

编 号：2013214027

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 副教授
答 辩 组 负 责 人	熊仕勇 高级工程师

二〇一七 年 六 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《酒店预订系统的设计与实现》，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 2013 级

专业 软件工程

班级 1301309

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。本系统主要实现了客户预订客房，查询自己的订单以及管理员对客房进行管理，例如：增加客房，修改客房信息，删除客房，更改客房状态，管理员对订单进行查询，按照时间范围查询、按照客户身份证号查询、按照订单号查询，管理员对会员信息的管理，查看和删除操作。

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

In all industries, management plays a very important role. What's more, it is the priority for all in hotel industry. The level of hotel management determines the survival and development of a hotel. With the rapid development of internet industry and traffic industry, the hotel industry also keeps pace of the times, meanwhile, the hotel management style also requires transformation: Traditional management models to modernization management methods. It now appears that rapid and efficient whole network and information management have become the necessity.

In order to reduce the management cost and improve the efficiency of management, hotels need to choose an informationalized, intellectualized and stable online hotel reservation system. Therefore, the hotel reservation system has become the essential element of management and development of modern hotels. This system realizes room reservations of guests, order inquiries and room management of administrators, such as adding rooms, modifying room information, deleting guest rooms, changing room status, and for administrators, it can help them track the order by a time range, ID number and order number, furthermore, administrators can manage information of members to view or delete theirs.

Foreground design of the system uses HTML as development language, the background uses Java language for logic judgment which integrates the SSM framework and uses the MySQL database for data persistence management. In development project, this system uses Eclipse as the integrated development tools, Maven as the project management tool, Git as the version control tool, Tomcat as the middleware of data transfer, Linux as the application platform of its server to design a hotel reservation system based on Web and B/S structure.

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

目录

第 1 章 引言	1
1.1 研究背景	1
1.2 国内外现状	1
1.2.1 国外研究现状	1
1.2.2 国内研究现状	2
1.3 论文研究的主要内容	2
第 2 章 关键技术介绍	3
2.1 Java	3
2.2 JavaScript.....	3
2.3 Html	3
2.4 三层架构模式	4
2.5 MyBatis 框架.....	5
2.6 Spring 框架.....	6
第 3 章 系统分析	9
3.1 可行性分析	9
3.1.1 技术可行性	9
3.1.2 经济可行性	9
3.1.3 市场可行性	9
3.2 系统分析	10
3.2.1 系统功能概述	10
3.2.2 功能分析	10
3.3 系统用例分析	11
3.3.1 会员用例图	11
3.3.2 管理员用例图	12
第 4 章 系统设计	15
4.1 系统概要设计	15
4.1.1 系统功能模块设计	15

4.1.2 整体流程设计	16
4.1.3 系统功能流程图	18
4.2 系统详细设计	23
4.2.1 系统架构设计	23
4.2.2 系统设计模型时序图	24
4.2.3 数据库设计	26
4.2.4 界面设计	30
4.2.5 系统接口设计	32
第 5 章 系统实现及测试	37
5.1 系统开发及运行环境	37
5.2 数据库连接	37
5.3 实现部分展示	38
5.3.1 会员登录界面实现	38
5.3.2 个人中心界面实现	39
5.3.3 后台管理中心界面实现	42
5.4 系统测试	43
5.4.1 测试目的和意义	43
5.4.2 测试方法	44
5.4.3 测试用例	44
第 6 章 总结与展望	45
6.1 个人工作总结	45
6.2 后续研究工作展望	45
参考文献	47
致谢	49
附录	51
一、英文原文	51
二、英文翻译	56

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，给人们的生活带来了很多好处，也在改变着人们的生活方式。大量研究表明，网络已经成为人们获取旅游信息的主要来源。而且，随着智能机、平板成为大众化的产品，人们可以很方便地在网上搜索旅游信息、选择购买自己喜欢的旅游产品和服务，这极大的提高了旅游者的体验。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

1.2.1 国外研究现状

就日本经济新闻社对东京、大阪的主要宾馆酒店进行调查显示，日本酒店业对网络服务的依存度大大提高。在日本经济萎靡的背景下，日本酒店业面临的竞争变得越来越激烈，酒店不得不进行打折促销，而更实惠的网上预订也受到了越来越多顾客的青睐。

在美国，绝大多数网民会通过互联网预订的方式进行酒店预订。在线订房方式的产生，无疑会给消费者和企业带来最实惠的利益。

在国外，酒店预订业作为旅游业的一个分支，已经发展到了相对成熟的阶段。

1.2.2 国内研究现状

在上个世纪 80 年代初，是国内的酒店计算机预订管理系统的起点。从事该方面工作的有清华大学自动化系的金国芬教授、西安交大和浙江省计算机研究所。到了 80 年代中后期，随着国外酒店计算机系统和先进管理技术的大规模引进，进一步促进了我国酒店预订管理技术的发展。国内酒店预订系统正是在充分吸收国外预订管理系统的精华，结合国内实际的情况下逐步发展成熟，到了 90 年代初期形成了几个较为成熟的软件系统了。

到了 90 年代中期，随着计算机在酒店中的普及应用，以及计算机技术的不断发展，酒店计算机系统的发展到了一个新的实际，新的系统平台、新的软件功能、新的系统特点及发展方向不断涌现。

如果用动态的发展的眼光来看，目前该行业的竞争只是刚刚开始，资本的投入程度还不算多，但相信随着时间的退役，可能会持续性地涌入各路资本，竞争会逐渐显现出来。而且随着竞争程度的加剧，这个行业会得以发展和繁荣。

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 章为引言，主要描述了论文研究的背景和国内外现状；

第 2 章为关键技术介绍，主要描述了开发时使用的技术、分层的设计思想；

第 3 章为系统分析，可行性分析和需求分析；

第 4 章为系统设计，概要设计和详细设计；

第 5 章为系统实现及测试，系统开发及运行环境，各个页面的实现结果进行展示及部分系统测试用例；

第 6 章为总结与展望，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 Java

Java 是一种跨平台的纯面向对象的编程语言，具有较高的安全性和较好的性能。除此之外，Java 并没有 C++ 中的多继承和指针的概念，所以简单易用。Java 包含的特点有面向对象、平台独立、分布式、简单性、安全性、动态性、可移植性、多线程、健壮性等^[1,2]。

Java 是一门强类型的面向对象编程语言^[3]，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。

2.2 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。

JavaScript 脚本语言同其他语言一样，有它自身的基本数据类型，表达式和算术运算符及程序的基本程序框架。JavaScript 提供了四种基本的数据类型和两种特殊数据类型用来处理数据和文字。而变量提供存放信息的地方，表达式则可以完成较复杂的信息处理^[4]。

2.3 Html

Html（超级文本标记语言）是标准通用标记语言下的一个应用。它是一种规范，也是一种标准，它通过标记符号来标记要显示的网页中的各个部分^[5]。网页文件本身是一种文本文件，通过在文本文件中添加标记符，可以告诉浏览器如何显示其中的内容（如：文字如何处理，画面如何安排，图片如何显示等）。浏览器

按顺序阅读网页文件，然后根据标记符解释和显示其标记的内容，对书写出错的标记将不指出其错误，且不停止其解释执行过程，编制者只能通过显示效果来分析出错原因和出错部位。但需要注意的是，对于不同的浏览器，对同一标记符可能会有不完全相同的解释，因而可能会有不同的显示效果。

2.4 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：界面层、业务逻辑层和数据访问层^[6,13]。界面层是主要表示WEB方式，如果逻辑层相当强大和完善，无论表现层如何定义和更改，逻辑层都能完善地提供服务。业务逻辑层主要是针对具体的问题的操作，也可以理解成对数据层的操作，对数据业务逻辑处理，如果说数据层是积木，那逻辑层就是对这些积木的搭建。数据访问层主要是对非原始数据（数据库或者文本文件等存放数据的形式）的操作层，而不是指原始数据，也就是说，是对数据库的操作，而不是数据，具体为业务逻辑层或表示层提供数据服务。

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^[6]。

分层模式如图 2.1 所示：

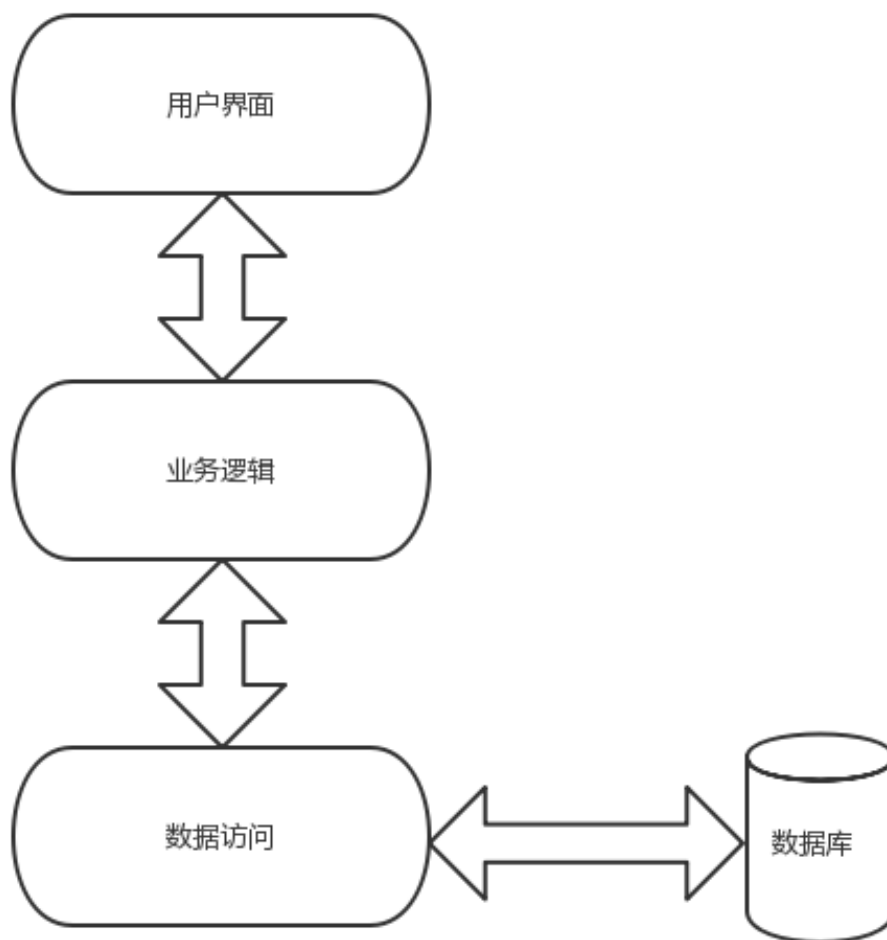


图 2.1 三层架构模式

2.5 MyBatis 框架

MyBatis 框架不是一个典型的 ORM^[7]。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 可以使用简单的 XML 文件或注解去配置 MyBatis 参数，将 Java 的 POJOs（Plain Old Java Objects，普通的 Java 对象）和接口映射成数据库中对应的记录。MyBatis 不仅实现领域层的对象范例和数据源层的关系范例之间的信息交互，且使用 XML 配置文件将访问数据层的数据库相关配置及 SQL 语言和应用层的程

序代码分离，所开发的程序在很大程度上提高了可移植性、可扩展性和可维护性^[8]。
MyBatis 的核心接口一共有 5 个，如图 2.2 所示：

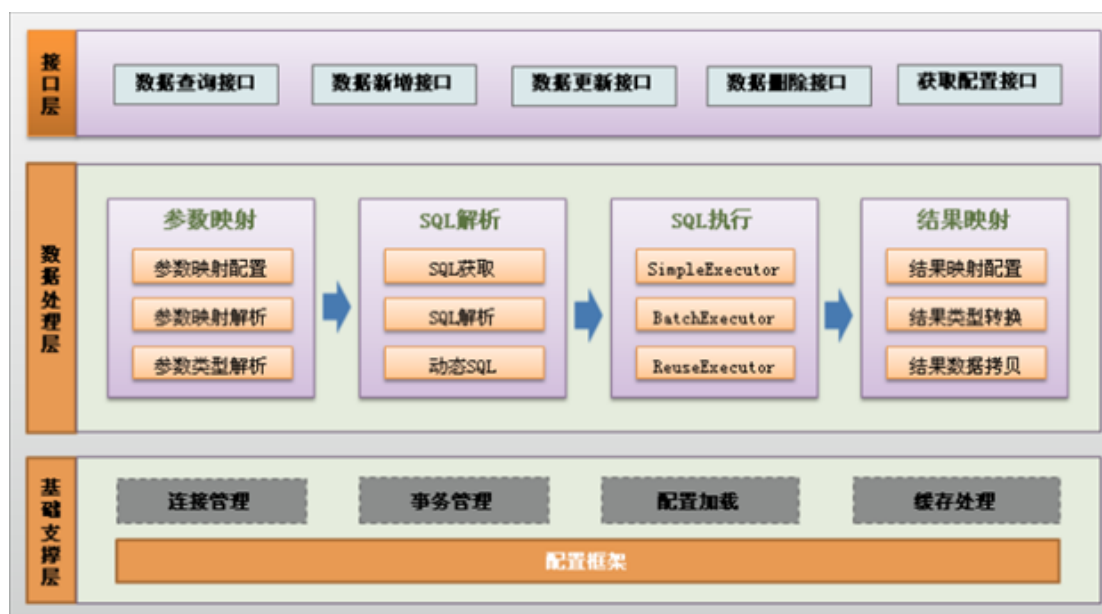


图 2.2 MyBatis 功能架构设计

2.6 Spring 框架

在 Spring 整体架构中，大致可以将其划分为几个层次，比如驱动组件、封装的 Java EE 服务、AOP 核心模块、IOC 容器和上层应用。Spring 体系中最为核心的的是 IoC 容器和 AOP 模块，通过 AOP 模块以非侵入和动态的方式来增强系统服务的功能，通过 IoC 容器管理 POJO 对象和它们之间的耦合关系，使应用开发资源可以通过简单的 POJO 对象来抽象和描述^[12]。

