

实 训 手 册

实习类型： 生产实训

学 院： 软件学院

专 业： 软件工程

班 级： 18级软件测试班

学 号： 201826704019

姓 名： 王君

指导教师： 张鹏飞

实习单位： 软通动力技术服务有限公司

**目录**

[一、项目名称 11](#_Toc80803785)

[二、意义背景 11](#_Toc80803786)

[三、需求分析 11](#_Toc80803787)

[四、功能设计 13](#_Toc80803788)

[4.1 概要设计 13](#_Toc80803789)

[4.1.1 功能概要 13](#_Toc80803790)

[4.1.2 系统整体功能流程图 15](#_Toc80803791)

[4.1.3 系统数据流图 16](#_Toc80803792)

[4.2 详细设计 16](#_Toc80803793)

[4.2.1登录注册 16](#_Toc80803794)

[4.2.2 主界面 17](#_Toc80803795)

[五、 数据库设计 19](#_Toc80803796)

[5.1 数据库设计概述 19](#_Toc80803797)

[5.2 数据库逻辑结构设计 19](#_Toc80803798)

[5.3 数据库物理结构设计 19](#_Toc80803799)

[5.4 数据库安全设计 22](#_Toc80803800)

[5.5 数据表结构 23](#_Toc80803801)

[六、 亮点设计 25](#_Toc80803802)

[6.1 用户密码加密处理 25](#_Toc80803803)

[6.2 手机号+验证码登录 26](#_Toc80803804)

[6.3 回收站 27](#_Toc80803805)

[6.3 分享文件 28](#_Toc80803806)

[6.4 群组管理 29](#_Toc80803807)

[6.5 签到管理 31](#_Toc80803808)

[七、 运行截图 32](#_Toc80803809)

[7.1 登录注册 32](#_Toc80803810)

[7.2 首页个人账号操作 35](#_Toc80803811)

[7.3 文件操作 37](#_Toc80803812)

[7.4 群组操作 40](#_Toc80803813)

[八、 实习总结 42](#_Toc80803814)

实习报告

# 一、项目名称

简存云

# 二、意义背景

由于实习公司的要求，运用主要以hadoop的技术框架为主，配合后端Java的代码的综合使用，学会使用springboot。同时，运用在实习期间学到的知识，基于springboot+hadoop+mysql+mybatis框架，前端使用bootstrap框架，后端使用spring boot为主，完成项目的登录、注册、密码找回、数据展示、上传、下载、重命名、删除、签到等功能。

# 三、需求分析

|  |  |  |  |
| --- | --- | --- | --- |
| **主要业务** | **Story模块** | **主要功能** | |
| 创  建  用  户  空  间 | 注册 | 用户可以通过手机号+验证码和邮箱账号+验证码两种方式进行注册，以防止恶意注册。注册成功后，用户可通过输入用户名和密码登录账号、手机号+验证码、邮箱账号+密码三种方式登录。用户忘记密码时可通过邮箱找回密码。用户在登录平台后可以创建群组发起签到和加入他人群组进行签到。为提高用户账号的安全性，用户退出登录账号后需要重新登录才能查看网盘内容。 |

|  |  |  |
| --- | --- | --- |
| 文  件  云  盘  存  储 | 上传文件 | 用户可以把自己的文件上传到网盘上，并可跨终端随时随地下载。为提升用户使用体验，用户可以选择按文件类型显示文件。同时，用户可根据自己的需求选择将文件从网盘中移除，若用户想同时删除同一目录下的多个文件，用户可选择需要删除的文件后进行批量删除。为防止用户误删导致文件丢失，用户删除的文件会放入回收站，用户可以在回收站选择恢复文件。为方便用户对网盘中的文件进行识别和管理，用户可对网盘中的文件或文件夹进行重命名，并且用户可以随时预览 |
| 下载文件 |
| 文件分类 |
| 删除文件 |
| 回收站 |
| 修改文件/文件夹名 |
|  | 文件预览 | 文件。 |
| 群  组  管  理 | 创建群组 | 用户在登录后，可以自己创建群组和加入他人群组。作为群主可以删除、上传、下载、修改群文件和删除、添加群成员，而作为群成员只可以上传、下载群文件，对于其他群成员不能进行任何操作。 |
| 加入群组 |
| 签  到  管  理 | 发起签到和签到 | 用户在创建或进入群组后，作为群主就可以发布签到和查看签到记录，作为群成员只有在规定时间内完成签到，才算签到成功，超过此时间范围，则视为无效。 |
| 查看签到记录 |

# 四、功能设计

## 4.1 概要设计

### 4.1.1 功能概要

**（1）注册**：新用户在第一次使用本网盘系统时，需要进行注册。用户输入自己的用户名、密码以及要绑定的邮箱等信息，并且进行真人验证码验证后即可注册成功。

**（2）登录/退出**：用户可以输入用户名和密码、手机号+验证码、邮箱账号+密码三种方式来登录自己的账号，进入主界面。

**（3）找回密码**：用户忘记密码时，可以点击找回密码按钮进入到找回密码界面，通过输入账号所绑定邮箱找回密码，或者输入手机号获得验证码进行登录。

**（4）修改密码：**用户若有修改密码需求，可以在登录平台后，点击修改密码按钮，通过输入原密码验证通过后，设置新的密码。

**（5）群组：**用户可以通过输入群组名和口令便可以创建一个群组，通过输入口令便可加入一个群组。在群组中，群成员可以上传、下载文件。

**（6）签到：**用户在加入群组后就可参加签到，如果用户自己创建一个群组，则可以发起签到。只有在规定时间内完成签到才有效，超过此时间则视为无效，用户可以随时查看签到记录。

**（7）上传**:用户可以在界面点击上传按钮就可以上传本地的文件至个人文件，在此文件夹下的所有文件只有用户本人才可以查看。

**（8）下载**: 用户在选择所要下载的文件后，就可以实现单个或者批量下载。

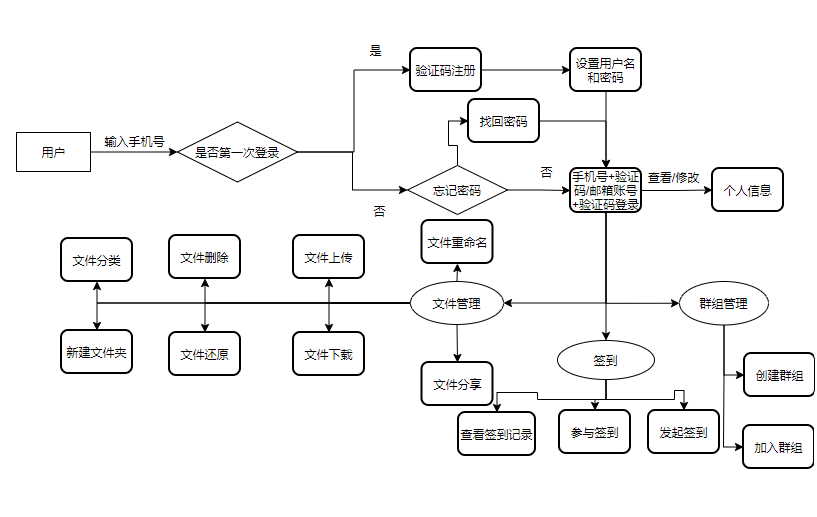
**（9）文件分类**: 为方便用户查看和管理网盘文件，网盘对用户上传的文件进行了文件类型分类，其中包括：全部文件、文档、图片、音频、视频、压缩文件和其他文件。用户点击不同的分类，将会出现符合类型的文件。

