重庆第二师范学院

**2018届全日制本科生毕业论文**

**题 目： 基于Java SSM框架的酒**

**店管理系统的设计与实现**

**学 院： 数学与信息工程学院**

**专业年级： 2014级物联网工程**

**学生姓名： 张百全 学号 1410504147**

**指导教师： 罗 军 职称 副教授**

**2018 年 3 月 4 日**

基于Java SSM框架的酒店管理系统的设计与实现

数学与信息工程学院　物联网工程专业2014级　张百全

指导教师 罗军

**摘 要：**随着科技与经济的迅速发展，尤其是互联网以及移动互联网的广泛普及，企业对公司业务的管理也越来越走向网络化、智能化。在本文中，根据酒店对业务管理的的需求，设计并着手实施了可供酒店管理者工作人员使用的酒店管理系统。此平台基于Java EE技术，使用了开源的Java SSM（Spring，SpringMVC，MyBatis）框架，是一个当今非常流行的MVC架构，旨在打造一个低耦合、高重用，可靠性高，方便维护，易根据酒店业务修改功能的酒店管理系统。本系统前端界面由jsp来实现，后台功能使用Java语言实现，使用MySQL数据库，Tomcat作为整个系统的后台服务器构建。

**关键词：**Java EE； SSM开源框架；MVC架构；低耦合；MySql；Tomcat

**Abstract:** With the rapid development of technology and economy, especially the popularization of the Internet and mobile Internet, the management of the company's business is becoming more and more networked and intelligent. In this paper, according to the demand of the hotel for the business management, the hotel management system designed and implemented for the staff of the hotel manager is designed and implemented. Based on Java EE technology and using the open source Java SSM (Spring, SpringMVC, MyBatis) framework, this platform is a very popular MVC architecture today designed to create a low-coupled, high-reuse, high-reliability, easy-to-maintain Hotel business modification function of hotel management system. The system front-end interface to achieve by the jsp, background functions using the Java language to achieve, the use of MySQL database, Tomcat as the entire system background server built.

**Keywords:** Java EE; SSM open source framework; MVC architecture; low coupling; MySQl; Tomcat。

**目录**

目录

1 绪论 5

1.1 开发背景 5

1.2 选课目的 5

1.3 研究内容 6

1.4 研究方法 6

2 开发环境与技术 6

2.1 开发环境 6

2.1.1 Eclipse简介 6

2.1.2 Tomcat简介 7

2.1.3 Mysql数据库简介 8

2.2 开发技术 10

2.2.1 Java EE 简介： 10

2.2.1 Java SSM(Spring,SPringMVC,Mybatis)简介 11

3 概要设计 12

3.1 功能模块 13

3.1.1 酒店管理系统模块设计 13

3.2 设计思想以及架构 13

3.2.1 设计思想 13

3.2.2 MVC架构 13

3.2.3 使用框架的好处 14

4 基于Java SSM框架的酒店管理系统的总设计 14

4.1 系统相关技术介绍 14

4.2 可行性分析 14

4.2.1 管理上的可行性 14

4.2.2 技术上的可行性 15

4.3 功能需求 15

4.3.1 系统框架结构示意流程图 16

4.3.2 数据库概述 16

4.3.3 数据表说明 16

4.4 系统测试、运行与维护 18

4.4.1 系统测试 18

4.4.2 系统运行与维护 19

4.4.3 系统评估 20

结论 20

致谢 20

参考文献 21

附录 22

# 1 绪论

## 1.1 开发背景

酒店业是一个前景广阔而又竞争激烈的行业。改革开放以来，我国的酒店业迅速发展，已经成为一个具有相当规模的产业。由于我国的旅游业迅速发展，通过调查，我国在2020年将成为世界上第四大旅游国家；同时我国加入世界贸易组织，酒店业将完全开放，这个时候，我国的酒店业将面临着前所未有的机遇和挑战。但是，现在甚至还有一些酒店还停留在由人工操作和管理阶段，这样已经无法适应当前的发展趋势。因此，要想使酒店的工作质量和效率提高，采用先进的计算机网络通信技术改变酒店业务模式，实现酒店业务管理的自动化已经成为一种必然。酒店管理系统采用B/S架构。按照国家星级宾馆标准化业务程序流式设计，会采用主流开发工具来开发，搭建比较稳定可靠安全的服务端来辅佐客户管理，并且还采用数据库MySQL，会通过测试用例来达到测试效果。

## 1.2 选课目的

对酒店整个来说，对酒店经营状况起决定作用的是酒店的服务管理水平。如何利用先进的管理手段来提高酒店的管理水平成为酒店业务发展的当务之急。面对信息时代的机遇和挑战，利用科技手段提高酒店的管理无疑是一条行之有效的途径。虽然计算机管理并不是酒店管理走向成功的关键元素，但它可以最大限度地发挥准确、快捷、高效等作用，对酒店的业务管理提供强有力的支持。因此，采用全新的计算机网络和酒店业务管理系统，已成为提高酒店的管理效率，使作业人员与管理系统之间灵活互动，实现流畅的工作流衔接，帮助酒店有效地进行业务管理，释放最大价值。酒店业务管理系统在达到在节省人力资源成本的同时，可以提高业务效率，并能够及时、准确、迅速地满足顾客服务的需求。

## 1.3 研究内容

本系统为管理者提供了完整的管理平台。其提供的主要功能有：客房类型管理，客房状态管理，客房预定管理，客房入住管理，客户换房管理，客房结算管理，退出系统。

## 1.4 研究方法

1.通过指导老师的帮助。

2.通过网络查阅相关文档资料，以及运用相关开发文档。

3.用Eclipse、Navicat Premium、Mysql、Maven、Notepad++等工具进行开发。

4.数据库采用Mysql数据库。

5.和同学们之间的交流和讨论

6.用Java语言编写代码，反复调试程序，以实现该系统的功能

# 2 开发环境与技术

## 2.1 开发环境

### 2.1.1 Eclipse简介

Eclipse是一个社区为个人和组织希望协作commercially-friendly开源软件。其项目大都集中于构建开放的开发平台由可扩展的框架、工具和运行时构建、部署和管理软件的整个生命周期。Eclipse基金会是一个非营利,成员支持承载Eclipse项目,帮助公司培养一个开源社区和互补的生态系统产品和服务。它是功能丰富的JavaEE集成开发环境，包括了完备的编码、调试、测试和发布功能，完整支持HTML,Struts,JSP,CSS,Javascript,SQL, Hibernate。

在结构上，Eclipse的特征可以被分为5类：

1. J2EE模型

2. WEB开发工具

3. 应用程序服务器的连接器

4. J2EE项目部署服务

5. 数据库服务

对于以上每一种功能上的类别，在Eclipse中都有相应的功能部件，并通过一系列的插件来实现它们。Eclipse结构上的这种模块化，可以让我们在不影响其他模块的情况下，对任一模块进行单独的扩展和升级。

