**本科毕业论文（设计）**

毕业设计是一个漫长的过程，有需要可以联系微信，选题最好提前沟通，有可能当你不了解难度乱写一个题目或者开题报告里技术和功能写错，后期有可能就不一定能做啦，全职代写一直陪你到答辩结束，全程答疑，可以看朋友圈案例。

**毕业设计全程指导包括 选题/开题报告/任务书/程序（代码）/论文/答辩指导**

**联系微信：hao609719**

本论文模板只是参考学习，切勿直接使用，论文已经入驻各大论文平台，分享的人数也很多，查重时将不会通过，抽查出后果比较严重，希望大家以学习为主，需要学习怎么写论文，以及需要论文画图工具可看我主页视频学习。

|  |  |
| --- | --- |
| **论文（设计）题目：** | 基于SSM的XXXX市福利院管理系统的设计与实现 |

|  |  |
| --- | --- |
| **学 院：** |  |
| **专 业：** |  |
| **班 级：** |  |
| **学 号：** |  |
| **学生姓名：** |  |
| **指导教师：** |  |

**2023年4月25日**

XXXXX学院毕业论文（设计）

诚信责任书

本人郑重声明：本人所呈交的毕业论文（设计），是在导师的指导下独立进行研究所完成。毕业论文（设计）中凡引用他人已经发表或未发表的成果、数据、观点等，均已明确注明出处。

特此声明。

论文（设计）作者签名：

日期：2023年 4 月 25 日

基于SSM的XXX市福利院管理系统的设计与实现

# 摘要

随着当前福利院事业的发展之迅速，时代不断的进步、经济在不断的增长、科技在不断的创新。福利院机构信息化、标准化、规范化建设的要求日益迫切，紧随着“互联网+”技术的兴起。国内的部分福利院在业务管理上，还有停留在传统的人工记录管理方式和纸质单据流转的阶段，传统的人工记录管理方式存在着信息不能共享，信息之间流转不方便的弊端。因此，针对六盘水市福利院的管理模式，以福利院的主要业务为线索，研究设计一款基于SSM的六盘水市福利院管理系统，其目的在于优化建设福利院管理模式，意在使信息与信息之间流转通畅，信息采集方便。

论文介绍了福利院管理系统的背景及现状，对技术进行分析，选用系统开发技术，通过需求分析得出工作流程，明确实现系统的登录、管理、居住、后勤管理、医疗工作站五个模块的功能。系统采用B/S模式，使用Thymeleaf、JavaScript、HTML、CSS3、Bootstrap作为前端技术，后端采用SpringBoot+MyBatisPlus（MyBatis扩展版）+Sa-Toke安全框架技术。通过设计该系统可以进一步提高福利院机构的管理服务工作标准化、信息化和规范化。在推进社会主义现代化建设新发展的同时，做好福利院事业管理工作。

关键词：SSM；福利院管理；B/S

LITERATURE REVIEW OF LIUPANSHUI WELFARE HOUSE MANAGEMENT SYSTEM BASED ON SSM

# Abstract

With the rapid development of welfare institutions, The Times are progressing, the economy is growing, and science and technology are innovating.With the rise of “Internet+” technology, the informatization, standardization and standardization construction of welfare institutions are increasingly urgent. In terms of business management, some domestic welfare homes still remain in the stage of traditional manual record management and paper document circulation. The traditional manual record management has the disadvantages of information sharing and inconvenient information transfer. Therefore, aiming at the management mode of Liupanshui Welfare home, taking the main business of the welfare home as the clue, the research and design of a MANAGEMENT system of Liupanshui Welfare home based on SSM is aimed at optimizing the construction of welfare home management mode, aiming at making the flow of information smooth and information collection convenient.

This paper introduces the background and current situation of the welfare home management system, analyzes the technology, selects the system development technology, obtains the work flow through the demand analysis, and clearly realizes the functions of the system login, management, residence, logistics management, medical workstation five modules. B/S mode is adopted, Thymeleaf, JavaScript, HTML, CSS3 and Bootstrap are used as the front-end technology, and SpringBoot+MyBatisPlus(MyBatis Extension version)+ SA-TOKE security framework technology is used as the back-end. Through the design of the system can further improve the welfare institutions management service standardization, information and standardization. While promoting the new development of socialist modernization, we will do a good job in the management of welfare institutions.

Keywords:SSM;Administration of welfare homes;B/S

目 录

[摘要 I](#_Toc6499)

[Abstract II](#_Toc32201)

[第1章 绪论 1](#_Toc29381)

[1.1 研究目的和意义 1](#_Toc11194)

[1.1.1 研究目的 1](#_Toc32207)

[1.1.2 研究意义 2](#_Toc676)

[1.2 国内外研究现状和趋势 2](#_Toc27927)

[1.2.1 国外研究现状和趋势 2](#_Toc17498)

[1.2.2 国内研究现状和趋势 3](#_Toc13935)

[第2章 基于SSM的六盘水市福利院管理系统的关键技术 5](#_Toc1412)

[2.1 系统开发环境 5](#_Toc16760)

[2.2 后台开发框架SSM框架 5](#_Toc1651)

[2.2.1 Sping框架 5](#_Toc12200)

[2.2.2 Spring MVC框架 6](#_Toc25483)

[2.2.3 Spring Boot框架 7](#_Toc12689)

[2.2.4 MyBatis-Plus数据库持久层框架 7](#_Toc25057)

[2.2.5 Thymeleaf模板引擎框架 7](#_Toc15223)

[2.2.6 Spring Security安全框架 7](#_Toc526)

[2.3 LayUI前端框架 8](#_Toc15219)

[2.4 系统开发工具 8](#_Toc4667)

[2.4.1 IDEA 8](#_Toc17835)

[2.4.2 Apache Maven项目管理 8](#_Toc501)

[第3章 基于SSM的六盘水市福利院管理系统的分析 9](#_Toc3955)

[3.1 可行性分析 9](#_Toc3412)

[3.1.1 技术可行性 9](#_Toc6226)

[3.1.2 操作可行性 9](#_Toc16997)

[3.1.3 经济可行性 9](#_Toc6200)

[3.2 功能需求分析 9](#_Toc15112)

[3.2.1 系统用户 10](#_Toc17521)

[3.2.2 用户功能描述 10](#_Toc30365)

[第4章 基于SSM的六盘水市福利院管理系统的设计 13](#_Toc18706)

[4.1 系统总体设计原则 13](#_Toc23093)

[4.2 系统功能模块设计 13](#_Toc760)

[4.3 系统详细设计 13](#_Toc21278)

[4.3.1 登录模块设计 13](#_Toc9869)

[4.3.2 管理模块设计 14](#_Toc14087)

[4.3.3 居住模块设计 19](#_Toc8692)

[4.3.4 后勤管理模块设计 21](#_Toc5647)

[4.3.5 医疗工作站模块设计 23](#_Toc15687)

[4.4 数据库设计 25](#_Toc5753)

[4.4.1 E-R图设计 25](#_Toc20999)

[4.4.2 数据表设计 28](#_Toc28383)

[第5章 基于SSM的六盘水市福利院管理系统的实现 43](#_Toc31486)

[5.1 业务功能实现 43](#_Toc31512)

[5.2 登录模块实现 43](#_Toc26645)

[5.3 管理模块实现 44](#_Toc2348)

[5.4 居住模块实现 45](#_Toc7661)

[5.5 后勤管理模块实现 47](#_Toc9994)

[5.6 医疗工作站模块实现 49](#_Toc26433)

[第6章 基于SSM的六盘水市福利院管理系统的测试 51](#_Toc16100)

[6.1 测试目的 51](#_Toc11865)

[6.2 单元测试 51](#_Toc32592)

[6.3 功能模块测试 52](#_Toc26392)

[第7章 结论 59](#_Toc13738)

[7.1 主要结论 59](#_Toc13927)

[7.2 研究展望 59](#_Toc13384)

[参考文献 61](#_Toc10763)

[致 谢 63](#_Toc27515)

[附 录 65](#_Toc11312)

# 第1章 绪论

## 1.1 研究目的和意义

### 1.1.1 研究目的

随着当前福利院事业发展的状况，速度在不断的飞跃。在福利院发展的进程上，反映出来了这个国家的社会文明在不停的进步，为当前社会创造了最基本的保障。同时，在整个社会都提倡关注社会中弱势群体的情况下，全社会在不停的为福利院事业提供强有的力量。也在不断的改变福利院的具体服务形式。伴随着“残疾儿童、孤寡老人、孤儿”的生活医疗等问题，针对无人照顾的“残疾儿童、孤寡老人、孤儿”等人员，福利院是一个能够为他们提供生活医疗住宿的地方。无人照顾的这些人群，在生活中，难免会有遇到生病的时候，比如感冒、发烧或者是一些突发疾病。这群人在生活中不能自理，许多时候可能会发生病症。在发生病症没有人照顾的情况下，不能第一时间赶到医院进行有效的治疗。如果居家的话在医疗方面技术水平存在不足，这样的话，患者就错过了一些宝贵的时间。尤其是部分慢性疾病和卧床不起的“残疾儿童、孤寡老人、孤儿”住在福利院不仅能享受专业职员提供日常生活和医疗服务，万一有紧急情况医院也能立即抢救[1]。在当前社会已有多种档次福利院以发展迅速，福利院在不停增加床位，以满足社会需求，在不停探索和展现出更细致化的老年公寓，儿童公寓，以房养老型，休闲养生型[1]。大部分“残疾儿童、孤寡老人、孤儿”不是重视在物质生活上，而是需要一个丰富多彩的生活。随着农村青壮年劳动力逐渐转入城市，导致农村出现了新问题，农村“残疾儿童、孤寡老人、孤儿”便成为社会一个大问题。于是“残疾儿童、孤寡老人、孤儿”等人员便只能依靠福利院的帮助，而传统的人工管理信息方式已不能满足当今时代的管理需求，如何使福利院信息管理更加标准化、规范化、信息化，减少管理成本，提高福利院工作人员的工作质量便成为福利院服务工作的重要问题。

就当前六盘水市而言，“残疾儿童、孤寡老人、孤儿”等人员入住福利院需要做信息登记、出入登记、平时各项工作登记等是一个重要的问题。在福利院各项工作开展过程中，人工登记办理，难免登记信息出现缺漏，纸质人工管理方式存放信息难免对后期工作带来不便。随着时代科技进步，信息技术的不断发展，信息化管理方式已是大势所趋。因此，设计一个基于SSM的六盘水市福利院管理系统，充分利用互联网技术，为六盘水市福利院入住人员信息进行统一信息化管理，减轻福利院工作人员的工作压力，为福利院人员管理工作提高效率和管理水平[2]。

### 1.1.2 研究意义

在目前社会经济的突飞猛进下，福利院信息管理的完善也是在逐步上升，现代医疗技术水平的不断提高，以至于影响到了福利院内部发展得到了更好的帮助。随着“互联网+”技术的兴起，抓住时代的契机[3]。设计开发一款基于SSM的六盘水市福利院管理系统，实现福利院的信息管理、居住管理、医护管理、后勤管理和系统管理等方面的集中管理[3]。做一个具有高效性、稳定性、安全可靠和良好的信息管理系统，该系统的设计和开发以福利院日常业务为基础，福利院的具体开发做需求分析，然后对福利院管理系统进行具体的设计。通过设计福利院管理系统，使福利院工作人员在工作中，提高工作效率。以更加有效的方法对福利院人员信息标准化、信息化、规范化管理。以减少不必要的工作流程，随着福利院事业发展的不断推广进行，管理系统发展迅速，利用“互联网+”技术管理信息，福利院工作人员可以通过该系统增加对福利院信息查看的透明度。福利院管理系统能够有效的对福利院的日常业务进行网络化管理[4]。因此，开发设计一个福利院管理系统具有非常重要的意义。

论文通过对目前现有的福利院管理系统的研究和学习，并对福利院运行特点以及性能进行分析，旨在建立一套福利院管理系统，通过福利院对于管理方面的特定需求，来配置相应的行为功能，从而更好的对福利院进行管理。

## 1.2 国内外研究现状和趋势

### 1.2.1 国外研究现状和趋势

目前，国外福利院管理系统发展日趋成熟，在福利院管理系统上，已完善功能需求，在模块中分类明确，国外的福利院形式都有一定的不同之处，但是共同点都是注重“残疾儿童、孤寡老人、孤儿”的生活环境和精神娱乐等方面[4]。而不仅仅是为了保障“残疾儿童、孤寡老人、孤儿”的基本生活的需要。有的发达国家对“残疾儿童、孤寡老人、孤儿”的生活条件这块十分重视，在系统上得以优化完善，节约工作成本，增加信息之间的联系性、互通性，加快信息流转通畅性，已有效的运用了“互联网+”技术对福利院管理标准化、信息化、规范化管理。为保证福利院良性运转，决定在福利院机构管理上做相应的检查，引入更为客观和公正的第三方评价体系。使福利院的制度不断走向完善。

随着时代科技的迅速发展，世界各地都在尝试通过一些网络技术、智能化来改变原来传统的人工录入信息管理行业的一些流程。在很多国外发达国家采用信息化技术来创新福利院管理的传统方式的同时，越来越多的管理系统被开发出来，功能也逐步得到改善和稳定。

### 1.2.2 国内研究现状和趋势

在现在的阶段看来，现在国内的部分福利院在管理上面仍然存在有不足的地方，有的福利院在信息管理方面，还在是使用过去的人工用纸登记上面。人工填写的信息在许多事情的处理上，存在了一定的误差，用人工登记的信息导致了信息和信息之间没有一个良好的桥梁。在查询调取信息的时候不方便。

在当前社会的福利院中，有大部分福利院是以满足“残疾儿童、孤寡老人、孤儿”的基本生活才建立的，在建立这些福利院的同时是离不开国家在经济方面的投入。也是与人们的观念有一定关系的。在信息的交互关系不是很好的情况下，福利院的工作人员就不能及时的掌握“残疾儿童、孤寡老人、孤儿”的具体状况，根据以上的原因，就会影响到福利院不能给这些人群提供更好的服务。以前的管理方式影响到了信息采集不方便。在信息采集方面的关键问题没有解决的情况下，那对基本的数据信息进行分析的时候就会遇到困难[4]。

现在的所有福利院中，中、高级别的福利院在设施方面都已经得到了改善，在许多设备上，已经取得了相应的升级模式。在福利院中设定了专门的设施以及工作人员，在福利院里面，服务制度时间都是24小时，医疗也是24小时在线，有专业的值班医生，专业的主治医生，全天候的为“残疾儿童、孤寡老人、孤儿”服务，让“残疾儿童、孤寡老人、孤儿”在心理上不会存在想法。在生活上关心“残疾儿童、孤寡老人、孤儿”的生活状态。同时，也是在心理上为大家进行梳理，进入到福利院的人都是无依靠的，在各个方面进行优化完善，也是让更多无依靠的人愿意相信福利院。福利院是社会中无依靠的人群选择最好的地方。在这里，可以使他们无需担忧，生活起居都是有人照顾的，也为成长提供了更好的帮助。

随着科技发展的迅速，福利院管理系统最主要的功能就是让信息流转通畅，福利院信息资源共享，加大福利院人员信息管理的融洽，使信息与信息之间能有效互通。有优质的管理效果，工作人员操作更加便捷，利用互联网节约工作时间成本。利用互联网的优势，加快福利院信息管理的步伐，在现有的国内管理系统的基础上使其更加完善。

# 第2章 基于SSM的六盘水市福利院管理系统的关键技术

## 2.1 系统开发环境

该系统的开发环境主要分为以下几个方面：程序运行时的整体运行环境、开发系统时所采用的操作系统、开发集成工具等。如图2.1所示。



图2.1 系统开发环境分析

根据需求，选择使用Windows-10教育版作为该系统开发时的操作系统。安装Jdk\_1.8、Tomcat\_8.0及以上、MySQL\_5.7.0及以上版本作为该系统开发的编译运行环境，系统前端使用Thymeleaf、JavaScript、HTML、CSS3、Bootstrap技术，后端采用SpringBoot+MyBatisPlus（MyBatis扩展版）+Sa-Toke安全框架技术开发而成[5]。系统利用MySQL存储数据、使用MySQL-Connector-Java包用于JDBC实现数据的访问。使用Tomcat作为Web服务器。

选择使用Maven\_3.6.0对系统开发所需要的第三方包进行管理。使用开源的数据库MySQL\_5.7.0版本。使用IntelliJ IDEA集成工具编写代码。

## 2.2 后台开发框架SSM框架

### 2.2.1 Sping框架

福利院管理系统的后端采用SpringBoot+MyBatisPlus（MyBatis扩展版）+Sa-Toke安全框架技术来进行开发。Spring框架是做系统开发的时候，系统的复杂而所需要的一种框架技术[5]。因此，在系统开发中搭建了Spring框架。Spring是使用最基本的JavaBean来完成EJB的事情，在开发的过程中，Spring不仅仅是用于服务器端的开发。Spring在系统开发中提供了一个高效统一的方式去搭建一个完美的开发过程。在开发人员开发的过程中，Spring将单个的层次架构通过处理后统一在一起。然后通过一定的具体操作，去建立起了一个连贯的体系[5]。从简单性、可测试性和松耦合性角度而言，许多Java应用都可以从Spring中受益。

通俗的说Spring框架给系统开发人员提供了一个完善的开发环境，可以为POJO对象提供优质的服务。在福利院管理系统中有多个子项目称为组件，而每个组件除了依赖部分基础组件外都是相互独立的，既满足开发者按需求所取，也使系统整体得到解耦进一步符合软件工程中的高内聚和低耦合的要求[6]。

### 2.2.2 Spring MVC框架

Spring框架在系统开发中，给Web应用程序提供了MVC功能模块。MVC作为福利院管理系统的分层思想，主要为：模型层（Model）、视图层（View）、控制器（Controller）[6]。

Spring MVC是一个归属于SpringFrameWork的产品，主要是融合在Spring Web Flow里面的一个框架结构。在开发系统时使用Spring可以插入的一个MVC架构，开发人员在使用Spring进行WEB开发的时候，可以选择使用Spring的Spring MVC框架或者其它的MVC开发框架[7]。如图2.2所示。

