 未标题-1

**信息科学与工程学院**

**实 习 报 告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 课程名称： | 项目开发实践（企业学习） | | | | | |
| 报告题目： | 基于Springboot的权限管理系统 | | | | | |
| 学生姓名： | 宗阔雄 | | 学号： | | 201516010209 | |
| 专业班级： | 计算机科学与技术（卓越计划）F1502班 | | | | | |
| 实习单位： |  | | | | | |
| 实习时间： | 2018-7-23— —2018-12-28 | | | | | 共 15 周 |
| 企业指导教师： | | 张慧 | | 职称/职务： | | AD |
| 学校指导教师： | | 唐建国 | | 职 称： | | 讲师 |

2018 年 12 月 16 日

中文摘要

1. 目的

权限系统一直以来是我们应用系统不可缺少的一个部分，若每个应用系统都重新对系统的权限进行设计，以满足不同系统用户的需求，将会浪费我们不少宝贵时间，所以花时间来设计一个相对通用的权限系统是很有意义的。该文档是关于客户对于权限管理网站的功能和性能的要求，重点描述了权限管理网站的功能规格。其主要内容包括：

* 系统功能简介
* 系统结构设计
* 系统接口设计
* 数据设计
* 模块设计
* 界面设计

1. 方法

本项目后台采用Spring Boot作为基础框架，数据库访问层使用Mybatis，安全框架使用Shiro，模板引擎为Thymeleaf，数据库连接池为Druid，并使用Redis作为缓存。其它的也包括Logback，quartz，poi等技术。通过对当前常见系统中的数据库用户，角色，权限进行分析，得出的一套通用的数据库用户权限管理系统。

1. 结果

本系统提供一个调用简单、可复用性高、满足一般需求的权限管理模块，并为需要对权限控制,比如应用系统的功能菜单、各个界面的按钮控件等进行权限的操控，以及实现系统监控和任务调度的功能。

1. 结论

本系统可以套用在多数系统中辅助使用，实现系统中的用户权限管理，可以达到减少重复开发的目的，方便快速上手，减少项目成本。

英文摘要

1. Purpose

Permission system has always been an indispensable part of our application system, If each application system is redesigned for system permissions to meet the needs of different system users，It will waste a lot of our valuable time，So it makes sense to take the time to design a relatively generic permission system。This document is about the functional and performance requirements of the client for the permission management website. This paper mainly describes the functional specification of permission management website。Its main content includes：

* system function introduction
* System structure design
* system interface design
* data design
* module design
* UI design

1. Method

Spring Boot is adopted as the basic framework in the background of this project，The database access layer use Mybatis，The security framework uses Shiro, the template engine is Thymeleaf, the database connection pool Druid, and Redis as the cache. Other technologies include Logback, quartz, poi, etc. Through the analysis of database users, roles and permissions in current common systems, a set of universal database user permission management system is obtained.

1. Results

This system provides a simple call, high reusability, to meet the general needs of the permission management module, and for the need to control the authority, such as the application system function menu, the interface button control and other control permissions, as well as the implementation of system monitoring and task scheduling functions.

1. Conclusion

This system can be applied in most systems to assist the use of the system, to achieve the user rights management system, can achieve the purpose of reducing repeated development, convenient and quick to start, reduce project costs.

目录

[1 可行性分析 1](#_Toc534034961)

[1.1 市场分析 1](#_Toc534034962)

[1.2 政策分析 1](#_Toc534034963)

[1.3 竞争实力分析 1](#_Toc534034964)

[1.4 技术可行性分析 1](#_Toc534034965)

[1.5 时间和资源可行性分析 1](#_Toc534034966)

[1.6 知识产权分析 1](#_Toc534034967)

[2 项目计划 2](#_Toc534034968)

[2.1 项目介绍 2](#_Toc534034969)

[2.1.1 项目范围 2](#_Toc534034970)

[2.1.2 项目目标 2](#_Toc534034971)

[2.2 项目过程定义 2](#_Toc534034972)

[2.2.1 过程模型 2](#_Toc534034973)

[2.2.2 方法与工具 2](#_Toc534034974)

[2.3 人力资源计划 3](#_Toc534034975)

[2.4 任务与进度 3](#_Toc534034976)

[3 需求分析 4](#_Toc534034977)

[3.1 产品介绍 4](#_Toc534034978)

[3.2 产品面向的用户群体 4](#_Toc534034979)

[3.3 产品应当遵循的标准或规范 4](#_Toc534034980)

[3.4 产品范围 4](#_Toc534034981)

[3.5 产品的功能性需求 4](#_Toc534034982)

[3.5.1 功能性需求分类 4](#_Toc534034983)

[3.6 产品的非功能性需求 5](#_Toc534034984)

[3.6.1 用户界面需求 5](#_Toc534034985)

[3.6.2 硬件环境需求 5](#_Toc534034986)

[4 体系结构设计 5](#_Toc534034987)

[4.1 系统概述 5](#_Toc534034988)

[4.2 设计约束 5](#_Toc534034989)

[4.3 设计策略 6](#_Toc534034990)

[4.4 系统总体结构 6](#_Toc534034991)

[4.5 子系统的结构与功能 6](#_Toc534034992)

[5 用户界面设计 8](#_Toc534034993)

[5.1 应当遵循的界面设计规范 8](#_Toc534034994)

[5.2 界面的关系图和工作流程图 8](#_Toc534034995)

[5.3 主界面 9](#_Toc534034996)

[5.4 系统管理 10](#_Toc534034997)

[5.5 系统监控 12](#_Toc534034998)

[5.6 任务调度 14](#_Toc534034999)

[6 模块设计 17](#_Toc534035000)

[6.1 模块命名规则 17](#_Toc534035001)

[6.2 模块汇总 17](#_Toc534035002)

[6.3 登陆注册及系统管理的模块设计 18](#_Toc534035003)

[6.4 任务调度的模块设计 19](#_Toc534035004)

[6.5 系统监控的模块设计 20](#_Toc534035005)

[7 数据库设计 20](#_Toc534035006)

[7.1 数据库环境说明 21](#_Toc534035007)

[7.2 数据库的命名规则 21](#_Toc534035008)

[7.3 逻辑设计 21](#_Toc534035009)

[7.4 物理设计 24](#_Toc534035010)

[7.5 安全性设计 27](#_Toc534035011)

[7.5.1 用户帐号密码的加密方法 27](#_Toc534035012)

[7.5.2 角色与权限 27](#_Toc534035013)

[8 系统测试 28](#_Toc534035014)

[8.1 测试基础 28](#_Toc534035015)

[8.2 各模块测试 28](#_Toc534035016)

[8.3 集成测试 28](#_Toc534035017)

[8.4 路径测试检查表 29](#_Toc534035018)

[8.5图形用户界面测试用例 29](#_Toc534035019)

# 可行性分析

## 市场分析

当前的软件系统中,有很大一部分都使用了用户权限的数据库设计,于是设计一个能够通用的数据库用户权限管理系统是很有意义的。

## 政策分析

本项目符合我国政府的相关法律法规。

## 竞争实力分析

本系统适用性广，包括企业人事，项目管理以及一些其它的系统中涉及到多用户的情况等，均可以使用。

## 技术可行性分析

本系统中设计到核心框架Springboot已经发展到2.1.X，再加上背后Spring家族的社区活跃度，该框架有着极为优秀的发展前景。SpringBoot帮助开发者快速启动一个Web容器；SpringBoot继承了原有Spring框架的优秀基因；SpringBoot简化了使用Spring的过程。而安全框架Shrio也有一下优点：易于使用 - 易用性是这个项目的最终目标。提供易于理解的 Java Security API。

广泛性 - 没有其他安全框架可以达到Apache Shiro宣称的广度，它可以为你的安全需求提供“一站式”服务。灵活性 - Apache Shiro可以工作在任何应用环境中。虽然它工作在Web、EJB和IoC环境中，但它并不依赖这些环境。Shiro既不强加任何规范，也无需过多依赖。

## 时间和资源可行性分析

本项目预计花费时间为两个月，由本人独立完成。

## 知识产权分析

本项目中使用的框架均为开源产品，遵循开源协议：Apache Licence，无知识产权纠纷。

# 项目计划

## 项目介绍

### 项目范围

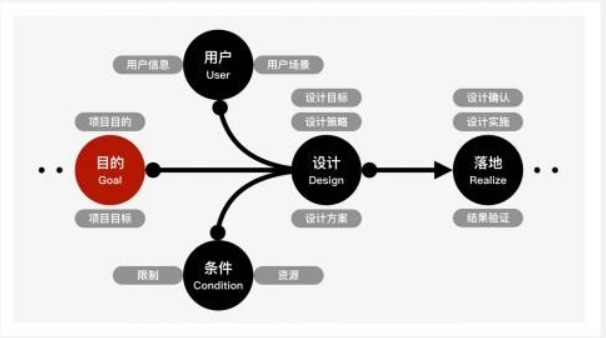
本项目是基于Springboot的权限管理系统，用来实现数据库用户权限在WEB端的管理，同时实现系统监控任务调度的功能。本项目应当包含良好的前台交互以及后台数据交换的功能，本项目适用于多用户多权限的系统中，不适用于单用户或仅管理员系统。

