

数据库课程设计

一. 项目简介

1.1 系统结构

1. Web 表现层
2. 业务逻辑层
3. 数据访问层

1.2 开发语言

Java

1.3 开发工具

IntelliJ IDEA, JDK 17.0.1

1.4 数据库

MySQL Server 8.0

1.5 操作系统

Microsoft Windows 10

1.6 技术栈

MyBatis, HTML+CSS+JavaScript, Bootstrap, Servlet, Tomcat, Axios, Cookie, Session, Filter, Vue, ElementUI, MySQL

二. 数据库规划

2.1 任务陈述

作业管理系统是一个帮助教师管理学生作业的系统, 学校工作繁杂, 资料众多, 特别对于学生作业管理来说, 免不了使用计算机, 一个完善的学生作业管理系统, 可以帮助教师能够更加轻松的工作, 摆脱纸质管理的麻烦, 为教师和学生减轻作业管理上的负担.

该作业管理系统共有三个角色: 系统管理员, 教师和学生. 管理员可以在管理员界面实现对学生信息, 教师信息, 课程信息, 班级信息的增删改查. 教师可以在教师界面实现对作业题目的发布, 修改和删除功能, 对学生上传答案的打分功能, 对学生成绩的查看统计功能. 学生可以在学生界面实现对作业题目的查看, 上交作业答案, 修改作业答案, 查看得分等功能

2.2 任务目标

教师信息管理	教师登录, 对教师信息的增, 删, 改, 查
班级信息管理	对班级信息的增, 删, 改, 查
学生信息管理	对学生信息的增, 删, 改, 查
课程信息管理	对课程信息的增, 删, 改, 查
习题信息管理	对习题信息的增, 删, 改, 查
作业信息管理	对教师布置作业的增, 删, 改, 查
批改作业模块	对所有信息的查询, 对学生作业的增, 删, 改, 查, 批改学生作业
学生作业管理	查看作业当前作业并完成作业, 查看已完成的作业, 修改个人信息

三. 系统定义

3.1 系统边界

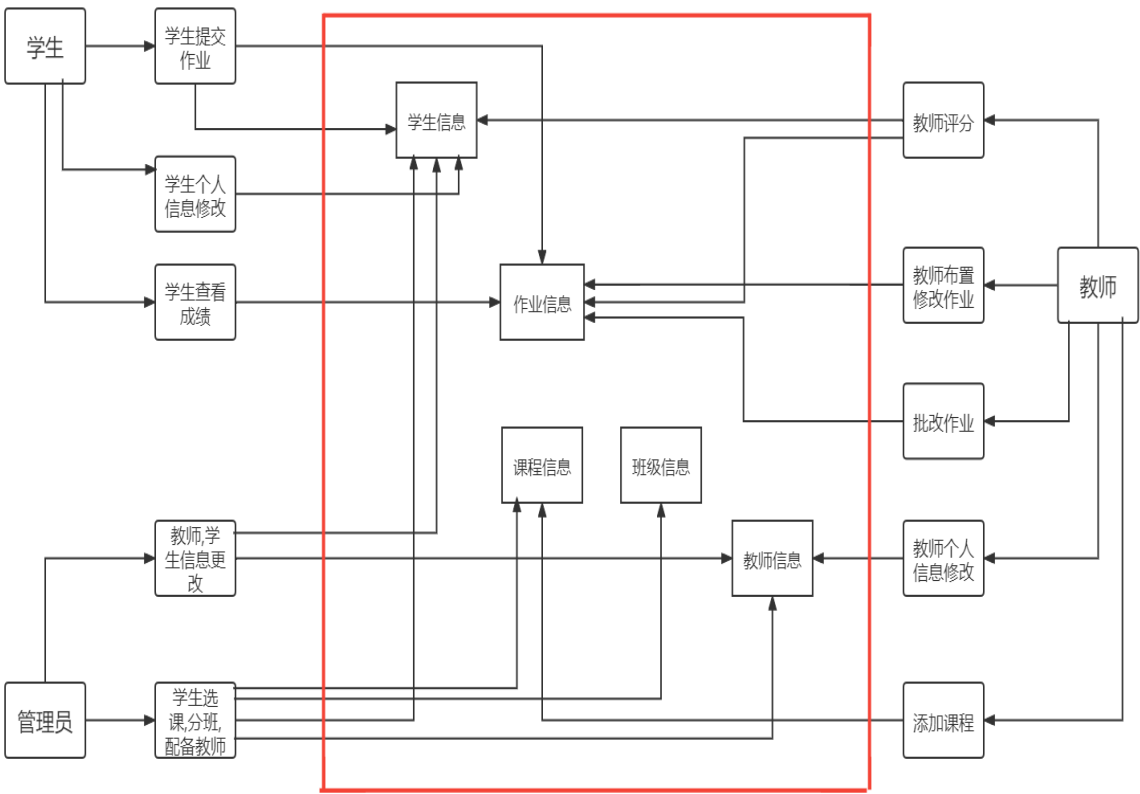


Figure 1: 系统边界

3.2 用户视图

管理员视图:

1. 用户信息管理

教师管理: 查询, 添加, 修改, 删除教师信息, 重置密码

学生管理: 查询, 添加, 修改, 删除学生信息, 重置密码

2. 班级管理信息

班级管理: 查询, 添加, 修改, 删除班级信息

3. 课程信息管理

课程管理: 查询, 添加, 修改, 删除课程信息

4. 习题库

习题管理: 查询, 添加, 修改, 删除习题信息

5. 作业表

作业内容: 查询, 添加, 修改, 删除作业信息

批改作业: 批改作业并打分

教师视图:

1. 个人信息管理

个人信息管理: 修改, 更新个人信息

2. 作业管理

习题管理: 习题库增, 删, 改, 查

布置作业: 作业单增删改查

学生作业: 查看每个学生作业完成情况并批改打分

学生视图:

1. 个人信息管理

个人信息管理: 修改, 更新个人信息

2. 作业管理

查看作业: 查看当前作业

四. 需求分析

4.1 用户需求说明

4.1.1 数据需求

1. 教师基本信息:

教师 id, 姓名, 密码, 教师权限

2. 学生基本信息:

学生 id, 姓名, 班级, 密码

3. 班级信息:

班级 id, 班级名

4. 课程表:

课程 id, 课程名

5. 题库表:

习题 id, 习题类型, 题目内容, 习题分值, 习题答案

6. 作业表:

作业 id, 作业内容, 作业目前属性, 作业发布日期, 作业截止日期

7. 学生作业表:

学生 id, 作业 id

8. 作业习题联系表:
作业 id, 习题 id

4.1.2 事务需求

4.1.2.1 数据录入

1. 教师数据录入:

教师 id, 姓名, 账户名, 密码

2. 学生信息录入:

学生 id, 姓名, 账户名, 密码

3. 课程信息录入:

课程 id, 课程名

4. 题库信息录入:

习题 id, 习题题目, 类型, 答案, 科目

4.1.2.2 数据查询

1. 教师信息查询:

教师 id, 姓名

2. 学生信息查询:

学生 id, 姓名

3. 课程信息查询:

课程 id, 姓名

4. 习题信息查询:

习题 id, 习题科目

5. 作业信息查询:

作业 id, 作业内容, 作业班级

4.1.2.3 数据更新

1. 教师信息更改

2. 学生信息更改

3. 课程信息更改

4. 习题信息更改

5. 作业信息更改

6. 布置作业: 布置作业 id, 班级

7. 批改作业: 作业打分, 评语

8. 提交作业: 上传作业

4.2 系统需求说明

4.2.1 初始数据库大小

10 名教师
4 个班级, 每个班级 15 名学生
20 门课程
20 道习题
两次作业

4.2.1 网络和共享需求

能够支持至少三分之一人数的人同时访问, 需要考虑并发访问的许可需求

4.2.3 性能

高峰期: 每周周末提交作业的时间
(1) 单个记录查询时间少于 1 秒, 高峰期少于 5 秒
(2) 多个记录查询时间少于 5 秒, 高峰期少于 10 秒
(3) 更新/保存记录时间少于 1 秒, 高峰期少于 5 秒

4.2.4 安全性

(1) 数据库必须要口令保护
(2) 为每类成员分配特定的用户视图所应有的访问权限 (管理员, 教师, 学生)
(3) 用户只能在适合他们完成工作需要的窗口中看到特定的数据

4.2.5 备份和恢复

每天 24 点备份

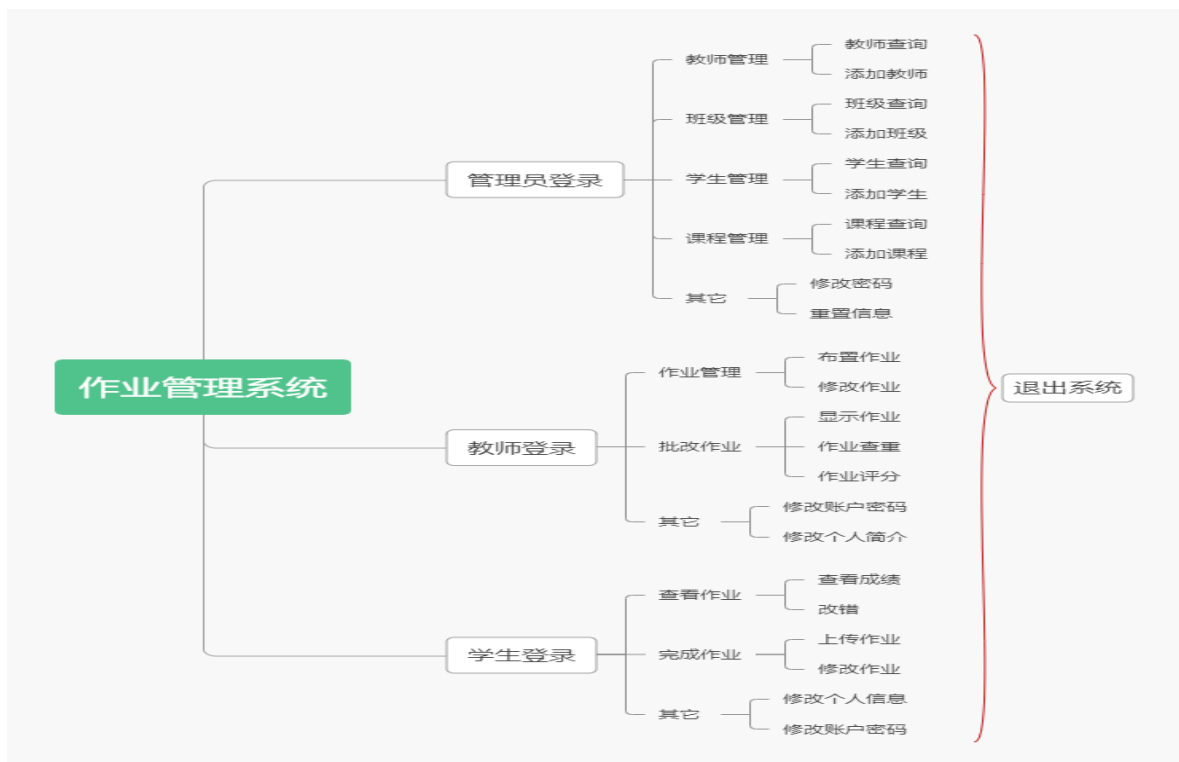
4.2.6 用户界面

菜单驱动, 联机帮助

4.2.7 法律问题

对用户信息管理, 遵守法律

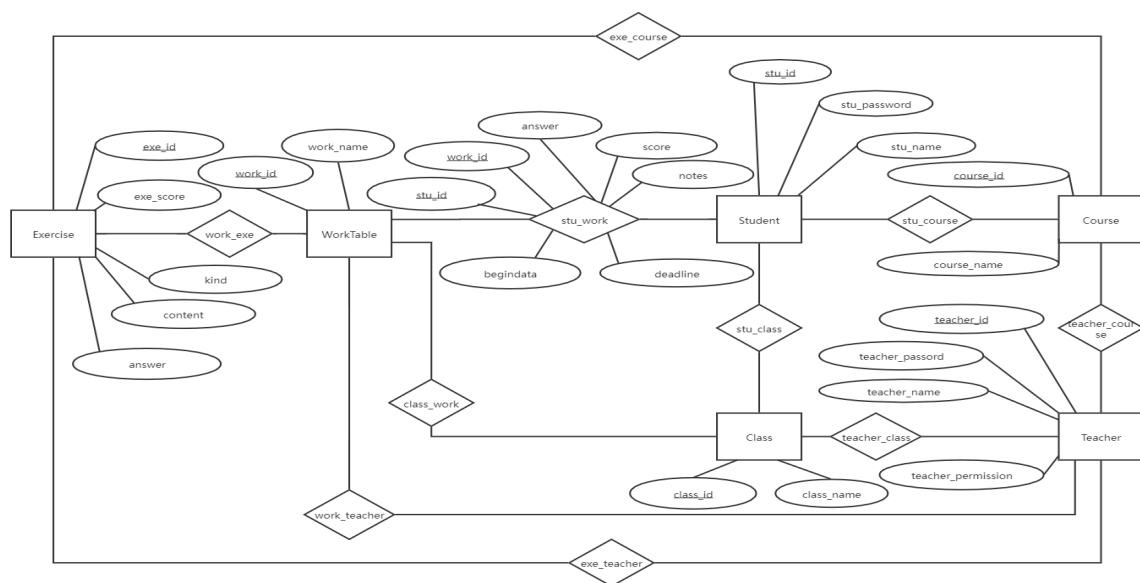
此作业管理系统主要功能如下:



作业管理系统

五. 数据库逻辑设计

5.1 ER 图



ER 图

5.2 数据字典

5.2.1 从数据字典中抽取出来的系统实体描述

教师实体表:

名	类型	长度	小数点	不是 null	虚拟	键	注释
▶ id	int			<input checked="" type="checkbox"/>	<input type="checkbox"/>	1	
password	varchar	18		<input checked="" type="checkbox"/>	<input type="checkbox"/>		
name	varchar	18		<input type="checkbox"/>	<input type="checkbox"/>		
permission	varchar	1		<input type="checkbox"/>	<input type="checkbox"/>		1为系统管理员,2为普通教!

