文件系统开发文档

# 一、需求

## （一）身份认证

1.系统有唯一的一个管理员admin

2.管理员权限：

·进行所有文件操作：浏览、下载（文件、目录）、上传、复制、移动、删除、压缩

·更改管理员密码

·添加、删除用户账号

·更改用户的密码

·控制游客、用户、管理员的可见目录（添加、删除）

3.用户权限：

·浏览、下载（文件、目录）、上传

·更改自己的密码

·忘记密码，可请求管理员重置密码

4.游客权限：

·浏览、下载（文件）

## （二）文件系统

1.上传时，限制大小为2GB（设置整个系统的HTTP传输最大数据量为2GB）

2.对于游客、用户、管理员，有不同的可见目录，管理员可控制可见目录（添加、删除）

3.浏览文件时：

·文件分页显示，每页20行，文件按照类型排序：目录 -- 视频 -- 音频 -- 图片 -- 文本 -- 压缩包 -- 其他

·每一行以图标 -- 文件名 -- 文件大小 -- 操作按钮的顺序排列

·支持显示的文件类型图标：目录、文本、图片、音频、视频、压缩包、其他

·文件名过长时，后面以省略号结束；操作按钮中添加“查看完整文件名”按钮

·文件大小至多保留2位小数，文本所占布局宽度要固定（4位数字.2位数字MB）

4.文件可以在线预览，与浏览器兼容性有关：文本、图片、音频、视频

5.操作按钮中要有文件下载

6.下载目录时，服务端先把目录压缩成压缩包，然后再传输，传输完毕后，服务端删除压缩包。**倘若此功能无法实现，删除用户、管理员的下载目录功能**。

7.删除文件（或目录）时，弹出确认框

8.复制/移动文件（或目录）时，有两种方式：

·输入目标目录的绝对路径

·在弹出的交互窗口中选择目标目录

9.管理员添加可见目录时，有两种方式：

·输入目录的绝对路径

·在弹出的交互窗口中选择目标目录

10.多选操作：

·下载/复制/移动/删除：建立操作队列，优先操作文件，再操作目录

·压缩：建立压缩文件队列，先压缩文件，再压缩目录

## （三）前端界面

1.系统有两套主题：浅色、深色，默认为深色

2.尽可能地显示动画、过渡等效果

3.响应式布局，移动端优先，rem适配文字等大小

4.手机端测试浏览器：华为内置浏览器、UC、火狐

5.电脑端测试浏览器：Edge、Chrome

## （四）性能

1.按文件类型进行排序

2.当一个目录文件很多时，能快速地响应一页的内容（用4000张壁纸测试）

# 二、技术栈

## （一）前端

1.使用Bootstrap5框架进行开发：

·原生HTML + JinJa2模板引擎

·CSS扩展语言SASS

·原生JavaScript

2.HTML：

·用好JinJa2的变量定义、继承、部件、宏，改善HTML重复代码

·前后端交互中，多考虑使用JinJa2的过滤器

·能不用JavaScript渲染HTML就不用，尽可能地用JinJa2渲染

3.CSS：

·尽可能地使用Bootstrap的基类、工具类来改变CSS，能不用CSS就不用CSS

4.JavaScript：

·JavaScript最好只用在少数的AJAX

## （二）后端

1.使用Flask Web框架进行开发：

·使用Flask扩展的ROM技术来管理数据库

·分成两个蓝图：auth（身份认证）、main（文件系统）

·在static下区分蓝图的静态文件

·程序的启动要在命令提示符中进行

2.代码结构：

·config.py用于保存项目基本信息，以及应用启动模式

·项目根目录下的\_\_init\_\_.py编写工厂函数启动app

·utils.py保存通用工具函数、工具类

·尽可能地使用类作为代码的基本元素

3.使用SQLite数据库

4.单元测试：

·每实现一个功能，就要编写完整的单元测试

·单元测试包括：数据库模型测试、视图函数测试、功能函数或类测试

·每编写一个前端元素，就要进行电脑端、手机端浏览器的测试

5.其他：

·requirements.txt

·Git版本控制

·sql测试文件

·应用启动的bat文件：包含开发环境和生产环境

·开发文档

## （三）开发模式

1.构思，参考开发文档的需求、技术栈部分

2.将开发的具体步骤添加到开发文档

3.先编写简单视图函数，作为前后端交互的桥梁

4.编写前端页面，注意参考开发文档的技术栈

5.由前端页面驱动：

·设计数据库

·编写功能函数或

6.将后端功能放入视图函数，实现前后端连接

7.编写单元测试

8.用电脑、手机浏览器测试前端页面功能

9.更新requirements.txt文件

10.上述步骤没有问题后，用Git把代码推送到GitHub

# 三、项目基本架构

## （一）目录结构

- app

- auth

- \_\_init\_\_.py

- views.py

- main

- \_\_init\_\_.py

- views.py

- static

- auth

- sass

- css

- js

- main

- sass

- css

- js

- templates

- database\_sqlite

- untils.py

- model.py

- \_\_init\_\_.py

- admin\_dir

- run\_product.bat

- run\_development.bat

- tests

- auth

- \_\_init\_\_.py

- main

- \_\_init\_\_.py

- model

- \_\_init\_\_.py

- untils

- \_\_init\_\_.py

- base

- \_\_init\_\_.py

- \_\_init\_\_.py

- config.py

- app.py

- .gitignore

- requirements.txt

## （二）构建项目基本结构

1.环境支持：

·Python环境

·Git环境

·PyCharm环境

·Bootstrap5环境

·创建目录结构

2.编写.gitignore

3.编写config.py

·定义项目全局变量

·定义三种应用启动模式：开发（默认）、生产、测试

4.编写项目根目录下的\_\_init\_\_.py

·根据应用启动模式初始化应用

·添加Flask扩展

·注册蓝图

5.编写app.py

·创建应用

·注册命令提示符命令flask shell

6.编写admin\_dir下的启动应用bat文件

7.编写单元测试：

·编写base测试：应用是否创建成功，是否能以三种模式启动应用

·注册命令提示符命令flask test

8.简单测试一下是否能实现视图函数的前后端连接

9.更新requirements.txt文件

10.用Git把代码推送到GitHub：

·创建GitHub仓库

·将项目代码推到创建的GitHub仓库

## （三）前端界面设计

1.参考：

·百度网盘

·GitHub

·手机端ES文件管理器

·向日葵文件传输

·windows11资源管理器

2.寻找免费的网页模板

3.logo设计

# 四、身份认证

## （一）唯一管理员机制

1.数据库模型Users

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 约束 | 键 | 备注 |
| user\_id | int |  | 主键，自增 | 用户ID |
| user\_name | varchar(255) | 非空，唯一 |  | 用户名 |
| user\_password\_hash | varchar(128) | 非空 |  | 用户密码哈希值 |

**建表语句：**

CREATE TABLE IF NOT EXISTS `users`

(

`user\_id` INTEGER PRIMARY KEY AUTOINCREMENT,

`user\_name` VARCHAR(255) NOT NULL UNIQUE,

`user\_password\_hash` VARCHAR(128) NOT NULL

);

**建立ORM映射：**

在model.py中，编写实体类Users，