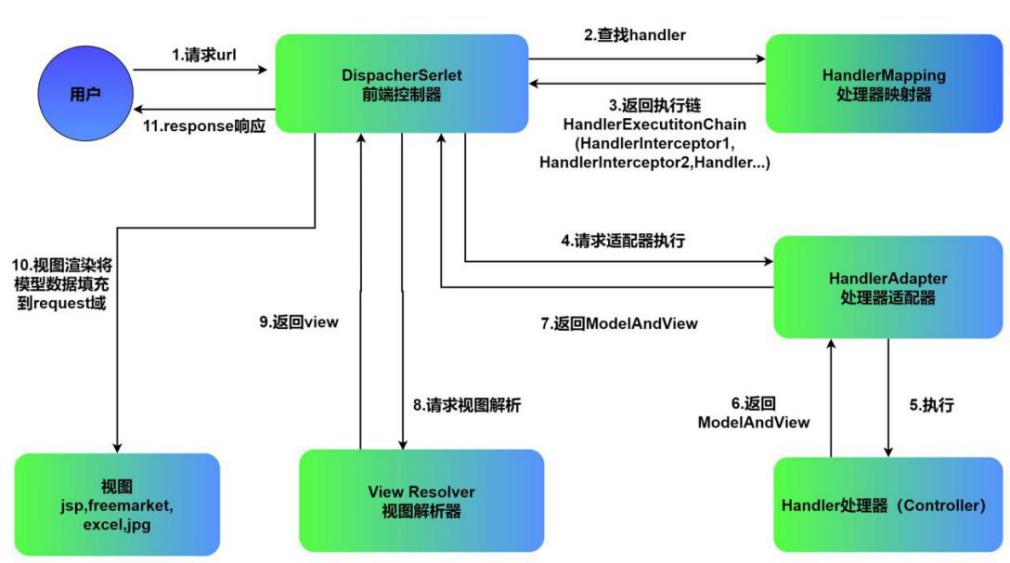


图2.2 Spring MVC执行流程图

### 2.2.3 Spring Boot框架

Spring Boot是当前系统开发的一个全新框架，主要的设计目的就是用来把Spring进行简单化，从而应用到开始系统开发搭建以及整个开发过程。此框架是进行了特殊的一种方式来进行框架配置的。在开发中，开发人员就不需要再对框架配置有重新的定义。通过这种方式，Spring Boot在现在社会中系统开发方向成为了至关重要的领导者[7]。如图2.3所示。

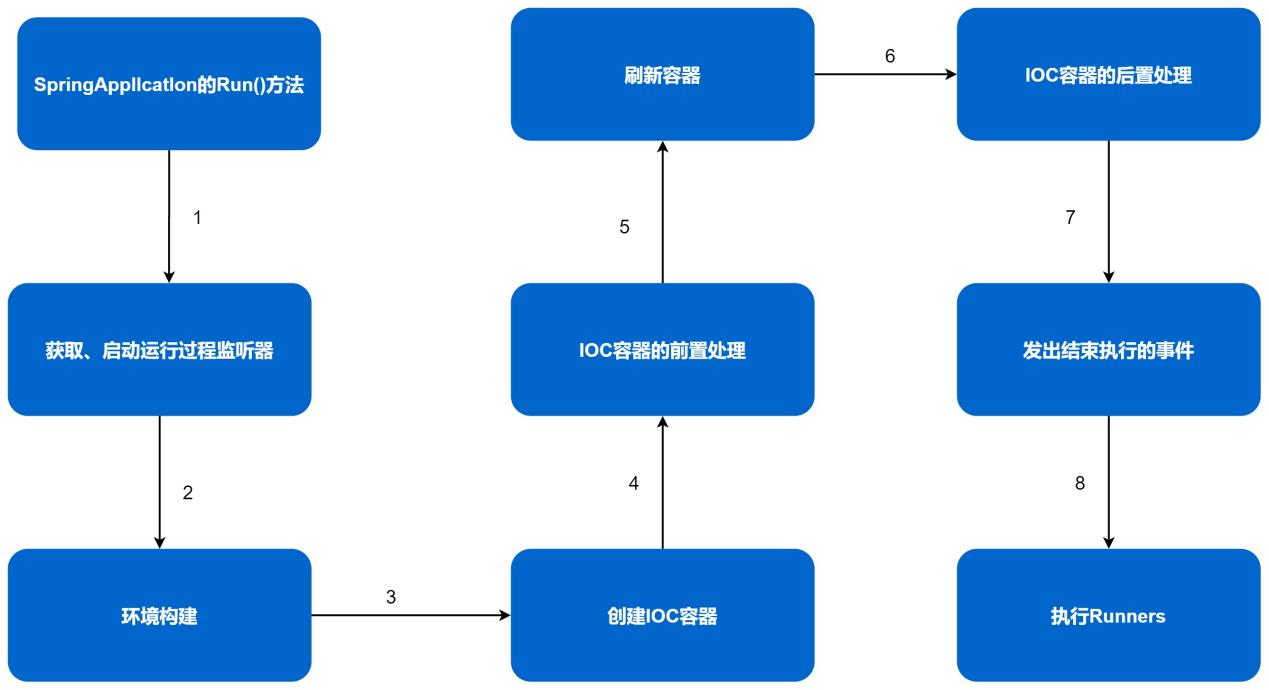


图2.3 Spring Boot启动流程图

### 2.2.4 MyBatis-Plus数据库持久层框架

MyBatis-Plus是MyBatis的升级版工具，在系统开发中需要用到的必要框架[8]。同时，在MyBatis的基础上只是做了增强而不做改变，主要就是为了把系统开发流程简单化，提高开发效率。

### 2.2.5 Thymeleaf模板引擎框架

Thymeleaf是现代的开发服务器端Java的一个模板的引擎，主要适用现在的前端设计Web以及在开发时的一个特有的独立环境[8]。Thymeleaf的主要目标是为开发人员在系统开发的整个工作流程中带来了比较美观的自然模板。HTML前端页面可以在浏览器中正确的显示，在此同时，还可以作为一个静态原型进行工作，在开发人员小组之间起到了团结协作的作用。Thymeleaf具有SpringFrameWork模块的功能，在做系统设计时成为了人们喜爱的工具。

### 2.2.6 Spring Security安全框架

Spring Security是一个现有的功能强大的框架，在除了功能强大之外还是一个可以专门进行身份验证制定的一个框架。是一个主要为了保护基于Spring的应用程序的基本标准。Spring Security也是一个归属于为Java应用程序提供用户的身份验证和用户权限授权的框架[9]。像现在开发中所有的Spring项目是一样的，Spring Security的强大之处就是，在操作时可以很快的实现一定的要求。

主要特征有以下四点：

（1）对身份验证和授权的支持。

（2）防止会话固定、跨站点请求伪造等攻击。

（3）Servlet API集成。

（4）可选与Spring Web MVC集成。

## 2.3 LayUI前端框架

LayUI是现在前端页面开发的一个良好的开源的Web UI解决方案，使用原有的模块化规范，并遵循 HTML、CSS、JS的基本开发方式，最适用于现在的前端网页界面开发[10]。LayUI是主要使用本身的模块间规范编写的前端UI框架。LayUI区别于那些基于MVVM底层的前端框架，更多时候主要是为Java后端开发人员所用的。在开发中只需要面对浏览器，在所有的数据之间起到交互的作用。

## 2.4 系统开发工具

### 2.4.1 IDEA

IDEA全称IntelliJ IDEA，是现在Java编程语言开发的一种集成环境[11]。在目前社会上，深受人们的喜爱。也是一种极为较好的开发工具。许多开发人员都是选用IDEA。尤其是在目前比较火的人工智能方面。可以支持代码重构。在许多工具设计中都是属于超前的。IDEA是JetBrains公司的产品，主要支持HTML，CSS，PHP，MySQL，Python等[12]。在开发过程中，极为省力，是现在系统开发过程中的必需品。

### 2.4.2 Apache Maven项目管理

Apache Maven是一个软件项目管理的工具。基于项目对象模型（POM）的概念，Maven可以从一条中心信息中，对每个管理项目进行标准的构建，以及管理合格标准的报告和文档[13]。Maven除了原有的特定的搭建项目运行能力外，为高级的项目提供了有效的、标准的管理工具[14]。虽然Maven有一定的缺点，主要就是在构建中具有可重用性，因此，在每个开发人员开发的过程中，通常用两三行Maven的主要脚本就可以搭建一个项目。

# 第3章 基于SSM的六盘水市福利院管理系统的分析

## 3.1 可行性分析

### 3.1.1 技术可行性

六盘水市福利院管理系统是采用Java面向对象编程语言进行编写，继承强，高兼容(代码移植性强)，采用主流开源Spring框架，主要秉承了高内聚和低耦合的开发思想，没有明显的入侵性，对其他技术也能够做到合理的相互兼容性，而且还避免了重复的可能性，利用Mybaits-Plus持久层框架提高了开发的效率[15]。而我国相关技术文档的完善程度及大量的社区论坛及项目案例可以借鉴，对应的技术案例讨论探究及示范案例，IDEA开发工具通过合理组合，成功地与MySQL数据库结合成一个完美的团队，在安全性和精密性都成功达到了很好的效果，也是最开始的想法，达到了开发的初衷。在之后的系统开发过程中，也可以达到预期的要求和效果，在具体的技术上是可以行得通的。

### 3.1.2 操作可行性

六盘水市福利院管理系统，主要是为“残疾儿童、孤寡老人、孤儿”等人员入住福利院需要做信息登记、出入登记、平时各项工作登记等。在前端UI设计上采用简单且大方的设计风格，给福利院管理系统的主要功能提供直观的交互操作按钮。用户只需点击按钮即可完成具体的功能，用户在系统操作上能做到简单易懂且不需要额外的培训学习。因此，设计该系统在操作上是可行的。

### 3.1.3 经济可行性

六盘水市福利院管理系统，首先从最初的设计到后期的技术上都是采用开源的相关工具以及技术，其次该系统的数据来源为老人管理、儿童管理及工作人员的信息，最后该系统的设计与实现全过程都是在指导教师的指导下来完成的，从经济层面、法律层面、社会层面没有产生争议，而且在开发的过程中不需要支付任何的费用。因此对于六盘水市福利院管理系统在经济上是可行的。

## 3.2 功能需求分析

六盘水市福利院管理系统从系统功能业务上划分主要有登录模块、管理模块、居住模块、后勤管理模块、医疗工作站模块五个核心业务。主要目的是为了简化系统数据收集的复杂流程、统一管理人员的信息、居住、生活以及医疗等方面。优化人员数据的收集管理，并能够及时对人员信息进行调整。

### 3.2.1 系统用户

系统主要分为三个角色：管理员、工作人员、医护人员。除管理员具备系统一切权限外，其它不同角色登录后显示不同的业务功能和界面，每个角色同时具备的操作权限也会不相同。

### 3.2.2 用户功能描述

管理员用户，作为系统的最高权限使用者，在登录系统后可在系统给其他用户给定一定权限，可以让不同角色得到不同的权限，进入不同的界面进行不一样的操作。可以修改系统里面的一些相关信息。可以修改所有用户的密码。同时也是权限最高者，可以给角色进行分配。可以给其他用户实现一些功能属性。可以增加、删除、修改、查询员工信息、医护人员信息、居住人员信息、后勤信息管理、公告管理、财务管理。可以对系统所有信息进行操作管理。如图3.1管理员用例图所示。

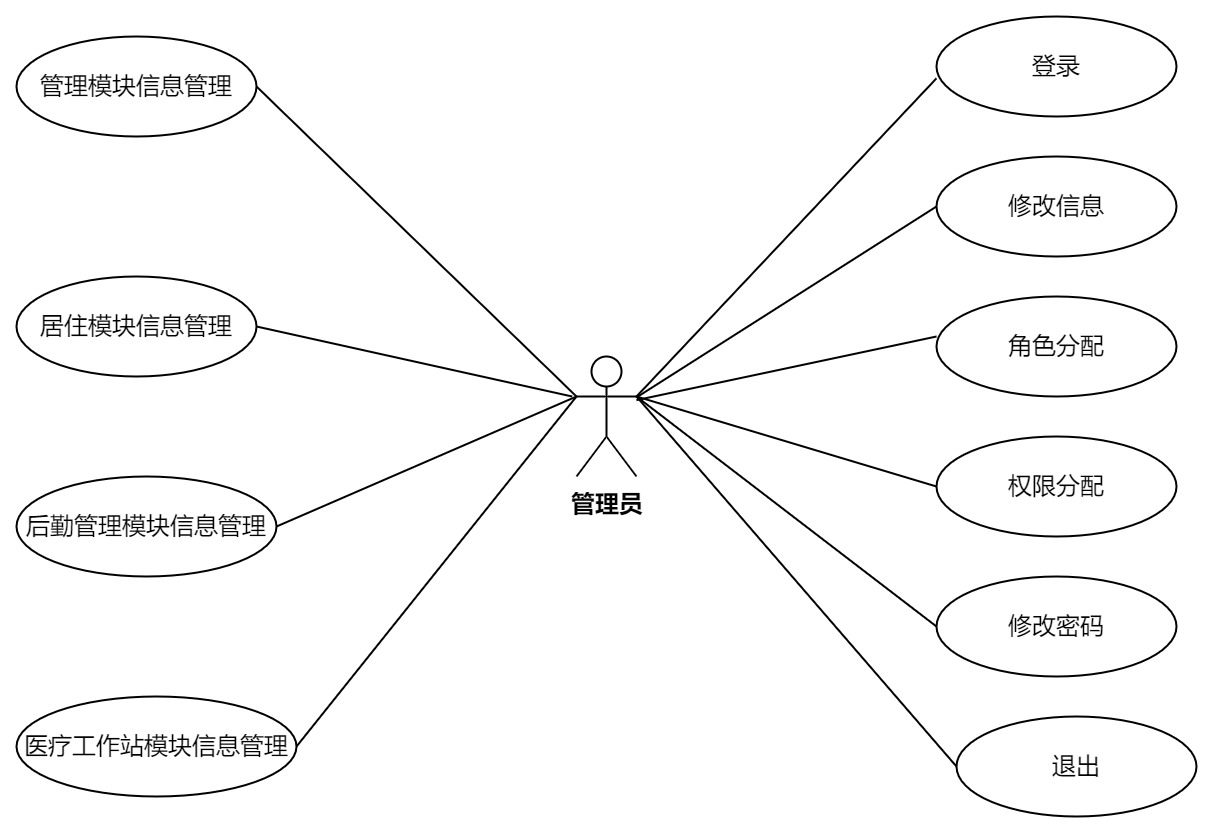


图3.1 管理员用例图

工作人员用户作为系统的使用用户。同时，可以修改个人信息，修改个人密码和实现一些基本操作，可以查看人员信息。可以对以下信息管理进行操作，有办理入住、接待登记、财务管理、公告管理、儿童管理、老人管理、员工信息管理、居住信息管理、后勤信息管理。在每一个管理中，可以实现增加、删除、修改、查询的功能，在特有的管理中，如公告管理，在里面可以发布公告、查询公告，在居住管理中，可以记录人员居住的基本信息等。如图3.2工作人员用例图所示。

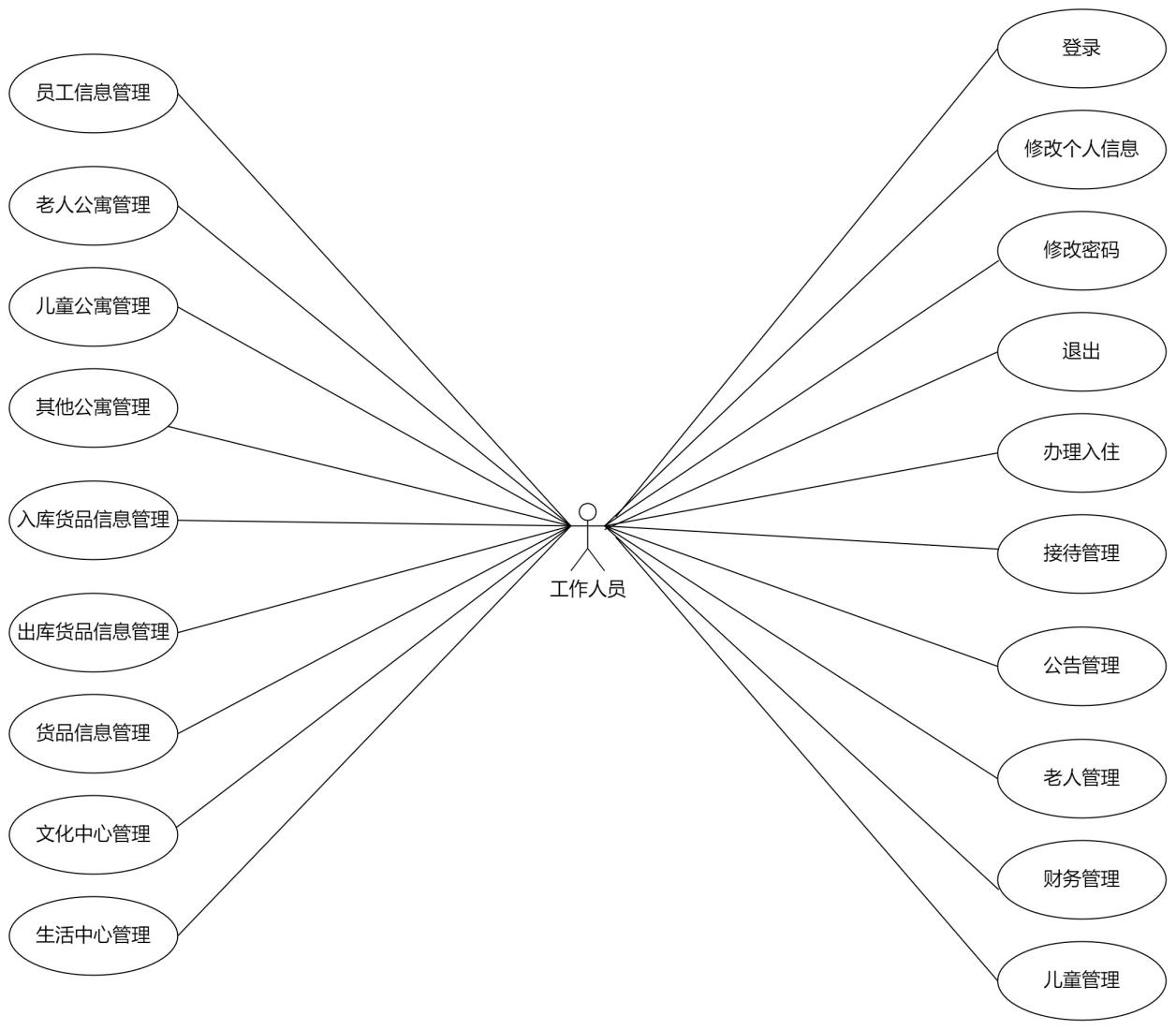


图3.2 工作人员用例图

医护人员用户作为系统的使用用户。同时，可以修改个人信息、修改密码，以及实现一些基本操作，可以查看医护人员的信息，可以对医院的挂号登记、入院人员信息管理、出院人员信息管理、医院药品管理、医院床位信息管理进行管理。在每一个管理中，可以实现增加、删除、修改、查询的功能，在医疗工作站模块中，可以完成一定的操作，可以查询每一个人的基本信息。对各项信息进行详细的记录。如图3.3医护人员用例图所示。

