目 录

中文摘要

现如今，社会信息化的发展速度越来越快，各行各业的竞争也越来越激烈，因此会议数量急剧增加，但仍有有些中小型企业或者单位继续延用人工预订通知会议的方式，使得会议管理方面出现种种问题，比如效率低，通知慢，预订冲突，会议室一直空闲不被得知等。为此，本文设计了一个把会议室信息化，预约方式简单化，通知流程高效化的会议室预约管理系统，提高这些中小型企业内部会议管理工作的效率，核心工作内容如下：

首先，对使用的 MVC 开发模式思想及 SSM 框架进行了介绍，并简单概述了 SSM 中三大框架的概念及优点，介绍了面向对象的方法，RUP 方法和 UML 建模。在方法的指导下描述了系统需求的功能性需求和非功能性需求的内容，并进行系统用例捕获，利用事件流，活动图和时序图对系统用例进行描述，为下文系统设计与实现垫下基础。然后，简单说明了系统设计的主要任务、内容、原则，通过类和包图的设计说明了系统的总体设计，并通过数据库 E-R 图和数据库表的设计完成系统的数据库设计。紧接着介绍本系统项目系统层次设计和 SSM 整合三大框架的过程，并以预约会议功能为例，对具体实现流程和代码进行一一地说明。最后，对实现的系统模块功能进行测试，并简要描述各功能的操作流程及测试情况，并把记录的测试结果和实现效果图进行展示。

关键词：会议室；SSM；管理系统

作　　者：刘汉文

指导教师：凌兴宏

Abstract

Nowadays, the development of social informatization is faster and faster, and the competition from all walks of life is more and more fierce. Therefore, the number of meetings is increasing rapidly. However, some small and medium-sized enterprises or units continue to use the way of manual reservation to notify the meeting, which leads to various problems in meeting management, such as low efficiency, slow notification, reservation conflict, etc. To this end, this paper designs a meeting room reservation management system, which makes the meeting room information-based, the appointment method simple, and the notification process efficient, to improve the efficiency of the internal meeting management of these small and medium-sized enterprises：

Firstly, the MVC development pattern and SSM framework are introduced, and the concepts and advantages of the three frameworks in SSM are briefly summarized. The object-oriented method, RUP method and UML modeling are introduced. Under the guidance of the method, the functional requirements and non functional requirements of the system requirements are described, and the system use cases are captured. The event flow, activity diagram and sequence diagram are used to describe the system use cases, which lays the foundation for the following system design and implementation. Then, the main tasks, contents and principles of the system design are briefly described. The overall design of the system is explained through the design of class and package diagram. The database design of the system is completed through the design of database E-R diagram and database table. Then it introduces the process of system level design and SSM integration of the system project, and takes the appointment meeting function as an example to explain the specific implementation process and code one by one. Finally, the function of the system module is tested to show the test results and the effect diagram.

Keywords: conference room; SSM; Management system;

Written by: Liu Han-wen

Supervised by: Ling Xing-hong

第一章 绪论

1.1 课题背景

随着行业间的竞争越来越激烈，企业事务规模越来越大，导致各个部门都重视沟通，但是进行沟通的有效方法之一是举行有效的会议，而会议室是恰恰是一个公司举行集体决策、讨论、调查、总结、表彰等会议的重要场所[1]，并且会议室预约管理更是每个企业都将面临的管理问题，选择合适的方式让有限的会议室资源在有限的时间内发挥最大的作用是一个企业管理者应慎重思考的重要问题之一，可是一些中小型企业管理者却选择了继续使用人工管理的方式预约管理会议室。然而，随着会议数量的不断增多,会议室资源极度有限，会议管理工作任务变得异常繁重，导致使用这种方式不仅操作难度大，不便于信息发布与管理，而且容易出错。

随着现在科学技术的飞速发展,特别是计算机产业的大量应用在人们的现实生活中，给人们的生活带来了巨大的变化。而会议室预约管理系统是正是通过计算机进行管理，显示会议室的使用情况，会议的主题，参议的人员，会议的位置等。它的优势很明显，通过互联网在计算机上登录会议室预约管理系统就可以进行会议室的预订，能直接避免手工化管理会议室的使用冲突等问题，方便快捷。从更深层次来讲，就是它能帮助企业更规范更信息化地通过互联网管理会议室资源，使会议室的使用效率大大提升，让企业的管理制度更加完善。

1.2 国内外现状

面对移动互联网的蓬勃发展和 OA 系统不断发展，国外不仅有完善的会议室管理系统，而且对于现在新兴的智能能化产业，许多企业采用移动互联技术和终端设备相结合的方式，把会议室管理系统融入到智能化中，从不同方面对传统会议服务模式进行了修正,使会议室的预约更加简便，使会议室更加智能。比如，国外 cvent 公司通过发售与会议服务相关的移动产品，迅速改变了本公司的会议服务的商业模式，占据了业界的首位，使大部分中小型企业体会到会议服务的便利。反观我们国内，现在，还有大多数中小型企业都是人工化管理各个会议室的日常使用，手动分配和查询会议室，不仅率低下，而且对各楼层的会议室空闲、预定情况、会议室各种资源状态、后勤保障等不能做到了如指掌，所以导致会议室资源冲突，会议室空闲，不能及时预订或者预订情况不能及时通知等现状。由于这些因素影响，国内的会议室服务行业的发展趋势也没有多剧烈，只有一些大型企业才会用到。

目前国内，会议室管理系统的研究采用的主流框架技术是SSM框架，刚开始主要采用SSH框架结构，之后随着技术的发展，为了更加便捷易用，SSM框架被创建并成为了该系统的主流开发框架技术，并且在我国对该系统的开发模式有四种：对已有系统进行二开；自主设计实现的系统；在Windows.Net框架基础上开发系统；基于java开发的系统。并且这些会议室管理系统的除了基本功能以外，还具有以下特殊功能的开发：

（1）网上实时预订

借助互联网技术，把会议室状态上传到云端，用户只需要在只需在手机或平板电脑上轻点几下，就可以快速查找到所需的会议室,进行会议预订。通过红外感应模块，还可以即时感知会议室是否有人占用，帮助用户解决预订时间的麻烦。

（2）终端系统显示

使用移动端的会议室管理系统进行预订会议室，实时的会议预订信息会发送到每个会议室的门口终端显示设备上面，方便会议人员的查看和寻找。更是把移动和终端管理功能二合一。

（3）通信软件的通知

如果预订会议成功,将及时通过通信软件通知会议人员该会议的预订消息。

1.3 课题主要内容

为了增加会议管理工作的效率，结束手工化的会议使管理，本次设计目的在于开发一个能够解决会议室预约管理，通知人员会议内容，方便人员预约的软件。通过掌握的基本知识实现一个集性能稳定，外观合适，功能完善的软件系统，能够实现用户登录、人员管理、会议室管理、会议信息管理等功能。本文主要内容通过对SSM框架技术、MVC开发思想、面向对象的开发方法及UML建模语言等技术分析了对会议室预约管理系统开发的意义，然后在需求阶段的系统需求和设计方面使用用例图分析，接着描述实现系统和系统测试内容。通过这些内容展示系统实现的过程以及系统对会议室管理信息化的作用和优点。

1.4 论文组织结构

通过对本课题的理解和研究，我把本次论文分成7个章节，详细阐述会议室预约管理怎么基于SSM框架结构设计与实现的，具体章节安排如下：

第一章绪论：介绍了背景，国内外现状，课题主要内容，以及论文的组织结构。

第二章相关技术及开发方法：介绍设计与实现会议室预约管理系统所需要的方法和

技术。

第三章系统分析：根据调查情况对系统的需求进行分析，并进行用例的描述。

第四章系统设计：根据需求进行系统设计及数据库设计。

第五章系统实现：阐述系统具体内容如何实现。

第六章系统测试：系统功能模块测试，并展示测试结果。

第七章总结与展望：总结本文的内容，并提出未来系统提升的方案。

第二章 相关技术及开发方法

2.1 MVC 模式

MVC 模式(Model-View-Controller)是软件工程中一种常用的软件架构模式，模型(Model)，视图(View)，控制器(Controller)是它的三个基本部分，通过它软件系统也分成了这个三个基本部分[2]。

Model：程序员编写实现某种功能算法，实现数据库管理与设计，封装并应用某程序应用的业务逻辑对应的相关数据处理方法。而且 Model 不依赖 View 和 Controller，Model 中的数据变化一般通过某种刷新机制被公布。所以，为了实现这种机制，Model 的 View 必须现在 Model 上注册，之后，View 便可以知道在数据 Model 上发生的改变。

View：图形界面设计，对数据进行显示。View 没有程序逻辑，它需要刷新功能，就得事先在它监视的数据模型（Model）的数据那里进行注册。

Controller：负责界面及功能请求的处理，在不同层面间起到组织作用以便于控制整个应用程序的流程，对请求事件（用户的行为及数据的改变）进行处理并做出响应。

MVC 工作流程示意图如图 2-1 所示。

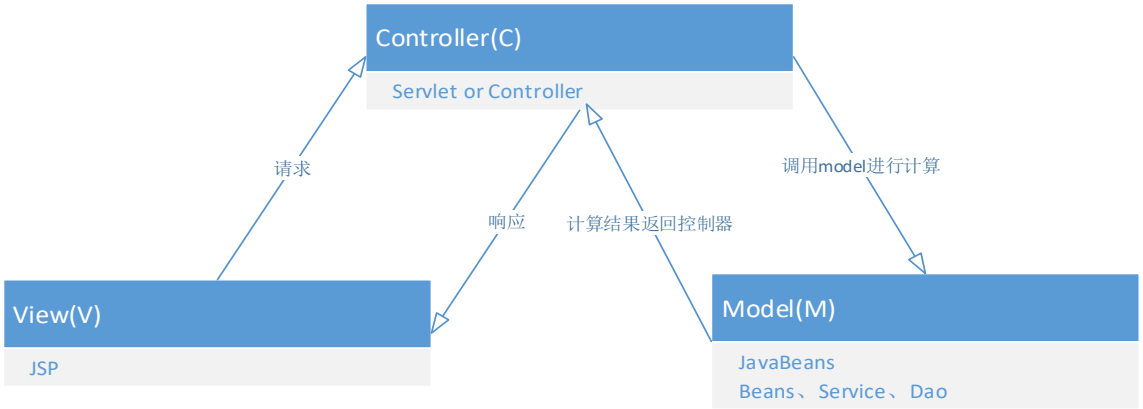

图 2-1 MVC 架构图

MVC 工作流程：

(1) 用户通过 View 界面，调用 Controller 控制器，对它发送请求指令；

(2) Controller 控制器接受请求，选择合适的模型 Model；

(3) Model 接受控制指令之后获取相应数据，返回给控制器；

(4) 控制器接受数据之后，按照相应的请求指令获取相应的视图；

(5) 视图 View 便显示用户想要的数据显示出来。

2.2 SSM 框架

SSM 框架是 Spring+SpringMVC+Mybatis 的缩写，是目前主流的 java EE 企业级框架，可以适用搭建众多大型企业级应用系统，使用它具体易复用，简化开发的好处，能快速掌握每个框架的核心思想[3]，如图 2-2 所示的 SSM 框架结构图。

(1) SpringMVC：在项目运行过程中拦截用户请求，它的核心 Servlet 即 Dispatcher Servlet 承担中介或是前台这样的职责，将用户请求通过 HandlerMapping 去匹配 Controller(具体对应请求所执行的操作)。

