目录

[一、框架设计 2](#_Toc28085)

[二、规范 2](#_Toc2697)

[1、文件夹： 2](#_Toc24627)

[2、变量命名： 2](#_Toc22007)

[3、function命名： 2](#_Toc24280)

[4、代码规范： 2](#_Toc23131)

[三、C/S的身份识别 2](#_Toc26443)

[1、管理员登录 2](#_Toc15426)

[2、仿照cookie模式简化的用户登录系统。 3](#_Toc7036)

[四、网络编程 3](#_Toc12062)

[1、客户端发送请求格式： 3](#_Toc15665)

[2、服务器向客户端发送信息格式： 3](#_Toc24895)

[五、数据库设计 4](#_Toc20146)

[1、考研信息数据库 4](#_Toc27699)

[2、反馈信息数据库 4](#_Toc18965)

[六、用户功能设计 4](#_Toc5490)

[1、查询 4](#_Toc11200)

[2、反馈 5](#_Toc18771)

[七、管理员功能设计 6](#_Toc12822)

[1、管理员登录 6](#_Toc19082)

[2、查看数据库情况 7](#_Toc873)

[3、处理反馈 8](#_Toc8904)

[4、备份 9](#_Toc13481)

[5、修改数据库信息 10](#_Toc1334)

[6、删库跑路 10](#_Toc15982)

[八、界面设计 10](#_Toc6621)

[1、框架设计 10](#_Toc6975)

[2、主界面 10](#_Toc8506)

[3、查询结果 10](#_Toc24916)

[4、反馈界面 11](#_Toc6547)

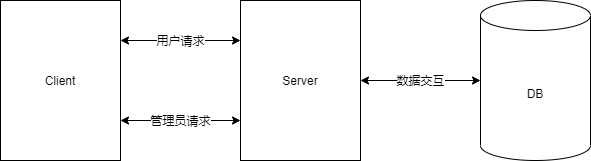
[5、提交反馈弹窗 11](#_Toc31522)

[6、管理员登录 11](#_Toc7106)

[7、管理员界面 12](#_Toc32104)

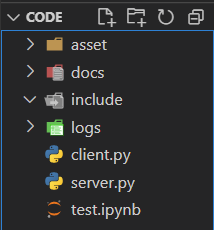
[8、反馈处理界面 12](#_Toc5013)

一、框架设计



二、规范

1、文件夹：



code文件夹——main，include文件夹、logs文件夹、asset文件夹、docs文件夹

asset文件夹——素材（图片等）

docs文件夹——文档（开发计划等）

include文件夹——头文件

logs文件夹——临时文件（主要是cookie文件）

后续可能会根据测试需要添加test文件夹

2、变量命名：

首单词首字母小写，第二个单词起首字母大写或下划线分隔。

3、function命名：

所有单词首字母大写，命名尽量体现功能，function前注释表明input、output。

4、代码规范：

尽量保证面向对象的格式（方便后面画类图……）

三、C/S的身份识别

1、管理员登录

服务器本地维护管理员账号密码信息txt文件（提前写好）。

信息保存在logs文件夹下。

admin.txt：

name：admin

password：111

2、仿照cookie模式简化的用户登录系统。

cookie检查发生在客户端和服务器连接时，在所有操作之前。

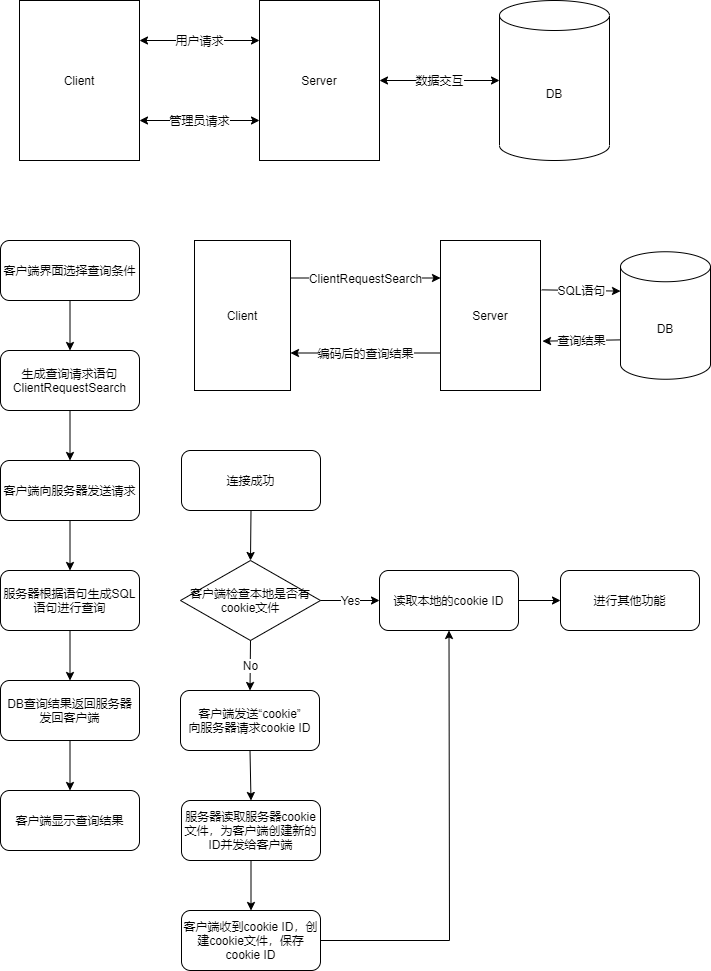
服务器和客户端都维护一个本地缓存的cookie文件，客户端cookie保存本客户端的ID标识，服务器的cookie保存访问过本服务器的所有用户的ID标识。

cookie文件用txt。

用户第一次访问服务器时，首先检查客户端本地是否已有cookie文件：

若有，则之间进行其他功能；若没有，则向服务器发送生成cookie请求“cookie”，服务器创建新的cookie ID存入服务器的cookie文件并发送给客户端，客户端生成cookie文件保存cookie ID。

cookie文件存储在logs目录下。



四、网络编程

使用socket，UDP连接（正常来说我们这个应用场景应该用TCP，但是TCP太麻烦了略……）。

一对多通信，部署了数据库的电脑作为服务器。

1、客户端发送请求格式：

（1）用户功能：

“*cookie ID\_功能（search或feedback或fsearch）\_内容*”