### 项目目标

本系统提供一个调用简单、可复用性高、满足一般需求的权限管理模块，并为需要对权限管理的系统节省开发本。

## 项目过程定义

### 过程模型



### 方法与工具

|  |  |
| --- | --- |
| 过程域 | 方法与工具 |
| GUCDR 模型 | 画布 |

## 人力资源计划

|  |  |
| --- | --- |
| 角色 | 职责 |
| 需求分析员 | 负责需求分析 |
| 系统设计员 | 负责数据库设计 |
| 程序员 | 负责代码逻辑编写 |
| 测试员 | 负责系统健壮性测试 |

## 任务与进度

|  |  |  |
| --- | --- | --- |
| 任务名称 | 起止时间 | 工作量 |
| 数据库设计 | 10.9~10.15 | 40 |
| 系统管理模块 | 10.16~11.16 | 160 |
| 系统监控模块 | 11.17~11.27 | 60 |
| 任务调度模块 | 11.28~12.12 | 60 |
| 前台设计 | 12.12~12.20 | 60 |

# 需求分析

## 产品介绍

本项目是基于SPringboot的权限管理系统，用来实现数据库用户权限在WEB端的管理，同时实现系统监控任务调度的功能。

## 产品面向的用户群体

使用多用户多权限的数据库系统的用户

## 产品应当遵循的标准或规范

(一)符合国家有关法律、法规和规章的规定; (二)符合国家产业发展方针、政策; (三)符合强制性的国家标准、行业标准和地方标准要求;

## 产品范围

本产品适用于多用户多权限的系统用来辅助管理用户权限。

## 产品的功能性需求

### 功能性需求分类

|  |  |  |
| --- | --- | --- |
| 功能类别 | 功能名称、标识符 | 描述 |
| 系统管理 | 用户管理 | 维护系统用户的信息 |
| 菜单管理 | 维护系统主页显示的菜单内容 |
| 部门管理 | 维护系统中用户所属部门信息 |
| 字典管理 | 维护数据库数据字典 |
| 角色管理 | 维护数据库角色，并维护相关角色的权限 |
| 系统监控 | 在线用户 | 显示在线用户的相关信息 |
| Redis监控 | 显示服务器调用Redis的相关信息 |
| Redis终端 | 提供Web端的Redis终端服务 |
| 系统日志 | 维护系统日志信息 |
| 任务调度 | 定时任务 | 设计维护定时任务 |
| 调度日志 | 维护系统的调度日志信息 |

## 产品的非功能性需求

### 用户界面需求

|  |  |
| --- | --- |
| 需求名称 | 详细要求 |
| 登录注册界面 | 用户进入界面之后可以进行登录注册操作 |
| 主页面 | 用户可在主界面进行相关操作 |

### 硬件环境需求

|  |  |
| --- | --- |
| 需求名称 | 详细要求 |
| Redis | Redis |
| Mysql | Mysql5.6 |
| Java | JDK1.8 |

# 体系结构设计

## 系统概述

本项目是基于SPringboot的权限管理系统，用来实现数据库用户权限在WEB端的管理，同时实现系统监控任务调度的功能。

## 设计约束

1、严格按照程序语言编写顺序编写程序。

2、规范使用各各程序之间的名称

3、编写代码需要有明显的结构性

4、界面设计。我们设计软件需要简单移动的操作界面，这是设计软件最基本的原则，对于优质的软件我们还需要提供美观的界面和人性化如何常人正常逻辑思维的操作界面。

5、运行设计。在我们设计软件的时候需要优化程序运行数度，越快的响应速度越能被用户喜欢。而我们在网络传输的数据也是越精简越好，如果你短短的一个表情你都需要通过一串很长的代码来传输，那就会导致消息发送响应数度慢。所以在图片中我们可以使用些如压缩技术来减少传输间的数据。送达数据后解压来提高运行速度。

6、呈现优先设计，如我们通过可以优先选择文字优先呈现，图片后呈现的的顺序来为优化用户体验。还有分区块呈现顺序，通过切割一整张内容，让最上方的内容优先呈现出来，而通过用户浏览的间隙来呈现接下来的内容。

## 设计策略

扩展策略：为了方便本系统以后的扩展，本系统中设计了更为灵活的mybatis-generator和mapper插件，对不同的数据库表结构有更强的适应性。

复用策略：本系统中的Shiro认证可以快速的进行重复利用，在原有配置的基础上，通过Shiro的注解即可实现后期添加的对不同的功能进行权限设定。

门面模式:调用slf4j-api作为日志应用, 它只服务于各种各样的日志系统。SLF4J是一个用于日志系统的简单Facade，允许最终用户在部署其应用时使用其所希望的日志System,本系统的实现为logbck,而调用的API均为slf4j-api,方便后期日志框架的替换

## 系统总体结构

本系统为权限管理网站，可以实现数据库用户权限管理等基本功能。管理员用户可以登录系统后对系统中的用户、菜单、角色、部门、数据字典进行相关维护，也可以进行任务调度，设置定时任务以及日志检查的功能。

## 子系统的结构与功能

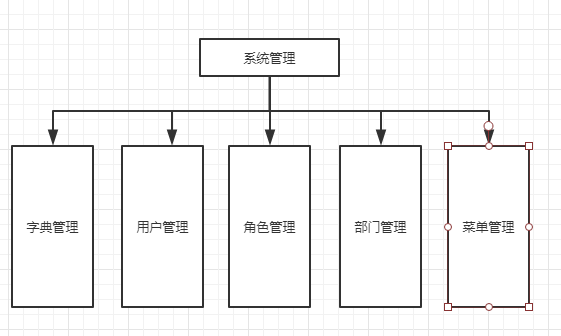


图2-1 系统管理组成结构图

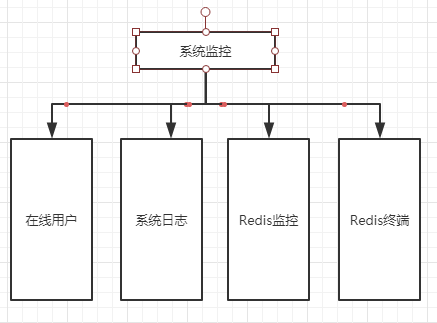


图2 -2系统监控组成结构图

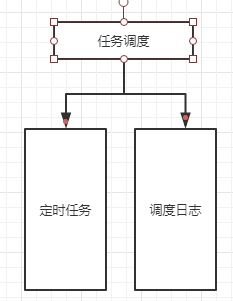


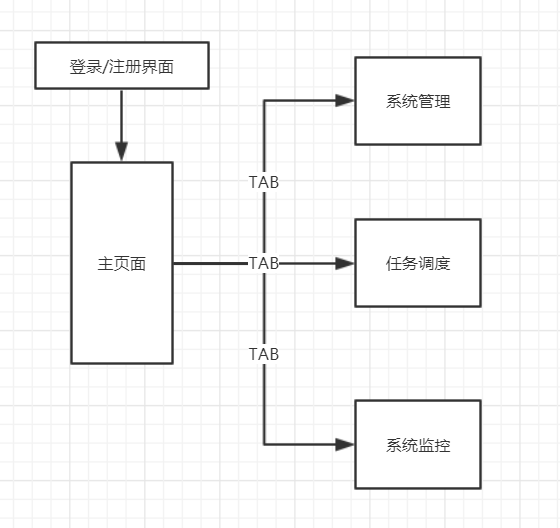
图2 -3任务调度组成结构图

# 用户界面设计

## 应当遵循的界面设计规范

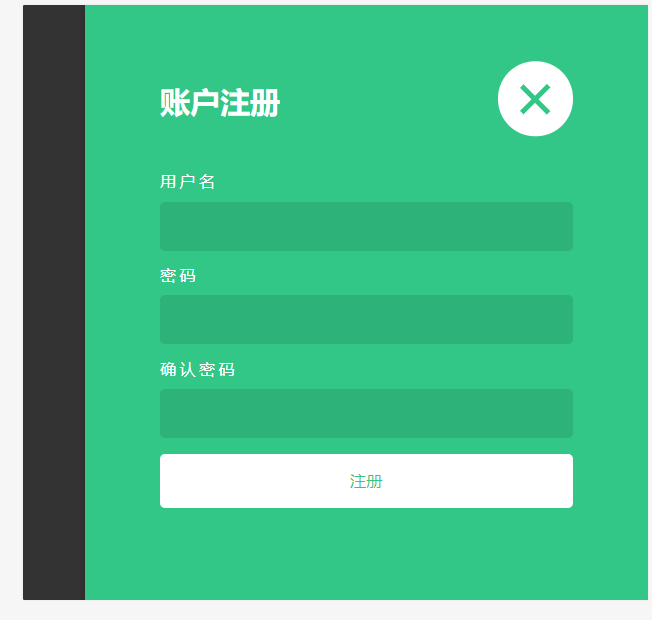
本系统前台页面参考谷歌的Material Design风格设计实现，界面简洁，逻辑清晰。