(2) MyBatis：对 jdbc 封装，它让数据库底层操作变得透明。

(3) Spring：整个项目中装配 bean 的大工厂，在配置文件中可以指定使用特定的参数去调用实体类的构造方法来实例化对象。

SSM 框架就是 MVC 模式的重要应用，它与 MVC 模式的 MVC 架构中的功能对应如图 2-3 所示。

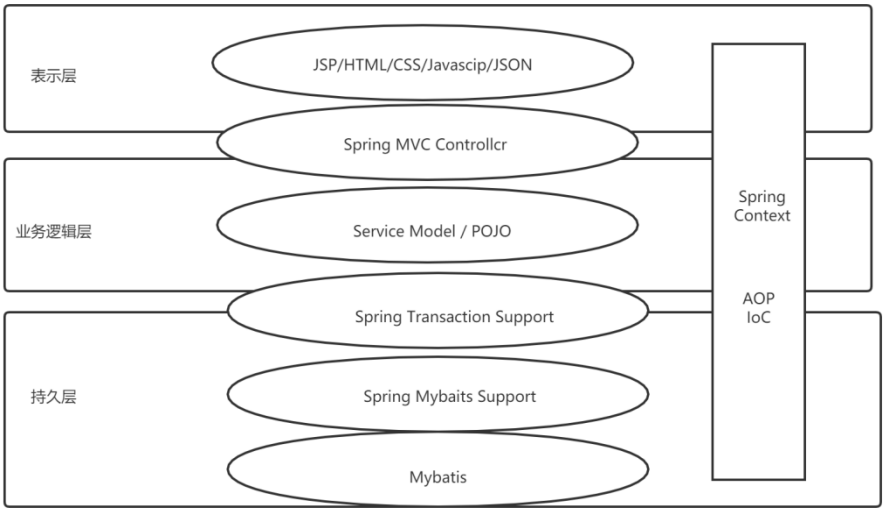

图 2-2 SSM 框架结构

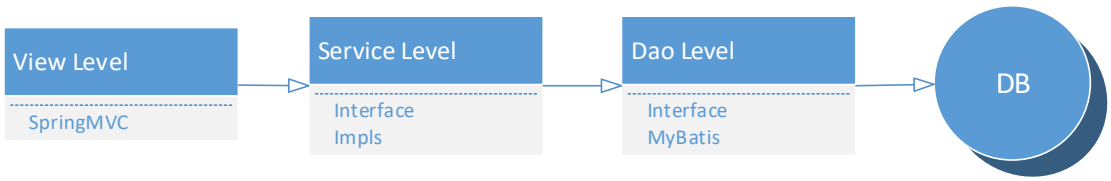

图 2-3 SSM 与 MVC 关系图

SSM 框架与 SSH 框架对比通过以下三个方面对比：

（1）MyBatis:SSM 框架中 MyBatis 主要负责对数据持久化的操作，相比 SSH 框架中的 HIbernate 的优势在于更加容易上手。

（2）SpringMVC：SSM 框架中 SpringMVC 负责实现 MVC 设计模式，作用于用户在 View 层进行通过 jsp，HTML 等方式进行交互，还有 Controller 层业务跳转。与 SSH 框架的 Structs 相比，优势在于更加的轻量灵活。

（3）Spring：主要负责管理 SpringMVC 和 MyBatis 相关对象的创建和依赖的注入。它被定义为一个轻量级的 IoC 容器，能使 SpringMVC 和 MyBatis 更好的工作。

2.2.1 Spring 介绍

Spring 为了解决企业级应用开发的复杂性而创建的[4]。在 Spring 之前，有一个重量级工具叫做 EJB，作用使让 Java Bean 之前进行有效的解耦，但是它过于臃肿，所以使用比较少，现在基本使用 Spring。Spring 不仅在服务端的开发使用，而且在测试性和松藕方面的使用效果也特别地表现好。一般我们主要掌握 Spring 四个方面的功能：

(1) IOC/DI:控制反转，指对一个对象的控制权反转；

(2) AOP：面向切面编程；

(3) 事务：利用 Aop 思想，简化事务配置；

(4) JdbcTemplate：Spring 利用 Aop 思想封装 JDBC 操作工具。

Spring 的优点：

(1) 通过 Spring 提供的 Ioc 容器控制对象之间的依赖关系，避免硬编码造成的过度耦合关系，并且可以使用户专注上层应用，不用编写底层需求代[15]；

(2) AOP 编程的支持；

(3) 支持灵活事务管理，通过声明式的方式，提高开发效率和质量；

(4) 对 Java EE API 的使用难度降低；

(5) 程序测试更加方便；

(6) 不排斥优秀的开源的框架，降低框架使用难度，方便集成各种优秀的框架。

2.2.2 SpringMVC 介绍

SpringMVC 是一种基于 Java 的实现了 web MVC 设计模式的请求驱动类型的轻量级 Web 框架[5]。它使用 MVC 架构模式的思想，将 Web 层进行职责解耦，有着简化我们日常的 Web 开发的目的。它的设计很好的把数据，业务与展现进行分离，与 Struts，Struts 等类似，并且设计是围绕着 DispatcherServlet 展开的，DispatcherServlet 的功能是负责请求发送特定的 handler，然后它通过可配置的 handler mappings、view resolution、locale 以及 theme resolution 来处理请求，把对应的数据转到对应的视图[13]。

SpringMVC 的优点：

(1) SpringMVC 框架提供了一套完整的组件；

(2) SpringMVC 是 Spring 框架的一部分，对 Spring 所提供的其他功能可以方便的利用；

(3) 提供前端控制器；

(4) 灵活性强，易于其他框架集成；

(5) 可自动绑定用户输入，并正确转换数据类型；

(6) 为了检验用户输入，内置了常见的检验器；

(7) 国际化的功能可以满足用户显示多国语言；

(8) 有多种视图的技术，而不仅仅局限于 JSP；

(9) 编辑后，基于 XML 的配置文件可以自动编译应用程序。

2.2.3 MyBatis 介绍

MyBatis 本是 apachede 的一个的开源项目 iBatis 迁移到 goodcode 上改名而来的，实质上改进了 iBtis。MyBatis 是一个优秀的持久成框架，它封装了 jdbc 的操作数据库的过程，简化开发者写对数据库进行相关操作代码的过程，使他们只需要关注 SQL 本身。MyBatis 把将要执行的各种 statement 通过 XML 或者注解的方式配置起来,并通过 Java 对象和 statement 中的 sql 进行映射生成最终执行的 sql 语句，最后由 MyBatis 框架执行 sql 获得的结果映射到 java 对象上并返回[6]。

MyBatis 的优点：

(1) 学习简单，上手容易；

(2) 消除了 JDBC 大量冗余代码，解决了手动开关连接；

(3) 可以更好与各种数据库兼容；

(4) 有分页插件/逆向工程等第三方软件；

(5) 在 xml 标签帮助下，可以编写动态 SQL 语句；

(6) 提供映射编辑，支持对象与数据库的 ORM 字段关系映射。

2.3 面向对象的方法与 UML

2.3.1 面向对象的方法介绍

面向对象方法，英文 Object-Oriented，简称 OO，自然地模拟了人类认识客观世界的方式，是当前主流开发软件方法之一，也是当前软件工程领域的一种重要的指导软件开

发的技术[7]。面向对象方法是一种建立在以对象为基础来分析、理解、构建、开发相应软件系统的方法，原因在于它把程序员面向编程的程序设计思想巧妙地应用到了软件分析与设计这个重要的过程中，它把软件开发内容比作一个是由对象组成的客观世界，所有需要完成的不同需求定义不同的对象，每个对象有属性和操作，把具有相似属性和操作的对象集合在一起定义为类，然后采用方法，继承，封装等面向对象的概念来完成需求。

面向对象方法的主要特点:

(1) 按照人类习惯的思维方法，对软件开发过程所有阶段进行综合考虑；

(2) 软件生存各阶段所使用的方法、技术具有高度的连续性；

(3) 软件开发各个阶段有机集成，有利于系统的稳定性；

(4) 具有良好的重用性。

2.3.2 统一软件开发过程-RUP

RUP（rational unified process）是一个面向对象且基于网络的程序开发方法论。它是以面向对象方法为基础的方法，RUP 坚持以用例驱动，以架构为中心，迭代和增量的开发方法[12]。

1.开发过程：

(1) 初始阶段：为建立商业案例并确定项目的边界，需要识别所有与系统交互的外部的实体，通过较高层次定义交互的特性。

(2) 细化阶段：分析问题领域，建立健全的体系结构基础，编制项目计划，避免中高风险元素，确定非功能需求，为项目建立支持的环境（开发案例，模板等），同时准备工具。

(3) 构造阶段：所有构件和应用程序被开发成产品，并且所有功能进行测试完毕。

(4) 交付阶段：通过几次迭代，结构问题，安装，可用性问题得到解决，确定软件对最终用户可以用。

2. 二维开发模型

RUP 软件开发生命周期是一个二维的软件开发模型。横轴通过时间组织，是过程展开的生命周期特征，体现开发过程的动态结构，用来描述它的术语主要包括周期、阶段、迭代和里程碑；纵轴以内容来组织为自然的逻辑活动，体现开发过程的静态结构，用来描述它的术语主要包括活动、产物、工作者和工作流[14]。如图 2-4。

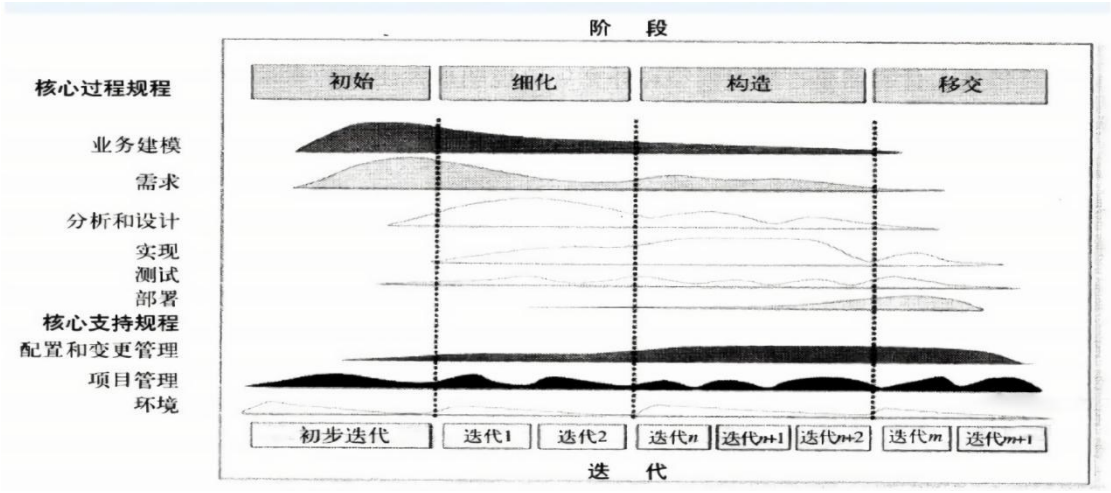

图 2-4 二维开发模型图

2.3.3 UML 建模

UML 全称 Unified Modeling Language，中文名统一建模语言，是一种利用图形符号直观地表示软件系统地建模语言[8]。在软件工程地开发中采用面向对象的技术就可以使用这些图形符号就行表示，并且 UML 可以描述从系统需求到系统测试的不同阶段。其中，用例图，活动图，时序图等用例模型可以用来在需求分析阶段来描述分析结论，在设计阶段是系统实现的准备需要考虑技术细节，所以这个阶段需要用到类图，包图，构件图等进行详细描述。UML 建模的一般流程：

(1) 分析阶段：模型的目的是捕获系统的需求，建立“现实世界”的类和协作的模型；

(2) 设计阶段：分析实际环境，把分析模型扩展为可行性方案；

(3) 实现阶段：通过设计模型完成实现代码的编写；

(4) 部署阶段：系统在物理结构中部署。

运用 UML 进行面向对象的系统分析时，首先要对项目进行需求调研，分析项目的业务流程图和数据流程图，以及项目中涉及的各级操作人员，识别出系统中的所有用例和角色；接着分析系统中各角色和用例间的联系，使用 UML 建模工具画出系统的用例图；勾画系统的概念层模型，借助 UML 建模工具描述概念层的类图和活动图。

UML 建模的优点：

(1) 统一标准，使面向对象的方法流派统一和提供了标准的模型元素定义与表示；

(2) 集中了面向对象技术领域的长处；

(3) 可以清晰表示可视化模型，描述能力强，适用于复杂系统建模；

(4) 系统建模语言，独立于过程；

(5) 建模表示简洁易懂，易于掌握和使用。

2.5 本章小结

本章介绍了开发中使用的MVC模式和系统开发使用的SSM框架基本内容及其三大框架Spring，SpringMVC，Mybatis的基本介绍和优点，以及系统分析设计使用面向对象的开发方法和UML建模语言介绍，为系统的正式设计开发垫下基础。

第三章 系统分析

3.1 系统需求概述

3.1.1 需求概述

企业的发展缺少不了人员之间的沟通，所以召开会议是不可少的，但是大部分中小企业还在采用手工化的方式管理预约会议，导致会议室资源冲突，会议室无法及时预订，预订情况无法及时通知等问题的出现。所以，为了规范会议室的使用，方便会议预约与通知，减少人工管理会议室的预订的时间，我设计并实现了一款会议室预约管理系统，通过它提高会议的管理水平和会议执行的效率。

