**信息系统设计报告**

**题目：** 教育互评子系统

**姓 名：** 杨攀原

**学 号： 2013391078**

**专业班级：** 专硕3班

目录

[第一章 项目计划 5](#_Toc76671602)

[1.1 项目由来 5](#_Toc76671603)

[1.2 提案人员 5](#_Toc76671604)

[1.3 目的 5](#_Toc76671605)

[第二章 项目提案分析 5](#_Toc76671606)

[2.1 项目说明 5](#_Toc76671607)

[2.2 项目背景 5](#_Toc76671608)

[2.3 项目目标 6](#_Toc76671609)

[2.4 项目可行性分析 6](#_Toc76671610)

[2.4.1 技术可行性 6](#_Toc76671611)

[2.4.2 操作可行性 6](#_Toc76671612)

[第三章 需求建模 7](#_Toc76671613)

[3.1 整体需求 7](#_Toc76671614)

[3.2 功能需求 8](#_Toc76671615)

[需求编号规则 8](#_Toc76671616)

[总体功能模块划分 9](#_Toc76671617)

[3.3 功能性需求 9](#_Toc76671618)

[3.3.1 用户管理功能(um) 9](#_Toc76671619)

[3.3.2 课程管理功能(cm) 10](#_Toc76671620)

[3.3.3 作业管理功能(hm) 11](#_Toc76671621)

[3.4 非功能性能需求 12](#_Toc76671622)

[3.4.1 性能需求 12](#_Toc76671623)

[3.4.2 可靠性与可用性需求 12](#_Toc76671624)

[3.4.3 安全性需求 12](#_Toc76671625)

[3.5 数据流图 13](#_Toc76671626)

[3.6 数据字典 14](#_Toc76671627)

[第四章 系统设计 15](#_Toc76671628)

[4.1 参与者用例分析 15](#_Toc76671629)

[4.2 系统设计 16](#_Toc76671630)

[4.2.1 系统架构 16](#_Toc76671631)

[4.2.2 系统具体功能实现流程图 16](#_Toc76671632)

[4.3 数据库设计 18](#_Toc76671633)

[4.3.1 E-R图 18](#_Toc76671634)

[4.3.2 数据建表 19](#_Toc76671635)

[第五章 系统实现 24](#_Toc76671636)

[5.1 开发工具 24](#_Toc76671637)

[开发环境 24](#_Toc76671638)

[开发框架 24](#_Toc76671639)

[5.2 关键代码模块 25](#_Toc76671640)

[5.2.1 课程管理模块 25](#_Toc76671641)

[5.2.2 作业管理模块 34](#_Toc76671642)

[第六章 系统测试 41](#_Toc76671643)

[6.1 测试项目说明 41](#_Toc76671644)

[数据与数据库完整性测试 41](#_Toc76671645)

[功能测试 43](#_Toc76671646)

[性能测试 44](#_Toc76671647)

[6.2 缺项分析 45](#_Toc76671648)

[第七章 用户手册 46](#_Toc76671649)

# 第一章 项目计划

## 1.1 项目由来

最近某实验室有同学正在研究同行互评方向，但在研究过程中遇到了一些问题。首先是网上相关的研究只有少量公布的同伴互评的数据集，而且这些少量公开的数据集只有互评的分数信息，缺少题目等相关具体信息，如果使用这样的数据集进行研究，使得研究方向严重受限。其次对于一些已经存在的相关网上授课平台，例如中国大学生慕课，广西大学慕课提供了相关的互评作业功能。但是这样的大型公开平台收集到的数据没有公开。因此，我们决定开发一个可以进行作业互评的相关教学平台，通过这个平台，我们可以让老师参与其中的教学环节，让教师布置作业，学生在其中完成作业并参与互评流程。这样的系统既可以帮助老师提供平台布置作业，采用同伴互评让学生自己打分减轻老师的负担，又可以收集学生在打分时的相关数据，为我们的同伴互评研究提供真实且与研究方向相符合的数据集。由于之前实验室已经完成并正在维护一个教学系统，因此将互评系统作为整个教学系统的子系统进行嵌入。

## 1.2 提案人员

提案人员：数据与无线网络实验室（dawnlab）主任，从事同行互评研究的研究者。

## 1.3 目的

在原有的教学系统“会了吗”上建立一个子系统，这个系统整合了系统中的作业模块，提供了学生作业互评这一环节，可以通过这一环节栋为学生打分，并且将收集到的同伴互评数据存储到数据库中。

# 第二章 项目提案分析

## 项目说明

产品名称：教育互评系统；

开发者：杨攀原；

用户群体：面向学生和教师；

## 项目背景

近年来，massive open-access online courses，简称MOOCs，越来越受欢迎。只要能上网，任何人都可以报名参加免费的大学课程。网上授课从线下教育模式演变而来，但两者之间仍然存在一些差异，导致传统教育的所有功能不能全部适用于线上教育。例如，在评估主观的学生作业(如数学证明、设计问题和论文)方面，线上教育受到了较大的约束。这是因为传统的线下教育人数通常不会超过一百人，一个老师就可以进行一对一的作业批改，但是对于一些热门的MOOCs课程，学生的人数可能会达到成百上千人，传统的老师评价方法无法适用。同行评估（Peer Assessment）则为这个问题提供了一个有前途的解决方案，它可以在短时间内对数万甚至数十万学生的课程中的主观作业进行评估和反馈。

## 项目目标

* 目前同伴互评的热门研究方向之一是使用概率模型的方法优化同伴互评的分数汇总模型，而目前还没有出现相关的公开数据集。
* 该项目的目的是建立一个学生互评的系统，可以为教师提供作业互评的平台，从而收集相关数据用于实验。这个系统的所有课程信息由自己创建而非外部导入。

## 项目可行性分析

### 2.4.1 技术可行性

对新软件功能的具体指标、运行环境及条件、响应时间、存储速度及容量、安全性和可靠性等要求；对网络通信功能的要求等；确定在现有资源条件下，技术风险及项目能否实现等。其中的资源包括已有的或可以取得的硬件、软件和其他资源，现有技术人员的技术水平和已有的工作基础。

服务端：使用springboot框架快速开发，需要部署在稳定的操作系统上，用户在交互中的响应平均时间不超过0.5秒，最大响应时间不超过3秒。

客户端：互评系统可以在网页端运行，浏览器需要支持html5。

### 2.4.2 操作可行性

该系统的逻辑与实际互评的逻辑相同：

首先，由教师在系统上发布作业，学生进入答题阶段。然后，待所有学生都答完提交后，进入互评阶段。最后，当所有学生互评完成后，对作业分数进行汇总，得到每个作业的估分。互评系统目前主要研究领域有两个，一个是作业的分配给合适的用户的算法，另一个是作业的汇总算法。以下是系统的流程图。



系统流程图

# 第三章 需求建模

## 3.1 整体需求

**教师的需求**：

教师可以在系统上建立一门课程，并在课程中添加若干个学生，这门课程可以与老师在线下的教学相对应。老师在建立课程之后，可以设置需要学生完成的作业，为了面向学生互评的批改方式，这里使用主观题。