### 2.1.2 Tomcat简介

Tomcat 是在SUN 公司的JSWDK（JavaServer Web DevelopmentKit ,是SUN公司推出的小型Servlet/JSP调试工具）的基础上发展起来的一个优秀的Servlet容器，Tomcat本身完全用java语言编写。

Servlet是一种运行在服务器上的小插件，他所具有的的以下特点：

1、提供了可被服务器加载并执行的程序代码，并服务于来自客户端的请求。

2、Servlet服务器必须支持java。

3、Servlet不依赖浏览器，是允许在服务器端，所以不管浏览器是否支持java，都能请求访问servlet。

Tomcat做为Servlet的容器基本功能如图一

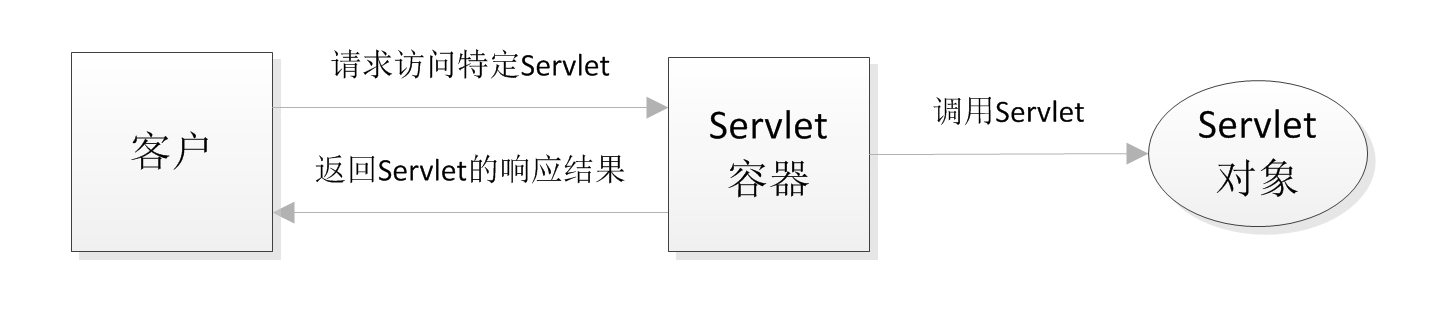


图2-2-1 Servlet原理图

Servlet 的规范规定Servlet不仅可以运行在基于HTTP协议的Web服务器上，还可以运行在基于其他应用协议的服务器上。不过，目前Servlet主要运行在Web服务器上，用来扩展Web服务器的功能。Servlet规范规定，Servlet容器响应客户请求访问特定的Servlet的流程如下：

1、客户发出要求访问特定的Servlet的请求。

2、Servlet 容器接收到客户请求并解析。

3、Servlet 容器创建一个ServletRequest对象，在ServletRequest对象中包含了客户请求信息及其他关于客户的信息，如请求头，请求正文，以及客户机的IP地址等。

4、Servlet容器创建一个ServletResponse对象

5、Servlet 容器调用客户所请求的Servlet的service服务方法，并且把ServletRequst对象和ServletResponse对象做为参数传给该服务方法。

6、Servlet从ServletRequest对象中可获取客户的请求信息。

7、Servlet利用ServletResponse对象来生成响应结果。

8、Servlet容器把Servlet生成的响应结果发送给客户。

### 2.1.3 Mysql数据库简介

MySQL是一个关系型数据库管理系统，由瑞典MySQL AB 公司开发，目前属于 [Oracle](http://baike.baidu.com/item/Oracle) 旗下产品。MySQL 是最流行的关系型数据库管理系统之一，在 WEB 应用方面，MySQL是最好的 RDBMS (Relational Database Management System，关系数据库管理系统) 应用软件。

MySQL是一种关系数据库管理系统，关系数据库将数据保存在不同的表中，而不是将所有数据放在一个大仓库内，这样就增加了速度并提高了灵活性。

MySQL所使用的 SQL 语言是用于访问数据库的最常用标准化语言。MySQL 软件采用了双授权政策，分为社区版和商业版，由于其体积小、速度快、总体拥有成本低，尤其是开放源码这一特点，一般中小型网站的开发都选择 MySQL 作为网站数据库。

MySQL 是一个真正的多用户、多线程SQL数据库服务器。SQL(结构化查询语言)是世界上最流行的和标准化的数据库语言，它使得存储、更新和存取信息更加容易。 MySQL是一个客户机/服务器结构的实现，它由一个服务器守护程序mysqld和许多不同的客户程序以及库组成的。

MySQL 的主要目标是快速、健壮和易用。最初是因为MySQL的创始人需要这样的一个SQL服务器，它能处理与任何不昂贵硬件平台上提供数据库的厂家在一个数量级 上的大型数据库，但速度更快，MySQL因此就开发出来了。它提供C，C++，JAVA(JDBC)，Perl，Python，PHP and TCL 的API接口;多平台支持，包括Solaris，SunOS，BSDI，SGI IRIX，AIX，DEC UNIX，Linux，FreeBSD，SCO OpenServer，NetBSD，OpenBSD，HPUX，Win9x and NT;多样的数据类型，包括有/无符号1，2，3，4，8字节integers，FLOAT，DOUBLE，CHAR，VARCHAR，TEXT， BLOB，DATE，DATETIME，YEAR，SET，ENUM;非常灵活和安全的权限系统，密码加密;为Windows提供ODBC接口，可通过 Access与之相联，另有第三方开发商提供多样的 ODBC驱动程序。

MySQL 自身不支持Windows的图形界面，因此，所有的数据库操作及管理功能都只能在MS-DOS方式下完成。首先，必须登陆MySQL服务器，即在提示符\ MySQL\BIN>下输入mysql –u root -p，提示符改为mysql>时表示登陆成功。然后，选择所操作的数据库，即mysql> use databasename。接着就可以用SQL语句进行查询了。如果需要退出数据库服务器，可执行quit命令，这样就成功退出服务器了。当然，由于 MySQL的知名度日益增加，许多第三方软件公司推出了MySQL在Windows环境中的具有图形界面的支持软件，如EMS公司的EMS MYSQL MANAGER就提供了Windows形式的MySQL数据库操作功能。

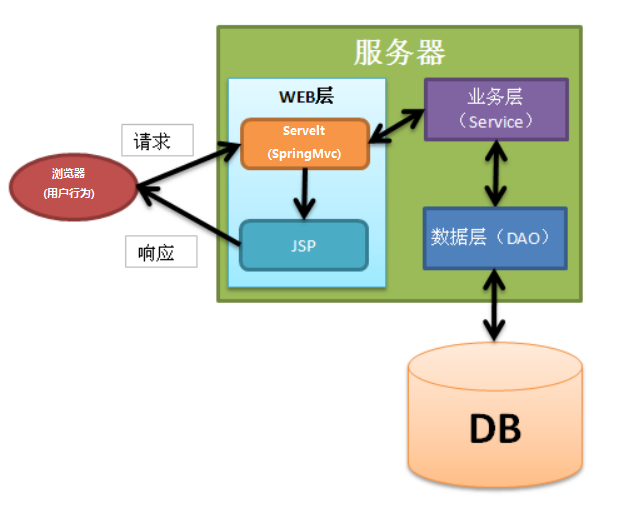


图2-1-3 数据库构架图

## 2.2 开发技术

### 2.2.1 Java EE 简介：

如上所述，Java EE平台旨在帮助开发人员创建大规模，多层次，可扩展，可靠和安全的网络应用程序。这种应用程序的缩写名称是“企业应用程序”，因为这些应用程序旨在解决大型企业遇到的问题。然而，企业应用不仅对大公司，机构和政府有用。企业应用程序的益处对于日益网络化的世界中的个人开发人员和小型组织是有帮助的，甚至是至关重要的。