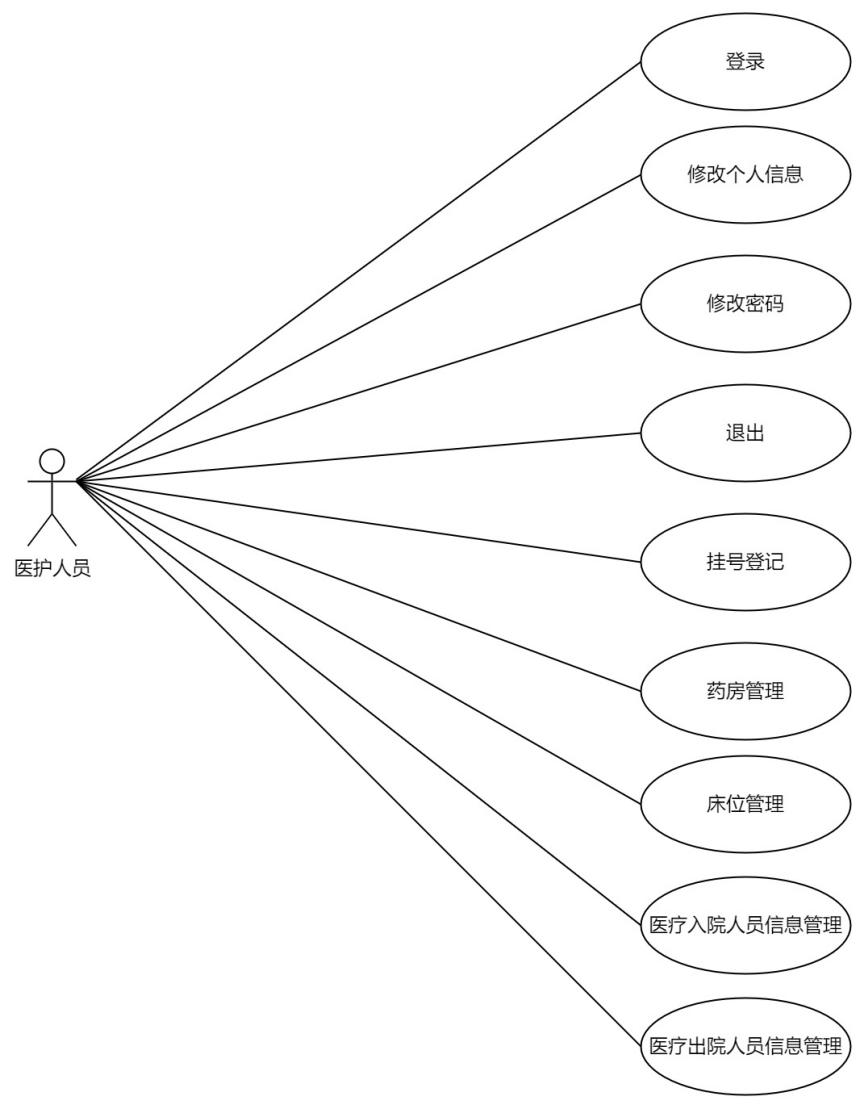


图3.3 医护人员用例图

# 第4章 基于SSM的六盘水市福利院管理系统的设计

## 4.1 系统总体设计原则

该系统在开发过程中应满足系统开发最基本的实用标准。开发规范化根据《阿里巴巴Java开发手册》有助于提高编码规范化水平。对于简单的单体应用，并发量少，只需满足功能需求即可。然而越是大型的应用，它所要考虑的因素就越来越多，通常来讲，架构设计应考虑三个关键点：高可用、高性能、可扩展性。除了上面说的三个关键点之外，架构设计还应考虑成本（可维护性，人力成本，运行成本等），安全性等等。设计中遵循CAP原理的一致性（Consistency），可用性（Availability），保证数据的完全一致性。并且使用单一职责（Simple Responsibility Pinciple，SRP）促使后期需求变更维护互不影响。这样的设计，可以降低类的复杂度，提高类的可读性，提高系统的可维护性，降低变更引起的风险.。

## 4.2 系统功能模块设计

该系统根据用户角色主要分为：管理员、工作人员、医护人员三大角色，功能模块由登录模块、管理员模块和用户模块组成。如图4.1所示。

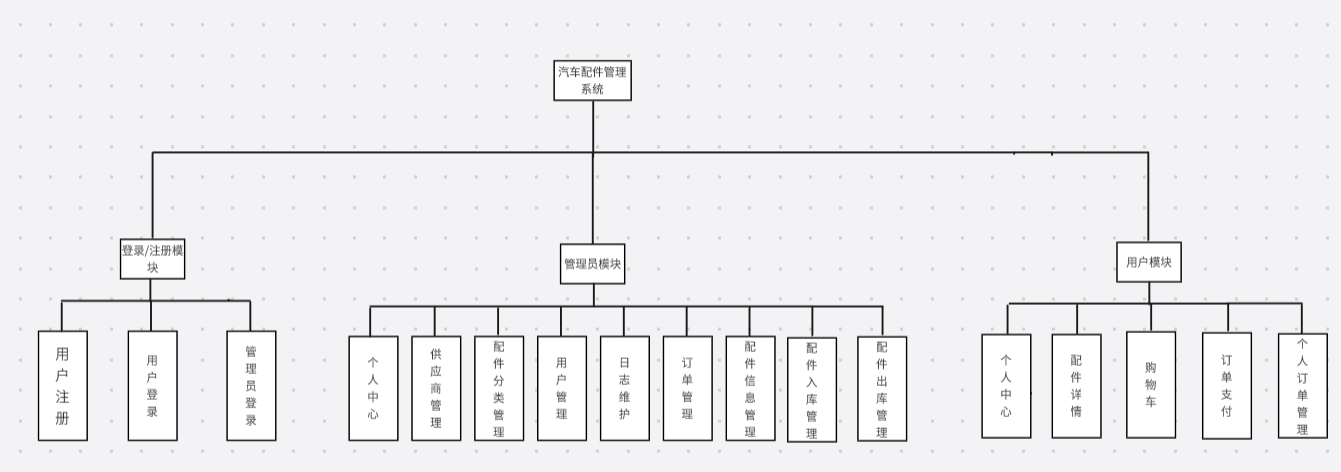
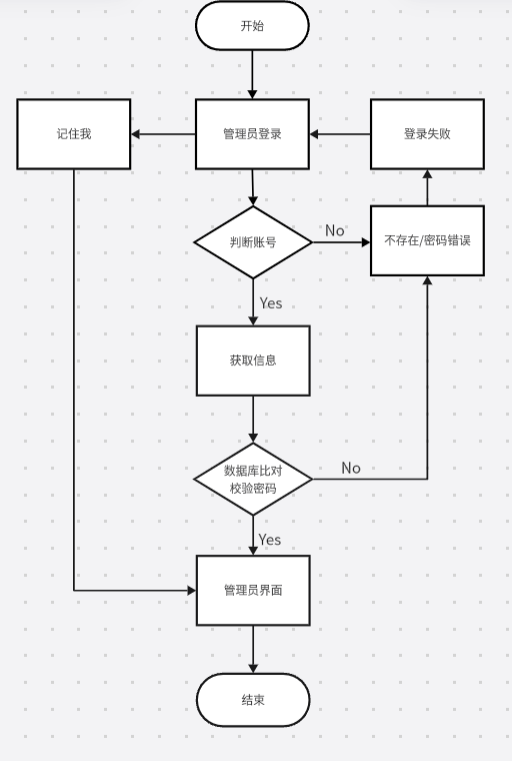


图4.1 系统功能模块图

## 4.3 系统详细设计

### 4.3.1 登录模块设计

该系统针对汽车配件商家的管理员以及购买汽车配件的用户使用，管理员按照系统注册的账号和密码进行登录，用户则可自行注册账号和密码进行登录。登录后可以选择修改原始密码并重新进行登录，以便提高个人的信息安全性。对使用该系统的用户进行必要的身份验证。在身份进行准确的验证之后，便可以进行登录操作。同时当用户或管理员进行数据信息操作时，例如：修改个人密码，修改个人信息等，都会对其数据在数据库中进行更改，同时管理员可以查看用户修改过的信息。如图4.2所示。



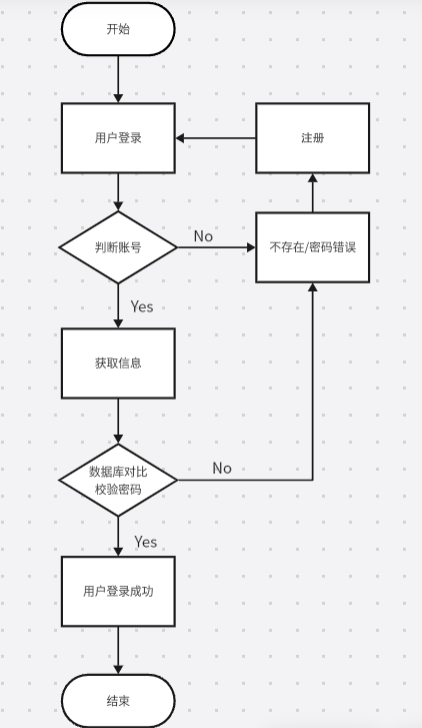


图4.2 登录流程图

### 4.3.2 管理员模块设计

汽车配件管理系统管理员通过登录验证后，进入到相应的管理员界面。管理员可以对所有的管理员进行管理，如对当前系统内的管理员进行增、删、改、查操作；可以对配件进行增、删、改、查以及对需要导出的数据进行导出操作和对配件的库存进行出库入库处理；可以对配件分类进行增、删、改、查操作，例如可以对汽车配件添加发动机配件类，在发动机配件类中还可进行二级分类，如发动机引擎等，也可以对二级分类进行增、删、改、查和数据导出操作；可以对订单进行删、查和确认发货操作，因为订单是由用户发起，故不能增加和修改；可以对注册用户管理进行删、查、改操作以及数据导出操作，因为添加为用户的注册，故不能进行添加操作；可以对轮播图进行增、删、改、查以及数据导出操作；可以对供应商管理进行增、删、改、查操作；可以对日志进行删和查操作。如图4.3所示。

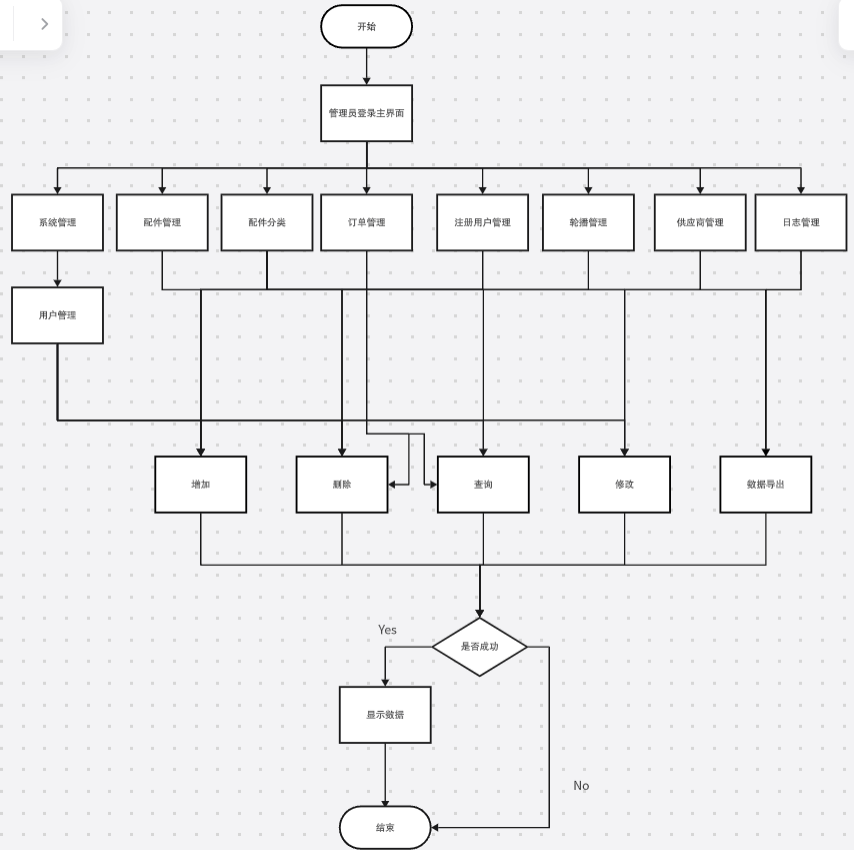


图4.3 管理员模块功能流程图

在以下类图中，主要的关系是，AdminReceiveContrller处理前台发送的请求，调用IAdminReceiveService定义的接口，通过AdminReceiveServiceImpl接口对接口调用实现方法后，由AdminReceiveMapper接口映射处理SQL语句操作数据库。通过调用公共ResultController数据返回类来处理数据的返回结果。如图4.4所示。

在以下类图中，主要的关系是，AdminFinanceContrller处理前台发送的请求，调用IAdminFinanceService定义的接口，通过AdminFinanceServiceImpl接口对接口调用实现方法后，由AdminFinanceMapper接口映射处理SQL语句操作数据库。通过调用公共ResultController数据返回类来处理数据的返回结果。如图4.5所示。

在以下类图中，存在的关系是，AdminChildrenContrller处理前台发送的请求，调用IAdminChildrenService定义的接口，通过AdminChildrenServiceImpl接口对接口调用实现方法后，由AdminChildrenMapper接口映射处理SQL语句操作数据库。通过调用公共ResultController数据返回类来处理数据的返回结果。如图4.6所示。