学生实体表:

名	类型	长度	小数点	不是 null	虚拟	键	注释
▶ id	int			<input checked="" type="checkbox"/>	<input type="checkbox"/>	1	
password	varchar	20		<input checked="" type="checkbox"/>	<input type="checkbox"/>		
name	varchar	255		<input checked="" type="checkbox"/>	<input type="checkbox"/>		
klass_id	int			<input checked="" type="checkbox"/>	<input type="checkbox"/>		


班级实体表:

名	类型	长度	小数点	不是 null	虚拟	键	注释
▸ id	int			<input checked="" type="checkbox"/>	<input type="checkbox"/>	 1	
name	varchar	20		<input checked="" type="checkbox"/>	<input type="checkbox"/>		

课程实体表:

名	类型	长度	小数点	不是 null	虚拟	键	注释
▸ id	int			<input checked="" type="checkbox"/>	<input type="checkbox"/>	 1	
name	varchar	40		<input checked="" type="checkbox"/>	<input type="checkbox"/>		

题库实体表:

名	类型	长度	小数点	不是 null	虚拟	键	注释
▸ id	int			<input checked="" type="checkbox"/>	<input type="checkbox"/>	 1	
score	int			<input checked="" type="checkbox"/>	<input type="checkbox"/>		
kind	varchar	1		<input checked="" type="checkbox"/>	<input type="checkbox"/>		题的类型:1为选择题,2为填
content	varchar	3000		<input checked="" type="checkbox"/>	<input type="checkbox"/>		题目内容
answer	varchar	6000		<input type="checkbox"/>	<input type="checkbox"/>		题目答案
course_id	int			<input checked="" type="checkbox"/>	<input type="checkbox"/>		课程表关联外键
teacher_id	int			<input checked="" type="checkbox"/>	<input type="checkbox"/>		teacher表的外键,出题人

作业实体表:

名	类型	长度	小数点	不是 null	虚拟	键	注释
id	int			<input checked="" type="checkbox"/>	<input type="checkbox"/>	1	
title	varchar	30		<input checked="" type="checkbox"/>	<input type="checkbox"/>		作业题目
teacher_id	int			<input checked="" type="checkbox"/>	<input type="checkbox"/>		布置作业教师
course_id	int			<input checked="" type="checkbox"/>	<input type="checkbox"/>		作业相关课程
klass_id	int			<input checked="" type="checkbox"/>	<input type="checkbox"/>		与klass关联的外键
createdate	timestamp			<input checked="" type="checkbox"/>	<input type="checkbox"/>		创建日期
deadline	timestamp			<input checked="" type="checkbox"/>	<input type="checkbox"/>		截止日期

5.2.2 从数据字典中抽取出来的联系的描述

实体一	多样性	联系	多样性	实体二
学生	m	学生班级	1	班级
学生	m	学生选课	n	课程
学生	m	学生作业	n	作业
作业	m	作业题目	n	题目
作业	m	教师布置作业	1	教师
作业	m	作业班级	n	班级
题目	m	题目出题人	1	教师
教师	m	教师班级	n	班级
教师	m	教师教课	n	课程

Table 1: NAME

六. 数据库物理设计

6.1 索引

表名	主键	外键
课程表	id	course_id,teacher_id
习题表	id	
班级表	id	
学生表	id	klass_id
教师表	id	
学生作业表	{stu_id,work_id}	stu_id,work_id
作业习题表	{work_id,exe_id}	work_id,exe_id
作业表	id	teacher_id,course_id,klass_id

Table 2: 索引表

6.2 视图

此环节设计在数据库应用生命周期的需求分析和收集阶段标识的用户视图。通常，视图使用 SQL 或类似 QBE 的工具创建。例如查询所有学生的全部信息

```
CREATE VIEW Student_View AS
```

```
SELECT *
```

```
FROM student;
```

主要视图: 学生当前作业视图

6.3 安全机制

6.3.1 系统安全

1. 本作业管理系统提供了充足的异常处理机制，能够捕获由各种错误引发的异常 (如：输入数据类型与数据库要求类型不一致, 查询过程中出现的错误, 操作不规范等等)。2. 管理员输入信息如果与数据库已有信息重复，则提示管理者重新输入 3. 修改数据时弹出确认修改的对话框, 防止误操作 4. 登陆时增加表单验证, 确保输入格式合法

6.3.2 数据安全

1. 在登陆界面需要输入账号和密码, 每个人的账号与学号绑定不允许随意修改 2. 为不同的用户设置不同的权限, 管理员可以对所有数据进行相应的增删改查操作, 教师只对作业方面的数据具有增删改查等权限, 学生只具有查找和提交作业的权限, 保证了数据安全. 3. 只有管理员可以对学生, 教师, 班级等信息进行增删改查, 都是通过事务实现的, 保护了数据库的强一致性, 体现了数据安全.

6.4 其它

规范化产生一个结构上一致且最小冗余的逻辑数据库设计，但是，规范化的数据库设计有时不能提供最大的处理效率。所以我们愿意接收规范化设计方面的一些损失而实现更好的性能。例如在该系统中，教师信息表中仍然有科室表的科室名称，更好的体现了数据库的 3NF，既加快了检索速度，也方便形成视图。

七. 应用程序设计

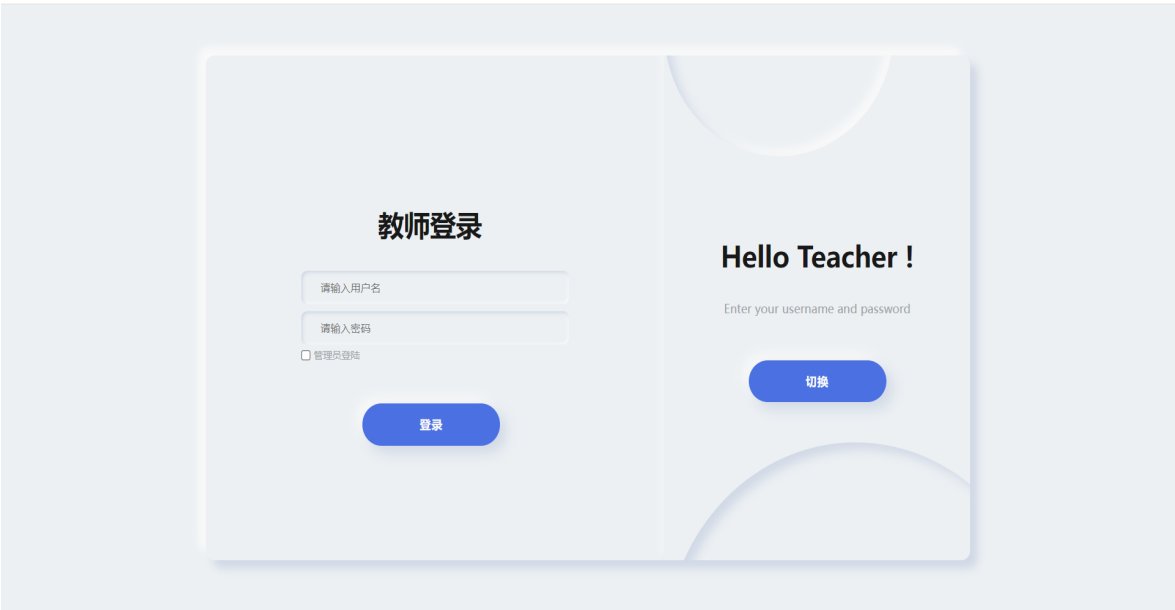
7.1 功能模块

1. 管理员对各种信息进行增删改查
2. 教师对作业信息的增删改查
3. 学生对作业信息的查看和作答

7.2 界面设计

7.2.1 登陆界面





7.2.2 管理员视图

作业管理系统

用户信息管理

学生用户信息管理

教师用户信息管理

班级信息管理

课程信息管理

选课信息管理

习题库

作业表

学生学号 学生学号 学生姓名 学生姓名 学生班级 学生班级 查询

批量删除 新增

<input type="checkbox"/>	学生学号	学生姓名	班级
<input type="checkbox"/>	1 10017	小鸣	2
<input checked="" type="checkbox"/>	2 10018	小一	2
<input type="checkbox"/>	3 10019	小张	2
<input checked="" type="checkbox"/>	4 10020	小爽	2
<input type="checkbox"/>	5 10021	小赵	2

共 8 条 5条/页 < 1 2 > 前往 1 页

作业管理系统

用户信息管理

学生用户信息管理

教师用户信息管理

班级管理

课程信息管理

选课信息管理

习题库

作业表

教师工号

教师工号

教师姓名

教师姓名

系名

系名

查询

批量删除

新增

	教师工号	教师姓名	系名
<input type="checkbox"/>			
<input type="checkbox"/> 1	100000	王老师	数学
<input checked="" type="checkbox"/> 2	100001	张鸣一	英语
<input type="checkbox"/> 3	100002	王老师	物理
<input checked="" type="checkbox"/> 4	100003	张老师	化学
<input type="checkbox"/> 5	100004	李老师	计算机

共 5 条

5条/页

< 1 >

前往

1

页

作业管理系统

用户信息管理

班级管理

班级管理

课程信息管理

选课信息管理

习题库

作业表

班级名称

班级名称

查询

批量删除

新增

	班级编号	班级名称
<input type="checkbox"/>		
<input type="checkbox"/> 1	1	计科一班
<input checked="" type="checkbox"/> 2	2	计科二班
<input type="checkbox"/> 3	3	计科三班
<input checked="" type="checkbox"/> 4	4	计科四班
<input type="checkbox"/> 5	5	大数据

共 6 条

5条/页

< 1 2 >

前往

1

页

作业管理系统

用户信息管理

班级管理

课程信息管理

课程管理

选课信息管理

习题库

作业表

课程名称

课程名称

查询

批量删除

新增

	课程编号	课程名称
<input type="checkbox"/>		
<input type="checkbox"/> 1	1000	新生研讨
<input checked="" type="checkbox"/> 2	1001	计算机导论与程序设计
<input type="checkbox"/> 3	1002	线性代数
<input checked="" type="checkbox"/> 4	1003	思想道德修养与法律基础

共 4 条

5条/页

< 1 >

前往

1

页

作业管理系统

用户信息管理

班级信息管理

课程信息管理

选课信息管理

习题库

习题管理

作业表

习题id课程名称习题类型课程名称出题人课程名称习题科目课程名称

查询

批量删除

新增

		习题编号	习题类型	习题分数	习题内容	习题答案	出题人	习题科目
<input type="checkbox"/>	1	1000000	1	3	如果n阶方阵A的行列式 A =0,则下列正确的是() A.A=0 B.r(A)>0 C.r(A)<n D.r(A)=0	D	100000	1002

共 1 条5条/页< 1 > 前往 1 页

作业管理系统

用户信息管理

班级信息管理

课程信息管理

选课信息管理

习题库

作业表

作业管理

名称名称教师教师相关课程相关课程班级班级

查询

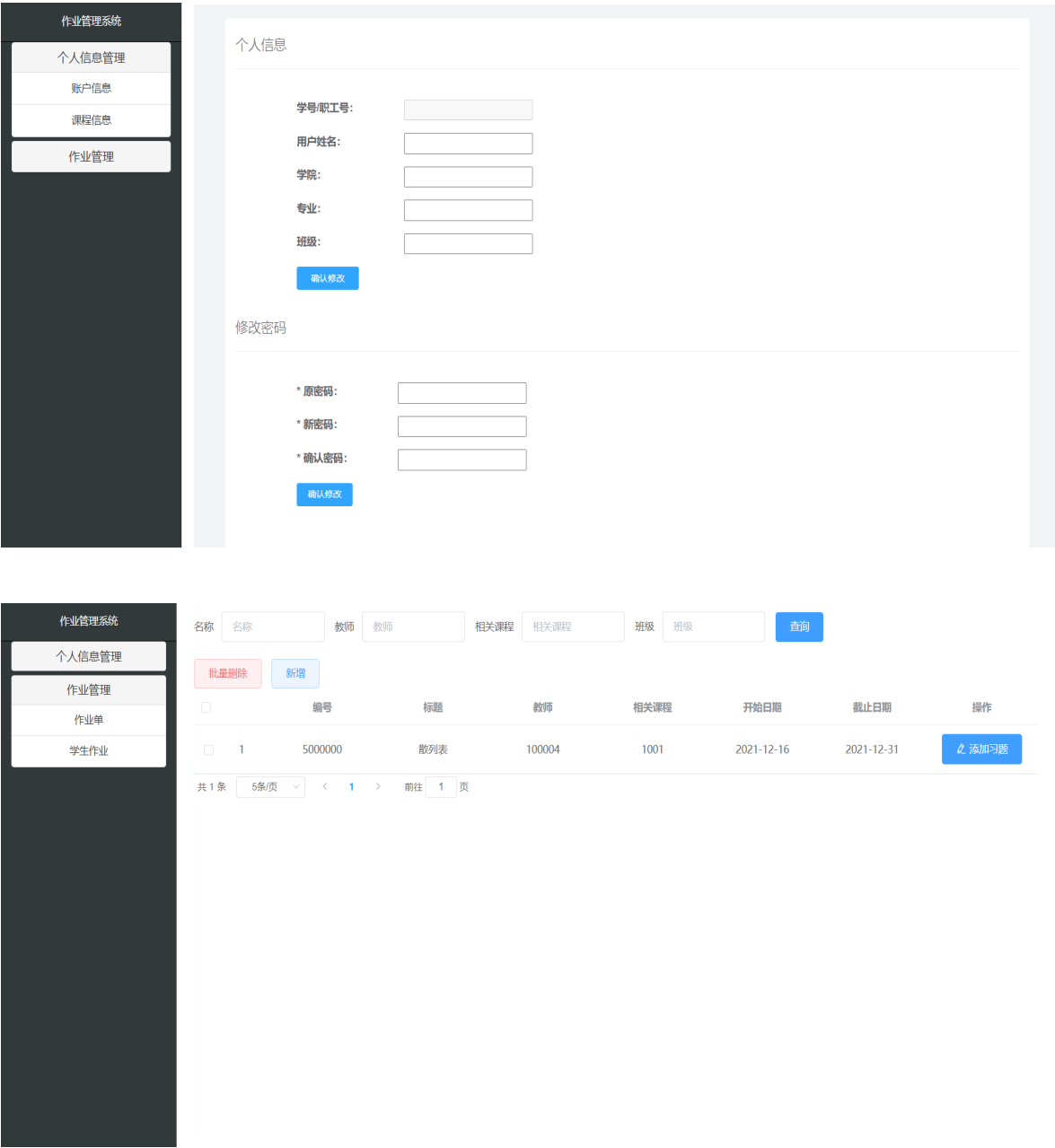
批量删除

新增

		编号	标题	教师	相关课程	开始日期	截止日期	操作
<input type="checkbox"/>	1	5000000	散列表	100004	1001	2021-12-16	2021-12-31	<div>添加习题</div>

