题目

姓名　学院 年级 专业

**摘　要：**内容

内容内容

**关键词**：关键字；关键字；关键字；关键字；关键字

**English Term**

**Zhengkejia Cisco message college**

**Abstract:** content

content

**Key words:** *Related Party Transaction，Related Party Transaction，Related Party Transaction，Related Party Transaction*

目　　录

**摘要I**

**AbstractII**

**1、绪论1**

1.1开发背景与现状2

1.2开发本系统意义..................................................................................................3

1.3目标4

**2、开发平台介绍5**

2.1开发语言5

2.2数据库5

2.3 Struts2框架5

2.4文件存储平台5

2.5文件检索平台5

2.6总体技术路线5

**3、系统功能模块分析5**

3.1系统总体设计5

3.2系统模块设计5

3.2.1登录模块6

键入章标题(第 3 级)6

键入章标题(第 3 级)6

键入章标题(第 3 级)6

键入章标题(第 3 级)6

**4、数据库设计与说明5**

4.1数据库设计5

4.2数据库说明5

**5、界面设计5**

5.1登录界面5

5.2用户操作界面5

键入章标题(第 2 级)5

键入章标题(第 2 级)5

键入章标题(第 2 级)5

**6、运行环境与测试5**

6.1软件运行环境5

6.2系统测试5

**参考文献5**

**附录5**

**致谢5**

*（插入分页符）*

1. **绪论（引论）**

**1.1开发背景与现状**

网盘，又称网络硬盘，是由互联网公司推出的在线存储服务，向用户提供文件的存储、共享、备份、访问等管理文件的功能。

随着移动互联网设备的持续繁荣，个人数据将继续呈现爆炸式的增长，而多种硬件设备之间的交互需求将使网络硬盘的用户增长持续呈迅速上升的趋势。据第三方调研公司的不完全统计，全球云存储市场的份额将达到16亿美元，而金山快盘在2012年的用户数量也突破到了1000万左右。

我们最初的愿望，就是希望有个地方可以永久地存放自己的文件不怕弄丢，然现在，这个产业已经以令人乍舌的速度在发展。

最新应用的云计算储存技术，为网盘行业发展带来了新的力量，传统的网盘将逐步被云存储取代，如邮箱的文件存储，本地存储等技术。云存储是搭建在高速分布式存储的网络上的高级数据中心，它将网络中大量不同类型的存储设备通过应用软件进行加工，集合起来协同工作，建立一个较为安全的数据存储空间软件和权限访问的系统，适用于各大中小型企业与个人用户的数据资料存储、备份、回档等一系列在日常生活中的必要需求。云存储最大特点就是在于将单一的存储产品转化成为数据存储与用户服务，在这个云存储的技术支持下，网盘发展行业可以像金融行业银行一样，在单一的存储服务基础上衍生出更多增值的服务，如会员服务，权限加速，存储空间加大等，只有这种与往常不一样的改变，才能拯救网盘的现状。

网盘的原理如同分蛋糕，就是将自己公司上的服务器作为一份蛋糕，将空间容量动态分配给各个注册用户，学术上来说，就是发展网盘的网络公司将其服务器的硬盘或硬盘阵列中的一部分容量分给注册用户使用，因此网盘一般来说投资都比较大，毕竟服务器租赁的费用比较高，所以免费网盘的普通注册用户的容量比较小，一般为5G到10G不等；另外为了防止注册用户滥用网盘空间资源，往往还会附加单个文件的最大限制要求。简单来分析，免费网盘的普通用户一般只用于存储较小的文件。而收费网盘或者VIP注册用户则具有速度快、安全性能好、容量高、允许大文件存储等优点，适合与对网盘资源要求比较高的用户使用。

如今在国内的发展得较为完善的当属百度云网盘。百度云网盘是百度科技有限公司2012年正式推出的一项免费云存储服务，首次注册即可获得5GB的空间，首次上传一个文件可以获得1GB，使用手机登录百度云移动端的注册用户，即可领取1024G的永久免费云盘存储空间。 到目前为止，具备Web版、Windows客户端、Android手机客户端，Mac客户端，IOS客户端和WP客户端。在开发如此完善多元的情况下，用户可以随时随地将自己的文件上传到网盘上，普通用户单个文件最大可达3G，并可以跨越不同终端随时随地查看和分享。不仅如此，百度网盘还提供离线下载、文件智能分类浏览、视频在线播放、文件在线解压缩、免费扩容等功能。

国内网盘事业发展得非常迅速，不仅仅有百度云网盘，还有咕咕网盘、360网盘、115网盘、联想企业网盘、金山快盘、华为网盘、搜狐企业网盘、彩云网盘、天翼云等各个企业的网盘软件，包括web端、移动端、不同服务器端的应用。当然，对于用户而言，方便的网盘软件就是一个好的软件应用。

当然在国外，在较早的时候就开始发展网盘技术，如OneDrive、MediaFire、DivShare、WikiUpload、Scribd、jooce、FanBox、onShare、Mozy、5upload、Nakido、Xdrive、Mofile、G宝盘、Box、vdisk、蜂盘、FileFront等；在2012年，Google推出了Google Drive，微软的Skydrive也开始进一步整合业务。可以负责任的说，云盘系统在用户心中占的比重也越来越大，与人们日常生活，工作的联系也越来越缜密。

随着网盘市场竞争的日益激烈和存储技术的不断发展，传统的网盘技术已经显得力不从心，传输速度慢及恢复能力低、安全性差、营运成本高等瓶颈一直困扰着网盘企业。同时，因为网盘技术的发展，可能因为竞争客户群体太激烈，很多企业也开始减少在网盘方面的发展，如较为出名的360云盘，是奇虎360科技的云存储服务产品。在2016年10月20日决定停止开发，2016年11月1日起停止云盘上传服务，2017年2月1日起关闭所有云盘账号并清空数据，2016年10月20日至2017年2月1日进行会员退款。

但是，我个人认为，随着硬件和宽带成本的降低以及无线移动网络的普及，网盘的发展将会有很大发展前景，因为服务群体非常的大，届时网盘的盈利模式将不再局限与云存储，而是会成为一个更好的产品，为人们提高更方便的服务。**1.2开发本系统意义**

