**安阳师范学院本科学生毕业论**

**牲畜养殖管理系统的设计与实现**

**作　 者　　 任 超**

**学 院　　 软件学院**

**专　　业　 软件工程(Java技术方向)**

**年　　级　　 2015**

**学　 号　　 150906078**

**指导教师　　 苏 静**

**论文成绩**

**日　　期　 　 2019.05.03**

**学生诚信承诺书**

郑重承诺：所呈交的论文是作者个人在导师指导下进行的研究工作及取得的研究成果。除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写的研究成果，也不包含为获得安阳师范学院或其他教育机构的学位或证书所使用过的材料。与作者一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

作者签名：　　　　　　　　　　 日期：

导师签名：　　　　　　　　　　 日期：

院长签名：　　　　　　　　　　 日期：

**论文使用授权说明**

本人完全了解安阳师范学院有关保留、使用学位论文的规定，即：学校有权保留送交论文的复印件，允许论文被查阅和借阅；学校可以公布论文的全部或部分内容，可以采用影印、缩印或其他复制手段保存论文。保密论文在解密后遵守此规定。

作者签名：　　　　　　　　　　 日期：

导师签名：　　　　　　　　　　 日期：

目录

[1 绪论 1](#_Toc8033740)

[1.1 项目背景 1](#_Toc8033741)

[1.2 项目研究的目的和意义 1](#_Toc8033742)

[1.3 牲畜养殖管理系统概述 2](#_Toc8033743)

[2 牲畜养殖管理系统的开发环境与相关技术 2](#_Toc8033744)

[2.1 开发与运行环境 2](#_Toc8033749)

[2.2 项目所用技术分析 2](#_Toc8033750)

[2.2.1 项目整体所用技术 2](#_Toc8033751)

[2.2.2 项目所用技术详细描述 2](#_Toc8033752)

[3 牲畜养殖管理系统需求分析. 3](#_Toc8033753)

[4 牲畜养殖管理系统概要设计 3](#_Toc8033754)

[4.1 系统总体设计 3](#_Toc8033755)

[4.2 数据库总体设计 5](#_Toc8033756)

[5 牲畜养殖管理系统的详细设计 8](#_Toc8033757)

[5.1 前后端分离的系统架构设计 8](#_Toc8033758)

[5.2 登录模块的实现 8](#_Toc8033759)

[5.3 注册模块的实现 10](#_Toc8033760)

[5.4 用户信息修改及密码修改 11](#_Toc8033761)

[5.5 猪只管理模块的实现 13](#_Toc8033762)

[5.6 物品管理模块 14](#_Toc8033763)

[5.7 安全管理模块 17](#_Toc8033764)

[5.8 数据导出模块 19](#_Toc8033765)

[5.9 溯源查询模块 20](#_Toc8033766)

[6 牲畜养殖管理系统功能测试 21](#_Toc8033767)

[7 总结 23](#_Toc8033768)

[致谢 23](#_Toc8033769)

[参考文献 24](#_Toc8033770)

[Abstract 24](#_Toc8033771)

[Keywords 25](#_Toc8033772)

**牲畜养殖管理系统的设计与实现**

任 超

（安阳师范学院 软件学院, 河南 安阳 455000）

摘 要：为了让牲畜养殖业更好的与互联网结合，享受到互联网所带来的便利，因此设计了牲畜养殖管理系统。牲畜养殖管理系统是基于JAVA平台设计和开发的，它可以让养殖人员更加方便的管理养殖场。设计系统时系统架构采用当前流行的B/S架构进行设计，利用SpringBoot进行整个项目的搭建，后端通过SSM框架完成，前端主要通过AngularJS框架完成。该管理系统主要针对养猪业，实现了用户登录注册，种猪、幼猪、肉猪和生产的管理，物品、生物安全的管理，报表的导出等功能。该管理系统可以让养殖业人员更好的管理自己的养殖场，减少不必要的损失和支出，提高收益。

关键字：SpringBoot；AngularJS；SpringSecurity; Layui

# 绪论

## 项目背景

在飞速发展的互联网时代，每时每刻都有许多新的互联网产品和技术出现，而最让人们关注的是如何才能和这个时代接轨，如何才能让自己在这个时代继续生存和发展下去。在这个时候，“互联网+”出现了，它通过让传统行业和互联网结合，完成了和这个时代的接轨。畜牧养殖业也可以跟上“互联网+”的浪潮，完成自身与互联网的结合，享受到互联网所带来的便利。

关于市场方面，针对养猪业的养殖管理，目前国外有很多管理系统，例如北美的生产管理和遗传评估软件系统herdsman、猪场管理软件PigCHAMP、新西兰开发的PigWIN等等，这些软件在美国等地十分流行，但是国内却很少有用到，一个原因是因为缺少汉化版本，另一个原因是购买不方便以及购买价格昂贵。我国有关的管理系统有GPS猪场生产管理信息系统以及GBS种猪育种数据管理与分析系统等，这些软件目前适合规模较大的养殖场使用，而小型养殖场很少有用到的，同时这些软件价格也是不菲，且用起来步骤繁琐。

关于技术方面，市场上目前JAVA所使用的的最多的框架为SSM框架，即Spring+SpringMVC+MyBatis框架，而针对Spring框架又衍生出了许多其他的框架，例如专门解决安全访问控制的SpringSecurity框架，简化Spring开发的SpringBoot框架【1】等等；前端目前使用最多的还是HTML5+CSS+JavaScript，另外为了简化开发，出现了许多对JavaScript进行封装框架，例如Bootstrap、Layui、AgularJS【2】等等，可以很方便完成前端页面的搭建并解决解决前后端交互的问题。利用这些技术，完成这个项目的开发是完全可行的。

## 项目研究的目的和意义

对于国内的养猪业来说，大型的养殖管理软件如GPS、GBS对于大型养猪场很适用，但是对于普通的养殖场来说还是用不到。该项目解决了这一问题，可以让中小型养殖场也用的上管理软件。牲畜养殖管理系统可以帮助小型养殖场记录种猪、幼猪、肉猪、生产等方面的数据，让整个养殖场的猪只数据一目了然，还可以记录猪只的生物安全情况，让养殖员了解到当前养殖场的猪只体况等信息，记录每天的物品消费信息，更加方便的了解到每笔支出都用到了哪里。牲畜养殖管理系统可以更好的服务于中小型养殖场，让这些养殖场也可以体验到互联网对生活以及对自己的工作所带来的的便利，提高整个养殖场的管理水平，提高养殖场的收入。

## 牲畜养殖管理系统概述

牲畜养殖管理系统主要分为用户模块，猪只模块，物品模块，生物安全模块，报表导出模块，溯源查询模块。用户模块主要功能包括用户信息的注册、用户的登录和用户信息的更新，可以完成使用的者的信息管理；猪只模块主要包括种猪、幼猪、肉猪和生产的管理，可以完成对所有养殖的猪只的管理；物品模块主要包括对整个养殖场所购物物品的记录，可以帮助养殖人员更清楚的了解支出情况；生物安全模块主要包括对猪只安全的管理，可以帮助养殖人员了解当前养殖场内所有猪只的状态等信息；报表导出即Excel表格的导出，可以由使用者将信息导出供其他相关人员查看记录等；溯源查询主要是根据输入的耳号信息，将对应的猪只信息查询并显示出来，可以帮助使用者更清楚的了解猪只相关信息。

