**编 号：**

**审定成绩：**

|  |  |  |  |
| --- | --- | --- | --- |
| **重庆邮电大学**  **毕业设计（论文）** | | | |
|  | | | |
|  | **中文题目** | **酒店预订系统的设计与实现** |  |
|  |  |
| **英文题目** | **The design and implementation of hotel** |
|  | **reservation system** |
| **学院名称** | **软件学院** |
| **学生姓名** | **韩昊** |
| **专 业** | **软件工程** |
| **班 级** | **1301309** |
| **学 号** | **2013214027** |
| **指导教师** | **张喜平** |
| **答 辩 组**  **负 责 人** | **姓名 职称** |
|  | |

**年 月**

**重庆邮电大学教务处制**

学院本科毕业设计(论文)诚信承诺书

本人郑重承诺：

我向学院呈交的论文《 》，是本人在指导教师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不含任何其他个人或集体已经发表或撰写过的作品成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明并致谢。本人完全意识到本声明的法律结果由本人承担。

年级

专业

班级

承诺人签名

年 月 日

摘要

无论在哪个行业，管理都起着非常重要的作用。而在酒店行业中，管理又是重中之重。管理酒店水平的高低，决定着酒店的生存和发展。在互联网行业、交通行业快速发展的大背景下，酒店行业也紧跟大时代的脚步，此时，酒店的管理方式也要求转型，由传统的管理方式到现代化的管理方式的转变。就现在来看，快速、高效的全方位网络化、信息化管理已成为必需的存在。

酒店为了减少管理成本，提高工作的管理效率，就需要选择使用一个信息化、智能化的稳定的网上酒店预订系统。因此，酒店预订系统成为了现代化酒店管理和发展的必不可少的元素组成。本系统前台设计采用HTML作为开发语言，后台使用Java语言进行逻辑判断，集成了SSM框架，并使用MySQL数据库进行数据的持久化管理。在项目开发中，用Eclipse作为集成开发工具、Maven作为项目管理工具、Git作为版本控制工具、Tomcat作为数据传递的中间件，Linux作为其服务器的应用平台。最终设计出一个基于Web、B/S结构的酒店预订系统。

**关键词：**酒店预订，Java，MySQL，B/S结构

**Abstract**

Management plays a very important role in any industry.In the hospitality industry, management is a top priority.Managing the level of the hotel determines the hotel's survival and development.In the Internet industry, transportation industry under the background of rapid development, the hotel industry also followed the footsteps of great age, at this point, the hotel management also requires transformation, from traditional management to modern management mode transformation.For now, rapid, efficient, all-directional networking and information-based management have become necessary.

Choose to use an information-based and intelligent online hotel reservation system, not only can reduce the management cost and improve management efficiency, simplify the management process, also can carry on the optimization of the hotel services. Therefore, in today's hotel management system, the modern hotel reservation system is an indispensable element. This design at the front desk system USES HTML as a development language, the background using the Java language logic, integrates the SSM framework, and using MySQL database for data persistence management. In project development, use Eclipse as an integrated development tool, Maven as a project management tool, Git as version control tool, Tomcat as the data transfer middleware, Linux as a server application platform. Finally a hotel reservation system based on the Web and B/S is designed.

**Keywords**:  Hotel Reservation, Java, MySQL, B/S Structure

目录

[第1章 引言 1](#_Toc483153996)

[1.1 研究背景 1](#_Toc483153997)

[1.2 国内外现状 1](#_Toc483153998)

[1.3 论文研究的主要内容 2](#_Toc483153999)

[第2章 关键技术介绍 3](#_Toc483154000)

[2.1 JavaScript 3](#_Toc483154001)

[2.2 Java 3](#_Toc483154002)

[2.3 三层架构模式 3](#_Toc483154003)

[2.4 Mybatis框架 4](#_Toc483154004)

[2.5 Spring框架 4](#_Toc483154005)

[第3章 系统需求分析 6](#_Toc483154006)

[3.1 系统总体需求 6](#_Toc483154007)

[3.2 系统详细设计 6](#_Toc483154008)

[3.3 可行性分析 7](#_Toc483154009)

[3.3.1 技术可行性 7](#_Toc483154010)

[3.3.2 经济可行性 8](#_Toc483154011)

[3.3.3 市场可行性 8](#_Toc483154012)

[第4章 系统概要设计 9](#_Toc483154013)

[4.1 系统用例图 9](#_Toc483154014)

[4.1.1 会员用例图 9](#_Toc483154015)

[4.1.2 管理员用例图 10](#_Toc483154016)

[4.2 整体流程设计 11](#_Toc483154017)

[4.2.1 会员中心操作流程图 11](#_Toc483154018)

[4.2.2 后台管理中心操作流程图 12](#_Toc483154019)

[4.3 数据库设计 14](#_Toc483154020)

[4.3.1 数据库概念结构设计 14](#_Toc483154021)

[4.3.2 数据库逻辑结构设计 15](#_Toc483154022)

[第5章 系统详细设计 18](#_Toc483154023)

[5.1 系统框架设计 18](#_Toc483154024)

[5.2 系统功能流程图 19](#_Toc483154025)

[5.2.1 会员登录流程图 19](#_Toc483154026)

[5.2.2 会员预订客房流程图 19](#_Toc483154027)

[5.2.3 会员查看个人订单流程图 20](#_Toc483154028)

[5.2.4 密码修改流程图 21](#_Toc483154029)

[5.2.5 管理员操作客房流程图 22](#_Toc483154030)

[5.3 系统接口设计 22](#_Toc483154031)

[5.3.1 dao层接口设计 23](#_Toc483154032)

[5.3.2 service接口设计 24](#_Toc483154033)

[5.3.3 controller层接口设计 25](#_Toc483154034)

[第6章 系统实现 26](#_Toc483154035)

[6.1 系统开发及运行环境 26](#_Toc483154036)

[6.2 数据库连接 27](#_Toc483154037)

[6.3 实现展示 27](#_Toc483154038)

[6.3.1 会员登录界面实现 27](#_Toc483154039)

[6.3.2 个人中心界面实现 28](#_Toc483154040)

[第7章 结束语 31](#_Toc483154041)

[参考文献 32](#_Toc483154042)

[附录 33](#_Toc483154043)

[Spring Framework 33](#_Toc483154044)

[Getting Started with Spring 33](#_Toc483154045)

[Introduction to the Spring Framework 34](#_Toc483154046)

[Spring 框架 38](#_Toc483154047)

[开始使用Spring 39](#_Toc483154048)

# 第1章 引言

# 1.1 研究背景

在人们的生活水平、消费水平提高的大背景下，旅游业得到长足的发展，在这良好的发展环境下，酒店行业得以快速发展。但是，酒店的规模不断的扩大，管理成本不断增加，因此需要有一套实用且方便的系统进行酒店客房的管理。与此同时，计算机科学技术不断提高，移动互联网迅速发展，智能机、平板已经成为大众化的产品，人们上网浏览、查询自己所需要的信息已经称为常态。也正因为此，酒店预订系统随着大众需求、酒店需求而产生，这为酒店行业解决了很多实际问题。例如：极低的出错概率，极高的保密性质和很低的成本管理等。这些优势能极大的提高酒店管理人员的运作效率和客户的入住率。

## 1.2 国内外现状

随着旅游业、交通行业的极速发展，人员流动也越来越频繁。酒店行业得以迅速发展。酒店数目迅速增加，但是，面对不断扩大的酒店规模，与此相关的订房管理类信息量也几何倍的增长。面对越来越激烈的市场竞争，也为了在这场持久战中能够生存发展下去，作为酒店方，不但需要良好的经济基础，还需要快速优质的客户服务，而这就需要一个最优的解决方案，解决酒店客房和顾客之间的供应关系，从而使得酒店的顾客入住率达到最大，从而获得最大的利益。

在互联网技术的成熟背景和个人可携带通信设备的普及下，酒店可以与时俱进，跟随互联网的脚步，做“互联网+”酒店，开设网上酒店预订系统，使得酒店线上和线下联动，达到最佳的服务状态，获得最大的利益。这使得计算机应用的优势得以显现：提高酒店客房管理的运作效率、规范化酒店客房信息，提高服务质量、减少酒店管理成本，提高了酒店的管理水平和经济效益。

## 1.3 论文研究的主要内容

全文共分为6章，内容结构安排如下：

第1章 引言：主要描述了论文研究的背景和国内外现状；

第2章 关键技术介绍：主要描述了开发时使用的技术、分层的设计思想；

第3章 系统需求分析：主要大致描述了酒店预订系统所需要实现的功能，并对其技术、经济和市场三个方面进行了可行性分析。

第4章 系统概要设计：主要展现了系统用例图、整体功能和数据库的设计，并列出数据库表结构及字段信息；

第5章 系统详细设计：主要介绍了系统框架和系统几个主要功能流程图设计情况；

第 6 章 系统实现：系统开发及运行环境，各个页面的实现结果进行展示。

第 7 章 结束语：总结毕业设计，并对未来自己的职业发展有一定的展望。

# 第2章 关键技术介绍

## 2.1 JavaScript

JavaScript是一种解释性脚本语言，即代码不进行预编译。在本系统中很多地方使用了JavaScript技术，比如说，便捷使用的JQuery工具类、Bootstrap时间组件，以及对会员输入的数据进行合法性检验。