开发本系统的灵感源至于我们的日常生活，在使用百度云网盘的过程中，发现云盘在各种网络状态下的使用限制，为此，我决定深入探究云盘系统的搭建与构造，并尝试完成一个具有完整网盘功能的系统。

该系统实现了文件的上传、下载、删除、分享，通过利用hadoop平台来实现数据的备份，防止了数据丢失，增强了数据的安全性。用户在使用过程中可以将自己的文件分享给其他用户，实现了数据共享。本系统还引入了Lucene全文检索技术，可以对用户的文件进行内容检索。

**1.3目标**

通过注册页面实现用户的注册账号功能，注册的同时用户可以选择邮箱验证注册和默认注册两种方式，其中默认注册方式分配给该账号的云盘空间有10G，若选择邮箱验证注册方式系统会发送验证码到用户的邮箱，以此方式成功注册后用户将获得100G云盘空间。

用户注册成功后将跳转到登陆界面，登陆成功便跳转到系统主界面，主界面将显示用户的账号、邮箱、电话等信息，用户头像为系统默认的头像，用户可自行上传更改。主界面还有新建文件夹功能、上传新文件功能、文件名搜索和文件内容搜索功能。若用户想扩大云盘容量可通过系统的签到功能进行升级。

1. **开发平台介绍**

2.1开发语言

Java是一门由Sun公司开发出来的面向对象编程语言，其既拥有C++语言的多种优点，同时也舍弃掉了C++里面很多难以理解的概念，例如多继承、指针等。得益于此Java语言有简单易用和功能强大两个特点。Java语言作为一门静态面向对象编程语言，能够快速进行面向对象编程。我们开发的系统运行于window和linux上，因此可利用Java的跨平台和可移植性进行开发，并且本人对Java语言相对熟悉。

JSP全称是Java Server Pages，即java服务器页面，由Sun公司主导多公司贡献建立成的一种动态网页技术标准。JSP可以在传统的HTML文件中插入Java script代码和jsp标签，从而形成功能更强大的JSP界面。 用JSP开发出来的Web应用能运行于多种系统上，这也符合我们要开发的系统的要求。

2.2数据库

MySQL是一种关系型数据库管理系统，其由MySQL AB 公司开发后来归属于甲骨文公司。MySQL关系数据库没有将数据保存在一个大数据库里面，而是将数据保存到了多张不同的表里面，如此一来增加了数据库的安全性与灵活性。由于MySQL拥有体积小、搭建拥有平台的成本较低，并且速度快，又是一个开源项目，因此其成了大部分中小型网站开发的不二之选。这些特点也适合我们的云盘系统。

2.3 Struts2框架

Struts2是基于MVC三层架构模式设计出来的一种轻量级的Web应用框架， Struts2的主要作用是建立模型与视图之间的数据交互。Struts 2的核心是WebWork，其通过在strut.xml文件中配置拦截器来控制处理用户的请求，这样配置之后业务的处理逻辑和界面的展示层完全分开了，更有利于系统的开发与维护，大大提高了灵活性。

2.4文件存储平台

由于开发的云盘系统是用来存储用户数据的，因此系统存储文件的安全性就显得极为重要，要保证用户数据不能因为系统原因或者机器宕机而丢失。所以数据的存储平台我们选择了hadoop平台。Hadoop是一个由Apache开发的开源的分布式系统基础架构。其实现了分布式文件系统，英文简称为HDFS。Hadoop框架有两个核心设计，分别是HDFS和MapReduce。HDFS主要负责为海量的数据提供成熟的存储机制，而MapReduce即为海量的数据集提供了高可靠的分布式计算功能。因为HDFS可以通过配置来设定每个文件的备份数，因此其有高容错性的特点，可以部署在廉价的机器上。得益于其分布式的特点，Hadoop可以通过流的形式来访问HDFS中的数据，其有吞吐量高的特点，适合于大数据集访问的应用程序。因此HDFS非常适合用来开发云盘系统，事实上百度云盘也是基于Hadoop开发的。

2.5文件检索平台

由于我们的云盘系统还提供了以内容搜索文件的功能，所以我们引入了Lucene全文检索机制。Lucene也是apache软件基金会的一个开源项目，其是一个全文检索架构而不是全文检索引擎。Lucene拥有完善的查询和搜索引擎，并提供部分文本分析引擎，但是其没有提供中文文本分析引擎，所以我们还要引入庖丁中文文本分词器来进行文本切分。

2.6总体技术路线

首先是开发系统的搭建，我们的数据存储平台搭建于本地虚拟机的Linux系统下。安装好linux系统后将Linux系统的IP及防火墙等配置好。在Linux系统上我们搭建了hadoop平台及MySQL数据库。

本云盘系统采用的技术路线如下：

（1）采用Struts2框架进行代码逻辑与界面的交互。

（2）虚拟机与宿主机的通信采用hadoop提供的RPC机制进行数据交互。

（3）开发系统环境：windows系统、linux系统

（4）数据库：MySQL。

（5）开发工具：IntelliJ IDEA 2016.3，Tomcat，JDK。

1. **系统功能模块分析**
2. **部分功能模块分析**

**3.1系统总体设计**

**3.2部分功能模块设计**

3.2.1登录模块

用户登录系统拥有独属于自己的账号，不同等级的用户所具有的存储空间也不相同。不同的角色拥有不同的权限，所能看到的页面，操作的数据也不一样。不同的权限用户只可以对自己权限下的功能进行操作，不能越级操作。[5]

管理员账号，可以查看网盘上所有用户的存储文件信息并允许管理员进行增删查改的操作。

3.2.2注册模块

用户通过填写账号、密码、手机号码、邮箱来注册用户账号并使用云盘系统。在此处，加入了邮箱验证的功能，增强了系统的保密性与稳定性。

3.2.3搜索模块

用户可以通过选择按钮选择通过文件名或文件内容进行搜索，大大增强了系统的可用性与方便性。使用Hadoop框架，运用数据流的技术，使文件搜索速度大大增强。

3.2.4进度条模块

通过Bootstrap框架，编写了显示网盘内存空间的进度条。通过操作进行动态变化，客观可见的显示存储空间的变化，同时具有变色提醒功能，当网盘空间达到80%时，进度条会从蓝色变为具有警示意义的红色，大大提高了用户的观感体现。

