

# 浙江大学实验报告

专业： 生物医学工程  
姓名： 沈家乐  
学号： 3190100438  
日期： 2021.12.22  
地点： 教 7-502

课程名称： 高级程序设计 指导老师： 耿晨歌、陆涛涛 成绩：  
实验名称： 高级程序设计课大作业 实验类型： 研究型 同组学生姓名： /  
一、实验目的和要求  
二、实验内容和原理  
三、主要仪器设备  
四、操作方法和实验步骤  
五、实验数据记录和处理  
六、实验结果与分析  
七、讨论、心得

## 实验 高级程序设计课大作业

### 一、实验目的和要求

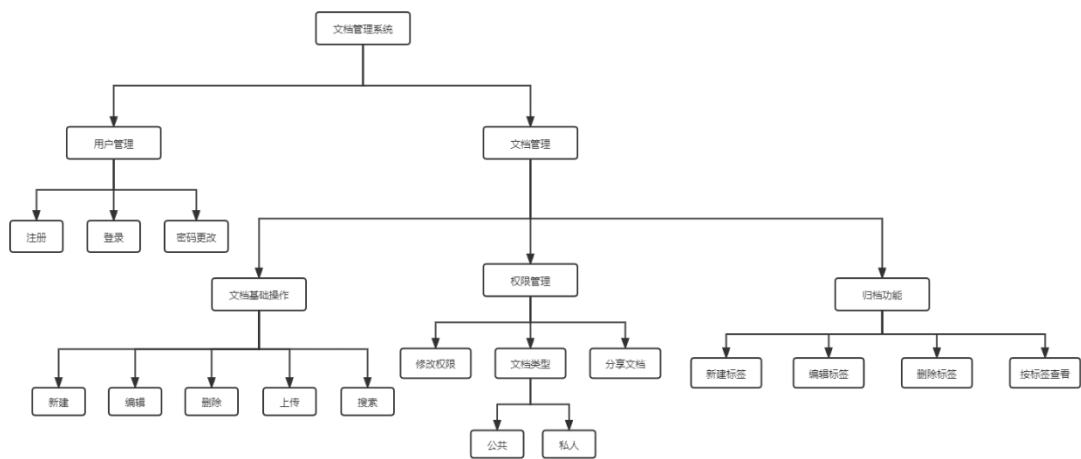
1. 实现类语雀的文档管理与编辑页面
2. 学习使用 Maven 构建 SpringBoot 项目
3. 了解熟悉使用 MyBatis 实现与数据库的交互
4. 掌握 Thymeleaf 和 BootStrap 实现前端页面
5. 基础要求
  - ✓ 个人版，不需要登录
  - ✓ 文档的增删改
    - ✓ 新建文档
    - ✓ 删除现有文档
    - ✓ 修改文档，即可在网页编辑文档
  - ✓ 主页面文档列表展示文档
6. 拓展要求
  - ✓ 多人版，包含登录
  - ✓ 多人编辑
  - ✓ 文档共享
  - ✓ 文档上传
  - ✓ 支持 Markdown 语法
  - ✓ 历史版本

- 更好的 UI

## 二、项目设计

### 1. 项目主体框架设计

本项目将着重从文档权限与归档功能出发，构建整个项目框架。

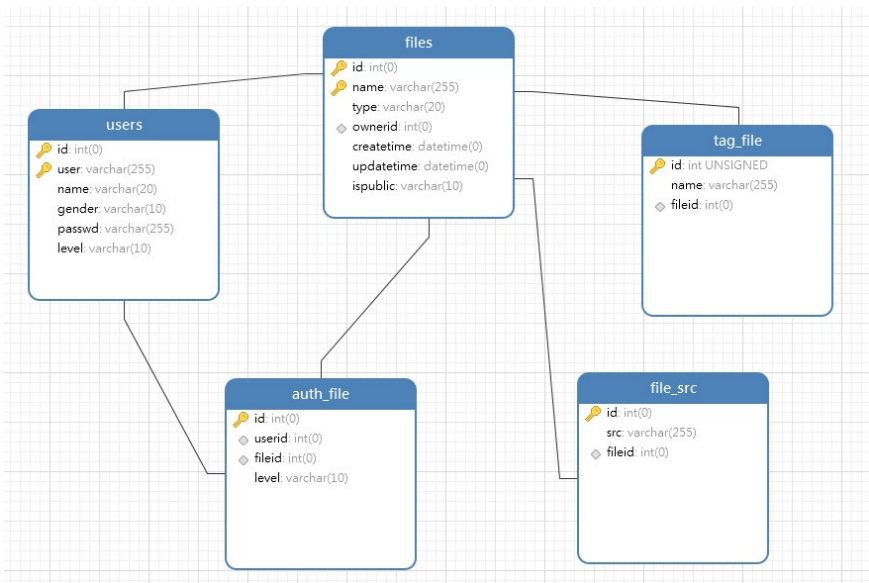


项目框架图

项目主要包含用户管理和文档管理，用户管理有登录、注册和修改密码等基础操作，而文档管理主要涉及到文档基础操作、权限管理、归档功能三个方面。

### 2. 项目数据库设计

基于项目主体框架，经过合理的分析，建立了以下数据库。



项目数据库 ER 图

其中 **files** 用于记录每一个文档的信息，**users** 用于记录每一个用户的信息。而 **tag\_file** 用于记录文档与标签一对多的关系记录，**file\_src** 用于记录文档的物理储存位置，**authfile** 则用于记录用户和文档之间多对多的权限记录信息。

