题有一个更加直观的了解。

当用户挑选试题完毕后，点击试题篮中的生成试卷，页面将跳到当前试卷的页面，用户会看见自己已选的试题组成的一套试卷，也可以直接点击当前试卷跳转页面。



图13. 手动组卷界面

1. 选择试题时的部分后台代码：

String id = request.getParameter("id");

Set<String> nowList;

if(session.getAttribute("nowList") == null){

nowList = new TreeSet<>();

}else{

nowList = (TreeSet)session.getAttribute("nowList");

}

//将已选择的题目的id放入noeList中然后放入session中

nowList.add(id);

1. 当点击生成试卷时，后台需要将已经选择的试题进行处理，即将之前存在session中的试题ID的链表取出来，然后分类传到当前试卷的页面上进行展示。要注意的是需要考虑到试题篮中没有试题的情况。

if(session.getAttribute(“nowList”) == null){

model.addAttribute(“msg” , “当前没有选择题目！”);

}else{

Set<String> nowList;

nowList=(TreeSet)session.getAttribute(“nowList”);

}

因为设置试题的唯一id开头1为单选，2为填空，3为大题，所以根据试题id将试题分类存储在List中。

for (String s : nowList) {

if(s.startsWith("1")){

c\_list.add(qsl.doSelectQuestionById(s));

}else if(s.startsWith("2")){

f\_list.add(qsl.doSelectQuestionById(s));

}else{

b\_list.add(qsl.doSelectQuestionById(s));

}

}

### 4.2.2自动组卷

自动组卷的功能就是将试卷必需的属性信息填写完整后，点击生成试卷就能生成一套试卷。



图14. 自动组卷界面

1. 填写题目数量的时候，系统要限制它只能填写0~100之内的正整数，这需要在前端使用JS监听这三个input。

$(document).on("input propertychange",".question-set input",function(){

var limitNum = $(this).val().replace(/[^0-9]+/g, "");

if(limitNum>=0&&limitNum<=100){

$(this).val(limitNum);

}else{

$(this).val("99");

}

});

1. 自动组卷中随机获取试题的方法：

//随机取得type类型的num道题目

public List<String> randGetQuestion(int num , String type , String sub){

List<String> ids = qsl.doSelectIdByType(type,sub);

Collections.shuffle(ids);

return ids.subList(0,num);

}

## 5.3试卷下载

组卷的最终目的还是为了考试，由于学校环境不同，考试科目知识繁琐度不一致等等一些因素的存在，如今的考试是线上线下并存的模式。如果是线上考试需要，那系统可以创建一个接口，与需要试卷的考试系统相对接，考试系统只需要调用接口就能获取该系统组成的试卷。而如果是线下考试需要，仍然需要从计算机上打印成纸质试卷，于是本系统提供了一个试卷下载的功能，其中使用FreeMarker模板引擎制作word模板[21]。

用户可以在当前试卷的页面直接下载试卷，而不需要保存。



图15. 试卷下载按钮

也可以保存试卷后在试卷库下载试卷。同样的，管理员在后台也能下载试卷。



图16. 后台试卷管理界面

用户可以点击组卷记录中的试卷标题查看或下载试卷。

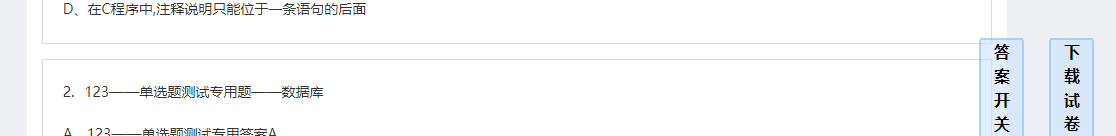


图17. 试卷查看界面

1. 试卷word文档下载的方法代码：

public void createWord(Configuration configuration,

Map<String, Object> dataMap,HttpServletResponse response){

Template t=null;

Writer out = null;

try {

configuration.setClassForTemplateLoading(this.getClass(),"/");

t = configuration.getTemplate("word.xml"); //获取模板文件

response.setContentType("application/x-msdownload;");

response.setHeader("Content-Disposition","attachment;filename="+

URLEncoder.encode((String)dataMap.get("title")+"-"+

(String)dataMap.get("sub")+"-"+(String)dataMap.get("grade"),

"UTF-8") + ".doc");

out = new BufferedWriter(new OutputStreamWriter

(response.getOutputStream(),"utf-8"));

t.process(dataMap, out); //将填充数据填入模板文件并输出到目标文件

} catch (Exception e1) {

e1.printStackTrace();

}finally {

try {

out.close();

} catch (IOException e) {

e.printStackTrace();

}

}

}

# 总结与展望

经过三个多月的努力，毕业设计终于完成了。因为也是想经过毕业设计这一过程来检验一下自己，所以在选择课题时，选了一个设计类的毕业课题。回想整个制作毕业设计的过程，有苦有甜，苦的是在开发的过程中经常会因为一些问题扰乱自己的心态，也会经常因为一些BUG而不得不去熬夜修改；甜的是自己在这次完整的毕业设计过程中学到了许多，也对自己的心态有了一个更好的磨炼。

当我选到这个课题时，一开始虽然脑海里有一个开发的大体方向，但是有许多细节方面的东西在我脑海里是比较模糊的。幸运的时遇到了一个好的指导老师，在毕设开始前，导师开会对我所选课题的系统进行了描述和指导，通过老师的指导，我脑海中开发的整个方向以及细节终于有了明显的框架。

从一开始的课题研究，课题背景等去查阅资料，到设计系统的需求分析，数据库设计等初期设计，再到最后的系统开发测试。通过此次的毕业设计，完成了整个开发前中后期的流程，不仅检验巩固了我之前所学的知识，也对自己以后的工作学习积累了大量的经验。

本系统的设计与开发主要是为了实现线上的手动和自动组卷，给学校的教师节省出了考试流程中的大部分时间，对现代化互联网环境下的教育有着很深的意义。但是由于技术能力有限，整个系统还是存在有一些问题的，对于系统的安全性和稳定性有待进一步提高，而且可能还存在一些小小的BUG。总之希望之后能够对该系统进行完善，在组卷方面添加一些难度等因素，使用更加高效的组卷算法，并且希望增加一些新的业务功能，既能提供线上考试的组卷，也能对线下考试的提供支持。