# 牲畜养殖管理系统的开发环境与相关技术



## 开发与运行环境

1. 开发语言：JAVA，JavaScript
2. 开发设备：Windows电脑一台，OS 10及以上版本系统
3. 开发工具：Java SE Development Kit 8 191及以上版本；IntelliJ IDEA 2019.1 x64 及以上版本；apache maven 3.6.0及以上版本开发工具：；Git version 2.21.0.windows.1 及以上版本；HBuilderX 1.2.1.20181126.及以上版本；Navicat Premium 12.1.17及以上版本
4. 数据库：MySQL 5.5.27；Redis version 2.8 及以上版本
5. Web服务器：Tomcat9.0

## 项目所用技术分析

### 项目整体所用技术

牲畜养殖管理系统采用目前开发软件首选的前后端分离的系统架构，前后端各自独立开发，互不关联，极大地降低了耦合度，同时测试功能模块时方便前后端独立测试，测试时产生的问题可以及时找出并解决。本系统前端采用HTML5、JavaScript、JQuery、AngularJS进行编写；后端采用SpringBoot、SSM框架进行编写，采用Git、Maven等辅助代码开发；数据库采用关系型数据库MySQL，使用Navicat进行数据库的视图化管理，使用Redis存储缓存数据，系统使用了SpringSecurity框架来进行安全处理。

### 项目所用技术详细描述

1. 后端采用SpringBoot+SSM框架，其中SpringBoot框架主要用于整个系统的搭建，SSM即SpringMVC+Spring+MyBatis框架【3】。SpringMVC框架主要用于控制层，提供接口，负责与前端进行数据交互；Spring框架贯穿整个后端，管理着所有的JavaBean，同时是用户业务层，负责业务逻辑部分；MyBatis框架主要用于持久层，负责与数据库的交互【4】。
2. 前端采用AngularJS框架，该框架具有Ajax异步调用和元素双向绑定的特点。Ajax异步调用可以使用户在不刷新页面的情况下与后端进行通信，并根据返回结果进行相应的处理。元素的双向绑定方便开发者对数据的操作，增加了代码的可读性。
3. 数据库使用MySQL和Redis，分别存放持久化数据和缓存数据。
4. 使用SpringBoot搭建系统框架，让Maven引入jar包的时候更加方便，不用花费大量时间去书写jar包依赖，直接引入SpringBoot已经集成好的依赖即可。
5. 通过引入SpringBoot与Tomcat集成的依赖，使得项目运行时不需要额外的Tomcat服务器，直接就可以运行项目，使得测试和运行都变得更加方便。
6. 通过引入SpringSecurity安全框架，解决了用户登录时的问题和权限控制方面的问题，使得整个项目更加安全、可控。

# 牲畜养殖管理系统需求分析.

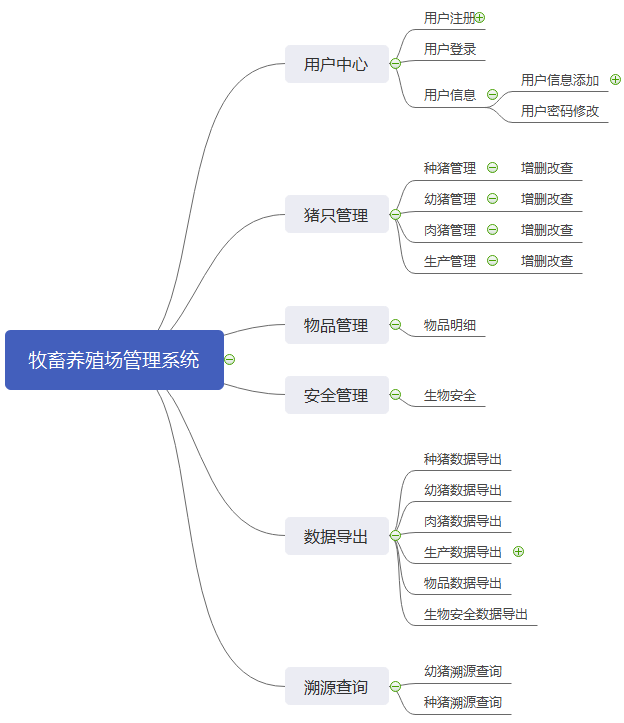
一个软件管理系统是否真的好用取决于是否管理了需要管理的信息，根据去附近几个养殖场的实地调研，总结出了以下的几个需求：

1. 要有最基本的用户登录、注册和退出功能，让用户有使用系统的入口。
2. 要有对猪只的管理，包括对种猪的管理、幼猪的管理、肉猪的管理和生产方面的管理，具体包括对种猪、幼猪、肉猪的信息记录，对种猪生产信息的记录。
3. 需要有对日常所用物品的管理，记录日常所用物品详细信息，方便查询显示。
4. 对猪只安全的管理，记录猪只的体检状况和目前状态等有关猪只安全的信息，便于及时查询并了解猪只体况等信息。
5. 数据报表的导出，要可以下载对应的Excel文件。
6. 对猪只的溯源查询，要可以根据输入的猪只耳号，查询出对应的猪只详细信息。

# 牲畜养殖管理系统概要设计

## 系统总体设计

通过对需求的分析，查询相关书籍和资料，经过多次分析和修改，最终设计出了产品需求文档，根据需求文档设计出了牲畜养殖管理系统的总体设计图，如图1所示



**图1 牲畜养殖管理系统的总体设计图**

牲畜养殖管理系统主要包括用户中心、猪只管理、物品管理、安全管理、数据导出、溯源查询六个模块，其中猪只管理又分为种猪管理、幼猪管理、肉猪管理、生产管理。这些模块的具体实现逻辑如下：

1. 用户中心模块：用户中心模块包括了用户信息注册、用户登录、用户信息更新的功能；用户注册模块中，用户可以输入用户名、密码、手机号等信息，根据输入的手机号发送验证码进行验证成功后完成注册；用户登录模块中，用户需要输入注册过的用户名和正确的密码才能登录系统；用户信息模块中，用户可以完善自己的个人信息，包括邮箱，所在地等等，还可以对密码进行修改。
2. 猪只管理模块：猪只管理模块包括了种猪管理、幼猪管理、肉猪管理和生产管理模块；这几个模块都包括了对有关信息的新增、更新、删除和查询的功能。
3. 物品管理模块：主要是对日常所购买的物品的管理，可以详细的记录所购买的每件物品的信息，记录每笔支出，便于日后进行盈亏计算。
4. 安全管理模块：主要是针对猪只生物安全的管理，包括对猪只的体况、疫苗、疾病方面的记录和管理。
5. 数据导出模块：用户可以将猪只资料和其他信息通过导出功能导出Excel表格到本地进行查看。
6. 溯源查询模块：用户可以通过输入想要查询的猪只耳号，进行对应猪只信息的查询。

## 数据库总体设计

数据库表的设计为下图表所示，其中表1为种猪管理表，对应的实体为种猪的详细信息；表2为幼猪管理表，对应的实体为幼猪的详细信息；表3为肉猪的数据库表，对应的实体为肉猪的详细信息；表4为用户表，对应的实体为用户的一些基本信息；表5为生产管理表，对应的实体为种猪生产的详细信息；表6为物品表，对应的实体为物品的详细信息；表7为生物安全表，对应的实体为记录生物安全信息所需的内容。