**（10）文件搜索：**为方便用户快速查看网盘文件，用户可以通过输入文件名来实现快速查找。

**（11）回收站**: 用户删除文件后, 文件将进入回收站. 进入回收站的文件, 会形成文件列表。用户可以选择对回收站内的文件进行恢复操作或者删除，同时还有批量删除、批量还原的便捷操作。

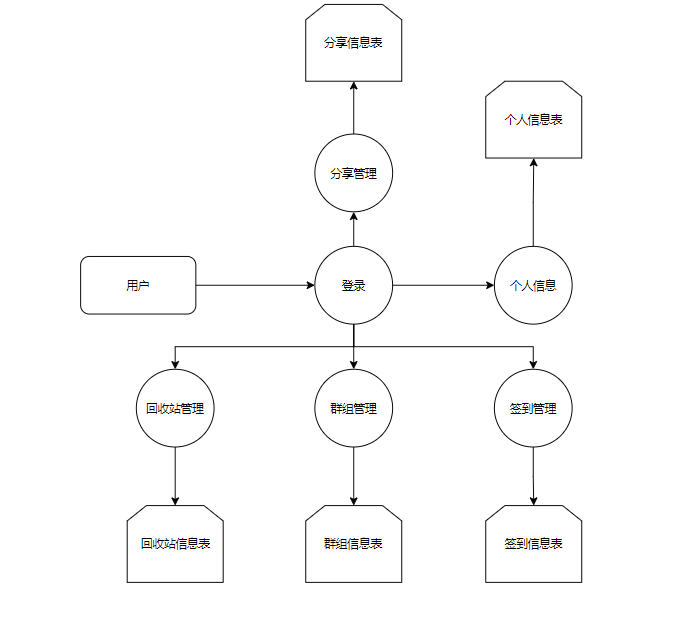
**（12）文件操作**：用户可以对网盘上的文件进行创建新文件夹，文件及文件夹重命名等功能。

### 4.1.2 系统整体功能流程图

**

4-1-1 系统整体功能流程图

### 4.1.3 系统数据流图



4-1-2 系统数据流图

## 4.2 详细设计

### 4.2.1登录注册

1. 注册

用户输入用户名、密码以及个人邮箱在简存云云盘上注册自己的账号。

（2）登录

用户完成注册后，可以通过用户名+密码、邮箱账号+密码、手机号+验证码三种方式登录简存云云盘。

（3）修改密码：用户可以根据个人需求，通过输入原密码验证的方式设置新密码。

（4）忘记密码：若用户忘记密码，可以通过邮箱或者手机号的方式找回原密码。

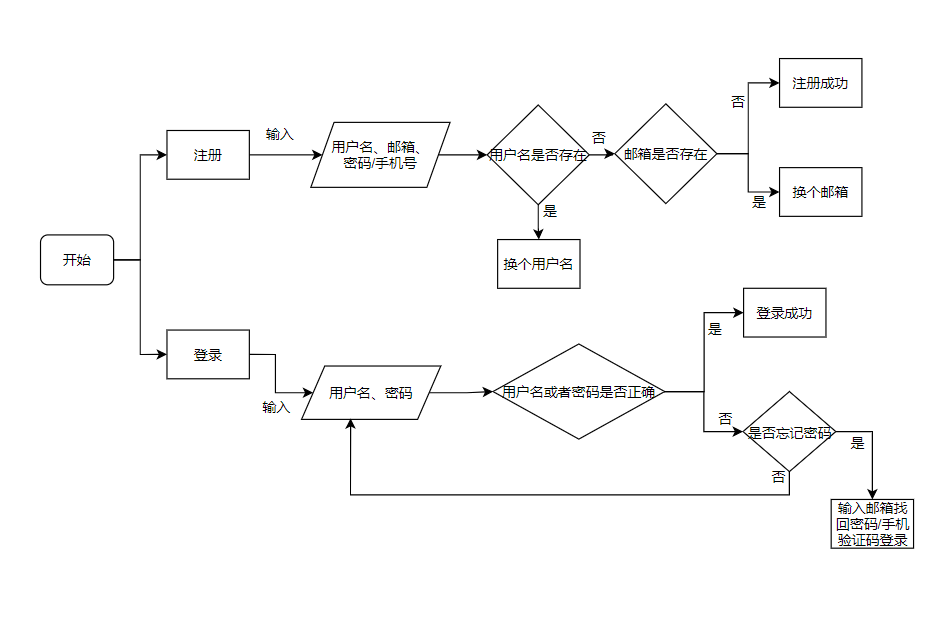


图4-2-1 登录注册功能流程图

### 4.2.2 主界面

用户登录成功后进入平台主界面。主界面主要由四部分组成：顶部导航栏、侧边导航栏、快捷操作以及文件管理。

其中顶部导航栏包含个人用户名、修改密码、文件搜索、签到以及退出；

侧边导航栏包括个人文件、群组文件、收藏文件、回收站、文件分类（视频、音频、图片、压缩包、文档）、签到记录；

快捷操作包括上传文件、创建文件夹、批量删除和批量下载操作按钮。

快捷操作包括上传文件、创建文件夹以及批量删除操作按钮；

文件管理负责文件的删除、重命名、下载、分享、收藏操作。

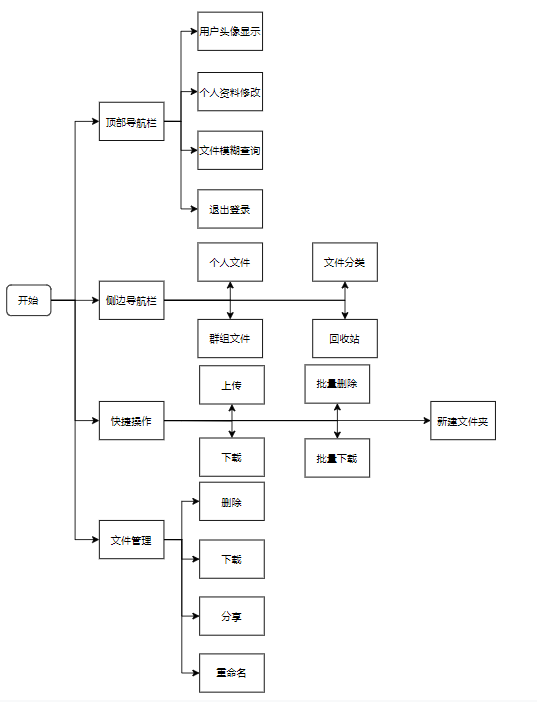


图4-2-2 主页面功能流程图

# 数据库设计

## 5.1 数据库设计概述

本系统采用MySQL数据库进行开发，本系统共有三张表。

## 5.2 数据库逻辑结构设计

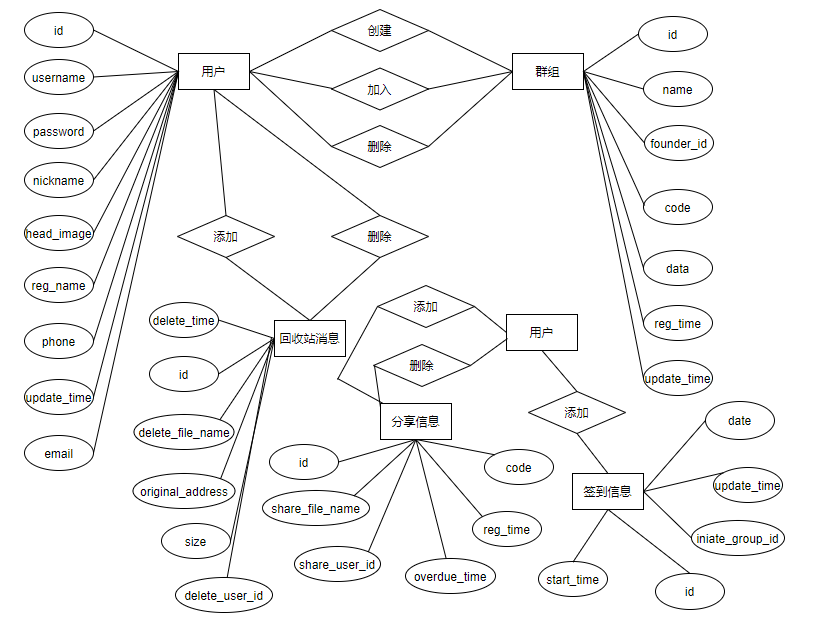


图5-2-1 数据库逻辑结构设计图

## 5.3 数据库物理结构设计

CREATE TABLE `users` (

`id` int(11) AUTO\_INCREMENT COMMENT '主键Id',

`username` varchar(20) NOT NULL COMMENT '用户名',

`password` varchar(255) NOT NULL COMMENT '密码',

`nickname` varchar(20) NOT NULL DEFAULT '' COMMENT '昵称',

`head\_image` varchar(255)NOT NULL DEFAULT 'default.jpg' COMMENT '头像地址',

`phone` varchar(20) COMMENT '手机号',

`email` varchar(20) COMMENT '邮箱',

`reg\_time` datetime NOT NULL DEFAULT CURRENT\_TIMESTAMP COMMENT '注册时间',

`update\_time` datetime NOT NULL DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP COMMENT '修改时间',