教师在布置作业时需要输入以下参数：作业名称、作业需要的试卷名，作业开始的时间和截止时间、作业的评价方式。

教师在布置作业前，可以优先录入试题和试卷。其中试卷由若干个试题组成，教师可以在布置作业时直接选择试卷来布置相应的题目。

作业的评价方式分别为：互评和教师批改。如果教师选择了评价方式为教师批改，则所有的作业全部由教师一个人批改，不使用互评功能。如果选择了互评模式，则需要额外输入互评的起止时间。互评的时间必须在全部作业提交之后。互评结束后，系统将会对作业进行汇总，计算出每份作业的最终的评分。

老师布置作业后，互评作业就开始了，互评的阶段主要分为3个阶段：布置作业阶段，作业发布阶段，互评作业阶段。

* 布置作业阶段，这个阶段老师需要对作业进行初始化，设置作业的内容、起止时间等，此阶段一直持续到作业的开始时间之前。这个阶段学生无法查看作业内容。
* 作业发布阶段，这个阶段从作业开始时间一直到作业结束时间，这个时间段里学生可以提交答案并能在结束时间前修改作业。此阶段结束后学生无法修改自己的答案。
* 互评作业阶段，如果老师没有在互评模式时选择“互评”，则跳过该阶段。进入互评阶段，每个学生会分配到若干份作业，需要对这些作业进行打分。打分的时间需要在互评的起止时间内。

**学生的需求**：

学生可以登录系统，在规定的时间里提交作业答案和评价作业，最终获得在作业中的得分。

**管理人员的需求**：

管理人员可以查看并修改用户的信息，可以查看课程、作业的相关信息。对于已经超过了起止时间的课程和作业，不能进行修改。同时系统提供收集数据的功能，可以下载同伴互评的相关规范化数据。

## 功能需求

### 需求编号规则

同伴互评系统-用户管理模块-001权限设置

同伴互评系统-用户管理模块-002登录功能

同伴互评系统-用户管理模块-003创建不同角色账号

同伴互评系统-用户管理模块-004用户信息修改

同伴互评系统-课程管理模块-001创建课程

同伴互评系统-课程管理模块-002修改课程信息

同伴互评系统-作业管理模块-001创建题目

同伴互评系统-作业管理模块-002创建试卷

同伴互评系统-作业管理模块-003创建作业

同伴互评系统-作业管理模块-004提交答案

同伴互评系统-作业管理模块-005进行互评

### 总体功能模块划分

根据业务和功能划分，可以划分为：用户管理模块（um）、课程管理模块（cm）、作业管理模块（hm）。

## 功能性需求

### 用户管理功能(um)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **模块名称** | | 用户管理功能 | | | |
| **模块简介** | | 用户管理功能描述 | | | |
| **模块功能列表** | | | | | |
| **序号** | **一级功能** | | | **二级功能** | |
| **功能名** | | **功能编号** | **功能名称** | **功能编号** |
| 1 | 权限设置 | | pasum001 |  |  |
| 2 | 登录功能 | | pasum002 |  |  |
| 3 | 创建不同角色账号 | | pasum003 |  |  |
| 4 | 用户信息修改 | | pasum004 |  |  |

#### 3.3.1.1 权限设置pasum001

在互评系统中，每一个用户都被赋予了一个角色，每一个角色都根据自身的特点被分布了相应的权限，角色不能跨越自己的权限来对系统进行操作。

#### 3.3.1.2 登录功能pasum002

进入系统时，每个用户根据自己的账号进行登录，登录时，系统会根据用户的角色权限显示相应的交互界面。

#### 创建不同角色账号pasum003

对于管理员角色，可以创建教师和学生的账号。

对于教师角色，可以创建自己学生的账号。

对于学生角色，不可以创建账号。

#### 用户信息修改pasum004

管理员可以修改任意用户的信息，教师可以修改自己和自己课程下的学生的部分信息，学生只能修改自己用户的信息。

### 课程管理功能(cm)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **模块名称** | | 课程管理功能 | | | |
| **模块简介** | | 课程管理功能描述 | | | |
| **模块功能列表** | | | | | |
| **序号** | **一级功能** | | | **二级功能** | |
| **功能名称** | | **功能编号** | **功能名称** | **功能编号** |
| 1 | 创建课程 | | pascm001 | 创建课程 | pascm001001 |
| 添加课程学生 | pascm001002 |
| 2 | 管理课程 | | pascm002 | 修改课程信息 | pascm002001 |
| 修改课程学生 | pascm002002 |

#### 创建课程pascm001

##### 3.3.2.1.1 创建课程pascm001001

“课程”概念对于物理世界中教师所教授的一门课程。教师可以通过创建一个课程来关联选择这门课的所有学生。

##### 添加课程学生pascm001002

新建的课程下是没有学生账号的，需要教师手动添加学生账号。

#### 管理课程pascm002

##### 修改课程信息pascm002001

创建课程后可以对课程的基本信息进行修改。

##### 修改课程学生pascm002002

可以随时对课程中的学生进行删减。

### 作业管理功能(hm)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **模块名称** | | 作业管理功能 | | | |
| **模块简介** | | 作业管理功能描述 | | | |
| **模块功能列表** | | | | | |
| **序号** | **一级功能** | | | **二级功能** | |
| **功能名称** | | **功能编号** | **功能名称** | **功能编号** |
| 1 | 创建题目 | | pashm001 |  |  |
| 2 | 创建试卷 | | pashm002 |  |  |
| 3 | 创建作业 | | pashm003 |  |  |
| 4 | 提交答案 | | pashm004 |  |  |
| 5 | 进行互评 | | pashm005 |  |  |

#### 创建题目pashm001

题目包括单选题、多选题、填空题、判断题、主观题。题目与课程无关，只与教师有关，教师可以事先创建题目用于创建试卷。

#### 创建试卷pashm002

试卷表示事先定义好的一个题目集。创建试卷的过程中可以为随意添加、创建题目，并且为每个习题调整分数。创建作业时可以直接选择试题。

#### 创建作业pashm003

教师在布置作业时需要输入以下参数：作业名称、作业需要的试卷名，作业开始的时间和截止时间、作业的评价方式。

#### 提交答案pashm004

这个时间段里学生可以提交答案并能在结束时间前修改作业。此阶段结束后学生无法修改自己的答案。

#### 进行互评pashm005

如果老师没有在互评模式时选择“互评”，则跳过该阶段。进入互评阶段，每个学生会分配到若干份作业，需要对这些作业进行打分。打分的时间需要在互评的起止时间内。

## 非功能性能需求

### 3.4.1 性能需求

使用springboot框架快速开发，需要部署在稳定的操作系统上，用户在交互中的响应平均时间不超过0.5秒，最大响应时间不超过3秒。

### 3.4.2 可靠性与可用性需求

系统在启动后一直到系统关闭停止运行期间，不能出现系统崩溃的情况，系统运行流畅，同时运行期间不会受到其他应用软件的影响。

### 3.4.3 安全性需求

对数据的安全需要有一定的保障，数据的存储和备份等都必须满足要求。

## 数据流图



0层数据流图



1层数据流图

## 数据字典

用户（用户ID， 姓名， 性别， 密码， 盐值， 邮箱， 电话， 头像， 角色， 是否注销）

课程（课程ID， 课程名， 授课教师ID）

试题（问题ID， 试题类型， 题干， 选项， 答案， 答案解析， 试卷ID， 所属用户ID）

试卷（试卷ID， 试卷所属用户ID， 试卷名称， 试卷描述）

作业（作业ID， 所属课程ID， 所属教师ID， 作业名称， 作业描述， 所属课程ID， 评分类型， 开始时间， 结束时间， 互评开始时间， 互评结束时间， 是否删除）