**表1 breeding\_pig表**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名称 | 字段类型 | 长度 | 小数点 | 允许空值 | 备注 |
| id | int | 60 | 0 | 不允许 | 主键 |
| ear\_number | varchar | 255 | 0 | 不允许 |  |
| gender | varchar | 255 | 0 | 允许 |  |
| is\_present | varchar | 255 | 0 | 允许 |  |
| age | int | 60 | 0 | 允许 |  |
| varieties | varchar | 255 | 0 | 允许 |  |
| production\_situation | varchar | 255 | 0 | 允许 |  |
| is\_transfer | varchar | 255 | 0 | 允许 |  |
| current\_house\_number | varchar | 255 | 0 | 允许 |  |
| before\_house\_number | varchar | 255 | 0 | 允许 |  |
| approach\_weight | varchar | 255 | 0 | 允许 |  |
| approach\_time | varchar | 255 | 0 | 允许 |  |
| approach\_type | varchar | 255 | 0 | 允许 |  |
| leaving\_time | varchar | 255 | 0 | 允许 |  |
| departure\_type | varchar | 255 | 0 | 允许 |  |

**表2 young\_pig表**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名称 | 字段类型 | 长度 | 小数点 | 允许空值 | 备注 |
| id | int | 60 | 0 | 不允许 | 主键 |
| ear\_number | varchar | 255 | 0 | 不允许 |  |
| mear\_number | varchar | 255 | 0 | 不允许 |  |
| gender | varchar | 255 | 0 | 允许 |  |
| age | int | 60 | 0 | 允许 |  |
| is\_transfer | varchar | 255 | 0 | 允许 |  |
| current\_house\_number | varchar | 255 | 0 | 允许 |  |
| before\_house\_number | varchar | 255 | 0 | 允许 |  |
| approach\_weight | varchar | 255 | 0 | 允许 |  |
| approach\_time | varchar | 255 | 0 | 允许 |  |
| approach\_type | varchar | 255 | 0 | 允许 |  |
| leaving\_time | varchar | 255 | 0 | 允许 |  |
| departure\_type | varchar | 255 | 0 | 允许 |  |

**表3 porker\_pig表**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名称 | 字段类型 | 长度 | 小数点 | 允许空值 | 备注 |
| id | int | 60 | 0 | 不允许 | 主键 |
| ear\_number | varchar | 255 | 0 | 不允许 |  |
| age | int | 60 | 0 | 允许 |  |
| varieties | varchar | 255 | 0 | 允许 |  |
| is\_transfer | varchar | 255 | 0 | 允许 |  |
| current\_house\_number | varchar | 255 | 0 | 允许 |  |
| before\_house\_number | varchar | 255 | 0 | 允许 |  |
| approach\_weight | varchar | 255 | 0 | 允许 |  |
| approach\_time | varchar | 255 | 0 | 允许 |  |
| approach\_type | varchar | 255 | 0 | 允许 |  |
| leaving\_time | varchar | 255 | 0 | 允许 |  |
| departure\_type | varchar | 255 | 0 | 允许 |  |

**表4 user表**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名称 | 字段类型 | 长度 | 小数点 | 允许空值 | 备注 |
| id | int | 60 | 0 | 不允许 | 主键 |
| username | varchar | 255 | 0 | 不允许 |  |
| password | varchar | 255 | 0 | 允许 |  |
| gender | varchar | 255 | 0 | 允许 |  |
| phone | varchar | 255 | 0 | 允许 |  |
| email | varchar | 255 | 0 | 允许 |  |
| location | varchar | 255 | 0 | 允许 |  |
| create\_time | varchar | 255 | 0 | 允许 |  |
| modify\_time | varchar | 255 | 0 | 允许 |  |

**表5 production表**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名称 | 字段类型 | 长度 | 小数点 | 允许空值 | 备注 |
| id | int | 60 | 0 | 不允许 | 主键 |
| ear\_number | varchar | 255 | 0 | 不允许 |  |
| age | int | 60 | 0 | 允许 |  |
| current\_house\_number | varchar | 255 | 0 | 允许 |  |
| parity | varchar | 255 | 0 | 允许 |  |
| litter\_size | varchar | 255 | 0 | 允许 |  |
| mating\_times | varchar | 255 | 0 | 允许 |  |
| abortion\_number | int | 60 | 0 | 允许 |  |
| survival\_number | int | 60 | 0 | 允许 |  |

**表6 goods表**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名称 | 字段类型 | 长度 | 小数点 | 允许空值 | 备注 |
| id | int | 60 | 0 | 不允许 | 主键 |
| recording\_time | varchar | 255 | 0 | 允许 |  |
| type | varchar | 255 | 0 | 允许 |  |
| name | varchar | 255 | 0 | 允许 |  |
| unit | varchar | 255 | 0 | 允许 |  |
| supplier | varchar | 255 | 0 | 允许 |  |
| number | varchar | 255 | 0 | 允许 |  |
| unit\_price | varchar | 255 | 0 | 允许 |  |
| actual\_expenditure | varchar | 255 | 0 | 允许 |  |

**表7 bio\_safety表**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名称 | 字段类型 | 长度 | 小数点 | 允许空值 | 备注 |
| id | int | 60 | 0 | 不允许 | 主键 |
| ear\_number | varchar | 255 | 0 | 允许 |  |
| physical\_examination | varchar | 255 | 0 | 允许 |  |
| immune\_status | varchar | 255 | 0 | 允许 |  |
| vaccination\_type | varchar | 255 | 0 | 允许 |  |
| disinfection\_situation | varchar | 255 | 0 | 允许 |  |
| disease\_condition | varchar | 255 | 0 | 允许 |  |

# 牲畜养殖管理系统的详细设计

## 前后端分离的系统架构设计

牲畜养殖管理系统采用前后端分离的架构来进行设计，这样做可以让项目的开发变得清晰明了，也可以更好的维护整个系统。前端完全使用静态页面来进行实现，使用了Layui框架可以更快的搭建出静态模板【5】，使用AngularJS框架封装的Ajsx异步请求来和后端进行数据交互，后端只提供可以供前台调用的接口服务即可；因为使用AngularJS，前端同时也使用了MVC的三层架构模式，将调用后端接口的方法分为了Service层和Controller层，页面需要和后端交互，在页面中引入对应的Service和Controller即可，不需要在HTML页面中书写大量的方法，让前端代码的可复用性提高，同时使耦合度降低；后端使用的SpringBoot和SSM框架，也让项目的耦合度极大地降低。

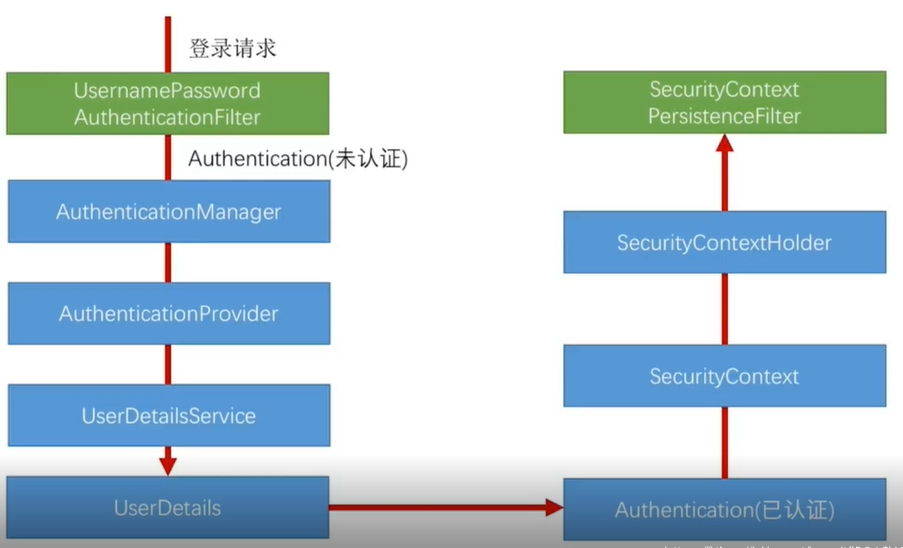
## 登录模块的实现

系统的登录界面如图2所示，当启动项目后，除了注册页面和一些必要的请求外，其他的请求都会被拦截，无论访问哪个页面都会强制跳转到登录页面，如果有账户和密码，即可登录系统，若没有，可以点击注册跳转到注册页面进行账户的注册。



