**重庆交通大学信息科学与工程学院**

**课程设计报告**

**(成员设计报告)**

|  |  |
| --- | --- |
| 题 目 | 交通运输人员就业与再就业教育培训平台设计与实现 |
| 课程名称 | 企业级软件系统开发实践 |
| 专业班级 | 计算机科学与技术20级2班 |
| 学 号 | 632007060519 |
| 姓 名 | 张峰银 |
| 指导教师 | 梁宗保 |

2023 年 2 月

《企业级软件系统开发实践》答辩评分标准

|  |  |  |  |
| --- | --- | --- | --- |
| **序号** | **项目** | **评分标准与等级** | **评分等级** |
| 1 | 设计解决方案及实现60% | A、根据建立的系统逻辑模型，从多个角度合理的设计了架构，功能设计合理、数据模型设计合理；并根据设计的架构，实现的功能完善，测试充分，系统运行正确，人机接口界面友好。 |  |
| B、根据建立的系统逻辑模型，进行了架构设计，功能设计合理、数据模型设计合理；并根据设计的架构，实现功能较为完善，测试合理，系统运行较正确，人机接口界面友好。 |  |
| C、根据建立的系统逻辑模型，有架构设计，功能设计基本合理、数据模型设计基本合理；能实现分工的功能，系统能运行基本正确，能进行单元测试和集成测试，有基本的人机接口界面。 |  |
| D、能设计架构，功能分解基本合理，数据模型设计基本合理，能完成基本数据的增、删、改、查功能；有基本的测试，系统能运行。 |  |
| E、未进行架构设计，功能分解不合理，数据模型设计不合理，没有实现分工的功能，或整个设计抄袭。 |  |
| 2 | 使用现代工具20% | A、在可行性分析和需求分析阶段，根据需要选择了合适的企业常用分析工具；在设计阶段选择了合适的、稳定的、目前企业常用的开发框架。 |  |
| B、在可行性分析和需求分析阶段，根据需要能选择较合适的企业常用分析工具；在设计阶段选择了满足本项目的开发要求的开发框架。 |  |
| C、在可行性分析和需求分析阶段，根据需要能选择除OFFICE外的分析工具；在设计阶段能选择开发框架进行开发。 |  |
| D、在可行性分析和需求分析阶段，主要采用OFFICE工具记性分析；在设计阶段没有选择使用开发框架进行。 |  |
| E、未完成可行性分析、需求分析和设计，或者抄袭。 |  |
| 3 | 团队协作（20%） | A、独立完成了分工任务的可行性和需求分析、架构设计和实现，分析设计完善，并能有机的集成到整个系统；个人工作量饱满，个人设计能很好地与其它功能协作工作。 |  |
| B、独立完成了分工任务的可行性和需求分析、架构设计和实现，分析设计合理，并能有机的集成到整个系统；个人工作量较饱满，个人设计能很好地与其它功能协作工作。 |  |
| C、独立完成了分工任务的可行性和需求分析、架构设计和实现，并能集成到整个系统；个人工作量适中，个人设计能与其它功能协作工作。 |  |
| D、能独立完成分工任务的可行性、需求分析、架构设计和实现，在协助下能集成到整个系统；个人工作量偏少，个人设计基本能与其它功能协作工作。 |  |
| E、未完成分工任务的需求分析、架构设计和实现或抄袭。 |  |
| **答辩总分**： | | |  |

## 成员设计报告评分标准

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 项目 | 分值 | 评分标准与等级 | 等级 | 得分 |
| 个  人  设  计  报  告 | 100 | A、个人设计文档内容详细、功能模块设计合理，单元测试和集成测试详细，格式规范。 |  |  |
| B、个人设计文档内容较为详细、功能模块设计较为合理，单元测试和集成测试较为详细，格式规范。 |  |
| C、个人设计文档内容较为详细、功能模块设计基本合理，能实现个人功能模块的测试，格式较规范。 |  |
| D、个人设计文档内容较为详细、功能模块设计基本合理，但存在30%以内条目书写不完全符合要求。有个人功能模块的测试，格式基本规范。 |  |
| E、报告不规范或存在30%以上条目书写不完全符合要求，或抄袭、复制别人程序与文档。 |  |
| 教师签字 | |  | 得分 |  |

注：1）答辩和成员设计报告质量分别按百分制打分；

2）综合成绩=过程表现\*0.1 + 团队设计报告\*0.2 + 成员设计报告\*0.2+项目答辩\*0.5

3）单边和评阅报告时在相应等级画“√”，分数换算为A\B\C\D\E等级分别计为95、85、75、65、30分

**目 录**

目录

[成员设计报告评分标准 2](#_Toc18414)

[第一章 功能设计列表 4](#_Toc783)

[第二章 功能模块设计 4](#_Toc1804)

[2.1 用户管理模块 4](#_Toc15913)

[2.1.1 用户管理模块功能描述 4](#_Toc26121)

[2.1.2 用户管理模块增删改查 4](#_Toc883)

[2.1.3 XXX模块设计流程 5](#_Toc13831)

[2.1.4 用户管理模块用户界面设计 5](#_Toc24578)

[2.1.5 接口说明 7](#_Toc13916)

[2.1.6调用方式 8](#_Toc10785)

[2.2 角色管理模块 9](#_Toc20291)

[2.2.1 角色管理模块功能描述 9](#_Toc28598)

[2.2.2 角色管理模块界面设计 10](#_Toc27462)

[2.2.3 接口设计 10](#_Toc31889)

[2.2.4 调用分析 11](#_Toc11437)

[2.3 考试管理模块 12](#_Toc24718)

[2.4 试题管理模块 13](#_Toc26555)

[2.5 机构管理模块 19](#_Toc23411)

[2.6 资料管理模块 21](#_Toc20355)

[第三章 测试 25](#_Toc1249)

[3.1 测试环境 25](#_Toc29325)

[3.2 功能模块测试 25](#_Toc30949)

[3.3 模块功能测试 25](#_Toc19895)

[3.4 用户管理模块的集成测试 26](#_Toc21986)

[3.5 角色管理模块的集成测试 27](#_Toc12010)

[3.6 考试管理模块的集成测试 28](#_Toc32488)

[3.7 试题管理模块的集成测试 29](#_Toc11928)

[3.8 机构管理模块的集成测试 30](#_Toc28092)

[3.9 资料管理模块的集成测试 30](#_Toc32295)

[第四章 谈谈你对项目管理的理解 31](#_Toc31253)

**第一章 功能设计列表**

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 模块编号 | 模块名称 | 功能模块简述 |
| 1 | 1 | 用户管理模块 | 负责用户的增删改查 |
| 2 | 2 | 角色管理模块 | 不同的用户拥有不同的角色，不同的角色拥有不同的功能 |
| 3 | 3 | 考试管理模块 | 管理员可以发布考试，为考试布置试卷 |
| 4 | 4 | 试题管理模块 | 可以为题库进行增删改查，也可以选择试题来组成试卷 |
| 5 | 5 | 机构管理模块 | 负责机构资料的增删改查 |
| 6 | 6 | 资料管理模块 | 管理员上传文件，视频等资料 |

