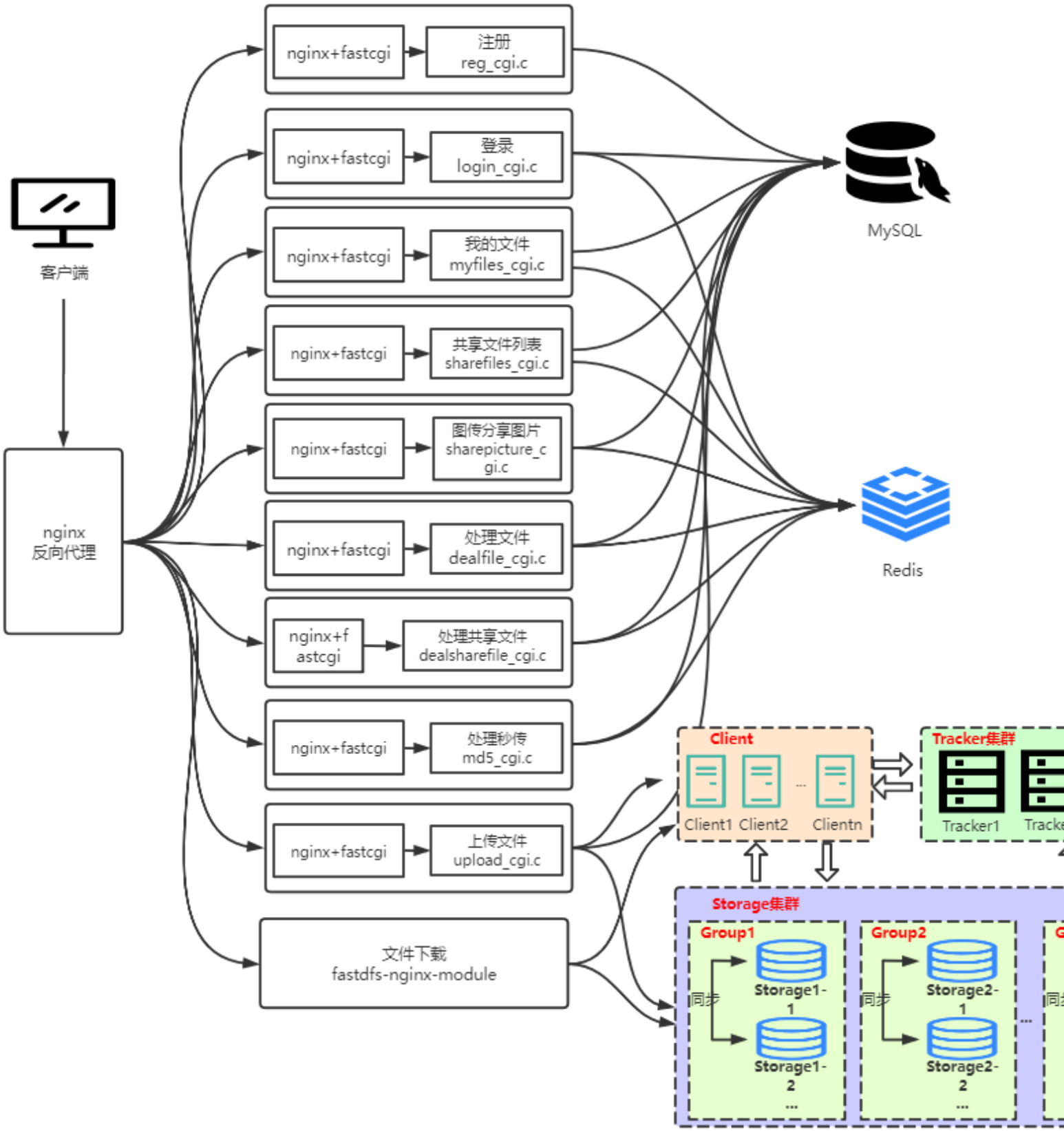
1. 后台数据处理



该框架需要结合数据库设计来看：我这里先把数据库的一些重点字段列举处理：

file_info 表描述一个文件的存储：

- md5：文件的 md5 值，文件上传先可以先匹对数据库是否存在相同的 md5 值，如果存在意味着是同一文件。
- file_id：文件 id，对应 fastdfs 的文件路径
- url：文件的完整存储路径，比如 192.168.52.139:80/group1/M00/00/00/xxx.png
- count：文件引用计数，每增加一个用户拥有此文件，此计数器+1

share_file_list 分享文件列表

- user：文件所属用户
- md5：对应的 md5 值
- file_name：文件名
- pv：文件下载量

user_file_count 用户文件数量，记录每个用户拥有的文件数量

- user：用户名
- count：文件数量

user_file_list 用户个人文件列表，记录每个人拥有的文件

- user：用户名
- md5 值：根据该值去 **file_info 进一步查找具体的文件**
- file_name：文件名
- shared_status：分享状态
- pv：下载量

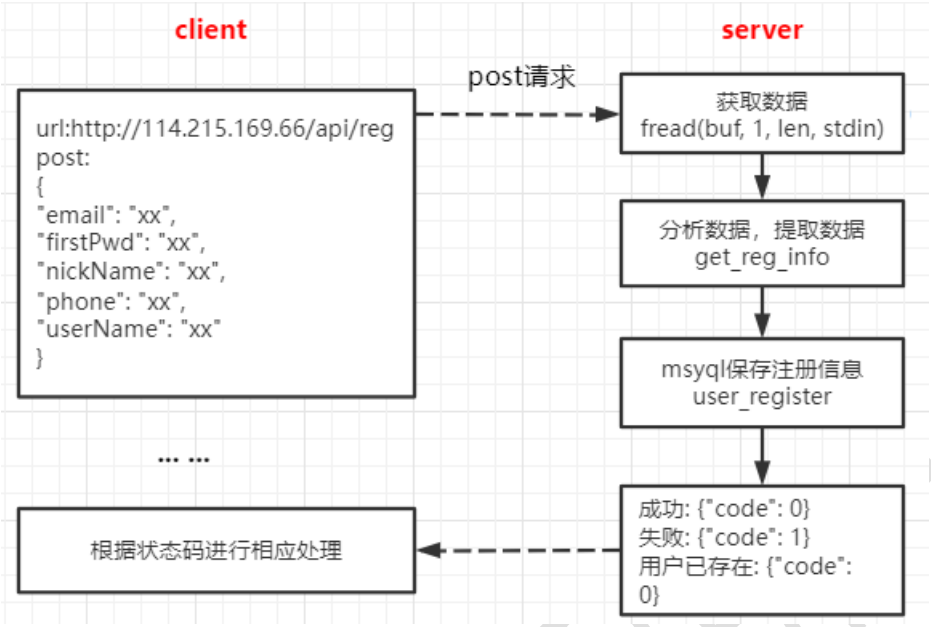
user_info 用户信息

- user_name：用户名，唯一
- password：密码，md5 加密

code 为 0 的时候是正常的，非 0 就是异常。

1.1 注册功能 reg_cgi.c

1.1.1 业务流程



1.1.2 MD5

MD5 即 Message-Digest Algorithm 5（信息-摘要算法 5），用于确保信息传输完整一致。是计算机广泛使用的杂凑算法之一（又译摘要算法、哈希算法），主流编程语言普遍已有 MD5 实现。