### 3. 项目页面设计

登录页面：

JFiles

Home

Sign up

User

Username

Password

Password

Sign in

注册页面：

JFiles

Home

Sign up

User

Username

Name

Your Name

Gender

Choose... ▾

Password

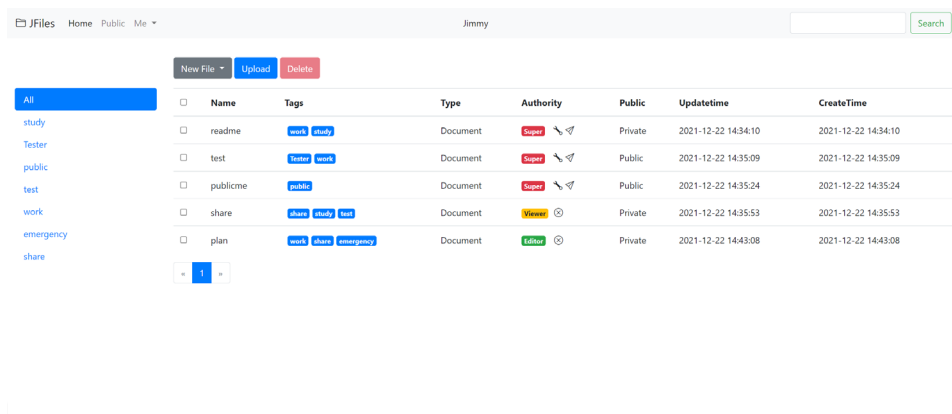
Password

Code

Invite Code

Sign up

用户主页面：



项目将以这些页面为基础，结合各种弹窗、模态框等组件构建前端界面，实现前端功能的交互和体验。

### 三、项目实施

#### 1. 利用 Maven 构建 StringBoot 项目

首先，根据助教指导和查阅相关资料完成对 pom.xml 的编写，然后构建整个项目，具体文件框架如下所示。

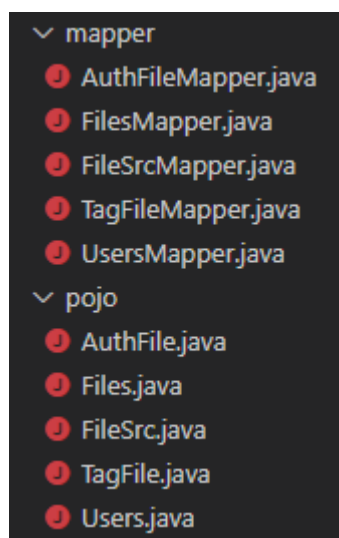
```
.
├── .mvn
├── .wrapper
├── .settings
├── .vscode
├── src
│   ├── main
│   │   ├── java
│   │   │   ├── com
│   │   │   │   ├── project
│   │   │   │   │   ├── javaWeb
│   │   │   │   │   │   ├── controller
│   │   │   │   │   │   ├── mapper
│   │   │   │   │   │   ├── pojo
│   │   │   │   │   │   ├── service
│   │   │   │   │   │   │   ├── impl
│   │   │   │   │   │   └── util
│   │   └── resources
│   │       ├── static
│   │       ├── cloudfiles
│   │       ├── css
│   │       ├── js
│   │       └── templates
│   └── test
│       ├── java
│       │   ├── com
│       │   │   ├── project
│       │   │   └── javaWeb
├── target
│   ├── classes
│   │   ├── com
│   │   │   ├── project
│   │   │   └── javaWeb
│   │       ├── controller
│   │       ├── mapper
│   │       ├── pojo
│   │       └── service
│   ├── maven-compiler-plugin
│   │   ├── compile
│   │   ├── default-compile
│   │   ├── testCompile
│   │   └── default-testCompile
│   ├── surefire-reports
│   ├── test-classes
│   │   ├── com
│   │   │   ├── project
│   │   │   └── javaWeb
```

文件架构

然后，根据数据库的设计，在 pojo 中建立相对应的类，在 mapper 中编写相对应的操作代码，并在 service 中写好相应应用层的接口，于 impl 下编写具体服务的实现方法。

## 2. 简化数据库信息读取

为了简化和减少一些重复代码的编写，笔者将 Mybatis 替换为 Mybatis-Plus，简化了一部分 mapper 的编写。同时在 lombok 库的帮助下进一步简化了 pojo 的代码编写，基于此，笔者完成了对 mapper 和 pojo 的编写。



其中各五个类，分别对应数据库的五个数据表。

## 3. Service 服务层的编写

User 的服务接口：

```
You, 2 days ago | 1 author (You)
public interface UsersService {
    void insert(Users user);
    Users selectById(int id);
    List<Users> selectAll();
    void deleteById(int id);
    Users selectByUserId(String userid);
    Users selectByName(String name);
    void update(Users user);
}
```

Tag 的服务接口:

```
You, 3 weeks ago | 1 author (You)
public interface TagFileService {
    void insert(TagFile tagFile);
    TagFile selectById(int id);
    void deleteById(int id);
    List<TagFile> selectAll();
    List<TagFile> selectByFileId(Integer fileId);
    List<String> getAllTagName();
    HashSet<String> getTagNameByFileList(List<Files> filesList);
    List<Integer> getFileIdListByName(String name);
    List<String> getTagNameByFileId(Integer fileId);
    void update(TagFile obj);
}
```

Auth 的服务接口:

```
public interface AuthFileService {
    void insert(AuthFile obj);
    AuthFile selectById(int id);
    void deleteById(int id);
    List<AuthFile> selectAll();
    AuthFile selectByFileId(int fileId,int userid);
    List<Integer> selectFileIdByUserId(int userid);
    List<AuthFile> selectByFileId(int fileId);
    void update(AuthFile authFile);
    void deleteByRelate(Integer userid, Integer fileId);
}
```

File 的服务接口:

```
You, 2 days ago | 1 author (You)
public interface FilesService {
    void insert(Files file);
    Files selectById(int id);
    void deleteByIds(List<Integer> ids);
    List<Files> selectAll();
    Files selectByName(String name);
    List<Files> selectByOwner(int ownerid);
    Page<Files> selectByPublic(String string, Integer pageNum);
    boolean uploadFile(String fileName, MultipartFile fileContent);
    void update(Files files);
    Page<Files> selectByIds(List<Integer> idList,Integer pageNum);
    List<Files> selectByIds(List<Integer> selectFileIdByUserId);
    List<Files> selectByPublic(String string);
    List<Files> searchByName(String name);
}
```

FileSrc 的服务接口：

```
You, 3 weeks ago | 1 author (You)
public interface FileSrcService {
    void insert(FileSrc obj);
    FileSrc selectById(int id);
    void deleteById(int id);
    List<FileSrc> selectAll();
    void update(FileSrc obj);
    FileSrc selectByFileId(int fileId);
}
```