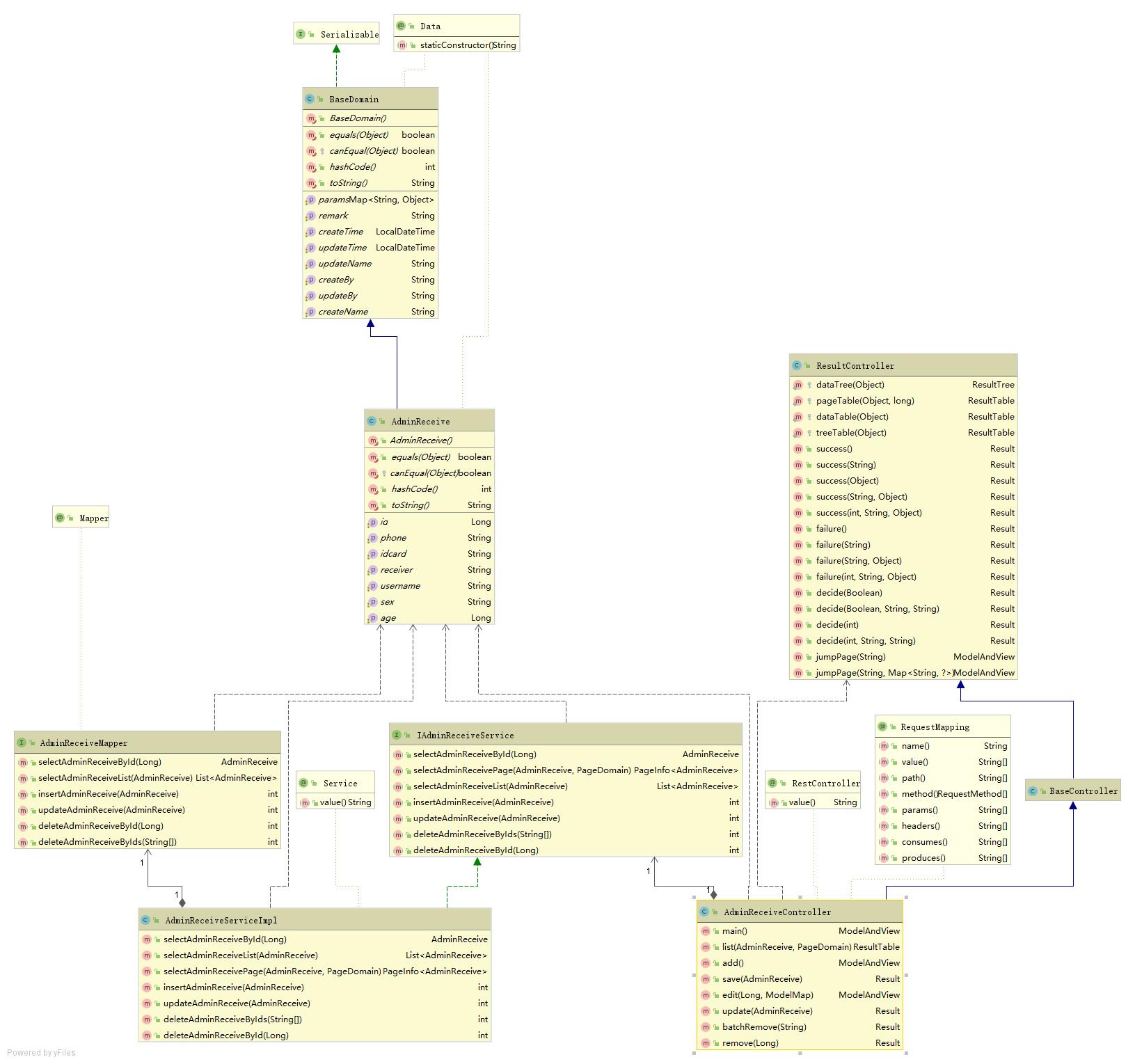


图4.4 接待管理类图

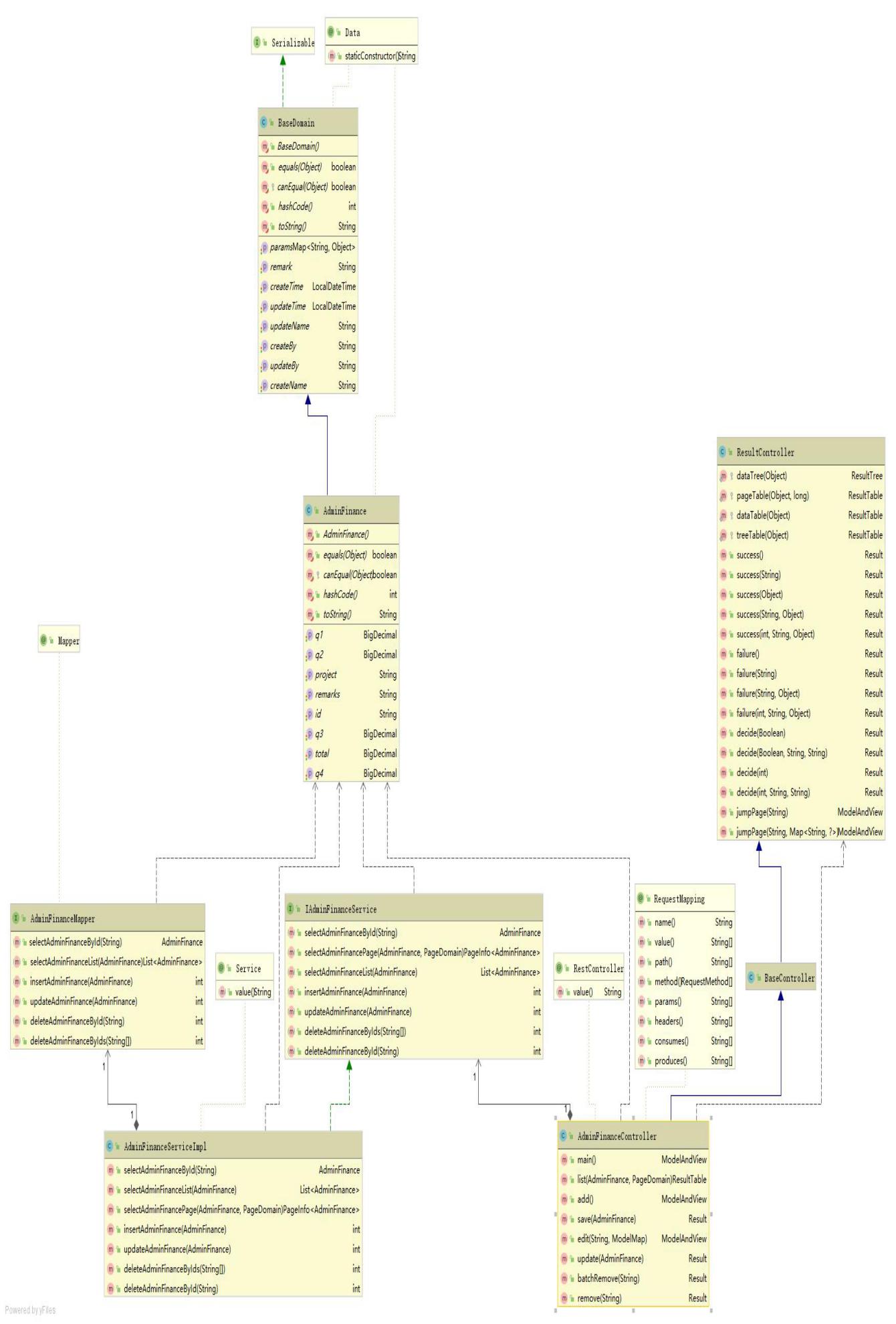


图4.5 财务管理类图

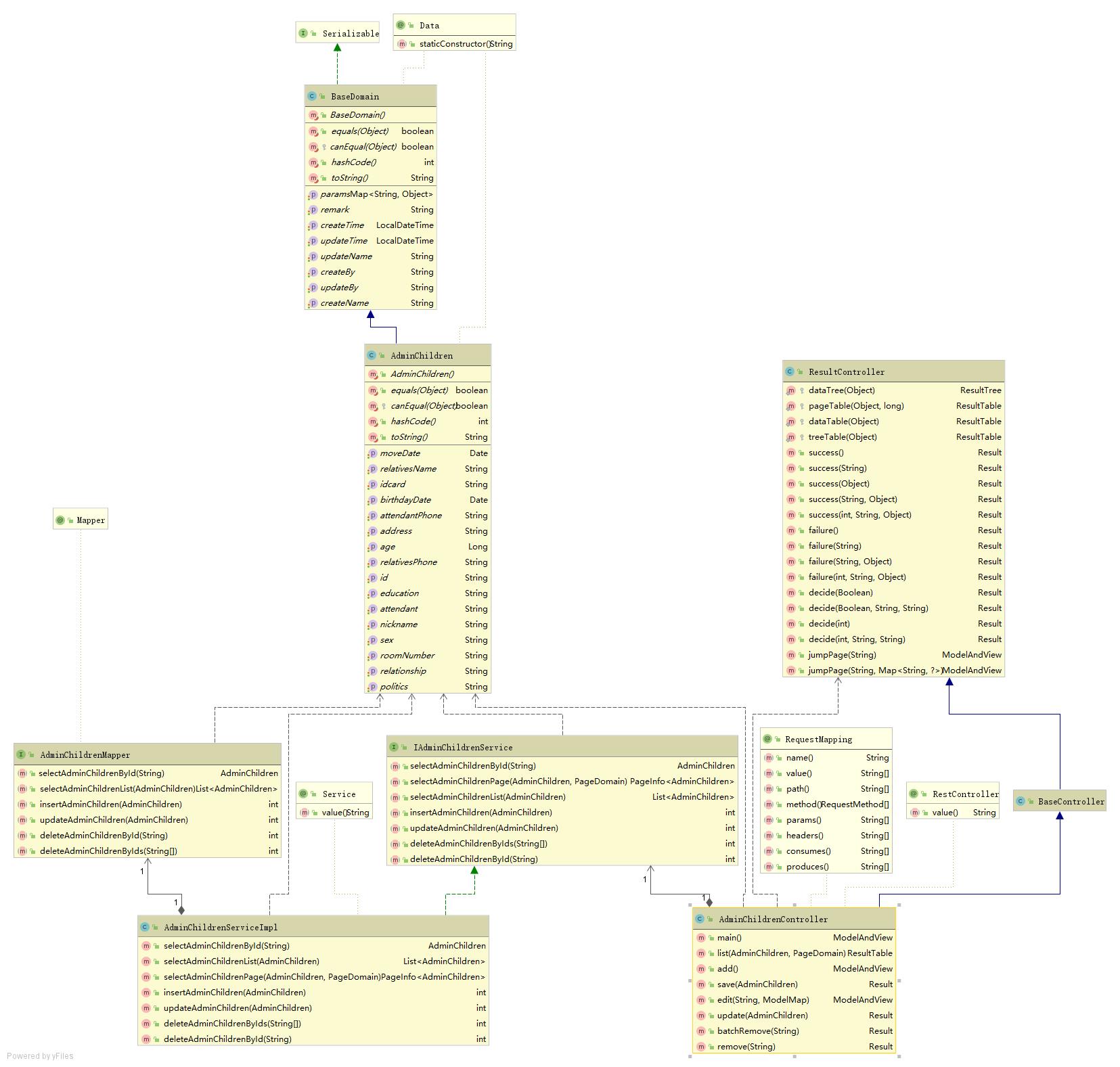


图4.6 儿童管理类图

### 4.3.3 用户模块设计

用户模块的用户可以实现的功能为：用户在登录之后可以查看系统中所有的配件，可以通过分类和关键字查询搜索自己想要的配件；用户还可以将配件加入自己的购物车，在购物车中进行配件的购买和支付功能；用户还可以查看或修改自己的个人信息；用户还可以查看我的订单。如图4.7所示。

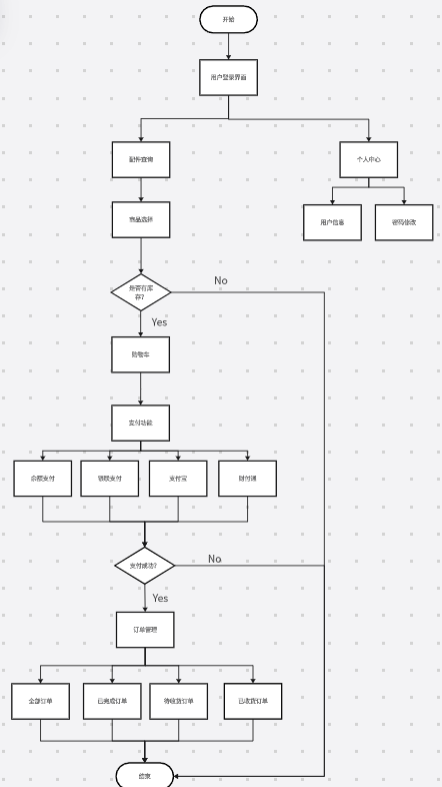
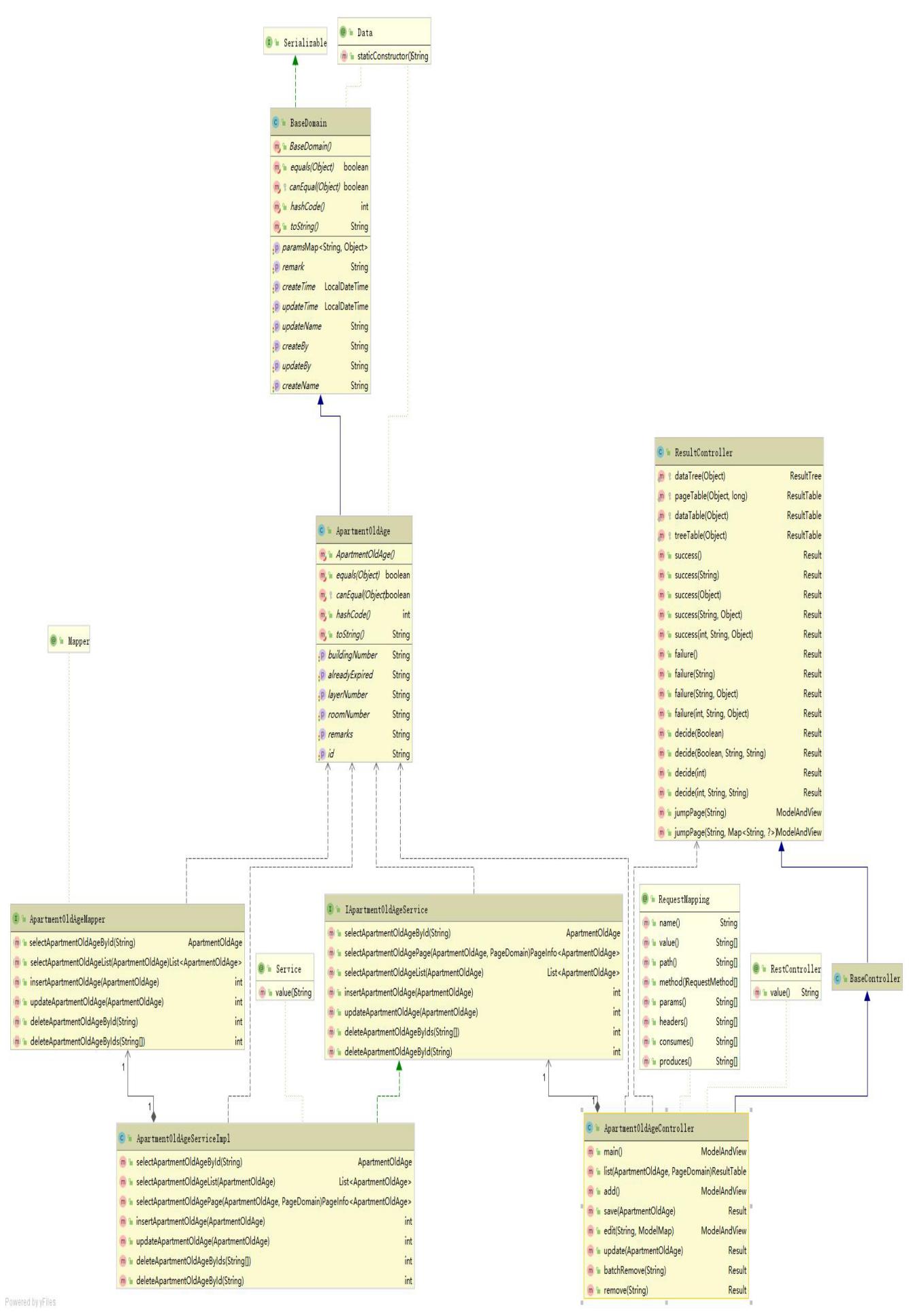


图4.7 用户模块功能流程图

在该类图中，存在的关系是，ApartmentOldAgeContrller处理前台发送的请求，调用IApartmentOldAgeService定义的接口，通过ApartmentOldAgeServiceImpl接口对接口调用实现方法后，由ApartmentOldAgeMapper接口映射处理SQL语句操作数据库。通过调用公共ResultController数据返回类来处理数据的返回结果。如图4.8所示。



## 4.4 数据库设计

### 4.4.1 E-R图设计

E-R图就是实体之间的联系图，主要是用于表述实体与属性之间、实体与实体之间存在的关系，同时，是数据库设计的主要技术支撑。该系统主要包含以下实体。

（1）用户实体（用户id，账户，密码，盐，状态，姓名，邮箱，头像，性别，电话，创建时间，创建人，修改时间，修改人，备注，是否启用，是否登录，部门编号，最后一次登录时间）主要：用于记录账号信息。

（2）角色实体（角色id，角色编号，角色名称，角色标识，是否启用，创建时间，创建人，修改时间，修改人，备注，详情，排序）主要：记录角色信息。

（3）用户与角色实体（用户与角色id，用户编号，角色编号）主要：用于角色与用户之间关系记录。

（4）角色与权限实体（角色与权限id，角色id，权限id，创建人，创建时间，更新人，更新时间，备注）主要：用于角色与权限之间关系记录。

（5）权限实体（权限id，权限名称，权限类型，权限标识，权限路径，打开方式，父类编号，图标，排序，创建人，创建时间，修改人，修改时间，备注，是否开启）主要：用于记录权限信息。

（6）接待管理实体（接待id，姓名，性别，年龄，身份证，电话，接待人，创建时间，修改时间）主要：用于记录接待管理信息。

（7）办理入住实体（办理入住id，姓名，性别，年龄，身份证，出生日期，政治面貌，学历，亲属姓名，与亲属关系，亲属电话，家庭住址，入住时间，房号，责任护理员，护理员联系方式）主要：用于记录存放办理入住信息。

（8）老人管理实体（老人id，姓名，性别，年龄，身份证，出生日期，政治面貌，学历，亲属姓名，与亲属关系，亲属电话，家庭住址，入住时间，房号，责任护理员，护理员联系方式）主要：用于记录存放老人管理的人员基本信息。

（9）儿童管理实体（儿童id，姓名，性别，年龄，身份证，出生日期，政治面貌，学历，亲属姓名，与亲属关系，亲属电话，家庭住址，入住时间，房号，责任护理员，护理员联系方式）主要：用于存留儿童管理的基本信息。

（10）财务管理实体（财务id，开支项目，第一季度，第二季度，第三季度，第四季度，合计，备注）主要：记录财务信息。

（11）公告管理实体（公告id，发布日期，公告名称，类别，点赞数，状态，发布人）主要：记录和发布公告的相关信息。

（12）员工信息管理实体（员工id，姓名，性别，年龄，身份证，政治面貌，家庭地址，毕业学校，学历，学位，职务，联系电话，部门）主要：用于记录员工个人信息的保存。

（13）医护人员管理实体（医护人员id，姓名，性别，年龄，身份证，政治面貌，家庭地址，毕业学校，学历，学位，职务，联系电话，备注）主要：记录医护人员的个人基本信息。

（14）老年公寓实体（老年公寓id，楼号，层号，房号，是否已满，备注）主要：用于记录老年公寓的基本住宿寝室信息。

（15）儿童公寓实体（儿童公寓id，楼号，层号，房号，是否已满，备注）主要：用于记录儿童公寓的基本住宿寝室信息。

（16）其他公寓实体（其他公寓id，楼号，层号，房号，是否已满，备注）主要：用于记录其他公寓的基本住宿寝室信息。

（17）入库货品管理实体（入库id，名称，单价，数量，入库日期，货品编号，经办人，部门，备注）主要：用于记录存入仓库的货品信息记录，登记。

（18）出库货品管理实体（出库id，名称，单价，数量，出库日期，货品编号，经办人，部门，备注）主要：用来记录出库货品信息管理。

（19）货品信息管理实体（货品id，名称，单价，数量，入库日期，出库日期，货品编号，经办人，部门，备注）主要：记录货品所有的基本信息。

（20）生活中心管理实体（生活id，类别名称，参与人数，参与时长，备注）主要：记录参加生活活动中心的信息流动记录。

（21）文化中心管理实体（文化id，类别名称，参与人数，参与时长，备注）主要：记录参加文化中心活动的人员信息，统计。

（22）挂号登记实体（挂号id，姓名，性别，年龄，身份证，联系电话，挂号科室，入院时间，就诊医生，就诊时间，备注）主要：记录医院病人挂号登记的信息。

（23）入院信息管理实体（入院id，姓名，性别，年龄，身份证，联系电话，入院时间，就诊医生，就诊状态，就诊结果，备注）主要：专门用于存留记录入院病人的基本信息。

（24）出院信息管理实体（出院id，姓名，性别，年龄，身份证，联系电话，入院时间，就诊医生，就诊状态，就诊结果，入院时间，出院时间，备注）主要：记录医院出院人信息的登记。

（25）药房管理实体（药房id，药品名称，药品批号，药品规格，库存数量，处方总量，药房单位，零售价，组别，数量单位，录入人，收款人，开方时间，备注）主要：用于存放药品信息以及药品的数量种类。

（26）床位管理实体（床位id，楼层，房号，床位号，入住类别，入住人姓名，入住人身份证，入住时间，出院时间，记录人）主要：记录住院的床位信息。在以下的E-R图中，用户与角色和权限是多对多的关系，用户与其它各个实体类的关系是一对多。一个用户可以对多个实体进行管理。如图4.13所示。

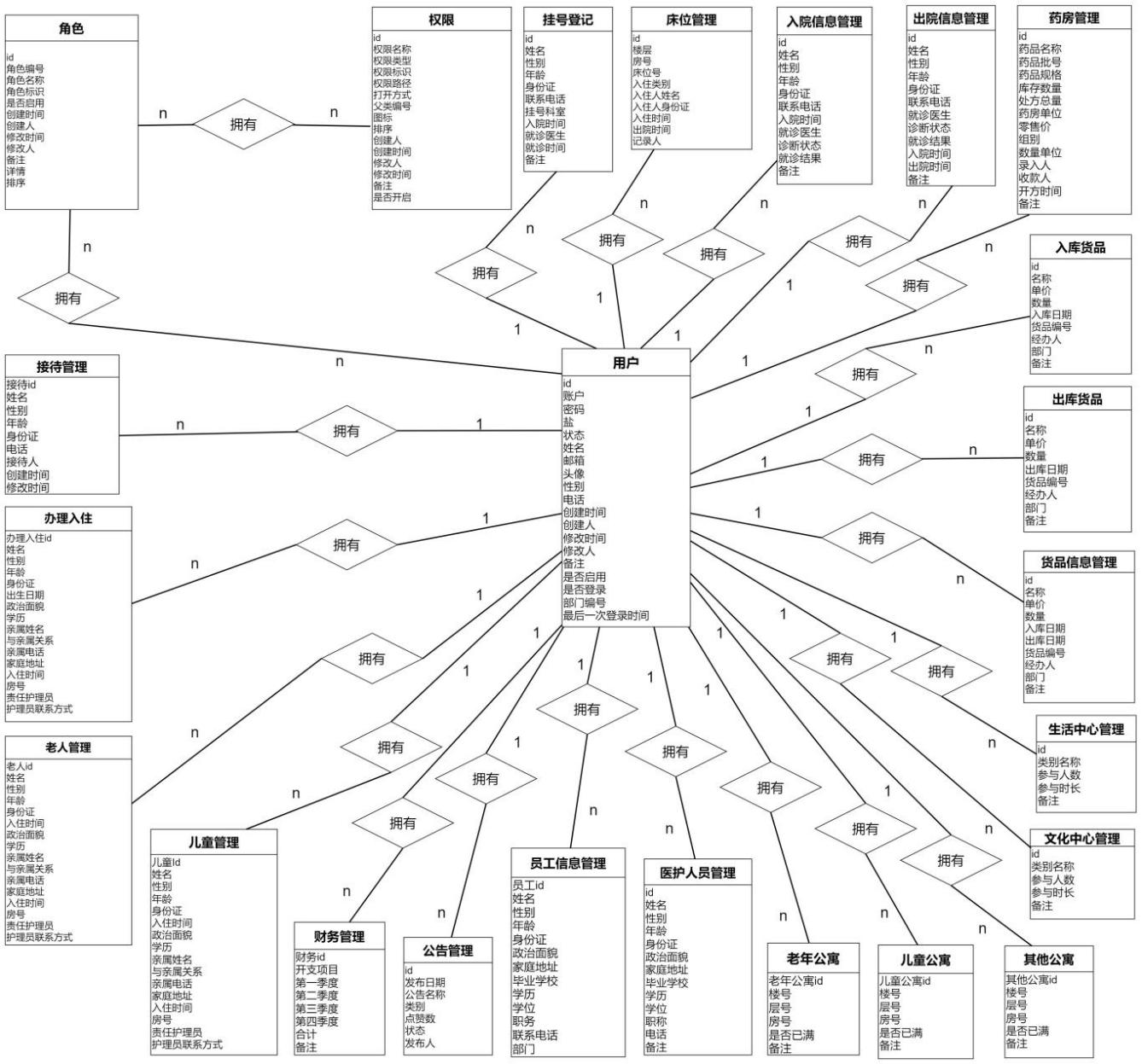


图4.13 系统E-R图

### 4.4.2 数据表设计

（1）菜单表主要有主键id、菜单编号、菜单父编号、当前菜单的所有父菜单编号、菜单名称、菜单图标、url地址、菜单排序号、菜单层级、是否是菜单、备注、菜单状态、是否打开新页面的标识、是否打开、创建时间、修改时间、创建人、修改人。如表4.1所示。

