**理财产品管理系统**

**概要设计说明书**

不想编码就去理财队

修订历史

|  |  |  |  |
| --- | --- | --- | --- |
| 修订日期 | 修订内容 | 修订人 | 版本号 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# 概述

## 系统简述

## 软件设计目标

### 功能性设计目标

#### 最高优先级：

1. 供应商认证
2. 登录
3. 退出
4. 员工账号管理
5. 产品录入
6. 产品评估
7. 入库审批
8. 产品上架
9. 产品展示

#### 次要优先级：

1. 供应商资料管理
2. 黑白名单
3. 产品配置
4. 产品配置入库审批
5. 用户评价

#### 最低优先级：

1. 产品导入
2. 呼叫中心系统
3. 购买产品

### 非功能性设计目标

#### 可用性

|  |  |
| --- | --- |
| 名称 | 描述 |
| 可操作性 | 界面简约清晰，操作便捷，容易上手 |
| 一致性 | 在不同设备上保持数据一致性 |

#### 可靠性

|  |  |
| --- | --- |
| 名称 | 描述 |
| 故障发生周期 | 平均每次故障发生的周期大于1个月 |
| 平均故障修复时间 | 平均每次故障修复的时间小于3天 |
| 安全性 | 恶意用户不能损坏或窃取该系统隐私数据 |

#### 性能

|  |  |
| --- | --- |
| 名称 | 描述 |
| 事务响应时间 | 平均每个事务的响应时间小于1秒 |
| 吞吐量 | 整个系统平均每秒处理大于20个事务 |
| 容量 | 整个系统支持大于50个用户同时操作 |
| 资源消耗 | 内存占用少，磁盘空间占用少，对网络信号强度和网络速度依赖不高 |

#### 可支持性

|  |  |
| --- | --- |
| 名称 | 描述 |
| 编码标准 | utf-8 |
| 类库 | jdk-8 |
| 可移植性 | 使用HTML，CSS，JavaScript和Java进行该网站的开发，保证了该网站的可移植性 |

## 参考资料

a) UML和模式应用（原书第3版）/（美）拉曼（Larman, C.）著；李洋等译.—北京：机械工业出版社，2006.5。

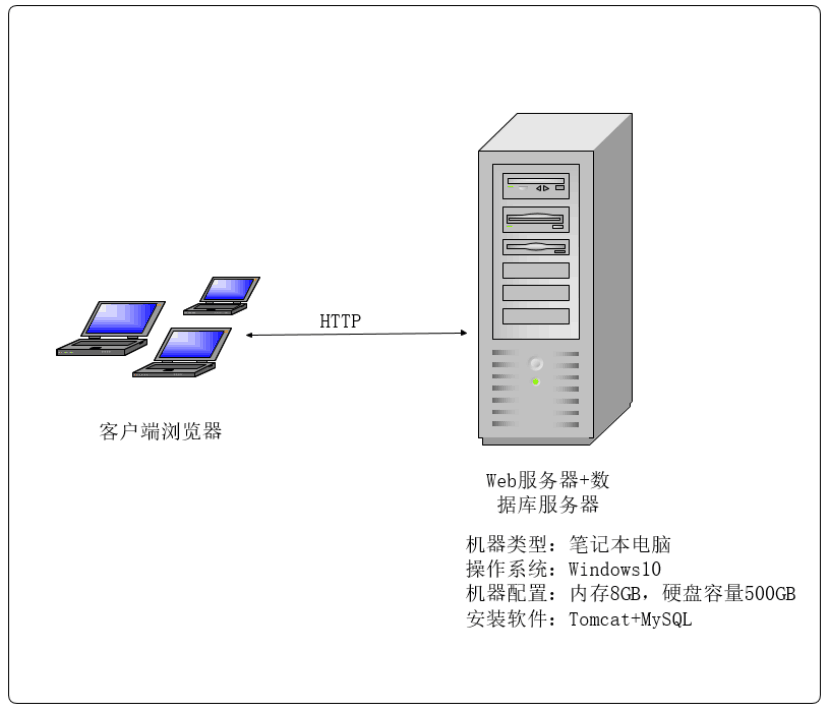
b) IT项目管理（原书第7版）/（美）施瓦尔贝（Schwalbe, K.）著；邢春晓等译.—北京：机械工业出版社，2015.8。