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。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，如图 2.3 所示。

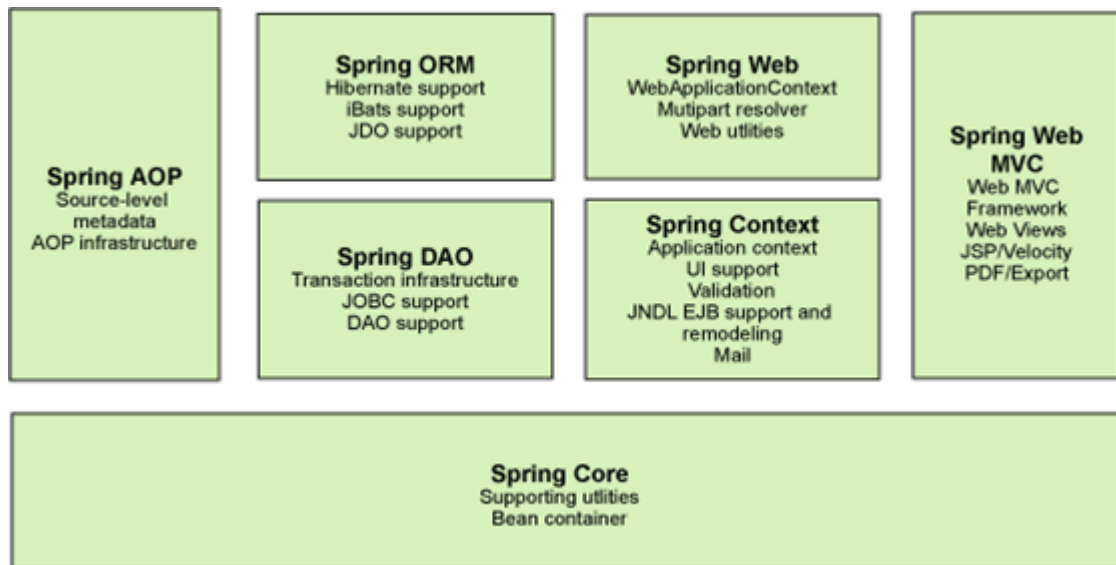


图 2.3 Spring 框架

第3章 系统分析

3.1 可行性分析

3.1.1 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。

3.1.2 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，且服务器租用的费用是很低的。

3.1.3 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。

3.2 系统分析

3.2.1 系统功能概述

本课题旨在设计和实现作用于国内小型酒店预订系统。方便为客户提供便捷的预订方式和优质的服务的同时，避免人工管理的一些弊端，提高酒店管理的效率。传统的手工记账存在着验证的效率和安全问题，使酒店管理混乱，客户预订客房手续繁琐，并不能很好的利用客房资源。

经过对相关资料和文献的查阅以及对现有酒店预订系统的分析，一个较为完备的酒店预订系统应该包含客房预订、订单查询、订单记录、用户信息管理和客房信息管理等基本功能。系统需要会员和管理员操作，会员注册并登录系统，完成客房预订，进入酒店，支付费用；管理员登录系统，管理客房和会员信息。

3.2.2 功能分析

通过对本课题的进一步研究和实际情况的调研，确定该系统面向会员（普通用户）和管理员两种角色。酒店预订系统的主要功能如下：

1) 对于管理员，该系统应该提供以下功能：

- (1) 对会员基本信息的查看，以及对会员账号的注销。
- (2) 对客房信息的修改操作和客房状态的更改操作。
- (3) 对订单按时间、订单号、身份证件号的查询。
- (4) 修改自己的登录密码。
- (5) 登录、退出系统。

2) 对于会员，该系统应该提供以下功能：

- (1) 对个人基本信息的查看。
- (2) 对可预订客房基本信息的展示及对客房的预订。
- (3) 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。
- (5) 注册、登录和退出系统。

3.3 系统用例分析

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。

3.3.1 会员用例图

在此用例图中，会员可以进行登录、注册、退出、查看订单、修改密码、查看个人信息和预订客房的操作。如图 3.1 所示：

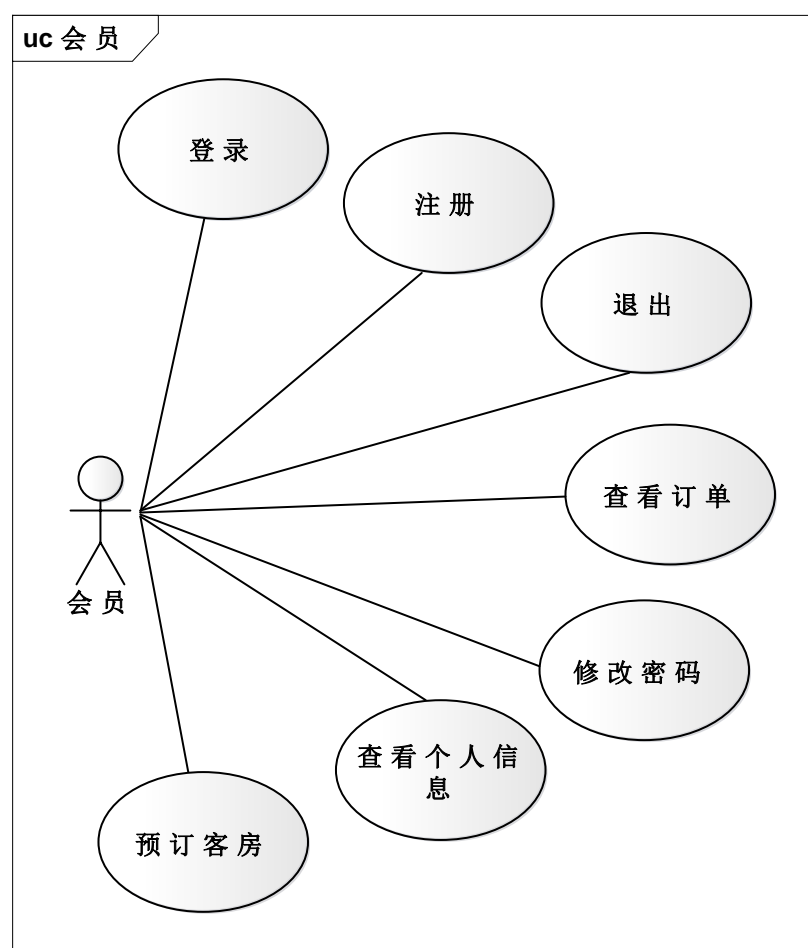


图 3.1 会员用例图

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账号和密码进行操作。会员可以输入注册时填写的账号和密码登录到个人中心页面。