1. 角色赋予了用户权限，有管理员、学生、教师。
2. 试题类型分为选择、填空、简答
3. 作业的评分类型可以选择为教师评分和学生互评

# 第四章 系统设计

## 4.1 参与者用例分析

教师功能：

1. 课程管理：

* 创建课程：课程是将学生与作业关联起来的组件，一个课程由一个老师和多个学生组成，课程对应老师线下所开设的一门课程。老师需要创建一个课程来关联其他学生账号。
* 管理课程：教师可以通过对课程的描述个性化课程，可以使课程更容易被学生选择。
* 管理学生：教师可以通过课程来添加学生账号，使得学生能够加入课程，从而看到课程的内容和作业。

1. 作业管理：

* 创建习题：教师在创建作业之前，需要事先录入习题。习题分为单选、多选、主观、填空、判断。习题可以在创建试卷的时候添加。
* 创建试卷：教师可以创建试卷，每张试卷中可以添加若干个习题，试卷还为每个习题赋予了分值。
* 创建作业：创建作业需要输入以下参数：作业名称、作业需要的试卷名，作业开始的时间和截止时间、作业的评价方式。作业的评价方式分别为：互评和教师批改。如果教师选择了评价方式为教师批改，则所有的作业全部由教师一个人批改，不使用互评功能。如果选择了互评模式，则需要额外输入互评的起止时间。互评的时间必须在全部作业提交之后。互评结束后，系统将会对作业进行汇总，计算出每份作业的最终的评分。
* 查看作业完成结果：作业提交时间结束之后，教师可以查看、批改学生的作业完成情况，从而了解到学生的学习状况。

学生功能：

1. 提交作业答案：学生登录系统后，可以通过选择相应的课程来完成老师布置的作业。作业在规定时间内都可以进行修改，规定时间结束之后无法进行修改。
2. 互评功能：如果老师选择了互评方式批改作业，学生需要在作业提交截止时间之后的规定互评阶段时间内对作业进行评分。每个学生会被分配若干份作业，需要对所有的作业进行打分。未参加互评的学生的成绩将会受到影响。

## 系统设计

4.2.1 系统架构



* + 1. 系统具体功能实现流程图
       1. **课程管理流程**



教师管理课程的流程为：

1. 创建课程，输入课程的名称、起止时间、课程信息
2. 为一个课程关联学生账号，每个学生账号可以关联多个课程。
   * + 1. **作业管理流程**

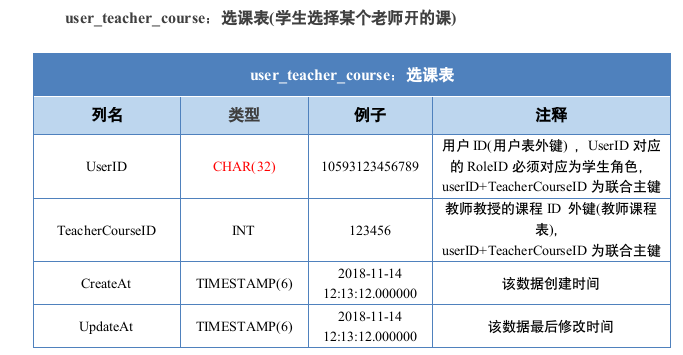
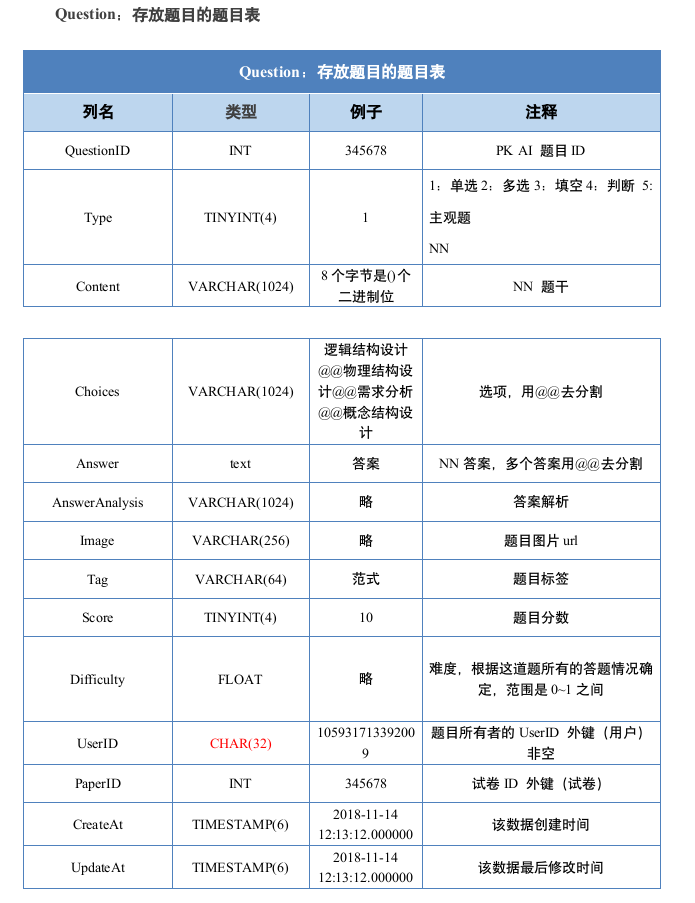


教师创建作业的流程为：

1. 录入试题
2. 筛选已经录入的试题，创建试卷
3. 选择试卷来创建作业
   1. 数据库设计
      1. E-R图



* + 1. 数据建表

1. 

