**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：供应商认证，登录，退出，员工账号管理，产品录入，产品评估，入库审批，产品上架，产品展示

次要优先级：供应商资料管理，黑白名单，产品配置，产品配置入库审批，用户评价

最低优先级：产品导入，呼叫中心系统，购买产品

### 非功能性设计目标

### 1.2.2.1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 |

### 1.2.2.2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 |

### 1.2.2.3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 |

### 1.2.2.4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

# 术语表

统计一些在项目过程中可能会发生歧义、一定要清晰定义的术语

# 设计概述

## 系统结构设计

### 系统逻辑架构

### 系统物理架构

### 系统逻辑数据模型

## 系统功能划分与设计

### 子系统-模块名字

#### 功能描述

#### 各个用例的流程图

#### 概念类图

## 系统接口设计

### 提供给用户的接口

各个网页

### 系统内部的接口

各个模块在工作的时候需要用到的接口,不需要具体到参数类型等详细内容,但是起码给接口一个名字

### 系统外部的接口

无

## 约束和假定

约束:

时间日程

项目质量

资金预算

假设:

比如说我们都是全栈工程师

项目三个约束条件(Triple Constraint)：时间日程（项目周期）、性能规规（质量）、以及资金（成本费用）预算。项目的假设没找到具体定义。

约束条件和假设是可以贯穿整个项目的。比如，项目必须在10月份完成（约束条件），预算是100万（约束条件）。我们提供5个熟练的队员（假设有5个人，假设他们是熟练的）。到后期，5个人来了4个（这时就不是假设了，是问题了）。4个人中，还有一个菜鸟（又成问题了）。假设也是风险的来源，如果不成立，就会转为问题。

## 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。