# 资产管理系统项目开发文档

<版本：13.0>

## 修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **描述** | **作者** |
| 2020年3月25日 | 1.0 | 初始化 | 陈刚老师 |
| 2020年4月9日 | 14.0 | 申请单双状态助攻资产状态转换  修改位置：   1. [需求列表](#_二、需求列表) 2. [修改申请单类](#_3.申请单类（Application）-_t_application) 3. [类图](#_2._UML类图) 4. √[变更管理第7条](源状态#_（7）、（√）申请单添加两个字段) 5. √[变更管理第8条](预定#_（8）、（√）取消空闲-已使用中间的) 6. [新增变更第16条](#_（16）、（_）统一描述，申请单类型和资产状态一致) | 陈刚老师 |
| 2020年4月11日 | 15.0 | 解决维修操作选项的问题   1. 需求列表-v11 2. 业务建模-6 3. 修改数据结构-3申请单 4. [新增测试项](#_（2）*测试维修操作的结果)-2 5. 新增服务类图-UML类图 6. 新增系统建模-12 7. 新增变更管理项 | 陈刚老师 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

目录

[资产管理系统项目开发文档 1](#_Toc37347033)

[一、需求调研 2](#_Toc37347034)

[二、需求列表 2](#_Toc37347035)

[三、概要设计 14](#_Toc37347036)

[1. HIPO图层次结构图 14](#_Toc37347037)

[2. 访问资源 14](#_Toc37347038)

[四、业务建模 15](#_Toc37347039)

[（1）资产生命周期状态图 15](#_Toc37347040)

[（2）资产领用流程 17](#_Toc37347041)

[（3）同意和拒绝申请单流程 20](#_Toc37347042)

[（4）登录流程 21](#_Toc37347043)

[（5）扫码显示资产页（不同用户显示不同内容） 23](#_Toc37347044)

[五、数据表结构-类结构分析 25](#_Toc37347045)

[（1）数据结构分析 25](#_Toc37347046)

[（2）类图建模 28](#_Toc37347047)

[六、系统建模 33](#_Toc37347048)

[（1）资产管理员可以登记资产（资产入库） 33](#_Toc37347049)

[（2）、资产管理员可以资产出库（分配使用者） 34](#_Toc37347050)

[（3）、资产使用者可以提交领用申请 35](#_Toc37347051)

[（4）、管理员可以查看所有申请 39](#_Toc37347052)

[（5）、管理员可以查看空闲资产 40](#_Toc37347053)

[（6）、管理员可以查看剩余数量 40](#_Toc37347054)

[（7）、资产管理员可以处理申领请求 40](#_Toc37347055)

[（8）、用户登录流程 43](#_Toc37347056)

[（9）、查看资产信息 45](#_Toc37347057)

[（10）、管理员扫码设置资产状态 46](#_Toc37347058)

[（11）、使用者扫码设置提交申请 47](#_Toc37347059)

[七、测试管理 47](#_Toc37347060)

[（1）测试扫码功能-设置资产状态和提交申请 47](#_Toc37347061)

[八、变更管理 49](#_Toc37347062)

[（1）、（√）添加申请单类型字段和修改申请单的入库单id为资产id 49](#_Toc37347063)

[（2）、（√）新增需求 49](#_Toc37347064)

[（3）、（ ）新增判断登录逻辑 50](#_Toc37347065)

[（4）、（√）新增入库单类属性——剩余量 50](#_Toc37347066)

[（5）、（ ）出库时考虑库存量 51](#_Toc37347067)

[（6）、（√）拒绝领用申请时修改资产状态“预定”到“空闲” 51](#_Toc37347068)

[（7）、（ ）申请单添加两个字段“源状态”和“终状态” 53](#_Toc37347069)

[（8）、（ ）取消空闲-已使用中间的“预定”状态 53](#_Toc37347070)

[（9）、（√）查看资产基本信息页网址——访问接口变更 54](#_Toc37347071)

[（10）、（√）给用户类添加“部门”属性 54](#_Toc37347072)

[（11）、（√）修复BUG管理员同意申请资产状态无变化 55](#_Toc37347073)

[（12）、（√）修改doLogin登录逻辑的返回视图 56](#_Toc37347074)

[（13）、（√）新增访问接口/user和/admin 56](#_Toc37347075)

[（14）、（√）更新过滤器规则，添加/user和/admin的拦截 57](#_Toc37347076)

[（15）、（ ）新增用户退出功能 58](#_Toc37347077)

## 一、需求调研

时间：2020年3月24日 星期二

地点：湖北开放职业学院

客户：黄主任

一、系统名称：资产管理系统

二、聊天记录：

主要对学校内的资产进行登记，追踪，随时掌握资产的去向。

需要领取办公用品的可以提交申请，资产管理员同意后才可领取。

主要是统计剩余资产数量和使用资产数量，对资产的使用状况有详细的记载。

支持二维码扫描查看资产信息，支持生成资产的二维码，并且可以打印出来，贴在资产上。

管理员扫描二维码，可以显示对该资产的操作项（如：报废、收回、分配等）

资产的使用者扫描，就能显示维修申请、报废申请等

其他人扫描，就只显示资产基本信息

三、技术选型

资产数量1万件内，属小型，可用MySQL数据库

编程语言 java 语言可实现，使用spring cloud 整合到微服务大平台中。

前端WEB+微信小程序（该功能额外付费）

四、可行性

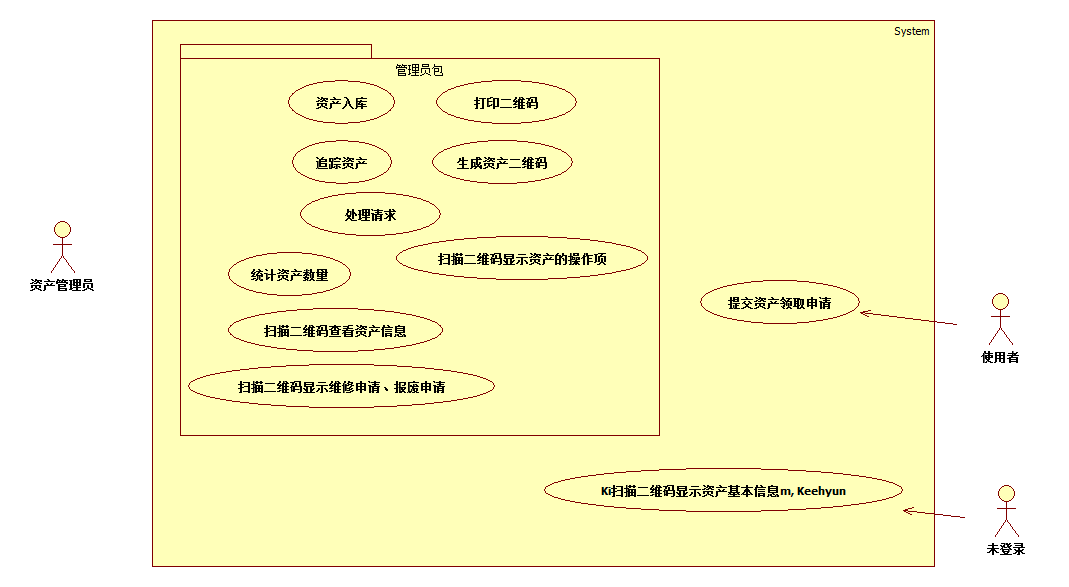
经济可行性：可行（不要钱，纯当练习）；技术可行性：可行（有人帮忙）；法律可行性：可行！