## 2.2 Java

Java是一门强类型的面向对象编程语言，它提出了万物皆可为对象的编程思想。因为其开源性，可以在社区中发现很多开源的、好用的组件，最常使用的就是Apache基金会下的各类开源组件，提高了程序员的编程效率。

Java可以编写桌面应用程序、Web应用程序、分布式系统和嵌入式系统应用程序等 ，尤其在服务端编程中，Java占有了很高的比例和极高的地位。

## 2.3 三层架构模式

三层架构模式是一种思想，一种分层的设计思想。这种思想将应用功能在逻辑上分为三个层次：View（视图）、Controller（控制器） 和 Model（模型层）。View是用于显示处理后数据的部分，Controller是处理会员交互及对请求进行分派和转发部分，Model是处理应用程序数据逻辑的部分。

分层模式的主要优点为：

1. 灵活和可扩展性。对应用层进行响应改变就可达到需求更改的要求。
2. 共享性。服务端可以为处在不同平台的客户端应用程序提供对应的服务。
3. 安全性。客户端无法直接访问数据库。服务器端可以控制访问的权限及数据更改的权限。
4. 可用性。可用性指的是逻辑代码的可重复利用性。不同的开发项目可以使用同一个组件。

## 2.4 Mybatis框架

Mybatis框架不是一个典型的ORM。它是同坐自定义SQL和关系型数据进行映射，然后得到我们需要的对象。因为其半ORM化，这使得我们能够写出高性能的SQL。Mybatis本身是轻量级框架，非常容易学习和上手。Mybatis的核心接口一共有5个，如图2.1.

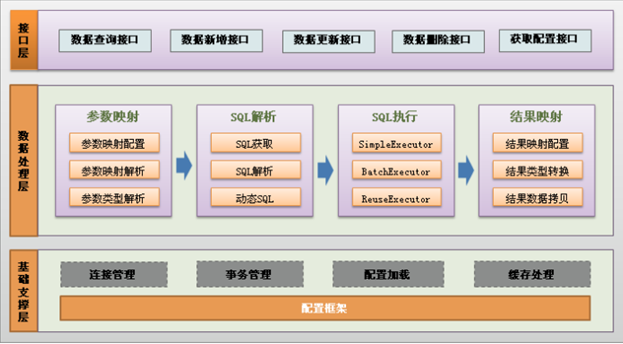


图 2.1 Mybatis功能架构设计

## 2.5 Spring框架

Spring是一款轻量级的开源框架，因其零入侵型受到开发者的欢迎。Spring架构设计借鉴了Unix的架构设计，所以说spring是模块化的集成框架，Spring框架由7个定义良好的模块组成，如图2.2。Spring很好的实现了IOC和AOP的思想，为使用的开发者大大减少了编码的工作量，更关注于业务逻辑的实现，提高了开发效率。

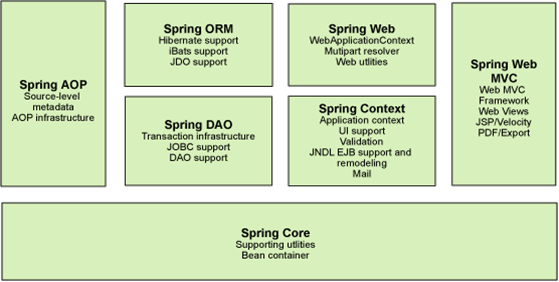


图2.3 spring框架

# 第3章 系统需求分析

### 3.1 系统总体需求

通过对该酒店管理人员的访谈和了解，需要根据酒店对客房预订自主研发一套酒店预订系统，为客户提供快捷优质的服务的同时，管理人员也能更加高效的管理客房。

根据业务需求，本系统可以划分为4个模块：注册模块、登录模块、个人中心管理模块、后台管理模块。

注册模块：顾客填写信息注册称为酒店会员。

登录模块：会员登录进入会员个人中心，管理员登录进入管理员个人中心。

个人中心管理模块：会员登陆后可以预定客房并查看个人订单。

后台管理模块：管理员登陆后，可以确定会员预订的客房状态。

## 3.2 系统详细设计

通过与酒店管理人员进行访谈以及对业务流程的调研，确定了系统应实现一下的功能：

管理员登录后台管理首页后，可以按照自己的需求进行以下操作。

（1）会员管理，包括对会员基本信息的查看，以及对会员帐号的注销。

（2）客房管理，包括对客房信息的修改操作和客房状态的更改操作。

（3）订单查询，包括对订单按时间、订单号、身份证件的查询。

（4）密码修改，管理员可以修改自己的登录密码。

会员登录个人中心首页后，会员可以按照自己的需求进行相应的操作。

（1）个人信息，包括对个人基本信息的查看。

（2）客房预订，包括对可预订客房基本信息的展示及对客房的预订。

（3）订单查询，仅可以查看自己的订单，并可以查询一定时间范围的订单。

（4）密码修改，用于会员更改自己的登录密码。

依据对系统的需求分析，系统要实现两大功能模块：管理员的功能界面、会员的功能界面。每个功能模块的具体的功能如图3.1。

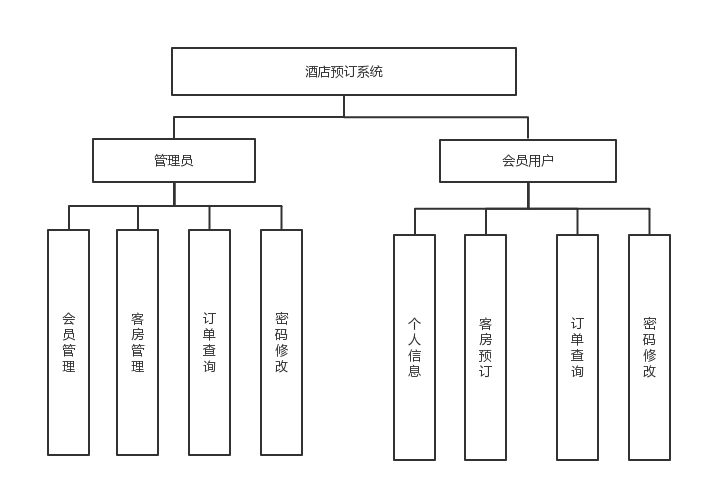


图3.1 系统功能图

## 3.3 可行性分析

### 3.3.1 技术可行性

本项目使用的开发工具是Eclipse，项目管理工具是Maven，项目版本控制工具为Git，采用标准的MVC三层架构的开发模式，使用的Spring、SpringMVC和Mybatis集成的SSM框架。通过Spring的IOC和AOP思想对应用进行解耦，并提高单组件的内聚，提高开发效率。

### 3.3.2 经济可行性

本项目除了应用服务器需要租用或购买外，其他都是免费产品，且服务器租用的维护费用是很低的。

### 3.3.3 市场可行性

为了能够在激烈的市场竞争存活发展下去，打好这场持久战，作为酒店，要始终坚持顾客至上的理念，提高酒店的服务和酒店的运作效率。为了使得酒店得以更好的管理和发展，这就要加强对酒店营业的分析和预测。酒店预订系统作为一款优秀的管理工具和分析工具，可以使得管理变得更简便，更高效。酒店客房的运营离不开管理人员的内部控制。酒店预订系统可以提供更加准确和及时的数据，这极大的帮助管理人员控制和决策，使得管理成本得以减少，酒店入住率得以提高，酒店利润达到最大化。

# 第4章 系统概要设计

## 4.1 系统用例图

就本系统而言参与者是指使用浏览器登录网站对房间进行预订的会员，以及通过网站登录后台对客房进行操作的人员。因此本系统中的会员分为两大类：分别是登录网站预订客房的会员和登录后台管理客房的人员。就此，用例图可分为会员用例图和管理员用例图。