**第二章 功能模块设计**

2.1 用户管理模块

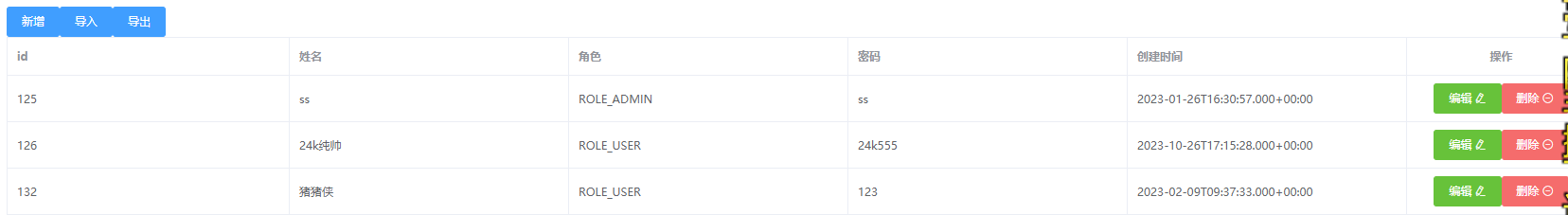
2.1.1 用户管理模块功能描述

可以对用户的信息进行增删改查

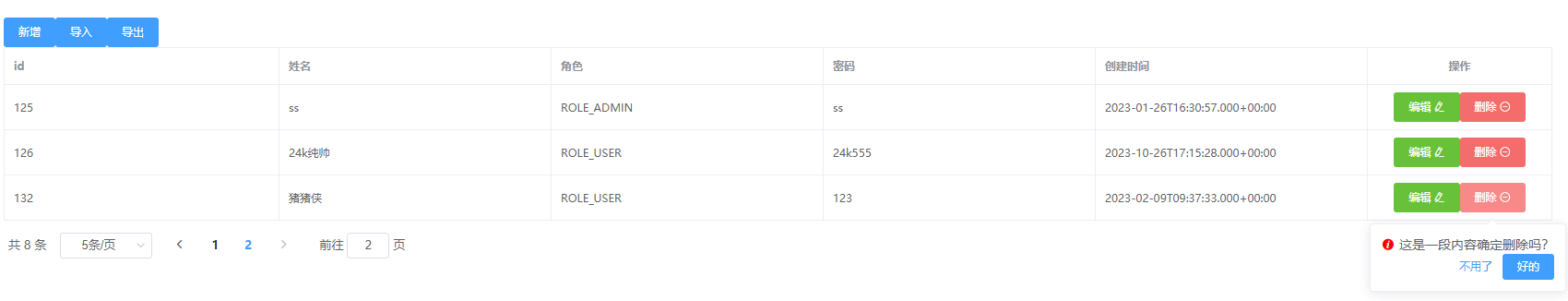
2.1.2 用户管理模块增删改查

管理员通过新增按钮来新增加一个用户，填写用户信息，并为用户分配角色。





点击删除按钮会提示是否删除用户，再次确认则可以删除该用户

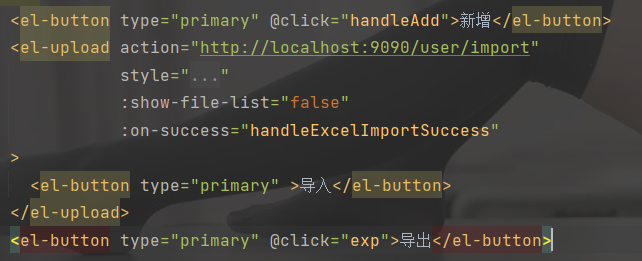


2.1.3 XXX模块设计流程

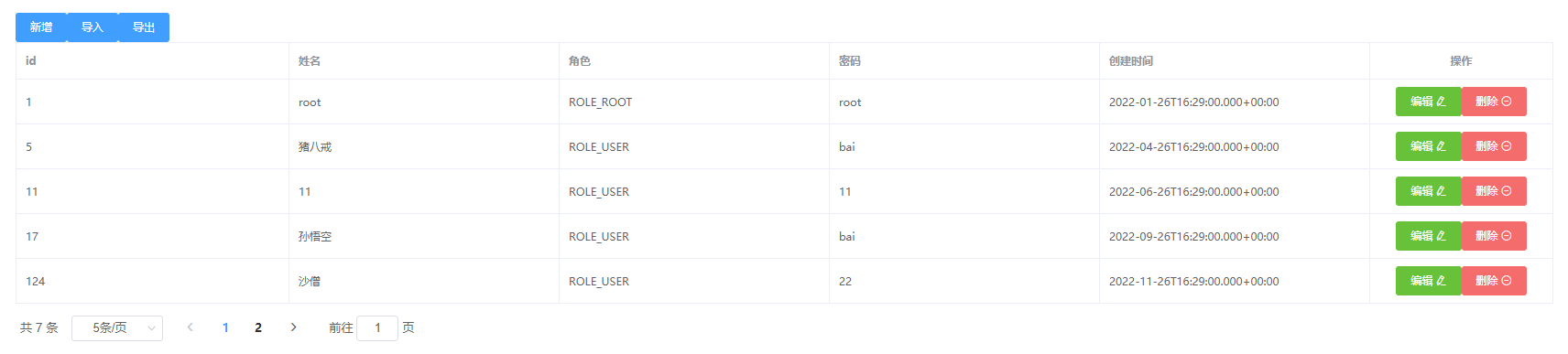
从业务角度详细描述根据输入数据产生输出数据的业务算法和流程。一般采用程序流程图描述实现的算法，并对实现算法进行描述

2.1.4 用户管理模块用户界面设计



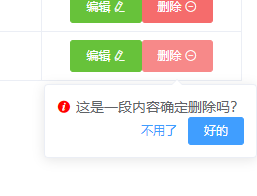


该用户界面中的按钮设计





该用户界面的表单设计





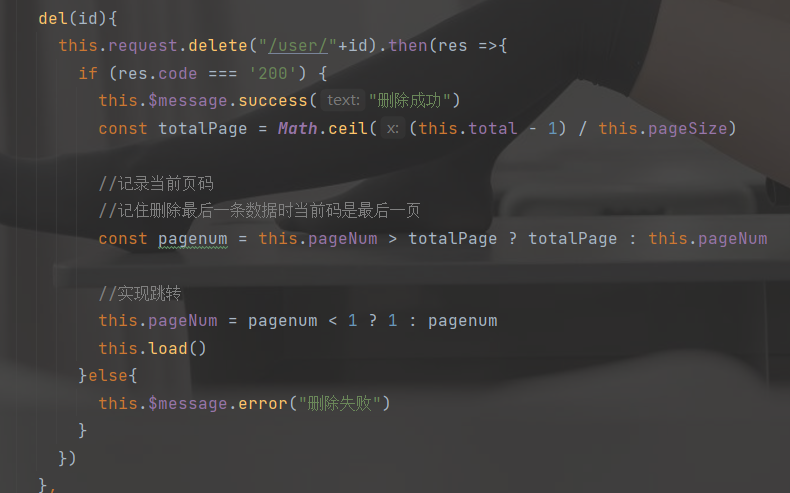
删除按钮提示删除确认设计

2.1.5 接口说明

增加或更新通过save()函数将创建的用户数据传给后台并保存到数据库之中，保存成功或失败都会在页面中显示对应的消息。



删除通过del()接口传入将要删除的用户id，在数据库中会找到该用户的id并将其删除



2.1.6调用方式

根据前端请求的接口路径，调用对应的方法

private IUserService userService;

//新增或更新

@PostMapping

public Result save(@RequestBody User user){

return Result.success(userService.saveOrUpdate(user));

}

@DeleteMapping("/{id}")

public Result delete(@PathVariable Integer id) {

return Result.success(userService.removeById(id));

}

//查询

@GetMapping

public Result findAll() {

return Result.success(userService.list());

}

@GetMapping("/username/{username}")

public Result findOne(@PathVariable String username) {

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

queryWrapper.eq("username",username);

return Result.success(userService.getOne(queryWrapper));