本系统中的两种用户角色的职责，如表 3-1 所示。

表 3-1 角色表

角色	职责需求
管理员	会议室管理，人员管理，部门管理，及其预约会议的操作权限
普通用户	预约会议的权限

3.1.2 功能性需求

(1) 普通用户需求：

1) 普通用户只需要注册之后，经过管理的审批同意后，方可登录系统。然后，普通用户也可以成为会议的发起者，而不是由特殊角色发起会议，这个考虑到有些公司的领导喜欢安排会议的任务让下属去处理通知好人员和预订会议室的情况；
2) 普通用户也可以查看各个会议室的使用情况，且对自己发起的会议有撤销的功能；
3) 查看自己有那些会议需要参加的需求；
4) 为了方便联系参加人员，用户还需要把自己的一些联系方式注册进入系统，并且可以随时修改自己的个人信息。

(2) 管理员需求

1) 普通用户的需求；
2) 具有会议室的增删改查功能；

3) 对公司部门信息删除，修改，增加，对新注册的人员有审批；

4) 对已存在的人员有删除的需求。

(3) 人员模块管理的需求：

1) 管理员可以通过具体条件进行搜索人员信息，比如部门，名字；

2) 管理员可以管控用户的账号；

3) 用户对自己的个人信息可以修改。

(4) 部门模块管理的需求：管理员可以对部门进行增删改查。

(5) 会议室模块管理的需求：

1) 管理员会议室进行增删改查；

2) 会议室需要显示当前状态；

3) 用户可以查看会议室具体内容；

4) 用户可以查看会议的预订情况。

(6) 会议模块管理的需求：

1) 用户可以预约会议的功能；

2) 会议可以被撤销的功能；

3) 可以显示会议的具体内容；

4) 预订的会议会通知参会人员；

5) 可以通过具体条件收索会议信息；

6) 可以查看个人参加的会议和个人预订的会议。

3.1.3 非功能性需求

（1）系统正常运行，各个功能模块能够稳定使用，保障数据安全，满足业务的基本要求；

（2）系统界面简洁美观；

（3）对输入信息有提示，防止数据异常，应进行数据检查；

（4）严格权限访问控制，之后经过身份确认才能进行权限内的操作；

（5）适配于各种主流浏览器；

（6）有可维护性的需求，接到普通修改请求，能在短时间能进行维护；

（7）对于未来有系统功能可扩展性的需求。

3.1.4 总体功能结构图

通过功能需求，为了使用户更加清晰的了解系统业务流程，通过业务流程图展示整个系统的运行流程，如图 3-1 所示。

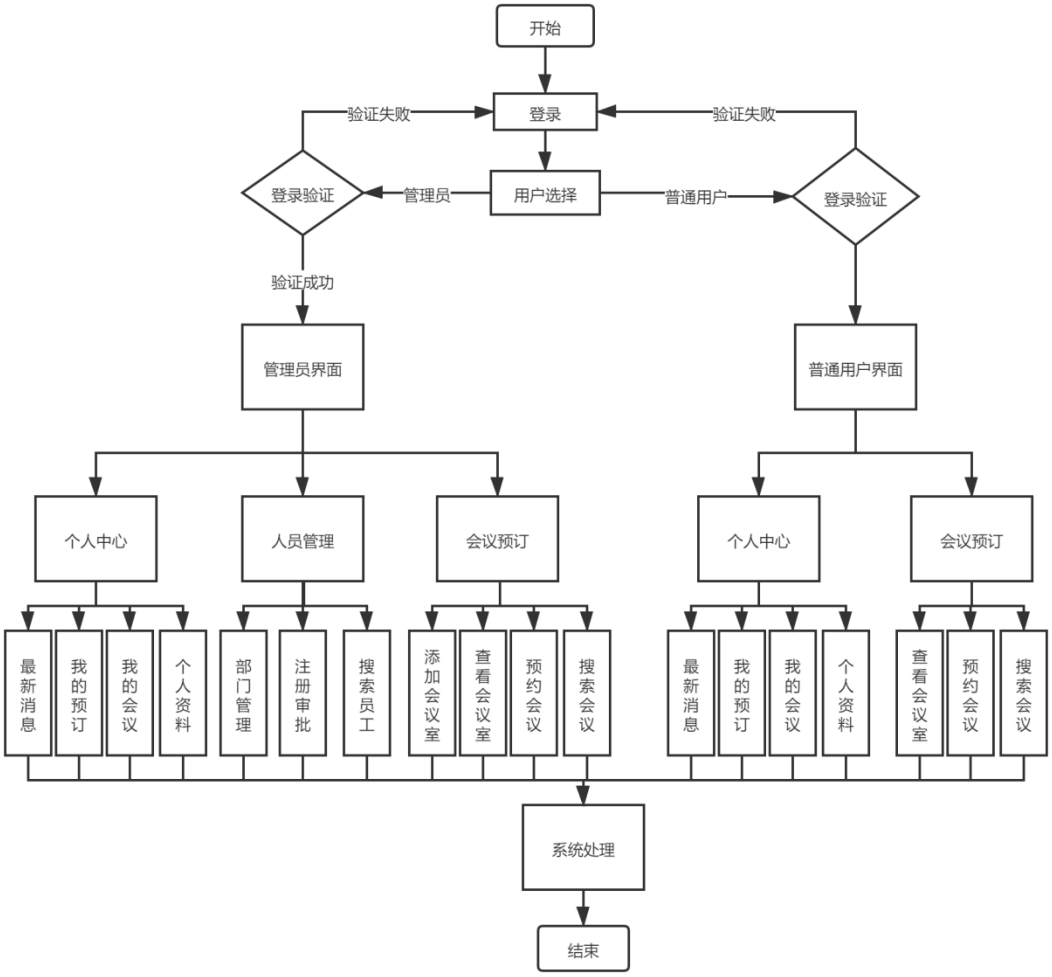

图 3-1 总体功能结构图

3.2 用例捕获

3.2.1 用例模型

用例模型就是描述一个系统做什么，也就是系统的功能，它是由若干个用例图组成，而用例图包含用例，参与者，用例之间的关系三种模型元素[9]。用例就是为了完成一个功能的完整操作描述，参与者就是使用用例功能的人或事物，用例之间的关系分为：

通信关系：参与者与用例之间的关系。

包含关系：一个用例使用另一个所提供的功能。

扩展关系：一个用例扩展到另一个用例所提供的功能。

3.2.2 系统用例

1. 通过系统需求的分析，可以确定该系统的参与者有两个：
   (1) 管理员：进行系统的维护功能，有添加，删除，修改特定功能；
   (2) 普通用户：预订会议，查看会议等功能。
2. 识别出参与者后，通过对需求的进一步分析可以确定系统核心用例进行捕获如下：
   (1) 人员管理用例捕获：本用例提供人员的添加、删除、修改、查看功能；
   (2) 部门管理用例捕获：本用例提供部门的添加、删除、修改、查看的功能；
   (3) 会议室管理用例捕获：本用例提供对会议室的添加、删除、修改、查看的功能；
   (4) 用户预约会议管理用例捕获：本用例提供会议预订的添加、删除、查看功能，并且还具有通知会议人员和检查会议室空闲状态的功能。
3. 识别出参与者和用例后，建立用例图：如图 3-2 所示：

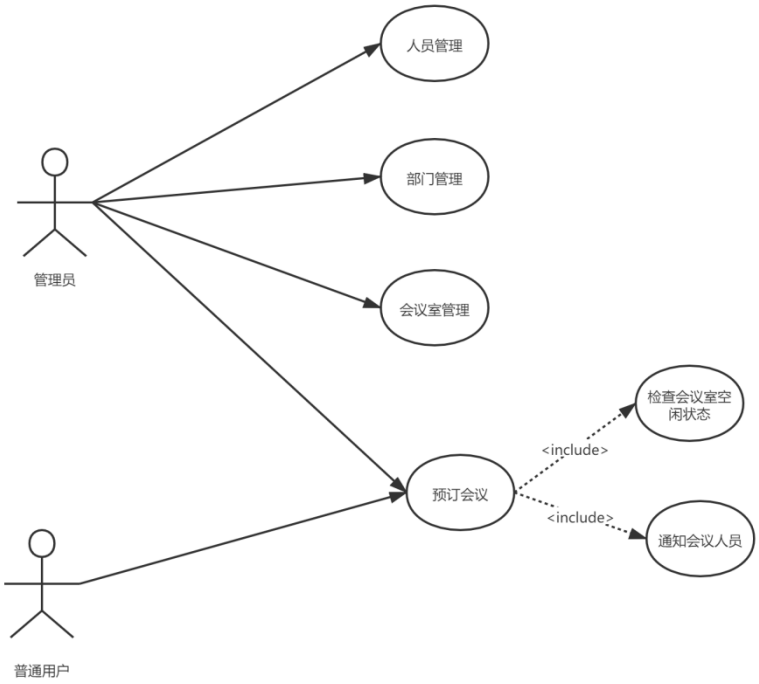

图 3-2 系统用例图

3.3 用例描述及时序图

3.2.1 文本事件流

用例的事件流是对完成用例行为所需的事件的描述。它描述系统应该做什么，而不是描述系统应该怎样做。

会议室管理系统的部分用例事件如下：

（一）用例名名称：用户登录

用例描述：用户输入账户和密码，系统开始身份验证

参与者：企业工作人员

前置条件：无

基本路径：

(1) 输入账号，密码

(2) 点击登录

(3) 验证用户权限，进入主界面

（二）用例名名称：预订会议

用例描述：用户进入预订界面，输入正确预订信息，完成会议预订。

参与者：企业工作人员

前置条件：用户登录成功

基本路径：

(1) 点击预订会议

(2) 输入会议信息

(3) 确认会议

(4) 跳转到搜索会议

主事件流：

(1) 用户登录系统，点开预约会议，用例开始。

(2) 用户进入预订界面，提示输入预订信息。

(3) 用户输入会议名称，会议人数，会议室，会议人员，会议室描述，会议时间。

(4) 填写信息完成，点击确认预订。

(5) 预订会议成功之后，该会议信息就会出现在所有参会人员的个人中心的最新通知和我的会议页面上，及时通知。

(6) 用户预订完毕便会跳转到搜索会议界面，显示刚才自己预订的会议。

(7) 用例结束。

异常事件流：

(1) 如果选择的时间段，该会议室被占，则预订失败。

(2) 如果用户预订时没有输入会议名，会议时间，参会人员信息，系统会提醒用户重新预订。

3.2.2 活动图

该活动图是描述查看会议室信息的用例如图 3-3 所示，用户在发起会议室查询请求，然后显示会议室，在查看某一个会议室的具体信息，其中普通用户和管理员用户有不同的功能显示。文本事件流显示的信息大致相同，其中，实心黑色圆点作为开始节点，平滑的圆角矩形作为动作的状态，带箭头的直线表示动作之间的转换，菱形代表选择条件分支判断，黑色粗线代表分叉与汇合，圆圈内部的实心黑色圆点代表活动结束。

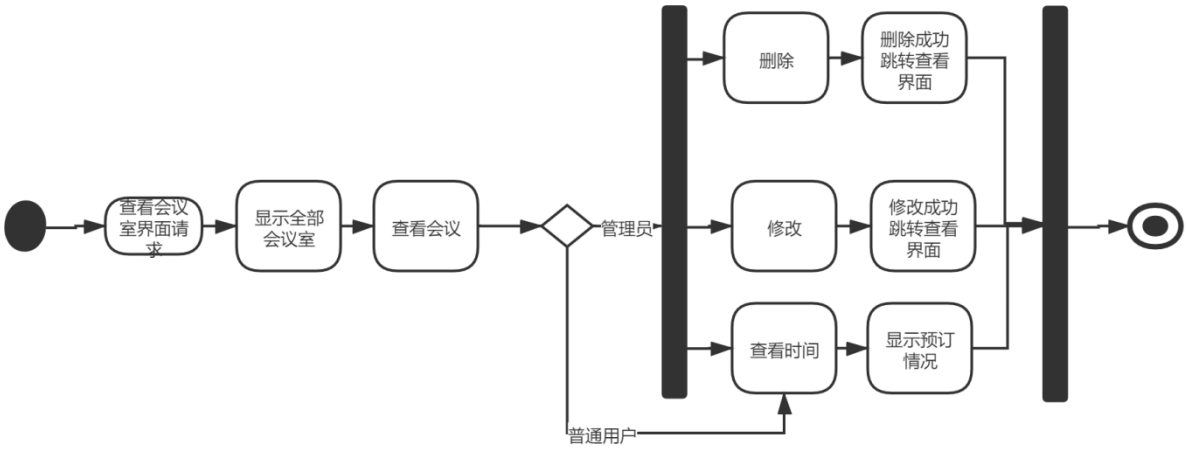

图 3-3 查看会议室活动图

3.2.3 时序图

为了更加清晰直观的显示会议预约的流程，我采用时序图的方式，再次描述用户预约会议的过程，从填写会议预订信息，判断预订信息的正确，再预订成功之后跳转到显示界面的过程，如图 3-4 所示。