# 第五章 系统实现

## 5.1 开发工具

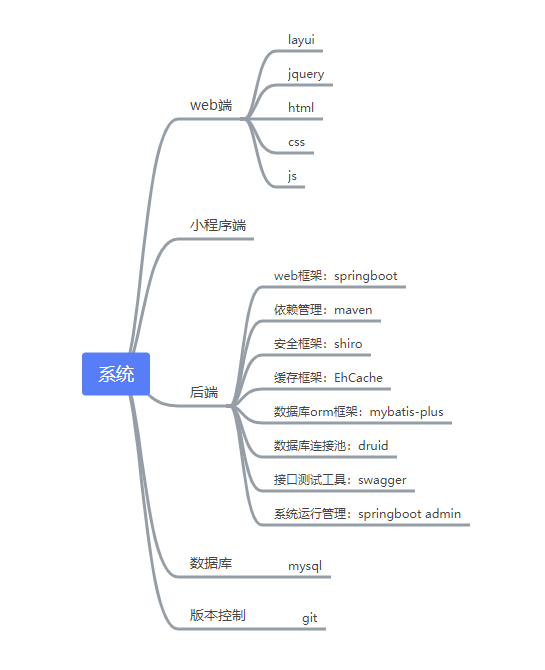
### 开发环境

系统： windows 10

IDE： IntelliJ IDEA

JDK: jdk1.8

### 开发框架



## 5.2 关键代码模块

### 5.2.1 课程管理模块

#### 接口类

public interface TeacherCourseService extends IService<TeacherCourse> {

// 教师选择一门课新建课程

void createTeacherCourse(TeacherCourse teacherCourse, String courseName, String name);

int createTeacherCourse(String courseName, String name, Date startDate, Date endDate, String position, String description);

int createTeacherCourse2(String userID, String courseName, Date startDate, Date endDate, String position, String description);

JSONObject CourseCreate(String userID, String courseName, Date startDate, Date endDate, String position,

String description);

// 根据CourseID和UserID查询该老师教授的课程，结果不一定唯一，因为一位老师可以开几个班的同一门课，按照TeacherCourseID查询的教授教授课程才是唯一的。

JSONObject viewCourseInfo(int courseID, String userID);

// 根据UserID得到该教师所教授的全部课程，并将其按照所需信息自定义Jason对象

JSONObject getTeacherCourseByUserID(String userID);

// 根据teacherCourseID查询某老师教授的某一课程

JSONObject getTeacherCourseByTeacherCourseID(int teacherCourseID);

// 更新教师课程信息

JSONObject editTeacherCourseInfo(int teacherCourseID, Date startDate, Date endDate, String position, String description,Integer courseID, String courseName);

// 根据teacherCourseID删除某老师教授的某一课程

int deleteTeacherCourseByTeacherCourseID(int teacherCourseID);

// 根据TeacherCourseID获取教授该课的教师信息

JSONObject getTchInfoByTeacherCourseID(int teacherCourseID);

List<String> getCourseByUserID(int userID);

//根据TeacherCourse添加课程

JSONObject insertTeacherCourse(TeacherCourse teacherCourse, String userID, String courseName);

}

#### 实现类

@Service

public class TeacherCourseServiceImpl extends ServiceImpl<TeacherCourseDao, TeacherCourse>

implements TeacherCourseService {

@Autowired

TeacherCourseDao teacherCourseDao;

@Autowired

UserDao userDao;

@Autowired

CourseListDao courseListDao;

@Override

public void createTeacherCourse(TeacherCourse teacherCourse, String courseName, String name) {

int courseID = teacherCourseDao.getCourseIDByName(courseName);

String userID = teacherCourseDao.getUserIDByName(name);

teacherCourseDao.createTeacherCourse1(teacherCourse, courseID, userID);

}

@Override

public int createTeacherCourse(String courseName, String name, Date startDate, Date endDate, String position,

String description) {

int courseID = teacherCourseDao.getCourseIDByName(courseName);

String userID = teacherCourseDao.getUserIDByName(name);

return teacherCourseDao.createTeacherCourse(courseID, userID, startDate, endDate, position, description);

}

@Override

public int createTeacherCourse2(String userID, String courseName, Date startDate, Date endDate, String position,

String description) {

int courseID = teacherCourseDao.getCourseIDByName(courseName);

return teacherCourseDao.createTeacherCourse(courseID, userID, startDate, endDate, position, description);

}

@Override

public JSONObject getTeacherCourseByTeacherCourseID(int teacherCourseID) {

// 得到UserID和CourseID

String userID = teacherCourseDao.selectById(teacherCourseID).getUserID();

int courseID = teacherCourseDao.selectById(teacherCourseID).getCourseID();

// 给出UserID，得到教师名字

String teacherName = userDao.selectById(userID).getName();

String avatar = userDao.selectById(userID).getAvatar();

// 根据CourseID得到课程名字

String courseName = courseListDao.selectById(courseID).getCourseName();

// 根据teacherCourseID得到TeacherCourse对象

TeacherCourse teacherCourse = teacherCourseDao.selectById(teacherCourseID);

// 构造我们需要的json对象

JSONObject jsonObject = new JSONObject();

JSONObject tempObject = new JSONObject();

tempObject.put("teacherName", teacherName);

tempObject.put("courseID",courseID);

tempObject.put("courseName", courseName);

tempObject.put("avatar", avatar);

tempObject.put("teacherCourseID", teacherCourse.getTeacherCourseID());

tempObject.put("position", teacherCourse.getPosition());

tempObject.put("startDate", DateFormatter.FormatDate(teacherCourse.getStartDate()));

tempObject.put("endDate", DateFormatter.FormatDate(teacherCourse.getEndDate()));

tempObject.put("description", teacherCourse.getDescription());

jsonObject.put("errcode", "0");

jsonObject.put("data", tempObject);// courseTeacherInfo

jsonObject.put("errmsg", "正常");

return jsonObject;

}

@Override

public JSONObject editTeacherCourseInfo(int teacherCourseID, Date startDate, Date endDate, String position,

String description,Integer courseID, String courseName) {

JSONObject jsonObject = new JSONObject();

try {

TeacherCourse teacherCourse = new TeacherCourse();

// teacherCourse.setTeacherCourseID();// set 子句，不需要set TeacherCourseID =

// 1,而是根据TeacherCourseID去update

teacherCourse.setCourseID(courseID);

teacherCourse.setStartDate(startDate);

teacherCourse.setEndDate(endDate);

teacherCourse.setPosition(position);

teacherCourse.setDescription(description);

LocalDateTime nowtime = LocalDateTime.now();

teacherCourse.setUpdateAt(nowtime);

//更新课程名称

CourseList courseList = courseListDao.selectById(courseID);

courseList.setCourseName(courseName);

courseListDao.updateById(courseList);

UpdateWrapper<TeacherCourse> teacherCourseUpdateWrapper = new UpdateWrapper<>();

teacherCourseUpdateWrapper.eq("teacherCourseID", teacherCourseID);

int res = teacherCourseDao.update(teacherCourse, teacherCourseUpdateWrapper);

// teacherCourseDao.updateById(teacherCourse);

jsonObject.put("errcode", "0");

jsonObject.put("data", res);

jsonObject.put("errmsg", "正常");

} catch (Exception e) {

jsonObject.put("errcode", "1");

jsonObject.put("errmsg", "更新课程信息失败！");

e.printStackTrace();

}

return jsonObject;

}

@Override

public JSONObject viewCourseInfo(int courseID, String userID) {

// 给出UserID，得到教师名字

String teacherName = userDao.selectById(userID).getName();

// 根据UserID得到TeacherCourse对象集合

QueryWrapper<TeacherCourse> queryWrapper = new QueryWrapper<>();

queryWrapper.eq("CourseID", courseID);

queryWrapper.eq("UserID", userID);

List<TeacherCourse> teacherCourseList = teacherCourseDao.selectList(queryWrapper);

int teacherCourseListLength = teacherCourseList.size();

