

编 号

江南大学

# 本科生毕业设计（论文）

题目： 基于二维码管理的仓库管理系统

---

物联网工程 学 院 计算机科学与技术 专 业

学 号 1030413210

学生姓名 张秀芳

指导教师 钱 瑛 讲师

---

二〇一七年六月



## 设计总说明

随着科技的飞速发展，国内外企业单位逐步引入计算机网络化管理来提高工作过程中各环节的效率。仓库管理作为供应链管理的重要组成部分，其工作效率的提升对供应链管理运作质量的改善具有重要意义。但目前一些中小型企业及事业单位仍旧沿袭使用传统简单的人工记录仓储信息的管理模式，该模式在现网络飞速发展的时代已完全无法满足仓库管理的需求，大量的人工信息记录不仅低效、容易出错，还造成了巨大的人力资源浪费，且对于后续存储信息查询也无法进行提高。

本次系统是针对在送检设备校验期间，校验机构需对送检设备的存储仓库进行信息管理这一需求而设计的。校验机构对送检设备进行委托校验时需要一定的校验周期，而在检验期间对送检设备的存储管理已然成为检验机构运作管理的一个重要部分。存储仓库管理效率的提升不仅可以提高设备存储日流量，还因正确划分存储位置分类存储使得货物提取时易准确确定位置，查找方便，避免误拿设备的情况发生。

送检设备仓库存储管理虽然属于仓库存储管理，但却与普通货物存储管理有所不同。普通货品的存储只有进库和出库两次操作，而对于送检设备，在存储管理操作上需要有未检入库、送检出库、已检入库、检毕出库四次存储操作，这种存储操作方式的不同也使得送检设备仓库管理这一“两进两出”存储管理方式的系统设计对同类送检设备的存储管理有相当大的借鉴意义。

基于以上现状，现选取需定期送检设备中的安全阀为例，通过 PC 端和手机端两方面入手，对该类设备仓库存储管理系统进行详细系统设计，使单位送检安全阀存储管理信息由人工表格记录转化为信息化管理。对于安全阀入库存放、校验出库、校验完毕入库和单位取阀出库环节通过现代化仓储技术实现科学管理，使不同种类安全阀迅速入库以及快速出库变得准确可靠、灵活高效。

本次管理系统的开发平台技术和环境为：PC 端以 Eclipse 为开发平台，使用 Java 开发语言结合 JSP 以及 AJAX 技术实现 PC 前端与后台服务器开发，使用 MySQL 作为底层数据库开发工具。而在手机端，操作系统选用了现阶段手机端系统中使用广泛的 Android 系统，通过 Android Studio 为主要开发平台，使用 Java 语言以及 XML 语言结合二维码扫描技术进行手机端 APP 开发。

由于本次毕设是截取检验流程中的设备存储管理部分进行系统设计，故为了使整体流程顺利运行，在进行设备存储操作之前需要先模拟了一个委托单录入过程生成委托单。使用生成的委托单编号和安全阀自身编号进行存储操作。对于 PC 端，主要包括七个模块功能，分别是：设备信息录入生成委托单、送检安全阀入库存储、待检安全阀出库送检、已检安全阀入库存储、已检安全阀提取出库和安全阀信息浏览以及移动端预操作信息。

由于网络的全体覆盖以及智能手机的普遍性和便携性使得交接员可通过手机端运行配套 APP 实地对存储位置进行合理考察后进行安全阀入库和出库。而手机端的移动便捷性搭配上移动摄像头功能使得现阶段成熟的二维码技术与需要频繁进库存储设备的仓储系统完美结合。无需输入数据，只需利用手机进行入库操作时对相应委托单二维码以及存储位置二维码进行扫描后确认预入库。而在出库时，直接扫描要出库的委托单或单个安全

阀编号即可。此二维码技术与手机移动端的结合使用使得进行安全阀信息预入库和预出库时操作简单、方便又高效。

手机端主要有五个功能模块，分别是：送检安全阀扫描委托单信息预入库、库内未检安全阀预出库送检、已检安全阀扫描委托单二维码对安全阀检验结果确认后预入库、已检安全阀提取出库和安全阀存储信息浏览。

通过 PC 端结合手机端实现安全阀仓储有效管理，使存储管理操作简单高效且能减少因为人工信息的大量输入而出现的信息误差。同时数据库信息能选择性的通过筛选生成数据报表显示出来，这也使有关人员在浏览需要核查确认的信息时简单明了，在存档保存时方便快捷。

**关键词：**安全阀；仓储管理；Android；Java；二维码

## ABSTRACT

With the rapid development of science and technology, domestic and foreign enterprises gradually introduce computer network management to improve the efficiency of the work process. As an important part of supply chain management, warehouse management is of great significance to improve the quality of supply chain management. But some small and medium enterprises and institutions still follow the use of traditional simple manual records of storage information management model. This model is completely unable to meet the needs of warehouse management in the era of rapid development of the network. A large number of manual information records are not only inefficient and error prone, but also cause a lot of human resources to waste, and can not improve the subsequent storage information query.

This system is designed according to the inspection agency need to use information technology to manage the warehouse during the equipment inspection period. The inspection agency needs a certain inspection period when commissioning the inspection equipment, and the storage management of the inspection equipment during the inspection period has become an important part of the operation and management of the inspection agency. Storage management efficiency of the warehouse can not only improve the storage capacity of the equipment, but also because of the correct allocation of storage location classification storage makes the goods easy to determine the location of the extraction, easy to find, to avoid mistakenly take the equipment situation.

Inspection equipment Warehouse management, although belonging to a warehouse storage management, but with the general cargo storage management is different. Ordinary goods storage management only includes the preservation and extraction of two operations, and for the inspection equipment, storage management operations need to save, extract to verify, save after verification, extract four storage operations. This kind of storage operation mode also makes the system design of the "two-in-two-out" storage management mode of the inspection and dispatching equipment warehouse management has a great reference significance for the storage management of the similar inspection equipment.

Based on the above situation, now, I will take valve which need to send inspection regularly as an example. I am going to do a detailed system design for the safety valve warehouse management system which will makes the storage management information of the safety valve delivered by the unit converted from manual form record to information management through the PC side and mobile terminal two aspects. The system has four storage operations for the safety valve, and realizes scientific management through modern storage technology, so that different kinds of safety valves can be quickly stored in the warehouse as well as fast and accurate, reliable and flexible.

The management system development platform and environment is: PC side use eclipse as a development platform and use Java language combined with JSP and Ajax technology to realize PC front-end and back-end server development, using MySQL as the underlying database development tool. In the mobile phone terminal, operating system selection using Android system is widely used at present mobile phone terminal system in. Through the Android Studio as the main development platform, and use Java language and XML combined with the use of two-dimensional code scanning technology for mobile terminal APP development.

The graduation design is part of the verification process for device storage management, in order to make the whole process run smoothly, the order entry process first set before the equipment storage operation. Repeated use of the generated order number and safety valve number for storage operations. For the PC side, mainly including seven module functions, namely: equipment information entry to generate a single order, not verified safety valve storage, to be verified safety valve out of the library, has been checked safety valve storage, has been verified safety valve out of the library, the safety valve operation information browsing and mobile pre-operation information.

Due to the overall coverage of the network and the universality and portability of the smart phone, the operator can carry out the safety valve pre-storage and the pre-out of the library operation after carrying out the storage location on the mobile terminal. And the mobile side of the mobile convenience equipped with mobile camera function makes the maturity of the two-dimensional code technology and the need for frequent access to the storage warehouse storage system perfect combination. Do not need to enter the data, in the storage operation, just use the phone to the corresponding order two-dimensional code and storage location two-dimensional code to scan and then confirm the pre-library. And in the library, the direct scan to the library of the order number or a single safety valve number can be. This combination of two-dimensional code technology and mobile phone mobile terminal makes the safety valve information pre-storage and pre-library, simple, convenient and efficient.

There are five main function modules on the mobile phone side, namely: safety valve scanning order information pre-storage, the warehouse is not verified safety valve pre-library inspection, scan the ticket single-dimensional code to confirm the safety valve results and then pre-storage operation, check the safety valve to extract the library and safety valve storage operation information browsing.

Through the PC side with the mobile phone side to achieve effective management of the safety valve storage, so that the storage management becomes simple, efficient and can reduce the manual information due to a large number of input information errors. At the same time, the database information can be selectively filtered through the production of data reports displayed, which also allows the staff to check the need to verify the confirmation of the information is simple and clear, in the archive save convenient.

**Keywords:** Safety valve; Warehouse management; Android; Java; Two-dimensional code

# 目 录

第 1 章 绪论 .....	1
1.1 研究背景 .....	1
1.2 研究现状 .....	1
1.2.1 国际仓储现状 .....	1
1.2.2 国内仓储现状 .....	1
1.3 研究内容及意义 .....	2
1.3.1 研究内容 .....	2
1.3.2 研究意义 .....	2
1.4 主要开发阶段 .....	3
第 2 章 需求分析 .....	5
2.1 系统需求分析 .....	5
2.2 可行性分析 .....	5
2.2.1 技术可行性 .....	5
2.2.2 软件使用可行性 .....	5
2.2.3 单位使用可行性 .....	6
2.3 系统业务流图 .....	6
第 3 章 系统设计 .....	7
3.1 系统总体设计 .....	7
3.1.1 设计目标 .....	7
3.1.2 选用的相关技术介绍 .....	7
3.2 系统详细设计 .....	9
3.2.1 前期工作 .....	9
3.2.2 PC 端模块功能设计 .....	10
3.2.3 手机端功能模块设计 .....	12
第 4 章 系统数据库设计 .....	15
4.1 数据库需求分析 .....	15
4.2 数据库概要设计 .....	17
4.3 数据库逻辑设计 .....	17
4.4 数据库物理设计 .....	18
第 5 章 系统实现 .....	21
5.1 建立数据库连接 .....	21
5.2 PC 端模块 .....	23
5.2.1 PC 端登入 .....	23

5.2.2 委托单信息录入 .....	24
5.2.3 未检入库 .....	27
5.2.4 已检入库 .....	29
5.2.5 信息浏览 .....	31
5.2.6 出库 .....	35
5.2.7 进出库操作确认 .....	39
5.3 手机端功能模块.....	40
5.3.1 手机端与服务器通信 .....	40
5.3.2 登陆模块 .....	43
5.3.3 信息浏览 .....	44
5.3.4 安全阀入库 .....	46
5.3.5 委托单未检批量出库 .....	50
5.3.6 安全阀已检出库 .....	52
第 6 章 系统测试.....	55
6.1 ANDROID 手机连接 MACOS 系统 PC 设置环境 .....	55
6.2 功能测试.....	55
6.2.1 PC 端整体功能测试 .....	56
6.2.2 手机端整体功能测试 .....	56
第 7 章 结论与展望 .....	59
7.1 结论.....	59
7.2 不足之处及未来展望.....	59
参考文献 .....	61
致 谢 .....	63



# 第 1 章 绪论

## 1.1 研究背景

随着科技的飞速发展，计算机类科技应用的广泛普及，仓储管理在这种情况下呈现两类情况发展。分别是以物流为基础的大型物流仓储管理和中小型企业自身货物存储仓库管理。

以物流为基础的大型物流仓储发展迅速，通过高科技设备的运用以及计算机信息系统方式的管理，使得仓储管理高效，条理性强，出错率低。

中小型企业运作的仓储管理模式停留在由纯人工纸张记录管理转向先纸张记录，后通过计算机可存储大量数据这一大容量特性，把所得数据统一人工录入计算机表格，再针对计算机数据进行统计归纳。这种仓库管理方式虽对数据的保存有一个大幅度安全提升，降低数据丢失缺损情况出现的概率，但由于数据记录仍旧是人工处理，人为因素使得存储数据录入不仅速度慢且准确率低，由此仓库存储工作效率仍旧得不到显著提升。对于频繁需要进出库的仓储操作，出入库频率的剧增使得单纯的人工先记录后存储的仓库管理方式已难以满足现仓储管理所需的快且准的要求。仓库管理作为供应链管理的一个重要组成部分，其工作效率的提升对供应链管理运作质量的改善具有重要意义。工业企业的发展以及社会需求模式变化使得仓储模式更新迫在眉睫。现阶段，国内企业单位逐步引入全计算机网络化管理替代大量人工记录，减少人力资源的浪费同时大幅度提高了仓储工作过程中各环节的效率。

人工输入信息存在的手动输入延迟以及信息错误率不管用什么手段都很难提高以及百分百保证准确。因此，为尽量避免这种人工不确定因素带来的信息输入速率无法提高以及准确率无法保证问题，在此系统中，特别在手机移动端的 APP 中引入二维码扫描技术，一旦二维码生成，后续的扫描过程只需很短的时间就能准确的获取所需信息，在提高速率的同时还保证了信息输入的准确性。

本次系统设计是以校验机构对定期送检设备的仓库存储为基础，选取了送检设备中的安全阀为例进行仓库存储管理系统设计。

## 1.2 研究现状

### 1.2.1 国际仓储现状

国际仓储走在仓储科技发展的前端，其仓储技术设备先进，引入条形码扫描技术，EDI 技术以及仓储管理信息系统大幅度降低人力资源的使用，高科技设备与仓储理念使整个仓储运作流程简单高效且准确。

### 1.2.2 国内仓储现状

国内企业单位逐步引入计算机网络化管理来提高工作过程中各环节的效率。大型企业仓储开始投入重视，自国外先进的仓储技术传入我国以来，我国仓储技术已有显著的提高。但现阶段国内仓库管理出现两极状态，以物流为主的大型企业仓储技术先进，而一些中小型企业及事业单位仍旧沿袭使用以传统简单的人工记录仓储信息的管理模式，人工记录使得货物进出库滞留关口时间久，不仅低效且容易出错，而且不能够满足自上而下全程

监管的高效管理工作需求。目前安全阀的仓库管理停留在由人工纸质表单登记后用 Excel 表格录入保存数据的人工管理水平。

### 1.3 研究内容及意义

#### 1.3.1 研究内容

本次毕业课程设计是通过研究以及运用 J2EE 技术设计基于二维码的仓库管理系统，其中主要内容包括：1) PC 端设计；2) 移动端 APP 设计；3) 服务器后台数据处理；4) 底层数据库搭建。

PC 端主要是采用 JSP 技术进行界面显示开发，并结合 AJAX 技术以及 jQuery 技术调用后台 Java 逻辑代码，而后把获取到的结果重新显示到网页界面。其主要操作流程是先登入信息操作员账户，对使用单位的委托单信息网上信息录入，随后通过未检入库选择已录入信息的委托单表单的委托单号以及交接工人和存储位置实现存储。未检出库以及最终已检出库都只需要直接在出库界面点击要出库的表单安全阀实现出库。已检入库需要在选择入库表单后在右边显示的安全法信息里选择合格安全法后再进行存储信息填写后入库。

手机端主要通过 XML 设计移动端手机界面，通过 Java 语言设计移动端数据处理。对于移动端二维码技术使用，主要是通过调用关于二维码扫描的开源 jar 包 simplezxing.jar，使用相应的获取字符串函数获取二维码中的字符串信息，随后对获取的字符串进行后台处理，通过客户端与服务器后台建立的通信连接把要处理的数据传输到服务器后台进行处理，并返回相应应答信息。二维码扫描功能在安全阀进出库过程中都有涉及使用，入库时，通过扫描委托单编号以及存储地址进行预入库操作，出库时通过扫描委托单进行整单出库操作，通过扫描单个安全阀进行委托单内单个安全阀提前出库。

服务器后台主要是使用 JSP 技术进行代码编程，通过搭建好的数据库连接池与通过 Java 代码生成的 JavaBean 与底层数据库建立连接后对数据库表进行数据信息的修改或获取，随后对数据进行相应逻辑处理。

底层数据的保存使用了开源数据库 MySQL，通过数据库语言对数据库进行表的搭建，而后多次调用数据库信息对其简单操作就能实现对安全阀进出库实时信息记录。

#### 1.3.2 研究意义

传统的人工数据记录管理模式，极易出现人力资源需求量大，效率低下，记录信息准确率低等影响仓储工作整体效率的问题。而本次研究通过对安全阀仓库信息管理系统的搭建，使得安全阀仓库管理系统摆脱人工数据记录模式，减少其人力资源的消耗，显著提升安全阀进出库效率，极大的改善安全阀仓储管理中由于人工记录数据而出现的错误率高，信息不易整理归纳，历史存储信息不易查询等问题。若小型企业都重视仓储管理系统的搭建，使用计算机网络管理系统代替人工，那么将节省大量的人力资源，提升仓储工作效率，对整个供应链管理运作质量的提升也有所帮助。

本次毕业设计课题针对定期送检的安全阀校验期间的安全阀存储管理，利用计算机网络化管理替代人工记录，保证了安全阀仓储管理的准确性，降低存储管理过程中人力资源消耗的同时提升了安全阀仓储管理的条理性和运作流程的效率，使得安全阀存储过程变得简单、便捷、高效。