在以上的接口中涵盖了一些常用的方法，诸如按照 id、name 等获取数据等等，在实际编写调用时将会十分便捷。当然这些接口函数都将在 impl 中编写具体的实现方式，依据不同情况，实现不同的功能。

#### 4. Controller 层编写

这一部分的编写与前端紧密相连，所以依照之前的前端页面初步设计，我在这一部分共编写了 5 个类如下所示。

```
1 EditorController.java
2 FilesController.java
3 LoginController.java
4 UploadController.java
5 UsersController.java
```

其中 FilesController 中主要与文档信息、标签等等相关内容，UsersController 中主要是与用户相关的功能，LoginController 中则是登录、登出相关功能，UploadController 与编辑器中内嵌图片相关，最后 EditorController 中则是与在线文档编辑相关的功能。

##### FilesController

- /{op}/: op 可选参数为 home 和 public，用于返回用户个人文档列表和公共文档列表
- /gettags: 用于获取文档的所有 Tag 标签

- `/editfile`: 用于编辑文档信息和 Tag 标签
- `/addfile`: 用于新增文档
- `/delfile`: 用于删除文档（可批量）
- `/upload`: 用于上传文档
- `/op/{tagname}`: `op` 可选参数为 `home` 和 `public`, `tagname` 为标签名, 用于文档按标签归档显示
- `/op/refreshTaglist`: `op` 可选参数为 `home` 和 `public`, 用于刷新标签列表
- `/getauth`: 用于获取用户对文档的权限
- `/getauthuser`: 用于获取文档全部权限用户
- `/changeauth`: 用于改变用户对文档的权限
- `/delauth`: 用于删除用户对文档的权限
- `/pageadd`: 显示下一页文档列表
- `/pageminus`: 显示前一页文档列表
- `/changeage/{op}`: `op` 为指定页数, 用于显示指定页的文档列表
- `/search`: 用于检索文档
- `/share`: 用于生成文档分享链接
- `/share/{code}`: 用于获取别人分享的文档

#### UserController

- `/changepasswd`: 用于修改用户密码
- `/signup`: GET 请求方式, 跳转至注册界面
- `/signup`: POST 请求方式, 用于用户注册

#### EditorController

- `/editor`: GET 请求, 用于显示编辑页面
- `/editor/{fileid}`: POST 请求, 用于获取文档内容, `fileid` 参数为文档 id
- `/editor/{fileid}`: GET 请求, 用于显示文档编辑页面, `fileid` 参数为文档 id
- `/save/{fileid}`: POST 请求, 用于保存文档内容



## LoginController

- /login: GET 请求，用于访问登录界面
- /logout: 用于用户退出登录
- /login: POST 请求，用于用户登录

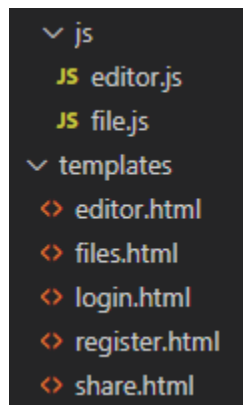
## UploadController

- /upload/img: 用于编辑器内部图片上传

以上是 Controller 部分的相关说明，具体实现可以查看详细文件内容。

## 5. 前端界面与 JS 的编写

根据项目需求，在 js 和 html 部分共编写了以下几个文件。



editor.js: 编写编辑器相关配置，以及自动保存等相关内容

file.js: 利用 jquery 和 ajax 实现一些动态刷新和局部刷新，以及内容的加载

editor.html: 编辑页面

files.html: 文档列表页面

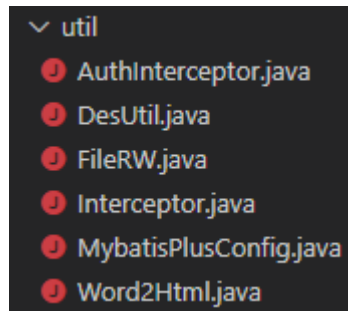
login.html: 用户登录页面

register.html: 用户注册界面

share.html: 文档分享相关界面

## 6. 其他相关组件

根据项目需求，还编写了一些工具类，用于实现部分功能，具体内容如下所示



**AuthInterceptor.java:** 拦截器，用于未登录时自动跳转至登录界面

**DesUtil.java:** DES 加解密相关，用于加密和解密分享链接等

**FileRW.java:** 文件读写相关，用于文档内容的读写

**Interceptor.java:** 对拦截器进行配置，使其能正常访问静态内容

**MybatisPlusConfig.java:** Mybatis-Plus 相关配置

**Word2Html.java:** 将 word 和 html 互相转换

## 7. 部分功能实现简单说明（具体详见代码）

### 分享功能：

通过对文档 fileid 用自己的密码对其进行 DES 加密后，再用 Base64 编码，形成分享链接，分享链接访问后会自动解码解密后，赋予用户对此文档的查看权限。

### 文档编辑功能：

这里主要在前端编写，依靠 vditor，进行相关响应函数的配置，做到了更改实时保存的功能，后端利用文件流对文件进行读写。

### 用户功能：

结合数据库，利用 session 进行登录用户信息状态的存取。

### 分页功能：

利用 Mybatis-Plus 提供的 Page 实现数据拉取的分页功能

权限功能：

利用一个单独的权限数据表储存用户与文档之间的权限关系，然后相应地在前后端进行设置。

文档上传：

利用 JQuery 的 ajax 函数实现文档的上传，后端利用文件流进行文档的读取与储存。

文档导出：

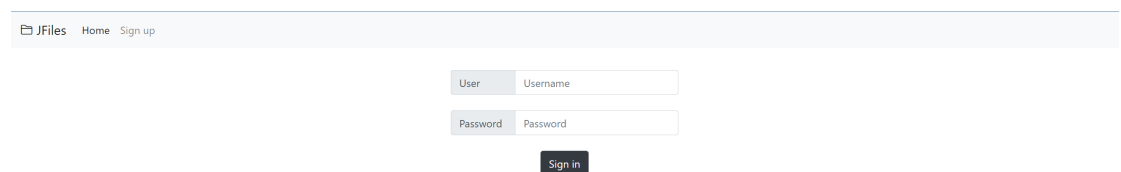
利用 vditor 编辑器，前端实现导出 pdf、markdown、html 多种格式文件。

## 四、成果展示

总的来说，项目实现了基础要求的全部功能，同时也实现了除了历史版本外的其他拓展要求功能。

### 1. 登录与注册

因为做了拦截功能，并将登录信息储存在 session 之中，所以在未登录状态下访问任意路由都会自动跳转到登录界面。在验证登录后，登录信息就会储存在 session 之中，访问登录界面时会自动跳转至主页



JFiles Home Sign up

User Username

Password Password

Sign in

登录界面

JFiles Home Sign up

User

Username

Name

Your Name

Gender

Choose...