4) 修改密码

已登录的会员可以点击密码修改跳转到密码修改界面，进行密码修改操作。

5) 退出

已登录的会员可以点击退出按钮登出账户，并会跳转到登录界面。

6) 预订客房

已登录会员可以点击客房预订跳转到客房预订界面，选择要预订的客房进行预订。

7) 查看订单

已登录的会员可以点击订单查询跳转到订单查询界面，选择一定的时间范围，查看此段时间内的个人订单，默认订单按照时间倒序展示。

8) 查看个人信息

已登录的会员可以点击个人信息跳转到个人信息界面，此界面会直接显示个人的基本信息。

3.3.2 管理员用例图

在此用例图中，管理员可以进行登录、退出、查询订单、管理客房、查看会员信息和删除会员账号的操作。如图 3.2 所示：

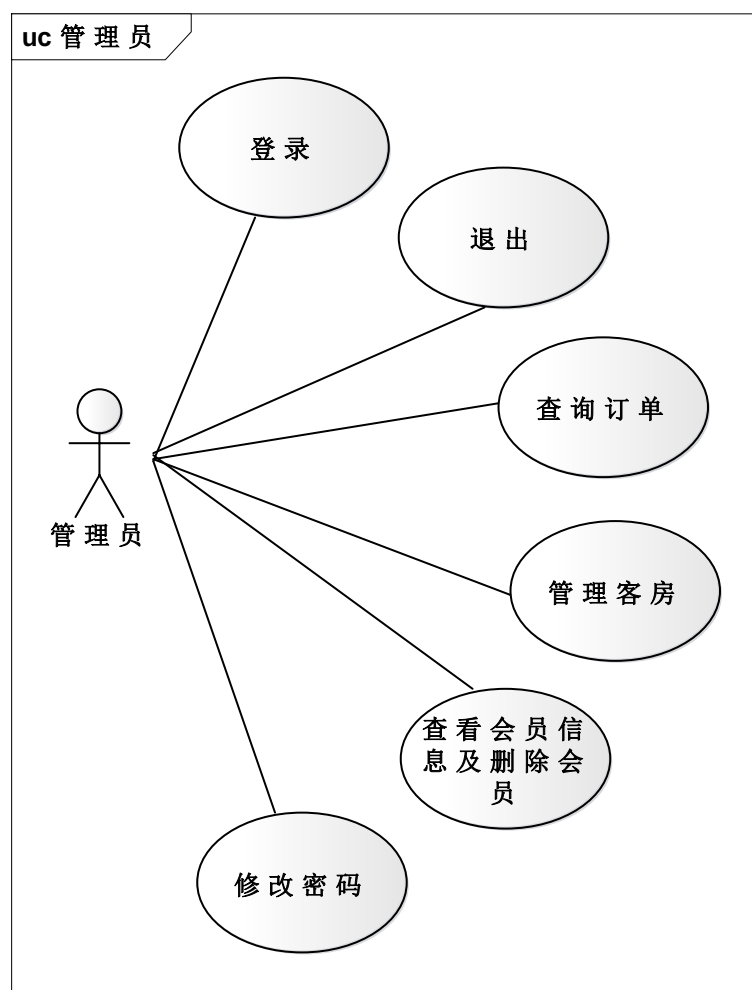


图 3.2 管理员用例图

1) 管理员

酒店内部管理人员，并通过内部特殊账号登录到后台管理中心的使用者。

2) 登录

在管理员登录界面，管理员输入内部账号和密码即可登录到后台管理中心。

3) 退出

管理员退出时需要执行的操作，退出当前账号，并会跳转到管理员登录界面。

4) 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账号。

5) 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。

6) 管理客房

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。

7) 修改密码

已登录的管理员可以点击密码修改跳转到密码修改界面，可以进行密码修改操作。

第 4 章 系统设计

4.1 系统概要设计

4.1.1 系统功能模块设计

依据对系统的需求分析，系统要实现两大功能模块：会员的功能模块、管理员的功能模块。具体的功能如图 4.1 和 4.2 所示：

1) 会员功能模块

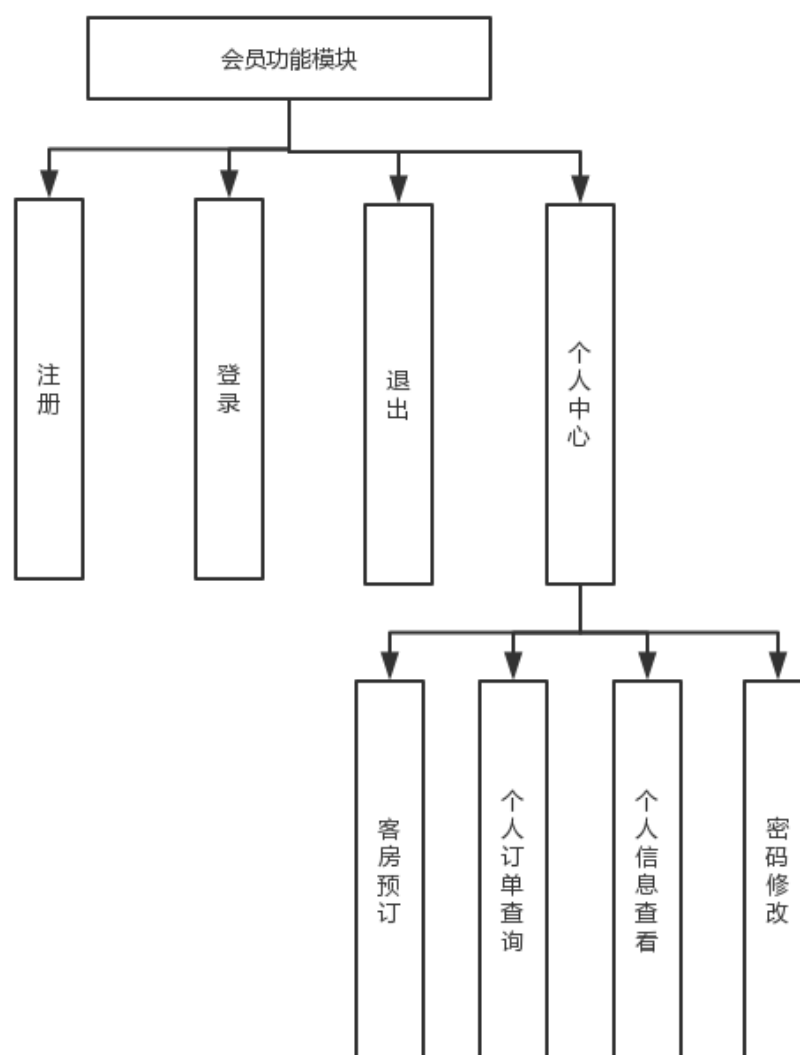


图 4.1 会员功能模块

2) 管理员功能模块

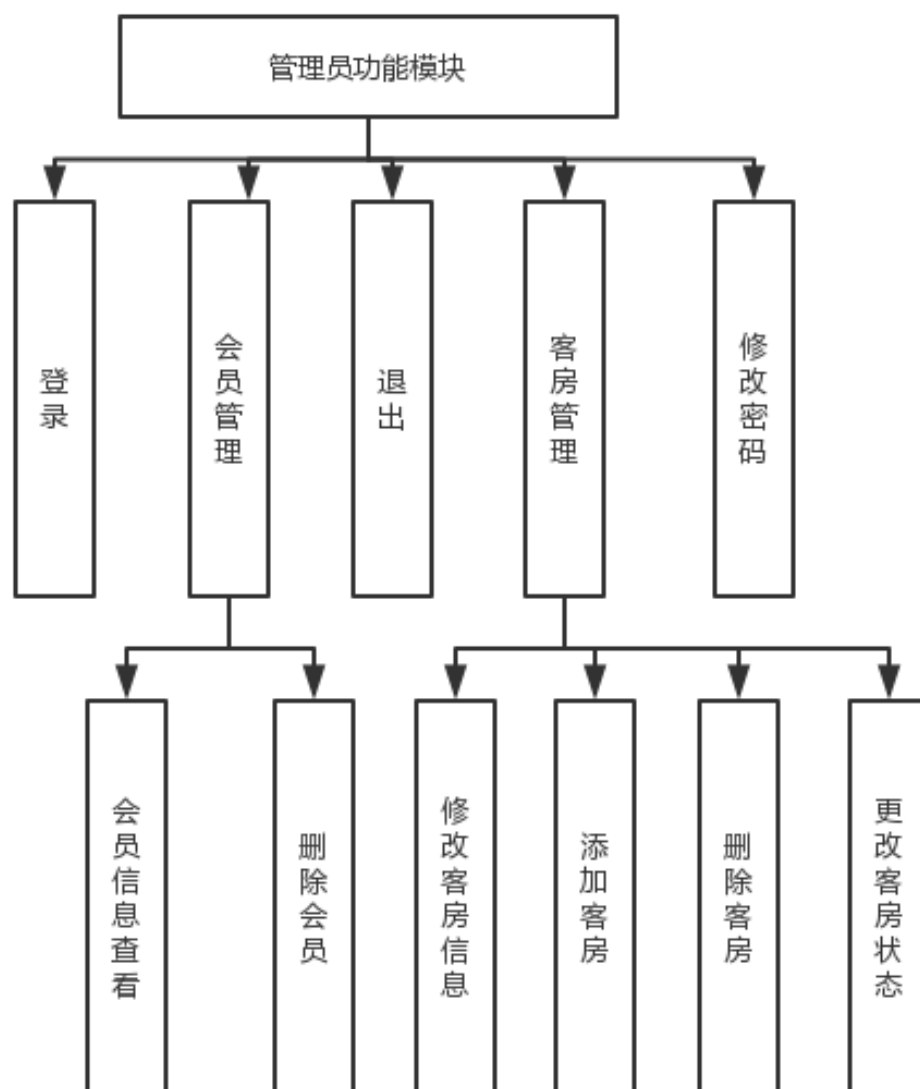


图 4.2 管理员功能模块

4.1.2 整体流程设计

1) 会员操作整体流程图

会员输入账号密码进行登录，验证无误后跳转到个人中心界面，会员主要进行的操作是查看订单和预订客房，点击退出，则退出个人中心界面，如图 4.3 所示：