使企业应用程序功能强大的功能（如安全性和可靠性）常常使这些应用程序变得复杂。Java EE平台通过提供允许开发人员专注于功能的开发模型，API和运行时环境来降低企业应用程序开发的复杂性。

[表2-2-1](https://docs.oracle.com/javaee/7/firstcup/java-ee001.htm#GCTOG)列出了Java EE应用程序中用于Web层的一些主要Java EE技术。

| **技术** | **目的** |
| --- | --- |
| JavaServer Faces技术 | 用于Web应用程序的用户界面组件框架，允许您在XHTML页面上包含UI组件（如字段和按钮），称为Facelets页面; 转换和验证UI组件数据; 将UI组件数据保存到服务器端数据存储; 并保持组件状态 |
| 表达语言 | Facelets页面中使用的一组标准标记来引用Java EE组件 |
| Servlet | Java编程语言类动态处理请求并构建响应，通常用于HTML页面 |
| Java EE的上下文和依赖注入 | 一组上下文服务，使开发人员可以在Web应用程序中使用企业bean和JavaServer Faces技术 |

表2-2-1 Web层Java EE技术

### 2.2.1 Java SSM(Spring,SPringMVC,Mybatis)简介

SSM框架，是Spring + Spring MVC + MyBatis的缩写，这个是继SSH之后，目前比较主流的Java EE企业级框架，适用于搭建各种大型的企业级应用系统。

Spring简介：

Spring是一个开源框架，Spring是于2003年兴起的一个轻量级的Java开发框架，由Rod Johnson在其著作Expert One-On-One J2EE Development and Design中阐述的部分理念和原型衍生而来。它是为了解决企业应用开发的复杂性而创建的。Spring使用基本的JavaBean来完成以前只可能由EJB完成的事情。然而，Spring的用途不仅限于服务器端的开发。从简单性、可测试性和松耦合的角度而言，任何Java应用都可以从Spring中受益。 简单来说，Spring是一个轻量级的控制反转（IoC）和面向切面（AOP）的容器框架。

Spring MVC简介：

Spring MVC属于Spring Framework的后续产品，已经融合在Spring Web Flow里面，它原生支持的Spring特性，让开发变得非常简单规范。Spring MVC 分离了控制器、模型对象、分派器以及处理程序对象的角色，这种分离让它们更容易进行定制。  
Spring MVC的架构是一个实现代码间低耦合很流行的软件设计的架构，下面这个是它主要的工作原理图：



图2-2-1 Spring MVC原理图

MyBatis简介：

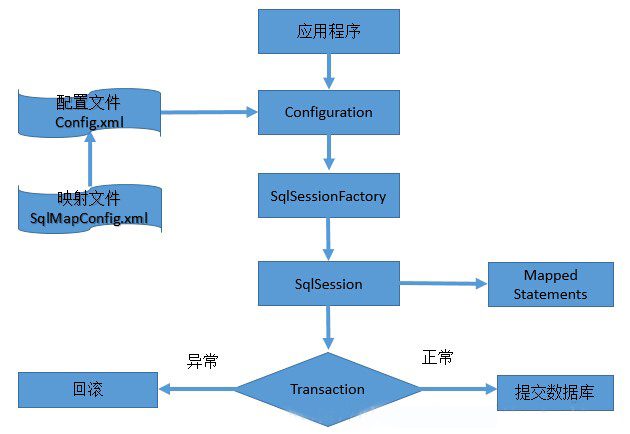
MyBatis本是apache的一个开源项目iBatis, 2010年这个项目由apache software foundation 迁移到了google code，并且改名为MyBatis 。MyBatis是一个基于Java的持久层框架。iBATIS提供的持久层框架包括SQL Maps和Data Access Objects（DAO）MyBatis消除了几乎所有的JDBC代码和参数的手工设置以及结果集的检索。MyBatis使用简单的XML或注解用于配置和原始映射，将接口和Java的POJOs（Plain Old Java Objects，普通的 Java对象）映射成数据库中的记录。可以这么理解，MyBatis是一个用来帮你管理数据增删改查的框架。它的结构如下图所示：  


图2-2-1 Mybatis原理图

# 3 概要设计

概要设计主要是将需求分析进行细化，划分功能模块，以及采用的何种技术和设计思想以及数据库的设计

## 3.1 功能模块

### 3.1.1 酒店管理系统模块设计

房间管理模块：预订房间管理、房间查询、退房管理。

前台服务员的主要职能是负责订房和退房，以及查询入住的客户信息。所有该角色只可以使用部分功能，包括预定管理、客户信息查询，入住管理，退房管理功能。

营销管理模块：分为普通客户和vip客户。进行客户资料查询、新增客户信息，修改客户信息，删除客户信息，以及每位客户在酒店的消费。

系统设置模块：整个酒店管理系统的中枢控制，对酒店房间的增删改查酒店中新增房间，即可在此模块中新增信息。

## 3.2 设计思想以及架构

### 3.2.1 设计思想

将整个系统按模块进行划分，采用多种设计模式，提高代码的可读性以便对代码进行修改和对功能的完善，降低类之间的耦合性。按照分层的思想来实现，是程序模块化。

### 3.2.2 MVC架构

本系统采用的是MVC架构，模型层（Model）、视图层（View）、控制层（Controller）三层结构。

1）、模型(Model)：通常Action类中主要处理控制功能，尽量不包括业务逻辑，而是把业务逻辑的处理委托给其他组件处理。

2)、视图(View)视图层就是前台终端界面，我们把数据通过Action返回到视图层，显示在前台终端上。

3)、控制器(Controller)：MVC架构中，用一个总控制器，然后再根据请求的URL转到相应的业务Action里面去，在由业务Action去访问service层, service层调用DAO层。

### 3.2.3 使用框架的好处

由于运用MVC的应用程序的三个部件是相互对立，改变其中一个不会影响其它两个，所以依据这种设计思想能构建良好的松偶合的组件，增大复用可能。对我们来说，控制器的也提供了一个额外好处，就是可以使用控制器来连接不同的模型和视图去完成用户的需求，这样控制器可以为构造应用程序提供强有力的手段。给定一些可重用的模型和视图，控制器就可以根据用户的需求选择模型进行处理，然后选择视图将处理结果显示给用户。