# 设计概述

## 系统结构设计

### 系统逻辑架构4bea0b8dcffcec41cf8d90cd91fe6b9

### 系统物理架构



### 系统逻辑数据模型

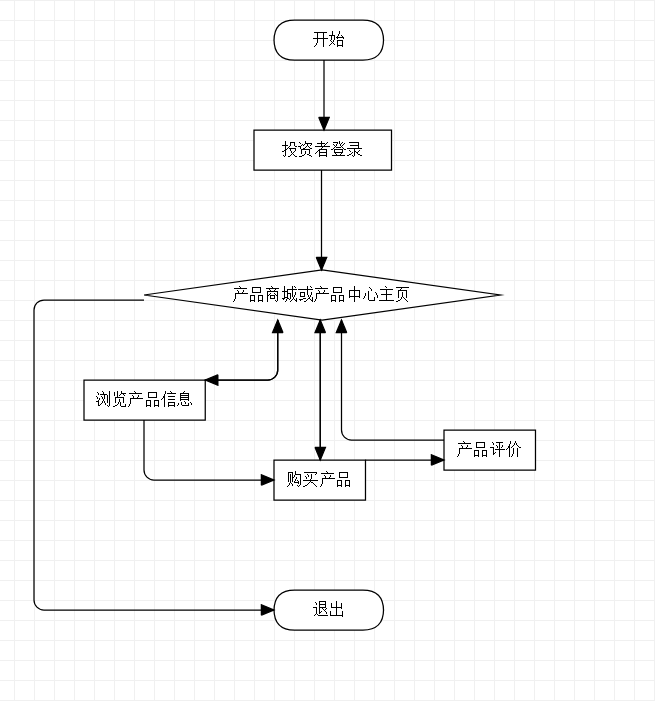
## 系统功能划分与设计

### 子功能-用户评价

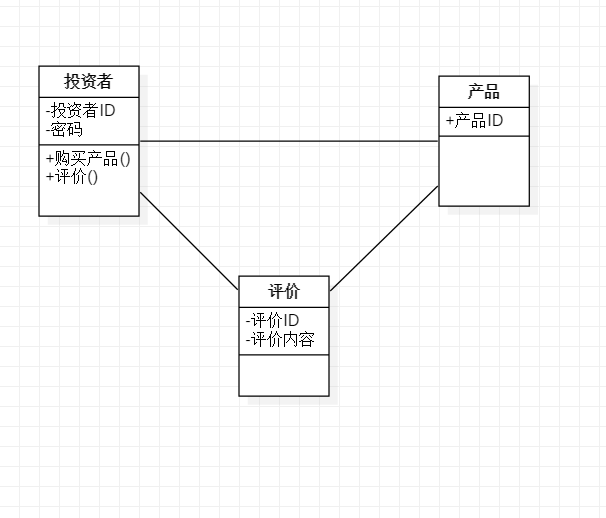