在本次设计中，通过研究 J2EE 技术架构、网页前端框架 Bootstrap、数据库连接池、AJAX 技术和 Android APP 开发技术，从而掌握网站搭建过程如何实现前台与后台数据交互技术和 Android 客户端与服务器端数据传递方法，熟悉一个完整系统由系统分析到系统详细设计再到系统功能实现的具体开发过程。在开发过程中，通过不断的学习和研究，本人对问题的分析解决能力有一个大幅度提升。

## 1.4 主要开发阶段

本次研究课题主要是根据安全阀校验期间需要对仓库进行系统化存储管理这一实际需求进行系统开发，故大致部分分为以下四个步骤。

第一，对安全阀仓储系统进行需求分析。根据任务书中描述的系统所需的功能进行分析，在分析过程中通过与指导老师的交流，了解安全阀存储的具体操作流程，针对安全阀“两进两出”这一存储方式的独特性结合普通仓库管理系统设计方式对这种定检类设备的系统开发进行需求分析。通过对用户需求进行调研，确定各用户的需求功能，实现定检类设备由人工记录信息管理转向仓库存储系统管理从而达到提高存储效率，简化工作人员存储操作，降低人力资源浪费的结果。

第二，针对需求分析内容进行系统设计。按照需求分析需要完成的系统具体功能选用合适的开发工具。为实现 PC 端以及手机端安全阀存储一体化要求，选定相应程序开发平台，利用开源数据库 MySQL 以及 Java 语言结合 JSP、AJAX 以 XML 技术来设计安全阀仓库管理系统，根据需求分析划分的功能块对系统进行总体模块设计，确认各模块之间的相互联系。确定手机端和 PC 端如何通过相互合作使用户对仓储系统的使用更加方便快捷以及合理管理。对系统中需要用的数据合理设计数据库，确认数据库表之间的数据联系，确定需要用到框架模型，确定系统整体界面布局 and 具体开发流程。

第三，针对系统设计中已确定的模块，数据库进行具体模块功能程序编写，编写前台界面部分以及后台处理部分，编写手机客户端 APP 程序，由于本次系统需要频繁调用数据库信息进行处理，故为了方便数据库连接，搭建 tomcat 数据库连接池，并在业务层实现所需连接的具体方法，简化了数据库频繁调用的重复繁琐的连接过程。

第四，对初步开发的系统进行系统测试。对开发完成的系统整体功能先单元后整体的模式进行调试。通过分步骤测试系统各功能模块使用，调整运行过程各种因疏忽因素以及为考虑因素对系统使用的影响并对其进行调整。整体运行一遍该系统的所有功能，测试功能之间数据是否出现因前后数据不一致而产生的问题，并对其进行调整，确保系统准确流畅的运行。



## 第 2 章 需求分析

### 2.1 系统需求分析

由于此系统是基于需定期校验的设备的特殊存储方式进行仓库存储系统开发的。故以安全阀校验存储为例，经过调研可知，目前安全阀设备校验存储管理方式还停留在人工管理阶段，当安全阀设备送检时，在前台登记送检设备信息，生成三份委托单，一份返还给使用单位，使其在安全阀校验完成后可通过委托单进行安全阀提取，一份通过工作人员保存，并将安全阀送到管理仓库进行堆存放待取，最后一份作为信息记录收纳保存。

为取代人工记录安全阀存储信息数据，降低人力资源的浪费，提高仓储管理流程效率，确保存储信息更加准确易查询，现安全阀管理系统对不同用户需要具有以下功能。

第一，仓库存储信息管理人员要求安全阀在存储过程对记录的委托单信息查找方便，最好能在第一次填写好设备信息后，把数据存储，以后的入库出库都能通过简单信息获取填写的信息。对安全阀存储时，输入委托单编号就能进行快速存储，出库时可以对出库信息进行选择，然后确认出库。并要求能浏览查询安全阀具体存储信息。

第二，仓库设备搬运交接员工要求提交预入库预出库信息时信息操作人员能实时查看到，并且在存储完后经过仓库口与操作员进行信息确认，随后操作员对手机端的信息进行确认存储，提高仓库存储信息的准确性。

第三，仓库企业管理者要求系统易操作，易安装，不需要对仓库工作人员有过多的技术要求。

第四，委托单位要求安全阀提交入库和提取出库的时候过程能尽量简单，并减少等待存储或等待提取安全阀的时间。

第五，设备搬运交接员工要求设备分类存储，能通过编号迅速找到安全阀的存储位置，并能准确核对表单内安全阀具体信息。

### 2.2 可行性分析

#### 2.2.1 技术可行性

本次基于二维码的安全阀仓库管理系统是以 Java 语言为主要开发语结合网页前端框架 Bootstrap 以及 Android 客户端的 xml 语言进行系统开发。在开发过程中为实现网页界面的部分更新，数据处理调用，界面功能实现，还运用到 AJAX，Javascript 以及 css 部分知识。手机端使用的二维码扫描技术，当前已十分成熟。而选用的 Android 系统也对后台 JSP 有较好的支持性。若要重新学习全部技术在短期内进行开发还是较有挑战的，但是由于已拥有 Java 语言以及数据库语言知识的基础，而 Bootstrap 框架也让编写前端界面工程量变小，故在老师的提点指导下，技术是可达到完成标准的。

#### 2.2.2 软件使用可行性

本次管理系统开发主要是使用 eclipse Mars.1 Release (4.5.1)、MySQL 14.14 、Android studio 2.3.1 以及 MySQL 管理工具 MySQL Workbench 等软件。系统运行的环境是 jdk1.8，服务器部署环境为 Apache tomcat 7.0。由于 MySQL 在数据库系统选择方面具有体积小，

速度快，使用简便的等优点，企业型数据库以及 web 网站搭建中一般都选用 MySQL 作为后台数据库，故数据库方面选用 MySQL 是可行的。

### 2.2.3 单位使用可行性

仓库管理系统的使用，只需要局域网内搭建服务器，且单位会分发工作用手机，故手机 APP 安装方便，连入局域网 wifi 就可实现客户端操作，对于现在手机使用广泛，无需投入成本，且节省记录信息人员开支，提高了仓储效率，对单位来说，无论是降低成本还是提升工作效率来说都是可行的。

## 2.3 系统业务流程图

本次对基于二维码的安全阀仓储管理系统的设计通过调研各用户需求，对所需功能进行分析，通过各用户对数据信息的操作实现安全阀从检验委托单信息输入到安全阀检验完毕出库的进出库完整存过程信息记录，安全阀仓库管理系统的基本系统模型如图 2-1 所示。

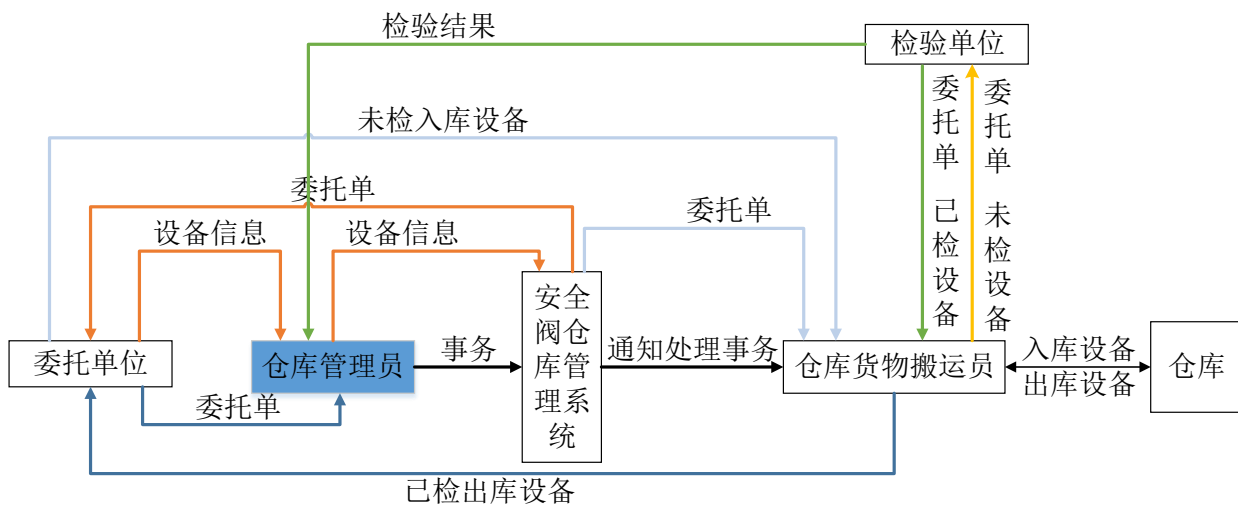


图2-1 安全阀仓库管理系统基本系统模型

## 第3章 系统设计

根据上文的需求分析，本阶段对基于二维码的仓库管理系统通过安全阀仓储管理为例进行系统设计并对其进行阐述。

### 3.1 系统总体设计

#### 3.1.1 设计目标

本次安全阀仓库管理系统使用用户有两种，一种是 PC 端安全阀信息存储记录员，另一种是 Android 移动客户端对交接处理具体安全阀存储位置，带领搬运公进行安全阀具体位置入库和出库的仓库货物搬运工。按照系统需求分析，确定安全阀仓库管理系统总体功能模块设计。安全阀仓库存储系统主要分为 PC 端和 Android 端。

PC 端可以实现安全阀委托单信息录入，安全阀未检入库，未检出库，已检入库，检毕出库以及移动端信息确认，安全阀仓储信息浏览和查询。

Android 端可以实现对 PC 端已录入的委托单进行安全阀信息扫描预入库，入库信息选择进行未检出库，扫描安全阀信息确认是否检验合格后实现已检安全阀再入库，扫描表单二维码实现安全阀出库以及对仓储管理信息浏览。

整体设计模块图如下图 3-1 所示。

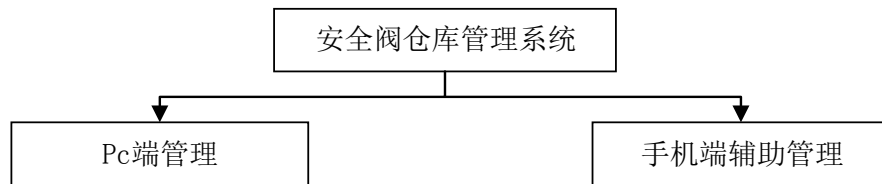


图 3-1 整体设计模块

#### 3.1.2 选用的相关技术介绍

##### 1. 系统结构的选用

对于管理系统的模式结构选用，一般有两种结构模式，一种是 C/S 结构，另一种是 B/S 模式。对于 C/S 模式，每个用户都需要在电脑上安装客户端，而 B/S 技术利用已趋于成熟的 WEB 浏览器技术，在 C/S 模式的基础上，对其进行优化，去繁变简，将不同种类的客户端统一为 WEB 浏览器<sup>[1]</sup>。本次管理系统设计时选用 B/S 结构模式对系统进行搭建，简单事件的逻辑处理在浏览器运行的网页前端实现，但核心功能逻辑方法的实现需要在服务器端由服务器运行处理。用户只需要有浏览器就可以通过网络访问服务器进行信息操作，服务器处理完信息后再返回处理结果。下图 3-2 为 B/S 结构图。

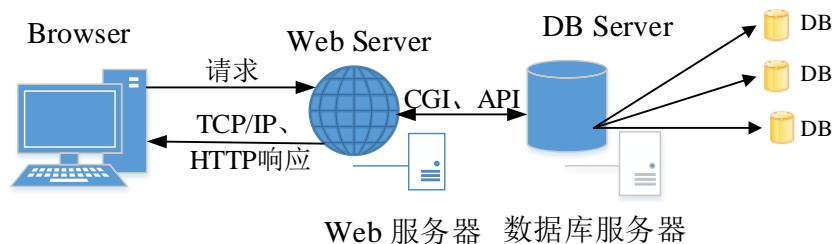


图 3-2 B/S 结构图

##### 2. 网页运行服务器的选用

本次系统设计选用的网页服务器是 Tomcat。Tomcat 是一款免费的开源的轻量级 Web

应用服务器，由于其运行占用系统资源小，扩展性好，性能稳定，是当前较流行的 Web 应用服务器。

Tomcat 是一个能发布运行 JavaWeb 应用程序的 web 服务器，也称为 Servlet 容器。它对客户端请求的响应流程为，当客户端对某一特定服务器发出请求时，Tomcat 接受请求并创建一个包含客户端请求信息的 ServletRequest 对象和一个 ServletResponse 对象。而后 Tomcat 调用客户端对部署在 Servlet 上的服务方法的请求，同时将已创建的请求信息对象 ServletRequest 和响应信息对象 ServletResponse 作为参数传入。Servlet 从传入的对象 ServletRequest 中获取客户端请求信息，对信息进行处理后，把响应结果传入对象 ServletResponse，最终由 Tomcat 把结果返回给客户端。

### 3. 底层数据库的选用

本次系统设计选用的底层数据库是 MySQL。MySQL 是一个在 WEB 应用方面表现最好的关系型数据库管理系统的应用软件。它由瑞典公司 MySQL aAB 开发，它不是把所有的数据保存在一个大型仓库内，而是把信息数据保存在不同表中，通过这种方法来增加数据的读取速度，提高数据库的灵活性<sup>[2]</sup>。

MySQL 访问数据库所使用的 SQL 语言是最常用的标准化数据库语言。由于 MySQL 的体积容量占用小，运行速度快，总体拥有成本低，对于本次毕业设计来说，MySQL 满足本次系统数据的存储需求。而由于 MySQL 是开源软件，故节省了很大一部分的开发成本。

### 4. JSP 动态网页技术介绍

JSP(Java Server Pages, java 服务器页面)是一个简化的 Servlet 设计。JSP 技术是在传统静态网页 HTML 文件中插入 Java 程序和 JSP 标记后形成的一种后缀名为(\*.jsp)的 JSP 文件。JSP 开发的 Web 应用是跨平台的能在多种操作系统上运行的应用。

JSP 以<%,%>形式实现了 HTML 语法中的 java 扩展。与 Servlet 一样，JSP 同样是在服务器端运行的，当浏览器向服务器发出某个 JSP 页面请求时，Web 服务器在服务器端对 JSP 文件进行编译处理，运行 JSP 页面中的 Java 脚本，把生成的结果嵌入 HTML 文本生成 HTML 页面并发送回客户端，故客户端浏览网站只需要有浏览器就可以了<sup>[3]</sup>。

JSP 语言特点：

- 1) 一次编写，多处运行，不受客户端系统环境影响；
- 2) 系统支持多平台，可以在任意操作系统以及任意环境中进行系统环境部署并运行；
- 3) 后台功能强大，由于 JSP 的后台是 Java 语言和 Servlet，故 JSP 可以进行复杂的逻辑业务处理

### 5. AJAX 技术选用

由于在动态网页中，很多时候需要对数据进行后台处理返回结果，达到局部刷新网页的效果。此时 AJAX 技术选用便能圆满的解决这个问题。

AJAX 全称为 Asynchronous Javascript And XML，其中文名为异步 Javascript 和 XML，是一种用于创建交互式动态网页应用的 Web 开发技术。AJAX 通过 JavaScript 与服务器进行少量的数据交换，在不重新加载页面的情况下使局部页面实现异步更新<sup>[4]</sup>。



AJAX 是一种独立于 Web 服务器的浏览器技术,使用 AJAX 技术不仅可以使因特网应用程序更小更快,交互性更强,更友好,提高系统性能,还可以优化用户界面. AJAX 工作原理如下图 3-3 所示.

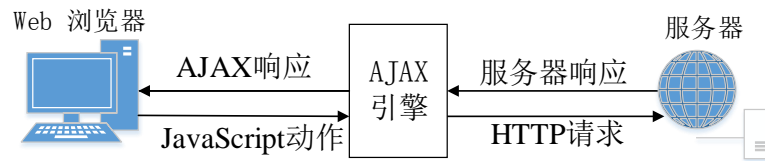


图 3-3 AJAX 工作原理

## 6. 手机端 Android 系统的选用

就目前手机移动端系统主要分为 Android 和 ios 系统,而在这两者中,Android 系统在市场上有一个绝佳的占有率,且由于 Android 编写环境提供方便的调试功能以及虚拟设备模拟器<sup>[5]</sup>,并且在与服务器连接对数据进行处理方面,Android 对本次服务器后台选用的 JSP 技术有较好的支持性,故此次手机端系统选用 Android 进行开发.

## 3.2 系统详细设计

现对概要设计的模块具体分析,对模块功能细节和实现方法进行详细设计.

### 3.2.1 前期工作

由于系统功能的设计是基于网络通信的基础上对数据库数据进行操作,故在功能设计之前,需要先建立 PC 端的数据库连接、手机端 APP 与服务器数据通信以及手机二维码扫描方法.

#### 1. PC 端与后台的数据库连接

对于 PC 端,由于服务器是直接部署在 PC 端的,故 PC 端主要需要建立与后台数据库连接.由于本系统需要频繁的对数据库进行连接,故为了提高对数据库的操作性能,放弃传统数据库连接方式,改为效率更高的数据库连接池进行数据库连接.