理论上 MD5 是不可逆的，而且 MD5 本来也不是作加密使用，而是用来校验数据的完整性，只是因为其不可逆且稳定、快速的特点，被广泛用于对明文密码的加密。

但是简单密码来说，破解者完全可以将一定范围内的密码字典全部计算出来之后存为数据库，之后直接查询进行破解。

用户重要信息(如密码)不应该明文保存到数据库，可以通过 MD5 加密后再保存：

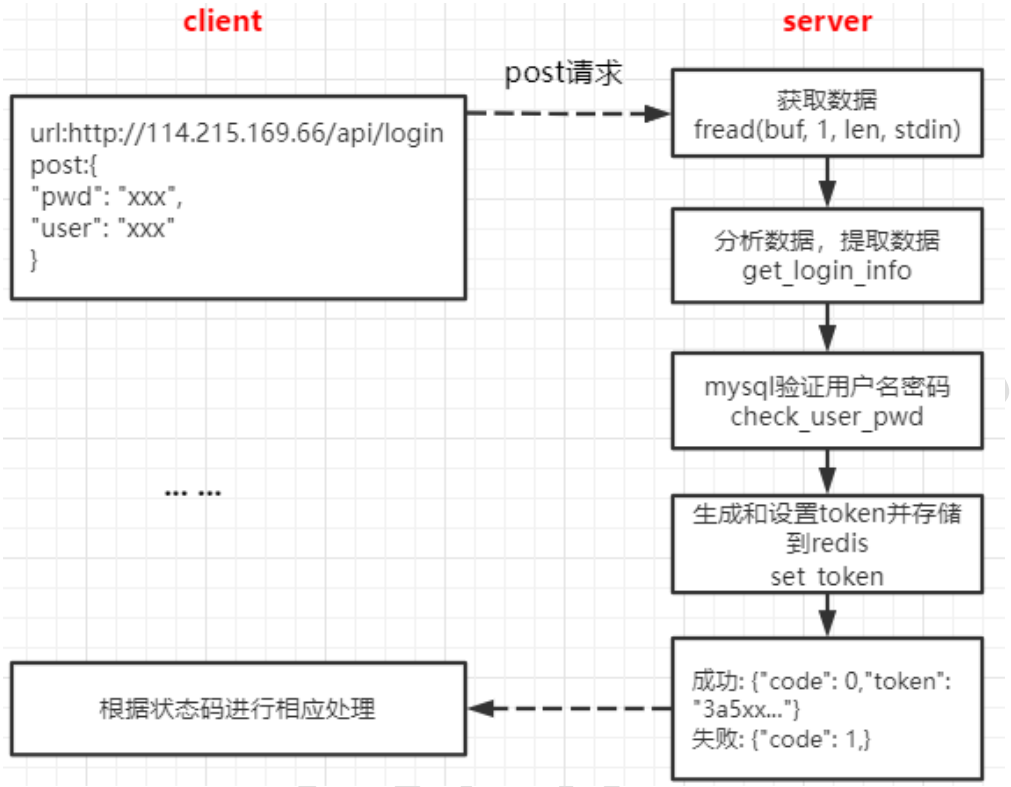
id	user_name	nick_name	password
9	milo	milo	e10adc3949ba59abbe56e057f20f883e

1.1.3 盐值加密方式拓展

https://www.yuque.com/docs/share/9276a004-7bc0-4d15-a477-c88068fb37ba?# 《给密码加盐》

1.2 登陆功能 login_cgi.c

1.2.1 业务流程



1.2.2 Token 验证

Token 的中文有人翻译成“令牌”，意思就是，你拿着这个令牌，才能过一些关卡。

Token 是一个用户自定义的任意字符串。在成功提交了开发者自定义的这个字符串之后，Token 的值会保存到服务器后台。只有服务器和客户端前端知道这个字符串，于是 Token 就成了这两者之间的密钥，它可以让服务器确认请求是来自客户端还是恶意的第三方。

这里所说的 Token，本质上就是 http session：《http session 介绍》

使用基于 Token 的身份验证方法，在服务端不需要存储用户的登录记录。大概的流程是这样的：

1) 客户端使用用户名跟密码请求登录
2) 服务端收到请求，去验证用户名与密码
3) 验证成功后，服务端生成一个 Token，这个 Token 可以存储在内存、磁盘、或者数据库里，再把这个 Token 发送给客户端
4) 客户端收到 Token 以后可以把它存储起来，比如放在 Cookie 里或者 Local Storage
5) 客户端每次向服务端请求资源的时候需要带着服务端签发的 Token
6) 服务端收到请求，然后去验证客户端请求里面带着的 Token，如果验证成功，就向客户端返回请求的数据

简单而言，本服务器程序是采用随机数+base64+md5 生成的 token，对于过期时间是通过 redis 的 key 超时机制实现。
更进一步的阅读：https://www.yuque.com/docs/share/18838587-4e59-4f2a-abd2-8a35471ead9d?# 《还分不清 Cookie、Session、Token、JWT》

1.2.3 Base64

用记事本打开exe、jpg、pdf这些文件时，我们都会看到一大堆乱码，因为二进制文件包含很多无法显示和打印的字符。

当不可见字符在网络上传输时，比如说从 A 计算机传到 B 计算机，往往要经过多个路由设备，由于不同的设备对字符的处理方式有一些不同，这样那些不可见字符就有可能被处理错误，这是不利于传输的。

为了解决这个问题，我们可以先对数据进行编码，比如 编码，变成可见字符，也就是 ASCII 码可表示的可见字符，从而确保数据可靠传输。 的内容是有 0 ～ 9，a ～ z，A ～ Z，+，/ 组成，正好 64 个字符，这些字符是在 ASCII 可表示的范围内，属于 95 个可见字符的一部分。

所以，如果要让记事本这样的文本处理软件能处理二进制数据，如使用json保存二进制信息，需要先把数据先做一个Base64编码，统统变成可见字符，再保存。

在Base64中的可打印字符包括大写英文字母A-Z、小写英文字母a-z、阿拉伯数字0-9，这样共有62个字符，此外两个可打印符号在不同的系统中而不同，通常用加号（+）和正斜杠（/）。外加“补全符号”，通常用等号（=）。

Base64是一种用64个字符来表示任意二进制数据的方法，常用于在URL、Cookie、网页中传输少量二进制数据。

Base64要求把每三个8Bit的字节转换为四个6Bit的字节（3*8 = 4*6 = 24），然后把6Bit再添两位高位0，组成四个8Bit的字节，也就是说，转换后的字符串理论上将要比原来的长1/3。

```
(原文)转换前                  11111111, 11111111, 11111111 （二进制）

(Base64 编码)转换后                  00111111, 00111111, 00111111, 00111111 （二进制）
```