Password

Password

Code

Invite Code

Sign up

## 注册界面

## 2. 文档列表

文档列表共分两种页面，Home 页上展示是登录用户拥有相关权限的文档，而 Public 页面显示的是所有属性为 public 的文档。即所有被设为 public 的文档，任何用户都能看到。

JFiles Home Public Me Jimmy  Search

New File Upload Delete

All  
note  
study  
Tester  
public  
test  
work  
emergency  
share

<input type="checkbox"/>	Name	Tags	Type	Authority	Public	Udattetime	CreateTime
<input type="checkbox"/>	readme	work study	Document	Super	Private	2021-12-22 14:34:10	2021-12-22 14:34:10
<input type="checkbox"/>	test	tester work	Document	Super	Public	2021-12-22 14:35:09	2021-12-22 14:35:09
<input type="checkbox"/>	publicme	public	Document	Super	Public	2021-12-22 14:35:24	2021-12-22 14:35:24
<input type="checkbox"/>	share	share study test	Document	Viewer	Private	2021-12-22 14:35:53	2021-12-22 14:35:53
<input type="checkbox"/>	plan	work share emergency	Document	Editor	Private	2021-12-22 14:43:08	2021-12-22 14:43:08
<input type="checkbox"/>	programnote	study note	Document	Super	Public	2021-12-22 21:37:58	2021-12-22 21:37:58
<input type="checkbox"/>	zyq	study	Document	Super	Private	2021-12-22 21:38:23	2021-12-22 21:38:23
<input type="checkbox"/>	bme	study test note	Document	Super	Private	2021-12-22 21:41:47	2021-12-22 21:41:47

<

1

2

>

## 个人文档列表

JFilesHomePublicMe

Jimmy

Search

AllnotestudyTesterpublicwork

<input type="checkbox"/>	Name	Tags	Type	Authority	Public	Uptime	CreateTime
<input type="checkbox"/>	test	Testerwork	Document	Super	Public	2021-12-22 14:35:09	2021-12-22 14:35:09
<input type="checkbox"/>	publicme	public	Document	Super	Public	2021-12-22 14:35:24	2021-12-22 14:35:24
<input type="checkbox"/>	programnote	studynote	Document	Super	Public	2021-12-22 21:37:58	2021-12-22 21:37:58
<input type="checkbox"/>	videolist	public	Document	Super	Public	2021-12-22 21:42:28	2021-12-22 21:42:28

<1>

## 公共文档列表

在页面的左侧显示的是标签导航条，点击相应的标签名，就能归档显示相应标签的文档列表。

JFilesHomePublicMe

Jimmy

Search

New FileUploadDelete

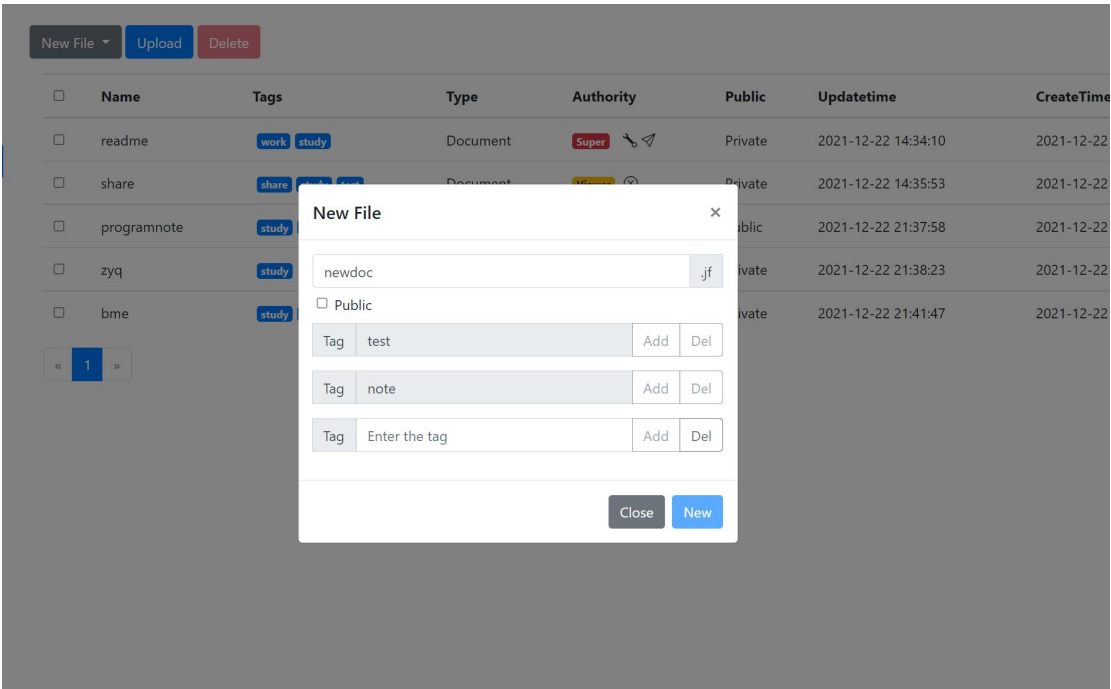
AllnotestudyTesterpublictestworkemergencyshare

<input type="checkbox"/>	Name	Tags	Type	Authority	Public	Uptime	CreateTime
<input type="checkbox"/>	readme	workstudy	Document	Super	Private	2021-12-22 14:34:10	2021-12-22 14:34:10
<input type="checkbox"/>	share	sharestudytest	Document	Viewer	Private	2021-12-22 14:35:53	2021-12-22 14:35:53
<input type="checkbox"/>	programnote	studynote	Document	Super	Public	2021-12-22 21:37:58	2021-12-22 21:37:58
<input type="checkbox"/>	zyq	study	Document	Super	Private	2021-12-22 21:38:23	2021-12-22 21:38:23
<input type="checkbox"/>	bme	studytestnote	Document	Super	Private	2021-12-22 21:41:47	2021-12-22 21:41:47