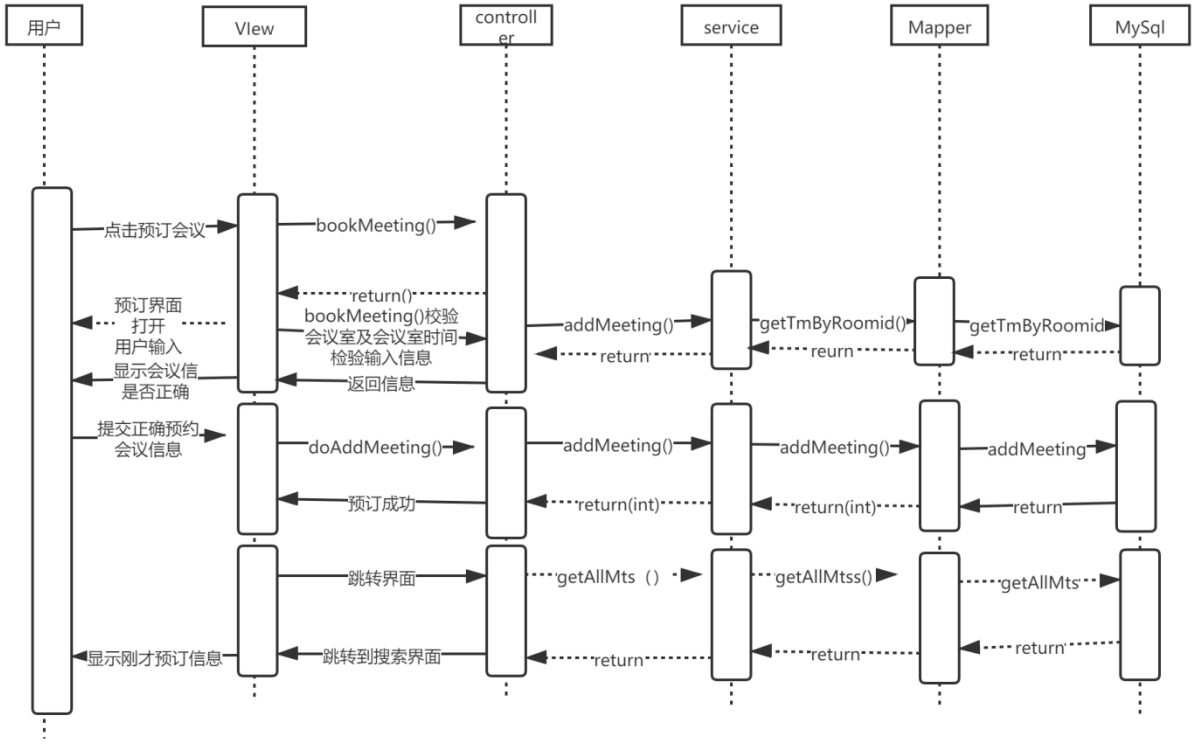

图 3-4 会议预约时序图

3.4 本章小结

本章对系统的功能性需求和非功能性需求进行概述，通过业务流程图直观展示了需求。再描述了用例的概念，通过文本事件流和时序图对系统的用例捕获进行了分析与描述。

第四章 系统设计

4.1 系统设计概述

4.1.1 主要任务和内容

主要任务通过系统需求分析阶段产生的文档资料，采用正确的方式确定实现会议室管理系统需要哪几种模块组合，并且确定这几个模块之间是什么关系来构成该系统的内部结构，还有使用什么工具或者技术来实现对实现成果的表示，以及对系统所有的数据库进行如何的设计等。

主要内容通过类图、包图以及数据库的 E-R 图方式，描述系统每个部分的设计以及系统内部结构的详细设计和对使用的方法的描述，以及数据库各表之间的关系和数据字典的展示。

4.1.2 系统设计的原则

在满足业务需要、方便、安全、准确的前提，还需要满足系统的科学化，合理化，经济化的要求，充分利用现有的软硬件设备和开发技术，达到可靠的性价比，所以系统设计应满足两种原则：

1.非功能需求原则：

(1) 完备性：功能齐全完备，满足用户的需求；

(2) 系统性：空间数据和系统属性有机融合在一起，各种参数可以互相传输；

(3) 可靠性：系统运行稳定安全，数据精确完整；

(4) 实用性：系统数据组织灵活，可以满足不同应用分析需求；

(5) 可扩充性：考虑未来技术的不断发展对系统的影响，应使用模块化的设计，使系统具有可扩展的功能。

(6) 易操作性：计算机的技术都朝着方便用户使用的方向发展，节约用户时间，方便用户快速对功能的使用和掌握，所以为了使系统使用效率提高，需要具有易操作。

2.开发设计原则：

(1) 开闭原则：尽量不修改原代码的情况下进行扩展；

(2) 单一职责原则：一个类负责一个功能领域的相应职责；

(3) 依赖倒转原则：实现编程不重要，针对接口编程才重要；

(4) 接口隔离原则：每个接口对应每个功能，而不是所有功能都调用一个接口；

(5) 里氏代换原则：引用父类的地方都可以透明的使用其子类的对象；

(6) 迪米特法制：软件实体间尽量小地产生相互作用。

4.2 系统总设计

4.2.1 包图

设计实现一个比较复杂的系统时，会产生大量的类和接口，如果把他们都放在一起使我们维护的时候不易区分和查询，所以就需要把相同类型的类放在同一个包中，并且通过包图来描述各个包以及包之间的关系，展示出系统模块与模块之间的依赖关系，而且包图是维护和控制系统总体结构的重要建模工具[10]。

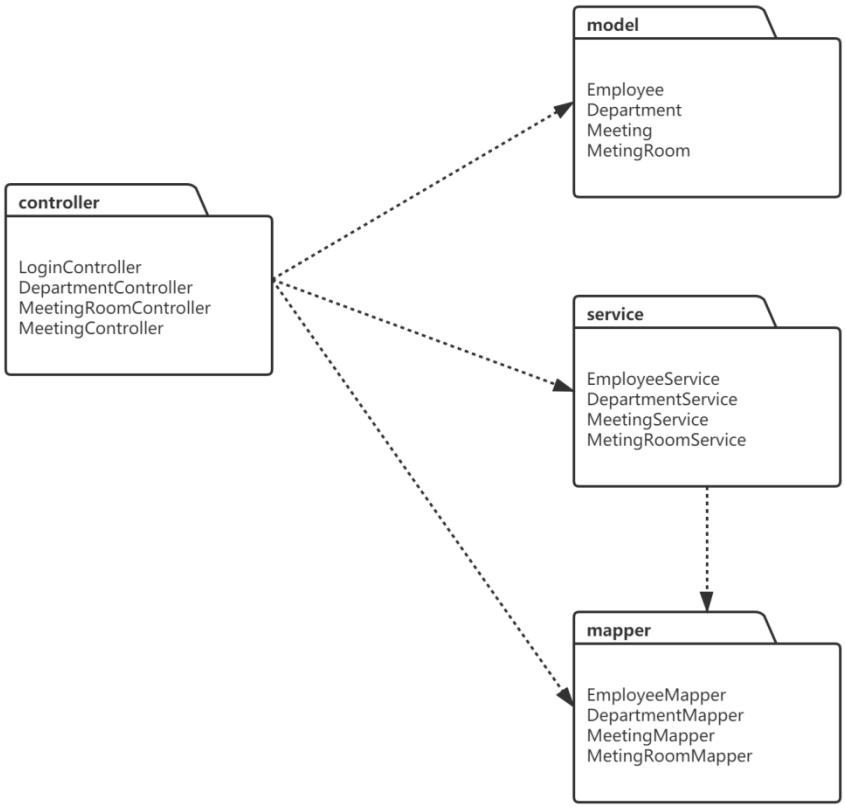

图 4-1 系统包图

controller 包是处理前端页面请求的控制类的集合；

model 包是系统所有实体的实体类集合；

service 包是所有功能业务逻辑实现的实现类；

mapper 包是系统数据库与 dao 层持久化操作的接口类，还有一些配置好相应 sql 语句的 xml 文件。

4.2.2 类图

1、登录模块设计：

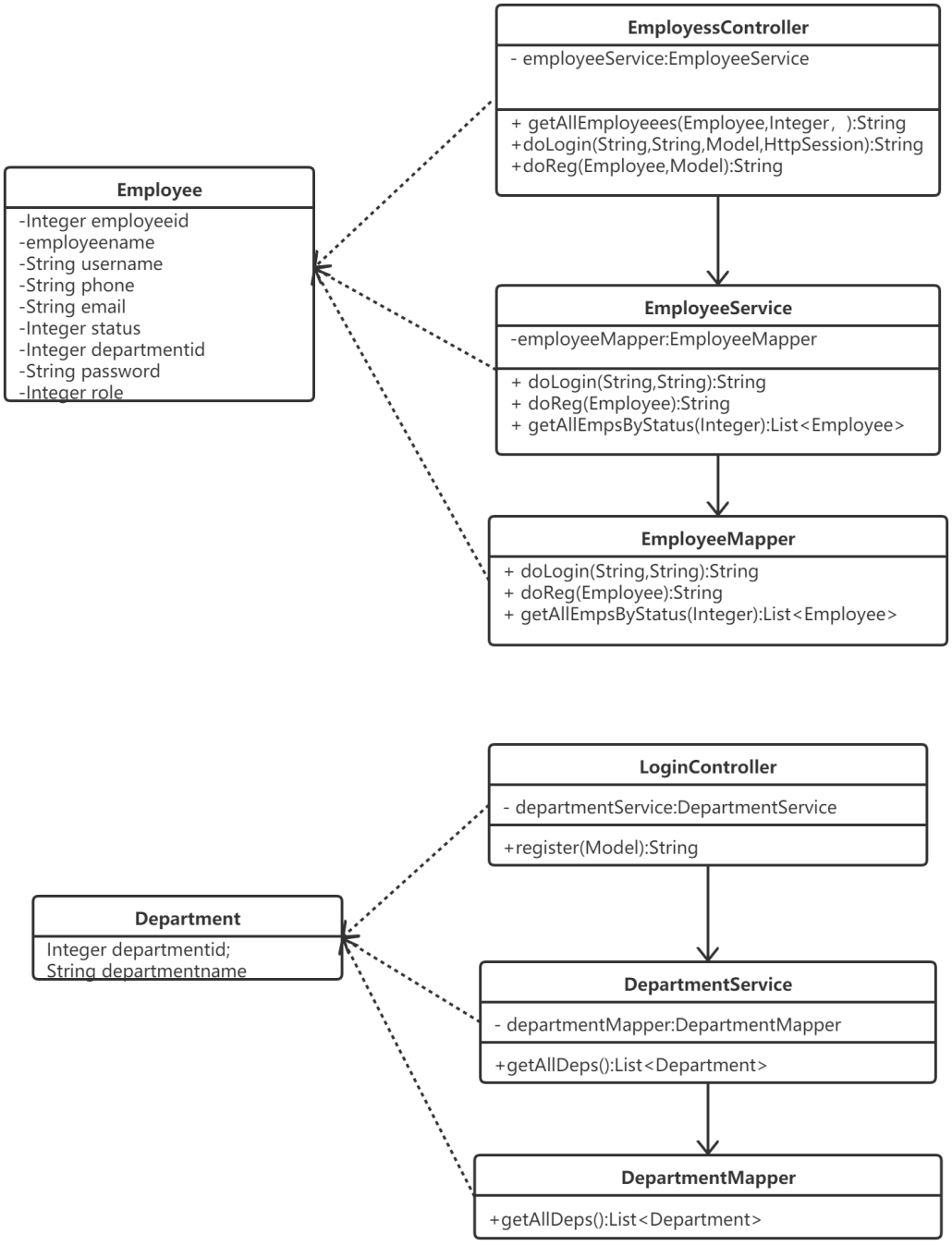

图 4-2 登录模块类图

(1) LoginController 类是为了处理用户登录时相关操作的请求，其中主要方法如下：

1) Login:用户请求登录页面请求；

2) doLogin：用户进行登录操作请求；

3) register：用户进入注册界面操作请求；

4) doReg：用户进行注册操作请求。

(2) EmployeeService 类为了解决用户登录时相关逻辑，其中的主要方法如下：

1) doLogin：用户进行登录操作；

2) doReg：用户进行注册操作；

(3) EmployeeMapper 接口类主要实现 Dao 层与数据库持久化操作，通过 EmployeeMapper.xml 中映射好的 SQL 语句对 employee 表进行基本操作获取结果。

