报价管理系统的设计与实现

摘 要

改革开放以来,随着互联网的兴起和普及,全球的电子商务得到了迅速的发展。利用最新的网络编程技术构建高效可靠的电子商务方案已经成为了当前的热点问题。在互联网开放的网络环境下,以电子交易方式进行交易活动和相关服务活动,实现消费者的网上购物、商户之间的网上交易和在线电子支付是一种新型的商业运营模式。进入21世纪.电子商务吸引了众多企业的加入,其主要原因就在于基于计算机网络的电子商务具有非常明显的优势。本文涉及的是报价管理系统,本系统是基于流行的J2EE架构,遵循MVC框架,综合利用ssm框架+mysql实现。本文描述的是报价管理系统的设计和实现过程,主要包括系统需求分析、系统概要设计、系统详细设计、系统实现与测试四个部分。系统需求分析部分讲述了报价管理系统的用例场景。其中以主要用例为例,展开用例场景描述,将系统的需求更详细的展现出来。系统概要设计阐述了系统包括的主要模块,各模块的主要功能和模块之间的关联。从功能方面更直观的介绍业务流程。系统详细设计采用UML将对该系统中的各模块进行全面的分析,讲解设计的细节,同时详细设计部分还介绍了数据库的建立和对数据库的操作逻辑。系统实现部分对在开发过程中遇到的难题的解决方法进行说明并展示相应的代码实现。

关键词：电子商务，报价管理，JavaEE，SSM框架，MySQL数据库

DESIGN AND IMPLEMENTATION OF QUOTATION MANAGEMENT SYSTEM

ABSTRACT

Since the reform and opening up, with the rise and popularization of the Internet, the global e-commerce has developed rapidly. Using the latest network programming technology to build efficient and reliable e-commerce solutions has become a hot issue. In the open network environment of the Internet, it is a new business operation mode to carry out transaction activities and related service activities by means of electronic transactions, to realize online shopping of consumers, online transactions among merchants and online electronic payment. The online shopping system has a powerful interactive function, which makes it convenient for businesses and users to transfer information and complete electronic trade. In the 21st century, e-commerce has attracted many enterprises, the main reason is that e-commerce based on computer network has very obvious advantages. This paper deals with the quotation management system, which is based on the popular J2EE architecture, follows the MVC framework, and uses the SSM framework + Mysql to realize. This paper describes the design and implementation process of quotation management system, including four parts: system requirements analysis, system outline design, system detailed design, system implementation and test. The part of system requirement analysis describes the use case scenario of quotation management system. Take the main use case as an example, expand the use case scenario description, and show the system requirements in more detail. System outline design describes the main modules of the system, the main functions of each module and the relationship between modules. Introduce the business process more intuitively from the function aspect. The detailed design of the system uses UML to analyze all the modules of the system and explain the details of the design. At the same time, the detailed design part also introduces the establishment of the database and the operation logic of the database. In the system implementation part, the solutions to the problems encountered in the development process are explained and the corresponding code implementation is shown.

**KEY WORDS**: Electronic Commerce，Quotation Management，JavaEE， SSM Framework，MySQL DataBase

目　录

[第1章 绪论 1](#_Toc25214)

[§1.1 研究课题的目的与背景 1](#_Toc11012)

[§1.2 国内外同类设计的概况 1](#_Toc15731)

[§1.3 本文主要研究的内容 2](#_Toc4243)

[第2章 系统分析 3](#_Toc19610)

[§2.1 需求分析 3](#_Toc14911)

[§2.2 可行性研究 3](#_Toc1622)

[第3章 系统设计 4](#_Toc13424)

[§3.1 系统总体结构与功能 4](#_Toc16051)

[§3.1.1 系统的设计目标 4](#_Toc16164)

[§3.1.2 系统的模块设计 4](#_Toc2391)

[§3.2 系统流程图 5](#_Toc3227)

[§3.3 数据库设计 6](#_Toc20457)

[§3.3.1 数据库设计 6](#_Toc1698)

[§3.3.2 数据库表设计 6](#_Toc26810)

[第4章 系统详细设计 11](#_Toc22297)

[§4.1 系统各模块功能设计与简介 11](#_Toc6071)

[§4.1.1 用户登录注册 11](#_Toc5530)

[§4.1.2 客户信息录入 12](#_Toc29677)

[§4.1.3 客户信息修改 13](#_Toc19311)

[§4.1.4 客户信息查看 14](#_Toc15910)

[§4.1.5 产品类型管理 15](#_Toc18496)

[§4.1.6 产品信息录入 15](#_Toc22881)

[§4.1.7 产品信息修改 16](#_Toc21789)

[§4.1.8 产品信息查看 17](#_Toc23616)

[§4.1.9 订单信息录入 18](#_Toc197)

[§4.1.10 订单信息查看 18](#_Toc10905)

[§4.1.11 报价信息录入 19](#_Toc22376)

[§4.1.12 报价信息查看 19](#_Toc16584)

[第5章 系统测试 21](#_Toc10547)

[§5.1 系统的功能测试 21](#_Toc26241)

[§5.1.1 测试登录注册功能 21](#_Toc24156)

[§5.1.2 测试客户录入功能 22](#_Toc27944)

[§5.1.3 测试客户信息修改功能 23](#_Toc3224)

[§5.1.4 测试客户信息查看功能 23](#_Toc32256)

[§5.1.5 测试删除客户功能 24](#_Toc24850)

[§5.1.6 测试产品类型录入 25](#_Toc24235)

[§5.1.7 测试产品信息录入 25](#_Toc19302)

[§5.1.8 测试产品信息查看 26](#_Toc1744)

[§5.1.9 测试订单信息录入 26](#_Toc31023)

[§5.1.10 测试订单信息查看 27](#_Toc30700)

[§5.1.11 测试报价信息录入 27](#_Toc28442)

[§5.1.12 测试报价信息查看 28](#_Toc4238)

[§5.2 系统测试结论 28](#_Toc13229)

[第6章 28](#_Toc7823)

[结 论 29](#_Toc24860)

[参考文献 30](#_Toc20493)

[致　谢 32](#_Toc24286)

[附　录 33](#_Toc24878)

# 绪论

## 研究课题的目的与背景

随着互联网技术应用的不断深入，传统的销售方式（如陌生拜访、电话销售、信件销售、传真销售等）因为成本、时间等多种阻碍因素，已经不能够适应当今激烈的商业竞争环境，店面销售、柜台销售、会议营销等多种销售方式也因为地域差异大，成本高等原因不便于企业扩大发展。因此产品的生产与销售链条也越来越多的融入了电子商务的元素。消费者通过网络能与来自世界各地、不同种类的产品建立直观的联系，并通过便捷、快速的操作完成购买流程。为了便于用户在千万计的产品中通过对不同制造商、不同产地的产品的价格、性能等参数进行比较，快速的找到自己需要的产品，尽可能缩短购买时间,并在制造商和消费者之间建立一个高效的平台。产品报价系统根据以上需求，产品管理、用户管理、订单管理、报价等后台模块，通过科学全面的测试，系统要实现产品报价和在线购物的主要功能是急不可待的。产品报价系统做为现有电子商务和在线购物模式的重要补充，在实际的应用中必将显现其突出的优势。

## 国内外同类设计的概况

随着互联网的不断发展，竞价推广已经普及了大多数的企业，市场竞争也是越来越激烈，企业面对竞争如此激烈的市场。传统的报价已经不足以满足对报价的精准定位。企业如何给客户一个准确的报价, -般有几种报价方式:成本导向定价法是企业定价首先需要考虑的方法。成本是企业生产经营过程中所发生的实际耗费,客观上要求通过商品的销售而得到补偿,且要获得大于其支出的收入,超出的部分表现为企业利润。以产品单位成本为基本依据,再加上预期利润来确定价格的成本导向定价法，是中外企业最常用、最基本的定价方法。需求导向定价是指按照顾客对商品的认知和需求程度制定价格,而不是根据卖方的成本定价。这类定价方法的出发点是顾客需求,认为企业生产产品就是为了满足顾客的需要,所以产品的价格应以顾客对商品价值的理解为依据来制定。还有一种是根据客户对公司产品的价值认知确定价格。

由此可观，虽然只是一个简单的报价,但是过程是比较复杂的,对于一个企业报价订单管理系统也是极其重要的。有的企业往往由于公司订单量和客户蚍较多,在跟进客户的时候没有办法及时立刻知道每个客户的一个报价情况，以吸在下次合作是,也难免报错价格,不是很清楚上次与同一个客户的成交价格。那通过线下的口头报价,到与客户成交的周期比较长,没有办法立即反应当时的一个情况,如果企业及时用了报价订单管理系统的话就会很好的解决这些问题。

## 本文主要研究的内容

本网站采用的编写语言是Java语言，使用的数据库为MySQL，SSM框架开发而成，本系统的主要功能有：

客户管理模块：该模块主要用来管理客户信息，包括客户的名称、联系电话、联系地址、联系人等信息。

产品管理模块：该模块主要是用来管理产品类别和产品信息，包括产品的名称、产品的单位、产品的价格等信息。

订单管理模块：该模块主要用来管理订单信息，包括下单客户名称，产品名称、产品数量、订单金额等信息。

报价管理模块：该模块主要用来管理报价信息，包括客户名称、产品名称、报价人、报价时间等信息。

# 系统分析

## 需求分析

对于销售公司来说，最大的需求就是能够对公司的各种信息进行管理。

首先因为公司的产品种类很多，所有需要对产品的种类进行管理。当公司推出一个系列的产品后，后台管理人员可以根据需要录入新的产品类别。对于客户而言，他关注的就是公司有哪些产品类别，所有客户需要可以对公司的产品类别进行查询。

