 分类号\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 密级\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

UDC \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 学号



毕业设计（论文）

|  |  |
| --- | --- |
| 论文题目 | SSM框架下的实验室管理系统 |
| **Thesis Topic** | Laboratory management system under SSM framework. |

|  |  |
| --- | --- |
| 学生姓名 | 杨嘉志 |
| 学 号 | 2014006174 |
| 所 在 院 系 | 软件工程 |
| 专 业 班 级 | 1422 |
| 导师姓名职称 | 谢红薇教授 |
| 完成日期 | 2018. 6.11 |

2018年 6月 11日

太　原　理　工　大　学

太　原　理　工　大　学

毕业设计（论文）任务书

第1页

|  |
| --- |
| 毕业设计（论文）题目：  SSM框架下的实验室管理系统  毕业设计（论文）要求及原始数据（资料）：  1．综述国内外SSM框架技术研究现状；  2. 深入了解SSM框架及其内部原理的相关技术；  3．熟练掌握该框架，训练编写web网页的能力；  4．设计并实现在该框架下，实验室管理系统的实现；  5．深入了解SSM框架的关键技术；  6．训练检索文献资料和利用文献资料的能力；  7．训练撰写技术文档与学位论文的能力。 |

第2页

|  |
| --- |
| 毕业设计（论文）主要内容：  1．综述SSM框架在项目开发中的应用；  2．了解SSM框架的相关技术；  3．熟悉SSM框架以及框架的开发环境；  4. 设计在SSM框架下搭建起实验室管理系统；  5．深入分析SSM框架的关键技术以及该框架的设计流程；  6．熟练掌握基于SSM框架下项目的编写；  7. 设计与实现在SSM框架下，实验室管理系统的实现。  学生应交出的设计文件（论文）：  1．内容完整、层次清晰、叙述流畅、排版规范的毕业设计论文；  2．包括毕业设计论文、源程序等内容在内的毕业设计电子文档及其它相关材料。 |

第3页

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 主要参考文献（资料）：   1. 丁毓峰, 毛雪涛. Java Web开发教程[M]. 北京:人民邮电出版社, 2017 2. 李新良, 游新娥, 禹云. JAVA面向对象程序设计教程[M]. 北京:北京理工大学出版社, 2016 3. Chris, Aquino, Todd, Gandee. Web开发权威指南[M]. 北京:人民邮电出版社, 2017. 4. Budi, Kurniawan, Paul, Deck. Servlet、JSP和Spring MVC初学指南[M]. 北京:人民邮电出版社, 2016 5. 孙卫琴. Tomcat与Java Web开发技术详解[M]. 第2版. 电子工业出版 6. 陈永政, 张正龙. Java EE框架技术[M]. 西安: 西安电子科技大学出版社, 2017 7. 丁旭. 基于B/S架构的软件项目实训[M]. 北京:清华大学出版社, 北京交通大学出版社, 2011 8. Walter, Savitch. Java[M]. Beijing:Higher Education Press, 2003 9. Alessandro, Bozzon, Philippe, Cudré-Mauroux, Cesare, Pautasso. Web engineering[M]. Switzerland :Springer, 2016. 10. Sestoft. Java precisely[M]. Cambridge, MA:The MIT Press, 2016 11. 杨海鹰, 潘华. 实验室信息管理系统[M]. 北京:化学工业出版社, 2007 12. 任光辉，赵扬，彭志广. 基于网络平台的实验室管理系统研究与实践[J]. 实验室科学, 2009, (05): 116-118 13. 侯亚彬, 宫德龙, 雷进生, 丁颖. 实验室研究与探索[J]. 实验室研究与探索, 2012, (04): 372-374  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | 专业班级 | 软件1422 | | | | | 学生 | 杨嘉志 | | 要求设计（论文）工作起止日期 | | | | | 2018年3月23日~2018年6月29日 | | | | 指导教师签字 | |  | | | | 日期 | 2018年6月29日 | | 教研室主任审查签字 | | | |  | | 日期 |  | | 系主任批准签字 | | |  | | | 日期 |  | |

SSM框架下的实验室管理系统

摘 要

多年以前，实验室的管理是靠手写，后来有了电脑，大多是用表格或者文件来进行管理。但是信息依然很多而且很复杂，管理起来需要花费很大的精力来记录，管理，添加，统计实验设备和实验室的相关信息。而且效率不高，随着计算机软件的兴起，用计算机来管理实验室就显得尤为重要。因此，建立一个SSM框架下的实验室管理系统，可以提高实验室管理的效率与使用价值。

本系统主要设计一个实验室管理系统，实现了对实验室及其实验室的设备管理，使得用户不用再花费额外精力在实验室的维护上。对于实验室的管理主要是通过计算机来管理。该系统采用了当下流行的SSM框架，使得每个功能之间的耦合度较低，方便后期的维护与功能迭代。

关键词：SSM；实验室；管理系统

Laboratory management system under SSM framework.

#### Abstract

Many years ago, the management of the lab was written by hand, and later computers, mostly in forms or documents, were managed. But it's still a lot of information and it's complicated, and it takes a lot of effort to manage, manage, add, statistical lab equipment and lab information. With the rise of computer software, it is very important to use computers to manage the laboratory. Therefore, the establishment of a laboratory management system under SSM can improve the efficiency and use value of laboratory management.

This system mainly designs a laboratory management system, which implements the management of the laboratory and its laboratory equipment, so that users no longer have to pay extra attention to the maintenance of the laboratory. The management of laboratory is mainly through computer. The system adopts the current popular SSM framework, which makes the coupling degree between each function low and facilitates the maintenance and function iteration in the later stage.

**Key words:** SSM; laboratory; management system.

目 录

[1 绪 论 1](#_Toc516123188)

[1.1 概述 1](#_Toc516123189)

[1.2 国外研究现状 1](#_Toc516123190)

[1.3 国内研究现状 2](#_Toc516123191)

[1.4 研究意义 2](#_Toc516123192)

[2 系统概述与技术介绍 3](#_Toc516123193)

[2.1 系统概述 3](#_Toc516123194)

[2.2 相关技术 3](#_Toc516123195)

[2.2.1 Spring 3](#_Toc516123196)

[2.2.2 SpringMVC 3](#_Toc516123197)

[2.2.3 MyBatis 3](#_Toc516123198)

[2.2.4 MySQL 3](#_Toc516123199)

[2.2.5 Tomcat 4](#_Toc516123200)

[3 需求分析说明书 5](#_Toc516123201)

[3.1 概述 5](#_Toc516123202)

[3.2 可行性分析 5](#_Toc516123203)

[3.3 用例分析 6](#_Toc516123204)

[3.4 数据描述 9](#_Toc516123205)

[3.5 功能需求分析 10](#_Toc516123206)

[3.5.1 功能划分 10](#_Toc516123207)

[3.5.2 功能描述 10](#_Toc516123208)

[3.6 非功能性需求分析 17](#_Toc516123209)

[3.6.1 操作便捷 17](#_Toc516123210)

[3.6.2 安全 17](#_Toc516123211)

[3.6.3 响应快 17](#_Toc516123212)

[4 系统总体设计 18](#_Toc516123213)

[4.1 软件总体功能结构 18](#_Toc516123214)

[4.2 用户界面设计 19](#_Toc516123215)

[4.2.1 注册登录界面 19](#_Toc516123216)

[4.2.2 主界面 19](#_Toc516123217)

[4.2.3 通用的列表界面 19](#_Toc516123218)

[4.2.4 申请购置设备界面 19](#_Toc516123219)

[4.3 数据库设计 19](#_Toc516123220)

[4.3.1 数据库概念模型设计 19](#_Toc516123221)

[4.3.2 数据表结构 19](#_Toc516123222)

[5 系统详细设计 25](#_Toc516123223)

[5.1 数据结构 25](#_Toc516123224)

[5.1.1 相关类名 25](#_Toc516123225)

[5.1.2 分页的数据结构 25](#_Toc516123226)

[5.1.3 一些常量 25](#_Toc516123227)

[5.1.4 命名规范 25](#_Toc516123228)

[5.2 核心功能设计 25](#_Toc516123229)

[5.2.1 登录 25](#_Toc516123230)

[5.2.2 购置设备 25](#_Toc516123231)

[5.2.3 删除设备 25](#_Toc516123232)

[5.2.4 用户管理 25](#_Toc516123233)

[5.2.5 实验室管理 25](#_Toc516123234)

[6 系统实现 26](#_Toc516123235)

[6.1 环境搭建 26](#_Toc516123236)

[6.2 功能模块 26](#_Toc516123237)

[6.2.1 注册 26](#_Toc516123238)

[6.2.2 个人信息管理 28](#_Toc516123239)

[6.2.3 实验室设备管理 30](#_Toc516123240)

[6.2.4 购置设备 32](#_Toc516123241)

[6.2.5 审批购置设备申请表 33](#_Toc516123242)

[6.2.6 盘点统计实验室数据 35](#_Toc516123243)

[6.2.7 通知管理 36](#_Toc516123244)

[6 系统测试 37](#_Toc516123245)

[6.1 软件测试目标 37](#_Toc516123246)

[6.2 软件测试方法 37](#_Toc516123247)

[6.3 主要模块测试 37](#_Toc516123248)

[6.3.1 登录模块测试 37](#_Toc516123249)

[6.3.2 申请购置设备模块测试 39](#_Toc516123250)

[6.3.3 修改个人信息模块测试 40](#_Toc516123251)

[6.3.4 查询实验室设备模块测试 41](#_Toc516123252)

[6.3.5 盘点统计模块测试 43](#_Toc516123253)

[6.4 测试总结 44](#_Toc516123254)

[总 结 46](#_Toc516123255)

[参考文献 47](#_Toc516123256)

[致 谢 48](#_Toc516123257)

[外文原文 49](#_Toc516123258)

[外文翻译 55](#_Toc516123259)

# 1 绪 论

## 1.1 概述

当今世界已经步入到信息化发展时代，社会各个地方都存在这被计算机处理过后的信息。可以说计算机已经广泛的被各个领域所接受，融入到了各行各业当中并且发挥着举足轻重的作用，成为当今科学发展的主要动力。

随着时代的进步，学校的规模不断的扩大。学校的设施也越来越好，数量也越来越多，也越来越复杂。实验室中的设备也是越来越先进，且实验室在高校无论是育人还是科研都发挥着至关的作用。但是，如何最大的利用实验室，管理实验室达到减轻开发人员的工作，提高他们在科研上的效率则显得尤为重要了。

因此，实验室管理系统就应运而生了。实验室管理系统是在传统的管理基础上，利用计算机技术，进行的一个改良。将以前的纸质版的记录与管理替换成了现在的数据库存储。以前通过纸笔计算的逻辑现在通过计算机即可优雅的展现出来。

## 1.2 国外研究现状

国外的实验室管理系统技术最早出现在上世纪六十年代末，随着计算机技术的发展，实验室管理系统已经得到了长足的进步。尽管最初的实验室管理系统只能在小型机上构建，但管理思想和应用模式得到了肯定和完善。这为实验室管理系统的进一步发展奠定了良好的基础。实验室管理系统主要强调对数据的管理，如数据的存储、查询等，转到了强调以管理为中心的新一代实验室信息管理系统，更加强调对实验室整体管理的实现，从对数据管理的计算机实现上面进展到应用管理理论全方位地对整个实验室的运行实施管理。伴随实验室认证的推广和要求，实验室管理系统成为实验室管理水平、提高实验室专业水准、提高效率的重要工具。实验室管理系统能够使实验室的管理水平提升到当前时代的前沿。近几年，国外在这方面的进展非常迅速，先进的信息处理以及计算机技术得到了很多应用。

第一套商品化的实验室管理系统是在1798年由HP公司的一位员自组公司开始开发，于1982年正式推出了商品化的实验室管理系统并取得了成功，首先在石油、化工、制药、冶金等大型企业中得到应用，现在已经成为企业IT架构的一部分。

2004年，美国的实验室管理系统供应商已经达到100多家，应用领域从传统的专业检测/校准实验室，制药企业，医院扩展到同样具有实验室特点的诊所等领域，出现了系统化的产品指导手册，如LIMSBook和LIMS Buyer Guide。

目前，实验室管理系统在欧、美、澳、日本等发达国家实验室已经广泛应用，并且遍及电子、石化、冶金、医药、环保、地质、农业、医学等领域。

## 1.3 国内研究现状

在我国，由于计算机的发展起步较晚，因此对于实验室管理系统的研究起步也算较晚。国内真正应用实验室管理系统是在20世纪90年代。1998年石油化工科学研究院首次开发出了国内需求的商业版的实验室管理系统。

自2000年以后，由于国内外实验室管理系统厂商以及有关人士的共同努力，实验室管理系统的技术逐渐开始为大众了解，逐步开发。

虽然实验室管理系统在国内的开发及应用与西方发达国家相比仍有不少差距，但是在研究追赶的步伐上我们也做了很多尝试。并且取得了一些成果，目前在各行业的检测和校准实验室中得到了广泛的应用。如高校，环保，食用药品，建筑，卫生等行业。

很多来自高校的研究者和来自实验室的应用者发表了很多关于实验室系统的成果，各种架构和实现方法。主要的系统架构有客户端/服务器结构（C/S结构）、浏览器/服务器结构（B/S结构）以及混合结构三种。

## 1.4 研究意义

当今世界信息化时代相对成熟，但大多数软件还是基于客户端的形式，如今也是慢慢的向B/S开始转变。这样可以随时随地只要有网就可以进行系统的管理。

随着时代的进步，学校的规模不断的扩大。学校的设施也越来越好，数量也越来越多，也越来越复杂。实验室中的设备也是越来越先进，且实验室在高校无论是育人还是科研都发挥着至关的作用。但是，如何最大的利用实验室，管理实验室达到减轻开发人员的工作，提高他们在科研上的效率则显得尤为重要了。

基于B/S模式的实验室管理系统，它所覆盖的功能范围包括有实验室设备管理功能，实验室预约管理功能，实验设备购置与报废、审批购置申请报废申请、基本的设备查询统计等。它是一个由计算机软件、数据库管理、以及管理人员，学生，教师有机组成的实验室管理系统。

# 2 系统概述与技术介绍

## 2.1 系统概述

实验室管理系统是一个基于B/S架构的项目，可以满足于在移动端，PC端上进行对实验室的管理操作。甚至于任何一台只要带有浏览器且可以上网的计算机上运行。该系统不仅具有实验室的管理功能，也具有对实验室设备的管理功能，还有通知公告的管理，以及对设备进行统计的功能。旨于打造一个智能化的实验室管理系统。为了系统的稳定运行以及项目的便捷性开发，采用了以下先进的技术作为本项目的技术栈：Spring、SpringMVC、MyBatis、MySQL以及Tomcat。

## 2.2 相关技术

### 2.2.1 Spring

Spring是一个开源框架，同时也是一个轻量级的框架,。并且Spring也是模块化的，这意味着我们可以使用其中的部分功能而不考虑其他不需要的功能。例如我们可以使用Spring中的AOP，IoC，DI并且舍弃其中的Dao使用Mybatis代替。

### 2.2.2 SpringMVC

Spring 框架提供了我们日常开发中需要的web应用的MVC模块。因此可以说是spirngMVC是spirng的后来产品。SpringMVC所有的设计都是基于一个servlet，其类名为DispatcherServlet。所有的请求都将被它拦截，并且在初始化该servlet的时候就将映射初始化进来，因此DispatchServlet可以返回映射之后的视图。

### 2.2.3 MyBatis

MyBatis 是一个不错的持久层框架，减少了代码的硬编码，也支持用户自己编写SQL语句，可以在xml中可以完成一对一或一对多的映射。使用起来非常轻便，易于上手。并且MyBatis支持多种数据库。

### 2.2.4 MySQL

MySQL是一个[关系型数据库管理系统](https://baike.baidu.com/item/%E5%85%B3%E7%B3%BB%E5%9E%8B%E6%95%B0%E6%8D%AE%E5%BA%93%E7%AE%A1%E7%90%86%E7%B3%BB%E7%BB%9F)，由瑞典MySQL AB 公司开发，目前属于。[Oracle](https://baike.baidu.com/item/Oracle)。旗下产品。MySQL是最流行的[关系型数据库管理系统](https://baike.baidu.com/item/%E5%85%B3%E7%B3%BB%E5%9E%8B%E6%95%B0%E6%8D%AE%E5%BA%93%E7%AE%A1%E7%90%86%E7%B3%BB%E7%BB%9F)之一，在 WEB 应用方面，MySQL是最好的 RDBMS (Relational Database Management System，关系数据库管理系统) 应用软件。