共 1 条5条/页< 1 > 前往 1 页

7.2.3 教师视图



作业管理系统

个人信息管理

作业管理

作业单

学生作业

作业单ID

作业单名称

作业状态

班级

查询

<input type="checkbox"/>	作业单编号	答案	得分	教师批注	作业状态	操作
<input type="checkbox"/>	1	5000000	0		0	<div>去作答</div>

共 1 条

5条/页

<

1

>

前往

1

页

7.2.4 学生视图

作业管理系统

个人信息管理

账户信息

作业管理

个人信息

学号/职工号:

用户姓名:

学院:

专业:

班级:

确认修改

修改密码

* 原密码:

* 新密码:

* 确认密码:

确认修改

作业管理系统

个人信息管理

作业管理

历史作业

当前作业

作业单ID

作业单名称

作业状态

班级

查询

<input type="checkbox"/>	作业单编号	答案	得分	教师批注	作业状态	操作
<input type="checkbox"/>	15000000		0		0	<div>去作答</div>

共 1 条

5条/页

<1>

前往1页

7.2.5 添加数据视图

编辑学生信息

学生姓名

班级

提交

取消

编辑教师信息

教师姓名

系

提交

取消

编辑班级信息

×

班级名称

提交

取消

编辑课程信息

×

课程名称

课程学分

提交

取消

编辑习题信息

×

习题类型

习题分数

习题内容

习题答案

出题人

习题科目

提交

取消

编辑作业信息

×

标题

教师

课程

班级

开始日期

截止日期

提交

取消

7.3 事务设计

利用 mybatis 注解和 xml 文件实现

7.3.1 课程事务

```
1 package zms.mapper;
2
3 import org.apache.ibatis.annotations.Insert;
4 import org.apache.ibatis.annotations.Param;
5 import org.apache.ibatis.annotations.ResultMap;
6 import org.apache.ibatis.annotations.Select;
7 import zms.pojo.Course;
8
9 import java.util.List;
10
11 public interface CourseMapper {
12
13     @Select("select id,name from course")
14     @ResultMap("courseResultMap")
15     List<Course> selectAll();
16
17     @Insert("insert into course values(#{courseId},#{courseName},#{credit})")
18     void add(Course course);
19
20     /**
21      * 批量删除
22      * @param ids
23      */
24     void deleteByIds(@Param("ids") int[] ids);
25
26     /* @Delete("delete from course where ")
27     void delete(Course course);*/
28
29     /**
30      * 分页查询
31      * @param begin
32      * @param size
33      * @return
34      */
35     @Select("select * from course limit #{begin},#{size}")
36     List<Course> selectByPage(@Param("begin") int begin,@Param("size") int size);
37
38     /**
39      * 查询总记录数
40      * @return
41      */
42     @Select("select count(*) from course")
43     int selectTotalCount();
44
45
46
47     /**
48      * 分页条件查询
49      * @param begin
50      * @param size
```

```

51     * @return
52     */
53     List<Course> selectByPageAndCondition(@Param("begin") int begin,@Param("size") int size,@Param("courseName") String courseName)
54
55     /**
56     * 根据条件查询总记录数
57     * @return
58     */
59     int selectTotalCountByCondition(@Param("course") Course course);
60
61
62 }

```

```

1  <?xml version="1.0" encoding="UTF-8" ?>
2  <!DOCTYPE mapper
3      PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
4      "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
5  <mapper namespace="zms.mapper.CourseMapper">
6
7      <resultMap id="courseResultMap" type="course">
8          <result property="courseId" column="id" />
9          <result property="courseName" column="name" />
10     </resultMap>
11     <delete id="deleteByIds">
12         delete from course where id in
13         <foreach collection="ids" item="id" separator="," open="(" close=")">
14             #{id}
15         </foreach>
16     </delete>
17
18
19     <select id="selectByPageAndCondition" resultMap="courseResultMap">
20         select *
21         from course
22         <where>
23             <if test="course.courseName!=null and course.courseName!=''">
24                 and name like #{course.courseName}
25             </if>
26         </where>
27
28         limit #{begin},#{size}
29
30     </select>
31
32     <select id="selectTotalCountByCondition" resultType="java.lang.Integer">
33         select count(*)
34         from course
35         <where>
36             <if test="course.courseName!=null and course.courseName!=''">
37                 and name like #{course.courseName}
38             </if>
39         </where>
40     </select>
41 </mapper>

```

7.3.2 习题事务

```
1 package zms.mapper;
2
3 import org.apache.ibatis.annotations.Insert;
4 import org.apache.ibatis.annotations.Param;
5 import org.apache.ibatis.annotations.ResultMap;
6 import org.apache.ibatis.annotations.Select;
7 import zms.pojo.Exercise;
8
9 import java.util.List;
10
11 public interface ExerciseMapper {
12
13     @Select("select id,name from exercise")
14     @ResultMap("exerciseResultMap")
15     List<Exercise> selectAll();
16
17     @Insert("insert into exercise values (null,#{exerciseScore},#{kind},#{content},#{answer},#{courseId},#{userId})")
18     @ResultMap("exerciseResultMap")
19     void add(Exercise exercise);
20
21     /**
22      * 批量删除
23      * @param ids
24      */
25     void deleteByIds(@Param("ids") int[] ids);
26
27     /* @Delete("delete from exercise where ")
28     void delete(Exercise exercise);*/
29
30     /**
31      * 分页查询
32      * @param begin
33      * @param size
34      * @return
35      */
36     @Select("select * from exercise limit #{begin},#{size}")
37     List<Exercise> selectByPage(@Param("begin") int begin,@Param("size") int size);
38
39     /**
40      * 查询总记录数
41      * @return
42      */
43     @Select("select count(*) from exercise")
44     int selectTotalCount();
45
46
47     /**
48      * 分页条件查询
49      * @param begin
50      * @param size
51      * @return
52      */
53     List<Exercise> selectByPageAndCondition(@Param("begin") int begin,@Param("size") int size,@Param("ex")
```

```

55     /**
56     * 根据条件查询总记录数
57     * @return
58     */
59     int selectTotalCountByCondition(@Param("exercise") Exercise exercise);
60
61
62 }

```

```

1  <?xml version="1.0" encoding="UTF-8" ?>
2  <!DOCTYPE mapper
3      PUBLIC "-//mybatis.org/DTD Mapper 3.0//EN"
4      "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
5  <mapper namespace="zms.mapper.ExerciseMapper">
6
7      <resultMap id="exerciseResultMap" type="exercise">
8          <result property="exerciseId" column="id" />
9          <result property="kind" column="kind" />
10         <result property="exerciseScore" column="score" />
11         <result property="content" column="content" />
12         <result property="answer" column="answer" />
13         <result property="teacherId" column="teacher_id" />
14         <result property="courseId" column="course_id" />
15     </resultMap>
16     <delete id="deleteByIds">
17         delete from exercise where id in
18         <foreach collection="ids" item="id" separator="," open="(" close=")">
19             #{id}
20         </foreach>
21     </delete>
22
23
24     <select id="selectByPageAndCondition" resultMap="exerciseResultMap">
25         select *
26         from exercise
27         <where>
28             <if test="exercise.exerciseId!=null">
29                 and exerciseId = #{exercise.exerciseId}
30             </if>
31             <if test="exercise.kind!=null">
32                 and kind = #{exercise.kind}
33             </if>
34             <if test="exercise.teacherId!=null">
35                 and teacherId = #{exercise.teacherId}
36             </if>
37             <if test="exercise.courseId!=null">
38                 and courseId = #{exercise.courseId}
39             </if>
40         </where>
41
42         limit #{begin},#{size}
43
44     </select>
45     <select id="selectTotalCountByCondition" resultType="java.lang.Integer">
46         select count(*)
47         from exercise

```

```

48         <where>
49             <if test="exercise.exerciseId!=null">
50                 and exerciseId = #{exercise.exerciseId}
51             </if>
52             <if test="exercise.kind!=null">
53                 and kind = #{exercise.kind}
54             </if>
55             <if test="exercise.teacherId!=null">
56                 and teacherId = #{exercise.teacherId}
57             </if>
58             <if test="exercise.courseId!=null">
59                 and courseId = #{exercise.courseId}
60             </if>
61         </where>
62     </select>
63 </mapper>

```

7.3.3 班级事务

```

1  package zms.mapper;
2
3  import org.apache.ibatis.annotations.Insert;
4  import org.apache.ibatis.annotations.Param;
5  import org.apache.ibatis.annotations.ResultMap;
6  import org.apache.ibatis.annotations.Select;
7  import zms.pojo.Klass;
8
9  import java.util.List;
10
11 public interface KlassMapper {
12
13     @Select("select id,name from klass")
14     @ResultMap("klassResultMap")
15     List<Klass> selectAll();
16
17     @Insert("insert into klass values(#{klassId},#{className})")
18     void add(Klass klass);
19
20     /**
21      * 批量删除
22      * @param ids
23      */
24     void deleteByIds(@Param("ids") int[] ids);
25
26     /**
27      * @Delete("delete from klass where ")
28      void delete(Klass klass);*/
29
30     /**
31      * 分页查询
32      * @param begin
33      * @param size
34      * @return
35      */
36     @Select("select * from klass limit #{begin},#{size}")
37     List<Klass> selectByPage(@Param("begin") int begin,@Param("size") int size);

```

```

37
38     /**
39      * 查询总记录数
40      * @return
41      */
42     @Select("select count(*) from klass")
43     int selectTotalCount();
44
45
46
47     /**
48      * 分页条件查询
49      * @param begin
50      * @param size
51      * @return
52      */
53     List<Klass> selectByPageAndCondition(@Param("begin") int begin, @Param("size") int size, @Param("klass")
54
55     /**
56      * 根据条件查询总记录数
57      * @return
58      */
59     int selectTotalCountByCondition(@Param("klass") Klass klass);
60
61
62 }

```

```

1  <?xml version="1.0" encoding="UTF-8" ?>
2  <!DOCTYPE mapper
3      PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
4      "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
5  <mapper namespace="zms.mapper.KlassMapper">
6
7      <resultMap id="klassResultMap" type="klass">
8          <result property="klassId" column="id" />
9          <result property="klassName" column="name" />
10     </resultMap>
11     <delete id="deleteByIds">
12         delete from klass where id in
13         <foreach collection="ids" item="id" separator="," open="(" close=")">
14             #{id}
15         </foreach>
16     </delete>
17
18
19     <select id="selectByPageAndCondition" resultMap="klassResultMap">
20         select *
21         from klass
22         <where>
23             <if test="klass.klassName!=null and klass.klassName!=''">
24                 and name like #{klass.klassName}
25             </if>
26         </where>
27
28         limit #{begin},#{size}
29

```



```

30     </select>
31     <select id="selectTotalCountByCondition" resultType="java.lang.Integer">
32         select count(*)
33         from klass
34         <where>
35             <if test="klass.klassName!=null and klass.klassName!=''">
36                 and name like #{klass.klassName}
37             </if>
38         </where>
39     </select>
40 </mapper>

```

7.3.4 学生事务

```

1  package zms.mapper;
2
3  import org.apache.ibatis.annotations.*;
4  import zms.pojo.Student;
5  import zms.service.StudentService;
6
7  import java.util.List;
8
9  public interface StudentMapper {
10
11     @Select("select id,name,klass_id from student")
12     @ResultMap("studentResultMap")
13     List<Student> selectAll();
14
15     @Select("select * from student where id=#{id} and password=#{password}")
16     Student select(@Param("id") Integer id,@Param("password") String password);
17
18
19     @Insert("insert into student values (null,#{studentName},#{klassId},12345)")
20     void add(Student student);
21
22     /**
23      * 批量删除
24      * @param ids
25      */
26     void deleteByIds(@Param("ids") int[] ids);
27
28     /*      @Delete("delete from student where ")
29     void delete(Student student);*/
30
31     /**
32      * 分页查询
33      * @param begin
34      * @param size
35      * @return
36      */
37     @Select("select * from student limit #{begin},#{size}")
38     List<Student> selectByPage(@Param("begin") int begin,@Param("size") int size);
39
40     /**
41      * 查询总记录数

```

```

42     * @return
43     */
44     @Select("select count(*) from student")
45     int selectTotalCount();
46
47
48
49     /**
50     * 分页条件查询
51     * @param begin
52     * @param size
53     * @return
54     */
55     List<Student> selectByPageAndCondition(@Param("begin") int begin, @Param("size") int size, @Param("student") Student student);
56
57     /**
58     * 根据条件查询总记录数
59     * @return
60     */
61     int selectTotalCountByCondition(@Param("student") Student student);
62
63
64 }

```

```

1  <?xml version="1.0" encoding="UTF-8" ?>
2  <!DOCTYPE mapper
3      PUBLIC "-//mybatis.org/DTD Mapper 3.0//EN"
4      "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
5  <mapper namespace="zms.mapper.StudentMapper">
6
7      <resultMap id="studentResultMap" type="student">
8          <result property="klassId" column="klass_id" />
9          <result property="studentId" column="id" />
10         <result property="studentName" column="name" />
11         <result property="studentPassword" column="password" />
12     </resultMap>
13     <delete id="deleteByIds">
14         delete from student where id in
15         <foreach collection="ids" item="id" separator="," open="(" close=")">
16             #{id}
17         </foreach>
18     </delete>
19
20
21     <select id="selectByPageAndCondition" resultMap="studentResultMap">
22         select *
23         from student
24         <where>
25             <if test="student.name_!=null and student.name_!=''">
26                 and name like #{student.name}
27             </if>
28             <if test="student.id_!=null">
29                 and id = #{student.id}
30             </if>
31             <if test="student.klass_id_!=null">
32                 and klass_id = #{student.klass_id}

```

```

33         </if>
34     </where>
35
36     limit #{begin},#{size}
37
38 </select>
39 <select id="selectTotalCountByCondition" resultType="java.lang.Integer">
40     select count(*)
41     from student
42     <where>
43         <if test="student.name_!=null and student.name_!=''">
44             and name like #{student.name}
45         </if>
46         <if test="student.id_!=null">
47             and id = #{student.id}
48         </if>
49         <if test="student.klass_id_!=null">
50             and klass_id = #{student.klass_id}
51         </if>
52     </where>
53 </select>
54 </mapper>

```

7.3.5 学生作业事务

```

1  package zms.web.servlet;
2
3
4  import com.alibaba.fastjson.JSON;
5  import zms.pojo.PageBean;
6  import zms.pojo.StuWork;
7  import zms.service.impl.StuWorkServiceImpl;
8  import zms.service.StuWorkService;
9
10 import javax.servlet.ServletException;
11 import javax.servlet.annotation.WebServlet;
12 import javax.servlet.http.HttpServletRequest;
13 import javax.servlet.http.HttpServletResponse;
14 import java.io.BufferedReader;
15 import java.io.IOException;
16 import java.util.List;
17
18 @WebServlet("/stuWork/*")
19 public class StuWorkServlet extends BaseServlet {
20
21     private StuWorkService stuWorkService = new StuWorkServiceImpl();
22
23     public void selectAll(HttpServletRequest request, HttpServletResponse response) throws ServletException {
24         //调用1.查询service
25         List<StuWork> stuWorks = stuWorkService.selectAll();
26
27         //转为2.JSON
28         String jsonString = JSON.toJSONString(stuWorks);
29
30         //写数据3.