同样公司还需要能够对产品进行管理操作，包括后台管理人员的录入新的产品，更新产品，删除产品以及查询产品。但是对于客户而言，只能够查询公司的产品。公司还需要对客户进行管理，包括客户录入、客户更新、客户删除以及根据查询条件查询满足条件的所有客户。对于报价和订=订单同样需要进行管理，其功能与客户基本相似。

## 可行性研究

首先，在社会可行性方面，对于一家销售公司，最重要的就是卖出产品。在卖出产品前，有一个最重要的环节，就是报价，客户往往会选择一个报价较低的公司来进货。通过该系统可以管理公司的产品、客户、订单以及报价，从而提高销售公司的竞争力以及销售能力。

其次，在技术可行性方面，为了提高开发效率，本网站采用SSM框架技术，在本科学习期间已经基本掌握SSM框架以及JavaEE的开发模式，使用Maven进行工程所需依赖包的管理，使用Mysql数据库保存系统数据。

最后，在开发环境可行性发面，本次采用的是window10作为开发系统。开发工具有Eclipse、WebStream、Navicat、NotePad++等。

# 系统设计

## 系统总体结构与功能

### 系统的设计目标

经过调查与研究，预期的报价管理系统应达到以下目标：

（1） 实现登录注册功能。

（2）系统能够稳定运行的同时也要安全可靠。

（3）报价管理系统是一个公共平台，对于管理员，可以对系统的所有信息进行管理。对于普通员工，可以录入和查询信息。对于客户，仅仅只能浏览查询信息。

### 系统的模块设计

用户登录注册：登录成功以后会根据权限的不同展示不同的菜单导航；客户管理模块：该模块只有管理员和操作员可以看到，并且操作员只有录入新客户、修改客户信息的功能；产品管理模块：该模块三个权限的用户都可以看到，但是普通用户只能根据查询条件浏览产品信息；订单管理模块：该模块也只有管理员和操作员可以看到，同样操作员只有录入新订单、修改订单信息的功能；报价管理模块：该模块三个权限的用户都可以看到，但是普通用户只能浏览报价信息。

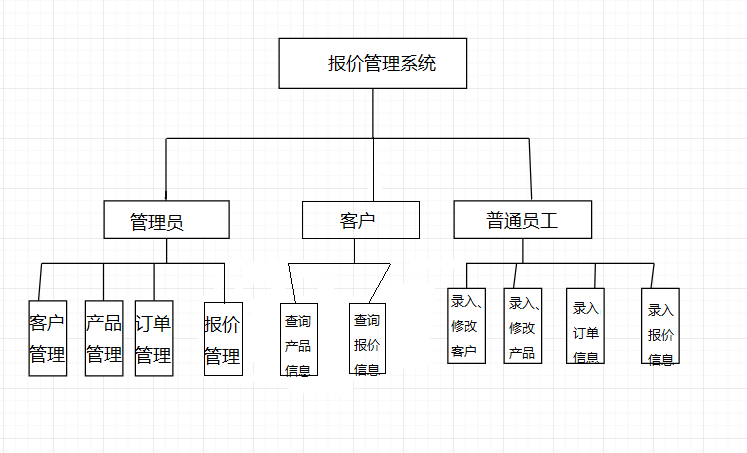


图3-1 模块总体结构图

## 系统流程图

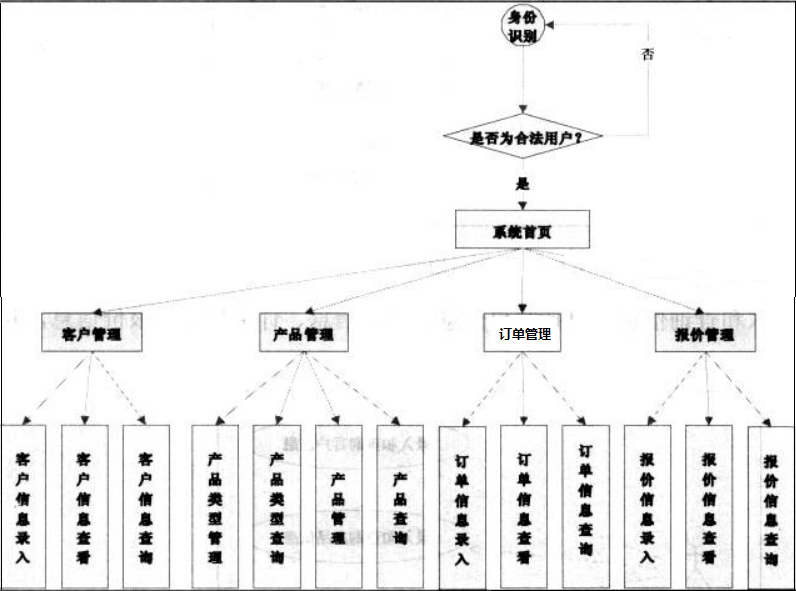


图3-2 系统流程图

## 数据库设计

### 数据库设计

数据库的设计是在遵循三大范式的基础上进行的，因此表结构设计合理且有相对较少的冗余。在报价管理系统中，需要对产品信息、产品分类信息、客户信息、报价信息、订单信息和用户信息进行管理，所有需要对这些信息创建相应的数据库表，即主表，分别是客户信息表、产品类型信息表、产品信息表、订单信息表、报价信息表以及用于存储登录用户的用户信息表。但是有的表可能会有关联的副表，比如：客户信息就需要为客户设置客户所属的行业、客户地址表（存储省市县）、订单信息会涉及到产品单位的产品单位信息表，所以系统有区域信息表、行业信息表、产品单位信息表。具体实体关系图如图3-3所示。

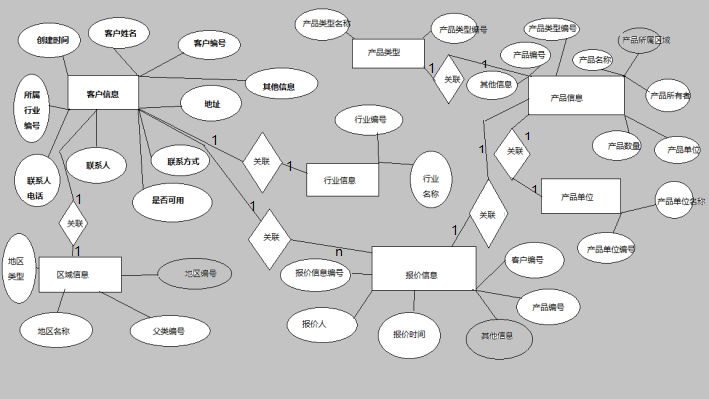


图3-3 E-R图

### 数据库表设计

根据以上分析，系统共需使用9张表，以下是本系统设计中所需要的主要的表的设计。

用户表用来存储用户的相关信息也可以用来验证登录信息是否正确，详情见表3-1。

表3-1 用户表

|  |  |  |
| --- | --- | --- |
| tb\_user | | |
| 字段名 | 类型 | 说明 |
| username | varchar(50) | 登录名 |
| password | varchar(100) | 登录密码 |
| email | varchar(50) | 邮箱 |
| grade | varchar(6) | 权限等级 |
| createDate | datetime | 创建时间 |
| address | varchar(255) | 用户地址 |

客户信息表是用来存储客户的相关信息的，详情见表3-2。

表3-2 客户信息表

|  |  |  |
| --- | --- | --- |
| tb\_customer | | |
| 字段名 | 类型 | 说明 |
| customerNo | varchar(30) | 客户编号 |
| customerName | varchar(50) | 客户名称 |
| address | varchar(200) | 地址 |
| phone | varchar(11) | 联系电话 |
| relationman | varchar（50） | 联系人 |
| relationmanPhone | varchar(11) | 联系人电话 |
| industryId | varchar(20) | 所属行业Id |
| createDate | datetime | 客户创建时间 |
| state | int | 是否可用 1 可用 0 不可用 |
| otherInfo | varchar | 其他信息 |

区域信息表用来存储客户所在省市县信息的，详情见表3-3。

表3-3 区域信息表

|  |  |  |
| --- | --- | --- |
| tb\_area | | |
| 字段名 | 类型 | 说明 |
| areaId | varchar(20) | 地区Id |
| parentId | varchar(20) | 父类Id |
| areaName | varchar(30) | 地区名称 |
| areaType | varchar(10) | 地区类型 |

行业信息表用来存储客户所属行业的，详情见表3-4。

表3-4 行业信息表

|  |  |  |
| --- | --- | --- |
| tb\_industry | | |
| 字段名 | 类型 | 说明 |
| industryId | varchar(20) | 行业Id |
| industryName | varchar(100) | 行业名称 |

产品类型信息表，用来储存产品类型信息，详情见表3-5。

表3-5 产品类型信息表

|  |  |  |
| --- | --- | --- |
| tb\_producttype | | |
| 字段名 | 类型 | 说明 |
| producttypeNo | varchar(10) | 产品类型编号 |
| producttypeName | varchar(50) | 产品类型名称 |

产品表是用于存储产品信息与产品类型信息表关联，详情见表3-6。

表3-6 产品信息表

|  |  |  |
| --- | --- | --- |
| tb\_product | | |
| 字段名 | 类型 | 说明 |
| productNo | varchar(30) | 产品编号 |
| producttypeNo | varchar(10) | 产品所属类型编号 |
| productName | varchar(50) | 产品名称 |
| productArea | varchar(100) | 产品所在区域 |
| productOwner | varchar(50) | 产品所有者 |
| unit | varchar(50) | 产品单位 |
| price | decimal(10,2) | 产品价格 |
| quantity | int | 产品数量 |
| otherInfo | varchar(500) | 产品其他信息 |