## 二、需求列表

通过需求调研文档，获取角色和用例

|  |  |
| --- | --- |
| v11.0备课  2020年4月10日 | 角色：资产管理员、资产使用者、未登录的其他人  颜色含义：红色表示已完成、绿色表示新增需求、蓝色表示正在处理  正在处理的需求  维修人员可以点击“维修成功”自动创建申请单  维修人员可以点击“维修失败”自动创建申请单  管理员可以显示申请单的操作项。  思考资产使用者在资产状态转换过程中应该有怎么样的变化？  已完成列表：   1. 资产管理员可以登记资产（资产入库）[已完成] 2. 管理员可以资产出库[完成] 3. 资产使用者可以提交资产领取申请[完成] 4. 资产管理员可以查看所有请求[新增，已完成] 5. 资产管理员可以查询指定入库单的空闲资产[新增，已完成] 6. 资产管理员可以查询库存剩余数量[新增，已完成] 7. 资产管理员可以查看同一个资产的申请单(检测冲突) 8. 管理员可以查看申领的申请单 9. 管理员可以查看维修的申请单 10. 管理员可以查看借用的申请单 11. 管理员可以查看领用申请的人 12. 资产管理员可以统计资产数量（剩余资产数量、当前使用的资产数量、资产使用记录） 13. 资产管理员可以统计当前正在使用的资产数量 14. 资产管理员可以统计正在维修的资产数量 15. 资产管理员可以生成资产二维码 16. 资产管理员可以扫描二维码查看资产信息 17. 其他未登录的人扫描二维码显示资产基本信息 18. 管理员可以登录后台 19. 使用者可以登录后台 20. 初始化测试数据 21. 权限认证过滤器 22. 资产管理员扫描二维码显示资产的操作项（报废、收回、分配）     1. 资产管理员扫描二维码显示资产完整信息     2. 资产管理员扫描二维码显示报废（转换为报废）操作项     3. 资产管理员扫描二维码显示收回（转换为空闲）操作项     4. 资产管理员扫描二维码显示分配（指定使用者）操作项 23. 资产使用者扫描二维码显示维修申请、报废申请     1. 资产使用者扫描二维码显示领用申请     2. 资产使用者扫描二维码显示维修申请     3. 资产使用者扫描二维码显示报废申请 24. 快速设置资产报废（转换为报废）操作 25. 快速设置资产收回（转换为空闲）操作 26. 快速设置资产维修（转换为维修）操作 27. 快速设置资产分配（转换为已使用）操作 28. 快速提交领用申请 29. 快速提交维修申请 30. 快速提交报废申请 31. 资产管理员可以处理请求（同意或拒绝、维修请求、报废请求）     1. 资产管理员可以处理借用请求【延后处理】     2. 资产管理员可以处理归还请求【延后处理】     3. 管理员可以直接修改资产状态     4. 资产管理员可以处理申领请求[完成同意、完成拒绝]     5. 资产管理员可以处理维护请求[完成同意、完成拒绝]     6. 资产管理员可以处理报废请求[完成]   未完成需求   1. 资产管理员可以追踪资产    1. 资产管理员可以追踪资产使用者变更记录    2. 资产管理员可以追踪资产维修记录 2. 资产管理员可以追踪资产当前位置（资产去向） |
| v10.0备课  2020年4月9日 | 角色：资产管理员、资产使用者、未登录的其他人  颜色含义：红色表示已完成、绿色表示新增需求、蓝色表示正在处理  正在处理的需求   1. 资产管理员可以处理请求（同意或拒绝、维修请求、报废请求）    1. 资产管理员可以处理借用请求【延后处理】    2. 资产管理员可以处理归还请求【延后处理】    3. 管理员可以直接修改资产状态    4. 资产管理员可以处理申领请求[完成同意、完成拒绝]    5. 资产管理员可以处理维护请求[完成同意、完成拒绝]    6. 资产管理员可以处理报废请求[完成]   已完成列表：   1. 资产管理员可以登记资产（资产入库）[已完成] 2. 管理员可以资产出库[完成] 3. 资产使用者可以提交资产领取申请[完成] 4. 资产管理员可以查看所有请求[新增，已完成] 5. 资产管理员可以查询指定入库单的空闲资产[新增，已完成] 6. 资产管理员可以查询库存剩余数量[新增，已完成] 7. 资产管理员可以查看同一个资产的申请单(检测冲突) 8. 管理员可以查看申领的申请单 9. 管理员可以查看维修的申请单 10. 管理员可以查看借用的申请单 11. 管理员可以查看领用申请的人 12. 资产管理员可以统计资产数量（剩余资产数量、当前使用的资产数量、资产使用记录） 13. 资产管理员可以统计当前正在使用的资产数量 14. 资产管理员可以统计正在维修的资产数量 15. 资产管理员可以生成资产二维码 16. 资产管理员可以扫描二维码查看资产信息 17. 其他未登录的人扫描二维码显示资产基本信息 18. 管理员可以登录后台 19. 使用者可以登录后台 20. 初始化测试数据 21. 权限认证过滤器 22. 资产管理员扫描二维码显示资产的操作项（报废、收回、分配）     1. 资产管理员扫描二维码显示资产完整信息     2. 资产管理员扫描二维码显示报废（转换为报废）操作项     3. 资产管理员扫描二维码显示收回（转换为空闲）操作项     4. 资产管理员扫描二维码显示分配（指定使用者）操作项 23. 资产使用者扫描二维码显示维修申请、报废申请     1. 资产使用者扫描二维码显示领用申请     2. 资产使用者扫描二维码显示维修申请     3. 资产使用者扫描二维码显示报废申请 24. 快速设置资产报废（转换为报废）操作 25. 快速设置资产收回（转换为空闲）操作 26. 快速设置资产维修（转换为维修）操作 27. 快速设置资产分配（转换为已使用）操作 28. 快速提交领用申请 29. 快速提交维修申请 30. 快速提交报废申请   未完成需求   1. 资产管理员可以追踪资产    1. 资产管理员可以追踪资产使用者变更记录    2. 资产管理员可以追踪资产维修记录 2. 资产管理员可以追踪资产当前位置（资产去向） |
| v9.0备课  2020年4月8日 | 角色：资产管理员、资产使用者、未登录的其他人  颜色含义：红色表示已完成、绿色表示新增需求、蓝色表示正在处理  正在处理的需求   1. 快速设置资产报废（转换为报废）操作 2. 快速设置资产收回（转换为空闲）操作 3. 快速设置资产维修（转换为维修）操作 4. 快速设置资产分配（转换为已使用）操作 5. 快速提交领用申请 6. 快速提交维修申请 7. 快速提交报废申请   已完成列表：   1. 资产管理员可以登记资产（资产入库）[已完成] 2. 管理员可以资产出库[完成] 3. 资产使用者可以提交资产领取申请[完成] 4. 资产管理员可以查看所有请求[新增，已完成] 5. 资产管理员可以查询指定入库单的空闲资产[新增，已完成] 6. 资产管理员可以查询库存剩余数量[新增，已完成] 7. 资产管理员可以处理申领请求[完成同意、完成拒绝] 8. 资产管理员可以处理维护请求[完成同意、完成拒绝] 9. 资产管理员可以处理报废请求[完成] 10. 资产管理员可以查看同一个资产的申请单(检测冲突) 11. 管理员可以查看申领的申请单 12. 管理员可以查看维修的申请单 13. 管理员可以查看借用的申请单 14. 管理员可以查看领用申请的人 15. 资产管理员可以统计资产数量（剩余资产数量、当前使用的资产数量、资产使用记录） 16. 资产管理员可以统计当前正在使用的资产数量 17. 资产管理员可以统计正在维修的资产数量 18. 资产管理员可以生成资产二维码 19. 资产管理员可以扫描二维码查看资产信息 20. 其他未登录的人扫描二维码显示资产基本信息 21. 管理员可以登录后台 22. 使用者可以登录后台 23. 初始化测试数据 24. 权限认证过滤器 25. 资产管理员扫描二维码显示资产的操作项（报废、收回、分配）     1. 资产管理员扫描二维码显示资产完整信息     2. 资产管理员扫描二维码显示报废（转换为报废）操作项     3. 资产管理员扫描二维码显示收回（转换为空闲）操作项     4. 资产管理员扫描二维码显示分配（指定使用者）操作项 26. 资产使用者扫描二维码显示维修申请、报废申请     1. 资产使用者扫描二维码显示领用申请     2. 资产使用者扫描二维码显示维修申请     3. 资产使用者扫描二维码显示报废申请   未完成需求   1. 资产管理员可以追踪资产    1. 资产管理员可以追踪资产使用者变更记录    2. 资产管理员可以追踪资产维修记录 2. 资产管理员可以处理请求（同意或拒绝、维修请求、报废请求）    1. 资产管理员可以处理借用请求【延后处理】    2. 资产管理员可以处理归还请求【延后处理】    3. 管理员可以直接修改资产状态 3. 资产管理员可以追踪资产当前位置（资产去向） |
| v8.0备课  2020年4月7日 | 角色：资产管理员、资产使用者、未登录的其他人  颜色含义：红色表示已完成、绿色表示新增需求、蓝色表示正在处理  正在处理的需求   1. 资产管理员扫描二维码显示资产的操作项（报废、收回、分配）    1. 资产管理员扫描二维码显示资产完整信息    2. 资产管理员扫描二维码显示报废（转换为报废）操作项    3. 资产管理员扫描二维码显示收回（转换为空闲）操作项    4. 资产管理员扫描二维码显示分配（指定使用者）操作项 2. 资产使用者扫描二维码显示维修申请、报废申请    1. 资产使用者扫描二维码显示领用申请    2. 资产使用者扫描二维码显示维修申请    3. 资产使用者扫描二维码显示报废申请 3. 资产管理员可以追踪资产当前位置（资产去向）   已完成列表：   1. 资产管理员可以登记资产（资产入库）[已完成] 2. 管理员可以资产出库[完成] 3. 资产使用者可以提交资产领取申请[完成] 4. 资产管理员可以查看所有请求[新增，已完成] 5. 资产管理员可以查询指定入库单的空闲资产[新增，已完成] 6. 资产管理员可以查询库存剩余数量[新增，已完成] 7. 资产管理员可以处理申领请求[完成同意、完成拒绝] 8. 资产管理员可以处理维护请求[完成同意、完成拒绝] 9. 资产管理员可以处理报废请求[完成] 10. 资产管理员可以查看同一个资产的申请单(检测冲突) 11. 管理员可以查看申领的申请单 12. 管理员可以查看维修的申请单 13. 管理员可以查看借用的申请单 14. 管理员可以查看领用申请的人 15. 资产管理员可以统计资产数量（剩余资产数量、当前使用的资产数量、资产使用记录） 16. 资产管理员可以统计当前正在使用的资产数量 17. 资产管理员可以统计正在维修的资产数量 18. 资产管理员可以生成资产二维码 19. 资产管理员可以扫描二维码查看资产信息 20. 其他未登录的人扫描二维码显示资产基本信息 21. 管理员可以登录后台 22. 使用者可以登录后台 23. 初始化测试数据 24. 权限认证过滤器   未完成需求   1. 资产管理员可以追踪资产    1. 资产管理员可以追踪资产使用者变更记录    2. 资产管理员可以追踪资产维修记录 2. 资产管理员可以处理请求（同意或拒绝、维修请求、报废请求）    1. 资产管理员可以处理借用请求【延后处理】    2. 资产管理员可以处理归还请求【延后处理】    3. 管理员可以直接修改资产状态 |
| v7.0备课 | 角色：资产管理员、资产使用者、未登录的其他人  颜色含义：红色表示已完成、绿色表示新增需求、蓝色表示正在处理  正在处理的需求  管理员可以登录后台  使用者可以登录后台  初始化测试数据  权限认证过滤器  **已完成列表**：   1. 资产管理员可以登记资产（资产入库）[已完成] 2. 管理员可以资产出库[完成] 3. 资产使用者可以提交资产领取申请[完成] 4. 资产管理员可以查看所有请求[新增，已完成] 5. 资产管理员可以查询指定入库单的空闲资产[新增，已完成] 6. 资产管理员可以查询库存剩余数量[新增，已完成] 7. 资产管理员可以处理申领请求[完成同意、完成拒绝] 8. 资产管理员可以处理维护请求[完成同意、完成拒绝] 9. 资产管理员可以处理报废请求[完成] 10. 资产管理员可以查看同一个资产的申请单(检测冲突) 11. 管理员可以查看申领的申请单 12. 管理员可以查看维修的申请单 13. 管理员可以查看借用的申请单 14. 管理员可以查看领用申请的人 15. 资产管理员可以统计资产数量（剩余资产数量、当前使用的资产数量、资产使用记录）     1. 资产管理员可以统计当前正在使用的资产数量     2. 资产管理员可以统计正在维修的资产数量 16. 资产管理员可以生成资产二维码     1. 资产管理员可以扫描二维码查看资产信息   其他未登录的人扫描二维码显示资产基本信息  未完成需求   1. 资产管理员可以追踪资产    1. 资产管理员可以追踪资产使用者变更记录    2. 资产管理员可以追踪资产维修记录    3. 资产管理员可以追踪资产当前位置（资产去向）    4. ~~资产管理员可以追踪资产当前状态~~    5. 资产管理员扫描二维码显示资产的操作项（报废、收回、分配） 2. 资产使用者扫描二维码显示维修申请、报废申请 3. ~~资产管理员可以打印二维码~~ 4. 资产管理员可以处理请求（同意或拒绝、维修请求、报废请求） 5. **资产管理员可以处理借用请求【延后处理】** 6. **资产管理员可以处理归还请求【延后处理】** 7. **管理员可以直接修改资产状态** |
| v6.0备课 | 角色：资产管理员、资产使用者、未登录的其他人  颜色含义：红色表示已完成、绿色表示新增需求、蓝色表示正在处理  正在处理的需求   1. **资产管理员可以查看同一个资产的申请单(检测冲突)** 2. **管理员可以查看申领的申请单** 3. **管理员可以查看维修的申请单** 4. **管理员可以查看借用的申请单** 5. **管理员可以查看领用申请的人** 6. 资产管理员可以统计资产数量（剩余资产数量、当前使用的资产数量、资产使用记录）    1. 资产管理员可以统计当前正在使用的资产数量    2. 资产管理员可以统计正在维修的资产数量 7. 资产管理员可以生成资产二维码    1. 资产管理员可以扫描二维码查看资产信息   **已完成列表**：   1. 资产管理员可以登记资产（资产入库）[已完成] 2. 管理员可以资产出库[完成] 3. 资产使用者可以提交资产领取申请[完成]    1. 资产管理员可以查看所有请求[新增，已完成]    2. 资产管理员可以查询指定入库单的空闲资产[新增，已完成]    3. 资产管理员可以查询库存剩余数量[新增，已完成]    4. 资产管理员可以处理申领请求[完成同意、完成拒绝]    5. 资产管理员可以处理维护请求[完成同意、完成拒绝]    6. 资产管理员可以处理报废请求[完成]   未完成需求   1. 资产管理员可以追踪资产    1. 资产管理员可以追踪资产使用者变更记录    2. 资产管理员可以追踪资产维修记录    3. 资产管理员可以追踪资产当前位置（资产去向）    4. 资产管理员可以追踪资产当前状态    5. 资产管理员可以扫描二维码查看资产信息    6. 资产管理员扫描二维码显示资产的操作项（报废、收回、分配） 2. 资产使用者扫描二维码显示维修申请、报废申请 3. 资产管理员可以打印二维码   其他未登录的人扫描二维码显示资产基本信息   1. 资产管理员可以处理请求（同意或拒绝、维修请求、报废请求） 2. **资产管理员可以处理借用请求【延后处理】** 3. **资产管理员可以处理归还请求【延后处理】** 4. **管理员可以直接修改资产状态** |
| v5.0备课 | 角色：资产管理员、资产使用者、未登录的其他人  颜色含义：红色表示已完成、绿色表示新增需求、蓝色表示正在处理  正在处理的需求   * 1. 资产管理员可以处理申领请求[完成同意、未完成拒绝]   2. 资产管理员可以处理维护请求[完成同意、未完成拒绝]  1. **资产管理员可以处理报废请求[正在处理]** 2. **资产管理员可以处理借用请求** 3. **资产管理员可以处理归还请求** 4. **资产管理员可以查看同一个资产的申请单(检测冲突)**   **已完成列表**：   1. 资产管理员可以登记资产（资产入库）[已完成] 2. 管理员可以资产出库[完成] 3. 资产使用者可以提交资产领取申请[完成]    1. 资产管理员可以查看所有请求[新增，已完成]    2. 资产管理员可以查询指定入库单的空闲资产[新增，已完成]    3. 资产管理员可以查询库存剩余数量[新增，已完成]   **新增需求**   1. **管理员可以查看申领的申请单** 2. **管理员可以查看维修的申请单** 3. **管理员可以查看借用的申请单** 4. **管理员可以查看领用申请的人** 5. **管理员可以直接修改资产状态**   未完成需求   1. 资产管理员可以追踪资产    1. 资产管理员可以追踪资产使用者变更记录    2. 资产管理员可以追踪资产维修记录    3. 资产管理员可以追踪资产当前位置（资产去向）    4. 资产管理员可以追踪资产当前状态 2. 资产管理员可以处理请求（同意或拒绝、维修请求、报废请求） 3. 资产管理员可以统计资产数量（剩余资产数量、当前使用的资产数量、资产使用记录）    1. 资产管理员可以统计库存剩余数量    2. 资产管理员可以统计当前正在使用的资产数量    3. 资产管理员可以统计正在维修的资产数量 4. 资产管理员可以扫描二维码    1. 资产管理员可以扫描二维码查看资产信息    2. 资产管理员扫描二维码显示资产的操作项（报废、收回、分配） 5. 资产使用者扫描二维码显示维修申请、报废申请 6. 资产管理员可以生成资产二维码 7. 资产管理员可以打印二维码   其他未登录的人扫描二维码显示资产基本信息 |
| v4.0课堂笔记 | 角色：资产管理员、资产使用者、未登录的其他人  颜色含义：红色表示已完成、绿色表示新增需求、蓝色表示正在处理  需求完整列表：   1. 资产管理员可以登记资产（资产入库）[已完成] 2. 管理员可以资产出库[完成一半] 3. 资产使用者可以提交资产领取申请[完成-制作UML图形] 4. 资产管理员可以追踪资产    1. 资产管理员可以追踪资产使用者变更记录    2. 资产管理员可以追踪资产维修记录    3. 资产管理员可以追踪资产当前位置（资产去向）    4. 资产管理员可以追踪资产当前状态 5. 资产管理员可以处理请求（同意或拒绝、维修请求、报废请求）    1. 资产管理员可以查看所有请求[新增，已完成]    2. 资产管理员可以查询指定入库单的空闲资产[新增，已完成]    3. 资产管理员可以查询库存剩余数量[新增，已完成]    4. **资产管理员可以处理申领请求[正在处理]**    5. **资产管理员可以处理维护请求[正在处理]**    6. **资产管理员可以处理报废求[正在处理]**    7. **资产管理员可以处理借用请求**    8. **资产管理员可以处理归还请求**    9. **资产管理员可以查看同一个资产的申请单(检测冲突)**    10. **查看申领的申请单**    11. **查看维修的申请单**    12. **查看借用的申请单**    13. **查看领用申请的人** 6. 资产管理员可以统计资产数量（剩余资产数量、当前使用的资产数量、资产使用记录）    1. 资产管理员可以统计库存剩余数量    2. 资产管理员可以统计当前正在使用的资产数量    3. 资产管理员可以统计正在维修的资产数量 7. 资产管理员可以扫描二维码    1. 资产管理员可以扫描二维码查看资产信息    2. 资产管理员扫描二维码显示资产的操作项（报废、收回、分配） 8. 资产使用者扫描二维码显示维修申请、报废申请 9. 资产管理员可以生成资产二维码 10. 资产管理员可以打印二维码   其他未登录的人扫描二维码显示资产基本信息 |
| v4.0备课 | 角色：资产管理员、资产使用者、未登录的其他人  颜色含义：红色表示已完成、绿色表示新增需求、蓝色表示正在处理  当前正在处理的需求：  5.  b) 资产管理员可以处理申领请求[正在处理]  c) 资产管理员可以处理维护请求[正在处理]  d) 资产管理员可以处理报废求[正在处理]  细化之后：  管理员可以同意领用申请  管理员可以同意维修申请  管理员可以同意报废申请  管理员可以同意借用申请  管理员可以同意归还申请  管理员可以拒绝领用申请  管理员可以拒绝维修申请  管理员可以拒绝报废申请  管理员可以拒绝借用申请  管理员可以拒绝归还申请  需求完整列表：   1. 资产管理员可以登记资产（资产入库）[已完成] 2. 管理员可以资产出库[完成一半] 3. 资产使用者可以提交资产领取申请[完成-制作UML图形] 4. 资产管理员可以追踪资产    1. 资产管理员可以追踪资产使用者变更记录    2. 资产管理员可以追踪资产维修记录    3. 资产管理员可以追踪资产当前位置（资产去向）    4. 资产管理员可以追踪资产当前状态 5. 资产管理员可以处理请求（同意或拒绝、维修请求、报废请求）    1. 资产管理员可以查看所有请求[新增，已完成]    2. 资产管理员可以查询指定入库单的空闲资产    3. 资产管理员可以查询库存剩余数量    4. **资产管理员可以处理申领请求[正在处理]**    5. **资产管理员可以处理维护请求[正在处理]**    6. **资产管理员可以处理报废求[正在处理]**    7. **资产管理员可以处理借用请求**    8. **资产管理员可以处理归还请求** 6. 资产管理员可以统计资产数量（剩余资产数量、当前使用的资产数量、资产使用记录）    1. 资产管理员可以统计库存剩余数量    2. 资产管理员可以统计当前正在使用的资产数量    3. 资产管理员可以统计正在维修的资产数量 7. 资产管理员可以扫描二维码    1. 资产管理员可以扫描二维码查看资产信息    2. 资产管理员扫描二维码显示资产的操作项（报废、收回、分配） 8. 资产使用者扫描二维码显示维修申请、报废申请 9. 资产管理员可以生成资产二维码 10. 资产管理员可以打印二维码 11. 其他未登录的人扫描二维码显示资产基本信息 |
| v3.0课堂笔记  2020年3月30日 | 角色：资产管理员、资产使用者、未登录的其他人   1. 资产管理员可以登记资产（资产入库）[已完成] 2. 管理员可以资产出库[完成一半] 3. 资产使用者可以提交资产领取申请[完成-制作UML图形] 4. 资产管理员可以追踪资产    1. 资产管理员可以追踪资产使用者变更记录    2. 资产管理员可以追踪资产维修记录    3. 资产管理员可以追踪资产当前位置（资产去向）    4. 资产管理员可以追踪资产当前状态 5. 资产管理员可以处理请求（同意或拒绝、维修请求、报废请求）    1. 资产管理员可以查看所有请求[新增，已完成]    2. 资产管理员可以处理申领请求    3. 资产管理员可以处理维护请求    4. 资产管理员可以处理报废请求 6. 资产管理员可以统计资产数量（剩余资产数量、当前使用的资产数量、资产使用记录）    1. 资产管理员可以统计库存剩余数量    2. 资产管理员可以统计当前正在使用的资产数量    3. 资产管理员可以统计正在维修的资产数量 7. 资产管理员可以扫描二维码    1. 资产管理员可以扫描二维码查看资产信息    2. 资产管理员扫描二维码显示资产的操作项（报废、收回、分配） 8. 资产使用者扫描二维码显示维修申请、报废申请 9. 资产管理员可以生成资产二维码 10. 资产管理员可以打印二维码 11. 其他未登录的人扫描二维码显示资产基本信息 |
| v2.0 | 角色：资产管理员、资产使用者、未登录的其他人  1. 资产管理员可以登记资产（资产入库）  2. 资产管理员可以追踪资产  a) 资产管理员可以追踪资产使用者变更记录  b) 资产管理员可以追踪资产维修记录  c) 资产管理员可以追踪资产当前位置（资产去向）  d) 资产管理员可以追踪资产当前状态  3. 资产管理员可以处理请求（同意或拒绝、维修请求、报废请求）  a) 资产管理员可以处理申领请求  b) 资产管理员可以处理维护请求  c) 资产管理员可以处理报废请求  4. 资产管理员可以统计资产数量（剩余资产数量、当前使用的资产数量、资产使用记录）  a) 资产管理员可以统计库存剩余数量  b) 资产管理员可以统计当前正在使用的资产数量  c) 资产管理员可以统计正在维修的资产数量  5. 资产管理员可以扫描二维码  a) 资产管理员可以扫描二维码查看资产信息  b) 资产管理员扫描二维码显示资产的操作项（报废、收回、分配）  6. 资产使用者扫描二维码显示维修申请、报废申请  7. 资产管理员可以生成资产二维码  8. 资产管理员可以打印二维码  9. 其他未登录的人扫描二维码显示资产基本信息  10. 资产使用者可以提交资产领取申请  **11. 管理员可以资产出库** |
| v1.0 | 角色：资产管理员、资产使用者、未登录的其他人   1. 资产管理员可以登记资产（资产入库） 2. 资产管理员可以追踪资产 3. 资产管理员可以处理请求（同意或拒绝、维修请求、报废请求、） 4. 资产管理员可以统计资产数量（剩余资产数量、当前使用的资产数量、资产使用记录） 5. 资产管理员可以扫描二维码查看资产信息 6. 资产管理员可以生成资产二维码 7. 资产管理员可以打印二维码 8. 资产管理员扫描二维码显示资产的操作项（报废、收回、分配） 9. 资产使用者扫描二维码显示维修申请、报废申请 10. 其他未登录的人扫描二维码显示资产基本信息   资产使用者可以提交资产领取申请 |

用例图



## 三、概要设计

### 1. HIPO图层次结构图

资产管理系统

首页控制器

申请单控制器

添加资产

资产出库

。。。

提交领用申请

拒绝领用申请

资产控制器

入库单控制器

### 2. 访问资源

新增访问资源 2020年4月9日

授权访问

|  |  |
| --- | --- |
| 资源地址 | 备注 |
| /user | 使用者后台，必须经过登录后才可以进入 |
| /admin | 管理员后台，必须经过登录后才可以进入 |

授权访问

|  |  |
| --- | --- |
| 资源地址 | 备注 |
| /application/\* | 申请单的相关功能（添加、修改、删除、查看等） |
| /asset/\* | 资产的相关功能（出库、修改资产状态、查看资产状态等） |
| /rukudan/\* | 入库单的相关功能（入库、查看入库单等） |

非授权访问（未登录用户可访问）

|  |  |
| --- | --- |
| 资源地址 | 备注 |
| /login/\* | 登录相关功能（登录页、登录检查、退出登录） |
| /index | 首页 |
| /assetById | 查看资产页 备注：已经执行变更管理第9条 |

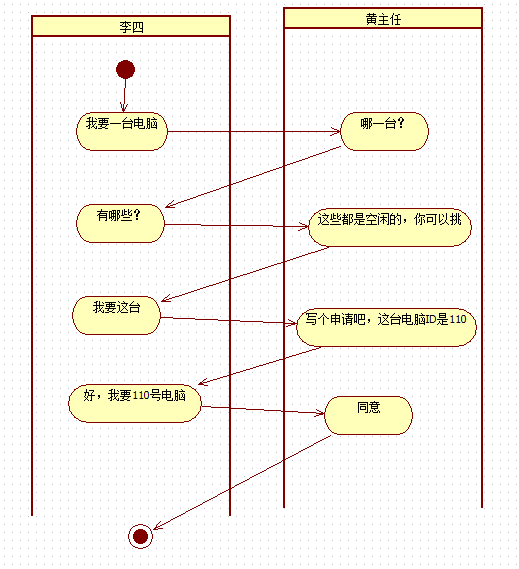
## 四、业务建模

### （1）资产生命周期状态图

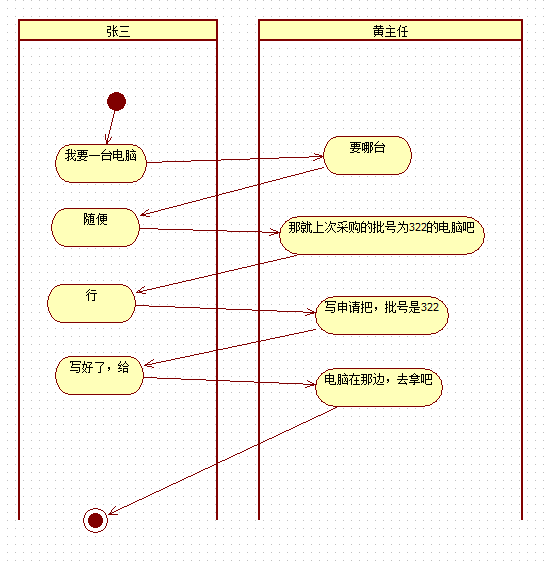
|  |  |
| --- | --- |
| v4.0  取消预订状态  取消管理员直接设置状态 |  |
| v3.0  新增预定状态 |  |
| v2.0  状态转换是有限制条件的 |  |
| v1.0 |  |