表4.1 菜单表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 数据字段 | 数据类型 | 长度 | 为空 | 主键 | 默认值 | 字段描述 |
| menu\_id | bigint |  | 否 | 是 |  | 主键id |
| code | varchar | 255 | 是 | 否 |  | 菜单编号 |
| pcode | varchar | 255 | 是 | 否 |  | 菜单父编号 |
| pcodes | varchar | 255 | 是 | 否 |  | 当前菜单的所有父菜单编号 |
| name | varchar | 255 | 是 | 否 |  | 菜单名称 |
| icon | varchar | 255 | 是 | 否 |  | 菜单图标 |
| url | varchar | 255 | 是 | 否 |  | url地址 |
| sort | int |  | 是 | 否 |  | 菜单排序号 |
| levels | int |  | 是 | 否 |  | 菜单层级 |
| menu\_flag | varchar | 32 | 是 | 否 |  | 是否是菜单 |
| description | varchar | 255 | 是 | 否 |  | 备注 |
| status | varchar | 32 | 是 | 否 |  | 菜单状态 |
| new\_page\_flag | varchar | 32 | 是 | 否 |  | 是否打开新页面的标识 |
| open\_flag | varchar | 32 | 是 | 否 |  | 是否打开 |
| create\_time | datetime |  | 是 | 否 |  | 创建时间 |
| update\_time | datetime |  | 是 | 否 |  | 修改时间 |
| create\_user | bigint |  | 是 | 否 |  | 创始人 |
| update\_user | bigint |  | 是 | 否 |  | 修改人 |

（2）管理员表主要有主键id、头像、账号、密码、md5密码盐、名字、生日、性别、电子邮件、电话、角色id、部门id、状态、创建时间、创建人、更新时间、更新人、乐观锁。如表4.2所示。

表4.2 管理员表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 数据字段 | 数据类型 | 长度 | 为空 | 主键 | 默认值 | 字段描述 |
| user\_id | bigint |  | 否 | 是 |  | 主键id |
| avatar | varchar | 255 | 是 | 否 |  | 头像 |
| account | varchar | 45 | 是 | 否 |  | 账号 |
| password | varchar | 45 | 是 | 否 |  | 密码 |
| salt | varchar | 45 | 是 | 否 |  | md5密码盐 |
| name | varchar | 45 | 是 | 否 |  | 名字 |
| birthday | datetime |  | 是 | 否 |  | 生日 |
| sex | varchar | 32 | 是 | 否 |  | 性别 |
| email | varchar | 45 | 是 | 否 |  | 电子邮件 |
| phone | varchar | 45 | 是 | 否 |  | 电话 |
| role\_id | varchar | 255 | 是 | 否 |  | 角色id |
| dept\_id | bigint |  | 是 | 否 |  | 部门id |
| status | varchar | 32 | 是 | 否 |  | 状态 |
| create\_time | datetime |  | 是 | 否 |  | 创建时间 |
| create\_user | bigint |  | 是 | 否 |  | 创建人 |
| update\_time | datetime |  | 是 | 否 |  | 更新时间 |
| update\_user | bigint |  | 是 | 否 |  | 更新人 |
| version | int |  | 是 | 否 |  | 乐观锁 |

（3）角色表主要包括主键id、父角色id、角色名称、提示、序号、乐观锁、创建时间、修改时间、创建用户和修改用户。如表4.3所示。

表4.3 角色表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 数据字段 | 数据类型 | 长度 | 为空 | 主键 | 默认值 | 字段描述 |
| role\_id | bigint |  | 否 | 是 |  | 主键id |
| pid | bigint |  | 是 | 否 |  | 父角色ID |
| name | varchar | 255 | 是 | 否 |  | 角色名称 |
| description | varchar | 255 | 是 | 否 |  | 提示 |
| sort | int |  | 是 | 否 |  | 序号 |
| version | int |  | 是 | 否 |  | 乐观锁 |
| create\_time | datetime |  | 是 | 否 |  | 创建时间 |
| update\_time | datetime |  | 是 | 否 |  | 修改时间 |
| create\_user | bigint |  | 是 | 否 |  | 创建用户 |
| update\_user | bigint |  | 是 | 否 |  | 修改用户 |

（4）购物车表主要有主键、用户、商品、价格和数量如表4.4所示。

表4.4 购物车表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 数据字段 | 数据类型 | 长度 | 为空 | 主键 | 默认值 | 字段描述 |
| cart\_id | bigint |  | 否 | 是 |  | 主键 |
| user\_id | bigint |  | 是 | 否 |  | 用户 |
| mall\_id | bigint |  | 是 | 否 |  | 商品 |
| price | decimal | (10,2) | 是 | 否 |  | 价格 |
| num | int |  | 是 | 否 |  | 数量 |

（5）商品管理表主要有主键id、商品图片、名称、价格、商品详情、售卖类型、分类、一级分类、创建时间。如表4.5所示。

表4.5 商品管理表

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 数据字段 | 数据类型 | 长度 | 为空 | 主键 | 默认值 | | 字段描述 |
| mall\_id | bigint |  | 否 | 是 |  | | 主键id |
| goods\_pic | varchar | 5000 | 是 | 否 |  | 商品图片 | |
| title | varchar | 255 | 是 | 否 |  | 名称 | |
| price | decimal | (10,2) | 是 | 否 |  | 价格 | |
| details | text |  | 是 | 否 |  | 商品详情 | |
| type | int |  | 是 | 否 |  | 售卖类型 | |
| type\_id | bigint |  | 是 | 否 |  | 一级分类 | |
| time | datetime |  | 是 | 否 |  | 创建时间 | |

（6）订单管理表主要有主键id、用户id、购买时间、订单号、收货地址、总价、状态、通知状态、退货原因。如表4.6所示。

表4.6 订单管理表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 数据字段 | 数据类型 | 长度 | 为空 | 主键 | 默认值 | 字段描述 |
| order\_id | bigint |  | 否 | 是 |  | 主键id |
| user\_id | bigint |  | 是 | 否 |  | 用户id |
| create\_time | datetime |  | 是 | 否 |  | 购买时间 |
| order\_no | varchar | 255 | 是 | 否 |  | 订单号 |
| address\_id | bigint |  | 是 | 否 |  | 收货地址 |
| price | decimal | (10,1) | 是 | 否 |  | 总价 |
| status | int |  | 是 | 否 |  | 状态 |
| broadcast | int |  | 是 | 否 |  | 通知状态 |
| mark | varchar | 255 | 是 | 否 |  | 退货原因 |

（7）注册用户表中主要有主键id、用户名、登录密码、手机号、姓名、头像、状态、收货地址。如表4.7所示。

表4.7 注册用户表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 数据字段 | 数据类型 | 长度 | 为空 | 主键 | 默认值 | 字段描述 |
| register\_id | bigint |  | 否 | 是 |  | 主键id |
| user\_name | varchar | 255 | 是 | 否 |  | 用户名 |
| pass\_word | varchar | 255 | 是 | 否 |  | 登录密码 |
| tel\_phone | varchar | 255 | 是 | 否 |  | 手机号 |
| nick\_name | varchar | 255 | 是 | 否 |  | 姓名 |
| img | varchar | 255 | 是 | 否 |  | 头像 |
| status | int |  | 是 | 否 |  | 状态 |
| address | varchar | 255 | 是 | 否 |  | 收货地址 |

（8）库存表主要有主键id、商品、数量、类型（1为出库，2为入库）、出入库时间。如表4.8所示。

表4.8 库存表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 数据字段 | 数据类型 | 长度 | 为空 | 主键 | 默认值 | 字段描述 |
| stock\_id | bigint |  | 否 | 是 |  | 主键id |
| goods\_id | bigint |  | 是 | 否 |  | 商品 |
| stock\_num | int |  | 是 | 否 |  | 数量 |
| type | int |  | 是 | 否 |  | 类型（1为出库，2为入库） |
| adds\_time | timestamp |  | 是 | 否 |  | 出入库时间 |

（9）日志表主要有主键id、日志名、用户id、时间。如表4.9所示。

表4.9 日志管理表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 数据字段 | 数据类型 | 长度 | 为空 | 主键 | 默认值 | 字段描述 |
| log\_id | bigint |  | 否 | 是 |  | 主键id |
| name | varchar | 255 | 是 | 否 |  | 日志名 |
| user\_id | bigint |  | 是 | 否 |  | 用户id |
| time | datetime |  | 是 | 否 |  | 时间 |

（11）订单中商品表主要包括主键id、商品id、订单id、数量。如表4.10所示。

表4.10 订单中商品表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 数据字段 | 数据类型 | 长度 | 为空 | 主键 | 默认值 | 字段描述 |
| item\_id | bigint |  | 否 | 是 |  | 主键id |
| mall\_id | bigint |  | 是 | 否 |  | 商品id |
| order\_id | bigint |  | 是 | 否 |  | 订单id |
| num | int |  | 是 | 否 |  | 数量 |

（11）供应商管理表主要包括主键id、名称、联系方式、厂家地址。如表4.11所示。

表4.11 供应商管理表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 数据字段 | 数据类型 | 长度 | 为空 | 主键 | 默认值 | 字段描述 |
| supplier\_id | bigint |  | 否 | 是 |  | 主键id |
| supplier\_name | varchar | 255 | 是 | 否 |  | 名称 |
| supplier\_tel | varchar | 255 | 是 | 否 |  | 联系方式 |
| address | varchar | 255 | 是 | 否 |  | 厂家地址 |

（12）轮播图管理表主要包括主键id、图片地址、状态。如表4.12所示。

表4.12 轮播图管理表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 数据字段 | 数据类型 | 长度 | 为空 | 主键 | 默认值 | 字段描述 |
| pic\_id | bigint |  | 否 | 是 |  | 主键id |
| url | varchar | 255 | 是 | 否 |  | 图片地址 |
| status | int |  | 是 | 否 |  | 状态 |

（13）分类管理表主要包括主键id、分类名称、父id。如表4.13所示。

表4.13 分类管理表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 数据字段 | 数据类型 | 长度 | 为空 | 主键 | 默认值 | 字段描述 |
| types\_id | bigint |  | 否 | 是 |  | 主键id |
| name | varchar | 255 | 是 | 否 |  | 分类名称 |
| pid | bigint |  | 是 | 否 |  | 父id |

（14）收货地址表主要包括主键id、姓名、手机号、详细地址、邮箱、用户。如表4.14所示。

表4.14 收货地址表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 数据字段 | 数据类型 | 长度 | 为空 | 主键 | 默认值 | 字段描述 |
| address\_id | bigint |  | 否 | 是 |  | 主键id |
| name | varchar | 255 | 是 | 否 |  | 姓名 |
| tel | varchar | 255 | 是 | 否 |  | 手机号 |
| address | varchar | 255 | 是 | 否 |  | 详细地址 |
| email | varchar | 255 | 是 | 否 |  | 邮箱 |
| user\_id | bigint |  | 是 | 否 |  | 用户 |

（15）角色和菜单关联表主要包括主键id、菜单id、角色id如表4.15所示。

表4.15 角色和菜单关联表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 数据字段 | 数据类型 | 长度 | 为空 | 主键 | 默认值 | 字段描述 |
| relation\_id | bigint |  | 否 | 是 |  | 主键id |
| menu\_id | bigint |  | 是 | 否 |  | 菜单id |
| role\_id | bigint |  | 是 | 否 |  | 角色id |

# 第5章 基于SSM的六盘水市福利院管理系统的实现

该系统主要由前端和后端组成。后端作为该系统运行时稳定、安全的支撑，也是系统的核心。因此，在进行具体代码实现时需要具体的业务逻辑以及数据类型的选择。前端作为该系统的外观展示页面，于此同时还需深入了解该系统的工作要求，在颜色的搭配上设计的简单、清洁、明了、合理，并且在用户交互界面上做到了清晰明确。

## 5.1 业务功能实现

为了简化开发效率，采用Mybatis Plus持久层框架访问数据库进行操作。前端调用Controller定义的接口时@PreAuthorize注解拦截判断是否具备权限，具备权限访问Service层业务逻辑实现业务功能并配合Mapper层对数据库的CRUD操作。得到返回结果调用ResultController 统一封装数据接口，返回统一格式JSON数据。

## 5.2 登录模块实现

采用RBAC权限与角色设计方式，极大的简化了权限的管理，前端发起登录请求，后端接收到请求后通过Spring Security 拦截处理认证授权，从数据库中查找是否存在此用户，存在后获取用户角色和菜单权限并放行登录返回JSESSIONID到客户端存储。在Redis中存放用户登录信息和实现登录未过期时的用户登录状态存活，同时将登录日志存放数据库。登录界面如图5.1所示。

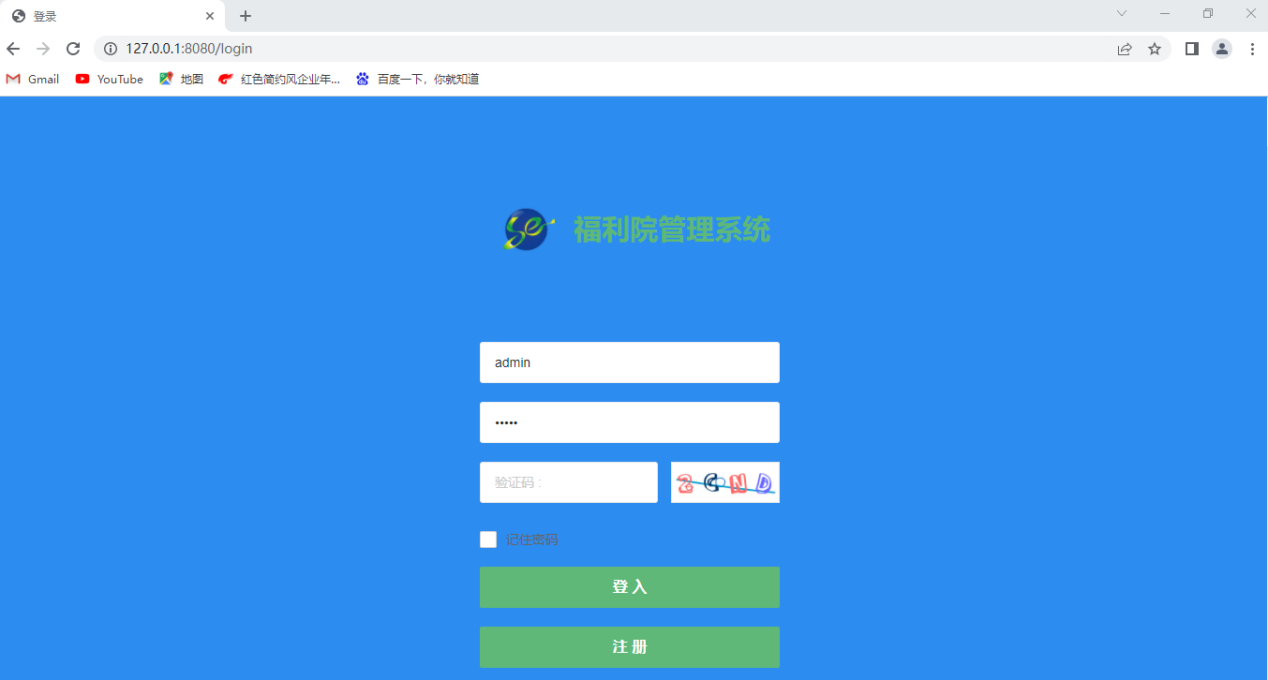


图5.1 登录模块界面图

在该模块登录界面中，主要包含了账号（Account）、密码(Passeord)、验证码(Code)三个输入框。在注册账号后，正确输入账号和密码点击登录则可以进入系统，在登录系统后管理员根据类别指定角色权限，使用Main()方法获取权限。在数据库中进行对比，判断角色是否符合，如果符合后台业务处理返回响应信息，就按照数据库中的定义进行角色权限分配，得到权限的角色方可执行相应的步骤。

## 5.3 管理模块实现

管理模块主要是由接待管理、办理入住、老人管理、儿童管理、财务管理、公告管理、员工信息管理、医护人员信息管理组成。以上各个管理都具有增加、删除、修改、查询的功能，可以对具体单一的信息进行修改处理。只有福利院管理系统的工作人员可以对该功能进行操作。通过该页面，展示系统管理的具体页面样式。管理模块界面如图5.2所示。

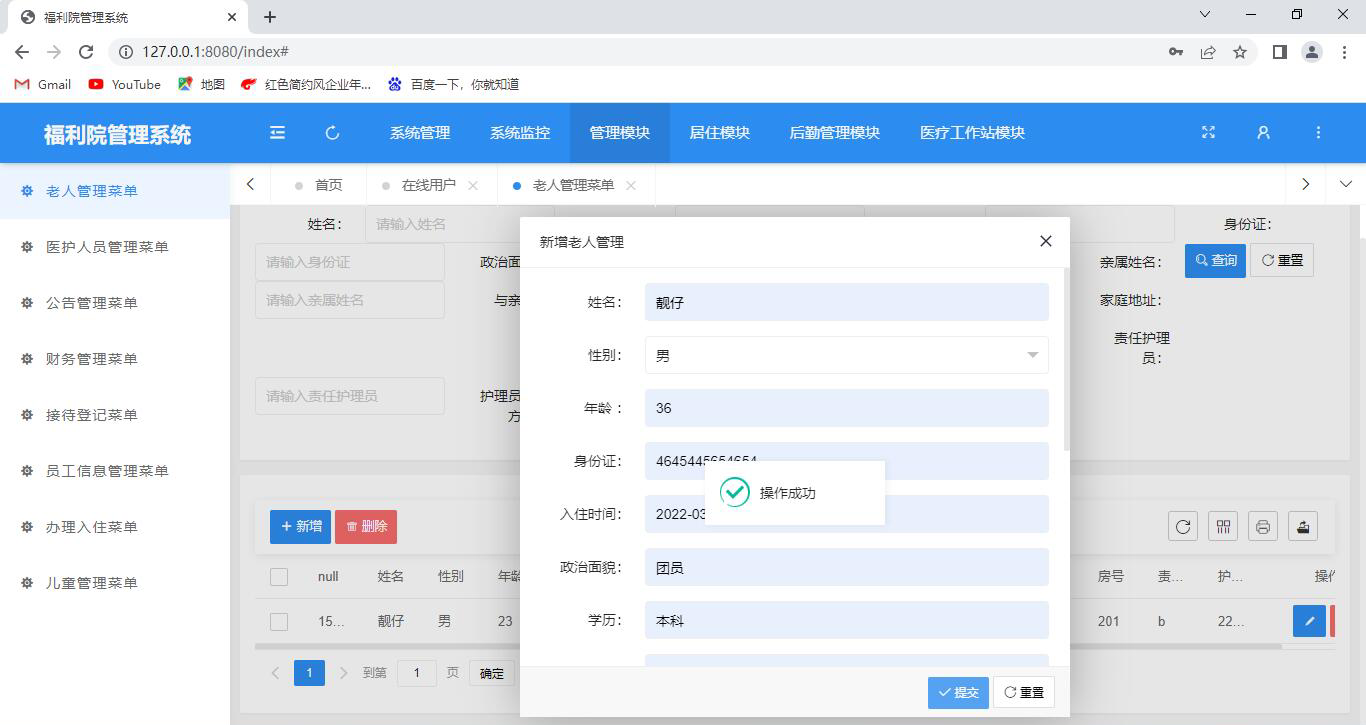


图5.2 管理模块界面图

在该模块中，主要选择说明一个类，相关具体的方法和关键代码，使用到了Domain目录下的Admin\_Elderly（老人管理）对象，以及在Controller目录下使用的Save()、Update()、BatchRemove()、List()方法分别向数据库实现新增、修改、删除、查询数据的操作。在每个方法操作后则通过代码来判断是否成功。具体操作执行后，会返回前端页面，在数据与数据之间交互后得到反馈。具体详细代码见附录1.1。在管理模块下老人管理类中具体功能执行代码如下：