1.3 上传文件 md5_cgi.c 和 upload_cgi.c

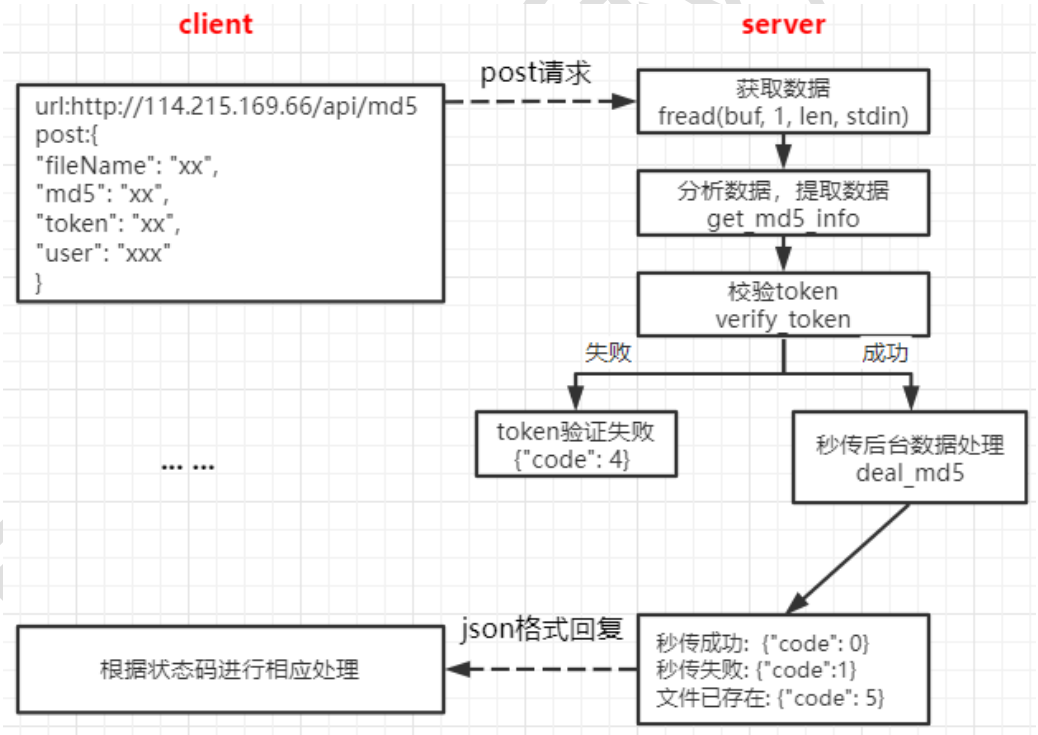
1.3.1 业务流程

每个文件都有一个唯一的 MD5 值（比如 2bf8170b42cc7124b04a8886c83a9c6f），就好比每个人的指纹都是唯一的一样，效验 MD5 就是用来确保文件在传输过程中未被修改过。

- 客户端在上传文件之前将文件的 MD5 码上传到服务器
- 服务器端判断是否已存在此 MD5 码，如果存在，说明该文件已存在，则此文件无需再上传，在此文件的计数器加 1，说明此文件多了一个用户共用。
  file_info　count
  插入用户文件列表
  用户文件计数+1