（2）管理员功能：

登录：“*admin\_login\_name\_password*”

查看数据库：“*admin\_monitor*”

获取反馈列表：“*admin\_feedbacklist*”

处理反馈：“*admin\_reply\_feedback ID\_回复内容*”

备份：“*admin\_backup\_编号*”（1-院校信息，2-反馈信息，3-全部）

2、服务器向客户端发送信息格式：

（1）用户功能：

返回查询结果：“*院校1-专业-数学-英语-科目一-科目二\_院校2-专业-数学-英语-科目一-科目二*”，每条信息以\_分隔，每条信息内的不同信息以-分隔。

返回反馈状态：100（反馈成功），111（反馈失败）

返回反馈进度：“*反馈内容1-反馈状态1-回复1\_反馈内容2-反馈状态2-回复2*”，回复在没有时可以为空（NULL），其他同查询结果。

（2）管理员功能

返回登录结果：200（匹配成功）、211（匹配失败）

返回数据库信息：“*已收录信息条数\_已收录院校数量\_数据库上次更新时间*”，三个编码即可，如10\_2\_2023.04.15 20:38。

返回反馈列表：“*cookie ID1-feedback ID1-反馈内容1-状态1-回复1\_cookie ID2-feedback ID2-反馈内容2-状态2-回复2*”，\_分隔每条信息，-分隔每条信息内不同类型的内容。

返回处理结果：300（修改成功）、311（修改失败）

备份结果：400（备份成功）、411（备份失败）

五、数据库设计

1、考研信息数据库

参考院校信息文件

2、反馈信息数据库

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| cookie ID | 反馈信息ID | 反馈内容 | 状态 | 回复 |

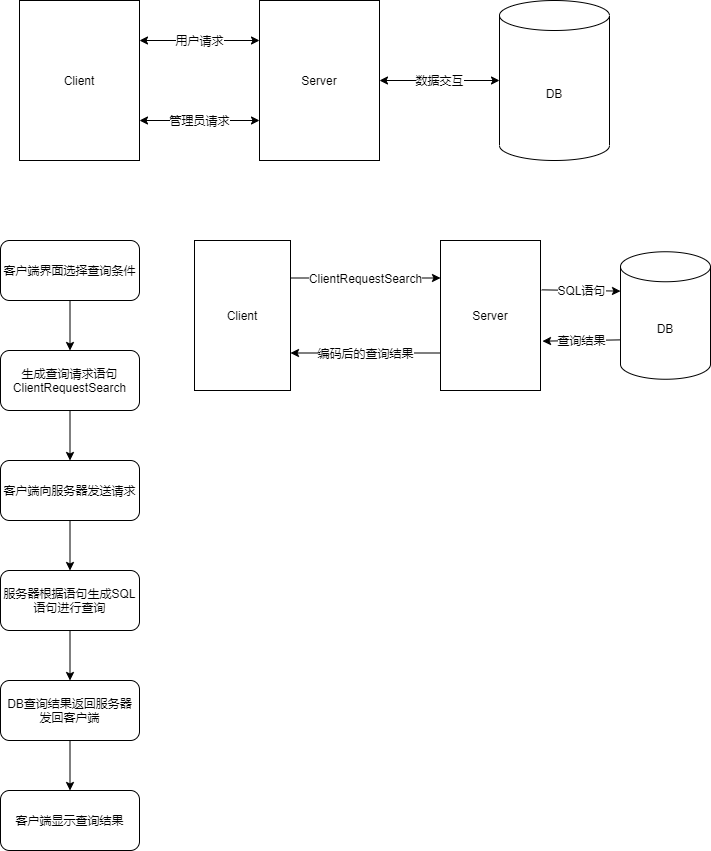
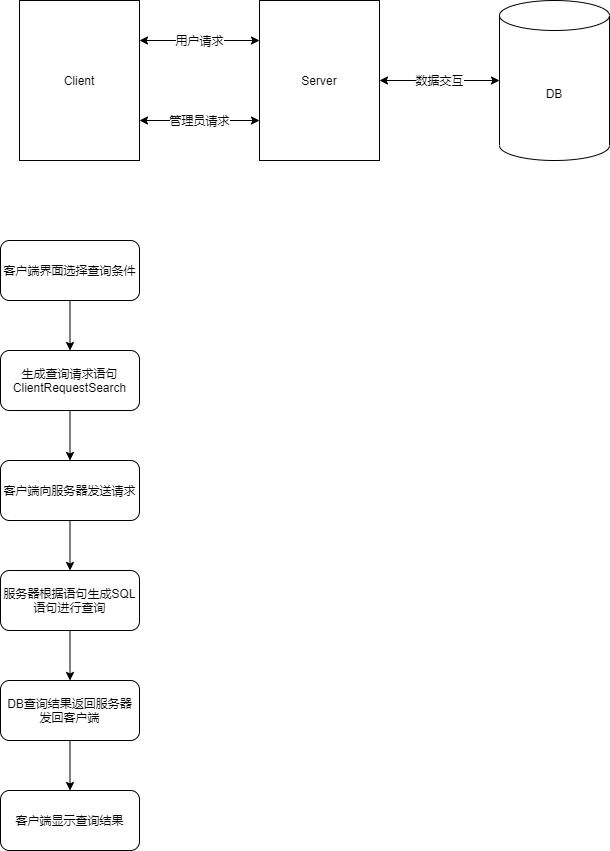
状态包括：已接收，已处理

