8319205241442265013257

本 科 毕 业 设 计 (论 文)

**题目：web 云存储硬盘系统的设计与实现**

|  |  |
| --- | --- |
| **学生姓名：** | **李政** |
| **学 号：** | **1606031127** |
| **班 级：** | **16软件工程1班** |
| **专 业：** | **软件工程** |
| **院 （系） ：** | **计算机工程学院** |
| **指导教师：** | **刘明纲** |

二零二零年五月二十二日

毕业设计(论文)原创性声明

本人郑重声明：所呈交毕业论文，是本人在指导教师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不包含任何其他人或集体已经发表或撰写过的作品成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

作者签名： 导师签名：

日期： 年 月 日 日期： 年 月 日

毕业设计(论文)使用授权声明

本人完全了解成都工业学院有关保留、使用论文的规定，即：学校有权保留论文并向国家主管部门或其指定机构送交论文的电子版和纸质版，有权将论文用于非赢利目的的少量复制并允许论文进入学校图书馆、院（系）资料室被查阅，有权将论文的内容编入有关数据库进行检索，可以采用复印、缩印或其他方法保存论文。

作者签名： 导师签名：

日期： 年 月 日 日期： 年 月 日

摘 要

近年来，随着计算机技术的飞速发展，信息数据化越来越普遍，计算机的使用者每天都要存储大量的文件。但是，在某些情况下，用户存储的文件量受到用户计算机自身存储硬件容量的严重限制。 其次，数据对于用户来说十分重要，用户某些时候不能把重要数据放置在同一个地方。这就涉及到文件的备份，对于普通用户来说，单独维护一台文件备份服务器，成本太过高昂。所以，如今云存储成为当下火热的技术焦点。

本文对云存储结合校园环境进行了介绍，分析了校园中师生的存储需求，并给出了具体的系统实现。本文的内容包括：对云存储技术的介绍，对系统进行了需求分析，数据库设计，系统概要设计，系统详细设计等。

所实现的系统具有一定的可用性，对高校中师生的文件存储具有一定的帮助，能够提高教师和学生的工作学习效率。

**关键词：**云存储；文件；云盘

Abstract

In recent years, with the development of computer science, information digitalization has become more and more common. Users need to storage a large number of files every day. However, in some cases, the amount of files a user can store is severely limited by the storage capacity of the storage hardware of the user's computer. Otherwise, data is so important to users that sometimes they can't put important data in the same place. This involves backing up files, which is too expensive for the average user to maintain on their own. As a result, cloud storage has become a hot technology focus.

This paper introduces the cloud storage combined with the campus environment, analyzes the storage needs of teachers and students on campus, and gives the concrete system implementation. The content of this paper includes: the introduction of cloud storage technology, requirements analysis of the system, database design, system outline design, detailed system design and so on.

The system can be used for the file storage of teachers and students in colleges and universities, and can improve the efficiency of teachers and students.

**Key Words:** Cloud Storage；File；Network Disk

目 录

[第1章 绪 论 1](#_Toc41141832)

[1.1 研究背景及意义 1](#_Toc41141833)

[1.2国内外研究现状 1](#_Toc41141834)

[1.2.1国外研究现状 1](#_Toc41141835)

[1.2.2国内研究现状 1](#_Toc41141836)

[1.3 本文研究内容 2](#_Toc41141837)

[1.4 文章结构组织 2](#_Toc41141838)

[第2章 相关技术介绍 3](#_Toc41141839)

[2.1开发技术和工具简述 3](#_Toc41141840)

[2.2 Vue.js以及Nuxt.js前端框架介绍 3](#_Toc41141841)

[2.3 Spring后端框架介绍 4](#_Toc41141842)

[2.4 MySQL数据库介绍 4](#_Toc41141843)

[第3章 系统分析 5](#_Toc41141844)

[3.1 需求分析 5](#_Toc41141845)

[3.2 使用角色分析 5](#_Toc41141846)

[3.3 功能性需求 6](#_Toc41141847)

[3.4 性能需求 11](#_Toc41141848)

[3.4.1 速度 11](#_Toc41141849)

[3.4.2 容量 11](#_Toc41141850)

[3.4.3 负载 11](#_Toc41141851)

[第4章 系统设计 12](#_Toc41141852)

[4.1 常用系统设计方法 12](#_Toc41141853)

[4.2 系统架构 12](#_Toc41141854)

[4.3 功能模块设计 13](#_Toc41141855)

[4.3.1登录 13](#_Toc41141856)

[4.3.2 文件管理 14](#_Toc41141857)

[4.3.3 分享管理 14](#_Toc41141858)

[4.3.4 共享管理 14](#_Toc41141859)

[4.3.5 用户管理 14](#_Toc41141860)

[4.3.6 存储空间管理 14](#_Toc41141861)

[4.3.7 角色管理 15](#_Toc41141862)

[4.3.8部门管理 15](#_Toc41141863)

[4.4 详细设计 15](#_Toc41141864)

[4.4.1登录 15](#_Toc41141865)

[4.4.2 文件管理 16](#_Toc41141866)

[4.4.3 分享管理 25](#_Toc41141867)

[4.4.4 共享管理 26](#_Toc41141868)

[4.4.5 用户管理 27](#_Toc41141869)

[4.4.6 存储空间管理 28](#_Toc41141870)

[4.4.7 角色管理 28](#_Toc41141871)

[4.4.8 部门管理 28](#_Toc41141872)

[4.5数据库设计 28](#_Toc41141873)

[4.5.1概念数据模型 29](#_Toc41141874)

[4.5.2物理数据模型 30](#_Toc41141875)

[4.5.3数据库表结构 30](#_Toc41141876)

[第5章 系统实现详情 36](#_Toc41141877)

[5.1 登录模块 36](#_Toc41141878)

[5.1.1 路由鉴权 36](#_Toc41141879)

[5.1.2 用户登录 38](#_Toc41141880)

[5.2 文件管理 39](#_Toc41141881)

[5.3 组织管理 44](#_Toc41141882)

[第6章 系统测试 47](#_Toc41141883)

[6.1 系统界面测试 47](#_Toc41141884)

[6.2 系统功能性测试 48](#_Toc41141885)

[6.2.1 登录功能测试用例 48](#_Toc41141886)

[6.2.2 用户上传文件测试 48](#_Toc41141887)

[6.2.3分享功能测试 49](#_Toc41141888)

[6.2.4共享功能测试 49](#_Toc41141889)

[6.2.5 用户管理功能测试 49](#_Toc41141890)

[6.3 测试总结 50](#_Toc41141891)

[第7章 结束语 51](#_Toc41141892)

[**参考文献** 52](#_Toc41141893)

[**致 谢** 54](#_Toc41141894)

第1章 绪论

## 1.1 研究背景及意义

根据我这四年的高校学习和生活，发现目前师生的文件交互主要依靠物理媒介（如移动硬盘等）或第三方平台 （如即时通讯工具、公共云存储等），虽然这给师生们提供了一定的便利，但是还有很多问题。比如网络问题，是一个很大的瓶颈。而且大部分文件共享是在校园内部进行的，使用外部网络就大大增加了延时，导致文件分享不能第一时间进行。除了通过外部网络，目前师生常用的工具还有物理媒介，这也有缺点，比如容量有限，需要随身携带，安全性等问题。

而且师生之间常常有这样的场景，比如教师发布课件，学生提交作业，学生在学校机房做实验，需要及时保存数据，学生在专周需要分组进行开发，这些都涉及到了文件的分享和共享。

## 1.2国内外研究现状

### 1.2.1国外研究现状

Amazon公司是最早推出云存储服务的企业，也是最成功的企业之一。Amazon公司于2006年开始对外出租存储服务，即简单存储服务，该服务是Amazon网络服务的一部分。Amazon网络服务由四个部分组成：简单存储服务，简单排列服务，简单数据库和弹性计算云。由于存储费用的低廉性，服务的稳定性，由此获得了大量客户的青睐。此外，Amazon还推出了弹性块存储技术，支持数据的持续性存储。

IBM于2009年推出了云计算领域的存储战略计划—“企业级智能云存储”。通过存储虚拟化和基于私有云的存储和归档技术，这项服务旨在为企业客户提供应用程序方面的支持。于其他云存储的提供商不同，IBM提出的解决方案可以对企业现有的基础架构进行整合。通过虚拟化技术和自动化技术，IBM可以帮助企业构建属于自己的云计算中心，实现企业硬件资源和软件资源的统一分配、部署、备份和监控管理。打破了应用对资源的独占，帮助企业内部实现云计算和云存储的理念。

### 1.2.2国内研究现状

清华大学的Corsair系统是国内众多高校开展的云存储研究中的典型案例，该系统实现了由分布式文件系统Carrie和数据共享服务系统Corsair组成，为该校师生提供个人数据存储、社区型数据分享以及公共资源数据下载等服务的云存储平台。

解放军理工大学的MassCloud系统是一个海量云存储平台。与传统的大规模存储系统相比，它具有构建成本低、性能高效可靠、使用简单方便的特点。该系统提供符合POSIX规范的访问接口，无论是哪种系统下的应用程序，都可以不经修改就将云存储当成自己的硬盘来使用，同时也提供了专用的API接口。

## 1.3 本文研究内容

本文的研究内容是设计并实现一个基于Web的云存储系统，基于软件工程生命周期，对该系统进行需求分析，概要设计，详细设计，数据库设计，系统实现，系统测试等阶段。在上述阶段中，需求分析通过需求说明和用例图的形式进行研究，概要设计使用功能模块图的形式，详细设计采用时序图和详细设计说明的形式，数据库设计建立系统的概念数据模型和物理数据模型。整个流程符合软件工程规范。

## 1.4 文章结构组织

第一章，绪论。主要介绍了研究的背景以及国内外研究现状。

第二章，介绍了系统的开发环境，以及所使用的技术，并对技术做了一个简单的介绍。

第三章，系统需求分析，对系统用例进行了比较深入的分析。

第四章，系统设计，包括概要设计，详细设计，数据库设计，对部分核心功能使用时序图论述其业务流程。

第五章，描述系统实现的详情，给出了系统的界面图，并用流程图对核心功能的详细流程进行介绍，还给出了部分功能的实现代码。

第六章，系统的测试，介绍了系统测试的必要性，以及做了界面测试和用例测试。

第2章 相关技术介绍

## 2.1开发技术和工具简述

本系统的开发环境为Windows 10操作系统，本系统开发中用到的工具包括：

* + IntelliJ IDEA 2019.3 x64。
  + Microsoft Visual Studio Code
  + Microsoft Visio
  + Navicat

本系统开发技术包括：

后端：SpringBoot, MyBatis, MySQL, Redis, Maven

前端：Vue.js,Nuxt.js, ElementUI

前后端数据交互：Axios

在开发时，使用SpringBoot内嵌的Tomcat作为容器，并打包为jar形式，实际部署时，采用外部的Tomcat, 需要排除内嵌的Tomcat, 并修改SpringBoot项目的主类。

## Vue.js以及Nuxt.js前端框架介绍

Vue是一套用于构建用户界面的渐进式框架。与其它大型框架不同的是，Vue 被设计为可以自底向上逐层应用。Vue 的核心库只关注视图层，不仅易于上手，还便于与第三方库或既有项目整合。另一方面，当与现代化的工具链以及各种支持类库结合使用时，Vue 也完全能够为复杂的单页应用提供驱动。

Nuxt.js 是一个基于 Vue.js 的通用应用框架。通过对客户端/服务端基础架构的抽象组织，Nuxt.js 主要关注的是应用的 UI渲染。Nuxt.js的目标是创建一个灵活的应用框架，你可以基于它初始化新项目的基础结构代码，或者在已有 Node.js 项目中使用 Nuxt.js。Nuxt.js 预设了利用 Vue.js 开发服务端渲染的应用所需要的各种配置。除此之外，Nuxt.js还提供了一种命令叫：nuxt generate ，为基于 Vue.js 的应用提供生成对应的静态站点的功能。Nuxt.js为这个命令所提供的功能，是向开发集成各种微服务（Microservices）的 Web 应用迈开的新一步。作为框架，Nuxt.js 为 客户端/服务端 这种典型的应用架构模式提供了许多有用的特性，例如异步数据加载、中间件支持、布局支持等。