#### 2. 手机端与服务器数据通信

而对于手机端,因为其功能的实现主要是通过与服务服务器连接后,通过服务器访问后台数据库,故首先需要对 APP 与服务器端建立通信连接,一般网络通信传递消息有两种 HTTP 方法,分别是 POST 和 GET 方法<sup>[6]</sup>.故通过引用这两种方法构建通信类,实现相应通信方法,简化 APP 与服务器之间的信息.

关于手机端 APP 编程要点的几点说明如下.

第一,由于 Android 端与服务器通信取决于网络问题,易造成阻塞,故有关于网络连接通信的 Android 端代码需要新开一个线程进行处理.

第二,关于用户界面组件的信息显示,为保证信息的前后一致性,需要保证在同一线程中完成,而主线程是专门运行信息队列的,故新线程通过网络通信获得的数据信息若要更新到 Android UI 界面,则需要通过 handler 异步信息传输机制把数据显示部分代码放到主线程来运行.从而达到传递消息和 runnable 队像到主线程的消息队列中,消息执行完毕后即从消息队列退出的作用.

第三，关于请求参数的格式，是通过 Map 参数传入通信方法进行参数传递的，故所有要传递的数据都要通过 Map 对象进行传递。

Map 参数赋值示例如下图 3-4 所示。

```
Map<String,String> addmessage=new HashMap<>();
addmessage.put("valorgroupnumber",valnumber);
addmessage.put("storagelocationnum",storagenumber);
addmessage.put("opaction",opaction);
addmessage.put("manindex",Account);
addmessage.put("checkedinfo",selectcheck.toString());
```

图 3-4 Map 参数赋值

### 3. 手机端二维码信息扫描

手机端建立通信后，由于多个操作需要设计二维码扫描，此处引入外部 simplezxing.jar 包，通过对包内方法的调用实现对相应二维码进行扫描<sup>[7]</sup>，随后对获取的二维码信息进行相关处理。具体实现内容详见第五章系统实现。

#### 3.2.2 PC 端模块功能设计

PC 端用户进行登入后可对仓储信息进行管理，PC 端应包含如下功能：委托单信息录入、安全阀进出库操作、安全阀仓储信息浏览和查询、对移动端已操作的预入库和预出库实现确认等功能。

其功能模块图如下图 3-5 所示。

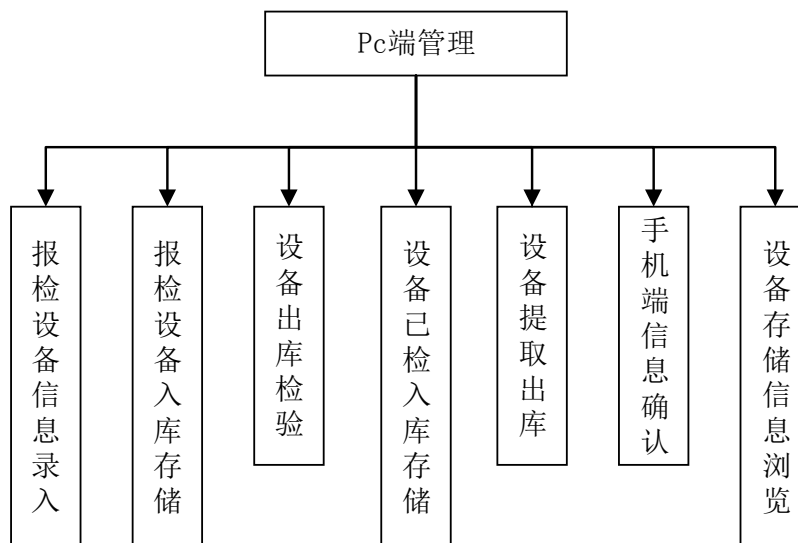


图 3-5 PC 端功能模块图

#### 1. 设备信息录入

设备信息录入分为两种类型，一种是单个安全阀报检时信息录入，另一种是多个安全阀报检时信息录入。报检设备信息录入流程图如下图 3-6 所示。

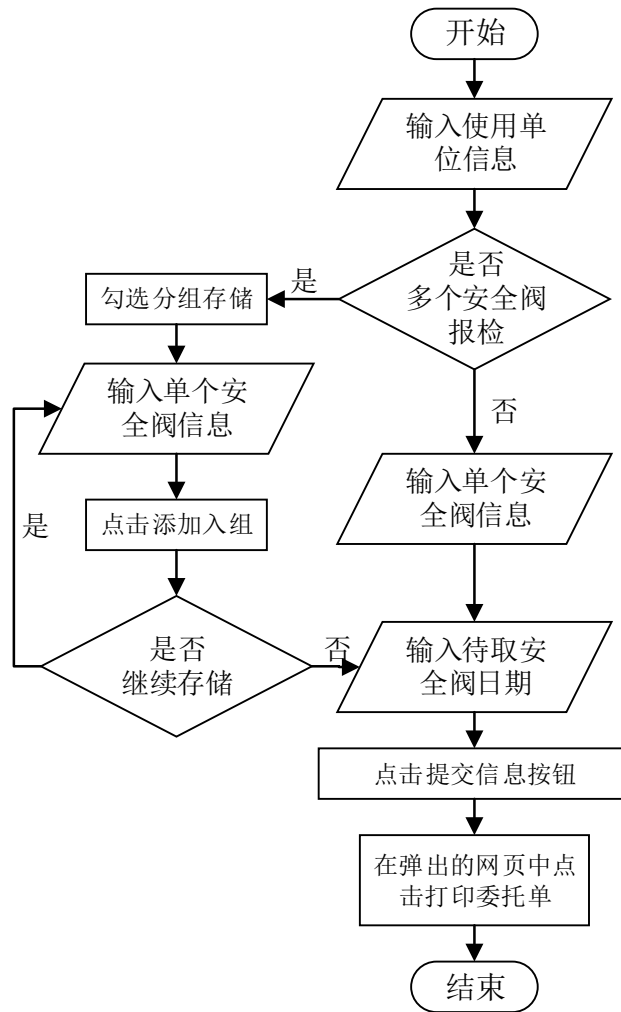


图 3-6 报检设备信息录入流程图

#### 1) 单个安全阀报检信息录入

进入信息录入模块后，直接填写委托单内包含的信息，点击提交信息后会生成委托单，实现单个安全阀委托单信息录入过程。

#### 2) 多个安全阀报检信息录入

进入委托单信息录入模块后，在安全阀信息栏勾选分组存储，每填完一个安全阀信息后点击安全阀信息填写模块右上方按钮对安全阀单个信息进行提交，界面会对提交成功的安全阀编号进行显示，最终再点击提交信息对整体信息进行委托单信息存储添加。

### 2. 安全阀进出库操作

安全阀存储进出库包括四个部分：未检入库、已检入库、出库、移动端确认。

#### 1) 未检入库

未检入库是对已经填好委托单的安全阀进行入库存储的操作。在委托单 ID 栏输入委托单编号，信息显示栏会自动显示委托单内包含的安全阀信息。填好具体存储事项后确认存储，后台对填入数据进行处理，添加存储信息，修改存储位置的状态后返回结果。

#### 2) 已检入库

已检入库是从已检待入库表内的委托单编号填入后，在信息栏显示安全阀信息，对安全阀进行合格选择，点击确认后根据合格或不合格安全阀单个还是分组情况分配两个合适存储位置，确认存储后进行已检入库，已检入库运行流程如下图 3-7 所示。

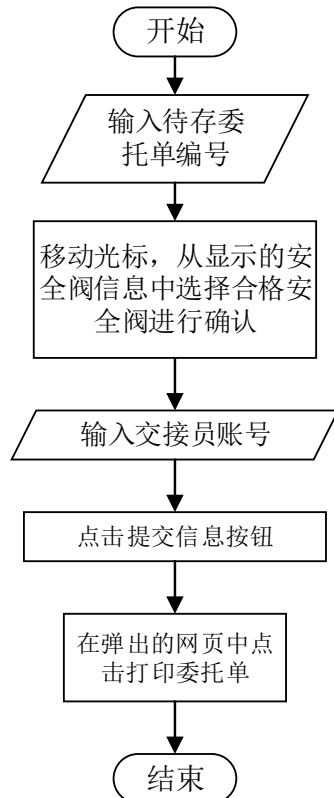


图 3-7 已检入库运行流程

### 3) 出库

出库直接在出库模块点击相应表单信息实现，后台根据点击的信息对其状态进行判断，确认此次出库是未检出库还是检毕出库，随后做出安全阀具体出库处理，即修改存储位置状态，添加出库动态信息等。对于未检出库，需要把出库安全阀的委托单编号添加到待入库表中。而由于委托检验的单位可能会出现对委托单内单个安全阀进行提前出库应急使用，故对于检毕出库可实现单个安全阀先出库问题，此时需要把检毕已出库的安全阀标志为失效状态。

### 4) 移动端确认

PC 端对 Android 端进行的预入库预出库信息进行确认，后台对信息进行判断后对存储位置状态进行改变，进行相应状态信息存储。

## 3.设备操作信息浏览

对于操作信息，通过表格信息显示操作的时间、委托单号、操作动作、安全阀状态等信息，并可通过输入检索内容过滤信息，以及把所需信息转换为 excel 表格文件形式进行下载保存。

### 3.2.3 手机端功能模块设计

手机端模块分为搬运工登入、信息扫描预入库、未检预出库、已检预出库、安全阀仓储信息浏览这五个模块。其功能模块图如下图 3-8 所示。

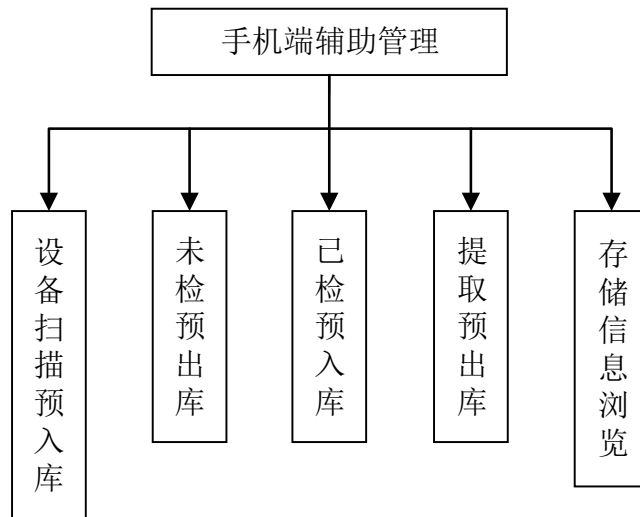


图 3-8 手机端功能模块图

#### 1. 安全阀信息扫描预入库

1) 未检预入库：对安全阀委托单编号进行二维码扫描，自动显示委托单内包含的安全阀信息，并对存储位置进行二维码扫描，把所得信息进行处理，存储在预存储表中。

2) 已检预入库：对安全阀委托单编号进行二维码扫描，自动显示委托单内的安全阀以及可勾选选项，选中即代表该安全阀校验合格，随后扫描合格存储区域编号以及不合格存储区域编号进行安全阀已检入库信息预存储。

#### 2. 未检预出库

系统会对可以预出库的信息进行显示，通过勾选信息前的选项点击确认进行预出库。

#### 3. 已检预出库

直接扫描委托单编号可实现整个表单出库，直接扫描安全阀编号，若安全阀为单一存储则表单出库，若安全阀为多个安全阀入库，则对单一安全阀实现出库，并不影响后续安全阀委托单出库操作或后续安全阀单一出库操作。

#### 4. 仓储记录信息浏览

可以在此浏览安全阀进出库有关操作信息，如操作时间、操作委托单编号，安全阀状态、进出库情况等信息。



## 第4章 系统数据库设计

本次基于二维码的仓库管理系统的数据库设计通过对不同用户的需求进行需求调研，随后通过不同用户需求对仓库管理系统的数据库进行系统需求分析，通过数据流图描述整个系统的功能结构，进一步对其进行系统概要结构设计<sup>[7]</sup>。最后对该仓库系统的数据库进行逻辑设计和物理设计，形成一个完整的数据库结构模型，通过 MySQL 将数据库物理设计转换为数据库表格。

### 4.1 数据库需求分析

通过对系统需求分析内容，对整个系统中数据的处理流程有一个明确方向。委托单录入后，信息操作员对可操作安全阀进行入库或出库等操作处理，并且可以随时查看安全阀进出库情况。安全阀仓库管理系统的功能级数据流图如图4-1所示。