回复：处于“已接收”状态时回复可以为空（NULL）

反馈信息ID自动生成（编号）。

六、用户功能设计

1、查询



客户端需对查询请求语句进行编码。

**ClientRequestSearch格式：**

*cookie ID\_search\_条件1\_条件2\_条件3*

（条件数量不一定，至少一条）

服务器根据解码后的查询条件生成SQL语句进行查询。

**查询结果格式：**

*院校1-专业-方向-数学-英语-科目一-科目二\_院校2-专业-方向-数学-英语-科目一-科目二*

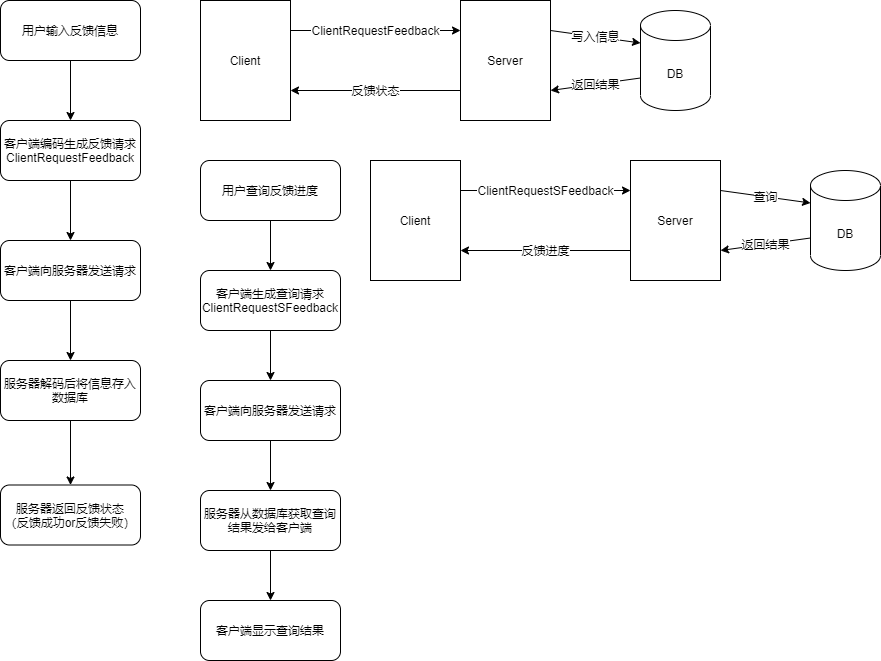
（每条信息以\_分隔，每条信息内的不同信息以-分隔）

查询结果为空时：

院校1-专业-NULL-数学-英语-科目一-科目二

2、反馈

（1）提交反馈信息：



**ClientRequestFeedback格式：**

*cookie ID\_Feedback\_反馈内容*

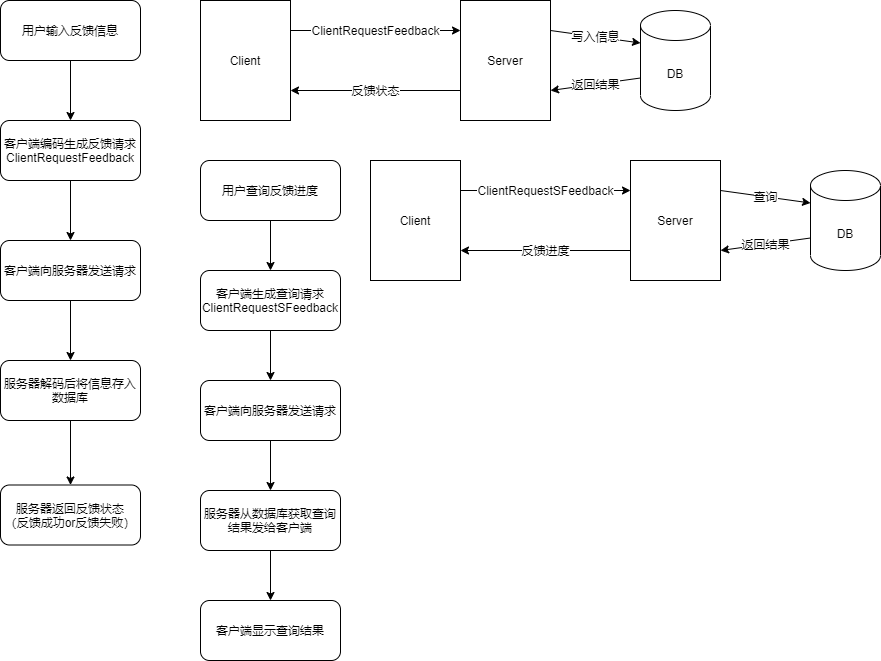
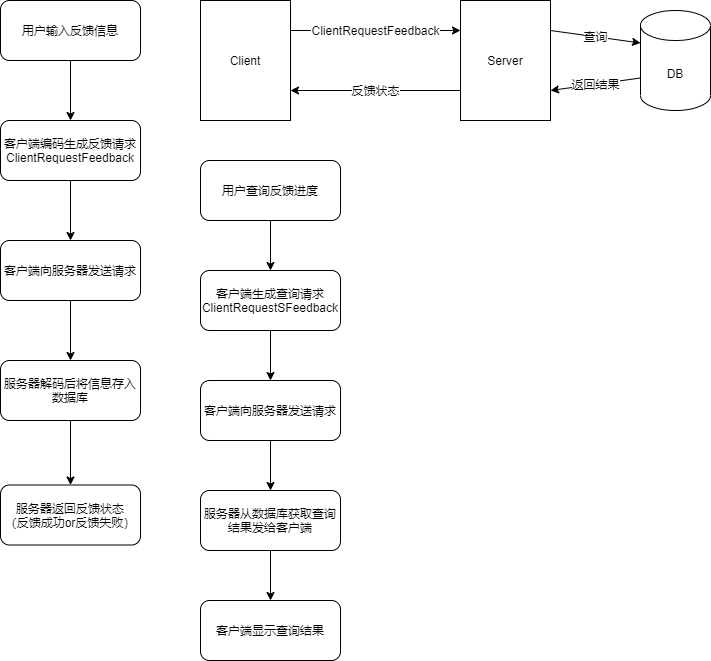
**服务器反馈结果格式：**