3.2.5文件处理模块

通过对文件后缀的处理，给予不同文件类型的图标icon，同时可以通过点击标签对文件进行选取与分类。每个文件都具有下载、删除、分享功能，体现了网盘系统的完整性、

3.2.6任务功能模块

设有任务功能，用户可以通过完成每日任务提升网盘存储空间，从而大大增强了系统的可用性与寿命，这种人性化的设置让用户得以长久的使用系统。

1. **数据库设计与说明**

数据库E-R图：

1

1

用户

文件

拥有

n

属于

1

回收站

数据字典：

User 用户信息表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 默认值 | 允许非空 | 自动递增 | 备注 |
| 用户名 | varchar(50) |  | 否 | 否 | 用户昵称 |
| 密码 | varchar(50) | NULL | 是 | 否 | 用户密码 |
| 电话号码 | varchar(50) | NULL | 是 | 否 | 用户电话号码 |
| 头像 | mediumblob | NULL | 是 | 否 | 用户头像 |
| 邮箱 | varchar(50) | NULL | 是 | 否 | 用户邮箱 |
| 签到日期 | varchar(50) | 0 | 是 | 否 | 记录用户上次签到日期 |
| 签到次数 | int(10) | 0 | 是 | 否 | 用户连续签到次数 |
| 已用空间 | double(4,2) | NULL | 是 | 否 | 用户已用云盘空间 |
| 总空间 | double(10,0) | NULL | 是 | 否 | 用户可用云盘总空间 |

file 文件信息表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 默认值 | 允许非空 | 自动递增 | 备注 |
| id | int(50) |  | 否 | 是 | 文件唯一标记号码 |
| 文件名 | varchar(50) |  | 否 | 否 | 文件名称 |
| 数据库路径 | varchar(50) |  | 否 | 否 | 文件在数据库中的路径 |
| 拥有者 | varchar(50) |  | 否 | 否 | 文件的拥有者 |
| 标签 | varchar(50) | NULL | 是 | 否 | 文件标签 |
| 大小 | varchar(50) | NULL | 是 | 否 | 文件大小 |
| 类型 | varchar(50) | NULL | 是 | 否 | 文件类型 |
| MD5值 | varchar(50) | NULL | 是 | 否 | 文件的MD5值 |
| 云盘路径 | varchar(50) | NULL | 是 | 否 | 文件在云盘的路径 |

recycle 删除文件缓存云盘路径表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 默认值 | 允许非空 | 自动递增 | 备注 |
| 云盘路径 | varchar(50) | NULL | 是 | 否 | 已删除文件在云盘的路径 |

**6系统逻辑控制**

**6.1总体设计概述**

本系统的逻辑设计思想为将对文件的操作转换为对数据库的操作，除了文件的上传和删除会真正对云盘上的文件进行操作外，其他的操作都是对数据库进行操作。

**6.2注册逻辑**

从前端获取到用户填写的信息之后，若用户填写的验证码为0，即该注册方式为不验证邮箱注册，直接将用户填写的用户信息写入数据库，即注册成功，若用户没有填写验证码，即默认通过邮箱验证码注册，此时将用户的填写数据放进session中，方便以后展示，并且通过验证码生成器生成一个随机的6位数字验证码，并且将该验证码发送到用户填写的邮箱中，用户获取到验证码后填写进注册页面，若用户填写的验证码与之前生成的验证码相同，即注册成功，将用户的信息写进数据库，即注册成功，否则不成功。其中不通过邮箱验证注册的用户云盘空间为10G，通过邮箱验证的用户云盘空间为100G。

if (code.equals("0")) {

selectsql="insert into user(username,password,phonenum,email,totalsize,used) values(\""

+ username + "\",\"" + password + "\",\"" + phonenum + "\",\"" + email + "\",10,0)";

} else {

selectsql = "insert into user(username,password,phonenum,email,totalsize,used) values(\""

+ username + "\",\"" + password + "\",\"" + phonenum + "\",\"" + email + "\",100,0)";

if (!identify.equals(code)) {

session.put("code", jcode);

MailUtil.sendEmail(email, icode, username);

return "again";

}

}

**6.3登陆逻辑**

用户注册成功后将跳转到登陆页面，用户输入注册成功的用户名和密码，后台接收到数据之后将会和数据库比对，如果数据库存在这条数据即为登陆成功，系统将会跳转到主页面，若输入错误系统将继续跳转到登陆页面。

String selectUsed = "select used,totalsize from user where username=\"" + username + "\"";

ResultSet resultSet1 = dataBaseOperation.querySql(selectUsed);

Map<String, Object> session = ActionContext.getContext().getSession();

if (resultSet1.next()) {

session.put("used", resultSet1.getDouble("used"));

session.put("totalsize", resultSet1.getDouble("totalsize"));

}

String selectsql = "select \* from user where username=\"" + username + "\" and password=\"" + password + "\"";

resultSet = dataBaseOperation.querySql(selectsql);

if (resultSet.next()) {

setEmail(resultSet.getString("email"));

setPhonenum(resultSet.getString("phonenum"));

session.put("username", username);

session.put("email", resultSet.getString("email"));

session.put("phonenum", resultSet.getString("phonenum"));

return "ok";

}

return "failed";

**6.4头像修改与显示逻辑**

在登陆时系统还将进行逻辑判断用户表的photo字段是否为空，如果不为空即将个是否有自定义头像标志放进session中，前端进行头像显示时进行判断该标志的值动态显示用户头像。

<s:if test='%{#session.hasphoto=="1"}'>

<img src="showtupian.action?username=<s:property value="#session.username"/>" alt="个人图片"

style="width:160px;height: 200px;">

</s:if>

<s:else>

<img src="/images/person.svg" alt="个人图片" style="width:160px;height: 200px;">

</s:else>

用户可在主界面修改头像，用户选择了图片文件后后台获取到该文件的输入流，并转换成二进制流写进数据库里面。

FileInputStream fileInputStream = new FileInputStream(picture);

