目录

[1. 引言 2](#_Toc198287787)

[1.1 目的 2](#_Toc198287788)

[1.2 范围 2](#_Toc198287789)

[1.3 术语与缩略词 2](#_Toc198287790)

[2. 系统概述 2](#_Toc198287791)

[2.1 系统环境 2](#_Toc198287792)

[2.2 模块划分 2](#_Toc198287793)

[3. 设计观图 3](#_Toc198287794)

[3.1 逻辑观图 3](#_Toc198287795)

[3.2 运行观图 (登录流程) 3](#_Toc198287796)

[3.3 部署观图 3](#_Toc198287797)

[4. 质量属性 3](#_Toc198287798)

[4.1 性能 3](#_Toc198287799)

[4.2 可维护性 3](#_Toc198287800)

[4.3 安全性 3](#_Toc198287801)

[4.4 可扩展性 4](#_Toc198287802)

[5. 架构决策 4](#_Toc198287803)

[6. 风险分析与应对 4](#_Toc198287804)

[7. 参考文献 4](#_Toc198287805)

# 1. 引言

## 1.1 目的

本文档用于详细描述博客网站系统的软件体系结构，以指导开发团队在需求分析、编码实现、测试和维护过程中有效执行，确保系统担保开发成本低、性能稳定、维护方便。

### 1.2 范围

本系统是一个基于B/S架构的响应式博客平台，支持用户登录/注册，文章发布与管理，评论交互，应用统计和后台管理功能，面向普通用户、注册作者和管理员提供多级别权限的不同功能。

### 1.3 术语与缩略词

- JWT: JSON Web Token，用于无状态认证  
- RESTful API: 基于资源组织的接口设计样式  
- ORM: Object-Relational Mapping，对象关系映射

# 2. 系统概述

## 2.1 系统环境

前端技术：React.js + Bootstrap 5  
后端技术：Node.js + Express.js  
数据库：MySQL  
保存并可选的Redis用于缓存  
部署环境：Docker容器化 + AWS EC2

## 2.2 模块划分

|  |  |
| --- | --- |
| 模块 | 功能描述 |
| 用户认证模块 | 登录/注册，JWT认证，密码加密 |
| 文章管理模块 | CRUD操作，Markdown编辑器，添加图片 |
| 评论模块 | 用户评论，管理员删除评论 |
| 统计分析模块 | 访问量统计，热门博客排行 |
| 后台管理模块 | 用户管理，评论/文章审核 |

# 3. 设计观图

## 3.1 逻辑观图

用户操作 --> 前端UI --> API接口 --> 商业逻辑层 --> DAO --> MySQL/Redis

## 3.2 运行观图 (登录流程)

1. 用户输入账号/密码  
2. 前端发送 POST /api/login 请求  
3. 后端校验密码（BCryptHash）  
4. 生成JWT Token，返回前端  
5. 前端保存Token，用于后续请求

## 3.3 部署观图

前端文件通过Nginx部署  
后端服务器Docker容器运行Node.js应用  
MySQL + Redis安装在不同实例/服务器

# 4. 质量属性

## 4.1 性能

首页加载 < 1.2s  
登录接口支持2000TPS  
文章列表支挅5000QPS

## 4.2 可维护性

模块化设计，每个模块独立开发  
提供完整的API文档和数据库ER图

## 4.3 安全性

用户密码BCrypt加密存储  
HTTPS数据传输加密  
后台管理接入多因素认证  
防XSS/SQL注入/CSRF

## 4.4 可扩展性

模块化开发，方便后期添加资讯推送/付费订阅等功能  
支持服务组织扩展，通过增加节点实现无中断扩展

# 5. 架构决策

|  |  |
| --- | --- |
| 决策项 | 原因分析 |
| SpringBoot/Express | 运行维护成本低，配置简单 |
| RESTful API | 规范化设计，容易集成外部系统 |
| MySQL + Redis | 实现性能与一致性并量，降低总访问时间 |
| Docker + CI/CD | 断电恢复、快速部署，减少人为操作风险 |

# 6. 风险分析与应对

|  |  |  |
| --- | --- | --- |
| 风险级别 | 情况 | 应对策略 |
| 三级 | 性能压力大 | Redis缓存 + CDN加速 |
| 二级 | 时间估算误差 | Sprint预留定量空间应对 |
| 一级 | 数据丢失（未备份） | 每日安排备份到云存储 |
| 一级 | 需求蚂蚁藏穿 | 采用需求冻结策略，系统版本约束 |

# 7. 参考文献

《IEEE 1471-2000》