100（反馈成功），111（反馈失败）

当反馈操作失败时（主要考虑写入数据库成功与否），返回“反馈失败”；成功时返回“反馈成功”。

客户端界面弹窗显示返回结果。

（2）查询反馈进度：



**ClientRequestSFeedback格式：**

*cookie ID\_fsearch*

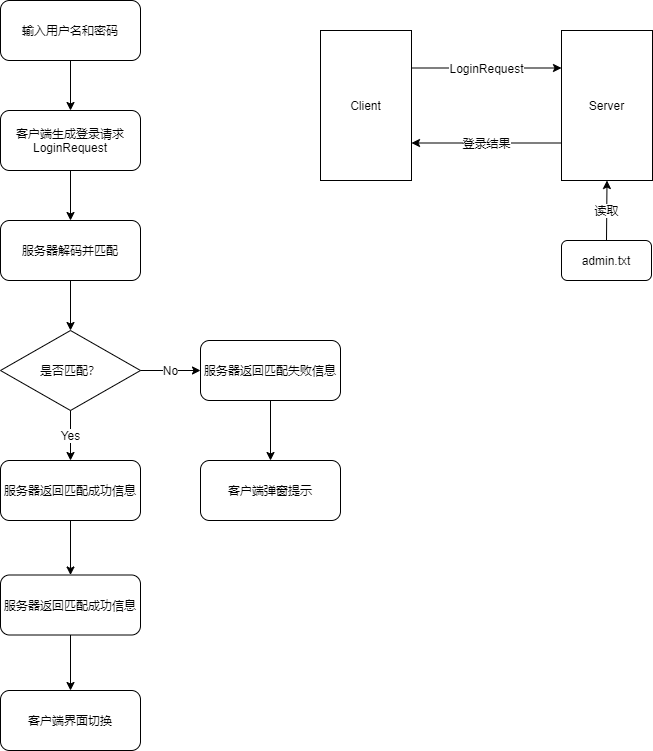
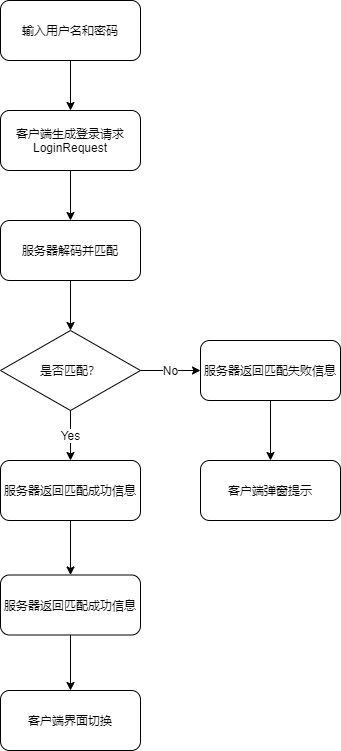
**服务器返回反馈进度格式：**

