

平顶山高新区

智慧园区项目之项目管理系统

概要设计说明书

江苏风云科技服务有限公司

JIANGSU FENGYUN TECHNOLOGY SERVICES CO.,LTD.

2017年09月

版本记录

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本号 | 时间 | 记录人 | 变更原因 | 变更描述 |
| 1.0 | 2017.11.07 |  |  |  |

目 录

[版本记录 1](#_Toc497921374)

[第一章 引言 3](#_Toc497921375)

[**1.1编写目的** 3](#_Toc497921376)

[**1.2编写背景** 3](#_Toc497921377)

[**1.3 参考资料** 4](#_Toc497921378)

[**1.4 术语和关键字** 4](#_Toc497921379)

[**1.5 读者对象** 4](#_Toc497921380)

[第二章 总体框架设计 4](#_Toc497921381)

[**2.1假设条件** 4](#_Toc497921382)

[**2.2业务架构** 4](#_Toc497921383)

[**2.3功能架构** 5](#_Toc497921384)

[**2.4系统架构** 6](#_Toc497921385)

[**2.5物理架构** 8](#_Toc497921386)

[第三章 主要模块架构和概要设计 9](#_Toc497921387)

[**3.1开发框架** 9](#_Toc497921388)

[**3.2.系统认证和权限体系** 12](#_Toc497921389)

[3.2.1.系统认证 12](#_Toc497921390)

[3.2.2.系统权限管理 13](#_Toc497921391)

[**3.3.项目管理过程** 13](#_Toc497921392)

[3.3.1.项目基础管理流程 13](#_Toc497921393)

[3.3.2.项目流程权限控制 14](#_Toc497921394)

[**3.4.外部接口** 14](#_Toc497921395)

[3.4.1.统一认证平台接口 14](#_Toc497921396)

[3.4.2.招商管理系统接口 14](#_Toc497921397)

[3.4.3.OA办公系统接口 14](#_Toc497921398)

[第四章非功能性设计 14](#_Toc497921399)

[**4.1安全性设计** 14](#_Toc497921400)

[**4.2性能和扩展设计** 15](#_Toc497921401)

[4.2.1.基本性能指标 15](#_Toc497921402)

[4.4.2.性能提升可扩展设计 15](#_Toc497921403)

# **第一章 引言**

**1.1编写目的**

本文档的目的是阐述“智慧园区项目”之一“项目管理系统”的概要设计文档，以指导开发人员、项目实施人员及其它相关人员进行相关的详细设计与开发，是详细设计说明书的编写依据，同时也是该项目最终验收的参考依据。

**1.2编写背景**

智慧园区建设，是指在产业集聚区内，按照科学的园区发展理念，以新一代信息技术为手段，以智慧应用为支撑，全面整合园区内外的资源，使园区管理服务等更高效便捷，实现基础设施网络化、开发管理信息化、功能服务精细化和产业发展智能化。推进智慧园区建设，旨在创新园区管理、服务产业发展，构建“互联网+”集聚区新业态新模式，促进产业集聚区发展由企业集中型向产业集聚型、由资源招商型向品牌价值型转变。

平顶山高新区智慧园区项目是在高新区信息化顶层规划设计下围绕以政务云为中心，建设符合平顶山高新区实际需求的智慧政务、城市管理、民生服务、产业服务等方面的服务内容，项目建设覆盖到高新区全部管辖区域。力争建设在全国视野更具前瞻性的智慧园区，打造成为平顶山高新区的一张新名片，使政府管理行为更加智能、决策能力更加科学，为企业提供服务更加高效、行政审批更加便捷；让居民公共服务更加全面、日常生活更加便利。

**1.3 参考资料**

1、《平公资采2017340号]招标文件》；

2、《平顶山高新区智慧园区项目整体规划方案》；

3、《项目管理系统建设方案》

4、《项目管理系统需求规格说明书》

**1.4 术语和关键字**

暂无

**1.5 读者对象**

1. 建设方：高新区管委会智慧园区项目建设领导小组；
2. 监理方：智慧园区项目建设监理单位相关人员；
3. 承建方：项目经理、开发经理、开发人员、测试人员；
4. 用户方：高新区管委会相关局办用户单位系统使用人员。

