{AchieveIt}  
软件架构设计说明书

|  |  |  |
| --- | --- | --- |
| 文件状态：  [ ] 草稿  [√] 正式发布  [ ] 正在修改 | 文件标识： |  |
| 当前版本： | 1.0 |
| 作 者： | 倪事通 |
| 完成日期： | 2020-03-15 |

版 本 历 史

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本/状态 | 作者 | 参与者 | 起止日期 | 备注 |
| v0.5 | 倪事通 | 万洋 | 2020/3/2-2020/3/8 | 草稿版 |
| v1.0 | 倪事通 | 万洋 | 2020/3/9-2020/3/16 | 正式版 |
|  |  |  |  |  |

目录

[图目录 3](#_Toc18960)

[1 概述 4](#_Toc15209)

[2 设计目标和约束 4](#_Toc22014)

[3 架构设计 5](#_Toc19603)

[3.1 总体方案 5](#_Toc8009)

[3.2 架构说明 5](#_Toc10102)

[3.2.1 架构图及说明 5](#_Toc25326)

[3.2.2 架构设计关键点 7](#_Toc30139)

[3.2.3 高可用性设计 7](#_Toc27174)

[3.2.4 高性能设计 8](#_Toc20765)

[3.2.5 可扩展性设计 8](#_Toc12163)

[3.2.6 安全性设计 8](#_Toc31394)

[3.2.7 其他设计 8](#_Toc17856)

[4 部署方案 9](#_Toc162)

[4.1 组件图 9](#_Toc22079)

[4.2 配置图 10](#_Toc29138)

# 图目录

[图 1 总体架构图 5](#_Toc17010)

[图 2 项目生命流程图 7](#_Toc21429)

[图 3 组件图 9](#_Toc21672)

[图 4 配置图 10](#_Toc32318)

# 概述

本文档用于描述AchieveIt项目的软件设计架构，对软件的范围、功能、实现技术进行规范说明，以满足系统设计的目标和相应约束。

# 设计目标和约束

AchieveIt是一个web项目。应满足项目立项说明书中描述的功能性需求，实现包括管理项目生命流程等功能，以及实现与外部系统的对接。在安全性上应有可靠的用户身份验证以及权限控制机制。同时，本项目应当满足用户7\*24的访问需求，并且在访问期间保证相应的性能及数据安全。本项目在数据保存、运营等方面应满足当地在相关方面的法律法规。