// 根据CourseID得到CourseName

String[] courseNameList = new String[teacherCourseListLength];

for (int i = 0; i < teacherCourseListLength; i++) {

courseNameList[i] = courseListDao.selectById(courseID).getCourseName();

}

// 自定义json对象，json对象包含errcode和threads，threads对应于包含所需信息的json对象的json数组

JSONObject jsonObject = new JSONObject();

JSONArray courseJsonArray = new JSONArray();

for (int i = 0; i < teacherCourseListLength; i++) {

JSONObject tempObject = new JSONObject();

tempObject.put("teacherName", teacherName);

tempObject.put("courseName", courseNameList[i]);

tempObject.put("teacherCourseID", teacherCourseList.get(i).getTeacherCourseID());

tempObject.put("position", teacherCourseList.get(i).getPosition());

tempObject.put("startDate", DateFormatter.FormatDate(teacherCourseList.get(i).getStartDate()));

tempObject.put("endDate", DateFormatter.FormatDate(teacherCourseList.get(i).getEndDate()));

tempObject.put("description", teacherCourseList.get(i).getDescription());

courseJsonArray.add(tempObject);

}

jsonObject.put("errcode", "0");// addProperty方法用来添加原生数据类型

jsonObject.put("threads", courseJsonArray);// add方法用来添加JsonElement（包括JsonObject、JsonArray、JsonPrimitive和JsonNull）对象

return jsonObject;

}

// @Transactional

@Override

public JSONObject getTeacherCourseByUserID(String userID) {

// 自定义json对象，json对象包含errcode和threads，threads对应于包含所需信息的json对象的json数组

JSONObject jsonObject = new JSONObject();

JSONArray courseJsonArray = new JSONArray();

try {

// 给出UserID，得到教师名字

String teacherName = userDao.selectById(userID).getName();

String avatar = userDao.selectById(userID).getAvatar();

// 根据UserID得到TeacherCourse对象集合

QueryWrapper<TeacherCourse> queryWrapper = new QueryWrapper<>();

// IsDeleted为0表示没有被删除;按照创建时间降序排列，保证最新的在最前面

queryWrapper.eq("UserID", userID).eq("IsDeleted", 0).orderByDesc("CreateAt");

List<TeacherCourse> teacherCourseList = teacherCourseDao.selectList(queryWrapper);

int teacherCourseListLength = teacherCourseList.size();

// 根据CourseID得到CourseName

String[] courseNameList = new String[teacherCourseListLength];

for (int i = 0; i < teacherCourseListLength; i++) {

int courseID = teacherCourseList.get(i).getCourseID();

courseNameList[i] = courseListDao.selectById(courseID).getCourseName();

}

// 自定义json对象，json对象包含errcode和threads，threads对应于包含所需信息的json对象的json数组

// JSONObject jsonObject = new JSONObject();

// JSONArray courseJsonArray = new JSONArray();

for (int i = 0; i < teacherCourseListLength; i++) {

JSONObject tempObject = new JSONObject();

tempObject.put("teacherName", teacherName);

tempObject.put("avatar", avatar);

tempObject.put("courseName", courseNameList[i]);

tempObject.put("teacherCourseID", teacherCourseList.get(i).getTeacherCourseID());

tempObject.put("position", teacherCourseList.get(i).getPosition());

tempObject.put("description", teacherCourseList.get(i).getDescription());

tempObject.put("startDate", DateFormatter.FormatDate(teacherCourseList.get(i).getStartDate()));

tempObject.put("endDate", DateFormatter.FormatDate(teacherCourseList.get(i).getEndDate()));

courseJsonArray.add(tempObject);

}

jsonObject.put("errcode", "0");// addProperty方法用来添加原生数据类型

jsonObject.put("threads", courseJsonArray);// add方法用来添加JsonElement（包括JsonObject、JsonArray、JsonPrimitive和JsonNull）对象

jsonObject.put("errmsg", "正常");

} catch (Exception e) {

jsonObject.put("errcode", "1");// addProperty方法用来添加原生数据类型

jsonObject.put("errmsg", "查询教师教授信息错误");// 大部分用户都没有头像，null异常

System.out.println(e.toString());

}

return jsonObject;

}

@Override

public int deleteTeacherCourseByTeacherCourseID(int teacherCourseID) {

// result表示什么？result > 0表示删除成功

// int result = teacherCourseDao.deleteById(teacherCourseID);

// 软删除

TeacherCourse teacherCourse = new TeacherCourse();

teacherCourse.setIsDeleted(1);

LocalDateTime nowtime = LocalDateTime.now();

teacherCourse.setUpdateAt(nowtime);

UpdateWrapper<TeacherCourse> teacherCourseUpdateWrapper = new UpdateWrapper<>();

teacherCourseUpdateWrapper.eq("teacherCourseID", teacherCourseID);

int result = teacherCourseDao.update(teacherCourse, teacherCourseUpdateWrapper);

return result;

}

@Override

public JSONObject CourseCreate(String userID, String courseName, Date startDate, Date endDate, String position,

String description) {

JSONObject jsonObject = new JSONObject();

try {

int courseID = teacherCourseDao.getCourseIDByName(courseName);

int res = teacherCourseDao.createTeacherCourse(courseID, userID, startDate, endDate, position, description);

System.out.println("CourseCreate res: " + res);

jsonObject.put("errcode", "0");

jsonObject.put("data", res);

jsonObject.put("errmsg", "正常");

} catch (Exception e) {

jsonObject.put("errcode", "1");

jsonObject.put("errmsg", "课程创建失败！请检查课程信息！");

e.printStackTrace();

}

return jsonObject;

}

@Override

public JSONObject getTchInfoByTeacherCourseID(int teacherCourseID) {

JSONObject jsonObject = new JSONObject();

try {

// 得到UserID和CourseID

String userID = teacherCourseDao.selectById(teacherCourseID).getUserID();

// 给出UserID，得到教师名字

String teacherName = userDao.selectById(userID).getName();

String avatar = userDao.selectById(userID).getAvatar();

JSONObject tempObject = new JSONObject();

tempObject.put("userID", userID);

tempObject.put("teacherName", teacherName);

tempObject.put("avatar", avatar);

jsonObject.put("errcode", "0");

jsonObject.put("data", tempObject);

jsonObject.put("errmsg", "正常");

} catch (Exception e) {

jsonObject.put("errcode", "1");

jsonObject.put("errmsg", "获取教师信息失败！");

e.printStackTrace();

}

return jsonObject;

}

@Override

public List<String> getCourseByUserID(int userID){

return teacherCourseDao.selecCoursetById(userID);

}

@Override

public JSONObject insertTeacherCourse(TeacherCourse teacherCourse, String userID, String courseName) {

JSONObject jsonObject = new JSONObject();

try {

User user = userDao.getByID(userID);

CourseList courseList = new CourseList();

courseList.setCollegeID(user.getCollegeID());

courseList.setUniversityID(user.getUniversityID());

courseList.setCourseName(courseName);

//添加到课程列表

courseListDao.insert(courseList);

teacherCourse.setCourseID(courseList.getCourseID());