*反馈内容1-反馈状态1-回复1\_反馈内容2-反馈状态2-回复2*

回复在没有时可以为空（NULL），其他同查询结果。

七、管理员功能设计

1、管理员登录



**LoginRequest格式：**

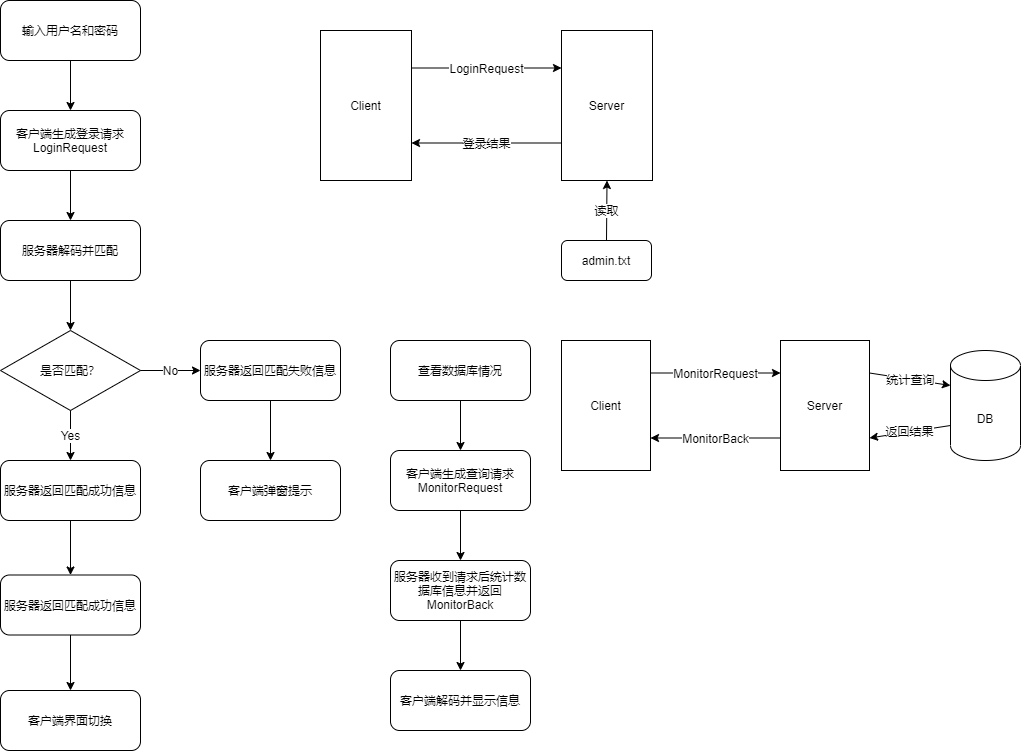
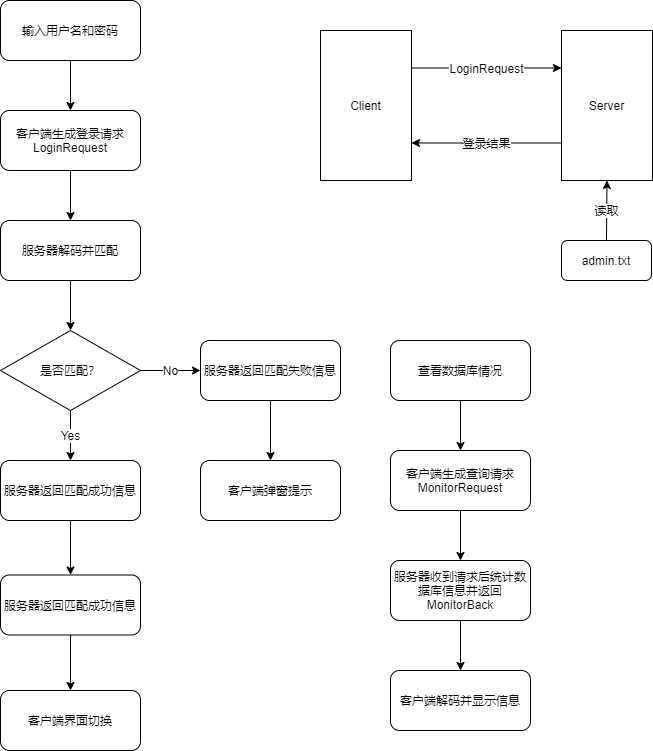
*admin\_login\_name\_password*