## Spring后端框架介绍

Spring 于 2003 年成立，是对早期J2EE规范复杂性的回应。虽然有些人认为 Java EE 和 Spring 处于竞争中，但实际上，Spring 是对 Java EE 的补充。 Spring 编程模型不包含 Java EE 平台规范；相反它集成了从Java EE 下精心挑选出的来规则：

* Servlet API(JSR 340)
* WebSocket API(JSR 356)
* Concurrency Utilities(JSR 236)
* JSON Binding API(JSR 367)
* Bean Validation(JSR 303)
* JPA(JSR 338)
* JMS(JSR 914)
* 以及必须要时用于事务协调的 JTA/JCA

Spring Framework 还支持Dependency Injection(JSR 330)和 Common Annotations(JSR 250)规范，应用程序开发人员可以选择使用这些规范，而不是 Spring Framework 提供的 Spring-specific 机制。

随着时间推移，Java EE 在应用开发中的作用已经变得更重要。在 Java EE 和 Spring 的早期，应用被创建并部署在服务器上。而今天，在 Spring Boot 的帮助下，应用是以可持续的方式创建的，嵌入了 Servlet 容器并且很容易更改。

Spring 保持不断的创新和发展。除了 Spring Framework 之外，还有其他项目，例如 Spring Boot，Spring Security，Spring Data，Spring Cloud，Spring Batch 等等。

## MySQL数据库介绍

MySQL是最流行的开放源码SQL数据库管理系统，它是由MySQL AB公司开发、发布并支持的。MySQL AB是由多名MySQL开发人创办的一家商业公司。它是一家第二代开放源码公司，结合了开放源码价值取向、方法和成功的商业模型。数据库是数据的结构化集合。它可以是任何东西，从简单的购物清单到画展，或企业网络中的海量信息。要想将数据添加到数据库，或访问、处理计算机数据库中保存的数据，需要使用数据库管理系统，如MySQL服务器。计算机是处理大量数据的理想工具，因此，数据库管理系统在计算方面扮演着关键的中心角色，或是作为独立的实用工具，或是作为其他应用程序的组成部分。

第3章 系统分析

## 3.1 需求分析

需求分析是整个软件生命周期过程中的一个比较重要的过程，在该阶段，系统分析人员收集信息，分析问题，整合观点，记录需求并验证其正确性，最终描述出软件被应用后与其环境互动形成的期望效应。

需求分析阶段有三个主要任务：

* 需求分析必须说明软件系统将被应用的环境及其目标，说明用来达到这些目标的软件功能，即要说明软件“需要做什么”和“为什么需要做”。
* 需求分析必须将目标和功能反映到软件系统中，映射为可行的软件行为，并对软件行为进行准确的规格说明。
* 现实世界是不断变化的世界，因此需求工程还需要妥善处理目标和功能随着时间演化的变动情况。

## 3.2 使用角色分析

系统的使用角色如图3-1所示：

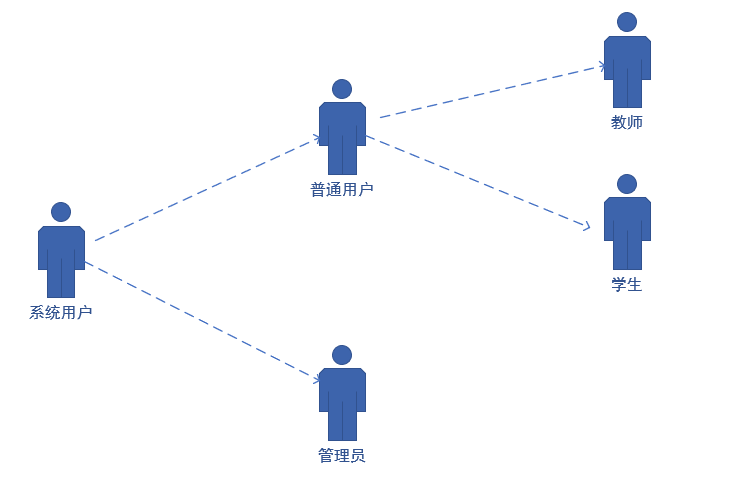
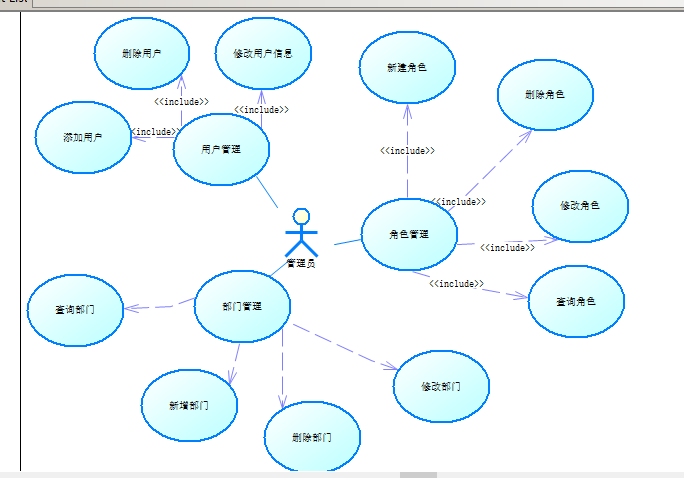


图3-1 系统使用角色

## 功能性需求

系统用例图如下图所示:

图3-2系统管理员用例图

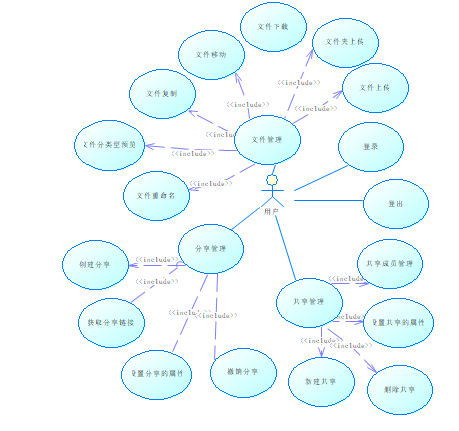


图3-3系统用户用例图

1. 登录模块

登录模块是最基本的模块，包括登录和登出两个功能，只有用户在登录后，系统后台保存有用户的登录信息，那么用户才能够使用几乎所有其他的系统功能。系统登出目的是销毁用户的登录凭证，避免用户的账号被他人恶意使用。用户每次重新登录后都会产生不一样的凭证。如果用户直接关闭浏览器退出，那么系统也应该在一定时间内销毁用户的登录凭证，目的是为了确保用户账号的安全性。

1. 文件管理模块

该模块包括文件上传，文件夹上传，文件下载，文件移动，文件复制，文件重命名，文件分类浏览。文件上传需要实现秒传，断点续传，即如果文件已经存在，那么用户无需再上传，如果部分存在，要求用户只上传不存在的部分，如果确实不存在，才要求用户上传整个文件。对于前两种情况，同样需要保存用户与文件的关联关系。

1. 分享管理

分享管理模块包括创建分享，获取分享链接，设置分享的属性，撤销分享四个功能。其中，分享属性里面包括分享的有效期以及口令。如果分享具有口令，那么系统要求用户必须输入对应的口令才能访问分享空间。

1. 共享管理

该模块包括用户新建共享，删除共享，设置共享的属性，管理共享成员。其中共享属性包括共享的名称，头像，可读权限，可写权限。用户可以申请加入共享，共享的创建者具有删除共享的权限，并且可以设置用户的类型是管理员还是普通用户。管理员以及创建者不受共享权限的约束。

1. 用户管理

该模块对于具有用户管理权限的用户可以执行添加用户、删除用户、修改用户信息（如密码）等操作，可以设置或撤销用户的管理员角色。

1. 系统空间管理

该模块包括一个功能，就是处理用户的空间扩容申请，具有该权限的管理员可以执行该操作。

1. 角色管理

该模块对于具有角色管理权限的管理员可以执行查询角色，新建角色，删除角色，修改角色等操作。

1. 部门管理

该模块对于具有部门管理权限的用户可以执行查询部门、新增部门、删除部门、修改部门等操作。

## 性能需求

### 3.4.1 速度

用户访问系统，首屏应该在1s内渲染完页面，用户点击路由跳转时间不应该超过1s。后台数据库查询及数据处理的时间不应该超过5s。

### 3.4.2 容量

系统磁盘容量应该能够满足所有用户的可用空间的总和。数据库容量至少应该能够存储100万条数据。

### 3.4.3 负载

系统应该能够满足所有用户同时进行文件的上传和下载。

第4章 系统设计

## 4.1 常用系统设计方法

软件设计的方法可以分为以下几种：结构化设计、面向对象程序设计、数据结构为中心的设计、基于构建的设计、形式化方法设计。

结构化设计是一个经典的软件设计方法，采取自上向下和逐步求精的思路，按照功能对系统进行分解。

面向对象设计的思想源于数据抽象和职责驱动，利用封装、继承、多态等方法，提高软件的可扩展性和可复用性。

数据结构为中心设计方法，开始于系统操纵的数据结构而不是它所表现的功能。软件工程师首先描述的是输入/输出的数据结构和基于这些数据结构的控制逻辑。

基于构件的设计重点在于构件的提供、开发和集成，以提高系统的可复用性。

形式化方法设计通过数学方法来对复杂的系统进行建模，通过严格的数学模型来验证系统的相关属性。

本系统的设计方法采用结构化设计方法。

## 4.2 系统架构

本系统采用前后端分离的架构，其核心思想是前端html页面通过异步请求调用后端的Restful API接口并主要使用json数据进行交互。

前后端分离的架构具有下列优势：

1. 可以实现真正的前后端解耦。
2. 在大并发情况下，可以同时水平扩展前后端服务器。
3. 减少后端服务器的并发/负载压力。
4. 提升开发效率，因为可以前后端并行开发，而不是像传统开发的强依赖。
5. 前端大量的组件代码得以复用，组件化。
6. 其他。

## 4.3 功能模块设计

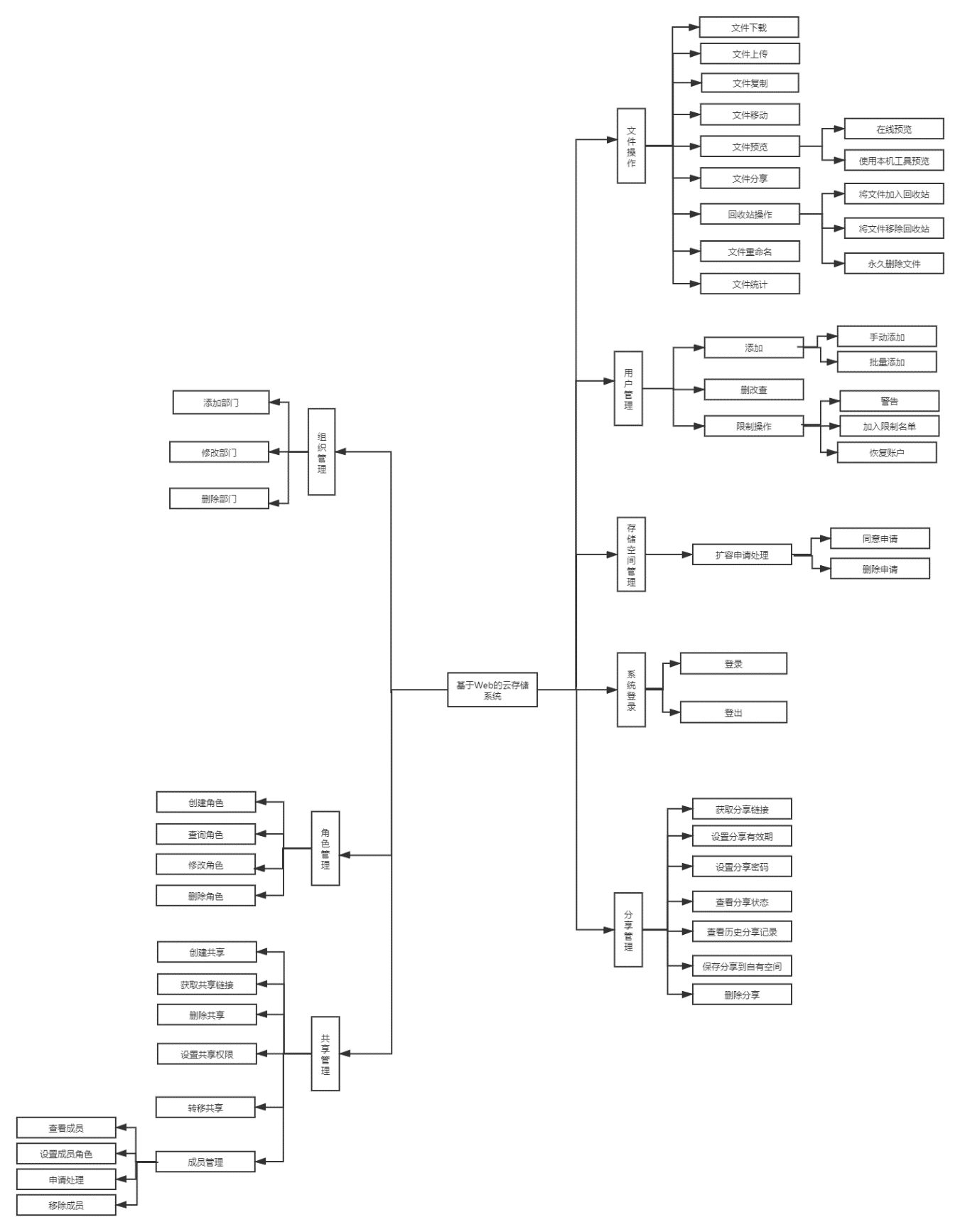


图4-1 系统总体功能模块图

### 4.3.1登录

此模块包括普通用户以及管理员的登录功能，未经登录的用户不能直接访问系统，系统需要能够拦截用户请求，并对用户当前状态做出判断，如果用户在一定时间内没有操作，为了安全起见，那么系统需要该用户重新登录。另外，被限制使用系统的用户不能登录。