PRIMARY KEY (`id`),

UNIQUE KEY `sku\_users\_username` (`username`) USING BTREE,

UNIQUE KEY `sku\_users\_phone` (`phone`) USING BTREE,

UNIQUE KEY `sku\_users\_email` (`email`) USING BTREE

)ENGINE=INNODB AUTO\_INCREMENT=1 DEFAULT CHARSET=utf8 ROW\_FORMAT=REDUNDANT COMMENT='用户表';

CREATE TABLE `group` (

`id` int(11) AUTO\_INCREMENT COMMENT '主键Id',

`name` varchar(20) NOT NULL COMMENT '群组名',

`founder\_id` int(11) NOT NULL COMMENT '创始人Id',

`code` varchar(20)COMMENT '邀请码',

`data` blob COMMENT '群组数据',

`reg\_time` datetime NOT NULL DEFAULT CURRENT\_TIMESTAMP COMMENT '创建时间',

`update\_time` datetime NOT NULL DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP COMMENT '修改时间',

PRIMARY KEY (`id`),

CONSTRAINT `fk\_emp\_group\_founderid` FOREIGN KEY(`founder\_id`) REFERENCES `users`(`id`)

)ENGINE=INNODB AUTO\_INCREMENT=1 DEFAULT CHARSET=utf8 ROW\_FORMAT=REDUNDANT COMMENT='群组表';

CREATE TABLE `share` (

`id` int(11) AUTO\_INCREMENT COMMENT '主键Id',

`share\_file\_name` varchar(255) NOT NULL COMMENT '分享文件名',

`share\_user\_id` int(11) NOT NULL COMMENT '分享用户Id',

`code` varchar(20) COMMENT '分享码',

`reg\_time` datetime NOT NULL DEFAULT CURRENT\_TIMESTAMP COMMENT '分享时间',

`overdue\_time` datetime NOT NULL DEFAULT CURRENT\_TIMESTAMP COMMENT '过期时间',

PRIMARY KEY (`id`),

CONSTRAINT `fk\_emp\_share\_share\_user\_id` FOREIGN KEY(`share\_user\_id`) REFERENCES `users`(`id`)

)ENGINE=INNODB AUTO\_INCREMENT=1 DEFAULT CHARSET=utf8 ROW\_FORMAT=REDUNDANT COMMENT='分享表';

CREATE TABLE `recyclebin` (

`id` int(11) AUTO\_INCREMENT COMMENT '主键Id',

`delete\_file\_name` varchar(255) NOT NULL DEFAULT '' COMMENT '删除文件名',

`original\_address` varchar(255) NOT NULL COMMENT '原始路径',

`size` bigint NOT NULL DEFAULT 0 COMMENT '文件大小',

`delete\_user\_id` int(11) NOT NULL COMMENT '删除用户Id',

`delete\_time` datetime NOT NULL DEFAULT CURRENT\_TIMESTAMP COMMENT '删除时间',

PRIMARY KEY (`id`),

CONSTRAINT `fk\_emp\_recyclebin\_delete\_user\_id` FOREIGN KEY(`delete\_user\_id`) REFERENCES `users`(`id`)

)ENGINE=INNODB AUTO\_INCREMENT=1 DEFAULT CHARSET=utf8 ROW\_FORMAT=REDUNDANT COMMENT='回收站';

CREATE TABLE `sign` (

`id` int(11) AUTO\_INCREMENT COMMENT '主键Id',

`initiate\_group\_id` int(11) NOT NULL COMMENT '发起群组Id',

`data` blob COMMENT '签到数据',

`start\_time` datetime NOT NULL DEFAULT CURRENT\_TIMESTAMP COMMENT '开始时间',

`update\_time` datetime NOT NULL DEFAULT CURRENT\_TIMESTAMP COMMENT '修改时间',

PRIMARY KEY (`id`),

CONSTRAINT `fk\_emp\_sign\_initiate\_group\_id` FOREIGN KEY(`initiate\_group\_id`) REFERENCES `group`(`id`)

)ENGINE=INNODB AUTO\_INCREMENT=1 DEFAULT CHARSET=utf8 ROW\_FORMAT=REDUNDANT COMMENT='回收站';

## 5.4 数据库安全设计

注册

手机注册短信验证

隐私空间

离开

个人文件只有本人有操作权限

一旦离开网页，则需再次验证登录

图5-4-1 数据库安全设计示意图

## 5.5 数据表结构

users

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段 | 类型 | 字段长度 | 描述 | 是否可为空 | 备注 |
| id | int | 11 | id | not null | 自增，主键 |
| username | varchar | 20 | 用户名 | not null |  |
| password | varchar | 255 | 密码 | not null |  |
| nickname | varchar | 20 | 昵称 | not null | DEFULT  “ ” |
| head\_image | varchar | 255 | 头像地址 | not null | DEFAULT  default.jpg |
| phone | varchar | 20 | 手机号 | not null |  |
| email | varchar | 20 | 邮箱 | not null |  |
| reg\_time | datetime |  | 注册时间 | not null | DEFAULT  CURRENT\_TIMESTAMP |
| update\_time | datetime |  | 修改时间 | not null | DEFAULT  CURRENT\_TIMESTAMP |

group

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段 | 类型 | 字段长度 | 描述 | 是否可为空 | 备注 |
| id | int | 11 | Id | not null | 自增，主键 |
| name | varchar | 20 | 群组名 | not null |  |
| founder\_id | int | 11 | 创始人id | not null |  |
| code | varchar | 20 | 邀请码 | not null |  |
| data | blob |  | 群组数据 | null |  |
| reg\_time | datetime |  | 创建时间 | not null | DEFAULT  CURRENT\_TIMESTAMP |
| update\_time | datetime |  | 修改时间 | not null | DEFAULT  CURRENT\_TIMESTAMP |

share

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段 | 类型 | 字段长度 | 描述 | 可否为空 | 备注 |
| id | int | 11 | id | not null | 自增，主键 |
| share\_file\_name | varchar | 255 | 分享文件名 | not null |  |
| share\_user\_id | int | 11 | 分享用户id | not null |  |
| code | varchar | 20 | 分享码 | not null |  |
| reg\_time | datetime |  | 分享时间 | not null | DEFAULT  CURRENT\_TIMESTAMP |
| overdue\_time | datetime |  | 过期时间 |  | DEFAULT  CURRENT\_TIMESTAMP |

recyclebin

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段 | 类型 | 字段长度 | 描述 | 是否可为空 | 备注 |
| id | int | 11 | id | not null | 自增，主键 |
| delete\_file\_name | varchar | 255 | 删除文件名 | not null | DEFAULT “” |
| original\_address | varchar | 255 | 原始路径 | not null | DEFAULT “” |
| size | bigint |  | 文件大小 | not null | DEFAULT 0 |
| delete\_user\_id | int | 11 | 删除用户id | not null |  |
| delete\_time | datetime |  | 删除时间 | not null | DEFAULT  CURRENT\_TIMESTAMP |

sign

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段 | 类型 | 字段长度 | 描述 | 是否可为空 | 备注 |
| id | int | 11 | id | not null | 自增，主键 |
| initiate\_group\_id | int | 11 | 发起群组id | not null |  |
| data | blob |  | 签到数据 | null |  |
| start\_time | datetime |  | 开始时间 | not null | DEFAULT  CURRENT\_TIMESTAMP |
| update\_time | datetime |  | 修改时间 | not null | DEFAULT  CURRENT\_TIMESTAMP |

# 亮点设计

## 6.1 用户密码加密处理

@Override

**public** **void** **userRegister**(**User** user) {

**if** (userDao.findUserByUsername(user) != **null**) {

**throw** **new** **RuntimeException**("用户已经存在");

} **else** **if** (userDao.findUserByPhone(user) != **null**) {

**throw** **new** **RuntimeException**("手机号已经被注册");

} **else** **if** (userDao.findUserByEmail(user) != **null**) {

**throw** **new** **RuntimeException**("邮箱已经注册");

} **else** {

//添加用户前先对密码进行加密处理

user.setPassword(shaEncryptUtil.encryptPwd(user.getPassword()));

userDao.addUser(user);

}

}

**public** **void** **userRegister**(**User** user) {

**if** (userDao.findUserByUsername(user) != **null**) {

**throw** **new** **RuntimeException**("用户已经存在");

} **else** **if** (userDao.findUserByPhone(user) != **null**) {

**throw** **new** **RuntimeException**("手机号已经被注册");

} **else** **if** (userDao.findUserByEmail(user) != **null**) {

**throw** **new** **RuntimeException**("邮箱已经注册");

} **else** {

//添加用户前先对密码进行加密处理

user.setPassword(shaEncryptUtil.encryptPwd(user.getPassword()));

userDao.addUser(user);

}

}

## 6.2 手机号+验证码登录

@Override

**public** **User** **fastLogin**(**User** user, **String** code) {

//先将用户真实数据从数据库取出

**User** userInfo = userDao.findUser(user);

**if** (userInfo == **null**)

**throw** **new** **RuntimeException**("用户不存在");

//先判断用户手机号是不是空的,是则进行邮箱验证码的验证,否则进行手机验证码的验证,验证成功返回用户数据,失败返回null.