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

MySQL所使用的 SQL 语言是用于访问[数据库](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93)的最常用标准化语言。MySQL 软件采用了双授权政策，分为社区版和商业版，由于其体积小、速度快、总体拥有成本低，尤其是[开放源码](https://baike.baidu.com/item/%E5%BC%80%E6%94%BE%E6%BA%90%E7%A0%81)这一特点，一般中小型网站的开发都选择 MySQL 作为网站数据库。

### 2.2.5 Tomcat

Tomcat是Apache下的一个开源的服务器项目，由于其开源以及代码效率执行非常好，深受广大开发人员的热爱。并且Tomcat服务器实现了jsp的解析，而且提供了一些不错的web功能。

# 3 需求分析说明书

## 3.1 概述

开发设计一个好的管理信息系统应该从实用、好用的角度出发，建立设备的管理系 统，在开发设计中应该考虑以下的原则：

可操作性：系统应该尽量减少操作员的数据录入量，数据输入的格式应符合业务习惯，并且直观、方便。要求系统处理的数据能准确无误，同时输出信息要求直观、简洁。

实用性：从用户的实际需要出发进行系统开发，不盲目追求高新技术的应用。

可靠性：系统运行具有较高的可靠性，提供严格的并发控制，确保数据的一致性和正确性。

安全性：系统安全措施可靠、高效、可维护性好，有权限控制、口令控制、临时锁定控制，其中口令录入界面便于系统识别登录用户。

可维护性：为了保证系统的可维护性，要求具有详细的文档资料，同时，要求系统在功能设计上考虑可扩展性，以满足业务变动的需求。

可移植性：系统开发完成后，要能运行于任何由Windows操作系统所构成的计算机网络环境下。

实验室不仅是大学教学或是研究学术都极其重要且不可或缺的一个场所，在教学中扮演了很主要的角色。但是实验室中也有充斥着各种各样的需求，如实验室的仪器、设备的管理，各种各样的通知公告管理，各种各样的申请表的管理，显然，如果现在仍使用古老的人工管理现在先进的设备以及复杂的人与人之间的关系（预约），甚至影响研究人员的效率。因此，人们迫切希望通过计算机来只能的管理这些设备，提高我们的工作效率。使得管理起来简捷，清晰。

## 3.2 可行性分析

一个完备的实验室管理信息系统具有以下优越性：

方便领导查询实验室设备的使用情况，节省时间，可以提高系统的工作效率和准确率。为了适应新形势的发展，我进行了这一系统的初步设计工作，也可以说是做一个初步的探索，希望它能够在查询实验室设备信息时能够快速的定位到用户想要的结果上，不仅仅是提高领导的效率，也要提高到使用者的效率。让人们更有精力去面对那些更加复杂的研究当中，该实验室管理系统是以MySQL数据库为持久化层、以B/S架构为服务的管理平台，对想要的设备信息能够快速的检索，同时也能对复杂的业务能够简捷的处理。

（1）经济性

经济可行性主要依据是成本/效益分析，利用网上成熟已久的技术快速进行开发，使系统能减少很多不必要的资源，使得现在的系统不像以前那么古老，并且还提高了实验室信息管理的精确度与简捷的操作。方便快速的操作，可减少实验室信息管理上的缺陷，又减少因工作的堆积而可能导致的错误，并且操作便捷，可减少许多不很必要的人员。目标系统开发需求比较低，加上现在已是信息时代，所以在软硬件上的花费比较少，在为使用者带来便利的同时，也为系统的进一步推广创造了条件。因此从经济角度考虑，此实验室设备管理系统开发可行。

（2）技术性

本系统采用的技术是基于SSM框架下的J2EE开发，现阶段，SSM框架相对已经成熟，不仅在官网上具有详细的参考文档，在各大博客站中也具有详细的实例教程。且基于Java面向对象的思想，使得业务逻辑也变得非常简单清晰，因此完成该项目的开发不是很难。从技术上来讲，是可行的。

（3）法律性

从法律因素和软件免费角度考虑，所有技术都从网上的博客获取，开源软件的官方文档介绍，所有软件都选开源软件。所以没有侵权这一说，我们使用的都相当于授权软件，因此在法律上也是可行的。

（4）操作性

用户仅需具有基础的电脑操作能力即可，经过本系统的智能化处理后，操作也很简单方便，功能列表都放在了界面上。用户只需点击以及少量的键盘输入即可完成办公。从操作性上来讲是可行的。

依据以上因素，本实验室管理系统开发项目的可行性还是可行的。不仅方便较好实现吗，而且技术成熟，参考资料丰富。综上所述，此项目在技术、经济、操作和社会效益上是完全可行的。

## 3.3 用例分析

（1）角色分析

该实验室管理系统主要是对实验室及其设备进行管理，因此在使用过程中，用户角色分为四种：学生，老师，管理员，系统管理员。学生可以通过该系统查看实验室设备的相关信息，以及申请实验室的使用，以及实验室的相关信息。老师可以管理实验室的相关信息，也可以申请购置设备等操作。管理员则可以对设备进行购置，统计，维修，设备入库操作，可以修改或者发表通知，可以对自己进行信息维护管理。

（2）用例分析

学生：

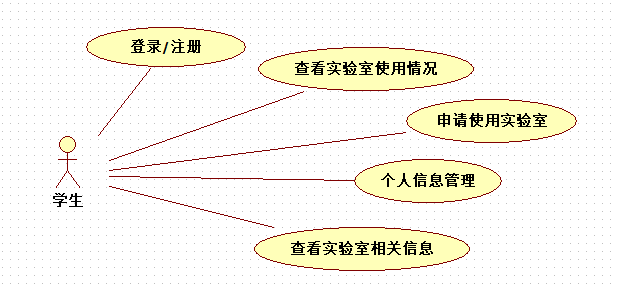


图3-1 学生用例图

学生最先接触的部分便是登录注册，在该模块内，学生可以注册一个属于自己的账号密码。紧接着学生可以查看实验室相关信息，如果需要申请实验室则可以对该实验室进行申请。

当然，学生还可以对自身的信息进行管理，可以更改相关的信息

教师：

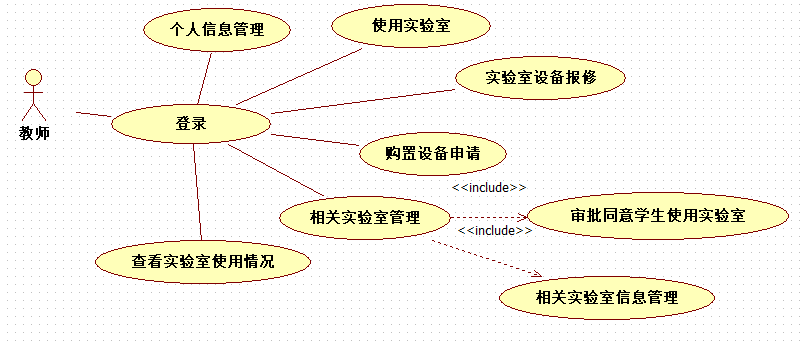


图3-2 教师用例图

教师无需注册，由管理员分配账号密码。教师可以修改一些与自己相关的信息。教师可以使用自己所属的实验室，如果需要使用其他的实验室，则需要申请。当教师需要一些设备但实验室没有的时候，教师可以申请设备的购置。教师也可以修改所属实验室的信息，如简介，设备的相关信息。教师可以查看该实验室的使用情况，如果有学生申请使用该实验室，则可以审批通过并且安排学生做相关实验。

管理人员：

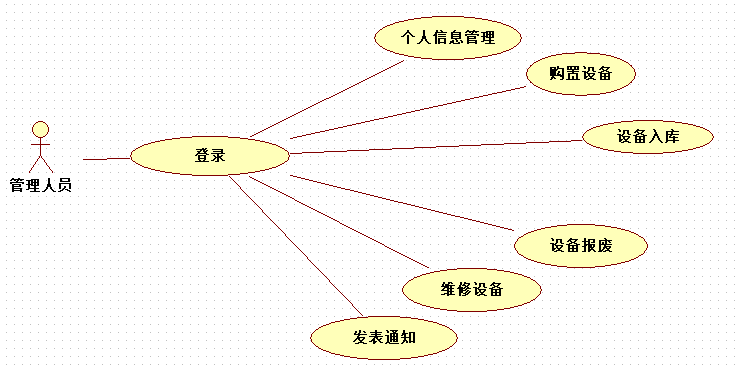


图3-3 管理人员用例图

管理人员无需注册，由系统管理员分配账号密码。管理人员登录之后，可以进行个人信息的管理。可以对设备进行购置，报废，维修，入库操作。可以发表通知。

系统管理员：

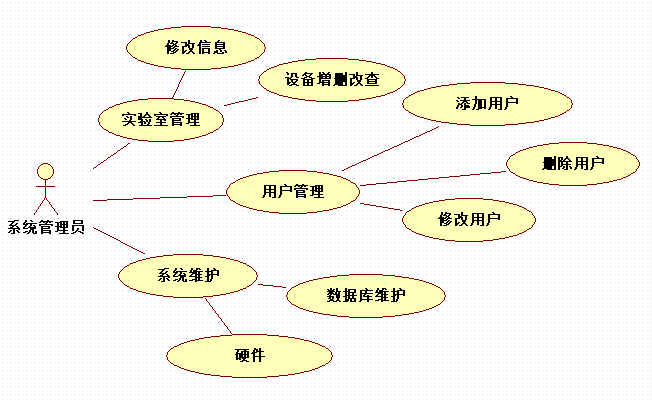


图 3-4 系统管理员用例图

系统管理员拥有所有角色的权限，并且可以对用户进行增删改查，系统管理员也可以对系统进行维护，例如停机维护。

## 3.4 数据描述

数据字典：

①登陆信息

输入：用户信息

输出：身份验证

描述：登陆信息=用户名+密码

②查询信息

输入：合法人员对设备的查询操作，统计使用情况

输出：设备信息表

描述：查询信息=设备名+编号+设备型号+价格+损坏程度+实验室

③维修信息

输入：所需维修设备的信息

输出：确定维修设备的记录

描述：维修信息=设备名+修理日期+修理厂家+修理费+负责人

④新设备

输入：购买设备的信息

输出：确定购买的新设备的记录

描述：新设备=申请表的id+单价+数量+等

⑤报废设备

输入：所需报废的设备的信息

输出：报废的设备记录

描述：报废设备= 表id +负责人+ 报废日期

⑥申请购置设备信息

输入：需要购买的设备信息

输出：申请记录

描述：申请购置设备信息 = 表id+设备名+设备型号+设备数量+购置预算+负责人

⑦申请报废设备信息

输入：需要报废的设备信息

输出：报废记录

描述：申请报废设备信息 = 设备id + 申请表id +负责人

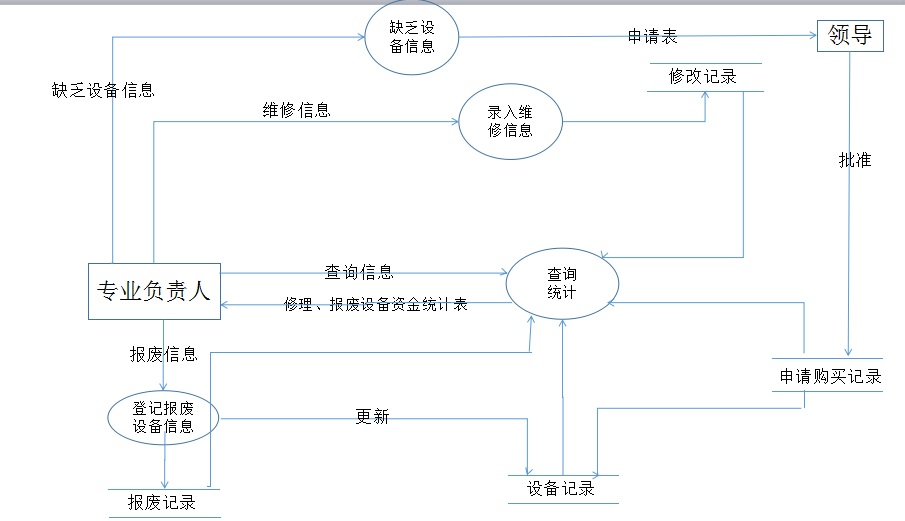


图3-5 数据流图

## 3.5 功能需求分析

### 3.5.1 功能划分

（1）管理员可以删除设备，但是每删除一次都要记录下来。  
（2）对于有问题的设备，要能够及时修理，并且记录该次修理记录。

（3）对于实验室缺少的设备，用户可以向管理员申请，并且将该次申请表记录到数据库当中。   
（4）对于实验室设备，应做到随时都能查询，并且是根据id，设备名，产地或是实验室名模糊查询。

（5）有些必要的通知，或重大的消息，应及时发表在首页。

（6）实验室的管理，应能显示各个实验室的相关信息，并且能够统计出各个实验室的设备信息。

### 3.5.2 功能描述

（1）注册登录

该功能除了对权限的管理，并将用户名放到session中，对于用户需要购买设备申请或者进行其他操作的时候不用再输入自己的姓名，页面上会自动填写。

需要注意几点：

1. session中不得存放用户密码
2. 登录校验时不得显示“用户名不存在”这样的提示消息。
3. 存入数据库中的密码应当加密，且在后台加密。
4. 注册时应校验用户名是否存在，且及时提示给用户。

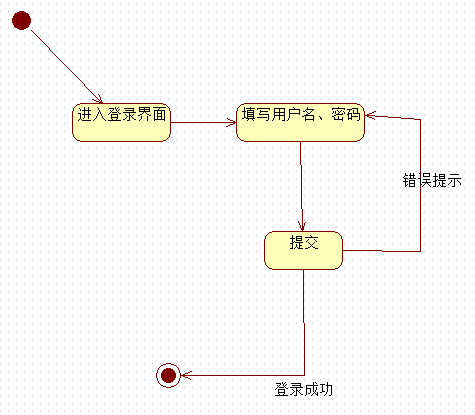


图3-6 登录状态图

（2）个人信息管理

该模块会显示用户自己的信息，用户可以对自己需要修改的信息进行操作，但是部分信息不能让用户编辑，例如，用户名，所属的实验室。

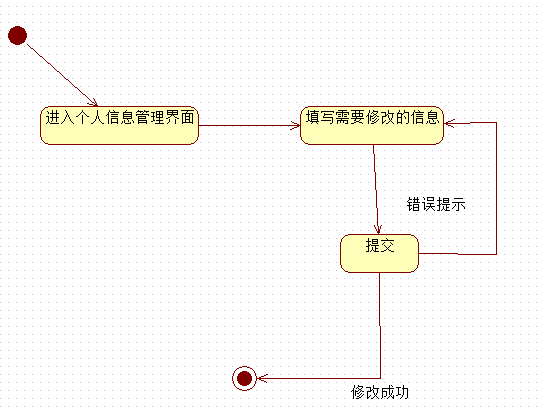


图3-7 修改信息状态图

（3）实验室设备管理

要求用户点开该界面后，可以查看所有设备的信息，但是是分页之后的效果。用户可以在该界面搜索设备，也可以点击页数进行浏览。也可以根据实验室，时间，进行排序过滤。

用户可根据展示的设备列表，进行选中删除，和修改，还有查看其详细信息功能。

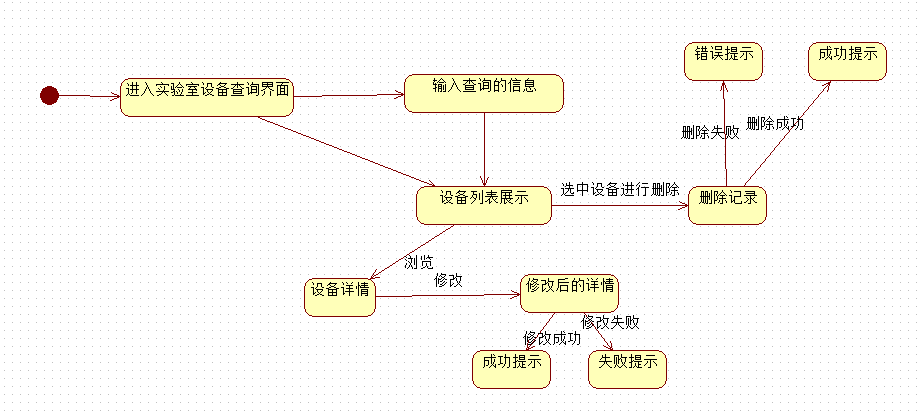


图3-8 实验室设备管理状态图

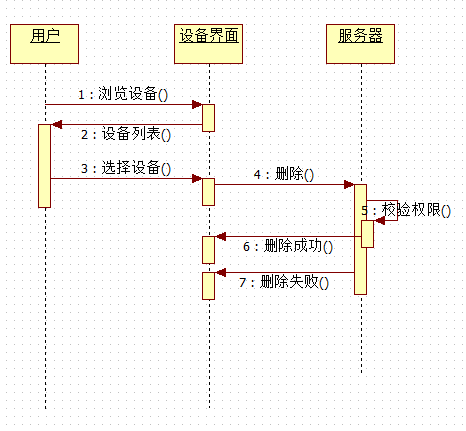


图3-9 实验室设备删除时序图

（4）购置设备

用户应该根据实验室所缺少的设备来进行购置，用户可以先在市场进行调研，将所需要的设备信息记录起来。然后在“申请购买设备”界面填写相关的信息，如: 申请表的标题， 设备名， 单价， 供应商的地址， 该设备放置的实验室名称。当然，不必要输入的信息不应该让用户手动输入，避免效率问题以及繁琐的操作问题。例如： 用户的个人相关信息， 可以系统自动生成， 以及实验室名称，可以以下拉框的形式展现给用户。