//添加到课程表中

int res = teacherCourseDao.insert(teacherCourse);

jsonObject.put("errcode", "0");

jsonObject.put("data", res);

jsonObject.put("errmsg", "正常");

} catch (Exception e) {

e.printStackTrace();

jsonObject.put("errcode", "1");

jsonObject.put("errmsg", "更新课程信息失败！");

}

return jsonObject;

}

}

### 5.2.2 作业管理模块

#### 接口类

public interface HomeworkService extends IService<Homework> {

// 新增作业

JSONObject homeworkCreate(String userID,int courseID, String homeworkName,String description, int gradeStyle, String startTime, String submitAt, String peerStart, String peerEnd, String feedbackStart, String feedbackEnd) ;

// 删除作业

JSONObject homeworkDelete(int homeworkID);

// 修改作业

JSONObject homeworkEdit(String homeworkName,String description, int gradeStyle, String startTime, String submitAt, int homeworkID, String userID, String peerStart, String peerEnd, String feedbackStart, String feedbackEnd);

// 通过作业ID查询作业

JSONObject homeworkGetByHomeworkID(int homeworkID);

// 通过课程ID查询作业

JSONObject homeworkGetByCourseID(int CourseID);

JSONObject homeworkGetbyhwName(String hwNameFliter, int courseID);

}

#### 实现类

@Service

public class HomeworkServiceImpl extends ServiceImpl<HomeworkDao, Homework> implements HomeworkService {

@Autowired

HomeworkDao homeworkDao;

@Autowired

UserDao userDao;

@Override

@Transactional

public JSONObject homeworkCreate(String userID,int courseID, String homeworkName,String description, int gradeStyle, String startTime, String submitAt, String peerStart, String peerEnd, String feedbackStart, String feedbackEnd) {

JSONObject jsonObject = new JSONObject();

try{

// 将String转为LocalDateTime

LocalDateTime \_startTime = DateFormatter.StrToLocalDateTime(startTime);

LocalDateTime \_submitAt = DateFormatter.StrToLocalDateTime(submitAt);

LocalDateTime \_peerStart = DateFormatter.StrToLocalDateTime(peerStart);

// 因手动分配作业的需要，作业提交时间与互评开始时间相差1小时

LocalDateTime \_peerStart\_new = \_peerStart.plusHours(1);

LocalDateTime \_peerEnd = DateFormatter.StrToLocalDateTime(peerEnd);

LocalDateTime \_feedbackStart = DateFormatter.StrToLocalDateTime(feedbackStart);

LocalDateTime \_feedbackEnd = DateFormatter.StrToLocalDateTime(feedbackEnd);

Homework homework = new Homework();

homework.setUserID(userID);

homework.setCourseID(courseID);

homework.setHomeworkName(homeworkName);

homework.setDescription(description);

homework.setGradeStyle(gradeStyle);

homework.setStartTime(\_startTime);

homework.setSubmitAt(\_submitAt);

homework.setPeerStart(\_peerStart\_new);

homework.setPeerEnd(\_peerEnd);

homework.setFeedbackStart(\_feedbackStart);

homework.setFeedbackEnd(\_feedbackEnd);

homework.setIsDeleted(0);

LocalDateTime nowtime = LocalDateTime.now();

homework.setCreateAt(nowtime);

homework.setUpdateAt(nowtime);

if(gradeStyle>0){

homeworkDao.homeworkCreate(homework);

}else{

homeworkDao.insert(homework);

}

jsonObject.put("errcode","0");

jsonObject.put("data",homework.getHomeworkID());

jsonObject.put("errmsg","新增作业成功！");

}catch (Exception e) {

jsonObject.put("errcode", "1");

jsonObject.put("errmsg", "新增作业失败！");

System.out.println(e.toString());

}

return jsonObject;

}

@Override

@Transactional

public JSONObject homeworkDelete(int homeworkID) {

JSONObject jsonObject = new JSONObject();

try{

Homework delHomework=new Homework();

delHomework.setHomeworkID(homeworkID);

delHomework.setIsDeleted(1);

LocalDateTime nowtime = LocalDateTime.now();

delHomework.setUpdateAt(nowtime);

int res = homeworkDao.updateById(delHomework);

jsonObject.put("errcode","0");

jsonObject.put("data",res);

jsonObject.put("errmsg","删除作业成功！");

}catch (Exception e){

jsonObject.put("errcode","1");

jsonObject.put("errmsg","删除作业失败！");

}

return jsonObject;

}

@Override

@Transactional

public JSONObject homeworkEdit(String homeworkName,String description, int gradeStyle, String startTime, String submitAt, int homeworkID, String userID, String peerStart, String peerEnd, String feedbackStart, String feedbackEnd) {

JSONObject jsonObject=new JSONObject();

try{

Homework homework = new Homework();

LocalDateTime \_startTime = DateFormatter.StrToLocalDateTime(startTime);

LocalDateTime \_submitAt = DateFormatter.StrToLocalDateTime(submitAt);

LocalDateTime \_peerStart = DateFormatter.StrToLocalDateTime(peerStart);

LocalDateTime \_peerEnd = DateFormatter.StrToLocalDateTime(peerEnd);

LocalDateTime \_feedbackStart = DateFormatter.StrToLocalDateTime(feedbackStart);

LocalDateTime \_feedbackEnd = DateFormatter.StrToLocalDateTime(feedbackEnd);

// homework.setUserID(userID);

// homework.setPaperID(paperID);

homework.setHomeworkName(homeworkName);

homework.setDescription(description);

homework.setGradeStyle(gradeStyle);

homework.setStartTime(\_startTime);

homework.setSubmitAt(\_submitAt);

homework.setPeerStart(\_peerStart);

homework.setPeerEnd(\_peerEnd);

homework.setFeedbackStart(\_feedbackStart);

homework.setFeedbackEnd(\_feedbackEnd);

LocalDateTime nowtime = LocalDateTime.now();

homework.setUpdateAt(nowtime);

// homework.setIsDeleted(0);

if(gradeStyle>0){

homeworkDao.editHomework(homework,homeworkID,userID);

}else{

homeworkDao.editHomework\_Peer(homework,homeworkID,userID);

}

jsonObject.put("errcode","0");

jsonObject.put("data",homework.getHomeworkID());

jsonObject.put("errmsg","修改作业成功！");

}catch (Exception e){

jsonObject.put("errcode","1");

jsonObject.put("errmsg","修改作业失败！");

}

return jsonObject;

}

@Override

@Transactional

public JSONObject homeworkGetByHomeworkID(int homeworkID) {

JSONObject jsonObject = new JSONObject();