### 4.3.2 文件管理

该模块是本系统的核心模块，主要包括用户文件上传，下载，复制，移动，重命名，回收站以及文件统计等操作。为了实现良好的用户交互体验，系统需要实时显示上传进度，并且可以由用户暂停或恢复上传进度。用户空间具有一定的大小，用户不能上传超过自己空间剩余容量大小的文件。上传功能包括文件上传和文件夹上传，上传文件时不能选择空文件，上传文件夹时只能选择单个文件夹，而且在同一目录下不能存在两个相同名称的文件夹，其实，用户空间内部每一个文件夹的路径都是唯一的。但同一目录下，可以存在相同文件，或同一文件的不同命名。其实，这并非意味着系统磁盘上存储了两个同样的文件，只是两条不同的用户与文件的关联关系记录。 也就是说，系统对于重复上传的文件，不要求用户上传，只需要在数据库中添加一条记录就行了，这就是所谓的“秒传”。同样，对于文件的移动，只需要修改所移动的目录子树的根节点。文件统计是管理员查看用户所上传的文件的详情，包括按照部门进行统计以及按照文件类型进行分类统计。

### 4.3.3 分享管理

该模块包括用户对文件进行分享的功能，用户可以选择自有空间内部的文件进行分享，对设置分享的权限及有效期，并且随时可以对分享进行管理，查看分享的访问次数，被保存次数，下载次数等，并在分享有效期内撤销分享。

### 4.3.4 共享管理

该模块包括用户对文件进行共享的功能，多个用户共同组成一个共享组，组内成员具有不同的角色，如组的创建者，组的管理员以及普通成员。组空间具有读写权限，但是组的创建者以及管理员不受权限的约束。这里的权限包括两类，只有写权限存在且未过期，用户才能上传文件到该目录，只有读权限开启，用户才能查看空间内部的文件详情。

### 4.3.5 用户管理

本系统不提供用户注册的功能，所有的用户需由管理员进行添加，包括手动添加以及自动批量添加，另外还包括用户查询，修改，删除等操作。用户自己可以对自己的信息进行修改。

### 4.3.6 存储空间管理

用户存储空间的大小有限，如果用户当前空间不够使用，那么可以提交用户空间扩容申请，填写目标空间大小以及申请理由，管理员可以根据用户的申请，选择通过或忽略该申请。

### 4.3.7 角色管理

该模块的功能主要由管理员使用，管理员可以通过设置角色来对后台的权限进行控制，每位用户只能关联一种角色，一种角色可以关联多个用户，一种角色代表一类权限的组合，一种权限代表一类系统界面菜单选项。

### 4.3.8部门管理

该模块提供给管理员实现部门的管理，如部门的添加，修改，修改包括对部门名称修改或修改部门的父子级关系。另外，文件统计部分中有一项统计指标也是根据部门来统计。

## 4.4 详细设计

### 4.4.1登录

用户访问系统的任意界面（无需鉴权的请求除外）时，都伴随着权限的鉴定，如果当前用户的登录状态存在，那么允许用户访问，如果不存在与当前用户关联的登录状态，那么系统会将页面重定向到登录页面要求用户登录。用户输入用户名和密码，首先要经过前端验证，用户名和密码不能为空，如果为空，前端可以直接提示登录信息不合法。验证通过后才向后台发起请求，后台同样需要验证，如果信息不为空，那么根据用户ID查询数据库，如果该用户记录不存在，则返回错误响应。如果该用户账户被限制使用，那么同样返回错误响应。接下来，就是进行密码匹配，需要用加密后的密码串比对数据库中的记录，如果一致，则说明登录成功，否则登录失败。并把登录状态写入数据库。如果是登录成功，系统为用户生成token, 并和登录信息一并保存到Redis中, 并把token写入响应头的cookie字段中。

登录过程的时序过程如图4-2所示：

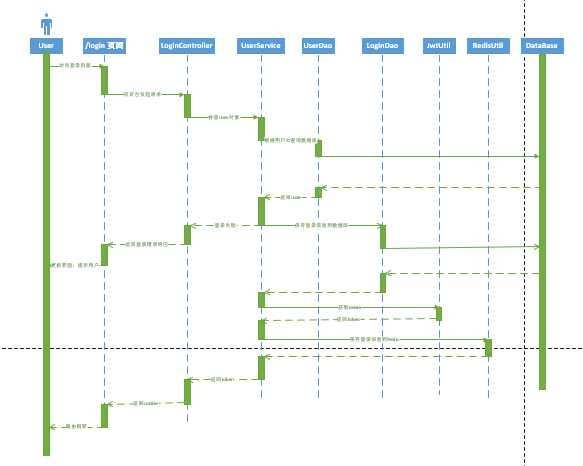


图4-2 登录过程时序图

### 4.4.2 文件管理

1. 文件上传

首先用户选择需要上传的文件，然后系统缓存待上传文件，并将其放到上传任务队列中，在时序图中对应” UploadFile”函数。之后，为了防止服务器请求量过大而宕机，根据并发上传文件的数量进行控制任务队列，只有在任务队列中的任务正常完成上传或者是报错后，才能将后续任务设置或活动任务，在时序图中对应” uploadHelper”函数。 接下来，系统计算文件的MD5值，为了防止直接对大文件进行计算而导致系统崩溃，所以采用分片计算，该分片值不宜过大，设置为1MB比较合适。经过测试，不同于计算MD5，上传时的分片不宜过小，否则会大大增加上传消耗的总时间，上传分片大小设置为5~10MB比较合适，。在时序图中对应”computeMD5”函数。然后系统向后台发送分片信息，包括文件的MD5值。后台返回两个值，一个是状态标志，一个是已上传分片的ID信息。前端首先判断状态标志，该标志共有三个取值：” success”表示文件已经完整存在，”null”表示文件不存在，”partial”表示部分存在，对于完整存在的情况，当前任务到此就结束了，这就实现了所谓的“秒传“。对于另外两种情况，需要绑定分块ID列表到当前任务中，便于下一步进行判断，如果状态位为”null”，那么也需要绑定一个空列表，便于后续的统一操作，这一步对应时序图中的”chunkCheck”函数。接下来，系统开始真正的上传，通过对整个文件进行分片， 然后遍历每一个分片，根据上一步获取到的分片ID，判断当前分片是否需要上传，直到所有的分片遍历完毕，该过程对应时序图中的”upload”函数。接下来，执行前端的最后一个步骤，即向后台发送合并分片的请求。

文件上传前端部分的时序图如下如所示：

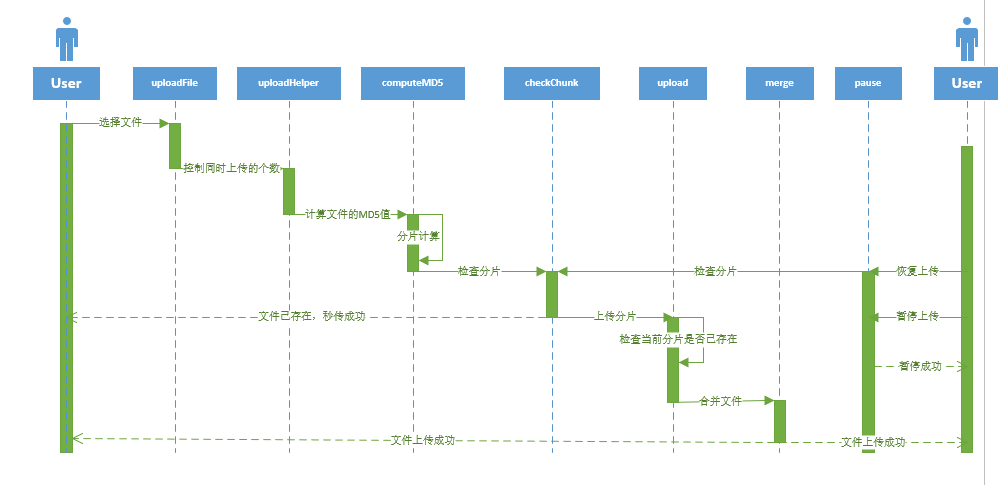


图4-3 文件上传前端部分时序图

根据前端请求的顺序，后端依次接收到“chunk/check”，“chunk/upload”, “chunk/merge”的请求。但是在最开始需要对用户鉴权，判断用户是否对目标目录具有上传的权限，这里的权限分为两部分，一是对应自有空间，二是对应组空间，对于第一种情况，直接判断当前目录是否属于当前用户即可，对于第二种，目标空间是共享空间，对比各种权限信息是否与当前用户吻合。如果不满足，将响应头状态置为401（unauthorized），前端收到响应后会拒绝继续上传该文件，并提示用户无权限。如果用户具备相应的权限，系统会先根据当前待上传文件的md5值，找数据库中是否存在对应的文件记录，如果没有，则创建并保存当前文件的信息，然后返回。如果对应记录已存在，而且完整性标记为完整，那么表示系统内部已经存在完整的文件数据了，直接返回并通知前端无需上传。第三种情况是，存在文件记录，但是文件不完整，系统通过返回文件的存储路径访问磁盘，查询分片记录，这里的分片记录命名格式为“文件md5值-分片序号”,读取后返回，前端根据此分片信息决定哪些分片需要上传。如果用户上传超过空间剩余大小的文件，系统应该拒绝上传。

文件上传的后端部分的时序图如下图所示：

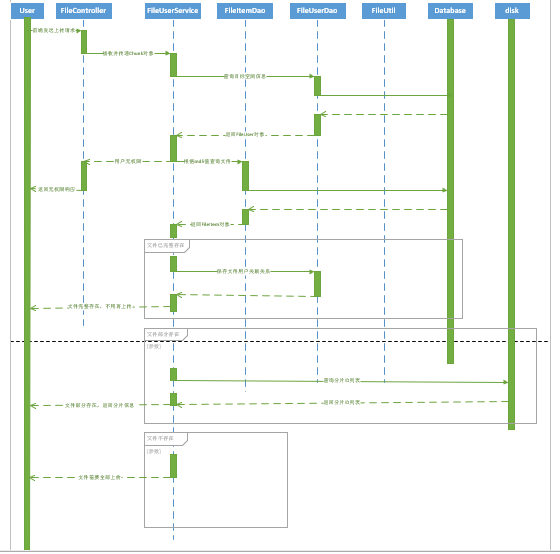


图4-4 文件上传后端部分时序图

2. 文件下载

文件下载，用户需要在页面选择要下载的文件，将ID统一发送到后台，注意，这里的ID不是文件对象的ID，而是文件-用户关联关系记录的ID，后台FileController收到ID后，转发数据给FileUserService, 该类中的download方法会先根据ID，调用FileUserMapper的selectByPrimaryKey方法依次查询每个关联关系，和上传一样，这里包括两种情况，一种是自有空间，另一种是组空间，如果是自有空间，那么直接访问磁盘，读取文件流，并返回给FileController写入到HttpServletResponse对象的响应流中，并设置响应头“Content-Type”为“application/octet-stream”，“Content-Disposition”为“attachment;filename=？”（这里的”?”指的是具体的文件名），第一个响应头的作用是，告诉浏览器，响应内容为应用程序文件；第二个响应头的作用是告述浏览器该响应作为附件处理，浏览器的实际行为是会弹出下载窗口，让用户选择保存目录。如果是组空间，那么需要判断当前是否存在读权限，如果是，才能同意用户下载，如果不是，那么拒绝用户的下载，并返回错误信息。如果用户选择多个文件或目录，那么系统先在本地创建一个临时目录，保存所有的文件，并打包成为一个zip压缩文件，将该压缩文件写入到响应流中。