}

@GetMapping("/page")

public Result findPage(@RequestParam Integer pageNum,

@RequestParam Integer pageSize,

@RequestParam(defaultValue = "") String name) {

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

queryWrapper.like("username",name);

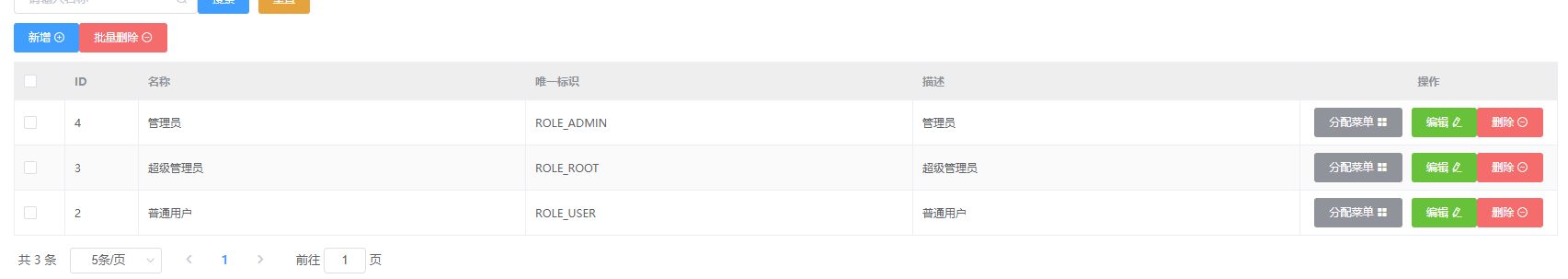
return Result.success(userService.page(new Page<>(pageNum, pageSize),queryWrapper));

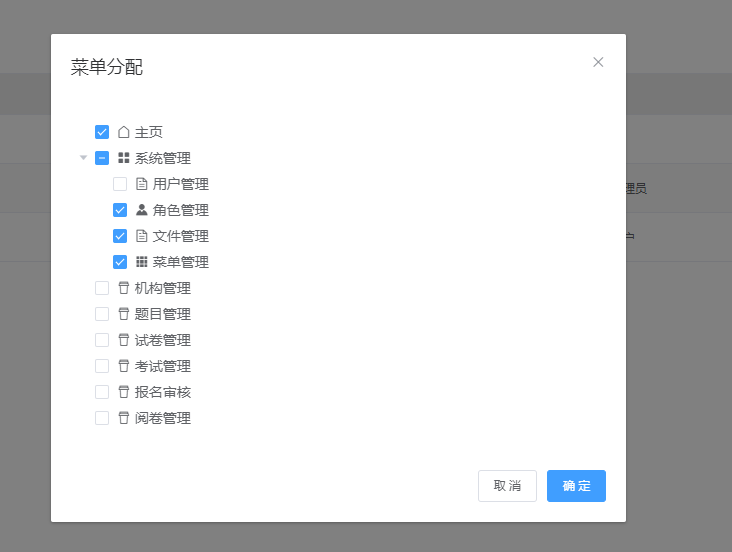
}

2.2 角色管理模块

2.2.1 角色管理模块功能描述

为系统创建不同的角色，分配不同的权限，管理不同的功能





2.2.2 角色管理模块界面设计



2.2.3 接口设计

将角色信息传入后端，在后端获取所有的菜单，根据角色信息提供他权限以内的功能菜单



2.2.4 调用分析

将角色id和菜单id一一对应，传入后台数据库中储存。调用时从数据库中取出角色对应的菜单id并显示。

@PostMapping("/roleMenu/{roleId}")

public Result roleMenu(@PathVariable Integer roleId, @RequestBody List<Integer> menuIds) {

roleService.setRoleMenu(roleId, menuIds);

return Result.success();

}

@GetMapping("/roleMenu/{roleId}")

public Result getRoleMenu(@PathVariable Integer roleId) {

return Result.success( roleService.getRoleMenu(roleId));

}

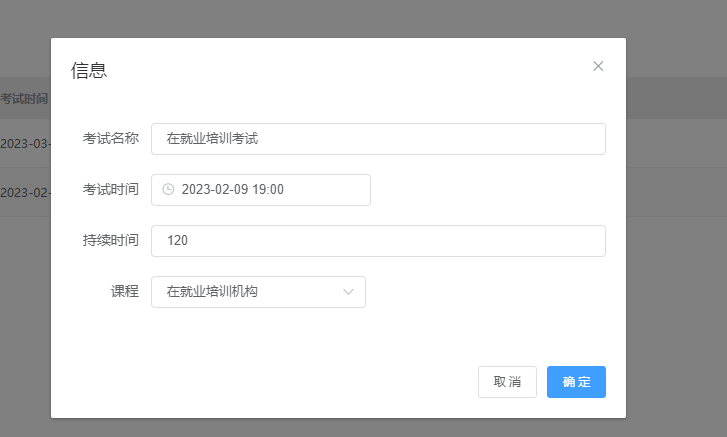
2.3 考试管理模块

2.3.1 考试管理功能功能描述

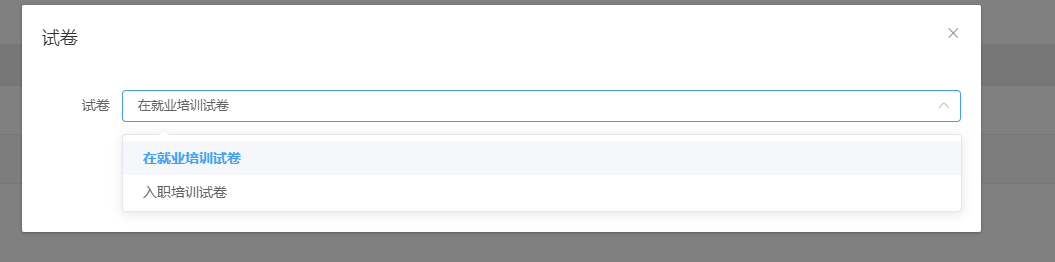
管理员可以发布考试，学员在前端就会看到发布的考试信息，学生报名参加后，需要管理员审核通过后才可以参加考试。