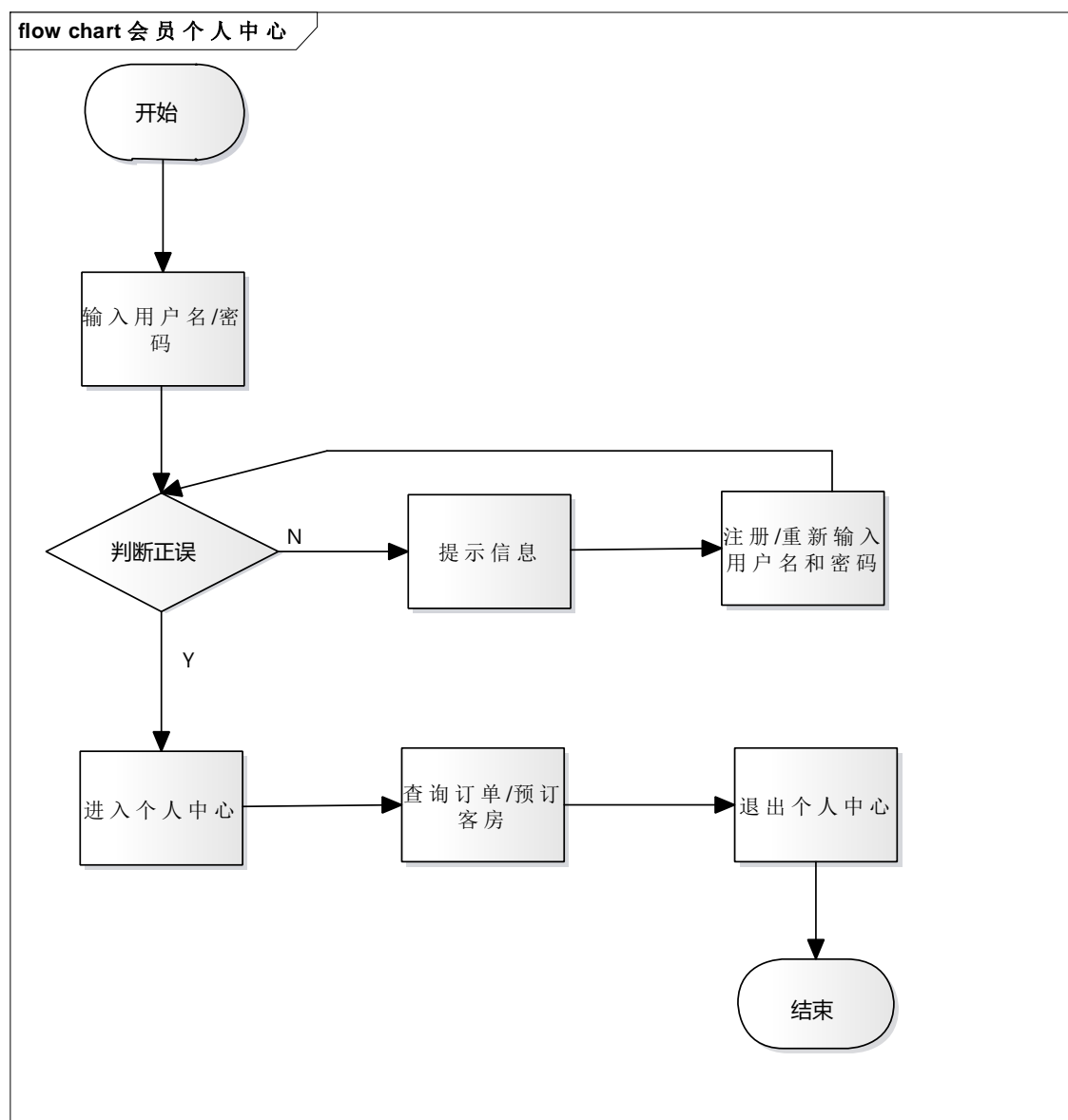


图 4.3 会员中心操作流程

2) 管理员操作流程

管理员输入账号密码且正确无误后进入后台管理中心，管理员主要的操作是查询订单和管理客房，操作结束后点击退出便可以退出后台管理中心。如图 4.4 所示：

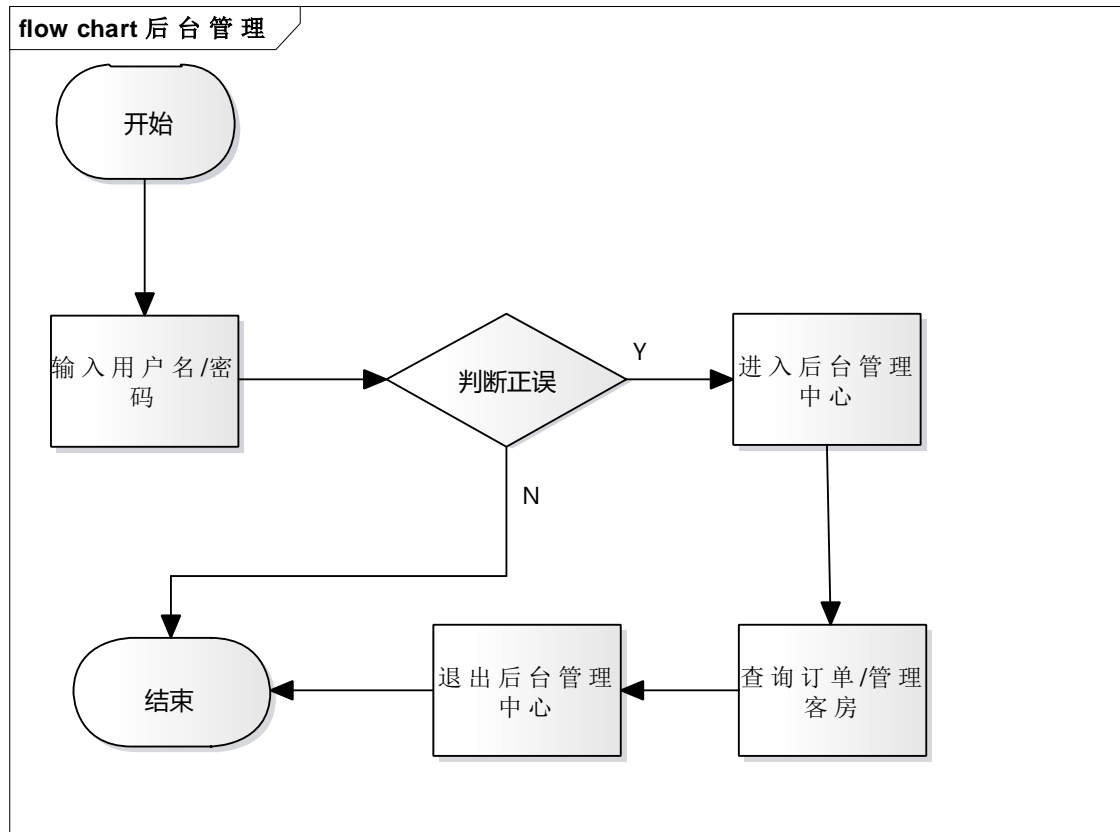


图 4.4 后台管理中心操作流程

4.1.3 系统功能流程图

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 4.5 所示：

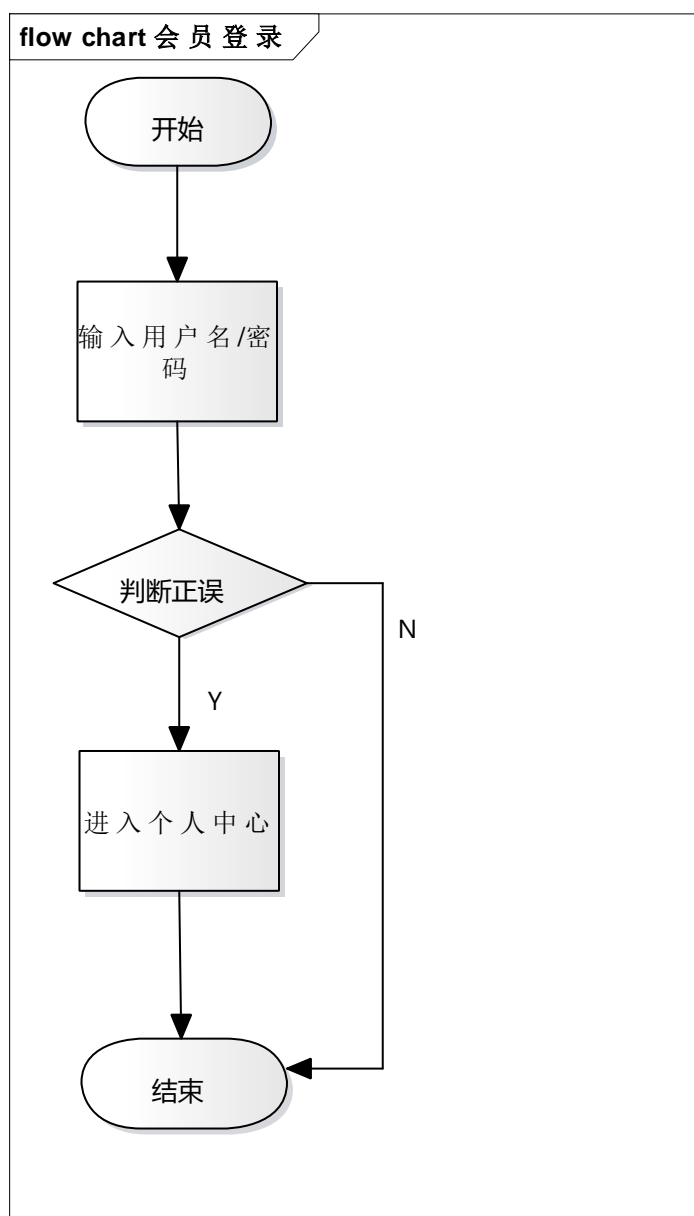


图 4.5 会员登录流程图

2) 会员预订客房流程图

已经登录的会员可以通过点击客房预订进入客房预订界面。选择好自己想要预订的客房后，点击预订，弹出选择框，是否确认预订。若确认预订，则会继续判断此客房是否已经被其他会员预订，若已经被预订，弹出提示框，让会员重新预订客房。若预订成功，则提示预订成功信息。整个流程图如图 4.6 所示：

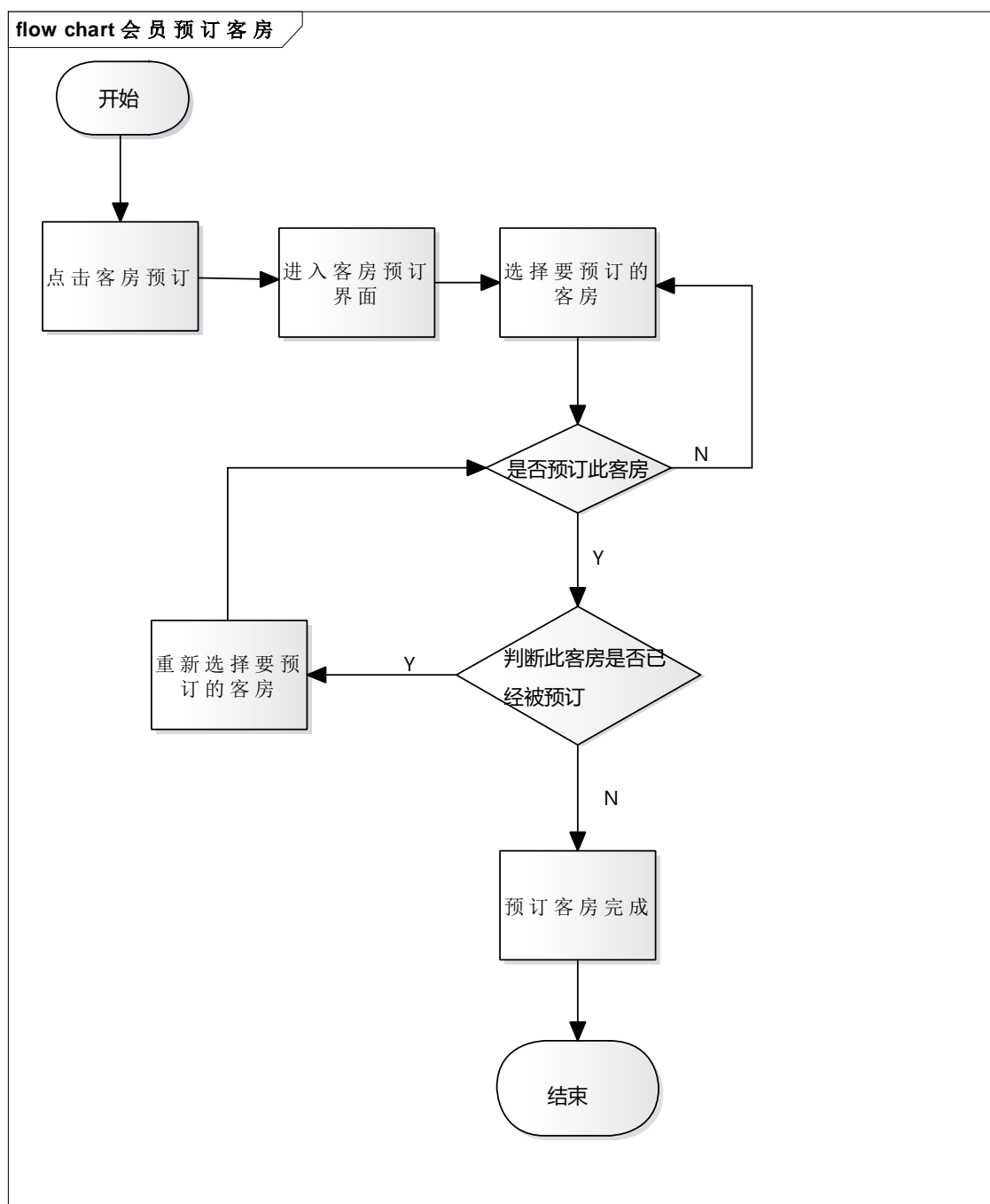


图 4.6 会员预订客房流程图

3) 会员查看个人订单流程图

已登录的会员点击订单查询，则会跳转到订单查询界面，并默认显示 10 条按时间倒序排序的订单（不足 10 条则有几条显示几条）。可通过选择要查询的时间范围，显示订单信息。如图 4.7 所示：

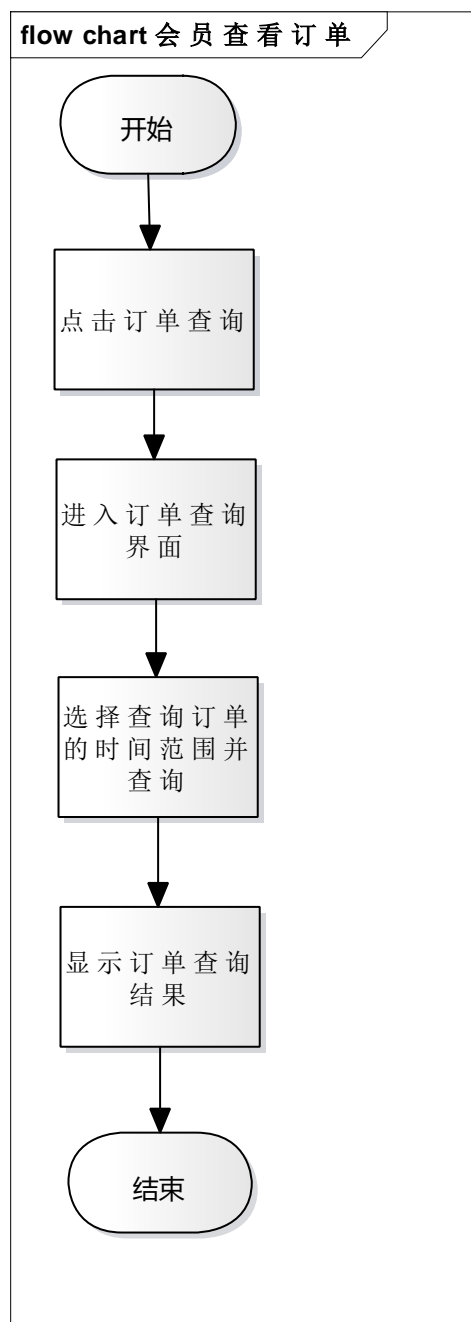


图 4.7 会员查看订单流程图

4) 密码修改流程图

已登录会员中心的会员点击密码修改，则跳转到密码修改界面，会员输入旧密码、新密码、确认新密码后，前端验证两次输入的新密码是否正确，若不正确，提示错误信息，操作结束。若正确，后台将验证旧密码是否正确，若不正确，提示错误信息，操作结束，若正确，密码修改完成。如图 4.8 所示：

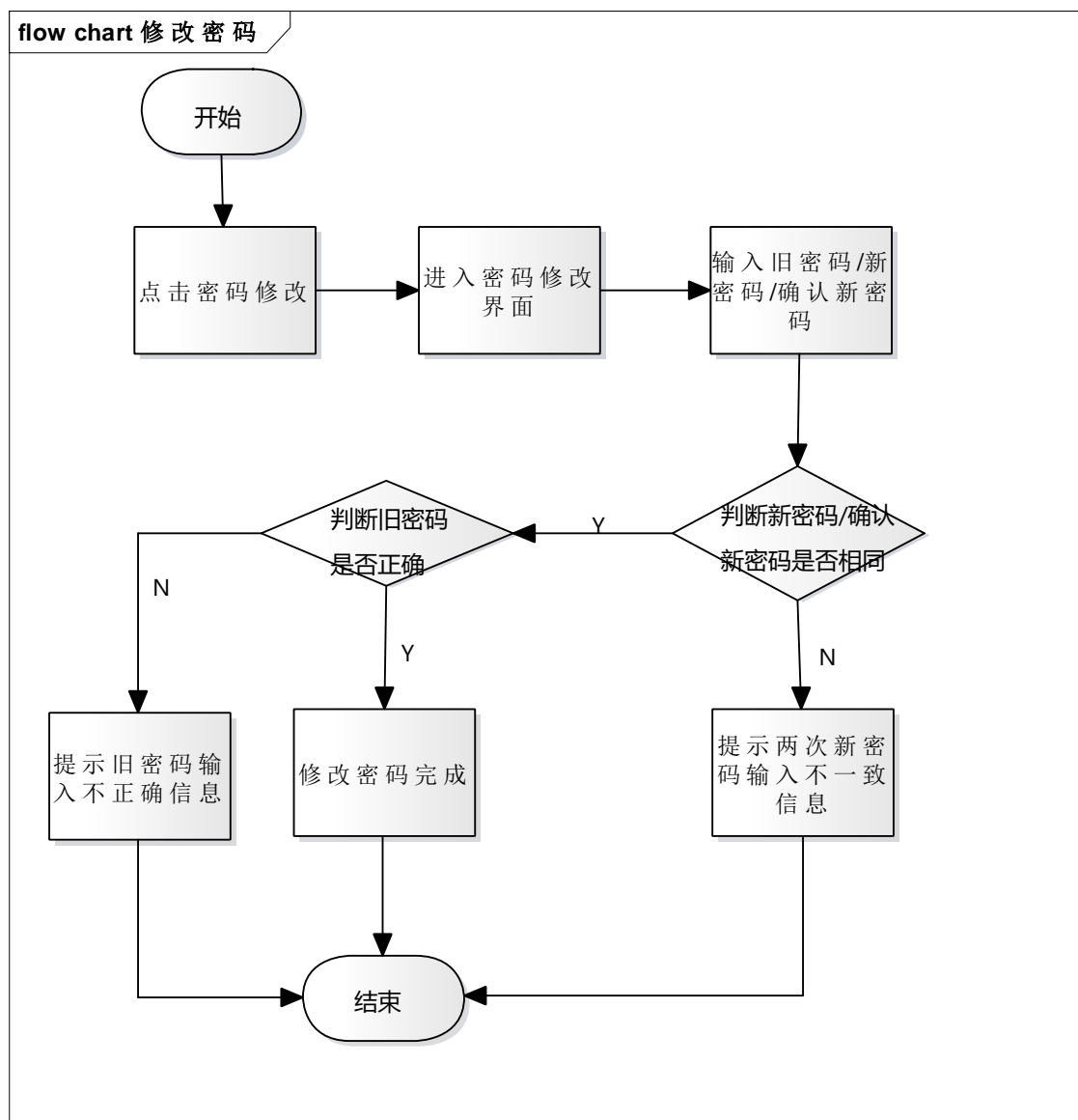


图 4.8 会员及管理员修改密码流程图

5) 管理员更改客房状态流程图

已经登录的管理员可以通过点击客房操作，进入客房操作界面。然后选择要操作的客房，选择此客房需要更改的状态（入住状态/退房状态），点击确定修改后会弹出确认框是否确认此操作，若选择否，操作结束。若选择是，客房更改状态完成。如图 4.9 所示