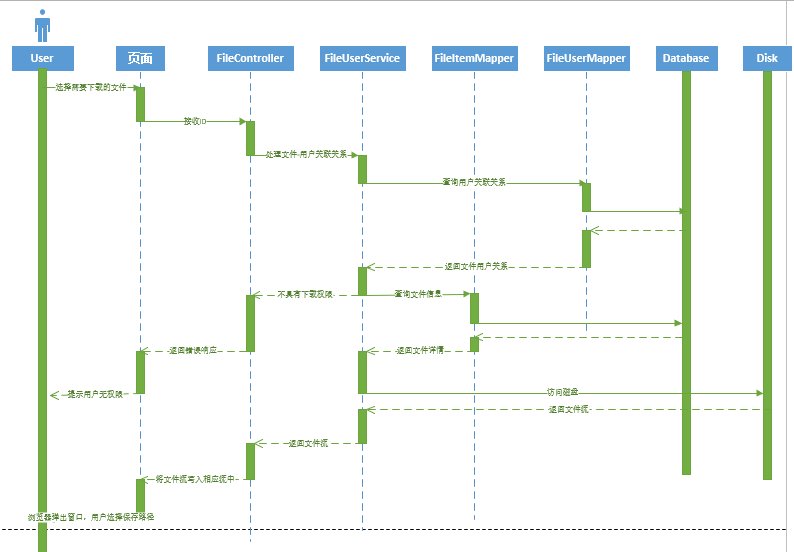


图4-5 下载文件时序图

3. 文件移动

文件移动，用户需要选择被移动的文件（文件夹），以及目标文件夹，后台FileController接收到数据后，传递数据到FileUserService中的move方法，该方法首先根据传入的值调用FileUserMapper中的selectByPrimaryKey方法查询文件-用户关联关系是否存在，如果不存在，直接返回错误响应。如果存在，判断目标文件是否是文件夹，如果不是，则返回响应并告知用户目标空间不是目录，而是文件，不能写入。然后，首先判断用户的空间类型，如果是组空间，判断源文件是否有读权限，目标目录是否有写权限，如果有一项条件不满足，那么拒绝操作。如果所有的条件都满足, 调用FileUserMapper中的updateByPrimaryKey方法修改该条记录。

例如对于下图，假如要移动目录B到目录A下，只需修改数据库中B记录的parent\_id属性即可，类似改变指针指向的操作。

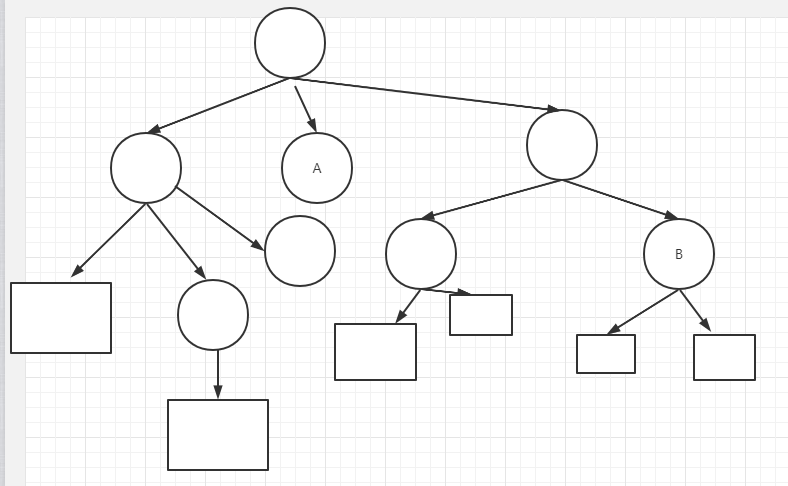


图4-6 文件树例图（圆形表示目录，矩形表示文件）

文件移动的时序图如下图所示：

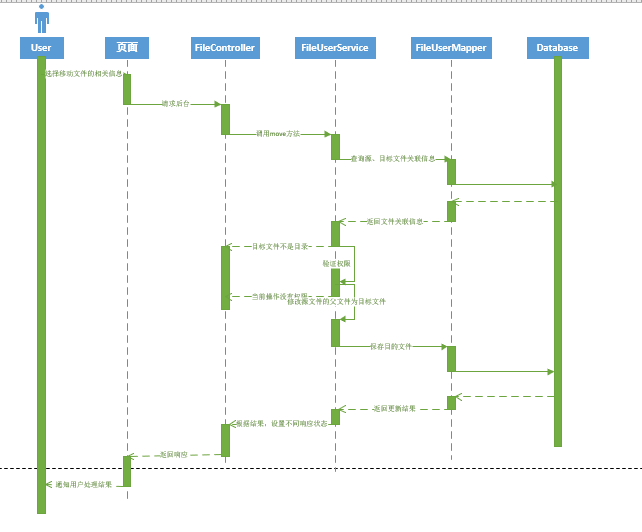


图4-7 文件移动时序图

4. 文件复制

文件复制，不同于在本地系统文件资源管理器的操作，分为复制和粘贴步骤。本系统将两个操作合并为一个，言下之意就是选择复制操作时，必须同时选择源文件和目标文件，同样，用户需要对源文件具有读权限，对目标文件具有写权限，才能满足权限要求。首先，用户在页面选择目标文件和源文件，请求后台FileController，然后传递数据给FileUserService中的copy方法，该方法根据源、目标文件的ID查询文件-用户关联关系。如果两者都存在，且目标文件是目录类型，那么才能执行接下来的操作，否则直接返回并响应错误信息。然后，根据上一步的查询操作，判断用户对源文件是否有读权限，对目标文件是否有写权限，如果两者都满足，才能执行下一步操作，否则返回响应并通知用户当前操作无权限。接下来，需要修改源文件子树内部的信息，下图中所示为一棵目录树，

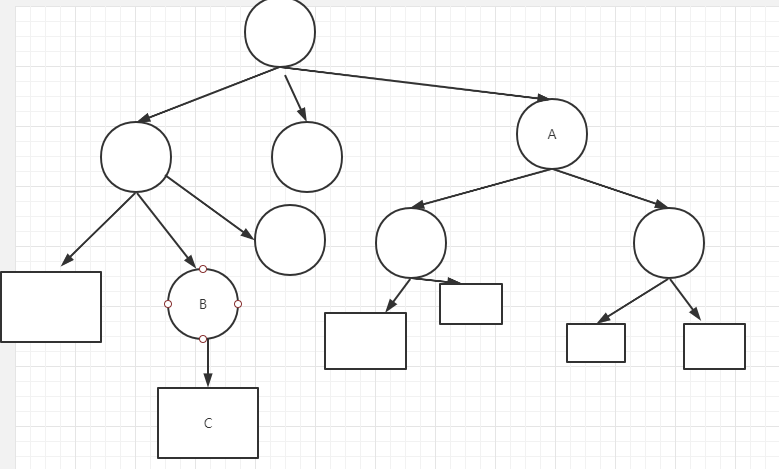


图4-8 文件树例图（圆形表示目录，矩形表示文件）

假如需要把B目录拷贝到A目录下，那么需要先查询出B目录子树中每一个节点，然后将其的“path”属性修改为在A目录下的路径信息。该步骤采用层次遍历方法遍历子树，调用fileUserMapper中的selectByParentIds方法，该方法会返回B目录下的子元素（这里指的是直接关联的子元素，而不是孙元素之类的），然后每遍历一层将结果写入数据库，同时将所有ID保存到一个临时列表中，待遍历完毕后，根据列表中的数据，再去查询新的子元素，循环直到临时列表为空，则表示子树遍历完毕。写入完毕后，返回操作结果，FileController根据操作结果设置不同的响应状态，前端通过响应状态对用户发出不同类型的通知。

复制文件的时序图如下：

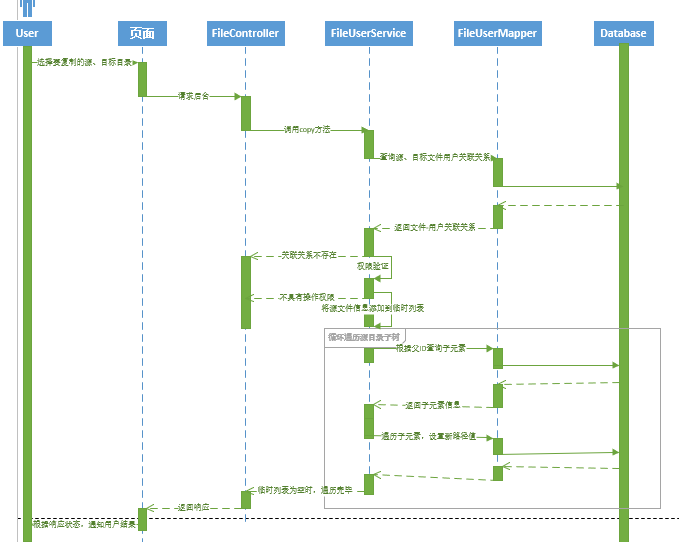


图4-9 复制文件时序图

5. 查询目录树

上面第3、4个功能都需要用到本功能，该功能查询目录，生成目录树并在前端显示，让用户选择移动或复制的目标目录。由于系统使用的前端UI框架对树型结构的特定数据类型限制，所以在后台需要将FileUser对象转化为所要求的类型，Node表示前端显示的树节点。为了减少数据库的查询次数，这里将该空间内部所有目录一次性查询出来，然后在内存中构建树。首先在列表中找到根目录，然后通过深度遍历，递归构建目录树。

6. 文件重命名与新建文件夹

文件重命名与复制文件中的修改子树中每个节点的“path”属性的操作类似，用户选择需要重命名的文件，输入新名称，系统除了要判断用户的权限操作之外，这里如果文件是目录的话还需要注意，新名称不能和当前目录下已存在的目录名称相同，因为用户查询文件的时候，是根据路径来查询的，如果此路径不唯一，就会造成查询错误的结果。对于新建文件夹也类似，必须确保每一级文件夹的名称不重复性。

7. 回收站操作

回收组操作包括三类，加入到回收站，从回收站移除，永久删除回收站内的文件。对于前两类操作，只需要更改数据库中的“is\_delete”字段即可。第三类直接删除即可。在FileUserService中设置两个方法，“handleRecycle”和“delete”方法，“handleRecycle”需要接收一个标志位参数，该参数值仅限0或1，分别表示从回收站移除和加入到回收站，该值也对应了数据库中“is\_delete”字段的值。删除回收站文件操作通过调用fileUserMapper中的delete方法即可。如果删除的文件属于组空间，那么操作用户只能是管理员或者组的创建者。