2.3.2 考试管理功能演示

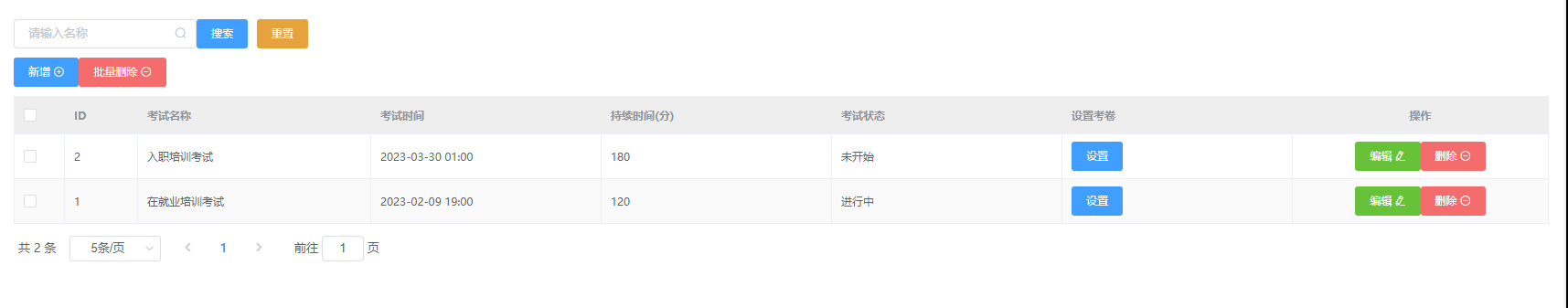
发布考试，填写考试相关信息



为考试配置对应的考卷

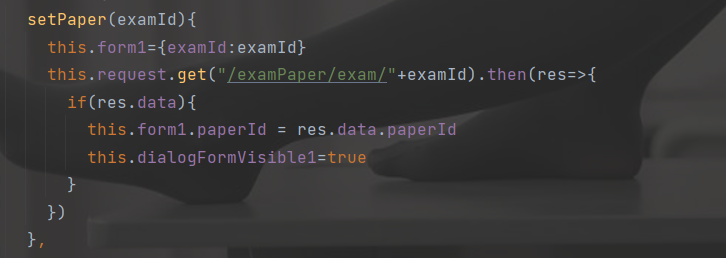


2.3.4 考试管理功能界面设计



2.3.5 接口设计

将考试id传入后台，获取数据库中对应机构的试卷，将其分配给此次考试



2.3.6 调用分析

根据前端传入的考试id，在数据库中查询过后，将其结果返回

@GetMapping("/exam/{examId}")

public Result findByExam(@PathVariable Integer examId) {

return Result.success(examPaperService.getOne(new QueryWrapper<ExamPaper>().eq("exam\_id",examId)));

}

2.4 试题管理模块

2.4.1 试题管理功能功能描述

可以为题库进行增删改查，也可以选择试题来组成试卷

2.4.2 试题管理功能演示

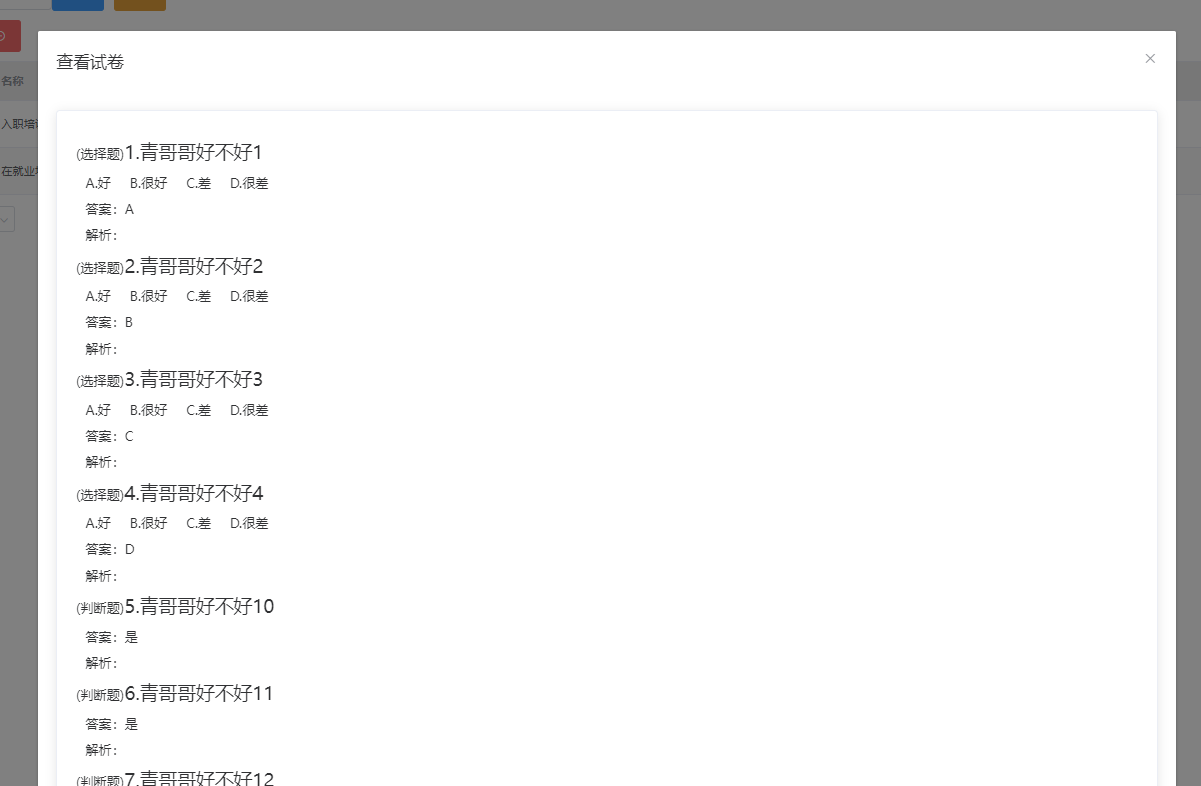
可以对试题进行基础的增删改查



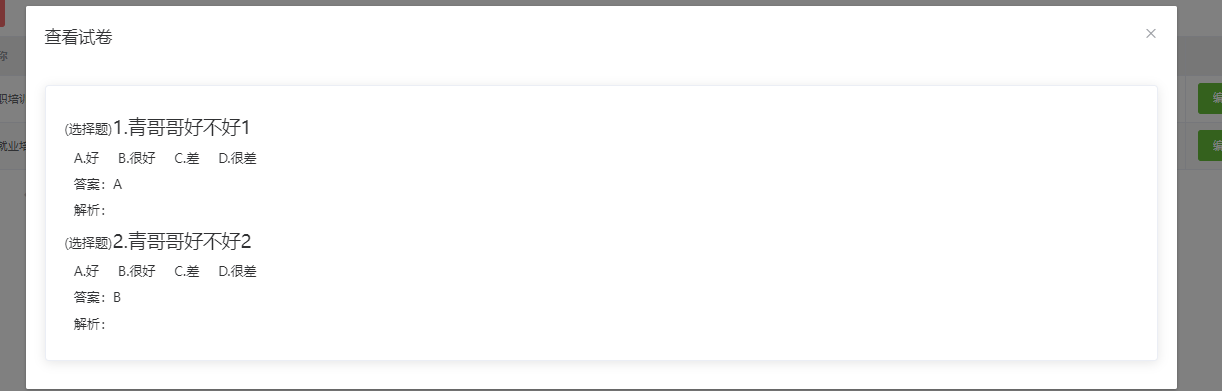


也可以为题库增加，删除试卷，有自动组卷和手动组卷两个模式可供选择









2.4.4 试题管理功能界面设计





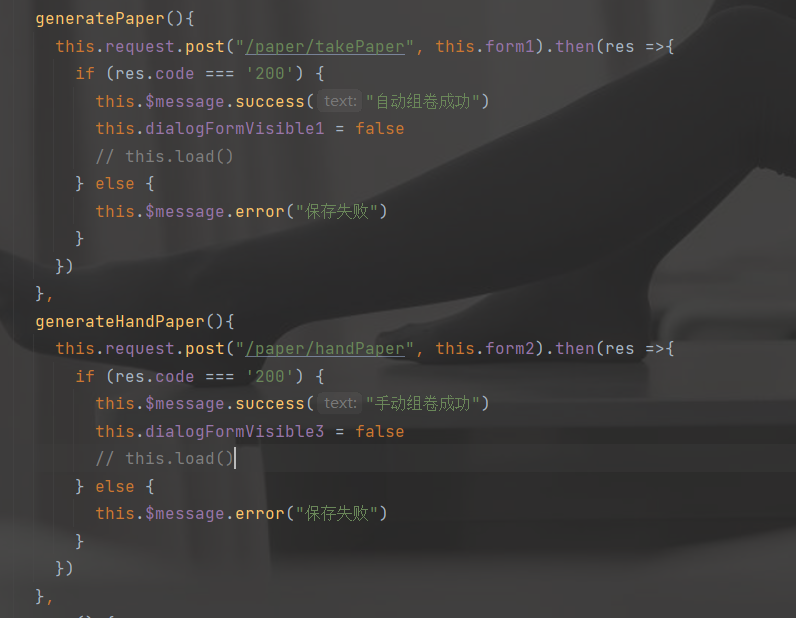
2.4.5 接口设计

查询数据库中的题目信息，并将其呈现在界面上



自动组卷将前端选好的各种类型的题目数量传到后台，由后台产生随机数，取出题目组成试卷。

手动组卷将所有题目获取到前端，选择题目后将题目id记录下，存储到试卷中，组成试卷



2.4.6 调用分析

@PostMapping("/takePaper")