- 如果服务器没有此 MD5 码，说明上传的文件是新文件，则真正上传此文件

1.3.2 秒传文件 md5_cgi.c

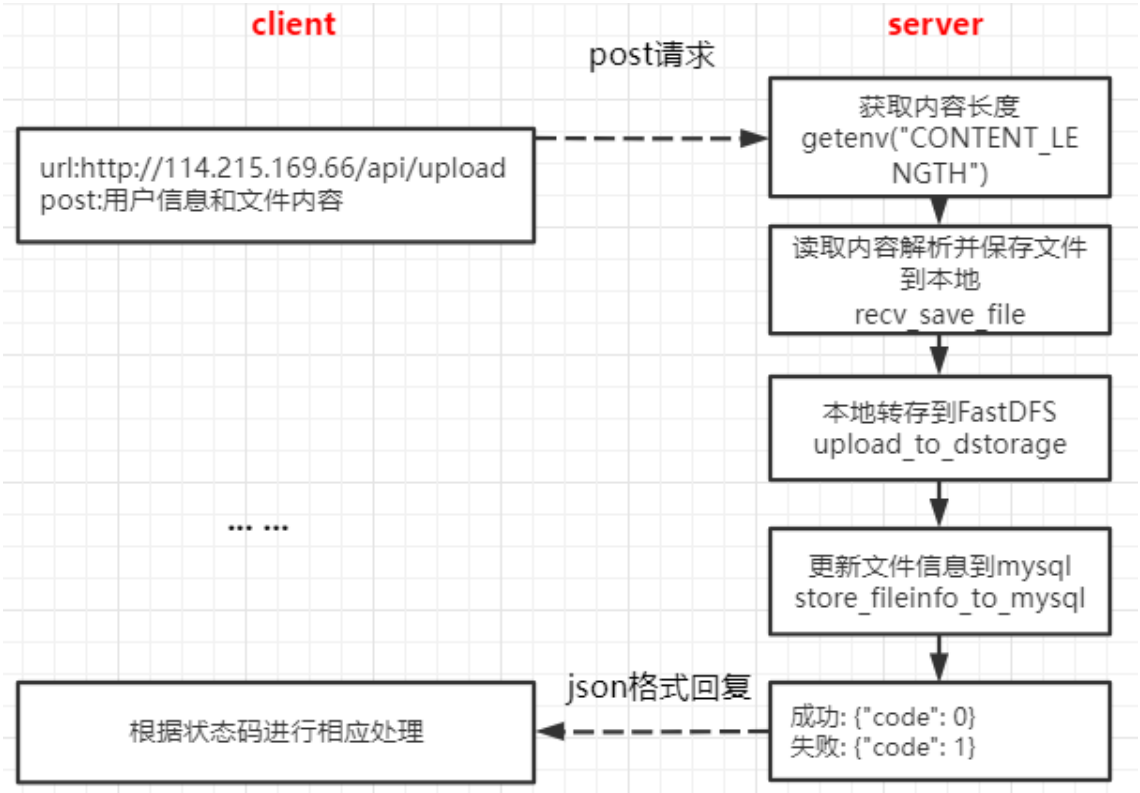


1.3.3 上传文件 upload_cgi.c

客户端上传文件信息具有一定的格式：

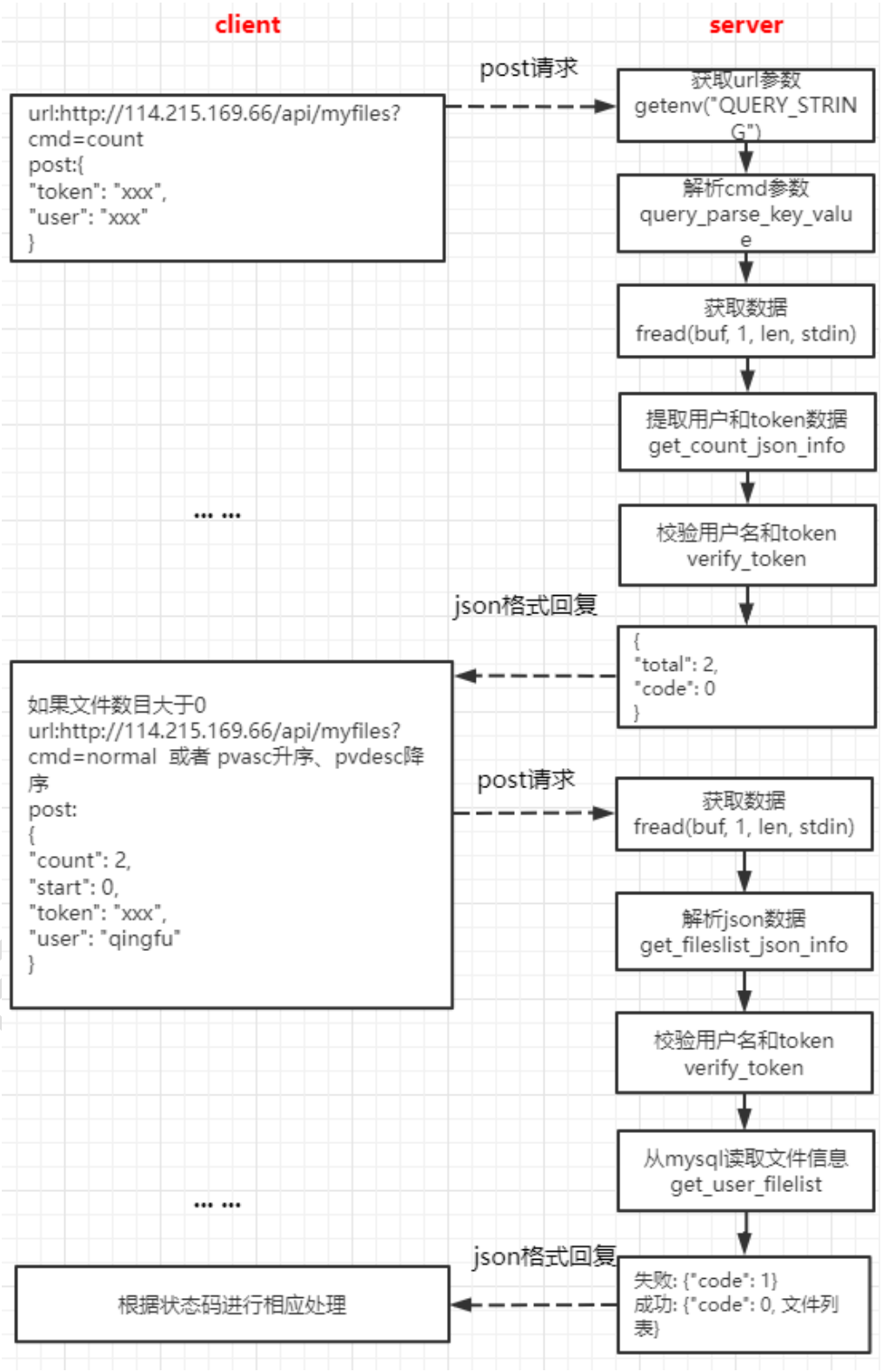
```
------WebKitFormBoundary88asdgewtgewx\r\n

Content-Disposition: form-data; user = "milo"; filename = "xxx.jpg"; md5 = "xxxx"; size = 1024\r\n
```

```
Content-Type: application/octet-stream\r\n

\r\n

真正的文件内容\r\n

------WebKitFormBoundary88asdgewtgewx
```



1.4 获取用户文件列表 myfiles_cgi.c

1.4.1 业务流程



1.4.2 文件列表 json 包

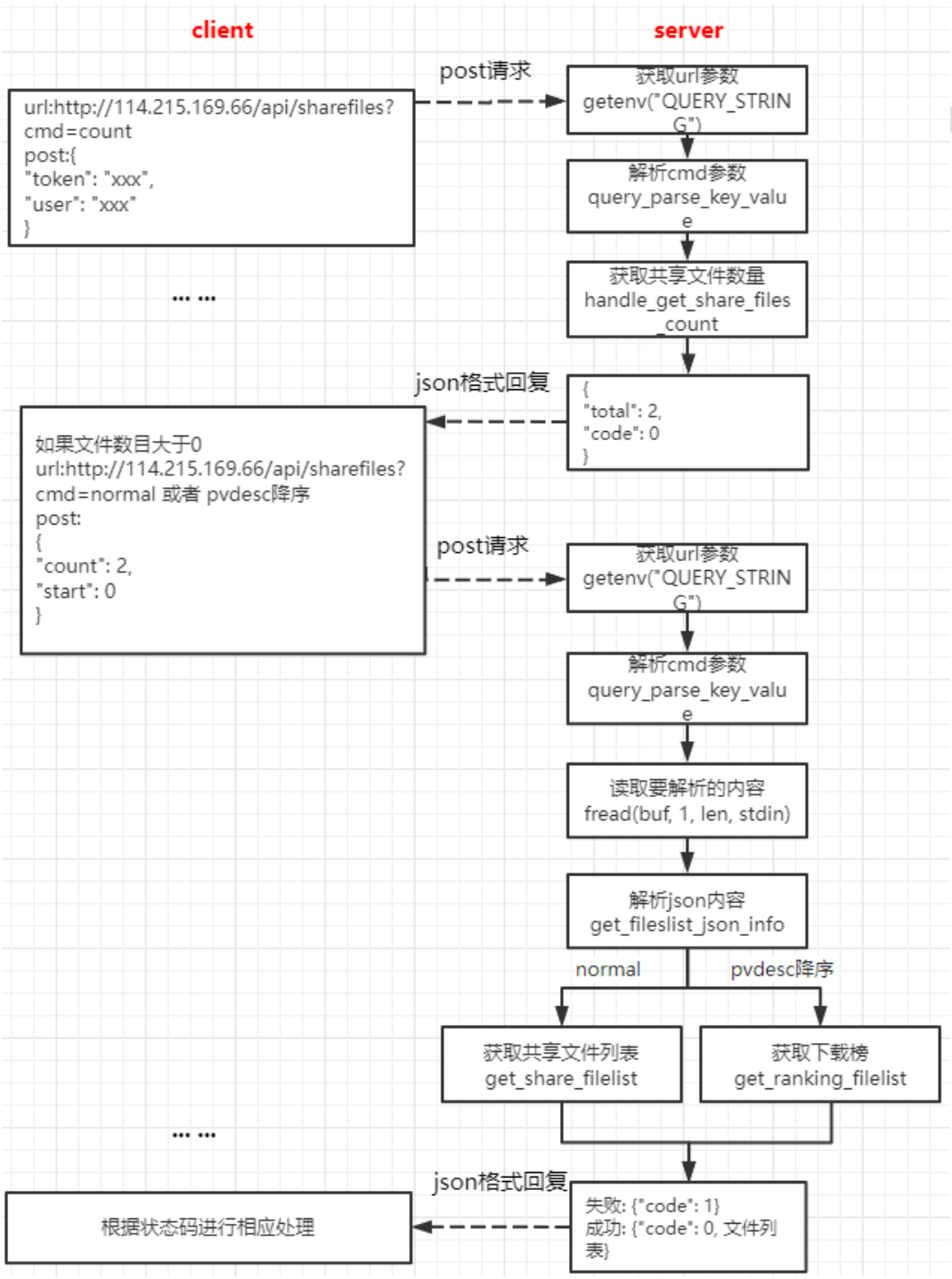
```
{
"code": 0,
"count": 2, // 如果为 0 则不需要解析 files
"total": 2, 个人文件总共的数量
"files": [
{
"user": "qingfu",
"md5": "6c5fa2864bb264c91167258b3e478fa0",
"create_time": "2021-05-15 11:31:00",
"file_name": "Qt5Svg.dll",
"share_status": 0,
"pv": 1,
"url":
"http://114.215.169.66:80/group1/M00/00/00/eBuDxWCfQHSATopyAAV8AJV_1mw866.dll",
"size": 359424,
"type": "dll"
},
{
"user": "qingfu",
"md5": "c32b5d82be3d2fcec96de06e81f87814",
"create_time": "2021-05-15 11:33:04",
"file_name": "1.png",
"share_status": 0,
"pv": 0,
"url":        "http://114.215.169.66:80/group1/M00/00/00/eBuDxWCfQPCAapRFAACtu7ei-
L0000.png",
"size": 44475,
"type": "png"
}
]
}
```