产品单位表用来存储产品单位信息，详情见表3-7。

表3-7 产品单位表

|  |  |  |
| --- | --- | --- |
| tb\_unit | | |
| 字段名 | 类型 | 说明 |
| unitId | int | 单位Id |
| unitName | varchar(255) | 单位名称 |

订单信息表用来存储订单信息，详情见表3-8。

表3-8 订单信息表

|  |  |  |
| --- | --- | --- |
| tb\_order | | |
| 字段名 | 类型 | 说明 |
| orderNo | varchar(30) | 订单编号 |
| customerNo | varchar(30) | 下单客户编号 |
| productNo | varchar(30) | 产品编号 |
| productCount | decimal(10,2) | 产品数量 |
| orderAmount | decimal(10,2) | 订单金额 |
| orderTime | datetime | 下单时间 |
| otherInfo | varchar(500) | 订单其他信息 |

报价信息表用来联系用户与收货地址的，详情见表3-9。

表3-9 报价信息表

|  |  |  |
| --- | --- | --- |
| tb\_quotation | | |
| 字段名 | 类型 | 说明 |
| quotationNo | bigint(30) | 报价编号 |
| quotationMan | varchar(20) | 报价人 |
| quotationTime | datetime | 报价时间 |
| otherInfo | varchar(500) | 报价其他信息 |
| customerNo | varchar(30) | 客户编号 |
| productNo | varchar(30) | 产品编号 |

# 系统详细设计

## 系统各模块功能设计与简介

### 用户登录注册

在地址栏输入localhost:8088/qms会默认打开登录页面。如图4-1

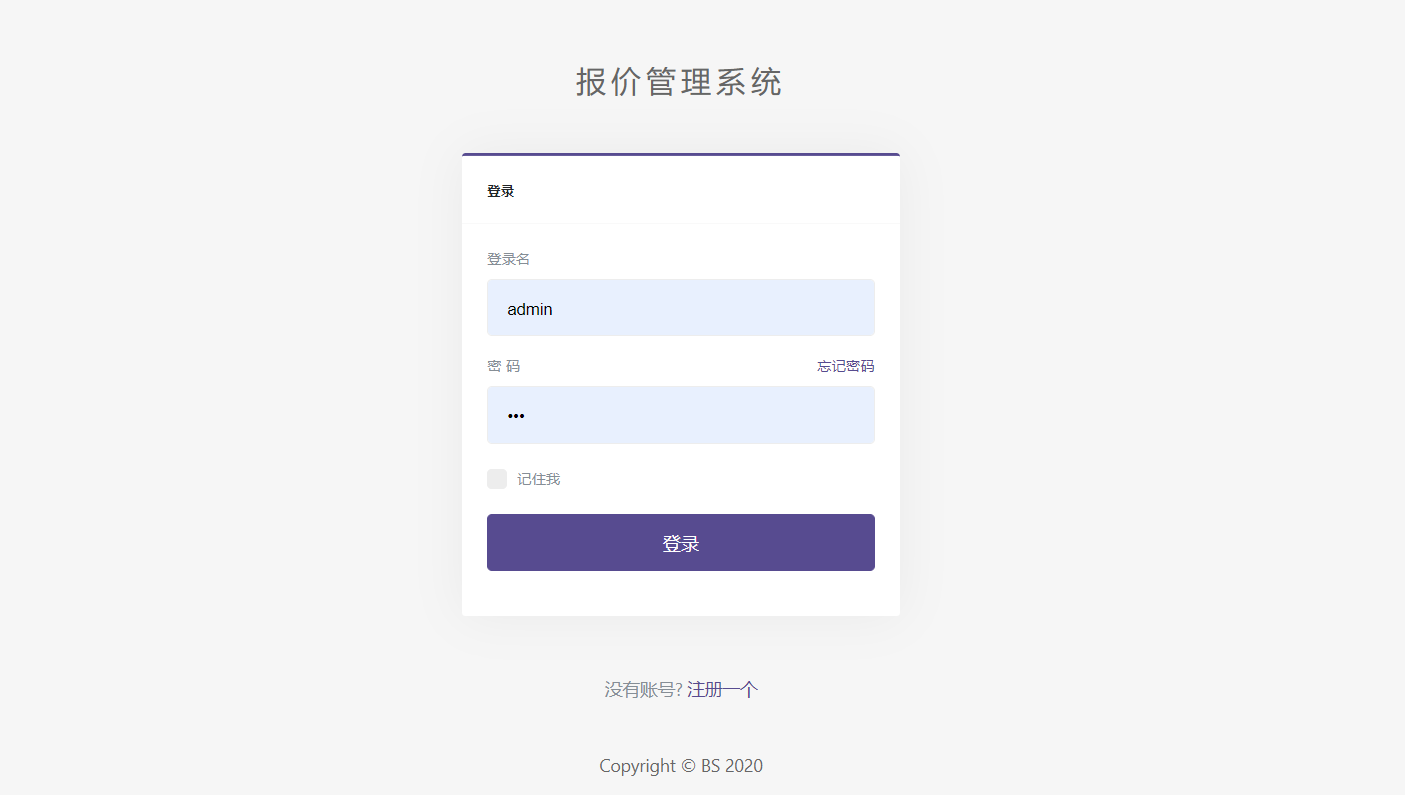


图4-1系统登录界面

用户输入登录名和密码，系统就会在数据库中检索相应的用户信息，如果登录名和密码都正确即登录成功，进入报价管理系统主页。若不成功则页面刷新，需要重新输入正确的登录名和密码。如果没有账号的话，我们可以点击”注册一个“按钮跳转至用户注册页面。如图4-2

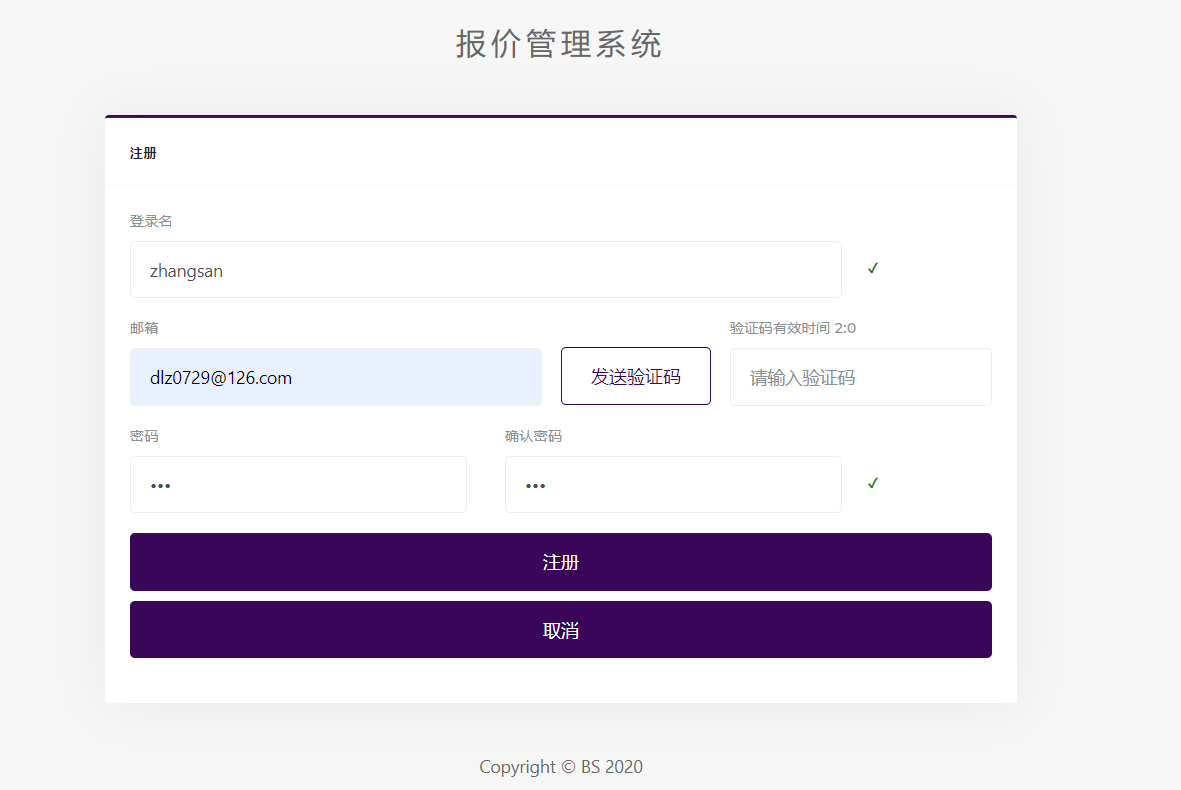


图4-2系统注册界面

当输入的登录名经过校验未被注册过，则在登录名文本框后方提示“√”，如果输入的登录名是注册过的，则在登录名文本框后方提示“登录名已存在”。如果输入的邮箱合法，就可以点击“发送验证码”按钮给输入的邮箱发送6位纯数字的验证码，验证码的有时间为两分钟，“请输入验证码”文本框上方会出现两分钟的倒计时，如果密码与确认密码一致，则在确认密码文本框后方提示“√”，验证码输入正确并且两次密码一致，点击注册按钮即可完成注册。如果期间超过两分钟后点击注册会提示验证码已失效，可以重新发送验证码。如果输入的验证码不正确，会提示验证码不正确。当完成注册之后，页面跳转至登录页面。期间如果我们不需要注册了，点击“取消”按钮，页面即可定向跳转至登录页面。