# **第二章 总体框架设计**

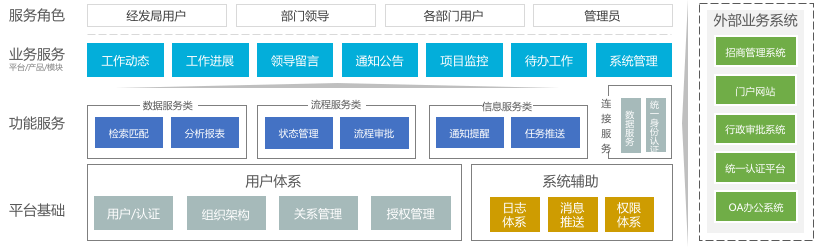
**2.1假设条件**

平台的设计通过不同的角度提出了一些假设和依赖，如下：

* 服务器端应用的运行环境：jdk1.8+；

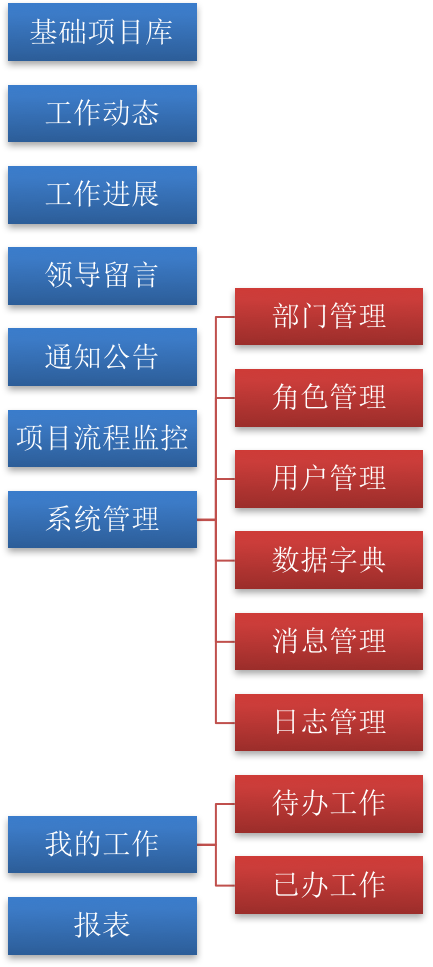
**2.2业务架构**

从业务角度对整个平台进行框架设计，业务架构图如下：



**2.3功能架构**

按照业务功能组织，以层级结构为基础，平台功能架构如下：



**2.4系统架构**

平顶山项目管理系统与多个由多个应用程系统并行，构建完善的应用体系。 以下是平顶山项目管理系统的体系结构图



系统组件：

|  |  |  |
| --- | --- | --- |
| **系统** | **描述** | **使用技术** |
| 门户 | 承接大众在技术产权咨询、评估及交易等业务的线上网站，能够提供政府及专业机构在技术产权领域的最新资讯信息。 | Java, avalonJS, Spring Framework |
| 管理平台 | 提供数据维护统一管理入口，为门户的运转提供数据支持。 | Java, avalonJS, Spring Framework |
| 应用服务API |  | Java, Spring Framework，SQLServer |

第三方开源技术：

|  |  |  |  |
| --- | --- | --- | --- |
| **Type** | **Selection** | **Version** | **Description** |
| 前端框架 | Jquery | 1.11.2 |  |
| avalonJS | 2.2.1,1.5 |  |
| Bootstrap | 3.3.7 |  |
| Echarts | 3.1.5 |  |
| 后端框架 | Spring | 4.2.5.RELEASE |  |
| Shiro | 1.3.2 |  |
| Environment | JDK | 1.7+ |  |
| Tomcat | 7.0+ |  |
| Back end  Common Tools | Slf4j | 1.7.21 |  |
| Commons-lang3 | 3.3.2 |  |
| Commons-io | 2.4 |  |
| Commons-codec | 1.9 |  |
| Commons-fileupload | 1.3.1 |  |
| Commons-beanutils | 1.9.1 |  |
| Jackson | 2.6.6 |  |
| Fastjson | 1.2.7 |  |
| Xstream | 1.4.7 |  |
| Poi | 3.9 |  |