# 4 基于Java SSM框架的酒店管理系统的总设计

## 4.1 系统相关技术介绍

本管理信息系统将采用技术成熟的Java语言作为前台开发工具。后台数据库采用MySQL数据库，因为它的灵活性、安全性和易用性为数据库编程提供了良好的条件。因此，系统的软件开发平台已成熟可行。硬件方面，科技的飞速发展使硬件的更新速度越来越快，容量越来越大，可靠性越来越高，价格越来越低，其硬件平台完全能满足此系统的需要。系统采用的是原型法的开发方法。本着开发人员对用户需求的理解，通过简单的系统反洗、系统设计，快速实现一个原型系统，然后通过反复修改来实现管理信息系统。

## 4.2 可行性分析

### 4.2.1 管理上的可行性

酒店管理系统能记录酒店客人的基本信息，提供查询、删除等工作，使工作人员从烦琐的手工操作中解脱出来，同时，能为企业的运作提供提供管理和决策支持。由于酒店电脑系统24小时连续运行，数据量大，可靠性要求高，因此整个电脑系统供电需采用专线方式，加配UPS（不间断供电系统）合理接地，以便保障整套系统的正常运行。

### 4.2.2 技术上的可行性

技术采用成熟的java作为前台开发工具。后台数据库采用MySQL数据库，灵活性、安全性和易用性为数据库编程提供了良好的条件。

### 4.3 功能需求

酒店住房业务的基本流程主要包括：

客房的预定，顾客通过电话或者其他途径向酒店进行客房预定，接待人员查询客房状态表后确定有空房则接受预定，预定成功后发给顾客预定凭证，并对客房状态表做出相应的变化。

客房的入住，如果顾客已经预定房间，则根据预定凭证进行预定入住操作，发给顾客房卡，客房状态由“预定”变为“入住”。

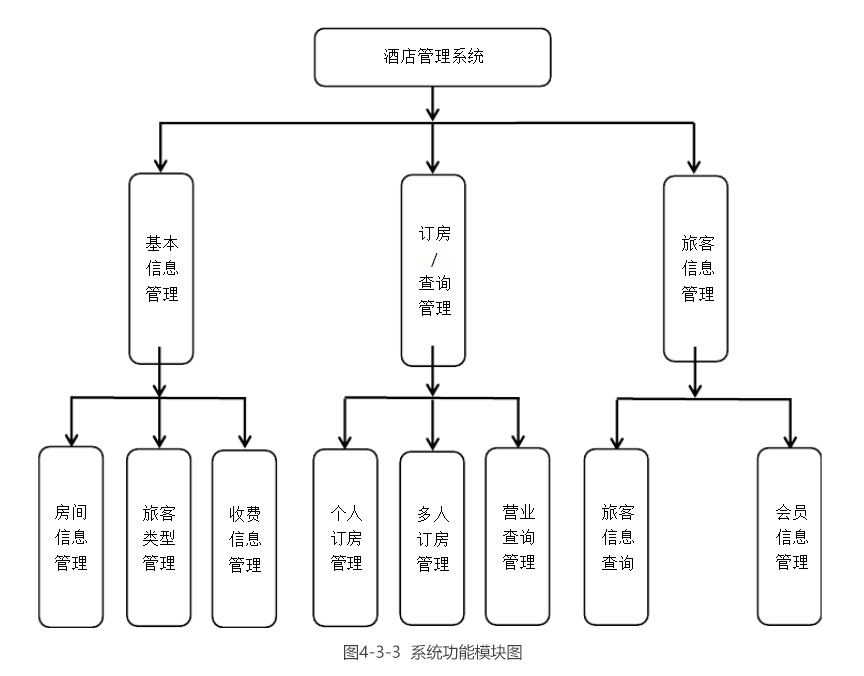
如果顾客没有预定房间，接待人员查询客房状态后，有空房则进行客房登记操作，发给顾客房卡，客房状态由“空房”变为“入住”。

顾客换房，如果顾客对入住的客房条件不满意，则可进行顾客换房操作。接待人员将顾客的房间进行更换，并更换顾客房卡，客房状态也进行相应的调整。

补交押金，如果顾客换房后押金太少，则要补交押金。顾客交纳押金，接待人员开具押金凭证。

顾客退房，顾客退房时，接待人员根据顾客的房卡，进行退房登记。计算顾客入住期间的房费，餐费，话费，各种消费和押金总额。对顾客收取费用或者退回押金。顾客信息转存到历史顾客表中。客房状态由“入住”转为“空房”。

### 4.3.1 系统框架结构示意流程图



4-3-3 系统框架结构示意流

### 4.3.2 数据库概述

该课题旨在实现本项目核心的权限管理的功能，就是在同一个登陆界面不同角色的用户拥有不同的权限。在登陆本系统后会显示有不同的菜单选项。经查阅资料后我清楚了实现这样的一个功能需求，需要建立这样的五张数据表，分别为用户表，角色表，权限表，用户角色权限关系表，角色权限关系表，如下为本系统涉及到的主要数据表。

### 4.3.3 数据表说明

员工信息表（employee）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 中文名 | 字段类型 | 长度 | 备注 |
| emplyeeId | 员工id | int | 11 | 主键 |
| usename | 员工名称 | varchar | 255 |  |
| realname | 员工姓名 | varchar | 255 |  |
| phone | 手机号 | varchar | 255 |  |
| titleId | 职务 | varchar | 255 |  |
| deptId | 部门编号 | varchar | 255 |  |
| deptName | 部门名称 | varchar | 255 |  |
| groupId | 分组类别 | varchar | 255 |  |
| Isdelete | 删除标识符 | int | 1 | 0未删除，1删除 |
| Ex1 | 备用字段 | varchar | 255 | 预留字段1 |
| Ex2 | 备用字段 | varchar | 255 | 预留字段2 |
| Ex3 | 备用字段 | varchar | 255 | 预留字段3 |

角色表（userclassinfo）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 中文名 | 字段类型 | 长度 | 备注 |
| UserClassID | 用户类别id | int | 11 | 主键 |
| UserClassName | 用户类别名称 | varchar | 255 |  |
| ClassNotes | 类别备注 | longtext | 0 |  |

职务信息表(title)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 中文名 | 字段类型 | 长度 | 备注 |
| titleId | 职务ID | Varchar | 50 | 主键 |
| titleName | 职务名称 | Varchar | 50 |  |
| EX1 | 备用字段 | Varchar | 50 |  |
| EX2 | 备用字段 | Varchar | 50 |  |
| EX3 | 备用字段 | Varchar | 50 |  |

角色权限关系表（m\_functioncontrol）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 中文名 | 字段类型 | 长度 | 备注 |
| ID | 功能编号 | varchar | 11 | 主键 |
| PXID | 功能排序号 | int | 11 |  |
| NAME | 功能名称 | varchar | 255 |  |
| URL | 功能链接 | varchar | 55 |  |
| PID | 上级编号 | varchar | 255 |  |
| ACTIVATE | 是否启用 | int | 1 | 0：未启用  1：启用 |