```

```

31         response.setContentType("text/json;charset=utf-8");
32         response.getWriter().write(jsonString);
33     }
34
35     public void add(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
36         //接受学生数据1.
37         BufferedReader br = request.getReader();
38         String params = br.readLine();//字符串json
39
40         //转为对应对象StuWork
41         StuWork stuWork = JSON.parseObject(params, StuWork.class);
42
43         //调用2.添加service
44         stuWorkService.add(stuWork);
45
46         //响应成功的标识.
47         response.getWriter().write("success");
48     }
49
50     /**
51     * 批量删除
52     *
53     * @param request
54     * @param response
55     * @throws ServletException
56     * @throws IOException
57     */
58     public void deleteByIds(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
59         //接受学生数据1. [1,2,3]
60         BufferedReader br = request.getReader();
61         String params = br.readLine();//字符串json
62
63         //转为对应int[]
64         int[] ids = JSON.parseObject(params, int[].class);
65
66         //调用2.添加service
67         stuWorkService.deleteByIds(ids);
68
69         //响应成功的标识.
70         response.getWriter().write("success");
71     }
72
73
74     /**
75     * 分页查询
76     *
77     * @param request
78     * @param response
79     * @throws ServletException
80     * @throws IOException
81     */
82     public void selectByPage(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
83         //接受当前页码和每页展示条数1. url?currentPage=1&pageSize=5
84         String _currentPage = request.getParameter("currentPage");
85         String _pageSize = request.getParameter("pageSize");
86         ;
87

```

```

88         int currentPage = Integer.parseInt(_currentPage);
89         int pageSize = Integer.parseInt(_pageSize);
90
91         //调用2.查询service
92         PageBean<StuWork> pageBean = stuWorkService.selectByPage(currentPage, pageSize);
93
94         //转为3.JSON
95         String jsonString = JSON.toJSONString(pageBean);
96
97         //写数据4.
98         response.setContentType("text/json;charset=utf-8");
99         response.getWriter().write(jsonString);
100     }
101
102     /**
103      * 分页条件查询
104      *
105      * @param request
106      * @param response
107      * @throws ServletException
108      * @throws IOException
109      */
110     public void selectByPageAndCondition(HttpServletRequest request, HttpServletResponse response) throws
111         //接受当前页码和每页展示条数1. url?currentPage=1&pageSize=5
112         String _currentPage = request.getParameter("currentPage");
113         String _pageSize = request.getParameter("pageSize");
114         ;
115
116         int currentPage = Integer.parseInt(_currentPage);
117         int pageSize = Integer.parseInt(_pageSize);
118
119         //获取对应的查询条件对象
120         BufferedReader br = request.getReader();
121         String params = br.readLine(); //字符串json
122
123         //转为对应对象StuWork
124         StuWork stuWork = JSON.parseObject(params, StuWork.class);
125
126         //调用2.查询service
127         PageBean<StuWork> pageBean = stuWorkService.selectByPageAndCondition(currentPage, pageSize, stuW
128
129         //转为3.JSON
130         String jsonString = JSON.toJSONString(pageBean);
131
132         //写数据4.
133         response.setContentType("text/json;charset=utf-8");
134         response.getWriter().write(jsonString);
135     }
136 }
137 }

```

```

1 <?xml version="1.0" encoding="UTF-8" ?>
2 <!DOCTYPE mapper
3     PUBLIC "-//mybatis.org/DTD Mapper 3.0//EN"
4     "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
5 <mapper namespace="zms.mapper.StuWorkMapper">

```

```

6
7     <resultMap id="stuWorkResultMap" type="stuWork">
8         <result property="stuId" column="stu_id"/>
9         <result property="workId" column="work_id"/>
10        <result property="stuAnswer" column="stu_answer"/>
11        <result property="stuScore" column="score"/>
12        <result property="notes" column="notes"/>
13        <result property="state" column="state"/>
14    </resultMap>
15    <delete id="deleteByIds">
16        delete from stu_work where id in
17        <foreach collection="ids" item="id" separator="," open="(" close=")">
18            #{id}
19        </foreach>
20    </delete>
21
22
23    <select id="selectByPageAndCondition" resultMap="stuWorkResultMap">
24        select *
25        from stu_work
26        <where>
27            <if test="stuWork.workId!=null">
28                and workId = #{stuWork.workId}
29            </if>
30            <if test="stuWork.state!=null">
31                and state = #{stuWork.state}
32            </if>
33        </where>
34
35        limit #{begin},#{size}
36
37    </select>
38    <select id="selectTotalCountByCondition" resultType="java.lang.Integer">
39        select count(*)
40        from stu_work
41        <where>
42            <if test="stuWork.workId!=null">
43                and workId = #{stuWork.workId}
44            </if>
45            <if test="stuWork.state!=null">
46                and state = #{stuWork.state}
47            </if>
48        </where>
49    </select>
50 </mapper>

```

7.3.6 教师事务

```

1 package zms.mapper;
2
3 import org.apache.ibatis.annotations.Insert;
4 import org.apache.ibatis.annotations.Param;
5 import org.apache.ibatis.annotations.ResultMap;
6 import org.apache.ibatis.annotations.Select;
7 import zms.pojo.Teacher;

```

```

8
9 import java.util.List;
10
11 public interface TeacherMapper {
12
13     @Select("select id,name,department from teacher")
14     @ResultMap("teacherResultMap")
15     List<Teacher> selectAll();
16
17     @Select("select * from teacher where id=#{id} and password=#{password} and permission=#{permission}")
18     Teacher select(@Param("id") Integer id,@Param("password") String password,@Param("permission") Integer permission);
19
20
21     @Insert("insert into teacher values (null,#{teacherName},#{department},1,123456)")
22     void add(Teacher teacher);
23
24     /**
25      * 批量删除
26      * @param ids
27      */
28     void deleteByIds(@Param("ids") int[] ids);
29
30     /*      @Delete("delete from teacher where ")
31     void delete(Teacher teacher);*/
32
33     /**
34      * 分页查询
35      * @param begin
36      * @param size
37      * @return
38      */
39     @Select("select * from teacher limit #{begin},#{size}")
40     List<Teacher> selectByPage(@Param("begin") int begin,@Param("size") int size);
41
42     /**
43      * 查询总记录数
44      * @return
45      */
46     @Select("select count(*) from teacher")
47     int selectTotalCount();
48
49
50
51     /**
52      * 分页条件查询
53      * @param begin
54      * @param size
55      * @return
56      */
57     List<Teacher> selectByPageAndCondition(@Param("begin") int begin,@Param("size") int size,@Param("teacher") Teacher teacher);
58
59     /**
60      * 根据条件查询总记录数
61      * @return
62      */
63     int selectTotalCountByCondition(@Param("teacher") Teacher teacher);
64

```

```
65
66 }
```

```
1  <?xml version="1.0" encoding="UTF-8" ?>
2  <!DOCTYPE mapper
3      PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
4      "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
5  <mapper namespace="zms.mapper.TeacherMapper">
6
7      <resultMap id="teacherResultMap" type="teacher">
8          <result property="teacherId" column="id" />
9          <result property="teacherName" column="name" />
10         <result property="department" column="department" />
11         <result property="teacherPassword" column="password" />
12     </resultMap>
13     <delete id="deleteByIds">
14         delete from teacher where id in
15         <foreach collection="ids" item="id" separator="," open="(" close=")">
16             #{id}
17         </foreach>
18     </delete>
19
20
21     <select id="selectByPageAndCondition" resultMap="teacherResultMap">
22         select *
23         from teacher
24         <where>
25             <if test="teacher.teacherName_!=_null_and_ teacher.teacherName_!=_''">
26                 and name like #{teacher.teacherName}
27             </if>
28             <if test="teacher.teacherId_!=_null">
29                 and id = #{teacher.teacherId}
30             </if>
31             <if test="teacher.department_!=_null_and_ teacher.department_!=_''">
32                 and department like #{teacher.department}
33             </if>
34         </where>
35
36         limit #{begin},#{size}
37
38     </select>
39     <select id="selectTotalCountByCondition" resultType="java.lang.Integer">
40         select count(*)
41         from teacher
42         <where>
43             <if test="teacher.teacherName_!=_null_and_ teacher.teacherName_!=_''">
44                 and name like #{teacher.teacherName}
45             </if>
46             <if test="teacher.teacherId_!=_null">
47                 and id = #{teacher.teacherId}
48             </if>
49             <if test="teacher.department_!=_null_and_ teacher.department_!=_''">
50                 and department like #{teacher.department}
51             </if>
52         </where>
53     </select>
```


7.3.7 作业事务

```

1      package zms.web.servlet;
2
3
4      import com.alibaba.fastjson.JSON;
5      import zms.pojo.Worktable;
6      import zms.pojo.PageBean;
7      import zms.service.WorktableService;
8      import zms.service.impl.WorktableServiceImpl;
9
10     import javax.servlet.ServletException;
11     import javax.servlet.annotation.WebServlet;
12     import javax.servlet.http.HttpServletRequest;
13     import javax.servlet.http.HttpServletResponse;
14     import java.io.BufferedReader;
15     import java.io.IOException;
16     import java.util.List;
17
18     @WebServlet("/worktable/*")
19     public class WorktableServlet extends BaseServlet {
20
21         private WorktableService worktableService = new WorktableServiceImpl();
22
23         public void selectAll(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
24             //调用1.查询service
25             List<Worktable> worktables = worktableService.selectAll();
26
27             //转为2.JSON
28             String jsonString = JSON.toJSONString(worktables);
29
30             //写数据3.
31             response.setContentType("text/json;charset=utf-8");
32             response.getWriter().write(jsonString);
33         }
34
35         public void add(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
36             //接受学生数据1.
37             BufferedReader br = request.getReader();
38             String params = br.readLine(); //字符串json
39
40             //转为对应对象Worktable
41             Worktable worktable = JSON.parseObject(params, Worktable.class);
42
43             //调用2.添加service
44             worktableService.add(worktable);
45
46             //响应成功的标识.
47             response.getWriter().write("success");
48         }
49
50         /**
51          * 批量删除

```

```

52     *
53     * @param request
54     * @param response
55     * @throws ServletException
56     * @throws IOException
57     */
58     public void deleteByIds(HttpServletRequest request, HttpServletResponse response) throws ServletException,
59         //接受学生数据1. [1,2,3]
60         BufferedReader br = request.getReader();
61         String params = br.readLine(); //字符串json
62
63         //转为对应int[]
64         int[] ids = JSON.parseObject(params, int[].class);
65
66         //调用2.添加service
67         worktableService.deleteByIds(ids);
68
69         //响应成功的标识
70         response.getWriter().write("success");
71     }
72
73     /**
74     * 分页查询
75     *
76     * @param request
77     * @param response
78     * @throws ServletException
79     * @throws IOException
80     */
81     public void selectByPage(HttpServletRequest request, HttpServletResponse response) throws ServletException,
82         //接受当前页码和每页展示条数1. url?currentPage=1&pageSize=5
83         String __currentPage = request.getParameter("currentPage");
84         String __pageSize = request.getParameter("pageSize");
85         ;
86
87         int currentPage = Integer.parseInt(__currentPage);
88         int pageSize = Integer.parseInt(__pageSize);
89
90         //调用2.查询service
91         PageBean<Worktable> pageBean = worktableService.selectByPage(currentPage, pageSize);
92
93         //转为3.JSON
94         String jsonString = JSON.toJSONString(pageBean);
95
96         //写数据4.
97         response.setContentType("text/json; charset=utf-8");
98         response.getWriter().write(jsonString);
99     }
100
101     /**
102     * 分页条件查询
103     *
104     * @param request
105     * @param response
106     * @throws ServletException

```

```

109     * @throws IOException
110     */
111     public void selectByPageAndCondition(HttpServletRequest request, HttpServletResponse response) throws
112         //接受当前页码和每页展示条数1. url?currentPage=1&pageSize=5
113         String _currentPage = request.getParameter("currentPage");
114         String _pageSize = request.getParameter("pageSize");
115         ;
116
117         int currentPage = Integer.parseInt(_currentPage);
118         int pageSize = Integer.parseInt(_pageSize);
119
120         //获取对应的查询条件对象
121         BufferedReader br = request.getReader();
122         String params = br.readLine(); //字符串json
123
124         //转为对应对象Worktable
125         Worktable worktable = JSON.parseObject(params, Worktable.class);
126
127         //调用2.查询service
128         PageBean<Worktable> pageBean = worktableService.selectByPageAndCondition(currentPage, pageSize, v
129
130         //转为3.JSON
131         String jsonString = JSON.toJSONString(pageBean);
132
133         //写数据4.
134         response.setContentType("text/json; charset=utf-8");
135         response.getWriter().write(jsonString);
136     }
137 }

```

```

1  <?xml version="1.0" encoding="UTF-8" ?>
2  <!DOCTYPE mapper
3      PUBLIC "-//mybatis.org/DTD Mapper 3.0//EN"
4      "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
5  <mapper namespace="zms.mapper.WorktableMapper">
6
7      <resultMap id="worktableResultMap" type="worktable">
8          <result property="worktableId" column="id"/>
9          <result property="title" column="title"/>
10         <result property="teacherId" column="teacher_id"/>
11         <result property="courseId" column="course_id"/>
12         <result property="klassId" column="klass_id"/>
13         <result property="createDate" column="createdate"/>
14         <result property="deadLine" column="deadline"/>
15     </resultMap>
16     <delete id="deleteByIds">
17         delete from worktable where id in
18         <foreach collection="ids" item="id" separator="," open="(" close=")">
19             #{id}
20         </foreach>
21     </delete>
22
23
24     <select id="selectByPageAndCondition" resultMap="worktableResultMap">
25         select *
26         from worktable