ImageOperation.readImage2DB(username, fileInputStream);

session.put("hasphoto","1");

return "ok";

public static void readImage2DB( String username, FileInputStream in ) throws Exception {

DataBaseOperation dataBaseOperation = new DataBaseOperation();

PreparedStatement ps = null;

try {

String sql = "UPDATE user set photo=? where username=\"" + username + "\"";

ps = dataBaseOperation.conn.prepareStatement(sql);

ps.setBinaryStream(1, in, in.available());

int count = ps.executeUpdate();

if (count > 0) {

System.out.println("插入成功！");

} else {

System.out.println("插入失败！");

}

} catch (Exception e) {

e.printStackTrace();

}

}

若头像标志位有头像数据，则jsp页面将会调用后台action进行页面刷新将数据库中的图片数据显示出来。

try {

HttpServletResponse response = ServletActionContext.getResponse();

DataBaseOperation dataBaseOperation = new DataBaseOperation();

String sql = " SELECT \* FROM user WHERE username = \"" + username + "\"";

ResultSet resultSet = dataBaseOperation.querySql(sql);

Blob blob = null;

if (resultSet.next()) {

blob = resultSet.getBlob("photo");

}

if (blob.equals(null)) {

session.put("hasphoto","0");

}else {

session.put("hasphoto","1");

}

InputStream inputStream = blob.getBinaryStream();// IO流

int length = (int) blob.length();

byte[] b = new byte[length];

inputStream.read(b, 0, length);

PrintWriter out = response.getWriter();

InputStream is = new ByteArrayInputStream(b);

int a = is.read();

while(a != -1) {

out.print((char) a);

a = is.read();

}

out.flush();

out.close();

} catch (Exception e) {

System.out.println("error");

}

return NONE;

**6.5文件上传逻辑**

前端选择上传文件后点击提交，后台将获取到该文件的输入流和文件名。由于系统设计为将云盘主界面显示的文件目录与云盘真正的数据存储目录分离，因此需要在文件上传时对文件进行转换。

1. 文件名转换

同一个用户所有的文件在云盘的真正存储目录为/Disk/用户名/，因此有可能存在用户上传了两个同一个文件名文件内容却不同的文件，此时云盘真正的目录下将值存在一个文件，所以文件上传时我们的文件在云盘上真是的文件名为在该文件名后面加上“\_(序号)”,若如此转化后云盘仍存在此文件名，即序号加一。如：原文件名：a.txt，该文件在云盘上的路径为/Disk/用户名/a\_0.txt，若云盘已存在该名称的文件，即转化为/Disk/用户名/a\_1.txt。

//相同文件名相同路径下不允许上传

setDbPath((String) session.get("dir"));

String selectSql = "select \* from file where filename=\"" + filenameFileName + "\" and dbpath=\"" + dbPath + "\""

+ " and owner=\"" + username + "\"";

ResultSet resultSet = dataBaseOperation.querySql(selectSql);

if (resultSet.next()) {

//弹出提示框

request.setAttribute("errorMessage", "目录：" + dbPath + "下已经存在" + filenameFileName + "文件！");

return "ok";

}

String hdfsName = filename1 + "\_";

//重命名文件在网盘的名字

String fileSql = "select hdfsPath from file where owner=\"" + username + "\" order by hdfsPath desc";

ResultSet resultSet0 = dataBaseOperation.querySql(fileSql);

String hdfsName1 = "";

//如果网盘已存在文件

while(resultSet0.next()) {

hdfsName1 = resultSet0.getString("hdfsPath").substring(resultSet0.getString("hdfsPath").lastIndexOf("/") + 1,

resultSet0.getString("hdfsPath").lastIndexOf("\_"));

if (filename1.equals(hdfsName1)) {

int index = Integer.parseInt(resultSet0.getString("hdfsPath").substring(resultSet0.getString("hdfsPath").lastIndexOf("\_") + 1,

resultSet0.getString("hdfsPath").lastIndexOf(".")), 10) + 1;

hdfsName += index + "." + type;

System.out.println(hdfsName);

break;

}

}

1. 文件类型转换

为了简化前端页面显示文件的图标，因此将文件的类型同意分成几类，如jpg、png等图片文件通属于image类。

String image = ".jpg,.png,.gif";//图片类型

String zip = ".zip,.rar,.gzip";//压缩包类型

String word = ".doc,.docx";//word文档类型

String excel = ".xls";//excel文件类型

String TXT = ".txt";//TXT文件类型

String EXE = ".exe";//EXE文件类型

if (image.contains(type)) {

type = "image";

}

if (zip.contains(type)) {

type = "zip";

}

if (word.contains(type)) {

type = "word";

}

if (excel.contains(type)) {

type = "excel";

}

if (TXT.contains(type)) {

type = "TXT";

}

if (EXE.contains(type)) {

type = "EXE";

}

1. 文件哈希值比较

用户可能上传相同内容的文件到想不同的位置，在文件上传时会计算文件的MD5值，并与数据库中的记录进行比较，若已存在相同MD5值的文件，即实际上不上传该新文件，只需将该文件的其他信息写进数据库并将hdfspath云盘路径指向云盘已存在的文件，并将是否上传文件的标志置为false。

//如果网盘存在内容相同的文件

String fileExistHdfs = "select md5,hdfsPath from file where owner=\"" + username + "\"";

ResultSet resultSet2 = dataBaseOperation.querySql(fileExistHdfs);

String hdfsPath = "/Disk/" + username + "/" + hdfsName;

boolean isUploadFile = true;

while(resultSet2.next()) {

if (resultSet2.getString("md5").equals(fileMd5)) {

hdfsPath = resultSet2.getString("hdfsPath");

isUploadFile = false;

break;

}

}

1. 用户云盘空间容量判断

若该上传文件的大小已超出用户的剩余云盘空间，则不上传文件并弹出提示。若没有超出，则上传文件，并更新用户剩余空间。

if (resultSet1.next()) {

double used = resultSet1.getDouble("used");

used += (double) filename.length() / (1024 \* 1024);

if (resultSet1.getDouble("totalsize") < used) {

//弹出提示框

request.setAttribute("errorMessage", "您的空间已达到上限，请升级会员继续使用！");

return "ok";