加入到回收站和从回收站移除的操作的时序图如下图所示：

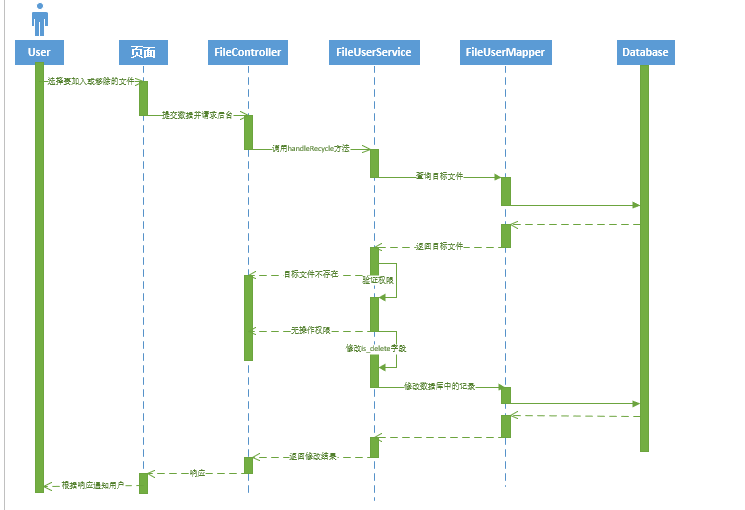


图4-10 回收站操作时序图

### 4.4.3 分享管理

用户只能对自有空间的文件进行分享，不能对组空间内的文件进行分享。

1. 创建分享

用户创建分享时需要指定需要分享的内容，可以是单个文件，也可以是目录，同时需要指定是否需要口令以及分享的有效期。用户选择后提交数据到后台ShareController，然后调用ShareService中的create方法，首先进行一系列的权限验证，包括分享的内容是否是属于操作本人的自有空间。如果不是，直接返回错误响应。否则，新建Share对象，然后调用ShareMapper的insert方法。最后返回该分享的ID，前端将接收到的ID拼接成为URL, 并回显给用户，用户可以选择复制链接，从而分享给其他人。

2. 查看分享与取消分享

用户可以查看自己创建的当前有效的分享，并对某些分享执行取消分享的操作。首先，系统登录后，点击主界面的我的分享菜单，前端路由跳转到我的分享主界面，在该界面的created阶段（Vue的生命周期阶段之一），向后台请求数据，分页显示历次以来的分享记录，默认显示第一页。后台ShareController接收到请求后，把页码和页大小参数传递给ShareService中的selectAll方法，该方法使用MyBatis的分页插件PageHelper进行分页，然后调用ShareMapper查询与该用户相关的记录。查询完毕后，返回到前端，前端再进行页面渲染。

3. 访问分享

在用户创建分享后，系统会为该分析生成一个id, 该id是分享的URL的组成部分，前端根据该ID拼接成URL, 用户可以通过该URL访问到该分享。

首先用户在地址栏输入分享对应的URL，会立即请求后台ShareController, 调用ShareService中的getShare方法，判断该分享是否需要口令，如果是，直接返回到前端页面，然后重定向到输入口令的页面，用户输入口令并提交后，后台再次验证，调用ShareService中的checkShare方法，验证口令是否一致，并将验证结果返回到前端，如果口令一致，那么跳转到该分享的页面，如果口令不正确，系统提示用户，不执行路由跳转操作，用户可以选择继续输入。

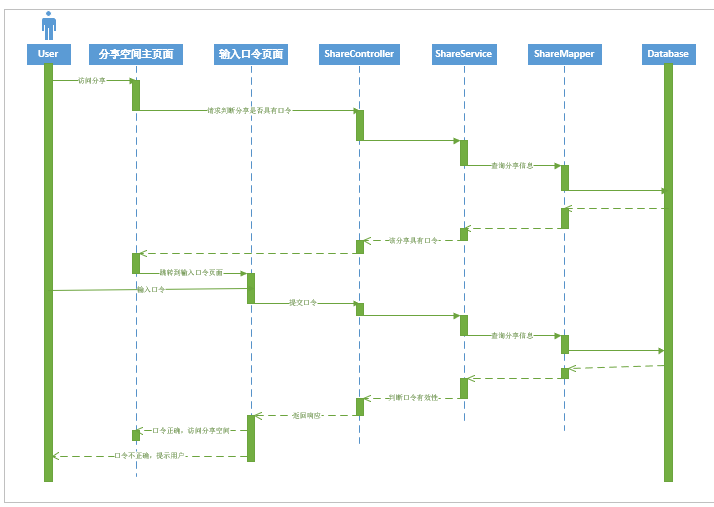


图4-11 用户访问分享时序图

4. 保存分享

用户访问了分享过后，可以查看分享的内容，如果有必要，用户可以选择将内容保存到自己的空间。这里又调用了上一个模块中的查询目录树的操作。这里查询的是访问者的目录树，而不是分享者的目录树。用户选择需要保存的文件信息然后提交到后台ShareController，然后调用ShareService中的saveShare方法。和上一个模块中多次论述到的子树操作一样，需要查询出所有的子树元素，并复制一份，设置新的路径信息后保存到数据库中。

5. 下载分享

用户访问了分享过后，可以选择直接下载，该操作本质上还是文件管理模块的文件下载操作，这里不再赘述。

### 4.4.4 共享管理

1. 创建共享

用户登录页面后，可以点击我的共享目录，跳转到我的共享页面，系统会分页显示当前用户加入的共享组。另外，用户可以点击创建共享按钮，点击后，系统弹出一个对话框，要求用户输入一些共享的属性，比如名称，介绍，头像，权限等。输入完毕后点击确认按钮提交到后台GroupController，再调用GroupService中的createGroup方法，该方法创建一个Group对象，并调用GroupMapper中的insert方法插入到数据库中。操作完毕后，和创建分享类似，返回新建的共享ID到前端后，前端根据该ID拼接成URL，用户可以通过分享该URL邀请其他用户加入。

2. 成员管理

用户访问某个组的URL，系统会先判断当前用户是否已经加入了该组，如果没有，那么会要求该用户先加入，用户需要提交申请加入的请求到后台GroupController, 然后调用GroupService中的createGroupUser方法，创建一条用户与组的关联并设置状态为未通过，然后调用GroupUserMapper中的insert方法保存到数据库。等待共享组的创建者或管理员审核通过后，才能访问该组的空间。加入该组后，用户可以查看到所有的该组成员，并且组的创建者可以选择设置某些用户为管理员，管理员不受该组权限的约束，可以对组空间进行任意的修改。组的创建者和管理员可以将某些组成员踢出该共享组。

3. 共享删除

只有共享组的创建者有删除该共享的权限，删除后，数据库中的该条Group记录以及该Group对应的Group-User记录、Group-File记录都将被删除。

### 4.4.5 用户管理

该模块主要由具有用户管理权限的管理员进行操作。

1. 添加用户

管理员登陆后台系统后，可以在显示的主界面的菜单栏中点击用户管理。会跳转到用户管理页面。管理员可以选择添加单条数据，该操作适合用户量较少的情况下使用，更多的时候推荐使用批量添加，即选择用户数据的excel文件上传，系统自动解析该文件，并保存记录到数据库中。这里的excel文件需要与用户的某些字段关联，否则系统会拒绝添加。

2. 修改用户信息

该功能可由管理员，也可由用户操作。但是有一些差异，用户可以修改用户自身的所有信息，但是管理员只有在用户忘记密码的时候，帮用户重置密码。

3. 删除用户

某些用户不再使用系统时，管理员可以执行删除用户，管理员先选择需要删除的用户部门，比如对于学生用户来说，毕业生离校后，不再使用该系统，管理员可以选择毕业班级，所有属于这些班级的用户都会被删除。

4. 限制用户

某些用户在使用过程中可能违反系统的使用规范，管理员可以限制这些用户继续使用本系统。直接修改数据库中user表的“u\_limit”字段为1即可。修改后，在登录模块，用户将不能通过登录校验。

### 4.4.6 存储空间管理

该模块由具有存储空间管理权限的管理员进行操作。

用户文件上传是具有大小限制的，不能超过用户空间大小的限制，用户如果需要扩大存储空间容量，需要先提交请求，然后输入扩容后的空间大小以及申请理由，系统创建申请记录，并存入数据库中。管理员访问扩容申请处理页面，系统分页查询未被处理的申请，管理员然后选择每项申请进行处理。每一项申请有一个字段“status”表示该项申请的状态，0表示未处理，1表示通过，2表示拒绝。

### 4.4.7 角色管理

管理员登录后台系统，可以在菜单栏里面选择角色管理。角色管理包括查询角色，删除角色，修改角色，添加角色。

用户选择添加角色后，需要输入该角色的必要信息，比如角色的名称，以及选择所关联的权限。同样，可以修改角色的信息以及直接删除某项角色。角色删除后，与该角色所关联的用户的管理员身份都失效了，如果该身份需要延续，那么管理员需要为该用户关联其他角色。

### 4.4.8 部门管理

管理员登录后台系统后，可以在菜单栏里面选择部门管理。部门管理包括部门添加，部门修改。部门修改指的是调整部门的父部门属性，修改时必须确保父部门已经存在，否则系统拒绝添加。

## 4.5数据库设计

通过需求分析，可以确定系统的ER模型如下图所示：

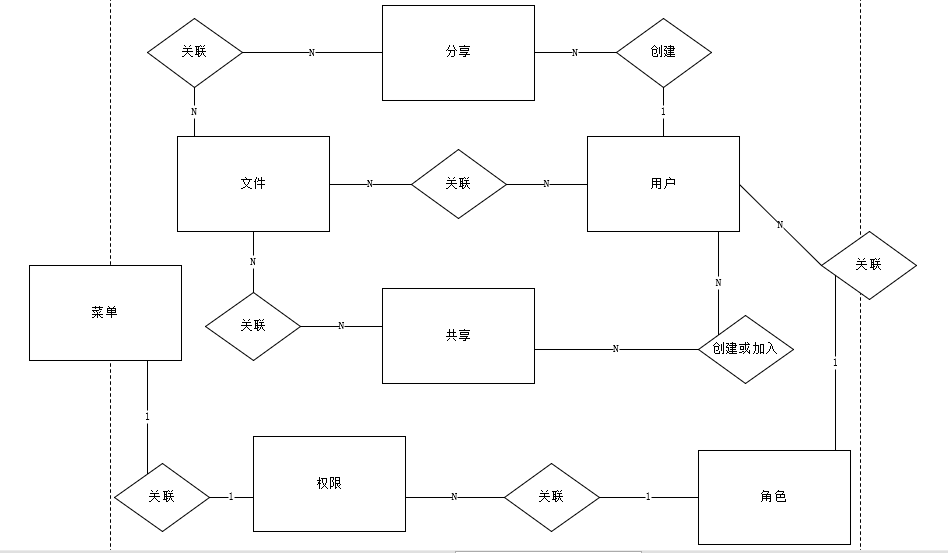


图4-12 系统ER图

上图中显示了本系统的几个实体，包括用户，文件，分享，共享，角色，权限，菜单。下面开始进行概念数据模型的建立，主要包括对各个实体的属性字段进行设置，并确定实体之间的关联关系。

### 4.5.1概念数据模型

概念数据模型重点考察需求中的实体，对实体的属性，标识符进行分析。实体一般是需求中抽象出来对应现实世界的事物。事物的属性指的是可以描述事物特性的指标。标识符指的是能够在多条记录中唯一标识事物的属性，一般由一个字段指定即可，但可以根据具体业务需求扩展到多个属性。