# 架构设计

## 总体方案

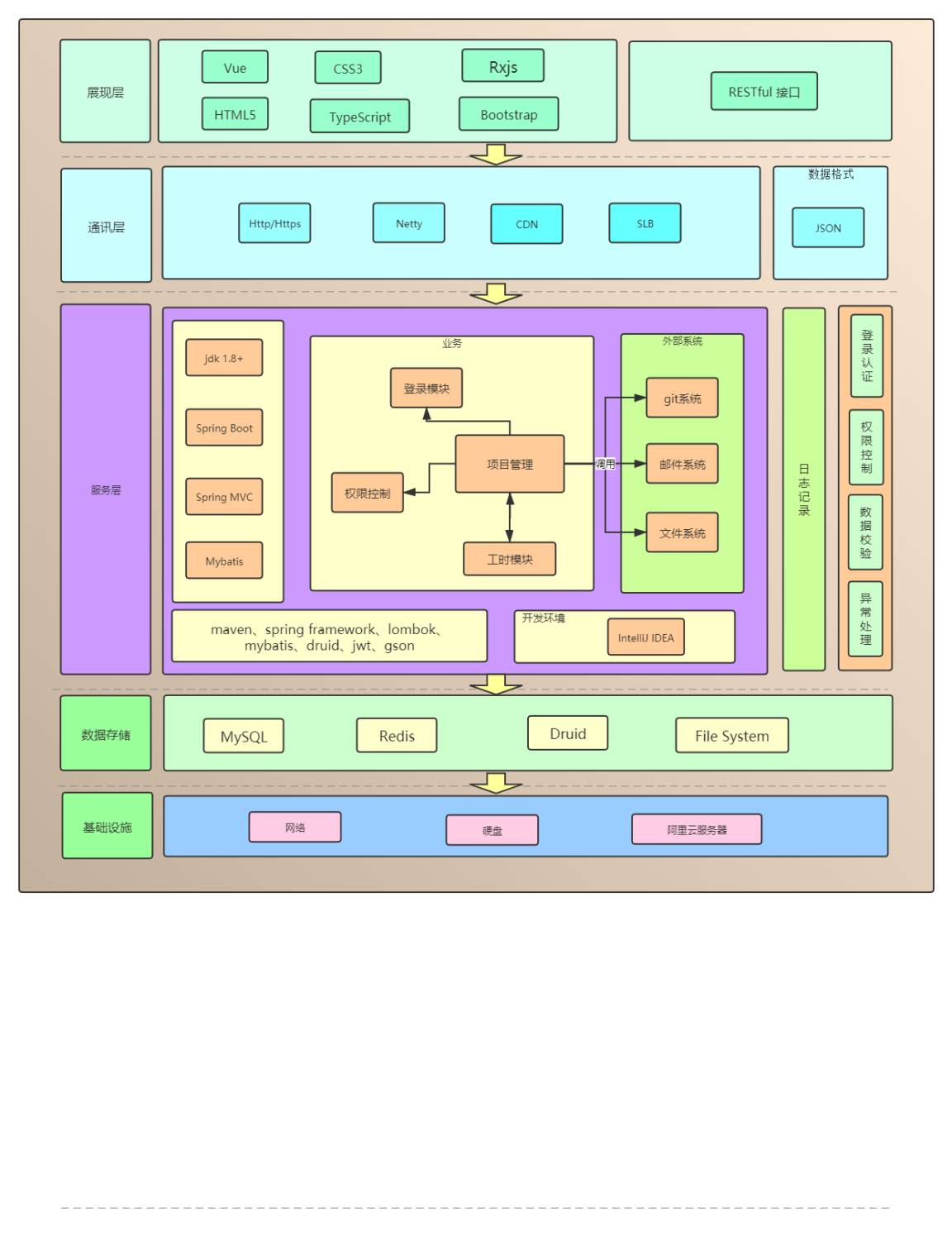


图 1总体架构图

软件开发前后端分离，并部署在云上。前端使用Vue框架，基于HTML5、CSS3、Typescript等设计前端UI界面，并与后端交互。前后端通过JSON格式数据对接。后端使用Spring Boot + Spring MVC + Mybatis框架实现功能，为前端踢狗RESTful接口，并通过Durid管理数据库连接池与数据库交互。数据库使用MySQL进行持久化存储，并通过Redis缓存。项目作为微服务部署到云端，通过CDN实现静态数据加速、SLB实现服务器负载均衡。

## 架构说明

### 架构图及说明

整个软件的架构分为五个部分，自底向上分别是基础设施、数据存储、服务层、通讯层和展现层。

**基础设施：**主要指软件运行所需的基础设施支持，包括网络、硬盘和云服务器。云服务器用于支持项目部署。

**数据存储：**主要是使用MySQL和文件系统来支持数据存储。

**服务层：**主要包含立项、管理成员、管理项目信息、公示系统、管理设备等业务，以及支持业务所使用的技术，这些技术包括编程语言Java、编程框架Spring Boot、Spring MVC、Mybatis以及缓存支持Redis、数据连接池Durid。除此之外，为了保障客户明确的需求，还需要进行日志记录、登录认证、权限控制、数据校验、异常处理。其中日志记录起着错误检查等作用，登录认证是保证软件中各用户以各自的角色各司其职，权限控制结合登录认证保障业务正常进行；数据校验和异常处理则是为了保障软件产品的可靠性。

**通讯层：**主要包括Http/Https、Netty、CDN、SLB。项目通过Http/https请求进行通信及传输数据，使用Netty框架进行客户端和服务器的通信，项目作为微服务部署到云端，通过CDN实现静态数据加速、SLB实现服务器负载均衡。

**展现层：**主要使用到Vue、CSS3、Rxjs、HTML5等技术，以及使用到RESTful接口进行前后端通信，以JSON格式的数据进行交互。前端使用Vue框架，基于HTML5、CSS3、Typescript、Bootstrap、Rxjs设计前端UI界面，使用RESTful规范进行前后端交互。

