# API文档

## 后端返回给前端的状态码规范：

**2xx Success（成功）**

* 200 OK：请求成功，并且服务器返回了请求的数据。
* 201 Created：请求成功，并且服务器创建了新的资源。
* 202 Accepted：服务器已接受请求，但尚未处理完成。
* 204 No Content：服务器成功处理了请求，但没有返回任何内容。

**3xx Redirection（重定向）**

* 301 Moved Permanently：请求的资源已被永久移动到新位置。
* 302 Found：请求的资源临时移动到另一个URI。
* 304 Not Modified：如果请求头中包含If-Modified-Since或If-None-Match，并且资源未修改，则返回此状态码。

**4xx Client Error（客户端错误）**

* 400 Bad Request：服务器无法理解请求，可能是请求格式错误。
* 401 Unauthorized：请求需要用户的身份认证。
* 403 Forbidden：服务器理解请求，但是拒绝执行。
* 404 Not Found：服务器找不到请求的资源。
* 405 Method Not Allowed：请求方法（如GET、POST、DELETE等）不被允许。
* 406 Not Acceptable：请求的资源的内容特性无法满足请求头中的条件。
* 408 Request Timeout：请求超时。
* 409 Conflict：请求与服务器当前的状态冲突。
* 410 Gone：请求的资源已被永久删除。
* 422 Unprocessable Entity：请求格式正确，但是由于含有语义错误，无法响应。
* 429 Too Many Requests：请求过多，超出了服务器的限流。

**5xx Server Error（服务器错误）**

* 500 Internal Server Error：服务器遇到了一个未曾预料的状况，无法完成请求。
* 501 Not Implemented：服务器不支持请求的功能，无法完成请求。
* 502 Bad Gateway：服务器作为网关或代理，从上游服务器收到无效响应。
* 503 Service Unavailable：服务器目前无法使用（由于超载或停机维护）。
* 504 Gateway Timeout：服务器作为网关或代理，但是没有及时从上游服务器收到请求。

自定义状态码

10000 Request Too Frequent：请求太频繁

10001 File is Empty：文件为空

10002 SqlException: 数据库异常

……待定

## API

{

    "status": ,

    "message": ,

    "data": {

    }

}

所有的接口都以以上形式返回，前端参考这个格式拿取数据

1. 用户登录(GET)

<https://10.69.252.102/api/user/login>

前端传递：token：string

后端返回：token：string

userid: string

name：string（用户名）

message：string

status：int

1. 文件加密(POST)

<https://localhost:8443/api/fileManage/fileEncryption>

前端传递：file：file（加密前的文件）

后端返回：file：file(加密后的文件)

message：string

status：int

1. 上传文件(POST)

<https://10.69.252.102/api/fileManage/uploadFile?path=>

前端传递：file：file

path：string（文件的绝对路径）

userid：string

后端返回：message：string

status：int

1. 下载文件(POST)

<https://10.69.252.102/api/fileManage/downloadFile?path=>

前端传递：path：string（文件的绝对路径）

userid: string

后端返回：message：string

status：int

1. 搜索文件(POST)

<https://10.69.252.102/api/fileManage/searchFile>

前端传递：name：string（用户输入）

userid：用户id

后端返回：message：string

status：int

files: List<File>（查询到的所有文件的信息）

1. 创建文件夹(POST)

<https://10.69.252.102/api/fileManage/createDirectory>?path=

前端传递：path：string（某个文件夹的路径）

name：string（文件夹名称）

userid：string

后端返回：message：string

status：int

1. 获取某个文件夹下的文件(GET)

<https://10.69.252.102/api/fileManage/showFile>

前端传递：path：string（某个文件夹的路径）

后端返回：message：string

status：int

files: List<File>（查询到的所有文件的信息）

1. 获取最近上传文件（GET）（deprecated）

<https://10.69.252.102/api/fileManage/showRecentFile>

前端传递：无

后端返回：message：string

status：int

files: List<File>（查询到的所有文件的信息）

1. 获取用户文件状态信息（GET）

<https://10.69.252.102/api/fileManage/showDetails>

前端传递：userid: 用户id

后端返回：

data(array):

userid：用户ID（string）

total\_size: 文件总存储空间（int）

total\_files：总文件数（int）  
total\_uploads：总上传数（int）  
total\_downloads：总下载数（int）  
total\_documents：总文件数（int）  
total\_images：总图片数（int）  
total\_videos：总视频数（int）  
total\_audios：总音频数（int）  
total\_others：总其它文件数（int）