### （2）资产领用流程

#### 情况一、从入库单中领取资产



#### 情况二、从空闲里面领取资产



#### 情况三、由系统自动决定从哪里获得资产



#### 情况三、改版



### （3）同意和拒绝申请单流程

#### 拒绝领用申请



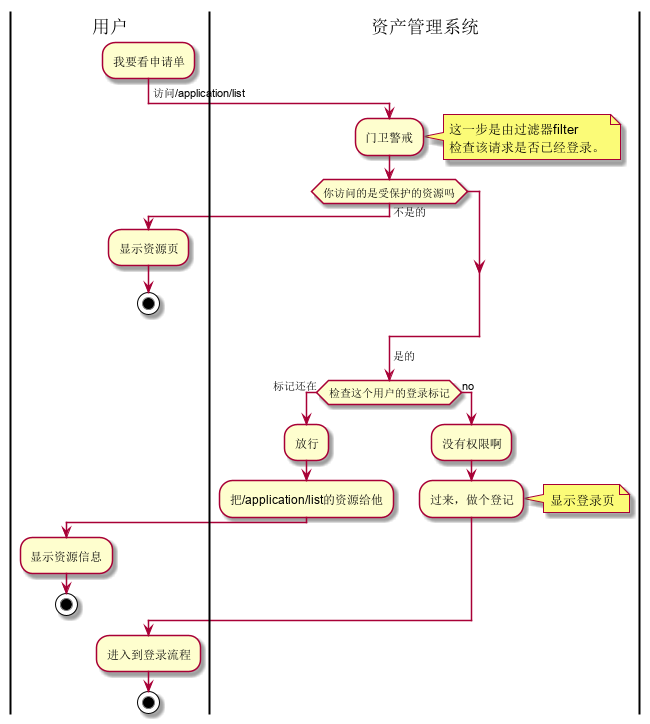
#### 拒绝维修申请单



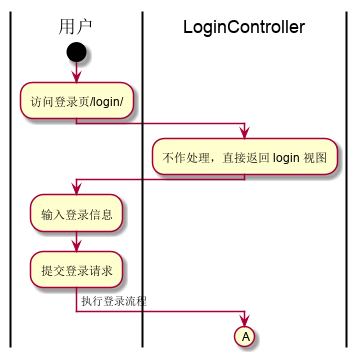
### （4）登录流程

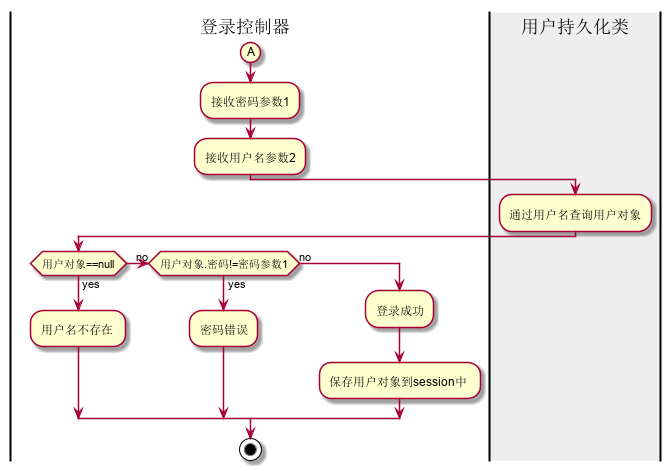
#### 1. 为什么要登录

为了保护资源，只允许授权的用户访问。

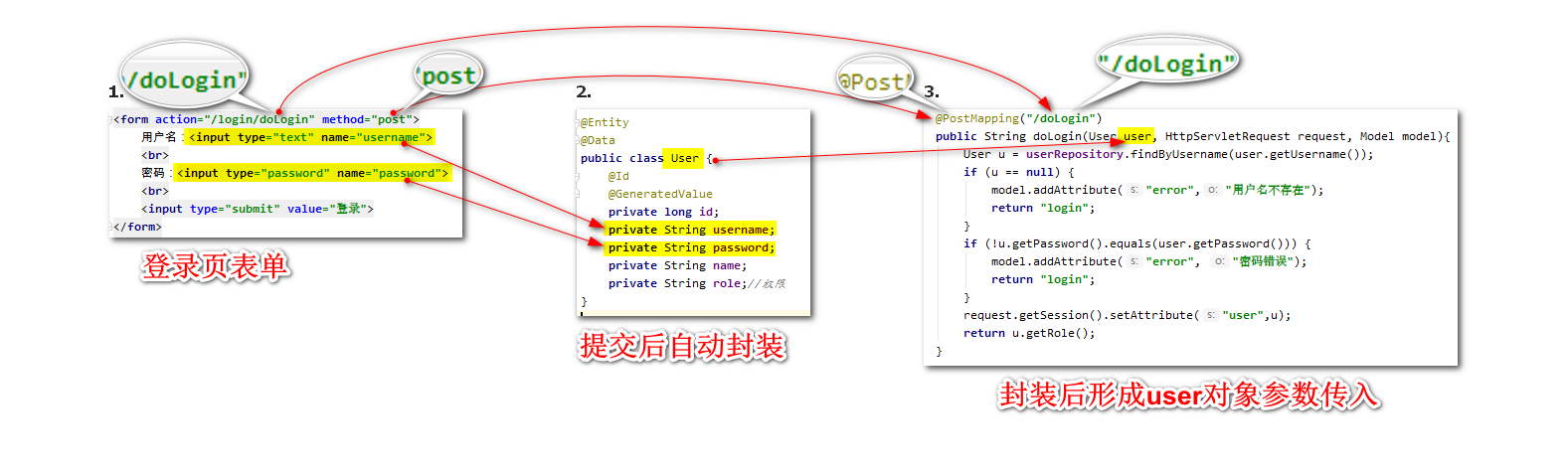


#### 2. 怎么获得授权



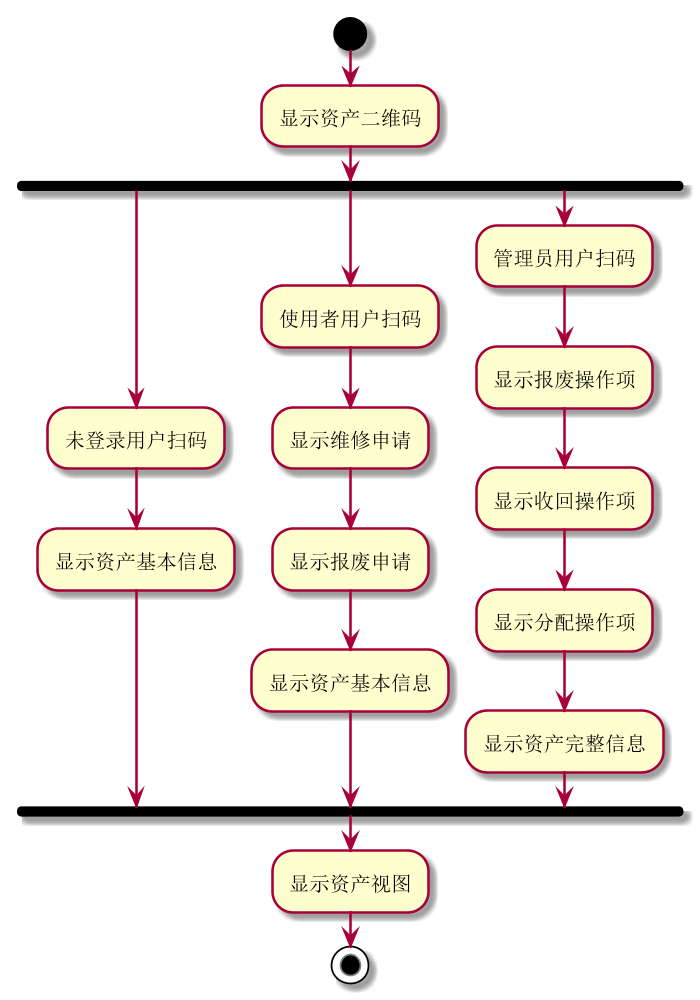


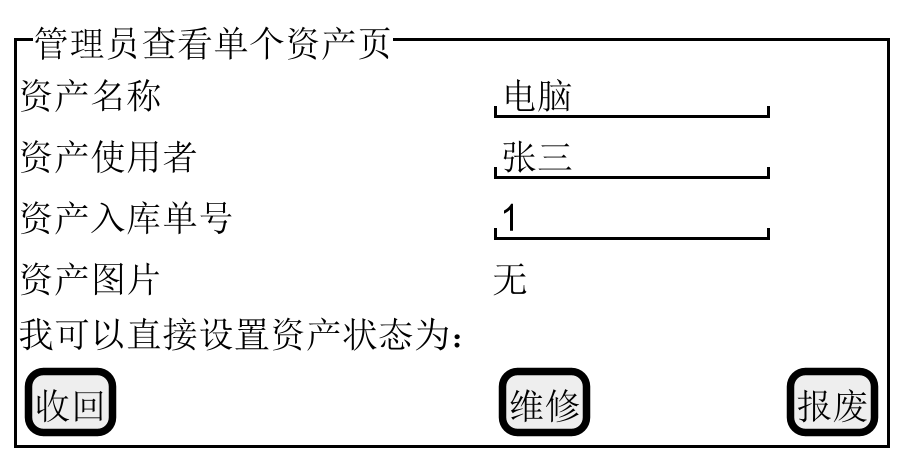
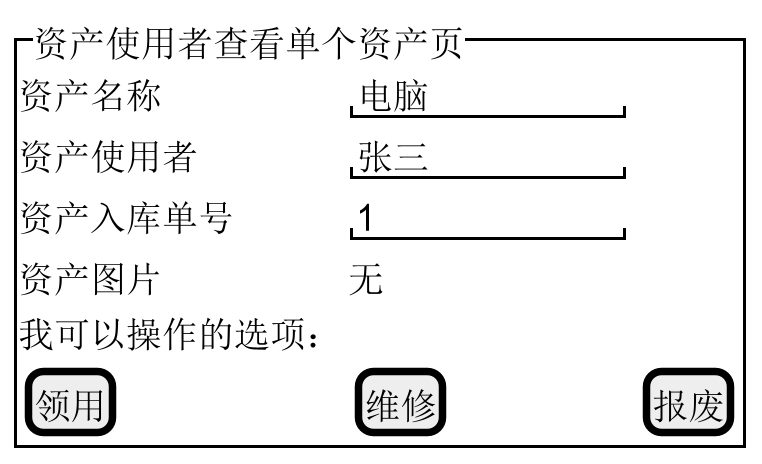
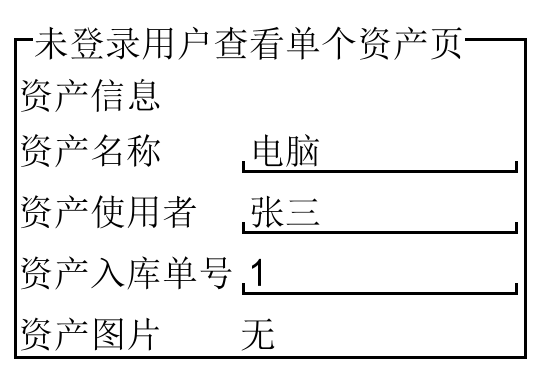
#### 3. 理解登录代码