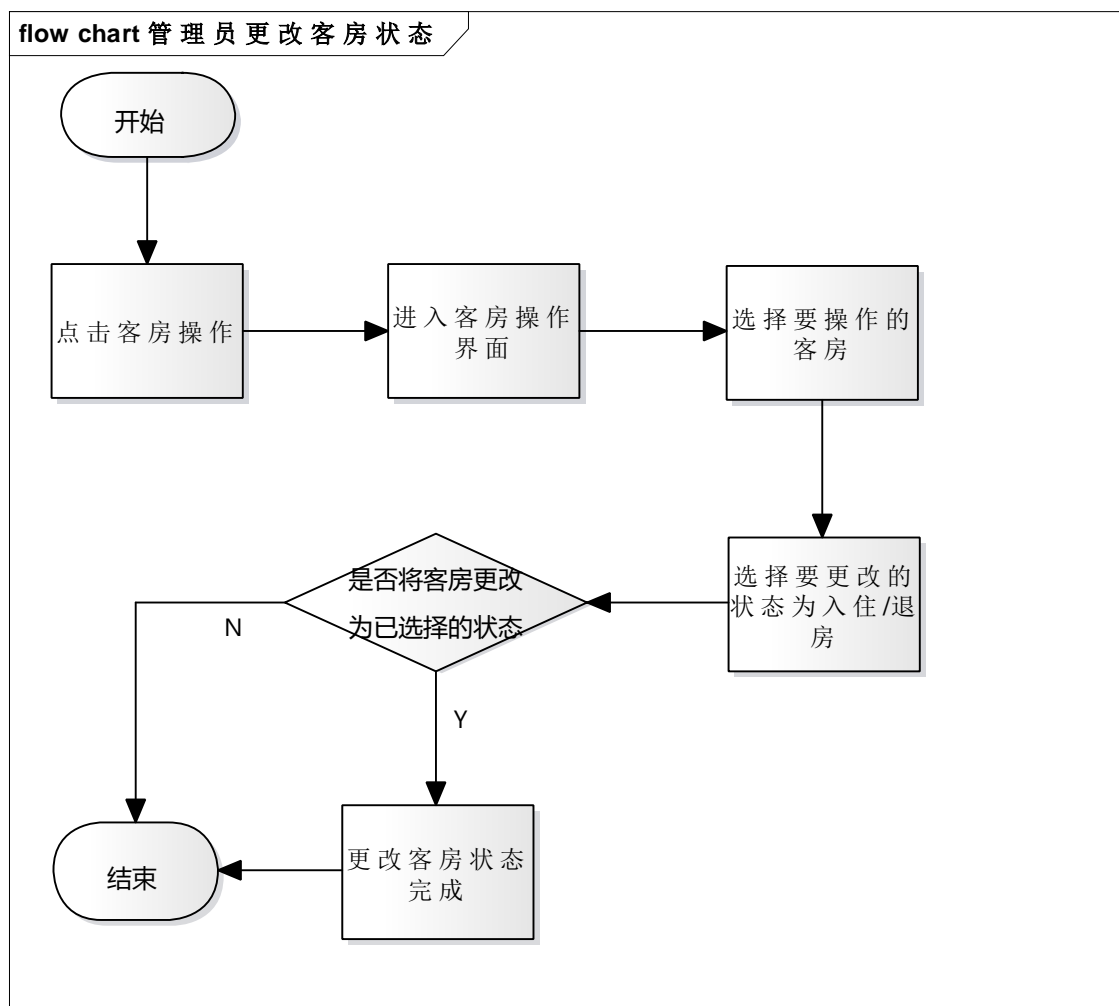


图 4.9 管理员更改客房状态流程图

4.2 系统详细设计

4.2.1 系统架构设计

酒店预订系统网站选择使用三层架构设计。分别为表示层（Member 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 4.10 所示：

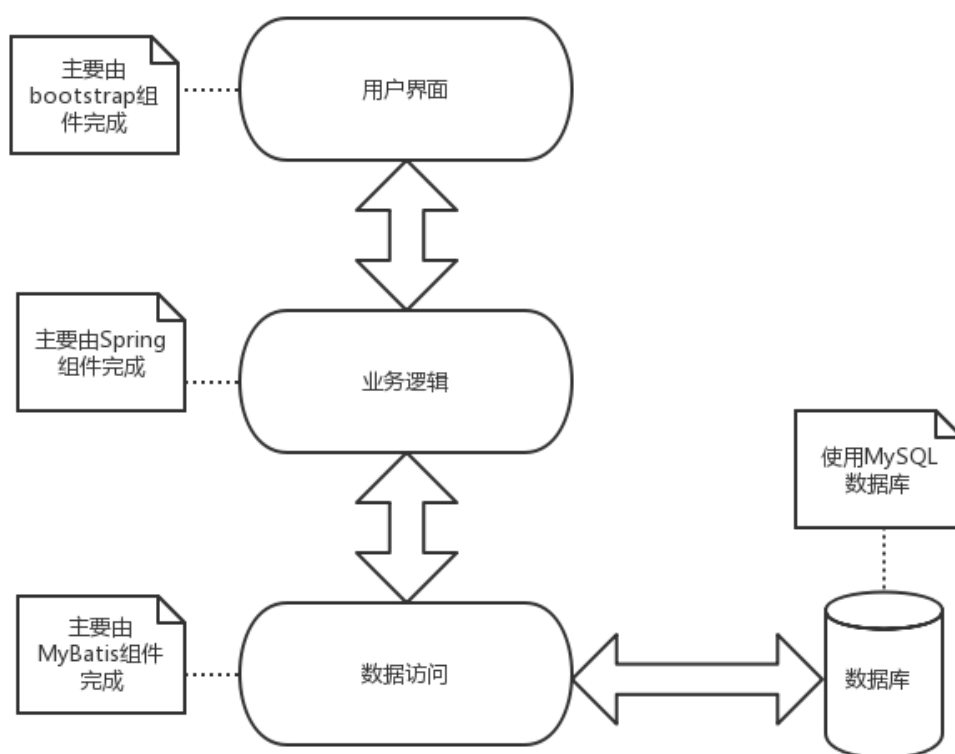


图 4.10 系统框架设计

4.2.2 系统设计模型时序图

在会员进行客房预订的时候会涉及到资源竞争问题，即多个会员同时预订一件客房的时候，这是必须要解决的。因流程图无法完全表达出对资源竞争的过程，用时序图加以补充说明。

1) 会员查看客房时序图

已登录的会员浏览客房的时候，一次完整的查询到显示过程如图 4.11 示：

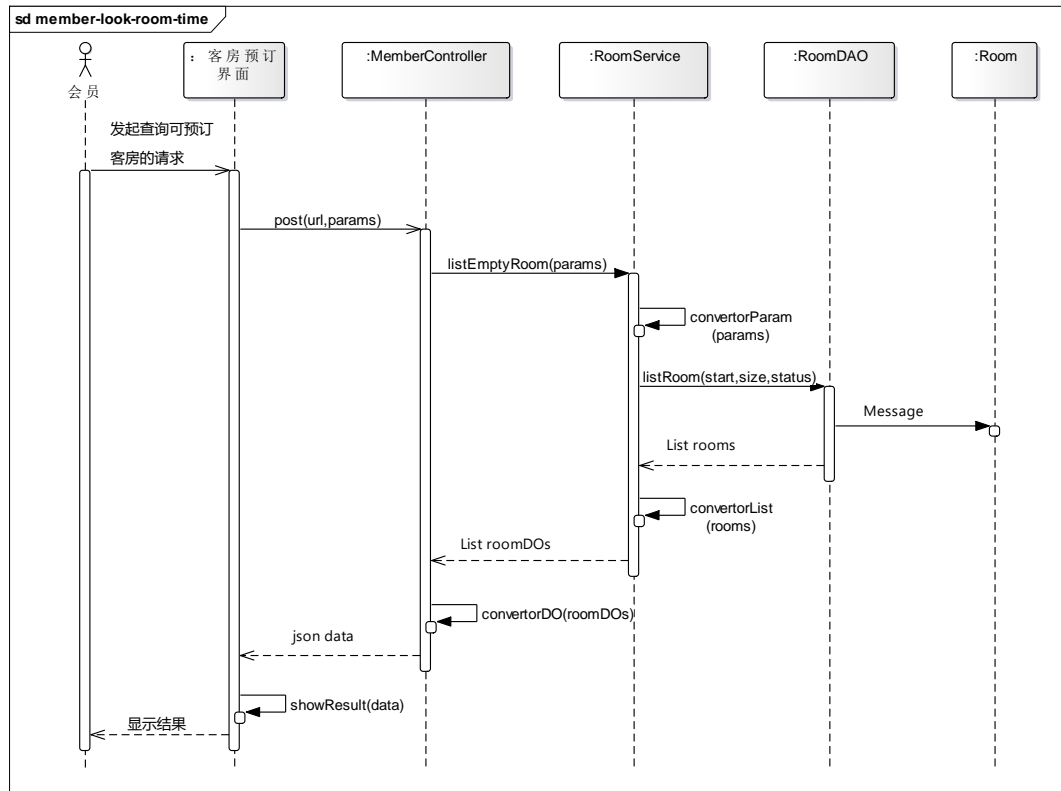


图 4.11 会员查看客房时序图

2) 会员预订客房时序图

已登录的会员预订客房时，一次完整的客房预订如图 4.12 所示：

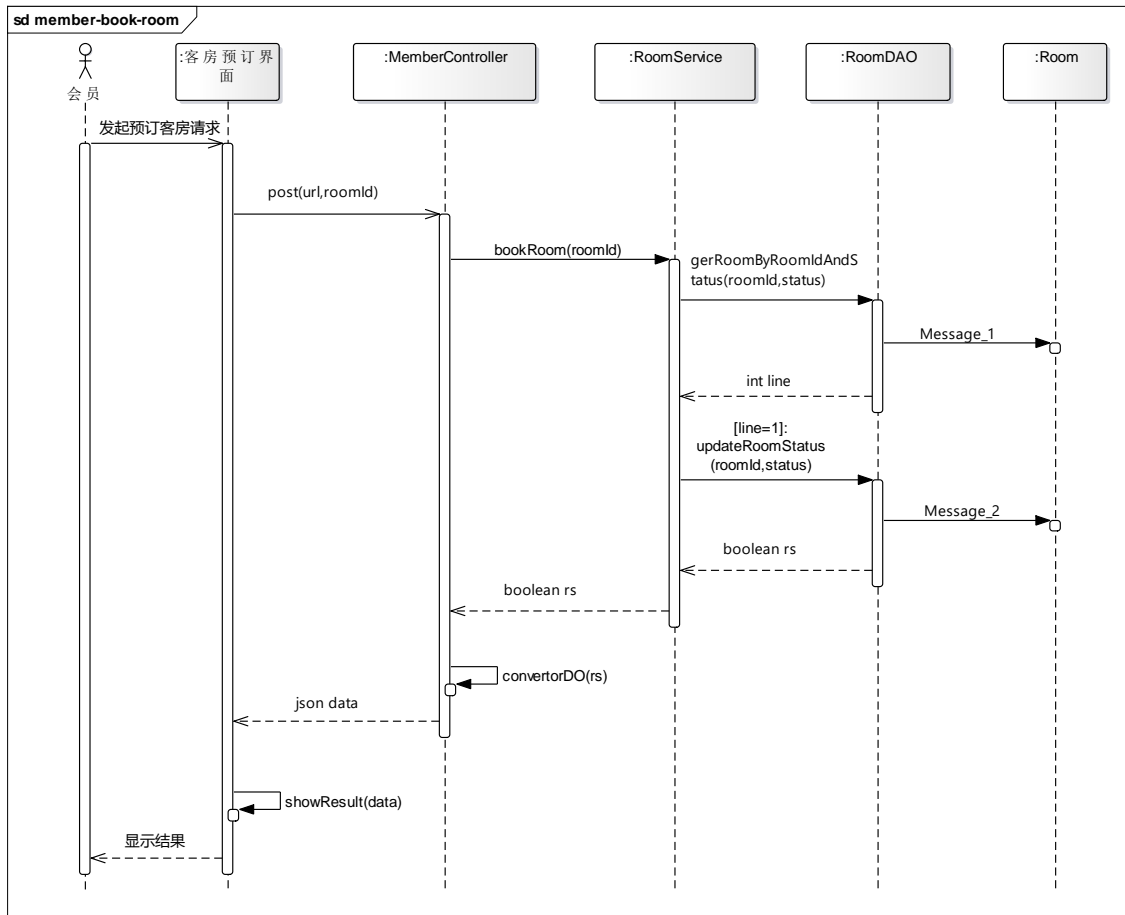


图 4.12 会员客房预订时序图

4.2.3 数据库设计

1) 类图设计

根据数据库建模的原则和设计技巧，结合本系统的业务流程，需要建立会员实体（Member），管理员实体（Admin），订单实体（Order）和客房实体（Room）。

会员首次使用酒店预订系统的时候需要注册账户并成功登录，在完成注册需要填写用户的基本信息，这些信息包括用户名（username）、密码（password）、联系电话（phone）、身份证号（idcard）、邮箱（email）、真实姓名（name）。