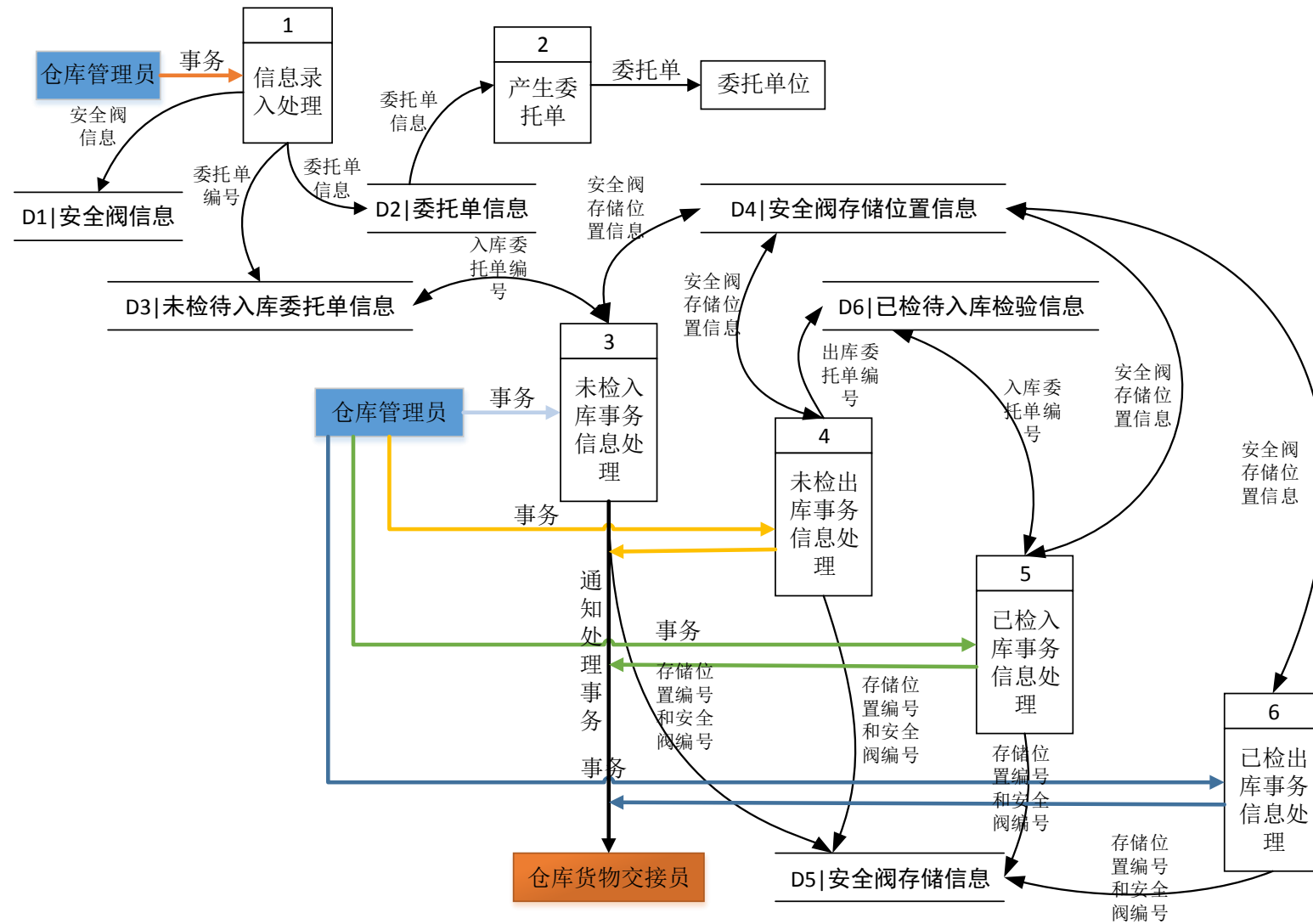


图4-1 安全阀仓库管理系统的功能级数据流图



## 4.2 数据库概要设计

现根据现具体安全阀仓库管理系统的需求分析，设计概念模型，其实体包括安全阀、使用单位、被委托单位、记录员、仓库区域，最终作出如下图 4-2 所示 E-R 模型。

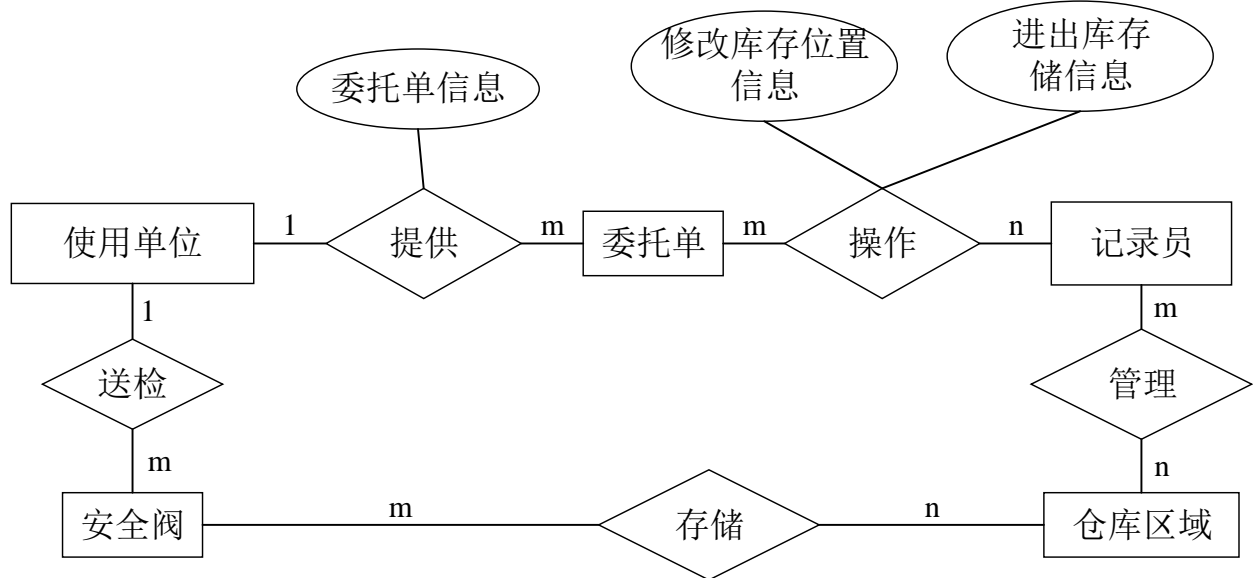


图4-2 仓库管理系统E-R图

## 4.3 数据库逻辑设计

根据本次安全阀仓库管理系统的具体情况以及上文设计的 E-R 图，设计数据库关系模式如下。

1. 安全阀（安全阀出厂编号，安全阀制造单位编号，安全阀系统编制编号，型号，工作介质，工称压力，工称通径，阀座口径，要求整定压力，压力级别范围，出厂日期，背压修正系数，制造单位许可证编号，设计压力，设计温度，阀门位号，回座压力，是否进口阀，是否丝口阀，组编号，是否检验合格，委托单编号，是否有效）；
2. 使用单位（使用单位编号，使用单位名称，使用单位地址，使用单位邮编，联系人姓名，联系人电话）；
3. 信息记录员（账号，密码，姓名，联系电话）；
4. 货物搬运员（账号，姓名，联系电话，密码）；
5. 存储位置（存储位置编号，存储位置状态，标记，存储安全阀编号或组编号）；
6. 委托单（委托单编号，安全阀编号或组编号，使用单位编号，使用设备编号，外观检查，送检时间，检验标准，检验结果编号，要求取件时间）；
7. 存储信息（安全阀或组编号，操作时间，存储位置模式，存储位置编号，存储动作，交接员账号，记录员账号，安全阀当前状态，附加存储位置）；
8. 预备存储信息（安全阀或组编号，存储位置模式，存储位置编号，存储动作，交接员账号，操作时间，安全阀当前状态，附加存储位置）；
9. 未检待存（未检待存委托单编号）；
10. 已检待存（已检待存委托单编号，存储位置模式）。

## 4.4 数据库物理设计

根据仓库管理系统逻辑模型确定需要建立的表有：安全阀基本信息表、使用单位信息表、信息记录员表，货物搬运员表、委托单信息表、存储位置信息表、安全阀存储信息操作表、未检待存委托单表、已检待存委托单表，具体内容如下表 4-1 至 4-10 所示。

1. 安全阀基本信息表 val\_information 如下表 4-1 所示。

表 4-1 安全阀基本信息表 val\_information

字段名	字段类型	字段说明	非空	数据类型	备注
productno	char(20)	安全阀出厂编号	√	String	主键
manufature	char(40)	制造单位编号	√	String	主键
valnumber	varchar(10)	系统编制编号	√	String	
valvecate	char(20)	安全阀型号	√	String	
media	varchar(8)	工作介质		String	
diapress	float	工称压力		float	
diameter	float	工称通径		float	
valdiameter	float	阀座口径		float	
requiredpress	float	要求整定压力		float	
pressgrade	char(20)	压力级别范围		String	
outputime	date	出厂日期		date	
revise	char(15)	背压修正系数		String	
manucode	char(20)	制造单位许可证编号		String	
designpress	float	设计压力		float	
designtemper	float(6,2)	设计温度		float	
valvepno	int(11)	阀门位号		int	
reseatpress	float	回座压力		float	
inportvalve	char(5)	是否进口阀		String	
svalve	char(5)	是否丝口阀		String	
groupnum	char(8)	组编号		String	
isqualify	char(5)	是否检验合格		String	
acceptno	char(20)	委托单编号		String	
isvalid	char(5)	是否有效		String	若安全阀已检出库, 则失效

2. 货物搬运员 workman 如下表 4-2 所示。

表 4-2 货物搬运员 workman

字段名	字段类型	字段说明	非空	数据类型	备注
manindex	char(12)	账号	√	String	主键
manname	char(8)	密码	√	String	
mantelephone	varchar(12)	姓名	√	String	
manpassword	char(8)	联系电话	√	String	

3. 已检待存委托单 checkedwillbesaved 如下表 4-3 所示.

表 4-3 已检待存委托单 checkedwillbesaved

字段名	字段类型	字段说明	非空	数据类型	备注
acceptno	char(20)	已检待存委托单编号	√	String	主键
valvolume	char(2)	存储位置模式	√	String	

4. 未检待存委托单 willbesaved 如下表 4-4 所示.

表 4-4 未检待存委托单 willbesaved

字段名	字段类型	字段说明	非空	数据类型	备注
acceptno	varchar(10)	未检待存委托单编号	√	String	主键

5. 委托单信息 checkorder 如下表 4-5 所示.

表 4-5 委托单信息 checkorder

字段名	字段类型	字段说明	非空	数据类型	备注
acceptno	char(20)	委托单编号	√	String	主键
valnumber	varchar(10)	安全阀编号或组编号	√	String	
factoryindex	char(20)	使用单位编号	√	String	
equipindex	char(10)	使用设备编号		String	
appearance	varchar(100)	外观检查		String	
sendtime	datetime	送检时间	√	date	
standard	varchar(20)	检验标准		String	
reportno	char(10)	检验结果编号		String	
requireddrawtime	date	要求取件时间	√	String	

6. 信息记录员 info\_op\_man 如下表 4-6 所示.

表 4-6 信息记录员 info\_op\_man

字段名	字段类型	字段说明	非空	数据类型	备注
useraccount	char(20)	账号	√	String	主键
userpassword	char(20)	密码	√	String	
username	char(30)	姓名	√	String	
usertelephone	varchar(11)	联系电话	√	String	

7. 存储位置信息 locationinfo 如下表 4-7 所示.

表 4-7 存储位置信息 locationinfo

字段名	字段类型	字段说明	非空	数据类型	备注
storagelocationnum	char(12)	存储位置编号	√	String	主键
locationstatus	char(8)	存储位置状态	√	String	
mark	varchar(12)	标记	√	String	
valgrouppnumber	char(8)	存储安全阀编号或组编号	√	String	

## 8. 使用单位信息 userfactory 如下表 4-8 所示.

表 4-8 使用单位信息 userfactory

字段名	字段类型	字段说明	非空	数据类型	备注
factoryindex	char(20)	使用单位编号	√	String	主键
factory	char(20)	使用单位名称	√	String	
address	char(30)	使用单位地址		String	
postcode	char(8)	使用单位邮编	√	String	
contact	char(8)	联系人名	√	String	
telephone	varchar(11)	联系人电话	√	String	

## 9. 预存储操作信息表 preparetochangeinfo 如下表 4-9 所示.

表 4-9 委托单信息 preparetochangeinfo

字段名	字段类型	字段说明	非空	数据类型	备注
valnumber	varchar(10)	安全阀编号或组编号	√	String	主键
optime	datetime	操作时间	√	String	
valvolume	char(2)	存储位置模式	√	String	
storagelocationnum	char(8)	存储位置编号	√	String	
opaction	char(1)	存储动作	√		
manindex	char(12)	交接员账号	√		
useraccount	char(10)	记录员账号	√		
valstatus	char(2)	安全阀当前状态	√		
exlocationnum	char(8)	附加存储位置	√	String	

## 10. 安全阀存储操作信息 valsavestatusinfo 如下表 4-10 所示.

表 4-10 安全阀存储操作信息 valsavestatusinfo

字段名	字段类型	字段说明	非空	数据类型	备注
valnumber	varchar(10)	安全阀编号或组编号	√	String	主键
optime	datetime	操作时间	√	date	主键
valvolume	char(2)	存储位置模式	√	String	
storagelocationnum	char(8)	存储位置编号	√	String	
opaction	char(1)	存储动作	√	String	
manindex	char(12)	交接员账号	√	String	
valstatus	char(2)	安全阀当前状态	√	String	
exlocationnum	char(8)	附加存储位置	√	String	

## 第 5 章 系统实现

根据前期系统功能分析，现针对上述系统设计中提到的功能分电脑端和手机端两大部分对其具体结构进行详细介绍。

### 5.1 建立数据库连接

在具体功能实现之前，需要建立数据库与网页服务器的连接。根据系统详细设计中的设计，实现服务器与后台数据库连接<sup>[10]</sup>。

#### 1. 数据库连接池建立连接

- 1) 导入数据库连接需要用到的(\*.jar)文件；
- 2) 修改工程路径 WebContent/META-INF/下的 context.xml 文件，如下图 5-1 所示。

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <Context>
3      <Resource
4          name="jdbc/mydb"
5          auth="Container"
6          type="javax.sql.DataSource"
7          driverClassName="com.mysql.jdbc.Driver"
8          url="jdbc:mysql://localhost:3306/db_valmanage?"
9          useUnicode=true&characterEncoding=utf8&useSSL=true"
10         username="xfzhang"
11         password="xfzhang"
12         maxActive="100"
13         maxIdle="30"
14         maxWait="10000" />
15 </Context>

```

图 5-1 context.xml tomcat 连接池数据初始化

注意，此时数据库建立的 database 名为：db\_valmanage。

- 3) 修改路径 WebContent/WEB-INF/下的文件 web.xml 中的 resource-ref，如下图 5-2。

```

<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://java.sun.com/xml/ns/javaee"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
        http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
    id="WebApp_ID" version="3.0">
    <display-name>web</display-name>
    <welcome-file-list>
        <welcome-file>index.jsp</welcome-file>
        <welcome-file>default.html</welcome-file>
        <welcome-file>default.htm</welcome-file>
        <welcome-file>default.jsp</welcome-file>
    </welcome-file-list>
    <resource-ref>
        <description>my DB Connection</description>
        <res-ref-name>jdbc/mydb</res-ref-name>
        <res-type>javax.sql.DataSource</res-type>
        <res-auth>Container</res-auth>
    </resource-ref>
</web-app>

```

图 5-2 对要连接的数据库命名

2. 对数据库连接建立连接类

为了使服务器在操作安全阀进出库信息过程中方便对数据库数据进行访问处理，本人建立了对数据库访问的连接类，通过把访问数据库所需的步骤方法封装在类中，而后访问时只需要一句话就能建立数据库连接，对数据库实现增删改查，该连接类的具体方法如下。

1) 创建文件 DBConnection.java，关键代码如下图 5-3 所示。

```
public class DBConnection {
    public static Connection getConnection() {
        Connection con = null;    //创建用于连接数据库的Connection对象
        DataSource ds = null;
        try {
            Context initContext = new InitialContext();
            Context envContext = (Context)initContext.lookup("java:/comp/env");
            ds = (DataSource)envContext.lookup("jdbc/mydb");
            System.out.println(ds.getConnection());
        } catch (Exception e1) {
            System.out.println("加载数据库驱动失败"+e1);
            return null;
        } // 加载MySQL数据驱动
        try {
            con = ds.getConnection();// 创建数据连接
            System.out.println("数据库连接成功了");
            return con;    //返回所建立的数据库连接
        } catch (Exception e) {
            System.out.println("数据库连接失败" + e.getMessage());
            return null;
        }
    }
}
```

图 5-3 创建数据库连接池的类

2) 实现连接方法类 connection

该类主要实现数据库的信息查询和信息修改两个方法，信息修改方法包括信息插入，修改，删除语句的实现。

数据库的查询方法语句执行方法 query(查询语句)的实现如下图 5-4 所示。

```
public static ResultSet query(String sql ) {
    System.out.println("函数DBQuery日志");
    if(conn == null){
        //同样先要获取连接，即连接到数据库
        conn = DBConnection.getConnection();
        if(conn == null){
            System.out.println("数据库连接失败" );
            return null;
        }
    }
    System.out.println("查询函数中连接到数据库数据成功"+conn);
    try {
        //创建用于执行静态sql语句的Statement对象，st属局部变量
        st = (Statement) conn.createStatement();
        //执行sql查询语句，返回查询数据的结果集
        rs = (ResultSet) st.executeQuery(sql);
        return rs;
    } catch (SQLException e) {
        System.out.println("数据库中查数据失败");
        return null;
    }
}
```

图 5-4 数据库查询方法 query 的实现

数据库信息修改方法 addquery(添加 / 修改 / 删除语句)的实现如下图 5-5 所示。

```

public static int addquery(String sql ) {
    System.out.println("函数DBQueryB");
    if(conn == null){
        //同样先要获取连接，即连接到数据库
        conn = DBConnection.getConnection();
        if(conn == null){
            System.out.println("数据库连接失败");
            return 0;
        }
    }
    System.out.println("插入函数中连接到数据库数据成功"+conn);
    try {
        //创建用于执行静态sql语句的Statement对象，st是局部变量
        st = (Statement) conn.createStatement();
        //执行sql查询语句，返回查询数据的结果集
        flag = (int) st.executeUpdate(sql);
        return flag;
    } catch (SQLException e) {
        System.out.println("数据库中插入数据失败");
        return 0;
    }
}

```

图 5-5 addquery 方法的实现

实现后，网页服务器连接数据库可以直接使用 <jsp:useBean id="connect" class="com.xfzhang.bean.connection"/>进行数据库连接，然后调用的时候直接在(\*.jsp)文件的 java 代码段使用 ResultSet rs=connect.query(sql)或 int flag=connect.addquery(sql)对方法进行调用。

## 5.2 PC 端模块

### 5.2.1 PC 端登入

index.jsp 网页界面使用表单传递登入数据，登入界面如下图 5-6 所示。

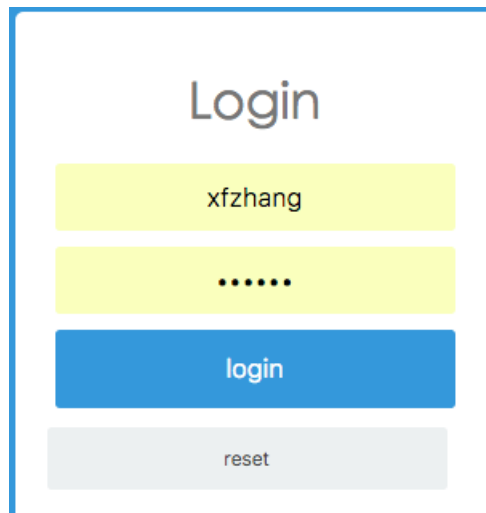


图 5-6 PC 端登入界面

通过 ajax 方式把登入信息传入后台 login.jsp 文件内，若判断登入成功，则把登入信息传入 session，使得在整个用户运行过程 session 都存在，在后续用户使用该系统时，右上方总会显示用户登入账号信息。后台登入代码如下图 5-7 所示。

```

<jsp:useBean id="connect" class="com.xfzhang.bean.connection"></jsp:useBean>
<%
    request.setCharacterEncoding("UTF-8");
    ResultSet rs=null;
    String account=request.getParameter("useraccount");
    String passwd=request.getParameter("password");
    String lo="select * from info_op_man where useraccount='"+account+"'";
    rs=connect.query(lo);
    String ss="";
    out.println(lo);
    while(rs.next()) {
        String pd=rs.getString("userpassword");
        if(rs.getString("userpassword").equals(passwd)){
            String useraccount=rs.getString("useraccount");
            String username=rs.getString("username");
            String usertelephone=rs.getString("usertelephone");
            String usepassword=rs.getString("userpassword");
            session.setAttribute("useraccount",useraccount);
            session.setAttribute("username",username);
            session.setAttribute("usertelephone",usertelephone);
            ss="sucess";
            System.out.println(account+passwd+ss);
        }else{
            ss="failed";
            System.out.println(account+passwd+ss);
        }
    }
    PrintWriter pw=response.getWriter();
    response.setContentType("text");
    pw.write(ss);
    pw.close();
%>

```

图 5-7 登录后台账户密码核对代码

当用户点击账号信息的下拉菜单退出登入时，系统会清除 session 信息，随后退出当前界面，重新回到登入界面。登出代码如下图 5-8 所示。