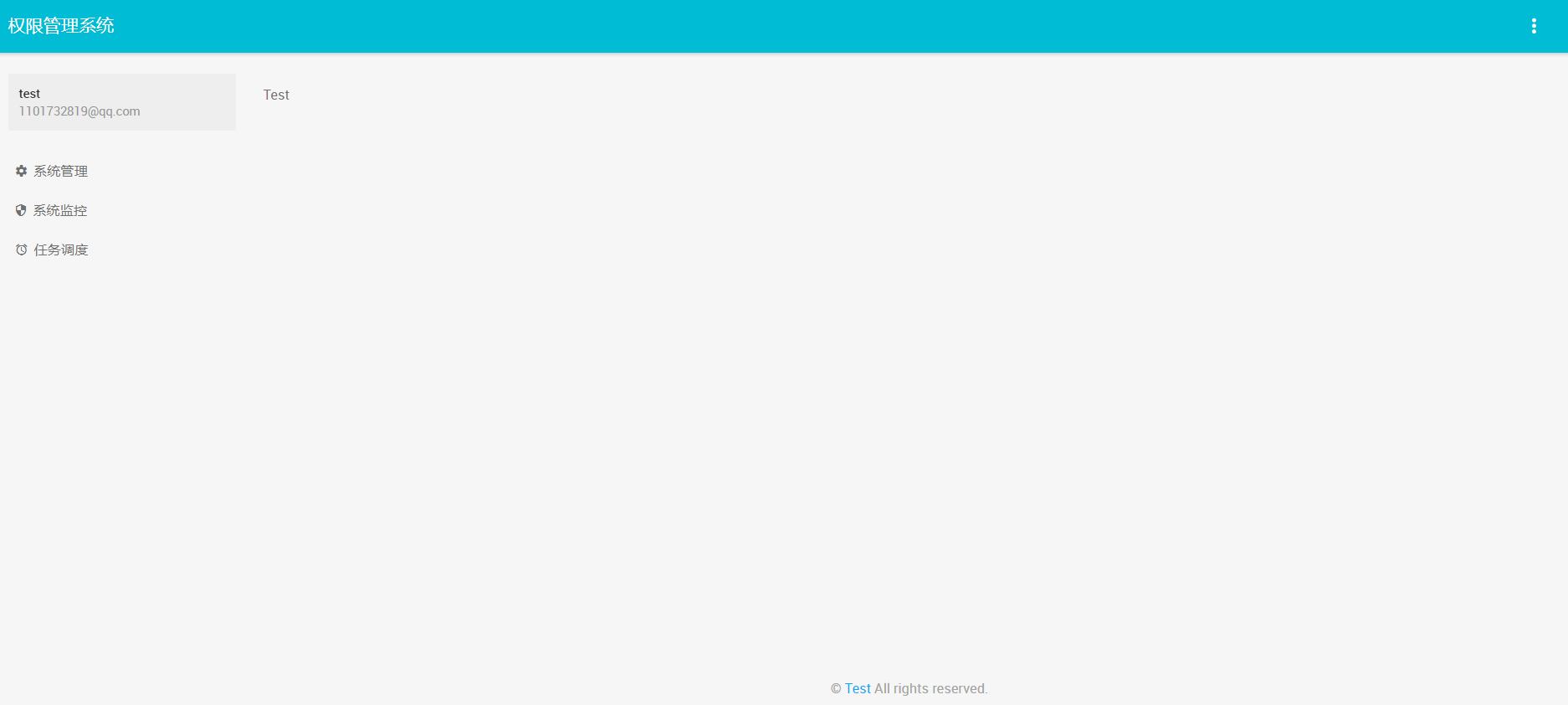
## 界面的关系图和工作流程图



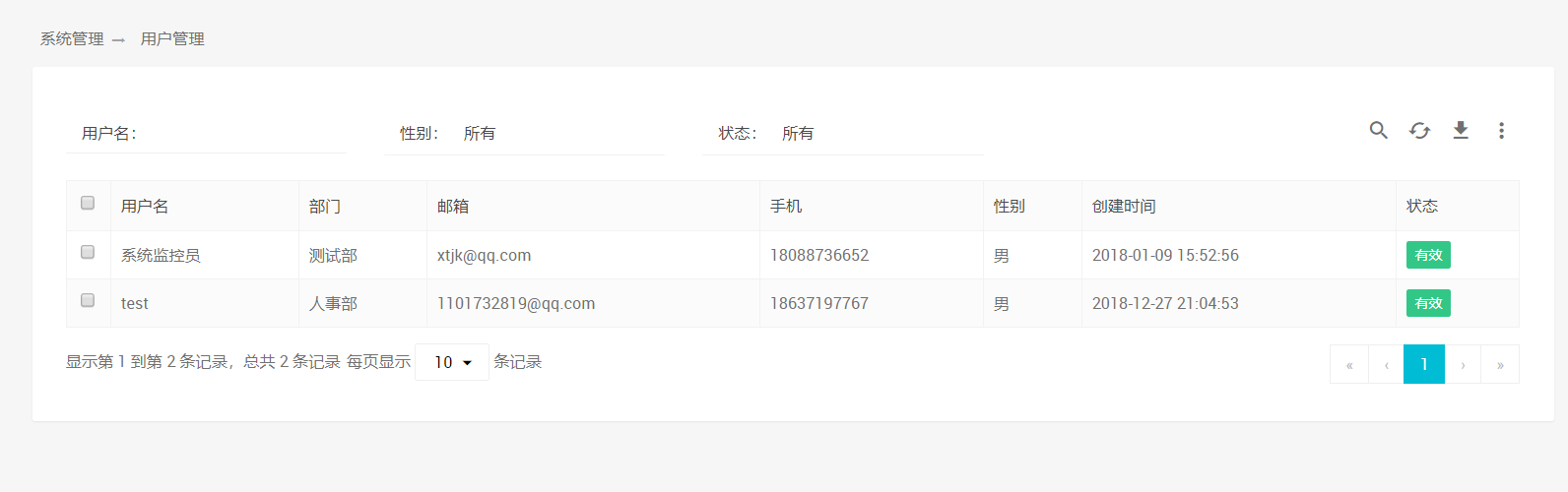
## 主界面







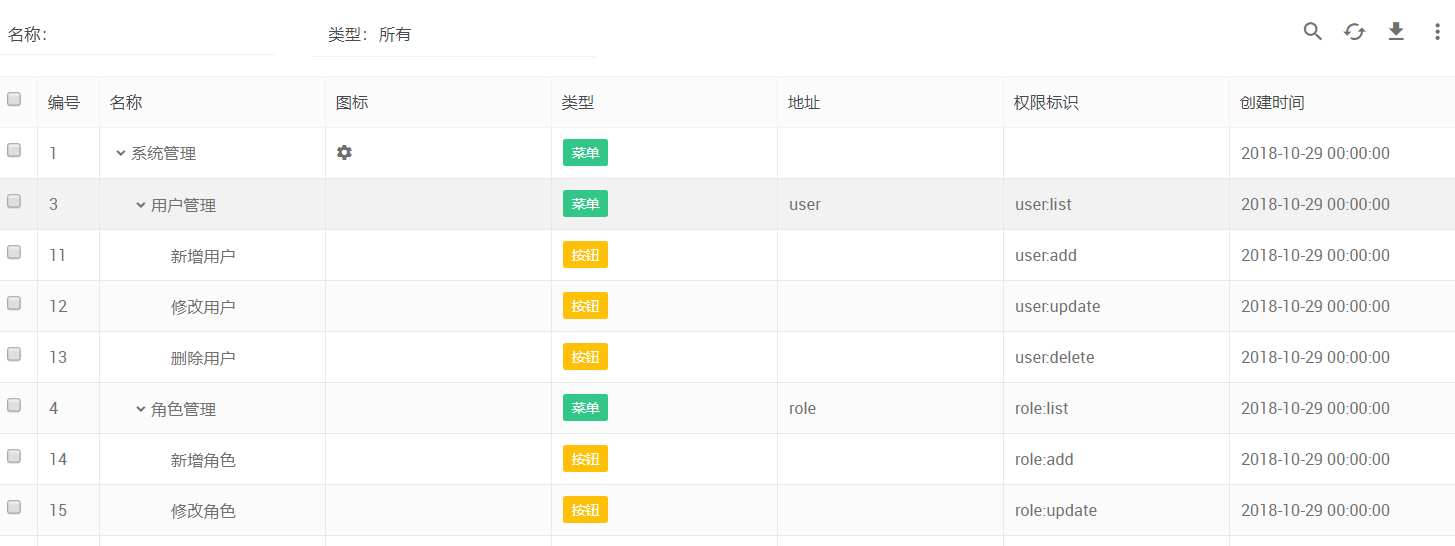
## 系统管理





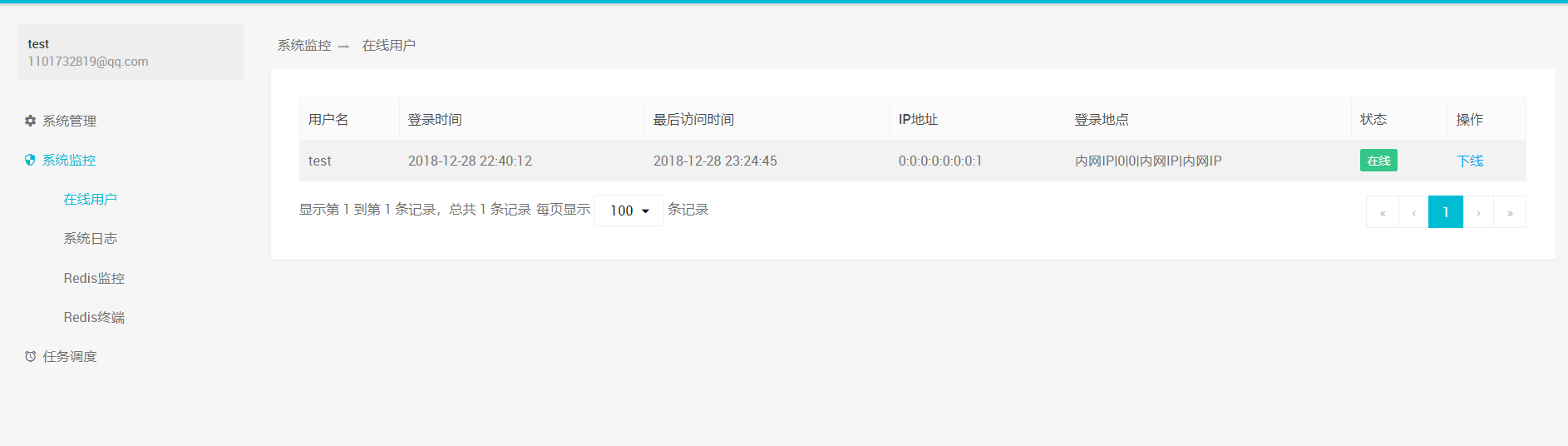


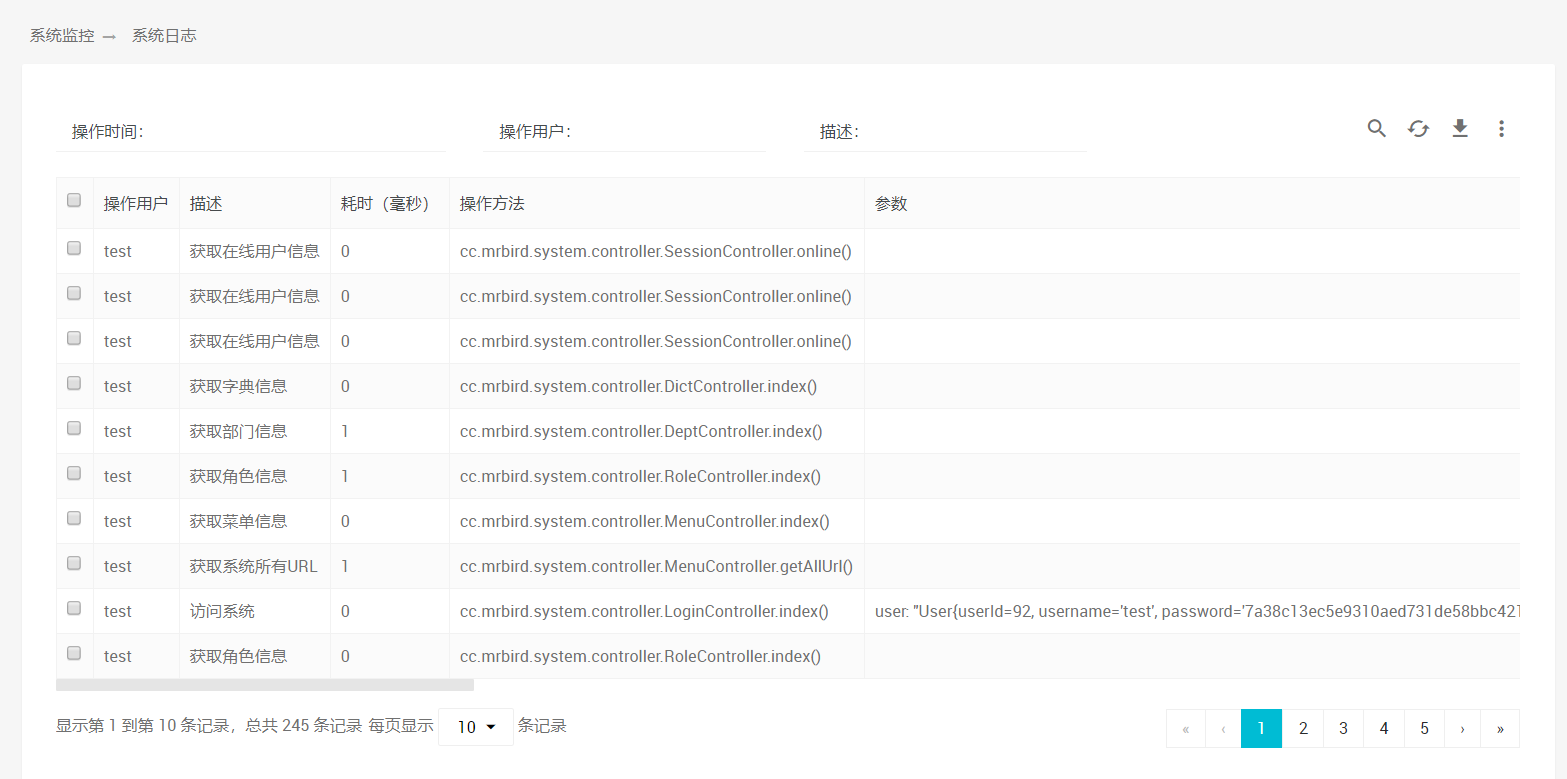


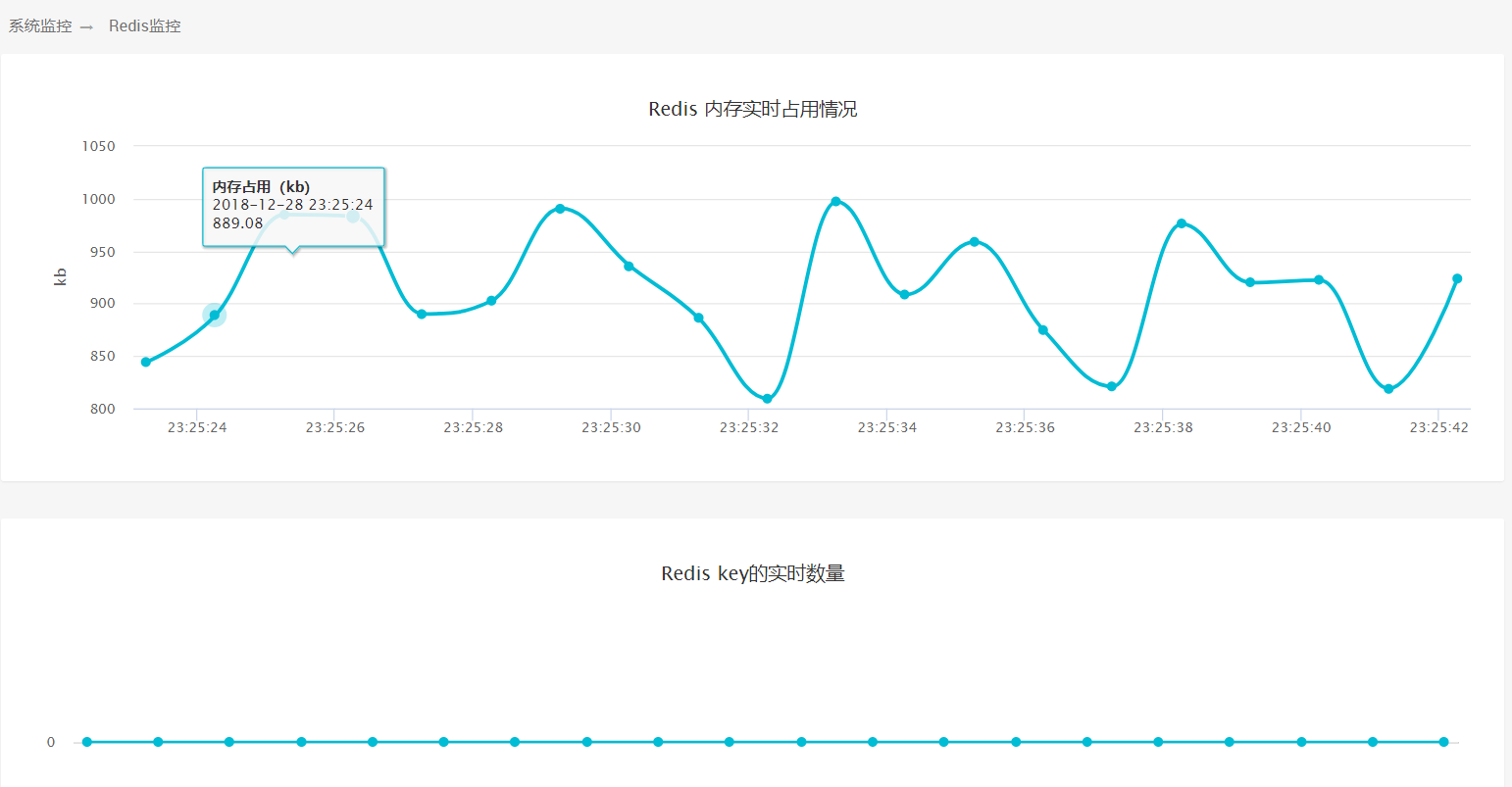




## 系统监控

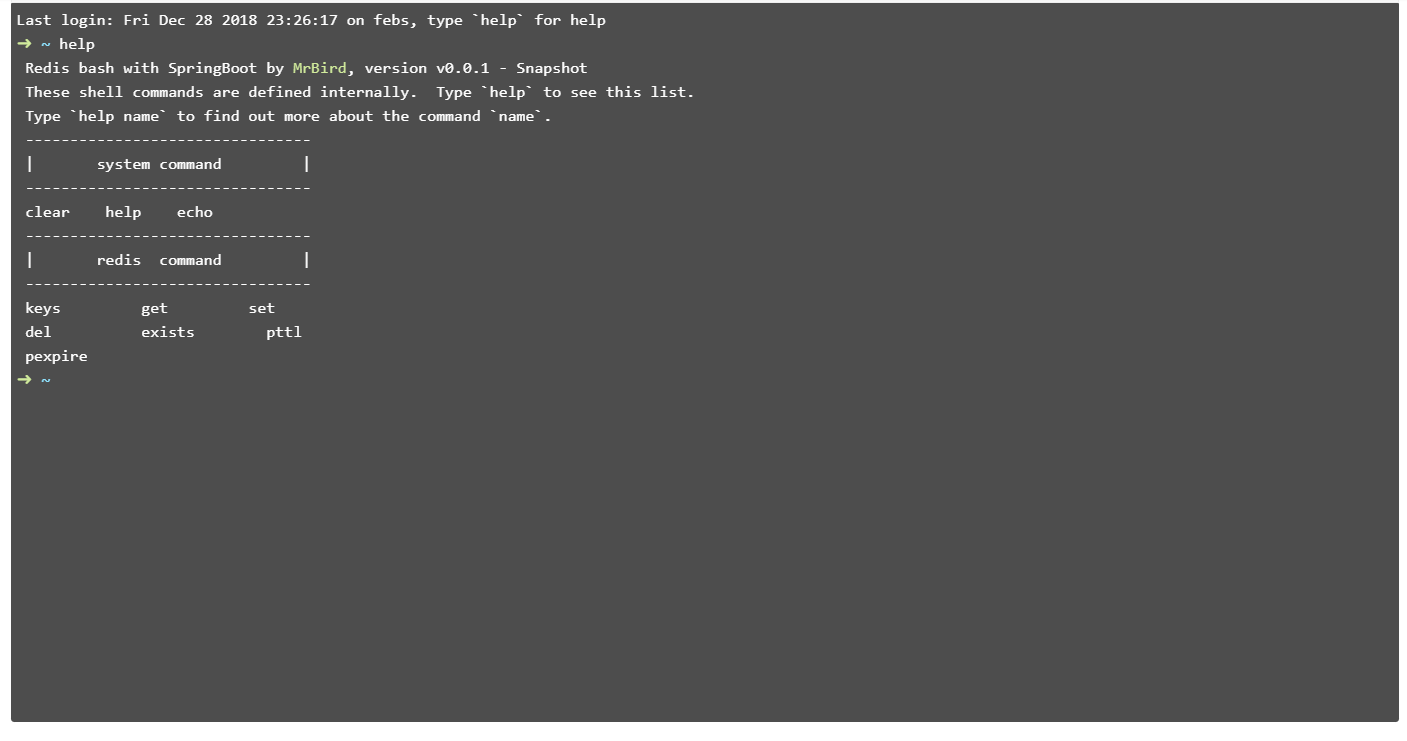








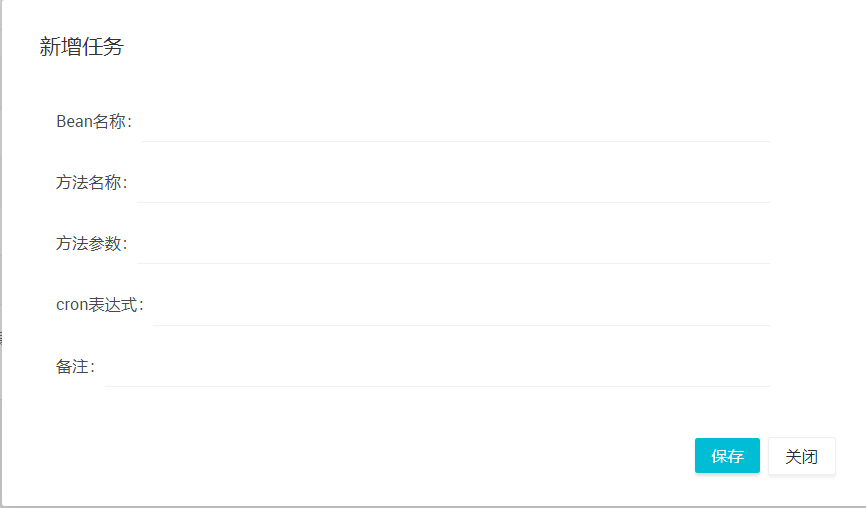


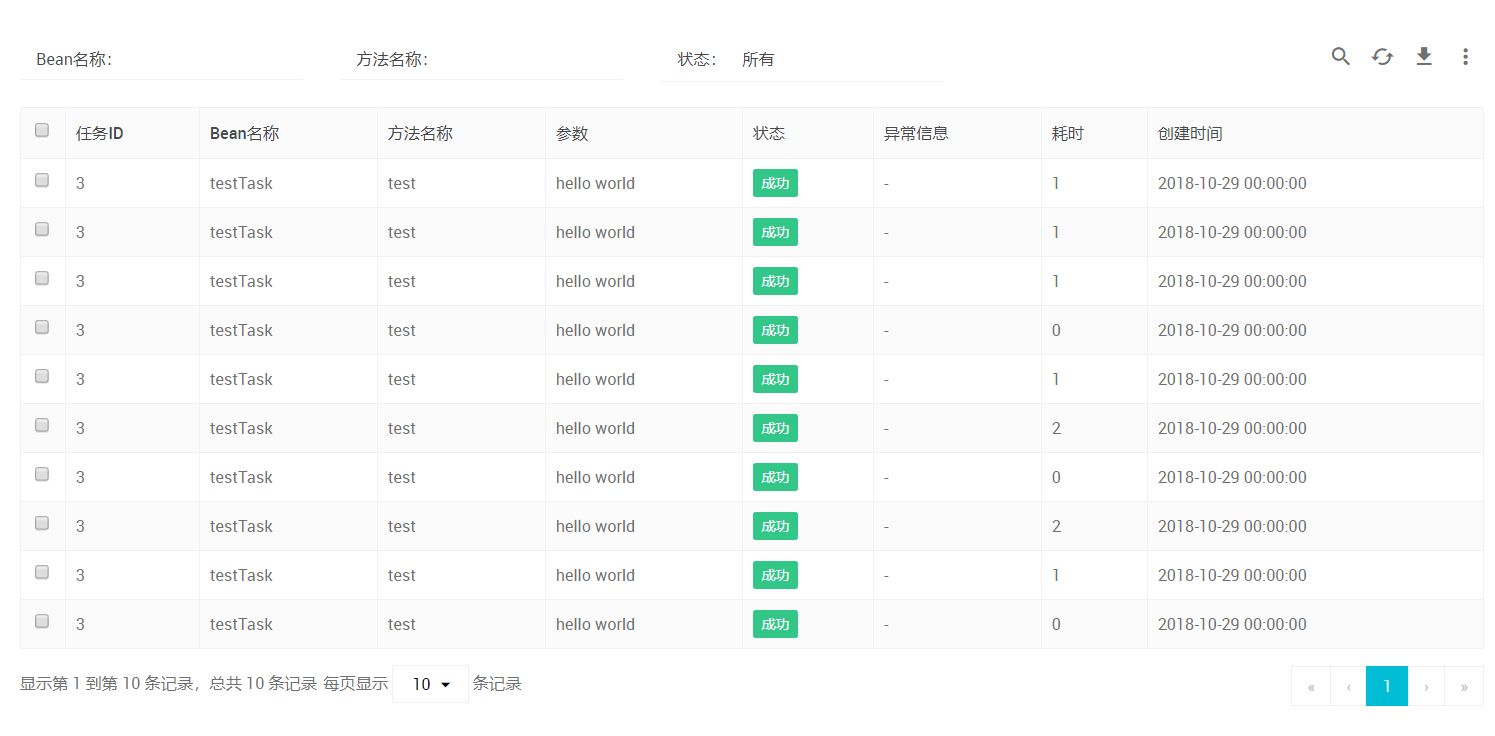