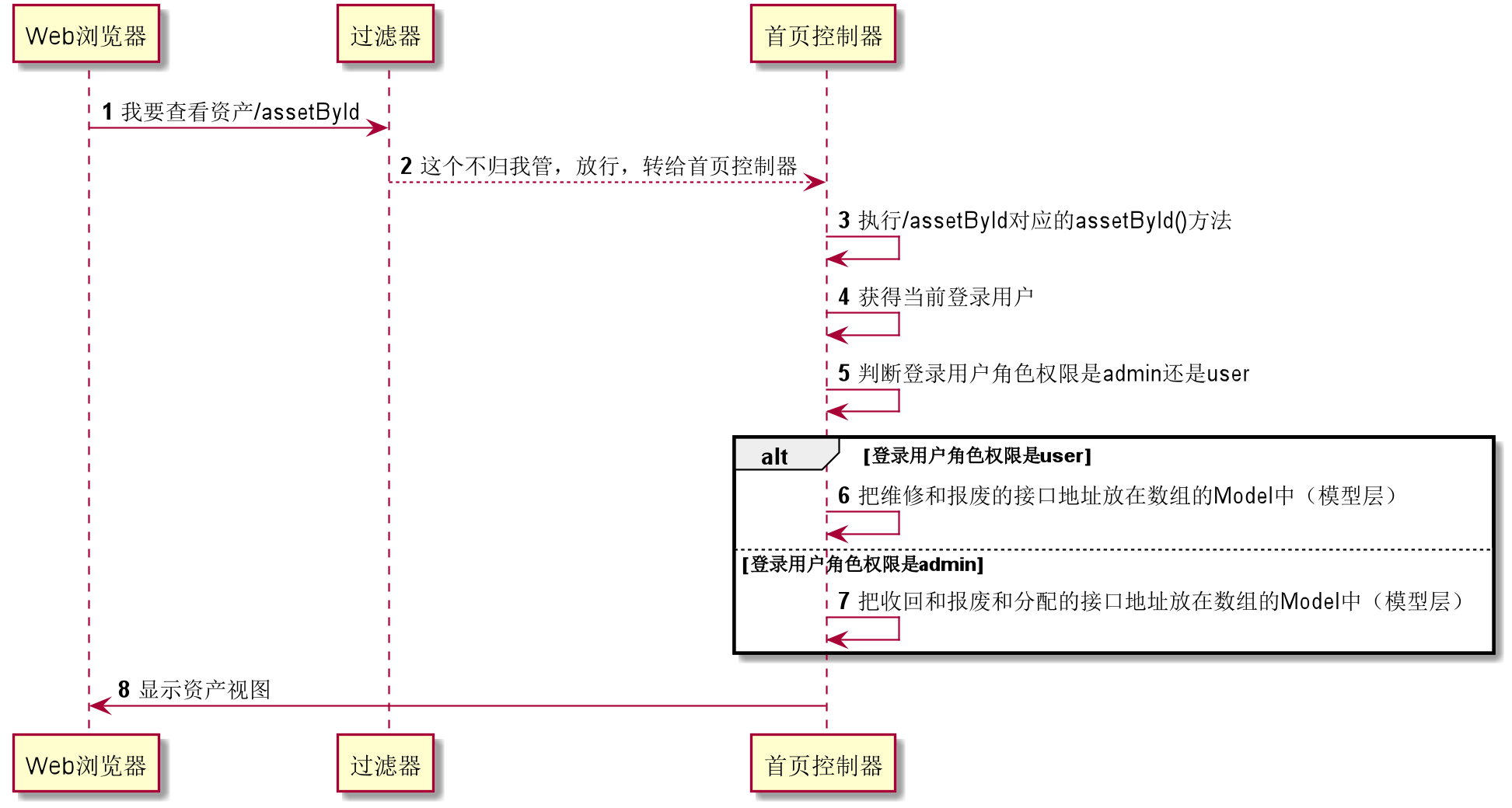
### （5）扫码显示资产页（不同用户显示不同内容）

#### 1. 查看资产二维码的业务流程

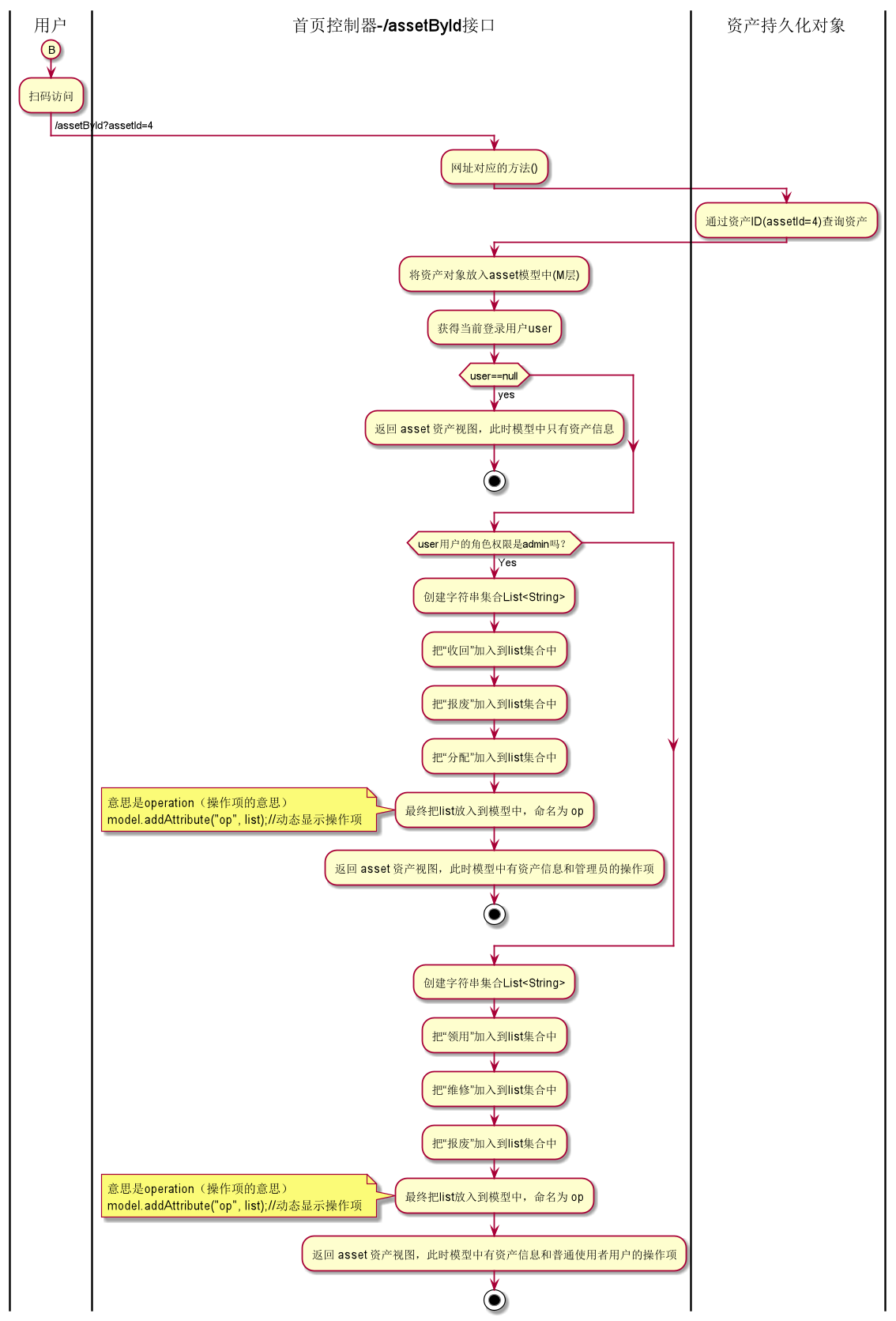




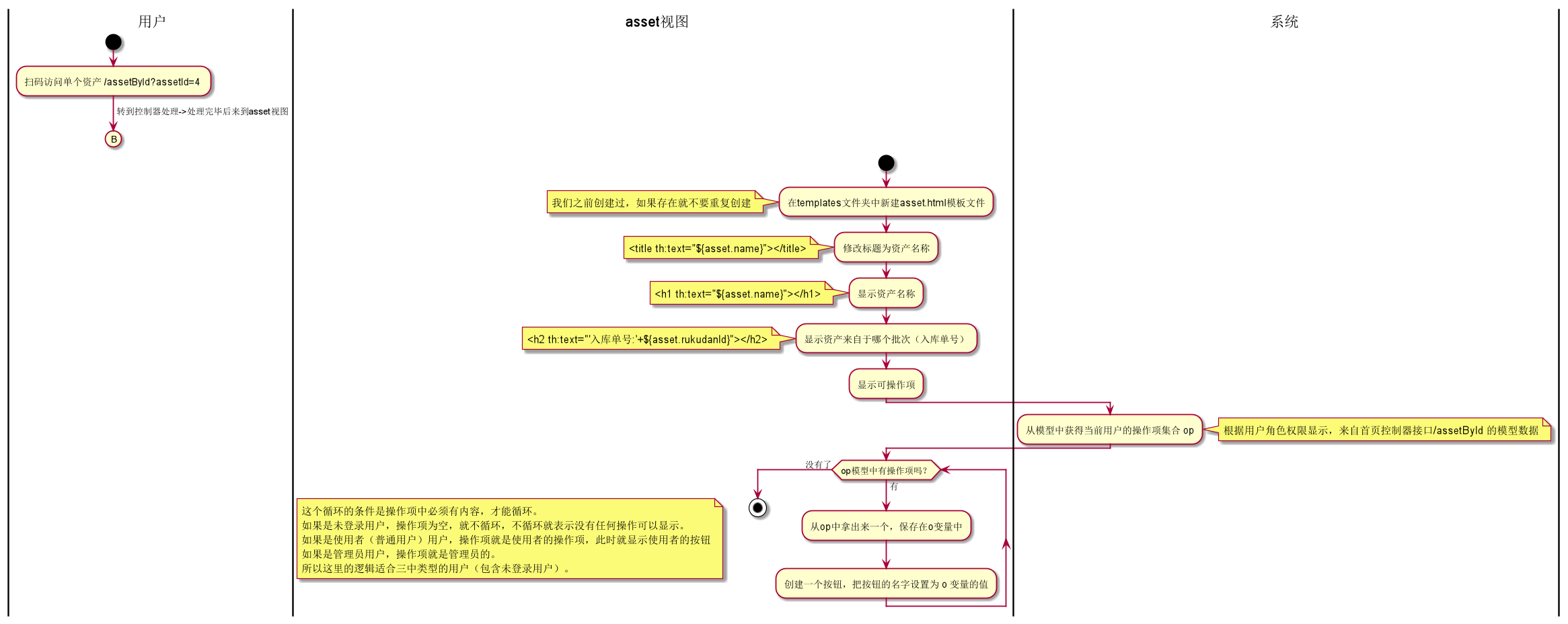
#### 2. 换个角度观察扫描资产二维码的顺序图



#### 3. 具体实施代码（程序流程图）

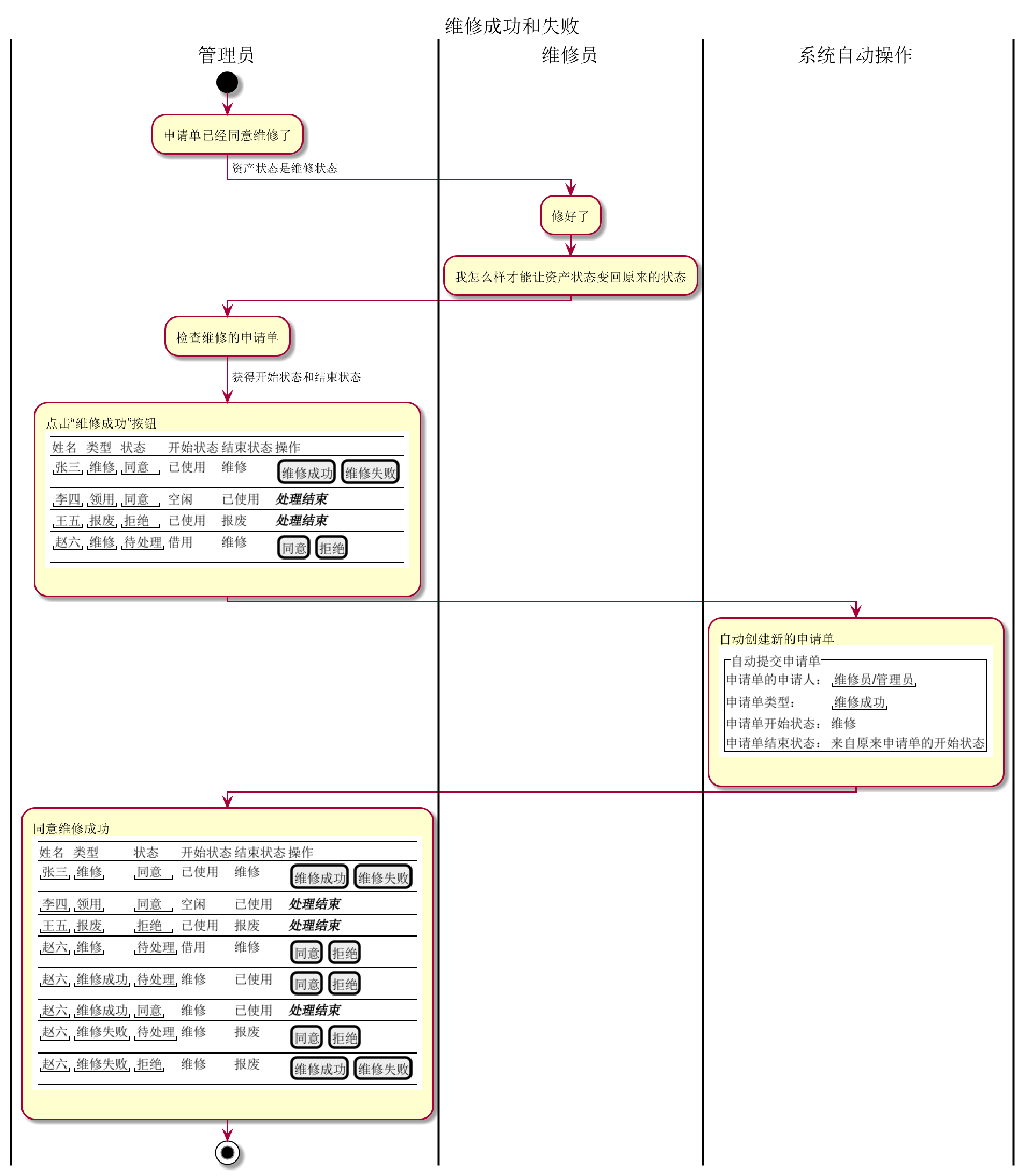


#### 4. 视图层逻辑（程序流程图）



### （6）维修处理流程

#### 1. 业务流程



#### 2. 原型模型



#### 3. 分析结论

管理员的操作项有三种：

1、同意和拒绝

2、维修成功、维修失败

3、处理结束

#### 4. 提炼系统需求

在什么情况下，我们显示 1、同意和拒绝？

在什么情况下，我们显示 2、维修成功、维修失败？

在什么情况下，我们显示 3、处理结束？

管理员可以显示申请单的操作项。

## 五、数据表结构-类结构分析

### （1）数据结构分析

#### 1.资产入库单类（AssetRukudan）- t\_assetrukudan

|  |  |  |  |
| --- | --- | --- | --- |
| 字段（属性） | 数据类型 | 约束 | 备注 |
| id | int | 自动递增+主键 |  |
| 入库日期(rukuDate) | Date |  | 2020年3月27日 |
| 资产名称(name) | String (varchar) |  | Dell品牌机 |
| 资产数量(amount) | int |  | 50个 |
| 资产类型(type) | String |  | 0=办公用品、  1=耗材、  2=电脑、  3=投影仪、  4=电子、  5=百货 |
| 资产说明书(readme) | String |  | 资产的使用说明和配置参数说明等等 |
| 资产单价(price) | float |  | 23.00/个 |
| 资产生产日期(scrq) | Date |  | 2020年3月26日 |
| 资产保质期(bzq) | String |  | 3年、5年、3个月  2023年3月26日过期 |

#### 2.资产类（Asset）- t\_asset

|  |  |  |  |
| --- | --- | --- | --- |
| 字段（属性） | 数据类型 | 约束 | 备注 |
| id | int | 自动递增+主键 |  |
| 资产名称(name) | String (varchar) |  | Dell品牌机 |
| 资产类型(type) | String |  | 0=办公用品、  1=耗材、  2=电脑、  3=投影仪、  4=电子、  5=百货 |
| 资产说明书(readme) | String |  | 资产的使用说明和配置参数说明等等 |
| 资产单价(price) | float |  | 23.00/个 |
| 资产状态(status) | String |  | 已使用状态、空闲状态、报废状态、未使用状态、维修状态、借用、预定（收到申请单） |
| 资产生产日期(scrq) | Date |  | 2020年3月26日 |
| 资产保质期(bzq) | String |  | 3年、5年、3个月  2023年3月26日过期 |
| 入库单号（rukudanId） | int |  | 记录该资产属于那一次采购的入库单 |
| 使用者(username) | String(varchar) |  | 使用该资产的人（用户名） |

#### 3.申请单类（Application）- t\_application

|  |  |  |  |
| --- | --- | --- | --- |
| 字段（属性） | 数据类型 | 约束 | 备注 |
| id | int | 自动递增+主键 |  |
| 申请人(username) | String |  | 需要资产的使用者(用户名) |
| 资产ID(assetId) | long |  | 具体的资产ID |
|  |  |  |  |
| 资产名称(assetName) | String |  | 来自于资产入库单的名称 |
| 申请时间(beginDate) | Date |  | 提交申请的时间 |
| 资产数量(amount) | int | 准备删除 | 我申请要2台显示器 |
| 资产用途(content) | String |  | 我要用来开发软件 |
| 处理人(manager) | String |  | 处理该申请的管理员 |
| 处理结果(resultContent) | String |  | 处理结果，处理意见、处理批示，已经没有电脑可用了 |
| 处理时间(resultDate) | Date |  | 还是处理申请的时间？ |
| 申请单状态(status) | String |  | 同意、拒绝、待处理、维修成功 |
| 资产原始状态(start) | String |  | 空闲、已使用、维修、报废、借用 |
| 资产目的状态(stop) | String |  | 空闲、已使用、维修、报废、借用 |
| 申请单类型(type) | String | [变更管理1](#_1、申请单类添加类型字段) | 领用、维修、报废、借用 |
| 申请单操作项(opertion) | String | 变更管理17 | 操作项第1种：同意、拒绝  操作项第2种：维修成功、维修失败  操作项第3种：处理结束 |

#### 4.用户类（User）- t\_user

后端数据表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段（属性） | 数据类型 | 约束 | 备注 |
| id | int | 自动递增+主键 |  |
| 用户名(username) | String | 唯一 | 初始用户:user/123456，admin/123456789 |
| 密码(password) | String |  |  |
| 姓名(name) | String |  |  |
| 角色权限(role) | String |  | 普通用户 user； 管理员 admin |

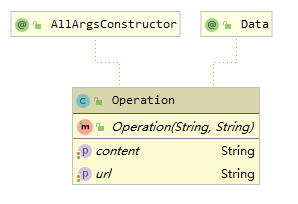
前端表单页

|  |  |  |  |
| --- | --- | --- | --- |
| 标签 | name属性 | 标签类型type | 显示标签 |
| form | - | - | action=/login/ method=post |
| input | username | text | 用户名 |
| input | password | password | 密码 |
| input | - | submit | 登录 |

#### 5.操作项类（Operation）- t\_operation

操作项（非持久化类）

|  |  |  |  |
| --- | --- | --- | --- |
| 字段（属性） | 数据类型 | 约束 | 备注 |
| id | int | 自动递增+主键 |  |
| 跳转url(url) | String |  | 比如：/application/list |
| 标签内容 (content) | String |  | 比如：维修、领用、报废、收回等 |



### （2）类图建模

#### 1. 控制器方法列表

v2.0

##### IndexController

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 方法 | URL | 备注 |
| 1 | index | @GetMapping({**"/index"**,**"/"**}) | 网页界面类 |
| 2 | assetById | @GetMapping(**"/asset/byId"**) | 通过资产ID查询资产视图 |

##### RukudanController

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 方法 | URL | 备注 |
| 1 | add | @GetMapping(**"/add"**) | 添加入库单 |
| 2 | list2 | @GetMapping(**"/list"**) | 查找所有输入单 |
| 3 | rukudanRemained | @GetMapping(**"/remained"**) | 查询入库单的剩余数量 |

##### AssetController

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 方法 | URL | 备注 |
| 1 | chuku | @GetMapping(**"/chuku"**) | 添加资产，资产出库 |
| 2 | list1 | @GetMapping(**"/listAsset"**) | 查询所有资产 |
| 3 | assetFree | @GetMapping(**"/free"**) | 查询资产来自于空闲状态 |
| 4 | findByStatus | @GetMapping(**"/findByStatus"**) | 通过状态查询资产 |

##### ApplicationController

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 方法 | URL | 备注 |
| 1 | applicationAdd | @PostMapping(**"/add"**) | 提交申请的接口 |
| 2 | applicationList | @GetMapping(**"/list"**) | 查询所有申请单 |
| 3 | applicationAgree | @GetMapping(**"/agree"**) | 处理同意申请单 |
| 4 | applicationAgreeRepair | @GetMapping(**"/agreeRepair"**) | 同意维修申请 |
| 5 | applicationRefuse | @GetMapping(**"/refuse"**) | 拒绝领用申请单 |
| 6 | applicationRefuseRepair | @GetMapping(**"/refuseRepair"**) | 拒绝维修申请单 |
| 7 | applicationAgreeScrap | @GetMapping(**"/agreeScrap"**) | 同意报废申请 |
| 8 | findByAsset | @GetMapping(**"/findByAsset"**) | 通过资产查询申请单 |
| 9 | findByType | @GetMapping(**"/findByType"**) | 通过申请单类型查询申请单 |
| 10 | findByTypeAndStatus | @GetMapping(**"/findByTypeAndStatus"**) | 通过状态和类型查询申请单 |

v1.0

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | IndexController方法 | URL | 备注 |
| 1 | index | @GetMapping({**"/index"**,**"/"**}) | 网页界面类 |
| 2 | add | @GetMapping(**"/add"**) | 添加资产，资产入库 |
| 3 | chuku | @GetMapping(**"/chuku"**) | 资产出库 |
| 4 | list1 | @GetMapping(**"/listAsset"**) | 列出所有资产 |
| 5 | list2 | @GetMapping(**"/listRukudan"**) | 列出所有资产入库单 |
| 6 | applicationAdd | @PostMapping(**"/application/add"**) | 申请单的操作 |
| 7 | applicationList | @GetMapping(**"application/list"**) | 查询所有申请单 |
| 8 | rukudanRemained | @GetMapping(**"rukudan/remained"**) | 查看入库单的库存量（剩余数量） |
| 9 | assetFree | @GetMapping(**"asset/free"**) | 查看当前空闲的电脑 |
| 10 | applicationAgree | @GetMapping(**"application/agree"**) | 同意领用申请单 |
| 11 | applicationAgreeRepair | @GetMapping(**"application/agreeRepair"**) | 同意维修申请 |
| 12 | applicationRefuse | @GetMapping(**"application/refuse"**) | 拒绝领用申请单 |

#### 2. UML类图

|  |  |
| --- | --- |
| v9  新增服务层和申请单服务类 |  |
| v8  申请单类新增两个属性 |  |
| v7  加入普通的操作项类 |  |
| v6  加入用户类 |  |
| v5 |  |
| v4 |  |
| v3 |  |
| v2.1 |  |
| v2.0 |  |
| v1.0 |  |

## 六、系统建模

### （1）资产管理员可以登记资产（资产入库）

#### （1）顺序图建模

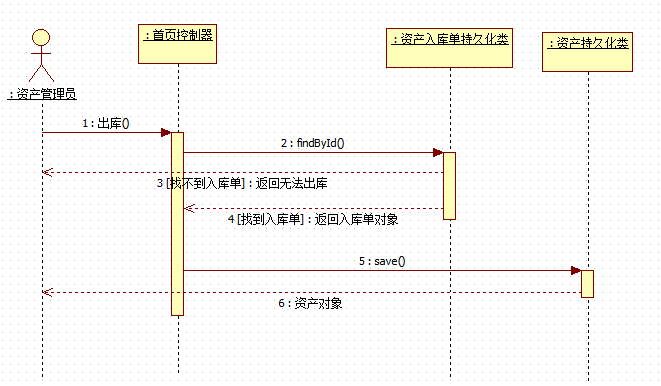
|  |  |
| --- | --- |
|  |  |
| v1.0 |  |

#### （2）程序流程图建模

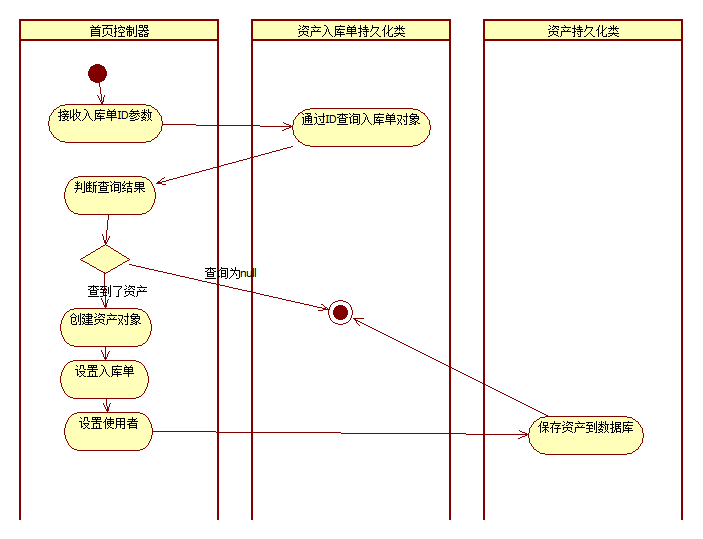
|  |  |
| --- | --- |
|  |  |
| v1.0 |  |

### （2）、资产管理员可以资产出库（分配使用者）

#### （1）顺序图建模



#### （2）程序流程图建模



### （3）、资产使用者可以提交领用申请

#### （1）参考代码

版本v2.0 Java 后端

*//申请单的操作* @PostMapping(**"/application/add"**) *//提交申请的接口* **public** Application applicationAdd(Application application){  
Asset asset= **assetRepository**.findById(application.getAssetId()).orElse(**null**);  
 **if**(asset == **null**){**return null**;}*//如果没有找到资产就返回null* application.setBeginDate(**new** Date()); *//设置申请时间为当前时间* application.setUsername(**"张三"**);*//****TODO 应该存放当前登录的用户*** application.setStatus(**"待处理"**);*//任何人提交申请都应该是 待处理状态* application.setResultContent(**""**);*//待处理状态下，是没有处理结果的* application.setManager(**""**);*//提交申请，默认进入无管理员处理* application.setAssetId(asset.getId());*//记录下申请单上指定的资产ID* application.setAssetName(asset.getName());*//申请单上的资产名称应该和入库单资产名称一致* application.setResultDate(**null**);*//提交申请后，没有处理时间* **return applicationRepository**.save(application);  
 }

版本v2.0 HTML前端

<!DOCTYPE **html**>  
<**html**>  
<**head**>  
 <**meta charset="utf-8"**>  
 <**title**>测试资产管理系统的功能</**title**>  
</**head**>  
<**body**>  
<**h1**>资产入库</**h1**>  
资产名称<**input type="text" id="assetName"**><**br**>  
资产数量<**input type="text" id="assetAmount"**>  
<**button onclick="**zichanruku()**"**>资产入库</**button**>  
  
<**h1**>资产出库</**h1**>  
入库单ID <**input type="text" id="chuku\_rukudanId"**><**br**>  
<**button onclick="**chuku()**"**>资产出库</**button**>  
  
<**h1**>检查入库单剩余量</**h1**>  
<**input type="text" id="rukudanId"**>  
<**button onclick="**rukudanRemained()**"**>检查入库单剩余量</**button**>  
  