该功能模块除以上基本需求之外，还应包括：用户可以自由的添加设备信息，使其一张表里可以有多种设备，且由用户自己决定。

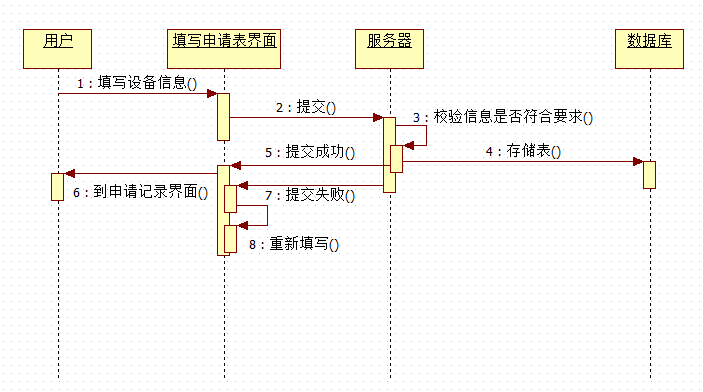


图3-10 购置设备时序图

（5）审批购置设备模块

拥有审批权限的用户可以对购置申请记录进行审批，且应对审批结果进行状态值分类。

1. 未审批的申请表的审批结果应显示“未审批”，对应状态值为 0。
2. 通过审批的申请表的审批结果应显示“通过”，对应状态值为1。
3. 未通过审批的申请表的审批结果应显示“未通过”，对应状态值为 -1。

用户可以随时查看自己的申请状态，且当申请记录得到审批后，应更新数据库中的设备表。

除以上需求外， 具有审批权限的用户还应在审批列表界面可以对申请列表进行查询操作，用以快速定位到用户想要审批的申请项。

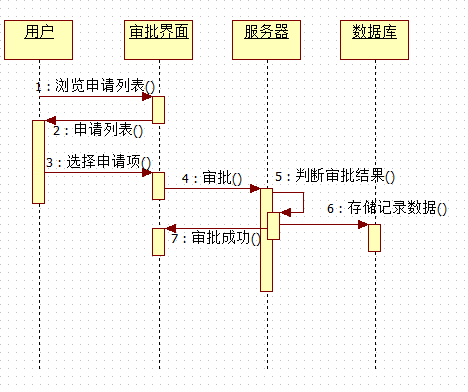


图3-11 购置设备时序图

（6）盘点统计实验室数据

在该功能模块下，用户不仅可以浏览每个实验室的相关信息，还可以查看在该实验室下，其设备的数据统计。该统计将显示，在该实验室下的前8个设备的数量占比情况。用以分析每个实验室的设备数量情况。

（7）实验室管理

1. 后台管理人员可以创建一个实验室，并且为其分配一个管理人员。

2. 后台管理人员与该实验室的管理人员可以对该实验室进行编辑操作。

3. 用户可以浏览所有实验室的相关信息，包括该负责人的姓名，工号等信息。

4. 后台管理人员可以对实验室进行删除操作。

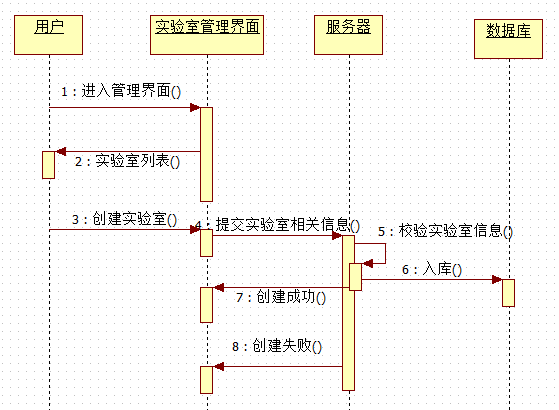


图3-12 创建实验室时序图

（8）通知管理

1. 后台管理人员可以创建一个通知信息，用于展示在首界面。如果该通知带有图片，则其将有可能会被放置在图片轮播当中。

2. 所有人可以浏览所有的通知，包括通知的图片，发布人，发布时间等信息。

3. 后台管理人员可以修改该通知。

后台管理人员可以删除该通知。

1. 除以上必须功能除外，还应具有如下功能：
2. 所有人可以对通知进行搜索，用以快速定位到用户想要浏览的通知。

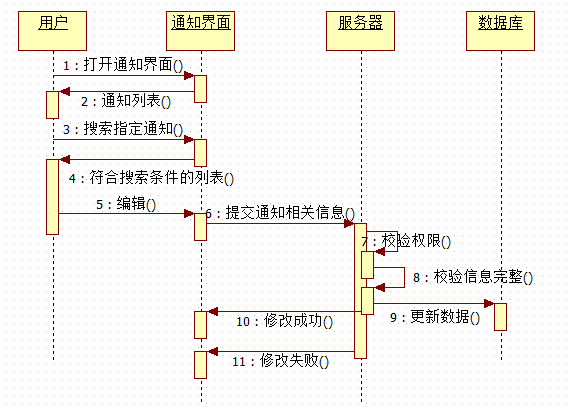


图3-13 修改通知管理模块

## 3.6 非功能性需求分析

### 3.6.1 操作便捷

操作方便是体现在界面上简单易于操作，但界面由不能太过简单而显得朴素单调。让用户应知道每一个按钮不用点击就能知道其作用是什么。也应能让用户快速找到自己所需要的功能。从而提高效率

### 3.6.2 安全

用户的每一次申请的记录或者操作的记录，除了自己和系统管理员外其他人应不可见。用户注册的密码也应加密。

### 3.6.3 响应快

用户点击的每一个功能，应能快速的响应其操作，这个需求应从代码、部署环境入手。首先将代码的功能模块之间耦合度降低，这样请求的时候功能调用就不会拖泥带水。再者是从部署环境上考虑，可以考虑高带宽的服务器，也可考虑服务器集群。

# 4 系统总体设计

## 4.1 软件总体功能结构

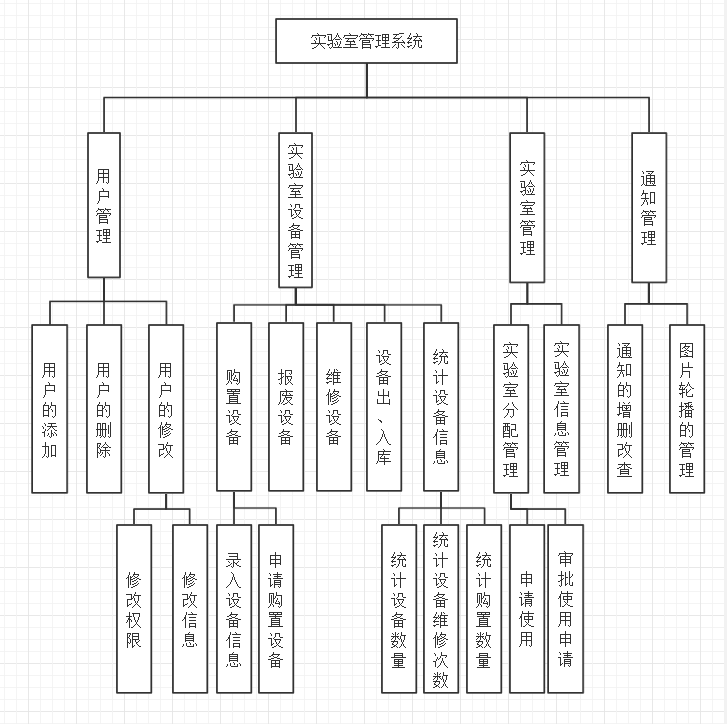


图4-1 系统功能层次图

该系统总共分为四大模块：用户管理、实验室管理、实验室设备管理、通知管理

（1）用户管理

系统管理员对于用户可以对其相关信息进行查看、编辑、删除。系统管理员也可以创建用户。为其分配权限。

（2）实验室设备管理

购置设备：管理员可以申请对设备进行购买，需要先录入设备的信息，再申请购置设备。

报废设备：管理员可以将实验室中损坏严重的设备进行报废处理，报废时应保存记录到数据库当中。

维修设备：管理员可以将实验室中的设备进行维修，维修时应保存记录到数据库当中。

设备的出入库：管理员将采购的设备放置到仓库当中，这时我们可以将对应的设备进行入库操作，将仓库中对应的设备移至到对应的实验室当中。相反，也可以将实验室中对应的设备移出到仓库当中。

统计设备信息：系统可以将各实验室的设备数量进行统计并以图的形式展现出来，让用户更加的了解当前该实验室的设备分布情况。也可以将设备的状况统计出来，方便管理员直观的了解与管理。

（3）实验室管理

实验室分配管理：当学生申请使用实验室的时候，管理员可以根据时间来分配实验室给对应的学生。

实验室的信息管理：管理员对于实验室可以进行增删改查的操作。增加一个实验室并且填写它的相关信息，也可以修改一个实验室的相关信息，也可以删除一个实验室。

（4）通知管理

通知的信息管理：管理员可以对通知进行增删改查。

图片轮播的管理：管理员可以控制首页的图片轮播的图片与标题，可以将符合一定条件的通知放到图片轮播当中。

## 4.2 用户界面设计

### 4.2.1 注册登录界面



图4-2 登录界面设计图



图4-3 注册界面设计图

登录界面如图4-2所示，其由两部分组成：左侧（宣传图），右侧（登录表单）。看似将这两个模块对半分，实则在其中间留有一部分空白，使其看起来不至于特别紧密。登录表单中，顶部是项目的名称，再下来则是提示用户失败的错误消息。然后就是账号密码。再下来就是复选框以及超链接分别对应的是记住密码与注册。再下来就是登录的按钮。登录表单底部与上中部隔了一条线，使界面看起来层次感丰富。底部将网站的版权以及备案号展现出来，这样突出了该网站是正规网站。

注册界面如图4-3所示，初步打算将其作为模态框的形式展现给用户。因此会看到右上角会有一个“关闭”按钮。接下来就是用户注册的基本的信息，然后就是提交与取消按钮。在底部会显示有一个“已有账号，点我登录”的超链接。目的是关闭该模态框，让用户输入账号密码去登录。

### 4.2.2 主界面



图4-4 主界面设计图

主界面如图4-4所示，该界面由三部分组成：头部，内容，尾部。头部则是一个网站的顶部导航栏，该顶部导航栏中包括了该项目的LOGO，还有导航的功能按钮。网站的内容由三部分组成：图片轮播，通知公告，业务处理。图片轮播可以用bootstrap框架中的图片轮播组件即可。通知公告中包含了标题，以及右侧的“更多”超链接，还有一些通知列表。左侧的“通知公告”与右侧的“业务处理”高度相同，目的是为了显得网站整齐大方。左侧与右侧采取了9:3的比例排放。右侧的业务处理则是显示了四个业务按钮，供用户来点击跳转到对应的业务界面。底部显示了网站的版权以及备案号，突出网站的正规性。整体看来，网站的设计上有天，下有底。整体看来不会显得特别突兀。

### 4.2.3 通用的列表界面

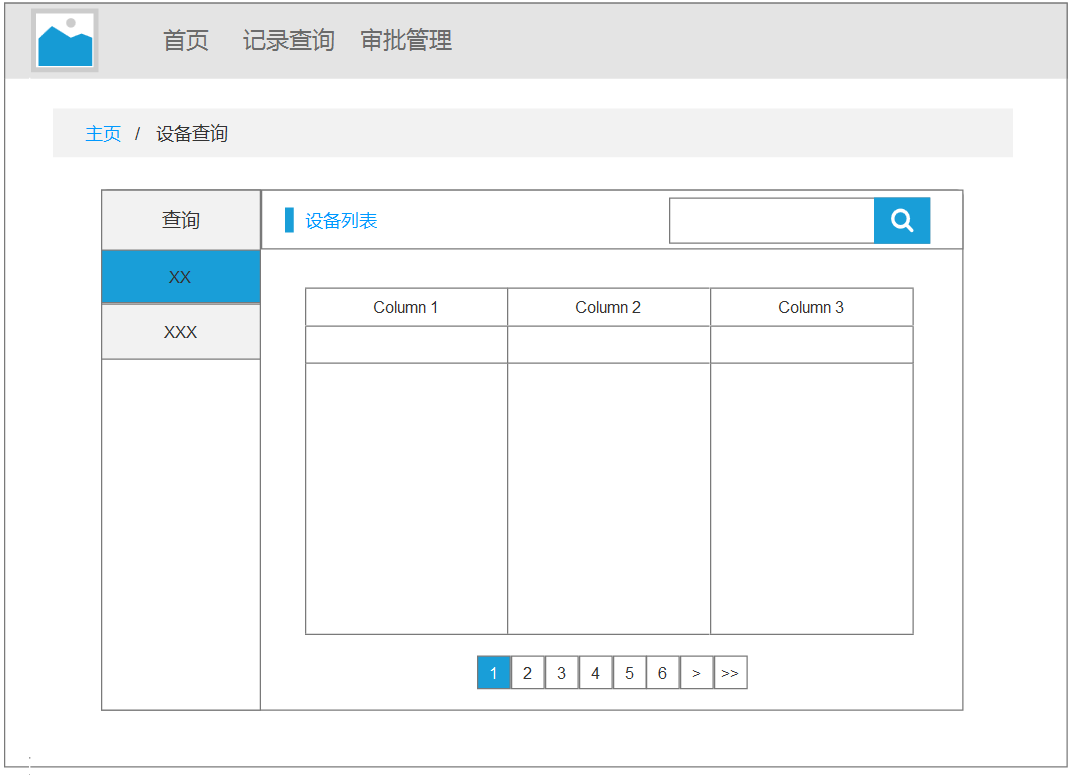


图4-5 通用的列表界面设计图

通用的列表界面设计图如图4-5所示，该界面是由三部分组成：头部，面包屑，主体。面包屑相当于一个辅助导航栏，既可以展现用户所在的位置，又可以让用户点击主页返回。该面包屑的宽度为页面的80%，居中，以灰色填充。主体部分宽度为980px，居中。主体部分分为两部分，左侧为导航栏，导航用户想要进行的操作。而右侧则是对应的列表。该列表由两部分组成：头部，内容。头部包含了一个小标题，和一个搜索输入框。内容则设计为列表+分页栏。

该页面整体颜色数量不超过4个颜色，使用户看起来不会那么的疲劳，且使用起来也很方便，将重要的信息用特别的颜色突出起来，使用户快速的看到所需要的信息。

### 4.2.4 申请购置设备界面



图4-6 申请购置设备界面设计图

申请购置设备界面设计图如图4-6所示，该界面由4部分组成：头部，面包屑，提示框，信息表单。信息表单与面包屑宽度相同，都为80%，信息表单由一些表格组合而成，值得注意的是右下角的“+”按钮，该按钮放置于信息表单的右下角处。颜色略显突兀，提示用户该按钮是一个功能键。

整体界面的颜色不超过5种，将每个功能键都用以突出的颜色标识出来。清晰地提示用户该按钮可以点击可以操作。

## 4.3 数据库设计

数据库是可以服务于各种应用程序的数据的组合。数据库是一个网站的灵魂。与静态网站不同，带有数据库的网页被称为动态网页，并且页面可以是多样的，特别是漂亮的，并且可以随着时间的数据变化而更新，这样页面就如同现在所做的非常好的网站（如网易和苹果）一样吸引人。MySQL数据库是一个小型数据库系统。操作简易，界面友好，用户数量多。