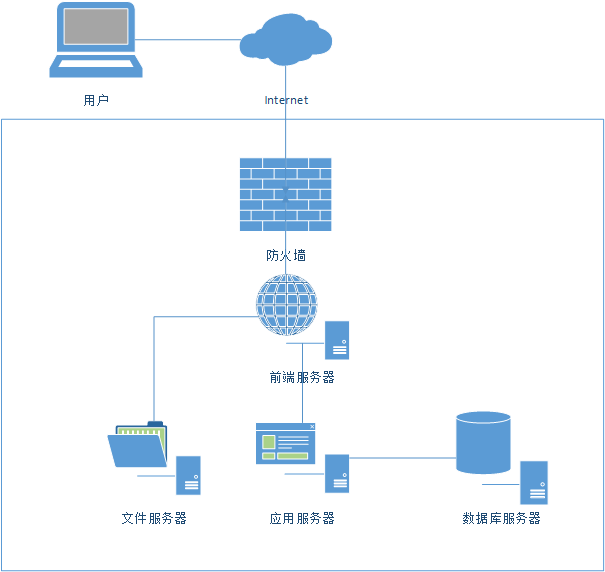
**2.5物理架构**

项目管理系统将有两套运行环境，以满足测试和生产。

测试环境：模拟生产环境的物理环境，在满足系统要求下使用最小化的托管服务，包括以下组件（角色）：

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **服务器角色** | **服务器类型** | **部署应用** | **数量** | **操作系统** | **CPU** | **内存**  **(GB)** | **硬盘**  **(GB)** | **备注** |
| 前端服务器 | 虚拟机 | 前端应用 | 1 | centOs7.0 | 4 core | 16 | 100 | 可与API服务器合用一台 |
| API服务器 | 虚拟机 | WSO2 |  | centOs7.0 | 4 core | 16 | 100 | 可与前端服务器合用一台 |
| 文件服务器 | 虚拟机 |  |  | centOs7.0 | 4 core | 16 | 200 |  |
| 数据库服务器 | 虚拟机 |  |  | centOs7.0 | 4 core | 16 | 200 |  |