#### 功能描述

投资者进入系统网站, 在产品中心和产品商城购买产品后评价产品

#### 业务流程图



#### 概念类图

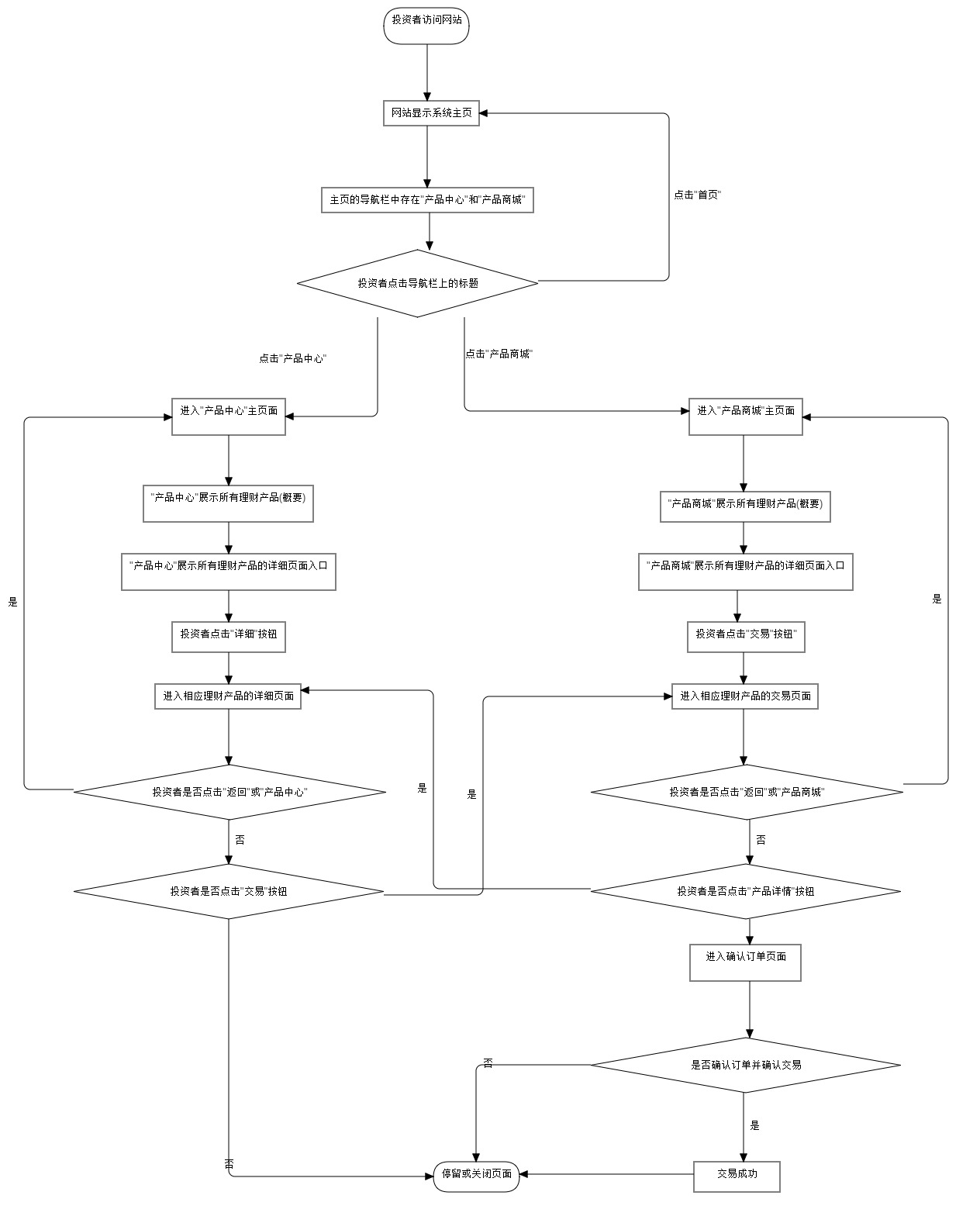


### 子功能-产品展示

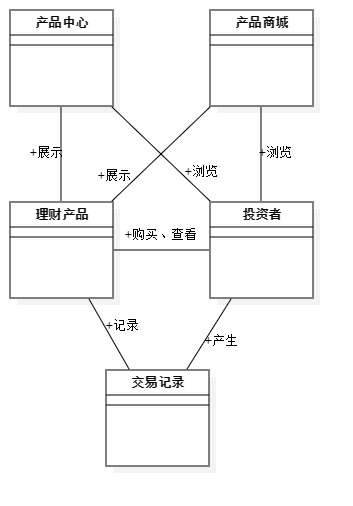
#### 功能描述

投资者进入系统网站, 在产品中心和产品商城中浏览产品和交易产品.