(4) DepartmentService 类用于处理部门管理于与登录注册时相关逻辑：getAllDeps：获取所有部门操作。

(5) DepartmentMApper 接口主要实现 Dao 层与数据库持久化操作，通过 DepartmentMApper.xml 中映射好的 SQL 语句对 department 表进行基本操作获取结果。

2、人员管理模块设计：

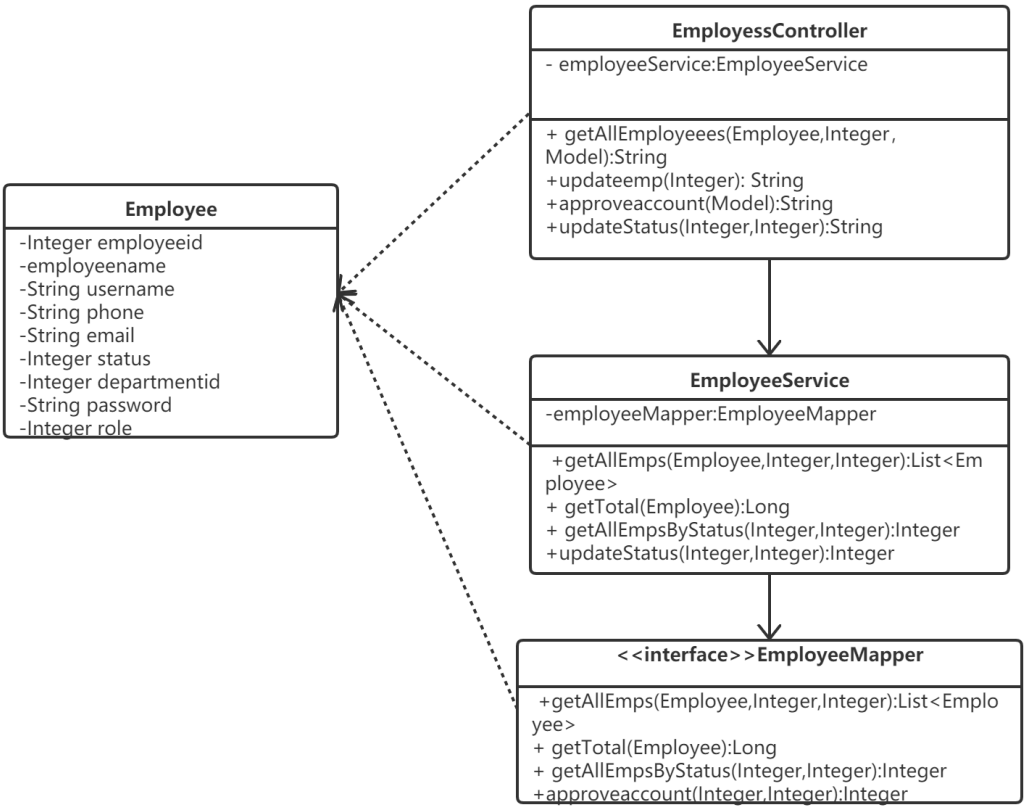

图 4-3 人员管理类图

(1) EmployeeController 类为了实现管理员处理与员工信息相关操作的请求，其中关键方法如下：

1) getAllEmployeees:获取所有员工请求；

2) updateemp：更新员工状态请求；

3) approveaccount：获取新注册员工请求；

4) updateStatus：更新新注册员工状态请求；

(2) EmployeeService 类用于管理员管理员工数据时相关逻辑，其中的主要方法如下：

1) getAllEmployeees:获取所有员工操作；

2) updateemp：更新员工状态操作；

3) approveaccount：获取新注册员工操作；

4) updateStatus：更新新注册员工状态操作；

(3) EmployeeMapper 接口主要实现 Dao 层与数据库持久化操作，通过 EmployeeMapper.xml 中映射好的 SQL 语句对 employee 表进行基本操作获取结果。

3、部门管理设计：

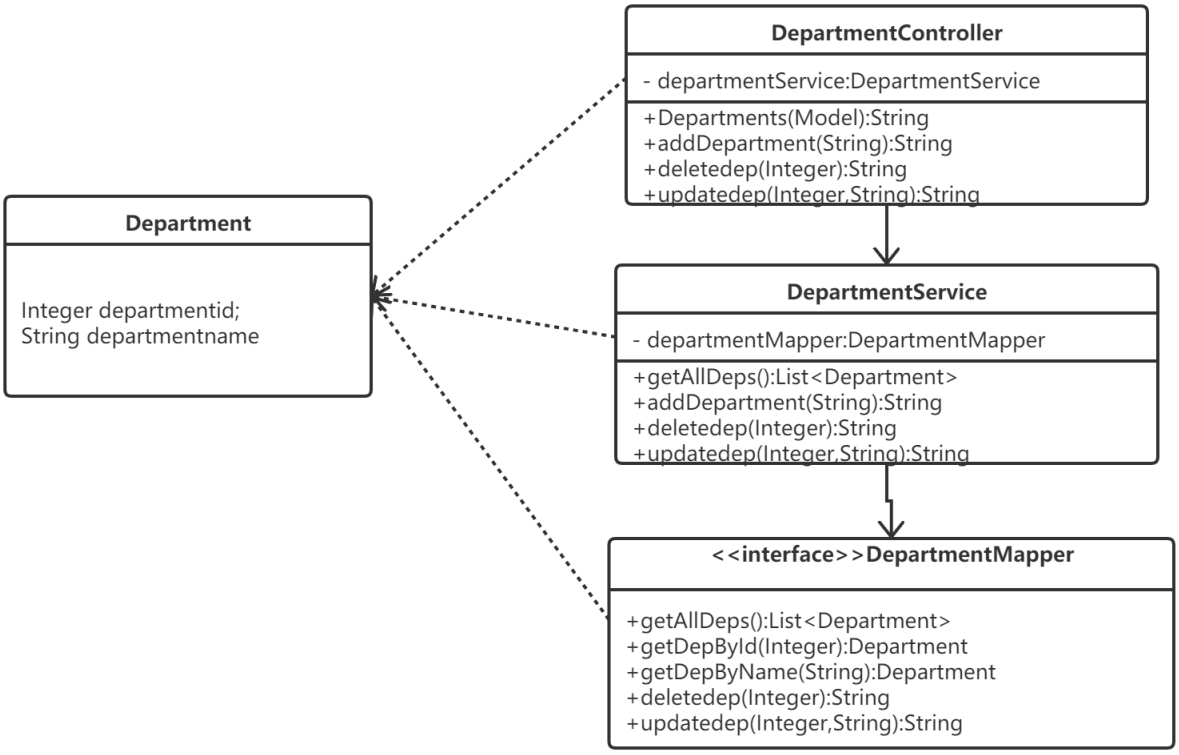

图 4-4 部门管理类图

(1) DepartmentController 类获取前端界面管理员管理部门信息相关操作的请求，其中关键方法如下：

1) Department:查看部门页面请求；

2) addDepartment：添加部门请求；

3) deletedep：删除部门请求；

4) updatedep：更新部门请求。

(2) DepartmentService 类用于管理员管理部门信息时相关逻辑，其中的关键方法如下：

1) Department:查看部门页面操作；

2) addDepartment：添加会议室操作；

3) deletedep：删除会议室操作；

4) updatedep：更新会议室操作。

(3) DepartmentMapper 接口主要实现 Dao 层与数据库持久化操作，通过 DepartmentMapper.xml 中映射好的 SQL 语句对 department 表进行基本操作获取结果。

4、会议室管理设计：

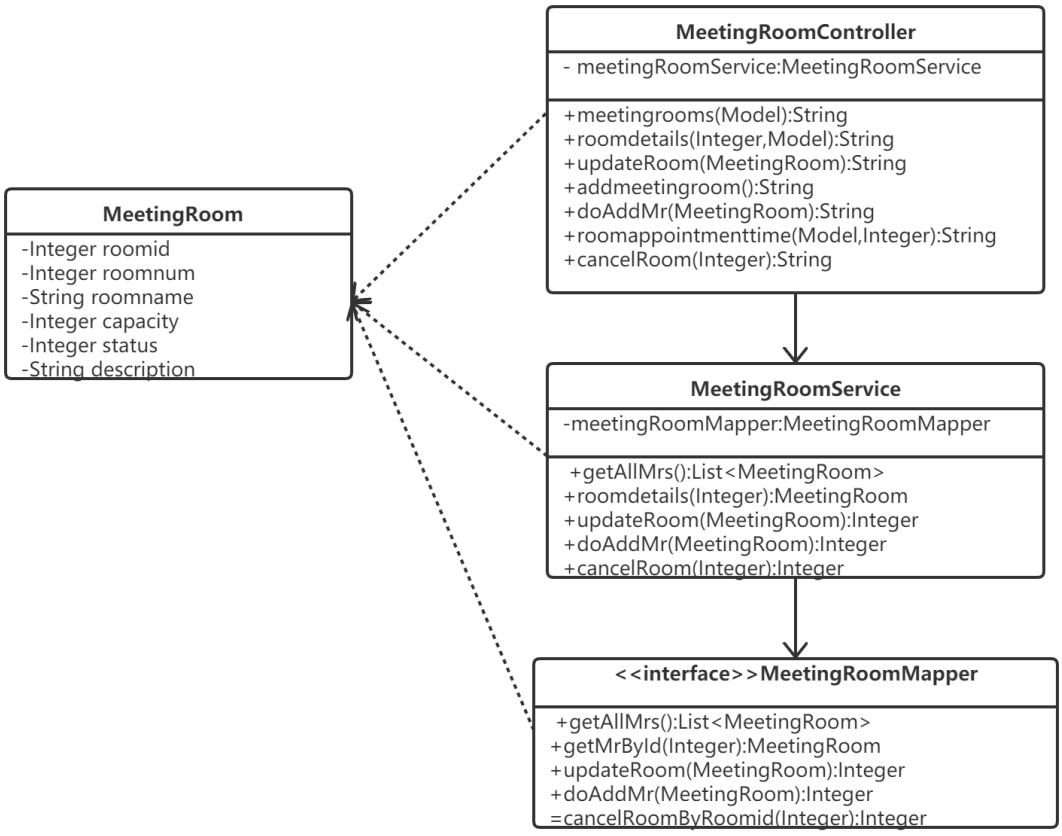

图 4-5 会议室管理类图

(1) MeetingRoomController 类获取前端界面用户查看会议室和管理员管理会议室信息相关操作的请求，其中主要方法如下：

1) Meetingrooms：查看会议室页面请求；

2) roomdetails：查看具体会议室内容请求；

3) updateRoom：管理员更新会议室请求；

4) addmeetingroom：管理员进入添加会议室页面请求；

5) doAddMr：管理员添加会议室操作请求；

6) cancelRoom:管理员关闭会议室操作请求。

(2) MeetingRoomService 类用于查看会议室和管理员管理会议室信息时相关逻辑，其中的主要方法如下：

1) getAllMr：获取全部会议室操作；

2) roomdetails：查看具体会议室内容操作；

3) updateRoom：管理员更新会议室操作；

4) doAddMr：管理员添加会议室操作操作；

5) cancelRoom:管理员关闭会议室操作操作。

(3) MeetingRoomMapper 接口主要实现 Dao 层与数据库持久化操作，通过 MeetingRoom Mapper.xml 中映射好的 SQL 语句对 meetingroom 表进行基本操作获取结果。