**服务器返回匹配结果代码：**

200（匹配成功）、211（匹配失败）

登录失败：客户端弹窗提示；登录成功：客户端切换界面。

2、查看数据库情况



统计信息包括已收录信息条数、已收录院校数量、数据库上次更新时间。

**MonitorRequest格式：**

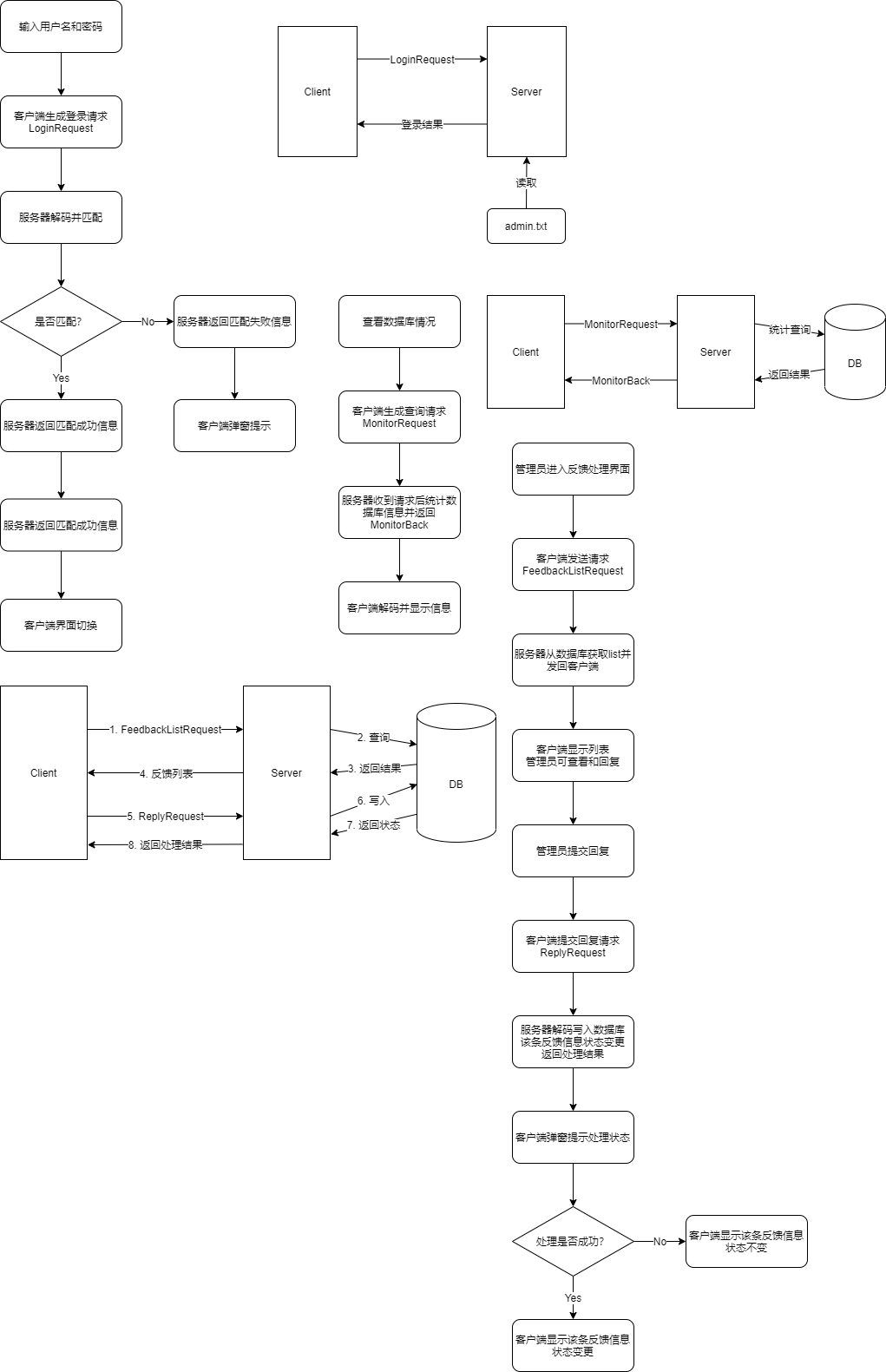
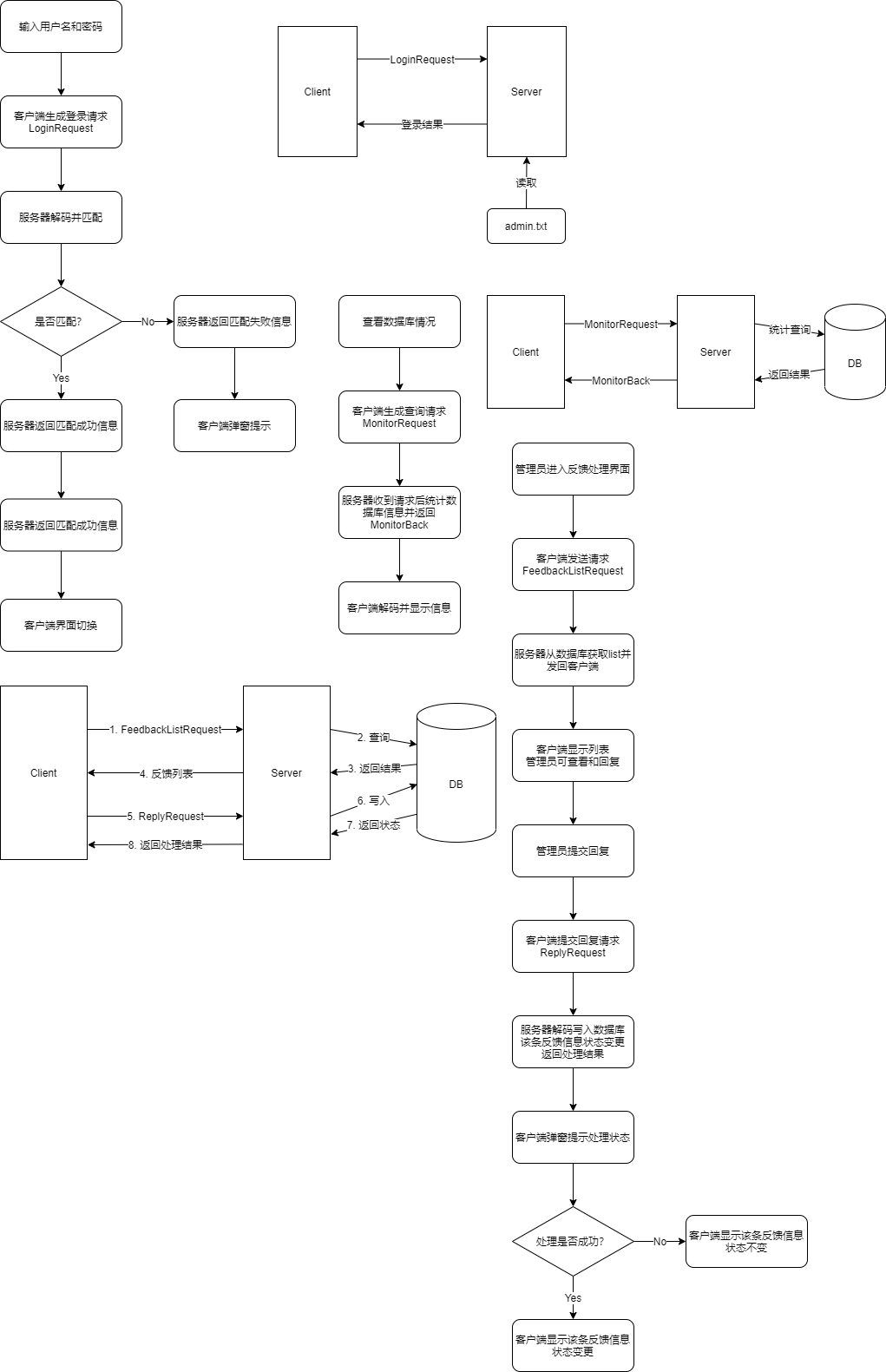
*admin\_monitor*

