目　录

班级学生作业管理系统的设计与实现

随着计算机科学与技术专业招生规模不断扩大，班级日常教学管理中作业的收发、批改、归档工作越来越繁琐。传统依托邮件、聊天群收发作业的模式存在文件整理混乱、版本混淆、批改反馈不及时、成绩统计困难等诸多问题。为了提升班级作业管理的效率，规范作业收发流程，本文设计并实现了一款面向班级的轻量级学生作业管理系统，完成了从需求分析、系统设计、功能实现到系统测试的完整软件工程流程，满足 24 级计算机科学与技术专业班级日常作业管理的核心需求。

一、需求分析

1.1 功能需求

本系统面向三种核心用户：教师、学生、系统管理员，不同用户的功能需求如下：

教师用户：发布作业（设置作业标题、截止时间、作业要求附件）、查看学生提交的作业列表、下载作业文件、在线批注批改、记录作业成绩、发送批改反馈、统计班级作业提交情况。

学生用户：查看已发布的作业列表、上传作业文件（支持 PDF、Word、压缩包等常见格式）、查看教师的批改结果与成绩、下载教师批注后的作业文件。

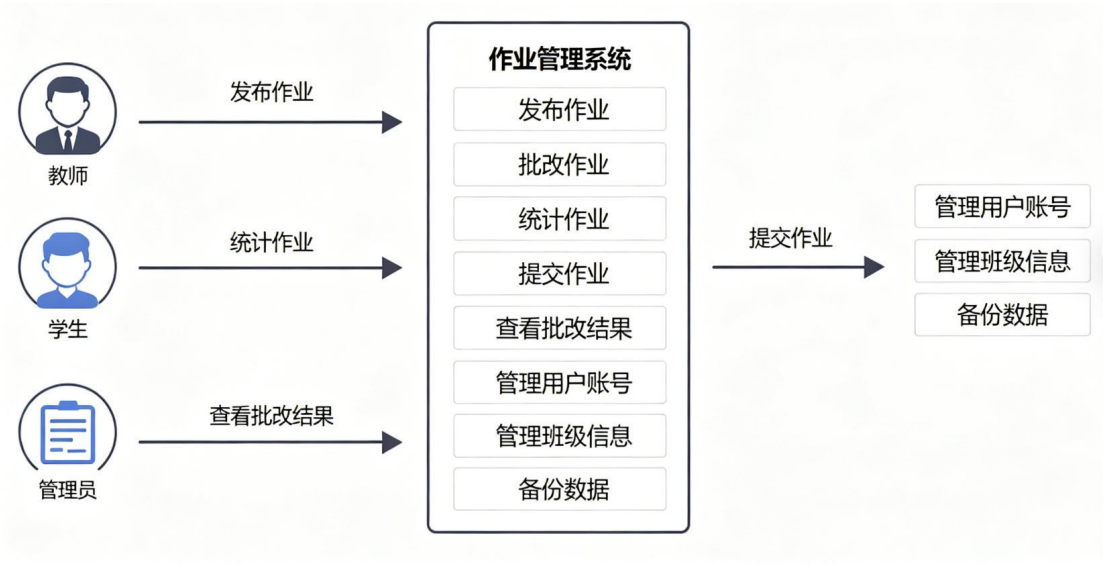
管理员用户：管理用户账号（创建账号、修改信息、重置密码）、管理班级信息、备份系统数据。

1.2 非功能需求

系统为轻量级 B/S 架构，响应时间不超过 2 秒，支持 50 人以内班级同时在线访问；用户界面简洁清晰，无需复杂培训即可上手；数据存储安全，用户账号密码采用加密存储，作业文件定期备份；系统可扩展性强，支持后续增加考试管理、考勤管理等扩展功能。