### 4.1.1 会员用例图

如图4.1.1所示。

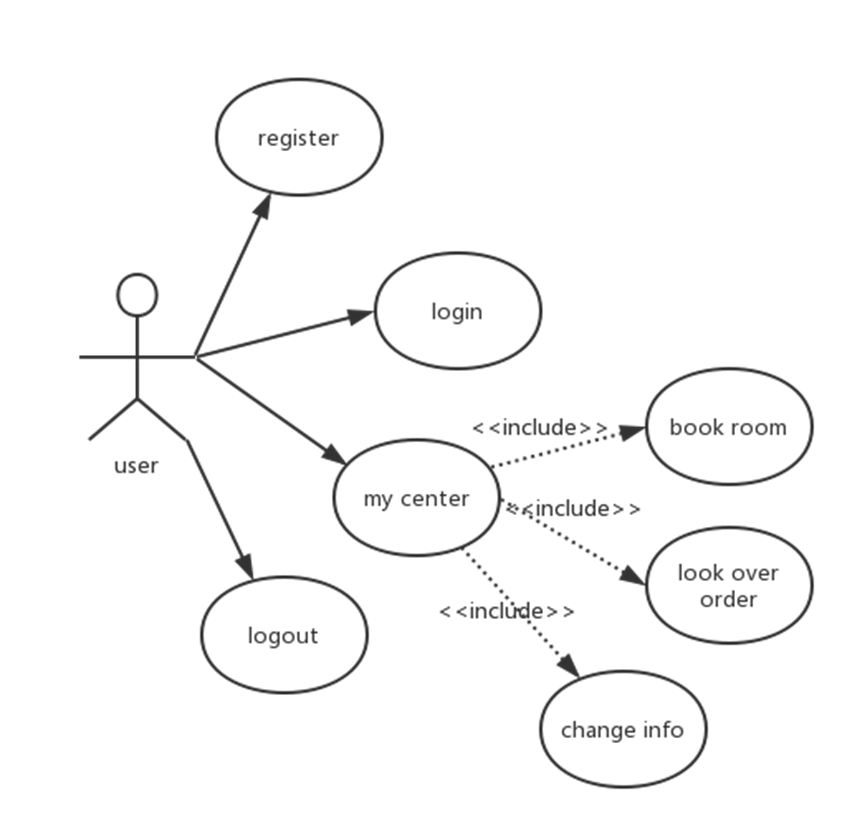


图 4.1.1

1. 会员

登录到网站个人中心的用户即为会员。

（2）注册

注册功能通过注册页面填写个人信息进行操作。用户可以通过填写个人信息注册成为会员。

（3）登录

登录功能通过登录页面输入帐号和密码进行操作。会员可以输入注册时填写的帐号和密码登录到个人中心页面。

（4）个人中心页面

已登录的会员可以在客人中心页面通过点击页面按钮进行个人信息查看、客房预订、个人订单查询、密码修改操作。

（5）登出

已登录的会员可以点击登出按钮登出账户，并会跳转到登录界面。

### 4.1.2 管理员用例图

如图4.1.2所示。

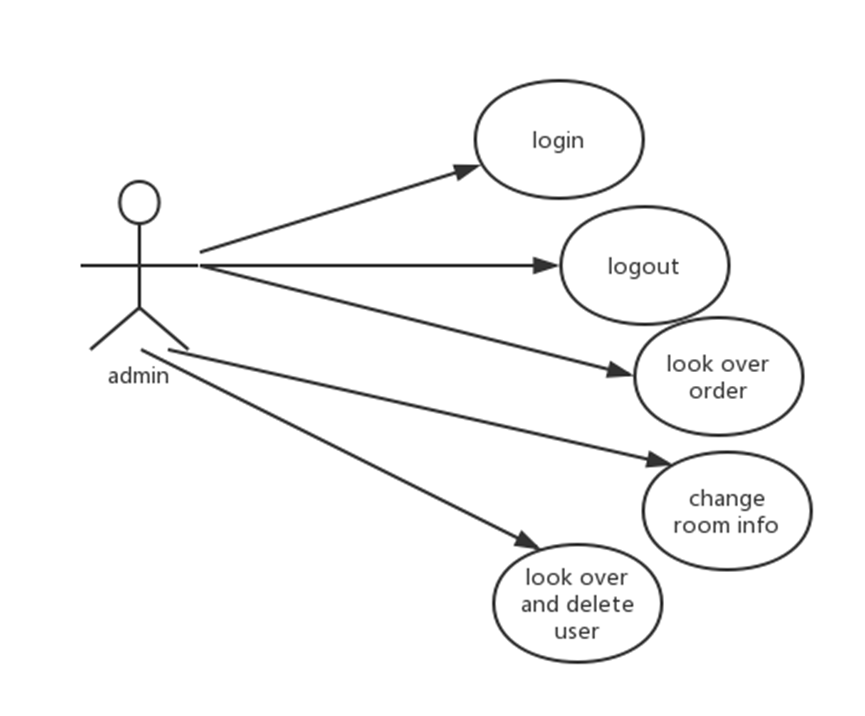


图 4.1.2

1. 管理员

酒店内部管理人员，并通过内部特殊帐号登录到后台管理中心的使用者。

1. 登录

在管理员登录界面，管理员输入内部帐号和密码即可登录到后台管理中心。

1. 登出

管理员退出时需要执行的操作，退出当前帐号，并会跳转到管理员登录界面。

1. 查看和删除会员信息

管理员可以通过点击会员管理来查看所有会员的基本信息，并可以删除指定的会员帐号。

1. 订单查询

管理员可以点击订单查询来查看所有订单，并可以依据会员信息来查看单个会员的订单。

1. 客房操作

管理员可以点击客房管理来查看所有客房，并在客房操作界面对客房信息进行修改、添加、删除操作，及对客房状态进行更改，包括：空客房、已预订、以入住状态。

## 4.2 整体流程设计

### 4.2.1 会员中心操作流程图

如图4.2.1所示。



图 4.2.1

### 4.2.2 后台管理中心操作流程图

如图4.2.2所示。



图4.2.2

## 4.3 数据库设计

### 4.3.1 数据库概念结构设计

单个会员可以多个未预订客房的信息，单个会员可以预订多个客房，所以会员和客房是1：n的关系，单个会员多次预定房间可以生成多张订单，所以会员和订单是1：n的关系。酒店预订系统设计的会员预订客房实体-关系图如图4.3.1-A所示。