- code: 0 正常，1 错误
- user：文件上传者
- md5： 文件 md5 码
- time：文件上传的时间
- filename ： 用户名
- share_status ： 文件是否已共享，已经共享值为 1，否则值为 0
- pv ： 文件当前的下载量，下载一次，pv++
- url ： 文件存放在 fastdfs 分布式存储服务器的 url 地址
- size ： 文件大小
- type ： 文件类型

1.5 获取共享文件或下载榜 sharefiles_cgi.c

分 3 个接口：

- 获取共享文件个数 http://114.215.169.66/api/sharefiles?cmd=count
- 获取共享文件列表 http://114.215.169.66/api/sharefiles?cmd=normal
- 获取共享文件下载排行榜 http://114.215.169.66/api/sharefiles?cmd=pvdesc



sharefiles_cgi.c

文件列表

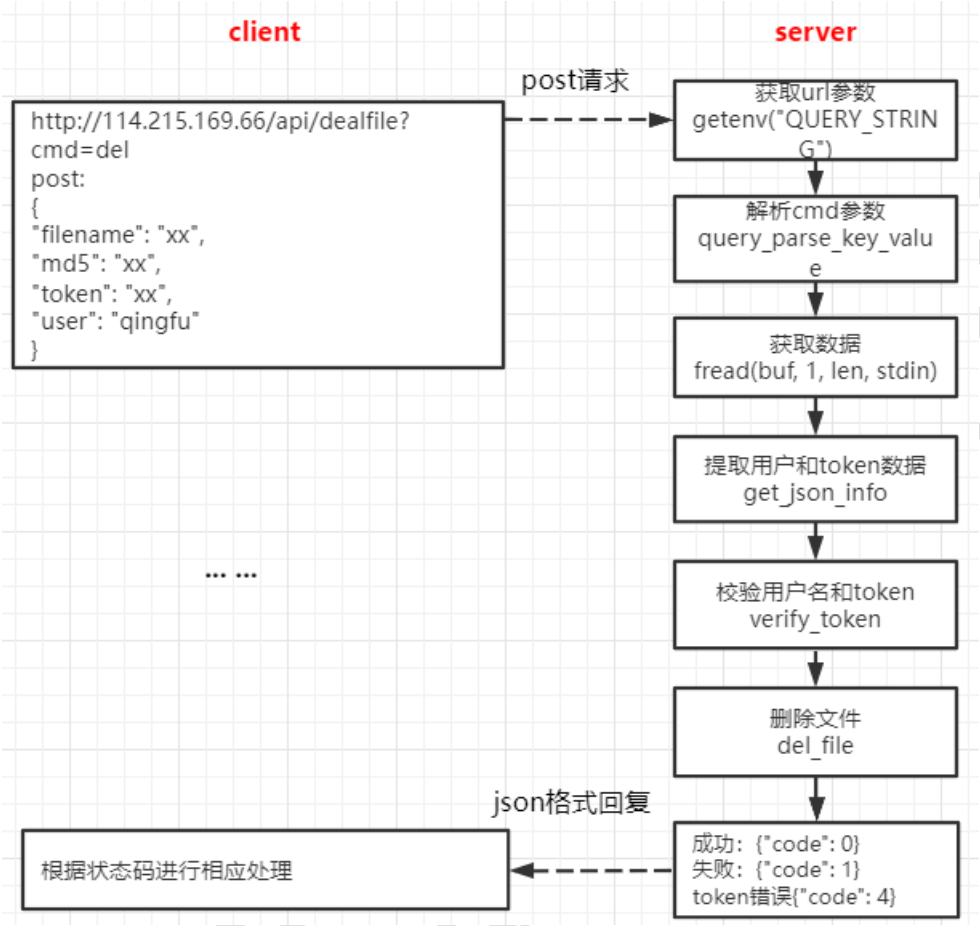
{
"code"：0,
"count": 1, **// 如果为 0 则不需要解析 files**
"total": 2, 总的文件数量
"files": [{
"user": "qingfu",
"md5": "602fdf30db2aacf517badf4565124f51",
"file_name": "Makefile",
"share_status": 1,
"pv": 0,
"create_time": "2020-07-03 23:41:15",
"url": "http://172.16.0.15:80/group1/M00/00/00/rBAAD17_Jn6AM-iuAACBBC5AIsc2532509",
"size": 33028,
"type": "null"
}]
}

重点内容：

- 共享文件和我的文件 指向的是同一个文件
- 一个人共享了一个文件，删除文件的时候也会删除共享文件；两个人同时删除文件则才会删除共享文件。
- 本质上而言，删除自己的文件时是否该删除共享文件，是业务的问题，确定了业务根据业务做代码实现即可。

1.6 处理文件 dealfile_cgi.c

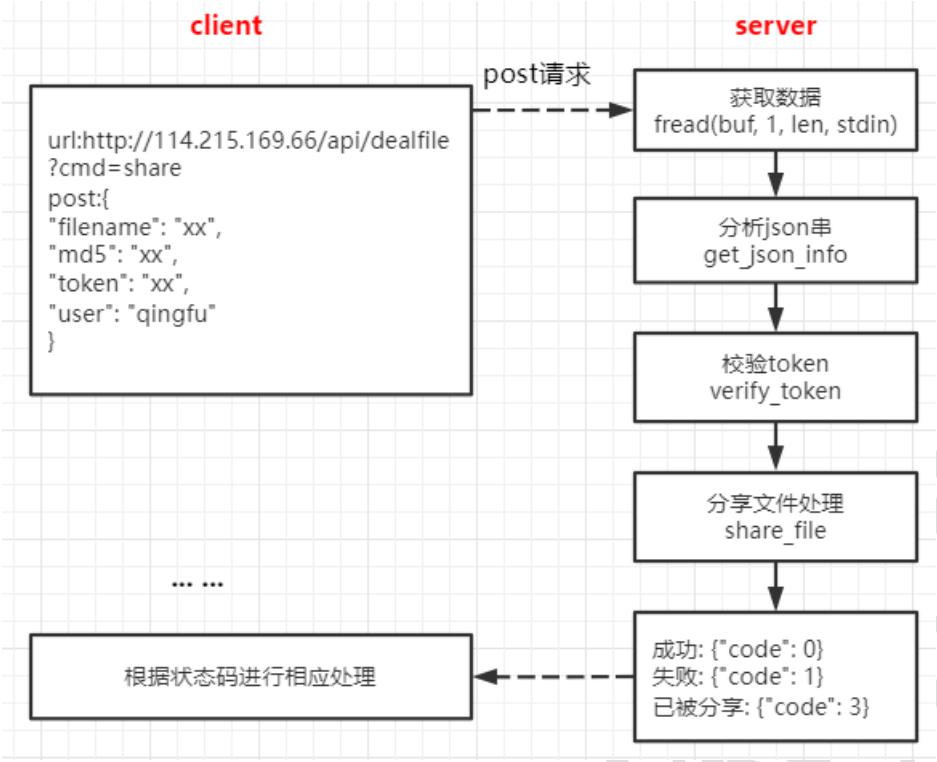
1.6.1 删除文件



- 先判断此文件是否已经分享
- 判断集合有没有这个文件，如果有，说明别人已经分享此文件(redis 操作)
- 如果集合没有此元素，可能因为 redis 中没有记录，再从 mysql 中查询，如果 mysql 也没有，说明真没有(mysql 操作)
- 如果 mysql 有记录，而 redis 没有记录，那么分享文件处理只需要处理 mysql (mysql 操作)
- 如果 redis 有记录，mysql 和 redis 都需要处理，删除相关记录