5、会议管理设计：

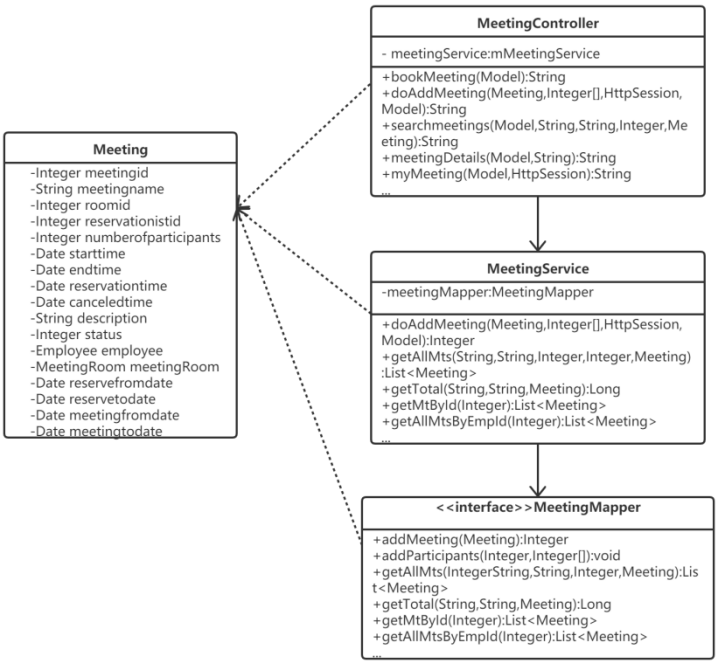

图 4-6 预订会议类图

(1) MeetingController 类获取前端界面用户预订会议相关操作的请求，其中主要方法如下：

1) bookMeeting：进入预订会议页面请求；

2) doAddMeeting：预订会议操作请求；

3) searchmeeting：搜索会议请求；

4) meetingDetails：查看会议具体内容请求；

5) myMeeting：查看我的会议请求；

(2) MeetingService 类用于处理用户预订会议时相关逻辑，其中的主要方法有：

1) doAddMeeting：预订会议操作；

2) getAllMts：获取全部会议操作；

3) getTotal：获取会议数量操作；

4) getMtById：通过会议 id 获取会议信息操作；

5) getAllMtsByEmpId：通过员工 id 获取会议信息。

(3) MeetingMapper 接口主要实现 Dao 层与数据库持久化操作，通过 MeetingMapper.xml 中映射好的 SQL 语句对 meeting 表进行基本操作获取结果。

个人中心模块，就是调用前面几个模块的类中的方法实现，内容基本重复，就不再独立展示。

4.3 数据库设计

4.3.1 数据库 E-R 图表

一个完整的系统，后台的数据都是存放在数据库中，以表的形式纯在，为了更容易理解各个表的关系，就需要使用概念模型对其建模描述，其中表示的方法有很多，最为常用的就是 E-R 模型。通过上文的系统需求分析阶段设计出本系统数据库的 E-R 图，如图 4-7 所示。

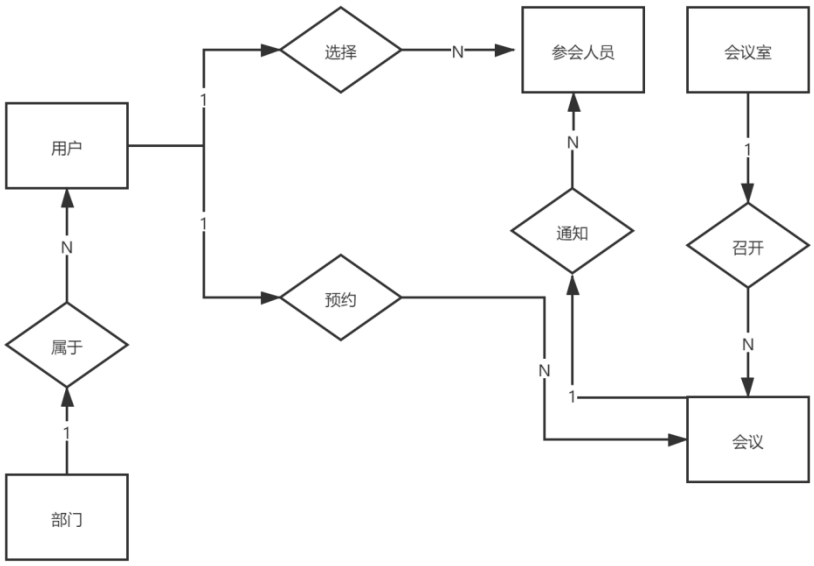

图 4-7 数据库的 E-R 图

4.3.2 数据库表

在系统设计过程中，设计数据库的任务显而易见的重要呀，合理的设计可以使系统开发逻辑清晰，也可以说系统的成败取决于数据库的设计，即数据是设计的基础。如果前期设计时，使数据库不够合理，不够完善，这会给系统数据的开发和后期维护带来严重的问题。

通过系统需求描述和 E-R 图的设计，设计了 6 张表，各表的详情如下

（1）员工表（employee)，记录企业员工的员工信息，employeeid 是主键，employeename 是员工登录的账号，status 是代表次员工账号的状态是注册审核中，还有注册完毕或者被管理员关闭了账号，role 是代表此账号是否是管理员或者普通用户，其他字段则是员工的基本信息，如表 4-1 所示。

表 4-1 员工表（employee）

字段名	数据类型	说明
employeeid	int	员工号
employeename	varchar(14)	员工账户名
username	varchar(20)	员工名
phone	varchar(20)	电话
email	varchar(100)	邮箱
status	varchar(20)	账号状态
departmentid	int	部门 id
password	varchar(50)	密码
role	varchar(12)	角色

（2）部门表（department），记录部门的名字和部门的编号，deparmentid 是主键，如表 4-2 所示。

表 4-2 部门表（department）

字段名	数据类型	说明
departmentid	int	部门 id
departmentname	varchar(20)	部门名

（3）会议室表（meetingroom），此表记录会议室的相关信息，其中 roomid 是主键，status 代表当前会议室的使用状态，如表 4-3 所示。

表 4-3 会议室表（meetingroom）

字段名	数据类型	说明
roomid	int	会议室编号
roomnum	int	会议室门牌号
roomname	varchar(20)	会议室名字
capacity	int	会议容纳的人数
status	varchar(20)	会议室的状态
description	varchar(200)	会议室的描述

（4）会议表（meeting），此表记录会议的相关信息，meetingid 是主键，其中 roomid 是会议召开的会议室，reservationistid 是预订会议的人员，如表 4-4 所示。

表 4-4 会议表（meeting）

字段名	数据类型	说明
meetingid	int	会议 id
meetingname	varchar(20)	会议名称
roomid	int	会议室 id
reservationistid	int	预订人 id
numberofparticipants	int	参加会议人数
starttime	datetime	开始时间
endtime	datetime	结束时间
reservationtime	datetime	预订时间
canceledtime	datetime	关闭时间
description	varchar(200)	会议描述
status	varchar(20)	会议状态

（5）参会人员表（meetingparticipants），此表记录每个会议对于哪些员工参加，如表 4-5 所示。

表 4-5 参会人员表（meetingparticipants）

字段名	数据类型	说明
meetingid	int	会议 id
employeeid	int	员工 id

（6）会议取消表（cancelmeeting），此表是记录被取消的会议数据，记录会议取消

人，取消的原因，如表 4-6 所示。

表 4-6 会议取消表（cancelmeeting）

字段名	数据类型	说明
meetingid	int	会议 id
cancelreason	varchar(101)	会议取消的原因
cancelerid	int	取消人 id

4.4 本章小结

本章描述了系统设计概念，具体说明了系统的主要任务，主要内容，设计原则及模块的总体设计图，并通过类图设计和包图设计两个内容描述了系统设计的内容。通过 E-R 的展示，描述了系统需要的数据库的设计，并给出了各张表的数据字典。

第五章 系统实现

5.1 项目系统层次设计

会议室预约管理系项目层次结构如图 5-1 所示。

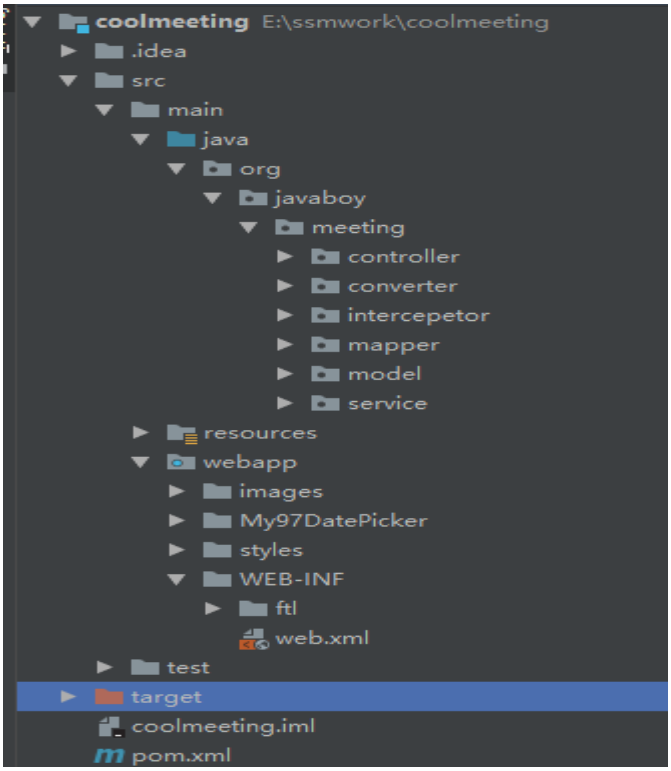

图 5-1 项目层次结构图

mapper 层：提供处理数据库操作的接口，映射数据操作方法的 SQL 语句的.xml 文件。

service 层：存放着整个系统的业务逻辑实现的方法,使用各种功能的实现方法。

controlle 层：负责业务模块的控制,接受所有前端传过来的业务参数,并调用 service 层进行业务逻辑进行处理,得到返回结果并返回给前端。

model 包:存放系统数据库中所有实体类属性。

intercepetor 包：则用于存放系统使用的一些工具类。

webapp 层：存放前端页面相关设计及参数。

5.2 SSM 框架搭建

本系统的实现采用 SSM 框架结构，下图 5-2 所示，是 SSM 框架整合 Spring，SpringMVC，MyBatis 的配置，过程相关介绍如下：

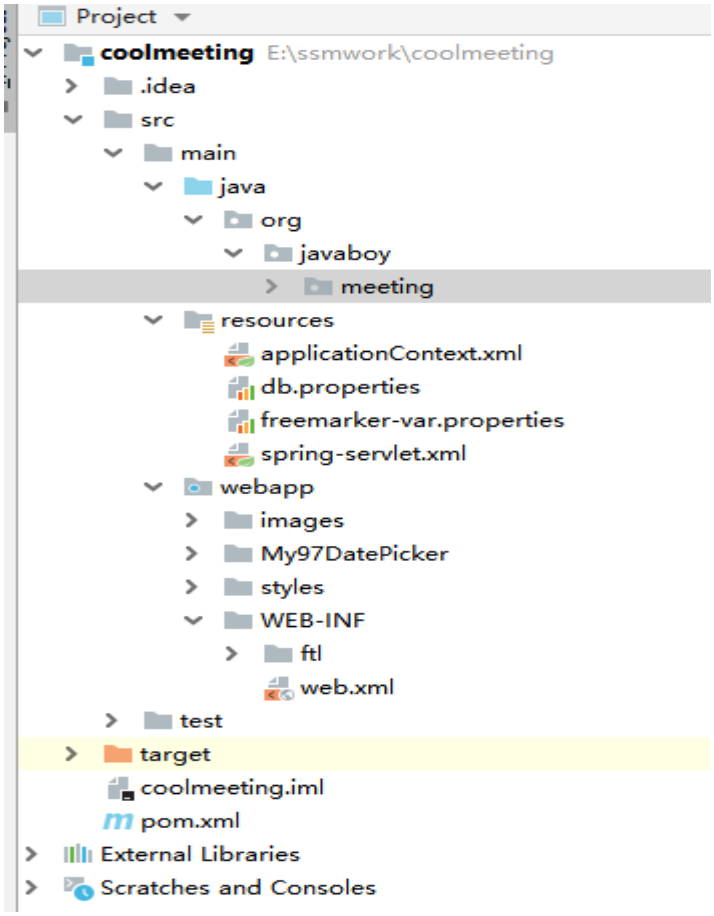

图 5-2 整合框架结构

上图是在 idea 工具中创建了一个依赖 maven 仓库的 web 项目结构，其中 pom.xml 文件中配置了使用 Spring,Springmvc,Mybatis 三大框架工具及数据连接池等其他相关技术的依赖，然后在 applicationContext.xml 文件中配置系统过滤器和扫描除了 controlle 包的包扫描工具及 Mybatis 的配置。再在 spring-servlet.xml 文件，配置视图解析器，模板基本属性和只扫描 controlle 包的包扫描器。之后再在 web.xml 文件中配置 applicationContext.xml 文件和 spring-servlet.xml 文件的使用，再对字符编码和文件解析器等配置等。通过这文件配置，本系统所使用的 ssm 框架已基本完成。