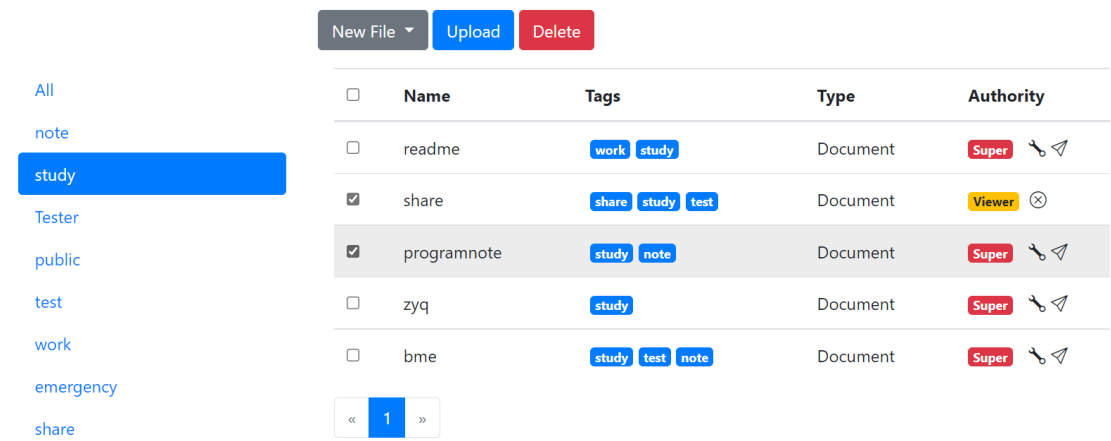
<1>

## 归档功能

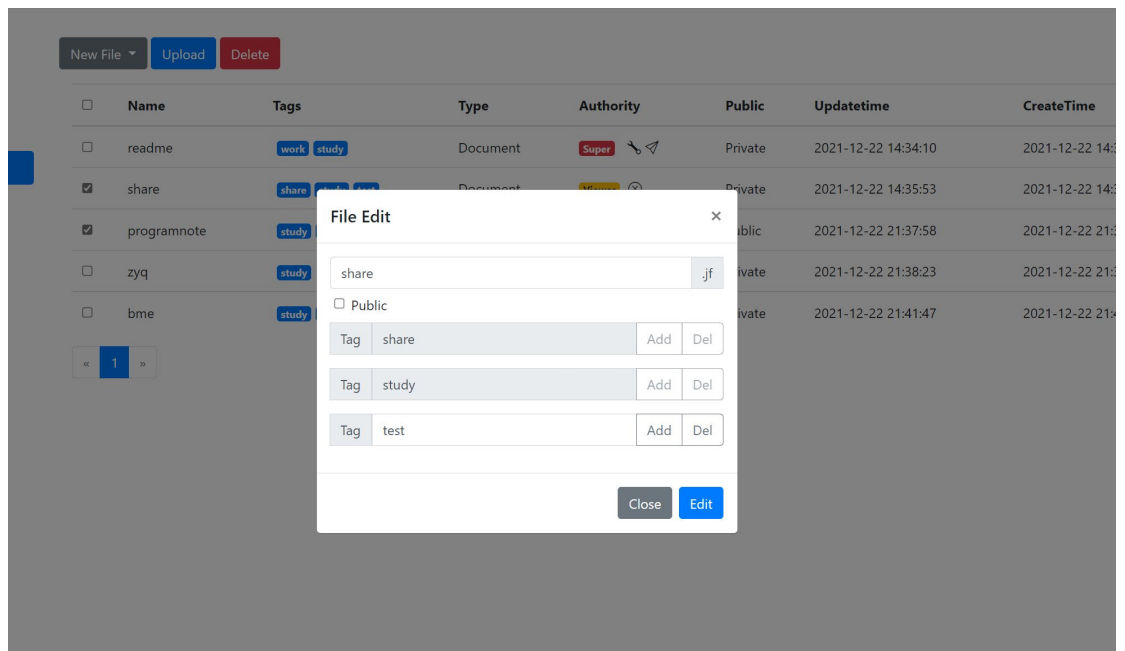
在文档列表界面，可以新建文档、编辑文档信息、修改删除文档标签、批量删除文档，以及更改文档权限。