如果删除文件，则也将其从共享列表删除，其他人如果想要保存共享文件则可以转存到个人文件列表。

16.2 分享文件

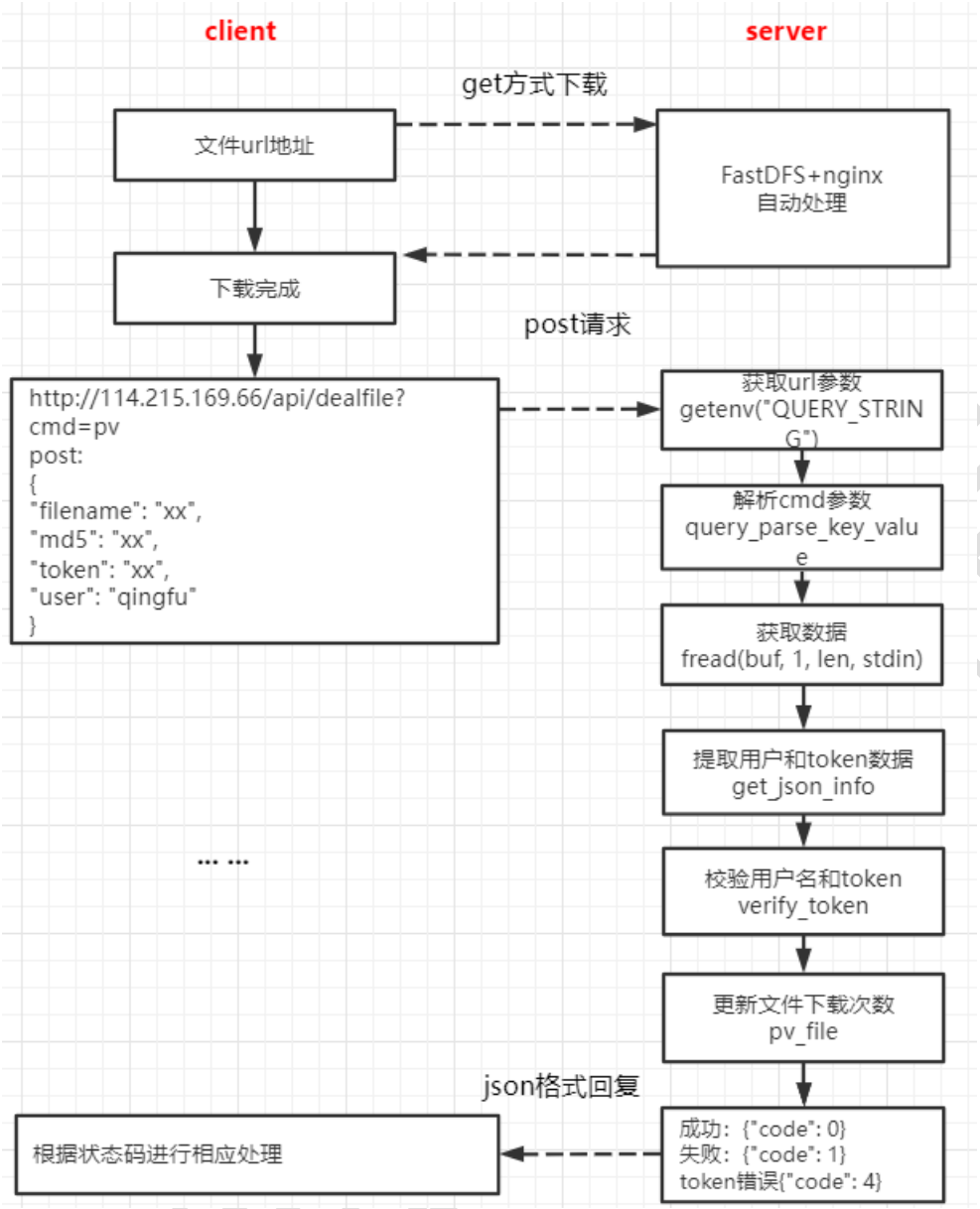


cmd=share

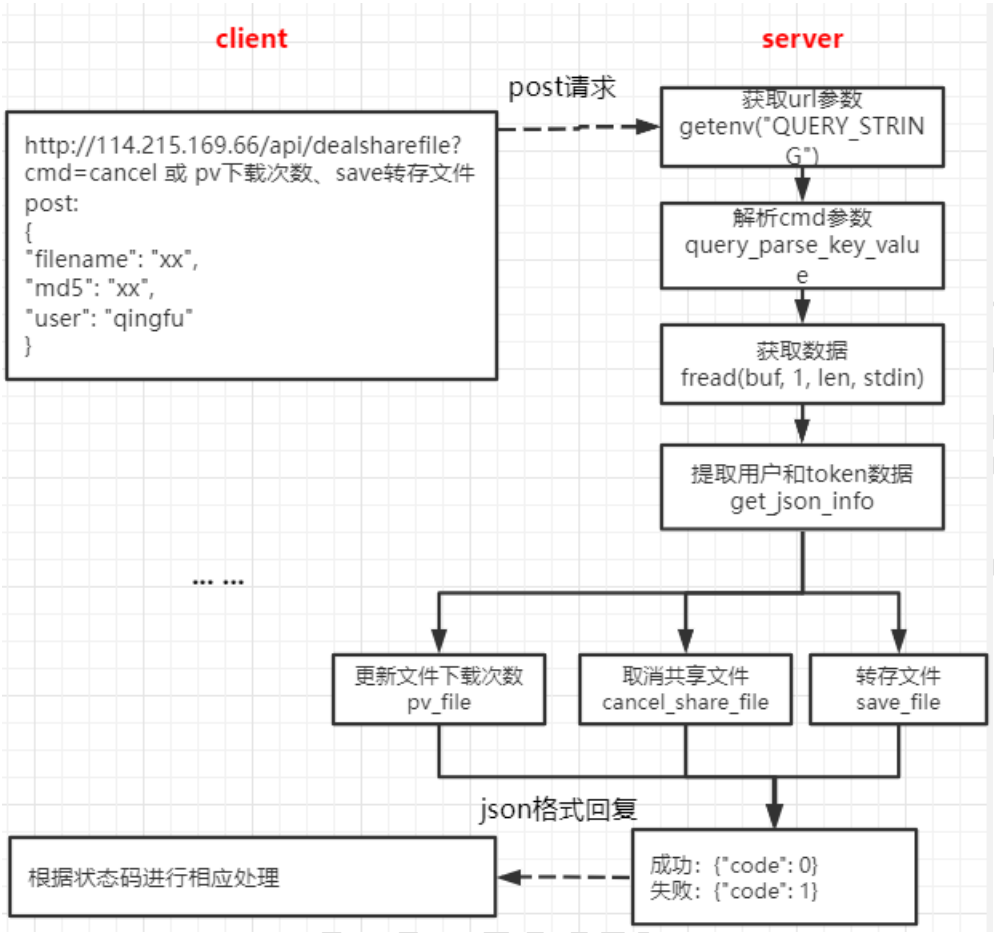
- 先在 FILE_PUBLIC_ZSET 判断是否已经有人分享该文件，对应的 fieldid 为 md5+filename，如果已经存在则返回结果。
- 在 user_file_list 将文件设置为分享状态
- 查询共享文件的数量
- 更新共享文件的数量 （共享后同属于 xxx_share_xxx_file_xxx_list_xxx_count_xxx 用户）
- 更新 FILE_PUBLIC_ZSET 排行榜
- 更新 FILE_NAME_HASH 标识和对应的文件名