**图2 系统登录界面**

系统会拦截其他的请求而放行需要的请求得益于SpringSecurity框架的引入。SpringSecurity框架是专门针对基于Spring的项目的安全框架，充分利用了Spring IoC，DI（IoC:Inversion of Control控制反转,DI:Dependency Injection依赖注入）和AOP（面向切面编程）功能。安全框架有两个重要的概念，认证（Authentication）和授权（Anthorization），认证后的用户可以访问当前系统，授权即确定当前用户在当前系统下所拥有的权限【6】，可以访问哪些页面，访问哪些请求等等，SpringSecurity的认证流程图如图3 所示



**图3 SpringSecurity的认证流程**

在项目中SpringSecurity的配置代码为：

http

.authorizeRequests()

.antMatchers("/login.html","/register.html","/css/\*\*", "/fonts/\*\*", "/images/\*\*", "/js/\*\*", "/lib.layui/\*\*","/lib/layui/\*\*","/lib/\*\*","/layui\_exts/\*\*","/register/add","/register/selectUser","/register/sendCode","/register/\*\*").permitAll()

.anyRequest().authenticated()

.and()

.formLogin().loginPage("/login.html")

.loginProcessingUrl("/login")

.defaultSuccessUrl("/index.html")

.permitAll()

.and()

.logout().logoutUrl("/logout")

.logoutSuccessUrl("/login.html")

.invalidateHttpSession(true)

.and()

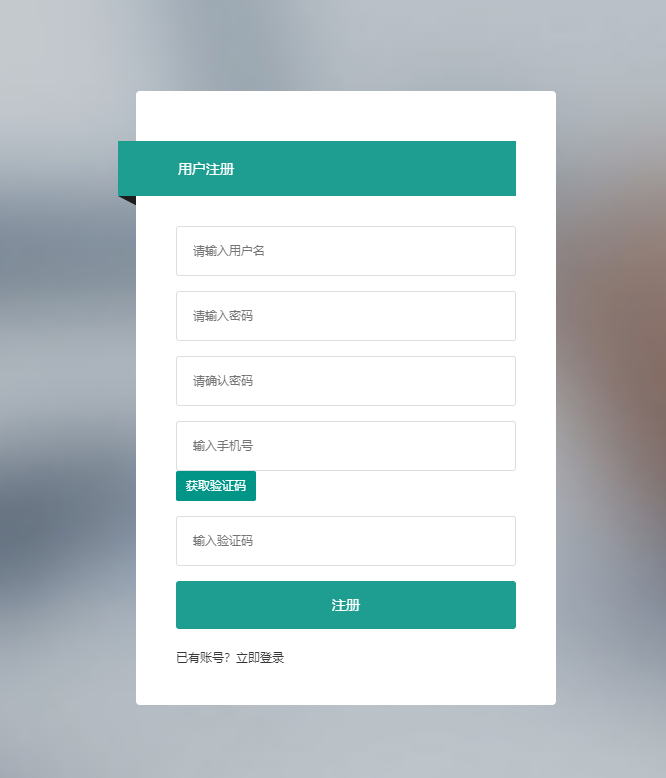
.csrf().disable()

.headers().frameOptions().sameOrigin();

自定义配置类继承WebSecurityConfigurerAdapter，重写configure方法，完成对页面的拦截与控制，重写 configureGlobal，注入自定义认证配置，自定义认证配置需要注入继承了UserDetailService 类的对象，配置完成后，就可以对用户输入的信息从数据库中进行查询，若输入的信息和查询出的信息一致，则登陆成功，否则登录失败。

## 注册模块的实现

用户注册界面如图4所示，用户通过输入注册所需的必要信息，完成注册。其中用户名需要保持其唯一性，当输入数据库中已存在的用户名时会弹窗进行提示，如果坚持注册会提示注册失败。密码输入后会在前台对两次输入的密码进行校验，如果不相同则清空并要求重新输入，输入手机号后对手机号的格式进行了校验，如果格式正确会进行短信验证码的发送，否则提示发送验证码失败，输入正确的验证码后完成用户注册。



**图4 用户注册界面**

发送验证码所利用的是阿里云的短信功能，通过完成对阿里云短信功能的SDK的修改，导入自己的配置信息，即可完成短信的发送，主要代码为：

String accessKeyId=env.getProperty("accessKeyId");

String accessKeySecret=env.getProperty("accessKeySecret");

System.setProperty("sun.net.client.defaultConnectTimeout", "10000");

System.setProperty("sun.net.client.defaultReadTimeout", "10000");

IClientProfile profile = DefaultProfile.getProfile("cn-hangzhou", accessKeyId, accessKeySecret);

DefaultProfile.addEndpoint("cn-hangzhou", "cn-hangzhou", product, domain);

IAcsClient acsClient = new DefaultAcsClient(profile);

SendSmsRequest request = new SendSmsRequest();

request.setPhoneNumbers(mobile);

request.setSignName(sign\_name);

request.setTemplateCode(template\_code);

request.setTemplateParam(param);

request.setOutId("yourOutId");

SendSmsResponse sendSmsResponse = acsClient.getAcsResponse(request);

return sendSmsResponse;

发送的验证码由后台随机生成6位数，之后将生成的验证码存入Redis缓存中，调用发送功能将验证码发送给用户，当用户输入验证码后，从Redis中读取之前存入的数据和用户输入的数据进行比较，相同则用户注册成功。进行操作Redis时使用了SpringDataRedis，SpringDataRedis是Spring框架对Redis封装，简化了Redis的操作，使用时只需注入RedisTemplate即可快速方便的操作Redis，比较的主要代码为：

String redisCode = (String) redisTemplate.boundHashOps("smsCode").get(phone);

if (redisCode == null) {

return false;

}

if (!code.equals(redisCode)) {

return false;

}

return true;

## 用户信息修改及密码修改

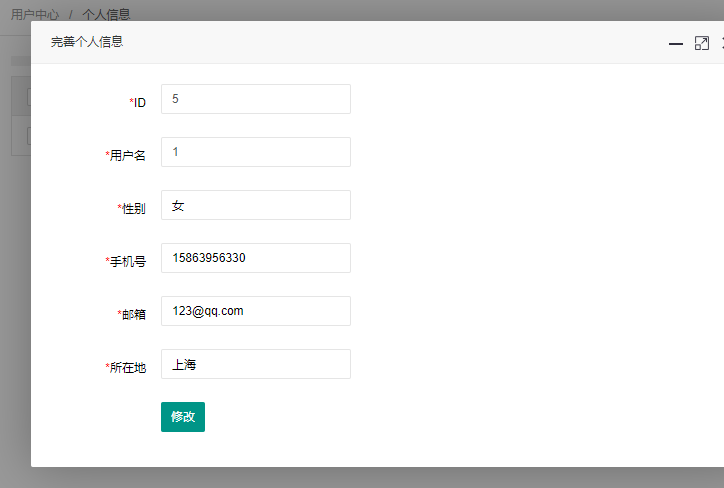
用户信息修改页面如图5所示，用户信息修改主要是对用户在注册时未填写的信息的补充，主要包括用户的性别、邮箱、所在地等信息的添加。用户信息的修改时查询已有信息是根据用户的用户名来进行查询的，因为系统使用了SpringSecurity框架，所以可以直接通过SpringSecurity的SecurityContextHolder.getContext().getAuthentication().getName()方法来进行用户名的获取。