本系统的概念数据模型如下图所示：

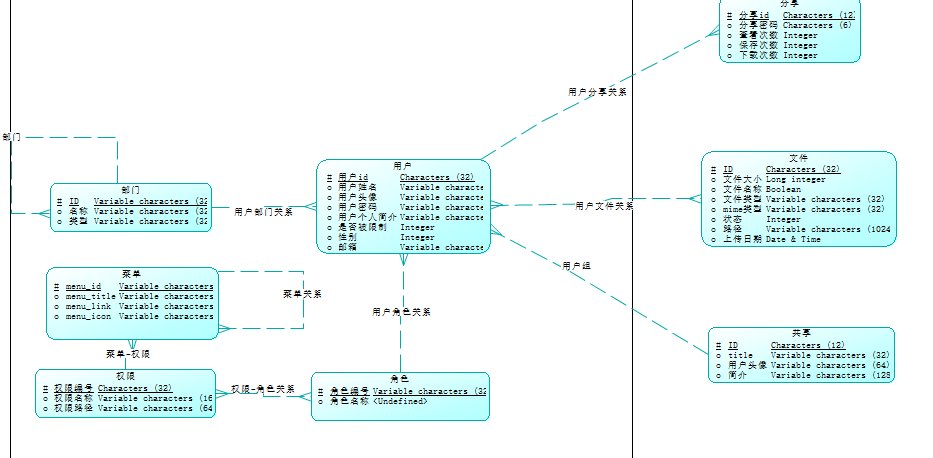


图4-13 系统概念数据模型图

### 4.5.2物理数据模型

物理数据模型在概念数据模型的基础上更加注重对实体之间关系的分析。描述数据是如何在计算机中存储的，如何表达记录结构、记录顺序和访问路径等信息。

本系统的物理数据模型如下图所示：

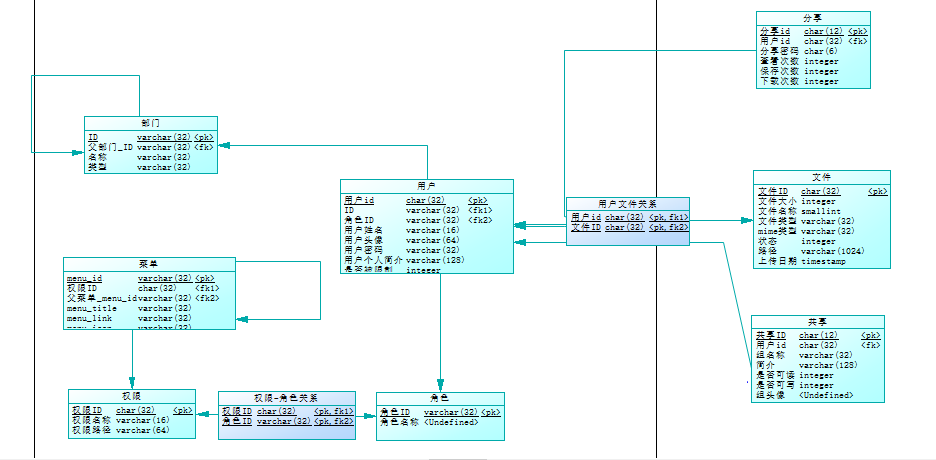


图4-14 系统物理数据模型

### 4.5.3数据库表结构

本节是对前两节的具体化，即实体以及实体的关系在计算机中的最终存储结构。

本系统所有涉及的数据库表都将在下面以表格形式展示，表格内容分为字段名称，字段类型，类型长度，是否为空，是否是主键，以及说明等列。

部门表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 长度 | 是否可为空 | 是否是主键 | 说明 |
| id | varchar | 32 | 否 | 是 | 部门ID |
| title | varchar | 64 | 否 | 否 | 部门名称 |
| d\_type | int | 1 | 否 | 否 | 部门类型，0表示学校，1表示学院，2表示专业，3表示班级。 |
| p\_did | varchar | 32 | 否 | 否 | 父级部门的ID，外键，部门表的主键 |

用户表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 长度 | 是否可为空 | 是否是主键 | 说明 |
| id | varchar | 32 | 否 | 是 | 用户ID |
| username | varchar | 32 | 否 | 否 | 用户名 |
| pwd | varchar | 32 | 否 | 否 | 用户密码 |
| gender | int | 1 | 是 | 否 | 用户性别 |
| avatar | varchar | 255 | 是 | 否 | 用户头像 |
| brief | varchar | 255 | 是 | 否 | 用户自我介绍 |
| email | varchar | 64 | 是 | 否 | 用户邮箱 |
| u\_type | int | 1 | 否 | 否 | 用户类型 |
| t\_ss | bigint | 20 | 否 | 否 | 空间总大小 |
| u\_ss | bigint | 20 | 否 | 否 | 空间已使用大小 |
| dept\_id | varchar | 32 | 否 | 否 | 部门id，外键，部门表的主键 |
| u\_limit | int | 1 | 否 | 否 | 受限 |
| role\_id | int | 11 | 是 | 否 | 角色id， 外键，角色表的主键 |

文件表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 长度 | 是否可为空 | 是否是主键 | 说明 |
| id | char | 32 | 否 | 是 | 文件id, 32位的文件md5值 |
| f\_name | varchar | 255 | 否 | 否 | 文件名称 |
| f\_size | bigint | 20 | 否 | 否 | 文件大小 |
| mime | varchar | 64 | 是 | 否 | 文件的mime类型 |
| f\_type | varchar | 64 | 是 | 否 | 文件的自定义类型 |
| status | int | 1 | 否 | 否 | 文件状态 |
| path | varchar | 10240 | 否 | 否 | 磁盘存储路径 |
| u\_date | datetime | - | 否 | 否 | 上传日期 |
| u\_id | varchar | 32 | 否 | 否 | 上传用户的id, 外键，用户表的主键 |

文件-用户关系表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 长度 | 是否可为空 | 是否是主键 | 说明 |
| id | char | 32 | 否 | 是 | 文件用户关系的ID |
| u\_id | u\_id | 32 | 否 | 否 | 用户id, 外键，用户表的主键 |
| f\_id | char | 32 | 是 | 否 | 关联的文件id，外键，文件表的主键 |
| f\_path | varchar | 10240 | 否 | 否 | 在所属空间内部的根目录 |
| f\_name | varchar | 256 | 否 | 否 | 文件的名称 |
| is\_folder | int | 1 | 否 | 否 | 是否是文件夹 |
| f\_pid | char | 32 | 是 | 否 | 父文件夹的id，引用该表的主键作为外键 |
| last\_update\_date | datetime | - | 否 | 否 | 上次更新日期 |
| f\_type | varchar | 32 | 否 | 否 | 文件类型 |
| is\_delete | int | 1 | 否 | 否 | 是否删除 |
| f\_size | bigint | 20 | 否 | 否 | 文件大小 |
| g\_id | char | 12 | 是 | 否 | 关联的共享id，外键，共享表的主键 |
| share\_id | char | 12 | 是 | 否 | 关联的分享id, 外键，分享表的主键 |

共享表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 长度 | 是否可为空 | 是否是主键 | 说明 |
| id | char | 12 | 否 | 是 | 共享的ID |
| title | varchar | 255 | 否 | 否 | 共享组的名称 |
| avatar | varchar | 255 | 是 | 否 | 共享组的头像文件存储路径 |
| brief | varchar | 1024 | 是 | 否 | 共享组简介 |
| limit | int | 1 | 否 | 否 | 共享组是否受限 |
| is\_writeable | int | 1 | 否 | 否 | 是否可写 |
| is\_readable | int | 1 | 否 | 否 | 是否可读 |

共享-用户关系表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 长度 | 是否可为空 | 是否是主键 | 说明 |
| id | char | 12 | 否 | 是 | 共享用户关系记录的id |
| g\_id | varchar | 32 | 否 | 否 | 共享组的id，外键，共享表的主键 |
| u\_id | int | 1 | 否 | 否 | 用户id，外键，用户表的主键 |
| gu\_type | int | 1 | 否 | 否 | 用户在组中的类型，0表示创建者，1表示管理员，2表示普通用户， |
| join\_date | datetime | - | 否 | 否 | 用户加入的日期时间 |
| status | int | 1 | 否 | 否 | 状态，表示该用户的申请是否被处理 |

登录表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 长度 | 是否可为空 | 是否是主键 | 说明 |
| u\_id | varchar | 32 | 否 | 否 | 登录用户的id, 外键，用户表的主键 |
| ip | varchar | 15 | 否 | 否 | 用户登录时的ip地址 |
| login\_date | date\_time | - | 否 | 否 | 登录日期时间 |
| status | int | 1 | 否 | 否 | 登录结果状态，0表示未登录成功，1表示登录成功 |

空间申请扩容记录表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 长度 | 是否可为空 | 是否是主键 | 说明 |
| id | int | 11 | 否 | 是 | 申请记录的id |
| exp\_size | bigint | 20 | 否 | 否 | 期待的空间大小 |
| reason | varchar | 2000 | 是 | 否 | 申请原因 |
| u\_id | varchar | 32 | 否 | 否 | 用户id,外键，用户表的主键 |

角色表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 长度 | 是否可为空 | 是否是主键 | 说明 |
| id | int | 11 | 否 | 是 | 角色的id |
| title | varchar | 32 | 否 | 否 | 角色的名称 |
| create\_date | datetime | - | 否 | 否 | 角色的创建日期 |
| last\_update\_date | datetime | - | 否 | 否 | 角色的上次修改日期 |

权限表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 长度 | 是否可为空 | 是否是主键 | 说明 |
| id | int | 11 | 否 | 是 | 权限的id |
| title | varchar | 32 | 否 | 否 | 权限的名称 |
| remark | varchar | 1024 | 否 | 否 | 权限的备注 |
| menu\_id | int | 11 | 否 | 否 | 权限关联的菜单的id, 菜单表的主键 |

菜单表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 长度 | 是否为空 | 是否是主键 | 说明 |
| id | int | 11 | 否 | 是 | 菜单的id |
| title | varchar | 255 | 否 | 否 | 菜单的名称 |
| link | varchar | 255 | 否 | 否 | 菜单的路径 |
| icon | varchar | 255 | 是 | 否 | 菜单的图标 |
| pm\_id | int | 11 | 是 | 否 | 父菜单的id |

角色-权限关联表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 长度 | 是否为空 | 是否是主键 | 说明 |
| r\_id | int | 11 | 否 | 是 | 角色的id, 外键，角色表的主键 |
| p\_id | int | 11 | 否 | 是 | 权限的id, 外键，权限表的主键 |