用户角色关系表（m\_sys\_funcsecurity）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 中文名 | 字段类型 | 长度 | 备注 |
| FID | 编号 | int | 11 | 主键 |
| MIS\_USERID | 用户分组 | int | 50 |  |
| FUNCTION\_ID | 功能编号 | varchar | 12 |  |

部门信息表（department）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 中文名 | 字段类型 | 长度 | 备注 |
| deptId | 部门ID | varchar | 50 | 主键 |
| deptName | 部门名称 | varchar | 50 |  |
| company | 公司 | varchar | 50 |  |
| parentId | 上级部门 | varchar | 50 |  |
| Isdelete | 删除标识符 | int | 1 | 0：未删除  1：已删除 |
| EX1 | 备用字段 | varchar | 50 | 预留字段1 |
| EX2 | 备用字段 | varchar | 50 | 预留字段2 |
| EX3 | 备用字段 | varchar | 50 | 预留字段3 |

客房信息表（room）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 中文名 | 字段类型 | 长度 | 备注 |
| roomid | 主键id | int | 11 | 主键 |
| roomnum | 房间号 | varchar | 255 |  |
| floor | 房间楼层 | varchar | 255 |  |
| type | 房间类型 | varchar | 255 |  |
| price | 房间价格 | decimal | 10 |  |
| state | 房间状态 | varchar | 255 |  |
| isreservation | 是否有预定 | varchar | 255 |  |
| reservationtime | 预定时间 | datetime | 0 |  |
| arrive | 到达时间 | datetime | 0 |  |
| deposit | 押金 | varchar | 255 |  |
| instructions | 房间说明 | varchar | 255 |  |
| EX1 | 备用字段1 | varchar | 255 |  |
| EX2 | 备用字段2 | varchar | 255 |  |
| EX3 | 备用字段3 | varchar | 255 |  |
| EX4 | 备用字段3 | varchar | 255 |  |
| EX5 | 备用字段5 | varchar | 255 |  |

## 4.4 系统测试、运行与维护

### 4.4.1 系统测试

(1)检查本系统的查询、修改和增加、删除、批量删除功能是否正常实现。

(2)检查系统流程图和模块之间关系图、模块内部之间关系图。  
(3)识别特殊情况，如错误处理，异常处理，错误提示是否合理。  
(4)检查用户界面是否符合合理美观，界面操作是否便捷易用。

### 4.4.2 系统运行与维护

本管理系统的维护是针对系统的各种功能组件，按照不同模块的维护，系统维护的内容大致可分为以下几类：

（1）系统应用维护。应用软件维护是系统维护的主要内容。指相应申请和相关文件的修改和完善。业务流程系统通过应用操作实现，一旦有问题或业务变化，将不可避免地导致修改和调整过程，所以主要的活动是维护系统维护程序。

（2）数据维护。数据库是支持业务运营的基础平台。对数据处理业务的需求不断变化，除了主要业务系统数据定期更新外，还需要定期更新数据，并根据环境或业务进行调整或调整，增加数据结构的内容和数据结构调整。此外，数据备份和恢复等是数据维护的工作。

（3）代码维护。代码维护是指扩展原代码，添加或删除维护。随着系统应用范围的扩大和应用环境的变化，系统中的各种代码需要增加，修改，删除和设置新的代码。

（4）硬件维护。主要是指对计算机和外围设备的日常维护和管理，如机器零件的清洁，润滑，维修设备故障，如更换易损件，工作应由负责人定期进行，以确保正常工作有效。

（5）机构和人员变动。信息系统是人机系统，也起着重要的作用。为了使信息系统的过程更加合理，有时涉及机构和人员的变更。这种变化也会影响设备和程序的维护。

### 4.4.3 系统评估

评估有五个主要因素：评价的主体，评价的对象，评价的目标，评价手段（标准，方法，工具）和评价的实施。

# 结论

2018年1月，我开始了毕业论文和毕业设计的工作，到目前为止，该系统预期功能已基本完成。 从最初的如同一张空白的纸，到慢慢进入研究开发状态，然后逐渐理清思路，明确方向。整个的过程很难用语言说清楚。 经过几个月的努力，紧张和实质的毕业项目终于结束了。回想起一路来的经历和感受，虽然很辛苦，越到很多困难和问题，但最终还是圆满的完成了毕业设计，在这整个过程中，真的是有太多令人难忘的记忆和收获。

本课题设计并实现基于了Java SSM（Spring，SpringMVC，Mybatis）和MySQL组合的酒店管理系统。

# 致谢

历时两个多月，我最终完成了论文的写作和毕业设计。从最初的拿到论文题目到系统的设计实现，再到论文文章的写作完成。每一个阶段都充满的很多的挑战，但正是由于存在挑战，使得我完成了我整个大学阶段最大也是最得意的一个项目。通过这个项目的实现，我也发掘了我的潜力，也学会了很多以前不了解不熟悉的知识，更多的是学会了去尝试解决问题的勇气和方法。同时也收获了很多人的关心与帮助，很感谢我的指导老师在我的项目设计和论文写作上对我的建议与指导，还有我的同学朋友们甚至当我遇到问题难以解决时论坛上一些不知名的对我的帮助。

我的论文和设计仍有很多不成熟，待完善的地方，这也是和我目前的知识技术阅历有关，但是此次论文写作的过程已经让我受益终身了。我感受到了知识只有通过实践才能表达发挥出来它的价值，同时也能检验自己的掌握程度。学习同时也是一需要个持续的过程，只有通过不断的学习，才能掌握更深的技术，解决问题的方法和思路也会更加广泛。希望这次的经历能引领我在以后的工作与学习中继续努力，不断进步。

参考文献

[1] 王洪波. 基于Java的小型酒店管理信息系统的设计与实现[J]. 贺州学院计算机科学与信息工程学院,2016.

[2] 杨旭.J2EE企业级开发.[M].北京:清华大学出版社2016.3.

[3] 关兵. 基于B/S模式的酒店管理信息系统[D].2010(11).

[4] 王辰. 基于 Java 的中小型酒店管理系统设计与实现[D],2015.12

[5] 萨师煊,王珊.数据库系统概论.北京:高等教育出版社[M], 2005.7.

[6] 朱福喜,余振坤.J2EE 经典实例详解[M]. 北京:人民邮电出版社, 2006, 26-183.

[7] 陈刚.Eclipse 从入门到精通[M]. 北京:清华大学出版社, 2007:6-20.