密码修改页面如图6所示，进行密码的修改时，需要先输入原密码与数据库中存放的原密码进行比较，原密码无误后才能输入新的密码进行密码的修改操作，原密码的查询也是通过获取到用户名来进行查询。因为在用户注册时，对密码进行了MD5加密，所以在对输入的原密码进行校验的时，也需要对用户现在输入的密码进行加密，加密主要代码为：

userPo.setPassword(DigestUtils.md5DigestAsHex(passwordPo.getPassword().getBytes()));

userService.modifyPassword(userPo);

return new Result(true, "密码修改成功");



**图5 用户信息修改页面**



**图6 密码修改页面**

## 猪只管理模块的实现

猪只管理模块主要包括了对种猪、幼猪、肉猪和生产的管理。

种猪管理模块包括了主要包括了对种猪数据的录入，对录入信息的更新，对录入信息的删除以及批量删除，对已录入信息的条件查询等功能；录入信息时需要保证耳号的唯一性，否则会提示录入信息失败，主要代码为：

if (null != porkerPigService.selectByEarNum(porkerPigPo.getEarNumber())) {

return new Result(false, "添加失败");

} else {

porkerPigService.addPorkerPig(porkerPigPo);

return new Result(true, "添加成功");

}

幼猪管理模块也包括了对信息的增加、更新、查询和删除等功能。添加信息的时候，需要对种猪的耳号信息进行添加，该耳号必须存在，否则会提示添加失败。

肉猪管理的信息相对较少，不过也包含了基本的增加、更新、查询和删除操作。

生产管理主要是针对种猪的生产管理，包括对每个种猪的胎次、产仔数、流产数、存活数等信息的记录，可以详细的看出实际生产情况。

对数据交互类Po的封装均用到了lombok插件，只需要加入@Data、@AllArgsConstructor、@NoArgsConstructor这三个注解，即可省去构造方法、getter()、setter()等方法书写，让整个数据交互类变得简洁明了。所有的数据交互类都实现了Serializable，保证数据的序列化存储，主要代码如下：

@Data

@AllArgsConstructor

@NoArgsConstructor

public class BreedingPigPo implements Serializable {}

在实现查询功能时，分页查询模块的后端使用了Mybatis的PageHelper插件。因为使用了SpringBoot框架，所以需要在启动类中对该插件进行属性的配置，否则将会无法使用,主要配置代码如下：

@Bean

public PageHelper pageHelper() {

PageHelper pageHelper = new PageHelper();

Properties properties = new Properties();

properties.setProperty("offsetAsPageNum", "true");

properties.setProperty("rowBoundsWithCount", "true");

properties.setProperty("reasonable", "true");

properties.setProperty("dialect", "mysql");

pageHelper.setProperties(properties);

return pageHelper;

}

使用时仅需调用PageHelper.startPage()方法，传入当前页和页面大小即可实现分页功能。分页的查询结果通过PageResult进行封装，PageResult中包括了当前查询的总记录数（Long类型）和当前页的结果（List）类型，主要代码如下：

@Override

public PageResult findProduction(ProductionPo productionPo, int pageNum, int pageSize) {

PageHelper.startPage(pageNum, pageSize);

Page<ProductionPo> page = (Page<ProductionPo>) productionMapper.selectByExample(productionPo);

return new PageResult(page.getTotal(),page.getResult());

}

前端页面的分页模块使用了AngularJS的pagination插件，该插件通过配置paginationConf中的currentPage、totalItems、itemsPerPage、perPageOptions属性，即可完成对分页的实现，主要代码如下：

$scope.paginationConf = {

currentPage: 1,

totalItems: 10,

itemsPerPage: 10,

perPageOptions: [10, 20, 30, 40, 50],

onChange: function(){

$scope.reloadList();

}

};

前端调用后端接口除了返回基本类型的数据外，额外封装了一个Result结果类来进行数据的返回。该Result类包括了一个Boolean类型的result属性和一个String类型的message属性，前端通过获取这两个属性的值来完成判断和提示的操作，该结果类设计如下：

public class Result implements Serializable {

private boolean result;

private String message;

}

后端Controller层的接口统一使用了@ResponseBody注解，用来将返回给前端的结果转换成json格式；需要接收实体类的接口使用了@RequestBody属性，用来将前端传过来的对象自动转换成JavaBean对象，极大地简化了开发的难度，注解使用方式如下：

@RequestMapping("/searchByExample")

@ResponseBody

public PageResult query(@RequestBody ProductionPo productionPo, int page, int size) {

return productionService.findProduction(productionPo, page, size);

}

## 物品管理模块

物品管理模块主要包括物品明细功能，物品明细的页面如图7所示。用户可以通过这个功能记录日常的物品购买情况，消费支出，如图8所示。还可以通过对购买的物品通过类型、名称等进行查询，也可以对之前的物品进行删除和修改等操作。

其中删除和修改是根据这条记录的id来进行的。当点击修改操作时，会弹出新的页面，如图9所示，同时前端将这条记录的id传入修改页面，之后先根据传入的id进行这条记录的查询，将查询出的数据回显到页面中进行显示，用户输入新数据后，根据id进行数据的更新。删除操作也是根据用户id来进行的，其中删除包括单条记录的删除和批量删除。批量删除的实现是通过前端获取并记录用户所勾选的记录id，勾选的id添加到集合中，取消勾选后从该集合中删除此id，点击批量删除时将该集合传入后台，后台接收到这个集合后，在Service层对该集合进行遍历删除操作，前端获取用户所勾选的id代码为：

$scope.selectIds=[];//选中的id集合

//更新复选框

$scope.updateSelection=function ($event,id) {

if($event.target.checked){

$scope.selectIds.push(id);