5.3 系统核心功能模块实现

5.3.1 用户权限判断

登录开始，用户输入账号和密码后，系统通过前端获取用户输入信息，然后返回给后台 LoginController 并调用 doLogin()方法，先调用 EmployeeService 的 doLogin 方法去调用 EmployeeMapper 中 LoadEmployeeUsername 方法使用 Dao 层的 EmployeeMappe.xml 查询用户信息，再 EmployeeService 的 doLogin 方法中判断用户时候存在，最后在

LoginController 的 doLogin 方法中判断该账号登录情况，返回用户信息跟登录情况信息给前端界面。

LoginController 的 doLogin():

```
 @PostMapping("/all/dologin")
    public String doLogin(String username , String password, Model model , HttpSession httpSession){
        Employee employee=employeeService.doLogin(username,password);
        if (employee==null){
            model.addAttribute("error","用户名或者密码输入错误,登录失败");
            return  "forward:/";/*登录成功时重定向，登录失败时页面跳转，即可重定向也可以页面跳转*/
        }
        else {
            if(employee.getStatus()==0){
                model.addAttribute("error","用户待审批");
                return "forward:/";
            } else if (employee.getStatus()==2) {
                model.addAttribute("error","用户审批未通过");
                return "forward:/";
            }else {
    .....
```

EmployeeService 中的 doLogin():

```
public Employee doLogin(String username, String password) {
        Employee employee=employeeMapper.LoadEmployeeUsername(username);
        if (employee==null||!employee.getPassword().equals(password)){
            return null;
        }
        return employee;
    }
```

EmployeeMappe.xml：

```
    <select id="LoadEmployeeUsername" resultType="org.javaboy.meeting.model.Employee">
        select * from employee where username= #{username};
    </select>
```

登录成功之后，leftMenu.ftl 通过获取到的用户数据，提取用户的权限字段信息，判断该用户是管理员还是普通用户，具体判断方法是判断 role 字段值是否等于 2，然后显示相关页面信息。如果未登录成功，login.ftl 获取未登录成功的原因，通过 LoginController 的 doLogin 方法中 model 对象，把失败原因加入对象中，前端再获取它保存的内容，显示未成功的原因。

leftMenu.ftl：

```
.....
<div class="sidebar-menugroup">
    <div class="sidebar-grouptitle">会议预定</div>
    <ul class="sidebar-menu">
        <#if currentuser??&& (currentuser.role==2)>
            <li class="sidebar-menuitem"><a href="/admin/addmeetingroom">添加会议室</a></li>
        </#if>
        <li class="sidebar-menuitem"><a href="/ordinary/meetingrooms">查看会议室</a></li>
        <li class="sidebar-menuitem"><a href="/ordinary/bookMeeting">预定会议</a></li>
        <li class="sidebar-menuitem"><a href="/ordinary/searchmeetings">搜索会议</a></li>
    </ul>
</div>.....
```

5.3.2 预约会议功能实现

（1）查看会议室

点击会议预约菜单中的查看会议室，就可以了解你想预约相关会议室的预约情况，，点击某一个会议室，前端界面便会获取该会议室的编号给到后端，通过 MeetingRoomController 中 roomappointmenttime 方法使用 Dao 层中 MeetingRoomController.xml 的 getMrByRoomid（）查询相关该会议室预约时间的状况，并且把信息通过 roomappointmenttime 加入 model 对象中，前端页面 roomappointmenttime.ftl 调用该对象，并显示预订时间的情况。

Roomappointmenttime():

```
    @RequestMapping("/ordinary/roomappointmenttime")
     public String roomappointmenttime(Model model,Integer roomid){
          model.addAttribute("rats",meetingRoomService.getMrByRoomid(roomid));
          return "roomappointmenttime";
     }
```

MeetingRoomController.xml 的 getMrByRoomid()：

```
<select id="getMtrByRoomid" resultType="org.javaboy.meeting.model.MeetingRoom">
    select * from meetingroom where roomid=#{roomid};
</select>
```

Roomappointmenttime.ftl:

```
.....
<table class="listtable">
                        <caption>
                            未来 7 天该会议室的预约时间:
                        </caption>
```

```
                    <tr class="listheader">
                        <th>起始时间</th>
                        <th>结束时间</th>
                    </tr>
                    <#if rats??>
                        <#list rats as rat>
                            <tr>
                                <td>${rat.starttime?string('yyyy-MM-dd HH:mm:ss')}</td>
                                <td>${rat.endtime?string('yyyy-MM-dd HH:mm:ss')}</td>
                            </tr>
                        </#list>
                    </#if>
.....
```

（2）会议预订

用户登录成功之后，在预约会议菜单中，点击会议预约，进入填写界面。用户需要填写会议室的名称，参会人数，预订开始时间，预订结束时间，选择会议室，填写会议说明，选择参会人员。如图 5-3 所示。

图 5-3 会议预约界面

在填写信息时，如果用户不填写会议的名称，就提交申请，系统会提醒用户输入，如果选择某一个会议室的使用时间跟别人预订的时间冲突，那么系统也会提醒用户重新选择，最后用户没有选择参会人员，申请不会成功，系统会提醒用户重新选择参会人员。

每一次提交失败之后，都会返回当前预约界面，并且保留原来输入的信息。

具体代码实现，前端页面获取输入内容，然后提交到后端 MeetingController 的 doAddMeeting 方法中处理，当获取会议名称，预计开始时间，预计结束时间，参会人员的这几个中一个或者几个为空的话，doAddMeeting 就会进行判断对应的错误信息，加入到 model 对象中，返回到前端页面进行显示。当用户选择会议进行的时间时，doAddMeeting 方法会去调用 dao 层中 MeetingMapper.xml 的 getTmByRoomid 查询方法，返回查询内容到 MeetingService 的 addMeeting 方法中判断，用户选择的时间段是否已经被预定了，如果被预订了就就会返回一个 result 值为 0，给 MeetingController 的 doAddMeeting 中，判断对应的错误提示。

没有错误之后，用户点击提交，后端通过以上判断后，MeetingService 的 addMeeting 调用 MeetingMapper 接口的 addMeeting 方法使用 dao 层中 MeetingMapper.xml 的 addMeeting 增加方法增加会议数据到数据库中，然后继续使用 MeetingMapper 接口的 addParticipants 方法把参会人员与该会议一一对应加入参会人员表（meetingparticipants）中。

MeetingController 的 doAddMeeting():

```
 public   String doAddMeeting(Meeting meeting, Integer[] mps, HttpSession session,Model model) {
        Employee currentuser = (Employee) session.getAttribute("currentuser");
meeting.setReservationistid(currentuser.getEmployeeid());
        if(meeting.getMeetingname().equals("")){

            model.addAttribute("error1","会议名为空");
            return "forward:/ordinary/bookMeeting";
        }
        else if (meeting.getStarttime()==null){
            model.addAttribute("error1","会议开始时间为空");
            return "forward:/ordinary/bookMeeting";
        }
        else if(meeting.getEndtime()==null){
            model.addAttribute("error1","会议结束时间为空");
            return "forward:/ordinary/bookMeeting";
        }
        else if(mps==null){
    .......
```

MeetingService 的 doAddMeeting()：

```
    public Integer addMeeting(Meeting meeting, Integer[] mps) {
        meeting.setReservationtime(new Date());
        meeting.setCanceledtime(new Date());
        meeting.setStatus(0);
        Integer result=1;
```

```
        List<Meeting> meeting1 = meetingMapper.getTmByRoomid(meeting.getRoomid());
        for (int i = 0; i < meeting1.size(); i++) {
            if ((meeting.getStarttime().getTime()<meeting1.get(i).getStarttime().getTime() &&
meeting.getEndtime().getTime()<meeting1.get(i).getStarttime().getTime())
                    || (meeting.getStarttime().getTime()>meeting1.get(i).getEndtime().getTime())){
                result=1;
            } else {
    .......
```

MeetingMapper.xml：

```
    ......
        insert into meeting (meetingname,roomid,reservationistid,numberofparticipants,starttime,endtime,r
eservationtime,canceledtime,description,status) values (#{meetingname},#{roomid},#{reservationistid},#{nu
mberofparticipants},#{starttime},#{endtime},#{reservationtime},#{canceledtime},#{description},#{statu
s});.....
```

（3）搜索会议

当会议预订完成，便会跳转到搜索会议功能，显示刚才你提交的会议数据。在这里可以查看所有会议室内容，还可以通过具体条件查询具体的内容。

具体功能代码实现，前端页面 searchmeetings.ftl 获取当前条件框里输入的内容，若是没有，便默认搜索全部会议。然后把数据提交到后端进行处理，MeetingController 中 searchmeetings 进行数据的获取，调用 MeetingService，及其 MeetingMapper 接口使用 MeetingMapper.xml 中 getAllMts 查询方法获取对应条件的数据，并且进行分页查询，每页在界面上显示 10 条数据，在使用 MeetingMapper.xml 中 getTotal 查询在该条件下一共有多少条数据。之后，把数据返回到 MeetingController 中 searchmeetings 方法中，把数据内容，数据分成多少页，数据一共有多少条加入 model 对象中，返回到前端页面显示。前端页面通过首页，上页，下页，和跳转到多少页的按钮，显示对应页数的数据，每次点击这些按钮，前端页面就会返回一个 page 值到后端，通过此 page 值 MeetingMapper.xml 中 getAllMts 就搜索对应页数的内容返回到前端页面。

MeetingController 中的 searchmeetings():

```
    @RequestMapping("/ordinary/searchmeetings")
    public String searchmeetings(Model model, String employeename, String roomname,
@RequestParam(defaultValue = "1") Integer page, Meeting meeting){

model.addAttribute("mts",meetingService.getAllMts(roomname,employeename,meeting,page,PAGE_SIZE));
        Long total=meetingService.getTotal(roomname,employeename,meeting);
```

```
            model.addAttribute("total",total);
            model.addAttribute("page",page);
           model.addAttribute("pagenum",total%PAGE_SIZE==0?total/PAGE_SIZE:total/PAGE_SIZE+1);
            model.addAttribute("meeting",meeting);
            model.addAttribute("employeename",employeename);
            model.addAttribute("roomname",roomname);
            return "searchmeetings";
        }
```

MeetingMapper.xml:

```
.....
<select id="getAllMts" resultMap="BaseResultMaps" >
                SELECT
mt.meetingid,mt.meetingname,mr.roomname,emp.employeename,mt.starttime,mt.endtime,mt.reservationtime
FROM meeting mt
            LEFT JOIN meetingroom mr ON mr.roomid=mt.roomid
            LEFT JOIN employee emp ON emp.employeeid=mt.reservationistid
            where mt.status='0'
            <if test="meeting.meetingname!=null">
                  and mt.meetingname like concat ('%',#{meeting.meetingname},'%')
            </if>
            <if test="roomname !=null">
                   and mr.roomname like concat ('%',#{roomname},'%')
            </if>
.....
```

会议预约模块功能代码到此结束，其他模块功能代码与之类似，不再一一介绍。

5.4 本章小结

本章介绍了项目系统层次的设计和 SSM 框架搭建，对系统的登录功能模块和会议预约模块代码的讲解介绍了系统核心功能实现的过程。

第六章 系统测试

6.1 功能性测试

6.1.1 登录模块测试

登录测试主要是对登录时出现的问题进行测试，用户输入相关账号和密码之后登录时返回的相关反馈说明。如图 6-1 是登录界面展示，测试结果如表 6-1 所示。

表 6-1 登录模块测试表

功能模块	测试用例	预期结果	执行结果
登录测试	管理员	显示全部界面	通过
	普通用户	显示部门页面	通过
	账号错误	提示账号或者密码错误	通过
	密码错误	提示账号或者密码错误	通过
	未通过注册审批	用户待审批	通过
	被关闭账号	用户审批未通过	通过

如果用户没有账号，则需要在注册页面进行注册，如图 6-2 所示，输入姓名，账户，密码，联系电话，电子邮寄，所在部门，其中账号名和姓名是必须输入的，账户名不能重复，如表 6-2 测试结果。

表 6-2 注册模块测试表

功能模块	测试用例	预期结果	执行结果
注册	姓名或者账户名未填	注册失败	通过
	密码和确定密码不一致	两次密码输入不一致	通过
	账户名重复	注册失败	通过

图 6-1 登录界面图

图 6-2 注册界面

6.1.2 预订会议模块测试

预订会议模块主要有 4 个人功能，添加会议室，查看会议室，会议预订，搜索会议。但是添加会议室只有管理员可以操作，其他三个功能都是一样的。测试结果如表 6-3 所示。