第5章 系统实现详情

通过对系统的需求分析和功能模块设计、详细设计、数据库设计后，开始逐步对系统各个模块进行实现，本章对主要模块的功能的实现过程及原理进行论述。

## 5.1 登录模块

### 5.1.1 路由鉴权

该部分虽不是与用户直接交互的功能模块，但是在系统正常运行中发挥着至关重要的作用。由于这部分与该模块关联性最大，所以放在此部分来论述。

为了防止用户未登录就访问系统的某些页面，本系统采用前端路由验证+后端路由验证的逻辑。整个路由鉴权的大体流程如下图所示：

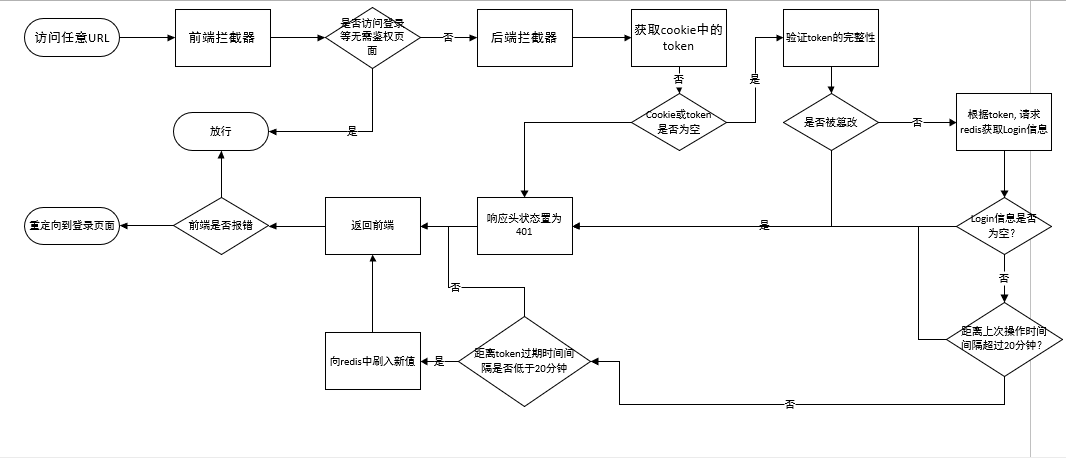


图5-1 路由鉴权流程图

前端路由鉴权实现代码：



图5-2 前端路由鉴权代码

后端拦截器实现代码：



图5-3 后端路由鉴权代码

### 5.1.2 用户登录

用户登录界面

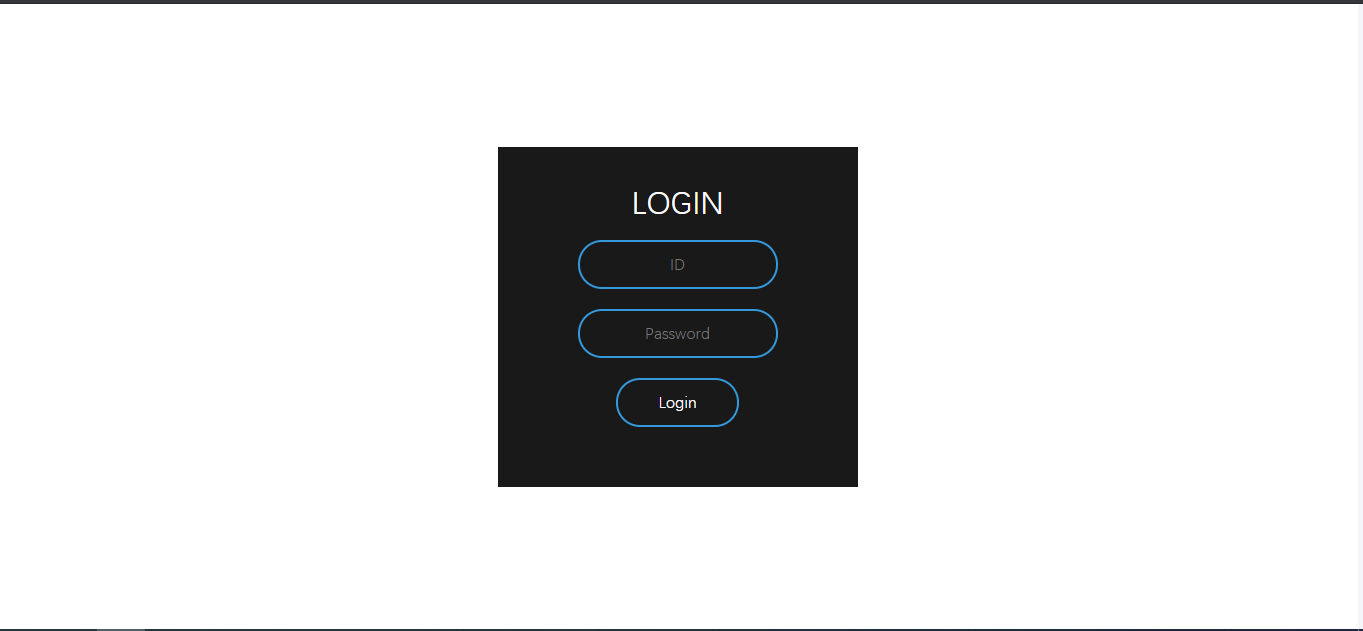


图5-4 用户登录界面图

用户登录操作的流程如下图所示：

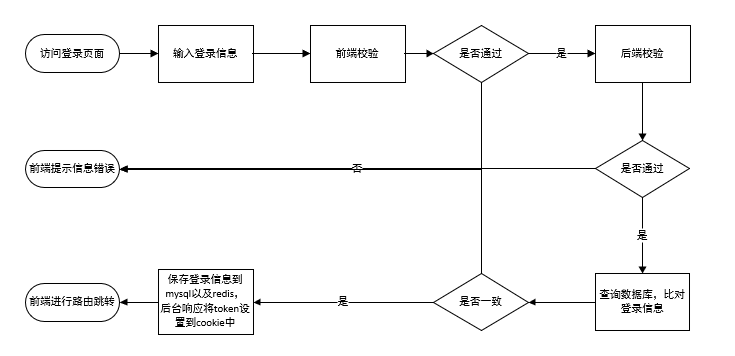


图5-5 用户登录流程图

## 5.2 文件管理

主窗口界面如下图所示：

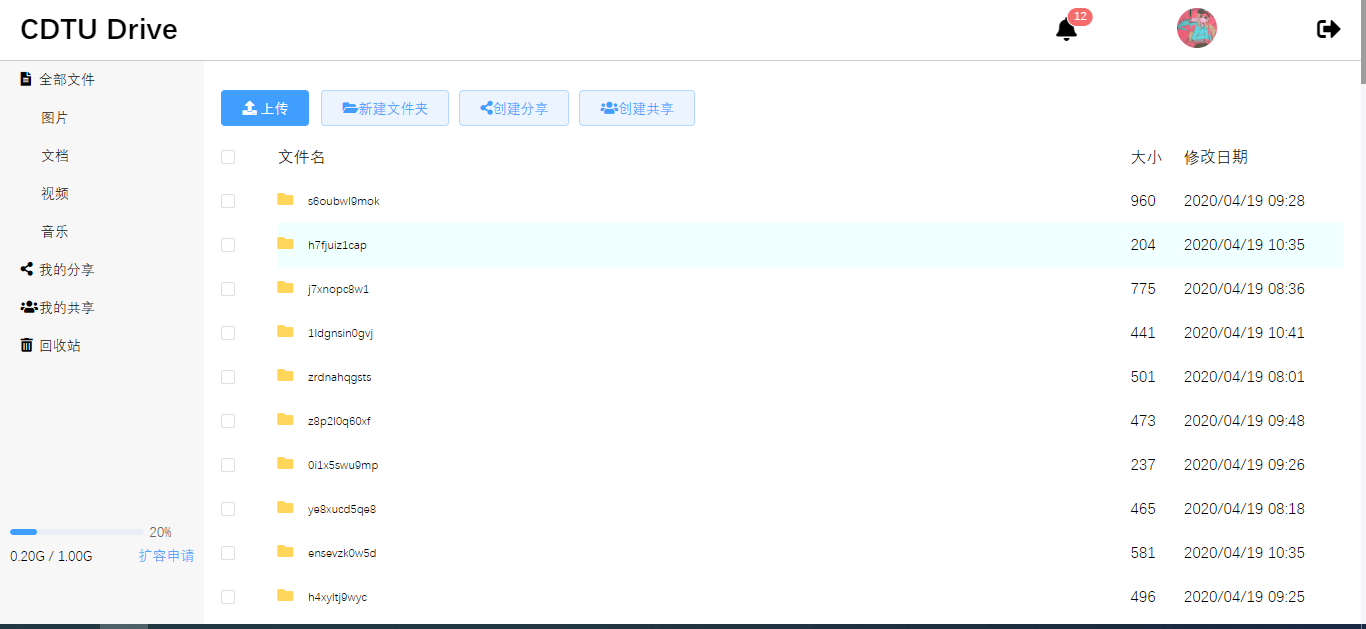


图5-6 主窗口界面图

用户点击上传按钮，选择上传文件还是文件夹，选择后，系统弹出上传信息面板。

上传信息面板显示实时的上传进度：



图5-7 文件上传信息面板界面图

上图中所示的为暂停中的上传任务，点击继续上传按钮，系统会继续上传。

由于文件上传的流程稍有些复杂，整个流程分为前后端两部分来展示：

前端上传逻辑流程：

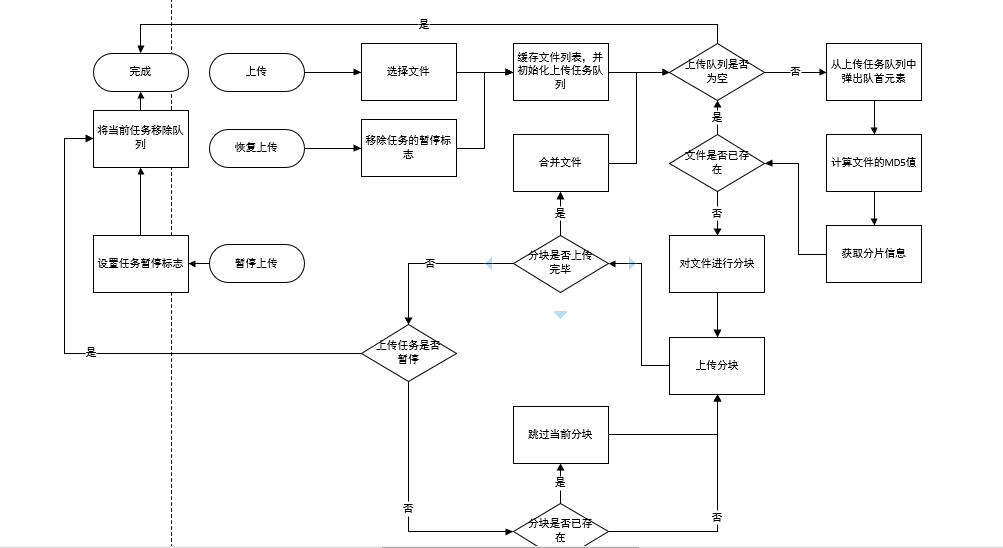


图5-8 前端文件上传流程图

后端上传逻辑流程：

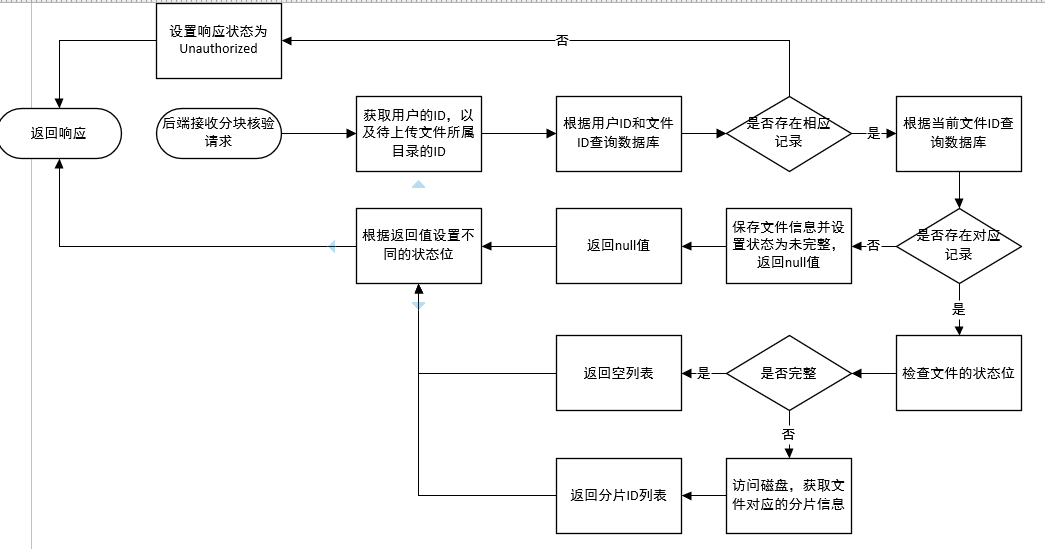


图5-9 后端文件上传流程图

文件上传前端部分的代码：

分块检验函数：



图5-10 分块校验函数代码

分块上传函数：



图5-11 分块上传函数

文件上传的后端部分代码：

合并分块的方法：



图5-12 后端分块合并方法代码

## 5.3 组织管理

该部分的界面如下图所示：



图5-13 后台管理员查看部门（表格）

上图展示的是部门信息的表格显示模式，可以点击按钮，查看详细的部门树结构信息：

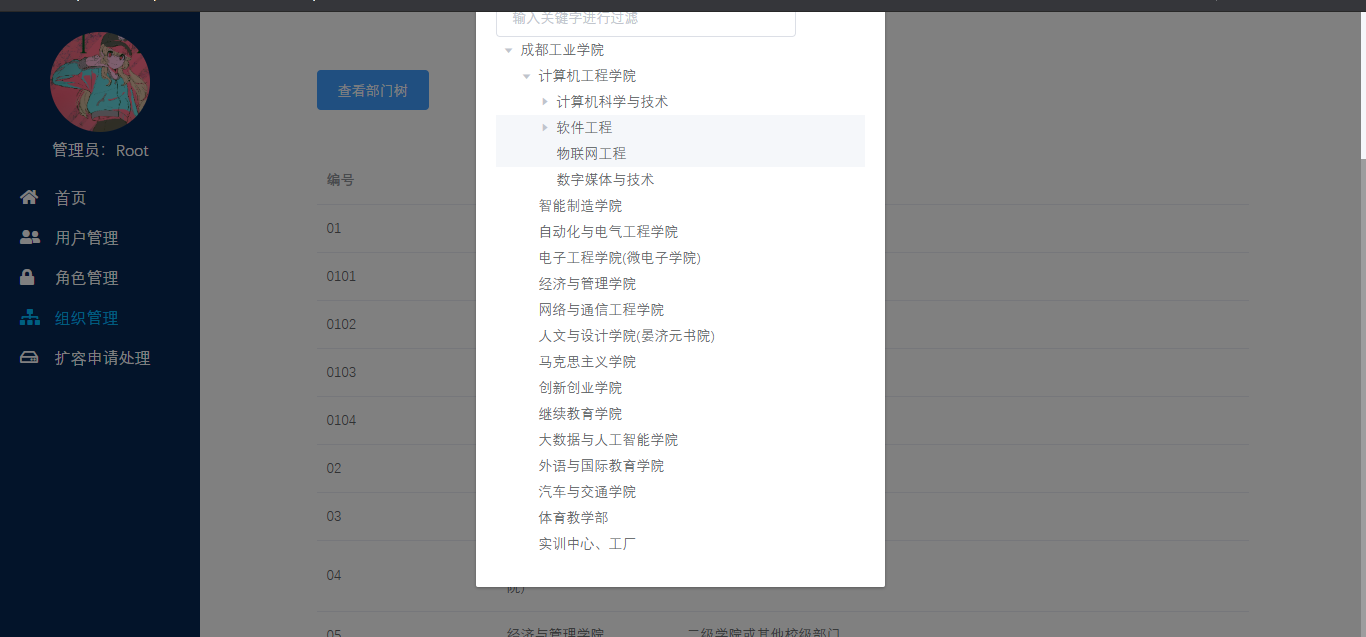


图5-14 后台管理员查看部门（树）

如果数据较多，可以通过顶部的输入框输入关键词进行搜索。

由于前端UI框架对数据格式的要求，所以后端做了转换，将存储的Department对象转为Node对象，具体代码如下图所示：



图5-15 树形结构生成方法代码

通过借助队列工具来辅助实现广度遍历操作。

第6章 系统测试

软件测试是软件质量保障的方法之一，是广泛意义上的“验证与测试”[IEEE1012-2004]的一部分。验证与确认常被简称为“V&V”。验证的目的是检查开发者是否正确的使用技术建立系统，确保系统能够在预期的环境中按照技术要求正确地运行。确认的目的是检查开发者是否建立了正确的系统，确保最终产品符合规格。

软件开发的验证与确认主要有两种手段，静态分析与动态测试。静态分析是在软件能够运行之前，依据开发文档、模型或者其他各种可用制品（例如原型）完成验证与确认任务的方法。动态测试就是软件测试。

## 6.1 系统界面测试

系统界面测试用例

|  |  |  |
| --- | --- | --- |
| 测试项 | 功能和描述 | 测试结果 |
| 用户登录页面 | 用户能够正常输入登录信息，并对用户登录信息进行初步验证，以及提示用户登录结果 | 正常 |
| 文件列表 | 显示文件的各级目录、点击文件夹会进行页面跳转 | 正常 |
| 分类显示自定义类型文件 | 根据常用文件的分类：文档、图片、音频、视频进行分类型浏览 | 正常 |
| 分享管理 | 创建分享、并通知用户创建的结果，分页显示分享及其详情 | 正常 |
| 文件目录树的显示 | 用户复制、移动，选择目标文件夹，会弹出对话框，将后台数据库存储的目录数据进行树型结构展示 | 正常 |
| 后台管理的菜单显示 | 根据管理员的角色动态渲染菜单 | 正常 |
| 用户管理 | 分页显示用户信息，能够根据分类显示不同类型的用户 | 正常 |
| 部门管理 | 分页显示部门的信息，能够根据部门分类显示不同类的部门，并显示部门树型结构 | 正常 |
| 角色管理 | 分页显示、添加、修改、删除角色数据 | 正常 |

## 6.2 系统功能性测试

### 6.2.1 登录功能测试用例

登录操作测试用例

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试用例描述：用户登录测试 | | | | |
| 编号 | 操作 | 预计结果 | 实际结果 | 测试状态 |
| 1 | 不输入登录信息就点击登录按钮 | 系统弹出警告信息：“请先输入登录信息” | 系统弹出警告信息：“请先输入登录信息” | 通过 |
| 2 | 输入不存在的用户ID，点击登录 | 系统弹出警告信息，“登录信息有误” | 系统弹出警告信息，“登录信息有误” | 通过 |
| 3 | 输入存在的用户ID，但是密码输入错误密码，点击登录 | 系统弹出警告信息，“登录信息有误” | 系统弹出警告信息，“登录信息有误” | 通过 |
| 4 | 输入正确的用户ID和密码，但是该用户已受限 | 系统弹出警告信息“该用户已被限制使用系统” | 系统弹出警告信息“该用户已被限制使用系统” | 通过 |
| 5 | 输入正确的用户ID和密码而且用户没有被限制 | 系统弹出“登录成功的”消息，并进行路由跳转 | 系统弹出“登录成功的”消息，并进行路由跳转 | 通过 |

### 6.2.2 用户上传文件测试

上传文件测试用例

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试用例描述：用户上传文件 | | | | |
| 编号 | 操作 | 预计结果 | 实际结果 | 测试状态 |
| 1 | 选择空文件上传 | 系统拒绝上传，并提示用户不能选择上传空文件 | 系统拒绝上传，并提示用户不能选择上传空文件 | 通过 |
| 2 | 选择已完整存在的文件上传 | 实现秒传 | 实现秒传 | 通过 |
| 3 | 选择部分存在的文件上传 | 只上传未存在的分片 | 只上传未存在的分片 | 通过 |
| 4 | 选择不存在的文件上传 | 上传整个文件 | 上传整个文件 | 通过 |
| 5 | 暂停和恢复上传 | 系统暂停发送分片，或从上次暂停的进度处开始上传 | 系统暂停发送分片，或从上次暂停的进度处开始上传 | 通过 |

### 6.2.3分享功能测试

分享功能测试用例

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试用例描述： 分享 | | | | |
| 编号 | 操作 | 预计结果 | 实际结果 | 测试状态 |
| 1 | 直接访问需要口令的分享 | 系统跳转到输入口令的页面 | 系统跳转到输入口令的页面 | 通过 |
| 2 | 输入错误的分享口令 | 系统弹出口令错误信息 | 系统弹出口令错误信息 | 通过 |
| 3 | 输入正确的分享口令 | 系统跳转到分享空间 | 系统跳转到分享空间 | 通过 |
| 4 | 系统访问已过期的分享 | 系统跳转到404页面 | 系统跳转到404页面 | 通过 |

### 6.2.4共享功能测试

共享功能测试用例

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试用例描述： 共享 | | | | |