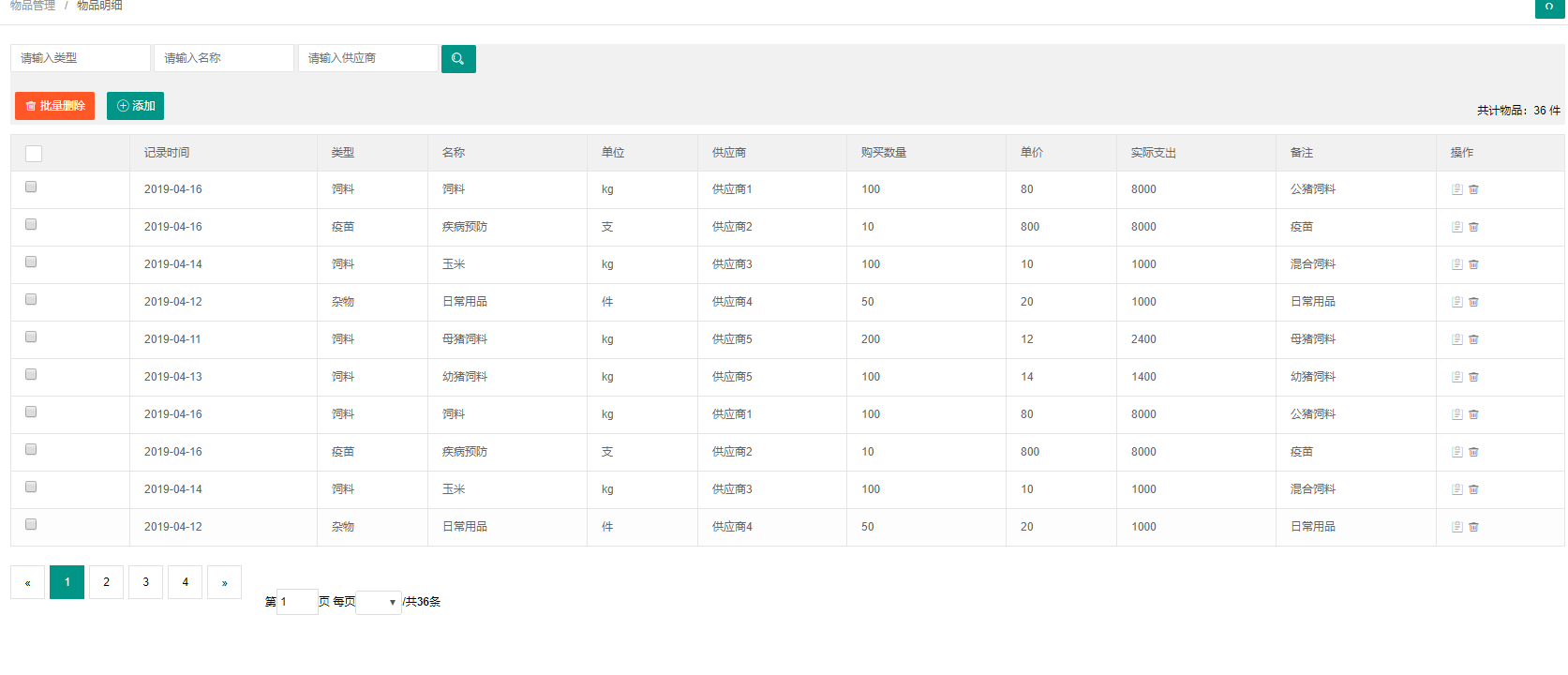
}else {

var idx = $scope.selectIds.indexOf(id);

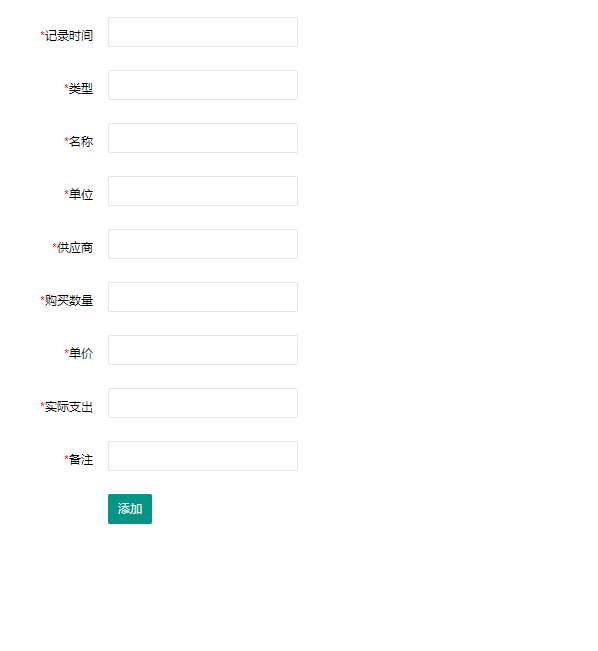
$scope.selectIds.splice(idx, 1);

}

};



**图7 物品明细页面**



**图8 物品添加页面**

![C:\Users\任超\AppData\Roaming\Tencent\Users\782912842\TIM\WinTemp\RichOle\7`49BS(1DGD7](@XW7J7M`R.png](data:image/png;base64,)

**图9 物品修改页面**

## 安全管理模块

安全管理模块主要是对生物安全的管理，记录猪只的体检情况、免疫情况、疫苗接种类型、消毒情况等数据，如图10所示，根据唯一标识耳号来进行查询和区分不同的猪只，完成对猪只生物安全的管理。

在添加数据时输入耳号后，如果鼠标离开输入框，会自动调用根据耳号查询的方法，返回Result结果集，如果该耳号数据库中已存在，会出现弹窗提示，如图11所示，不存在即可继续输入其他项，其中待输入项前有红色\*号标识的为必填项，如果不填将不允许数据的提交，如图12所示。添加已存在的耳号弹窗方法的Controller层代码为：

$scope.selectByEarNum=function (earNumber) {

bioSafetyService.selectByEarNum(earNumber).success(

function (response) {

if (response.result){

alert("该耳号已存在");

}

});

};

Service层代码为：

this.selectByEarNum=function (earNumber) {

return $http.post('../bioSafety/selectByEarNum?earNumber='+earNumber);

};

耳号弹窗的后端接口的代码为：

@RequestMapping("/selectByEarNum")

@ResponseBody

public Result selectByEarNum(String earNumber){

BioSafetyPo bioSafetyPo= bioSafetyService.selectByEarNum(earNumber);

if (bioSafetyPo != null) {

return new Result(true, "耳号已存在");

} else {

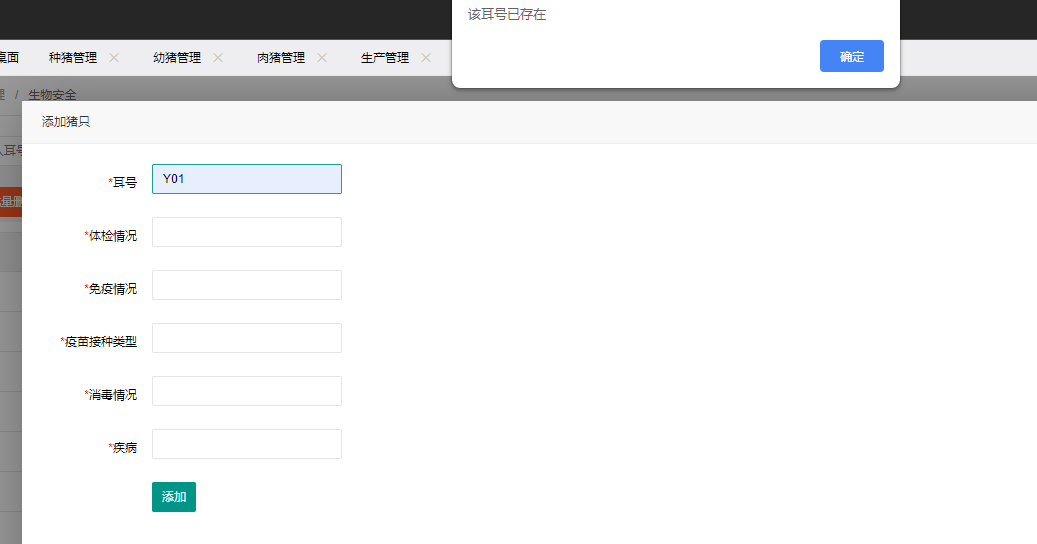
return new Result(false, "正常");