### 客户信息录入

该功能只有管理员和操作员可以看到，当点击二级菜单“客户信息录入”，页面跳转至客户信息录入界面。如图4-3

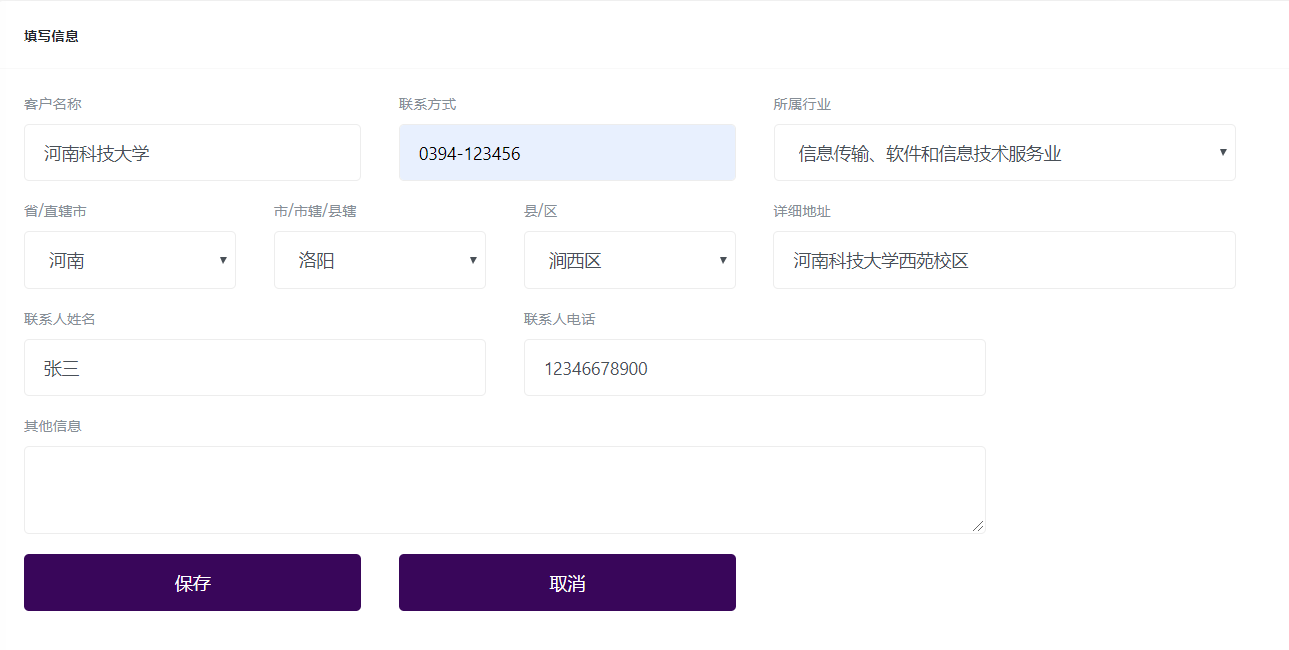


图4-3客户信息录入界面

页面中所属行业下拉框是页面加载时，查询行业信息表（tb\_industry）后绑定的，我们可用为客户选择对应的行业。页面中的省/直辖市，市/市辖/县辖，县/区是一个省市县三联动的下拉框。同样，页面加载的时候会查询区域信息表（tb\_area），优先绑定省/直辖市，当我们选择一个省/直辖市，触发事件，查询该省/直辖市下的市/市辖/县辖，县/区同理。当输入框的值都录入完毕之后，点击“保存”，就会将客户信息保存到客户信息表（tb\_customer）中。

保存成功之后页面会刷新，表单内容被清空，页面不跳转。如果点击“取消”，页面跳转至主页。

### 客户信息修改

该功能只有管理员和操作员可用看到，操作员只能通过二级菜单“客户信息修改”跳转至客户信息修改页面，管理员不仅能通过二级菜单，还能在客户信息查看页面，选中某一条数据，点击编辑按钮，这条数据会被带到客户信息修改页面对应的文本框中。如图4-4

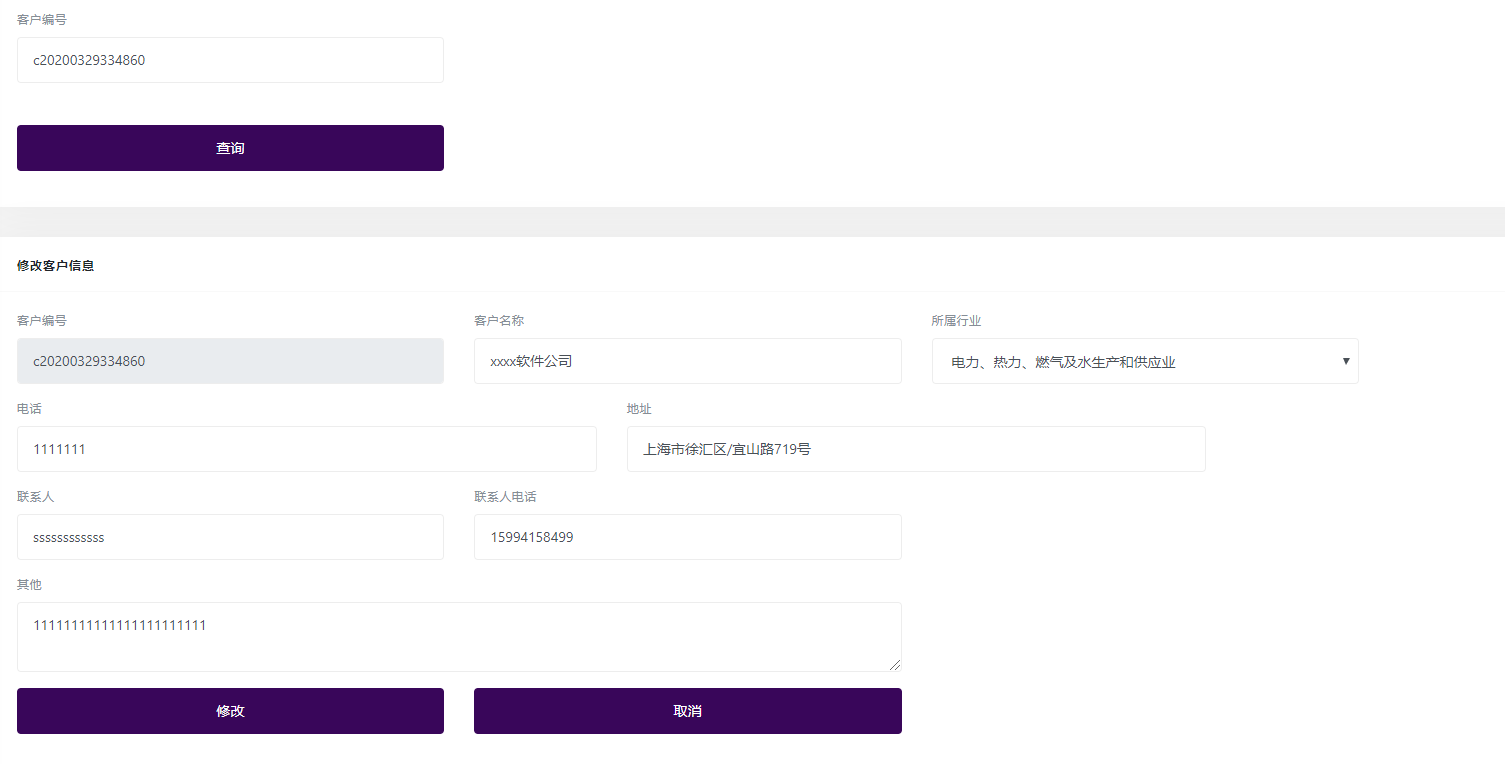


图4-4客户信息修改界面

如果是直接打开客户信息修改页面，页面中不会有数据，需要手动输入客户编号进行查询，查询结果会绑定到对应的输入框。需要修改的内容输入好之后，点击修改，即可修改客户的信息。最后，点击取消按钮，如果权限是管理员跳转至客户信息查看页面，如果权限是操作员跳转至主页。