```

<body>
<%
    session.removeAttribute("useraccount");
    session.removeAttribute("username");
    session.removeAttribute("usertelephone");
    response.sendRedirect("../index.jsp");
%>
</body>

```

图 5-8 退出时清除 session 代码

### 5.2.2 委托单信息录入

委托单信息录入主要是对提交信息按钮的具体响应过程，通过判断是否点击了多个安全阀分组存储确认接下来具体步骤，而是否点击分组选项通过参数 flag 传入后台参数 ischecked 中，其信息值为 yes 或 no。如果为 no，需要对当前安全阀信息进行存入数据库处理。由于后面多个模块功能都运用了 ajax 技术，此处委托单信息录入模块中，对 ajax 部分的代码实现以及相应代码作用进行详细解释，具体内容如下图 5-9 所示。



```

function doFind(){
    //获取id checkisgroup的勾选情况
    var checkArray = document.getElementById("checkisgroup");
    if(checkArray.checked){
        flag="yes";
    }else{
        flag="no";
    }
    alert(flag);
    $.ajax({
        cache: false,
        type: "POST",
        //把表单数据发送到groupaddinfo.jsp, 附带参数flag的值
        url:"jsp/groupaddinfo.jsp?option=addinformation&&flag="+flag,
        //传入id为addinfomation的表单内的参数值
        data:$('#addinformation').serialize(),
        async: false,
        error: function(request) { //ajax错误时执行报错
            alert("发送请求失败! ");
        },
        success: function(data) {
            //ajax正确返回数据时对数据进行处理
            if(data=="插入失败"){
                alert(data); //将返回的结果显示到ajaxDiv中
            }else{
                var name=document.getElementById("factory").value;
                name=encodeURIComponent(name);
                var a=data.split("&");
                var path=a[0];
                var acceptno=a[1];
                //打开打印委托单的页面CheckOrderPDF.html
                window.open("/valmanage/CheckOrderPDF.html?name="+name
                    +"&path="+path+"&acceptno="+acceptno,"", "modal=yes,width=500,"
                    +"height=500,resizable=no,scrollbars=no");
            }
        }
    });
}

```

图 5-9 委托单信息提交

### 1. 单个安全阀信息录入

单个安全阀信息录入界面操作时，直接填写委托单信息，点击提交按钮即可实现信息提交，后台会对提交的委托单信息进行添加入数据库委托单表，并把当前委托单编号添加入待存表内。委托单单个信息录入界面入下图 5-10 所示。

图 5-10 PC 端委托单录入界面

## 2. 多个安全阀信息录入

多个安全阀信息录入相比于单个安全阀信息录入，只是在点击分组选项时，通过 ajax 显示隐藏代码，其功能是添加安全阀信息入数据安全阀信息表并把确认其入组。需要先对安全阀进行信息添加后通过提交信息按钮统一建立委托单。隐藏代码显示的界面部分如下图 5-11 所示。

图 5-11 多个安全阀录入部分安全阀信息添加界面

现对包含多个安全阀的委托单信息录入流程中各步骤实现代码进行分析。

### 1) 添加安全阀入组按钮的 ajax 技术代码

此处用 ajax 技术把所需信息作为参数传入后台服务器，经服务器处理后把信息传入前台，显示在前台界面上，具体代码 ajax 实现大致相似，不再附具体代码。

2) 后台添加安全阀数据代码区分委托单是单个还是多个安全阀主要通过传入后台的 checkbox 状态编制参数 ischecked 区分。当 ischecked 为 yes 时，进入多个安全阀存储模式，即把此时多个安全阀存储系统自动分配的组好进行委托单内与委托单号进行一一对应添加委托单信息。而当 ischecked 为 no 时，需要先添加安全阀信息进入安全阀信息表，随后添加委托单信息。

5.2.3 未检入库

未检入库是对已成功填写好委托单后的安全阀委托单信息进行整体入库过程，实现界面如下图 5-12 所示。



图 5-12 PC 端未检入库界面

当委托单编号输入后，鼠标焦点移除输入框后，会自动在右边安全阀基本信息框显示委托单内的安全阀信息，确保委托单输入无误。

1) 自动获取提示委托单号

安全阀委托单号栏会直接提示可填写的委托单号，节省录入时间且很大程度的直接避免了由于人工输入数据带来的认为错误。该部分实现代码如下图 5-13 所示。

```
<label for="valorgrouppnumber">委托单编号ID: </label>
<input type="text" id="valorgrouppnumber" name="valorgrouppnumber"
placeholder="acceptno" list="idlist" onBlur="showmessage()">
<%
String sql="select * from willbesaved";
ResultSet rs=null;
rs=connect.query(sql);
%>
<datalist id="idlist">
<%while(rs.next()){ %>
<option value="<%=rs.getString("acceptno")%>"></option>
<%=rs.getString("acceptno")%>|
<%}%>
</datalist>
```

图 5-13 PC 端输入框自动提示可操作委托单号

2) 显示委托单安全阀具体信息

当委托单框失去焦点时，自动执行函数 showmessage()获取数据库中该委托单所含的安全阀具体信息，进一步确定信息的准确性。该部分前端 ajax 技术实现代码如下图 5-14 所示。

```
function showmessage(){
$.ajax({
    cache: false,
    type: "POST",
    url: "jsp/show.jsp?option=showvalongroupinfo",
    data: $('#nochecksave').serialize(),
    async: false,
    error: function(request) {
        alert("发送请求失败！");
    },
    success: function(data) {
        document.getElementById("showmessage").innerHTML=data;
    }
});
}
```

图 5-14 失去焦点时自动显示信息

通过获取后台得到的数据,并插入到 `showmessage` 部分显示委托单内的安全阀信息.后台调用数据库信息并返回安全阀信息代码如下图 5-15 所示.

```
String select_val="select * from val_information where valnumber='"+valorgroupnumber+"'";
ResultSet rs_select_val=connect.query(select_val);
String select_group="select * from val_information where groupnum='"+valorgroupnumber+"'";
ResultSet rs_select_group=connect.query(select_group);
String ss="";
while(rs_select_val.next()){
    ss+="安全阀编制ID: "+rs_select_val.getString("valnumber")+"<br>";
    ss+="&nbsp;&nbsp;&nbsp;安全阀出厂编号: "+rs_select_val.getString("productno")+"<br>";
    ss+="&nbsp;&nbsp;&nbsp;安全阀制造单位: "+rs_select_val.getString("manufacture")+"<br>";
    ss+="&nbsp;&nbsp;&nbsp;安全阀型号: "+rs_select_val.getString("valvecate")+"<br>";
}
if(rs_select_group.next()){
    int count=0;
    ss+="安全阀组('"+valorgroupnumber+"'):<br>";
    count++;
    ss+=count+". 安全阀编制ID: "+rs_select_group.getString("valnumber")+"<br>";
    ss+="&nbsp;&nbsp;&nbsp;安全阀出厂编号: "+rs_select_group.getString("productno")+"<br>";
    ss+="&nbsp;&nbsp;&nbsp;安全阀制造单位: "+rs_select_group.getString("manufacture")+"<br>";
    ss+="&nbsp;&nbsp;&nbsp;安全阀型号: "+rs_select_group.getString("valvecate")+"<br>";
    while(rs_select_group.next()){
        count++;
        ss+=count+". 安全阀编制ID: "+rs_select_group.getString("valnumber")+"<br>";
        ss+="&nbsp;&nbsp;&nbsp;安全阀出厂编号: "+rs_select_group.getString("productno")+"<br>";
        ss+="&nbsp;&nbsp;&nbsp;安全阀制造单位: "+rs_select_group.getString("manufacture")+"<br>";
        ss+="&nbsp;&nbsp;&nbsp;安全阀型号: "+rs_select_group.getString("valvecate")+"<br>";
    }
}
PrintWriter pw=response.getWriter();
response.setContentType("text");
pw.write(ss);
pw.close();
```

图 5-15 后台自动显示委托单内安全阀信息代码

### 3) 获取合适的存储位置

通过点击是单个存储还是多个存储后点击获取存储位置按钮，可获取一个合适的安全阀存储位置。后台处理数据则是根据存储位置信息编号，按照要求，筛选出合适位置，筛选条件为：存储位置是否为空，是否在未检区域，是否满足单个存储区域或组存储区域要求。后台 Java 代码如下图 5-16 所示。

```

<jsp:useBean id="connect" class="com.xfzhang.bean.connection" />
<%
String volum=request.getParameter("volume");
try {
    ResultSet rs=null;
    String lo=new String();
    if(volum.equals("S")){
        lo="select * from locationinfo where mark=1 and locationstatus=0 limit 1";
    }else if(volum.equals("L")){
        lo="select * from locationinfo where mark=2 and locationstatus=0 limit 1";
    }
    rs=connect.query(lo);
    while (rs.next()) {
        String sln=rs.getString("storagelocationnum");
        PrintWriter pw=response.getWriter();
        response.setContentType("html/text");
        pw.write(sln);
        pw.close();
    }
} catch (Exception e) {
    out.print("get data error!");
}
}

```

图 5-16 后台获取可存储位置

#### 4) 开始存储

点击存储按钮开始运行后台存储代码，需要先对存储的委托单进行信息核实，委托单时数据库内存在的委托单，且输入的仓库位置信息可用，则进行存储，若成功，则返回存储成功信号。后台存储部分代码如下图 5-17 所示。

```

if(rs.next()){
    f=1;
    String lo1="select * from locationinfo where storagelocationnum=\""+
    +storagelocationnum+"\"";
    rs=connect.query(lo1);
    if(rs.next()){
        ff=1;
        if(rs.getInt("locationstatus")==0){
            String modify="update locationinfo set locationstatus=1,"
            +"valorgroupnumber=\""+valorgroupnumber+"\" where storagelocationnum=\""+
            +storagelocationnum+"\"";
            String insert="insert into valsavestatusinfo values('"+valorgroupnumber
            +"', '"+valvolume+"', '"+storagelocationnum+"', '"+opaction+"', '"+
            +manindex+"', '"+useraccount+"', '"+optime+"', '"+valstatus+"', '"+null+"')";
            int flag_modify=connect.addquery(modify);
            int flad_insert=connect.addquery(insert);
            if(flag_modify!=0&&flad_insert!=0){
                String delete="delete from willbesaved where acceptno='"+acceptno+"";
                int flag_delete=connect.addquery(delete);
                if(flag_delete!=0){
                    PrintWriter pw=response.getWriter();
                    response.setContentType("text");
                    pw.write("存储成功!");
                    pw.close();
                }
            }
        }
    }
}

```

图 5-17 未检入库后台存储代码

### 5.2.4 已检入库

已检入库与未检入库流程相似，但由于检验之后的安全阀需要根据是否合格分开存放，故输入委托单后弹出的信息中的安全阀前面是 checkbox 选择框，当选择框确认后后台根据安全阀的数量自动分配合格存储区域，当然，如果操作员有合适的区域也可以修改。此

后点击确认入库后需要把委托单安全阀信息中的是否合格字段按照选择修改为 yes 或 no，随后进行存储。该部分界面实现如下图 5-18 所示。

图 5-18 已检入库界面

当输入安全阀委托单号时，右边的安全阀合格确认位置会出现委托单内安全阀信息，此处后台调用与上一节未检入库时的返回安全阀信息原理相同，只不过在信息前添加了一个 id 为安全阀编号的 checkbox 组件。此时的每个安全阀前面都有 checkbox，选择合格安全阀打钩后点击确认按钮，此时系统通过 ajax 把 checkbox 勾选的 id 的 id 信息传入后台，ajax 传输 checkbox 数据数组进后台的代码如下图 5-19 所示。

```
var volume=$('#input:radio:checked').val();
var checkedcount=0;
$('#input:checkbox:checked').each(function() {
    checkedcount++;
    checked.push($(this).val());
});
$.ajax({
    cache: false,
    type: "POST",
    url:"jsp/show.jsp?option=getcheckedgroup&&valorgroupnumber="+
        +valorgroupnumber+"&&volume="+volume,
    traditional :true,
    data:{"checkedid":checked}, //要发送的是ajaxForm表单中的数据
    dataType:'json',
    async: false,
    error: function(request) {
        alert("发送请求失败！");
    },
},
```

图 5-19 ajax 传输 checkbox 数组信息

此时，后台对数组信息进行接收，后台对已确认的安全阀信息更新其检验情况为合格，对未打钩的安全阀更新其检验信息为不合格。后台获取数组数据的代码为：

```
String s[]=request.getParameterValues("checkedid");
```

已检入库后台存储代码与未检入库及基本相同，不同的是已检入库时安全阀状态需要修改为已检状态（合格或不合格），未检入库安全阀状态是未检状态。



5.2.5 信息浏览

该部分主要是对安全阀存储状态表内容的显示，就是利用 ajax 技术把后台存储状态表数据通过 json 数据格式传入前台并显示的过程，界面如下图 5-20 所示。

存储信息

10 records per page

Search:

安全阀编号	委托单号	存储位置	操作时间	存入／取出	状态	更多
g0000002	20170415120337	60100	2017-05-21 18:23:12.0	存入	已检在库	浏览
00000021	20170427223519	10100	2017-05-17 23:05:42.0	取出	备检出库	浏览
g0000004	20170415125932	60101&40101	2017-05-17 23:01:00.0	取出	检毕出库	浏览
g0000009	20170514165744	60100&40100	2017-05-17 22:58:08.0	取出	检毕出库	浏览
00000027	20170514165744	60100	2017-05-17 21:42:16.0	取出	检毕出库	浏览
g0000004	20170415125932	60101&40101	2017-05-17 21:23:57.0	存入	已检在库	浏览
g0000004	20170415125932	20102	2017-05-17 21:20:24.0	取出	备检出库	浏览

图 5-20 安全阀操作信息浏览界面

此处的 ajax 部分成功获取后台信息后，把获取的 json 信息转换为 html 格式字符串，随后插入特定的表格信息显示区域，对返回后的 json 信息进行数据处理部分的代码如下图 5-21 所示。

```

success: function(data) {
    var N=new Array();
    var i=0;
    var insert="";
    for(var n=0;n<data.length;n++){
        insert+="|  |
| --- |
|";
        insert+="  |

```

图 5-21 返回的浏览信息前端处理代码

后台通过按时间倒叙限制条件对操作信息表（valsavestatusinfo）查询，随后把查询出的数据保存为 json 格式后，返回至前台。核心代码如下图 5-22 所示。



```

while(rs.next()){
    if(rs.getString("valnumber").substring(0,1).equals("g")){
        select_checkorder="select * from val_information where groupnum='"
+rs.getString("valnumber")+"'";
    }else{
        String isgroup="select * from val_information where valnumber='"
+rs.getString("valnumber")+"'";
        ResultSet rs_isgroup=connect.query(isgroup);
        if(rs_isgroup.next()){
            if(rs_isgroup.getString("groupnum")==null){
                select_checkorder="select * from checkorder where valnumber='"
+rs.getString("valnumber")+"'";
            }else{
                select_checkorder="select * from checkorder where valnumber='"
+rs_isgroup.getString("groupnum")+"'";
            }
        }
        ResultSet rs_checkorder=connect.query(select_checkorder);
        if(rs_checkorder.next()){
            acceptno=rs_checkorder.getString("acceptno");
        }
        JSONObject ob=new JSONObject();
        ob.addProperty("valnumber",rs.getString("valnumber"));
        ob.addProperty("valvolume",rs.getString("valvolume"));
        ob.addProperty("storagelocationnum",rs.getString("storagelocationnum"));
        ob.addProperty("opaction", rs.getString("opaction"));
        ob.addProperty("manindex",rs.getString("manindex"));
        ob.addProperty("useraccount", rs.getString("useraccount"));
        ob.addProperty("optime", rs.getString("optime"));
        ob.addProperty("valstatus", rs.getString("valstatus"));
        ob.addProperty("exlocationnum", rs.getString("exlocationnum"));
        ob.addProperty("acceptno", acceptno);
        array.add(ob);
    }
}
System.out.println(array.toString());
PrintWriter pw=response.getWriter();
response.setContentType("text/json");
pw.write(array.toString());
pw.close();

```

图 5-22 信息浏览后台数据的获取与返回

对于已显示的信息进行相应信息下载生成 Excel 表格形式文件保存内容，下载按钮 export 键如下图 5-23 所示。

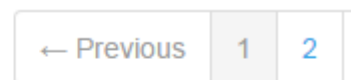
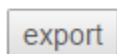


图 5-23 export 下载操作信息按钮

该 export 按钮功能实现代码如下图 5-24 所示。

```

function tabletoExcel(mytable) {
    // getExplore()返回1, 说明是不是Google Chrome、
    // Firefox、Opera、Safari, 那么就认为是IE了。
    if (getExplorer() == 1) {
        //是IE的话, 就调用toExcel()方法来导出Excel表格,
        //不依赖微软的Excel产品。(toExcel()方法的定义见下面)
        toExcel(mytable, '');
        return;
    }
    var table=document.getElementById("ddtable");
    // 克隆(复制)此table元素, 这样对复制品进行修改(如添加或改变table的标题等),
    //导出复制品, 而不影响原table在浏览器中的展示。
    table = table.cloneNode(true);

    var uri = 'data:application/vnd.ms-excel;base64,',
    template = '<html xmlns:o="urn:schemas-microsoft-com:office:office" '
        + 'xmlns:x="urn:schemas-microsoft-com:office:excel" '
        + 'xmlns=
```

5.2.6 出库

出库界面如下图 5-26 所示.



图 5-26 委托单出库界面

请求该页面时，页面自动调用后台程序获取可供出库条件的数据，并通过 json 传入前台，前台获取 json 数据后动态添加可出库委托单信息在网页上生成表格，此处界面显示原理与上节信息浏览类似，每一栏的信息右端一个出库按钮。点击按钮进行出库操作，出库操作主要分委托单整单出库和已检验的安全阀单个出库。

1. 委托单整单出库

点击该按钮实现方法 deleteRow(r)中获取该 r 行某一列信息实现方法代码如下图 5-27 所示.