服务器网络部署架构图：



备注：

以上需要四个服务器（角色）用于部署项目管理系统，理论上多个角色可以共用服务器，但建议文件服务器和数据库服务器相对独立。

# **第三章 主要模块架构和概要设计**

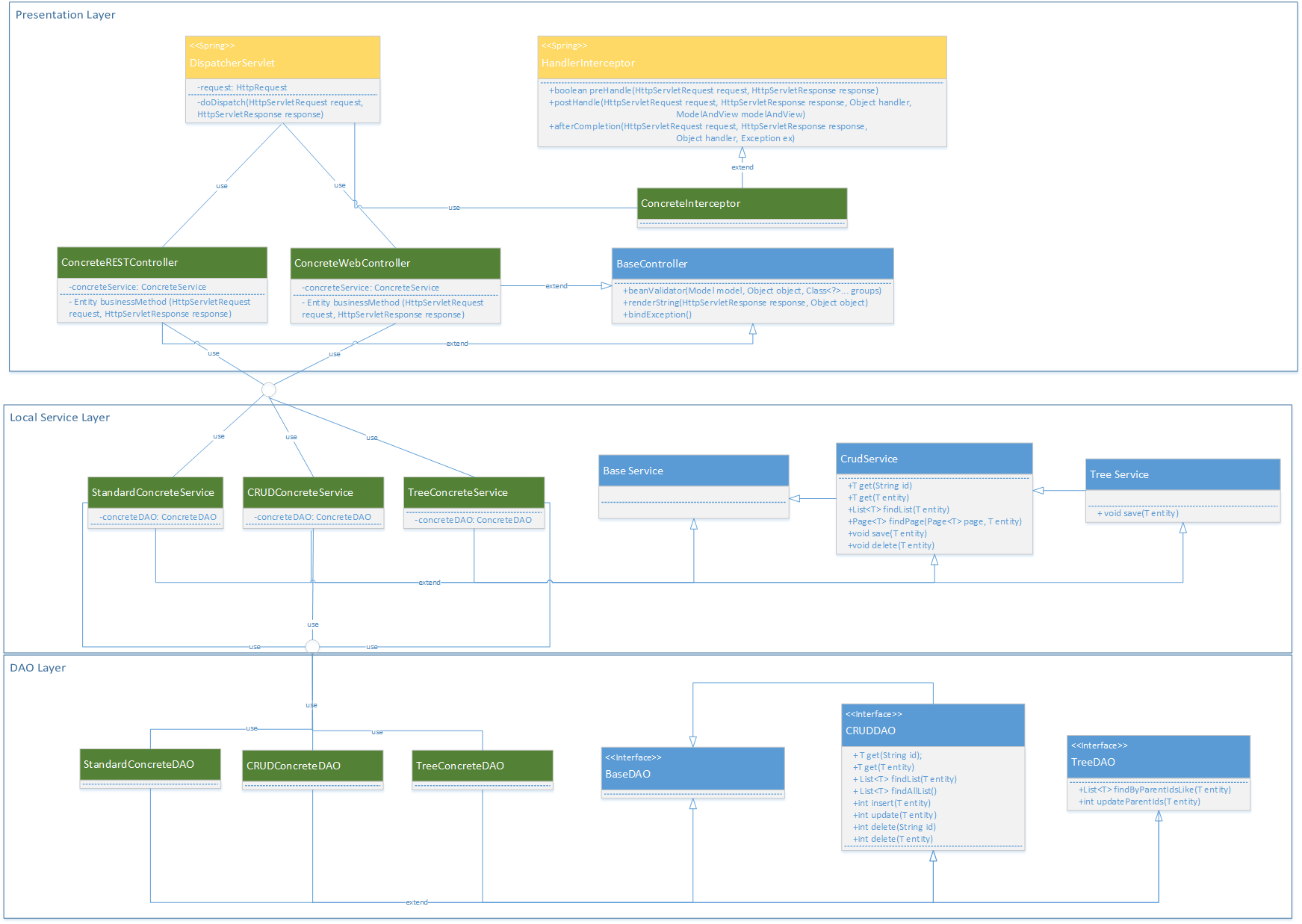
**3.1开发框架**

服务器端将使用的开发框架，主要是从各种开源框架进行构建的。下面是主要的组件列表：

|  |  |  |
| --- | --- | --- |
| **组件** | **依赖** | **描述** |
| Web组件 | Spring MVC | 用于请求、响应处理机制的开发框架 |
| Web服务组件 | Spring REST | 用于服务处理机制和服务开发框架 |
| 安全组件 | Apache Shiro | 用于权限检查、请求授权检查和网络服务权限检查 |
| 本地服务容器 | Spring IOC | 用于java Bean的注入 |
| 数据持久化组件 | SpringData | 用于数据库访问和数据对象的增删改查 |