**return** "".equals(user.getPhone()) ?

**CodeUtil**.codeCheck(user.getEmail(), code, **CodeUtil**.CodeTypeEnum.EMAIL\_CODE) ? userInfo : **null** :

**CodeUtil**.codeCheck(user.getPhone(), code, **CodeUtil**.CodeTypeEnum.PHONE\_CODE) ? userInfo : **null**;

}

## 6.3 回收站

//回收站自动清空

**public** **static** **void** **setAutoCleanTime**() {

**String** autoCleanTime= **ResourceBundle**.getBundle("application").getString("spring.hadoop.recycleBin.autoCleanTime");

**String** []ss=autoCleanTime.split("\\\*");

**long** autoTime=1;

**for** (**String** s:ss){

autoTime\*=**Integer**.parseInt(s);

}

**RecycleBinServiceImpl**.autoCleanTime = autoTime/1000>0?autoTime/1000:1;

}

**static** {

setAutoCleanTime();

**final** **int**[] count = {1};

**final** **Runnable** beeper = () -> {

log.info("第"+ count[0] +"次回收站清理开始");

**Set**<**String**> list;

**if** (redisTemplate.hasKey("recycleBin")) {

list= redisTemplate.boundZSetOps("recycleBin").rangeByScore(0, **System**.currentTimeMillis());

**if** (list!=**null**){

**int** num=list.size();

log.info("本次共有"+num+"个需要清理的文件");

list.forEach(v-> {

**try** {

hadoopService.completelyDelete(v);

redisTemplate.boundZSetOps("recycleBin").remove(v);

recycleBinDao.deleteRecycleRecord(**new** **RecycleBin**(**HadoopServiceImpl**.HadoopUtil.getUserId(v)));

} **catch** (**IOException** e) {

e.printStackTrace();

}

});

log.info("清理完成，本次共清理"+ num +"个文件");

}

}**else** {

log.info("回收站空空如也");

}

count[0]++;

};

*//文件还原*

@Override

**public** **boolean** **reduction**(**String** recyclePath, **User** user) **throws** **IOException** {

**RecycleBin** recycleBin=**new** **RecycleBin**(**Integer**.parseInt(recyclePath.substring(recyclePath.lastIndexOf("/")+1)));

recycleBin=recycleBinDao.select(recycleBin);

**if** (!recycleBin.getUser().equals(user))

**throw** **new** **RuntimeException**("这不是你的文件！！！");

**boolean** code=update(recyclePath,recycleBin.getOriginalAddress());

**if** (code){

recycleBinDao.deleteRecycleRecord(recycleBin);

redisTemplate.boundZSetOps("recycleBin").remove(recyclePath);

}

**return** code;

}

## 6.3 分享文件

//分享文件

@Override

**public** **void** **shareFile**(**String** path,**Share** share) **throws** **IOException** {

copyFile(path,shareRootPath+"/"+share.getId());

shareDao.addShare(share);

}

//获得分享文件列表

@Override

**public** **Share** **getShareFile**(**Share** share) {

//先取出真实数据

**Share** tempShare=shareDao.selectShare(share);

//判断验证码是否正确

**if** (tempShare.getCode()==**null**||tempShare.getCode().equals(share.getCode()))

**return** tempShare;

**else**

**throw** **new** **RuntimeException**("验证码错误");

}

## 6.4 群组管理

//加入群组

@Override

**public** **void** joinGroup(Group group, User user, Group.Data.Power power) {

//先从数据库取出群组数据

Group tempGroup = groupDao.select(group);

Group.Data data;

**if** (tempGroup == **null**)

**throw** **new** RuntimeException("群组不存在!!!");

**if** (tempGroup.getFounder().equals(user))

**throw** **new** RuntimeException("你是群主!!!!!");

**if** (tempGroup.getCode()!=**null**&&!tempGroup.getCode().equals(group.getCode()))

**throw** **new** RuntimeException("您输入的邀请码不正确!!!!!");

//如果群组中的data不为空

**if** (tempGroup.getData() != **null**) {

//将取出的byte[]转成Data对象

data = Group.bytesToData(tempGroup.getData());

Map<User, Group.Data.Power> map = data.getMap();

**if** (map.containsKey(user))

**throw** **new** RuntimeException("你已经在群组中，无法重复加入.");

map.put(user, power);

//将修改后的Data对象重新写回byte[]

group.dataToBytes(**new** Group.Data(map));

} **else** {

group.dataToBytes(**new** Group.Data(user, power));

}

groupDao.updateData(group);

}

//退出群组

@Override

**public** **void** exitGroup(Group group, User user) {

//先从数据库取出群组数据

group = groupDao.select(group);

**if** (group == **null**)

**throw** **new** RuntimeException("群组不存在");

Group.Data data;

**if** (group.getFounder().getId() == user.getId())

//群主只能解散,退出必须要先转让

**throw** **new** RuntimeException("你是群主!!!!!");

**if** (group.getData() == **null**) {

**throw** **new** RuntimeException("这个群组是空的!!!");

} **else** {

//将取出的byte[]转成Data对象

data = Group.bytesToData(group.getData());

Map<User, Group.Data.Power> map = data.getMap();

**if** (!map.containsKey(user))

**throw** **new** RuntimeException("你不在当前群组中，无法退出");

map.remove(user);

//将修改后的Data对象重新写回byte[]

group.dataToBytes(**new** Group.Data(map));

}

groupDao.updateData(group);

}

//获取群组所有成员

@Override

**public** List<User> getGroupMember(Group group) {

List<User> users = **new** ArrayList<>();

group = groupDao.select(group);

**if** (group == **null**)

**throw** **new** RuntimeException("群组不存在");

//首先将群主加入list中

users.add(group.getFounder());

**if** (group.getData() != **null**) {

users.addAll(Group.bytesToData(group.getData()).getMap().keySet());

}

**return** users;

}

## 6.5 签到管理

//发起签到

@Override

**public** **void** createSignInfo(Sign sign, User user) {

**if** (!user.equals(groupService.getGroupInfo(sign.getGroup()).getFounder()))

**throw** **new** RuntimeException("只有群主才能发起签到");

signDao.addSignInfo(sign);

}

//签到

@Override

**public** **void** sign(Sign sign, User user) {

sign=signDao.select(sign);

**if** (sign==**null**)

**throw** **new** RuntimeException("签到信息不存在");

**if** (System.currentTimeMillis()>sign.getEndTime().getTime())

**throw** **new** RuntimeException("签到已过期");

**if** (sign.getData()!=**null**) {

Sign.Data data = Sign.bytesToData(sign.getData());

Map<User, Object> map = data.getMap();

**if** (map.containsKey(user))

**throw** **new** RuntimeException("你已经签到过了");

map.put(user, **new** Date());

sign.dataToBytes(data);

}**else** {

sign.dataToBytes(**new** Sign.Data(user,**new** Date()));

}

signDao.updateSignInfo(sign);