public Result takePaper(@RequestBody PaperDto paperDto) {

//删除老的试卷

UpdateWrapper<PaperQuestion> updateWrapper=new UpdateWrapper<>();

updateWrapper.eq("paper\_id",paperDto.getPaperId());

paperQuestionService.remove(updateWrapper);

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

queryWrapper.eq("course\_id",paperDto.getCourseId());

//根据课程id查出所有该课程的题目，再根据type区分

List<Question> questionList = questionService.list(queryWrapper);

List<Question> type1List = questionList.stream().filter(question -> question.getType()==1).collect(Collectors.toList()); //选择

List<Question> type2List = questionList.stream().filter(question -> question.getType()==2).collect(Collectors.toList()); //判断

List<Question> type3List = questionList.stream().filter(question -> question.getType()==3).collect(Collectors.toList()); //问答

if(type1List.size()<paperDto.getType1()){

throw new ServiceException("-1","选择题数量不足");

}

if(type2List.size()<paperDto.getType2()){

throw new ServiceException("-1","判断题数量不足");

}

if(type3List.size()<paperDto.getType3()){

throw new ServiceException("-1","问答题数量不足");

}

//开始随机组卷

List<PaperQuestion> paperQuestion =getPaperQuestion(type1List.size(),paperDto.getType1(),type1List,paperDto.getPaperId());

paperQuestion.addAll(getPaperQuestion(type2List.size(),paperDto.getType2(),type2List,paperDto.getPaperId()));

paperQuestion.addAll(getPaperQuestion(type3List.size(),paperDto.getType3(),type3List,paperDto.getPaperId()));

paperQuestionService.saveBatch(paperQuestion);

// paperService.saveOrUpdate(paperDto);

return Result.success();

}

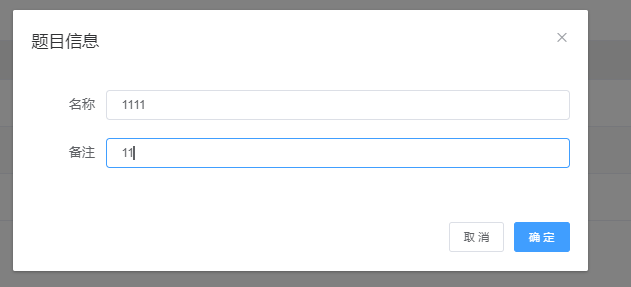
2.5 机构管理模块

2.5.1 机构管理功能描述

负责机构的增删改查

2.5.2 机构管理功能演示

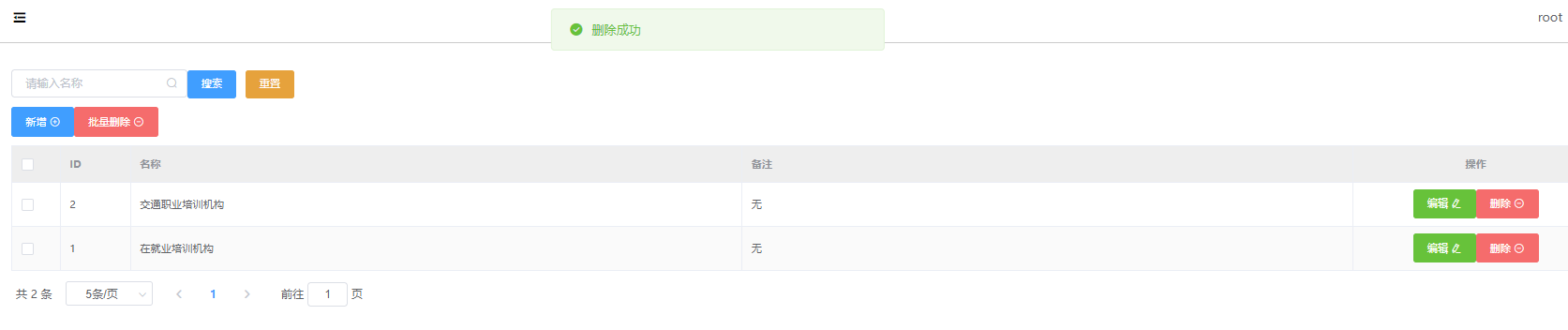
点击新增按钮来填写机构信息，确认后将机构信息存储到数据库里



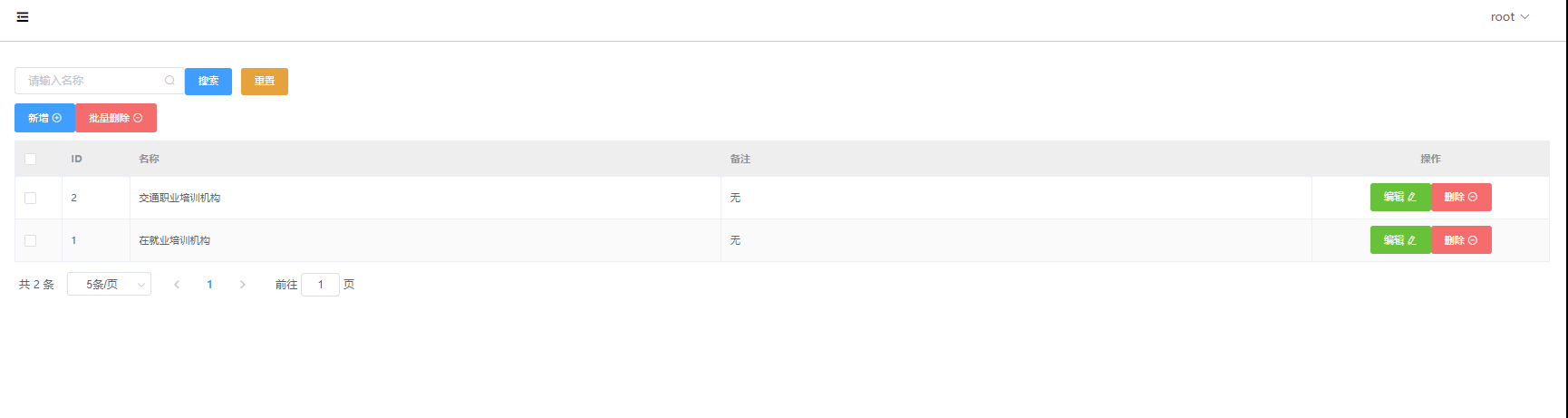


同样也可以对机构进行基础的增删改查





2.5.4 机构管理功能界面设计



2.5.5 接口设计





2.5.6 调用分析

将前端表单数据传到后台，后台将数据存入数据库中

// 新增或者更新

@PostMapping

public Result save(@RequestBody Course course) {

courseService.saveOrUpdate(course);

return Result.success();

