**<轻量博客引擎>**

**系统设计说明书**

**团队：菜鸟驿站 小组**

日期：2025年6月18号

目录

[**1.引言** 4](#_Toc201102034)

[**1.1. 编写**目的 4](#_Toc201102035)

[**1.2. 项目范围** 4](#_Toc201102036)

[**1.3. 术语定义** 5](#_Toc201102037)

[**1.4. 引用文件** 5](#_Toc201102038)

[**2. 总体描述** 5](#_Toc201102039)

[**2.1. 产品背景** 5](#_Toc201102040)

[**2.2. 产品描述** 6](#_Toc201102041)

[**2.2.1 系统接口** 6](#_Toc201102042)

[**2.2.2 用户界面** 7](#_Toc201102043)

[**2.2.3 硬件接口** 8](#_Toc201102044)

[**2.2.4 软件接口** 8](#_Toc201102045)

[**2.2.5 通信接口** 8](#_Toc201102046)

[**2.2.6 内存约束** 8](#_Toc201102047)

[**2.3. 产品功能** 8](#_Toc201102048)

[**2.5. 应用目标与作用范围** 9](#_Toc201102049)

[**2.6. 约束** 9](#_Toc201102050)

[**2.7. 假设与依赖关系** 9](#_Toc201102051)

[**3. 具体需求** 10](#_Toc201102052)

[**3.1外部接口** 10](#_Toc201102053)

[**3.2. 功能需求** 10](#_Toc201102054)

[**3.2.1 用例描述（Use Case）** 10](#_Toc201102055)

[**3.2.2 顺序图** 11](#_Toc201102056)

[**3.2.3 分析类图（逻辑建模）** 11](#_Toc201102057)

[**3.2.4 功能模块划分** 12](#_Toc201102058)

[**3.3. 性能需求** 12](#_Toc201102059)

[**3.3.1. 精度需求** 12](#_Toc201102060)

[**3.4 设计约束** 13](#_Toc201102061)

[**3.4.1 标准依从性** 13](#_Toc201102062)

[**3.5软件系统属性** 13](#_Toc201102063)

[**3.5.1 观感需求** 13](#_Toc201102064)

[**3.5.2 易用性需求** 13](#_Toc201102065)

[**3.5.3可执行性需求** 14](#_Toc201102066)

[**3.5.4安全性** 14](#_Toc201102067)

[**3.5.5 系统完整性** 14](#_Toc201102068)

[**3.5.6 可扩展性与可维护性** 14](#_Toc201102069)

**变更履历**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **版本** | **修订章节与内容** | **变更原因** | **修订人** | **审核人** | **修订日期** |
| **1** | **1.0** | **全文** | **建立初稿** | **杜嘉康** | **杜嘉康** | **2025-05-18** |
| **2** | **1.1** | **性能需求添加** | **内容修订** | **覃昶** | **杜嘉康** | **2025-06-18** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

**说明：“变更原因”主要是分为：**

1. **建立初稿**
2. **内容修订**
3. **正式发布**

**1.引言**

**1.1. 编写**目的

本说明文档旨在系统性地阐述博客系统在系统架构、功能模块、数据库结构、接口设计、权限机制等方面的技术实现方案。通过本说明书，开发人员能够统一系统设计思路，明确各模块之间的接口与数据流，为后续编码开发、测试验证和系统部署提供技术依据。本说明书适用于：

 系统架构设计人员

 后端实现开发人员

 数据库设计与维护人员

 前端开发人员（参考 API 交互部分）

 测试工程师

 项目维护与运维团队

**1.2. 背景**

本系统为一个基于 Rust 编程语言开发的轻量级博客系统，旨在满足技术博主、独立创作者及学习者对内容管理的核心需求。通过部署该系统，用户可以自由发布博客内容、管理分类与标签、浏览归档记录、进行评论互动等。

系统以高性能、高安全性、低资源消耗为目标，采用现代化 Web 框架 Axum 构建服务端 API，结合 SeaORM 实现数据库访问，所有功能模块围绕 RESTful 设计，方便后期前端集成或移动端接入。

该系统适合作为初中型博客平台的技术选型模板，也适合作为学习 Rust Web 开发的入门与实战项目。

**1.3. 术语定义**

|  |  |
| --- | --- |
| **术语** | **定义** |
| Rust | 一门注重性能和内存安全的系统编程语言，由 Mozilla 研究团队主导开发，适用于高并发、嵌入式、WebAssembly 和后端服务等场景。 |
| Axum | Rust 社区中用于构建 Web 应用的高性能异步框架，基于 Tokio 异步运行时和 Tower 服务抽象，设计简洁、类型安全、模块化强。 |
| SeaORM | Rust 中的异步 ORM 框架，支持自动实体生成、灵活查询构建和数据库迁移，适用于中大型项目的持久层开发。 |
| JWT | 一种开放标准（RFC 7519），用于以安全、紧凑的方式在客户端与服务器之间传输身份认证信息。 |
| SQLite | 嵌入式轻量级关系型数据库，支持 ACID 事务，适用于资源受限的场景或开发阶段的本地存储。 |