[8] 贾蓓,镇明敏,杜磊. Java Web整合开发实战:基于Struts 2+Hibernate+Spring [M] 清华大学出版社2013.7

[9]杨开振.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]2017.7.1

[10]疯狂软件.Spring+MyBatis企业应用实战[M],2017.1.1

附录

///HotelApr/src/main/java/com/zhangbaiquan/pro/controller/IndexController.java

package com.zhangbaiquan.pro.controller;

import java.util.ArrayList;

import java.util.List;

import javax.servlet.http.HttpSession;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.servlet.ModelAndView;

import com.zhangbaiquan.pro.po.base.Employee;

import com.zhangbaiquan.pro.po.power.MFunctioncontrol;

import com.zhangbaiquan.pro.po.power.MFunctioncontrolExample;

import com.zhangbaiquan.pro.po.power.MSysFuncsecurity;

import com.zhangbaiquan.pro.po.power.MSysFuncsecurityExample;

import com.zhangbaiquan.pro.po.power.MSysFuncsecurityExample.Criteria;

import com.zhangbaiquan.pro.service.power.FunctionControlService;

import com.zhangbaiquan.pro.service.power.SysFunctionService;

@Controller

public class IndexController {

@Autowired

private SysFunctionService sysFunctionService;

@Autowired

private FunctionControlService functionControlService;

@RequestMapping(value="index")

public ModelAndView showIndex(ModelAndView mv,HttpSession session){

Employee user=(Employee) session.getAttribute("user");

if(user==null){

mv.setViewName("redirect:/loginPage.action");

}

// if (!user.getEx1().equals(pwd) { // 密码错误

// result.put("result", -2);

// writeJSON(response, result);

// return;

// }

List<String> fidlist=new ArrayList<String>();

if(user!=null&&user.getGroupid()!=null) {

MSysFuncsecurityExample funcrexample=new MSysFuncsecurityExample();

Criteria funcCriteria=funcrexample.createCriteria();

funcCriteria.andMisUseridEqualTo(new Integer(user.getGroupid()));

funcCriteria.andLockEqualTo(0);

List<MSysFuncsecurity> sflist= sysFunctionService.selectByExample(funcrexample);

if(sflist!=null) {

for (MSysFuncsecurity mSysFuncsecurity : sflist) {

fidlist.add(mSysFuncsecurity.getFunctionId());

}

}

}else {

mv.setViewName("login");

return mv;

}

MFunctioncontrolExample example1 =new MFunctioncontrolExample();

com.zhangbaiquan.pro.po.power.MFunctioncontrolExample.Criteria criteria1=example1.createCriteria();

criteria1.andIdIn(fidlist);

criteria1.andActivateEqualTo(1);

criteria1.andPidEqualTo("0");

example1.setOrderByClause("pxid");

List<MFunctioncontrol> parentlist=functionControlService.selectByExample(example1);

MFunctioncontrolExample example =new MFunctioncontrolExample();

com.zhangbaiquan.pro.po.power.MFunctioncontrolExample.Criteria criteria=example.createCriteria();

criteria.andIdIn(fidlist);

criteria.andActivateEqualTo(1);

example.setOrderByClause("pxid");

List<MFunctioncontrol> funclist=functionControlService.selectByExample(example);

mv.addObject("parentlist", parentlist);

mv.addObject("funclist", funclist);

mv.setViewName("index");

return mv;

}

@RequestMapping(value="loginPage")

public ModelAndView loginPage(ModelAndView mv){

mv.setViewName("login");

return mv;

}

@RequestMapping(value="mainPage")

public ModelAndView mainPage(ModelAndView mv){

mv.setViewName("main");

return mv;

}

@RequestMapping(value="defaultPage")

public ModelAndView defaultPage(ModelAndView mv){

mv.addObject("message", "此功能暂未实现，请先使用其他功能");

mv.setViewName("default");

return mv;

}

}

// /HotelApr/src/main/java/com/zhangbaiquan/pro/controller/base/UserinfoController.java

package com.zhangbaiquan.pro.controller.base;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import javax.servlet.http.HttpSession;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.ResponseBody;

import org.springframework.web.servlet.ModelAndView;

import com.zhangbaiquan.pro.po.base.Department;

import com.zhangbaiquan.pro.po.base.DepartmentExample;

import com.zhangbaiquan.pro.po.base.Employee;

import com.zhangbaiquan.pro.po.base.EmployeeExample;

import com.zhangbaiquan.pro.po.base.Title;

import com.zhangbaiquan.pro.po.base.Userclassinfo;

import com.zhangbaiquan.pro.service.base.DeptService;

import com.zhangbaiquan.pro.service.base.EmployeeService;

import com.zhangbaiquan.pro.service.base.UserClassService;

import com.zhangbaiquan.pro.util.PageBean;

@Controller

@RequestMapping("user")

public class UserinfoController {

@Autowired

private DeptService deptService;

@Autowired

private EmployeeService employeeService;

@Autowired

private UserClassService userClassService;

@RequestMapping(value="list")

public ModelAndView list(ModelAndView mv,Integer page,String key,String dept){

PageBean paging=new PageBean();

paging.setPageSize(5);

if(page!=null&&page>0){

paging.setCurrentPage(page);

}

EmployeeExample example=new EmployeeExample();

EmployeeExample.Criteria criteria=example.createCriteria();

criteria.andIsdeleteEqualTo(0);

if(key!=null&&key!=""){

criteria.andRealnameLike("%"+key+"%");

}

if(dept!=null&&dept!=""){

criteria.andDeptnameEqualTo(dept);

}

criteria.andIsdeleteEqualTo(0);

example.setPageSize(paging.getPageSize());

example.setStartPos(paging.getStartPos());

List<Employee> list= employeeService.selectByPageAndExample(example);

List<Object> jsonList=new ArrayList<Object>();

for (int i = 0; i < list.size(); i++) {

Map<String, Object> map = new HashMap<String, Object>();

Employee employee=list.get(i);

if(employee.getTitleid()!=null&&!"".equals(employee.getTitleid())){

String[] idarray=employee.getTitleid().split(",");

StringBuffer permission = new StringBuffer();

for (String id : idarray) {

Title tList=employeeService.selectByTitleKey(id);

String titlename=null;

titlename=tList.getTitlename();

if (permission.toString().equals("")) {

permission.append(titlename);

}

else{

permission.append(","+titlename);

}

}

map.put("titlename", permission.toString());

}

//Map<String, Object> map = new HashMap<String, Object>();

map.put("emplyeeid",list.get(i).getEmplyeeid());

map.put("usename", list.get(i).getUsename());

map.put("realname", list.get(i).getRealname());

map.put("phone", list.get(i).getPhone());

map.put("deptname", list.get(i).getDeptname());

jsonList.add(map);

}

paging.setList(jsonList);

paging.setTotalRows(employeeService.countByExample(example));

mv.addObject("paging", paging);

mv.addObject("key", key);

mv.addObject("dept", dept);

mv.addObject("deptList", deptService.selectByExample(null));

mv.setViewName("base/user/list");

return mv;

}

@RequestMapping(value="sendAdd")

public ModelAndView sendAdd(ModelAndView mv,Integer deptid){

DepartmentExample example=new DepartmentExample();

DepartmentExample.Criteria criteria=example.createCriteria();

criteria.andIsdeleteEqualTo(0);