```

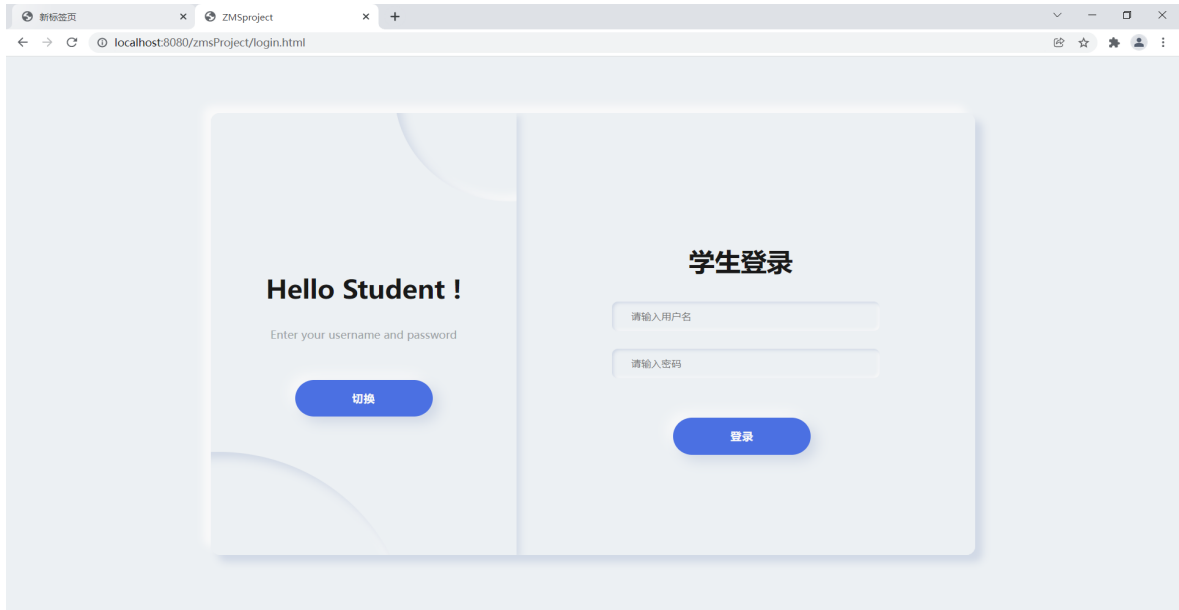
```

27     <where>
28         <if test="worktable.title!=null and worktable.title!=''">
29             and title like #{worktable.title}
30         </if>
31         <if test="worktable.teacherId!=null">
32             and teacherId = #{worktable.teacherId}
33         </if>
34         <if test="worktable.courseId!=null">
35             and courseId = #{worktable.courseId}
36         </if>
37         <if test="worktable.klassId!=null">
38             and klassId = #{worktable.klassId}
39         </if>
40     </where>
41
42     limit #{begin},#{size}
43
44 </select>
45 <select id="selectTotalCountByCondition" resultType="java.lang.Integer">
46     select count(*)
47     from worktable
48     <where>
49         <if test="worktable.title!=null and worktable.title!=''">
50             and title like #{worktable.title}
51         </if>
52         <if test="worktable.teacherId!=null">
53             and teacherId = #{worktable.teacherId}
54         </if>
55         <if test="worktable.courseId!=null">
56             and courseId = #{worktable.courseId}
57         </if>
58         <if test="worktable.klassId!=null">
59             and klassId = #{worktable.klassId}
60         </if>
61     </where>
62 </select>
63 </mapper>

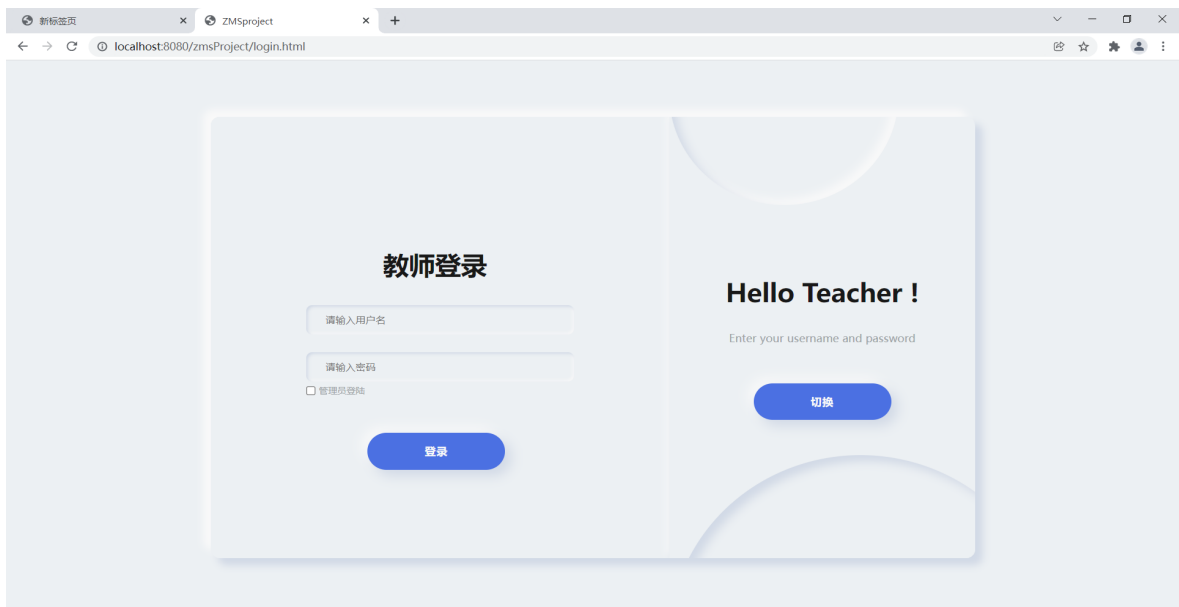
```