}

# 运行截图

## 7.1 登录注册

（1）注册

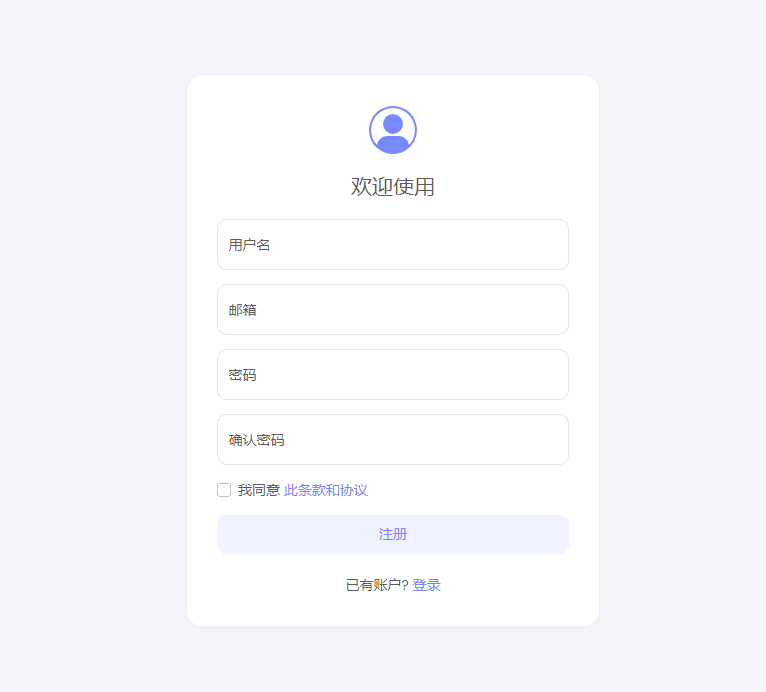


图7-1-1 注册界面效果图

1. 登录



图7-1-2登录界面效果图

1. 忘记密码

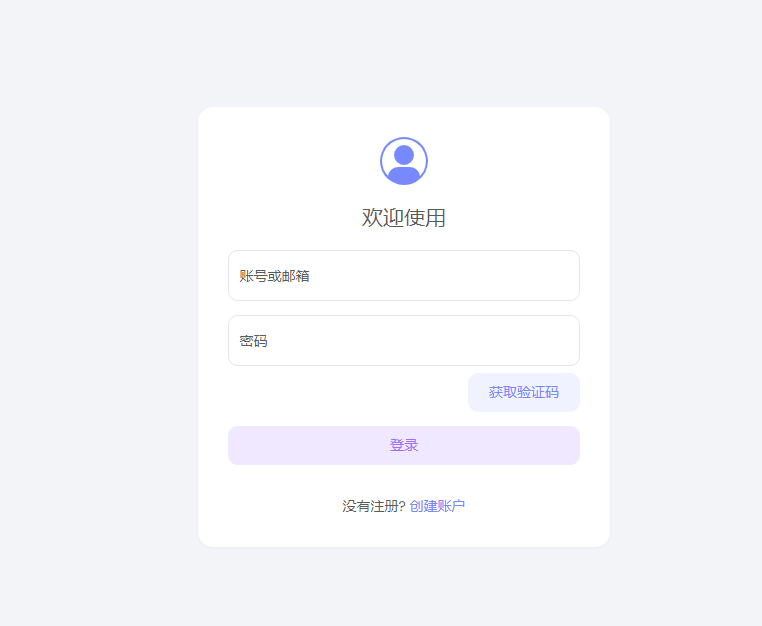


图7-1-3忘记密码界面效果图

## 7.2 首页个人账号操作

（1）修改密码

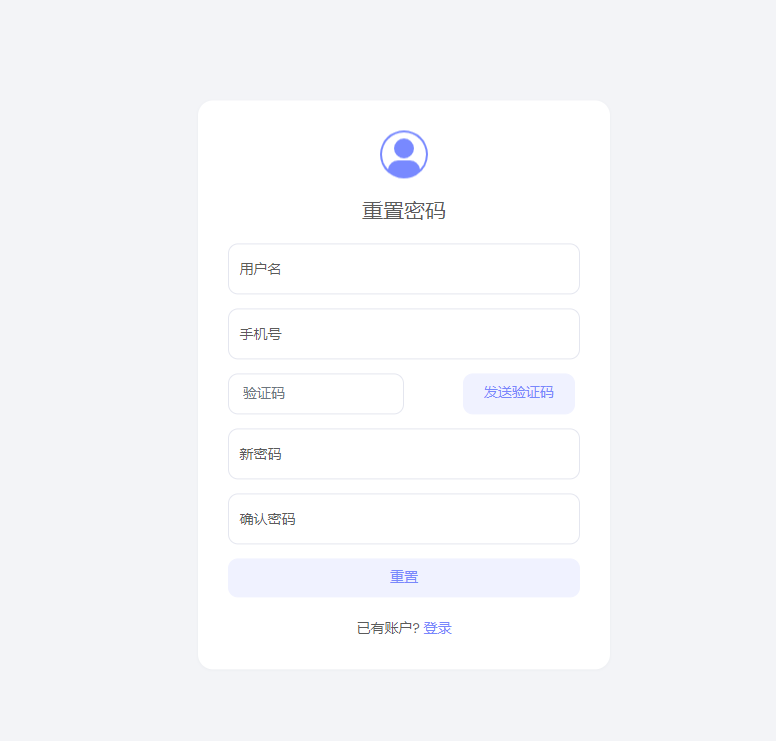


图7-2-1修改密码界面效果图

1. 签到

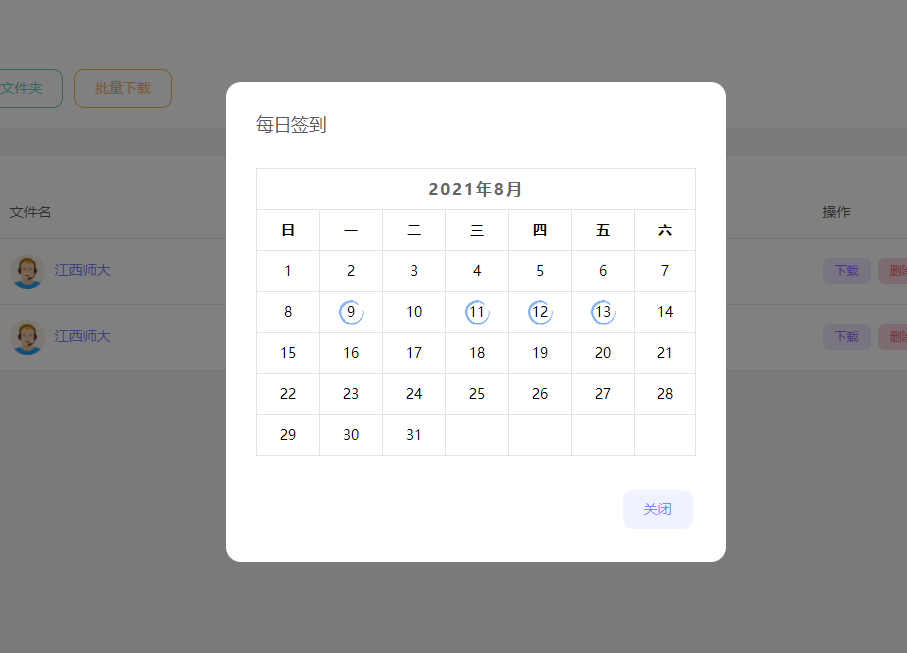


图7-2-2签到界面效果图

## 7.3 文件操作

（1）上传

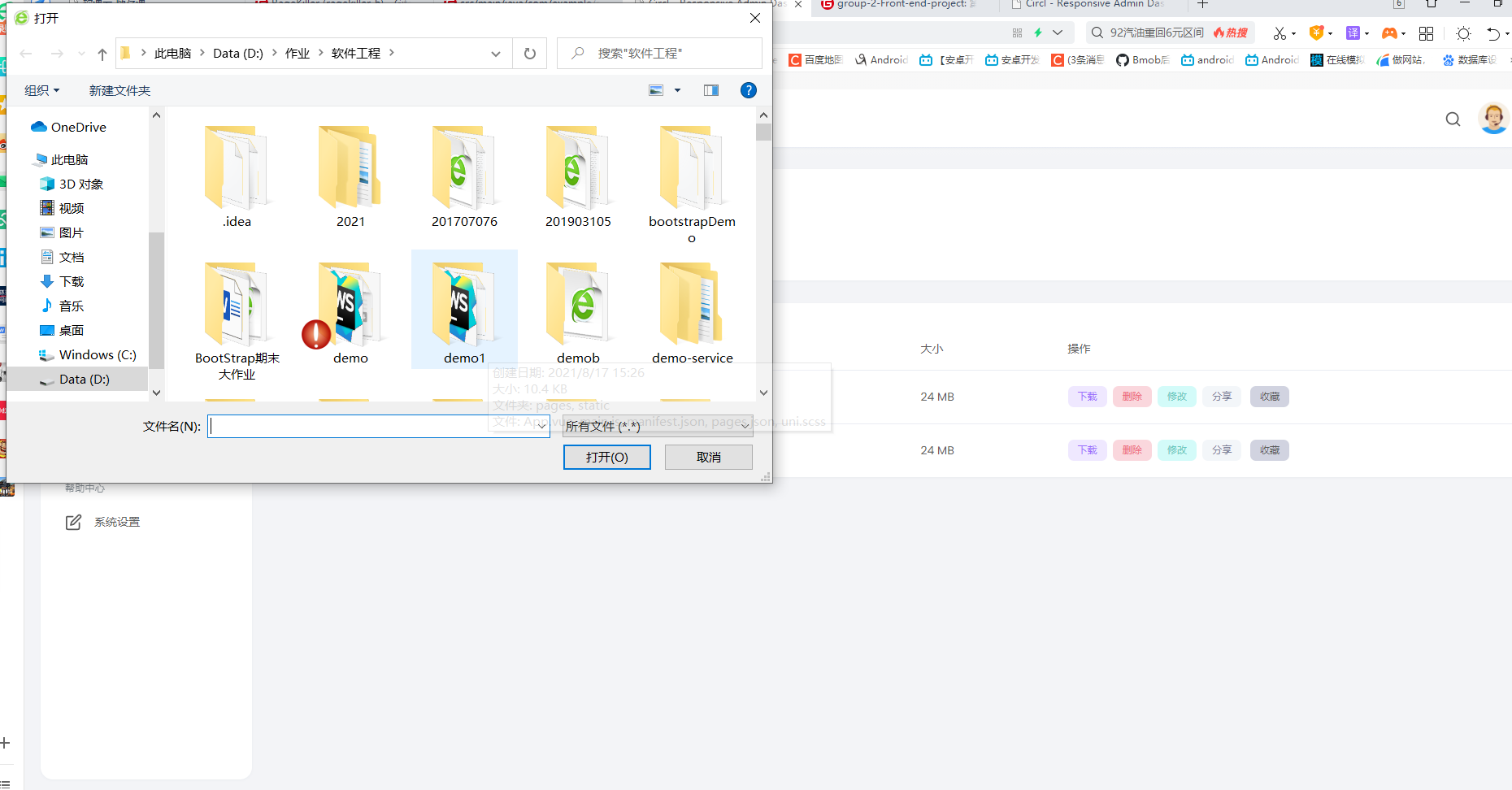


图7-3-1上传界面效果图

（2）新建文件夹

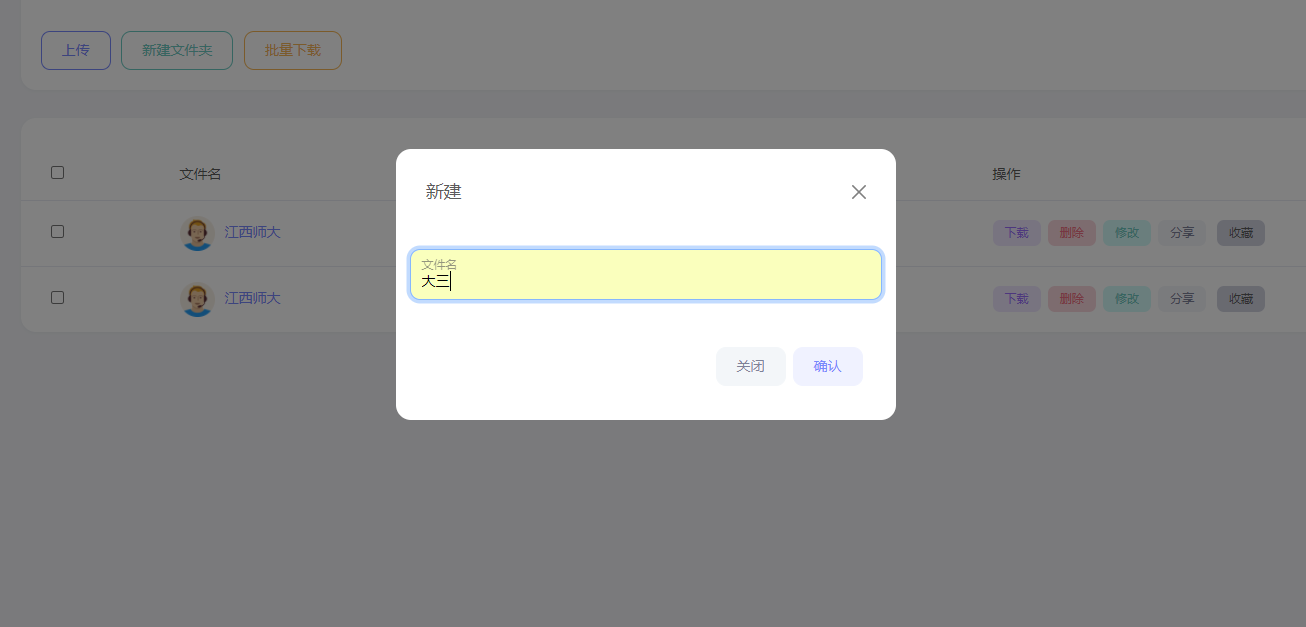


图7-3-2新建文件夹效果图

1. 批量下载



图7-3-3批量下载界面效果图

1. 回收站