```
var rows=r.parentNode.parentNode.rowIndex;
var valorgrouppnumber=document.getElementById('table').rows[rows].cells[0].innerText;
```

图 5-27 获取表格当前行某一列信息方法

后台实现代码分为两个部分，当对安全阀状态进行判断后，确认此次出库是已检出库还是未检出库，对该次操作安全阀状态进行判断，随后进行相应状态处理。重新加载可出库表单信息，出库时主要有两个操作，一个是对当前委托单安全阀存储位置进行查询后把该存储位置信息修改为空，另一个是插入出库操作信息。具体代码如下图 5-28 与 5-29 所示.

```

ResultSet rs = null;
String savevalve = "select * from locationinfo where locationstatus=1 "
+"and valorgrouppnumber='" + opnumber
    + "' order by mark desc";
String getinfo = "select * from valsavestatusinfo where valnumber='"
    + opnumber + "' order by optime desc";
ResultSet rs_select_getinfo = connect.query(getinfo);
if (rs_select_getinfo.next()) {
    valvolume = rs_select_getinfo.getString("valvolume");
    storagelocationnum = rs_select_getinfo.getString("storagelocationnum");
    ResultSet rs_select_save = connect.query(savevalve);
    if (status.equals("N") || status.equals("C")) {
        valstatus = "C";
    } else if (status.equals("Y") || status.equals("O")) {
        valstatus = "O";
    }
    if (rs_select_save.next()) {
        String exlocationnum = null;
        String location = rs_select_save.getString("storagelocationnum");
        String modify = "update locationinfo set locationstatus=0,"
            + "valorgrouppnumber=null where storagelocationnum='"
                + location + "'";
        int flag = 0, flag1 = 0;
        flag = connect.addquery(modify);
        if (rs_select_save.next()) {
            exlocationnum = rs_select_save.getString("storagelocationnum");
            modify = "update locationinfo set locationstatus=0,"
                + "valorgrouppnumber=null where storagelocationnum='"
                    + exlocationnum + "'";
            flag1 = connect.addquery(modify);
        }
    }
}

```

图 5-28 修改存储位置为空

```

String insert_out_info = "insert into valsavestatusinfo values('"
+ opnumber + "','" + valvolume + "','" + storagelocationnum + "','"
+ opaction + "','" + manindex + "','" + useraccount
+ "','" + optime + "','" + valstatus + "','" + exlocationnum + "')";
int rs_insert_out = connect.addquery(insert_out_info);

```

图 5-29 插入存储操作信息

## 2. 委托单内单个安全阀出库

该部分待出库信息不是以表单信息整体安全阀作为一个单位，是以每个安全阀作为一个整体单位，当点击委托单出库栏的浏览按钮时，下方单个出库表格会出现该委托单内能单个出库的安全阀，此处单个出库只在最后一次安全阀已检出库时可使用，在出库检验时还是需要根据表单出库。单个出库表格显示的每个安全阀前面都有 checkbox 按钮，以及其当前校验结果，当勾选后，点击出库按钮，实现单个安全阀单独出库。界面如下图 5-30 所示。

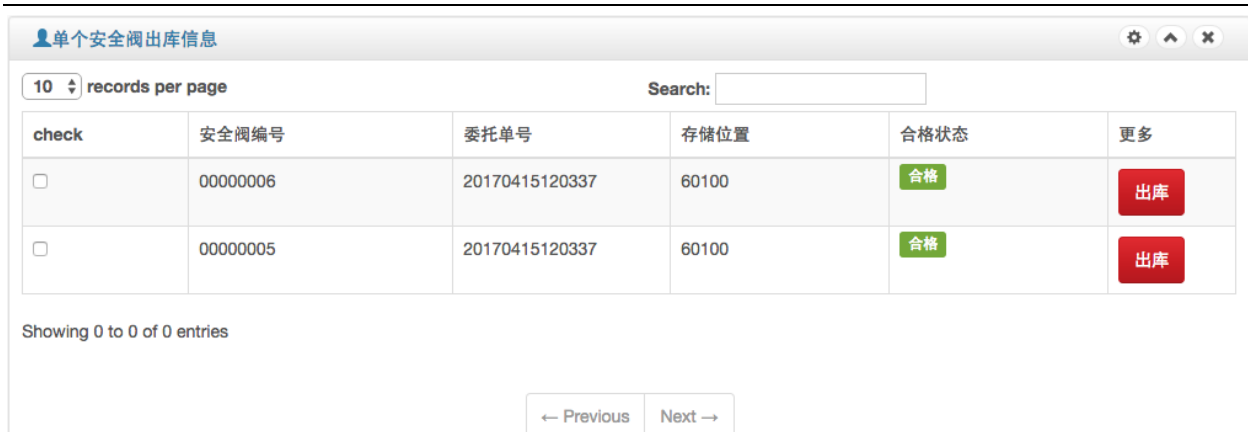


图 5-30 单个安全阀出库部分界面显示

此部分实现是把 checkbox 勾选的值作为数组传入后台(\*.jsp)文件中，具体 ajax 代码实现与上文已检入库确认安全阀合格部分类似，故不再详细说明。后台对单个安全阀的出库处理主要有三部分：1) 插入该安全阀出库操作信息；2) 判断该安全阀是否为当前委托单的最后一个安全阀，如果是，直接执行委托单出库操作；3) 判断该安全阀是否为当前存储位置的最后一个安全阀，如果是，对当前存储位置修改置空位，对安全阀出库的委托单操作进行插入。

插入安全阀操作信息的具体代码如下图 5-31 所示。

```
String insert = "insert into valsavestatusinfo values('" + s[i] + "','" +
+ valvolume + "','" + storagelocationnum + "','" + opaction + "','" + manindex
+ "','" + useraccount + "','" + optime + "','" + valstatus + "','" + null + "')";
int flag = connect.addquery(insert);
if (flag != 0 && FF == 1) {
    String update = "update val_information set isvalid='no' where valnumber='"
+ s[i] + "'";
    int flag_update = connect.addquery(update);
}
```

图 5-31 插入单个安全阀操作信息，并修改其状态为无效

当安全阀委托单含多个安全阀且其内分合格和不合格两部分存储，而此时安全阀只含一个存储位置，即委托单内的合格或不合格的一类安全阀已经全部出库了，则对安全阀备选存储位置值空，并插入一条该委托单的修改操作，具体代码如下图 5-32 所示。



```

storagelocationnum = rs_valsave.getString("storagelocationnum");
String select_val = "select * from val_information where groupnum='"
    + rs.getString("groupnum") + "'and isqualify='yes'";
String select_val1 = "select * from val_information where groupnum='"
    + rs.getString("groupnum") + "'and isqualify='no'";
ResultSet rs_val = connect.query(select_val);
ResultSet rs_val1 = connect.query(select_val1);
Boolean a = !rs_val.next(); Boolean c = !rs_val1.next();
if (FF == 1 && exlocationnum != null && c) {
    String update_location = "update locationinfo set locationstatus=0"
        + ",valorgroupnumber=null where storagelocationnum='"
        + exlocationnum + "'";
    String opa="X";
    String insertstatus = "insert into valsavestatusinfo values('"
        + rs.getString("groupnum") + "','"
        + rs_valsave.getString("valvolume")+ "','"
        + rs_valsave.getString("storagelocationnum") + "','"
        + opa + "','" + rs_valsave.getString("manindex") + "','"
        + rs_valsave.getString("useraccount") + "','" + optime + "','"
        + rs_valsave.getString("valstatus") + "','" + null + "')";
    int rs_insertstatus = connect.addquery(insertstatus);
    int rs_uplocation = connect.addquery(update_location);
}

```

图 5-32 委托单信息判定修改委托单内安全阀存储位置

如果校验结果为合格的安全阀已全部出库，而不合格的还在，则把不合格安全阀的存储位置放置在主存储位置，若安全阀已经全部出库，则添加委托单整体出库信息，具体代码如下图 5-33 所示。

```

if (FF == 0 || a) {
    String update_location = "update locationinfo set locationstatus=0,"
        + "valorgroupnumber=null where storagelocationnum='"
        + storagelocationnum + "'";
    System.out.println(2 + update_location);
    int rs_uplocation = connect.addquery(update_location);
    if (rs_uplocation == 0) {
        PrintWriter pw = response.getWriter();
        response.setContentType("text");
        pw.write("存储位置更新失败");
        pw.close();
        return;
    } else if (FF == 1) {
        String insertstatus = null;
        optime = sdf.format(d);
        if (exlocationnum != null && !c) {
            insertstatus = "insert into valsavestatusinfo values('"
                + rs.getString("groupnum") + "','"
                + rs_valsave.getString("valvolume") + "','"
                + exlocationnum + "','"
                + rs_valsave.getString("opaction") + "','"
                + rs_valsave.getString("manindex") + "','"
                + rs_valsave.getString("useraccount") + "','"
                + optime + "','" + rs_valsave.getString("valstatus")
                + "','" + null + "')";
        } else {
            insertstatus = "insert into valsavestatusinfo values('"
                + rs.getString("groupnum") + "','"
                + rs_valsave.getString("valvolume") + "','"
                + storagelocationnum + "','" + opaction + "','"
                + manindex + "','" + useraccount + "','" + optime
                + "','" + valstatus + "','" + null + "')";
        }
    } //当委托单内安全阀全部出库之后
    int flag_updatestatus = connect.addquery(insertstatus);
}

```

图 5-33 单个安全阀出库操作信息处理

5.2.7 进出库操作确认

该处信息的显示与信息浏览部分的实现方法基本一致，只是后台获取信息的表格修改了，故显示功能代码参考 5.2.5 信息浏览。该部分添加的主要功能是对手机端的存储操作的信息进行确认处理，即根据表内显示的安全阀状态进行相应的数据库信息处理。主要分为入库操作和出库操作以及操作取消。界面显示如下图 5-34 所示。



图 5-34 移动操作信息确认

当点击确认入库，按钮响应函数为 addinfo()，当点击确认出库后，按钮响应函数是 deleteinfo()，而当点击取消按钮后，系统自动删除手机端添加的这条存储操作，使得手机端操作端有一定的容错性。

1. 入库

当点击确认入库时，执行函数 addinfo()，通过 ajax 把确认入库的相关信息传入后台，具体后台代码如下图 5-35 所示。

```
String modify="update locationinfo set locationstatus=1,valgrouppnumber=\""+
+valgrouppnumber+"\" where storagelocationnum=\""+storagelocationnum+"\"";
String insert="insert into valsavestatusinfo values('"+valgrouppnumber+"',"+
+valvolume+"', '"+storagelocationnum+"', '"+opaction+"', '"+manindex+"', '"+
+useraccount+"', '"+optime+"', '"+valstatus+"', '"+exlocationnum+"')";
System.out.println(insert);
int flag_modify=connect.addquery(modify);
int flad_insert=connect.addquery(insert);
if(flag_modify!=0&&flad_insert!=0){
    String delete_pre="delete from preparetochangeinfo where valgrouppnumber='"+
+valgrouppnumber+"\"";
    String delete=null;
    if(valstatus.equals("N")){
        delete="delete from willbesaved where valgrouppnumber='"+
+valgrouppnumber+"\"";
    }else if(valstatus.equals("Y")){
        delete="delete from checkedwillbesaved where valgrouppnum='"+
+valgrouppnumber+"\"";
    }
    int flag_delete=connect.addquery(delete);
    int flag_delete_pre=connect.addquery(delete_pre);
}
```

图 5-35 进库操作信息确认处理

2. 出库

当点击确认出库时，执行函数 `deleteinfo()`，该函数先通过一个 `ajax` 技术核查此出库信息是表单里的单个安全阀信息还是整个表单信息，然后根据返回的信息，把数据传入不同的后台代码进行后台委托单出库信息执行。

后台安全阀出库代码包括表单整出库和表单内安全阀出库，表单整出库代码与出库操作代码基本重合，不同的是信息接收的途径，具体代码区分如下图 5-36 所示。

```
if (option.equals("pc")) {
    manindex = request.getParameter("manindex");
    status = request.getParameter("status");
    System.out.println(status+" test ");
    SimpleDateFormat sdf = new SimpleDateFormat("yyyyMMddHHmmss");
    Date d = new Date();
    optime= sdf.format(d);
}else if(option.equals("android")){
    String select_info="select * from preparetochangeinfo "
        +"where valorgrounumber='"+opnumber+"'";
    ResultSet rs_select_info=connect.query(select_info);
    if(rs_select_info.next()){
        manindex=rs_select_info.getString("manindex");
        optime=rs_select_info.getString("optime");
        optime = optime.replaceAll("-", "");
        optime = optime.replaceAll(" ", "");
        optime=optime.replaceAll(":", "");
        optime=optime.substring(0,14);
        status=rs_select_info.getString("valstatus");
    }
}
```

图 5-36 移动端操作出库和 PC 端操作出库信息途径区分

### 5.3 手机端功能模块

Android 功能模块主要登入模块，仓储信息浏览模块，安全阀入库（包括未检入库和已检入库）模块，安全阀未检出库模块，安全阀已检出库模块。

本次 Android 编程环境没有使用 Eclipse 和 ADT，而是使用 Google 发行的专门为 Android 开发者使用的开发工具 Android Studio。

#### 5.3.1 手机端与服务器通信

Android 与服务器通信通常采用 Http 通信方式或 Socket 通信方式，本次通信选用的是 Http 通信方式。而 Http 通信方式又分为 GET 和 POST 两种方式。POST 请求可以向服务器传送数据，而且数据放在 HTML HEADER 内一起传送到服务端，数据对用户不可见。而 GET 请求则是把参数数据依次添加到提交的 URL 中，用户可直接在 URL 上看到数据，其值和表单内字段一一对应。GET 方式数据传输安全性极地，而 POST 方式相比而言安全性较高。现针对 Android 与服务器通信功能这一需求构建手机端 APP 与服务器的通信类。

对于 GET 方法实现，由于 GET 方式传送机制中，对于数据的传送是直接把数据参数添加到 URL 上，故首先对 GET 方法传入的参数数据按固定格式添加到 URL 尾部，然后通过 URL 打开一个 `HttpURLConnection` 链接，该链接可以获得一个 `InputStream` 字节流对象，最后将返回的 `conn.getInputStream()`通过编码转换为字符串格式返回 Android 端显示。GET 方法的具体实现代码如下图 5-37 所示。



```

public static String localhost="192.168.1.101";
public static String getDataByGet(String url,Map<String,String>params,String charset) {
    if (url == null) {
        return null;
    }
    url = url.trim();
    URL targetUrl = null;
    try {
        if (params == null) {
            targetUrl = new URL(url);
        } else {
            StringBuilder sb = new StringBuilder(url + "?");
            for (Map.Entry<String, String> me : params.entrySet()) {
                //解决请求参数中含有中文导致乱码问题
                sb.append(me.getKey()).append("=").append(URLEncoder.encode(
                    me.getValue(), charset)).append("&");
            }
            sb.deleteCharAt(sb.length() - 1);
            targetUrl = new URL(sb.toString());
        }
        Log.i(TAG, "get:url----->" + targetUrl.toString()); //打印log
        HttpURLConnection conn = (HttpURLConnection) targetUrl.openConnection();
        conn.setConnectTimeout(3000);
        conn.setRequestMethod("GET");
        conn.setDoInput(true);
        int responseCode = conn.getResponseCode();
        if (responseCode == HttpURLConnection.HTTP_OK) {
            String ss=stream2String(conn.getInputStream(), charset);
            // Log.d("response.get",ss);
            return ss;
        }
    } catch (Exception e) {
        Log.i(TAG, e.getMessage());
    }
    return null;
}

```

图 5-37 通信类内 GET 方法

对于 POST 方法，因其数据传输不在 URL 中实现，对于数据可传输量比 GET 方式大。POST 传输是通过对 HttpURLConnection 进行设置，使支持 POST 传输，而后通过相关属性传递参数。具体代码如下图 5-38 所示。

```

public static String getDataByPost(String url, Map<String, String> params, String charset) {
    if(url == null)
    {return null;}
    url = url.trim();
    URL targetUrl = null;
    OutputStream out = null;
    try{
        targetUrl = new URL(url);
        HttpURLConnection conn = (HttpURLConnection) targetUrl.openConnection();
        conn.setConnectTimeout(3000);
        conn.setRequestMethod("POST");
        conn.setDoInput(true);
        conn.setDoOutput(true);