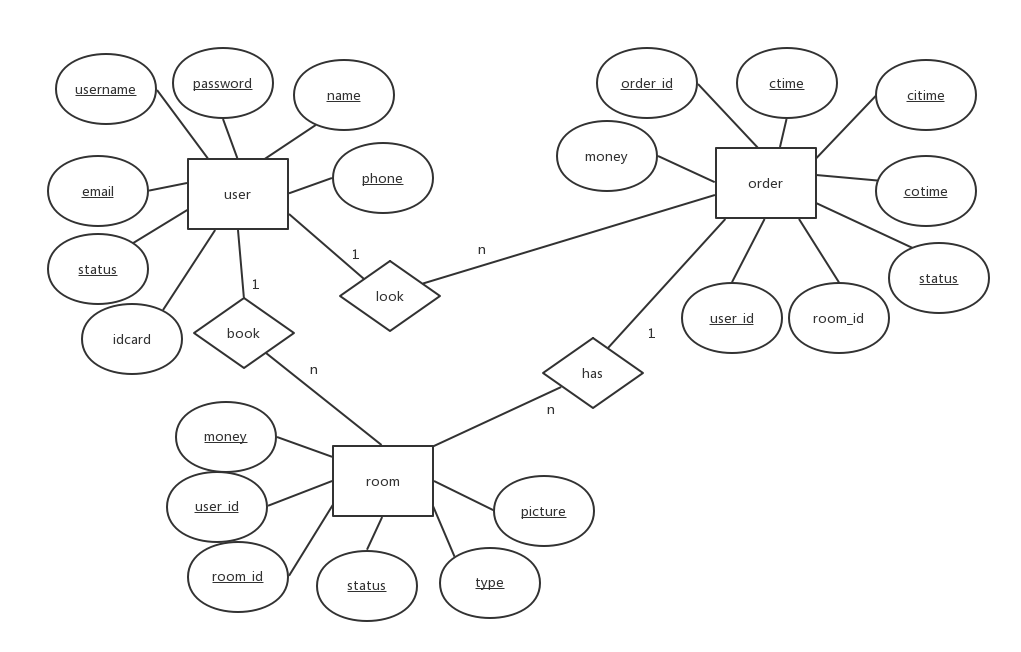


图 4.3.1-A

单个管理员可以对多个会员信息进行管理，所以是1：n的关系；会员用户对预定房间信息表进行管理时则是1：n的关系。管理员管理预订信息实体-关系图如图4.3.1-B。

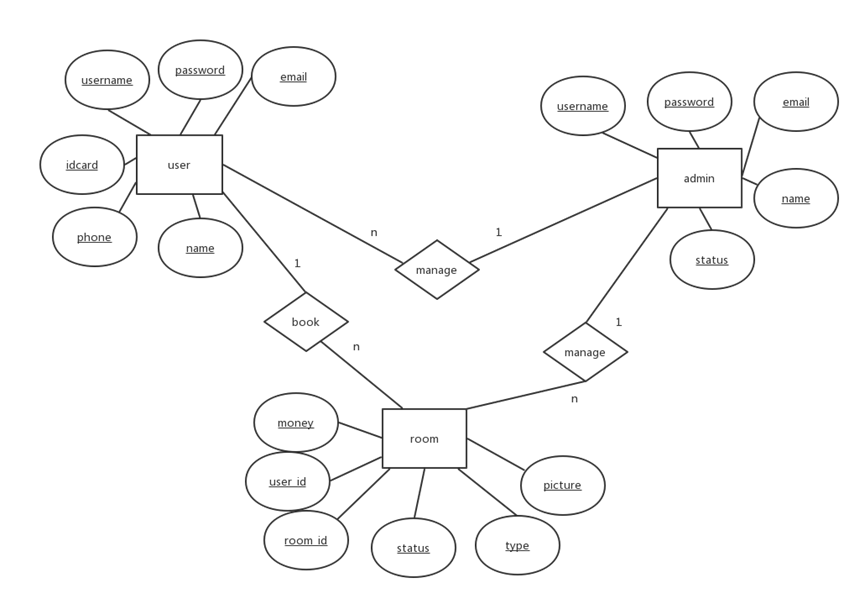


图4.3.1-B

### 4.3.2 数据库逻辑结构设计

（1）订单表

订单表（order）记录着会员预订客房的多种信息，其字段信息设计如图4.4.2-A所示。

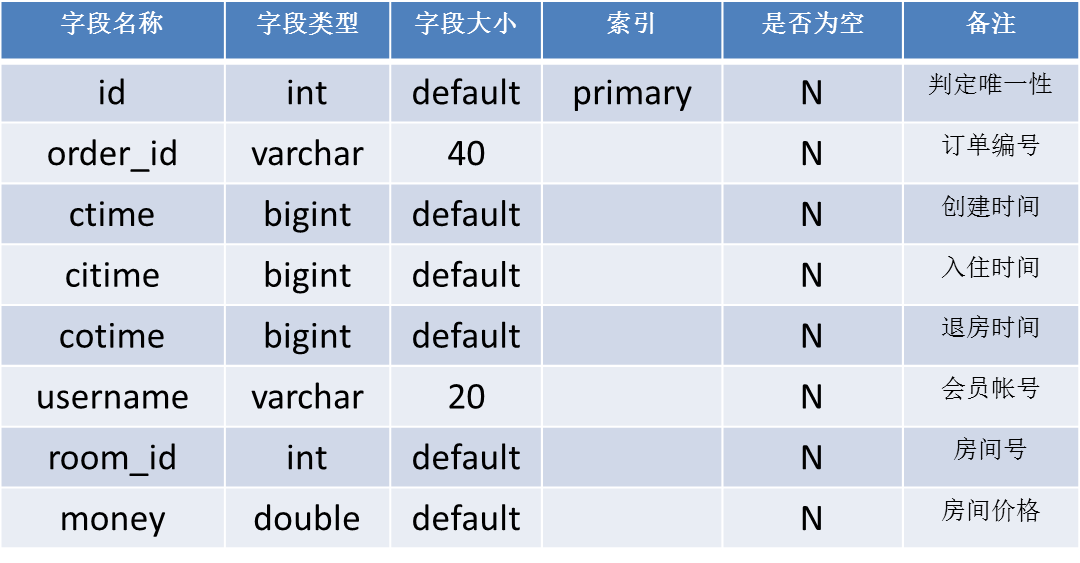


图4.4.2-A 客房预定表order

（2）客房表

客房表（room）记录着客房的各种详细信息，其字段具体设计如图4.4.2-B所示。

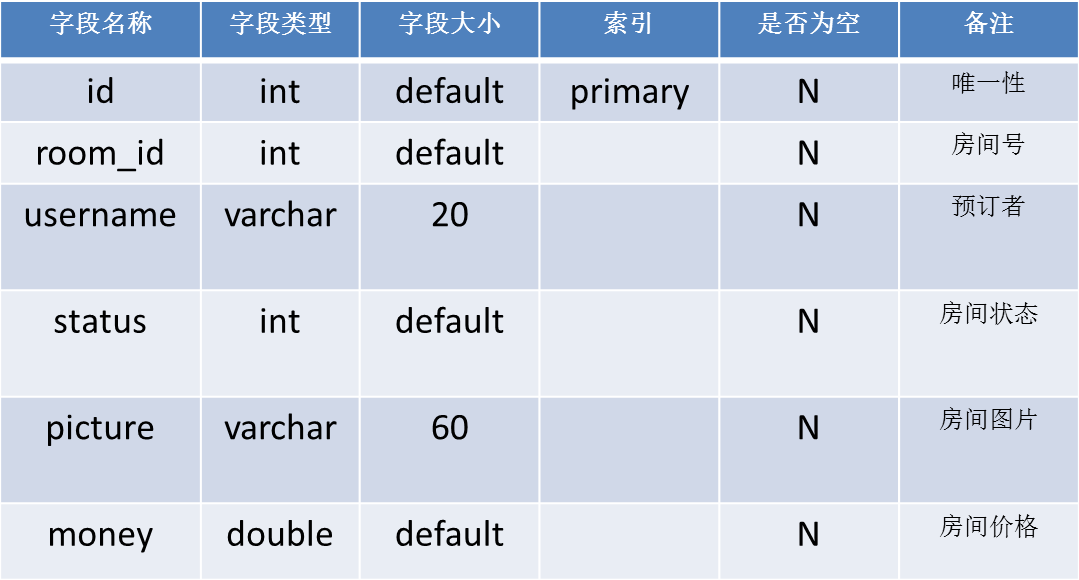


图4.4.2-B 客房表room

（3）会员表

会员表（user）记录着会员的各种信息，其字段信息具体设计如图4.4.2-C。

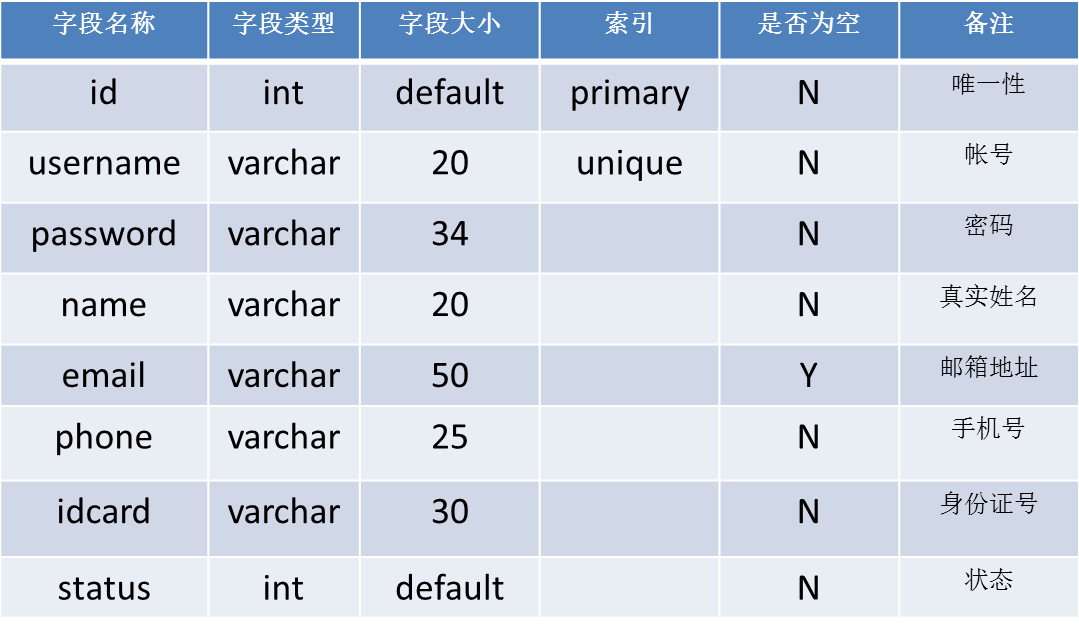


图4.4.2-C 会员表user

（4）管理员表

管理员表（admin）记录着管理员的各种信息。其字段信息具体设计如图 4.4.2-D。

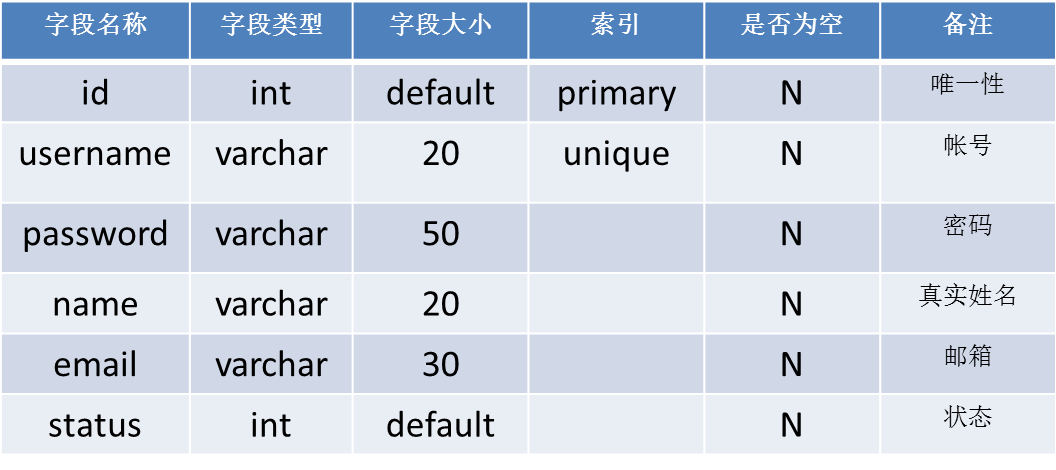


图 4.4.2-D 管理员表 admin

# 第5章 系统详细设计

## 5.1 系统框架设计

酒店预订系统网站选择使用三层架构设计。分别为表示层（User Interface layer）、业务逻辑（Business Logic Layer）、数据访问层(Data access layer)。使用的SSM框架设计如图5.1所示。