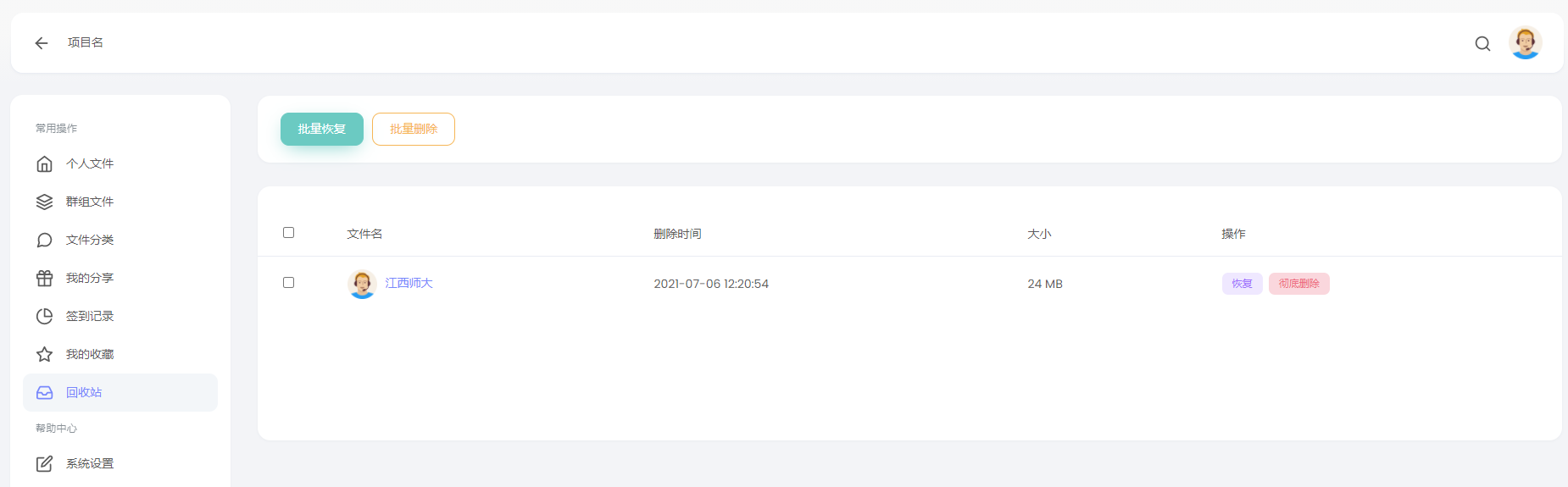


图7-3-4回收站界面效果图

1. 清空回收站

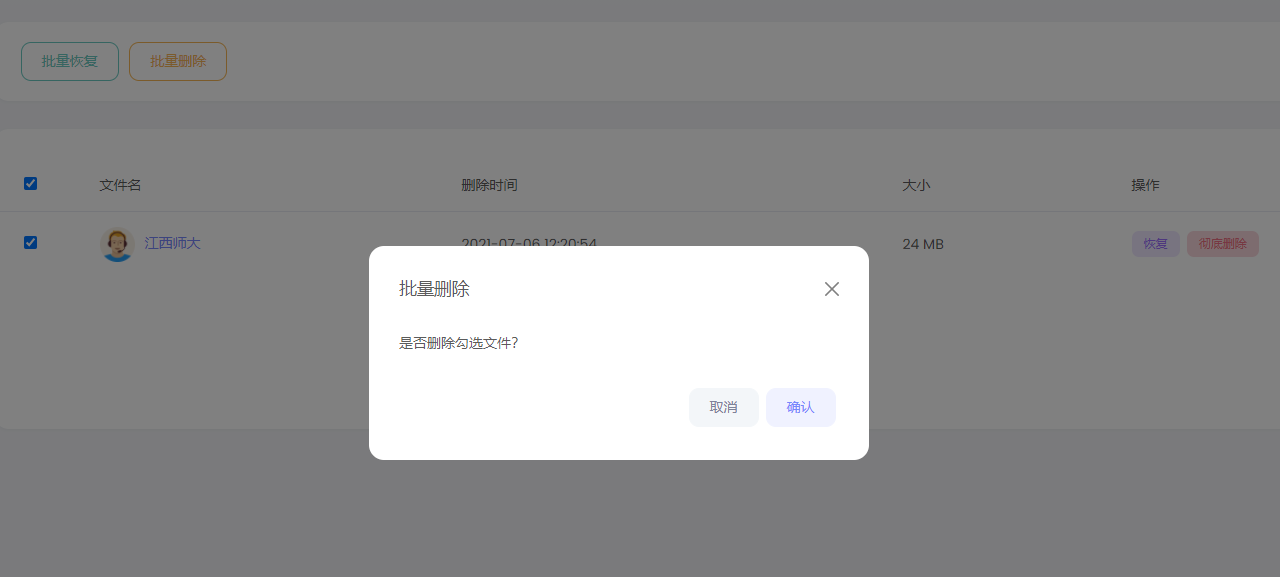


图7-3-5清空回收站界面效果图

1. 文件分类

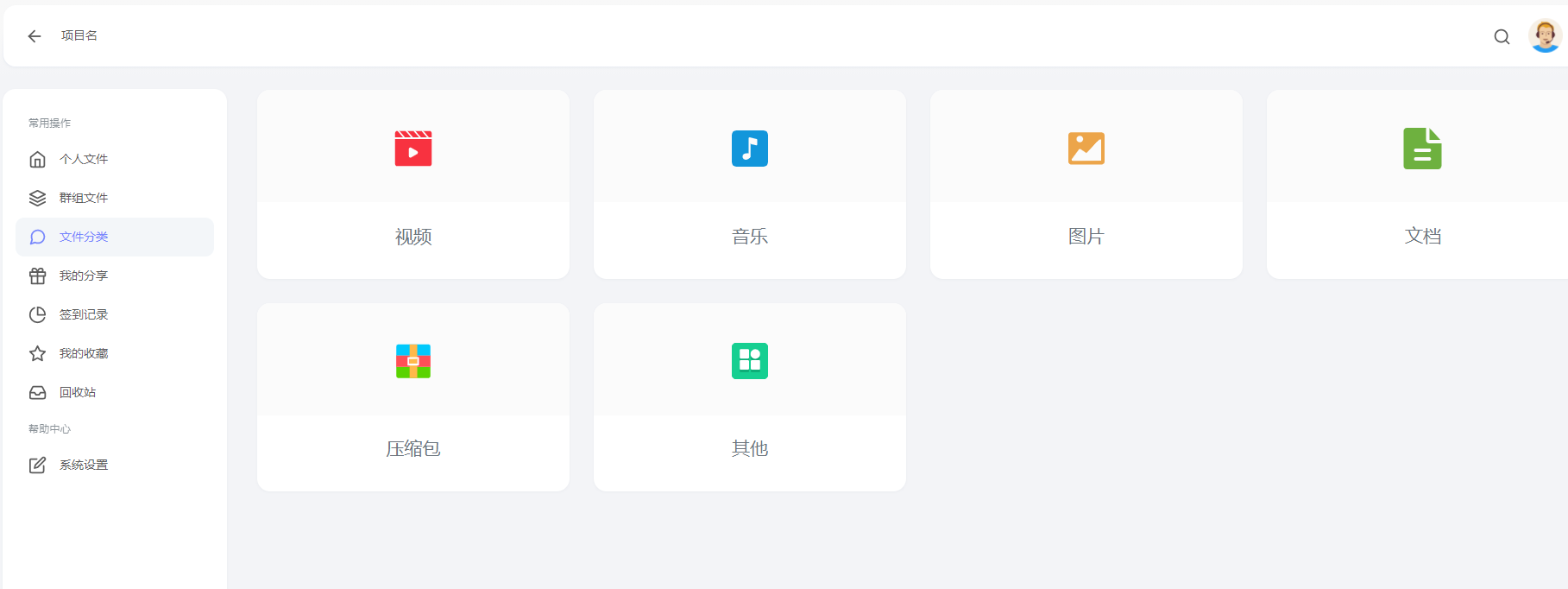


图7-3-6文件分类界面效果图

## 7.4 群组操作

(1)群组

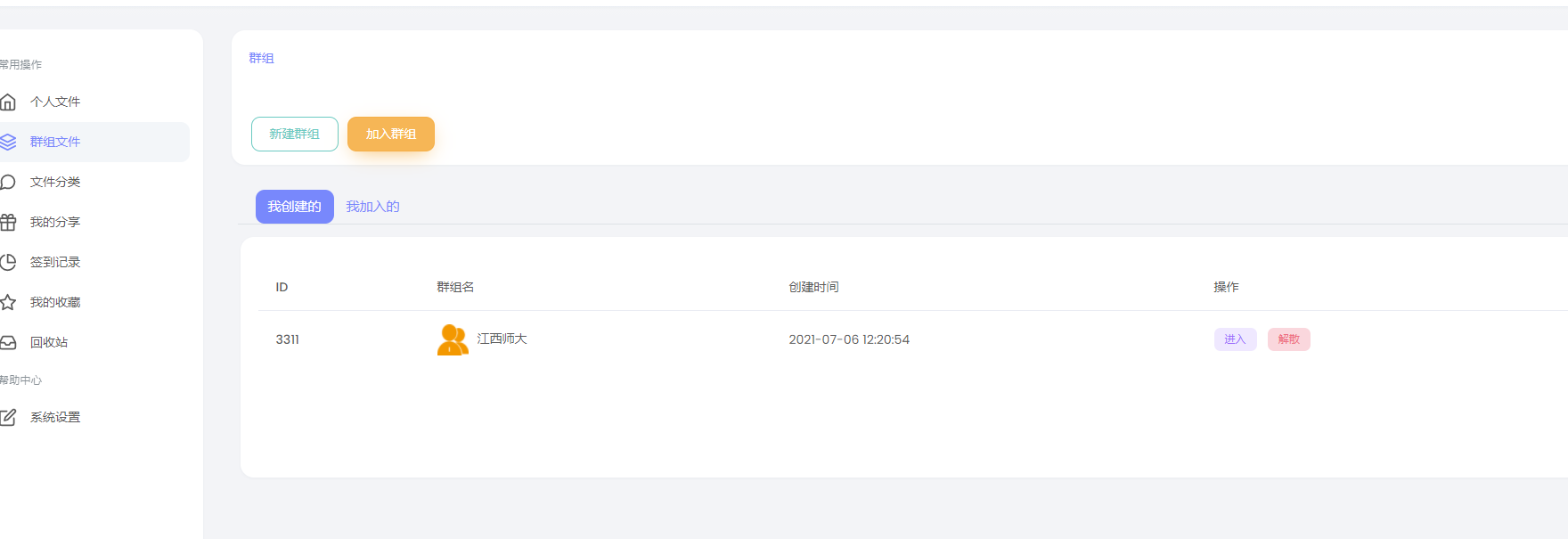


图7-4-1群组界面效果图

（2）新建群组



图7-4-2新建群组界面效果图

（3）加入群组

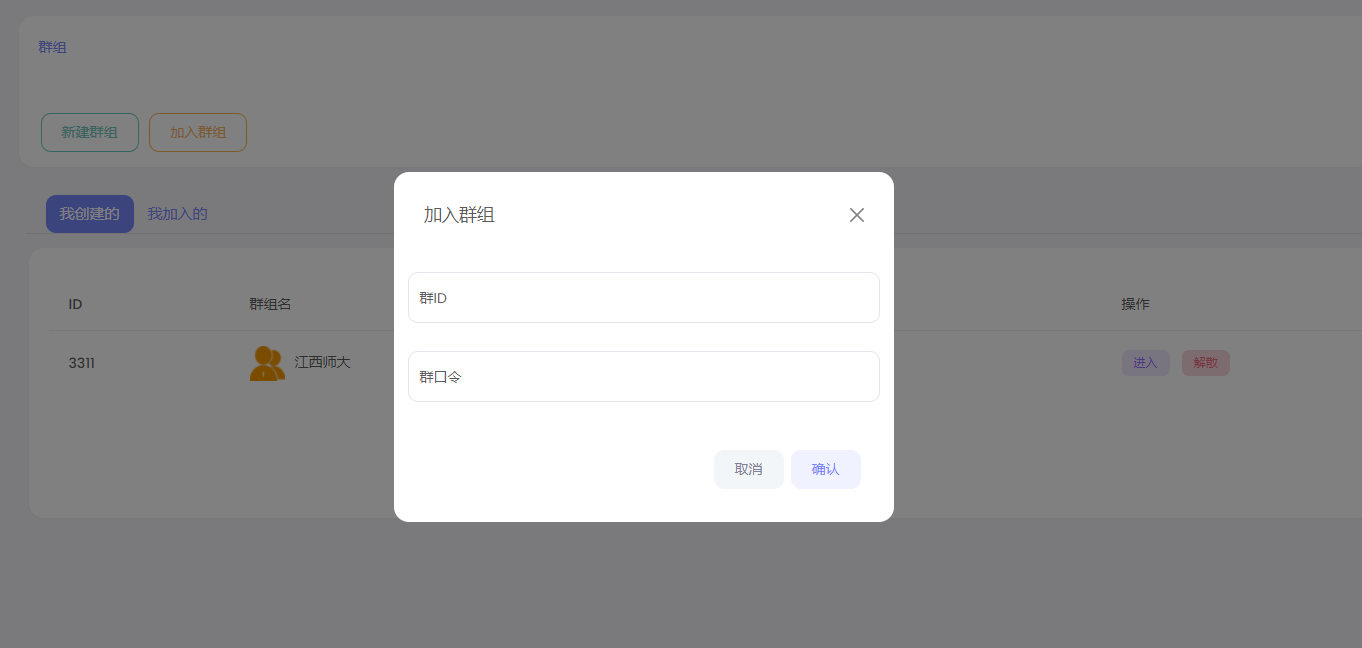


图7-4-3加入群组界面效果图

（4）群组文件

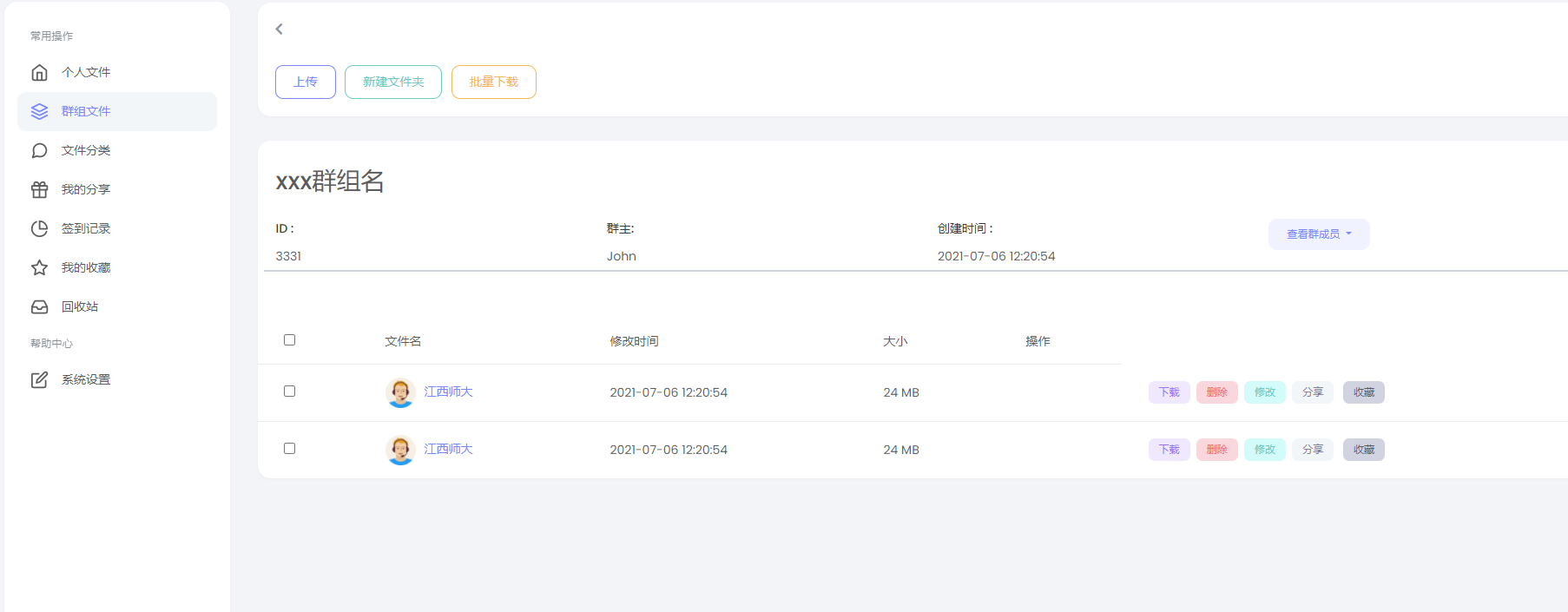


图7-4-4群组文件界面效果图

# 实习总结

一个月的实训时间匆匆而过，暑期实训即将结束，十分感恩，十分不舍，唯独没有遗憾。在本次实训中，我受益良多，感谢学院给予的培训机会，感谢软通动力张鹏飞老师和刘佳老师的指导和关照，接下来我对本次实训做一个总结。