}

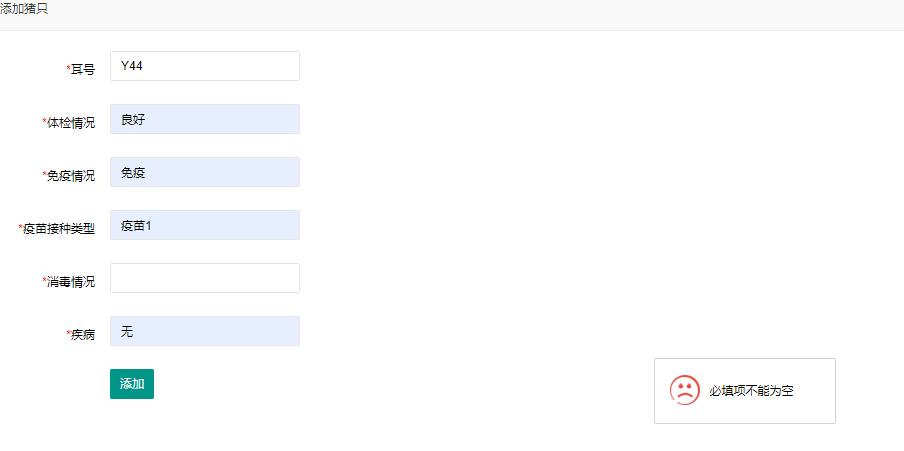
}



**图10 生物安全界面**



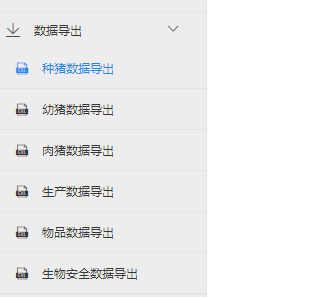
**图11 添加已存在耳号弹窗提示**



**图12 未添加必填项禁止数据提交**

## 数据导出模块

数据导出模块主要实现了将数据通过Excel表格导出，数据导出页面如图13所示，包括了对已存在的所有信息的导出。



**图13 数据报表导出的前端页面**

数据导出并未使用JAVA的插件在后台实现导出，而是将数据传入前端，在前端生成报表进行导出。因为前端使用了layui进行了页面的搭建设计，所以数据导出所使用的前端插件也是以Layui为基础设计的。layui\_exts插件，具有强大的SheetJS功能，依赖XLSX.js和FileSaver，可以出色的完成数据报表的导出。

在需要实现导出功能的页面调用Layui的方法手动加载Excel插件，代码如下：

<script type="text/javascript">

layui.config({

base: '../layui\_exts/'

}).extend({

excel: 'excel'

});

</script>

通过Ajax请求后端接口返回数据，后端依旧返回json格式的数据即可，请求代码如下：

$.ajax({

url: '../bioSafety/findAll'

,dataType: 'json'

,success: function(response) {}

通过执行梳理函数excel.filterExportData()，将后端给的数据顺序顺序和映射关系对应，通过执行data.unshift()函数，为导出的报表添加表头；最后调用excel.exportExcel()方法，传入导出报表的数据、报表的名称和报表的格式，即可完成报表的导出,主要实现代码为：

data = excel.filterExportData(data, {

earNumber: 'earNumber'

,physicalExamination: 'physicalExamination'

,immuneStatus: 'immuneStatus'

,vaccinationType: 'vaccinationType'

,disinfectionSituation: 'disinfectionSituation'

,diseaseCondition: 'diseaseCondition'

});

data.unshift({earNumber:'耳号',physicalExamination:'体检情况',immuneStatus:'免疫情况',

vaccinationType:'疫苗接种类型',disinfectionSituation:'消毒情况',diseaseCondition:'疾病'});

var timestart = Date.now();

excel.exportExcel({

sheet1: data

}, '生物安全数据.xlsx', 'xlsx');

## 溯源查询模块

溯源查询模块包括了幼猪溯源查询和种猪溯源查询，如图14所示，主要实现的是根据输入的耳号，从数据库中查询出所有和这个耳号相关的数据并返回。其中种猪溯源查询还会把对应的幼猪查询出来并显示。

将对应的AngularJS、 service和controller引入页面：

<script type="text/javascript" src="../js/angularjs/angular.min.js"></script>

<script src="../js/angularjs/pagination.js"></script>

<link rel="stylesheet" href="../js/angularjs/pagination.css">

<script type="text/javascript" src="../js/base.js"></script>

<script type="text/javascript" src="../js/service/sourceService.js"></script>

<script type="text/javascript" src="../js/controller/sourceController.js"></script>

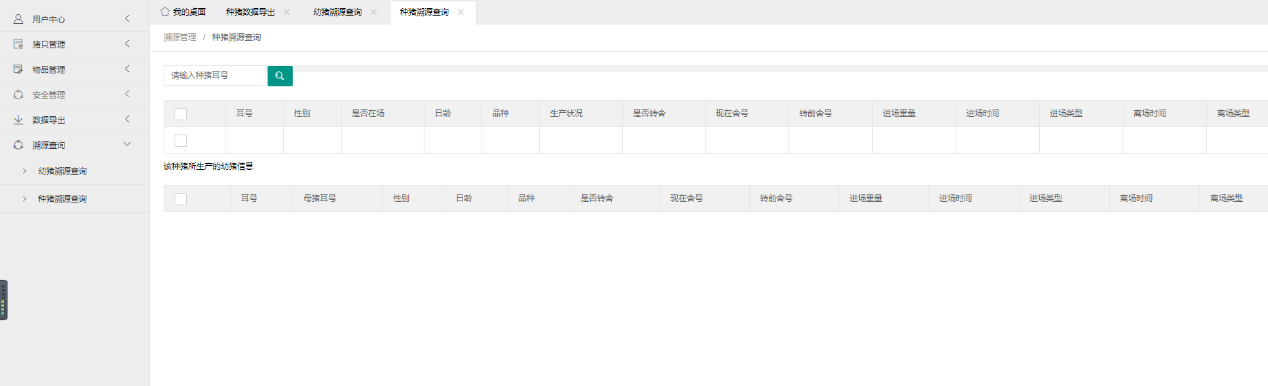
在body元素上添加ng-app和ng-controller引入服务：

<body ng-app="zcgj" ng-controller="sourceController" >

在点击按钮上添加ng-click()方法，调用controller中所写好的服务即可完成数据的查询，实现代码为：

<button class="layui-btn" lay-submit="" lay-filter="sreach" ng-click="selectBreeding(searchEntity.bearNumber)" ><i class="layui-icon">&#xe615;</i></button>

种猪溯源查询用时执行了两个方法来完成种猪和幼猪信息的查询，最后将返回的结果集映射绑定在对应的值上，页面就可以将数据显示出来，如图15所示：



**图14 溯源查询的前端页面**



**图15 页面查询出数据的显示**

# 牲畜养殖管理系统功能测试

一个软件项目的开发离不开测试环节，每完成一个功能模块，都需要对其进行详细的测试，找到该模块中存在的问题，对产生的问题进行排查，找到出现问题的代码对其进行修正，之后再次测试是否存在别的问题，保证项目完成时各个功能模块可以正常运行。