        StringBuilder sb = new StringBuilder();
        if(params!=null && !params.isEmpty()) {
            for(Map.Entry<String,String> me : params.entrySet()) {
                // 对请求数据中的中文进行编码
                sb.append(me.getKey()).append("=").append(URLEncoder.encode(
                    me.getValue(), charset)).append("&");
            }
            sb.deleteCharAt(sb.length()-1);
        }
        byte[] data = sb.toString().getBytes();
        conn.setRequestProperty("Content-Type", "application/x-www-form-urlencoded");
        conn.setRequestProperty("Content-Length", String.valueOf(data.length));
        out = conn.getOutputStream();
        out.write(data);
        Log.i(TAG, "post:url----->" + targetUrl.toString()); //打印log

        int responseCode = conn.getResponseCode();
        if(responseCode == HttpURLConnection.HTTP_OK)
        {
            String ss = stream2String(conn.getInputStream(), charset);
            return ss;
        }
    } catch (Exception e) {
        Log.i(TAG, e.getMessage());
    }
    return null;
}

```

图 5-38 通信类内 POST 方法

由于数据的传输是通过字节流传输的，通过数据转换，把字节流转换为字符串进行输出，具体方法实现代码如下图 5-39 所示。

```

private static String stream2String(InputStream in, String charset) throws IOException
{
    if(in == null)
        return null;
    byte[] buffer = new byte[1024];
    ByteArrayOutputStream bout = new ByteArrayOutputStream();
    int len = 0;
    while((len = in.read(buffer)) != -1)
    {
        bout.write(buffer, 0, len);
    }
    String result = new String(bout.toByteArray(), charset);
    in.close();
    return result;
}

```

图 5-39 通信类内方法文字编码转换

### 5.3.2 登陆模块

手机端的使用用户是仓库内的货物搬运员，用户通过输入用户名和密码后发送给服务端验证信息，服务器端接收来自 Android 的信息，进行用户信息核实，若成功返回 success，若失败返回 failed，Android 端再根据是否登入成功判定是否需要进行新界面。登陆界面如下图所示 5-40 所示，成功登陆后主页界面如下图所示 5-41 所示。

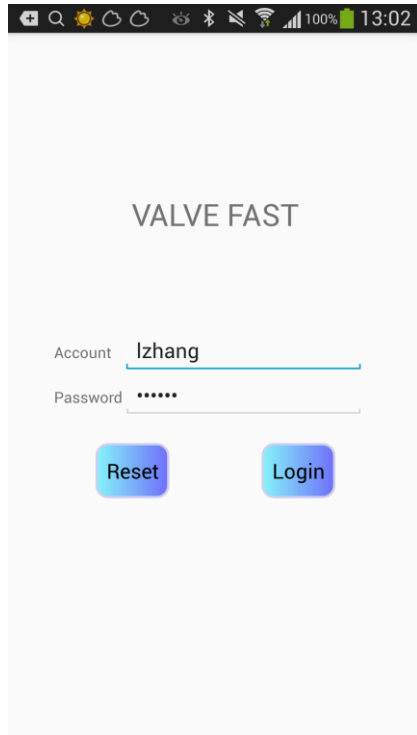


图 5-40 手机端登入界面



图 5-41 手机登录后主页

登陆信息提交到后台数据库具体代码如下图 5-42 所示。

```
Runnable newTread = new Runnable() {
    public void run() {
        EditText user = (EditText)findViewById(R.id.useraccount);
        String userac=user.getText().toString().trim();
        EditText password = (EditText) findViewById(R.id.password);
        String passwd=password.getText().toString().trim();
        Map<String,String> usermessage=new HashMap<>();
        usermessage.put("useraccount",userac);
        usermessage.put("password",passwd);
        response=PostUtils.getDataByPost(LOGIN_URL,usermessage,"utf8");
        if(response.equals("success")){
            result="登陆成功";
            handler.sendMessage(0x123);
            Intent it=new Intent().setClass(MainActivity.this,Homepage.class);
            Bundle bundle=new Bundle();
            bundle.putString("account", userac);
            it.putExtras(bundle);
            startActivity(it);
            MainActivity.this.finish();
        }else if(response.equals("failed")){
            result="密码或账号错误";
            handler.sendMessage(0x123);
        }
    }
};
```

图 5-42 新线程提交登入信息并对返回信息进行核查

服务器后台处理代码如下图 5-43 所示，成功返回 success，失败返回 failed.

```
<jsp:useBean id="connect" class="com.xfzhang.bean.connection"></jsp:useBean>
<% request.setCharacterEncoding("UTF-8");
    ResultSet rs=null;
    String account=request.getParameter("useraccount");
    String passwd=request.getParameter("password");
    String lo="select * from workman where manindex='"+account+"'";
    rs=connect.query(lo);
    String ss="";
    out.println(lo);
    while(rs.next()) {
        String pd=rs.getString("manpassword");
        if(rs.getString("manpassword").equals(passwd)){
            ss="sucess";
        }else{
            ss="failed";
        }
    }
    PrintWriter pw=response.getWriter();
    response.setContentType("text");
    pw.write(ss);
    pw.close();
}%>
```

图 5-43 登入信息核查

由于后续模块通信流程基本相似，故关于 Android 端通信则不再细讲.

### 5.3.3 信息浏览

手机端界面如下图 5-44 所示.

浏览				
编号	位置	动作	时间	状态
g0000002	60100	入库	2017-05-21	Y
00000021	10100	出库	2017-05-17	C
g0000004	60101	出库	2017-05-17	O
g0000009	60100	出库	2017-05-17	O
00000027	60100	出库	2017-05-17	O
g0000004	60101	入库	2017-05-17	Y
g0000004	20102	出库	2017-05-17	C
00000021	10100	入库	2017-05-17	N
g0000002	30100	出库	2017-05-17	O
g0000002	30100	入库	2017-05-17	Y
g0000002	20101	出库	2017-05-17	C
g0000004	20102	入库	2017-05-17	N
g0000002	20101	入库	2017-05-17	N
g0000005	20100	入库	2017-05-17	N
g0000009	60100	入库	2017-05-14	Y
g0000009	20100	出库	2017-05-14	C
g0000009	20100	入库	2017-05-14	N
00000002	50100	出库	2017-05-14	O
00000002	50100	入库	2017-05-14	Y
00000002	10100	出库	2017-05-14	C
00000002	10100	入库	2017-05-14	N

图 5-44 手机存储操作信息浏览界面

手机端信息浏览主要是连接服务器后台，获取服务器返回的 json 信息进行处理后通过把信息添加进 table 表格进行显示，故 Android 前端 activity 代码中需要声明 table id，具体代码如下图 5-45 所示。

```
<TableLayout
    android:id="@+id/tableshow"
    android:layout_weight="10"
    android:layout_marginTop="24px"
    android:layout_marginLeft="24px"
    android:layout_height="wrap_content"
    android:layout_width="match_parent">
</TableLayout>
```

图 5-45 手机界面 activity 代码表格声明

后台主要部分是对信息的显示，对象 response 是通信后获取的 json 信息，故手机端信息浏览后台对通信后获取 json 信息进行数据处理显示的 java 具体代码如下图 5-46 所示。

```
for (int i = 0; i < data.length()+1; i++) {
    TableRow tableRow = new TableRow(LookInfo.this);
    JSONObject mydata = data.getJSONObject(i);
    TableRow.LayoutParams lp1=new TableRow.LayoutParams(TableRow.LayoutParams
        .WRAP_CONTENT,TableRow.LayoutParams.WRAP_CONTENT);
    for (int j = 0; j < colume; j++) {
        TextView tv = new TextView(LookInfo.this);
        ViewGroup parent = (ViewGroup) tv.getParent();
        if (parent != null) {
            parent.removeAllViews();}
        tv = new TextView(LookInfo.this);
        if(colume_s[j].equals("optime")){
            String date=mydata.getString(colume_s[j]);
            date=date.substring(0,10);
            tv.setText(date);
            tv.setBackgroundResource(R.drawable.table_textview);
        }else if(colume_s[j].equals("opaction")){
            if(mydata.getString(colume_s[j]).equals("T")){
                tv.setText("出库");
                tv.setBackgroundResource(R.drawable.textview_send);
            }else if(mydata.getString(colume_s[j]).equals("S")){
                tv.setText("入库");
                tv.setBackgroundResource(R.drawable.textview_save);
            }
        }else if(mydata.getString(colume_s[j]).equals("X")){
            tv.setText("修改");
            tv.setBackgroundResource(R.drawable.textview_change);
        }
    }
    }else{
        tv.setText(mydata.getString(colume_s[j]));
        tv.setBackgroundResource(R.drawable.table_textview);}
    tv.setGravity(Gravity.CENTER);
    lp1.setMargins(5,5,5,5);
    tv.setLayoutParams(lp1);
    tableRow.addView(tv);}
ViewGroup parent = (ViewGroup) tableRow.getParent();
if (parent != null) {
    parent.removeAllViews();}
tableLayout.addView(tableRow, new TableRow.LayoutParams(MP, WC, 1));
}
```

图 5-46 对获取信息的处理显示代码



### 5.3.4 安全阀入库

安全阀入库包括安全阀未检入库和安全阀已检入库，通过扫描委托单编号以及存储位置添加安全阀预存信息进入数据库。安全阀入库操作界面如下图 5-47 所示，左半部分为直接显示界面，右半部分为点击委托单编号后显示的下拉菜单。



图 5-47 入库操作界面

#### 1. 二维码技术扫描

此处二维码扫描使用了 `simplezxing.jar` 包，进行二维码信息获取，选择扫描安全阀类型，点击开始扫描，随后，在成功获取栏对获取信息进行界面显示，通过函数 `onActivityResult` 在扫描成功部分编写对成功返回的二维码信息进行处理 Java 代码，即当 `CaptureActivity.REQ_CODE` 为 `RESULT_OK` 时，扫描成功，此时对扫描的二维码信息进行处理，具体代码如下图 5-48 所示。

```

protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    switch (requestCode) {
        case CaptureActivity.REQ_CODE:
            switch (resultCode) {
                case RESULT_OK:
                    photos=getResources().getStringArray(R.array.photoitem);
                    index=photo.getSelectedItemPosition();
                    String number=data.getStringExtra(CaptureActivity.EXTRA_SCAN_RESULT);
                    if(photos[index].equals("委托单编号")&&number.length()>6){
                        TextView tv = (TextView) findViewById(R.id.valnumber);
                        tv.setText("委托单编号:");
                        tv.setGravity(Gravity.CENTER);
                        TextView tv1 = (TextView) findViewById(R.id.addvalnumber);
                        tv1.setText(number); //or do sth
                        tv1.setGravity(Gravity.CENTER);
                        Thread t = new Thread(new Tread1);
                        t.start();
                    }else if(photos[index].equals("存储位置编号")&&number.length()<6){
                        TextView tv = (TextView) findViewById(R.id.storagenumber);
                        tv.setText("存储位置编号:");
                        tv.setGravity(Gravity.CENTER);
                        TextView tv1 = (TextView) findViewById(R.id.addstoragevalnumber);
                        tv1.setText(number); //or do sth
                        tv1.setGravity(Gravity.CENTER);
                    }else if(photos[index].equals("附加存储位置编号")&&number.length()<6){
                        TextView tv = (TextView) findViewById(R.id.exlocationnum);
                        tv.setText("附加存储位置编号:");
                        tv.setGravity(Gravity.CENTER);
                        TextView tv1 = (TextView) findViewById(R.id.addexlocationnum);
                        tv1.setText(number); //or do sth
                        tv1.setGravity(Gravity.CENTER);}
                    break;
                case RESULT_CANCELED:
                    if (data != null) {}
                    break;}
            break;
    }
}

```

图 5-48 二维码扫描技术对信息进行处理函数的使用

## 2. 委托单信息返回

扫描安全阀委托单编号后通过 POST 方式把委托单编号传入服务器，服务器对委托单表（checkorder）和安全阀信息表（val\_information）进行信息检索匹配，获取委托单内安全阀信息，并判断该委托单安全阀是否在可入库状态，以及是未检入库还是已检入库，随后把信息通过 json 格式传回手机端。服务器返回委托单入库存储状态以及委托单内安全阀信息具体代码如下图 5-49 所示。

```

if(option.equals("androidshowvalinfo")){
    String acceptno=request.getParameter("acceptno");
    String select_val="select * from val_information where acceptno='"+acceptno+"'";
    ResultSet rs_select_val=connect.query(select_val);
    String ss="";
    JsonObject object=new JsonObject();
    JSONArray array=new JSONArray();
    String select_willbe="select * from willbesaved where acceptno='"+acceptno+"'";
    ResultSet rs_willbe=connect.query(select_willbe);
    String select_checkedwillbe="select * from checkedwillbesaved where acceptno='"+acceptno+"'";
    ResultSet rs_checkedwillbe=connect.query(select_checkedwillbe);
    String status="";
    if(rs_willbe.next()){
        status="willbesaved";
    }else if(rs_checkedwillbe.next()){
        status="checkedwillbesaved";
    }else{
        status="Cantbesaved";
    }
    if(rs_select_val.next()){
        JsonObject ob=new JsonObject();
        int count=0;
        count++;
        ob.addProperty("index", count);
        ob.addProperty("valnumber",rs_select_val.getString("valnumber"));
        ob.addProperty("valproductno",rs_select_val.getString("productno"));
        ob.addProperty("manufacture",rs_select_val.getString("manufacture"));
        ob.addProperty("valvecate", rs_select_val.getString("valvecate"));
        object.addProperty("valgroupp", "val");
        array.add(ob);
        while(rs_select_val.next()){
            object.addProperty("valgroupp", "group");
            count++;
            JsonObject ob1=new JsonObject();
            ob1.addProperty("index", count);
            ob1.addProperty("valnumber",rs_select_val.getString("valnumber"));
            ob1.addProperty("valproductno",rs_select_val.getString("productno"));
            ob1.addProperty("manufacture",rs_select_val.getString("manufacture"));
            ob1.addProperty("valvecate", rs_select_val.getString("valvecate"));
            array.add(ob1);
        }
        object.addProperty("status",status);
        object.add("values", array);
        System.out.println(object.toString());
        PrintWriter pw=response.getWriter();
        response.setContentType("text / json");
        pw.write(object.toString());
        pw.close();
    }
}
}

```

图 5-49 服务器返回委托单内安全阀信息以及委托单状态

获取信息后，根据返回的委托单内安全阀已检还是未检确定返回的安全阀信息显示在手机界面时是否需要添加 checkbox，以便于存储时判断安全阀校验合格情况。显示时分为



已检和未检两种方式进行显示，已检需要确认安全阀合格情况，未检和已检两种委托单编号扫描后的手机界面显示如下图 5-50 和 5-51 所示。



图 5-50 未检委托单入库扫描编号



图 5-51 已检委托单扫描编号

### 3. 确认预存储

点击最下端确认存储按钮进行信息预存储，通过 json 格式把信息传入服务器。由于预存储分为未检入库确认和已检入库确认，已检入库确认安全阀为合格状态时的 checkbox 显示与响应方法如下图 5-52 所示。

```
if(data1.getString("status").equals("checkedwillbesaved")){
    checkBox[i] = new CheckBox(AddInfo.this);
    final int kk=i;
    JSONObject ob=new JSONObject();
    ob.put("cbvalnumber",val.getString("valnumber"));
    checkBox[i].setOnCheckedChangeListener((arg0, ischecked) → {
        if(ischecked){
            try {
                selectcheck.getJSONObject(kk).put("ischecked",true);
                Log.d("Json",selectcheck.toString());
            } catch (JSONException e) {
                e.printStackTrace();
            }
        }else{
            try {
                selectcheck.getJSONObject(kk).put("ischecked",false);
                Log.d("Json",selectcheck.toString());
            } catch (JSONException e) {
                e.printStackTrace();
            }
        }
    });
    ob.put("ischecked", false);
    selectcheck.put(i,ob);
    checkBox[i].setGravity(Gravity.CENTER);
    lp1.setMargins(5, 5, 5, 5);
    checkBox[i].setLayoutParams(lp1);
}
```

图 5-52 后台添加 checkbox 组件及响应代码

由于数据已经是手机端提供的，所以对于已检和未检两种预存储方式的后台代码可以重用，唯一的区别是对于已检入库，需要把确认合格的安全阀在安全阀信息表（val\_information）内的是否检验合格状态(isqualify)修改为“yes”，未选中安全阀的 isqualify 状态修改为 “no”。随后添加预存储信息进入预存储信息表（preparetochangeinfo），具体服务器端实现预存储操作代码如下图 5-53 所示，已检入库对安全阀检验信息修改代码如下图 5-54 所示。

```
String addtopre = "insert into preparetochangeinfo values('" + valorgrouppnumber
    + "','" + valvolume + "','" + storagelocationnum + "','" + opaction
    + "','" + manindex + "','" + optime + "','" + exlocationnum + "','"
    + valstatus + "')";
int addtopre_flag = connect.addquery(addtopre);
```

图 5-53 后台存储预存信息核心代码