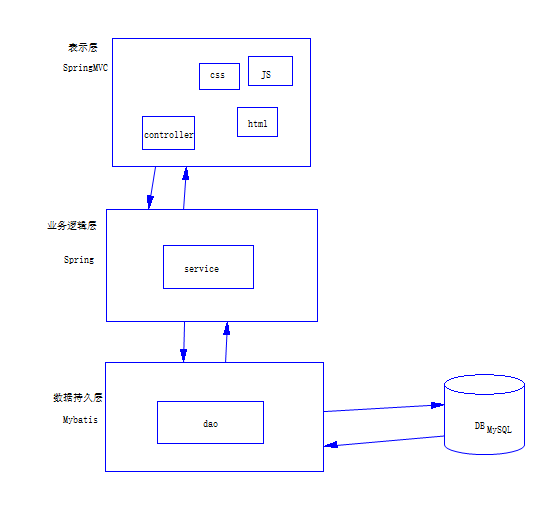


图 5.1

## 5.2 系统功能流程图

### 5.2.1 会员登录流程图

会员输入用户名和密码后，提交给服务器，服务器端对会员身份进行校验，如果验证通过则跳转至个人中心界面，过程如图5.2.1所示。



图 5.2.1

### 5.2.2 会员预订客房流程图

如图5.2.2所示。



图 5.2.2

### 5.2.3 会员查看个人订单流程图

如图5.2.3所示。



图 5.2.3

### 5.2.4 密码修改流程图

如图5.2.4所示。



图 5.2.4

### 5.2.5 管理员操作客房流程图

如图5.2.5所示。



图 5.2.5

## 5.3 系统接口设计

整体目录规范设计如图5.3所示。

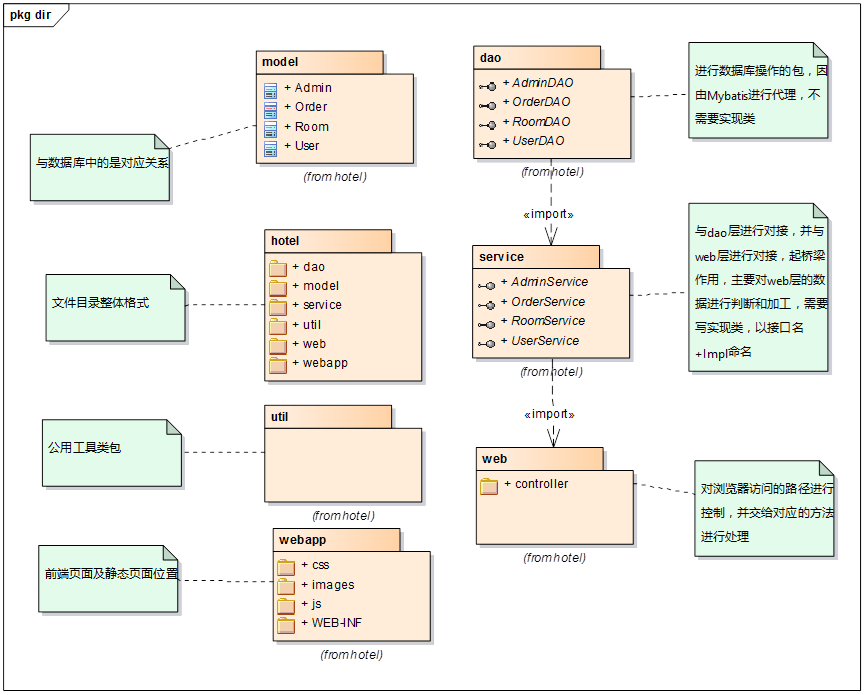


图 5.3

### 5.3.1 dao层接口设计

dao层用于与数据库进行对接，实现对数据库表中相应数据的增删改查，如图5.3.1。



图 5.3.1 dao层接口设计

### 5.3.2 service接口设计

service层主要负责对web层数据进进行判断和封装并将封装后的数据交由dao层处理并返回结果给service层再继续返回给web层。service层接口如图5.3.2所示。



图 5.3.2 service层接口设计

### 5.3.3 controller层接口设计

controller层主要负责对请求的地址进行分配和转发，并将service层处理的结果返回给客户端，这里主要指的是浏览器。controller层接口如图5.3.3所示。



图 5.3.3 controller接口设计

# 第6章 系统实现

## 6.1 系统开发及运行环境

（1）系统开发的硬件环境如表3.2 -A所示。

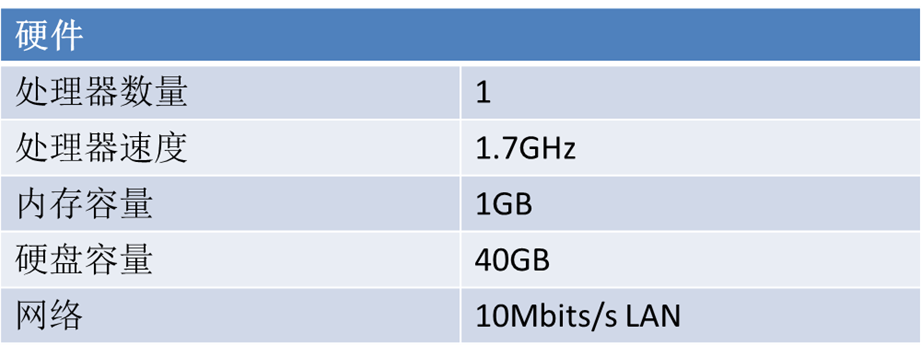


表 3.2-A

（2）软件开发所需的软件环境如表3.2-B所示。

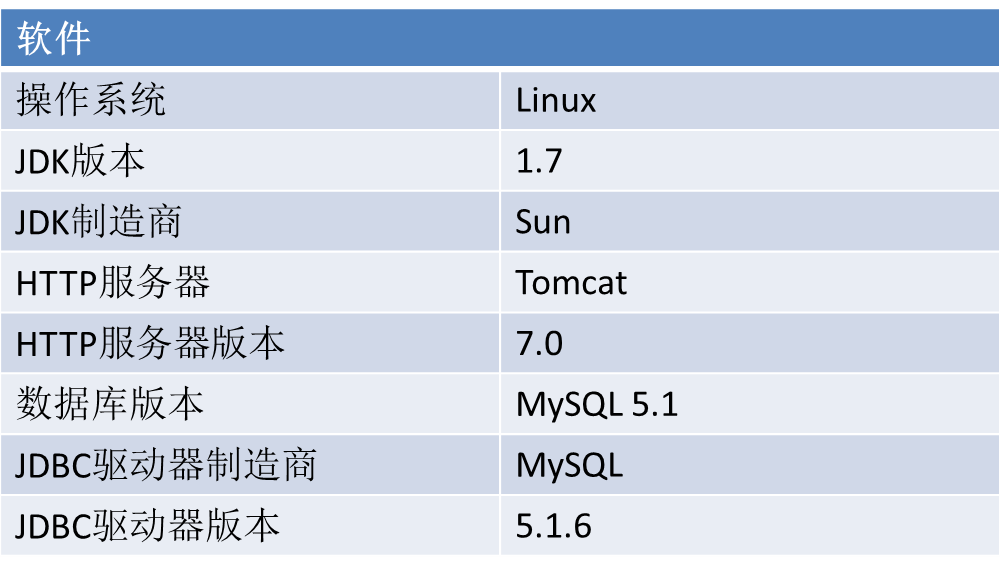


表3.2-B

## 6.2 数据库连接

数据库连接信息详情如图6.2所示。

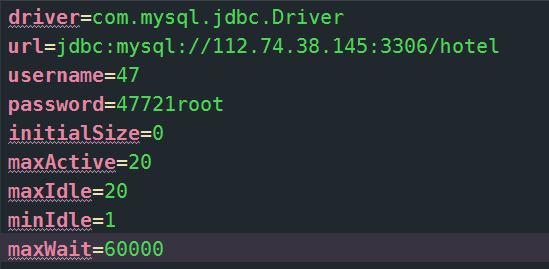


图 6.2

## 6.3 实现展示

### 6.3.1 会员登录界面实现

如图 6.3.1

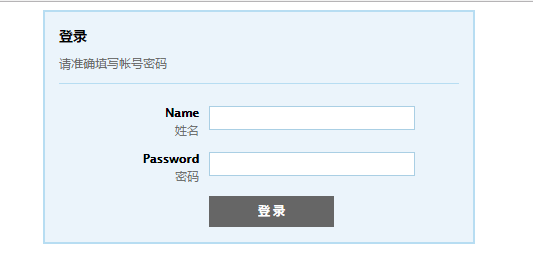
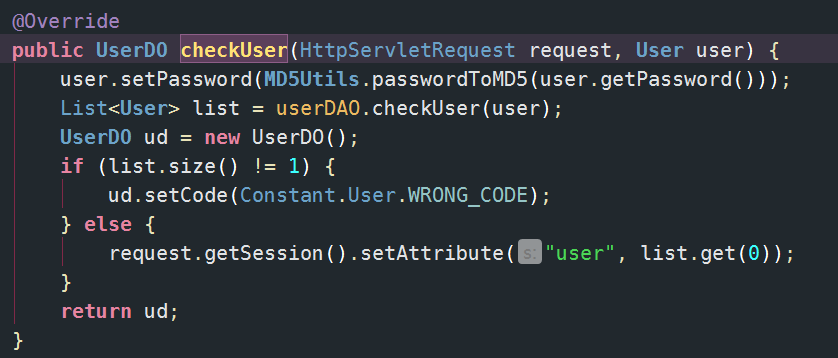