测试时所使用的测试工具为Postman软件，Postman是接口调试的利器，之前可以通过Chrome浏览器下载插件进行使用，后来单独形成了一个软件可以下载安装使用，Postman可以用来测试不同类型的接口请求方式，包括Get、Post、Put等常见请求方式，还可以在请求接口时其添加任意的参数进行测试，用起来十分方便。

在进行接口测试时，主要使用了黑盒测试中的等价类划分法【7】。等价类划分法的主要思想是软件系统的相关数据分成多个部分，也就是把软件数据划分为多个子集，然后在这多个数据子集中选取一定具有代表性的少量数据进行测试，省去将软件的所有数据进行测试【8】，下面是针对用户注册模块中输入手机号所进行的测试：

手机号录入，规定号码必须以（13,17,147,15(后一位除4的任意数),18(后一位除1和4的任意数)开头）。

划分等价类：1.数字，2.字符长度为11，3.以规定字段开头

**表8 等价类表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 有效等价类 | 编号 | 无效等价类 | 编号 |
|  | 数字 | 1 | 11位非纯数字 | 8 |
|  | 字符串长度为11 | 2 | 字符长度为0 | 9 |
| 字符长度>11 | 10 |
| 0<字符长度<11 | 11 |
| 输入条件 | 13开头11位数 | 3 |  |  |
|  | 17开头11位数 | 4 | 所有不是以13,17,147, 15（后一位除4）, 18（后一位除1和4）开头的11位数 | 12 |
|  | 147开头11位数 | 5 |  |  |
|  | 15（后一位除4）开头11位数 | 6 |  |  |
|  | 18（后一位除1和4）开头11位数 | 7 |  |  |

**表9 测试用例设计**

|  |  |
| --- | --- |
| 覆盖有效等价类的测试用例 | 编号 |
| 13412564596 | 1,2,3 |
| 17375693624 | 1,2,4 |
| 14745263695 | 1,2,5 |
| 15993622635 | 1,2,6 |
| 18236522369 | 1,2,7 |
|  |  |
| 覆盖无效等价类的测试用例 |  |
| 1aa523623695 | 8 |
| 无输入 | 9 |
| 159938556932 | 10 |
| 1596932236 | 11 |
| 15423699632 | 12 |
| 18136955623 | 12 |
| 18423699685 | 12 |
| 19326355874 | 12 |

**表10 测试结果**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试模块：手机号的输入 | | | | |
| 测试用例 | 预期结果 | 实际结果 | 测试结果 |
| 1,2,3 | 正常 | 正常 | 成功 |
| 1,2,4 | 正常 | 正常 | 成功 |
| 1,2,5 | 正常 | 正常 | 成功 |
| 1,2,6 | 正常 | 正常 | 成功 |
| 1,2,7 | 正常 | 正常 | 成功 |
| 8 | 提示有误 | 提示有误 | 成功 |
| 9 | 提示有误 | 提示有误 | 成功 |
| 10 | 提示有误 | 提示有误 | 成功 |
| 11 | 提示有误 | 提示有误 | 成功 |
| 12 | 提示有误 | 提示有误 | 成功 |

以上便是测试注册模块中对输入手机号格式的测试，可以看出等价类划分法测试是比较简单的但是也是比较全面的一种测试方法。

# 总结

通过这次系统的开发，使我对软件开发的流程有了更加清晰全面的认识，同时也丰富了自己的知识体系，增强了动手实践的能力，使自己的编程习惯和编程经验得到了不小的提升。从前端到后端再到数据库，让我对软件工程的结构体系有了更加深刻的认知。在开发的过程中，自己学习到了前后端分离的技术架构，学习到了SpringBoot为软件开发所带来的便利，学习并了解了许多优秀的前端框架和开发工具，让自己的知识水平获得了进一步的提升。

项目中所使用的一些技术和方法都是自己在网上或者实习过程中所学习到的，完成这个项目后，让我对这些知识有了更加深刻的了解，认识到了什么事都要自己手动去实现一次，才是真的会用了，才能将这些知识真的变成自己的东西。这种学习方式不仅可以提高自己的编程效率，同时也可以提高自己对新技术的学习。通过这次开发让我认识到了目前越来越多的行业正在向“互联网+”转型，我身为一名软件工程专业的学生，身为一名软件开发者，正处于这个时代的浪潮，而自己要想在这个时代有所作为，就要继续去学习更多的知识，通过自己的努力，让自己变得更加优秀，让自己对社会的进步尽一份微薄之力。

# 致谢

转眼间大学的时光就到了尾声，而我们也该要对自己的学生时代说一句再见了。大学是一个令人敬仰的地方，大学是一个学习知识的圣地，在这四年中，我在同学和老师的帮助下学习到了许多的专业知识，同时自学了许多的其他的知识，开拓了自己的视野，丰富了自己的知识储备。从大一开始，苏静老师就邀请我和其他同学一起加入了学习团队，而正是因为这个学习团队，让我的大学生活变得充满目标，让我从中学习到了很多很多的知识，为以后的实习奠定了扎实的基础。苏静老师在百忙之中抽空对我们团队的指导，也让我们受益匪浅。

由衷的感谢我的毕业论文指导老师苏静老师，苏静老师负责任的督促和定期的检查，让我的毕业设计有了保障，对我的指导也让我对毕业设计有了更加全面的认识，对于完成我的毕业设计给予了很大的帮助。

感谢大学里的每一位指导过我的老师，感谢你们的悉心指导，希望在以后的日子里，老师们的工作生活一帆风顺，而我也会在接下来的生活中继续努力，为了自己的梦想奋斗，体验精彩的人生。

# **参考文献**

1. 王丹,孙晓宇,杨路斌,高胜严.基于SpringBoot的软件统计分析系统设计与实现[J].软件工程,2019,22(03):40-42.
2. 刘艳.基于AngularJS的农业信息搜索系统设计与实现[J].无线互联科技,2018,15(19):31-32.
3. 彭程,高晶.Java与MySQL环境下图书馆信息化管理系统设计与实现[J].电脑知识与技术,2019(08):59-60.
4. 阳小兰,罗明.基于Spring+SpringMVC+MyBatis网上论坛的设计与实现[J].黑龙江科技信息,2016(36):279-280.
5. 曹灿,刘志刚.基于SSH和Layui的工程科学前沿与实践系统[J].工业控制计算机,2019,32(02):91-92+96.
6. 蒋丛萃,史卓.Spring security在电子商务中的设计应用[J].电脑知识与技术,2016,12(14):247-248.
7. 张新,焦豪.两种黑盒测试方法的比较分析[J].电子技术与软件工程,2018(07):61-63.
8. 钟睿.浅析软件黑盒测试[J].数字通信世界,2018(05):145.

**Design and Implementation of Livestock Breeding Management System**

REN Chao

(School of Software Engineering, Anyang Normal University, Anyang, Henan 455000)

Abstract：In order to make the livestock industry better integrated with the Internet and enjoy the convenience brought by the Internet, a livestock management system was designed.The Livestock Management System is designed and developed on the basis of the JAVA platform, which allows farmers to manage farms more easily.When designing the system, the system architecture is designed with the current popular B/S architecture. The entire project is built using SpringBoot. The back end is completed through the SSM framework. The front end is mainly completed through the AngularJS framework.The management system is mainly for the pig industry, and has realized the functions of user login registration, management of breeding pigs, young pigs, pigs and production, management of items, biosafety, and export of statements.The management system allows farmers to better manage their farms, reduce unnecessary losses and expenses, and increase profits.

Keywords： SpringBoot；AngularJS；SpringSecurity; Layui