16.3 下载文件和更新文件下载计数

cmd=pv
更新 user_file_list 对用的 pv 值。


client
server
get方式下载
文件url地址
FastDFS+nginx
自动处理
下载完成
post请求
http://114.215.169.66/api/dealfile?
cmd=pv
post:
{
"filename": "xx",
"md5": "xx",
"token": "xx",
"user": "qingfu"
}
获取url参数
getenv("QUERY_STRIN
G")
解析cmd参数
query_parse_key_valu
e
获取数据
fread(buf, 1, len, stdin)
提取用户和token数据
get_json_info
... ...
校验用户名和token
verify_token
更新文件下载次数
pv_file
json格式回复
根据状态码进行相应处理
成功：{"code": 0}
失败：{"code": 1}
token错误{"code": 4}

1.7 处理分享文件 dealsharefile_cgi.c



1.7.1 取消分享

http://114.215.169.66/api/dealsharefile?cmd=cancel

查询共享文件的数量 user_file_count

如果共享文件数量为 1 则删除共享文件数量对应的行

```
xxx_share_xxx_file_xxx_list_xxx_count_xxx
```

如果共享文件数量>1，则更新共享文件数量-1

将文件从 share_file_list 删除

将文件从 FILE_PUBLIC_ZSET 排行榜删除

1.7.2 转存文件

http://114.215.169.66/api/dealsharefile?cmd=save

- 先查询是个人文件列表是否已经存在该文件。
- 增加 file_info 表的 count 计数，表示多一个人保存了该文件。
- 个人的 user_file_list 增加一条文件记录
- 更新个人的 user_file_count

当我们转存该文件后，即使分享者删除自己的文件，不会影响到我们自己转存的文件。

1.7.3 下载共享文件

http://114.215.169.66/api/dealsharefile?cmd=pv

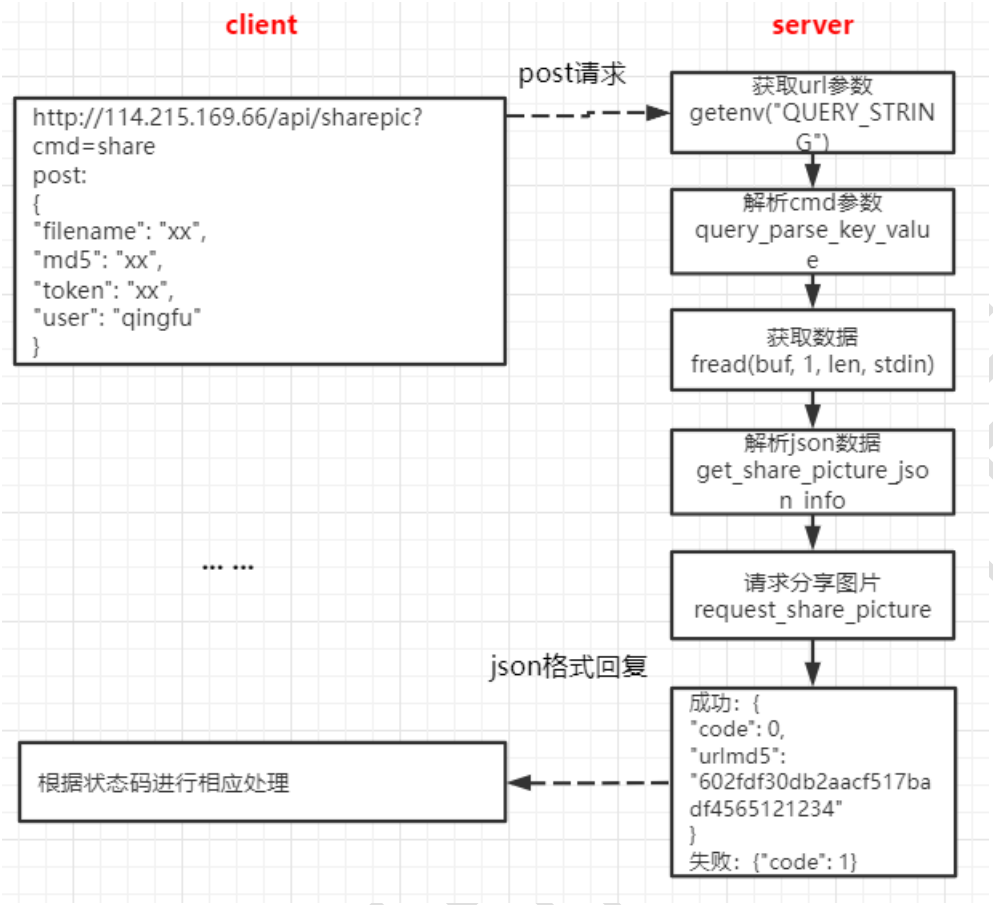
更新 share_file_list 的 pv 值
更新 redis 里的 FILE_PUBLIC_ZSET，用作排行榜