实训过程中，张鹏飞老师为我们传授了大数据存储和大数据计算的相关知识，这是未来软件行业的一个大方向，为我们今后的工作学习指明了方向。教学过程中理论知识和项目实践开发相结合。虽然本科学习的过程中对于软件工程有了一个系统的学习，也学习了相应的开发语言和技术，但从未真正地融入到真实的工作环境模拟开发，实训恰好给了我们一个很好的机会将学习和实践结合起来。

我们将hadoop大数据存储技术应用到具体项目中，开发了一个web个人存储云盘，用户可以将自己需要存储的文件存储到个人web云盘中，随时随地进行下载和上传，基于大数据存储使上传和下载速度更快，体验性更好。项目采用前后端分离方式开发，使用git版本控制工具，小组的代码存放在gitee，团队每个人都能进行查看和下载运行，模拟实际工作场景。同时在分工时，我们也在尽量模拟敏捷开发的工作模式，由于经验不足，遇到一些困难，但通过这些困难，我们也吸取了教训总结了经验，明确了团队沟通的重要性，尤其是前后端工程师的接口沟通，对于需求的分析。

作为一名测试人员，我在本次实训中拓展了大数据的知识、强化了Linux命令、串通了项目的开发细节和流程、模拟了实际的工作场景，张老师深入浅出的教学使软件开发知识的学习不再枯燥晦涩，我对今后的学习充满信心，未来也会更加努力，拓展自己的知识面，成为一名合格的软件测试工程师。

实习工作评价与成绩评定

|  |
| --- |
| 学生自我鉴定：  对于本次实训，我做到了不迟到、不早退、零缺勤，认真听课，课下能够用按时自行完成作业和任务。遇到的问题学会利用网络、书本自己独立解决。对于小组项目的制定能够积极参与，但对于分配的任务个人实战经验还不足在这点上需要改进。 |
| 实习单位或指导教师意见（由实习企业填写）：  该生演讲能力比较突出，团队协作能力比较强，学习态度认真，继续努力  单位(盖章):  2021 年 9月 1 日 |
| 实习工作成绩（占综合成绩 50%）  91  实习报告成绩（占综合成绩 50%）  88  实习综合成绩(百分制)  89．5  指导教师签字:张鹏飞  2021 年 9 月 1 日 |
| 评阅教师评定实习成绩:  评阅教师签名:  年 月 日 |

A：90~100分 A-：85~89分 B+：82~84分 B：78~81分 B-：75~77分

C+：72~74分 C：68~71分 C-：64~67分 D：60~63分 F：<60分