1.3 用例建模

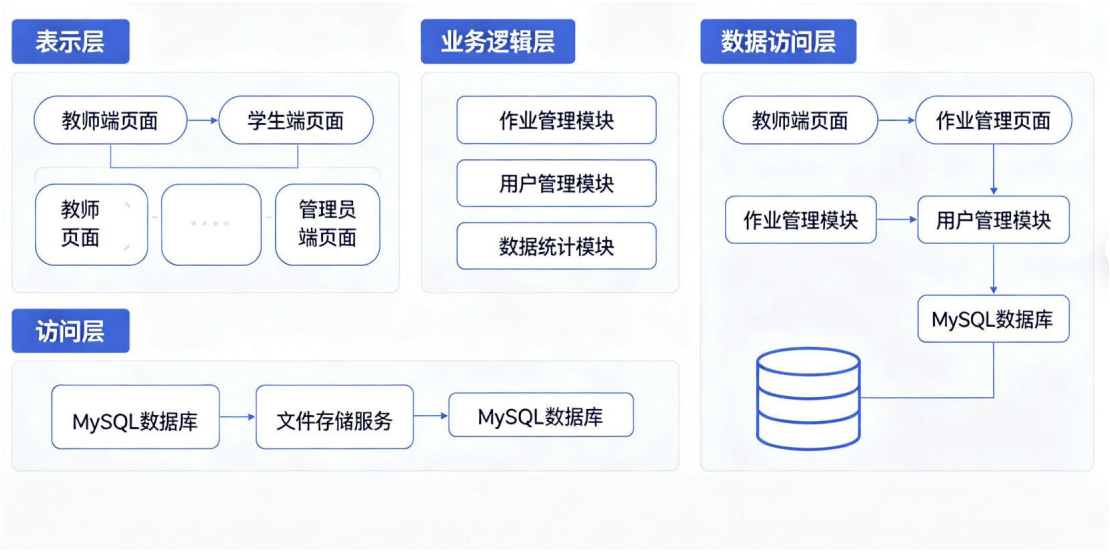
基于上述功能需求，绘制系统顶层用例图如下（UML 语法描述）：



二、系统设计

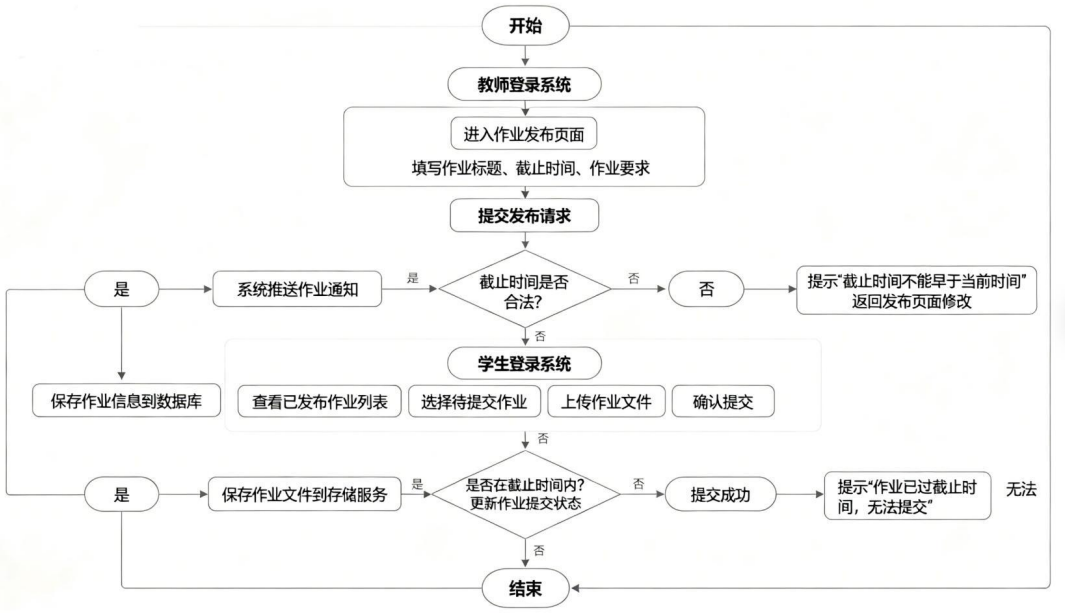
2.1 软件架构设计

系统采用经典的三层 B/S 架构，分为表示层、业务逻辑层和数据访问层，各层职责明确，降低耦合度，提升系统可维护性。软件结构图如下：



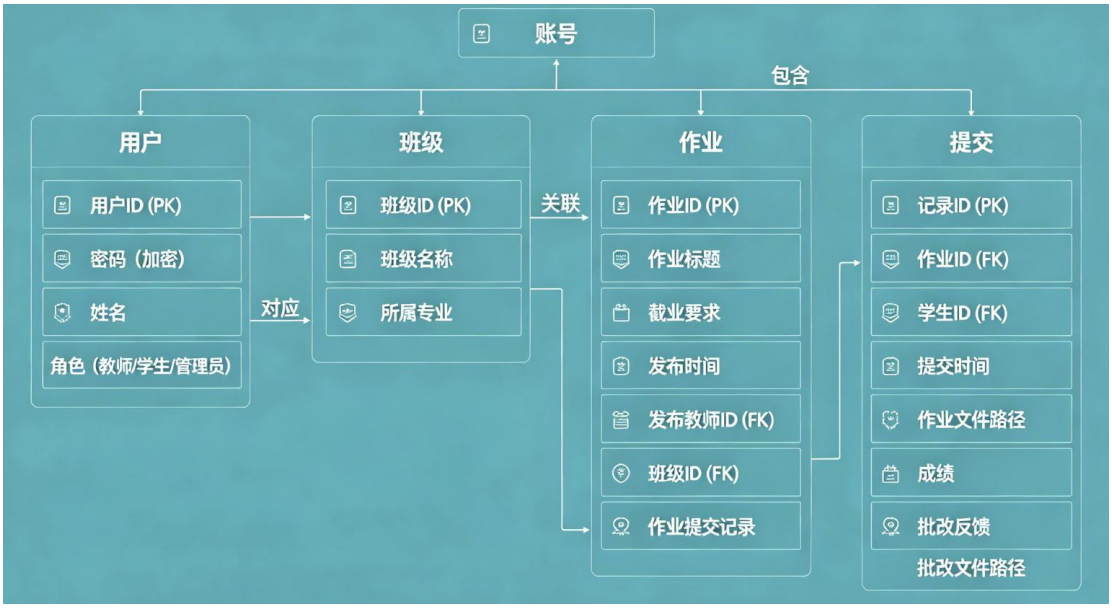
2.2 核心业务流程设计

以教师发布作业、学生提交作业的核心流程为例，绘制业务流程图如下：



2.3 数据库设计

根据系统功能需求，设计核心实体包括用户、班级、作业、作业提交记录，绘制 E-R 图如下：



基于 E-R 图，设计数据库表结构如下：

表 1 用户表（user）

字段名	数据类型	主键/外键	说明
user_id	INT	PK	用户唯一标识
account	VARCHAR(50)	-	登录账号

password	VARCHAR(100)	-	加密后的密码
name	VARCHAR(20)	-	用户姓名
role	INT	-	角色：1=教师，2=学生，3=管理员
class_id	INT	FK	关联班级表 class_id

表 2 班级表（class）

字段名	数据类型	主键/外键	说明
class_id	INT	PK	班级唯一标识