#### 业务流程图



#### 概念类图



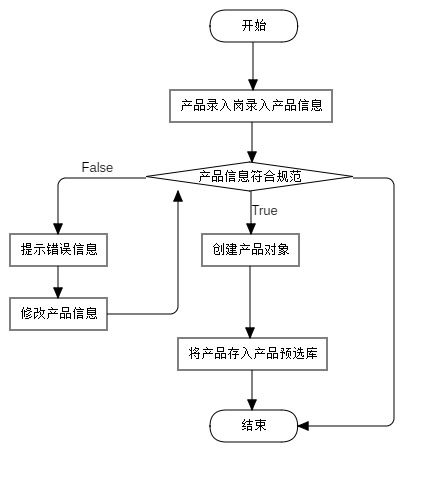
### 子功能-产品录入

#### 功能描述

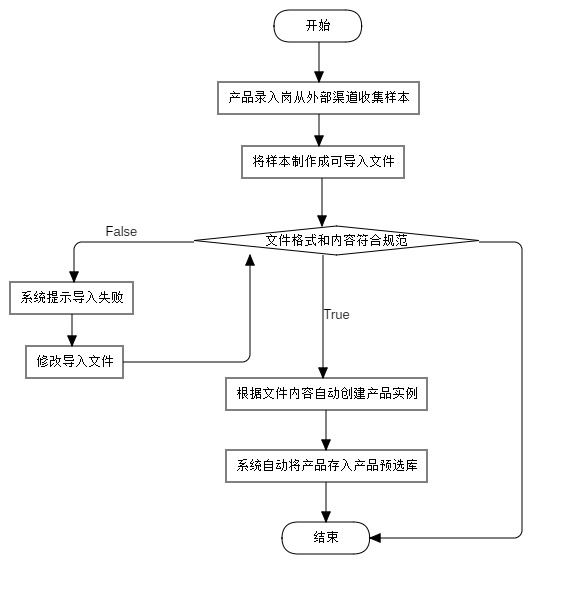
产品录入岗将产品信息录入到系统中

#### 业务流程图

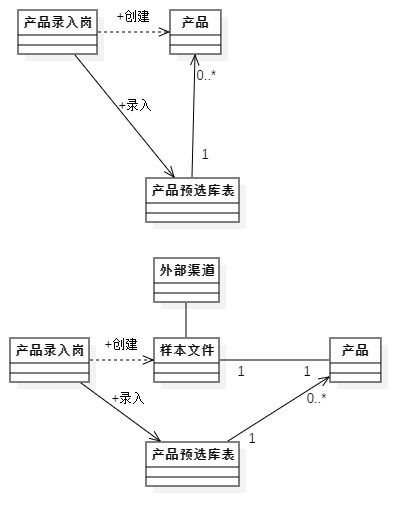
##### 产品录入



##### 产品导入



#### 概念类图

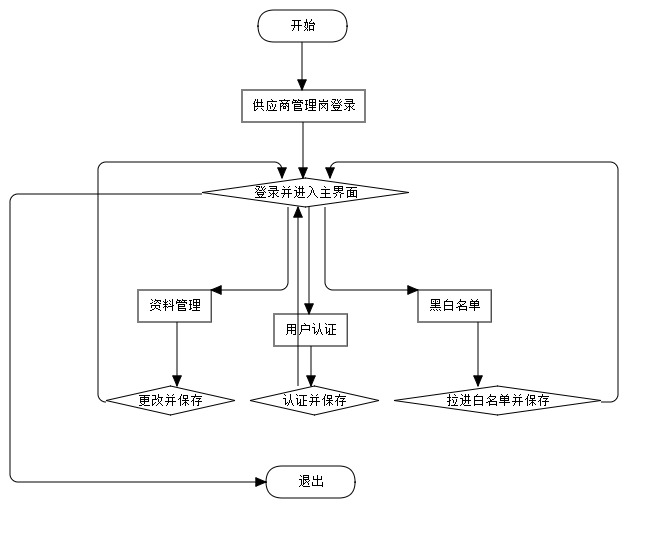


### 子功能-供应商管理

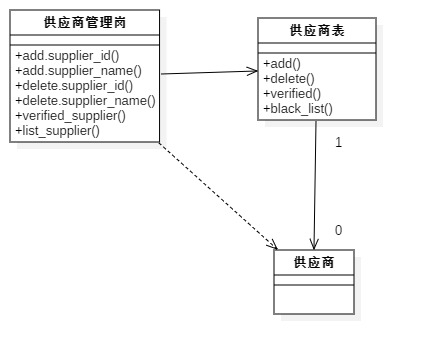
#### 功能描述

供应商管理岗将供应商资料录入到系统中;供应商管理岗将供应商资料进行认证;供应商管理岗将供应商进行黑白名单分类

#### 业务流程图



#### 概念类图

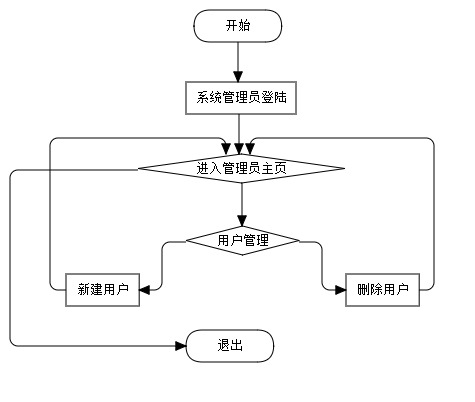


### 子功能-系统管理员模块

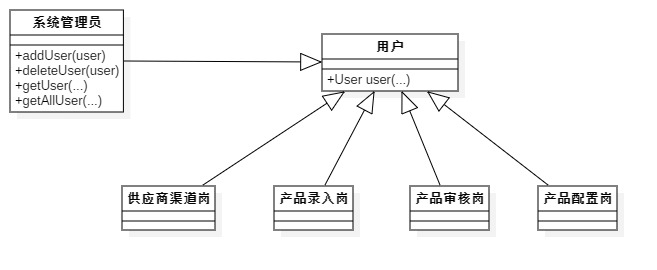
#### 功能描述

管理整个系统的员工账号

#### 业务流程图



#### 概念类图



### 子功能-系统名字

#### 功能描述

#### 业务流程图

#### 概念类图

### 子功能-系统名字

#### 功能描述

#### 业务流程图

#### 概念类图

## 系统接口设计

### 提供给用户的接口

### 系统内部的接口

|  |  |  |
| --- | --- | --- |
| **接口名称** | **接口描述** | **所属模块** |
| 获取产品信息 | 从标准库里获取已上架的产品信息列表 | 产品评价 |
| 获取评价 | 获取与产品有关的评价内容 | 产品评价 |
| 添加评价 | 添加用户的评价内容到数据库中 | 产品评价 |