<**h1**>查询空闲资产</**h1**>  
入库单ID<**input type="text" id="free\_rukudanId"**><**br**>  
<**button onclick="**freeAssets()**"**>查询空闲资产</**button**>  
<**script type="text/javascript"**>  
 **var** global = **"http://localhost:8080/"**;  
 */\*\*  
 \* 资产入库  
 \*/* **function** zichanruku(){  
 **var** aname = document.getElementById(**"assetName"**).value;  
 **var** amount = document.getElementById(**"assetAmount"**).value;  
 **var** url = global+**"add?name="**+aname+**"&amount="**+amount;  
 location.href=url;  
 }  
 */\*\*  
 \* 资产出库  
 \*/* **function** *chuku*(){  
 **var** rukudanId = document.getElementById(**"chuku\_rukudanId"**).value;  
 **var** url = global+**"chuku?rukudanId="**+rukudanId;  
 location.href=url;  
 }  
 */\*\*  
 \* 资产剩余量查询  
 \*/* **function** rukudanRemained(){  
 **var** rukudanId = document.getElementById(**"rukudanId"**).value;*//获得ID为rukudanId标签的值* **var** url = global+**"rukudan/remained?rukudanId="**+rukudanId;*//访问服务器的网址url接口* location.href=url;  
 }  
 */\*\*  
 \* 查询空闲资产  
 \*/* **function** freeAssets(){  
 **var** rukudanId = document.getElementById(**"free\_rukudanId"**).value;*//获得ID为rukudanId标签的值* **var** url = global+**"asset/free?rukudanId="**+rukudanId;*//访问服务器的网址url接口* location.href=url;  
 }  
</**script**>  
</**body**>  
</**html**>

版本v1.0 Java后端

*//申请单的操作*@PostMapping(**"/application/add"**) *//提交申请的接口***public** Application applicationAdd(Application application){  
 *//持久化保存到申请单数据表中。  
 //查询入库单，通过前台传递过来的参数中获得入库单号，再通过入库单持久化对象去查询* AssetRuKuDan assetRuKuDan = **assetRuKuDanRepository**.findById(application.getRukudanId()).orElse(**null**);  
 **if** (assetRuKuDan == **null**){  
 System.***out***.println(**"没有找到入库单:"**+application.getRukudanId());  
 **return null**;  
 } *//如果入库单填写错误，没有找到，则直接返回null，不做保存处理* application.setBeginDate(**new** Date()); *//设置申请时间为当前时间* application.setUsername(**"张三"**);*//****TODO 应该存放当前登录的用户*** application.setStatus(**"待处理"**);*//任何人提交申请都应该是 待处理状态* application.setResultContent(**""**);*//待处理状态下，是没有处理结果的* application.setManager(**""**);*//提交申请，默认进入无管理员处理* application.setAssetName(assetRuKuDan.getName());*//申请单上的资产名称应该和入库单资产名称一致* application.setResultDate(**null**);*//提交申请后，没有处理时间* **return applicationRepository**.save(application);  
}  
  
*/\*\*  
 \* 查询所有申请单  
 \** ***@return*** *\*/*@GetMapping(**"application/list"**)*//查询所有申请的接口***public** List<Application> applicationList(){  
 **return applicationRepository**.findAll();  
}

版本v1.0 HTML前端

<!DOCTYPE **html**>  
<**html**>  
<**head**>  
 <**meta charset="utf-8"**>  
 <**title**></**title**>  
</**head**>  
<**body**>  
<**h1**>测试提交申请</**h1**>  
<**form action="http://localhost:8080/application/add" method="post"**>  
 资产来源<**input type="text" name="rukudanId"** />  
 <**br**>  
 资产名称 来自于入库单ID，由系统自动完成  
 <**br**>  
 资产数量 <**input type="number" name="amount" id="" value="1"** />  
 <**br**>  
 处理人 <**input type="text" name="manager"**>  
 <**br**>  
 资产用途 <**textarea name="content" rows="4" cols="4"**></**textarea**>  
 <**input type="submit" value="提交申请"**/>  
</**form**>  
</**body**>  
</**html**>

#### （2）顺序图建模



#### （3）程序流程图建模



### （4）、管理员可以查看所有申请

因为该需求过小，无需建立模型，直接查看代码。

*/\*\*  
 \* 查询所有申请单  
 \** ***@return*** *\*/*@GetMapping(**"application/list"**)*//查询所有申请的接口***public** List<Application> applicationList(){  
 **return applicationRepository**.findAll();  
}

### （5）、管理员可以查看空闲资产

*//查看当前空闲的电脑*@GetMapping(**"asset/free"**) *//查询资产来自于空闲状态***public** List<Asset> assetFree(**long** rukudanId){*//是不是把所有的空闲资产都查出来？  
 //assetRepository; //****TODO 先去解决查询数据库的代码然后过来*** List<Asset> freeAssets = **assetRepository**.findAllByRukudanIdAndStatus(rukudanId, **"空闲"**);  
 **return** freeAssets;*//返回给前端，让使用者查看并选择哪一个资产作为领用申请单上的资产ID*}

### （6）、管理员可以查看剩余数量

*//查看入库单的库存量（剩余数量）*@GetMapping(**"rukudan/remained"**)  
**public int** rukudanRemained(**long** rukudanId){  
 *//查询入库单对象* AssetRuKuDan assetRuKuDan = **assetRuKuDanRepository**.findById(rukudanId).orElse(**null**);  
 **return** assetRuKuDan.getRemained();*//获得该入库单的剩余数量*}

### （7）、资产管理员可以处理申领请求

#### 同意领用申请单

流程步骤

1、接收两个参数，申请单号和同意意见。

2、调用【申请单持久化类对象】查找申请单号findById(申请单号)

3、通过申请单得到资产ID

4、通过【资产持久化类对象】查找资产

5、判断资产是否为null，

5a、如果是null就直接返回

5b、如果不是null就执行第6步

6、获得资产状态

7、判断资产状态

7a 资产状态不等于预定状态，就返回

7b 资产状态等于预定状态，进行第8步

8、设置资产状态为“已使用”

9、设置申请单状态为“同意”

10、设置申请单处理人【当前登录用户】

11、设置申请单处理意见为【参数result】

12、设置申请单处理日期为当前日期

13、调用【申请单持久化类】的保存方法，将申请单更新到数据库

14、返回更新后的申请单。

#### 同意维修申请单

流程步骤

1、接收申请单号参数

2、调用【申请单持久化类】的findById，查找申请单对象

3、判断申请单

3a、申请==null，直接返回null

3b、申请单不等于null，进行第4步

4、获得申请单中的资产ID

5、调用【资产持久化类对象】的findById，查找资产

6、判断资产

6a、如果资产==null，直接返回null

6b、如果资产不等于null，进行第7步

7、获得资产状态

8、获得申请单类型

9、判断申请单类型是不是维修

9a、申请单不是“维修”单就直接返回null

9b、申请是“维修”单，执行第10步

10、判断资产状态

10a、资产状态不是“已使用”，就返回

10b、资产状态是“已使用”，执行第11步

11、设置资产状态为“维修”

12、调用【资产持久化类】的save方法，保存资产到数据库

13、设置申请单状态“同意”

14、设置申请单处理人为【当前登录用户】

15、设置申请单处理意见为【参数result】

16、设置处理时间为【当前时间】

17、调用【申请单持久化类对象】的save方法保存申请单

18、返回保存后的申请单

#### 拒绝领用申请单

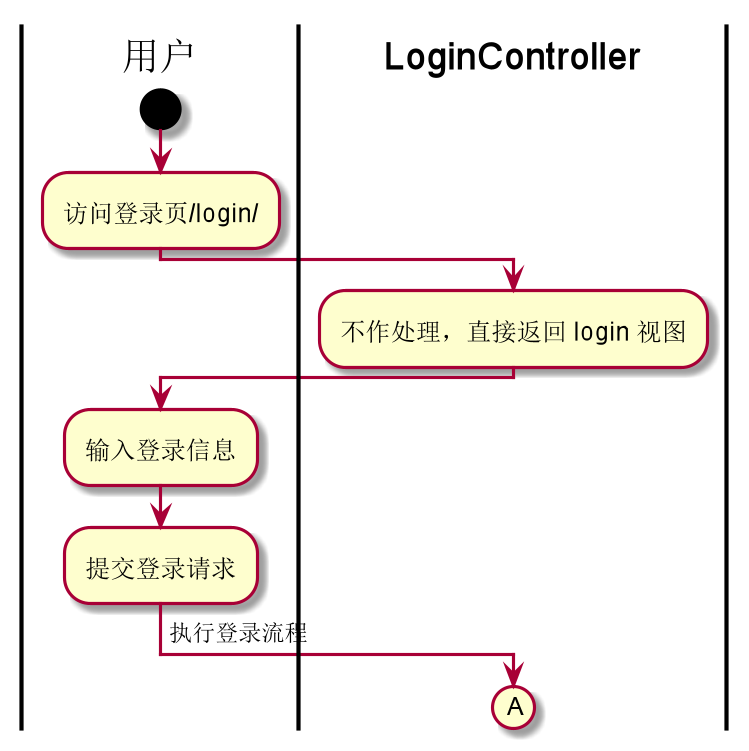
*/\*\*  
 \* //拒绝领用申请单  
 \** ***@param applicationId*** *申请单ID  
 \*/*@GetMapping(**"application/refuse"**)  
**public** Application applicationRefuse(**long** applicationId,String result){*//处理同意申请单的方法  
 //从数据库中查询申请单对象* Application application = **applicationRepository**.findById(applicationId).orElse(**null**);  
  
 *//新增内容 2020-4-1 14:39:21 begin* **long** assetId = application.getAssetId();*//获得资产ID* Asset asset = **assetRepository**.findById(assetId).orElse(**null**); *//查询资产对象* **if** (asset == **null**) { *//如果没找到* **return null**;*//就直接返回* }  
 String status = asset.getStatus();*//获得资产当前状态* **if** (status == **"预定"**) {*//必须是“预定”状态下才可以回到空闲。* asset.setStatus(**"空闲"**);  
 **assetRepository**.save(asset);  
 }*//否则不对资产对象做任何处理，原来是什么就是什么。  
 //新增内容 2020-4-1 14:39:21 end  
  
 //****TODO 是否考虑资产的状态，空闲可以拒绝吗？。*** application.setStatus(**"拒绝"**);*//设置申请单状态为 同意* application.setManager(**"黄主任"**);*//****TODO 应该是当前登录的用户（admin）*** application.setResultContent(result);*//****TODO 如果没有写处理意见呢？*** application.setResultDate(**new** Date());*//处理时间为当前时间  
 //把已经处理的申请单保存到数据库中并返回* **return applicationRepository**.save(application);  
}

#### 拒绝维修申请单

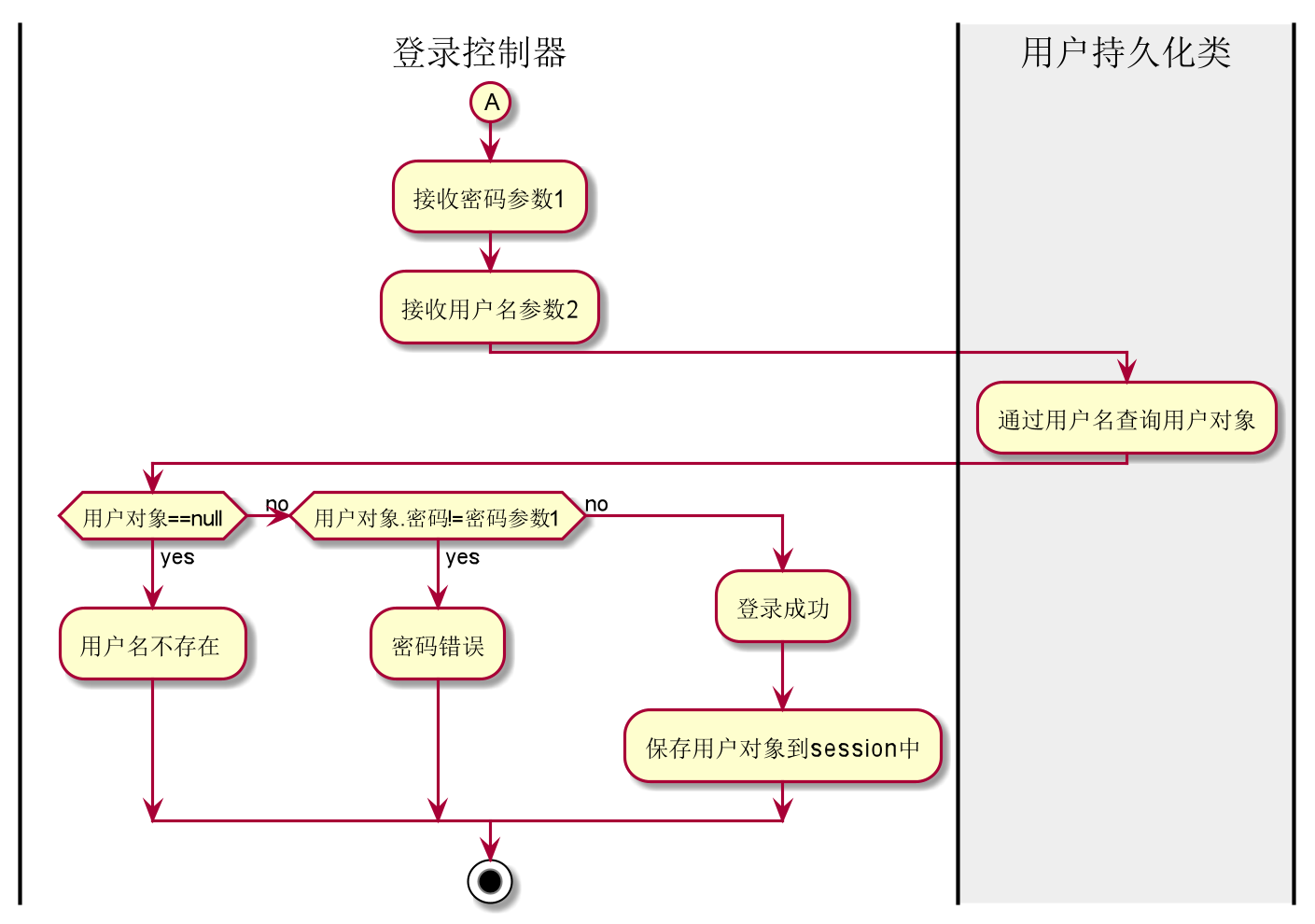
*/\*\*  
 \* //拒绝维修申请单  
 \** ***@param applicationId*** *申请单ID  
 \*/*@GetMapping(**"application/refuseRepair"**)  
**public** Application applicationRefuseRepair(**long** applicationId,String result){*//处理同意申请单的方法  
 //从数据库中查询申请单对象* Application application = **applicationRepository**.findById(applicationId).orElse(**null**);  
  
 *//****拒绝维修申请，不需要改变资产状态*** application.setStatus(**"拒绝"**); *//设置申请单状态为 同意* application.setManager(**"黄主任"**); *//****TODO 应该是当前登录的用户（admin）*** application.setResultContent(result); *//****TODO 如果没有写处理意见呢？*** application.setResultDate(**new** Date()); *//处理时间为当前时间  
 //把已经处理的申请单保存到数据库中并返回* **return applicationRepository**.save(application);  
}