### 客户信息查看

该功能只有管理员才可以看到，当点击二级菜单“客户信息查看”，页面跳转至客户信息查看页面。如图4-5

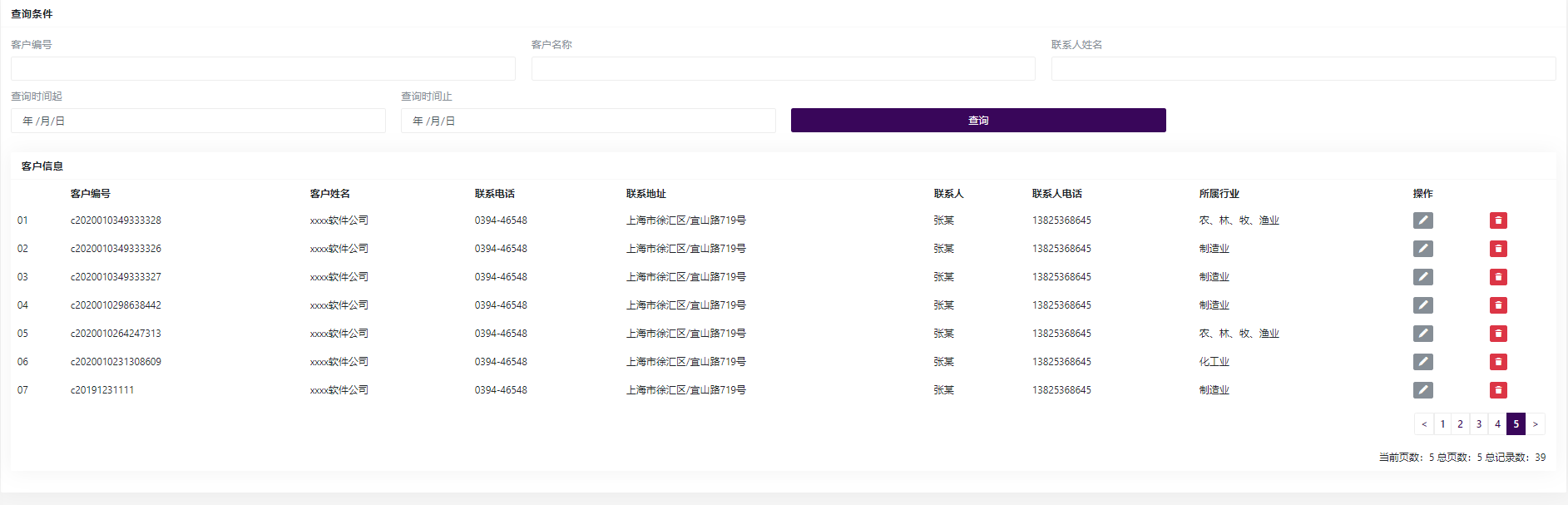


图4-5客户信息查看界面

页面采用数字分页的方式对客户信息进行展示，默认每页8条数据。可以点击上一页、下一页或者指定页数，查看当前页的数据。如果当前页已经是第一页或者已经是最后一页了，就不能再往前或者往后翻页。表头上面是五个可用用于模糊查询的参数，分别是根据客户编号唯一查询，根据客户名称模糊查询，根据联系人姓名模糊查询，根据查询时间起查询注册时间大于参数时间的数据，根据查询时间止查询注册时间小于参数时间的数据。点击查询按钮后对满足条件的数据进行筛选，最后分页展示。

### 产品类型管理

该功能只有管理员和操作员可以看到，当点击二级菜单“产品类型管理”，页面跳转至“产品类型管理”页面。如图4-6



图4-6产品类型管理界面

页面左侧展示出数据库中已经存在的所有的产品类型，页面右侧可以新增产品类型，挡在输入框输入内容后，会先校验该产品类型是否已经存在，如果不存在便可以新增，如果产品类型名称是已经存在的则页面提示，产品类型名称已存在，并且清空输入框，点击“取消”按钮的话，页面跳转至主页。

### 产品信息录入

该功能只有管理员和操作员才能够看到，点击“产品信息录入”页面跳转至产品信息录入页面。如图4-7



图4-7产品信息录入界面

对需要新增的产品输入各种参数以后，点击“保存”，即添加成功，如果，带年纪“保存”之前未输入参数，则无法提交表单信息，如果点击“取消”页面跳转至主页。

### 产品信息修改

该功能只有管理员和操作员可以看到，管理员有两种方式进入该页面，分别是二级菜单“产品信息修改”、产品信息查看页对面每条数据修改操作的按钮。操作员只能通过二级菜单“产品信息修改”进入产品信息修改页面。如图4-8

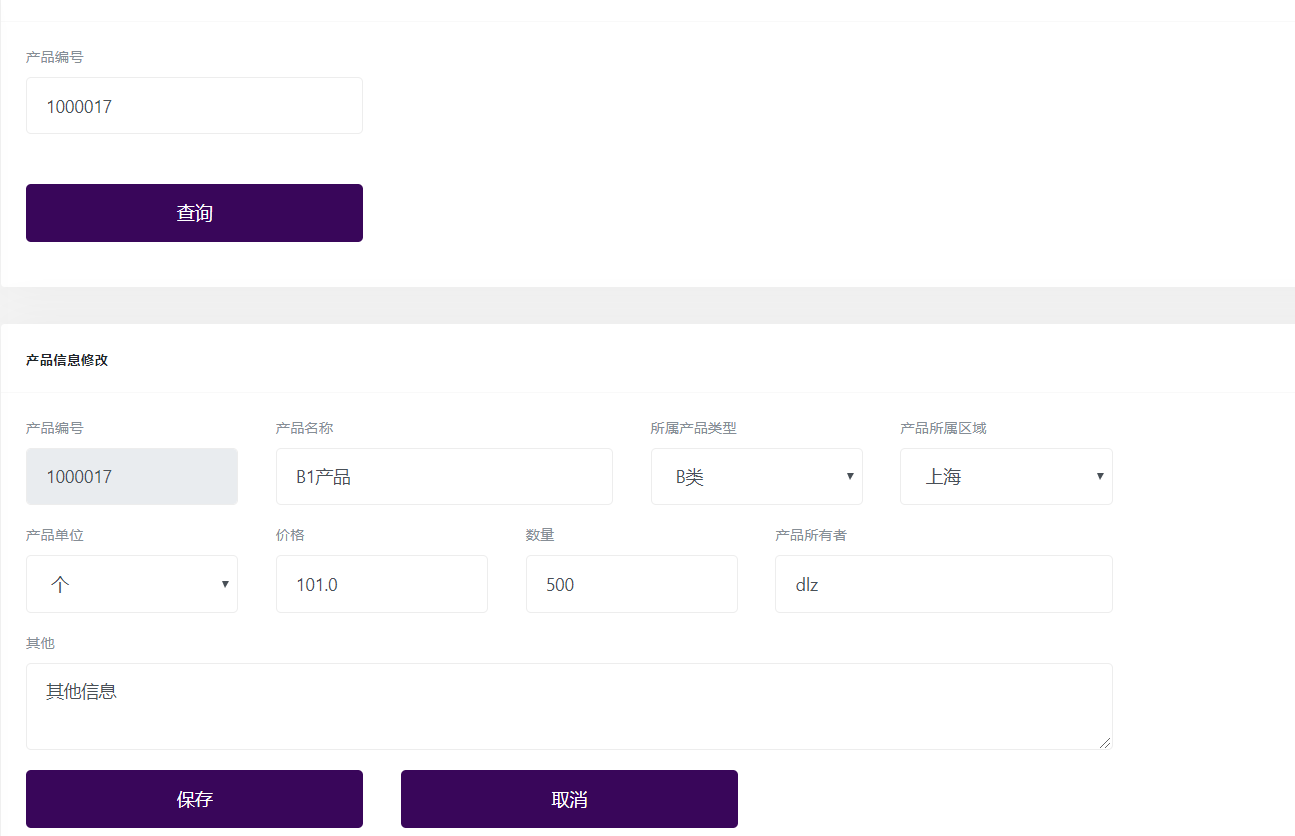


图4-8产品信息修改界面

页面中产品信息修改部分的产品编号文本框是只读的，填写需要修改的内容后，点击保存，即可修改成功，修改成功根据权限判断页面跳转，如果是管理员跳转至产品信息查看页面，如果是操作员跳转至主页。