本网站使用MySQL，库名字叫lab，下面内容给出设计图以及表结构。

### 4.3.1 数据库概念模型设计

数据库名为lab，其中有11张表。表有：申请购置的设备表buy\_equipment、删除设备的记录表delete\_record、设备表equipment、设备与修理表的多对多关联表equipment\_has\_repaire\_record、实验室相关表lab、通知类表notice、权限表permission、修理记录表repire\_record、待审批的申请购置表request\_buy\_record、角色表role、用户表user。

（1）用户基本信息管理E-R图

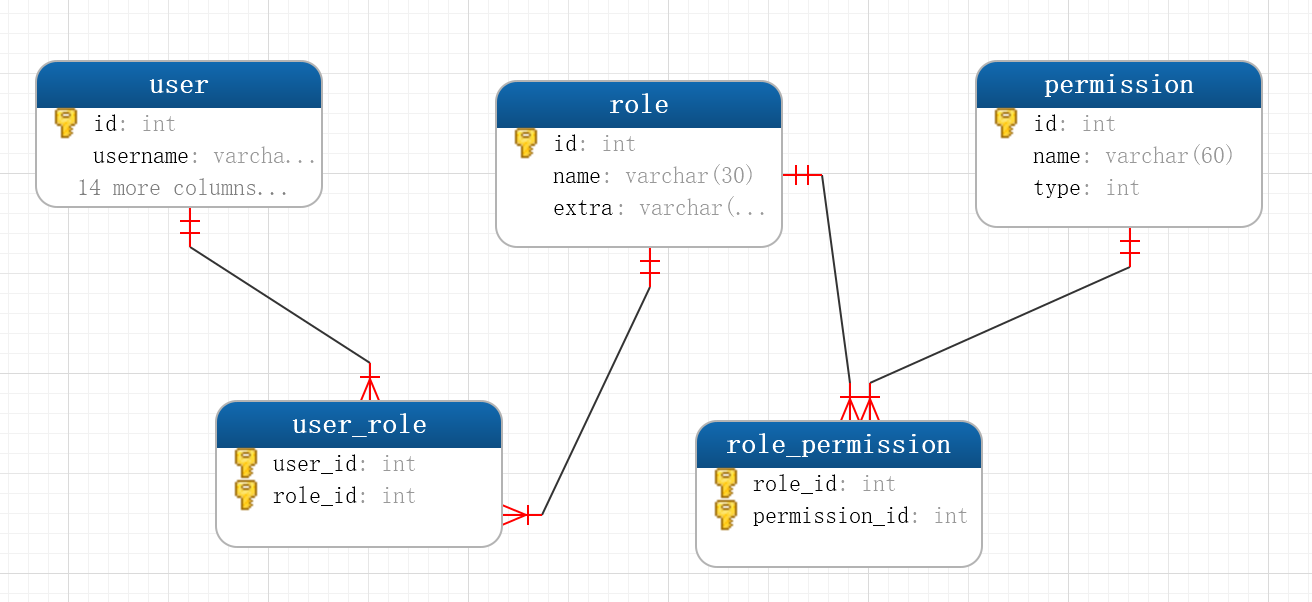


图4-7 用户基本信息管理E-R图

从图4-7可以看出，一个用户可以拥有多种身份，不同的身份拥有多种权限。这样设置为多对多的关系是由于一个用户可以拥有多种身份，反过来，一个身份可以被多个用户共同使用。因此需要建立一个额外的表来维持用户表和身份表的多对多关系。

（2）通知管理E-R图

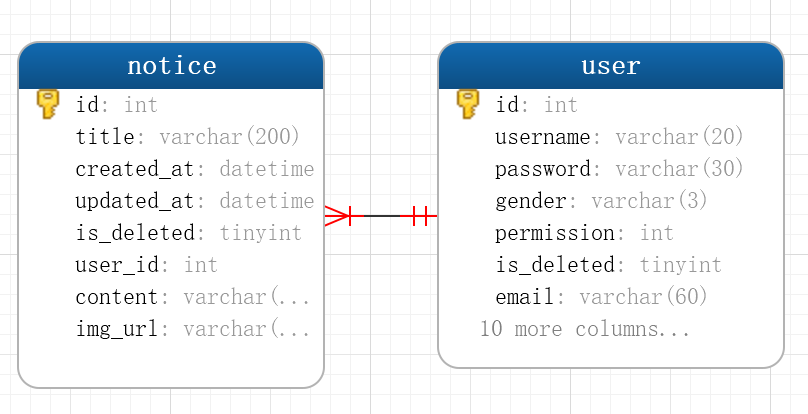


图4-8 通知管理E-R图

从图4-8可以看出，用户与通知是一对多的关系，即一个用户可以创建多个通知。

（3）实验室设备管理E-R图

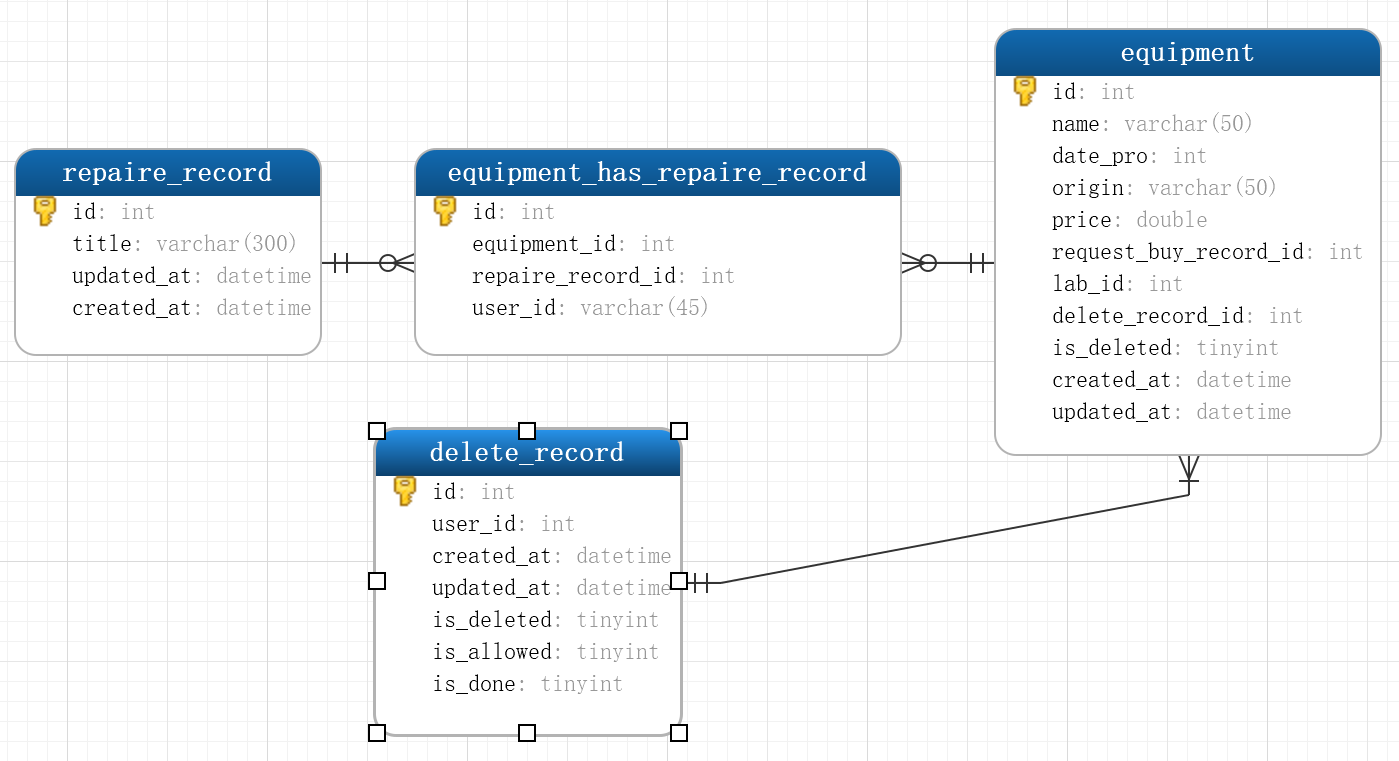


图4-9 实验室设备管理E-R图

由图4-9可以看出，设备与维修记录是多对多的关系，意味着该设备不会仅仅被修一次。设备与删除记录是多对一的关系，代表着用户可以一次性删除多个设备。

（4）实验室管理E-R图

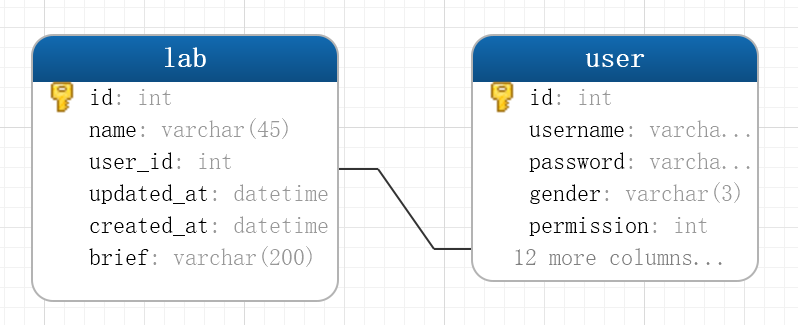


图4-9 实验室管理E-R图

由图4-9可以看出，实验室与用户之间的关系是一对一的关系。所以一个实验室当中只会存在一个负责人。

### 4.3.2 数据表结构

1. user (用户表)

用户表主要存储用户的相关信息，如用户名，密码，性别，住址，令牌，手机号，邮编等信息，以及角色的id用以表示该用户的权限。该表结构如表4-3-1所示。

表4-3-1 user表结构

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 主键 | 备注 |
| id | int | 11 | 是 | 主键 |
| username | varchar | 20 | 否 | 用户名 |
| password | varchar | 30 | 否 | 密码 |
| gender | varchar | 3 | 否 | 性别 |
| is\_deleted | tinyint | 1 | 否 | 是否被删除 |
| email | varchar | 60 | 否 | Email |
| name | varchar | 10 | 否 | 姓名 |
| address | varchar | 150 | 否 | 地址 |
| area | varchar | 60 | 否 | 地区 |
| home\_phone | varchar | 15 | 否 | 家庭电话 |
| role\_id | int | 11 | 否 | 权限角色 |
| tel\_phone | varchar | 11 | 否 | 手机号 |

续表4-3-1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| zip\_code | varchar | 20 | 否 | 邮编 |
| created\_at | datetime |  | 否 | 发布时间 |
| updated\_at | datetime |  | 否 | 更新时间 |
| token | varchar | 64 | 否 | 验证身份的令牌 |

2. equipment(设备表)

设备表主要存储一些设备的相关信息，如，设备名，产地，价格，购置记录的id，删除记录的id，创建时间等。同时该表与实验室表，购置记录表，删除记录表关联。因此含有它们的id。具体如表4-3-2所示。

表4-3-2 equipment表结构

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 主键 | 备注 |
| id | int | 11 | 是 | 主键 |
| name | varchar | 20 | 否 | 姓名 |
| origin | varchar | 3 | 否 | 供货商地址 |
| price | double | 1 | 否 | 单价 |
| request\_buy\_record\_id | int | 1 | 否 | 申请购置记录id |
| lab\_id | int | 30 | 否 | 实验室id |
| delete\_record\_id | tinyint | 1 | 否 | 删除记录id |
| is\_deleted | tinyint | 1 | 否 | 是否被删除 |
| created\_at | datetime | 15 | 否 | 创建时间 |
| updated\_at | datetime | 11 | 否 | 更新时间 |

1. lab（实验室表）

实验室表中主要存放的是实验室的相关信息，如实验室名，负责人的id，实验室的简介，以及实验室的创建时间等。具体表结构如表4-3-3所示

表4-3-3 lab表结构

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 主键 | 备注 |
| id | int | 11 | 是 | 主键 |
| name | varchar | 45 | 否 | 实验室名 |

续表4-3-3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| user\_id | Int | 11 | 否 | 负责人的id |
| updated\_at | datetime | 0 | 否 | 更新时间 |
| created\_at | datetime | 0 | 否 | 创建时间 |
| brief | varchar | 200 | 否 | 实验室简介 |

1. notice（通知表）

通知表主要存放的是用户登录之后的主界面的通知栏的内容。该表存放的主要字段有title，content，img\_url等，用以存放着通知文章的题目，内容，图片。具体表结构如表 4-3-4所示。

表4-3-4 notice表结构

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 主键 | 备注 |
| id | int | 11 | 是 | 主键 |
| title | varchar | 20 | 否 | 通知的标题 |
| created\_at | varchar | 3 | 否 | 创建时间 |
| updated\_at | double | 1 | 否 | 更新时间 |
| is\_deleted | int | 60 | 否 | 是否被删除 |
| user\_id | int | 30 | 否 | 发布的用户id |
| content | varchar | 2000 | 否 | 通知内容 |
| img\_url | varchar | 300 | 否 | 图片的链接 |

1. role (角色表)

角色表主要存放的是该系统中会存在的角色信息，如学生，老师，管理人员与系统管理员。不同的角色拥有不同的权限组。同时，该表又具有extra字段，可以记录一些备注的信息等。具体表结构如表 5-2-5所示。

表4-3-5 role表结构

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 主键 | 备注 |
| id | int | 11 | 是 | 主键 |
| name | varchar | 30 | 否 | 角色名 |
| extra | varchar | 200 | 否 | 备注 |

1. permission (权限表)

权限表存放着权限的名字，以及权限的id，还有角色的id，与role表是一对多的关系。这就代表着一个角色可以拥有多个权限。目的是让不同的角色具有不同的权限组。

表4-3-6 permission表结构

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 主键 | 备注 |
| id | int | 11 | 是 | 主键 |
| name | varchar | 60 | 否 | 权限名 |
| role\_id | int | 11 | 否 | 角色id |

1. buy\_equipment (购置设备表)

购置设备表单与设备表相似，只是用来与申请表绑定作为参考用的。该表拥有设备名，设备数量，设备单价以及实验室的id等字段，是与购置申请表具有多对一的关系。具体表结构如下图所示（表 4-3-7）。

表4-3-7 buy\_equipmen表结构

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 主键 | 备注 |
| id | int | 11 | 是 | 主键 |
| name | varchar | 60 | 否 | 设备名 |
| count | int | 11 | 否 | 设备数量 |
| price | double | 0 | 否 | 设备单价 |
| request\_buy\_record\_id | int | 11 | 否 | 申请购置记录id |
| lab\_id | int | 11 | 否 | 实验室id |
| date\_pro | datetime | 0 | 否 | 生产日期 |
| origin | varchar | 200 | 否 | 设备详细地址 |

1. request\_buy\_record (申请购置记录表)

申请购置记录表与购置设备表还有用户表关联，其存放的有购置记录的标题，创建时间，更新时间，审批时间以及审批状态等等，与用户表关联则可以显示审批人以及申请人的相关信息，展现的时候会更加清晰。具体表结构如下图所示（表4-3-8）

表4-3-8 request\_buy\_record表结构

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 主键 | 备注 |
| id | int | 11 | 是 | 主键 |
| title | varchar | 150 | 否 | 申请记录标题 |
| created\_at | datetime | 0 | 否 | 创建时间 |
| updated\_at | datetime | 0 | 否 | 更新时间 |

续表4-3-8

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| allowed\_at | datetime | 0 | 否 | 审批时间 |
| is\_deleted | tinyint | 1 | 否 | 是否删除 |
| is\_allowed | tinyint | 1 | 否 | 是否审批 |
| is\_done | tinyint | 1 | 否 | 是否操作 |
| cost | double | 0 | 否 | 总花费 |
| allow\_user\_id | int | 11 | 否 | 审批人id |
| request\_user\_id | int | 11 | 否 | 申请人id |

1. delete\_record (删除设备记录表)

删除设备记录表与用户表，设备表关联。其中与设备表的关系是多对一的关系，因此在设备表中可以看到delete\_record\_id的字段，且删除设备记录表中记录了创建时间以及更新时间还有用户的id。可以显示的时候与用户表关联，将用户的信息也一并输出出来。具体的表结构如下图(表5-2-9)所示。

表4-3-9 delete\_record表结构

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 主键 | 备注 |
| id | int | 11 | 是 | 主键 |
| user\_id | int | 11 | 否 | 操作的人的id |
| created\_at | datetime | 0 | 否 | 创建时间 |
| updated\_at | datetime | 0 | 否 | 更新时间 |
| is\_deleted | tinyint | 1 | 否 | 该记录是否被删除 |

# 5 系统详细设计

## 5.1 数据结构

### 5.1.1 相关类名

类名均使用数据库中的表名，首字母大写。且表中的数据结构应转成Java的数据结构。像varchar对应的是String，int对应的是int，datetime对应的是Date，tinyint对应的是boolean。