1. 放入回收站（PUT）

<https://10.69.252.102/api/fileManage/recycle>

前端传递：paths: List（所有要回收的文件的路径）

后端返回：message：string

status：int

1. 恢复文件（PUT）

<https://10.69.252.102/api/fileManage/recover>

前端传递：paths: List（所有要恢复的文件的路径）

后端返回：message：string

status：int

1. 删除文件（DELETE）（删除文件需要一个一个去调用，List当中只能包含一个路径）

<https://10.69.252.102/api/fileManage/deleteFile>?path=

前端传递：paths: List（所有要删除的文件的路径）

type：文件类型

userid：用户ID

后端返回：message：string

status：int

1. 获取用户上传的所有文件（GET）

<https://10.69.252.102/api/fileManage/getAllFiles>

前端传递：userid

后端返回：message：string

status：int

files: List<File>（查询到的所有文件的信息）

14.其它必要的接口……

备注：

1. 上传、下载文件、创建文件夹的path需要前端拼接好再传给后端。

示例：/home/kncsz/cloudstorage/userfile/username/name

前缀：/home/kncsz/cloudstorage/userfile/

userfile目录下存放的是所有用户的根目录文件夹，名称为用户名，所有用户上传的文件或文件夹都在根目录下。

username：替换为用户名

name: 替换为文件名或文件夹名

1. 后端需要注意，除文件加密外的所有接口都需要进行数据库操作（插入、删除、更新、查询）
2. 解释一下几个重要的参数

前端从数据库中查到的path是父目录的路径而不是绝对路径，name为名称。绝对路径为path + “/” + name做拼接。父目录路径不用拼接，直接为path

对于上传文件（POST请求）和删除文件（DELETE请求），前端不会将参数放在url中，举个例子<https://10.69.252.102/api/fileManage/downloadFile?path=> ?后面跟的就是请求参数(get方法请求参数才会在url中)

因此对于上传文件和删除文件，需要前端手动将参数path加在url当中（业务需要）

上传文件中path表示文件的绝对路径/home/kncsz/cloudstorage/userfile/username/filename

下载文件中path表示文件的绝对路径

/home/kncsz/cloudstorage/userfile/username/filename

搜索文件中name为用户的输入，并非文件名

创建文件夹中path表示文件夹的绝对路径

/home/kncsz/cloudstorage/userfile/username/directoryname

显示文件中path表示父目录的路径

例如要查询/home/kncsz/cloudstorage/userfile/username/directory1下的文件，path就为home/kncsz/cloudstorage/userfile/username/directory1

删除文件中paths每次只能传入一个文件的路径,如果要删除多个文件，需要多次调用这个接口（业务需要）

## 创建数据库

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

id：主键

userid：用户ID

name：用户名

datestamp：创建时间

CREATE TABLE user (

id INT AUTO\_INCREMENT PRIMARY KEY,

userid VARCHAR(255) UNIQUE NOT NULL,

name VARCHAR(255) NOT NULL,

datestamp VARCHAR(255) NOT NULL

);

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

id:主键

userid: 用户id（外键）

path：文件的父目录的绝对路径

name：文件名或文件夹名

size：文件大小或文件夹大小

type：类型（file，directory）

description：描述

datestamp：创建时间

lastmodified：最后修改时间

deletetime: 放入回收站的时间

intrash：是否被放入回收站

multiupload：是否是用户批量上传

CREATE TABLE file (

id INT AUTO\_INCREMENT PRIMARY KEY,

userid VARCHAR(255) NOT NULL,

path VARCHAR(255) NOT NULL,

name VARCHAR(255) NOT NULL,

size BIGINT NOT NULL,

type VARCHAR(100),

description VARCHAR(255),

datestamp VARCHAR(255),

lastmodified VARCHAR(255),

deletetime VARCHAR(255),

intrash BOOLEAN NOT NULL DEFAULT FALSE,

multiupload BOOLEAN NOT NULL DEFAULT FALSE,

FOREIGN KEY (userid) REFERENCES user(userid)

);

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

userid: 用户id（外键）

total\_size: 存储量

total\_files：总文件数

total\_uploads：总上传数

total\_downloads: 总下载数

total\_documents：总文档数

total\_images：总图片数

total\_videos ：总视频数

total\_audios：总音频数

total\_others：其它总数

CREATE TABLE filestats (

id INT AUTO\_INCREMENT PRIMARY KEY,

userid VARCHAR(255) NOT NULL,

total\_size BIGINT DEFAULT 0,

total\_files INT DEFAULT 0,

total\_uploads INT DEFAULT 0,

total\_downloads INT DEFAULT 0,

total\_documents INT DEFAULT 0,

total\_images INT DEFAULT 0,

total\_videos INT DEFAULT 0,

total\_audios INT DEFAULT 0,

total\_others INT DEFAULT 0,

FOREIGN KEY (userid) REFERENCES user(userid)

);

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/