### （8）、用户登录流程

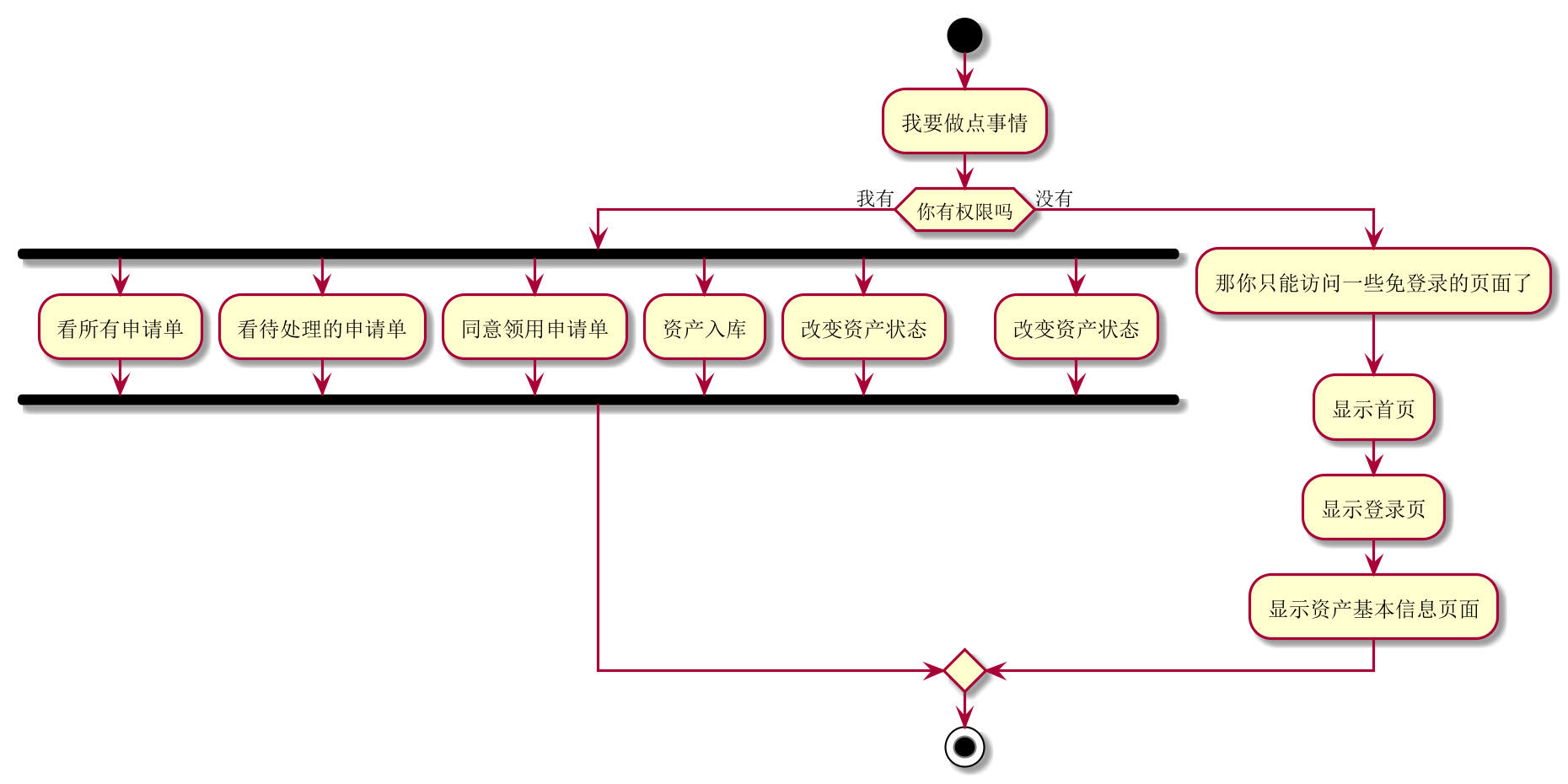
#### 1. LoginController-访问登录页

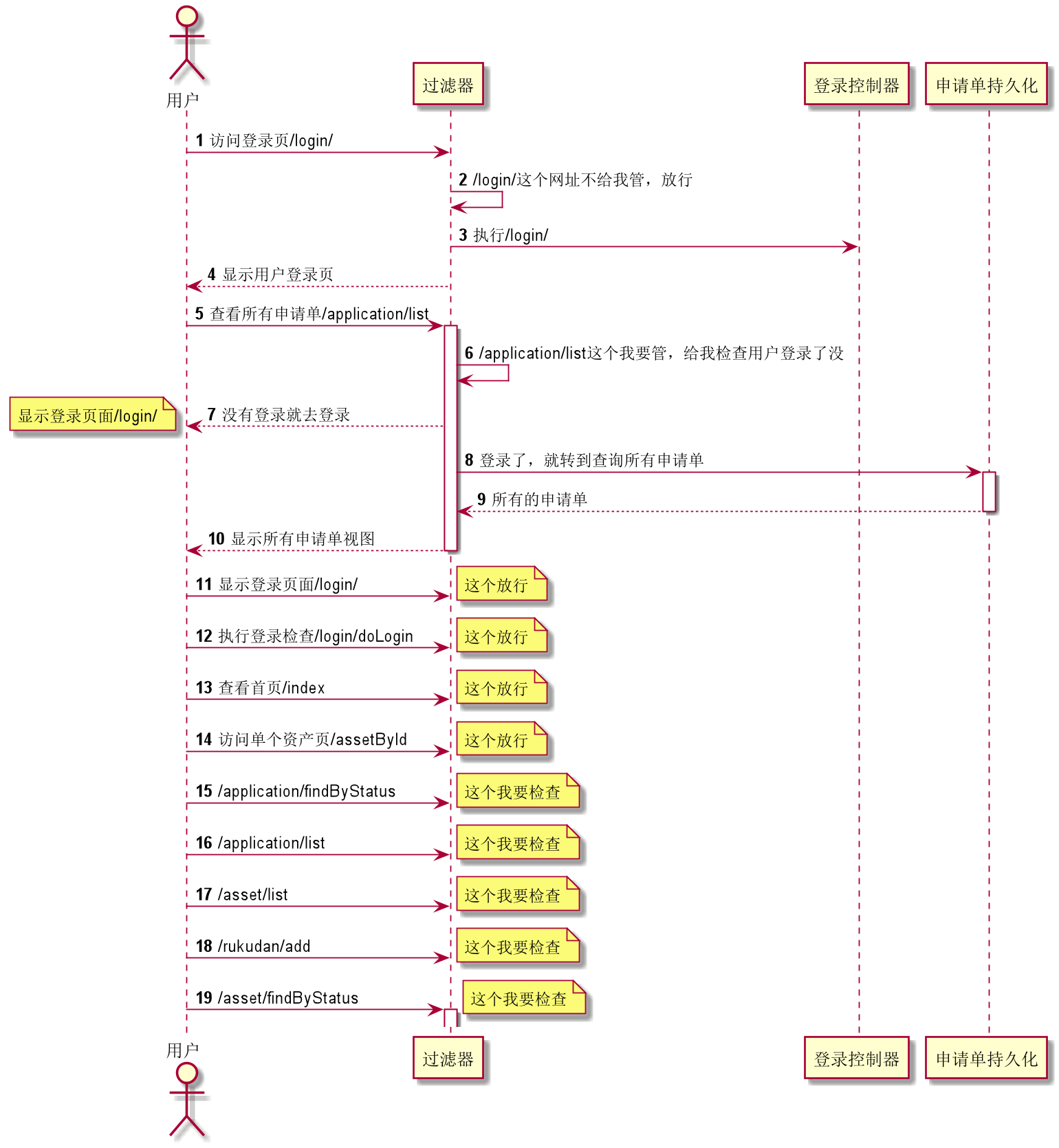


#### 2. LoginController-A登录流程



#### 3. 过滤器的原理



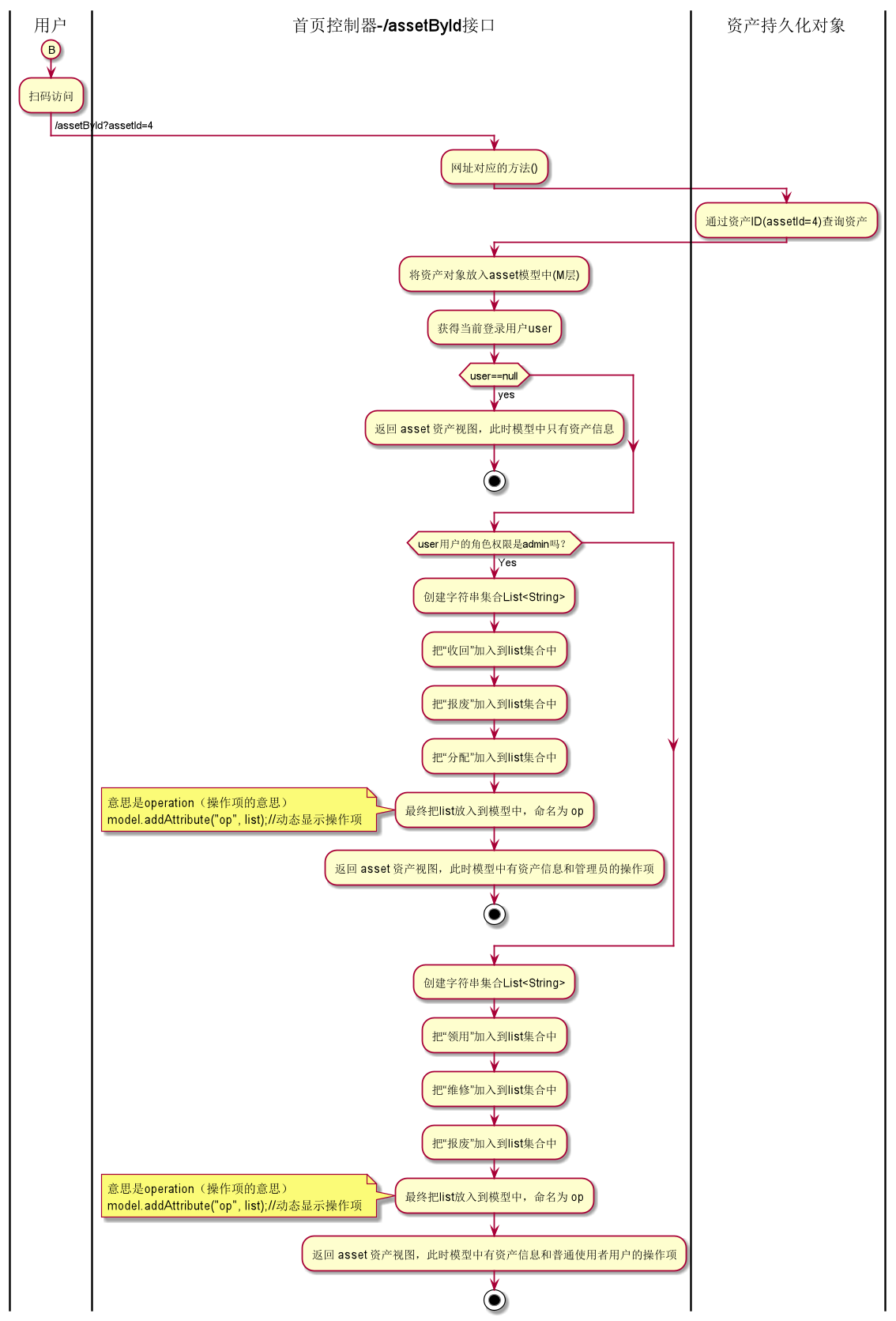


### （9）、查看资产信息

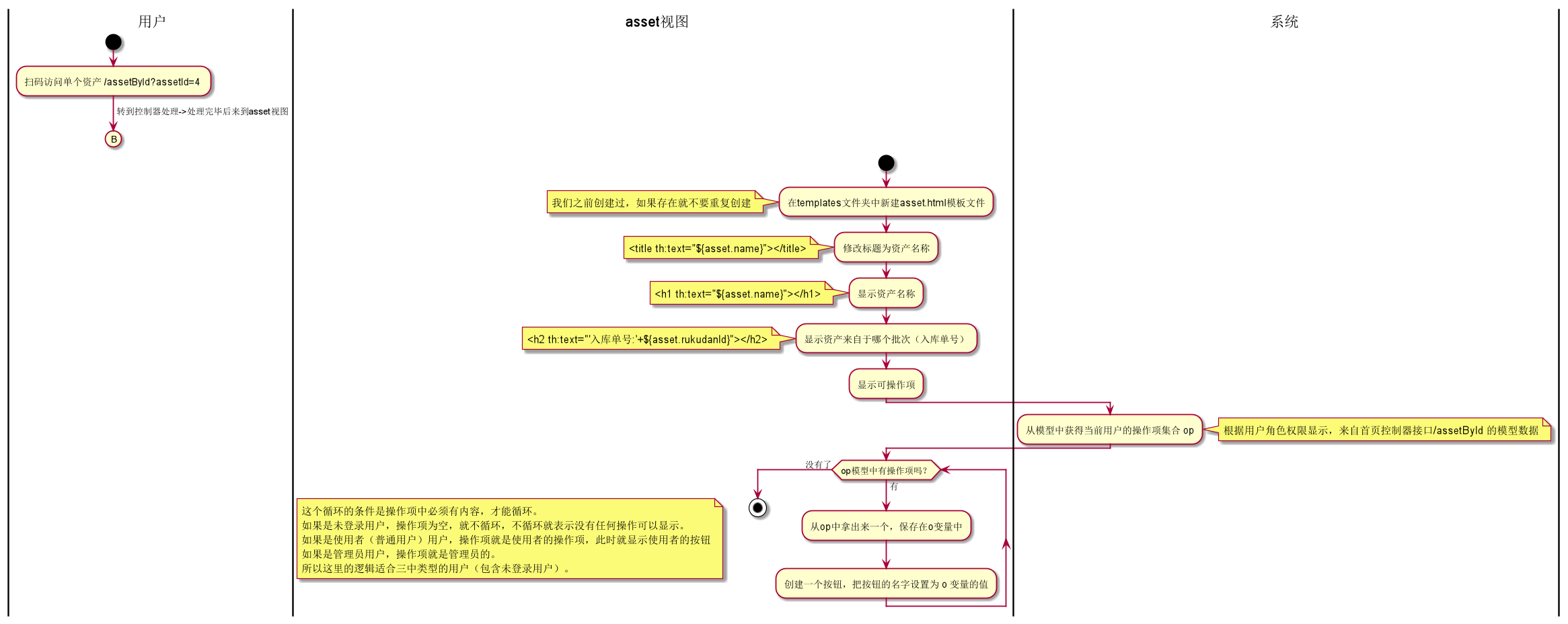
#### 1. 后端IndexController（首页控制器）

不同的用户查看资产页，显示的内容也不同。

后端IndexController（首页控制器）的/assetById接口对应的方法，程序流程图：

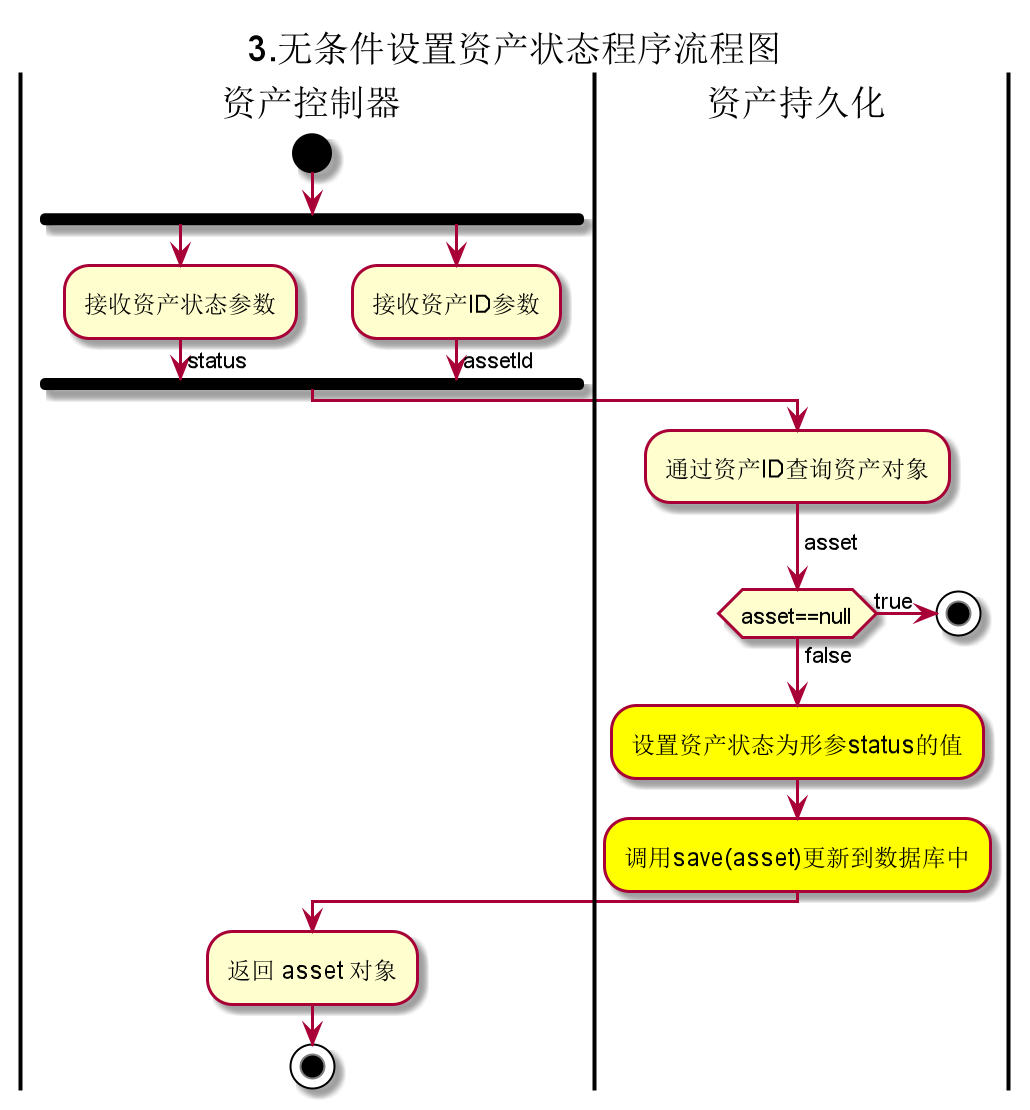


#### 2. 前端程序流程图

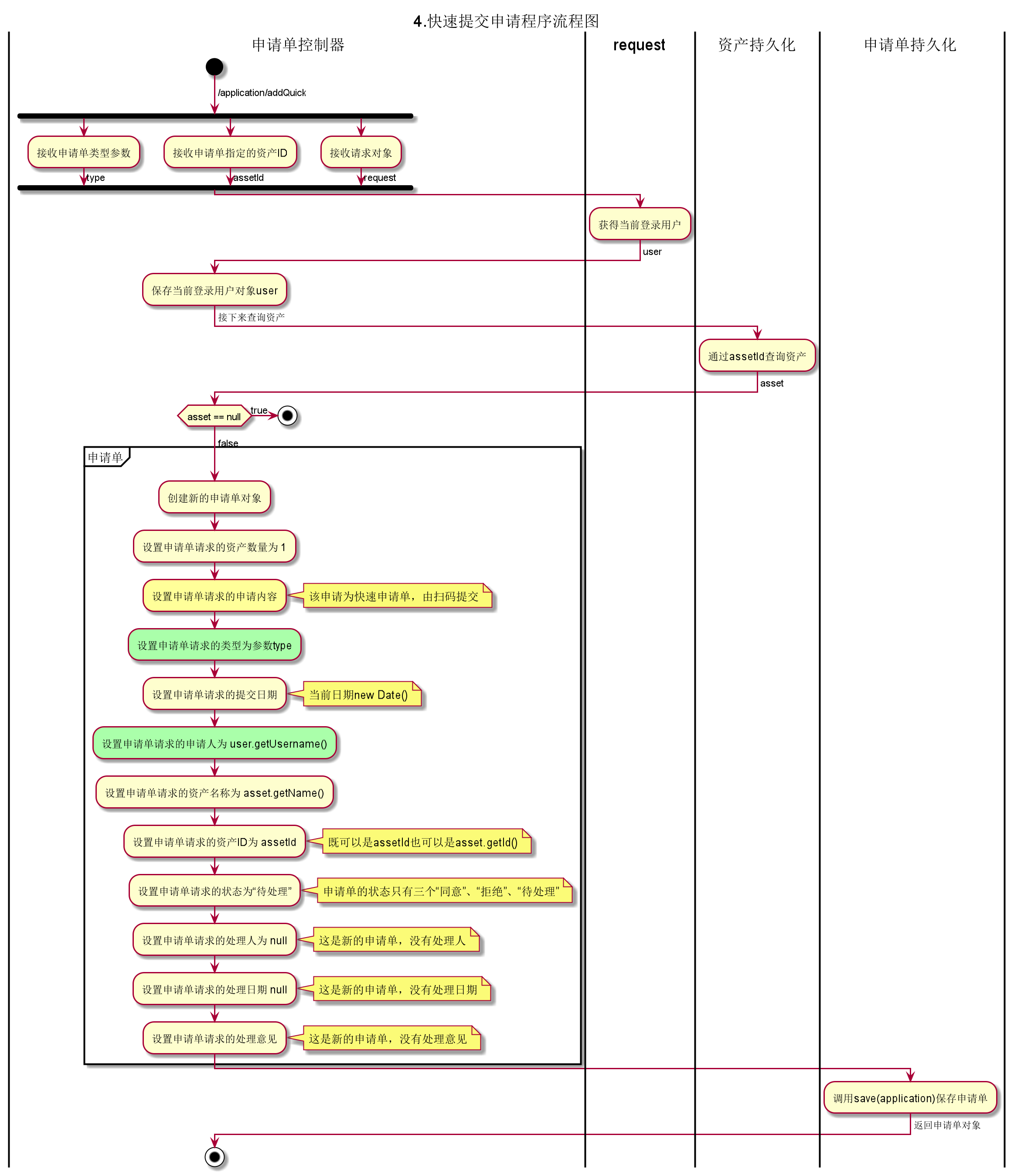


### （10）、管理员扫码设置资产状态

流程图

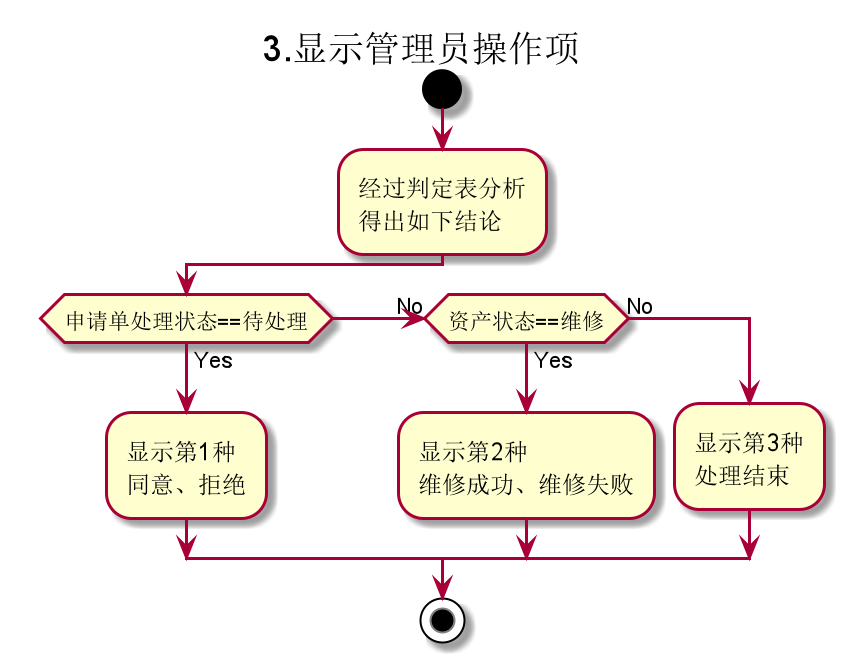


### （11）、使用者扫码设置提交申请

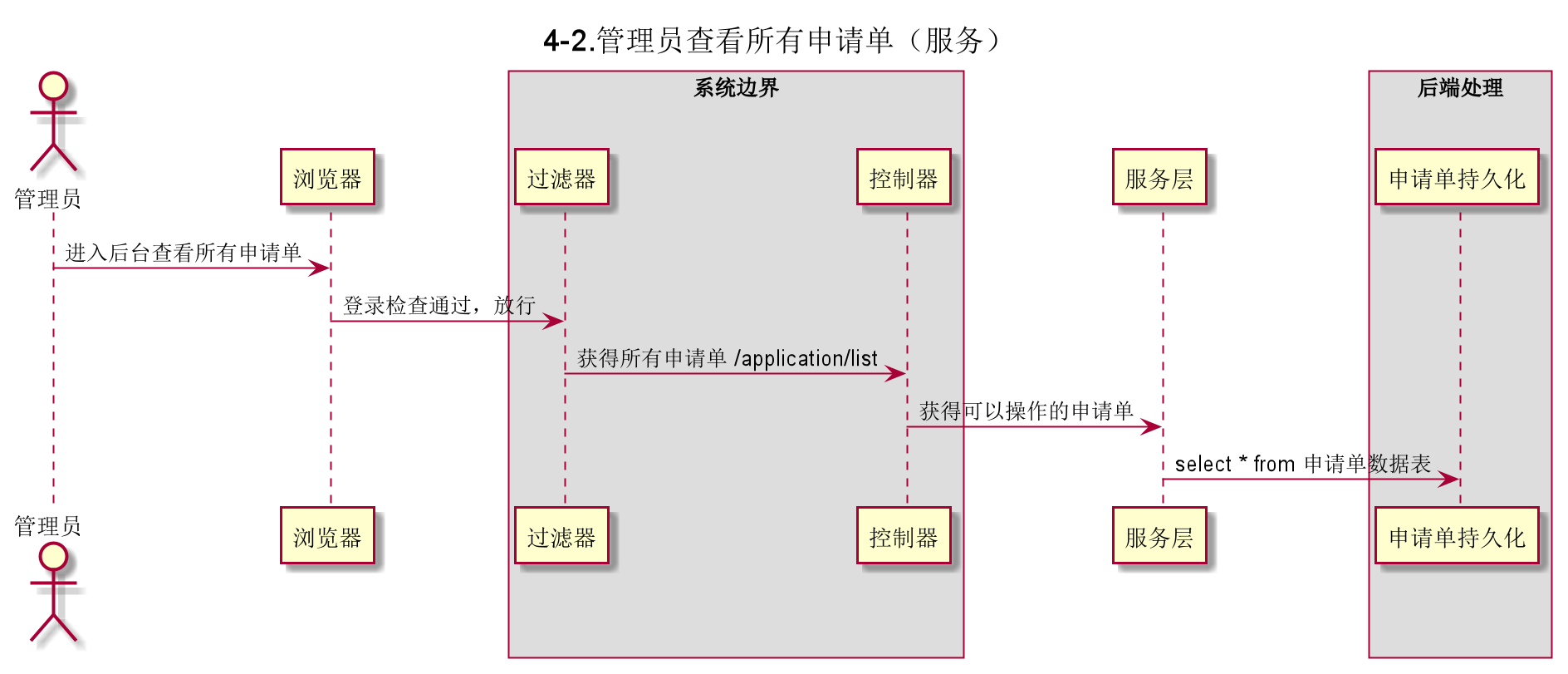


### （12）、管理员可以显示申请单的操作项

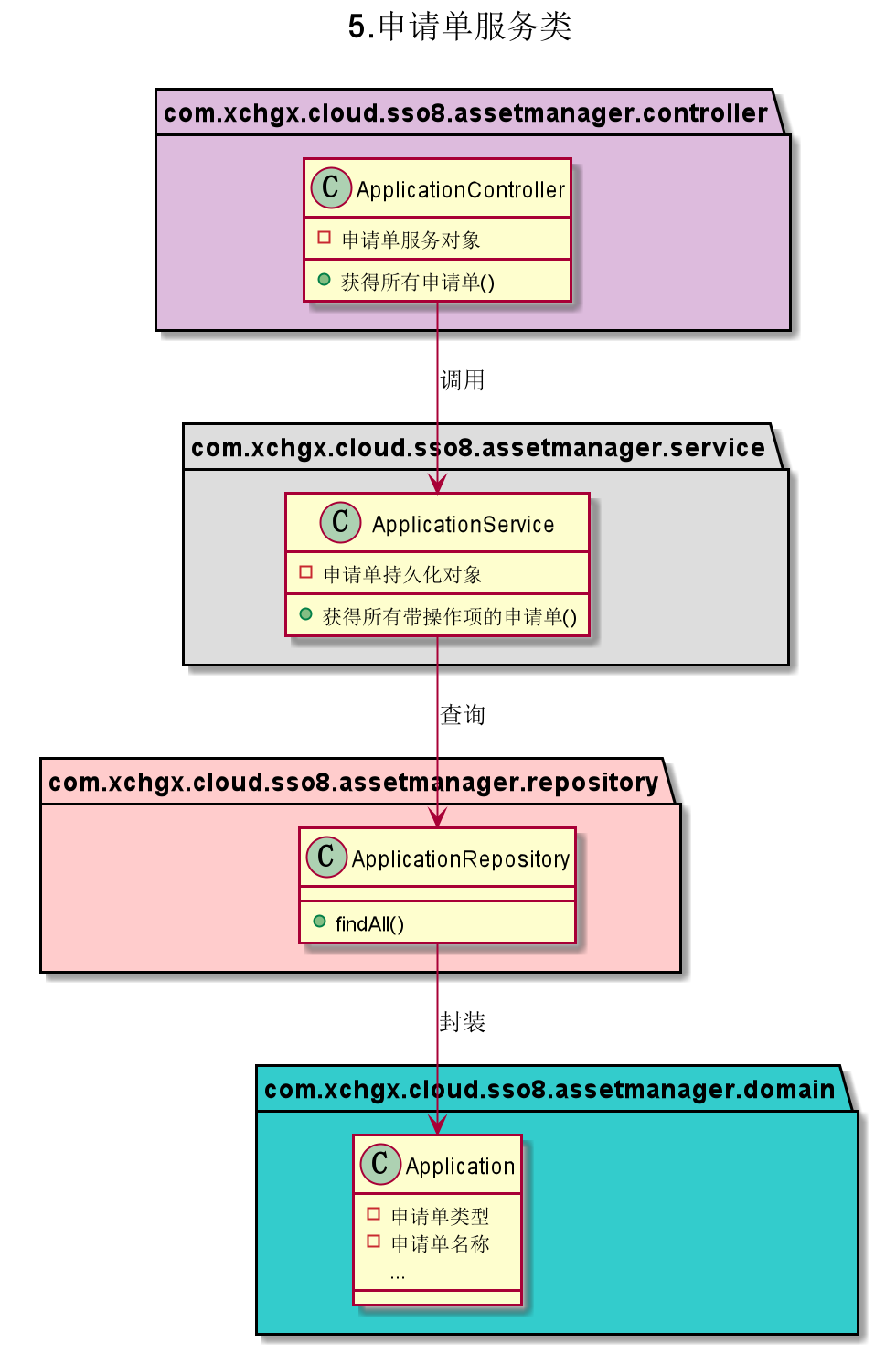
#### 1. 显示申请单操作选项的程序流程图



#### 2. 显示申请单操作选项的顺序图



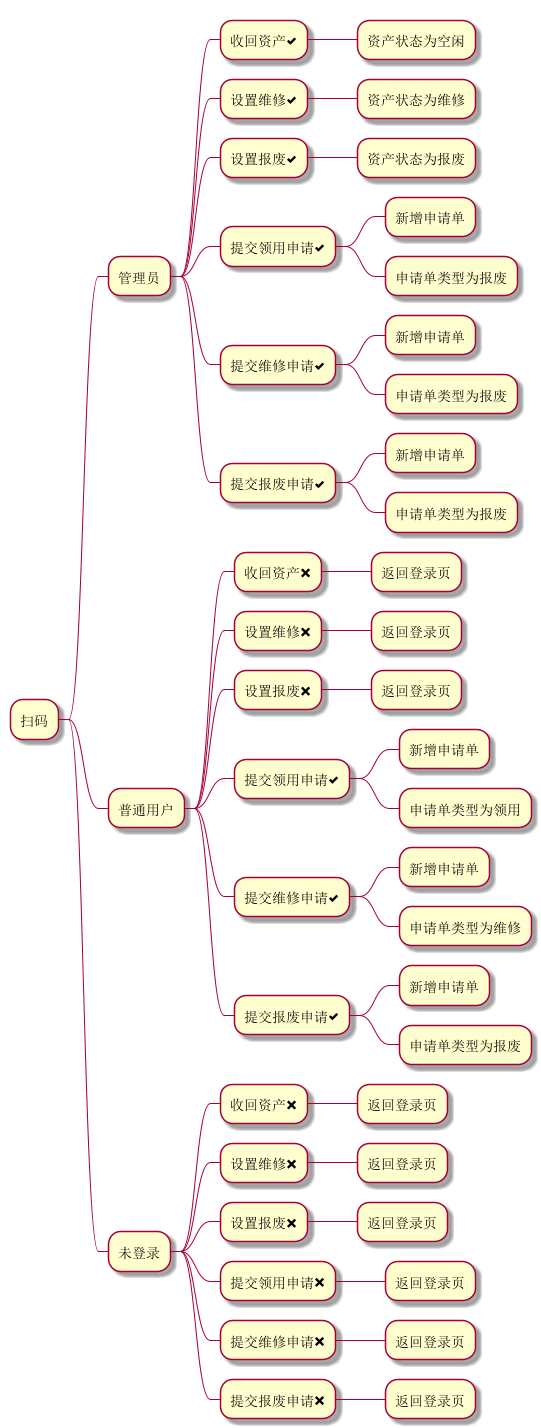
#### 3.显示申请单操作项的类图



## 七、\*测试管理

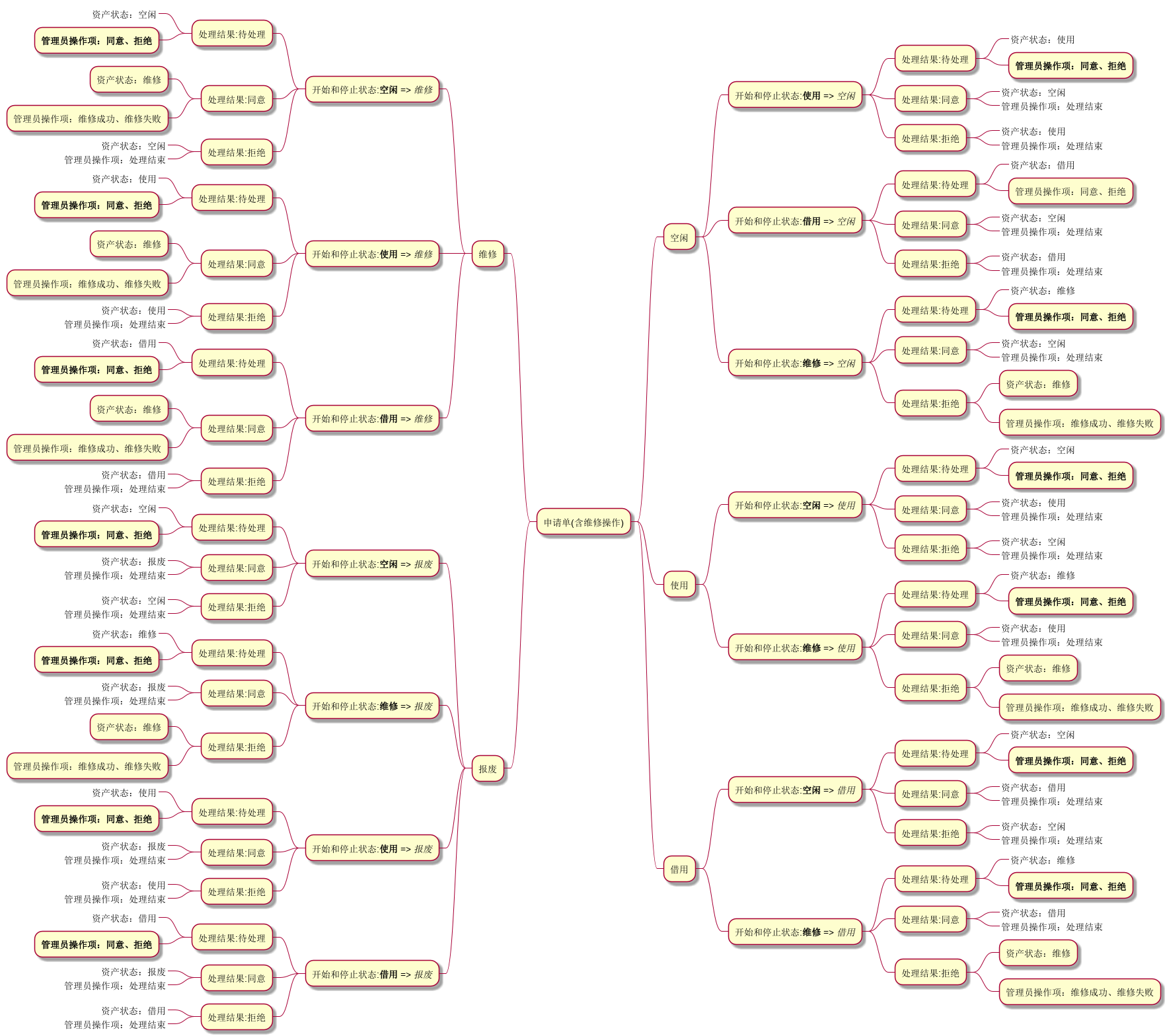
### （1）测试扫码功能-设置资产状态和提交申请

判定树



### （2）\*测试维修操作的结果

#### 判定树



从判定树中，我们可以得出如下结论。

1、申请单为待处理的，管理员操作项就是：同意、拒绝。