}

@DeleteMapping("/{id}")

public Result delete(@PathVariable Integer id) {

courseService.removeById(id);

return Result.success();

}

@PostMapping("/del/batch")

public Result deleteBatch(@RequestBody List<Integer> ids) {

courseService.removeByIds(ids);

return Result.success();

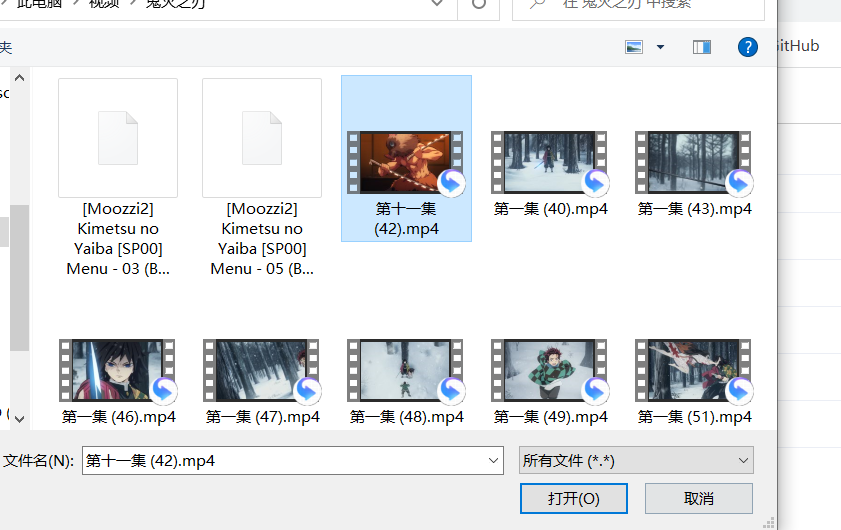
2.6 资料管理模块

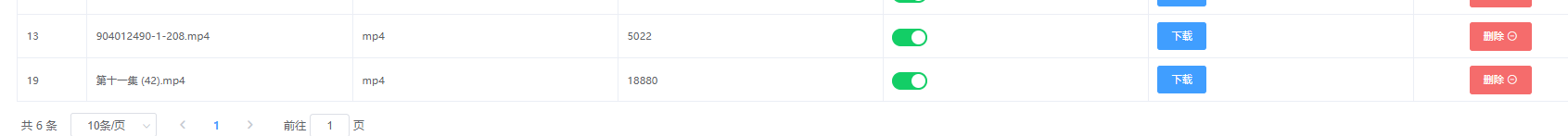
2.6.1 资料管理功能描述

管理员上传文件，视频等资料

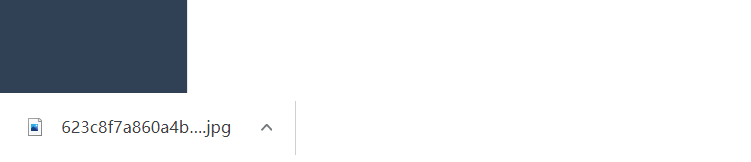
2.6.2 资料管理功能演示

点击上传文件，选择文件后可以进行上传

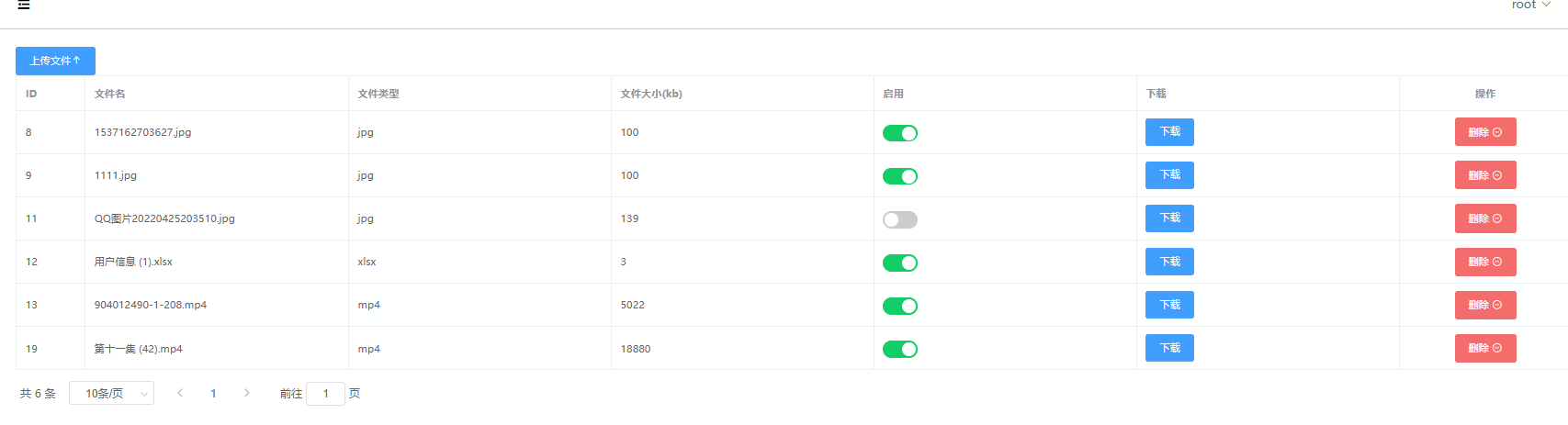




也可以通过下载按钮来在线获取文件，可以对文件进行删除



2.6.4 资料管理功能界面设计



2.6.5 接口设计

从数据库中请求存储的文件信息数据，将其展现给管理员，方便管理员查看



2.6.6 调用分析

选择文件上传后，先将文件存到磁盘，然后判断配置的文件目录是否存在，如果不存在则为其分配唯一的md5标识，将其信息存入数据库保存

@PostMapping("/upload")

public String upload(@RequestParam MultipartFile file) throws IOException {

String originalFilename =file.getOriginalFilename();

String type= FileUtil.extName(originalFilename);

long size =file.getSize();

//先存到磁盘

File uploadParentFile = new File(fileUploadPath);

//判断配置的文件目录是否存在

if(!uploadParentFile.exists()){

uploadParentFile.mkdir();

}

String uuid =IdUtil.fastSimpleUUID();

String fileUuid=uuid+ StrUtil.DOT+type;

String md5,url;

File uploadFile=new File(fileUploadPath+fileUuid);

if(uploadFile.exists()){

//获取文件的md5

md5= SecureUtil.md5(uploadFile);

//从数据库查询是否存在相同的记录

Files dbFiles=getFileByMd5(md5);

if(dbFiles!=null){

url=dbFiles.getUrl();

}else {

//获取的文件存储到磁盘文件

file.transferTo(uploadFile);

url="http://localhost:9090/file/"+fileUuid;

}

}else{

//获取的文件存储到磁盘文件

file.transferTo(uploadFile);

md5=SecureUtil.md5(uploadFile);

url="http://localhost:9090/file/"+fileUuid;

}

//存储数据库

Files saveFile=new Files();

saveFile.setName(originalFilename);

saveFile.setType(type);

saveFile.setSize(size/1024);

saveFile.setUrl(url);

saveFile.setMd5(md5);

fileMapper.insert(saveFile);

return url;

}

@GetMapping("/{fileUuid}")

public void download(@PathVariable String fileUuid, HttpServletResponse response) throws IOException {

File uploadFile =new File(fileUploadPath + fileUuid);

ServletOutputStream os = response.getOutputStream();