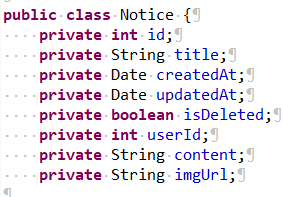


图5-1 Notice类的数据结构

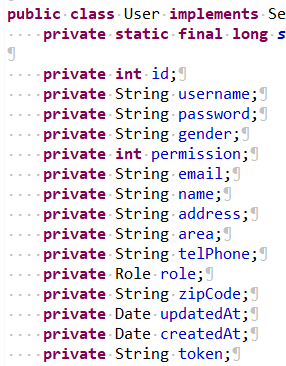


图5-2 User类的数据结构

### 5.1.2 分页的数据结构

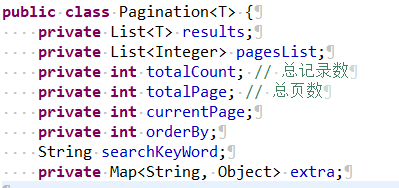


图5-3 Pagination类的数据结构

如图5-3所示，该数据结构应用于项目中的每一个列表界面。目的是封装一些数据到前台页面渲染出来。如results是从数据库中查询到的结果集，pagesList是生成好的分页列表，totalCount是根据关键字所能查到的所有结果数量，totalPage是总页数，currentPage是当前页数，orderBy是控制排序或是升序，searchKeyWord是搜索的关键字，extra是额外的信息，用于存储非公共的信息。

### 5.1.3 一些常量



图5-4 列表显示数量常量

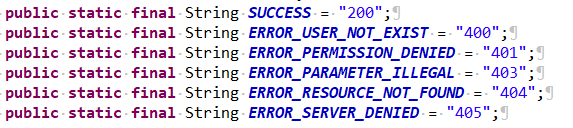


图5-5 错误代码常量

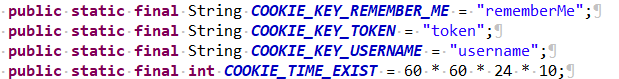


图5-5 cookie常量

该项目中存在一些常量，有些是可以写死的，有些是放到配置文件中，解决硬编码的。像图5-4中的常量，是写到properties文件中的常量，我们需要更改这些数值的时候就可以在配置文件中更改而不需要到代码中在做修改。非常友好的解决了一些硬编码问题。类似图5-5和图5-6的常量，是可以写死的。当然也是被提出来的常量，当开发人员想做修改的时候在常量类中修改即可。

### 5.1.4 命名规范

（1）工具类都以Util结尾。

（2）业务层和持久层都必须先写接口再写实现类，接口以Service/Dao结尾，实现类以ServiceImpl/DaoImpl结尾。例如：UserService，UserServiceImpl。

（3）在常量类中，常量的命名都必须大写，以下划线隔开。命名应能体现出该常量的作用。例如：COOKIE\_KEY\_TOKEN，该常量的意思是用于Cookie中的token的key值。应该能像这样看到名称就能理解其作用。

## 5.2 核心功能设计

### 5.2.1 登录

用户通过浏览器地址访问到本系统前应输入账户密码进行身份验证，验证通过后，才可以进行一些需要登录之后才能进行的操作。当验证不通过时，界面上会提示用户验证失败，重新输入账户和密码。登录流程图如图5-6所示：

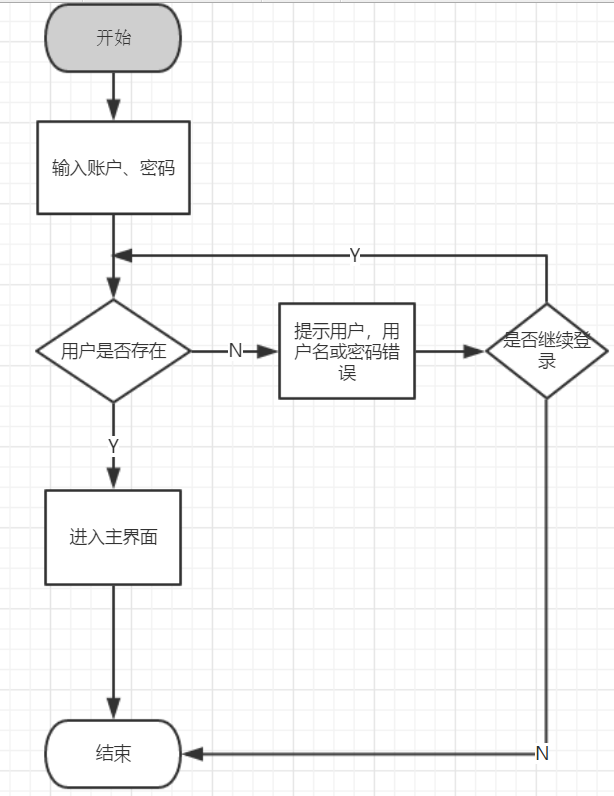


图5-6 登录流程图

### 5.2.2 购置设备

用户在“申请购买设备”界面填写相关的信息，如: 申请表的标题， 设备名， 单价， 供应商的地址， 该设备放置的实验室名称。点击提交会有一个验证，验证输入信息是否符合要求，如果验证通过。则提交申请成功，等待进一步审批。如果提交失败，如果审批成功，管理员会按照申请表中的设备信息将设备购入，并且入库。具体的流程图如图5-7所示。

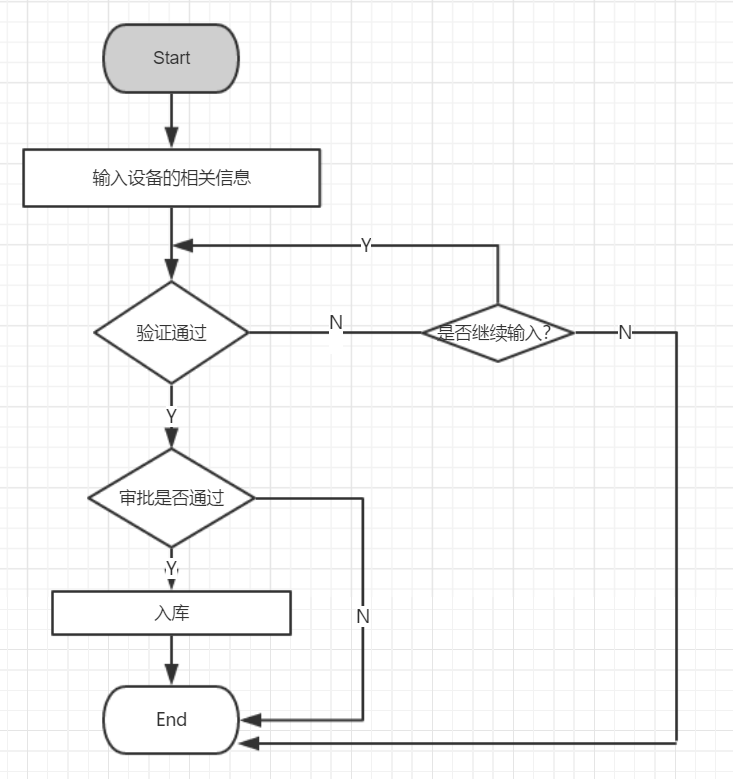


图5-7 购置设备流程图

### 5.2.3 删除设备

管理员可以对设备进行删除操作，首先是打开设备列表界面，然后选中设备，点击删除按钮。再确认弹出的确认框。即可完成对设备的删除操作，具体流程图如图5-8所示。

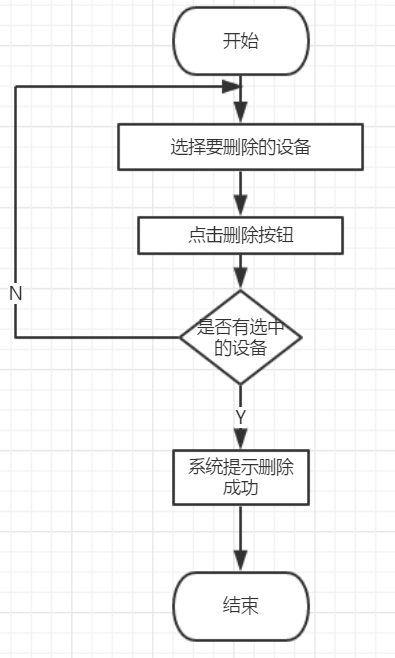


图5-8 删除设备流程图

### 5.2.4 用户管理

（1）用户的创建

系统管理员可以创建用户，输入用户的相关信息，如用户名密码，并且为其设定相应的角色，带有相应的权限。具体的流程图如图5-9所示。

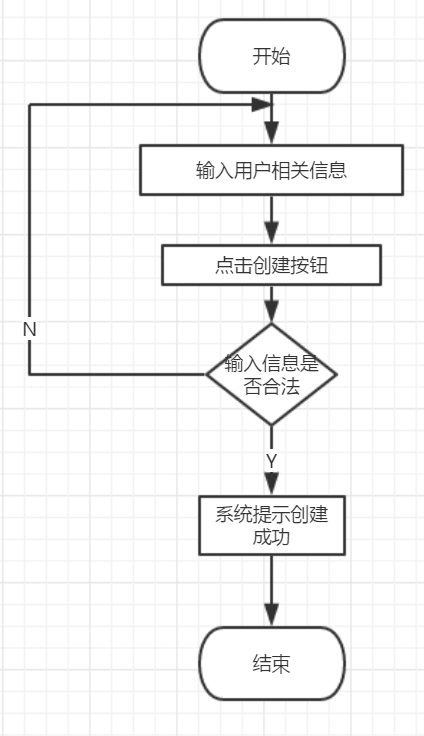


图5-9 用户创建流程图

（2）用户的删除

删除操作同设备的删除类似，同是将用户列表界面打开，然后选择想要删除的用户，点击删除按钮。点击确认确认框，选中的用户就可以被删除了。

（3）用户的修改

系统管理员可以修改用户的权限或者是个人信息。首先打开用户的详细信息界面，然后点击修改按钮，当前用户就处于可编辑状态。管理员可以输入其他信息代替该用户的个人信息，然后点击确认。这时就完成了对用户的修改。

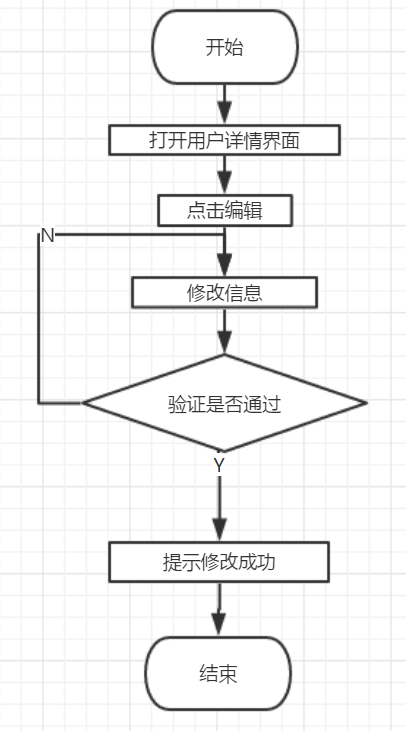


图5-10 修改用户流程图

### 5.2.5 实验室管理

（1）实验室信息管理

系统管理员或者实验室的负责人打开实验室详情界面可以浏览实验室的相关信息，也可以点击编辑按钮，对实验室的相关信息进行修改。当修改的信息符合要求的时候，如实验室名不能为空。这些条件都达到满足的时候，会提示修改成功。系统管理员也可以删除实验室。在实验室列表界面，选中要删除的实验室，点击删除即可完成删除操作。

（2）实验室分配管理

当学生申请了对某个实验室占用的申请后，如果经过该实验室负责人的审批后，该学生会被批注在该时间段中使用实验室。具体流程图如图所示。

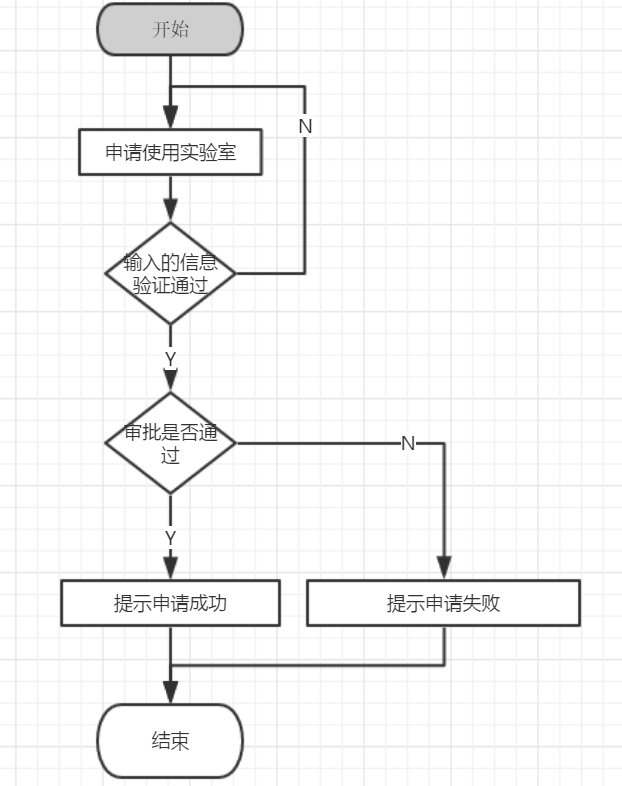


图5-11 申请实验室流程图

# 6 系统实现

## 6.1 环境搭建

本项目使用MyEclipse集成环境开发，使用其自带的Tomcat8.5，使用Java1.8作为编译语言。下载开源的SSM的jar包作为开发的框架，Spring和SpirngMVC均采用4.3版本，Mybatis采用3.4的版本。前端的界面采用了Bootstrap3框架。

## 6.2 功能模块

### 6.2.1 注册

1. 界面图



图6-1 注册界面部分截图

2. 实现逻辑

本模块是系统注册界面，实现的功能是当用户输入个人信息后，可以注册到一个可用的账号，利用该账号可以登录到实验室管理系统当中。从而知道使用该系统的人的相关信息。

本模块的逻辑很简单，界面也很朴素。当用户输入完这些信息会进行一个校验，校验所输入的信息是否为空，以及两次密码输入是否一致。当信息都输入正确之后，提交给后台，后台继续完成注册的逻辑。将该信息存入到数据库中，用以下次登录时候进行比对。

当注册成功后会提示用户注册成功，此时用户可以使用该账户进行登录了。

3. 核心代码

userService.register(user);

session.setAttribute(Constants.User.SESSION\_USER\_KEY, user);

5.1.2 登录

1. 界面图

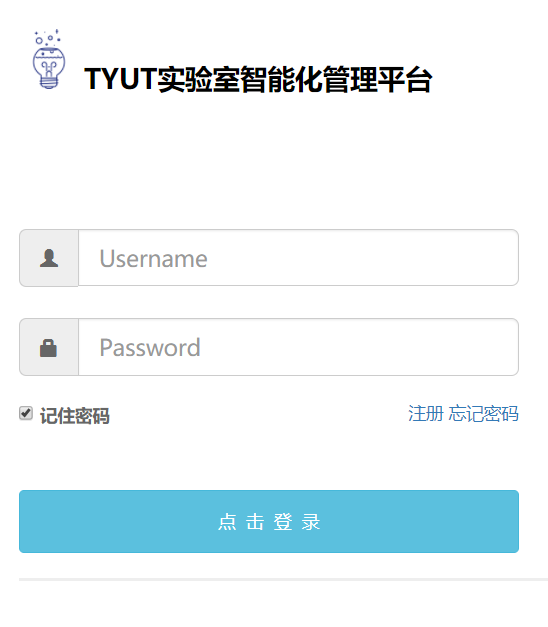


图6-2 登录界面部分截图

2. 实现逻辑

本模块是系统登录，实现的功能是当用户输入个人信息后，可以登录该系统当中。

本模块的逻辑很简单，界面也很朴素。当用户输入完这些信息会进行一个校验，校验所输入的信息是否为空，当信息都输入正确之后，提交给后台，后台继续完成登录的逻辑。在后台中根据用户输入的用户名和密码与数据库中的数据进行比对，如果查询出没有记录，则提示用户“用户名或密码错误”。

此时，还要校验用户是否点击了“记住密码”，如果用户点击记住密码，则在后台为该用户生成一个token，并将该token存到该用户表中，并且将token以及用户名都存放在cookie当中。以便完成下回在登录界面的时候，后台可以读取cookie然后读取token进行一个身份的验证。。

当登录成功后会跳转到主界面，届时用户可以各种各项业务的操作。

1. 核心代码

if (session.getAttribute(Constants.User.SESSION\_USER\_KEY) != null) {

return "redirect:/lab/toContent";