## 任务调度









# 模块设计

## 模块命名规则

模块命名按照英文加内容的命名规则,总共三个模块System/Common/Job，同时按照MVC架构进行命名限制。

## 模块汇总

|  |  |
| --- | --- |
| 登录注册以及系统管理模块 | |
| 模块名称 | 功能简述 |
| System.controller | 控制器,负责页面跳转的拦截控制 |
| System.service | 模块的服务层, |
| System.service.impl | 服务层的实现(接口的实现) |
| System.dao | 数据库对象操作层,ORM框架的实现内容 |
| System.domain | 数据库值对象 |
| 安全框架模块 | |
| 模块名称 | 功能简述 |
| Common.shrio | Shrio框架的配置类以及自定义ShrioRealm |
| 任务调度模块 | |
| 模块名称 | 功能简述 |
| Job.config | 定时任务的配置类 |
| Job.controller | 控制层,负责页面跳转控制 |
| Job.service | 服务层,提供接口 |
| Job.service.impl | 服务的实现 |
| Job.util | 任务模块的工具类 |
| Job.dao | 任务模块的数据库对象,ORM框架的实现内容 |
| Job.domain | 数据库值对象 |
| 异常模块 | |
| 模块名称 | 功能简述 |
| Common.exception | 系统异常的统一处理 |
| 工具包模块 | |
| 模块名称 | 功能简述 |
| Common.util | 集合了一些类似时间转换/MD5加密/Http等工具类 |
| Excel导出模块 | |
| 模块名称 | 功能简述 |
| Common.util.poi | POI的实现,负责从系统导出Excel文档 |