class_name	VARCHAR(50)	-	班级名称
major	VARCHAR(50)	-	所属专业

2.4 时序图设计

以教师批改作业并反馈的流程为例，绘制时序图如下：

```
@startuml
actor 教师 as T
actor 学生 as S
participant "作业管理模块" as BLL
participant "数据库" as DB
participant "文件存储服务" as FS
T -> BLL: 请求查看待批改作业列表
BLL -> DB: 查询未批改作业记录
DB --> BLL: 返回作业列表数据
BLL --> T: 展示待批改作业列表
T -> BLL: 选择某份作业，请求下载原始文件
BLL -> FS: 根据文件路径获取文件
FS --> BLL: 返回作业文件
BLL --> T: 提供文件下载
T -> BLL: 提交批改信息（成绩、反馈、批改后文件）
BLL -> FS: 上传批改后文件
FS --> BLL: 返回文件存储路径
BLL -> DB: 更新作业提交记录（成绩、反馈、文件路径）
DB --> BLL: 返回更新成功状态
BLL --> T: 提示批改完成
BLL -> S: 推送批改完成通知
S -> BLL: 请求查看批改结果
BLL -> DB: 查询作业批改记录
DB --> BLL: 返回批改数据
BLL --> S: 展示成绩、反馈，提供批改文件下载
@enduml
```