新建文档界面

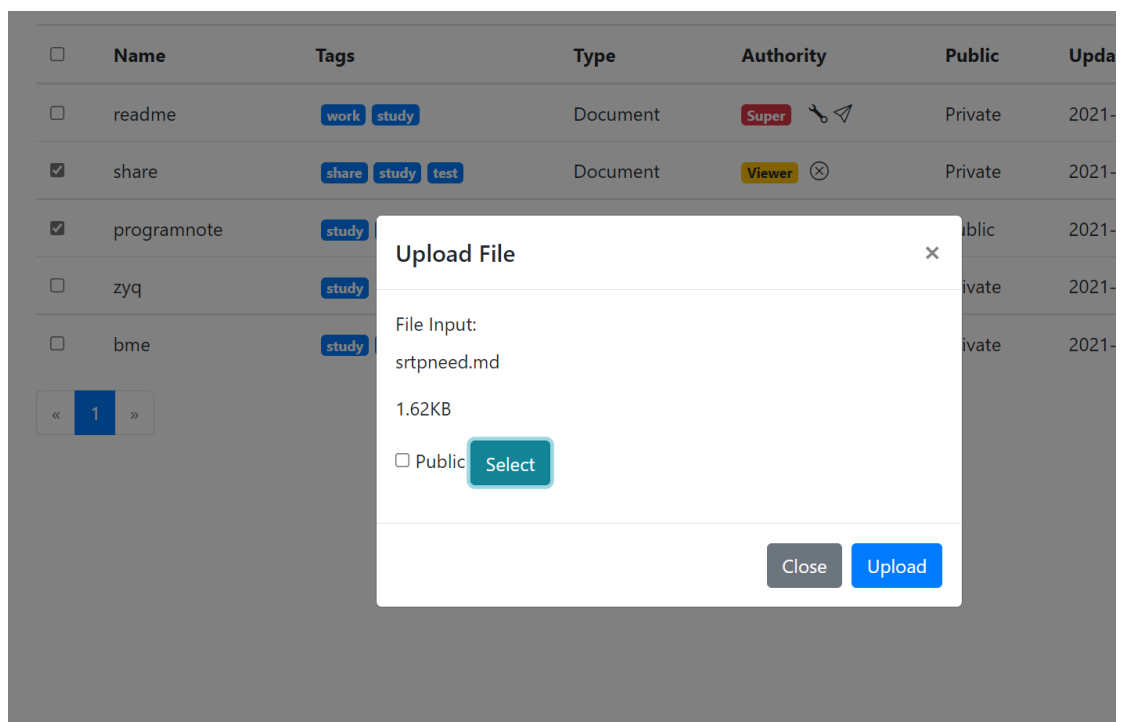


批量删除演示

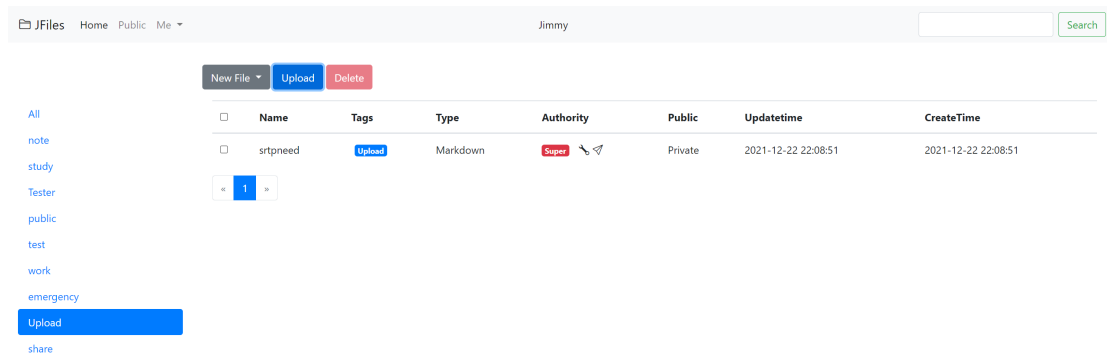


文档信息编辑界面

上传文档后，新上传的文档会自动带有 upload 标签

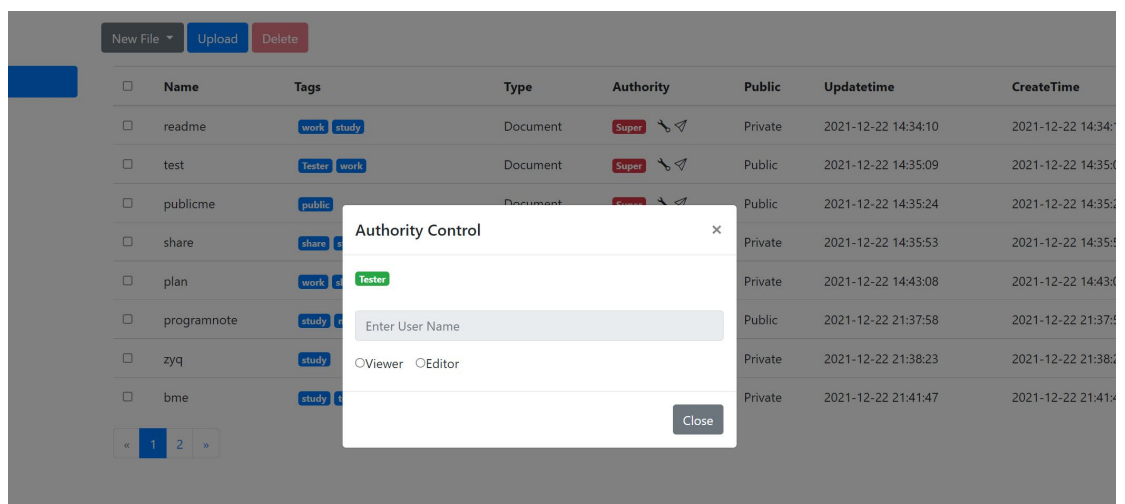


文档上传界面一



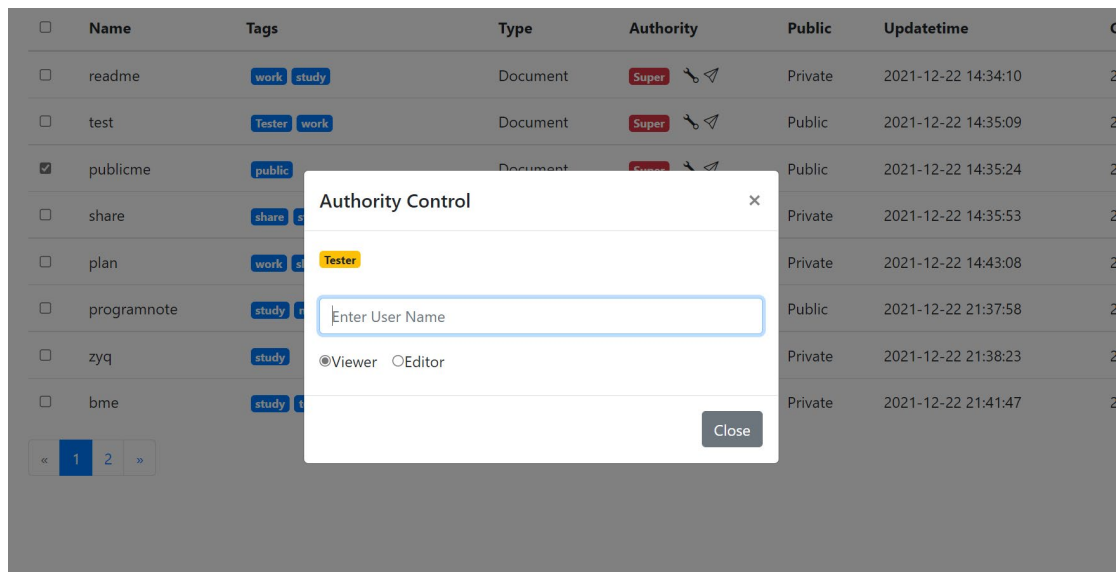
## 文档上传界面二

只有对文档拥有 **Super** 权限的用户可以对文档权限进行管理。非 **Super** 权限点击标签旁的 **x**，即可主动取消自己相应的权限。点击 **Authority** 旁的扳手图标即可以对文档权限进行修改，文档权限共分为三类，分别是 **Super**、**Editor**、**Viewer**，文档创建者拥有 **Super** 权限，拥有 **Editor** 权限的用户可对文档进行编辑，而 **Viewer** 权限对文档只拥有阅读权限。