会员实体和订单实体是一对多的关系，所以订单实体中必须存在用户的身份证号属性来标识订单属于那个用户的。

会员实体和客房实体是一对多的关系，所以客房实体中必须存在用户的身份证号属性来客房由哪个会员预订的。

客房实体和订单实体是一对多的关系，所以订单实体中必须存在客房号属性来标识此订单会员入住的是那个客房。另外，在添加客房的时候，需要输入客房号（room_id），客房类型（type）、价格（money）和客房图片（picture）。

在系统底层类图设计中，每个类都有属性的 Getter 和 Setter 方法，为了让类图更加清晰，类图中将不显示方法，如图 4.13 所示：

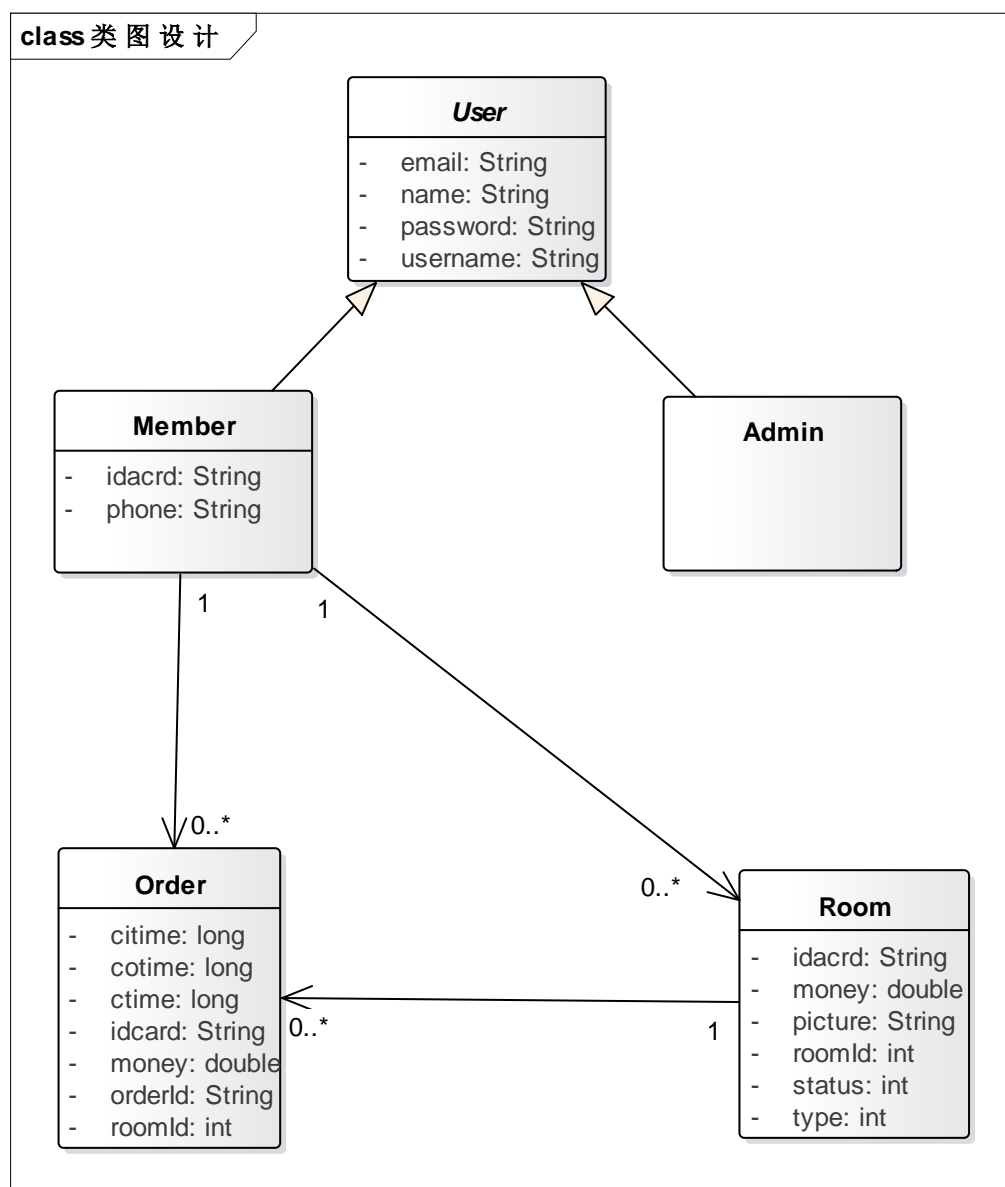


图 4.13 类图设计

用户 **User** 实体类泛化出会员 **Member** 实体类和管理员 **Admin** 实体类。

会员 **Member** 实体类和订单 **Order** 实体类是单向关联（1：N）关系，说明一个会员可以生成多条订单，但是一条订单只能由一个会员生成。

会员 Member 实体类和客房 Room 实体类是单向关联（1：N）关系，说明一个会员可以预订多个客房，但一个客房只能由一个会员预订。

客房 Room 实体类和订单 Order 实体类是单向关联（1：N）关系，说明一间客房信息可以被多条订单所记录，但是一条订单只能记录一间客房信息。

2) 数据库表设计

酒店预订系统中每个实体类都对象一张数据库表，每张数据库表的设计如下所示：

(1) 订单表

订单表（order）记录着会员预订客房信息，包括：订单编号（order_id）、订单创建时间（ctime）、会员入住时间（citime）、会员退房时间（cotime）、会员身份证号码（idcard）、客房号即房间号码（room_id）、消费金额（money），其中订单编号为此表的主键。其字段信息设计如表 4.1 所示。

表 4.1 order 数据库表

字段名称	字段类型	字段大小	索引	是否为空	备注
order_id	varchar	128	primary	N	订单编号
ctime	bigint	default		N	创建时间
citime	bigint	default		N	入住时间
cotime	bigint	default		N	退房时间
idcard	varchar	128		N	会员身份证
room_id	int	6		N	客房号
money	double	default		N	消费金额

(2) 客房表

客房表（room）记录着客房的各种详细信息，包括客房号即房间号码（room_id）、入住会员身份证号码（idcard）、客房状态（status）、客房图片地址（picture）、房间类型（type）和房间单日价格（money），其中房间号码为此表的主键。其字段具体设计如表 4.2 所示。

表 4.2 room 数据库表

字段名称	字段类型	字段大小	索引	是否为空	备注
room_id	int	6	primary	N	客房号
type	int	128		N	房间类型
status	int	3		N	客房状态
picture	varchar	512		N	图片地址
money	double	default		N	房间价格
idcard	varchar	128		N	预订者信息

(3) 会员表

会员表（member）记录着会员的各种信息，包括登录账号（username）、登录密码（password）、会员真实姓名（name）、会员邮箱地址（email）、会员电话号码（phone）和会员身份证号码（idcard），其中登录账号为此表的主键。其字段信息具体设计如表 4.3 所示。

表 4.3 member 数据库表

字段名称	字段类型	字段大小	索引	是否为空	备注
username	varchar	20	primary	N	账号
password	varchar	34		N	密码
name	varchar	30		N	真实姓名
email	varchar	64		Y	邮箱地址
phone	varchar	64		N	手机号
idcard	varchar	64		N	身份证

(4) 管理员表

管理员表（admin）记录着管理员的各种信息，包括登录账号（username，）、登录密码（password）、真实姓名（name）、邮箱地址（email），其中登录账号为此表的主键。其字段信息具体设计如表 4.4 所示。

表 4.4 admin 数据库表

字段名称	字段类型	字段大小	索引	是否为空	备注
username	varchar	20	primary	N	账号
password	varchar	50		N	密码
name	varchar	20		N	真实姓名
email	varchar	50		N	邮箱

4.2.4 界面设计

1) 会员登录界面设计

登录页面主要设置用户名和密码输入框，系统会将输入的信息与数据库中的存储的账号及加密后的密码比对。会员登录界面设计如图 4.14 所示。



图 4.14 会员登录界面

2) 会员个人中心界面设计

已登录的会员在点击导航栏系统菜单中的功能模块时，下册的功能模块界面也会随之切换，功能模块详细界面如图 4.15 所示：



图 4.15 个人中心界面

3) 会员注册界面设计

会员注册页面含有会员注册是所填的基本信息，包括账号、密码、真实姓名、身份证号码、手机号码 5 项必填信息和邮箱选填信息。如图 4.16 所示：

The image shows a browser window with the address bar displaying 'www.hotel.com/register.html'. The page has a title '会员注册' (Member Registration) at the top. Below the title is a registration form with the following fields and labels: '账号:' (Account), '密码:' (Password), '姓名:' (Name), '身份证:' (ID Card), '手机号:' (Phone Number), and '邮箱:' (Email). Each label is followed by a text input field. At the bottom of the form is a gray button labeled '注册' (Register).