**开发框架模式：**

使用如下框架按照下图模式进行开发：



**开发模式图**

**黄色部分**是Spring第三方框架核心类

**蓝色部分**是风云框架基于第三方框架继承并封装的基类

**绿色部分**是由开发者为具体的业务逻辑实现的具体类。具体实施主要有以下几个部分：

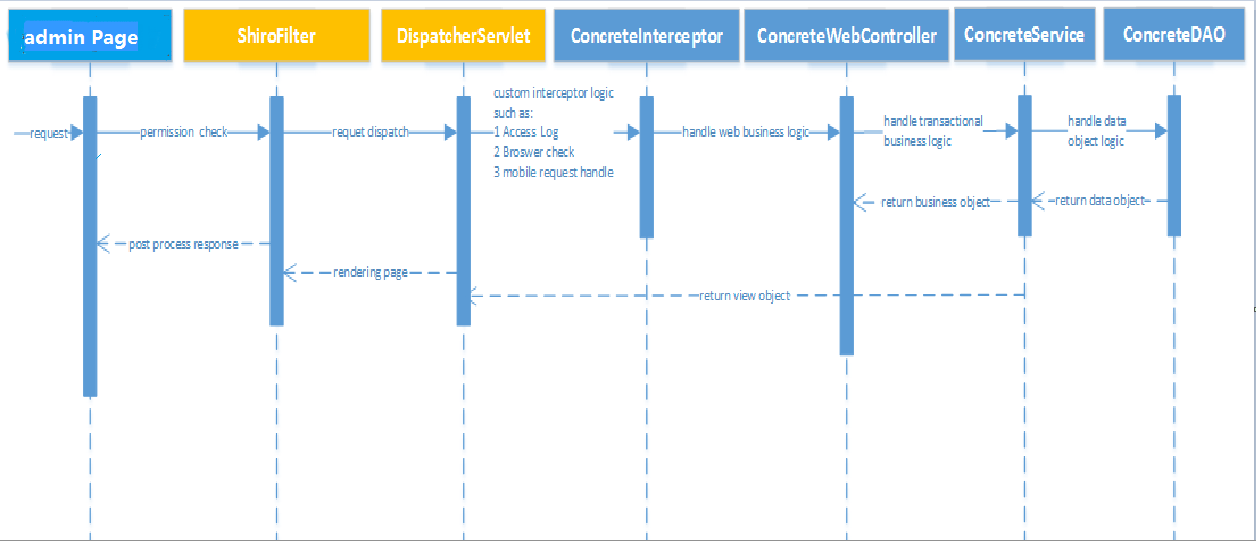
|  |  |
| --- | --- |
| **核心类** | **描述** |
| ConcreteWebController | 它是处理web请求的业务接口类，数据处理web请求并调用本地服务通过HTTP响应返回结果 |
| ConcreteRESTController | 它是处理APP请求的业务接口类，数据处理客户端请求并调用本地服务通过HTTP响应返回结果 |
| ConcreteInterceptor | Web和APP请求时先进入此类，由于验证请求的预处理 |
| ConcreteService | 这是核心的业务逻辑处理组件，完成业务逻辑，提供服务层的接口与事务控制 |
| ConcreteDAO | 这是数据处理过程的逻辑组件，管理数据库访问逻辑和提供service层的数据处理接口 |

**请求处理流程：**

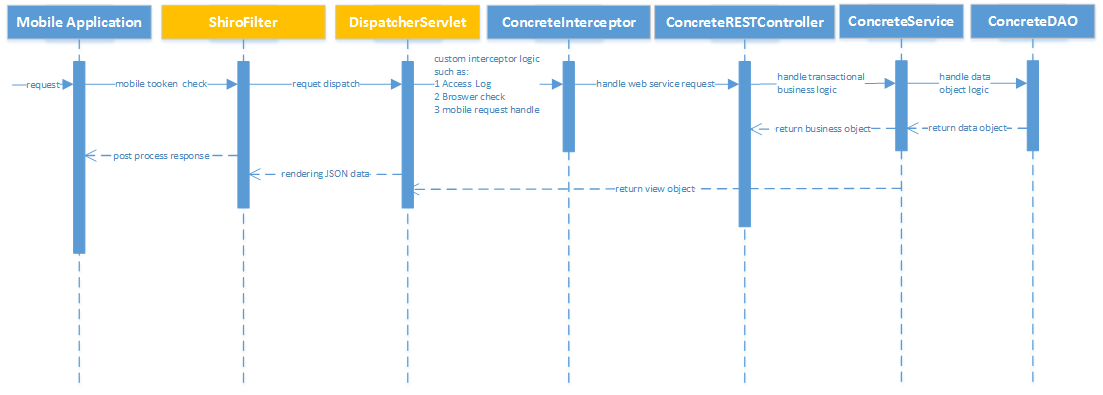
下面的时序图完整的描述了一个后端处理如何请求处理并如何返回处理结果。

从web请求和app请求两方面考虑并进行设计。

1. **Web请求处理：**



1. **APP REST请求处理：**



**3.2.系统认证和权限体系**

### 3.2.1.系统认证

#### 3.2.1.1.账号来源

系统账号数据来源于OA系统，由OA统一管理，OA数据库开放相关的账号、组织架构视图，提供给项目管理系统读取。

OA系统相关数据表：

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 表名 | 中文名 | 表说明 |
| 1 | user | 用户表 | 存储用户账号信息、密码等，日常用于获取用户信息，登录时用来校验密码 |
| 2 | department | 部门表 | 存储部门组织架构，用于获取用户所述部门信息、部门选择等操作时调用 |

#### 3.2.1.2.账号及组织架构管理

账号和组织架构管理均在OA系统实现，此处略。

#### 3.2.1.3.统一用户认证

统一用户认证系统，在网站门户项目中实现，并提供给各个系统统一调用。

统一用户系统采用CAS3.0标准协议实现，项目管理系统作为其中的一个Client端，按标准调用统一用户系统的服务。

需要调用的CAS接口和内部调用：

* 实现Filter，控制用户未登录状态跳转到CAS服务器
* 实现对CAS令牌的解析
* 实现本地Session存储
* 实现用户登出相关调用及本地Session的清理