| 编号 | 操作 | 预计结果 | 实际结果 | 测试状态 |
| 1 | 访问当前用户未加入的共享组 | 系统页面跳转到申请加入的页面 | 系统页面跳转到申请加入的页面 | 通过 |
| 2 | 组管理员通过用户申请 | 用户成为该分享的成员 | 用户成为该分享的成员 | 通过 |
| 3 | 组创建者将用户设置为管理员 | 被设置用户成为管理员 | 被设置用户成为管理员 | 通过 |

### 6.2.5 用户管理功能测试

用户管理功能测试用例

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试用例描述：管理员用户管理 | | | | |
| 编号 | 操作 | 预计结果 | 实际结果 | 测试状态 |
| 1 | 管理员上传符合要求的用户数据excel文件 | 系统解析数据，并将数据保存到数据库 | 系统解析数据，并将数据保存到数据库 | 通过 |
| 2 | 管理员上传不符合要求的excel文件 | 系统解析报错，并提示用户，新增用户失败 | 系统解析报错，并提示用户，新增用户失败 | 通过 |
| 3 | 管理员根据部门删除用户 | 对应部门下的用户数据被删除 | 对应部门下的用户数据被删除 | 通过 |

## 6.3 测试总结

通过对系统的各项测试，结果符合预期。本次测试在3个层次上保证了该系统的质量：

* 单元测试，验证了该系统的构造单元的内部实现的质量。
* 集成测试，验证了系统各个构造单元和部件按照体系结构要求进行集成的质量。
* 系统测试，验证了系统产品符合系统的需求。

第7章 结束语

至此，本次毕业设计就告一段落了，回顾整个过程，感觉自己当初有幸能够选择此研究题目。虽然自己以前也做过简单的项目，但是都是一些基本的CRUD操作，对于文件操作，一直不是很熟悉。在此系统实现初期，也是查阅了很多资料，了解文件上传的过程，以及如何实现分片上传，断点上传等。

尽管本人认真分析和设计该系统，但是由于本人技术能力有限，所以该系统距离实际使用还是有一定的差距。最大的问题就是未能实现分布式，实际情况中的系统应该实现集群存储，用户访问系统时，主服务器应该从从属服务器中选择一个来处理用户的请求。另外，系统还应具有单独的备份服务器，降低数据丢失的风险。

生命不止，学习不止。针对上述问题，我在以后的工作生活中会继续了解学习相关领域的技术，并对本系统做进一步的优化和升级。

# **参考文献**

[1] 霍斯特曼(Cay S. Horstmann) Java核心技术(卷Ⅰ 基础知识)[M].11.北京：机械工业出版社，2019.

[2] 霍斯特曼(Cay S. Horstmann) Java核心技术(卷Ⅱ 高级特性)[M].11.北京：机械工业出版社，2019：43-117.

[3] 骆斌,丁二玉,刘钦.软件开发的技术基础(卷二)[M].北京：机械工业出版社，2012.

[4] 田胜利,杜根远,胡子义. 基于用户级权限管理的网络硬盘的设计与实现[J]. 计算机与网络. 2007(06).

[5] 胡坤华,粟栗. 网络密文硬盘中的高效认证方案[J]. 微计算机信息. 2006(36).

[6] 王洋.浅谈网络硬盘的应用及其优势以及缺点[J]. 电脑学习. 2009(04).

[7] 刘杰. 浅谈数字化校园中网络硬盘的建设[J]. 福建电脑. 2010(09).

[8] 骆斌,丁二玉. 需求工程-软件建模与分析[M].北京：高等教育出版社，2009.

[9] 中国软件工程学科教程课题组.中国软件工程学科教程[M].北京：清华大学出版社，2005.

[10] 周志明.深入理解Java虚拟机：JVM高级特性与最佳实践[M].3.北京：机械工业出版社，2019（11）.

[11] Standish Group. Chaos.1995.

[12] Standish Group. Extreme Chaos.2001.

[13] Yusuf Dogan, Christi K. Madsen Optimization of ultrafast laser parameters for 3D micromachining of fused silica[J] Optics and Laser Technology, 2020, 123.

[14] Jiajia Zhang, Juan Bi, Guibo Chen; Inverse estimation of the material parameters at metal surface irradiated by laser pulse[J] Optik - International Journal for Light and Electron Optics, 2017, 129

[15] Sang Liu, Gaoyang Mi, Fei Yan et al. Correlation of high power laser welding parameters with real weld geometry and microstructure[J] Optics and Laser Technology, 2017, 94.

[16] A. Stancalie, S. Ciobanu, D. Sporea Investigation of the effect of laser parameters on the target, plume and plasma behavior during and after laser-solid interaction[J] Applied Surface Science, 2017.

[17] S. Jelvani Effect of Ag nanoparticles on the amplifying parameters of a dye laser amplifier[J] Optik, 2019, 199.

[18] K. Venkatesan, R. Ramanujam, P. Kuppan Parametric modeling and optimization of laser scanning parameters during laser assisted machining of Inconel 718[J] Optics and Laser Technology, 2016, 78.

[19] Andrey M. Popov, Sergey M. Zaytsev, Irina V. Seliverstova et al. Matrix effects on laser-induced plasma parameters for soils and ores[J] Spectrochimica Acta Part B: Atomic Spectroscopy, 2018, 148.

[20] Xiaohui Jiang, Tong Ye, Yihong Zhu. Effect of process parameters on residual stress in selective laser melting of AlSi10Mg[J] Materials Science and Technology, 2020, 36(3).

# **致 谢**

首先，我需要诚挚地感谢我的毕业设计导师刘明纲老师，在我完成毕业设计期间，向我提供悉心的关怀和指导。感谢学校和学院各级领导为成工学子创造舒适的学习环境。

本次毕业设计到此也接近尾声，标志着四年本科学习生活即将结束。尽管四年中学习了很多课程和技术，但是大部分都停留在比较浅显的地步，始终与实际应用有较大差距，对于很多技术点存在知识盲区。不过，通过此次毕业设计，提升了我解决问题的能力，从收集资料到实际问题分析，再到最后的实现，都有一定的进步。 在以后的工作学习生活中，我会保持这段时间以来的习惯，不断学习，不断进步。