图6.3.1

密码验证代码如下。



### 6.3.2 个人中心界面实现

（1）个人中心——个人信息界面展示，如图6.3.2-(1)所示。

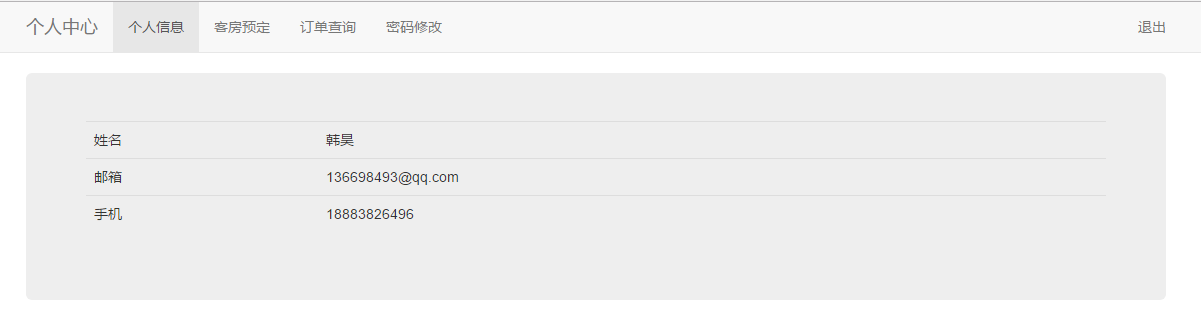
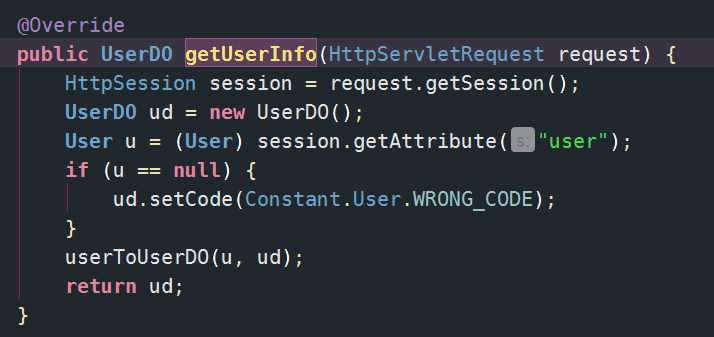


图6.3.2-(1)

查询个人信息代码如下：

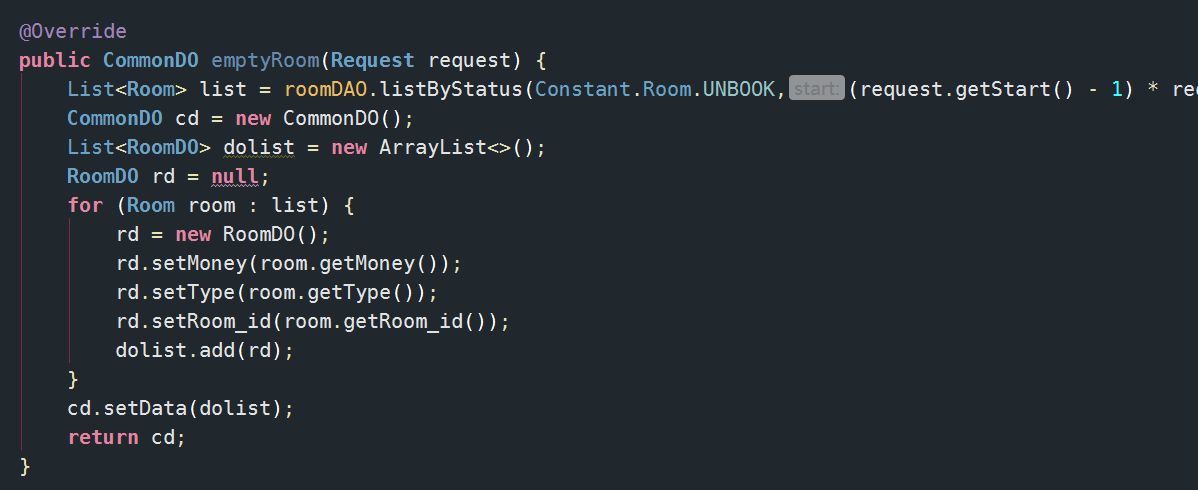


1. 客房预订界面展示如图6.3.2-(2)所示。



图 6.3.2-(2)

查询客房代码实现如下：

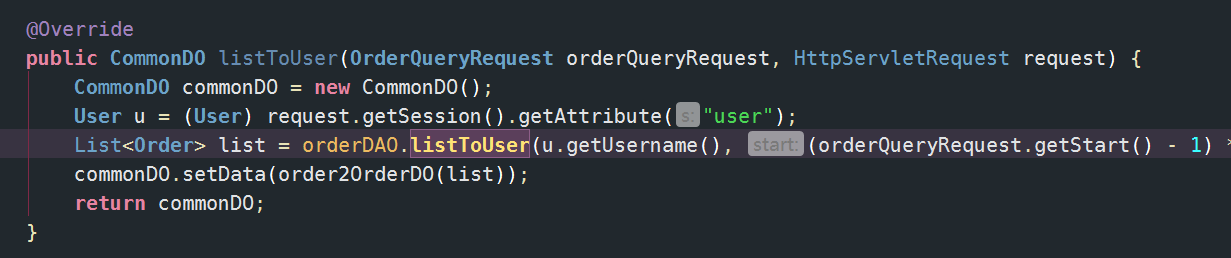


1. 订单查询界面展示如图6.3.2-(3)所示



图 6.3.2-(3)

订单查询代码如下：



1. 修改密码界面展示如图6.3.2-(4)所示。

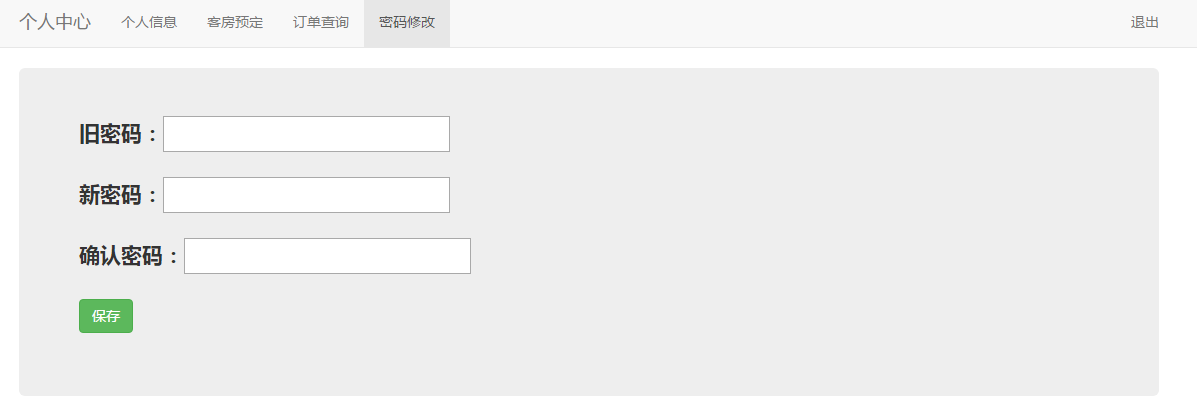
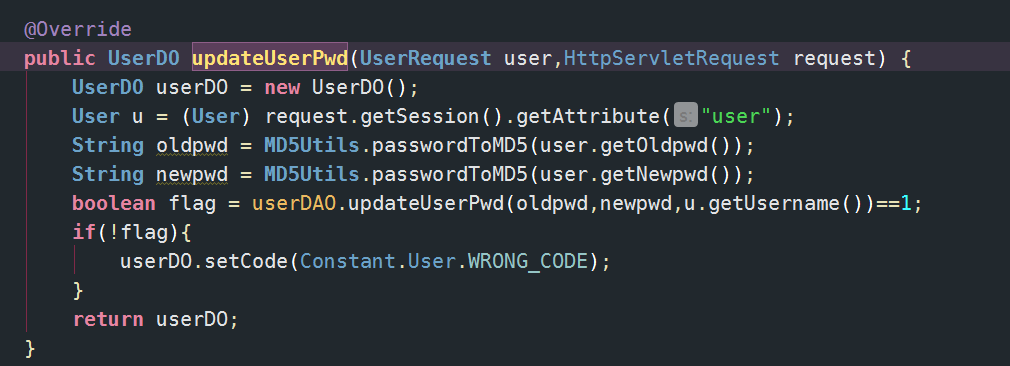