表 6-3 添加会议室测试表

功能模块	测试用例	预期结果	执行结果
添加会议室	添加会议室	新增会议室成功，并显示该会议室信息	通过

添加会议室，管理员对企业新添加的会议室在系统上进行数据的增加，需要填写会议室的门牌号，会议室的名称，最多容纳的人数，当前的状态及其备注，如图 6-3 所示。

图 6-3 添加会议室界面

查看会议室，用户点击查看会议室，可以查看所有会议室的内容及其使用情况，点击操作查看详情可以查看会议室的具体内容，点击查看时间，可以了解到该会议室的预订情况，如果时管理员点击查看会议室，可以修改会议室的具体内容和使用状态。如图 6-4 所示，测试结果如表 6-4 所示。

表 6-4 查看会议室测试表

功能模块	测试用例	预期结果
查看会议室	查看会议室	显示该会议室信息
	查看预订时间	显示该会议室的预订时间情况

图 6-4 查看会议室界面

会议预订，用户点击预订会议，就可以输入会议的具体内容，包括会议的名称，会议参加的人数，预计开始时间，预计结束时间，会议室名称，会议说明，选择参会人员。如图 6-5 所示，测试结果如表 6-5 所示。

表 6-5 会议预订测试表

功能模块	测试用例	预期结果	执行结果
会议预订	未输入会议名称	系统提示会议名称为空	通过
	未输入开始时间	系统提示开始时间为空	通过
	未输入结束时间	系统提示结束时间为空	通过
	会议室预订的时间冲突	系统提示该会议在这个时间段被占用	通过
	未选择参会人员	系统提示参会人员为空	通过
	信息填写正确	提交会议预约成功	通过

图 6-5 预订会议界面

搜索会议，搜索会议会显示全部会议记录，还可以通过条件搜索进行具体会议的查询。如图 6-6 所示，测试结果如表 6-6 所示。

表 6-6 搜索会议测试表

功能模块	测试用例	预期结果
搜索会议	添加条件	显示具体会议

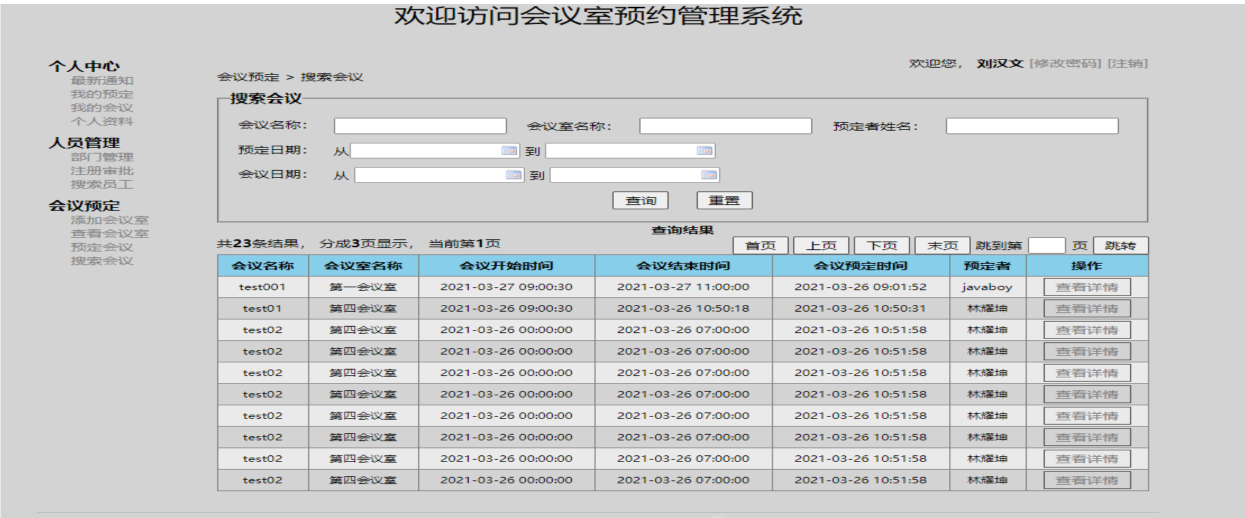

图 6-6 搜索会议界面

6.1.3 人员管理模块测试

人员管理模块一共有 4 个功能，分别是部门管理，注册审批，人员搜索。并且，这个模块只有管理才有权限。

（1）部门管理：管理员进行对部门增删改查的操作，显示部门的名称和编号，并且可以通过部门名字搜索，查看到具体的部门信息如图 6-7 所示，测试结果如表 6-7 所示。

表 6-7 部门管理测试表

功能模块	测试用例	预期结果	执行结果
部门管理	搜索部门	显示成功	通过
	修改部门	修改成功	通过
	添加部门	添加成功	通过
	删除部门	删除成功	通过

图 6-7 部门管理界面

（2）注册审批：管理员对新注册的员工进行进行同意或者拒绝注册的操作，如图 6-8 所示，测试结果如表 6-8 所示。

表 6-8 注册审批测试表

功能模块	测试用例	预期结果	执行结果
注册审批	注册通过	员工注册成功	通过
	注册不通过	员工注册失败	通过

图 6-8 注册审批界面

（3）搜索员工：管理员可以查看所有员工信息，还可以输入员工具体的条件进行搜索，对员工进行具体的操作，如图 6-9 所示，测试结果如表 6-9 所示。

表 6-9 搜索员工测试表

功能模块	测试用例	预期结果	执行结果
搜索员工	条件搜索	显示具体员工	通过

图 6-9 搜索员工界面

6.1.4 个人中心模块测试

个人中心模块有 4 个功能，分别是最新通知，我的预订，我的会议，个人资料。

（1）最新通知：登录成功之后，首先是跳转到最新消息界面，这里会显示最新未来7天我要参加的会议内容和未来7天已经取消的会议内容，如图6-10是最新通知页面效果，表6-10是测试结果。

表6-10 最新通知测试表

功能模块	测试用例	预期结果	执行结果
最新通知	查看最新通知	显示最新会议情况	通过

图6-10 最新通知界面

（2）我的预订：在我的预订界面能看到我预订过，还没有结束的会议，其中每一个会议会有一个操作按钮，点击查看/撤销后能够看这个会议的具体内容，会议名，会议时间，会议的说明，参加会议的人员，会议的人数。图6-11是我的预订界面，图6-12是查看界面，表6-11是测试结果。

表6-11 我的预约测试表

功能模块	测试用例	预期结果	执行结果
我的预订	查看我的预订会议	显示预订的会议	通过

图 6-11 我的预订

图 6-12 查看界面

当你点击界面的撤销会议按钮，跳转到撤销界面如图 6-13，撤销界面可以输入撤销会议的原因，当你点击确认撤销之后，就会跳转到最新消息界面。

图 6-13 撤销界面

（3）我的会议：我的会议界面会显示我即将参加的所有会议如图 6-14 所示，点击查看操作，会显示会议的具体内容。测试结果如表 6-12 所示。

表 6-12 我的会议测试表

功能模块	测试用例	预期结果	执行结果
我的会议	查看我的会议	显示全部会议	通过

图 6-14 我的会议界面

（4）个人资料：显示用户的基本信息，允许用户进行修改，但是修改账号名不能重复如图 6-15 所示。测试结果如表 6-13 所示。

表 6-13 个人资料测试表

功能模块	测试用例	预期结果	执行结果
个人资料	查看个人资料	显示个人资料	通过

图 6-15 个人资料界面

6.2 本章小结

本章主要介绍了系统登录模块测试，预订会议室模块测试，人员模块测试，个人中心模块测试，描述了各模块功能的流程，并且展示了测试结果的效果图。

第七章 总结与展望

7.1 总结

通过查阅相关文献资料了解到会议室及预约会议的发展历程，然后整理出本文会议室预约管理系统的需求分析，运用并结合了 MVC 开发模式，SSM 框架等相关技术进行了本系统的系统设计分析与描述，最终设计并实现了会议室预约管理系统中的主要功能及其功能的测试。在此对本次毕业设计与论文的主要工作进行总结。

通过对我国中小型企业在会议室预约管理方面存在的问题的进行了分析与调查，得出设计会议室预约管理系统的原因，根据原因总结了本系统所需要解决的需求问题，并且通过需求设计出相关功能模块及其数据库表，然后采用软件工程的用例图与建模描述了各个功能模块的流程和数据库表的对应关系，通过 SSM 框架和 MVC 模式的开发思想相结合，实现对系统的用户登录，用户管理，部门管理，会议室管理，会议预约，通过参会人员等主要功能的开发实现，最后测试并展示了系统实际效果图。

通过本会议预约管理系统可以将用户从传统的会议室预约方式中解出来，将大大减少管理员的工作任务量，将出错的概率降到最低并提高一个合理资源的利用率，使企业能方便预订会议，容易管理会议室，参会人员能及时得知消息，成为一个高效办公工具的一部分。

7.2 展望

通过大学期间所学的基本知识，设计并实现了会议室预约管理系统的主要功能，基本能够满足一个企业会议会议预约的日常需求，但是由于自己的知识水平有限，所以系统有许多不足之处，需要进一步的完善。

(1) 系统安全性能的优化，对用户数据的保护等；

(2) 通过短信，微信等通信软件进行通知参会人员；

(3) 此会议管理系统能与移动终端的会议设备进行信息联动；

(4) 移动端设备越来越普及，功能更加强大完善，智能化时代的也随之来临，会议室的智能能化终端设备，实时了解会议室动向，所以未来应该对系统进行改进，以适应智能化，与终端进行联动，使会议室服务行业走向高潮。

参考文献

[1] 谭书旺. 怎样做好会议室管理工作[J]. 秘书之友, 2009, 000(006):17-18.

[2] 崔岢, 邓威. 基于 MVC 思想的程序设计与管理[J]. 赤子, 2012, 000(019):P.172-172.

[3] 王鹤琴, 汪炜玮, 朱珍元. 基于 SSM 框架技术的办公管理系统的研究[J]. 安徽警官职业学院学报, 2017, 016(003):118-122.

[4] 谢晓东, 童敏, 朱建新. Spring Framework 与 AJAX[J]. 计算机与数字工程, 2007, 35(9):40-43.

[5] 唐章琨. 医疗电子商务平台的设计与实现[D]. 电子科技大学, 2018.

[6] 黄劲聪, 黄衍博, 叶浩斌,等. 一种基于 MyBatis 的 SQL 语句的配置方法,系统:, CN108121542A[P]. 2018.

[7] 乔婉风. 挑战传统方法的面向对象技术[J]. 国外建材科技, 2003, 24(3):106-106.

[8] 柳丹, 陈志刚, 雷卫军. 基于 UML 描述的"4+1"视图模型及应用[J]. 计算技术与自动化, 2001, 020(004):46-51.

[9] 张景峰, 胡晓红, 陈海燕,等. 基于 UML 的用例图模型创建[J]. 电脑知识与技术, 2019(32).

[10] 王文玲, 金茂忠. UML 模型及其应用[J]. 计算机工程与应用, 1999(11):47-50.

[11] 管才路, 叶刚, 耿伟,等. 基于 Java 的 Mybaits 生成持久层配置文件[J]. 电子技术与软件工程, 2018, 000(022):P.139-139.

[12] 侯文昊. 基于 RUP 的软件质量改进方法的研究[D]. 北京建筑大学.

[13] Container S . Green B eans : Get ting Started wit h Spring M VC Bootstrapping the DispatcherServlet and Spring Container. 2011.

[14] 邓容. RUP 方法在新浪微博测试管理中的应用和实践[D]. 兰州大学.

[15] 谌桂枝, 沈晓建, 龚兴艳. 基于 Spring 框架的 IoC 微内核的实现机制与应用[J]. 湖南工业大学学报, 2009, 23(003):50-53.

致　谢

我本次毕业设计的指导老师是凌老师，作为副教授的他平时很关心我们的毕业设计工作的进程，为了我们避免浪费了大量时间，总结相关软件工程写作思路，指出论文常见写作错误。凌老师平等对待每个同学，耐心为我们指出错误提出修改意见，为了不使我们在写论文的道路上迷茫，他会引导我们一个正确方向，对于不同课题都能给出不同的方向指导，就像黑夜里的灯塔指引着我们前进，这也让我体会到了凌老师扎实的知识储备水平和大量的人生阅历，在此我向凌老师表示由衷的感谢。