JSONObject tempObject = new JSONObject();

try{

Homework homework = homeworkDao.homeworkGetByHomeworkID(homeworkID);

tempObject.put("homeworkID",homework.getHomeworkName());

tempObject.put("paperID",homework.getPaperID());

tempObject.put("homeworkName",homework.getHomeworkName());

tempObject.put("description",homework.getDescription());

tempObject.put("gradeStyle",homework.getGradeStyle());

tempObject.put("courseID",homework.getCourseID());

tempObject.put("startTime",DateFormatter.FormatLocalDateTime(homework.getStartTime()));

tempObject.put("submitAt",DateFormatter.FormatLocalDateTime(homework.getSubmitAt()));

tempObject.put("isDeleted",homework.getIsDeleted());

tempObject.put("createAt",DateFormatter.FormatLocalDateTime(homework.getCreateAt()));

tempObject.put("updateAt",DateFormatter.FormatLocalDateTime(homework.getUpdateAt()));

if(homework.getGradeStyle()==0){

tempObject.put("peerStart",DateFormatter.FormatLocalDateTime(homework.getPeerStart()));

tempObject.put("peerEnd",DateFormatter.FormatLocalDateTime(homework.getPeerEnd()));

tempObject.put("feedbackStart",DateFormatter.FormatLocalDateTime(homework.getFeedbackStart()));

tempObject.put("feedbackEnd",DateFormatter.FormatLocalDateTime(homework.getFeedbackEnd()));

}else {

tempObject.put("peerStart",null);

tempObject.put("peerEnd",null);

tempObject.put("feedbackStart",null);

tempObject.put("feedbackEnd",null);

}

jsonObject.put("errcode","0");

jsonObject.put("data",tempObject);

jsonObject.put("errmsg","查询成功！");

}catch (Exception e){

jsonObject.put("errcode","1");

jsonObject.put("errmsg","查询失败！");

}

return jsonObject;

}

@Override

public JSONObject homeworkGetByCourseID(int courseID) {

JSONObject jsonObject = new JSONObject();

JSONArray homeworkJsonArray = new JSONArray();

try{

List<Homework> homeworkList = homeworkDao.homeworkGetByCourseID(courseID);

for (int i = 0; i < homeworkList.size(); i++) {

JSONObject tempObject = new JSONObject();

tempObject.put("homeworkID", homeworkList.get(i).getHomeworkID());

tempObject.put("courseID", homeworkList.get(i).getCourseID());

tempObject.put("description", homeworkList.get(i).getDescription());

tempObject.put("gradeStyle", homeworkList.get(i).getGradeStyle());

tempObject.put("homeworkName", homeworkList.get(i).getHomeworkName());

tempObject.put("userID",homeworkList.get(i).getUserID());

tempObject.put("paperID",homeworkList.get(i).getPaperID());

tempObject.put("isDeleted",homeworkList.get(i).getIsDeleted());

tempObject.put("startTime", DateFormatter.FormatLocalDateTime(homeworkList.get(i).getStartTime()));

tempObject.put("submitAt", DateFormatter.FormatLocalDateTime(homeworkList.get(i).getSubmitAt()));

tempObject.put("createAt", DateFormatter.FormatLocalDateTime(homeworkList.get(i).getCreateAt()));

tempObject.put("updatedAt", DateFormatter.FormatLocalDateTime(homeworkList.get(i).getUpdateAt()));

if(homeworkList.get(i).getGradeStyle()==0){

tempObject.put("peerStart", DateFormatter.FormatLocalDateTime(homeworkList.get(i).getPeerStart()));

tempObject.put("peerEnd", DateFormatter.FormatLocalDateTime(homeworkList.get(i).getPeerEnd()));

tempObject.put("feedbackStart", DateFormatter.FormatLocalDateTime(homeworkList.get(i).getFeedbackStart()));

tempObject.put("feedbackEnd", DateFormatter.FormatLocalDateTime(homeworkList.get(i).getFeedbackEnd()));

}else {

tempObject.put("peerStart",null);

tempObject.put("peerEnd",null);

tempObject.put("feedbackStart",null);

tempObject.put("feedbackEnd",null);

}

homeworkJsonArray.add(tempObject);

}

jsonObject.put("errcode", "0");// addProperty方法用来添加原生数据类型

jsonObject.put("data", homeworkJsonArray);// add方法用来添加JsonElement（包括JsonObject、JsonArray、JsonPrimitive和JsonNull）对象

jsonObject.put("errmsg","查询教师该课程下所有作业成功！");

}catch (Exception e){

jsonObject.put("errcode","1");

jsonObject.put("errmsg","查询教师该课程下所有作业失败！");

}

return jsonObject;

}

@Override

public JSONObject homeworkGetbyhwName(String hwNameFliter, int courseID){

JSONObject jsonObject = new JSONObject();

JSONArray homeworkJsonArray = new JSONArray();

try{

List<Homework> homeworkList = homeworkDao.homeworkGetbyhwName(hwNameFliter,courseID);

for (int i = 0; i < homeworkList.size(); i++) {

JSONObject tempObject = new JSONObject();

tempObject.put("homeworkID", homeworkList.get(i).getHomeworkID());

tempObject.put("courseID", homeworkList.get(i).getCourseID());

tempObject.put("description", homeworkList.get(i).getDescription());

tempObject.put("gradeStyle", homeworkList.get(i).getGradeStyle());

tempObject.put("homeworkName", homeworkList.get(i).getHomeworkName());

tempObject.put("userID",homeworkList.get(i).getUserID());

tempObject.put("paperID",homeworkList.get(i).getPaperID());

tempObject.put("isDeleted",homeworkList.get(i).getIsDeleted());

tempObject.put("startTime", DateFormatter.FormatLocalDateTime(homeworkList.get(i).getStartTime()));

tempObject.put("submitAt", DateFormatter.FormatLocalDateTime(homeworkList.get(i).getSubmitAt()));

tempObject.put("createAt", DateFormatter.FormatLocalDateTime(homeworkList.get(i).getCreateAt()));

tempObject.put("updatedAt", DateFormatter.FormatLocalDateTime(homeworkList.get(i).getUpdateAt()));

if(homeworkList.get(i).getGradeStyle()==0){

tempObject.put("peerStart", DateFormatter.FormatLocalDateTime(homeworkList.get(i).getPeerStart()));

tempObject.put("peerEnd", DateFormatter.FormatLocalDateTime(homeworkList.get(i).getPeerEnd()));

tempObject.put("feedbackStart", DateFormatter.FormatLocalDateTime(homeworkList.get(i).getFeedbackStart()));

tempObject.put("feedbackEnd", DateFormatter.FormatLocalDateTime(homeworkList.get(i).getFeedbackEnd()));

}else {

tempObject.put("peerStart",null);

tempObject.put("peerEnd",null);

tempObject.put("feedbackStart",null);

tempObject.put("feedbackEnd",null);

}

homeworkJsonArray.add(tempObject);

}

jsonObject.put("errcode", "0");// addProperty方法用来添加原生数据类型

jsonObject.put("data", homeworkJsonArray);// add方法用来添加JsonElement（包括JsonObject、JsonArray、JsonPrimitive和JsonNull）对象

jsonObject.put("errmsg","模糊查询作业成功！");

}catch (Exception e){

jsonObject.put("errcode","1");

jsonObject.put("errmsg","模糊查询作业失败！");

}

return jsonObject;

}

}

# 第六章 系统测试

## 6.1 测试项目说明

### 数据与数据库完整性测试

1、测试名称：数据的完整性测试

2、测试内容：界面中提供了对课程表的插入、修改、删除，题目表的插入、修改、删除，试卷表的插入、修改、删除，作业表的插入、修改、删除，用户表的插入，在测试时，通过对已有的接口测试