### 产品信息查看

该功能管理员和客户可以看到，但是客户不能对数据进行修改，点击产品信息查看页面跳转至产品信息查看页面，管理员权限的页面如图4-9

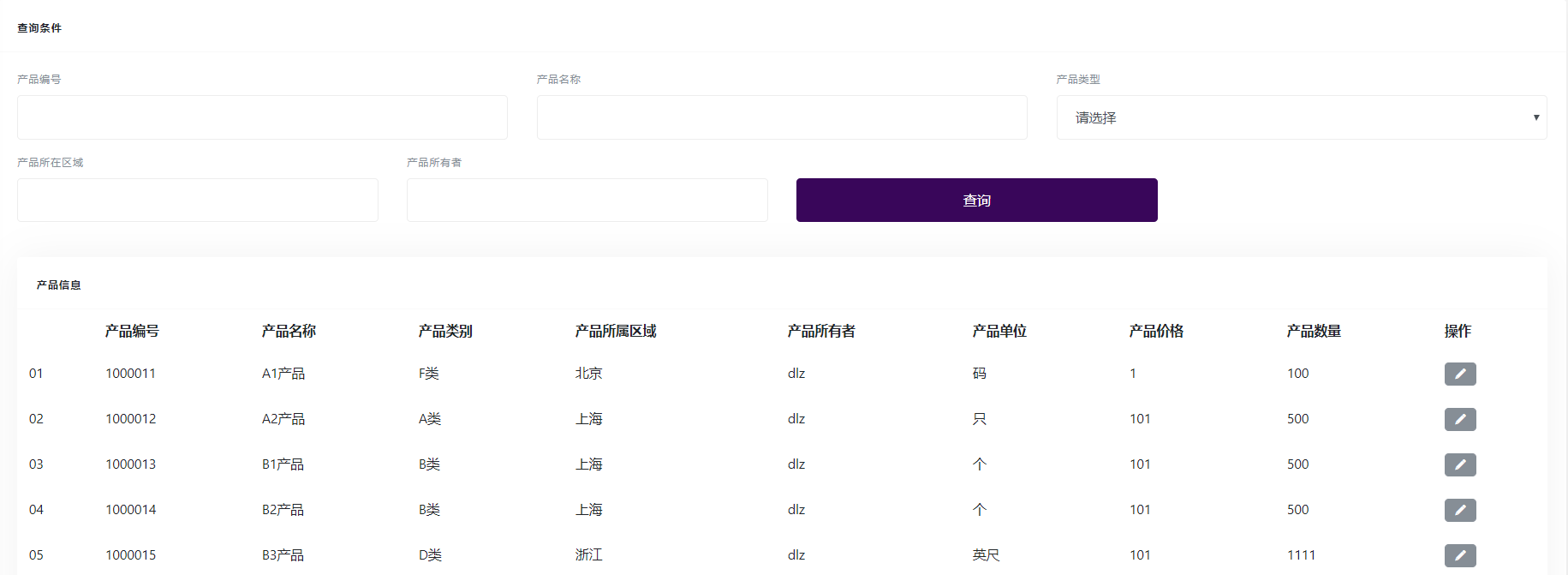


图4-9产品信息查看界面

管理员和客户都可以根据查询条件对产品信息进行查询。

### 订单信息录入

该功能管理员和操作员可以看到，点击二级菜单“订单信息录入”页面跳转至订单信息录入页面。如图4-10

图4-10订单信息录入界面

其中客户名称、产品类型、产品名称都是下拉框可以进行选择，产品数量文本框进行了js校验，只能输入数字。点击保存按钮，订单信息录入成功。点击取消，页面跳转至主页。

### 订单信息查看

该功能只有管理员可以看到，当登录的账号是管理员权限的，点击订单信息查看，页面跳转至订单信息查看页面。如图4-11

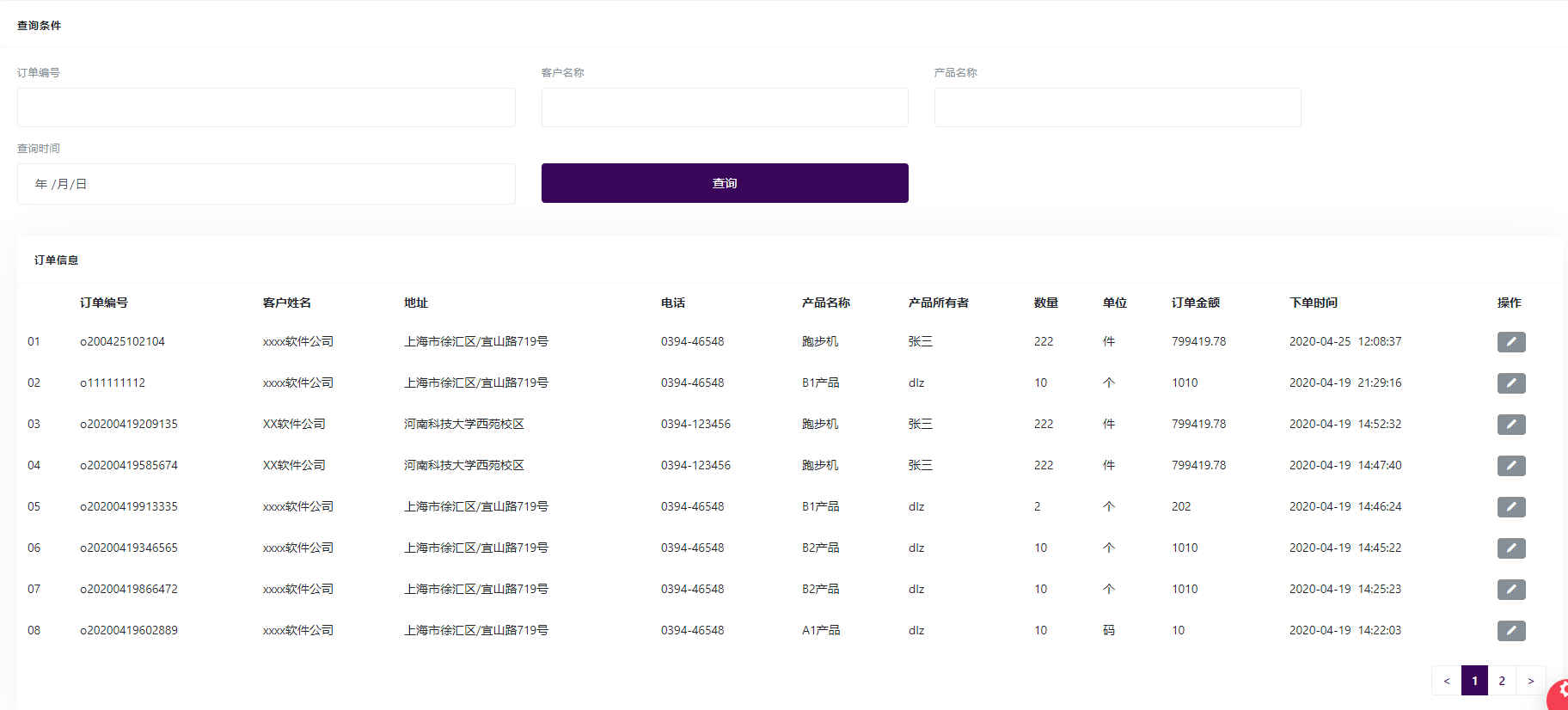


图4-11订单信息查看界面

页面数据采用数据分页的方式展示数据，默认每页8条数据。可以根据查询条件对数据进行筛选，最后展示在页面上。管理员可以点击每条数据的详细

按钮查看订单的完整信息。

### 报价信息录入

该功能只有管理员和操作员可以看到，如果登录的账号权限是管理员或操作员，点击纪报价信息录入，页面跳转至报价信息录入页面。如图4-12

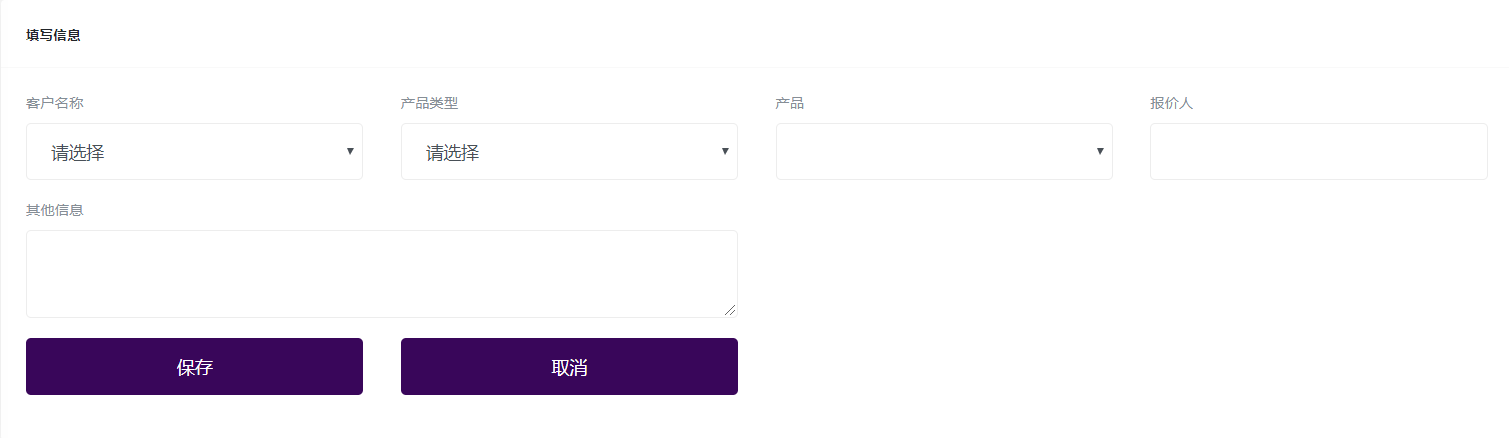


图4-12报价信息录入界面

填写好报价信息之后，点击保存，即可完成报价信息的录入，点击取消的话，页面跳转至主页。

### 报价信息查看

该功能只有管理员和客户可以看到，点击报价信息查看，页面跳转至报价信息查看页面，如果登录账号的权限是管理员，还有删除报价信息的权限。如图4-13



图4-13报价信息查看界面

页面同样采用数字分页的方式对报价信息进行分页展示，默认每页8条数据，如果是管理员，可以删除某条数据，只需要点击删除按钮，即可将该条数据删除。

# 系统测试

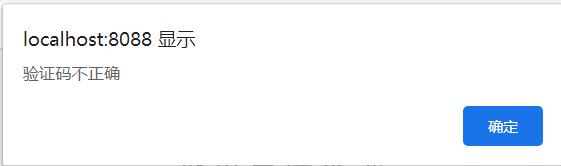
本系统的开发使用到了MySQL数据库做为数据存储的工具，并且使用数据库可视化工具Navicat，开发工具使用到的是Eclipse，项目使用Maven进行jar包管理，框架采用的是SSM（Spring+SpringMVC+Mybatis），使用JSP页面实现视图的展示，JSP页面中使用到了EL、JSTL表达式来简化代码量，前端使用到了Ajax、Jquery。最后系统功能基本上实现。现在在本地计算机上进行功能测试，在测试描述中，在前面模块的测试中进行详细展示，后面若有类似则不再展示。

以下仅展示部分测试内容。

## 系统的功能测试

### 测试登录注册功能

打开浏览器地址栏输入localhost:8088/qms 进入系统的登录页面，点击注册按钮，若输入的验证码不正确或验证码超时或者注册成功，页面弹窗给出提示信息。如图5-1