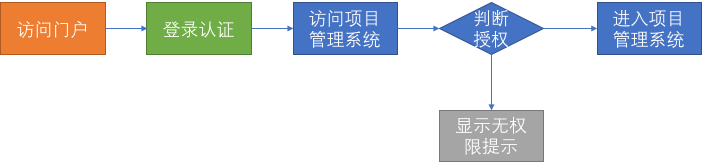
### 3.2.2.系统权限管理

#### 3.2.2.1.系统使用授权

OA系统账号覆盖整个政务网，但非所有网内用户都能使用项目管理系统，系统管理员需要对用户进行授权。

在项目管理系统创建用户授权表，只有表内已授权的用户，才能访问项目管理系统，否则显示相关的无权限界面进行提示。系统提供相关的用户授权维护。

具体的认证和授权验证流程：



#### 3.2.2.2.模块、功能点权限控制和分配

用户通过菜单控制模块权限，通过功能按键控制功能点权限，主要涉及体现：

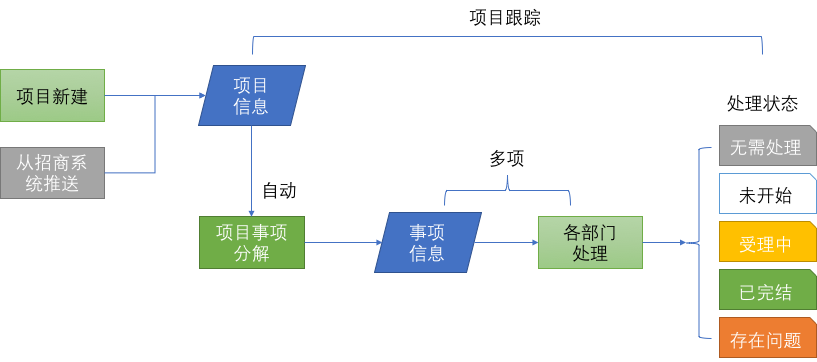
* 菜单的可见性（前端控制）
* 页面的可访问性（后端控制）
* 页面组件（含功能按钮）的可见性（前端控制）
* API的可调用性（后端控制）

菜单可见权限和用户拥有的权限角色，提供相关的配置功能。

**3.3.项目管理过程**

### 3.3.1.项目基础管理流程

项目基础管理流程图：



### 3.3.2.项目流程描述

1. 项目新建：由经发局用户操作，新增完成后，产生项目信息
2. 从招商系统推送：招商系统项目协议签署完成后，通过API调用，将项目信息推送到项目管理系统，此类项目仅展示信息，不做后续流程流转
3. 项目事项分解：自动过程，新建的项目，自动按照模板分解成多个项目事项，并分配到各个部门，由相关部门的相关角色用户进行后续跟进处理。期间产生用户待办任务，并推送到OA系统
4. 各部门处理：各部门用户对事项进行处理，可以处理成五种状态：无需处理、未开始、受理中、已完结、存在问题。用户可以随时更改状态
5. 项目跟踪：项目管理系统随时展示项目各个事项的处理状态，以图形化方式展示，供相关权限的用户和领导查看。

**3.4.外部接口**

### 3.4.1.统一认证平台接口

与统一认证平台对接包含登录和注销两个接口，接口符合CAS3规范，具体详见统一认证接入文档，此处略。

### 3.4.2.招商管理系统接口

招商管理系统需要把项目推送到项目管理系统，项目在立项完成后进行推送，项目管理系统提供API供调用。另外，项目管理系统可以退回项目，之后让招商系统重新提交，退回的项目在项目管理系统标注为“已退回”状态重新提交后，作为一条新记录提交。

相关接口：

#### 3.4.2.1.项目新增接口

|  |  |
| --- | --- |
| 接口名称 | 项目新增 |
| 接口提供方 | 项目管理系统 |
| 接口调用方 | 招商管理系统 |
| 输入参数 | 入园协议扫描件，项目简介，投资概况，征地，占地面积，总投资，投资单位，公司全称，选址位置，项目负责人姓名，电话，项目联系人姓名，电话，项目ID |
| 输出结果 | 是否新增成功/失败信息 |
| 调用时机 | 招商协议签署完成后 |
| 其他备注 | 招商管理系统须在调用完成后标注状态，以便处理后续退回 |