图 6.3.2-(4)

更改密码代码如下：



# 第7章 结束语

在这短短的几个月内，我能够顺利完成这次毕业设计，主要归功于以下方面：

(1) 利用在大学期间学过的软件工程专业知识，使用了包括擅长做服务端的编程语言Java和擅长对页面进行动态修改的脚本语言JavaScript。

(2) 参考各种相关书籍，网上查找相关资料信息。

(3) 指导老师张喜平喜喜老师的耐心指导。

毕业设计就要画上一个句号了，自己大学阶段的生活也将要过去了。自己也将走向职场，开始真正的大人生活。在这几个月里，我通过自己的学习和努力，通过老师的悉心帮助，我的专业技能也得到了长足的提升。毕业了，既是结束，也是新的开始。

# 参考文献

[1] 王宜贵.软件工程[M].北京机械工业出版社，2002：20-79

[2] 孙涌.现代软件工程[M].北京希望电子出版社，2003.8:1-246

[3] 朱丹丹.基于JAVAEE的毕业设计管理系统的设计研究[J].数字技术与应用，2015,12:146

[4] 张孝祥.深入Java Web开发内幕——核心基础[M].北京：电子工业出版社.2006

[5] 孙卫琴，李洪成.Tomcat与Java Web 开发技术详解[M].电子工业出版社,2003.6:1-205

[6] 徐建波，周新莲．Web设计原理于编程技术［M］．中南大学出版社，2005．185-193

[7] BruceEckel.Java编程思想[M].机械工业出版社，2003.10:1-378

[8]李倩.中小型酒店客房管理信息系统设计[J]. 电子测试, 2014(21):18-20.

[9] 刘亚宾,杨红.精通Eclipse[M].北京:电子工业出版,2005

[10] 李刚.基于J2EE的Ajax宝典[M].北京:电子工业出版社, 2007

[11] 唐汉明,翟振兴,关宝军,王洪权.深入浅出MySQL(第2版)[M]. 北京: 人民邮电出版社, 2014.

[12] 林信良. Spring2.0技术手册[M]. 北京：电子工业出版社，2005

[13] 徐雯;高建华;[基于Spring MVC及MyBatis的Web应用框架研究](http://www.cnki.com.cn/Article/CJFDTOTAL-WXDY201207000.htm)[J];微型电脑应用;2012年07期

[14] 叶雯;基于Spring MVC框架的Web登录模块的设计与实现[J];电脑知识与学术术;2013年35期

[15] (美)阿诺德，Ken Arnold，等. Java程序设计语言[M]. 北京：人民邮电出版社，2006

[16] 林上杰，林康司编著．JSP2.0技术手册［M］．电子工业出版社，2004．3-6

# 附录

1.英文原文

## Spring Framework

The Spring Framework is a lightweight solution and a potential one-stop-shop for building your enterprise-ready applications. However, Spring is modular, allowing you to use only those parts that you need, without having to bring in the rest. You can use the IoC container, with any web framework on top, but you can also use only the Hibernate integration code or the JDBC abstraction layer. The Spring Framework supports declarative transaction management, remote access to your logic through RMI or web services, and various options for persisting your data. It offers a full-featured MVC framework, and enables you to integrate AOP transparently into your software.

Spring is designed to be non-intrusive, meaning that your domain logic code generally has no dependencies on the framework itself. In your integration layer (such as the data access layer), some dependencies on the data access technology and the Spring libraries will exist. However, it should be easy to isolate these dependencies from the rest of your code base.

This document is a reference guide to Spring Framework features. If you have any requests, comments, or questions on this document, please post them on the user mailing list. Questions on the Framework itself should be asked on StackOverflow .

### Getting Started with Spring

This reference guide provides detailed information about the Spring Framework. It provides comprehensive documentation for all features, as well as some background about the underlying concepts (such as "Dependency Injection") that Spring has embraced.

If you are just getting started with Spring, you may want to begin using the Spring Framework by creating a Spring Boot based application. Spring Boot provides a quick (and opinionated) way to create a production-ready Spring based application. It is based on the Spring Framework, favors convention over configuration, and is designed to get you up and running as quickly as possible.

You can use start.spring.io to generate a basic project or follow one of the "Getting Started" guides like the Getting Started Building a RESTful Web Service one. As well as being easier to digest, these guides are very task focused, and most of them are based on Spring Boot. They also cover other projects from the Spring portfolio that you might want to consider when solving a particular problem.

### Introduction to the Spring Framework

The Spring Framework is a Java platform that provides comprehensive infrastructure support for developing Java applications. Spring handles the infrastructure so you can focus on your application.

Spring enables you to build applications from "plain old Java objects" (POJOs) and to apply enterprise services non-invasively to POJOs. This capability applies to the Java SE programming model and to full and partial Java EE.

Examples of how you, as an application developer, can benefit from the Spring platform:

* Make a Java method execute in a database transaction without having to deal with transaction APIs.
* Make a local Java method an HTTP endpoint without having to deal with the Servlet API.
* Make a local Java method a message handler without having to deal with the JMS API.
* Make a local Java method a management operation without having to deal with the JMX API.

#### Dependency Injection and Inversion of Control

A Java application — a loose term that runs the gamut from constrained, embedded applications to n-tier, server-side enterprise applications — typically consists of objects that collaborate to form the application proper. Thus the objects in an application have dependencies on each other.

Although the Java platform provides a wealth of application development functionality, it lacks the means to organize the basic building blocks into a coherent whole, leaving that task to architects and developers. Although you can use design patterns such as Factory, Abstract Factory, Builder, Decorator, and Service Locator to compose the various classes and object instances that make up an application, these patterns are simply that: best practices given a name, with a description of what the pattern does, where to apply it, the problems it addresses, and so forth. Patterns are formalized best practices that you must implement yourself in your application.

The Spring Framework Inversion of Control (IoC) component addresses this concern by providing a formalized means of composing disparate components into a fully working application ready for use. The Spring Framework codifies formalized design patterns as first-class objects that you can integrate into your own application(s). Numerous organizations and institutions use the Spring Framework in this manner to engineer robust, maintainable applications.

#### Core Container

The Core Container consists of the spring-core, spring-beans, spring-context, spring-context-support, and spring-expression (Spring Expression Language) modules.

The spring-core and spring-beans modules provide the fundamental parts of the framework, including the IoC and Dependency Injection features. The BeanFactory is a sophisticated implementation of the factory pattern. It removes the need for programmatic singletons and allows you to decouple the configuration and specification of dependencies from your actual program logic.

The Context (spring-context) module builds on the solid base provided by the Core and Beans modules: it is a means to access objects in a framework-style manner that is similar to a JNDI registry. The Context module inherits its features from the Beans module and adds support for internationalization (using, for example, resource bundles), event propagation, resource loading, and the transparent creation of contexts by, for example, a Servlet container. The Context module also supports Java EE features such as EJB, JMX, and basic remoting. The ApplicationContext interface is the focal point of the Context module. spring-context-support provides support for integrating common third-party libraries into a Spring application context for caching (EhCache, Guava, JCache), mailing (JavaMail), scheduling (CommonJ, Quartz) and template engines (FreeMarker, JasperReports, Velocity).

The spring-expression module provides a powerful Expression Language for querying and manipulating an object graph at runtime. It is an extension of the unified expression language (unified EL) as specified in the JSP 2.1 specification. The language supports setting and getting property values, property assignment, method invocation, accessing the content of arrays, collections and indexers, logical and arithmetic operators, named variables, and retrieval of objects by name from Spring’s IoC container. It also supports list projection and selection as well as common list aggregations.

#### AOP and Instrumentation

The spring-aop module provides an AOP Alliance-compliant aspect-oriented programming implementation allowing you to define, for example, method interceptors and pointcuts to cleanly decouple code that implements functionality that should be separated. Using source-level metadata functionality, you can also incorporate behavioral information into your code, in a manner similar to that of .NET attributes.

The separate spring-aspects module provides integration with AspectJ.

The spring-instrument module provides class instrumentation support and classloader implementations to be used in certain application servers. The spring-instrument-tomcat module contains Spring’s instrumentation agent for Tomcat.

#### Messaging

Spring Framework 4 includes a spring-messaging module with key abstractions from the Spring Integration project such as Message, MessageChannel, MessageHandler, and others to serve as a foundation for messaging-based applications. The module also includes a set of annotations for mapping messages to methods, similar to the Spring MVC annotation based programming model.

#### Data Access/Integration

The Data Access/Integration layer consists of the JDBC, ORM, OXM, JMS, and Transaction modules.

The spring-jdbc module provides a JDBC-abstraction layer that removes the need to do tedious JDBC coding and parsing of database-vendor specific error codes.