在文档权限编辑界面，只需选择相对应的权限，然后键入权限赋予对象的用户名，即可赋予权限，点击上面的权限标签即可取消相对应的权限。



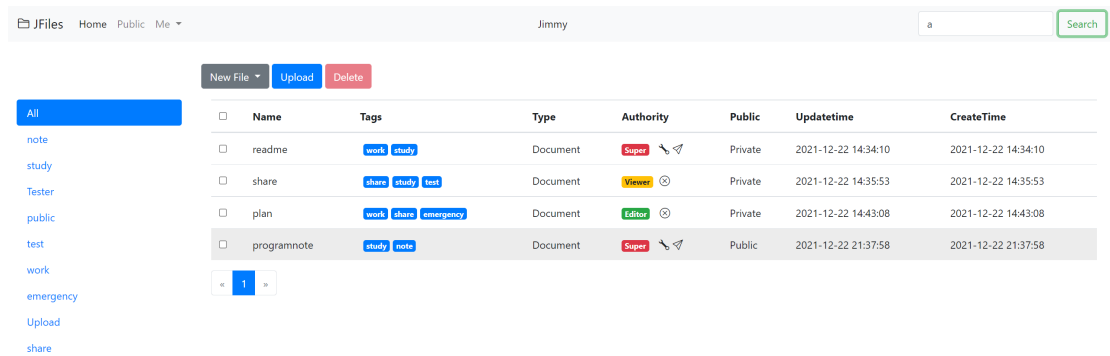
## 文档权限编辑界面一





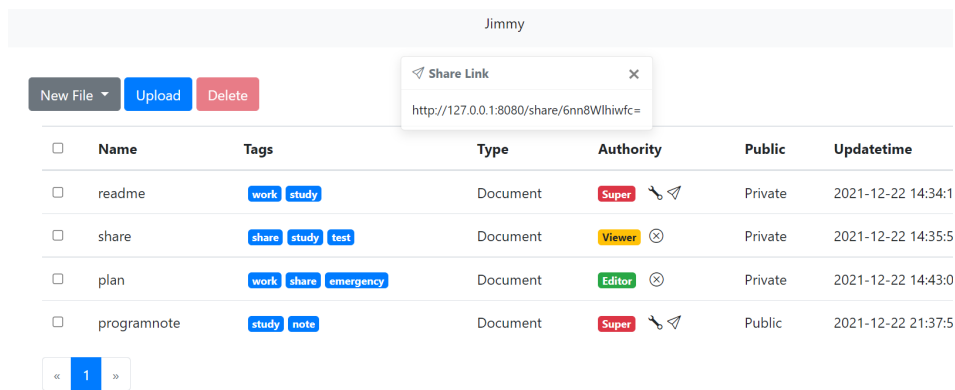
文档权限编辑界面二

在右上角的搜索框键入相应的关键字，即可搜索相对应的文档

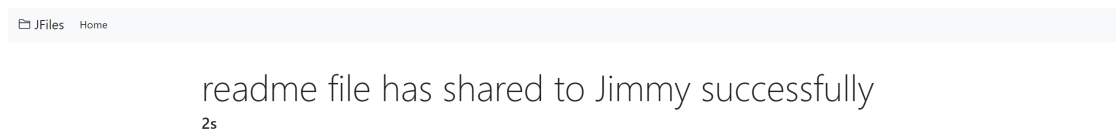


文档搜索功能演示（模糊查询）

点击 Super 权限标签旁边的小飞机，即可生成文档分享链接。文档的分享链接经过 DES 加密和 Base64 编码，访问分享链接的用户，即可拥有对此文档的 Viewer 权限。