}

DecimalFormat df = new DecimalFormat("0.00");

String updateUesd = "update user set used=" + used + " where username=\"" + username + "\"";

dataBaseOperation.updateSql(updateUesd);

}

1. 文件索引

为了提供全文检索功能，上传文件时要对上传文件内容进行索引，本系统的索引引用了Lucene全文检索功能。文件索引时只需对文件id和文件内容进行索引。搜索时返回包含搜索关键字的文件id。

public void index( String id, String contents ) throws Exception {

String indexDir = "d:\\luceneindex";

Analyzer analyzer = new PaodingAnalyzer();

Directory dir = FSDirectory.getDirectory(indexDir);

IndexWriter writer = new IndexWriter(dir, analyzer, false, IndexWriter.MaxFieldLength.UNLIMITED);

Document document = new Document();

document.add(new Field("id", id, Field.Store.YES, Field.Index.NOT\_ANALYZED));

document.add(new Field("contents", contents, Field.Store.YES, Field.Index.ANALYZED));

writer.addDocument(document);

writer.close();

dir.close();

}

**6.5文件搜索逻辑**

本系统提供的搜索有文件名搜索和文件内容搜索。

1. 文件名搜索

文件名搜索只需查询数据库的文件名字段，将符合要求的数据返回。

String sql = "select \* from file where owner=\"" + username + "\" and filename like \"%" + search + "%\"";

ResultSet resultSet = dataBaseOperation.querySql(sql);

while(resultSet.next()) {

File file = new File();

file.setId(resultSet.getString("id"));

file.setFilename(resultSet.getString("filename"));

file.setDbpath(resultSet.getString("dbpath"));

file.setOwner(resultSet.getString("owner"));

file.setTag(resultSet.getString("tag"));

file.setSize(resultSet.getString("size"));

file.setType(resultSet.getString("type"));

file.setMd5(resultSet.getString("md5"));

fileslist.add(file);

}

1. 文件内容搜索

文件内容搜索将对之前建立的索引文件进行搜索，并返回文件id，在数据库中获取该id对应的文件记录并进行显示。

public List<File> contentSearch( String contents, String username ) throws Exception {

List<File> fileslist = new ArrayList<File>();

String indexDir = "d:/luceneindex";

Directory dir = FSDirectory.getDirectory(indexDir);

IndexSearcher searcher = new IndexSearcher(dir);

ScoreDoc[] hits = null;

Term term = new Term("contents", contents);

TermQuery query = new TermQuery(term);

TopDocs topDocs = searcher.search(query, 100);

hits = topDocs.scoreDocs;

String key = "";

for (int i = 0; i < hits.length; i++) {

Document doc = searcher.doc(hits[i].doc);

key += doc.get("id") + ",";

}

if (!key.equals("")) {

key = key.substring(0, key.length() - 1);

String selectSql = "select \* from file where owner=\"" + username + "\" and id in(" + key + ")";

System.out.println(selectSql);

ResultSet resultSet = dataBaseOperation.querySql(selectSql);

while(resultSet.next()) {

File file = new File();

file.setFilename(resultSet.getString("filename"));

file.setDbpath(resultSet.getString("dbpath"));

file.setOwner(resultSet.getString("owner"));

file.setTag(resultSet.getString("tag"));

file.setSize(resultSet.getString("size"));

file.setType(resultSet.getString("type"));

file.setMd5(resultSet.getString("md5"));

fileslist.add(file);

}

}

searcher.close();

dir.close();

return fileslist;

}

**6.6文件目录显示逻辑**

由于系统设计将数据库和文件系统分开，因此无法通过获取目录下的文件进行展示。数据库记录的数据库路径用于展示给用户查看。

如图：



用户可点击的目录有两种方式，其中在区域1和区域2点击需要处理的逻辑不同。区域1点击的话点击路径只会小于或等于当前路径，这时只需显示点击路径之前的部分。区域2点击的话为进入该目录，所以点击路径肯定大于当前路径。

public String listDir() throws Exception {

fileslist = new ArrayList<File>();

dirList = new ArrayList<DirPath>();

String sessionDir = (String) session.get("dir");

if (dirType == null) {

if (dirName != null) {

session.put("dir", session.get("dir") + "/" + dirName);

}

} else {

if (sessionDir.length() < dirName.length()) {

session.put("dir", session.get("dir") + "/" + dirName);

} else {

if (dirName != null) {

session.put("dir", dirName);

}

}

}

dirName = (String) session.get("dir");

String[] splitDir = dirName.split("/");

for (int i = 1; i < splitDir.length; i++) {

DirPath dp = new DirPath();

dp.setDir(splitDir[i]);

String temp = "";

for (int j = 0; j <= i; j++) {

temp += splitDir[j] + "/";

}

dp.setDirPath(temp.substring(0, temp.length() - 1));

dirList.add(dp);

}

String sql = "select \* from file where dbpath=\"" + dirName + "\" and owner=\"" + username + "\"";

//"where dbpath=\"" + path + "\"";

ResultSet resultSet = dataBaseOperation.querySql(sql);

while(resultSet.next()) {

File file = new File();

file.setId(resultSet.getString("id"));

file.setFilename(resultSet.getString("filename"));

file.setDbpath(resultSet.getString("dbpath"));

file.setOwner(resultSet.getString("owner"));

file.setTag(resultSet.getString("tag"));

file.setSize(resultSet.getString("size"));

file.setType(resultSet.getString("type"));

file.setMd5(resultSet.getString("md5"));

fileslist.add(file);

}

return "ok";

}

**6.7签到逻辑**

本系统的签到升级空间要求连续签到七天才有效，因此在登陆的时候有判断当前日期相对于上次签到日期相比是否大于一天，若大于一天则该用户没有连续签到，之前的签到无效，将该用户的签到天数置为0。如果该用户签到的日期已经达到7天则升级该用户的空间并将用户的签到天数重新置为0。

if (!a.equals("0")) {

Date d2 = df.parse(a);

long diff = d1.getTime() - d2.getTime();

long days = diff / (1000 \* 60 \* 60 \* 24);

if (days > 1) {

String updateTime = "update user set signtime=0 where username=\"" + username + "\"";