} else {

User user = userService.loginByCookie(request, response);

if (user != null) {

user.setPassword("");

session.setAttribute(Constants.User.SESSION\_USER\_KEY, user);

return "redirect:/lab/toContent";

}

}

return Constants.UserControllerUrl.USER\_CONTROLLER\_LOGIN\_PAGE\_URL;

**5.2 主界面**

1. 界面图

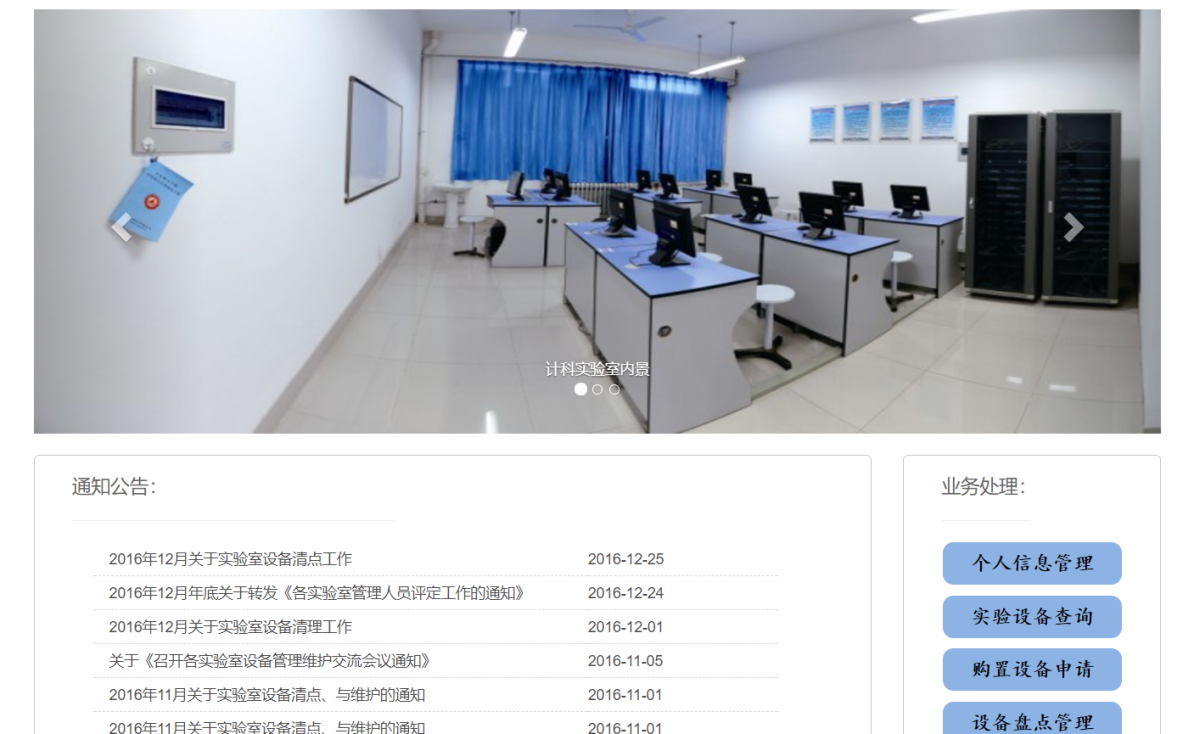


图6-3 主界面部分截图

2. 实现逻辑

该界面展现了通知公告栏，图片轮播栏，业务处理栏，还有导航栏（图片太大未显示）。可以说该界面是调度各业务的入口。该界面不需要登录，用以任何用户都可以看到通知公告一栏。

### 6.2.2 个人信息管理

1. 界面图



图6-4 修改个人信息界面部分截图

2. 实现逻辑

本模块是个人信息管理，实现的功能是当用户可以浏览自己的个人信息，如果发现某些信息需要修改，则可以输入新的数据用以更换。当然，也有部分数据不能修改，如用户名，以及身份。

本模块的逻辑与注册用户类似，当用户输入完信息，点击修改按钮，则先会在前端进行一个校验。如果用户输入信息不合法，或是某些必要的信息为空则提醒用户修改相应的信息。提示条使用SpringMVC中的FlashMessage即可实现重定向传参并且刷新后失效的功能。

当修改成功后顶部会出现个提示条，显示修改成功。

值得一提的是，前台界面选择住址可以三级联动，即当选择省的时候，市的可选内容也会随之而变化，当市选择完毕的时候，后面的区也会发生变化。当然实现该功能则需要将所有省市区的数据添加进来，当选择省的时候，动态生成市的数据，当选择市的时候，动态生成区的数据，如核心代码中的代码所示。

3. 核心代码

//循环第一步,把省循环进select

for (var i = 0; i < cityArray.length; i++) {

sheng.options[sheng.length] = new Option(city[i].name);

// 循环第二步,把所有的市都循环进select

sheng.onchange = function() {

shi.options.length = 0;

shi.options[shi.length] = new Option("请输入市");

for (var j = 0; j < city[sheng.selectedCount - 1].city.length; j++) {

shi.options[shi.length] = new Option(city[sheng. selectedCount- 1].city[j].name)

};

}

shi.onchange = function() {

qu.options.length = 0;

qu.options[qu.length] = new Option("请输入区");

for (var k = 0; k < city[sheng.selectedIndex - 1].city[shi.selectedIndex - 1].area.length; k++) {

qu.options[qu.length] = new Option(city[sheng.selectedIndex - 1].city[shi.selectedIndex - 1].area[k]);

};

}

};

### 6.2.3 实验室设备管理

1. 界面图



图6-5 实验室设备管理界面部分截图

2. 实现逻辑

用户进入该界面后，默认的显示全部实验室的设备，页面默认显示15个设备，多余的将会被分页，如果要进行各个实验室的分类，则需要点击右上角有个下拉列表，如图6-5所示，用户可以点击子选项则可以显示各实验室的设备。当然用户也可以进行搜索想要的设备，在右上角的文本输入框中输入关键词然后按回车即可搜索。该搜索逻辑是根据设备名，设备id，产地，实验室来进行模糊搜索。所以当用户输入一个信息的时候采用该方法会命中率较高。

用户也可以选择右侧的删除选择框，该选择框是多选的，因此用户可以在此页中删除多个设备。当用户选择多个选择框后，点击下方的删除按钮，会弹出一个提示框，是否删除这些设备。当删除了这些设备后，则会在该界面提示用户删除成功，并且提示用户总共删除了多少个设备。

3. 核心代码

（1）显示设备列表界面

Pagination<Equipment> pagination = equipmentService.listEquipmentsByPagination(searchKeyWord, page, orderBy, labId, user);

request.setAttribute(Constants.RecordController.REQUEST\_ATTRIBUTE\_KEY\_REQUEST\_BUY\_RECORD, pagination);

List<Lab> labs = labService.getLabs();

request.setAttribute("labs", labs);

（2）分页设备列表

Pagination<Equipment> pagination = new Pagination<Equipment>();

pagination.setSearchKeyWord(keyword);

orderBy = order == 0 ? "ASC" : "DESC";

keyword = "%" + keyword + "%";

int totalCount = equipmentDaoImpl.getCountBySearchWord(keyword, labId);

int maxPage = (int) Math.ceil(totalCount \* 1.0 / count);

if (page > maxPage) {

page = 1;

}

int start = (page - 1) \* count;

int offset = count;

List<Integer> pageIndexList = PaginationUtil.makePageIndexList(page, count, maxPage);

pagination.setCurrentPage(page);

pagination.setResults(equipmentDaoImpl.getByPagination(keyword, start, offset, labId, orderBy));

pagination.setTotalCount(count);

pagination.setTotalPage(maxPage);

pagination.setPagesList(pageIndexList);

pagination.setOrderBy(order);

（3）删除设备

String[] deletedIds = request.getParameterValues("deleted-checkboxes");

int count = 0;

try {

count = equipmentService.deleteEquipmentByIdArray(user, deletedIds);

} catch (EquipmentException e) {

}

redirectAttributes.addFlashAttribute("tip", "您成功删除了" + count + "条记录.");

### 6.2.4 购置设备

1. 界面图



图6-6 购置设备申请界面部分截图

2. 实现逻辑

用户进入该界面后，负责人信息会默认的填充为自己的，且不能更改。用以提醒用户负责人是他自己。紧接着要输入的是该申请表单的简要，可以填购买的目的。剩下的则是设备的相关信息，当然默认会显示一个设备信息表单。当用户点击右下角的加号键，在下方则会又出现一个一个设备信息的表单，用户可以填写第二个设备的相关信息。当填写完毕后，用户可以点击提交按钮。

点击“提交”后，首页将完成验证功能，以验证用户输入的数据的正确性以及是否为空。有效性验证被执行。验证通过后，首页将数据发送到背景并在后台再次检查。如果通过，信息将存储在数据库中。并通知用户创建成功。然后转到应用程序记录界面。用于查看此应用程序记录的状态。

值得一提的是，点击加号动态生成表单是用jquery操作dom元素动态生成html，用户也可以点击设备信息右侧的垃圾箱按钮，删除掉该项表单。当用户提交的时候，这些数据是通过jquery选择器获得的（数组）。

3. 核心代码

（1）前台获取用户输入的数据

let equipmentPriceElementes = $("table[id^='equipment-']").find("input[data-field='equipment\_price']");

（2）前台数据发往后台

if (isFilledTrue) {

// send ajax request

$.ajax({

url: path + "/equipment/request-buy",

type: "post",

data: JSON.stringify(equimentJson),

success: function (rdata) {

console.log(rdata);

},

error: function () {

$(".alert").attr("class", "alert alert-danger");

$(".alert").html("输入信息有误，请检查后再提交。");

}

});

}

（3）后台获取前台发来的设备（json）数据

ArrayList<Map<String, String>> equipmentList =

(ArrayList<Map<String, String>>) equipments.get(

Constants.EquipmentController.KE Y\_REQUEST\_BUY\_EQUIPMENTS\_JSON

);

### 6.2.5 审批购置设备申请表

1. 界面图



图6-7 审批购置设备界面部分截图

2. 实现逻辑

进入该界面后，可以浏览所有用户的购置请求，该页面会默认显示15条记录，默认按照时间进行倒序排序。多的记录会进行分页。如果想要快速定位到自己想看到的记录的时候，可以通过右上角的搜索按钮，输入你想检索的信息，点击搜索。后台会进行匹配过滤筛选（通过与申请人的姓名，申请表的标题比对）。然后展示出来。

也可以点击详情列的图标，点击之后则可以看到该申请表的详细内容，以及三个按钮，分别是返回，拒绝，同意（如图6-8所示）。可以视情况而审批。当审批完成后，页面会自动跳转到审批记录页面，在该页面会显示所有审批的记录。



图6-8 购置设备申请详情界面部分截图

3. 核心代码

（1）显示申请记录列表

Pagination<RequestBuyRecord> pagination = new Pagination<RequestBuyRecord>(); pagination.setSearchKeyWord(keyword);

orderBy = order == 0 ? "ASC" : "DESC";

keyword = "%" + keyword + "%";

int totalCount = recordDao.getNotReplyAllRequestBuyRecordCount(keyword);

int start = (page - 1) \* count;

int offset = count;

int maxPage = (int) Math.ceil(totalCount \* 1.0 / count);

if (page > maxPage) {

page = 1;

}

List<Integer> pageIndexList = PaginationUtil.makePageIndexList(page, count, maxPage);

pagination.setCurrentPage(page);

pagination.setResults(recordDao.getNotReplyRequestBuyRecordPagination(keyword, start, offset, orderBy));

pagination.setTotalCount(count);

pagination.setTotalPage(maxPage);

pagination.setPagesList(pageIndexList);

pagination.setOrderBy(order);

（2）审批

try {

recordService.approveRequestBuyRecord(id, operation, user);

result.put("code", Constants.ErrorCode.SUCCESS);

} catch (RecordException recordException) {

result.put("code", recordException.getErrorCode());

} catch (EquipmentException equipmentException) {

result.put("code", equipmentException.getErrorCode());

}

### 6.2.6 盘点统计实验室数据

1. 界面图

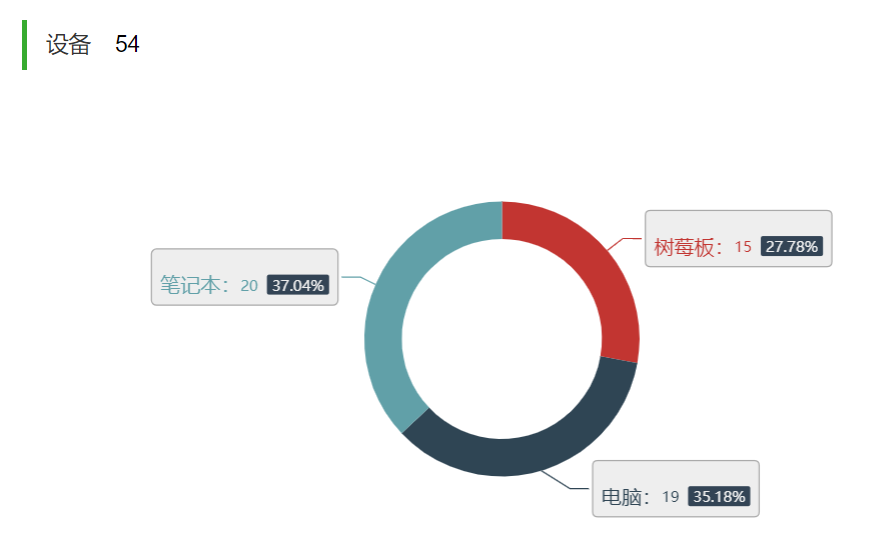


图6-9 盘点统计实验室数据界面部分截图

2. 实现逻辑

进入该界面后，可以浏览每个实验室的设备统计信息，用户可以点击任何一个实验室，即可显示相应的统计信息。还可以点击全部，则显示所有实验室的设备占比。如6-9图所示，设备右侧的54代表着该实验室的设备总数， 在下方环形图中，左侧笔记本后的20代表着它的数量为20，占该实验室的百分比是37.04%

该插件是引入的百度的echarts，上手简单。在官网示例图找到对应的模板，引入数据即可自己生成这种统计图。简捷实用。

3. 核心代码

（1）生成统计图

$.ajax({

url: path + "/lab/ajax/info",

type: "get",

data: {

labId: currentLabId

},

success: function (rdata) {

let labEquipCounts = rdata.counts;

let labEArray = new Array();

for (let i = 0; i < labEquipCounts.length; i++) {

let equipCount = {};

equipCount.value = labEquipCounts[i].count;

equipCount.name = labEquipCounts[i].equipmentName;

labEArray.unshift(equipCount);

}

labEcharts.setOption({

series:[

{

data: labEArray

}

]

})

},

error: function () {

}

});

### 6.2.7 通知管理

1. 界面图



图6-10 通知公告显示界面部分截图

2. 实现逻辑

用户点开该项目的主页的时候则会显示该通知公告栏，用以显示通知公告。默认显示最新的6条，可以点击更多以看到更多的通知公告信息。同时也是支持搜索api。用以快速定位到用户想要找的通知公告。

# 6 系统测试

## 6.1 软件测试目标

软件测试的目的是发现以前工作中发生的错误并改进整个项目日期。对于这种应用，主要从用户的角度来看，尽可能避免日常使用中的错误，并且总体体验受到影响。从总体逻辑结构上实施门禁检测，核查是否满足初步设计，是否完成要求的要求，符合管理体系的要求。因此，完全掌握了最终项目的总体应用，并对整个项目进行了修正。

## 6.2 软件测试方法

测试的方法有三种：传统测试、功能验证、系统测试。

传统的测试方法包括开发人员进行的单元测试基于系统功能设计测试用例。这些测试非常针对该系统。这种类型的测试非常适合很少与其他任何交互的代码。

功能验证作为测试过程，要求设计人员了解系统的核心功能并能够准确定位系统。此测试的目的是查看系统的核心功能是否不符合规范。

系统测试与其他测试不同之处在于核心功能验证再次运行此测试。他认为这个系统是一个很大的系统，从整个系统开始就是错误的。了解网络的工作方式以及Web服务如何相互交互。由于通常的系统测试等待生命周期的结束，所以时间很紧。系统测试阶段经常被忽视，所以一些小问题不能被基石发现。因此，在查找问题代码时，有必要完善系统测试设计。

## 6.3 主要模块测试

### 6.3.1 登录模块测试

表6-1 登录模块测试表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 项目功能模块 | 登录模块 | | | |
| 测试目的 | 验证用户登录的信息的有效性 | | | |
| 预置条件 | 项目正常运行 | | | |
| 测试数据 | 用户名=yjz 密码=yjz123 | | | |
| 操作步骤 | 操作描述 | 数据 | 期望结果 | 实际结果 |