/\*\* \* 新增保存老人管理\*/

@PreAuthorize("hasPermission('/system/elderly/add','system:elderly:add')")

public Result save(@RequestBody AdminElderly adminElderly)

{

SysUser sysUser = (SysUser) SecurityUtil.currentUser();

adminElderly.setId(SequenceUtil.makeStringId());

adminElderly.setCreateTime(LocalDateTime.now());

adminElderly.setCreateBy(sysUser.getUserId());

adminElderly.setCreateName(sysUser.getUsername());

return decide(adminElderlyService.insertAdminElderly(adminElderly));

}

/\*\* \* 删除老人管理 \*/

@PreAuthorize("hasPermission('/system/elderly/remove','system:elderly:remove')")

public Result batchRemove(String ids)

{

return decide(adminElderlyService.deleteAdminElderlyByIds(Convert.toStrArray(ids)));

}

/\*\* \* 修改保存老人管理\*/

@PreAuthorize("hasPermission('/system/elderly/edit','system:elderly:edit')")

public Result update(@RequestBody AdminElderly adminElderly) {

SysUser sysUser = (SysUser) SecurityUtil.currentUser();

adminElderly.setUpdateTime(LocalDateTime.now());

adminElderly.setUpdateBy(sysUser.getUserId());

adminElderly.setUpdateName(sysUser.getUsername());

return decide(adminElderlyService.updateAdminElderly(adminElderly));

}

## 5.4 居住模块实现

该模块主要由老人公寓管理、儿童管理公寓、其他公寓管理三个主要功能组成，同时，都具有增加、删除、修改、查询的功能，可以查询到公寓各个楼层的房间空余的床位查询，通过该页面展示，清楚的看到老人入住信息以及儿童入住信息和其他人员入住信息。只有福利院管理系统的工作人员可以对该功能进行操作。居住模块界面如图5.3所示。

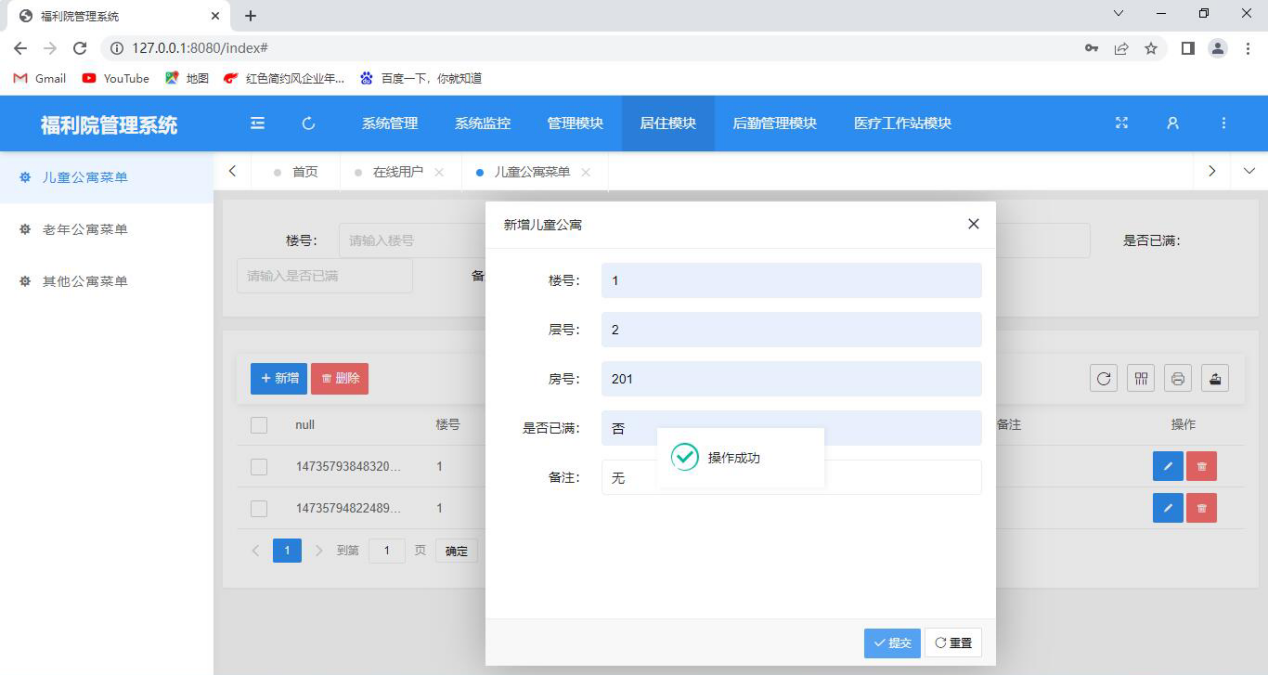


图5.3 居住模块界面图

在该模块中，选择一个类做具体技术说明，在该类中用到的方法和具体的关键代码。主要使用到了Domain目录下的Apartment\_Children（儿童公寓）对象，在Controller目录下使用以下几个方法来向数据库请求实现新增、修改、删除、查询的功能，主要方法如下：Save()、Update()、BatchRemove()、List()，在每个方法执行操作后，通过后台数据库代码对操作进行判断。通过判断后，数据与数据之间进行相应的交互后，返回到前端的页面，则可以看到反馈效果。具体详细代码见附录1.2。在居住模块下儿童公寓管理类中具体实现操作功能代码如下：

/\*\* \* 新增保存儿童公寓\*/

@PreAuthorize("hasPermission('/system/childrens/add','system:childrens:add')")

public Result save(@RequestBody ApartmentChildren apartmentChildren) {

SysUser sysUser = (SysUser) SecurityUtil.currentUser();

apartmentChildren.setId(SequenceUtil.makeStringId());

apartmentChildren.setCreateTime(LocalDateTime.now());

apartmentChildren.setCreateBy(sysUser.getUserId());

apartmentChildren.setCreateName(sysUser.getUsername());

return decide(apartmentChildrenService.insertApartmentChildren(apartmentChildren));

}

/\*\* \* 删除儿童公寓\*/

@PreAuthorize("hasPermission('/system/childrens/remove','system:childrens:remove')")

public Result batchRemove(String ids) {

return decide(apartmentChildrenService.deleteApartmentChildrenByIds(Convert.toStrArray(ids)));}

/\*\* \* 修改保存儿童公寓\*/

@PreAuthorize("hasPermission('/system/childrens/edit','system:childrens:edit')")

public Result update(@RequestBody ApartmentChildren apartmentChildren) {

SysUser sysUser = (SysUser) SecurityUtil.currentUser();

apartmentChildren.setUpdateTime(LocalDateTime.now());

apartmentChildren.setUpdateBy(sysUser.getUserId());

apartmentChildren.setUpdateName(sysUser.getUsername());

return decide(apartmentChildrenService.updateApartmentChildren(apartmentChildren));

}

## 5.5 后勤管理模块实现

后勤管理模块主要由入库货品登记、出库货品登记、货品信息管理、生活中心管理、文化中心管理组成。在每一个管理里面，可以实现增加、删除、修改、查询的功能。可以对信息进行修改，对各项信息可以进行查询。同时，只有福利院管理系统的工作人员可以对该功能进行操作。后勤管理模块界面如图5.4所示。

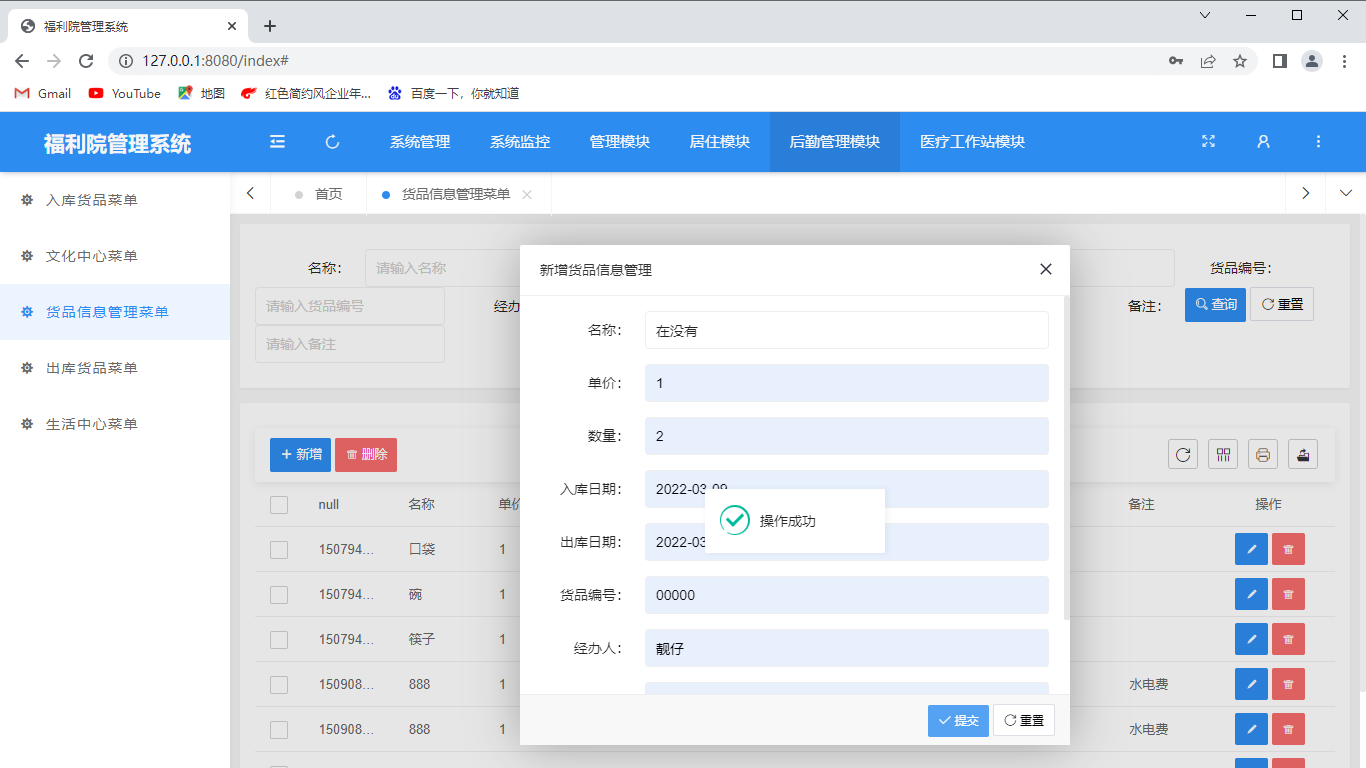


图5.4 后勤管理模块界面图

在该模块中，主要选择一个类做技术说明，包括使用到的方法以及部分关键代码。使用到了Domain目录下的Goods\_Information(货品信息管理)对象，在Controller目录下使用到了以下几个主要方法来向数据库发送请求，得以实现类中主要功能。方法如下：Save()、Update()、Remove()、List()，在进行各个方法的操作后，通过向后台数据库发送的请求，实现新增、修改、删除、查询的功能。在完成各个数据交互请求后返回到前端主界面，此时可以查看界面效果。具体详细代码见附录1.3。在后勤管理模块下货品信息管理类中具体实现功能代码如下：

/\*\* \* 新增保存货品信息管理\*/

@PreAuthorize("hasPermission('/system/information/add','system:information:add')")

public Result save(@RequestBody GoodsInformation goodsInformation) {

SysUser sysUser = (SysUser) SecurityUtil.currentUser();

goodsInformation.setId(SequenceUtil.makeStringId());

goodsInformation.setCreateTime(LocalDateTime.now());

goodsInformation.setCreateBy(sysUser.getUserId());

goodsInformation.setCreateName(sysUser.getUsername());

return decide(goodsInformationService.insertGoodsInformation(goodsInformation));

}

/\*\* \* 删除\*/

@PreAuthorize("hasPermission('/system/information/remove','system:information:remove')")

public Result remove(@PathVariable("id") String id) {

return decide(goodsInformationService.deleteGoodsInformationById(id));

}}

/\*\* \* 修改保存货品信息管理\*/

@PreAuthorize("hasPermission('/system/information/edit','system:information:edit')")

public Result update(@RequestBody GoodsInformation goodsInformation) {

SysUser sysUser = (SysUser) SecurityUtil.currentUser();

goodsInformation.setUpdateTime(LocalDateTime.now());

goodsInformation.setUpdateBy(sysUser.getUserId());

goodsInformation.setUpdateName(sysUser.getUsername());

return decide(goodsInformationService.updateGoodsInformation(goodsInformation));

}

## 5.6 医疗工作站模块实现

医疗工作站模块主要由挂号登记、入院信息管理、出院信息管理、药房管理、床位管理、医护人员管理组成。在该部分，只有医护工作人员可以执行该项工作，在工作中可以实现增加、删除、修改、查询的功能。对具体人员信息进行修改，查询、删除、增加。每一类人员的信息具有独立性，都有各自的存放空间。医疗工作站模块如图5.5所示。

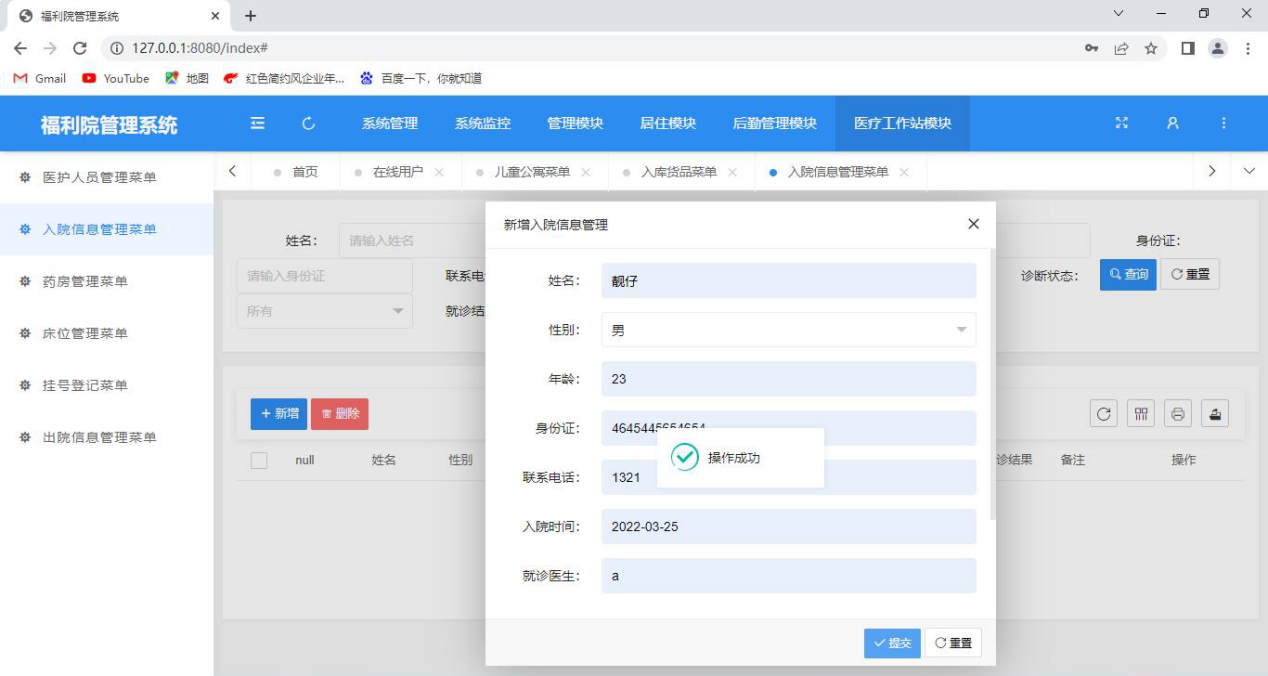


图5.5 医疗工作站模块界面图

在该模块中，主要选择一个类做具体说明，以及所用到的方法，主要使用Domain目录下的Medical\_Admisson\_Information(入院信息管理)对象，同时在目录Controller下使用方法Save()、Edit()、Remove()、List()来对数据进行检测，向数据库发送请求后，通过后台数据库代码识别判断，成功则返回前端页面。在具体操作执行后可以实现功能新增、修改、删除、查询。具体详细代码见附录1.4。在医疗工作站模块下入院信息管理类中具体执行功能代码如下：

/\*\* \* 新增保存入院信息管理\*/

@PreAuthorize("hasPermission('/system/admissionInformation/add','system:admissionInformation:add')")

publicResultsave(@RequestBodyMedicalAdmissionInformationmedicalAdmissionInformation) {

SysUser sysUser = (SysUser) SecurityUtil.currentUser();

medicalAdmissionInformation.setId(SequenceUtil.makeStringId());

medicalAdmissionInformation.setCreateTime(LocalDateTime.now());

medicalAdmissionInformation.setCreateBy(sysUser.getUserId());

medicalAdmissionInformation.setCreateName(sysUser.getUsername());

returndecide(medicalAdmissionInformationService.insertMedicalAdmissionInformation(medicalAdmissionInformation));}

/\*\* \* 删除\*/

@PreAuthorize("hasPermission('/system/admissionInformation/remove','system:admissionInformation:remove')")

public Result remove(@PathVariable("id") String id) {

returndecide(medicalAdmissionInformationService.deleteMedicalAdmissionInformationById(id));}

}

/\*\* \* 修改保存入院信息管理\*/

@PreAuthorize("hasPermission('/system/admissionInformation/edit','system:admissionInformation:edit')")

publicResultupdate(@RequestBodyMedicalAdmissionInformationmedicalAdmissionInformation) {