DataBaseOperation dataBaseOperation = new DataBaseOperation();

dataBaseOperation.updateSql(updateTime);

signtime = 0;

}

if (days >= 1 && signtime == 7) {

String updateTime = "update user set signtime=0 where username=\"" + username + "\"";

DataBaseOperation dataBaseOperation = new DataBaseOperation();

dataBaseOperation.updateSql(updateTime);

signtime = 0;

}

}

用户点击签到之后将进行判断当前日期和上次签到日期是否相同，若相同则该用户今天已经签到过了，系统将进行提示。如果不同系统提示签到成功，并显示已签到天数。

public String sign() throws Exception {

HttpServletRequest request = ServletActionContext.getRequest();

Map<String, Object> session = ActionContext.getContext().getSession();

String username = (String) session.get("username");

DataBaseOperation dataBaseOperation = new DataBaseOperation();

Date now = new Date();

SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy/MM/dd");

String signday = dateFormat.format(now);

int signtime;

String signdaydb = "";

String updateDay = "update user set signday=\"" + signday + "\" where username=\"" + username + "\"";

String updateTime = "update user set signtime=";

String sql = " SELECT \* FROM user WHERE username = \"" + username + "\"";

double size = 0;

ResultSet rs = dataBaseOperation.querySql(sql);

if (rs.next()) {

try {

signdaydb = (String) rs.getString("signday");

signtime = rs.getInt("signtime");

size = rs.getDouble("totalsize");

DateFormat df = new SimpleDateFormat("yyyy/MM/dd");

Date d1 = df.parse(signday);

if (signdaydb.equals("0")) {

dataBaseOperation.updateSql(updateDay);

updateTime += "1 where username=\"" + username + "\"";

dataBaseOperation.updateSql(updateTime);

session.put("signtime", "1");

request.setAttribute("signmessge", "签到成功,您已连续签到1天,请继续坚持！");

return "ok";

}

Date d2 = df.parse(signdaydb);

long diff = d1.getTime() - d2.getTime();

long days = diff / (1000 \* 60 \* 60 \* 24);

if (signdaydb.equals(signday)) {

request.setAttribute("signmessge", "您今天已经签到过了，明天再来吧！");

} else {

dataBaseOperation.updateSql(updateDay);

int temp = signtime + 1;

String a = "";

a += temp;

updateTime += temp + " where username=\"" + username + "\"";

session.put("signtime", a);

dataBaseOperation.updateSql(updateTime);

request.setAttribute("signmessge", "签到成功,您已连续签到" + a + "天,请继续坚持！");

if (temp == 7) {

size += 10;

updateTime = "update user set signtime=7,totalsize=" + size + " where username=\"" + username + "\"";

dataBaseOperation.updateSql(updateTime);

String aaa = "";

aaa += size;

session.put("totalsize", aaa);

request.setAttribute("signmessge", "您已连续签到7天,恭喜您获得10G免费空间，请再接再厉！");

}

}

} catch (Exception e) {

}

}

return "ok";

}

**7运行环境及测试**

**7.1系统运行环境**

本系统运行于宿主机上，hadoop和MySQL数据库运行于虚拟机上。具体环境如下：

工程运行环境：项目运行于Tomcat，也可用Jetty运行。

客户机：建议客户端使用CPU四核1.6G以上，内存4G以上，独立显卡最好是512M显存以上，CPU集成核显亦可。

数据库服务器：Linux系统，物理内存4G以上。

文件服务器：Linux系统，物理内存4G以上，硬盘100G以上。

**7.2系统测试**

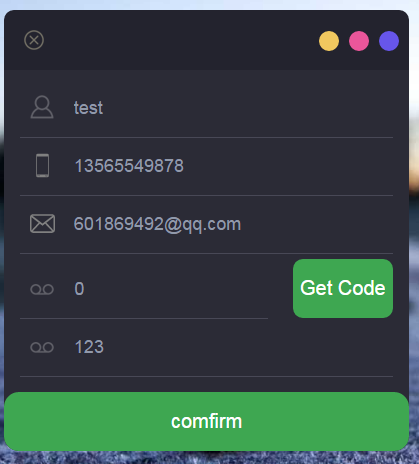
为确保云盘系统能健康正常的运行，测试在开发过程中是必不可少的一个组成部分。测试的工作分布在系统开发的各个阶段，如界面、功能的开发，都需要反复的调整以确保系统的正确性。

本次测试主要用到一种单元测试的测试方法，就是黑盒测试，黑盒测试顾名思义，就是类似在一个黑盒子中进行测试，无法查看内部数据，只通过输入输出来查看系统的正确性。通过测试来检查功能是否正确或遗漏或者存在数据库的访问问题等等。若发生问题，则去查看该部分的代码，这就是单元测试的方便之处，不需要再全局的寻找问题所在，只需查看某部分即可。