图 4.16 会员注册界面

4) 管理员后台管理界面设计

已登录的管理员在点击导航栏系统菜单中的功能模块时，下册的功能模块界面也会随之切换，功能模块详细界面如图 4.17 所示：



图 4.17 后台管理界面

4.2.5 系统接口设计

项目整体目录规范设计如图 4.18 所示：

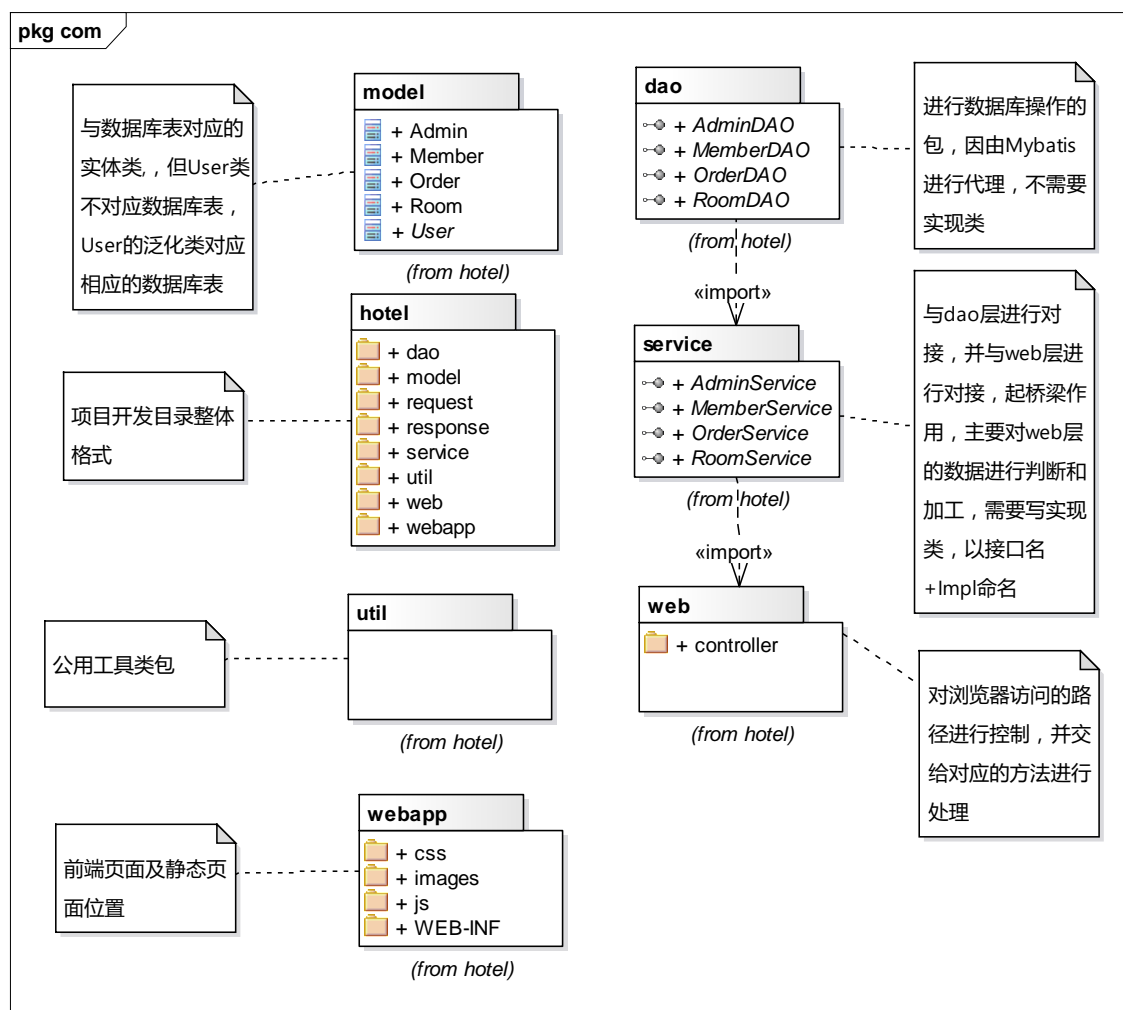


图 4.18 项目整体目录规范

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 4.19 所示：

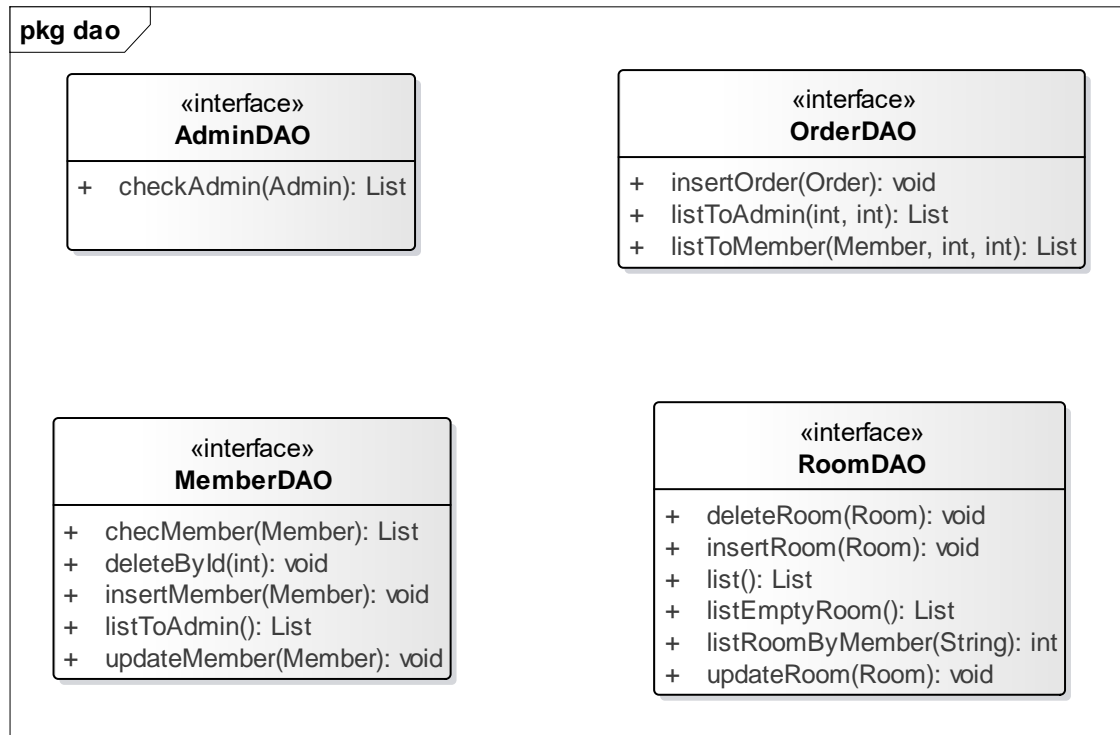


图 4.19 dao 层接口设计

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 4.20 所示：

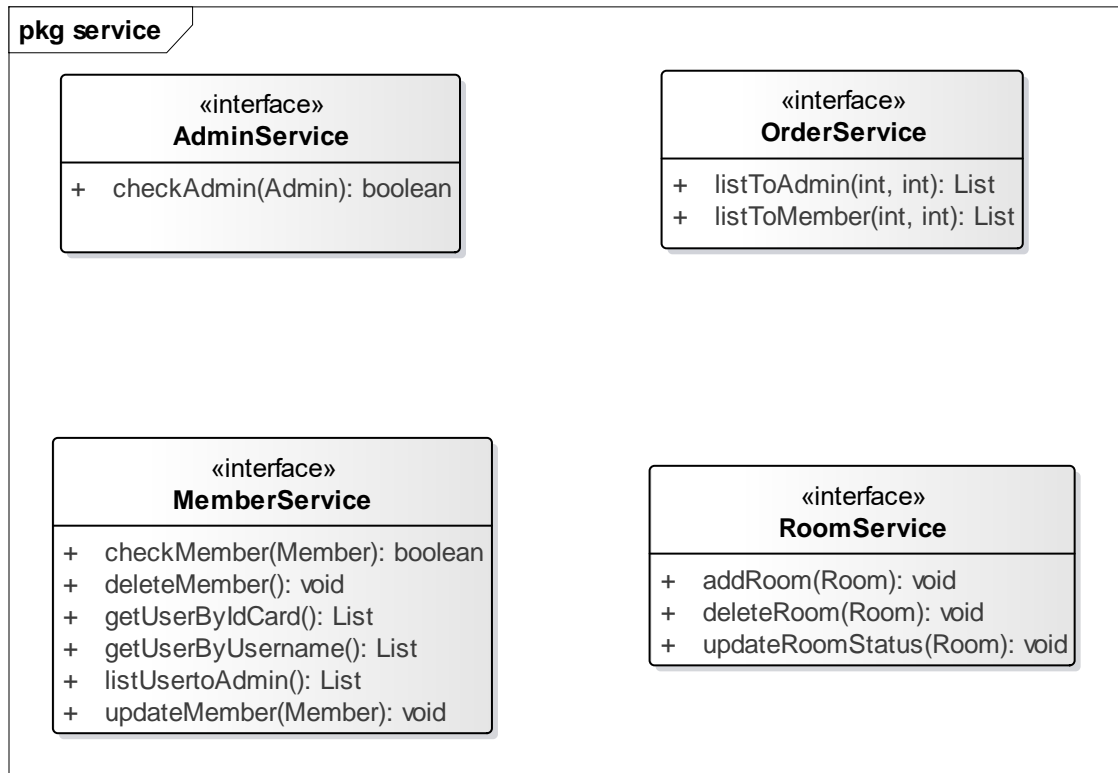


图 4.20 service 接口设计

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 4.21 所示：

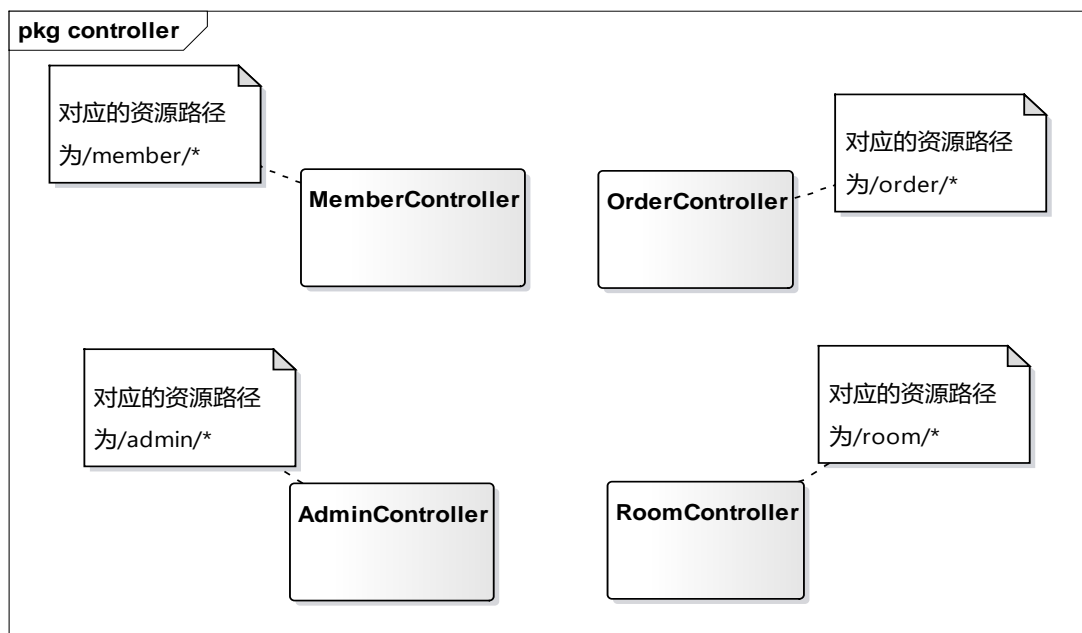


图 4.21 controller 接口设计

第 5 章 系统实现及测试

5.1 系统开发及运行环境

1) 系统开发的硬件环境如表 5.1 所示：

表 5.1 系统开发硬件环境

硬件	
处理器数量	1
处理器速度	1.7GHz
内存容量	1GB
硬盘容量	40GB
网络	10Mbps/s LAN

2) 软件开发所需的软件环境如表 5.2 所示：

表 5.2 系统开发软件环境

软件	
操作系统	Linux
JDK 版本	1.7
服务器中间件	Tomcat
服务器中间件版本	7.x
数据库版本	MySQL 5.1
JDBC 驱动器制造商	MySQL
JDBC 驱动器版本	5.1.6

5.2 数据库连接

数据库连接信息详情如表 5.3 所示：

表 5.3 数据库连接信息

属性名	属性值	备注
driver	com.mysql.jdbc.Driver	选择与数据库连接的驱动
url	jdbc:mysql://112.74.38.145:3306/hotel	选择连接的数据库地址
username	47	数据库账号
password	47721root	数据库密码

5.3 实现部分展示

5.3.1 会员登录界面实现

当会员登陆时，拦截器会拦截会员的登录请求，当后台校验页面出来的用户和密码有误的时候，提示错误信息。登录页面如图 5.1 所示：

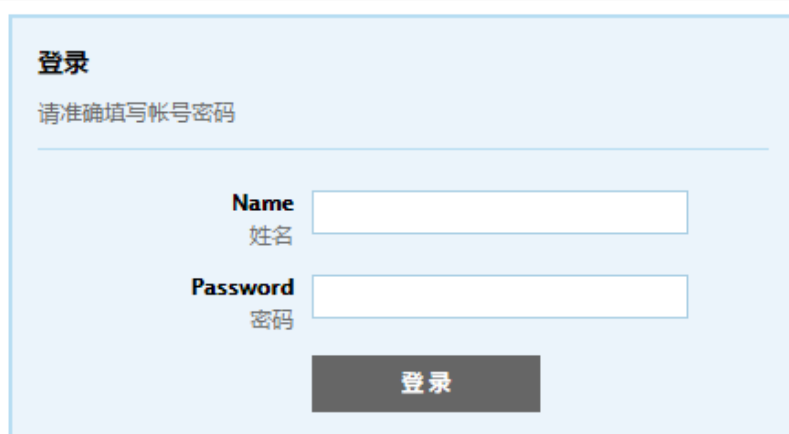


图 5.1 登录界面

会员登录拦截及密码验证代码如图 5.2 所示：

```
public boolean preHandle(HttpServletRequest request, HttpServletResponse response, Object handler)
    throws Exception {
    HttpSession session = request.getSession();
    if (session.getAttribute("user") == null) {
        response.sendRedirect("/login.html");
        return false;
    }
    return true;
}
```



```
@Override
public UserDO checkUser(HttpServletRequest request, User user) {
    //此处使用MD5加密验证登录
    user.setPassword(MD5Utils.passwordToMD5(user.getPassword()));
    List<User> list = userDAO.checkUser(user);
    UserDO ud = new UserDO();
    if (list.size() != 1) {
        ud.setCode(Constant.User.WRONG_CODE);
    } else {
        request.getSession().setAttribute(s: "user", list.get(0));
    }
    return ud;
}
```

图 5.2 会员登录拦截及密码验证代码

5.3.2 个人中心界面实现

1) 个人信息界面展示，如图 5.3 所示。

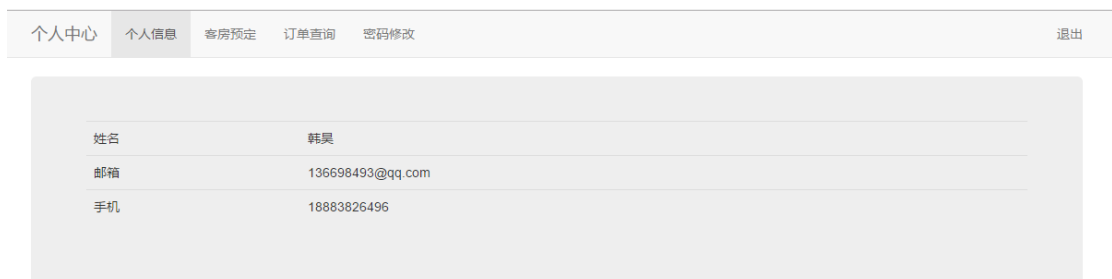


图 5.3 个人信息展示

查看个人信息代码如图 5.4 所示：

```
@Override
public UserDO getUserInfo(HttpServletRequest request) {
    HttpSession session = request.getSession();
    UserDO ud = new UserDO();
    User u = (User) session.getAttribute(s: "user");
    if (u == null) {
        ud.setCode(Constant.User.WRONG_CODE);
    }
    userToUserDO(u, ud);
    return ud;
}
```

