软件需求说明书

# 引言

## 项目背景

个人博客系统主要用于发表个人博客，记录个人生活日常、学习心得以及技术分享等，供他人浏览、查阅、评论等。个人博客往往作为技术人员的个人知识发布管理平台，以便于记录某项技术使用教程，也有利于与同行业的从业人员进行交流。

本项目致力于构建一个个人博客平台, 每个注册用户都可以发布,管理自己的博客, 也可以浏览,评论他人的博客. 系统还包含了搜索功能和关注功能, 用户可以方便地寻找到自己需要的功能, 并及时得到来自关注博主的消息. 此外, 由于博文采用富文本形式展现, 平台还包含了文件存储的功能, 用户可以将上传的图片嵌入到自己的博客中, 或上传资源分享给别人。

## 用户特征和水平

本软件主要面向计算机技术人员, 计算机专业的学生、从业者、爱好者。编写博客的人应该有基础的markdown标记语言的使用技能，愿意分享技术。

# 项目概述

## 软件系统描述

简要介绍该软件开发的意图、应用目标、作用范围以及其他应向读者说明的有关该软件开发的背景材料。

## 用户群体

列出本软件系统的最终用户的特点，充分说明操作人员和维护人员的教育水平和技术专长，以及本软件的预期使用频度。

## 功能描述

采用用况图描述软件系统功能，在绘制用况图的基础上，需给出每个用况的具体说明。

|  |  |  |  |
| --- | --- | --- | --- |
| **用况编号：** | | **用况名：** | **作者：** |
| **用况描述：** | | | |
| **执行者** |  | | |
| **相关用况** |  | | |
| **前置条件** |  | | |
| **后置条件** |  | | |
| **基本路径** |  | | |

# 功能需求（本次课程不做要求）

采用结构化分析方法对软件进行需求分析，使用数据流图、数据字典和加工小说明对功能需求进行规约。先给出顶层数据流图以及进一步分解得到的各层数据流图，进而对最底层数据流图的每一个加工（代表软件的一项功能）给出涉及的数据字典和加工小说明。

|  |  |
| --- | --- |
| **需求名** | 具体加工的编号和名字 |
| **功能描述** | 描述功能要达到的目标、所采用的方法和技术。 |
| **输入数据** | 采用数据字典，详细描述该功能的所有输入数据，如输入源、数量、度量单位、时间设定和有效输入范围。 |
| **加工** | 定义输入数据和中间参数，以获得预期输出结果的全部操作。 |
| **输出数据** | 采用数据字典，详细描述该功能的所有输出数据，如输出目的地、数量、度量单位、时间关系、有效输出范围、非法值的处理和出错信息。 |

# 非功能需求

## 性能需求

本节具体说明软件、或人与软件交互的静态或动态数值需求。静态数值需求可以能包括支持的终端数、支持并行操作的用户数、处理的记录数、表的大小等。动态数值需求可能包括欲处理的事务和任务的数量，以及在正常情况下和峰值工作条件下一定时间周期中处理的数据总量。所有这些需求都必须用可以度量的术语来叙述。

1. 系统必须有能力支持1000个以上的并发用户
2. 系统必须有能力支持5000个以上的用户注册
3. 系统不限制发布博文的字符数量
4. 系统中博客管理信息的生效时间应该应该小于1秒，最大生效时间应该小于5秒
5. 对于每个用户的访问，平均响应速度应该小于1秒，最大响应时间应该小于5秒
6. 在用户搜索文章时，系统返回搜索结果的时间应该小于1秒，最大时间应该小于5秒

为做到这一点, 我们会应用到如下技术:

1. 微服务架构. 微服务的架构可以方便地横向扩展瓶颈功能;
2. Redis缓存: 对于热点功能, 我们可以考虑使用redis作为缓存, 降低过多的数据库IO操作;
3. JMeter: 性能测试工具, 可用于找出性能瓶颈所在。

## 外部接口需求

### 用户接口

提供用户使用软件产品时的接口需求。

### 硬件接口

给出软件产品和系统硬件之间每一个接口的逻辑特点。

### 软件接口

指定需使用的其他软件产品，以及同其他应用系统之间的接口。

### 通信接口

指定各种通信接口，如局部网络的协议等。

## 设计约束

描述设计约束受其他标准和硬件限制等方面的影响。

1. 系统设计过程中必须使用Java语言编写系统内容，主页推荐算法模块可以采用Python实现，非必要不使用其他语言以免造成不同微服务之间在Nacos的注册问题。
2. 系统设计时需考虑在单机器部署时能够响应1000次/秒的首页响应请求，在多机器部署的时候能响应至少5000次/秒的首页响应请求。并且任取10秒，一个特定应用所消耗的可用计算能力平均不超过50%。
3. 系统设计时需前后端分离且所采用技术需为业界前沿技术，架构需采用微服务架构便于后续迭代和维护，不得因为方便而使用业界过期或是存在缺陷的技术进行前后端设计。
4. 系统在部署时应部署在Linux环境，因而设计时必须考虑实际生产环境和开发环境不同所带来的问题并进行处理。
5. 系统设计时前后端通信应该采用https加密形式进行，对于敏感信息应当在设计时考虑安全性问题。
6. 系统在设计时需考虑用户使用问题，应当以简洁明了为主。降低用户的使用和学习成本，且重要功能主题鲜明，界面美观。
7. 系统在设计时应该考虑容灾问题，具备出现一定故障时能够自动报警或不影响其他模块正常运行，不影响数据库和缓存的准确性。

## 质量属性

|  |  |
| --- | --- |
| **属性名称** | **详细要求** |
| 性能及效率 | 当系统在单个机器上部署时, 使集群至少能够响应1000次/秒的首页请求响应; 在多个节点上部署时, 能够响应5000次/秒的首页请求响应; 所有界面应该在5秒之内完成界面框架的加载, 7秒内填充来自后端的数据 对于搜索功能, 响应时间不应超过5秒. |
| 易用性, 清晰性 | 系统界面设计应与主流博客平台相似, 降低用户迁移的学习成本; 从页面设计上, 界面的主次设计鲜明, 主要部分展示数据, 次要部分包含全面的跳转链接. 用户上手时间不应超过5分钟, 在15分钟内了解绝大多数功能. |
| 安全性 | 前后端采用https加密; 对于用户敏感信息(如密码等)采用加密算法. |
| 可扩展性 | 后端API将搜索,热搜,排序等功能抽象, 可以在工期足够时采用不同的算法实现功能. |
| 可移植性 | 系统采用docker部署, 因此可以在任意x86-64架构的linux系统(包括WSL)节点上部署. |
| 健壮性 | 微服务节点由nacos管理, 节点失败重启时间不应超过10秒; 失败不应影响到持久化存储的数据, 尽量避免影响到缓存数据(如redis). |