## 登陆注册及系统管理的模块设计

6.3.1 登录注册

|  |  |
| --- | --- |
| 模块名称 | 登录注册 |
| 功能描述 | 为用户提供登录注册服务 |
| 接口与属性 | public ResponseBo addUser |
| 数据结构  与算法 | 对用户登录密码进行MD5 加密:return new SimpleHash(ALGORITH\_NAME, pswd,ByteSource.Util.bytes(username.toLowerCase()+ SALT),HASH\_ITERATIONS).toHex();MD5以512位分组来处理输入的信息，且每一分组又被划分为16个32位子分组，经过了一系列的处理后，算法的输出由四个32位分组组成，将这四个32位分组级联后将生成一个128位散列值。 |

6.3.2 系统管理

|  |  |
| --- | --- |
| 模块名称 | 系统权限管理 |
| 功能描述 | 为用户针对数据库内容的增删改查操作 |
| 接口与属性 | public ResponseBo addUser(User user, Long[] roles);//新增用户  public ResponseBo updateUser(User user, Long[] rolesSelect);//修改用户  public ResponseBo deleteUsers(String ids);//删除用户  public Map<String, Object> roleList(QueryRequest request, Role role);//角色列表  public ResponseBo getRole(Long roleId);获取角色  public boolean checkRoleName(String roleName, String oldRoleName);//check  public ResponseBo addRole(Role role, Long[] menuId)//添加角色  public ResponseBo deleteRoles(String ids)//删除角色  public ResponseBo updateRole(Role role, Long[] menuId)//修改角色  public ResponseBo getMenu(String userName);获取菜单  public ResponseBo getMenu(Long menuId);  public ResponseBo getMenuButtonTree();获取菜单按钮列表  public ResponseBo getUserMenu(String userName);获取用户的菜单  public List<Menu> menuList(Menu menu);获取菜单列表  public boolean checkMenuName(String menuName, String type, String oldMenuName);check  public ResponseBo addMenu(Menu menu);//添加  public ResponseBo deleteMenus(String ids);//删除菜单  public ResponseBo updateMenu(Menu menu);//更新菜单  public List<Map<String, String>> getAllUrl();//获取系统所有URL  public ResponseBo getDeptTree();获取部门树  public ResponseBo getDept(Long deptId)//获取部门信息  public List<Dept> deptList(Dept dept)//部门列表  public boolean checkDeptName(String deptName, String oldDeptName)//check  public ResponseBo addDept(Dept dept)//添加部门  public ResponseBo deleteDepts(String ids)//删除部门  public ResponseBo updateRole(Dept dept)//更新部门  public Map<String, Object> dictList(QueryRequest request, Dict dict)//数据字典列表  public ResponseBo getDict(Long dictId);获取数据字典信息  public ResponseBo addDict(Dict dict)//添加字段  public ResponseBo deleteDicts(String ids)//删除字段  public ResponseBo updateDict(Dict dict)//更新字段 |
| 技术实现 | 以上方法均使用Shrio的权限认证功能, 首先1.对subject进行授权，调用方法isPermitted（"permission串"）2、SecurityManager执行授权，通过ModularRealmAuthorizer执行授权3、ModularRealmAuthorizer执行realm（自定义的CustomRealm）从数据库查询权限数据调用realm的授权方法：doGetAuthorizationInfo4、realm从数据库查询权限数据，返回ModularRealmAuthorizer 5、ModularRealmAuthorizer调用PermissionResolver进行权限串比对6、如果比对后，isPermitted中"permission串"在realm查询到权限数据中，说明用户访问permission串有权限，否则 没有权限，抛出异常。 |
| 补充说明 | 返回的对象为ResponseBo的均为Ajax异步调用,该对象包装有结果msg和对应的代码code. |

## 任务调度的模块设计

6.4.1 任务调度模块

|  |  |
| --- | --- |
| 模块名称 | 任务调度 |
| 功能描述 | 设置定时任务，任务管理 |
| 接口与属性 | public Map<String, Object> jobList(QueryRequest request, Job job);任务列表  public boolean checkCron(String cron)//check  public ResponseBo addJob(Job job)//添加  public ResponseBo deleteJob(String ids)//删除  public ResponseBo getJob(Long jobId)//获取任务  public ResponseBo updateJob(Job job)//更新任务  public ResponseBo runJob(String jobIds)//运行  public ResponseBo pauseJob(String jobIds)//暂停  public ResponseBo resumeJob(String jobIds)//恢复任务  public ResponseBo getSysCronClazz(Job job)//定时任务  public Map<String, Object> jobLogList(QueryRequest request, JobLog log)//任务日志列表  public ResponseBo deleteJobLog(String ids)；//删除任务日志 |
| 技术实现 | Scheduler:任务调度器，是实际执行任务调度的控制器。  Trigger：触发器，用于定义任务调度的时间规则，有SimpleTrigger,CronTrigger,DateIntervalTrigger和NthIncludedDayTrigger，其中CronTrigger用的比较多，本文主要介绍这种方式。CronTrigger在spring中封装在CronTriggerFactoryBean中。  Calendar:它是一些日历特定时间点的集合。一个trigger可以包含多个Calendar，以便排除或包含某些时间点。  JobDetail:用来描述Job实现类及其它相关的静态信息，如Job名字、关联监听器等信息。在spring中有JobDetailFactoryBean和 MethodInvokingJobDetailFactoryBean两种实现，如果任务调度只需要执行某个类的某个方法，就可以通过MethodInvokingJobDetailFactoryBean来调用。  Job：是一个接口，只有一个方法void execute(JobExecutionContext context),开发者实现该接口定义运行任务，JobExecutionContext类提供了调度上下文的各种信息。Job运行时的信息保存在JobDataMap实例中。实现Job接口的任务，默认是无状态的，若要将Job设置成有状态的，在quartz中是给实现的Job添加@DisallowConcurrentExecution注解（以前是实现StatefulJob接口，现在已被Deprecated）,在与spring结合中可以在spring配置文件的job detail中配置concurrent参数。 |

## 系统监控的模块设计

6.4.1 系统监控模块

|  |  |
| --- | --- |
| 模块名称 | 系统监控，主要包括日志和Redis |
| 功能描述 | 日志监控，Redis监控管理 |
| 接口与属性 | public Map<String, Object> logList(QueryRequest request, SysLog log)//日志列表  public ResponseBo deleteLogss(String ids)//删除日志  public String getRedisInfo(Model model)//Redis信息  public String redisTerminal(Model model)//Redis终端  public String getKeysSize()//获取键Size  public String getMemoryInfo()//获取内存信息  public ResponseBo keys(String arg)//执行Redis Keys命令  public ResponseBo get(String arg)//执行Redis get命令  public ResponseBo set(String arg)//执行Redis set命令  public ResponseBo del(String arg)//执行Redis del命令  public ResponseBo exists(String arg)//执行Redis exists命令  public ResponseBo pttl(String arg)//执行Redis pttl命令  public ResponseBo pexpire(String arg)//执行Redis pexpire命令  private static boolean isValidLong(String str)//判断 |

# 数据库设计

概念设计是对用户要求描述的现实世界，通过对其中功能的分类、聚集和概括，建立抽象的概念数据模型。这个概念模型应反映现实世界各功能的信息结构、信息流动情况、信息间的互相制约关系以及各部门对信息储存、查询和加工的要求等。所建立的模型应避开数据库在计算机上的具体实现细节，用一种抽象的形式表示出来。以扩充的实体—（E-R模型）联系模型方法为例，第一步先明确现实世界所含的各种实体及其属性、实体间的联系以及对信息的制约条件等，从而给出所用信息的局部描述(在数据库中称为用户的局部视图)。第二步再将前面得到的多个用户的局部视图集成为一个全局视图，建立数据库的每一幅用户视图。

## 数据库环境说明

Mysql5.6

## 数据库的命名规则

定时任务由qrtz开头,其余表由t开头,用\_隔开,

## 逻辑设计

采用RBAC模式建表,RBAC 是基于角色的访问控制（Role-Based Access Control ）在 RBAC 中，权限与角色相关联，用户通过成为适当角色的成员而得到这些角色的权限。这就极大地简化了权限的管理。这样管理都是层级相互依赖的，权限赋予给角色，而把角色又赋予用户，这样的权限设计很清楚，管理起来很方便。

以下是初始化SQL

DROP TABLE IF EXISTS `t\_dept`;

CREATE TABLE `t\_dept` (

`DEPT\_ID` bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '部门ID',

`PARENT\_ID` bigint(20) NOT NULL COMMENT '上级部门ID',

`DEPT\_NAME` varchar(100) NOT NULL COMMENT '部门名称',

`ORDER\_NUM` bigint(20) DEFAULT NULL COMMENT '排序',

`CREATE\_TIME` datetime DEFAULT NULL COMMENT '创建时间',

PRIMARY KEY (`DEPT\_ID`)

) ENGINE=InnoDB AUTO\_INCREMENT=9 DEFAULT CHARSET=utf8;