图 5.4 查看个人信息代码

2) 客房预订界面展示如图 5.5 所示。

客房号	类型	价格	操作
105	小床房	100	+
106	小床房	100	+
107	小床房	100	+
108	中床房	200	+
109	大床房	300	+
110	中床房	200	+

图 5.5 客房预订界面

查询客房代码实现如图 5.6 所示：

```

@Override
public CommonDO emptyRoom(Request request) {
    List<Room> list = roomDAO.listByStatus(Constant.Room.UNBOOK,
        start: (request.getStart() - 1) * request.getSize(), request.getSize());
    CommonDO cd = new CommonDO();
    List<RoomDO> dolist = new ArrayList<>();
    RoomDO rd = null;
    for (Room room : list) {
        rd = new RoomDO();
        rd.setMoney(room.getMoney());
        rd.setType(room.getType());
        rd.setRoom_id(room.getRoom_id());
        dolist.add(rd);
    }
    cd.setData(dolist);
    return cd;
}

```

图 5.6 查询客房代码

3) 订单查询界面展示如图 5.7 所示:

个人中心	个人信息	客房预定	订单查询	密码修改	退出
------	------	------	------	------	----

订单号	房间号	入住时间	结束时间	消费金额
3707851994081455181494341792329	104	2017-05-09	2017-05-10	300

图 5.7 订单查询界面

订单查询代码如图 5.8 所示:

```

@Override
public CommonDO listToUser(OrderQueryRequest orderQueryRequest, HttpServletRequest request) {
    CommonDO commonDO = new CommonDO();
    //会员只能查看自己的订单，不可以查询其他人的订单
    User u = (User) request.getSession().getAttribute(s: "user");
    List<Order> list = orderDAO.listToUser(u.getUsername(),
        start: (orderQueryRequest.getStart() - 1) * orderQueryRequest.getSize(),
        orderQueryRequest.getSize());
    commonDO.setData(order2OrderDO(list));
    return commonDO;
}

```

图 5.8 订单查询代码

4) 修改密码界面展示如图 5.9 所示：

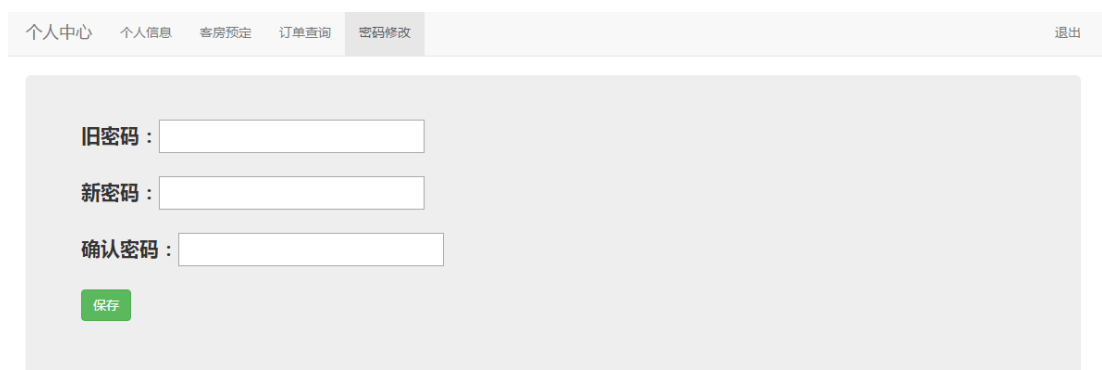


图 5.9 修改密码界面

更改密码代码如图 5.10 所示：

```
@Override
public UserDao updateUserPwd(UserRequest user, HttpServletRequest request) {
    UserDao userDao = new UserDao();
    User u = (User) request.getSession().getAttribute("user");
    //数据库存储的加密后的密码，这里需要将密码加密后再做校验
    String oldpwd = MD5Utils.passwordToMD5(user.getOldpwd());
    String newpwd = MD5Utils.passwordToMD5(user.getNewpwd());
    boolean flag = userDao.updateUserPwd(oldpwd, newpwd, u.getUsername())==1;
    if(!flag){
        userDao.setCode(Constant.User.WRONG_CODE);
    }
    return userDao;
}
```

图 5.10 更改密码代码

5.3.3 后台管理中心界面实现

1) 后台管理中心客房管理界面如图 5.11 所示：

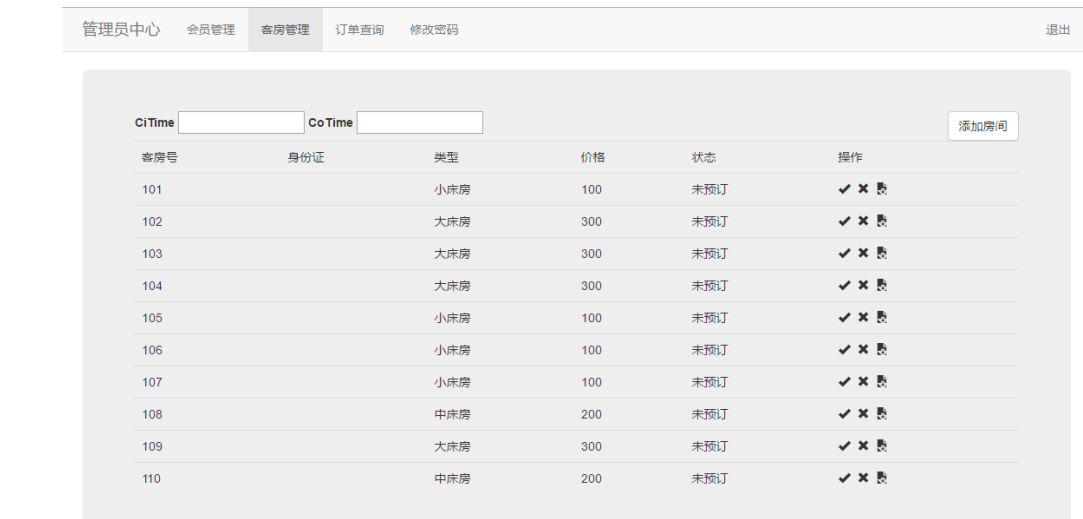


图 5.11 客房管理界面

5.4 系统测试

5.4.1 测试目的和意义

软件测试是程序的一种执行过程，目的是尽可能发现并改正被测试软件中的错误，提高软件的可靠性^[15]。它是软件生命周期中一项非常重要且非常复杂的工作，对软件可靠性保证具有极其重要的意义。在目前形式化方法和程序正确性证明技术还无望成为实用性方法的情况下，软件测试在将来相当一段时间内仍然是软件可靠性保证的有效方法。软件工程的总目标是充分利用有限的人力和物力资源，高效率、高质量地完成软件开发项目。不足的测试势必使软件带着一些未揭露的隐藏错误投入运行，这将意味着更大的危险让用户承担。过度测试则会浪费许多宝贵的资源。到测试后期，即使找到了错误，然而付出了过高的代价。E.W.Dijkstra 的一句名言说明了这一道理：“程序测试只能表明错误的存在，而不能表明错误不存在。”可见，测试是为了使软件中蕴涵的缺陷低于某一特定值，使产出、投入比达到最大^[16]。

5.4.2 测试方法

从软件开发流程来看，各项软件测试是在项目开发流程中交替进行的。软件测试包含单元测试、集成测试、系统测试和验收测试。

首先在完成单个功能实现时，就需要对其进行单元测试，只有保证单个功能内部不出现错误，在系统集成功能时才能不出错；然后在组装系统功能时进行集成测试，测试系统是否正常运行；接下来通过系统整体测试来完善和优化系统功能；最后在投入使用前对其进行验收测试。软件测试的方法主要是黑盒测试和白盒测试。

5.4.3 测试用例

由于本系统所涉及的功能模块较多，这里只讨论会员登录模块的测试用例，系统用户登录的测试用例描述如表 5.4 和表 5.5 所示：

表 5.4 用户登录测试用例

功能特性	用户登录
测试目的	验证输入的会员登录信息时，系统是否校验用户信息并对结果做出响应。
测试数据	账号：hanhao 密码：testhh

表 5.5 用户登录测试用例

测试内容	操作描述	数据	期望结果	实际结果	测试状态
1	只输入账号	账号：hanhao	提示“密码不能为空”	提示“密码不能为空”	结果一致
2	只输入密码	密码：testhh	提示“账号不能为空”	提示“账号不能为空”	结果一致
3	输入正确的账号和密码	账号：hanhao 密码：testhh	跳转到个人中心界面	跳转到个人中心界面	结果一致
4	输入正确账号和错误密码	账号：hanhao 密码：1233	提示“账号或密码错误”	提示“账号或密码错误”	结果一致
5	输入错误账号正确密码	账号：hanh 密码：testhh	提示“账号或密码错误”	提示：“账号或密码错误”	结果一致

第6章 总结与展望

6.1 个人工作总结

本次设计主要是酒店通过客户对客房的选择，帮主客户进行客房预定，实现酒店预订系统。让客户足不出户就可以预订自己想要的客房。本系统主要实现了客户预订客房，查询自己的订单以及管理员对客房进行管理，例如：增加客房，修改客房信息，删除客房，更改客房状态，管理员对订单进行查询，按照时间范围查询、按照客户身份证号查询、按照订单号查询，管理员对会员信息的管理，查看和删除操作。

通过这次毕业设计，我主要学习了如何使用 Java 进行服务端的开发，同时复习了关于软件工程以及数据库方面的知识。

在整个系统的开发中也遇到了一些问题，比如界面设计，如何对页面布局才能使页面显示的更加好看和优雅。又比如，多个顾客同时预订同一间客房的资源竞争问题，如何保持数据库数据的正确性等等问题。我不断的学习和查询资料，从老师和同学处获取帮助，终于完成了一个功能相对完善的酒店预订系统。

6.2 后续研究工作展望

在本次项目设计中，虽然后台功能基本完全实现，但是前台功能因为个人技术原因，页面设计做的不够美观优雅。后续将会对前端页面进行美化和部分页面重新设计。

在移动支付盛行的今天，后序系统可以加入二维码支付，做到真正的自动化预订和管理功能。二维码将附带个人身份信息，取消房卡，使用一张二维码即可实现支付功能、房卡功能，做到真正的是“一码一房”。

参考文献

- [1] 赵景晖. Java 程序设计[M].北京机械工业出版社,2005:1-2.
- [2] BruceEckel. Thinking In java [M]. Prentice Hall,2006:1-378
- [3] Joni Murti Mulyo Aji;Exploring Farmer-supplier Relationships in the East Java Seed Potato Market[J]; 爱思唯尔期刊;2016: 220-225
- [4] Tom Negrino, Dori Smith. JavaScript&Ajax Sixth Edition [M]. Peachpit Press, 2006:1-2
- [5] 李东博编著. HTML5+CSS3 从入门到精通 [M] . 清华大学出版社, 2013:1-10
- [6] 徐雯;高建华.基于 Spring MVC 及 MyBatis 的 Web 应用框架研究[J].微型电脑应用,2012 年 07 期
- [7] John Jorstad.SSM Provides Important Advantages; So, why has SSM Failed to Achieve Greater Market Share[J]; Trans Tech,2015: 481-486
- [8] 贺军, 焦荣. 基于 XML 应用的 MyBatis 技术[J].火力与指挥控制, 2013(s1):20-30
- [9] 刘亚宾,杨红.精通 Eclipse[M].北京:电子工业出版,2005
- [10] 张孝祥.深入 Java Web 开发内幕——核心基础[M].电子工业出版社,2006
- [11] Kartika Firdausy, Samadri, Anton Yudhana.Web based Library Information System Using PHP and MYSQL[J]. DOAJ,2008:50-60
- [12] 林信良. Spring2.0 技术手册[M].电子工业出版社,2005
- [13] 徐建波,周新莲.Web 设计原理与编程技术[M].中南大学出版社, 2005:185-193
- [14] 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,2015:146
- [15] 孙涌.现代软件工程[M].北京希望电子出版社,2003.8:1-246
- [16] 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

致谢

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。

附录

一、英文原文

I. 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 member mailing list. Questions on the Framework itself should be asked on StackOverflow .

II. 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.

III. 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.

IV. 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.

V. 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.

VI. 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.

VII. 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.

VIII. 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.

IX. 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.

X. 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.

二、英文翻译

1. 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。

2. 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。

3. 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。

4. 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。

5. 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。

6. 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。

7. 消息传递

Spring Framework 4 包含一个 Spring 消息模块,其中包含来自 Spring Integration 项目的关键抽象,比如消息、MessageChannel、MessageHandler,以及其他作为基于消息的应用程序的基础。该模块还包括一组用于将消息映射到方法的注释,类似于 Spring MVC 基于注释的编程模型。

8. 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。

9. 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。

10. 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。