测试的部分界面如下：

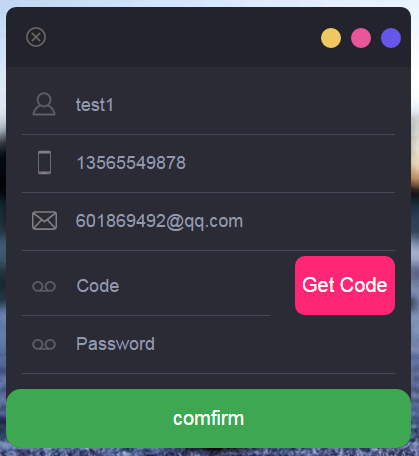
注册测试：

不认证邮箱注册：



认证邮箱注册：

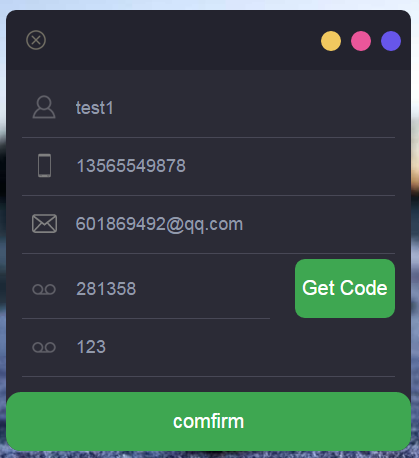
1. 获取验证码



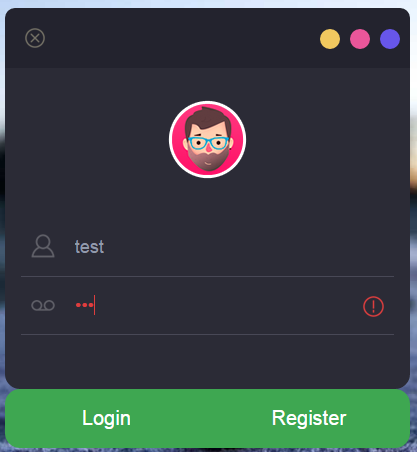
1. 邮箱获取到验证码



1. 输入验证码注册

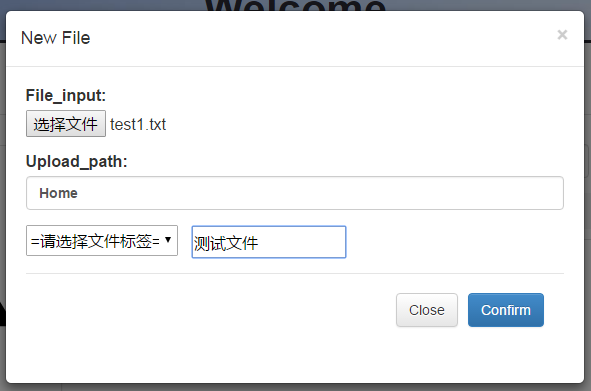


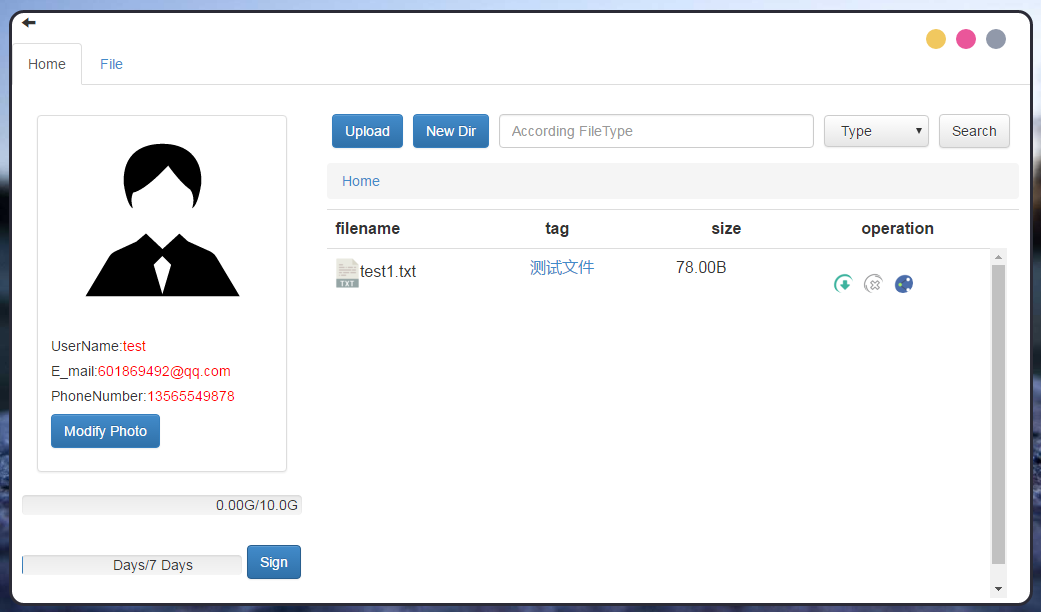
登录测试：



刚才注册的账号均能登录成功。

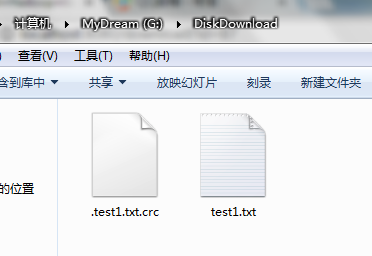
文件上传测试：





下载测试：

点击下载按钮，查看默认本地下载路径

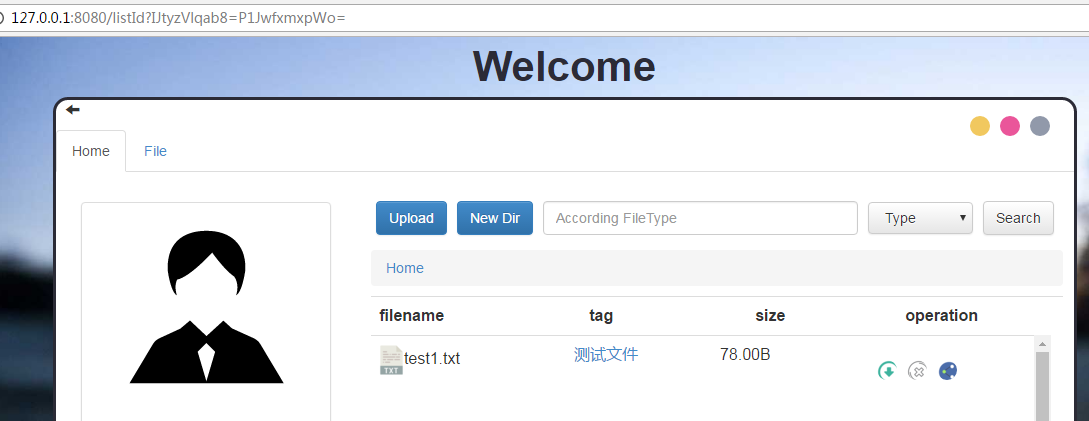


分享测试：

点击分享图标，显示分享地址。

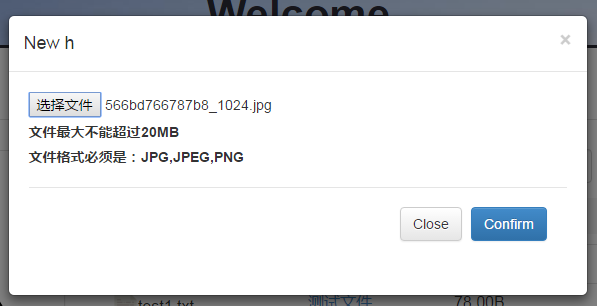


打开网址显示该文件

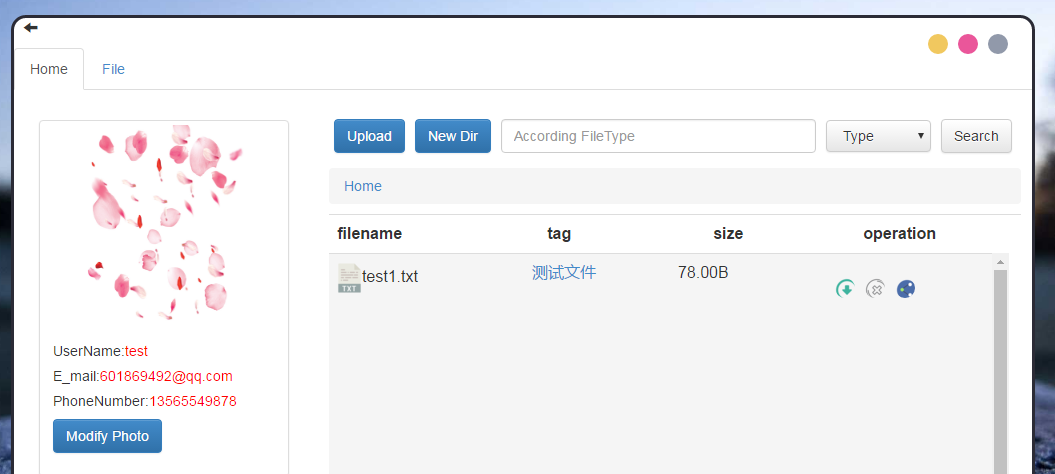


修改头像测试：

上传头像。

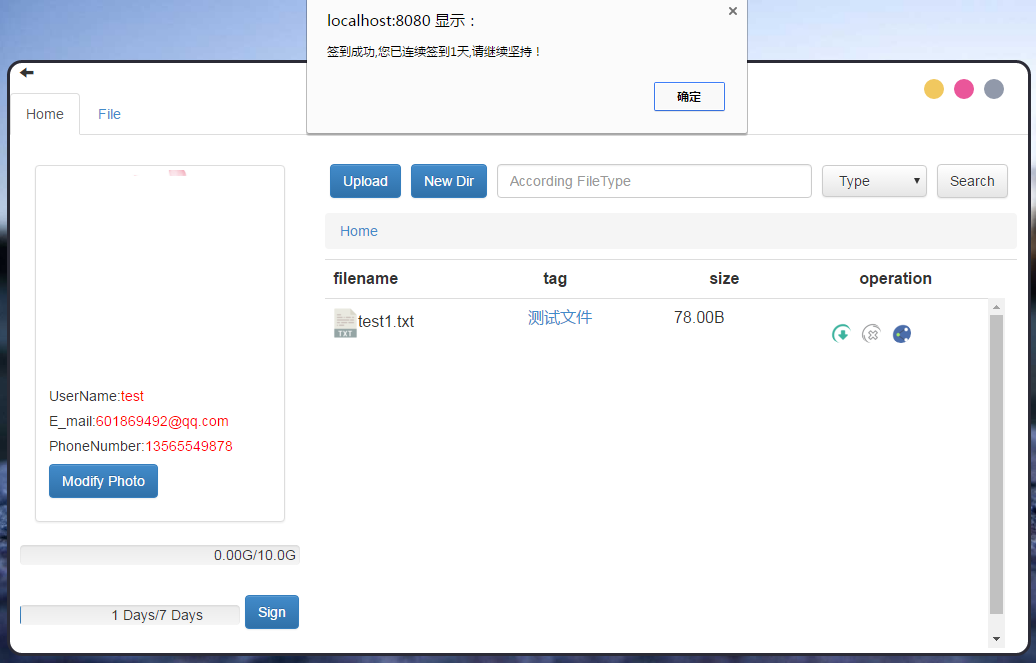


点击确定，主界面显示头像。

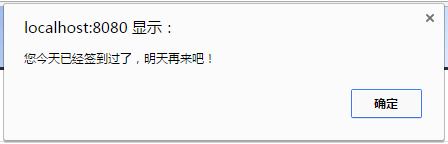


签到测试：

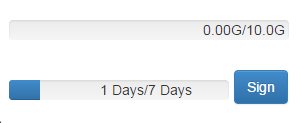
点击签到



再次点击签到，弹出提示



进度条显示



搜索测试：

文件名搜索



文件内容搜索