八. 测试和运行

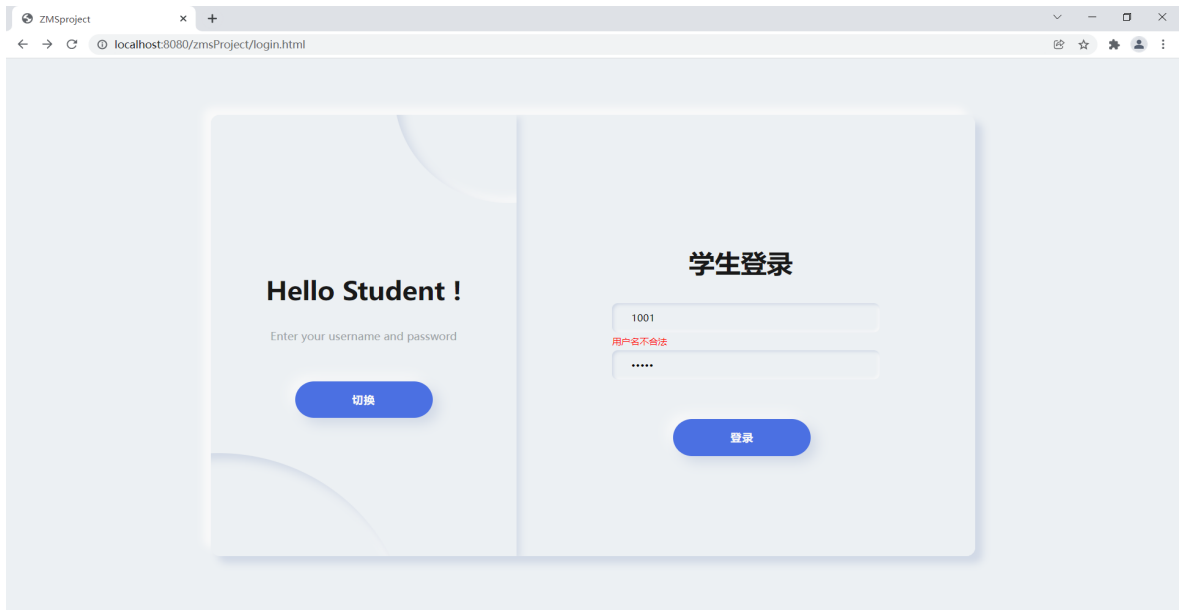
运行界面



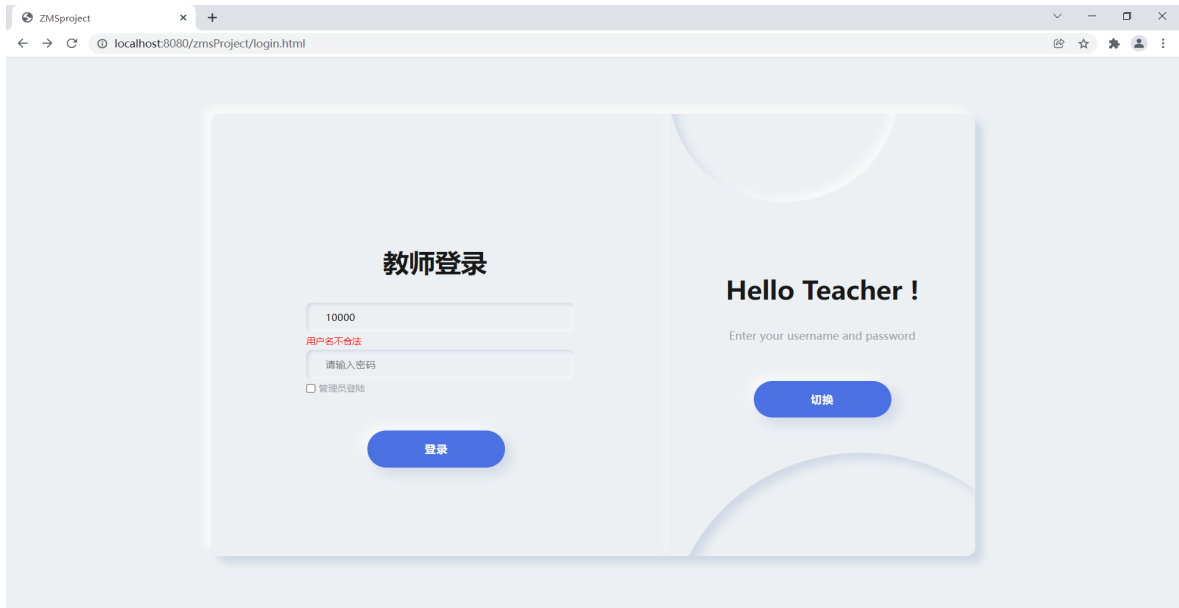
学生登陆界面



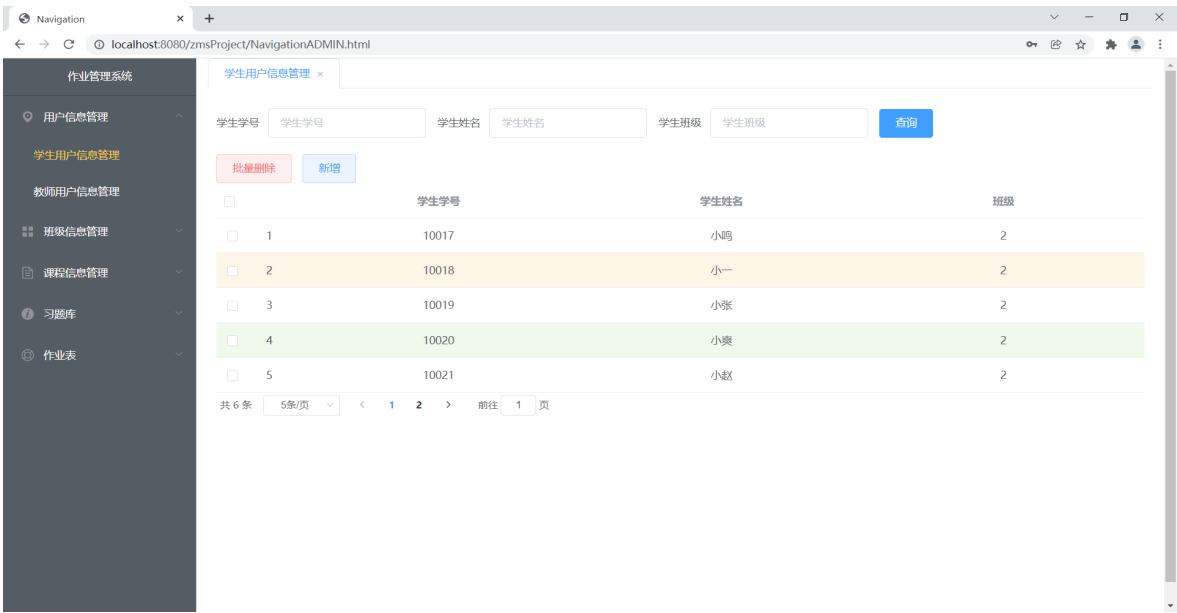
教师登录界面



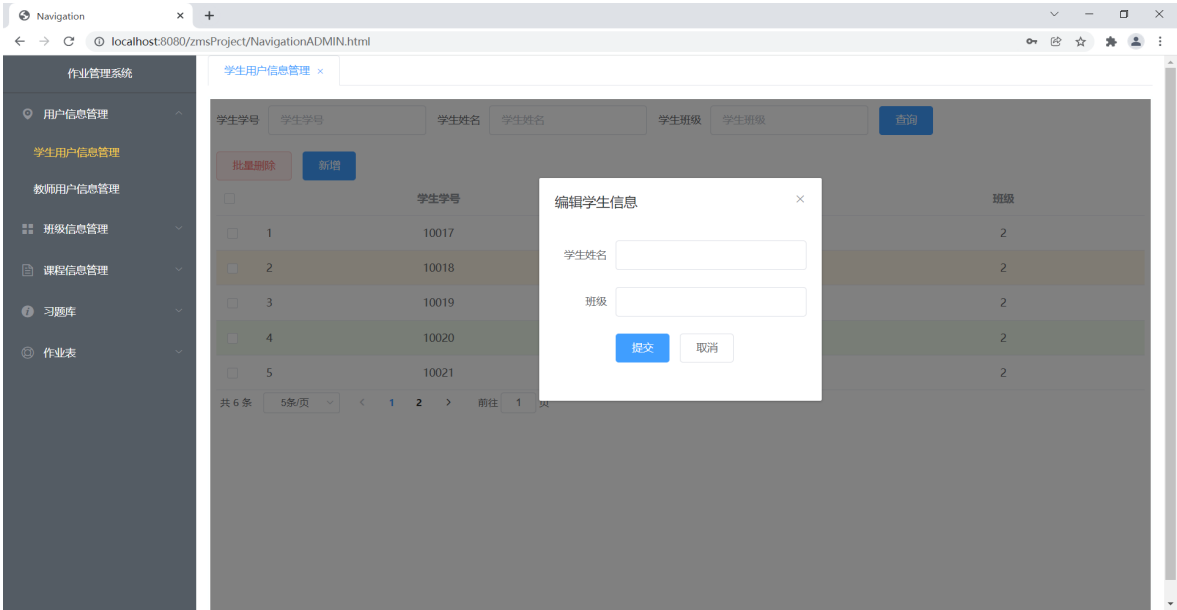
学生用户名不合法界面



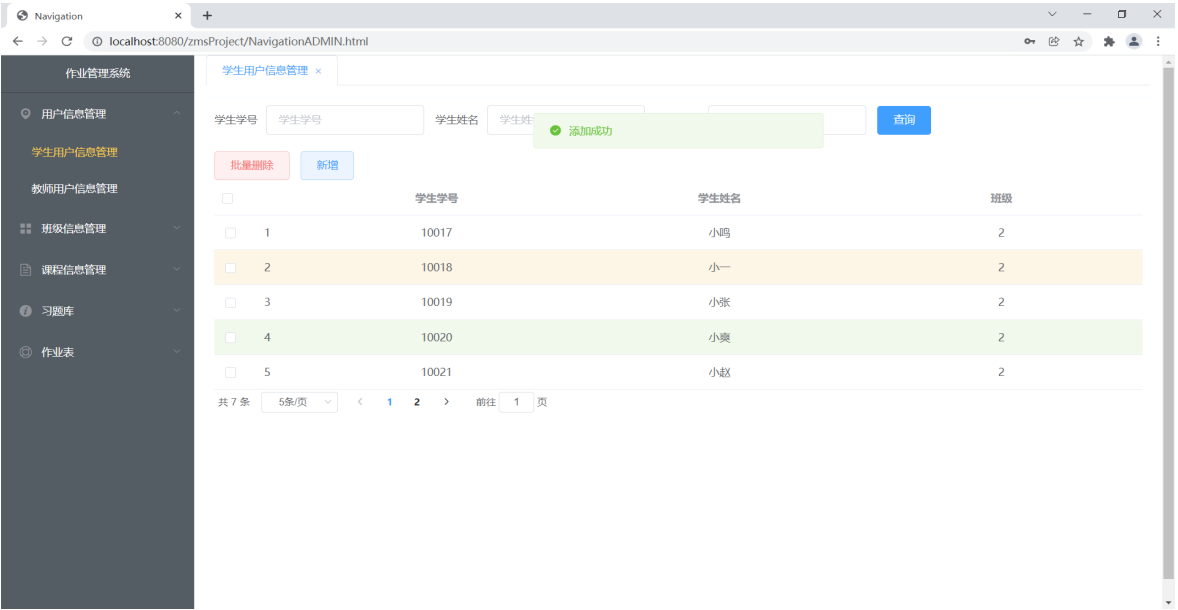
教师用户名不合法界面



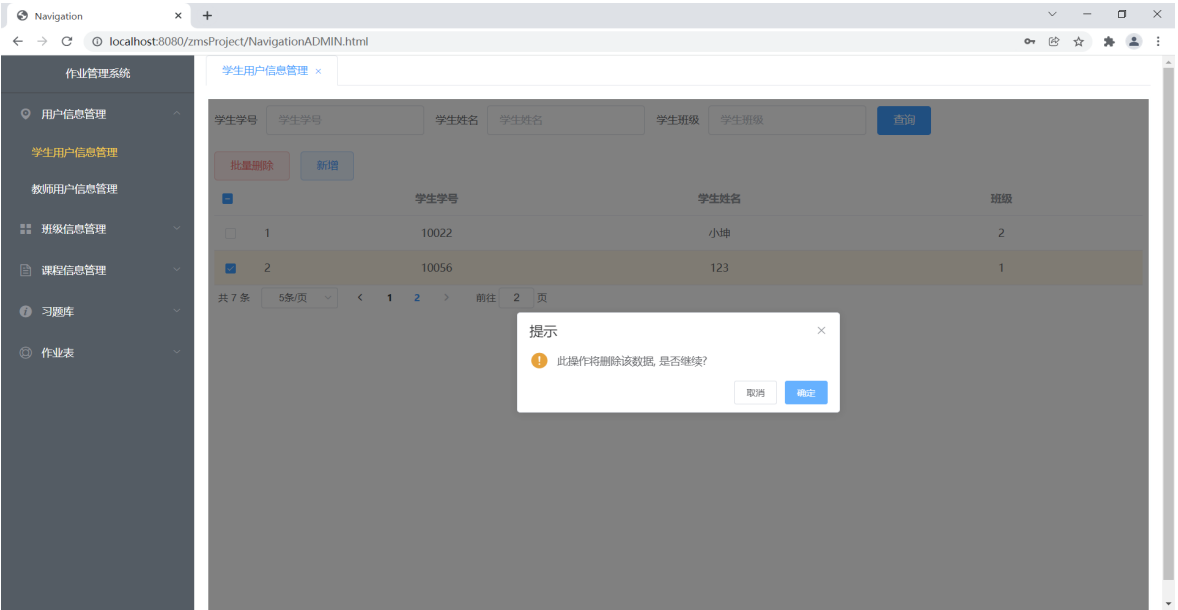
学生用户信息管理界面



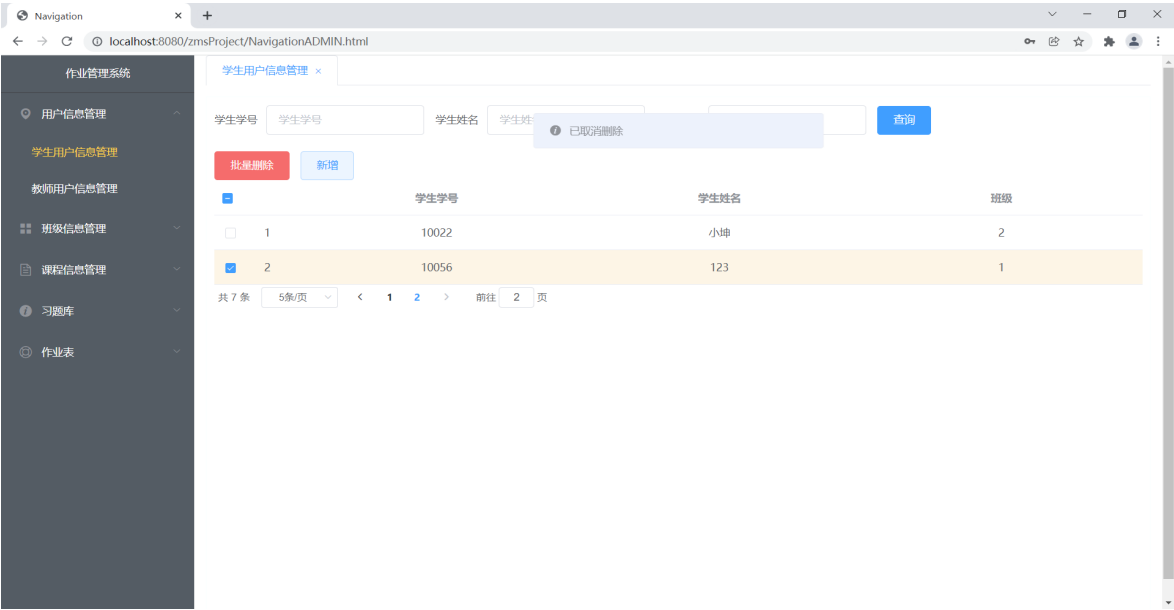
添加学生信息



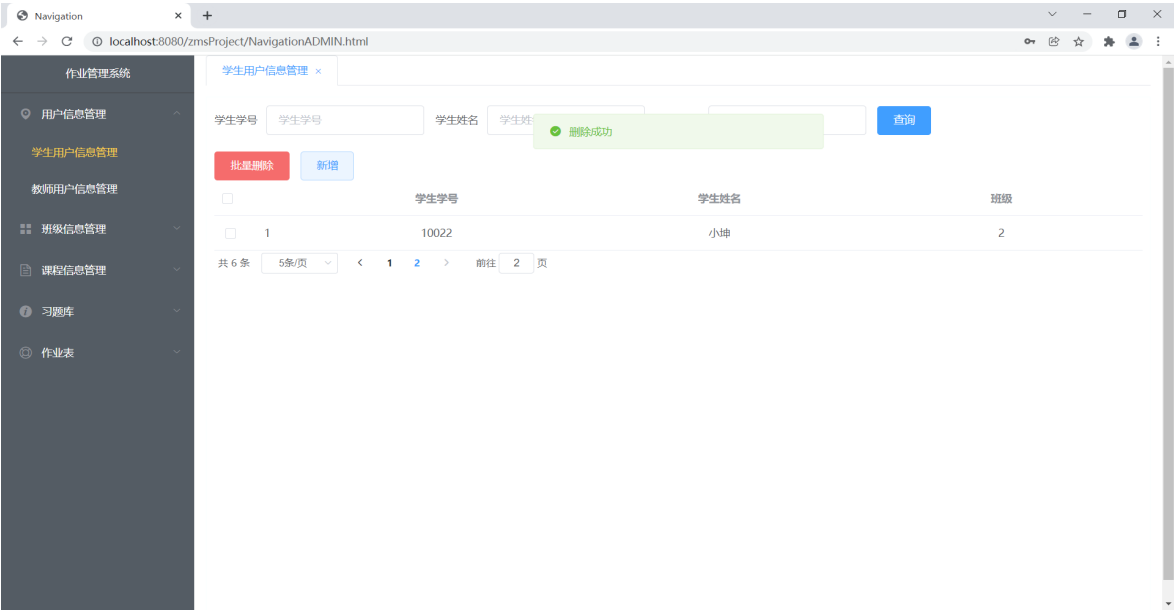
添加成功



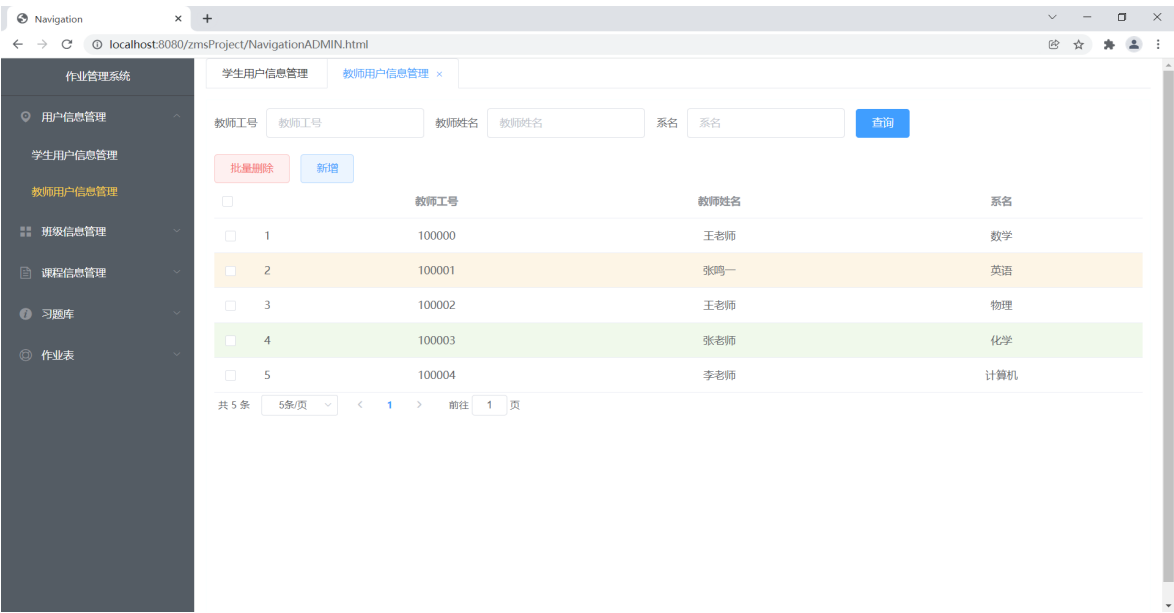
学生删除界面



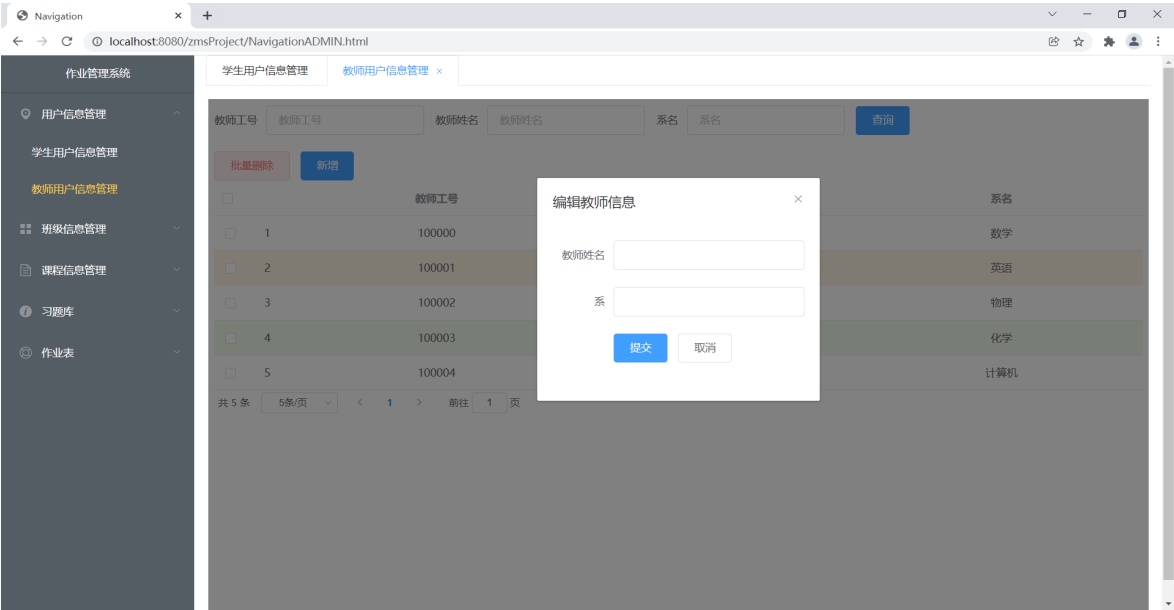
取消删除



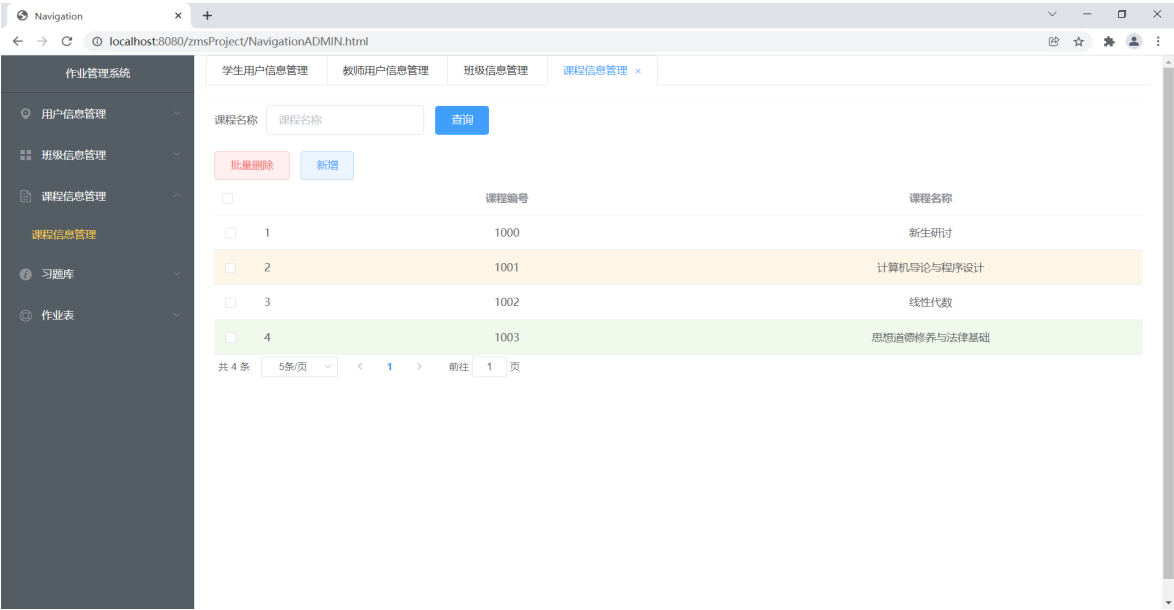
确认删除



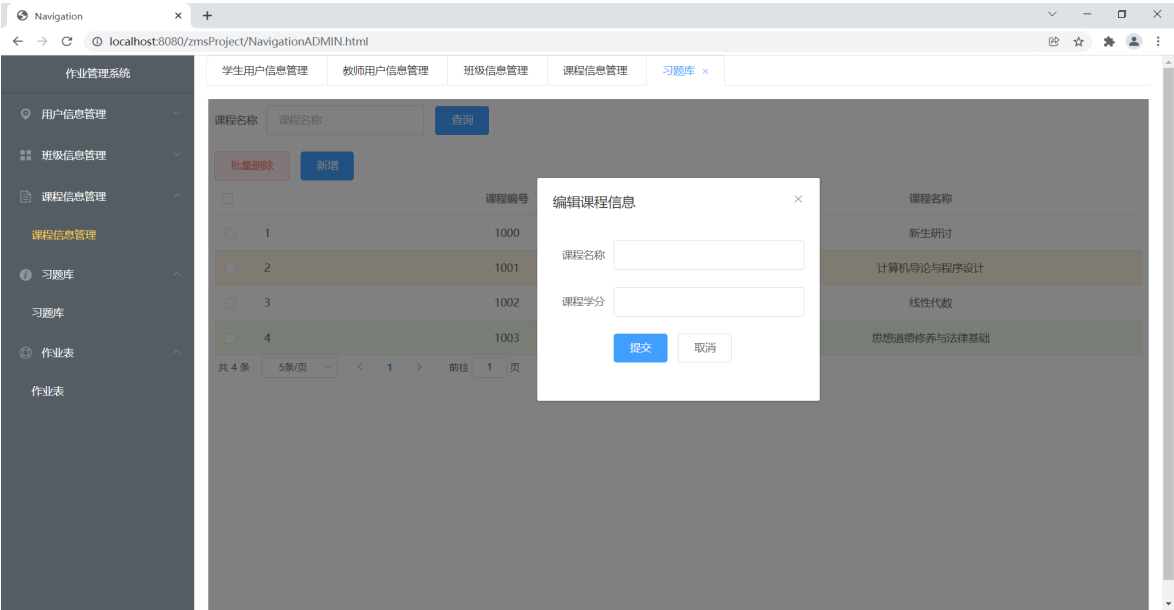
教师用户信息管理界面



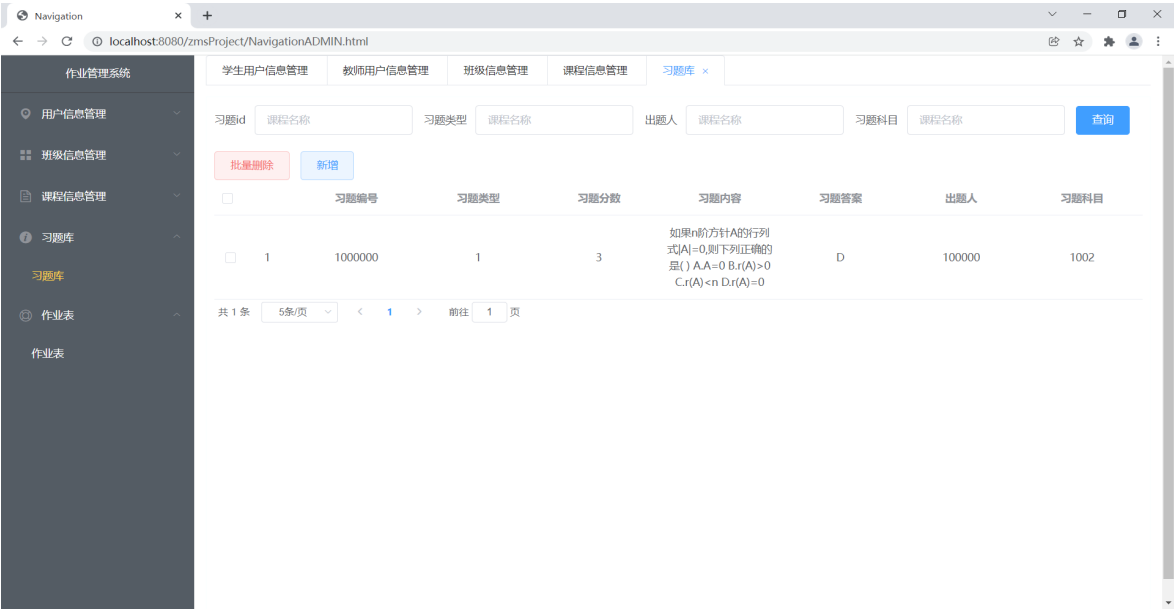
添加教师信息