URLEncoder urlEncoder=new URLEncoder();

response.addHeader("Content-Disposition","attachment;filename=" + urlEncoder.encode(fileUuid, StandardCharsets.UTF\_8));

response.setContentType("application/octet-stream");

os.write(FileUtil.readBytes(uploadFile));

os.flush();

os.close();

}

**第三章 测试**

3.1 测试环境

3.2 功能模块测试

采用路径测试法，对实现的每个功能模块的所有执行路径进行测试。

3.3 模块功能测试

测试对象的功能测试应该侧重于可以被直接追踪到用例或业务功能和业务规则的所有测试需求。这些测试的目标在于核实能否正确地接受、处理和检索数据以及业务规则是否正确实施。这种类型的测试基于黑盒方法，即通过图形用户界面 (GUI) 与应用程序交互并分析输出结果来验证应用程序及其内部进程。以下列出的是每个应用程序推荐的测试方法概要：

|  |  |
| --- | --- |
| 测试目标： | 确保测试对象的功能正常，其中包括导航、数据输入、处理和检索等。 |
| 方法： | 利用有效的和无效的数据来执行各个用例、用例流或功能，以核实以下内容：   在使用有效数据时得到预期的结果。   在使用无效数据时显示相应的错误消息或警告消息。   各业务规则都得到了正确的应用。 |
| 完成标准： |  所计划的测试已全部执行。   所发现的缺陷已全部解决。 |
| 需考虑的特殊事项： | 确定或说明那些将对功能测试的实施和执行造成影响的事项或因素（内部的或外部的） |