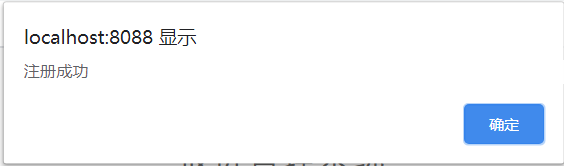


图5-1注册界面

### 测试客户录入功能

进入客户信息录入界面，如果未填写客户信息就点击“保存”按钮，则非空字段会在文本框的下面提示“请输入该字段”并且文本框的边框变为红色。如图5-2

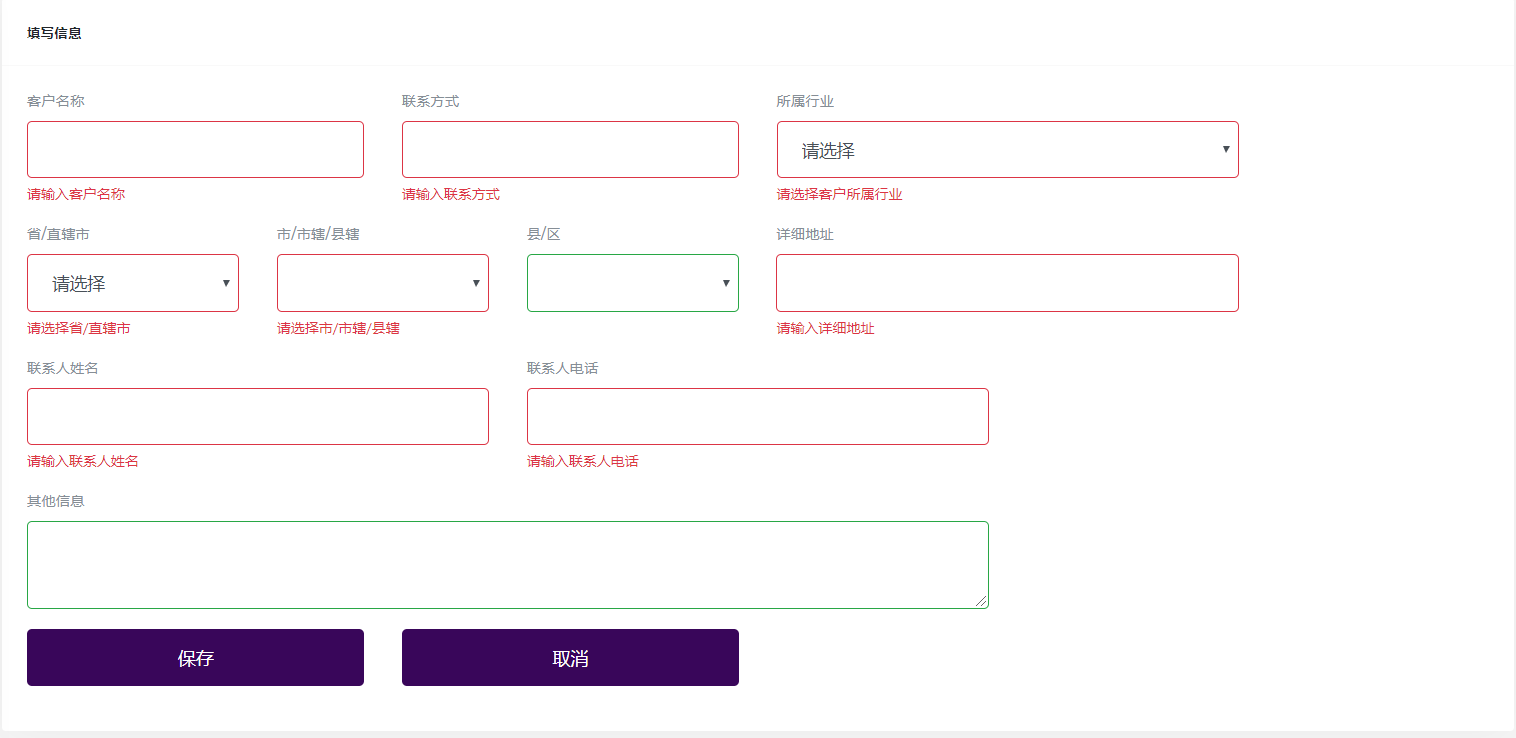


图5-2客户信息录入界面

### 测试客户信息修改功能

客户信息修改功能可对客户的信息进行修改，但要求某些信息不能为空，如果为空的话，会在文本框下方进行提示“请输入该字段”并且文本框的边框变为红色。如图5-3



图5-3客户信息修改界面

### 测试客户信息查看功能

点击客户信息查看，当展示的已经是首页或末页的时候，在点击上一页或者下一页的时候，页面弹窗给出提示信息“已经是第一页了！”，“已经是最后一页了！”。如图5-4





图5-4客户信息查看界面

### 测试删除客户功能

点击单条数据操作列中的“删除”按钮，页面弹出“确定删除客户吗？”。如图5-5



图5-5 客户信息删除确认界面

点击“确认”，客户信息删除成功，页面给出“删除成功！”提示。如图5-6



图5-6 客户信息删除成功界面

### 测试产品类型录入

如果输入的产品类型名称是已存在的，则页面弹窗给出提示“产品类型已存在”并且清空输入框。如图5-7

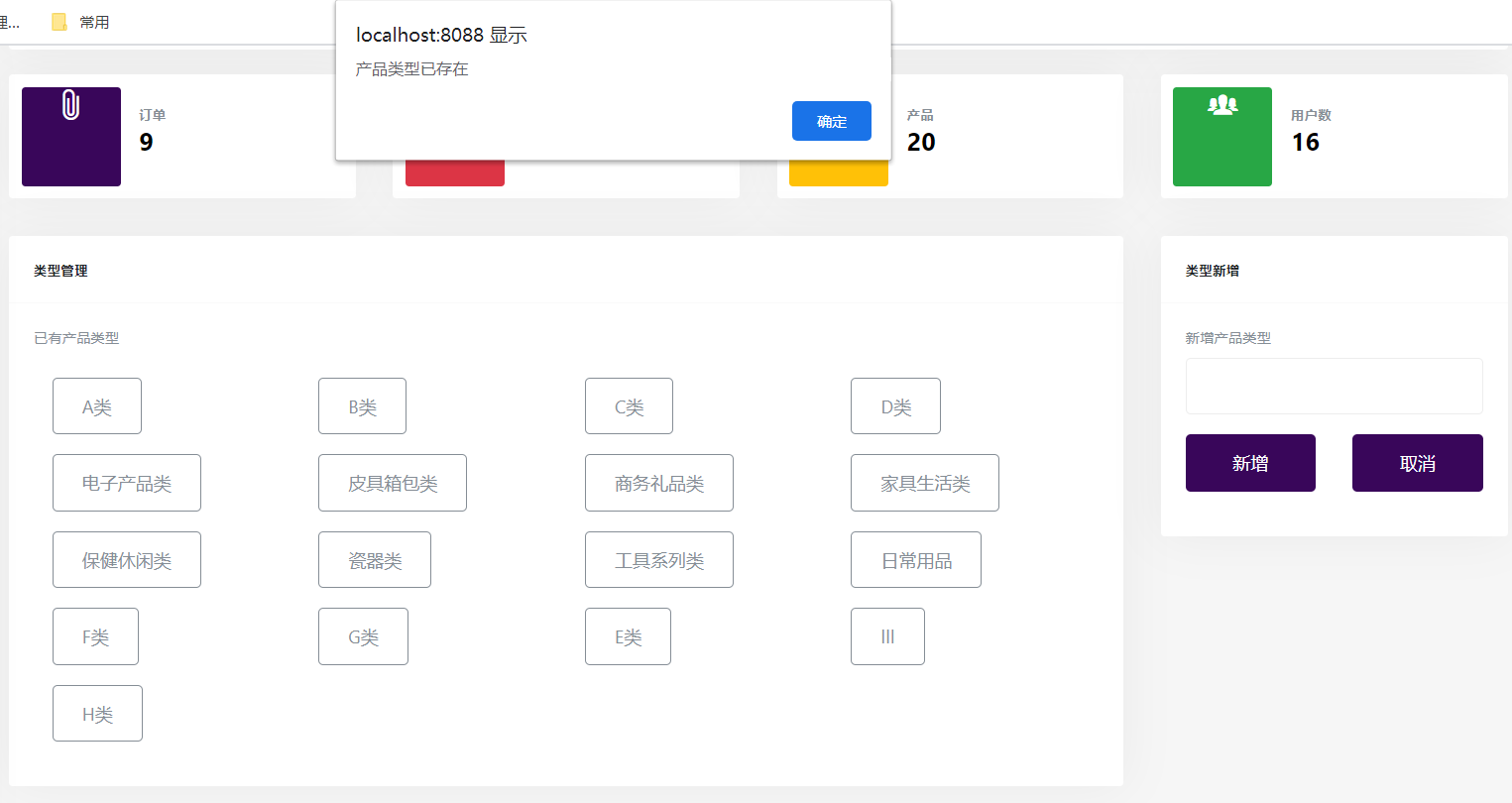


图5-7 产品类型录入界面

### 测试产品信息录入

进入产品信息录入页面，如果未填写产品信息就点击“保存”按钮，则非空字段会在文本框的下面提示“请输入该字段”并且文本框的边框变为红色。如图5-8



图5-8产品信息录入界面

### 测试产品信息查看

进入产品信息查看页面，页面对数据进行分页展示，默认第1页8条数据，数据展示上方为查询条件，可以根据其中一个或多个条件对数据进行筛选。如图5-9

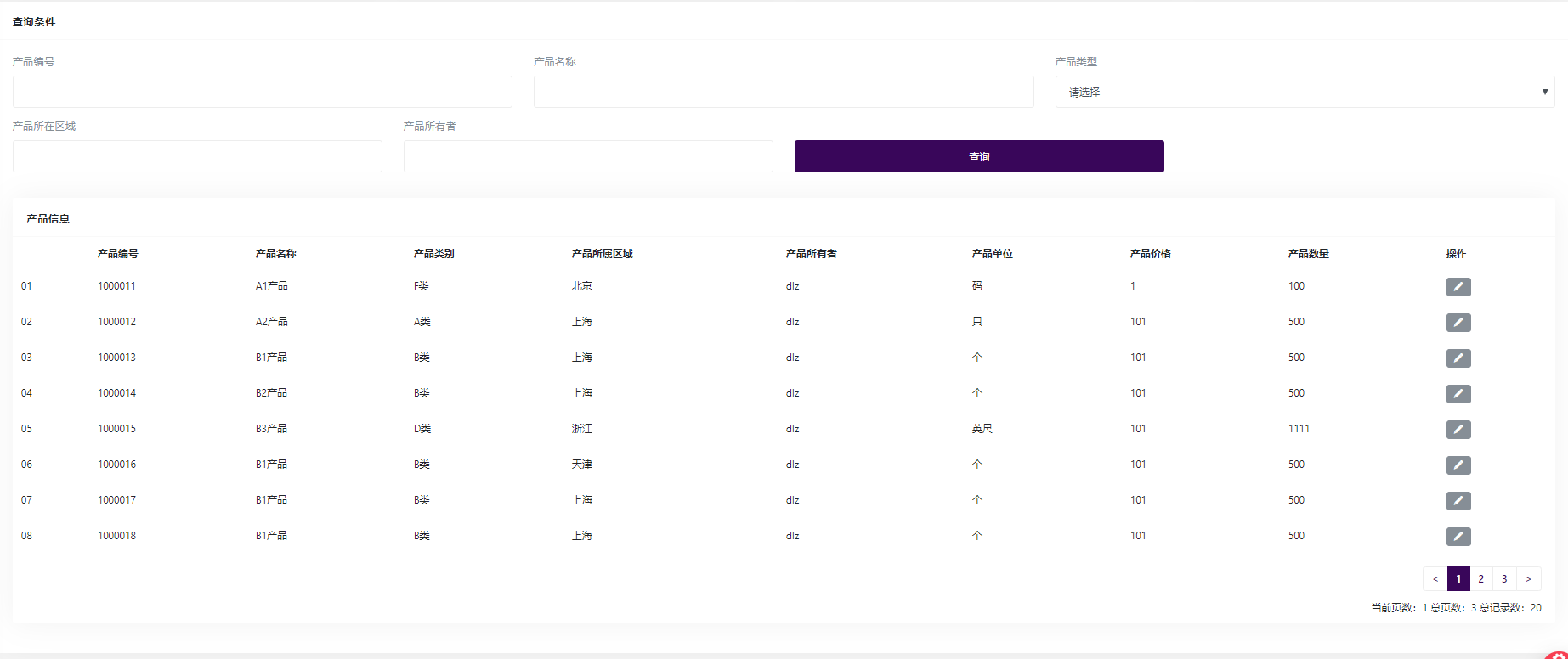