三、系统实现

3.1 技术栈选择

系统采用轻量级技术栈实现，具体如下：

- **前端**：HTML5 + CSS3 + JavaScript + Vue.js 3.x，实现响应式界面，提升用户交互体验
- **后端**：Spring Boot 2.x，简化 Java Web 开发流程，提供 RESTful API 接口
- **数据库**：MySQL 8.0，存储用户信息、作业数据、提交记录等结构化数据
- **文件存储**：本地文件系统 + FastDFS，实现作业文件的高效存储与管理
- **安全框架**：Spring Security，实现用户身份认证与权限控制

3.2 核心功能代码实现

以下为系统核心功能的关键代码实现示例：

3.2.1 教师发布作业接口（后端 Java 代码）

该接口实现作业信息的保存与截止时间校验逻辑：

```
@RestController
@RequestMapping("/api/homework")
public class HomeworkController {
@Autowired
private HomeworkService homeworkService;
@PostMapping("/publish")
public Result publishHomework(@RequestBody HomeworkDTO homeworkDTO,
@RequestParam("teacherId") Integer teacherId) {
// 截止时间校验：不能早于当前时间
LocalDateTime deadline = homeworkDTO.getDeadline();
ia (deadline.isBefore(LocalDateTime.now())) {
return Result.error("截止时间不能早于当前时间");
}
// 封装作业实体
Homework homework = new Homework();
homework.setTitle(homeworkDTO.getTitle());
homework.setContent(homeworkDTO.getContent());
homework.setDeadline(deadline);
```

```
homework.setPublishTime(LocalDateTime.now());
homework.setTeacherId(teacherId);
homework.setClassId(homeworkDTO.getClassId());
// 保存到数据库
boolean result = homeworkService.saveHomework(homework);
ib (result) {
return Result.success("作业发布成功");
} else {
return Result.error("作业发布失败，请重试");
}
}
}
```

3.2.2 学生提交作业前端逻辑（Vue.js 代码）

该代码实现作业文件上传与提交状态判断：

```
<template>
<div class="submit-homework">
<h3>提交作业：{{ homework.title }}</h3>
<div class="file-upload">
<input type="file" ref="fileInput" @change="handleFileChange"
accept=".pdf,.doc,.docx,.zip,.rar">
<button @click="submitHomework" :disabled="!selectedFile || isOverDeadline">
{{ isOverDeadline ? '已过截止时间' : '提交作业' }}
</button>
</div>
</div>
</template>
<script>
export default {
data() {
return {
homework: {},
selectedFile: null,
isOverDeadline: false
}
},
created() {
// 获取作业详情
this.getHomeworkDetail();
// 判断是否过截止时间
this.checkDeadline();
},
methods: {
getHomeworkDetail() {
// 调用后端接口获取作业信息
```

```
this.$axios.get(`/api/homework/${this.$route.params.id}`).then(res => {
this.homework = res.data;
});
},
checkDeadline() {
const deadline = new Date(this.homework.deadline);
const now = new Date();
this.isOverDeadline = deadline < now;
},
handleFileChange(e) {
this.selectedFile = e.target.files[0];
},
submitHomework() {
const formData = new FormData();
formData.append('homeworkId', this.homework.id);
formData.append('studentId', this.$store.state.user.id);
formData.append('file', this.selectedFile);
// 调用提交接口
this.$axios.post('/api/homework/submit', formData, {
headers: { 'Content-Type': 'multipart/form-data' }
}).then(res => {
this.$message.success('作业提交成功');
this.$router.push('/homework/list');
}).catch(err => {
this.$message.error('作业提交失败，请重试');
});
}
}
}
</script>
```