#### 3.4.2.2.项目新增接口

|  |  |
| --- | --- |
| 接口名称 | 项目退回 |
| 接口提供方 | 招商管理系统 |
| 接口调用方 | 项目管理系统 |
| 输入参数 | 项目ID、退回原因 |
| 输出结果 | 是否退回成功/失败信息 |
| 调用时机 | 随用户相关操作 |
| 其他备注 | 项目管理系统须在调用完成后将该项目标注成“已退回” |

### 3.4.3.OA办公系统接口

项目管理系统需要将待办事项推送到OA，以便OA系统进行个人事项的统一展示和管理。

相关接口：

#### 3.4.3.1.事项新增接口

|  |  |
| --- | --- |
| 接口名称 | 个人待办事项新增 |
| 接口提供方 | OA办公系统 |
| 接口调用方 | 项目管理系统 |
| 输入参数 | 事项ID、事项名称、归属人、截止时间、处理URL、当前处理状态 |
| 输出结果 | 是否新增成功/失败信息 |
| 调用时机 | 系统产生待办事项时 |
| 其他备注 |  |

#### 3.4.3.2.事项状态更新接口

|  |  |
| --- | --- |
| 接口名称 | 个人待办事项状态更新 |
| 接口提供方 | OA办公系统 |
| 接口调用方 | 项目管理系统 |
| 输入参数 | 事项ID、当前处理状态 |
| 输出结果 | 是否更新成功/失败信息 |
| 调用时机 | 待办事项状态变化时 |
| 其他备注 |  |

# 第四章非功能性设计

**4.1安全性设计**

保证基础信息系统的整体安全，以防止各种人为因素和非人为因素带来的隐患和来自互联网系统的破坏和攻击，平台提出了网络安全、系统安全、资源安全、应用安全、数据安全等方面的要求。

|  |  |  |
| --- | --- | --- |
| 序号 | 安全类型 | 安全设计 |
| 1 | 网络安全 | 防火墙控制，内外网隔离，通过防火墙屏蔽外来攻击，通过端口映射将服务提供给外部访问，并对关键服务做来源筛选。 |
| 2 | 系统安全 | 1. 可实现IP访问控制、登录时间控制；部分服务根据实际要求可控制连接数。 2. 认证模块实现统一的认证安全，各个认证客户端授权接入。 3. SQL防注入控制、JS防注入控制。 |
| 3 | 资源安全 | 资源服务器仅提供内网方位，并授权应用服务器访问。应用操作界面对相关的资源访问进行权限的控制。 |
| 4 | 应用安全 | 1. 采用权限体系，控制各个不同用户的使用权限 2. 对用户操作内容进行记录，对用户提交的信息进行筛查控制，对资源类上传控制大小。 |
| 5 | 数据安全 | 1. 关键数据资源价目存储，防止数据外泄 2. 数据库管理权限统一控制分配，数据库仅供内网访问，对个人访问数据库进行授权登记。 |

**4.2性能和扩展设计**

### 4.2.1.基本性能指标

本系统基础性能指标如下：

在线数在100内，并发数在20内，页面响应时间5秒内，数据量大，查询条件复杂的页面，页面响应时间控制在10秒内。

### 4.4.2.性能提升可扩展设计

系统根据实际访问情况，未来预留性能可扩展方案，如下：

* 负载均衡扩展：可通过在前端服务器之上增加一个反向代理服务，可部署Nginx服务实现，对访问请求进行内网分发，并构建第二台或更多的前端服务器，实现对外部请求的负载均衡。系统本身支持多前端分布式部署。
* 反向代理双机热备：为防止前端服务器的反向代理服务的单点故障，系统可实现反向代理双击热备，通过假设第二台Nginx服务器，并通过Keepalived中间件实现热备切换。
* 应用访问池：系统前后端API访问支持应用访问池，当后续并发量过大，可考虑扩展多台API服务器，实现API层面的负载均衡及热备。
* 数据访问高可用：系统数据库采用Mysql，支持数据库的主从复制、读写分离，如运营过程中出现数据访问瓶颈，可扩展相关的功能，确保数据访问的性能和高可用。