The spring-tx module supports programmatic and declarative transaction management for classes that implement special interfaces and for all your POJOs (Plain Old Java Objects).

The spring-orm module provides integration layers for popular object-relational mapping APIs, including JPA, JDO, and Hibernate. Using the spring-orm module you can use all of these O/R-mapping frameworks in combination with all of the other features Spring offers, such as the simple declarative transaction management feature mentioned previously.

The spring-oxm module provides an abstraction layer that supports Object/XML mapping implementations such as JAXB, Castor, XMLBeans, JiBX and XStream.

The spring-jms module (Java Messaging Service) contains features for producing and consuming messages. Since Spring Framework 4.1, it provides integration with the spring-messaging module.

#### Web

The Web layer consists of the spring-web, spring-webmvc, spring-websocket, and spring-webmvc-portlet modules.

The spring-web module provides basic web-oriented integration features such as multipart file upload functionality and the initialization of the IoC container using Servlet listeners and a web-oriented application context. It also contains an HTTP client and the web-related parts of Spring’s remoting support.

The spring-webmvc module (also known as the Web-Servlet module) contains Spring’s model-view-controller (MVC) and REST Web Services implementation for web applications. Spring’s MVC framework provides a clean separation between domain model code and web forms and integrates with all of the other features of the Spring Framework.

The spring-webmvc-portlet module (also known as the Web-Portlet module) provides the MVC implementation to be used in a Portlet environment and mirrors the functionality of the Servlet-based spring-webmvc module.

#### Test

The spring-test module supports the unit testing and integration testing of Spring components with JUnit or TestNG. It provides consistent loading of Spring ApplicationContexts and caching of those contexts. It also provides mock objects that you can use to test your code in isolation.

（2）英文翻译

## Spring 框架

Spring框架是一种轻量级解决方案，是构建您的企业应用程序的潜在一站式服务。

然而，Spring是模块化的，允许您只使用您需要的那些部分，而不需要引入其他部分。您可以使用IoC容器，上面有任何web框架，但是您也可以只使用Hibernate集成代码或JDBC抽象层。

Spring框架支持声明式事务管理、通过RMI或web服务远程访问您的逻辑，以及用于持久化数据的各种选项。它提供了一个全功能的MVC框架，并允许您将AOP透明地集成到您的软件中。

Spring的设计是非侵入性的，这意味着您的域逻辑代码通常对框架本身没有依赖性。在您的集成层(例如数据访问层)中，一些对数据访问技术和Spring库的依赖将会存在。但是，应该很容易将这些依赖项与您的代码库的其余部分隔离开来。

本文档是Spring框架特性的参考指南。如果您有任何关于此文件的请求、评论或问题，请将它们贴在用户邮件列表上。关于框架本身的问题应该在StackOverflow上被问到。

### 开始使用Spring

这个参考指南提供了关于Spring框架的详细信息。它为所有特性提供了全面的文档，以及Spring所支持的基本概念(如“依赖注入”)的一些背景知识。如果您刚刚开始使用Spring，那么您可能想要开始使用Spring框架，创建一个基于Spring的基于Spring的应用程序。

Spring引导提供了一种快速(且有主见的)方式来创建基于产品的基于Spring的应用程序。它基于Spring框架，支持约定优于配置，并且旨在让您尽可能快地启动和运行。

您可以使用start.spring.io生成一个基本的项目，或者遵循“开始”的指导，就像开始构建一个RESTful Web服务一样。除了更容易消化，这些指南都是非常专注于任务的，而且大多数都是基于Spring引导的。它们还涵盖了在解决特定问题时可能需要考虑的Spring投资组合中的其他项目。

#### 依赖注入和控制反转

Java应用程序是一个松散的术语，它从受约束的、嵌入式的应用程序到n层的、服务器端企业应用程序，通常由协作形成应用程序的对象组成。

因此，应用程序中的对象彼此依赖。

尽管Java平台提供了大量的应用程序开发功能，但是它缺乏将基本的构建块组织成一个一致的整体的方法，将该任务留给架构师和开发人员。

虽然可以使用设计模式,例如工厂,抽象工厂,建筑,装饰,和服务定位器组成的各种类和对象实例应用程序组成,这些模式是:最佳实践给出一个名字,有什么模式的描述,如何应用它,它解决的问题,等等。

模式是您必须在应用程序中实现的形式化的最佳实践。

Spring框架反转(IoC)组件解决了这个问题，它提供了一种形式化的方法，将完全不同的组件组合到一个可以使用的完全工作的应用程序中。

Spring框架将形式化的设计模式作为您可以集成到您自己的应用程序(s)的一级对象。

许多组织和机构以这种方式使用Spring框架来设计健壮的、可维护的应用程序。

#### 核心容器

核心容器由Spring核心、Spring bean、Spring上下文、Spring上下文支持和Spring表达式(Spring表达式语言)模块组成。

spring核心和spring bean模块提供了框架的基本部分，包括IoC和依赖注入特性。

BeanFactory是工厂模式的一个成熟的实现。它消除了对编程单例的需求，并允许您将依赖项的配置和规范从实际的程序逻辑中分离。

上下文(spring上下文)模块构建在核心和bean模块提供的坚实基础之上:它是一种以框架方式访问对象的方法，类似于JNDI注册表。

上下文模块从bean模块继承其特性，并添加对国际化的支持(例如，使用资源包)、事件传播、资源加载和上下文透明的创建，例如，Servlet容器。

上下文模块还支持Java EE特性，如EJB、JMX和基本远程控制。

ApplicationContext接口是上下文模块的焦点。

Spring上下文支持提供了对将常用的第三方库集成到Spring应用程序上下文中的支持，用于缓存(EhCache、Guava、JCache)、邮件(JavaMail)、调度(CommonJ、Quartz)和模板引擎(FreeMarker、JasperReports、Velocity)。spring表达式模块提供了一种强大的表达式语言，用于在运行时查询和操作对象图。它是JSP 2.1规范中指定的统一表达式语言(统一EL)的扩展。该语言支持设置和获取属性值、属性赋值、方法调用、访问数组的内容、集合和索引器、逻辑和算术运算符、命名变量以及从Spring的IoC容器中检索对象。它还支持列表投影和选择以及常见的列表聚合。

#### 面向切面编程

spring AOP模块提供了一种面向AOP的面向方面的编程实现，允许您定义方法拦截器和切入点，以清晰地分离实现应该分离的功能的代码。

使用源代码级别的元数据功能，您还可以将行为信息合并到您的代码中，以类似于这样的方式。

网络属性。独立的spring aspect模块提供了与AspectJ的集成。

spring仪表模块提供类的插装支持和类加载器实现，用于特定的应用程序服务器。Spring工具-Tomcat模块包含用于Tomcat的Spring的插装代理。

#### 消息传递

Spring Framework 4 includes a spring-messaging module with key abstractions from the Spring Integration project such as Message, MessageChannel, MessageHandler, and others to serve as a foundation for messaging-based applications. The module also includes a set of annotations for mapping messages to methods, similar to the Spring MVC annotation based programming model.

#### 数据访问和集成

数据访问/集成层由JDBC、ORM、OXM、JMS和事务模块组成。

spring JDBC模块提供了一个JDBC抽象层，消除了对数据库供应商特定的错误代码进行冗长的JDBC编码和解析的需要。

spring tx模块支持对实现特殊接口和所有pojo(普通Java对象)的类进行编程和声明性事务管理。

spring orm模块为流行的对象-关系映射api提供了集成层，包括JPA、JDO和Hibernate。使用Spring orm模块，您可以将所有这些o/r映射框架与Spring提供的所有其他特性结合使用，例如前面提到的简单的声明性事务管理特性。

spring-oxm模块提供了一个抽象层，支持对象/xml映射实现，如JAXB、Castor、XMLBeans、JiBX和XStream。

spring-jms模块(Java消息传递服务)包含用于生成和使用消息的功能。

自Spring Framework 4.1以来，它提供了与Spring消息传递模块的集成。

#### 网络

Web层由spring-Web、spring-webmvc、spring-websocket和spring-webmvc-portlet模块组成。spring web模块提供了基本的面向web的集成特性，例如多部件文件上传功能，以及使用Servlet侦听器和面向web的应用程序上下文的IoC容器的初始化。它还包含一个HTTP客户端和Spring的远程支持的与web相关的部分。

Spring-webmvc模块(也称为Web servlet模块)包含Spring的模型-视图-控制器(MVC)和Web应用程序的REST Web服务实现。

Spring的MVC框架提供了域模型代码和web forms之间的清晰分离，并与Spring框架的其他所有特性进行了集成。

spring-webmvc-Portlet模块(也称为web Portlet模块)提供了在Portlet环境中使用的MVC实现，并反映了基于servlet的spring-webmvc模块的功能。

#### 测试

Spring测试模块支持使用JUnit或TestNG的Spring组件的单元测试和集成测试。它提供了Spring Applicationcontext和这些上下文缓存的一致加载。它还提供了模拟对象，您可以使用这些对象来单独测试您的代码。