SysUser sysUser = (SysUser) SecurityUtil.currentUser();

medicalAdmissionInformation.setUpdateTime(LocalDateTime.now());

medicalAdmissionInformation.setUpdateBy(sysUser.getUserId());

medicalAdmissionInformation.setUpdateName(sysUser.getUsername());

returndecide(medicalAdmissionInformationService.updateMedicalAdmissionInformation(medicalAdmissionInformation));

}

# 

# 第6章 基于SSM的六盘水市福利院管理系统的测试

## 6.1 测试目的

系统测试主要就是为了发现系统在运行时出现的错误，在满足系统需求时执行整套程序的运行过程，因此，系统的成功测试就是为了从测试的过程中发现从开始到现在整个过程还未发现的错误。系统测试的目的就是在系统的工作环境下经过运行、检测后，发现系统中的缺陷，同时，就是希望用最小的时间和人工时长，发现系统中潜在的错误。然后，根据开发的需求来运行程序，在开发的过程中，对系统进行模块测试。按照一定的测试原则，发现系统的不足，因此，对系统进行全面的测试。对整体的开发具体细节进行有效的检测。

白盒测试是一种具有结构性的测试也是具有逻辑性的测试。同时也可以称之为是一种基于代码的测试方式。白盒测试就是测试系统的某一个被测试的单元的内部是如何进行工作的一种方法。同时，白盒测试的目的是通过测试过程来检查系统内部的逻辑结构，也是对系统中的逻辑路径进行全面的覆盖测试。在整套程序的不同的地方建设起检查点，检查整套程序的运行状态，以便于可以确定实际的运行状态和预期的状态是否是一致的。在此同时，它也是允许测试的人员在对程序的内部逻辑结构以及其他有关信息来进行设计和选择测试的用例，对整套程序的整体逻辑进行高质量的测试，提高整套程序代码的质量。

黑盒测试是一种具有数据性结构的测试。也是基于需求规格说明的一种测试。黑盒测试直白的说就是把现有的程序当做一个封闭的盒子，在一定的结构特性的情况下来进行测试整套程序的功能，系统测试者要在系统程序的关键的位置进行完整的测试，主要就是为了检查系统程序的功能是不是按照最开始的相关需求规定进行正常的使用，在运行时，是不是能够准确无误的接收输入时的数据信息，因此生成正确的输出的信息以及程序的性能是不是真正的满足用户想要的需求，从而是不是保证数据库或者外部的信息的完整性。在通过黑盒测试的检测后，可以准确的检测到系统的每一个功能是不是都能够正常的进行程序运行。

## 6.2 单元测试

单元测试简单而言就是降低程序开发的成本，提高开发质量的常用方式之一，单元测试就是由系统程序的开发人员或者是后期的测试工作人员来对程序的每个模块的正确性进行检测的一项工作。主要是用来检测程序代码的功能是不是正确的，进行单元测试，不仅可以提高程序代码的质量，还可以提升开发人员的编程技术以及编程的技巧。在进行单元测试时，对每一个模块、每一个类、每一个函数实现的功能进行正确的检测。因此，测试是一项特别重要的工作。

## 6.3 功能模块测试

本系统采用的是SSM框架，主要分为五个模块：分别是登录模块、管理模块、居住模块、后勤管理模块、医疗工作站模块。登录模块：在该模块中，主要实现系统管理员和工作人员登录该系统。由于本系统是针对六盘水市福利院管理系统管理员和工作人员，用户可根据注册的初始账号和密码进行登录，登录后可选择需要修改初始密码重新进行登录，提高个人信息安全性。管理模块：在该模块中，系统管理员对所有用户的权限进行管理。不同角色所拥有不同的权限及其不同权限所显示不同的页面视图。此外，管理员可以对所有个人信息进行管理，包括查看信息、办理登记、查询、添加、删除、修改。详细的分别有接待管理、办理入住、财务管理、公告管理、老人管理、儿童管理、和人员信息档案管理。居住模块：在该模块中，管理员可以查看老年公寓入住情况和儿童公寓入住情况以及其他公寓入住情况，其中包括可以查看已住满寝室和未住满寝室等。修改入住个人信息（如：姓名、年龄、性别等）以及添加入住人员、删除入住人员、查询空房间寝室等等。后勤管理模块：在该模块中，管理员可以查看后勤仓库存放的货品以及货品入库、出库，各类货品信息管理。可以对生活中心以及文化娱乐中心活动信息的管理，查看、修改、添加、删除等。医疗工作站管理模块：在该模块中，医护人员可以对该模块下的各个信息进行管理，有挂号登记管理，入院信息管理，出院信息管理，药房管理，床位管理，医护人员信息管理。为方便对系统进行测试，系统的使用人员分别为管理员、工作人员和医护人员。以上三种角色登录进入系统，可以进行测试。测试结果如以下几个表所示。

（1）多个角色用户登录功能测试表

在该模块的测试中，主要是为了测试每一个用户角色的权限范围是否正常，在不同角色登录系统所展示的界面是否与设计初衷一致。

表6.1 登录模块测试表

|  |  |
| --- | --- |
| 功能描述 | 不同角色的用户登录功能。 |
| 测试目的 | 判断用户的属性与权限，进入不同角色界面，角色分配正确否。 |

表6.1（续） 登录模块测试表

|  |  |  |
| --- | --- | --- |
| 测试步骤 | 1.打开浏览器窗口，然后跳转到登录页面。  2.输入不同的角色用户的账号、密码。  3.点击登录，查看运行显示界面。 | |
| 期待结果 | 是否存在不同角色用户登录成功并显示不同界面菜单，不存在用户登录显示错误的提示信息。 | |
| 实际结果 | 存在不同角色用户登录成功并显示不同界面菜单，不存在用户登录显示“用户账号或密码错误”没进入系统。 | |
| 测试结果 | √ 通过 ×不通过 ×无法测试 | |
| 备注 | 管理员权限指定，通用功能。 | |
| 测试人员 | 陈永量 | 测试时间 2022年03月08 日 |

（2）管理模块测试表

在该模块测试中，主要是为了测试每一个管理块的功能使用是否正常，增加、删除、修改、查询，是否可以正常操作。

表6.2 管理模块测试表

|  |  |
| --- | --- |
| 功能描述 | 接待管理、办理入住、老人管理、儿童管理、财务管理、公告管理、员工信息管理、医护人员管理。 |
| 测试目的 | 判断整个流程的合理性，信息反馈对否。 |
| 测试步骤 | 1.接待管理信息录入显示是否正确，在录入信息之中，功能是不是都是正确的。  2.办理入住界面信息填写录入字段是否正确。  3.老人管理中信息增加、删除、修改、查询，功能实现对否。  4.儿童管理中信息增加、删除、修改、查询，功能实现对否。  5.财务管理中字段格式显示、具体属性实现对否。  6.公告管理中信息发布的显示和预期的效果是否一致。  7.员工信息管理中信息存放功能实现对否，增删改查对否。  8.医护人员信息管理中信息存放功能实现对否，增删改查对否。 |

表6.2（续） 管理模块测试表

|  |  |  |
| --- | --- | --- |
| 期待结果 | 1.接待管理信息录入显示正确，在工作的同时不出现错误。  2.办理入住界面信息填写录入字段正确，工作是系统不出错。  3.老人管理中信息增加、删除、修改、查询，功能实现正确。  4.儿童管理中信息增加、删除、修改、查询，功能实现正确。  5.财务管理中字段格式显示、具体属性实现正确。  6.公告管理中信息发布的显示和预期的效果一致。  7.员工信息管理中信息存放功能实现对否，增删改查正确。  8.医护人员信息管理中信息存放功能实现对否，增删改查正确。 | |
| 实际结果 | 1.接待管理信息录入显示正确，在工作的同时不出现错误。  2.办理入住界面信息填写录入字段正确，工作是系统不出错。  3.老人管理中信息增加、删除、修改、查询，功能实现正确。  4.儿童管理中信息增加、删除、修改、查询，功能实现正确。  5.财务管理、公告管理中字段格式显示、具体属性实现正确。  6.员工信息管理中信息存放功能实现对否，增删改查正确。  7.医护人员信息管理中信息存放功能实现对否，增删改查正确。 | |
| 测试结果 | √ 通过 ×不通过 ×无法测试 | |
| 备注 | 通用功能。 | |
| 测试人员 | 陈永量 | 测试时间 2022年03月08 日 |

（3）居住模块测试表

在该模块的测试中，主要是为了测试管理员和工作人员在对该模块功能进行使用时是否存在异常。在添加、删除、修改、查询信息的时候，功能使用正常与否。

表6.3 居住模块测试表

|  |  |
| --- | --- |
| 功能描述 | 系统管理员或工作人员对该模块对象信息的增、删、改、查。 |
| 测试目的 | 在工作人员对信息进行管理时，是否功能显示正确。 |
| 测试步骤 | 1.测试老年公寓信息管理中，增加、删除、修改、查询。  2.测试儿童公寓中信息的增加、删除、修改、查询，是否正确。  3.测试其他公寓中信息的增加、删除、修改、查询，是否正确。 |

表6.3（续） 居住模块测试表

|  |  |  |
| --- | --- | --- |
| 期待结果 | 1.老年公寓管理中信息的显示的增加、删除、修改、查询，是否正确，对象信息显示界面是否正确。  2.儿童公寓管理中信息的显示的增加、删除、修改、查询，是否正确，对象信息显示界面是否正确。  3.老年公寓管理中信息的显示的增加、删除、修改、查询，是否正确，对象信息显示界面是否正确。 | |
| 实际结果 | 1.老年公寓管理中信息的显示的增加、删除、修改、查询，对象信息显示界面正确。  2.儿童公寓管理中信息的显示的增加、删除、修改、查询，对象信息显示界面正确。  3.老年公寓管理中信息的显示的增加、删除、修改、查询，对象信息显示界面正确。 | |
| 测试结果 | √ 通过 ×不通过 ×无法测试 | |
| 备注 | 通用功能。 | |
| 测试人员 | 陈永量 | 测试时间 2022年03月08 日 |

（4）后勤管理模块测试表

在该模块测试中，是为了测试管理员和工作人员在操作系统时，在该模块的功能中寻找不足的缺陷。查看功能使用是不是正常的，是否存在使用不了的功能。

表6.4 后勤管理模块测试表

|  |  |
| --- | --- |
| 功能描述 | 系统管理员或工作人员对该模块对象信息的增、删、改、查。 |
| 测试目的 | 在工作人员对信息进行管理时，是否功能显示正确。 |
| 测试步骤 | 1.测试入库货品信息登记管理中，增加、删除、修改、查询。  2.测试出库货品信息的增加、删除、修改、查询，是否正确。  3.测试货品信息管理中的增加、删除、修改、查询，是否正确。  4.生活中心管理界面显示，字段布局，功能使用是否合理。  5.文化中心管理界面显示，数据字段的布局，功能使用是否合理。 |

表6.4（续） 后勤管理模块测试表

|  |  |  |
| --- | --- | --- |
| 期待结果 | 1.入库货品管理中信息的显示的增加、删除、修改、查询，是否正确，对象信息显示界面是否正确。  2.出库货品管理中信息的显示的增加、删除、修改、查询，是否正确，对象信息显示界面是否正确。  3.货品信息管理中信息的显示的增加、删除、修改、查询，是否正确，对象信息显示界面是否正确。  4.生活中心管理界面显示，字段布局，功能使用一致。  5.文化中心管理界面显示，格式样式，功能使用一致。 | |
| 实际结果 | 1.入库货品管理中信息的显示的增加、删除、修改、查询，对象信息显示界面正确。  2.出库货品管理中信息的显示的增加、删除、修改、查询，对象信息显示界面正确。  3.货品信息管理中信息的显示的增加、删除、修改、查询，对象信息显示界面正确。  4.生活中心管理界面显示，字段布局，功能使用一致。  5.文化中心管理界面显示，格式样式，功能使用一致。 | |
| 测试结果 | √ 通过 ×不通过 ×无法测试 | |
| 备注 | 通用功能。 | |
| 测试人员 | 陈永量 | 测试时间 2022年03月08 日 |

（5）医疗工作站模块测试表

在该模块测试中，主要测试管理员和医护人员在对该模块界面进行操作时，是否存在界面显示不对或者功能无法实现。在录入信息时，添加、删除、修改、查询是否正常操作。

表6.5 医疗工作站模块测试表

|  |  |
| --- | --- |
| 功能描述 | 系统管理员或医护人员对该模块对象信息的增、删、改、查。 |
| 测试目的 | 在工作人员对信息进行管理时，是否功能显示正确。 |

表6.5（续） 医疗工作站模块测试表

|  |  |  |
| --- | --- | --- |
| 测试步骤 | 1.挂号登记管理中，增加、删除、修改、查询。  2.入院信息管理的增加、删除、修改、查询，是否正确。  3.出院信息管理中信息的增加、删除、修改、查询，是否正确。  4.药房管理界面显示，字段布局，功能使用。  5.床位管理界面显示，格式样式，功能使用。  6.医护人员信息的界面样式，信息查询等。 | |
| 期待结果 | 1.挂号登记管理、入院信息管理、出院信息管理中信息的显示的增加、删除、修改、查询，是否正确，对象信息显示界面是否正确。  2.药房管理界面显示，字段布局，功能使用一致。  3.床位管理界面显示，格式样式，功能使用一致。  4.医护人员信息的界面样式，信息查询、功能是否有效。 | |
| 实际结果 | 1.挂号登记管理中信息的显示的增加、删除、修改、查询，对象信息显示界面正确。  2.入院信息管理中信息的显示的增加、删除、修改、查询，对象信息显示界面正确。  3.出院信息管理中信息的显示的增加、删除、修改、查询，对象信息显示界面正确。  4.药房管理界面显示，字段布局，功能使用一致。  5.床位管理界面显示，格式样式，功能使用一致。  6.医护人员信息的界面样式，信息查询、功能有效。 | |
| 测试结果 | √ 通过 ×不通过 ×无法测试 | |
| 备注 | 通用功能。 | |
| 测试人员 | 陈永量 | 测试时间 2022年03月08 日 |

# 第7章 结论

## 7.1 主要结论

在了解了基于SSM的六盘水市福利院管理系统的设计与实现的相关技术后，根据指导老师的建议，进行资料查询，阅读参考资料，确定了毕业论文的研究目的和意义，并且了解了福利院管理系统目前在国内外发展的现状和趋势，在认真听取指导老师的建议后，根据建议设计系统的功能模块图。功能模块确立后，根据需求阅读相关文献以及学习在设计该系统时可能会遇到的问题的相关解决技术。从文献中认真学习技术和优点，然后灵活运用到该系统中去，在经过多次学习和研究，最后，确立了设计该系统所需要的相关技术，在此同时，还了解了管理系统设计的热门技术。深深的感受到管理系统正在逐步的取代传统的人工记录方式，管理系统正在快速的完善发展。

通过此次的资料查询，参考文献的阅读，首先了解了现在市面上相关管理系统和开发技术的发展现状以及今后的发展趋势，其次，认真学习撰写论文的标准规范，最后，通过借鉴和学习现在已经有的优秀毕业论文设计，能在管理系统开发过程中得到一定的启发，并能顺利完成毕业设计。

## 7.2 研究展望

经过了一系列的反复的测试，发现由于受到开发周期、开发设备、开发资源等的影响，还有极少的地方需要改进，对系统进行优化。但是，该系统也取得了相应的功能成效。最后总结出以下几点：

（1）前端交互设计上虽然功能都满足，但是在美观上仍然存在部分不足。

（2）在后端布局上，存在有少部分的不足。

（3）在系统信息录入时，信息之间互通性有待优化整改，有细节上的不足。

经过仔细的学习以及资料搜索查询之后，相信随着将来的学习和深入的实践经验的积累，在针对该系统现有的不足和以后可能出现的不足的可能性上做出更加优化的方法进行细节上的整合。为设计一个更加美观、使用成熟、全面的福利院管理系统而继续努力。

# 参考文献

1. 姜学勤.以品牌建设引领文化养老新风尚—基于上海市第四社会福利院文化品牌“时节润年华”的打造[J].社会福利,2021(04):56-85.
2. 杨世文,侯超钧.基于SSM框架的学术论坛管理系统设计与实现[J].计算机时代,2021(02):84-96.
3. 何莉.[儿童福利院档案管理工作浅析](https://kns.cnki.net/kcms/detail/detail.aspx?filename=LTLW201936044&dbcode=CJFQ&dbname=CJFD2019&v=F9678hcCqTqwzffbTqwSEGTnpfXA4Ut4rXFYhXrEJwgUVVYrPvpzTYxt3y_gPKfG" \t "https://kns.cnki.net/kcms/detail/frame/kcmstarget)[J].兰台内外,2019(36):305-364.
4. 滕小玢.[福利院孤残儿童照顾现状、问题与对策研究](https://kns.cnki.net/kcms/detail/detail.aspx?filename=1018127997.nh&dbcode=CMFD&dbname=CMFD2018&v=tyQ4cludF17QnD-ISTv-13IhZVYMwjt3ivXHOWFQNcr8DSq1GJ3tpv6dnaFhF42U" \t "https://kns.cnki.net/kcms/detail/frame/kcmstarget)[D].南昌:江西财经大学.2018.
5. 刘继华,路晓梦,张志宏,等.基于SSM框架的毕业设计管理系统设计与实现[J].吕梁学院学报,2020(02):203-226.
6. 龙达鑫.基于SSM框架的员工管理系统设计与实现[J].信息技术与信息化,2020(10):35-98.
7. 柯采.基于J2EE和MVC模式的Web应用研究[J].软件,2020,41(03):165-167.
8. 黑马程序员.Java EE企业级应用开发教程(Spring+ Spring MVC+MyBatis)[M].北京:人民邮电出版社,2018:145-149.
9. 黑马程序员.Java Web程序设计任务教程[M].北京:人民邮电出版社,2017:25-28.
10. 杨开振,周吉文,梁华辉,等.Java EE互联网轻量级框架整合开发:SSM框架(Spring MVC+Spring+MyBatis)和Redis实现[M].北京:电子工业出版社,2017:67-73.
11. 胡成.基于SSM框架的消防员智慧营区管理系统设计与实现[D].合肥:合肥工业大学,2019.
12. 杨帆.基于SSM框架的智慧社区系统设计与实现[D].武汉:武汉邮电科学研究院,2018.
13. 焦鹏珲.基于SpringBoot和Vue框架的电子招投标系统的设计与实现[D].南京:南京大学,2018.
14. Ding Lingling.Design And Implementation Of Student Achievement Information Management System Based On Java[J].International Journal of Computational and Engineering,2021,6(2):7-9.
15. Xiaojie Guo.Tingmei Wang. Design and Implementation of Graduate Information Management System Based on SSM Framework[J].Journal of Simulation,2020:63-69.