DROP TABLE IF EXISTS `t\_dict`;

CREATE TABLE `t\_dict` (

`DICT\_ID` bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '字典ID',

`KEYY` bigint(20) NOT NULL COMMENT '键',

`VALUEE` varchar(100) NOT NULL COMMENT '值',

`FIELD\_NAME` varchar(100) NOT NULL COMMENT '字段名称',

`TABLE\_NAME` varchar(100) NOT NULL COMMENT '表名',

PRIMARY KEY (`DICT\_ID`)

) ENGINE=InnoDB AUTO\_INCREMENT=37 DEFAULT CHARSET=utf8;

-- ----------------------------

-- Table structure for t\_job

-- ----------------------------

DROP TABLE IF EXISTS `t\_job`;

CREATE TABLE `t\_job` (

`JOB\_ID` bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '任务id',

`BEAN\_NAME` varchar(100) NOT NULL COMMENT 'spring bean名称',

`METHOD\_NAME` varchar(100) NOT NULL COMMENT '方法名',

`PARAMS` varchar(200) DEFAULT NULL COMMENT '参数',

`CRON\_EXPRESSION` varchar(100) NOT NULL COMMENT 'cron表达式',

`STATUS` char(2) NOT NULL COMMENT '任务状态 0：正常 1：暂停',

`REMARK` varchar(200) DEFAULT NULL COMMENT '备注',

`CREATE\_TIME` datetime DEFAULT NULL COMMENT '创建时间',

PRIMARY KEY (`JOB\_ID`)

) ENGINE=InnoDB AUTO\_INCREMENT=21 DEFAULT CHARSET=utf8;

-- ----------------------------

-- Table structure for t\_job\_log

-- ----------------------------

DROP TABLE IF EXISTS `t\_job\_log`;

CREATE TABLE `t\_job\_log` (

`LOG\_ID` bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '任务日志id',

`JOB\_ID` bigint(20) NOT NULL COMMENT '任务id',

`BEAN\_NAME` varchar(100) NOT NULL COMMENT 'spring bean名称',

`METHOD\_NAME` varchar(100) NOT NULL COMMENT '方法名',

`PARAMS` varchar(200) DEFAULT NULL COMMENT '参数',

`STATUS` char(2) NOT NULL COMMENT '任务状态 0：成功 1：失败',

`ERROR` text COMMENT '失败信息',

`TIMES` decimal(11,0) DEFAULT NULL COMMENT '耗时(单位：毫秒)',

`CREATE\_TIME` datetime DEFAULT NULL COMMENT '创建时间',

PRIMARY KEY (`LOG\_ID`)

) ENGINE=InnoDB AUTO\_INCREMENT=2476 DEFAULT CHARSET=utf8;

-- ----------------------------

-- Table structure for t\_log

-- ----------------------------

DROP TABLE IF EXISTS `t\_log`;

CREATE TABLE `t\_log` (

`ID` bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '日志ID',

`USERNAME` varchar(50) DEFAULT NULL COMMENT '操作用户',

`OPERATION` text COMMENT '操作内容',

`TIME` decimal(11,0) DEFAULT NULL COMMENT '耗时',

`METHOD` text COMMENT '操作方法',

`PARAMS` text COMMENT '方法参数',

`IP` varchar(64) DEFAULT NULL COMMENT '操作者IP',

`CREATE\_TIME` datetime DEFAULT NULL COMMENT '创建时间',

`location` varchar(50) DEFAULT NULL COMMENT '操作地点',

PRIMARY KEY (`ID`)

) ENGINE=InnoDB AUTO\_INCREMENT=860 DEFAULT CHARSET=utf8;

-- ----------------------------

-- Table structure for t\_menu

-- ----------------------------

DROP TABLE IF EXISTS `t\_menu`;

CREATE TABLE `t\_menu` (

`MENU\_ID` bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '菜单/按钮ID',

`PARENT\_ID` bigint(20) NOT NULL COMMENT '上级菜单ID',

`MENU\_NAME` varchar(50) NOT NULL COMMENT '菜单/按钮名称',

`URL` varchar(100) DEFAULT NULL COMMENT '菜单URL',

`PERMS` text COMMENT '权限标识',

`ICON` varchar(50) DEFAULT NULL COMMENT '图标',

`TYPE` char(2) NOT NULL COMMENT '类型 0菜单 1按钮',

`ORDER\_NUM` bigint(20) DEFAULT NULL COMMENT '排序',

`CREATE\_TIME` datetime NOT NULL COMMENT '创建时间',

`MODIFY\_TIME` datetime DEFAULT NULL COMMENT '修改时间',

PRIMARY KEY (`MENU\_ID`)

) ENGINE=InnoDB AUTO\_INCREMENT=112 DEFAULT CHARSET=utf8;

-- ----------------------------

-- Table structure for t\_role

-- ----------------------------

DROP TABLE IF EXISTS `t\_role`;

CREATE TABLE `t\_role` (

`ROLE\_ID` bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '角色ID',

`ROLE\_NAME` varchar(100) NOT NULL COMMENT '角色名称',

`REMARK` varchar(100) DEFAULT NULL COMMENT '角色描述',

`CREATE\_TIME` datetime NOT NULL COMMENT '创建时间',

`MODIFY\_TIME` datetime DEFAULT NULL COMMENT '修改时间',

PRIMARY KEY (`ROLE\_ID`)

) ENGINE=InnoDB AUTO\_INCREMENT=71 DEFAULT CHARSET=utf8;

-- ----------------------------

-- Table structure for t\_role\_menu

-- ----------------------------

DROP TABLE IF EXISTS `t\_role\_menu`;

CREATE TABLE `t\_role\_menu` (

`ROLE\_ID` bigint(20) NOT NULL COMMENT '角色ID',

`MENU\_ID` bigint(20) NOT NULL COMMENT '菜单/按钮ID'

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- ----------------------------

-- Table structure for t\_user

-- ----------------------------

DROP TABLE IF EXISTS `t\_user`;

CREATE TABLE `t\_user` (

`USER\_ID` bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '用户ID',

`USERNAME` varchar(50) NOT NULL COMMENT '用户名',

`PASSWORD` varchar(128) NOT NULL COMMENT '密码',

`DEPT\_ID` bigint(20) DEFAULT NULL COMMENT '部门ID',

`EMAIL` varchar(128) DEFAULT NULL COMMENT '邮箱',

`MOBILE` varchar(20) DEFAULT NULL COMMENT '联系电话',

`STATUS` char(1) NOT NULL COMMENT '状态 0锁定 1有效',

`CRATE\_TIME` datetime NOT NULL COMMENT '创建时间',

`MODIFY\_TIME` datetime DEFAULT NULL COMMENT '修改时间',

`LAST\_LOGIN\_TIME` datetime DEFAULT NULL COMMENT '最近访问时间',

`SSEX` char(1) DEFAULT NULL COMMENT '性别 0男 1女',

`THEME` varchar(10) DEFAULT NULL COMMENT '主题',

`AVATAR` varchar(100) DEFAULT NULL COMMENT '头像',

`DESCRIPTION` varchar(100) DEFAULT NULL COMMENT '描述',

PRIMARY KEY (`USER\_ID`)

) ENGINE=InnoDB AUTO\_INCREMENT=167 DEFAULT CHARSET=utf8;

-- ----------------------------

-- Table structure for t\_user\_role

-- ----------------------------

DROP TABLE IF EXISTS `t\_user\_role`;

CREATE TABLE `t\_user\_role` (

`USER\_ID` bigint(20) NOT NULL COMMENT '用户ID',

`ROLE\_ID` bigint(20) NOT NULL COMMENT '角色ID'

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

## 物理设计

4.0 表汇总

|  |  |
| --- | --- |
| 表名 | 功能说明 |
| t\_dept | 部门信息 |
| t\_dict | 数据字典信息 |
| t\_job | 任务信息 |
| t\_job\_log | 任务日志 |
| t\_log | 日志 |
| t\_menu | 菜单 |
| t\_role | 角色 |
| t\_role\_menu | 角色菜单关联表 |
| t\_user | 用户信息 |
| t\_user\_role | 用户角色关联表 |

4.1 表t\_dept

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | t\_dept | | |
| 列名 | 数据类型（精度范围） | 空/非空 | 约束条件 |
| DEPT\_ID | bigint | Not Null | PRIMARY KEY |
| PARENT\_ID | bigint | Not Null |  |
| DEPT\_NAME | varchar(100) | Not Null |  |
| ORDER\_NUM | bigint |  |  |
| CREATE\_TIME | datetime |  |  |

4.2 表t\_dict

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | t\_dict | | |
| 列名 | 数据类型（精度范围） | 空/非空 | 约束条件 |
| DICT\_ID | bigint | Not Null | PK，自增 |
| KEYY | bigint | Not Null |  |
| VALUEE | varchar(100) | Not Null |  |
| FIELD\_NAME | varchar(100) | Not Null |  |
| TABLE\_NAME | varchar(100) | Not Null |  |

4.3 表t\_job

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | t\_job | | |
| 列名 | 数据类型（精度范围） | 空/非空 | 约束条件 |
| JOB\_ID | bigint | Not Null | PK,自增 |
| BEAN\_NAME | bigint | Not Null |  |
| METHOD\_NAME | varchar(100) | Not Null |  |
| PARAMS | varchar(100) | Not Null |  |
| CRON\_EXPRESSION | varchar(100) | Not Null |  |
| STATUS | char(2) | Not Null |  |
| REMARK | varchar(200) |  |  |
| CREATE\_TIME | datetime |  |  |