2、资产状态为：维修的、管理员操作项就是：维修成功、维修失败。

3、同意操作的，资产状态为申请单的结束(stop)状态

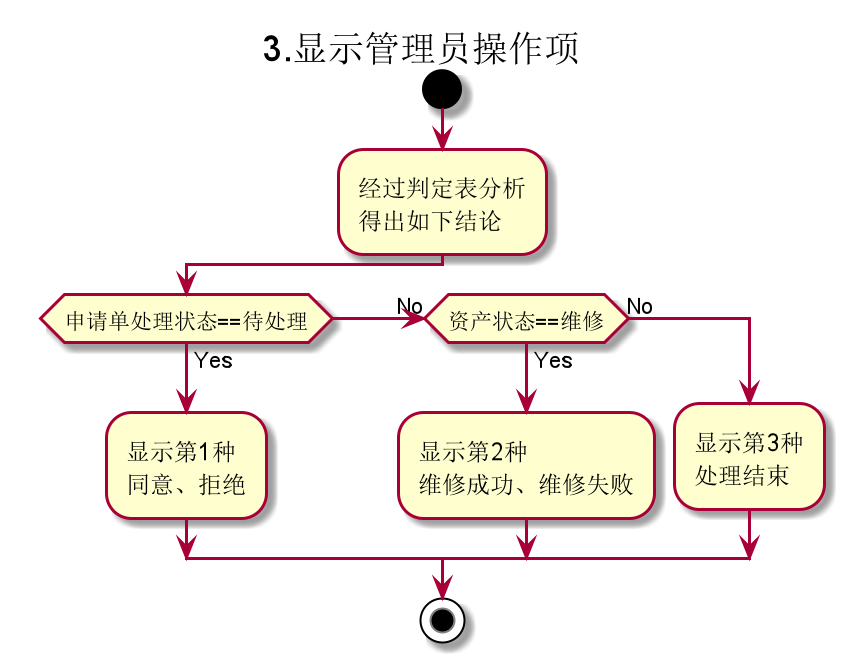
4、拒绝操作的，资产状态为申请单的开始(start)状态

#### 判定表

资产状态的其它状态包括：空闲、使用、借用、报废

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 决策规则号 | | 1 | 2 | 3 | 4 |
| 条件 | 申请单处理状态 | 待处理 | 待处理 | 同意或拒绝 | 同意或拒绝 |
| 资产状态 | 维修 | 其它状态 | 维修 | 其它状态 |
| 备注 | | 维修结束后的状态 |  |  |  |
| 操作 | 同意、拒绝 | √ | √ |  |  |
| 维修成功、维修失败 |  |  | √ |  |
| 处理结束 |  |  |  | √ |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 决策规则号 | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 条件 | 申请单处理状态 | 待处理 | 待处理 | 同意或拒绝 | 同意或拒绝 | 维修成功 | 维修成功 | 维修失败 | 维修失败 |
| 资产状态 | 维修 | 其它状态 | 维修 | 其它状态 | 维修 | 其它状态 | 维修 | 其它状态 |
| 申请单类型维修 | √ | √ | √ | √ | √ | √ | √ | √ |
| 申请单类型其它 |  |  |  |  |  |  |  |  |
| 备注 | | 维修结束后的状态 |  |  |  |  |  |  |  |
| 操作 | 同意、拒绝 | √ | √ |  |  |  |  |  |  |
| 维修成功、维修失败 |  |  | √ |  |  |  |  |  |
| 处理结束 |  |  |  | √ |  |  |  |  |
| 维修结束 |  |  |  |  | √ |  | √ |  |