List<Department> deptList= deptService.selectByExample(example);

mv.addObject("deptList", deptList);

mv.addObject("grouplist", userClassService.selectByExample(null));

mv.addObject("titlelist", employeeService.selectTitleExample(null));

mv.setViewName("base/user/add");

mv.addObject("deptid", deptid);

return mv;

}

@RequestMapping(value="titlelist")

@ResponseBody

public List<Title> titlelist(){

return null;

}

@RequestMapping(value="add")

public ModelAndView add(ModelAndView mv,Employee user){

DepartmentExample example=new DepartmentExample();

DepartmentExample.Criteria criteria=example.createCriteria();

criteria.andDeptidEqualTo(user.getDeptid());

List<Department> deptList= deptService.selectByExample(example);

user.setDeptname(deptList.get(0).getDeptname());

user.setIsdelete(0);

int row=employeeService.insertSelective(user);

if(row>0){

mv.setViewName("redirect://user/list.action");

}else{

mv.addObject("message", "添加失败，请重试");

mv.setViewName("redirect://user/sendAdd.action");

}

return mv;

}

@RequestMapping(value="deleteUser")

@ResponseBody

public String deleteUser(String id){

int row=employeeService.deleteByPrimaryKey(id);

if(row>0){

return "success";

}else{

return "error";

}

}

@RequestMapping(value="sendUpdate")

public ModelAndView sendUpdate(ModelAndView mv,String id){

Employee list=employeeService.selectByPrimaryKey(id);

String[] idarray=list.getTitleid().split(",");

StringBuffer permission = new StringBuffer();

for (String ids : idarray) {

Title tList=employeeService.selectByTitleKey(ids);

String titlename=null;

titlename=tList.getTitlename();

if (permission.toString().equals("")) {

permission.append(titlename);

}

else{

permission.append(","+titlename);

}

}

mv.addObject("user",list);

mv.addObject("titlename",permission.toString());

Integer groupid=Integer.parseInt(list.getGroupid());

Userclassinfo group=userClassService.selectByPrimaryKey(groupid);

mv.addObject("grouping",group);

DepartmentExample example=new DepartmentExample();

DepartmentExample.Criteria criteria=example.createCriteria();

criteria.andIsdeleteEqualTo(0);

List<Department> deptList= deptService.selectByExample(example);

mv.addObject("deptList", deptList);

mv.addObject("grouplist", userClassService.selectByExample(null));

mv.addObject("titlelist", employeeService.selectTitleExample(null));

mv.setViewName("base/user/update");

return mv;

}

@RequestMapping(value="update")

public ModelAndView update(ModelAndView mv,Employee user){

DepartmentExample example=new DepartmentExample();

DepartmentExample.Criteria criteria=example.createCriteria();

criteria.andDeptidEqualTo(user.getDeptid());

List<Department> deptList= deptService.selectByExample(example);

user.setDeptname(deptList.get(0).getDeptname());

int row=employeeService.updateByPrimaryKeySelective(user);

if(row>0){

mv.setViewName("redirect://user/list.action");

}else{

mv.addObject("message", "添加失败，请重试");

mv.setViewName("redirect://user/sendUpdate.action?id="+user.getEmplyeeid());

}

return mv;

}

@RequestMapping(value="login")

public ModelAndView login(ModelAndView mv,String username,String pwd,HttpSession session){

EmployeeExample example=new EmployeeExample();

com.zhangbaiquan.pro.po.base.EmployeeExample.Criteria criteria=example.createCriteria();

criteria.andUsenameEqualTo(username);

criteria.andIsdeleteNotEqualTo(1);

List<Employee> list=employeeService.selectByExample(example);

Employee user=null;

if(list!=null&&list.size()>0){

user=list.get(0);

session.setAttribute("user", user);

Department dept=deptService.selectByPrimaryKey(user.getDeptid());

session.setAttribute("dept", dept);

mv.setViewName("redirect:/index.action");

}else{

mv.setViewName("redirect:/loginPage.action");

}

return mv;

}

}

// /HotelApr/src/main/java/com/zhangbaiquan/pro/controller/base/DeptController.java

package com.zhangbaiquan.pro.controller.base;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.ResponseBody;

import org.springframework.web.servlet.ModelAndView;

import com.zhangbaiquan.pro.po.base.Department;

import com.zhangbaiquan.pro.po.base.DepartmentExample;

import com.zhangbaiquan.pro.po.base.Employee;

import com.zhangbaiquan.pro.po.base.EmployeeExample;

import com.zhangbaiquan.pro.po.base.Goods;

import com.zhangbaiquan.pro.po.base.GoodsExample;

import com.zhangbaiquan.pro.po.base.DepartmentExample.Criteria;

import com.zhangbaiquan.pro.service.base.DeptService;

import com.zhangbaiquan.pro.service.base.EmployeeService;

import com.zhangbaiquan.pro.util.PageBean;

@Controller

@RequestMapping("/dept")

public class DeptController {

@Autowired

private DeptService deptService;

@Autowired

private EmployeeService employeeService;

@RequestMapping(value="list")

public ModelAndView showList(ModelAndView mv,Integer page,String sname) {

DepartmentExample example=new DepartmentExample();

Criteria criteria=example.createCriteria();

PageBean paging=new PageBean();

paging.setPageSize(5);

if(page!=null&&page>0){

paging.setCurrentPage(page);

}

if (sname!=null&&sname!="") {

criteria.andDeptnameLike("%"+sname+"%");

}

criteria.andIsdeleteEqualTo(0);

example.setPageSize(paging.getPageSize());

example.setStartPos(paging.getStartPos());

paging.setList(deptService.selectByPageAndExample(example));

paging.setTotalRows(deptService.countByExample(example));

mv.addObject("sname",sname );

mv.addObject("childlist",paging );

mv.setViewName("base/dept/list");

return mv;

}

@RequestMapping(value="sendAdd")

public ModelAndView sendAdd(ModelAndView mv) {

DepartmentExample example=new DepartmentExample();

Criteria criteria=example.createCriteria();

criteria.andIsdeleteEqualTo(0);

mv.addObject("childlist", deptService.selectByExample(example));

mv.setViewName("base/dept/add");

return mv;

}

@RequestMapping(value="addDept")

public ModelAndView addDept(ModelAndView mv,Department dept) {

dept.setIsdelete(0);

int row=deptService.insert(dept);

if(row>0) {

mv.addObject("message", "添加成功");

mv.setViewName("redirect:/dept/list.action");

}else {

mv.addObject("message", "添加失败");

mv.setViewName("base/dept/add");

}

return mv;

}

@RequestMapping(value="sendUpdate")

public ModelAndView sendUpdate(ModelAndView mv,String deptid) {

DepartmentExample example=new DepartmentExample();

Criteria criteria=example.createCriteria();

criteria.andIsdeleteEqualTo(0);

mv.addObject("childlist", deptService.selectByExample(example));

mv.addObject("dept", deptService.selectByPrimaryKey(deptid));

EmployeeExample EMP=new EmployeeExample();