1.8 图床分享图片 sharepicture_cgi.c

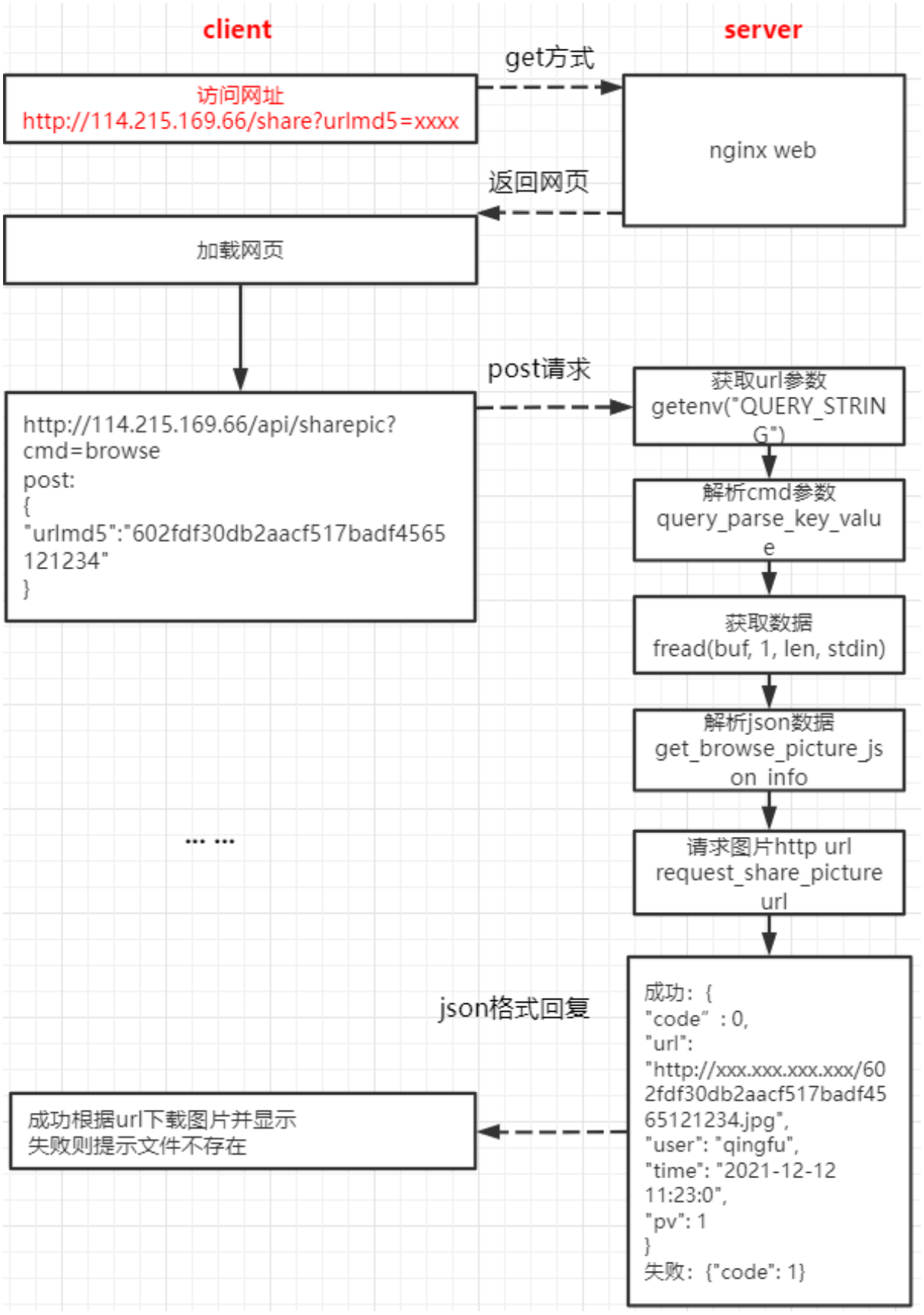
sharepicture_cgi.c 分享后每个人都可以看到。

- 图片分享：http://114.215.169.66/api/sharepic?cmd=share
- 图片浏览：http://114.215.169.66/api/sharepic?cmd=browse
- 我的图片分享：http://114.215.169.66/api/sharepic?cmd=normal
- 取消图片分享：http://114.215.169.66/api/sharepic?cmd=cancel

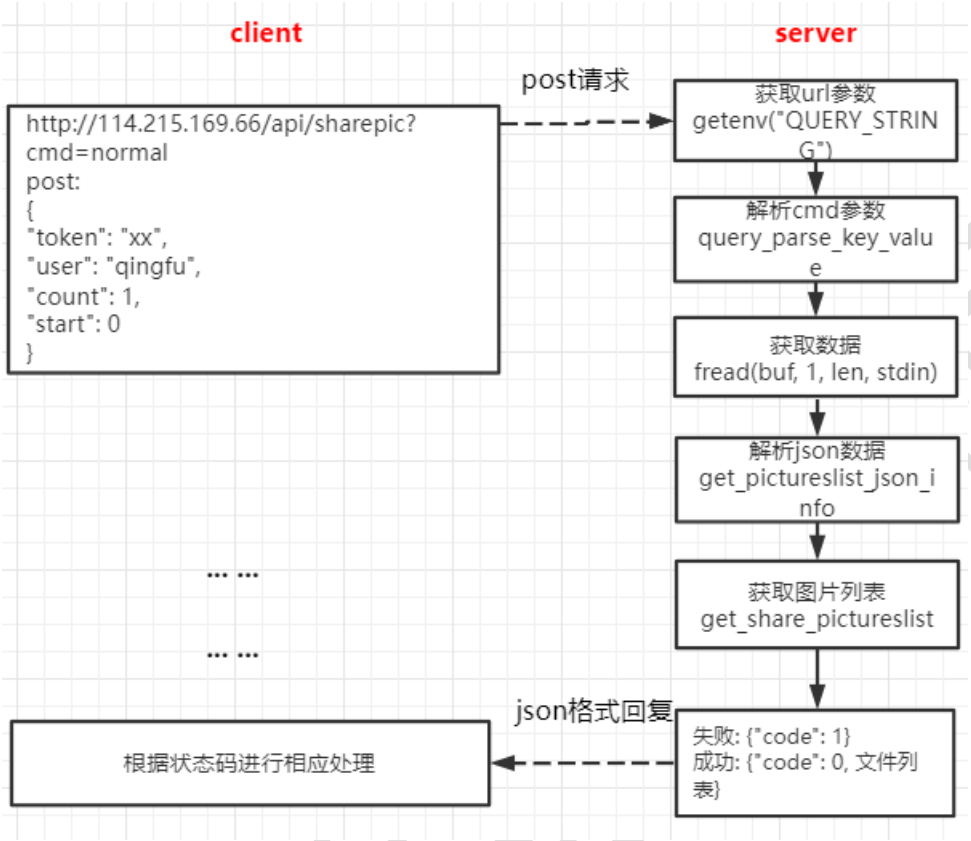
1.8.1 图片分享


client
server
post请求
http://114.215.169.66/api/sharepic?
cmd=share
post:
{
"filename": "xx",
"md5": "xx",
"token": "xx",
"user": "qingfu"
}
获取url参数
getenv("QUERY_STRIN
G")
解析cmd参数
query_parse_key_valu
e
获取数据
fread(buf, 1, len, stdin)
解析json数据
get_share_picture_jso
n_info
... ...
请求分享图片
request_share_picture
json格式回复
成功：{
"code": 0,
"urlmd5":
"602fdf30db2aacf517ba
df4565121234"
}
失败：{"code": 1}
根据状态码进行相应处理

1.8.2 图片浏览


client
server
get方式
访问网址
http://114.215.169.66/share?urlmd5=xxxx
nginx web
返回网页
加载网页
post请求
http://114.215.169.66/api/sharepic?
cmd=browse
post:
{
"urlmd5":"602fdf30db2aacf517badf4565
121234"
}
获取url参数
getenv("QUERY_STRIN
G")
解析cmd参数
query_parse_key_valu
e
获取数据
fread(buf, 1, len, stdin)
解析json数据
get_browse_picture_js
on_info
... ...
请求图片http url
request_share_picture
url
json格式回复
成功：{
"code" : 0,
"url":
"http://xxx.xxx.xxx.xxx/60
2fdf30db2aacf517badf45
65121234.jpg",
"user": "qingfu",
"time": "2021-12-12
11:23:0",
"pv": 1
}
失败：{"code": 1}
成功根据url下载图片并显示
失败则提示文件不存在

18.3 我的图片分享



返回结果范例：

```
{
"code"：0,
"count": 1, // 如果为 0 则不需要解析 files
"total": 2, 总的文件数量
"files": [{
"user": "qingfu",
"filemd5": "602fdf30db2aacf517badf4565124f51", // 不显示
"file_name": "Makefile",
    "urlmd5": "rBAAD17_Jn6AM-iuAACBBC5AIsc2532509"
"pv": 0, // 浏览量
"create_time": "2020-07-03 23:41:15", // 分享时间
"size": 33028
}]
}
```

18.4 取消图片分享