## 八、变更管理

### （1）、（√）添加申请单类型字段和修改申请单的入库单id为资产id

|  |  |
| --- | --- |
| 变更的目前状况 | 申请单类缺少类型字段，无法确认当前申请单是领用申请，还是维修申请。只能通过肉眼识别申请类型是领用还是维修。 |
| 变更理由 | 添加申请单类型字段，由系统自动判断该申请所属类型。 |
| 变更后的状况 | 完善申请单管理 |
| 风险评估 | 需要重新创建表，或者修改表字段，开发阶段可以。生产阶段需要考虑成本和停机。 |
| 领导审核意见 | 编号、登记和审核变更。编号应有追溯性  同意变更  张三思 |
| 实施变更人员 | 李四 |
| 实施细节 | 修改Application类属性，新增类型属性。  @Entity *//数据库实体对象 自动创建数据表* @Data *//自动创建属性的getter方法和setter方法* **public class** Application {  *//中间属性省略…* **private** String **type**;*//申请单类型 领用申请、维修申请、报废申请、借用申请。* }  申请类型由前端决定，后台不负责申请类型。 |

### （2）、（√）新增需求

|  |  |
| --- | --- |
| 变更的目前状况 | 需求任务数量适当，在可控范围。 |
| 变更理由 | 该需求属于处理请求的扩展，不影响主体框架 |
| 变更后的状况 | 申请类型新增2个，借用申请和归还申请 |
| 风险评估 | 影响很小 |
| 领导审核意见 | 编号、登记和审核变更。编号应有追溯性  同意变更  张三思 |
| 实施变更人员 | 李四 |
| 实施细节 | 新增如下两个需求：  d) 资产管理员可以处理借用请求  e) 资产管理员可以处理归还请求 |

### （3）、（ ）新增判断登录逻辑

|  |  |
| --- | --- |
| 变更的目前状况 | 需求任务数量适当，在可控范围。 |
| 变更理由 | 该需求需要新增用户表，需要判断用户权限，可以完善管理员和用户之间的权限分配 |
| 变更后的状况 | 方便可视化测试 |
| 风险评估 | 影响较大，工作量增加，将延长工期。 |
| 领导审核意见 | 编号、登记和审核变更。编号应有追溯性  不同意，根据项目进度，最后考虑该需求  张三思 |
| 实施变更人员 | 李四 |
| 实施细节 | 暂不实施 |

### （4）、（√）新增入库单类属性——剩余量

|  |  |
| --- | --- |
| 变更的目前状况 | 资产的库存需要通过计算得到。入库单数量-当前使用量 |
| 变更理由 | 每出库一次，剩余量-1，始终检查剩余量>=0的。方便获取剩余数 |
| 变更后的状况 | 方便查询库存剩余量 |
| 风险评估 | 影响不大，只是修改了入库单类结构 |
| 领导审核意见 | 编号、登记和审核变更。编号应有追溯性  同意变更  张三思 |
| 实施变更人员 | 李四 |
| 实施细节 | 修改Application 申请单类属性  @Entity @Data @ToString **public class** AssetRuKuDan {  *//省略其它属性….* **private int remained**;*//入库单中资产剩余数量 = 数量 - 出库次数* }  修改资产入库功能  @GetMapping(**"/add"**) *//添加资产，资产入库* **public** AssetRuKuDan add(@RequestParam String name,@RequestParam **int** amount){ *//接收前端传送过*  *//省略其它代码* assetRuKuDan.setRemained(amount);*//设置剩余数量为入库数量* }  //修改出库，添加入库单剩余量 – 1的代码  @GetMapping(**"/chuku"**) *//添加资产，资产出库* **public** Asset chuku(@RequestParam **long** rukudanId){  //….  *//以下为新增部分v2.0 2020年3月31日10:43:03* asset.setName(ruKuDan.getName());*//设置资产的名称为入库单中的资产名称* **int** remainded = ruKuDan.getRemained();*//获得入库单中的剩余数量* remainded--;*//减少一台* ruKuDan.setRemained(remainded);*//重新赋值剩余数量* **assetRuKuDanRepository**.save(ruKuDan);*//重新保存到数据库中 //以上为新增部分v2.0 2020年3月31日10:43:12*  } |
|  |  |

### （5）、（ ）出库时考虑库存量

|  |  |
| --- | --- |
| 变更的目前状况 | 出库没有考虑剩余量是否为0的情况，存在bug |
| 变更理由 | 该bug必须解决，影响正常使用 |
| 变更后的状况 | 无法出库表示没有空闲资源可用 |
| 风险评估 | 影响很大，必须解决 |
| 领导审核意见 | 编号、登记和审核变更。编号应有追溯性  修改量很小，逻辑简单，让新手练练手。  张三思 |
| 实施变更人员 | 李四 |
| 实施细节 |  |
|  |  |

### （6）、（√）拒绝领用申请时修改资产状态“预定”到“空闲”

|  |  |
| --- | --- |
| 变更的目前状况 | 领用申请让资产状态称为“预定”，拒绝申请时没有改回来 |
| 变更理由 | 导致预定资产越来越多。 |
| 变更后的状况 | 不会有“预定”资产处于“无人管理”状态。 |
| 风险评估 | 影响很大，必须解决 |
| 领导审核意见 | 编号、登记和审核变更。编号应有追溯性  修改量很小，逻辑简单，让新手练练手。注意必须是从预定回到空闲。  张三思 |
| 实施变更人员 | 李四 |
| 实施细节 | */\*\*  \* //同意维修申请，面向使用者的，使用者提出申请，一定是资产是他自己的。  \* 同意维修申请  \** ***@return*** *\*/* @GetMapping(**"/agreeRepair"**) **public** Application applicationAgreeRepair(**long** applicationId){*//申请单号* Application application = **applicationRepository**.findById(applicationId).orElse(**null**);  **if** (application==**null**){**return null**;}*//如果没有找到申请单号就返回null* **long** assetId = application.getAssetId();*//获得申请单中的资产ID* Asset asset = **assetRepository**.findById(assetId).orElse(**null**);  **if** (asset == **null**) {  **return null**;*//没有找到资产* }  String status = asset.getStatus();*//获得资产状态* String type = application.getType();*//获得申请单的类型* **if**(type != **"维修"**){  System.***out***.println(**"申请单类型不是维修，无法同意维修"**);  **return null**;  }  **if** (status != **"已使用"**){*//****TODO 判断资产的使用者是不是申请单的申请人。*** System.***out***.println(**"资产不是已使用状态，无法同意维修"**);  **return null**;  }  asset.setStatus(**"维修"**); *//设置申请单中的资产状态为 维修* **assetRepository**.save(asset);  *//以下处理申请单* application.setStatus(**"同意"**);*//设置申请单状态为 同意* application.setManager(**"黄主任"**);*//****TODO 应该是当前登录的用户（admin）*** application.setResultContent(**"黄主任已同意维修"**);*//****TODO 如果没有写处理意见呢？*** application.setResultDate(**new** Date());*//处理时间为当前时间  //把已经处理的申请单保存到数据库中并返回* **return applicationRepository**.save(application); } |
|  |  |

### （7）、（√）申请单添加两个字段“源状态”和“终状态”

|  |  |
| --- | --- |
| 变更的目前状况 | 申请单没有记录资产是如何转移、往什么方向转移。 |
| 变更理由 | 资产的状态应该只有1个，但是申请单上可以记录资产应该从哪个状态到哪个状态。这样可以追溯资产状态变化的痕迹。 |
| 变更后的状况 | 根据申请单上的状态可以让维修后回到申请前的状态。 |
| 风险评估 | 影响中等，等需求差不多后在解决。 |
| 领导审核意见 | 编号、登记和审核变更。编号应有追溯性  修改量很小，但是修改的位置较多，需要提炼成公共的代码，让老练的程序员来完成该功能。  张三思 |
| 实施变更人员 | 李四 |
| 实施细节 | @Entity *//数据库实体对象 自动创建数据表* @Data *//自动创建属性的getter方法和setter方法* **public class** Application { *//申请单类*  *//…* **private** String **stop**;*//资产停止状态、目标状态* **private** String **type**;*//申请单类型 领用申请、维修申请、报废申请、借用申请、归还申请。* } |
|  |  |

### （8）、（√）取消空闲-已使用中间的“预定”状态

|  |  |
| --- | --- |
| 变更的目前状况 | 领用申请需要单独提交URL，增加了重复工作量。 |
| 变更理由 | 在申请单中标记源状态是空闲，终状态是已使用，只要同意了就进入已使用状态，反正“预定”“空闲”都可以提交领用申请，不都一样么？要想看到多少人申请该资产，可以从申请单中查询呀。 |
| 变更后的状况 | 所有的申请都一律使用application/add，多方便，不用专门区分领用而另外做if判断。 |
| 风险评估 | 影响中等偏上，等需求差不多后在解决。 |
| 领导审核意见 | 编号、登记和审核变更。编号应有追溯性  修改量很大，变更反复不适合团队协作开发，目前特殊情况就一个“领用申请”，如果再发现这种类似的特殊处理的“申请”，就合并处理。  张三思 |
| 实施变更人员 | 李四 |
| 实施细节 | application.setStop(application.getType().equals(**"领用"**)?**"已使用"**:application.getType()); *//****TODO 建议申请单的类型和资产保持同步*** application.setStart(asset.getStatus());*//资产的状态不能让参数决定，应该是由系统内的读取数据为准。* |
|  |  |

### （9）、（√）查看资产基本信息页网址——访问接口变更

|  |  |
| --- | --- |
| 变更的目前状况 | 访问资产基本信息页是 /asset/byId |
| 变更理由 | /asset/byId 属于 /asset/\* 规则，属于受保护范畴，然后又要免登录访问，这就造成了冲突，需要解决冲突。 |
| 变更后的状况 | /asset/byId 返回的是Web网页，不是JSON，可以更改为**/assetById**，符合/\*规则。 |
| 风险评估 | 影响很小。 |
| 领导审核意见 | 编号、登记和审核变更。编号应有追溯性  修改量小，可以在用户登录完成后，改过来。  张三思 |
| 实施变更人员 | 李四 |
| 实施细节 | */\*\*  \* 通过资产ID查询资产视图  \* "http://xchgx.vicp.net/asset/byId?assetId="  \** ***@// TODO: 2020/4/7 变更管理第9条，执行变更。*** *\** ***@param assetId*** *资产ID  \** ***@return*** *\*/* @GetMapping(**"/assetById"**) *//* **public** String assetById(**long** assetId,Model model){  Asset asset = **assetRepository**.findById(assetId).orElse(**null**);  model.addAttribute(**"asset"**,asset);*//准备数据* **return "asset"**;*//返回资产asset视图,数据自动进入到视图* }  index.html同时更改  **new *QRCode***(***qrs***[***i***],**"http://xchgx.vicp.net/assetById?assetId="**+***qrs***[***i***].**id**); |
|  |  |

### （10）、（√）给用户类添加“部门”属性

|  |  |
| --- | --- |
| 变更的目前状况 | 目前用户类主要用来登录，没有其他作用 |
| 变更理由 | 为了完成“管理员可以追踪资产的位置”需求，位置可能包含部门和房间号，此时将用户类中添加“部门”属性，可以直接获得位置信息。 |
| 变更后的状况 | 变更后方便后续查询，可以显示资产更多信息。 |
| 风险评估 | 影响很少。 |
| 领导审核意见 | 编号、登记和审核变更。编号应有追溯性  修改量特别小，但不能太靠后，当有视图已经开发完成后，再添加“部门”属性，容易造成修改量增大，尽可能早点完成。  张三思 |
| 实施变更人员 | 李四 |
| 实施细节 | */\*\*  \* 用户类  \*/* @Entity @Data @ToString **public class** User {  *// 其它属性省略 。。。。* **private** String **department**;*//工作部门* }  2020-4-7 16:03:02 |
|  |  |

### （11）、（√）修复BUG管理员同意申请资产状态无变化

|  |  |
| --- | --- |
| 变更的目前状况 | BUG管理员同意申请资产状态无变化 |
| 变更理由 | 修复bug |
| 变更后的状况 | 修复bug |
| 风险评估 | 影响很少。 |
| 领导审核意见 | 编号、登记和审核变更。编号应有追溯性  修改量特别小，但不能太靠后，当有视图已经开发完成后，再添加“部门”属性，容易造成修改量增大，尽可能早点完成。  张三思 |
| 实施变更人员 | 李四 |
| 实施细节 | @GetMapping(**"/agree"**) **public** Application applicationAgree(**long** applicationId,String result){*//处理同意申请单的方法*  *//省略….*  asset.setUsername(application.getUsername());*//设置使用者*  **assetRepository**.save(asset);*//BUG 修改资产状态要持久化到数据库*  *}* |
|  |  |

### （12）、（√）修改doLogin登录逻辑的返回视图

|  |  |
| --- | --- |
| 变更的目前状况 | 只有经过登录doLogin才能够进入后台，刷新都不行。 |
| 变更理由 | 允许已经登录的用户通过刷新直接进入后台。 |
| 变更后的状况 | 支持数据刷新，保持最新数据内容 |
| 风险评估 | 影响很少。 |
| 领导审核意见 | 编号、登记和审核变更。编号应有追溯性  早点解决。  张三思 |
| 实施变更人员 | 李四 |
| 实施细节 | */\*\*  \* 执行登录判断  \** ***@param user*** *登录表单（含用户名和密码）  \** ***@param request*** *请求对象  \** ***@param model*** *Model模式层  \** ***@return*** *\*/* @PostMapping(**"/doLogin"**) **public** String doLogin(User user, HttpServletRequest request, Model model){  User u = **userRepository**.findByUsername(user.getUsername());  **if** (u == **null**) {  model.addAttribute(**"error"**,**"用户名不存在"**);  **return "login"**;  }  **if** (!u.getPassword().equals(user.getPassword())) {  model.addAttribute(**"error"**, **"密码错误"**);  **return "login"**;  }  request.getSession().setAttribute(**"user"**,u);*//记录下登录成功的标记。  //返回的视图名称，由用户的权限决定  //如果用户是user权限（普通用户），那么就返回user视图  //如果用户是admin权限（管理员），那么就返回admin视图* **return "redirect:/"**+u.getRole();*//返回角色视图，准备好user.html视图和admin.html视图* } |
|  |  |

### （13）、（√）新增访问接口/user和/admin

|  |  |
| --- | --- |
| 变更的目前状况 | 登录已经修复为重定向到/user 和/admin了，此时必须补上/user接口和/admin接口，否则造成404错误 |
| 变更理由 |
| 变更后的状况 |
| 风险评估 | 影响登录过程，紧急 |
| 领导审核意见 | 编号、登记和审核变更。编号应有追溯性  修改量特别小，影响很大，速度解决。  张三思 |
| 实施变更人员 | 李四 |
| 实施细节 | */\*\*  \* 使用者后台  \** ***@param model*** *\** ***@return*** *\*/* @GetMapping(**"/user"**) **public** String user(Model model){   List<Asset> assets = **assetRepository**.findAll();  model.addAttribute(**"assets"**,assets);*//在模型中添加数据* **return "user"**; }  */\*\*  \* 管理员者后台  \** ***@param model*** *\** ***@return*** *\*/* @GetMapping(**"/admin"**) **public** String admin(Model model){   List<Asset> assets = **assetRepository**.findAll();  model.addAttribute(**"assets"**,assets);*//在模型中添加数据* List<AssetRuKuDan> assetRuKuDans = **assetRuKuDanRepository**.findAll();  model.addAttribute(**"rukudans"**, assetRuKuDans);   List<Application> applications = **applicationRepository**.findAll();  model.addAttribute(**"applications"**, applications);  **return "admin"**; } |
|  |  |

### （14）、（√）更新过滤器规则，添加/user和/admin的拦截

|  |  |
| --- | --- |
| 变更的目前状况 | /user和/admin是新增的，不在过滤器的拦截规则中，然而这两个接口对应的分别是使用者的后台和管理员的后台，是必须要经过登录检查的，所以要加入到过滤器规则中。 |
| 变更理由 |
| 变更后的状况 |
| 风险评估 | 影响登录过程，紧急且重要。 |
| 领导审核意见 | 编号、登记和审核变更。编号应有追溯性  修改量小，影响很大，速度解决。  张三思 |
| 实施变更人员 | 李四 |
| 实施细节 | @WebFilter(filterName = **"loginFilter"**, urlPatterns = {**"/rukudan/\*"**,**"/asset/\*"**,**"/application/\*"**,**"/admin"**,**"/user"**}) |
|  |  |

### （15）、（ ）新增用户退出功能

|  |  |
| --- | --- |
| 变更的目前状况 | 没有退出功能，不方便在“未登录”和“已登录”之间切换。 |
| 变更理由 |
| 变更后的状况 |
| 风险评估 | 影响登录过程，紧急且重要。 |
| 领导审核意见 | 编号、登记和审核变更。编号应有追溯性  修改量小，影响很大，速度解决。  张三思 |
| 实施变更人员 | 李四 |
| 实施细节 |  |
|  |  |

### （16）、（ ）统一描述，申请单类型和资产状态一致

|  |  |
| --- | --- |
| 变更的目前状况 | 申请单类型和资产状态一致，这样方便从申请单中获取状态，而不需要第二次处理。 |
| 变更理由 |
| 变更后的状况 |
| 风险评估 | 影响登录过程，紧急且重要。 |
| 领导审核意见 | 编号、登记和审核变更。编号应有追溯性  修改量小，影响较大，尽快解决。  张三思 |
| 实施变更人员 | 李四 |
| 实施细节 | 等待 |
|  |  |

### （17）、（ ）申请单类添加新属性，管理员操作项opertion

|  |  |
| --- | --- |
| 变更的目前状况 | 为了方便视图层渲染，我们在申请单类中添加新属性operation，用来存放该申请单的操作内容。 |
| 变更理由 |
| 变更后的状况 |
| 风险评估 | 尽早处理。 |
| 领导审核意见 | 编号、登记和审核变更。编号应有追溯性  修改量小，影响较大，尽快解决。  张三思 |
| 实施变更人员 | 李四 |
| 实施细节 | 等待 |
|  |  |

### （18）、（ ）删除申请单类的amount数量属性

|  |  |
| --- | --- |
| 变更的目前状况 | 原本是为了提交领用申请时，可以多领几台。结果导致一张单子包含多个资产，然后维修往往都不是全部都能修好。这就给维修的逻辑带来复杂度，不如改成，每次申请只针对1个资产。所有的申请单都面对1个资产。统一数量，减少复杂度。 |
| 变更理由 |
| 变更后的状况 |
| 风险评估 | 尽早处理，已经有现有代码和测试页面中使用了amount属性。 |
| 领导审核意见 | 编号、登记和审核变更。编号应有追溯性  修改量大，影响较大，尽快解决。  张三思 |
| 实施变更人员 | 李四 |
| 实施细节 | 等待 |
|  |  |

### （19）、（ ）处理申请单后资产使用者如何变化

|  |  |
| --- | --- |
| 变更的目前状况 | 维修失败，资产使用者改为null  维修成功，资产使用者为【原资产使用者】。  归还后，资产使用者为null  使用、借用后，资产使用者为申请人 |
| 变更理由 |
| 变更后的状况 |
| 风险评估 | 尽早处理，已经有现有代码和测试页面中使用了amount属性。 |
| 领导审核意见 | 编号、登记和审核变更。编号应有追溯性  修改量大，影响较大，尽快解决。  张三思 |
| 实施变更人员 | 李四 |
| 实施细节 | 等待 |
|  |  |