**1.4. 引用文件**

[1] Steve Klabnik, Carol Nichols. *The Rust Programming Language*（Rust官方权威书籍）  
[2] The Axum Book. https://docs.rs/axum  
[3] SeaQL. *SeaORM 官方文档*：https://www.sea-ql.org/SeaORM/  
[4] SQLite. *SQLite Documentation*：https://www.sqlite.org/docs.html

**2. 总体设计**

**2.1. 需求规定**

本系统根据不同角色的使用需求，划分为“管理员端”与“普通访客端”两种角色权限，对功能进行详细规定如下：

**管理员功能需求**

管理员可通过登录后台管理系统完成以下操作：

 使用用户名与密码登录系统（JWT 鉴权）

 发布新博客文章

 编辑/修改已发布博客

 删除博客

 为博客设置分类

 为博客设置标签

 根据标题、分类、标签筛选博客

 创建新的分类

 修改已有分类

 删除分类

 根据名称查询分类

 创建新的标签

 修改已有标签

 删除标签

 根据名称查询标签

 管理用户评论（删除、审核等，预留接口）

**普通访客功能需求**

普通未登录用户可访问以下功能：

 分页浏览博客列表

 快速查看博客最多的前 6 个分类

 查看所有分类及其对应博客

 快速查看被最多博客引用的前 10 个标签

 查看所有标签及其对应博客

 按分类或标签查看博客列表

 按时间归档方式浏览历史博客（年度、月份）

 浏览单篇博客内容详情

 关键字全文搜索博客

 提交对博客的评论（需登录）

**2.2. 运行环境**

|  |  |
| --- | --- |
| **项目** | **配置描述** |
| 操作系统支持 | Windows 10/11（64位）、Ubuntu 20.04+、MacOS |
| 数据库 | 默认：SQLite3（单文件部署） 或 MySQL 8.0（生产环境推荐） |
| 后端语言环境 | Rust ≥ 1.70，支持 Cargo 包管理 |
| 依赖框架 | Axum（HTTP框架）、SeaORM（数据库ORM）、Serde（序列化）等 |
| 数据传输格式 | JSON（RESTful 交互） |
| 身份认证方式 | JWT（JSON Web Token），支持 HTTP Header 中 Authorization 鉴权 |
| 运行方式 | 后端可作为独立 HTTP 服务部署，或集成至 Docker 容器环境 |
| 推荐部署环境 | Linux + systemd（或 PM2/Runit） |

**2.3. 系统体系结构设计**

系统采用前后端分离架构，后端基于 Rust 的 Axum 框架构建异步 Web 服务，前端可基于 React/Vue 等任意技术栈接入。系统架构分层如下：

**2.4. 功能模块概览**

**3. 详细设计**

**3.1系统安全与权限设计**

本系统采用 **基于 JWT 的用户认证机制**，结合 Axum 的中间件体系实现权限控制，确保系统安全、接口访问清晰。

 所有需要登录权限的接口统一通过 Authorization: Bearer <Token> 鉴权头传入，服务端在中间件中解析并校验 token 有效性；

 访客用户（未登录）可访问博客浏览、分类、标签等只读接口；

 管理员用户需登录后获取 JWT，才能访问博客的发布、修改、删除、分类与标签管理等敏感操作；

 登录信息（密码）采用 bcrypt 加密存储，而非 MD5，确保现代密码安全性；

 系统部署推荐配置 HTTPS + 防火墙（云安全组），过滤非法端口及 IP 访问；

 未来可接入限流、中间件审计与 CSRF 防护策略以提升安全性。

**3.2. 用例图**

**3.3. 设计类图**

**3.4 ER分析图**

**3.5设计约束**

**3.5.1 软件约束**

 支持主流操作系统（Linux, Windows, Mac）

 后端依赖 Rust ≥1.70 + Cargo

 数据传输格式为 JSON

 API 接口需遵循 RESTful 规范

 浏览器兼容性要求：Chrome ≥ 90，Edge ≥ 90，Firefox ≥ 85

**3.5.2 硬件约束**

 服务器最低配置：1核CPU + 512MB RAM + 1GB 硬盘

 支持部署在云主机（如腾讯云、阿里云）

**3.6系统属性说明**

**3.6.1易用性**

 后台采用极简风格 UI，方便非专业人员上手操作；

 Markdown 编辑器内置预览功能，实时渲染内容；

 访客界面结构清晰，支持快捷搜索与跳转；

 支持键盘快捷键（如 Ctrl+S 保存草稿等）提升效率。

**3.6.2可执行性**

 首页加载时间控制在 300ms 内；

 博客详情页首屏加载时间小于 500ms；

 SQLite 模式下支持 1000 条博客数据快速分页；

 服务器接口平均响应时间小于 100ms（不含渲染）。

**3.6.3 安全性**

 所有后台接口需携带有效 JWT；

 密码加密采用 bcrypt 算法；

 支持日志记录、请求追踪（结合 Tracing 库）；

 未来可集成限流、中间件权限注解。

**3.6.4 完整性**

 所有博客数据可导出为 Markdown 文件；

 标签分类关系具备完整性检查；

 数据库设置级联删除保护；

 系统支持定期备份数据库快照。

**3.6.5 可扩展性与可维护性**

 所有业务模块通过模块化 crate 管理（controller/service/model 路径隔离）；

 配置支持 dotenv + config crate，便于环境适配；

 可接入 Redis、Elasticsearch 等服务以提升性能；