* 课程表的插入：
* 输入： 三个属性值（课程名称、课程时间、课程描述）
* 测试步骤：在新建课程界面输入课程名称、课程时间、课程描述，点击新建课程。
* 预期结果：跳转至查看课程界面，并显示添加成功课程
* 课程表的修改：
* 输入： 可选填的三个属性值（课程名称、课程时间、课程描述）
* 测试步骤：在编辑课程界面输入课程名称、课程时间、课程描述，点击编辑
* 预期结果： 提示编辑成功
* 课程表的删除：
* 输入： 无
* 测试步骤：在需要删除的课程右下角点击删除按钮，点击确定删除
* 预期结果：提示删除成功
* 题目表的插入：
* 输入：每种题目都有：题干，题目解析。根据题目类型输入信息：单选题给出4个选项并给出一个正确答案；多选题给出4个选项并给出多个正确答案；填空题、主观题给出若干个正确答案。
* 测试步骤：新建题目界面输入题干、答案、题目解析，点击创建。
* 预期结果：创建成功并在查看试题界面找到新题目
* 题目表的修改：
* 输入：需要修改的参数，题干、答案、题目解析。
* 测试步骤：输入修改内容，点击编辑按钮。
* 预期结果：题目修改成功并显示修改内容。
* 题目表的删除：
* 输入：无
* 测试步骤：在显示题目页面点击删除即可
* 预期结果：提示删除成功
* 试卷表的插入：
* 输入：试卷的名称及描述，组成试卷的习题及分值
* 测试步骤：点击新建试卷，输入试卷的名称及描述，并点击左侧界面选择习题添加至试卷中，右侧界面可以为题目修改分值，右下角按钮点击保存试卷。
* 预期结果：提示创建试卷成功
* 试卷表的修改：
* 输入：试卷的名称及描述
* 测试步骤：点击编辑试卷按钮，修改试卷的名称及描述
* 预期结果：试卷修改成功。
* 试卷表的删除：
* 输入：无
* 测试步骤：选择试卷后点击删除按钮
* 预期结果：提示删除试卷成功
* 作业表的插入：
* 输入：作业名称，起止时间，所需试卷。可选互评方式，互评起止时间
* 测试步骤：点击新建作业，输入作业名称，起止时间，所需试卷，这里互评方式作业需要的内容包含了普通作业，选择互评模式，输入互评起止时间。
* 预期结果：作业创建成功
* 作业表的删除：
* 输入：无
* 测试步骤：选择作业的删除按钮，确认删除
* 预期结果：作业删除成功
* 用户表的插入：
* 输入：用户的姓名、性别、所属大学、学工号。
* 测试步骤：管理员账号可以创建教师，需要输入教师的学工号、姓名、所属大学；教师账号可以选择相应的课程来添加学生账号、学工号，可通过EXCEL表格导入
* 预期结果：创建教师、学生账号，如果该学生账号已经被其他老师添加，就不会创建学生账号，只会将该学生的账号关联到课程。

### 功能测试

* 登录模块：
* 输入账号格式不正确，直接提示账号错误，
* 输入账号不正确或密码不正确，提示错误，
* 课程模块：
* 登录教师账号、学生账号界面，显示所有课程，点击相应课程进入课程操作界面
* 教师账号可以创建新的课程并修改删除已有的课程，学生只能查看
* 点击创建按钮成功进入创建页面，输入信息后点击确定成功创建
* 修改按钮可以成功进入修改页面，修改信息之后显示正常
* 点击删除按钮之后成功删除课程
* 试题模块：
* 教师账号在我的题库中可以成功查看自己的所有试题
* 教师账号创建试题成功之后，可以在我的题库中成功查看到此题
* 试卷模块：
* 教师账号在我的试卷中可以查看到之前所有的创建过的试卷
* 教师账号在新建试卷之后可以，在我的试卷中查看到已经创建过的试卷
* 管理员模块：
* 管理员可以查看所有的教师信息，并且可以自由添加教师信息

### 性能测试

核实所指定的事务或业务功能在以下情况下的性能下行为：

* 正常的预期工作量
* 预期的最繁重工作量

输入：

（1）通过修改数据文件增加数据库数据数量。

（2）同时登录学端、教师端、管理员端账号。

（3）同时登录多个教师端或管理员端账号。

测试步骤：

（1）修改数据文件来增加数据库中数据数量，测试各个界面显示及后台处理是否正确或性能开始下降。

（2）同时打开三个页面：学生、教师、管理员各一个。测试在三个端口同时操作时是否发生冲突或性能下降。

（3）同时登录多个教师端账号和多个学生端账号，测试同时操作时是否发生冲突或性能下降。

预期结果：

（1）每个界面在增加数据量后在所预期的时间内完成显示或操作，且不发生故障。

（2）多个用户登录时，在可接受的时间范围内成功完成测试且不发生冲突。

## 6.2 缺陷分析

|  |  |  |  |
| --- | --- | --- | --- |
| 缺陷报告0001 | | |  |
|  |
|  |
|  |
| 缺陷记录 | | 编号：0001 |  |
| 软件名称：互评管理系统 | 模块名称：试题 | 版本号：2000 |  |
|  |
| 严重程度：次要 | 优先级：低 | 状态：新建 |  |
| 测试人员：杨攀原 | | |  |
| 日期：2021-7-3 | | |  |
| 硬件平台 | | 软件平台 |  |
| PC： | | 操作系统： |  |
| CPU：Intel(R) Core(TM)i5-7200U 2.50GHz | | Windows 10 |  |
| RAM：8.00GB | |  |
|  | | |  |
| 缺陷概述：创建选择题时，无法修改选项个数 | | |  |
| 详细描述 | | |  |
| 1、登录教师账号，进入“我的题库”，点击“新建试题” | | |  |
| 2、选择创建选择题，出现四个选项，无法对选项的个数进行修改，说明题目功能没能很好的实现。 | | |  |
|  | | |  |
|  | | |  |

|  |  |  |
| --- | --- | --- |
| 缺陷报告0002 | | |
|
|
|
| 缺陷记录 | | 编号：0001 |
| 软件名称：互评管理系统 | 模块名称：试卷 | 版本号：2000 |
|
| 严重程度：次要 | 优先级：低 | 状态：新建 |
| 测试人员：杨攀原 | | |
| 日期：2021-7-3 | | |
| 硬件平台 | | 软件平台 |
| PC： | | 操作系统： |
| CPU：Intel(R) Core(TM)i5-7200U 2.50GHz | | Windows 10 |
| RAM：8.00GB | |
|  | | |
| 缺陷概述：修改试卷时，无法改正试卷的题目顺序和分值 | | |
| 详细描述 | | |
| 1、登录教师账号，进入“我的试卷”，选择任意试卷，点击编辑试卷 | | |
| 2、只能编辑试卷的名称和描述，无法修改试卷的题目个数和分值 | | |
|  | | |
|  | | |

# 第七章 用户手册

网址：https://www.yuque.com/books/share/9be16530-b7f0-4f58-bb9a-b8b2f6a23196?#