| 删除评价 | 删除数据库中的评价 | 产品评价 |
| 购买产品 | 添加用户购买产品的信息到数据库中 | 产品评价 |
| 获取产品中心列表 | 获取产品中心产品列表 | 产品展示 |
| 获取产品商城列表 | 获取产品商城产品列表 | 产品展示 |
| 获取产品详细信息 | 获取单个理财产品详细介绍信息 | 产品展示 |
| 获取产品购买信息 | 获取单个理财产品的购买相关信息 | 产品展示 |
| 购买产品 | 投资者购买理财产品 | 产品展示 |
| 上传产品信息 | 输入产品信息的页面，将产品信息传输到产品录入岗 | 产品录入 |
| 导入产品信息 | 导入产品信息的页面，选择文件，将文件信息传输到产品录入岗 | 产品录入 |
| 获取用户列表 | 从数据库中获取所有用户的信息 | 系统管理员 |
| 获取单个用户 | 从数据库中获取所查询的单个用户的信息 | 系统管理员 |
| 新建用户 | 添加新的用户信息到数据库中 | 系统管理员 |
| 删除用户 | 从数据库中删除选中的用户 | 系统管理员 |
| 获取供应商名称 | 从数据库获取供应商的名称 | 供应商管理模块 |
| 获取供应商ID | 从数据库获取供应商的ID | 供应商管理模块 |
| 获取供应商的认证状态 | 从数据库获取供应商的认证状态 | 供应商管理模块 |
| 获取供应商的黑白名单 | 从数据库获取供应商的黑白名单状态 | 供应商管理模块 |
| 获取供应商档案 | 从数据库获取供应商的档案 | 供应商管理模块 |
| 更改供应商名称 | 更改数据库中的供应商名称 | 供应商管理模块 |
| 更改供应商ID | 更改数据库中的供应商ID | 供应商管理模块 |
| 拉黑供应商 | 向数据库申请将对应的供应商拉入黑名单状态 | 供应商管理模块 |
| 拉白供应商 | 向数据库申请将对应的供应商拉入白名单状态 | 供应商管理模块 |
| 认证供应商 | 向数据库申请将对应的供应商拉入已认证状态 | 供应商管理模块 |