续表6-1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 输入用户名，按“登录”按钮 | 用户名=yjz  密码为null | 显示警告信息“请输入用户名和密码！” | 系统弹出模态框提示 “请输入用户名和密码！” |
| 2 | 输入密码，按“登陆”按钮。 | 用户名为null  密码=yjz123 | 显示警告信息“请输入用户名和密码！” | 系统弹出模态框提示 “请输入用户名和密码！” |
| 3 | 输入用户名和密码，按“登陆”按钮。 | 用户名=yjz  密码为yjz1234 | 显示警告信息“用户名或密码错误，请重新输入！” | 系统弹出模态框提示 “用户名或密码错误，请重新输入！” |
| 4 | 输入用户名，密码，按“登录”按钮 | 用户名=yjz  密码为yjz123 | 进入主界面 | 进入主界面 |
| 5 | 输入用户名，密码，按“登录”按钮 | 用户名=yjz4  密码为yjz123 | 显示警告信息“用户名或密码错误，请重新输入！” | 系统弹出模态框提示“用户名或密码错误，请重新输入！” |
| 6 | 输入用户名，按“登录”按钮 | 用户名=yjz  密码为null | 显示警告信息“请输入用户名和密码！” | 系统弹出模态框提示 “请输入用户名和密码！” |

测试结果：

登录模块测试首先是校验用户输入的用户名和密码，当用户名和密码都不能为空的时候才能登录。后台接受到用户名密码的时候会从数据库中进行比对，如果找不到该值的时候，会提示用户用户名和密码有误，且测试用例结果如表6-1所示，均达到预期结果。

### 6.3.2 申请购置设备模块测试

表6-2 申请购置设备模块测试表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 项目功能模块 | 申请购置设备模块 | | | |
| 测试目的 | 验证输入的信息是否满足数据库存储的要求 | | | |
| 预置条件 | 项目正常运行，用户拥有该权限 | | | |
| 测试数据 | labname=生物实验室，title=测试, ename=设备, ecount=10, eprice=12, eaddress=aa | | | |
| 操作步骤 | 操作描述 | 数据 | 期望结果 | 实际结果 |
| 1 | labname=生物实验室，title=测试, ename=设备, ecount=10, eprice=12, eaddress=aa  点击提交 | title=测试, ename=设备, ecount=10, eprice=12, eaddress=aa, labname=生物实验室 | 跳转到申请记录界面 | 跳转到申请记录界面 |
| 2 | labname=生物实验室，title=null, ename=设备, ecount=10, eprice=12, eaddress=aa点击提交 | title=null, ename=设备, ecount=10, eprice=12, eaddress=aa, labname=生物实验室 | 显示警告信息“申请简要不能密码为空显示警告信！” | 显示警告信息“申请简要不能为空！” |

续表6-2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 3 | labname=生物实验室，title=测试, ename=null, ecount=10, eprice=12, eaddress=aa点击提交 | title=测试, ename=null, ecount=10, eprice=12, eaddress=aa, labname=生物实验室 | 显示警告信息“设备的各项信息不能为空！” | 显示警告信息“设备的各项信息不能为空！” |
| 4 | labname=生物实验室，title=测试, ecount=eawe, ecount=10, eprice=12, eaddress=aa  点击提交 | title=测试, ename=设备, ecount=eawe, eprice=12, eaddress=aa, labname=生物实验室 | 显示警告信息“设备数量需为整数” | 显示警告信息“设备数量需为整数” |

测试结果：

该申请购置设备模块需要对用户输入的信息进行一个校验，当用户输入的参数不合法时，该系统会提醒用户有误，且需要重新输入再提交。当用户输入的数据符合系统的要求时，会跳转到申请记录页面，因而用户可以查看到自己的提交记录。且用例测试如表6-2所示，均符合预期结果。

### 6.3.3 修改个人信息模块测试

表6-3 修改个人信息模块测试用例表

|  |  |
| --- | --- |
| 项目功能模块 | 修改个人信息模块 |
| 测试目的 | 验证用户输入的个人信息能否正常的修改 |
| 预置条件 | 项目正常运行，用户登录成功且打开个人信息管理界面。 |

续表6-3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试数据 | gender=男，name=杨嘉志，province=山西，city=临汾，area=侯马市， address=山西省侯马市，zip=043000，email=171882105@qq.com | | | |
| 操作步骤 | 操作描述 | 数据 | 期望结果 | 实际结果 |
| 1 | 1.依次输入  gender=男，name=杨嘉志，province=山西，city=临汾，area=侯马市， address=山西省侯马市，zip=043000，email=171882105@qq.com 2.点击修改 | gender=男，  name=杨嘉志，  province=山西，  city=临汾，  area=侯马市，  address=山西省侯马市，zip=043000，email=171882105@qq.com | 页面提示修改成功 | 页面提示修改成功 |
| 2 | 1.清空数据，依次输入  gender=男，name=，province=请输入省，city=请输入市，area=请输入区， address=，zip=，email=  2.点击修改 | gender=男，  name=，  province=请输入省，  city=请输入市，  area=请输入区，  address=，  zip=，  email= | 页面提示修改成功 | 页面提示修改成功 |
| 3 | 1.清空某些数据，依次输入  gender=男，name=，province=山西，city=临汾，area=请输入区， address=山西省侯马市，zip=，email=  2.点击修改 | gender=男，  name=，  province=山西，  city=临汾，  area=请输入区，  address=山西省侯马市，  zip=，  email= | 页面提示修改成功 | 页面提示修改成功 |

测试结果：

该修改个人信息模块，用户可以任意的修改自己的信息。修改成功后会提示用户修改成功。测试的用例结果如表6-3所示，均符合预期结果。

因此，该修改个人信息模块符合预期结果。

### 6.3.4 查询实验室设备模块测试

表6-4 查询实验室设备模块测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 项目功能模块 | 查询实验室设备模块 | | | |
| 测试目的 | 验证根据当前页数，实验室的id，查询的关键词能否正常的显示设备列表 | | | |
| 预置条件 | 项目正常运行，用户登录成功且打开实验室设备查询界面。 | | | |
| 测试数据 | searchKeyWord=计算器，page=1，labId=4 | | | |
| 操作步骤 | 操作描述 | 数据 | 期望结果 | 实际结果 |
| 1 | 1. 输入搜索的内容“计算器”  2. 按回车或者点击搜索按钮。 | searchKeyWord=计算器，  page=1  labId=0 | 页面显示在全部实验室中搜索到的结果 | 页面显示在全部实验室中搜索到的结果 |
| 2 | 1. 操作步骤同1  2. 在下拉框中选择计科实验室 | searchKeyWord=计算器  page=1  labId=4 | 页面显示在计科实验室中搜索到的结果 | 页面显示在计科实验室中搜索到的结果 |
| 3 | 1. 操作步骤同2  2. 选择第2页 | searchKeyWord=计算器  page=2  labId=4 | 页面显示在计科实验室中搜索到的结果，且当前页是第二页 | 页面显示在计科实验室中搜索到的结果，且当前页是第二页 |
| 4 | 1. 操作步骤同1  2. 在URL中将page的参数改为“啊” | searchKeyWord=计算器  page=啊  labId=0 | 页面依然会显示搜索计算器的结果，且当前页是第一页。 | 页面依然会显示搜索计算器的结果，且当前页是第一页。 |

续表6-4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 5 | 1. 操作步骤同1  2. 在URL中将page的参数改为 -10086 | searchKeyWord=计算器  page=-10086  labId=0 | 页面依然会显示搜索计算器的结果，且当前页是第一页。 | 页面依然会显示搜索计算器的结果，且当前页是第一页。 |

测试结果：

该模块对输入的参数进行了校验，且设了默认值，当用户手动的修改url中的参数时。如果其不符合参数的格式，系统将会采用默认值代替不合法的参数。因而该模块符合预期结果。

### 6.3.5 盘点统计模块测试

表6-5 盘点统计模块测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 项目功能模块 | 盘点统计模块 | | | |
| 测试目的 | 验证该模块能否正确的统计到设备的结果 | | | |
| 预置条件 | 项目正常运行，用户登录成功且打开盘点统计模块界面。 | | | |
| 测试数据 | labId=4 | | | |
| 操作步骤 | 操作描述 | 数据 | 期望结果 | 实际结果 |
| 1 | 1. 点击左侧的计科实验室 | labId=4 | 该页面正常显示统计该实验室的设备情况结果 | 该页面正常显示统计该实验室的设备情况结果 |

续表6-5

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2 | 1. 到设备查询界面。  2. 删除任何一项计科实验室的设备。  3. 打开盘点设备统计界面。  4. 点击左侧的计科实验室。 | labId=4 | 该页面正常显示计科实验室的设备统计结果。且设备数量因删除而变化。 | 该页面正常显示计科实验室的设备统计结果。且设备数量因删除而变化。 |
| 3 | 1. 操作同1.  2. 点击左侧的物联网实验室。 | labId=5 | 该页面统计结果随左侧点击的实验室不同而产生不同的结果。且能显示到界面上。 | 该页面统计结果随左侧点击的实验室不同而产生不同的结果。且能显示到界面上。 |

测试结果：

该模块主要是测试盘点统计模块功能能否正常运行，当打开该界面的时候，会默认选择到labId=1的实验室上。因而能看到该实验室下的设备统计情况。当点击左侧的化学实验室的时候，数据也能切换到化学实验室的相关结果。当在设备界面中删除一两个设备时，再回到盘点界面，数据也会随之而变化。具体的测试用例如表6-5所示。测试结果该模块符合预期结果。

## 6.4 测试总结

通过运用不同 的测试方法对实验室管理系统进行测试，发现了在设计阶段基础考虑不周的地方以及最终实现效果与设计不一致的情况。首先，在申请购置设备界面，最初的设计并没有考虑到输入的数量是非整数，造成了应用的缺陷。在完善整个申请购置设备后，通过了该项测试。其次，作为web应用程序。session的存在时间最长为30分钟，如果当一个用户静止在某个页面30分钟后，那么当用户在点击这个页面的某个超链接向后台发送一个请求的时候，会找不到上一个会话。因此登录会失效，在该系统中，当静止30分钟后，用户发送一次请求，用户的界面会被重定向到登录界面。这是正确的，但是当用户紧接着登录了之后，无法跳转到刚刚失效但是请求的那个界面。

# 总 结

当初选择这个题目的时候，其实还未学习SSM框架，当时在学校学习的是移动互联，因此未涉及过Java的后台开发。更不用提及SSH框架。因此为了扩大知识面，此次毕设选择以开发web为主。在选择框架上，从网上搜集资料，从SSH与SSM两个框架中筛选一个最好的来学习与应用，因为从安全性上来讲，也从框架的使用最后定下来选择了SSM框架，MyBatis的自由的SQL语句编写，SpringMVC的注解映射，Spring的依赖注入等等框架的好处，优点体会的是淋漓尽致。

本次毕业设计不仅仅是单纯的为了完成一个项目，也锻炼了我的自学能力。这是尤为重要的，在公司实习的一段时间内，公司也是看重的是人的学习能力。此次毕业设计，我先学习了SSM框架的使用，后来又学习了SSM的底层。并在写毕业设计之前手写了一个简易的SpringMVC+Spring框架。这样使我对于框架的应用有个更深的了解，也看到了编程的魅力。

此次毕业设计，尝试将大学所学的理论知识应用到实践当中来，从开始定好题目后，开始编写文档，包括：需求分析，概要设计，详细设计。将功能明确之后，再将需要用到的技术栈罗列出来，将项目的框架勾勒出来。再将在企业学习到的时间管理应用到项目当中来，即罗列好功能列表，为每个功能定义一个最后期限。是根据自身能力而定。不能太长，也不能太短。

当然，在此次毕业设计中也难免遇到困难，如某个功能的实现方法不太会，某些bug不会改。每当遇到这些困难时候，我都会先自行解决，上博客网，或是外国的论坛寻求帮助，如Stack Overflow。虽然都是英文，但都是简单的词汇很容易理解。通过这些方法都无法解决的问题，我会寻找指导老师，同学以及我的同事寻求帮助，共同探讨解决。每当解决一个bug的时候，总会觉得自己的能力又得到了进一步的提升。

通过此次毕业设计，让我意识到了。未来如果希望能在程序员这条道路上走的更远的话，终身学习是非常必要的。对一个工具或是框架不能浅尝辄止，认为掌握了使用方法就沾沾自喜。要知其所以然，从它的源码下手，进行一个更深层次的学习。这样才能掌握的更精，使用起来也就会得心应手。最重要的是态度，对于自己所写的代码，要负责。这点我从实习到毕业设计都有深深的体会，对于自己的产品一定要负责，如果产出一个充满bug的产品，没有人会为你的产品买单。到头来被淘汰的终将会是自己。所以一定要抱有认真的态度做好每一件事。

# 参考文献

1. 丁毓峰, 毛雪涛. Java Web开发教程[M]. 北京:人民邮电出版社, 2017
2. 李新良, 游新娥, 禹云. JAVA面向对象程序设计教程[M]. 北京:北京理工大学出版社, 2016
3. Chris, Aquino, Todd, Gandee. Web开发权威指南[M]. 北京:人民邮电出版社, 2017.
4. Budi, Kurniawan, Paul, Deck. Servlet、JSP和Spring MVC初学指南[M]. 北京:人民邮电出版社, 2016
5. 孙卫琴. Tomcat与Java Web开发技术详解[M]. 第2版. 电子工业出版
6. 陈永政, 张正龙. Java EE框架技术[M]. 西安: 西安电子科技大学出版社, 2017
7. 丁旭. 基于B/S架构的软件项目实训[M]. 北京:清华大学出版社, 北京交通大学出版社, 2011
8. Walter, Savitch. Java[M]. Beijing:Higher Education Press, 2003
9. Alessandro, Bozzon, Philippe, Cudré-Mauroux, Cesare, Pautasso. Web engineering[M]. Switzerland :Springer, 2016.
10. Sestoft. Java precisely[M]. Cambridge, MA:The MIT Press, 2016
11. 杨海鹰, 潘华. 实验室信息管理系统[M]. 北京:化学工业出版社, 2007
12. 任光辉，赵扬，彭志广. 基于网络平台的实验室管理系统研究与实践[J]. 实验室科学, 2009, (05): 116-118
13. 侯亚彬, 宫德龙, 雷进生, 丁颖. 实验室研究与探索[J]. 实验室研究与探索, 2012, (04): 372-374

# 致 谢

作者在设计（论文）期间都是在谢红薇教授全面、具体指导下完成进行的。谢老师渊博的学识、敏锐的思维、民主而严谨的作风使学生受益非浅，并终生难忘。

感谢谢红薇教授在毕业设计工作中给予的帮助。

感谢我的学友和朋友对我的关心和帮助。

# 外文原文

**Database Management System**

**Source**：Database and Network Journal

**Author**：David Anderson

You know that a data is a collection of logically related data elements that may be structured in various ways to meet the multiple processing and retrieval needs of organizations and individuals. There’s nothing new about data base-early ones were chiseled in stone, penned on scrolls, and written on index cards. But now database are commonly recorded on magnetically media, and computer programs are required to perform the necessary storage and retrieval operations.

The system software package that handles the difficult tasks associated with created, accessing, and maintaining database records is in a DBMS package establish an interface between the database itself and the users of the database. (These users may be applications programmers, managers and others with information needs, and various OS programmers.)

A DBMS can organize, process, and present selected data elements from the database. This capability enables decision makers to search. Probe, and query data contents in order to extract answers to nonrecurring and unplanned questions that aren’t available in regular reports. These questions might initially be vague and/or poorly defined, but people can “browse” through the database until they have the needed information. In short, the DBMS will “manage” the stored data items and assemble the needed items from the common database in response to the queries of those who aren’t programmers. In a file-oriented system, users needing special information may communicate their needs to a programmers, who, when time permits, will information. The availability of a DBMS, however, offers users a much faster alternative communications patch (see figure).

Special, direct, and other file processing approaches ate used to organize and structure data in single files. But a DBMS is able to integrate data elements from several files to answer specific user inquiries fir information. This means that the DBMS is able to structure and tie together the logically related data from several large files.

Logical structures. Identifying these logical relationships is a job of the data administrator. A data definition language is used for this purpose. The DBMS may

then Employ one of the following logical structuring techniques during storage access, and retrieval operation: list structures, hierarchical (tree) structures, and network structures, relational structures.