**MonitorBack格式：**

*已收录信息条数\_已收录院校数量\_数据库上次更新时间*

（三个编码即可，如10\_2\_2023.04.15 20:38）

3、处理反馈



**FeedbackListRequest格式：**

*admin\_feedbacklist*

**服务器反馈列表格式：**

*cookie ID1-feedback ID1-反馈内容1-状态1-回复1\_cookie ID2-feedback ID2-反馈内容2-状态2-回复2*

（\_分隔每条信息，-分隔每条信息内不同类型的内容）

**ReplyRequest格式：**

*admin\_reply\_feedback ID\_回复内容*

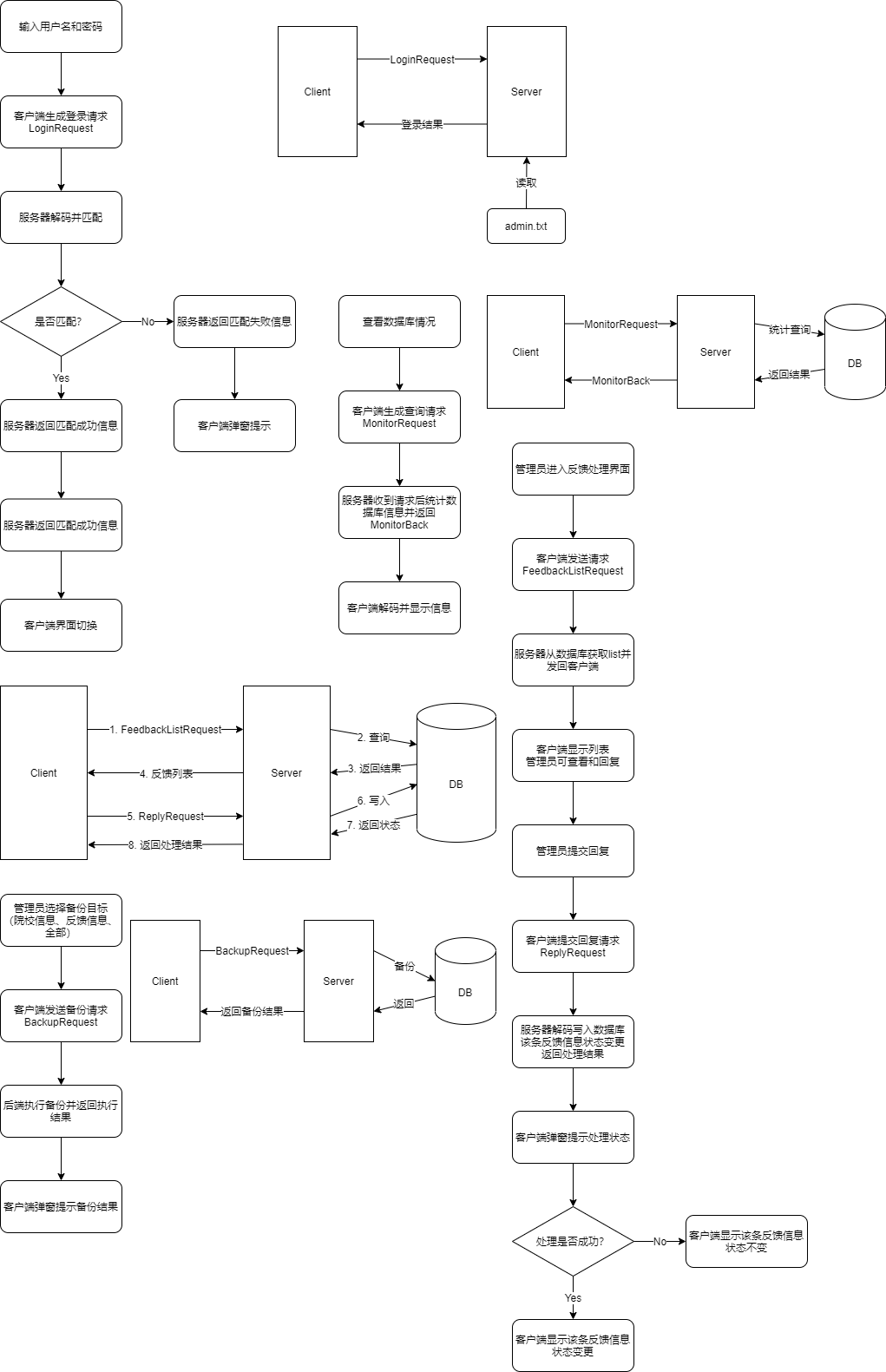
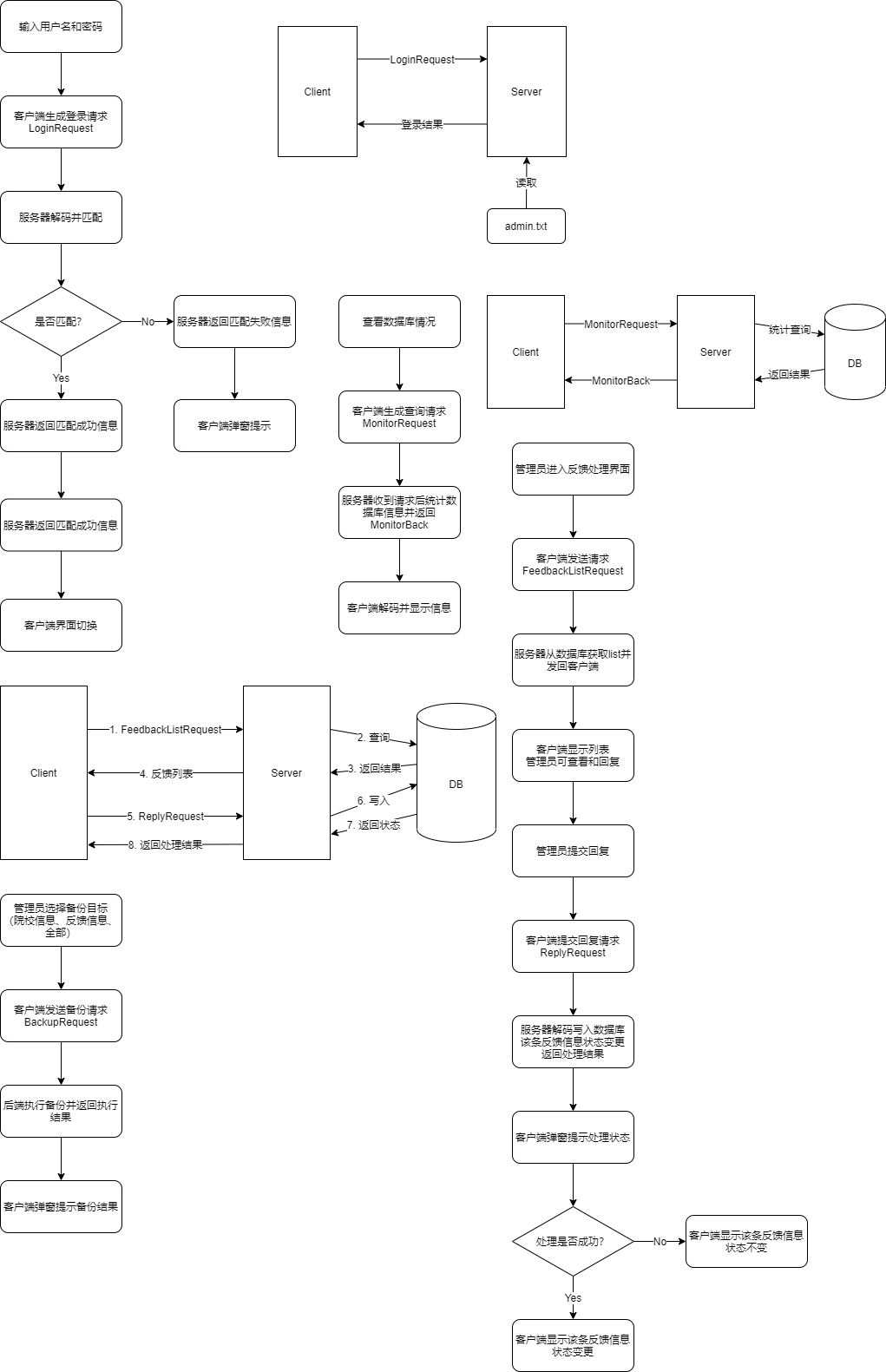
根据feedback ID将回复写入数据库并返回处理结果（状态）。