### 系统外部的接口

无

## 约束和假定

### 约束:

#### 时间日程：

|  |  |
| --- | --- |
| 日期 | 里程碑 |
| 第一周6月6日 | 项目需求分析 |
| 第二周6月13日 | 概要设计、详细设计 |
| 第三周6月20日 | 编码 |
| 第四周6月29日 | 编码、编码测试 |
| 第五周7月6日 | 测试 |
| 第六周7月11日 | 验收发布 |

#### 项目质量：

|  |  |
| --- | --- |
| 质量特性 | 要求 |
| 正确性 | 该系统必须正常运行并实现所有的预期功能 |
| 可靠性 | 在硬件发生故障、输入的数据无效或操作错误等意外环境下，系统能够在一定范围内承受住压力而不至于崩溃 |
| 高效性 | 该系统不消耗过多的硬盘空间与内存 |
| 安全性 | 未经授权的人无法侵入系统内破坏系统或窃取隐私数据 |
| 易用性 | 用户可以在短时间内学会熟练操作该系统 |
| 风险性 | 在预定的成本预算和期望的时间内完成系统的开发，并且让用户满意 |

#### 资金预算：

|  |  |
| --- | --- |
| 成本项 | 预算 |
| 人工成本 | 10000元 |
| 硬件成本 | 20000元 |
| 软件成本 | 20000元 |
| 场地成本 | 10000元 |
| 总计 | 60000元 |

### 假设:

1. 设计阶段能按时顺利完成
2. 核心编码人员都能在编码阶段开始前掌握基础JavaWeb开发技术以及Spring、Springmvc、Mybatics框架的搭建与使用
3. 各里程碑事件能按时完成，进度不滞后。

## 非功能性设计

### 可用性设计

1. 配备用户操作手册，系统配备指导新用户操作的教学流程，总结一些常见的问题并给出建议。
2. 采用迭代设计，从设计的早期阶段通过评估逐步优化设计。在评估的过程中能够使设计人员和开发人员整合用户和客户例如投资者、供应商、审核录入配置人员等的反馈，直到系统能达到一个可接受的可用性水平。另外确保可用性的首选方法是测试系统上的实际用户。实现高可用性需要将设计工作重点放在系统的预期时的最终用户身上。确定主要用户是谁，他们的工作方式以及必须完成的任务有很多种方法。

### 可靠性设计

1. 系统维护员不定期检查系统，当系统出现问题时及时维护。
2. 设置备份服务器，避免系统因服务器问题全面瘫痪。
3. 服务器上系统和软件的版本尽可能选择稳定且新的版本，以减少软件或系统漏洞，关闭服务器不用的端口，及时更新安全补丁。
4. 后台开发时加强进行安全性测试，包括密码加密、使用验证码等。
5. 常做数据备份，防止数据丢失。
6. 管理员本身的密码尽可能复杂，并且通常访问数据库或后台的时候尽可能使用低权限的账户。

### 性能设计

1. 根据需求配置更高性能的服务器，提升用户访问体验。
2. 可以增加带宽大小，保证多用户并发访问，防止系统因访问量过大响应迟缓。
3. 可以考虑将服务器搭建在云服务器上，依靠云服务保证网站服务器性能方面达到最大保障。

### 可支持性设计

1. 严格按照编码标准和编码规范进行开发，采用模块化设计降低耦合性，使系统更具有更高的移植性。