```
if (option.equals("ischeckedsave")) {
    exlocationnum = request.getParameter("exlocationnum");
    int flag_modify = 0;
    int qcount = 0;
    for (int i = 0; i < check.length(); i++) {
        JSONObject ob = check.getJSONObject(i);
        String valnumber = ob.getString("cbvalnumber");
        String modify_qualify = null;
        if (ob.getString("ischecked").equals("true")) {
            qcount++;
            modify_qualify = "update val_information set isqualify='yes'"
                + " where valnumber='" + valnumber + "'";
        } else if (ob.getString("ischecked").equals("false")) {
            modify_qualify = "update val_information set isqualify='no'"
                + " where valnumber='" + valnumber + "'";
        }
        System.out.println(modify_qualify);
        flag_modify = connect.addquery(modify_qualify);
        if (flag_modify == 0) {
            Flag = 1;
            why += "modify qualify is error";
        }
    }
    if (qcount > 0 && qcount < check.length()) {
        if (exlocationnum == null) {
            Flag = 1;
            why += "未添加附加存储位置";
        }
    }
}
```

图 5-54 已检入库对安全阀检验信息修改

### 5.3.5 委托单未检批量出库

委托单未检出库是对委托单进行快速多条委托单出库，由于在送安全阀去检验时一般都是多条委托单的安全阀同时送去校验，故通过点击多条委托单就能实现同时出库。此时手机端界面如下图 5-55 所示。



图 5-55 安全阀委托单批量出库显示界面

此处与信息浏览显示界面使用的显示代码基本相同，不同之处在于每条委托单信息前都有 checkbox，故可对可出库委托单进行选择后确认出库。Checkbox 显示部分及信息响应参考上文 5.3.4 安全阀入库第 3 条，确定合格入库部分。服务器接收预出库信息并存入数据库的具体代码如下图 5-56 所示。

```
for (int i = 0; i < check.length(); i++) {
    JSONObject ob = check.getJSONObject(i);
    String ischecked = ob.getString("ischecked");
    if (ischecked.equals("true")) {
        String valorgroupnumber = ob.getString("cbvalnumber");
        String select_outstatus = "select * from valsavestatusinfo where"
        + " valnumber='" + valorgroupnumber + "' order by optime desc limit 1";
        ResultSet rs_select_outstatus = connect.query(select_outstatus);
        if (rs_select_outstatus.next()) {
            String valvolume = rs_select_outstatus.getString("valvolume");
            String storagelocationnum = rs_select_outstatus.getString(
                "storagelocationnum");
            String opaction = "T";
            String exlocationnum = rs_select_outstatus.getString("exlocationnum");
            String valstatus = rs_select_outstatus.getString("valstatus");
            if (valstatus.equals("Y")) {
                valstatus = "O";
            } else if (valstatus.equals("N")) {
                valstatus = "C";
            }
            if (exlocationnum != null) {
                exlocationnum = "" + exlocationnum + "";
            }
            addtopre = "insert into preparetochangeinfo values('" + valorgroupnumber
            + "','" + valvolume + "','" + storagelocationnum + "','" + opaction
            + "','" + manindex + "','" + optime + "','" + exlocationnum + "','"
            + valstatus + "')";
            int flag_insert = connect.addquery(addtopre);
            if (flag_insert == 0) {
                Flag = 1;
                System.out.println(addtopre);
            }
        }
    }
}
```

图 5-56 手机接收出库委托单信息进行出库信息预存储

### 5.3.6 安全阀已检出库

直接扫描委托单号或者安全阀编号实现整单出库或紧急安全阀单个出库。扫描委托单编号会显示具体安全阀信息，可对安全阀进行选择后单个预出库，把预出库安全阀编号传输到服务器后台，也可对安全阀所有信息进行全选，此时表示整单安全阀信息出库，已检出库手机端界面如下图 5-57 所示。



图 5-57 手机端已检出库界面

扫描委托单显示安全阀具体信息参考安全阀已检入库部分显示。把选择的安全阀信息以 json 格式传入服务器后，服务器对接收到的安全阀编号以及其是否选中信息进行相应出库处理，分为只含单个安全阀的委托单单个出库，含多个安全阀的委托单不完全出库以及含多个安全阀的委托单全部出库。根据不同情况，对预存储信息判断是存入多个单条出库信息还是整合到一条信息进行预存储。同时判断该次出库确认后是否部分存储区域会空出来，若是，则需添加一条修改信息。具体后台代码如下图 5-58 所示。

```

if (checkedcount == check.length() && checkedcount != 0) {
    if (exlocationnum != null) {
        exlocationnum = "" + exlocationnum + "";
    }
    if(mark.equals("group")){
        addtopre = "insert into preparetochangeinfo values('" + valorgroupnumber
            + "','" + valvolume+ "','" + storagelocationnum + "','" + opaction
            + "','" + manindex + "','" + optime + "','" + exlocationnum + "','"
            + valstatus + "')";
    }else{
        String location=null;
        String select_isqualify="select * from val_information where valnumber='"
            +valorgroupnumber+""";
        ResultSet rs_isqualify=connect.query(select_isqualify);
        if(rs_isqualify.next()){
            if(rs_isqualify.getString("isqualify").equals("no")
                && exlocationnum != null){
                location=exlocationnum;
            }else{
                location=storagelocationnum;
            }
            valvolume="S";
            addtopre = "insert into preparetochangeinfo values('" + valorgroupnumber
                + "','" + valvolume+ "','" + location + "','" + opaction + "','"
                + manindex + "','" + optime + "','" + null + "','" + valstatus + "')";
        }
    }
    int flag_insert = connect.addquery(addtopre);
    if (flag_insert == 0) {
        Flag = 1;
    }
} else {
    valvolume = "S";
    String location = null;
    for (int i = 0; i < check.length(); i++) {
        JSONObject ob = check.getJSONObject(i);
        String ischecked = ob.getString("ischecked");
        if (ischecked.equals("true")) {
            String valnumber = ob.getString("cbvalnumber");
            String select = "select * from val_information where valnumber='"
                + valnumber + """;
            ResultSet rs_select = connect.query(select);
            if (rs_select.next()) {
                if (rs_select.getString("isqualify").equals("no") &&
                    exlocationnum != null) {
                    location = exlocationnum;
                } else {
                    location = storagelocationnum;
                }
            }
            addtopre = "insert into preparetochangeinfo values('" + valnumber
                + "','" + valvolume + "','" + location + "','" + opaction
                + "','" + manindex + "','" + optime + "','" + null + "','"
                + valstatus + "')";
            int flag_insert = connect.addquery(addtopre);
            if(flag_insert==0){
                ff=1;
                why+=addtopre;
            }
        }
    }
}

```

图 5-58 二维码扫描委托单已检出库



## 第 6 章 系统测试

由于 PC 端系统测试主要是通过 Eclipse 平台结合搭建的网页服务器端进行，故测试主要是对各操作流程不断重复运行，观察是否出现未考虑到的因素引出的错误，随后根据 Eclipse 软件显示出的错误进行改错。而在手机端，由于测试需要搭建虚拟 Android 环境或连接 Android 手机，故需要对测试环境进行搭建。

由于通过 PC 端搭建测试用的 Android 虚拟机对电脑配置要求比较高，故本次 Android 测试不使用电脑搭建 Android 虚拟机，而是通过电脑连接 Android 手机进行测试，而因为本次设计所用电脑为 Mac 系统，对手机调试连接无法直接数据线插入就使用，故需要进行电脑端数据配置。

### 6.1 Android 手机连接 macOS 系统 PC 设置环境

#### 1. 环境

操作系统：macOS

Android 编程环境：Android Studio 2.3.1

外接 Android 手机型号：三星 Galaxy S4

#### 2. 连接方法

由于 macOS 系统无法对 Android 手机直接 root 操作，故需要对连接的手机进行设置，具体步骤如下：

1) 在终端输入 `system_profiler SPUSBDataType` 查看 usb 连接信息

此处我的手机显示如下：

SAMSUNG\_Android:

Product ID: 0x6860

Vendor ID: 0x04e8 (Samsung Electronics Co., Ltd.)

Version: 4.00

Speed: Up to 12 Mb/sec

Location ID: 0x14200000 / 9

Current Available (mA): 500

Current Required (mA): 96

Extra Operating Current (mA): 0

记录 ID: 0x04e8.

2) 在终端输入 `vim ~/.android/adb_usb.ini`

添加：04e8 后保存退出。

3) 重启 adb

在终端依次输入两条命令 `adb kill-server`，`adb start-server` 重启 adb。

4) 显示 adb 列表

在终端输入 `adb devices`，若显示 List of device attached 就连接成功了。

### 6.2 功能测试

对于系统测试，一般需要先进行单元测试，随后进行整体测试。本次系统单元测试是在系统单元部分编码完成时直接进行功能运行测试的。而后为了保证系统整体流程运行正

确，需对系统进行整体测试。

整个测试分为两个部分，即 PC 端整体功能测试和手机端整体功能测试。

### 6.2.1 PC 端整体功能测试

测试过程及要点：

#### 1. 委托单信息录入

通过录入委托单表单信息提供后续需要用到的数据，录入数据的时候需要录入多组可对比的数据，如直接单个安全阀委托单数据、含多个安全阀的委托单、只一个安全阀但分组存储的委托单，比较后续操作的不同。

#### 2. 未检入库

未检入库通过单个安全阀的委托单和多个安全阀的委托单两组委托单进行测试。查看后台数据库存储位置信息是否正确，待检表内委托单号是否正确删除，操作信息状态表是否对该次操作信息添加成功且信息准确。

#### 3. 未检出库

点击库内未检委托单进行出库，查看后台数据库存储位置信息是否清除为空，待入库表是否增加了该次委托单编号信息，操作信息表是否正确添加了该次出库操作信息。

#### 4. 已检入库

通过对单个安全阀委托单，多个安全阀委托单中的安全阀不同合格情况，即全合格、全不合格和部分合格部分不合格等多种不同可能性的已检入库方式进行多次测试。查看数据库内安全阀信息合格状态的修改情况，查看存储位置信息修改情况是否正确，查看操作状态表是否正确添加此次操作信息，查看已检待入库表是否删除了该次操作的委托单信息。

#### 5. 已检出库

对委托单整单出库以及委托单内单个安全阀紧急出库进行测试，查看操作信息是否正确添加，存储位置是否根据实际情况发生变化。

#### 6. 信息浏览

查看信息操作表格的信息是否正确显示。

### 6.2.2 手机端整体功能测试

#### 1. 未检入库

按照不同类型委托单对委托单进行扫描委托单号以及存储位置，对委托单进行预入库操作，随后在电脑端进行该操作信息确认，查看存储位置信息是否正确修改，操作信息是否添加成功，未检待入库表内该次委托单号是否删除。

#### 2. 未检出库

勾选需要出库检验的委托单，确认后，在 PC 端进行信息确认操作，查看此时的存储位置状态是否改变，信息操作表是否添加此次操作信息等。

#### 3. 已检入库

对通过二维码扫描的委托单号进行合格安全阀信息确认，添加存储位置后确认存储，在 PC 端确认操作信息后，查看相关信息是否修改成功。



#### 4. 已检出库

扫描委托单选择需要出库的安全阀进行出库操作，若要委托单整体出库则在显示的安全阀信息里直接点击全选就好了，随后确认出库，在 PC 端进行信息确认，查看相关信息是否修改成功。



## 第7章 结论与展望

### 7.1 结论

本次基于二维码管理的仓库管理系统在系统管理上进行新的改进创新，与以往的普通仓库管理系统有所不同，本次的仓库管理系统针对需要定检的设备在定检期间需要“两进两出”物品存储管理这一需求进行设计。鉴于本次系统的特殊性，专门为其设计移动端辅助存储，使存储时更加方便快捷。而在WEB端的设计中，采用Javascript、jQuery以及AJAX结合JSP技术的使用，使网页系统前台界面与后台程序调用更加紧密结合，信息传递简单快捷，方便系统操作。对于手机端APP，采用预存预取方式，使得WEB端仓库信息管理者只需要简单对信息进行确认即可对存储设备进行进出库信息处理。

不管开发什么系统都离不开开发环境的搭建，也正是这样，让本人在开发系统过程中，掌握搭建网页平台环境以及Android开发环境。对系统的开发有一个深入了解。通过系统开发，更是学习并掌握了之前没有接触过得Android系统开发以及AJAX技术，在开发过程中收获到新领域知识。

通过本次毕业课程设计，第一次真正的从搭建系统底层数据库到网页端前后台以及手机端APP各平台整体进行系统开发。随着开发的逐渐深入，才越发觉得自己需要学习的东西还是有很多，而现在自己使用的只是最常规的技术，在之后的日子里，需要对这些知识加深了解，学习一些系统框架的使用，使得开发系统更加有条理性。从最初的对系统开发没有任何头绪，到最终能够通过自己独立完成整体开发，不仅让我在压力中成长，最终成果也是对自己的一种鼓励，给自己加了信心。更加让自己坚信，不管最初看上去多难完成的事，通过努力，都能够做到。

### 7.2 不足之处及未来展望

由于时间有限，很多细节问题并没有深入的去优化，只能说系统整体能达到最初要求，但并没有做到最好，这也是自己在一个多月的仓促时间内对整体任务作出的一个可以算得上满意却称不上最优的答卷。以下为对开发的系统以及自身发现的几点不足之处。

第一，数据库设计有些部分并没有达到最优结构，数据表和参数带一部分冗余。

第二，关于WEB端开发，有部分内容其实可以更加简化，比如出库时对于校验委托单内的安全阀实行单个或整体出库的设计，本人其实更希望实现的是当点击出库表格行的单个出库按钮后能弹出该委托单的单个安全阀信息，然后实现单个出库，这一点，由于时间问题，网页前端界面没有完整实现出来，让我有点遗憾。

第三，关于整体系统设计，因为没有使用后台开发框架，代码条理性以及功能实现区域性不够严谨，没有让后台代码形成各自的功能模块，这也是本人在这次开发过程意识到的关于初次开发者在系统开发过程中容易出现不足，以后还需要继续学习，提升对功能模块更加合理分割能力，设计能相互独立又能紧密结合的功能模块。

希望在接下来的学习过程中，能对自身不足加强改正，规范书写代码，增强代码的可读性。



## 参考文献

- [1] Xia X X, Xin B Y. Compare B/S mode with C/S mode[J]. Journal of Yanbian University, 2002.
- [2] 张桂珠. Java 面向对象程序设计 (第 4 版) [M]. 北京: 北京邮电大学出版社, 2015.
- [3] 贾志诚, 王云. JSP 程序设计: 慕课版[M]. 北京: 人民邮电出版社: 2016.
- [4] 林博辞. Ajax 框架及 JSON 技术在 J2EE 架构中的研究与应用[D]. 大连海事大学, 2012.
- [5] Bill Phillips, Brian Hardy. Android programming: the big nerd ranch guide[M]. United States of America : Addison-Wesley Professional, 2013.
- [6] Joshua Bloch. Effective Java 英文版: 第二版[M]. 北京: 人民邮电出版社: 2009.
- [7] 张海藩, 牟永敏. 软件工程导论 (第六版) [M]. 北京: 清华大学出版社: 2013.
- [8] 林存艳. 基于 Android 平台的 QR 二维码生成与解码的研究与实现[D]. 山东师范大学, 2015.
- [9] 钱雪忠, 李京. 数据库原理及应用[M]. 北京: 北京邮电大学出版社: 2010.
- [10] 韩建宁, 高波. 数据库连接池在动态 Web 网页开发中的设计与实现[J]. 电子设计工程, 2009, 17 (4): 7-9.
- [11] 卓国峰, 郭朗. Java Web 企业项目实战[M]. 北京: 清华大学出版社: 2015.
- [12] Cox, Brad. Web applications as Java servlets[J]. Dr. Dobbs's Journal, 2001, 26 (5): 99-104.
- [13] Perry B W. Java Servlet & JSP Cookbook[M]. United States of America: O'Reilly Media, 2004.
- [14] 罗荣, 唐学兵. 基于 JDBC 的数据库连接池的设计与实现[J]. 计算机工程, 2004, 30 (9): 92-93.
- [15] Wolber D, Abelson H, Spertus E, et al. App Inventor-Create Your Own Android Apps[M], DBLP, 2011.
- [16] Zhang L N, Xiao-Lin L I. Application of JavaBean in Web Database Access[J]. Journal of Gansu Lianhe University, 2007.
- [17] 李云山. 深入浅出 Java 语言程序设计[M]. 北京: 中国青年出版社, 2003.



## 致 谢

本次毕设的课程设计是在导师钱瑛讲师悉心指导下完成的。在系统开发前期，老师对整个系统方向的指导让我对整个系统设计有一个明确的方向。在我不明白系统运作方式的时候，钱瑛老师会耐心的跟我分析整个系统的运作流程，然后在我毕设期间对系统功能的大方向要求以及细节处理，让我明确系统设计的具体要求。而后在本篇论文的书写过程，也是老师悉心指导，指出本人在论文书写中的不足之处，并给予我很大的建议。

最后，再一次衷心的感谢钱瑛老师对我毕设和毕业论文的帮助，让我顺利完成毕业课程设计以及毕业论文的书写。