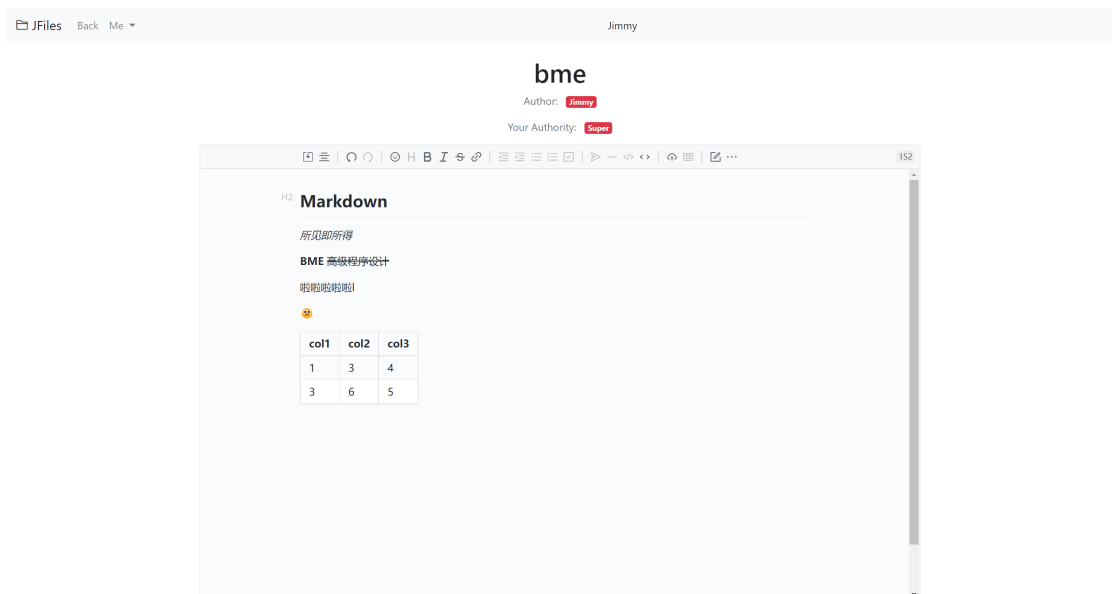
## 分享链接生成功能演示



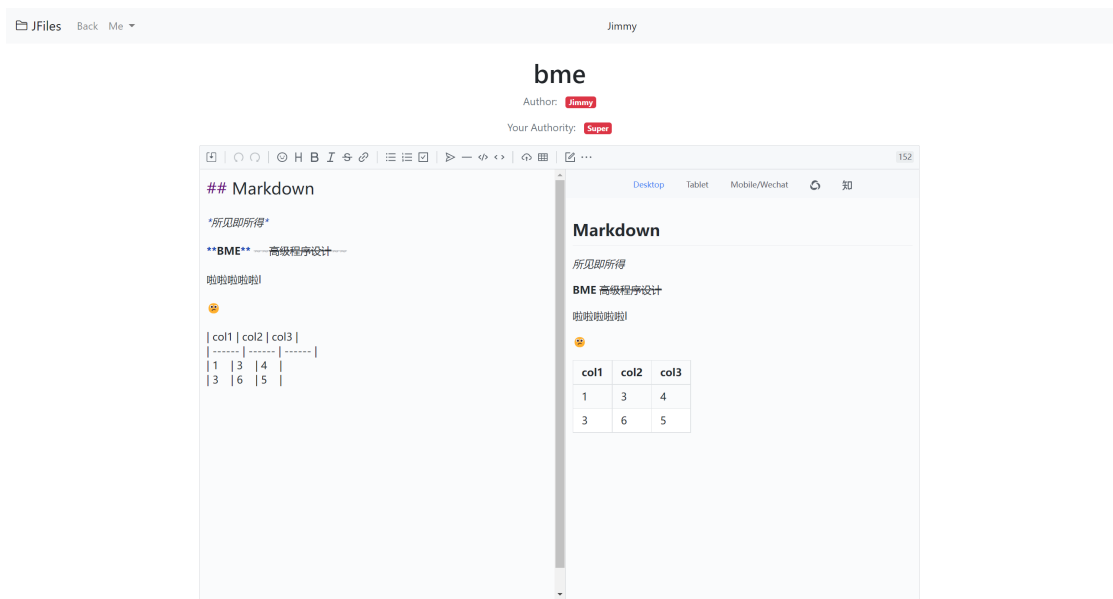
## 成功分享文档界面

### 3. 文档编辑界面

编辑器采用了 vditor，支持 markdown 语法和生成 pdf 功能，支持常见排版，支持插入图片、表格等等功能

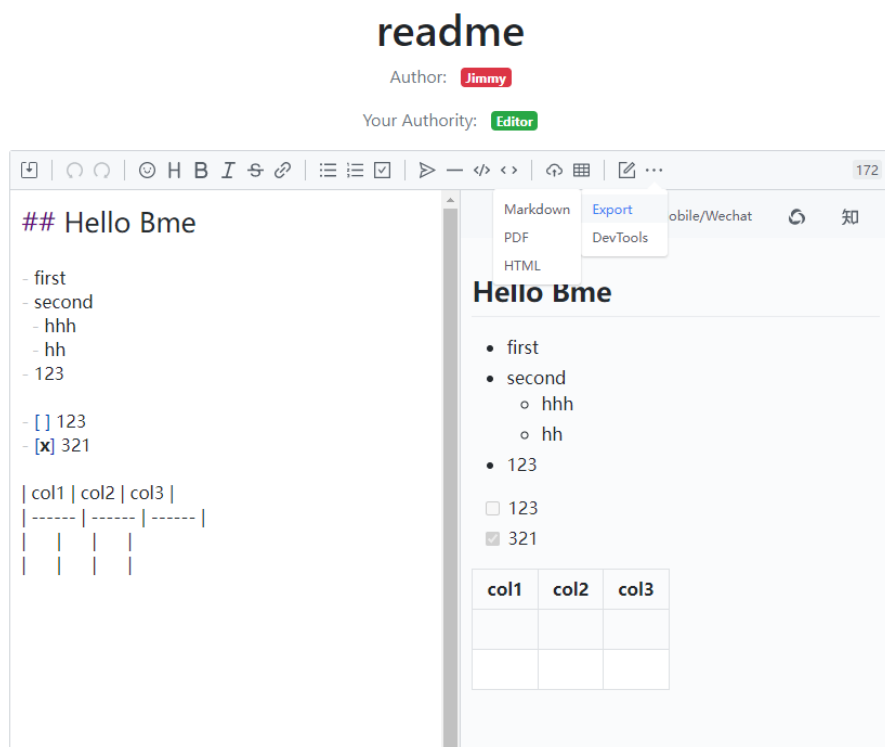


## 编辑界面演示一



编辑界面演示二

#### 4. 文档导出



共有三种导出方式，分别为 Markdown、PDF 和 HTML

### 五、讨论、心得

这次作业我从零学习了如何使用 Java 进行 Web 开发，接触到了 Maven、

Spring、SpringBoot、Mybatis、lombok 等等新的知识，也在不断的开发中，对 Java 的一些特性和语法有了更加深入的认识，尤其对 Springboot 框架有一个全面的认知，对于工业上的 Java 使用有了一定的认识。

同时，在前端界面的编写中，我学习了解了有关 html、js 和 jquery 等知识，也成功使用 ajax 实现了异步加载，让前端界面更加流畅，美观。在学习前端的时候，也了解到了诸如 vue、TypeScript 等新知识，虽然没有运用，但是也极大地拓宽了我的视野。

另外，在对程序的调试中，我也熟练地掌握了诸如 vscode、git、stackoverflow 等工具与网站，让我的学习更加高效。

最后，感谢耿老师和陆助教对我本次实验的支持与指导，让我学习到了许多实用且有趣的知识，能够更好地完成本次实验，让我对 Java 相关的知识点的掌握更加扎实。希望未来有机会，能够继续在两位的指导下学习！