3.4 用户管理模块的集成测试

点击新增用户



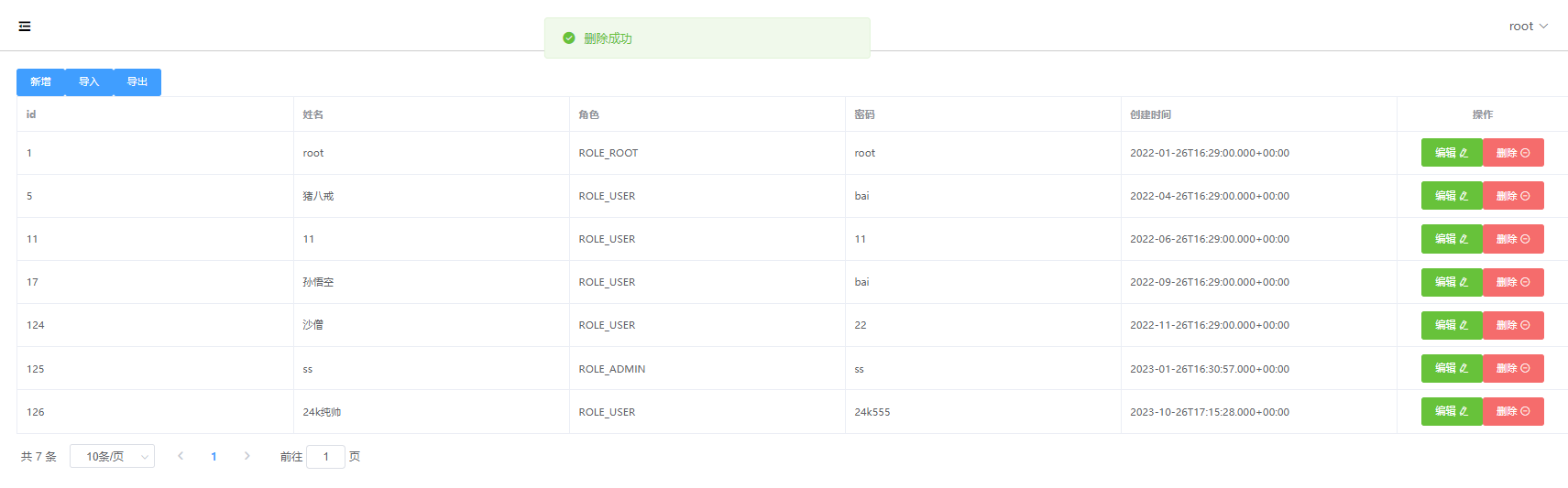
新增用户成功



编辑用户信息成功

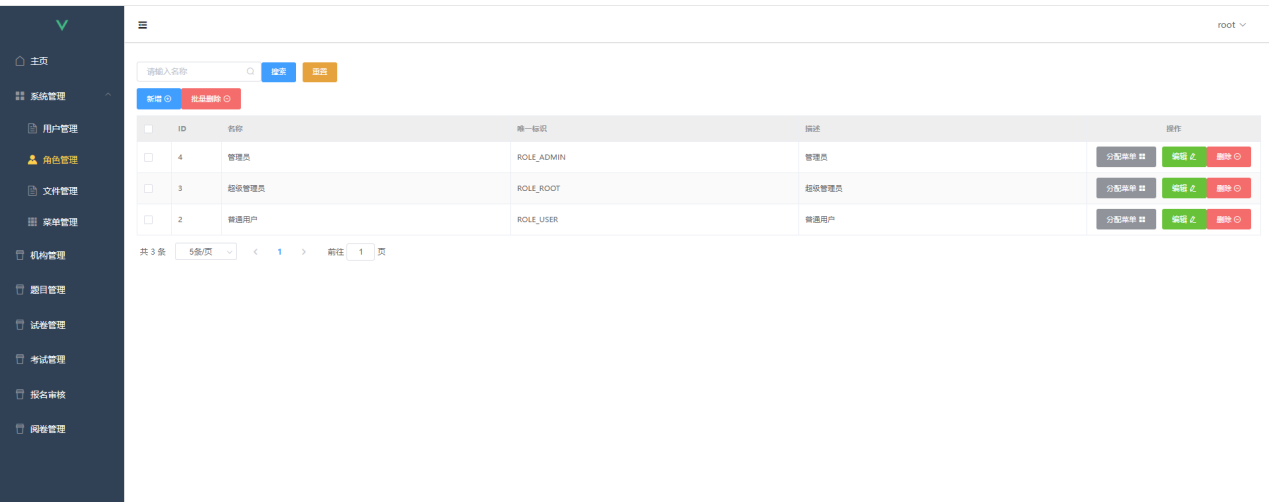


删除用户成功



3.5 角色管理模块的集成测试

测试角色的菜单分配功能





3.6 考试管理模块的集成测试

发布考试信息

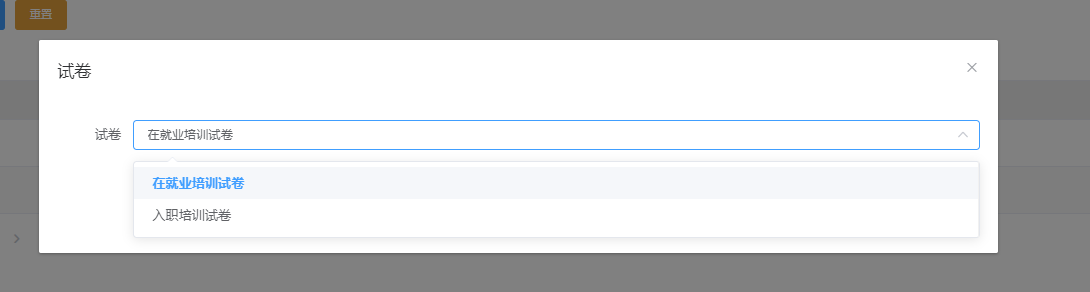




删除考试信息



为考试设置试卷



3.7 试题管理模块的集成测试

新增加一道题目





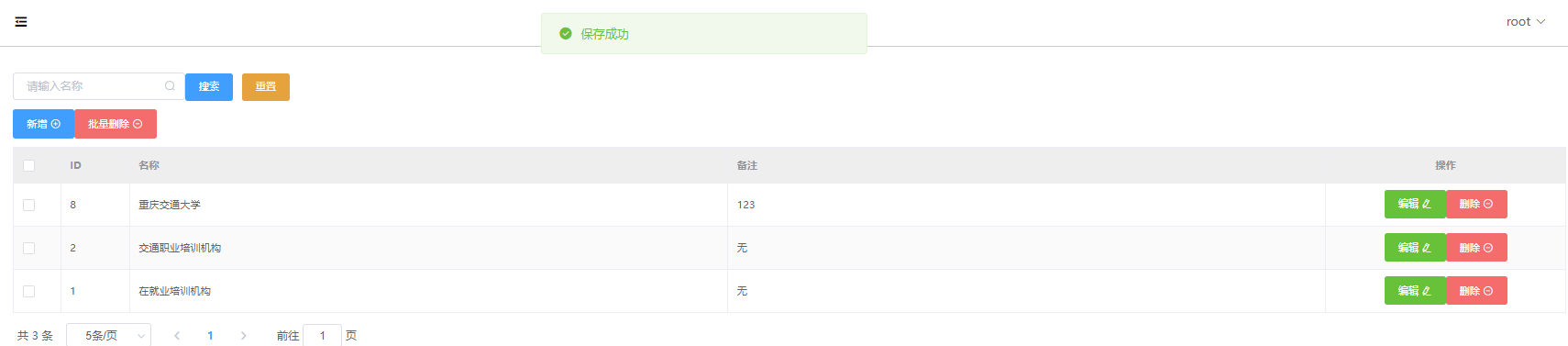
删除题目



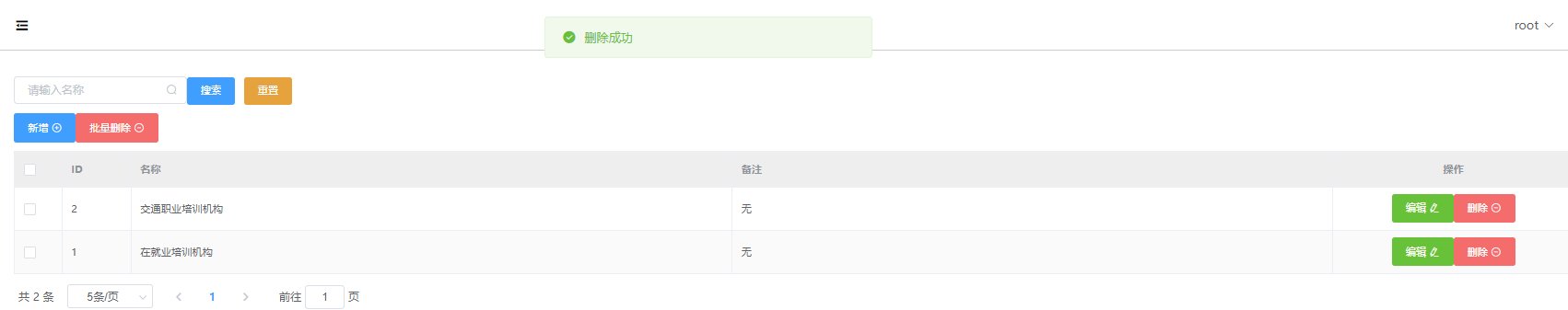
3.8 机构管理模块的集成测试

新增加一个机构



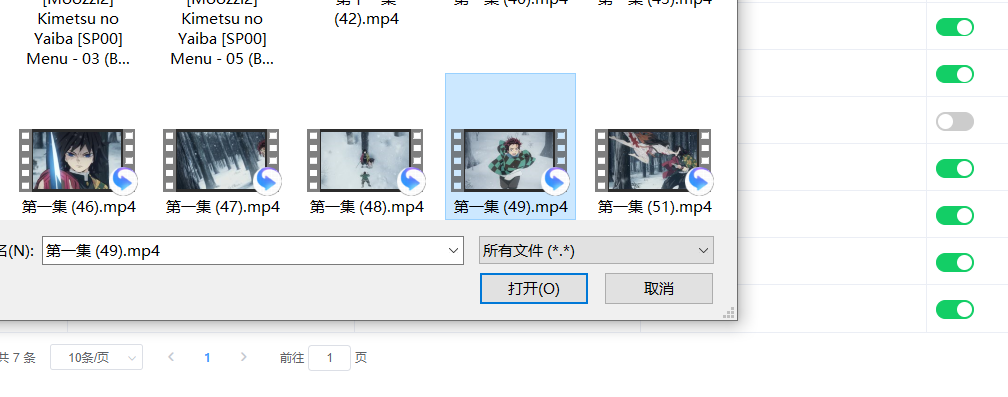


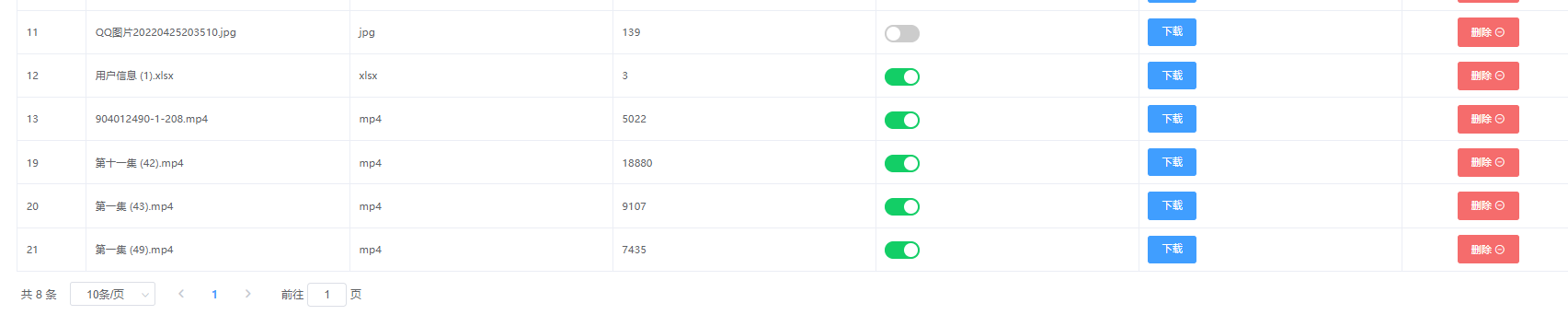
删除机构



3.9 资料管理模块的集成测试

上传视频资料





删除资料



**第四章 谈谈你对项目管理的理解**

项目管理是为了让项目井然有序的进行。项目管理是指把各种系统、方法和人员结合在一起,在规定的时间、预算和质量目标范围内完成项目的各项工作。即从项目的投资决策开始到项目结束的全过程进行计划、组织、指挥、协调、控制和评价，以实现项目的目标。在项目管理方法论上主要有：阶段化管理、量化管理和优化管理三个方面。

在本次项目实践中，我们首先对项目总体做了基础的需求分析，明确了项目需要用到的功能，然后通过学习、查阅资料来实现这些功能。本次项目中，我们使用了spring boot+vue的基础框架，引入了element-ui组件。在后续开发过程中，为了简化手写代码的繁杂工作，还使用了代码生成器。

开发过程中难免会遇到问题，不过我们会积极的与队友进行沟通交流，寻求解决问题的方法。也会在百度、谷歌等上面搜寻解决问题的途径。

在开发项目过程中，学会了不少的新知识，也进一步了解了项目前后端分离的开发流程，让我们感觉获益良多。