4.4 表t\_job\_log

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | t\_job\_log | | |
| 列名 | 数据类型（精度范围） | 空/非空 | 约束条件 |
| LOG\_ID | bigint | Not Null | PK,自增 |
| JOB\_ID | bigint | Not Null |  |
| BEAN\_NAME | varchar(100) | Not Null |  |
| METHOD\_NAME | varchar(100) | Not Null |  |
| PARAMS | varchar(100) |  |  |
| STATUS | char(2) | Not Null |  |
| ERROR | text |  |  |
| TIMES | decimal(11,0) |  |  |
| CREATE\_TIME | datetime |  |  |

4.5 表t\_log

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | t\_log | | |
| 列名 | 数据类型（精度范围） | 空/非空 | 约束条件 |
| ID | bigint | Not Null | PK,自增 |
| USERNAME | varchar(50) | Not Null |  |
| OPERATION | text |  |  |
| TIME | decimal(11,0) |  |  |
| METHOD | text |  |  |
| PARAMS | text |  |  |
| IP | varchar(64) |  |  |
| CREATE\_TIME | datetime |  |  |
| location | varchar(50) |  |  |
|  |  |  |  |
|  |  |  |  |

4.6 表t\_menu

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | t\_ menu | | |
| 列名 | 数据类型（精度范围） | 空/非空 | 约束条件 |
| MENU\_ID | bigint | Not Null | PK自增 |
| PARENT\_ID | bigint | Not Null |  |
| MENU\_NAME | varchar(100) | Not Null |  |
| URL | varchar(100) |  |  |
| PERMS | text | Not Null |  |
| ICON | varchar(50) |  |  |
| TYPE | char(2) | Not Null |  |
| ORDER\_NUM | bigint |  |  |
| CREATE\_TIME | datetime | Not Null |  |
| MODIFY\_TIME | datetime |  |  |
|  |  |  |  |

4.7 表t\_role\_menu

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | t\_role\_menu | | |
| 列名 | 数据类型（精度范围） | 空/非空 | 约束条件 |
| ROLE\_ID | bigint | Not Null |  |
| MENU\_ID | bigint | Not Null |  |

4.8 表t\_user

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | t\_user | | |
| 列名 | 数据类型（精度范围） | 空/非空 | 约束条件 |
| USER\_ID | bigint | Not Null | PK自增 |
| USERNAME | varchar(50) | Not Null |  |
| PASSWORD | varchar(128) | Not Null |  |
| DEPT\_ID | bigint |  |  |
| EMAIL | varchar(128) |  |  |
| MOBILE | varchar(20) |  |  |
| STATUS | char(1) |  |  |
| CRATE\_TIME | datetime | Not Null |  |
| MODIFY\_TIME | datetime |  |  |
| LAST\_LOGIN\_TIME | datetime |  |  |
| SSEX | char(1) |  |  |
| THEME | varchar(10) |  |  |
| AVATAR | varchar(100) |  |  |
| DESCRIPTION | varchar(100) |  |  |

4.9 表t\_user\_role

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | t\_user\_role | | |
| 列名 | 数据类型（精度范围） | 空/非空 | 约束条件 |
| USER\_ID | bigint | Not Null |  |
| ROLE\_ID | bigint | Not Null |  |

## 安全性设计

### 用户帐号密码的加密方法

系统中对用户密码进行了MD5加密，比如用户test：

test 7a38c13ec5e9310aed731de58bbc4214 5 1101732819@qq.com 18637197767 1 2018-12-27 21:04:53 2018-12-28 14:23:14 2018-12-30 17:50:49 0 cyan default.jpg

他的密码已经进行过加密，如下为加密代码：（salt默认为”test”）

SimpleHash(ALGORITH\_NAME, pswd, ByteSource.Util.bytes(username.toLowerCase() + SALT),

HASH\_ITERATIONS).toHex()

### 角色与权限

仅部分角色,admin可以新增用户,并赋予其不同的权限

|  |  |  |
| --- | --- | --- |
| 角色 | 可以访问的表与列 | 操作权限 |
| admin | 所有表 | 管理员,所有权限 |
| 系统监控员 | 所有表 | 可查看系统监控信息，但不可操作 |

# 系统测试

## 测试基础

测试是软件质量保证的关键阶段，是在软件投入运行前，对软件的需求分析、设计、实现编码的最终检查，是为了发现错误而执行程序的过程。

1) 测试的目的

最终目的是建立一个可靠性高的软件系统。

2) 测试方法

本系统的测试用到了静态测试和动态测试。动态测试一般用两种方法：一是黑盒测试法，一是白盒测试法。

3) 测试过程



## 各模块测试

各模块测试就是把各个模块看成一个小整体，采用静态测试和白盒测试法来测试各模块是否满足要求。

首先是对系统设置模块的测试，当输入用户名“admin”和密码“123456”并且选择“管理员”按钮时，就可以管理员的身份进入后台操作系统；当登录为“系统监控员”，就可以以一般用户的身份进入前台管理系统。此模块满足要求。

其次对信息管理模块的测试，当点击“信息管理”按钮时，可进入信息管理模块；

对一般用户模块的测试同对管理员模块的测试，测试后基本无误。

通过次测试，保证了数据能正确通过各模块，即实参和形参的参数个数、数据类型及对应关系是一致的。

## 集成测试

单个模块能正常，组装后不一定能正常工作，所以在各模块测试的基础上，将所有模块按照设计要求组装成一个完整的系统进行测试。本系统采用黑盒测试方法及自顶向下测试方法，从主模块开始，沿程序的控制路径逐步向下，挨个进行各个功能的测试，比如在后台数据库管理系统中添加一个用户，则在前台管理系统中就能看到所添加的客户信息，表明此系统没有毛病，满足用户的要求。经过一系列测试后结果基本无误。

## 路径测试检查表

|  |  |
| --- | --- |
| **检查项** | **结论** |
| 数据类型问题  （１）变量的数据类型有错误吗？  （２）存在不同数据类型的赋值吗？  （３）存在不同数据类型的比较吗？ | 没有类型错误  不存在  不存在 |
| 变量值问题  （１）变量的初始化或缺省值有错误吗？  （２）变量发生上溢或下溢吗？  （３）变量的精度不够吗？ | 没有  不发生  没有 |
| 逻辑判断问题  （１）由于精度原因导致比较无效吗？  （２）表达式中的优先级有误吗？  （３）逻辑判断结果颠倒吗？ | 没有  没有  不颠倒 |
| 循环问题  （１）循环终止条件不正确吗？  （２）无法正常终止（死循环）吗？  （３）错误地修改循环变量吗？  （４）存在误差累积吗？ | 正确  循环正常  无  不存在 |
| 错误处理问题  （１）忘记进行错误处理吗？  （２）错误处理程序块一直没有机会被运行？  （３）错误处理程序块本身就有毛病吗？如报告的错误与实际错误不一致，处理方式不正确等等。  （４）错误处理程序块是“马后炮”吗？如在被它被调用之前软件已经出错。 | 尚未出现此类问题 |

## 8.5图形用户界面测试用例

|  |  |
| --- | --- |
| 检查项 | 测试人员的类别及其评价 |
| 窗口切换、移动、改变大小时正常吗？ | 正常 |
| 各种界面元素的文字正确吗？（如标题、提示等） | 正确 |
| 各种界面元素的状态正确吗？（如有效、无效、选中等状态） | 正确 |
| 各种界面元素支持键盘操作吗？ | 支持 |
| 各种界面元素支持鼠标操作吗？ | 支持 |
| 对话框中的缺省焦点正确吗？ | 正确 |
| 数据项能正确回显吗？ | 可以 |
| 对于常用的功能，用户能否不必阅读手册就能使用？ | 可以 |
| 执行有风险的操作时，有“确认”、“放弃”等提示吗？ | 有 |
| 各种界面元素的布局合理吗？美观吗？ | 合理美观 |
| 各种界面元素的颜色协调吗？ | 颜色协调 |
| 各种界面元素的形状美观吗？ | 美观 |
| 字体美观吗？ | 美观 |
| 图标直观吗？ | 直观 |

# 结论

从系统调查到查资料再到系统的分析设计，经历了这么久的时间，经过这段时间的努力，这个系统终于是完成了。在实习项目设计的这段时间内，感觉在无锡基地培训期间所学的知识得到了很好的发挥，最重要的是学到了很多对自己今后的发展起推动作用的东西。自己各方面有了很大的提高，开发编程思想有了深化，开发经验也逐渐丰富。之后在易保的实习阶段学习到了更全面更专业的企业项目知识,加深了自己对软件开发行业的了解,希望有一天自己能够成为软件开发行业的一名有贡献程序员。

参 考 文 献

[1] 张鑫旭. CSS世界[M],人民邮电出版社，2017.

[2] 杨开振. 深入浅出Spring Boot 2.x[M]，人民邮电出版社，2018.

[3] 疯狂软件. Spring Boot 2企业应用实战[M]. 电子工业出版社，2018.

[4] 杨家炜.基于Spring Boot的web设计与实现[J].轻工科技，2016(7):86-89.