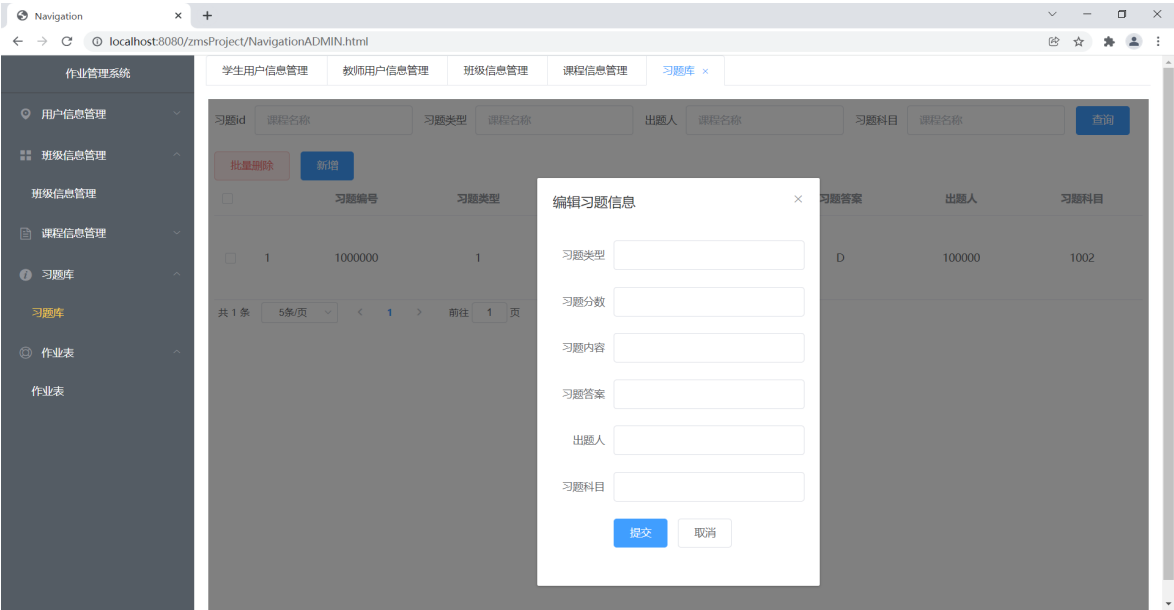
课程信息管理界面



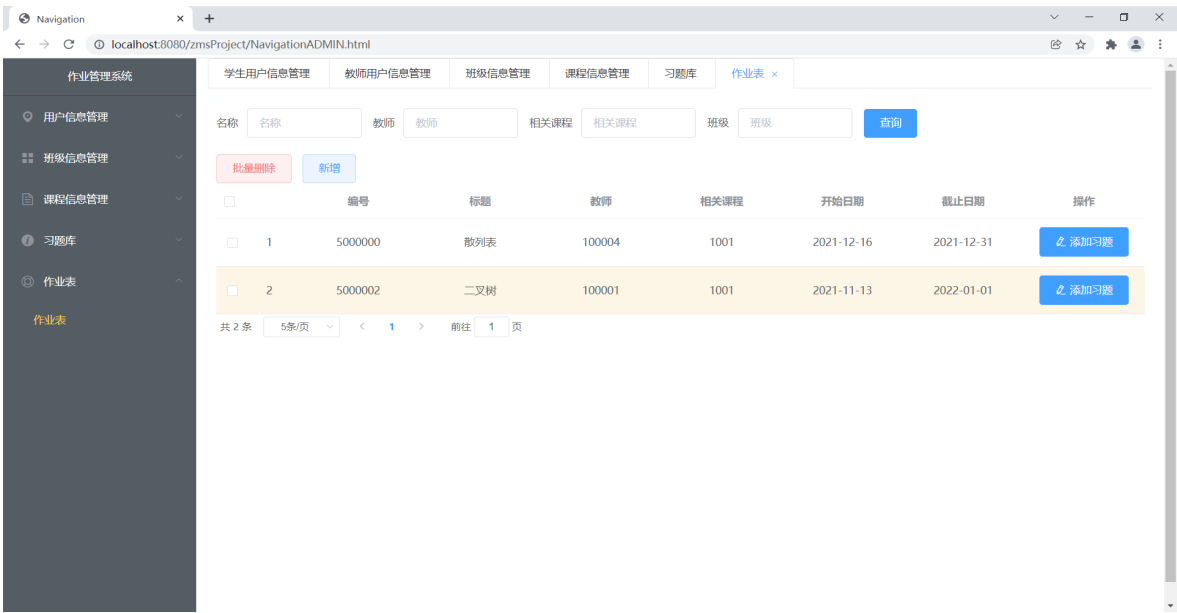
添加课程界面



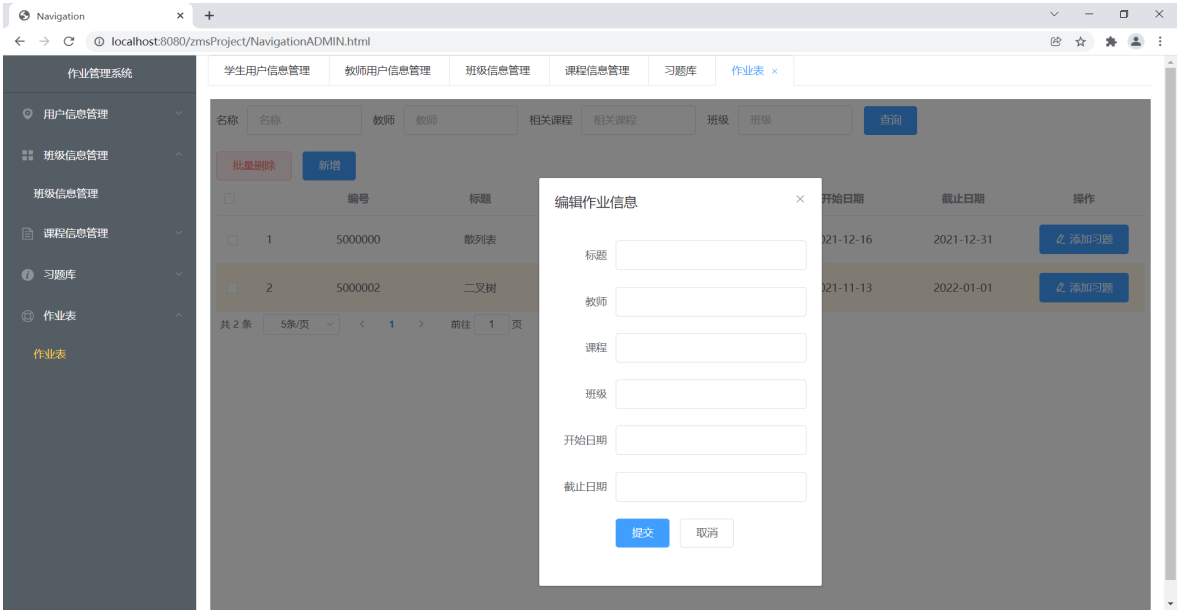
题库管理界面



添加习题界面



作业表管理界面



添加作业界面

Navigation

localhost:8080/zmsProject/NavigationTEACHER.html

作业管理系统

个人信息管理

账户信息

作业管理

个人信息

学号/职工号:

用户姓名:

学院:

专业:

班级:

确认修改

修改密码

* 原密码:

* 新密码:

* 确认密码:

确认修改

教师个人账户界面

Navigation

localhost:8080/zmsProject/NavigationTEACHER.html

作业管理系统

个人信息管理

作业管理

习题库

作业布置

习题id

课程名称

习题类型

课程名称

出题人

课程名称

习题科目

课程名称

查询

批量删除

新增

<input type="checkbox"/>	习题编号	习题类型	习题分数	习题内容	习题答案	出题人	习题科目	
<input type="checkbox"/>	1	1000000	1	3	如果n阶方阵A的行列式 $ A =0$,则下列正确的是() A. $A=0$ B. $r(A)>0$ C. $r(A)<n$ D. $r(A)=0$	D	100000	1002