**项目生命周期流程图：**

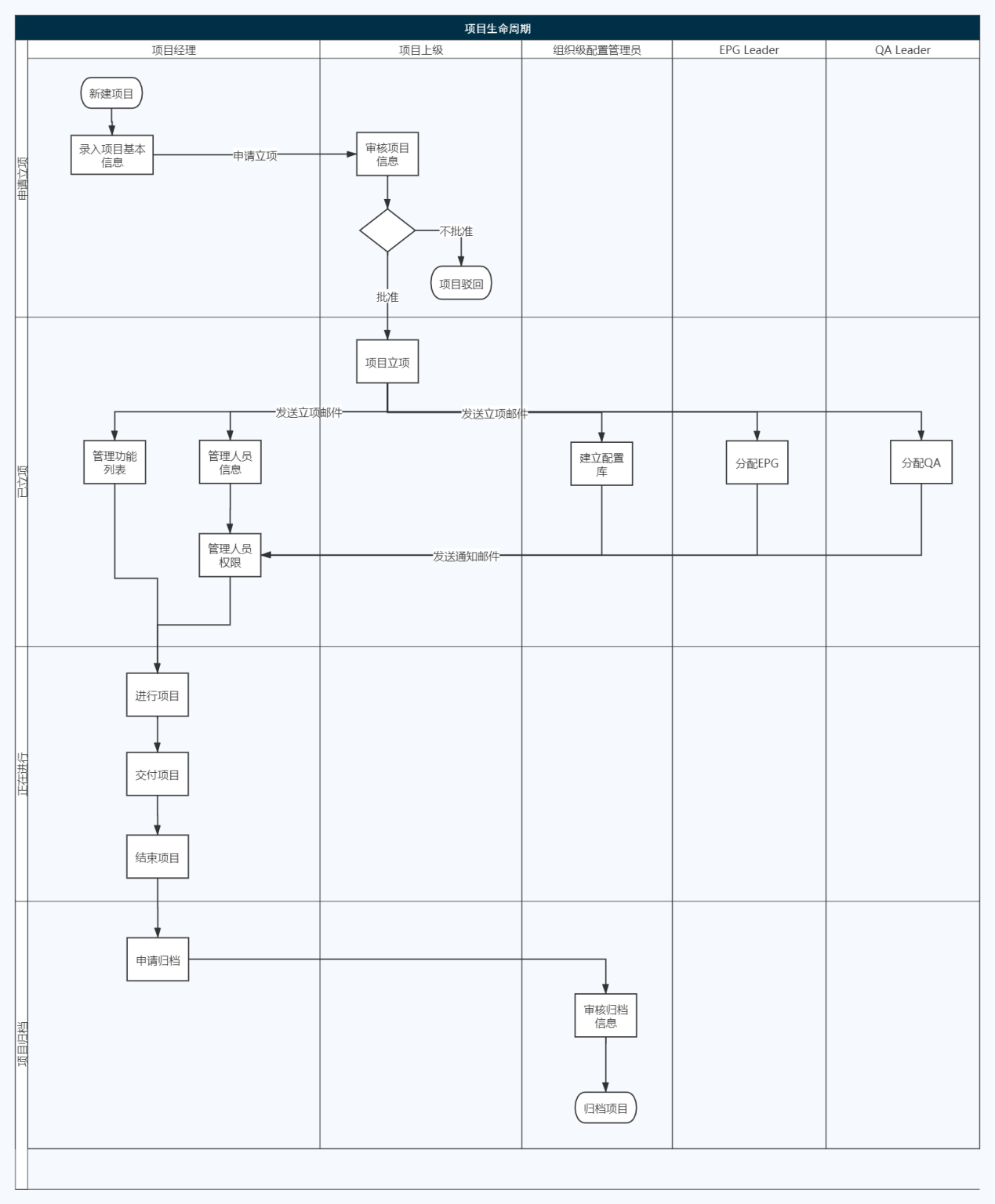


图 2项目生命流程图

### 架构设计关键点

前后端分离，Vue + Spring Boot + MySQL

JWT（JSON Web Token）实现用户身份识别；

### 高可用性设计

部署在公有云上，通过冗余链路、备份服务器、备份数据库消除单点故障，实现热备份、热启动。

### 高性能设计

基于云的CDN、SLB实现静态数据加速和服务器负载均衡；

在服务层，使用durid实现高效、功能强大的数据库连接池，同时使用redis对高频数据进行缓存；

在数据库层设计合理的index，加快数据库的访问；

部署到云上，动态扩展服务器内存、网络等，以应对访问徒增等情况。

### 可扩展性设计

使用Spring Boot启动项目，基于Spring MVC框架实现服务器开发。Spring具有控制反转、依赖注入等特性，使得系统能方便地增加新功能新特性；

前后端分离及RESTful接口的设计，便于移动端（Android、IOS）的开发。

### 安全性设计

服务器端通过jwt进行用户身份识别，用户登录后返回token，在后续交互过程中验证客户端传来的token是否有效，同时在超出规定时间后设置该token失效；

数据库表中保存用户权限信息，根据用户身份及管理员操作维护权限表，在用户执行操作时，校验用户相关权限；

使用sha1将用户密码单向散列（提供全局salt和用户独有salt），使得即便数据库泄露，他人也难以还原用户密码。

### 其他设计

前后端分离，使得开发人员可以相对独立地开发同一个功能，便于分工。

# 部署方案

## 组件图



图 3组件图

## 配置图

图 4配置图