1. List structures. In this logical approach, records are linked together by the use of pointers. A pointer is a data item in one record that identifies the storage location of another logically related record. Records in a customer master file, for example, will contain the name and address of each customer, and an account number identifies each record in this file. During an accounting period, a customer may maintain an invoice file to reflect these transactions. A list structure could be used in this situation to show the unpaid invoices at any given time. Each in the customer file would point to the record location of the first invoice for that customer in the invoice file. This invoice record, in turn would be linked to later invoice for the customer. The last invoice in the chain would be identified by the use of a special character as a pointer.
2. Hierarchical structures. In this logical approach, data units are structured in multiple levels that graphically resemble an “upside down” tree with the root at the top and the branches formed below, there’s a superior-subordinate relationship in a hierarchical structure. Below the single-root data component are subordinate elements (or one) has only a single owner. Thus, as we see in figure, a customer owns an invoice, and the invoice has subordinate items. The branches in a tree structure are not connected.
3. Network structures. Unlike the tree approach, which dose not permit the connection of branches, the network structure permits the connection of the nodes in a multidirectional manner. Thus, each node may have several owners and may, in turn, own any number of other data units. Data, management software permits the extraction of the needed information from such a structure by beginning with any record in a file.
4. Relational structures. A relational structure is made up of many tables. The data are stored in the form of “relations” in these tables. For example, relation tables could be established to link a college course with the instructor of the course, and with the location of the in order to find the name of the instructor and the location of the English class, the course/instructor relation is searched to get the name, and the course/location relation is searched to get the class location. Many other relations are of course, possible. This is a relatively new database structuring approach that’s expected to be widely implemented in the future.
5. Physical structure. People visualize or structure data in logical ways for there Own purposes. Thus, records R1 and R2 may always be logically linked and processed in sequence in one particular application. However, in a computer system it’s quite possible that these records that are logically contiguous in one application are not physically stored together. Rather, the physical structure of the I/O and storage devices techniques used, but also on the different logical relationships that users may assign to the data found on R1 and R2. For example, R1 and R2 may be records of credit customers who have shipments send to the same block in the same city every two weeks. From the shipping department manager’s perspective, then, R1 and R2 are sequential entries on a geographically organized shipping report. But may be identified, and their accounts may be processed, according to their account numbers which are widely separated. In short, then the physical location of the stored records in many computer-based information systems is invisible to users.

During the past five years, Microsoft has promoted Data Access Objects (DAO), and then Remote Data Objects (RDO), and now ActiveX Data Objects (ADO) as the primary data access technology for Visual Basic developers. It seems that Microsoft has been pushing a different data access technology with each successive version of Microsoft Visual Studio. Today, new versions of ADO are available on Microsoft's Web site and ship with other products and technologies, such as Microsoft Windows 2000, Microsoft Windows NT 4 Service Packs, Microsoft Internet Explorer versions 3 and later, Microsoft SQL Server 6.5 Service Pack 5 and SQL Server 7, Microsoft Office 2000, and even Microsoft Expedia Streets & Trips 2000.

One of the goals of ADO is to simplify data access. ADO is built upon some fairly complex technologies—OLE DB and ODBC (open database connectivity)—and is designed to allow you to programmatically access and modify data stored in a wide variety of databases. This broad reach is a departure from previous data access technologies. For the sake of comparison, let's take a quick glance at ADO's predecessors: DAO and RDO.

Data Access Objects

DAO was originally designed to interact with Microsoft Access databases. Although you can use DAO to access SQL Server and Oracle databases, many developers complain about DAO's performance with these large database systems.

Others complain that DAO doesn't permit programmers to access some of the richer, more powerful features of SQL Server and Oracle, such as output and return parameters on stored procedures.

One of my coworkers likes to say that using DAO to work with an Oracle database is like performing brain surgery on you…without anesthetics…while wearing oven mitts. Extreme? Yes—but he does have a point. DAO is tuned to work with desktop databases, not client/server databases. Frustrated by DAO's performance and access limitations, developers who wanted to work with SQL Server and Oracle databases generally sought other options.

Remote Data Objects

Microsoft provided another option in RDO, which originally released with Visual Basic 4 Enterprise Edition. RDO's object model closely resembles the hierarchy of structures in the ODBC API. Programmers found that RDO provided much faster access to client/server database systems, such as SQL Server and Oracle, than DAO did. Although those familiar with the ODBC API quickly learned how to work with the RDO object model, developers lacking experience with that API, such as those who had been using DAO, found the RDO technology difficult to use.

The object model itself wasn't the problem for most programmers learning RDO: the nuances inherited from the ODBC API posed the greatest obstacles. Suddenly, programmers had to bone up on cursors and bookmarks. They had to learn many of the ins and outs of specific database systems. Does the error message "The connection is busy with results from another hstmt" ring any bells out there? If you try to do the impossible on an ODBC connection to your database, RDO won't save you. Instead, you'll get that error. DAO hid the problem from you by automatically creating another connection to your database to perform the action you requested.

Another challenge that RDO posed for programmers accustomed to writing DAO code was that RDO lacked many of DAO's features, such as sorting, searching, and filtering. Other DAO functionality unavailable in the RDO world includes data definition language (DDL) interfaces to ODBC API functions such as Create Table and Create Field.

Best of Both Worlds: ActiveX Data Objects

Programmers clamored for a data access technology that combined the simplicity and relative ease of use of DAO with the speed, power, and control of RDO. Initially introduced as part of the Microsoft Internet Information Server 3 package,

ADO was intended to be all things to all people. Of course, such lofty goals are rarely fulfilled.

While the initial release of ADO lacked many of Rod’s features, I believe that ADO 2.0 offered comparable functionality. Certain RDO features, such as mixed cursors, have yet to be implemented in ADO, but these features are few and far between. In fact, I'm at a loss to name a single significant feature available in RDO that was not available in ADO 2.0 in one form or another. (I'm sure someone will tell me otherwise; a great way to find such features is to make a statement like that in a book like this.)

With the release of version 2.1, ADO and its supporting libraries began offering nearly all features available in DAO. DDL libraries were added to ADO in version 2.1 to provide functionality similar to functions available with DAO, such as Create Table, Create Field, and Create Index. Microsoft Jet and Replication Objects (JRO) in ADO 2.1 offers much of the Jet-specific functionality available via the DB Engine object in DAO. ADO 2.1 also added functionality to simplify the retrieval of newly generated identity values. ADO 2.5 adds no new functionality to more closely match the capabilities of DAO and RDO, because perhaps the only place where ADO lags behind DAO is in its searching and filtering capabilities.

So ADO has most of the functionality of RDO and DAO as well as many helpful features not available in previous data access technologies.

Database Management

There are problems with traditional data management. A more subtle problem is data dependency. When a problem’s logic is tied to it’s physical data structure, changing that structure will almost certainly require changing the program. As a result, programs using traditional access methods can be difficult to maintain. The solution to both problems id often organizing the data as a single, integrated database. The task of controlling access to all the data can then be concentrated in a centralized database management system.

How dose the use of a centralized database solve the data redundancy problem? All data are collected and stored in a single place; consequently, there is one and only one copy of any given data element. When the value of an element (an address, for example) changes, the single database copy is corrected. Any program requiring access to this data element gets the same value, because there is only one value.

How dose a database help to solve the data dependency problem? Since the

responsibility for accessing the physical data rests with the database management system, the programmer can ignore the physical data structure. As a result, programs tend to be much less dependent upon their data, and are generally much easier to maintain. Expect the trend toward database management to continue.

# 外文翻译

数据库管理系统

来源：数据库和网络杂志

作者：大卫·安德森

众所周知，数据库是逻辑上相关的数据源集合。这些数据源可以按照不同的结构组织起来，以满足单位和个人的多方面的要求。数据库本身并没有什么新东西——早期的数据库凿在石头上，记在名册上，以及写在索引卡中。而现在普遍记录在可磁化的介质上，并且需要用计算机程序来执行必须的存储和检索操作。

处理与创建、访问以及维护数据库记录有关的复杂任务的系统软件包叫做数据管理。DBMS软件包中的程序在数据库及其用户间建立接口（这些用户可以是应用程序员）。

DBMS可组织、处理和显示从数据库选择的数据源。该功能可以是决策者搜索、试探和查询数据库的内容，从而对正式报告中没有的、不再出现的、且无计划的问题做出回答。这些问题最初可能是模糊的并且/或者是定义拙劣的，但是人们可以浏览数据库知道获得数据的答案。简言之， DBMS将“管理”存储的数据项，并从公共数据库中汇集所需的数据项来响应那些非程序员的询问。在面向文件的系统中，需要特定的用户可以将它们的要求传送给数据员，并准备信息。但是使用DBMS可为用户提供一个更快的、用户可选择的通信方式。

顺序的直接的以及其它的文件处理方式通常用于单个文件中的组织和结构，而DBMS可综合多个文件的数据项，并回答某个用户对信息的查询。这意味着DBMS能够访问和检索非关键字字段的数据，即 DBMS能够将几个大文件中逻辑相关的数据组织并联在一起。

逻辑结构。确定这些逻辑关系是数据库管理者的任务，由数据定义语言完成。DBMS在存储、访问和检索操作过程中可以选用的逻辑技术有：1表结构2层次（树型）结构3网状结构4关系性结构。

1.表结构。在该逻辑方式中，记录通过指针连接在一起。指针是记录中的一个数据项，它指出另一个逻辑相关的记录的存储位置。例如，顾客主文件中的记录将包含每个顾客的姓名和地址，而且该文件中的每个记录都由一个账号标识。在记帐时期，顾客可以在不同时间购买许多东西。因此，公司保存一个发票文件来反应这些交易，这种情况下，可使用表结构显示任意时间为支付的发票。顾客文件中的每个记录都将包含这样一个记录，该链的最后一个记录有一个作为指针的特殊字符标识。

2.层次结构。该逻辑方式中，数据单元的多级结构类似一棵倒立的树，该树

的树根在顶部，而树枝向下延伸。在层次结构中存在主从关系，唯一的根数据下是从属的元或节点，而每个从属的元或节点又一次“拥有”一个或多个元或节点。该结构中根下面的每个元或树枝都只有一个所有者，这样一个用户拥有一个发票，而发票又有一个从属。在树型结构中，树枝不能相连。

3.网状结构。网状结构不像树型结构那样不允许树枝相连，它允许节点间多个方向相连。这样每个节点都有几个所有者，而它又可能有任意多个其它数据单元。数据管理软件允许从文件的任意记录开始提取该记录中的所有信息。

4.关系型结构。关系型结构由许多表组成，数据则以关系的形式存储在这些表中。例如，可以建立一些表将大学课程与该课程的老师以及上课的地点连接起来。为找到英语课的上课地点和教师名，现查询课程/教师关系得到名字，然后再查询课程/地点关系得到地点，当然也可以得到其它关系。这是一个新颖的数据组织技术，将来有望得到广泛应用。

5.物理结构：人们总是为了各自的目的，按逻辑方式设想或组织数据。这样，在一个具体应用中，记录R1和R2是逻辑相连且是按顺序处理的。但是，在计算机系统中，这些在一个应用中逻辑上邻接的记录，物理上完全可能不存储在一起。即录在介质和硬件中的物理结构不仅取决于用户定义的R1和R2中的数据的逻辑关系。例如，R1和R2可能是持有信用卡的用户记录而用户要求每两周将货物运送到同一个城市的同一个街区。而从运输部的管理规则看，R1和R2是按地理位置组织运输记录的顺序项。都是在A/R应用中，可找到R1和R2表示的顾客，并可根据其完全不同的账户处理它们的账目。简言之，在许计算机化的信息记录中，存储记录的物理单元，用户是看不见的。

关于数据访问技术的发展和最好的数据访问技术（ADO）

在过去的五年里，微软不断推出开发人员所用的主要数据访问技术，首先是Data Access Objects（DAO），然后是Remote Data Objects（RDO），现在是ActiveX Data Objects（ADO）。现在，新版本的ADO已经出现在微软的WEB站点上。ADO的目标之一是简化数据访问。ADO建立在一些相当复杂的技术上，如OLE Db 和 ODBC（开放式数据库连接），并且是为了能够对存放于很多种不同类型的数据库中的数据以编程方式进行访问和修改而设计的。正是这种更为广泛的访问范围，使ADO有别于以前的数据访问技术。为了便于比较，我们首先简单地看看ADO以前的技术：DAO 和RDO。

数据访问对象（DAO）

DAO 最初是为了与Microsoft Access 数据库进行交互而设计的。虽然可以使用DAO访问SQL Server 和 Oracle 数据库，但是很多开发人员都抱怨DAO应用于这些大型数据库时性能太差。还有人抱怨DAO不允许程序员访问SQL Server 和 Oracle 的一些更为珍贵的、功能更为前大的性能，例如存储过程中的输出和返回参数等。使用ADO处理Oracle 数据库就相对自己进行脑部外科手术，不进行麻醉。是的，DAO的设计意图是处理桌面数据库的，而不是客户机/服务器类型的数据库。由于DAO的性能和访问局限性令人感到沮丧，因此想要使用SQL Server 和 Oracle 数据库的开发人员通常寻找其他方法。

远程数据访问对象（RDO）

微软提供的另一种选择是RDO，它最初随Visual Basic 4企业版发行。RDO对象模型与ODBC API中的结构层次非常类似。程序员会发现与ADO相比，RDO大大提高了对诸如SQL Server 和 Oracle 的客户机/服务器类型数据库系统的访问速度。虽然熟悉ODBC API 的开发人员会很快学会如何使用RDO对象模型，但是缺乏使用API经验的开发人员，比如一直使用DAO的人，则发现RDO技术很难使用。

对于大多数学习RDO的程序员而言，对象模型的本身不是问题，而从ODBC API 所继承而来的细微差别造成了最大的障碍。突然之间，程序员们不得不专心致志于临时表和书签。他们必须学习特定数据库的很多入口和出口。错误消息“The connection is busy with results from another hstmt”是否会在出错得体放给出提示？基于数据库的ODBC 连接所不能实现的操作，RDO也不会支持。相反的，指挥得到上面的错误消息。而DAO则通过自动地创建另外一个数据库连接来执行要求的操作，从而隐含了这个问题。

对于习惯于写DAO代码的程序员而言，RDO所带来的另一个挑战是RDO 不具备ADO的很多功能，例如排序，查找和过滤等。RDO所不具备的其他DAO功能包括ODBC API 函数的数据定义语言（DDL）接口，例如 Create Table 和 Create Field。

4．最好的数据访问技术：ActiveX Data Objects

这种数据访问技术，它能够把DAO的简单性和使用上的方便性与RDO的速度、能力和控制相结合。而最初作为Microsoft Internet Information Server 3 软件包一部分出现的ADO，正是希望成为对于所有人都适合的技术。当然，这种理想的目标几乎不可能实现。

虽然 ADO 最初的版本缺少很多RDO 的功能，但是ADO2.0提供了同样的功能。某些RDO 功能，如混合临时表，在ADO中还没有实现，但是这些功能都是很少使用的。事实上，很难找到一个在RDO中非常有用的功能，而在ADO中找不到。

随着2.1版本的出现，ADO及其支持库开始提供DAO所具备的几乎所有功能。在2.1版中加入了DDL库，以提供类似于DAO中诸如Create Table 、Create Field 和Create Index 等函数的功能。ADO 2.1中的Microsoft Jet 和 Replication Objects (JRO) 提供在DAO中通过DBEngine 对想实现的Jet特有的大部分功能。ADO2.1还增加了简化检索新产生的标识值的性能。ADO2.5 并没有增加使之与DAO和RDO功能更为接近的新功能，这是因为ADO 落后于DAO的唯一方面是其搜索和过滤能力。因此,ADO拥有了RDO 和DAO的绝大多数功能，同时还有很多以前的数据访问技术所没有的有益功能。

数据库的管理

传统的数据库管理存在着一些问题，许多问题是由于单独地考虑应用而引起的。较错综复杂是数据的依赖性。当程序的逻辑结构受到他的物理数据结构的束缚的时候，改变了数据结构，肯定要改变程序，引出的结果是采用传统存取方法的程序很难维护。解决以上问题的方法经常是把数据组成一体化的数据库，这样就能把控制存取所有数据任务集中在一个中心的数据库管理系统中。

使用集中化数据库如何解决数据冗余问题呢？把所有的数据都收集和存放在一个地方，因此任意给定的数据元素只有一个拷贝。当一元素的值例如地址改变时，只修正唯一的数据拷贝，要求这些数据元素的任何一个程序都能得到同样的值。

数据库又如何协助解决数据依赖性问题呢？由于存取数据的责任由数据库系统承担，因此程序员就可不考虑实际的数据结构，导致了程序更少地以来于它的数据，一般是更容易维护了。可以预料，数据库管理会继续向前发展。