登录、注册、验证、上传、下载、分享、修改图片等功能都能成功运行，测试成功。

（正文结束）

*（空两行）*

**参考文献：***（小三号，宋体，加粗、缩进两字，参考文献数目应不少于10篇（部）, 参考文献中所有的标点符号全部在半角状态下输入）*

*（空一行，同时有外文文献的，先中文文献，后外文文献）*

[1]作者.书名[M].出版地:出版社，出版年份:起止页码.*（图书）*

[2]主办方.论文集[C].出版地:出版者,出版年份.*（论文集）*

[3]作者.题名[J].刊名，年，期（卷）:起止页码.*（期刊）*

[4]World Health Organization. Factors regulating the immune response: report of WHO Scientific Group[R]. Geneva: WHO, 1970.*（科技报告）*

[5] 萧钮.出版业信息化迈人快车道[EB/OL].(2001-12-19)[2002-04-15]. http://www.creader.com/news/20011219/200112190019.html *(网络文献)*

*（缩进两字，连续编号，宋体，五号，单倍行距。具体格式按《文后参考文献著录规则（GB/T 7714-2005）》节选）*

*（插入分页符）*

**附　　录**

*（“附录”两字中间在全角状态下中间空两格，居中、小三号，宋体，加粗）*

*（插入分页符）*

**致　　谢**

*（“致谢”两字中间在全角状态下中间空两格，，居中、小三号，宋体，加粗）*

*（空一行）*

致谢内容*（宋体，小四，首行缩进两字）*

**https://wenku.baidu.com/view/3d642236852458fb760b562d.html**