# 致 谢

一晃就是四年。还记得当初入学时的样子。是多么的开心、快乐，现如今，已不在有那种感觉，也许是自己长大了。在经过岁月不停的变迁，时光一眨眼就过了，转眼间，已经是即将要离开美好的大学。大学几年的求学生涯已经到了要结束的时候。回顾在学校的点点滴滴，心中已是感慨万千。在学校时不知时间是来也匆匆，去也匆匆。直到即将准备毕业的时候，才发现自己已经快不是学生身份了。

通过本次的毕业设计，发现了身边的老师、同学已经成为了我生命中重要的人，回想在校时，老师、同学们的细心帮助，有太多要感谢的话。在做毕业设计的过程中，在遇到技术性问题的时候，是真的感觉到很吃力，在不久前的一段时间也为此心情沉闷过。但是幸运的是，在那段沉闷枯燥的时间里，身边总是存在着一群好心的人。首先，感谢李惠老师，谢谢您！从最初选定题目的时候，是您一次次地为学生分析想法的不足之处，最终也是在您的推荐下选定了毕业设计的题目。接着在后来毕业论文完成的整个过程中，正是因为您的细心指导，学生在系统的整体设计、系统的实现等重要阶段中，始终坚定初心。最终脚踏实地的完成了本系统的实现和最终确定的论文。在每个人的生命中，都会遇到一些重要的人。而这群人就是在人生道路上的贵客。

在此期间，学校就读的那段时间，一直以学习为主。在遇到困难时，便向老师、同学请教知识难点。懂得了，在遇见问题的时候，要学会分析问题和解决问题。其次，感谢就读大学期间各个科任老师。感谢您们，因为有了各位老师孜孜不倦的教诲，学生得以沉淀知识，在授学期间，老师们的关心，细心的讲解知识难点，对学生不懂和存在质疑的地方得到一一解决，为之后做出的毕业设计系统打下了牢固的基础。感谢您们的关心与教育，让学生无论是在生活上，还是学习上都受益匪浅。

同时，感谢就读大学期间的所有同学们，因为大家的帮助，感受到了世界是美好的。正因为各位乐于助人的精神，在做毕业设计系统遇到困难的时候感受到了光明，为此，十分感动。感谢大家，生活因大家的助力而美丽。愿各位归来时仍是少年。

最后，在以六盘水市福利院管理系统为项目的系统设计中可能仍存在待优化地方，但是实现的整个过程才是最大的收获。

# 附 录

1.1 老人管理类

1.1.1 Domain目录下代码

public class AdminElderly extends BaseDomain{

private static final long serialVersionUID=1L;

/\*\* null \*/

private String id;

/\*\* 姓名 \*/

private String nickname;

/\*\* 性别 \*/

private String sex;

/\*\* 年龄 \*/

private Long age;

/\*\* 身份证 \*/

private String idcard;

/\*\* 入住时间 \*/

private Date birthdayDate;

/\*\* 政治面貌 \*/

private String politics;

/\*\* 学历 \*/

private String education;

/\*\* 亲属姓名 \*/

private String relativesName;

/\*\* 与亲属关系 \*/

private String relationship;

/\*\* 亲属电话 \*/

private String relativesPhone;

/\*\* 家庭地址 \*/

private String address;

/\*\* 入住时间 \*/

private Date moveDate;

/\*\* 房号 \*/

private String roomNumber;

/\*\* 责任护理员 \*/

private String attendant;

/\*\* 护理员联系方式 \*/

private String attendantPhone;

}

1.1.2 Controller目录下代码

public class AdminElderlyController extends BaseController {

private String prefix = "system/elderly";

@Autowired

private IAdminElderlyService adminElderlyService;

@GetMapping("/main")

@PreAuthorize("hasPermission('/system/elderly/main','system:elderly:main')")

public ModelAndView main() {

return jumpPage(prefix + "/main");

}

/\*\*

\* 查询老人管理列表

\*/

@ResponseBody

@GetMapping("/data")

@PreAuthorize("hasPermission('/system/elderly/data','system:elderly:data')")

public ResultTable list(@ModelAttribute AdminElderly adminElderly, PageDomain pageDomain) {

PageInfo<AdminElderly> pageInfo = adminElderlyService.selectAdminElderlyPage(adminElderly, pageDomain);

return pageTable(pageInfo.getList(), pageInfo.getTotal());

}

/\*\*

\* 新增老人管理

\*/

@GetMapping("/add")

@PreAuthorize("hasPermission('/system/elderly/add','system:elderly:add')")

public ModelAndView add() {

return jumpPage(prefix + "/add");

}

/\*\*

\* 新增保存老人管理

\*/

@ResponseBody

@PostMapping("/save")

@PreAuthorize("hasPermission('/system/elderly/add','system:elderly:add')")

public Result save(@RequestBody AdminElderly adminElderly) {

SysUser sysUser = (SysUser) SecurityUtil.currentUser();

adminElderly.setId(SequenceUtil.makeStringId());

adminElderly.setCreateTime(LocalDateTime.now());

adminElderly.setCreateBy(sysUser.getUserId());

adminElderly.setCreateName(sysUser.getUsername());

return decide(adminElderlyService.insertAdminElderly(adminElderly));

}

/\*\*

\* 修改老人管理

\*/

@GetMapping("/edit")

@PreAuthorize("hasPermission('/system/elderly/edit','system:elderly:edit')")

public ModelAndView edit(String id, ModelMap mmap) {

AdminElderly adminElderly =adminElderlyService.selectAdminElderlyById(id);

mmap.put("adminElderly", adminElderly);

return jumpPage(prefix + "/edit");

}

/\*\*

\* 修改保存老人管理

\*/

@ResponseBody

@PutMapping("/update")

@PreAuthorize("hasPermission('/system/elderly/edit','system:elderly:edit')")

public Result update(@RequestBody AdminElderly adminElderly) {

SysUser sysUser = (SysUser) SecurityUtil.currentUser();

adminElderly.setUpdateTime(LocalDateTime.now());

adminElderly.setUpdateBy(sysUser.getUserId());

adminElderly.setUpdateName(sysUser.getUsername());

return decide(adminElderlyService.updateAdminElderly(adminElderly));

}

/\*\*

\* 删除老人管理

\*/

@ResponseBody

@DeleteMapping("/batchRemove")

@PreAuthorize("hasPermission('/system/elderly/remove','system:elderly:remove')")

public Result batchRemove(String ids) {

return decide(adminElderlyService.deleteAdminElderlyByIds(Convert.toStrArray(ids)));

}

/\*\*

\* 删除

\*/

@ResponseBody

@DeleteMapping("/remove/{id}")

@PreAuthorize("hasPermission('/system/elderly/remove','system:elderly:remove')")

public Result remove(@PathVariable("id") String id) {

return decide(adminElderlyService.deleteAdminElderlyById(id));

}}

1.2 儿童公寓管理类

1.2.1 Domain目录下代码

public class ApartmentChildren extends BaseDomain{

private static final long serialVersionUID=1L;

/\*\* null \*/

private String id;

/\*\* 楼号 \*/

private String buildingNumber;

/\*\* 层号 \*/

private String layerNumber;

/\*\* 房号 \*/

private String roomNumber;

/\*\* 是否已满 \*/

private String alreadyExpired;

/\*\* 备注 \*/

private String remarks;

}

1.2.2 Controller目录下代码

public class ApartmentChildrenController extends BaseController {

private String prefix = "system/childrens";

@Autowired

private IApartmentChildrenService apartmentChildrenService;

@GetMapping("/main")

@PreAuthorize("hasPermission('/system/childrens/main','system:childrens:main')")

public ModelAndView main() {

return jumpPage(prefix + "/main");

}

/\*\*

\* 查询儿童公寓列表

\*/

@ResponseBody

@GetMapping("/data")

@PreAuthorize("hasPermission('/system/childrens/data','system:childrens:data')")

public ResultTable list(@ModelAttribute ApartmentChildren apartmentChildren, PageDomain pageDomain) {

PageInfo<ApartmentChildren> pageInfo = apartmentChildrenService.selectApartmentChildrenPage(apartmentChildren, pageDomain);

return pageTable(pageInfo.getList(), pageInfo.getTotal());

}

/\*\*

\* 新增儿童公寓

\*/

@GetMapping("/add")

@PreAuthorize("hasPermission('/system/childrens/add','system:childrens:add')")

public ModelAndView add() {

return jumpPage(prefix + "/add");

}

/\*\*

\* 新增保存儿童公寓

\*/

@ResponseBody

@PostMapping("/save")

@PreAuthorize("hasPermission('/system/childrens/add','system:childrens:add')")

public Result save(@RequestBody ApartmentChildren apartmentChildren) {

SysUser sysUser = (SysUser) SecurityUtil.currentUser();

apartmentChildren.setId(SequenceUtil.makeStringId());

apartmentChildren.setCreateTime(LocalDateTime.now());

apartmentChildren.setCreateBy(sysUser.getUserId());

apartmentChildren.setCreateName(sysUser.getUsername());

return decide(apartmentChildrenService.insertApartmentChildren(apartmentChildren));

}

/\*\*

\* 修改儿童公寓

\*/

@GetMapping("/edit")

@PreAuthorize("hasPermission('/system/childrens/edit','system:childrens:edit')")

public ModelAndView edit(String id, ModelMap mmap) {

ApartmentChildren apartmentChildren =apartmentChildrenService.selectApartmentChildrenById(id);

mmap.put("apartmentChildren", apartmentChildren);

return jumpPage(prefix + "/edit");

}

/\*\*

\* 修改保存儿童公寓

\*/

@ResponseBody

@PutMapping("/update")

@PreAuthorize("hasPermission('/system/childrens/edit','system:childrens:edit')")

public Result update(@RequestBody ApartmentChildren apartmentChildren) {

SysUser sysUser = (SysUser) SecurityUtil.currentUser();

apartmentChildren.setUpdateTime(LocalDateTime.now());

apartmentChildren.setUpdateBy(sysUser.getUserId());

apartmentChildren.setUpdateName(sysUser.getUsername());

return decide(apartmentChildrenService.updateApartmentChildren(apartmentChildren));

}

/\*\*

\* 删除儿童公寓

\*/

@ResponseBody

@DeleteMapping("/batchRemove")

@PreAuthorize("hasPermission('/system/childrens/remove','system:childrens:remove')")

public Result batchRemove(String ids) {

return decide(apartmentChildrenService.deleteApartmentChildrenByIds(Convert.toStrArray(ids)));

}

/\*\*

\* 删除

\*/

@ResponseBody

@DeleteMapping("/remove/{id}")

@PreAuthorize("hasPermission('/system/childrens/remove','system:childrens:remove')")

public Result remove(@PathVariable("id") String id) {

return decide(apartmentChildrenService.deleteApartmentChildrenById(id));

}}

1.3 货品信息管理类

1.3.1 Domain目录下代码

public class GoodsInformation extends BaseDomain{

private static final long serialVersionUID=1L;

/\*\* null \*/

private String id;

/\*\* 名称 \*/

private String name;

/\*\* 单价 \*/

private BigDecimal price;

/\*\* 数量 \*/

private Long number;

/\*\* 入库日期 \*/

private Date warehousingDate;

/\*\* 出库日期 \*/

private Date warehouseDate;

/\*\* 货品编号 \*/

private String goodsNumber;

/\*\* 经办人 \*/

private String handler;

/\*\* 部门 \*/

private String department;

/\*\* 备注 \*/

private String remarks;

}

1.3.2 Controller目录下代码

public class GoodsInformationController extends BaseController {

private String prefix = "system/information";

@Autowired

private IGoodsInformationService goodsInformationService;

@GetMapping("/main")

@PreAuthorize("hasPermission('/system/information/main','system:information:main')")

public ModelAndView main() {

return jumpPage(prefix + "/main");

}

/\*\*

\* 查询货品信息管理列表

\*/

@ResponseBody

@GetMapping("/data")

@PreAuthorize("hasPermission('/system/information/data','system:information:data')")

public ResultTable list(@ModelAttribute GoodsInformation goodsInformation, PageDomain pageDomain) {

PageInfo<GoodsInformation> pageInfo = goodsInformationService.selectGoodsInformationPage(goodsInformation, pageDomain);

return pageTable(pageInfo.getList(), pageInfo.getTotal());

}

/\*\*

\* 新增货品信息管理

\*/

@GetMapping("/add")

@PreAuthorize("hasPermission('/system/information/add','system:information:add')")

public ModelAndView add() {

return jumpPage(prefix + "/add");

}

/\*\*

\* 新增保存货品信息管理

\*/

@ResponseBody

@PostMapping("/save")

@PreAuthorize("hasPermission('/system/information/add','system:information:add')")

public Result save(@RequestBody GoodsInformation goodsInformation) {

SysUser sysUser = (SysUser) SecurityUtil.currentUser();

goodsInformation.setId(SequenceUtil.makeStringId());

goodsInformation.setCreateTime(LocalDateTime.now());

goodsInformation.setCreateBy(sysUser.getUserId());

goodsInformation.setCreateName(sysUser.getUsername());

return decide(goodsInformationService.insertGoodsInformation(goodsInformation));

}

/\*\*

\* 修改货品信息管理

\*/

@GetMapping("/edit")

@PreAuthorize("hasPermission('/system/information/edit','system:information:edit')")

public ModelAndView edit(String id, ModelMap mmap) {

GoodsInformation goodsInformation =goodsInformationService.selectGoodsInformationById(id);

mmap.put("goodsInformation", goodsInformation);

return jumpPage(prefix + "/edit");

}

/\*\*

\* 修改保存货品信息管理

\*/

@ResponseBody

@PutMapping("/update")

@PreAuthorize("hasPermission('/system/information/edit','system:information:edit')")

public Result update(@RequestBody GoodsInformation goodsInformation) {

SysUser sysUser = (SysUser) SecurityUtil.currentUser();

goodsInformation.setUpdateTime(LocalDateTime.now());

goodsInformation.setUpdateBy(sysUser.getUserId());

goodsInformation.setUpdateName(sysUser.getUsername());

return decide(goodsInformationService.updateGoodsInformation(goodsInformation));

}

/\*\*

\* 删除货品信息管理

\*/

@ResponseBody

@DeleteMapping("/batchRemove")

@PreAuthorize("hasPermission('/system/information/remove','system:information:remove')")

public Result batchRemove(String ids) {

return decide(goodsInformationService.deleteGoodsInformationByIds(Convert.toStrArray(ids)));

}

/\*\*

\* 删除

\*/

@ResponseBody

@DeleteMapping("/remove/{id}")

@PreAuthorize("hasPermission('/system/information/remove','system:information:remove')")

public Result remove(@PathVariable("id") String id) {

return decide(goodsInformationService.deleteGoodsInformationById(id));

}}

1.4 入院信息管理类

1.4.1 Domain目录下代码

public class MedicalAdmissionInformation extends BaseDomain{

private static final long serialVersionUID=1L;

/\*\* null \*/

private String id;

/\*\* 姓名 \*/

private String nickname;

/\*\* 性别 \*/

private String sex;

/\*\* 年龄 \*/

private Long age;

/\*\* 身份证 \*/

private String idcard;

/\*\* 联系电话 \*/

private String phone;

/\*\* 入院时间 \*/

private Date hospitalizedDate;

/\*\* 就诊医生 \*/

private String visitingDoctor;

/\*\* 诊断状态 \*/

private String diagnosticStatus;

/\*\* 就诊结果 \*/

private String visitResults;

/\*\* 备注 \*/

private String remarks;

}

1.4.2 Controller目录下代码

public class MedicalAdmissionInformationController extends BaseController {

private String prefix = "system/admissionInformation";

@Autowired

private IMedicalAdmissionInformationService medicalAdmissionInformationService;

@GetMapping("/main")

@PreAuthorize("hasPermission('/system/admissionInformation/main','system:admissionInformation:main')")

public ModelAndView main() {

return jumpPage(prefix + "/main");

}

/\*\*

\* 查询入院信息管理列表

\*/

@ResponseBody

@GetMapping("/data")

@PreAuthorize("hasPermission('/system/admissionInformation/data','system:admissionInformation:data')")

public ResultTable list(@ModelAttribute MedicalAdmissionInformation medicalAdmissionInformation, PageDomain pageDomain) {

PageInfo<MedicalAdmissionInformation> pageInfo = medicalAdmissionInformationService.selectMedicalAdmissionInformationPage(medicalAdmissionInformation, pageDomain);

return pageTable(pageInfo.getList(), pageInfo.getTotal());

}

/\*\*

\* 新增入院信息管理

\*/

@GetMapping("/add")

@PreAuthorize("hasPermission('/system/admissionInformation/add','system:admissionInformation:add')")

public ModelAndView add() {

return jumpPage(prefix + "/add");

}

/\*\*

\* 新增保存入院信息管理

\*/

@ResponseBody

@PostMapping("/save")

@PreAuthorize("hasPermission('/system/admissionInformation/add','system:admissionInformation:add')")

public Result save(@RequestBody MedicalAdmissionInformation medicalAdmissionInformation) {

SysUser sysUser = (SysUser) SecurityUtil.currentUser();

medicalAdmissionInformation.setId(SequenceUtil.makeStringId());

medicalAdmissionInformation.setCreateTime(LocalDateTime.now());

medicalAdmissionInformation.setCreateBy(sysUser.getUserId());

medicalAdmissionInformation.setCreateName(sysUser.getUsername());

return decide(medicalAdmissionInformationService.insertMedicalAdmissionInformation(medicalAdmissionInformation));

}

/\*\*

\* 修改入院信息管理

\*/

@GetMapping("/edit")

@PreAuthorize("hasPermission('/system/admissionInformation/edit','system:admissionInformation:edit')")

public ModelAndView edit(String id, ModelMap mmap) {

MedicalAdmissionInformation medicalAdmissionInformation =medicalAdmissionInformationService.selectMedicalAdmissionInformationById(id);

mmap.put("medicalAdmissionInformation", medicalAdmissionInformation);

return jumpPage(prefix + "/edit");

}

/\*\*

\* 修改保存入院信息管理

\*/

@ResponseBody

@PutMapping("/update")

@PreAuthorize("hasPermission('/system/admissionInformation/edit','system:admissionInformation:edit')")

public Result update(@RequestBody MedicalAdmissionInformation medicalAdmissionInformation) {

SysUser sysUser = (SysUser) SecurityUtil.currentUser();

medicalAdmissionInformation.setUpdateTime(LocalDateTime.now());

medicalAdmissionInformation.setUpdateBy(sysUser.getUserId());

medicalAdmissionInformation.setUpdateName(sysUser.getUsername());

return decide(medicalAdmissionInformationService.updateMedicalAdmissionInformation(medicalAdmissionInformation));

}

/\*\*

\* 删除入院信息管理

\*/

@ResponseBody

@DeleteMapping("/batchRemove")

@PreAuthorize("hasPermission('/system/admissionInformation/remove','system:admissionInformation:remove')")

public Result batchRemove(String ids) {

return decide(medicalAdmissionInformationService.deleteMedicalAdmissionInformationByIds(Convert.toStrArray(ids)));

}

/\*\*

\* 删除

\*/

@ResponseBody

@DeleteMapping("/remove/{id}")

@PreAuthorize("hasPermission('/system/admissionInformation/remove','system:admissionInformation:remove')")

public Result remove(@PathVariable("id") String id) {

return decide(medicalAdmissionInformationService.deleteMedicalAdmissionInformationById(id));

}}