EmployeeExample.Criteria CRT=EMP.createCriteria();

CRT.andDeptidEqualTo(deptid);

mv.addObject("employee", employeeService.selectByExample(EMP));

mv.setViewName("base/dept/update");

return mv;

}

@RequestMapping(value="updateDept")

public ModelAndView updateDept(ModelAndView mv,Department dept) {

//dept.setSortnum(deptService.selectByPrimaryKey(dept.getDeptid()).getSortnum());

Employee employee= employeeService.selectByPrimaryKey(dept.getEx3());

dept.setEx2(employee.getRealname());

dept.setIsdelete(0);

int row=deptService.updateByPrimaryKey(dept);

if(row>0) {

mv.addObject("message", "修改成功");

mv.setViewName("redirect:/dept/list.action");

}else {

mv.addObject("message", "添加失败");

mv.setViewName("base/dept/update");

}

return mv;

}

@ResponseBody

@RequestMapping(value="delete")

public String delete(String deptid) {

DepartmentExample example=new DepartmentExample();

Criteria criteria=example.createCriteria();

criteria.andParentidEqualTo(deptid);

deptService.deleteByExample(example);

int row=deptService.deleteByPrimaryKey(deptid);

if(row>0) {

return "success";

}else {

return "error";

}

}

///HotelApr/src/main/java/com/zhangbaiquan/pro/controller/roommanagement/RoomController.java

package com.zhangbaiquan.pro.controller.roommanagement;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpSession;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.ResponseBody;

import org.springframework.web.servlet.ModelAndView;

import org.springframework.web.servlet.mvc.support.RedirectAttributes;

import com.zhangbaiquan.pro.po.base.Employee;

import com.zhangbaiquan.pro.po.roommanagement.Room;

import com.zhangbaiquan.pro.po.roommanagement.RoomExample;

import com.zhangbaiquan.pro.service.roommanagement.RoomService;

import com.zhangbaiquan.pro.util.PageBean;

@Controller

@RequestMapping(value="/roommanagement")

public class RoomController {

@Autowired

private RoomService roomService;

@RequestMapping(value="/realtime\_room.action")

public ModelAndView realtime\_room(ModelAndView mv, Integer page, HttpSession session){

Employee user=(Employee) session.getAttribute("user");

if(user==null){

mv.setViewName("redirect:/loginPage.action");

return mv;

}

mv.setViewName("roommanagement/realtime\_room");

return mv;

}

@RequestMapping(value="/toroomadd.action")

public ModelAndView toroomadd(ModelAndView mv, HttpSession session){

Employee user=(Employee) session.getAttribute("user");

if(user==null){

mv.setViewName("redirect:/loginPage.action");

return mv;

}

mv.setViewName("roommanagement/roomadd");

return mv;

}

@RequestMapping(value="/roomadd.action")

public ModelAndView roomadd(ModelAndView mv, HttpSession session,Room room){

Employee user=(Employee) session.getAttribute("user");

if(user==null){

mv.setViewName("redirect:/loginPage.action");

return mv;

}

room.setRoomnum(room.getRoomnum());

room.setFloor(room.getFloor());

room.setType(room.getType());

room.setPrice(room.getPrice());

room.setState(room.getState());

room.setIsreservation(room.getIsreservation());

room.setInstructions(room.getInstructions());

roomService.insertSelective(room);

mv.setViewName("redirect:/roommanagement/realtime\_list.action");

return mv;

}

@RequestMapping(value="/toRoomModify")

public ModelAndView toRoomModify(ModelAndView mv,Integer roomid){

Room room = roomService.selectByPrimaryKey(roomid);

mv.addObject("room",room);

mv.setViewName("roommanagement/roomModify");

return mv;

}

@RequestMapping(value = "/modify.action")

public ModelAndView modify(ModelAndView mv, HttpServletRequest request,Room room, HttpSession session) {

Employee user = (Employee) session.getAttribute("user");

if (user == null) {

mv.addObject("msg", "登录超时，请重新登录");

mv.setViewName("login");

return mv;

}

Room rooma = roomService.selectByPrimaryKey(room.getRoomid());

// room.setRoomid(room.getRoomid());

room.setRoomnum(room.getRoomnum());

room.setFloor(room.getFloor());

room.setType(room.getType());

room.setPrice(room.getPrice());

room.setState(room.getState());

room.setIsreservation(room.getIsreservation());

room.setInstructions(room.getInstructions());

System.out.println(room);

roomService.updateByPrimaryKeySelective(room);

mv.setViewName("redirect:/roommanagement/realtime\_list.action");

return mv;

}

@RequestMapping(value="/delete")

public String delete(ModelAndView mv,Integer roomid){

roomService.deleteByPrimaryKey(roomid);

return "redirect:/roommanagement/realtime\_list.action";

}

@RequestMapping(value = "deletelist")

@ResponseBody

public String delete(String roomids) {

String[] idarray = roomids.split(",");

int rows = 0;

for (String id : idarray) {

int roomid = Integer.parseInt(id);

Room room = roomService.selectByPrimaryKey(roomid);

int row = roomService.deleteByPrimaryKey(roomid);

if (row > 0) {

RoomExample example = new RoomExample();

RoomExample.Criteria criteria = example.createCriteria();

criteria.andRoomidEqualTo(roomid);

roomService.deleteByExample(example);

rows += row;

}

}

if (rows > 0) {

return "success";

} else {

return "error";

}

}

@RequestMapping(value="/realtime\_list.action")

public ModelAndView realtime\_list(ModelAndView mv, Integer page,String roomnum,HttpSession session){

Employee user=(Employee) session.getAttribute("user");

if(user==null){

mv.setViewName("redirect:/loginPage.action");

return mv;

}

PageBean paging = new PageBean();

paging.setPageSize(5);

if (page != null && page > 0) {

paging.setCurrentPage(page);

}

RoomExample example = new RoomExample();

RoomExample.Criteria criteria = example.createCriteria();

if (roomnum != null && roomnum.length() > 0) {

criteria.andRoomnumEqualTo(roomnum);

mv.addObject("cardnum", roomnum);

}

example.setPageSize(paging.getPageSize());

example.setStartPos(paging.getStartPos());

paging.setTotalRows(roomService.countByExample(example));

// paging.setTotalRows(2);

paging.setList(roomService.selectByPageAndExample(example));

// PageBean pageBean = visitorinfoService.findListByPage(visitorinfoVo);

// mv.addObject("pageBean", paging);

mv.addObject("paging", paging);

System.out.println("PageSize = "+paging.getPageSize());

System.out.println("StartPos = "+paging.getStartPos());

// System.out.println("FirstPage = "+paging.getFirstPage());

System.out.println("TotalPages = "+paging.getTotalPages());

System.out.println("TotalRows = "+paging.getTotalRows());

System.out.println("CurrentPage = "+paging.getCurrentPage());

System.out.println("page = "+page);

mv.setViewName("roommanagement/realtime\_list");

return mv;

}

}