**服务器返回处理结果：**

300（修改成功）、311（修改失败）

客户端根据服务器返回结果弹窗提示处理结果，并根据返回结果修改客户端显示的内容（feedback状态）

4、备份



备份有三种可能：备份院校信息、备份反馈信息、备份全部。

**BackupRequest格式：**

*admin\_backup\_编号（1-院校信息，2-反馈信息，3-全部）*

**服务器返回备份结果状态：**

400（备份成功）、411（备份失败）

客户端根据服务器返回状态设置弹窗提醒。

5、修改数据库信息

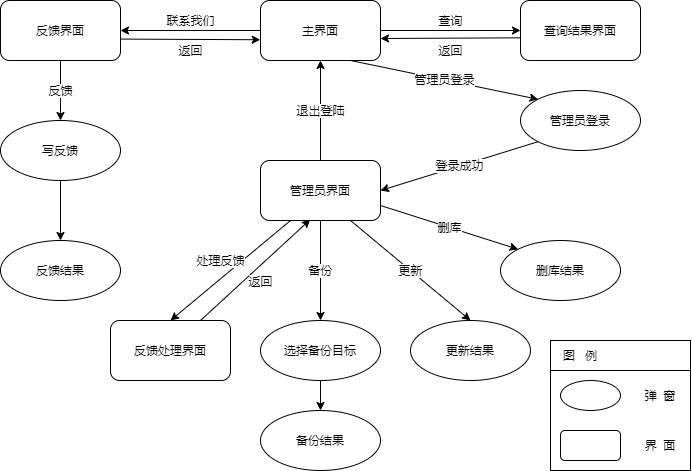
数据库更新，但是考虑到数据量带来的实现难度，所以这里只在前端做个无实意的按钮（雾）。

6、删库跑路

这也做个幌子，前端设计一个按钮，弹窗提示“数据库锁定，无法执行此操作”即可~

八、界面设计

1、框架设计



2、主界面



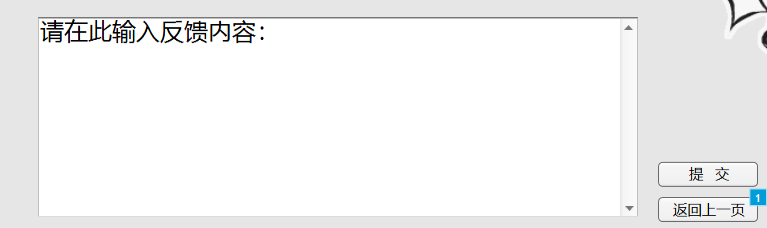
3、查询结果



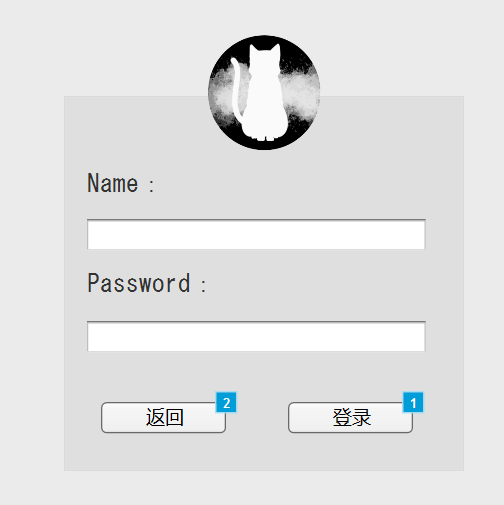
4、反馈界面



5、提交反馈弹窗



6、管理员登录



7、管理员界面



8、反馈处理界面