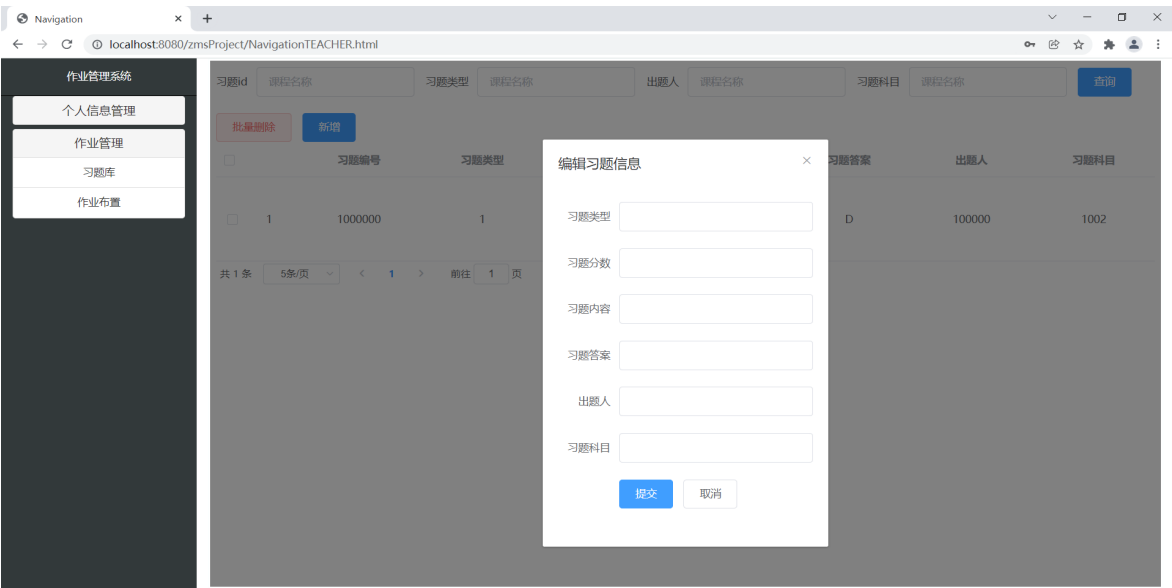
共 1 条

5条/页

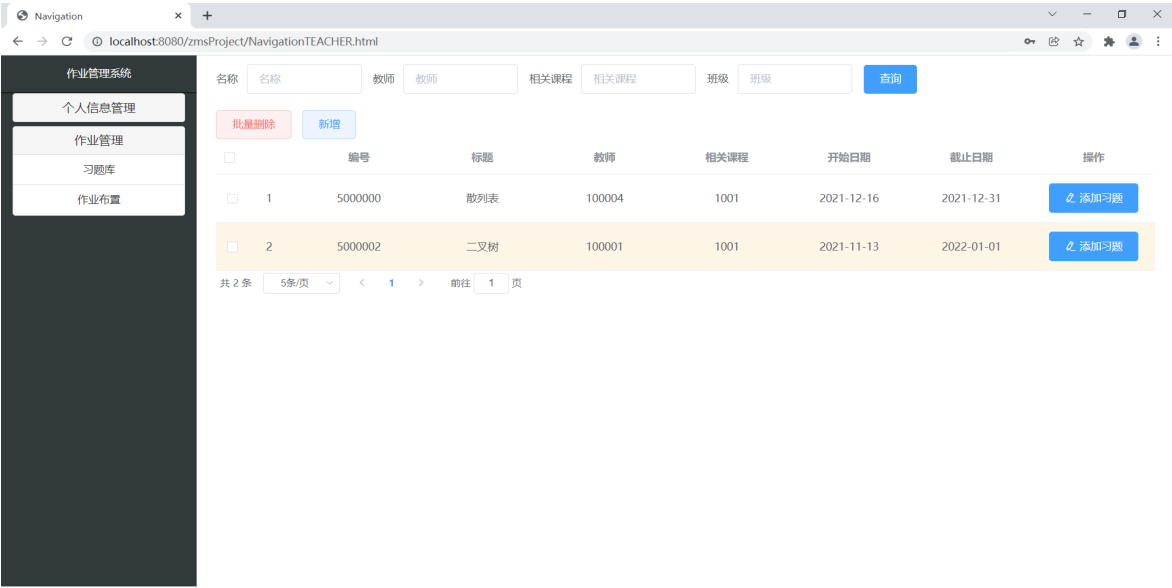
< 1 >

前往 1 页

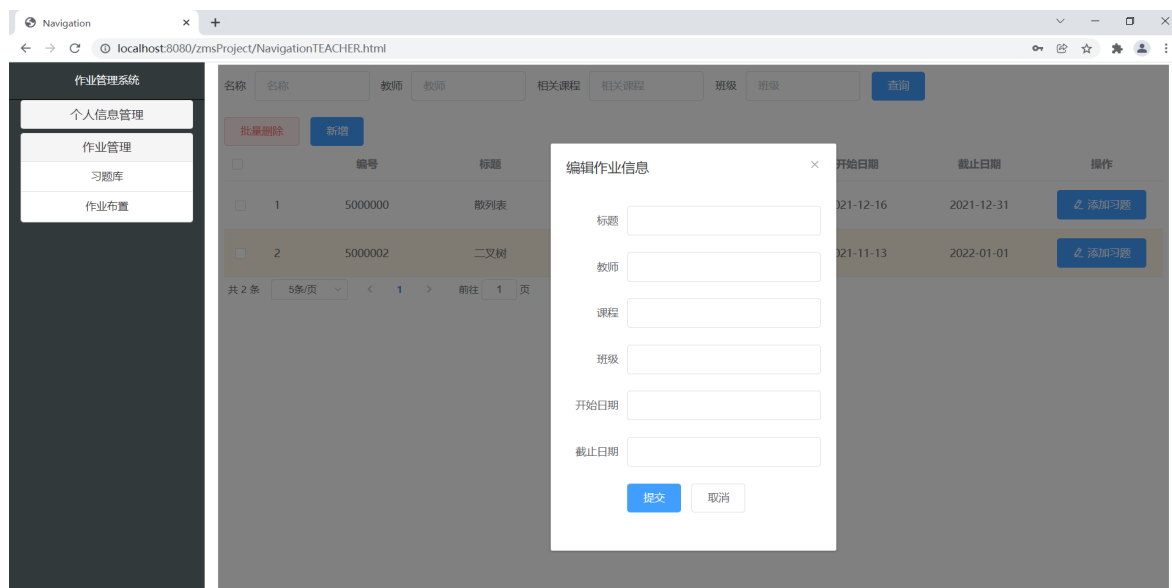
教师查看习题库界面



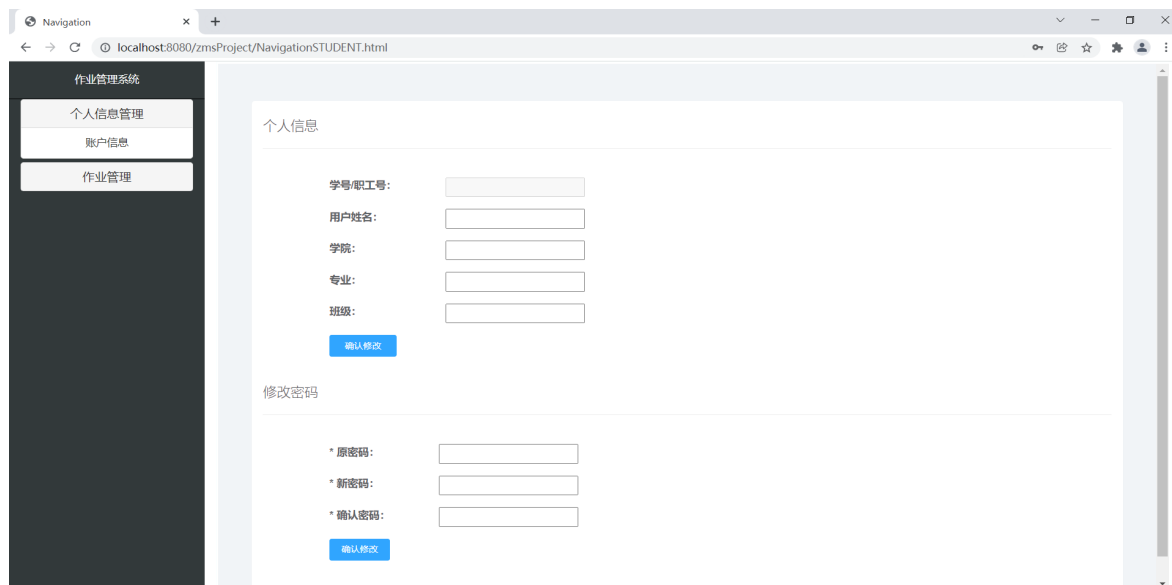
教师添加习题界面



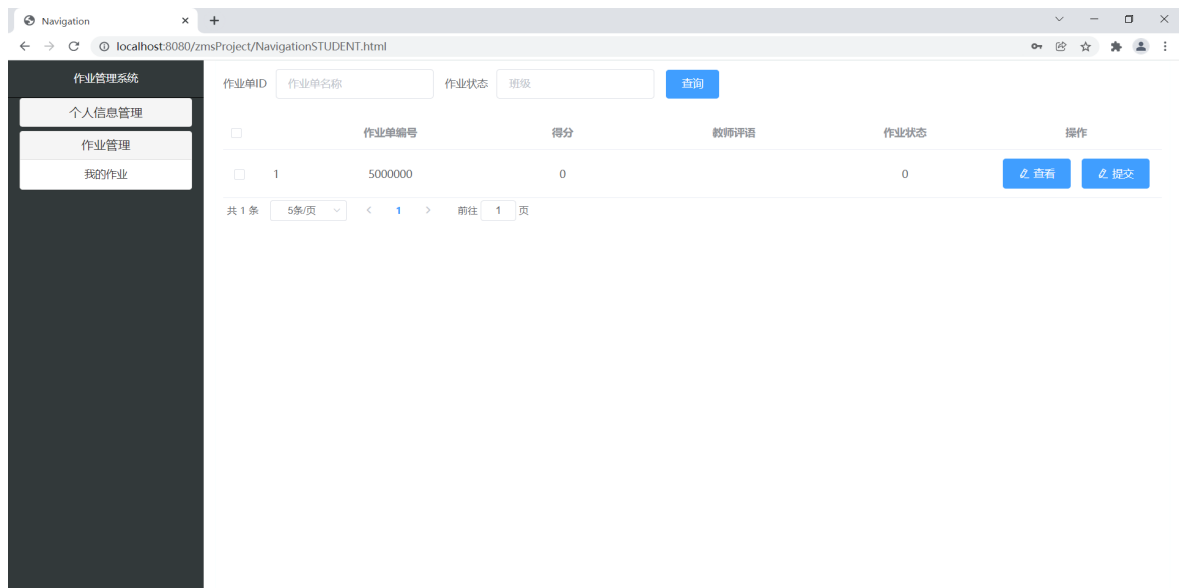
教师查看作业单界面



教师添加作业单界面



学生个人账户界面



学生查看个人作业界面

九. 总结

9.1 系统优点

1. 利用 axios 发送异步请求, 使页面不需刷新, 即可进行局部更新数据 2. 尽量是一个表只存储一个对象的信息, 并减少传递依赖, 使其尽量满足了 3NF 范式 3. 在对数据进行更改时, 增加提示框防止进行误操作 4. 界面美观, 软件易用性高, 操作简单, 能够很好的满足需求 5. 数据库设计内容具体详细, 条理清晰, 关系明确, 能够遵循数据库设计的准则来描述信息关系, 可以稳定地为系统提供服务。6. 系统设计全面、功能齐全, 在有限时间内将需求的功能完全实现。系统分为学生信息, 教师信息, 班级信息, 习题信息, 作业信息五个主要板块, 设计十分清晰明快能够很好的满足实际操作需求。7. 对异常进行了处理, 对错误操作给予提示, 例如登陆界面的表单验证 8. 利用各种接口进行解耦, 使项目耦合度较低, 利于维护和修改

9.2 系统不足

1. 在添加习题, 添加作业单等对话框中需要对其它实体中的元组 id 进行选择, 这里采用的是输入 id 的方式, 较为繁琐麻烦 2. 在显示其它实体中的依赖信息时显示其 id, 较为晦涩难懂, 对用户不友好 3. 登录界面在登陆失败时会失败信息输出到 servlet 界面中, 不简洁

9.3 系统改进

1. 利用 select 选择器对添加信息对话框中存在外码依赖的部分进行优化, 利于用户进行选择 2. 利用函数查询数据库, 将对应地 id 转化为其名称进行展示 3. 在登陆时, 利用 axios 发送异步请求到 servlet 中后对没有查询到对应数据的情况增加一个函数, 利用其打印登陆失败信息会 html 中

9.4 经验与收获

通过这次课程设计发现这其中需要的很多知识我们没有接触过，去图书馆查资料的时候发现我们前边所学到的仅仅是皮毛，还有很多需要我们掌握的东西我们根本不知道。同时也发现有很多已经学过的东西我们没有理解到位，不能灵活运用于实际，不能很好的用来解决问题，这就需要我们不断的大量的实践，通过不断的自学，不断地发现问题，思考问题，进而解决问题。在这个过程中我们将深刻理解所学知识，同时也可以学到不少很实用的东西。从各种文档的阅读到开始的需求分析、概念结构设计、逻辑结构设计、物理结构设计。亲身体验了一回系统的设计开发过程。很多东西书上写的很清楚，貌似看着也很简单，思路非常清晰。但真正需要自己想办法去设计一个系统的时候才发现其中的难度。经常做到后面突然就发现自己一开始的设计有问题，然后又回去翻工，在各种反复中不断完善自己的想法。

附. 参考文献

References

- [1] 基于关系数据库的关键词查询 [J]. 林子雨, 杨冬青, 王腾蛟, 张东站. 软件学报. 2010(10)
- [2] SQL Server 数据库应用系统中数据完整性的设计与实施 [J]. 刘敏贤. 航空计算技术. 2002(02)
- [3] XML 的关系数据库技术与应用 [J]. 骆炎民, 张全伙. 华侨大学学报 (自然科学版). 2003(03)
- [4] 基于 SQL Server 视图的数据库安全模型的研究 [J]. 陆静平, 何玉林. 计算机工程与应用. 2002(09)
- [5] 孙卫琴. Tomcat 与 Java Web 开发技术详解 (第 2 版)[M]. 北京: 电子工业出版社, 2009.
- [6] 毕建信. 基于 MVC 设计模式 WEB 应用研究与实现 [D]. 武汉: 武汉理工大学, 2006.
- [7] 王国辉. Java Web 编程宝典: 十年典藏版 [M]. 北京: 人民邮电出版社, 2011.
- [8] 赵利庆. Java Web 架构中数据库优化模式的研究与实现 [D]. 北京: 北京邮电大学, 2015.