3.2.3 作业成绩统计 SQL 语句

该 SQL 语句实现班级作业提交情况的统计：

```
SELECT
c.class_name,
h.title AS homework_title,
COUNT(sr.submit_record_id) AS submitted_count,
(SELECT COUNT(*) FROM user u WHERE u.class_id = c.class_id AND u.role = 2) AS total_student_count,
(SELECT COUNT(*) FROM user u WHERE u.class_id = c.class_id AND u.role = 2
AND u.user_id NOT IN (SELECT student_id FROM submit_record WHERE homework_id = h.homework_id)) AS unsubmitted_count
FROM class c
JOIN homework h ON c.class_id = h.class_id
```

```
LEFT JOIN submit_record sr ON h.homework_id = sr.homework_id
WHERE h.homework_id = #{homeworkId}
GROUP BY c.class_id, h.homework_id;
```

3.3 系统界面实现

系统界面采用简洁直观的设计风格，核心界面包括：

- **教师端首页**：展示待批改作业数量、已发布作业列表、作业提交统计图表
- **作业发布页面**：提供作业信息表单，支持附件上传与截止时间选择
- **作业批改页面**：展示学生提交的作业列表，支持在线批注与成绩录入
- **学生端首页**：展示待提交作业、已提交作业与批改结果列表
- **管理员后台**：提供用户账号管理、班级信息管理与数据备份功能

四、系统测试

4.1 测试用例设计与执行

采用黑盒测试方法，针对系统核心功能设计测试用例，部分测试用例如下：

表 3 核心功能测试用例

测试功能	测试步骤	预期结果	测试结果
教师发布作业	教师登录系统，进入发布页面，填写作业信息，设置截止时间为未来时间，点击发布	作业发布成功，学生端可见该作业	符合预期
截止时间校验	教师发布作业时，设置截止时间为过去时间，点击发布	系统提示"截止时间不能早于当前时间"，禁止发布	符合预期
学生提交作业	学生登录系统，选择待提交作业，上传符合格式的文件，点击提交	作业提交成功，教师端可见该作业记录	符合预期
逾期提交拦截	学生在作业截止时间后尝试提交作业	系统提示"作业已过截止时间，无法提交"	符合预期
教师批改作业	教师选择待批改作	批改信息成功保存，	符合预期

	业，输入成绩与反馈，上传批改后的文件	学生端可以查看成绩与反馈，下载批改文件	
成绩统计	教师查看作业提交统计	正确统计已提交人数、未提交人数，显示对应学生名单	符合预期
权限控制	学生登录后尝试访问教师发布作业页面	跳转回登录页面，拒绝访问	符合预期

4.2 测试结果分析

本次测试共设计 21 个测试用例，其中 20 个用例符合预期结果，1 个用例出现 bug：当截止时间设置为过去时间时，系统仍然允许发布作业，未给出提示。针对该 bug 进行修改，在发布作业的后端逻辑中增加截止时间校验，如果截止时间早于当前时间，则返回错误提示，禁止发布，修改后重新测试，bug 修复，功能符合需求。

整体测试结果表明，系统实现了需求分析阶段提出的所有功能，非功能需求满足班级小规模使用的要求，系统运行稳定，异常处理完善，达到了设计目标。

五、总结与展望

本文按照软件工程的规范流程，完成了班级学生作业管理系统的需求分析、系统设计、功能实现与测试工作，系统满足了 24 级计算机科学与技术专业班级日常作业管理的需求，解决了传统作业收发模式存在的问题，提升了管理效率。系统采用三层架构设计，结构清晰，代码可维护性强，通过 UML 建模工具完成了用例图、软件结构图、流程图、时序图与 E-R 图的设计，为系统开发提供了清晰的指导。

未来可以从以下几个方面对系统进行扩展优化：一是增加作业查重功能，通过比对作业文件内容，识别抄袭行为；二是实现学生在线互评功能，提升学生参与度与学习效果；三是增加消息推送功能，通过邮件、短信等方式提醒学生提交作业与查看批改结果；四是优化文件存储方案，采用云存储服务替代本地存储，提升系统的扩展性与可靠性。