图5-9 产品信息查看界面

### 测试订单信息录入

进入订单信息录入页面，如果未填写订单信息就点击“保存”按钮，则非空字段会在文本框的下面提示“请输入该字段”并且文本框的边框变为红色。如图5-10



图5-10 订单信息录入界面

### 测试订单信息查看

进入订单信息查看页面，页面对数据进行分页展示，默认第1页8条数据，数据展示上方为查询条件，可以根据其中一个或多个条件对数据进行筛选。如图5-11

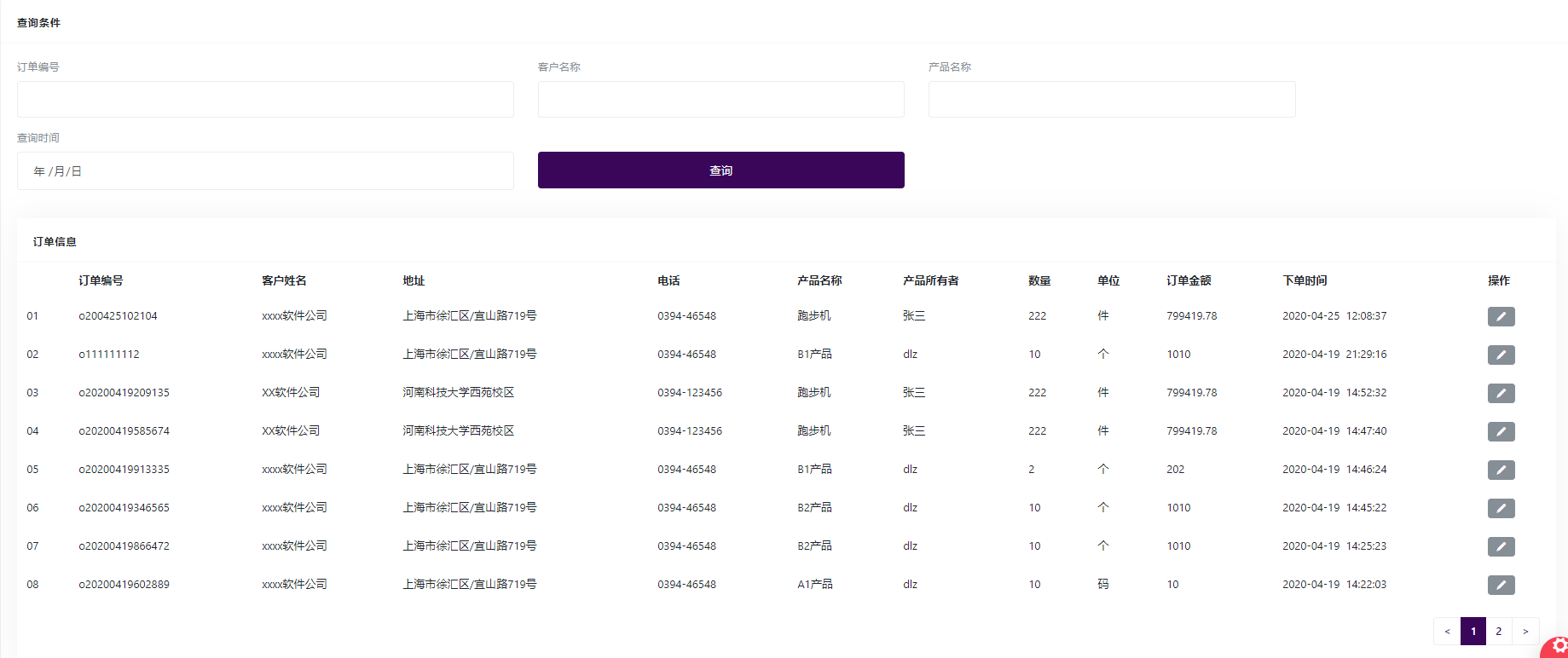


图5-11 订单信息查看界面

### 测试报价信息录入

进入报价信息录入页面，如果未填写订单信息就点击“保存”按钮，则非空字段会在文本框的下面提示“请输入该字段”并且文本框的边框变为红色。如图5-12

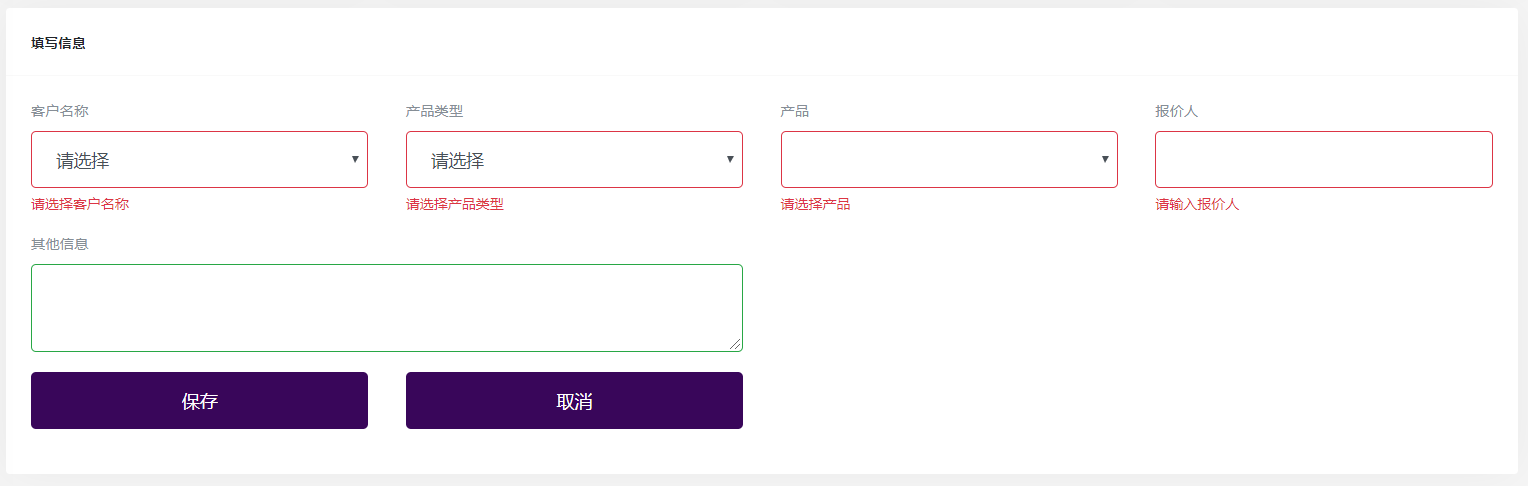


图5-12报价信息录入界面

### 测试报价信息查看

进入报价信息查看页面，页面对数据进行分页展示，默认第1页8条数据。页面展示的是管理员权限页面，可以删除报价信息。如图5-13



图5-13 报价信息查看界面

## 系统测试结论

通过对上述功能的测试，各模块的功能基本上都达到了预期的效果，运行相对稳定，测试成功。

# 

结 论

通过这次的毕业设计，我收获了很多东西。总结大概有以下几点：

第一，体会到了有了框架之后编写代码效率的提高，那是因为框架中封装了普通项目中需要重复书写的代码和简化了调用过程，就比如说在传统的jsp项目中，我们的controller接收到前端的请求后，需要我们去开发Dao层，里面涉及到了数据库的连接和存储过程的代码，大部分都是冗余的代码，但是有了SSM框架后就极大的简化了Dao层，只需要service层和mapper层就可以了，mapper层是用来连接mapper.xml文件的，而直接在mapper.xml中做sql语句的开发就可以了。而数据库连接和存储过程都直接有mybatis负责了，这样就完成了一次完成的数据库交互！

第二，就是对于mysql的使用，mysql做为系统数据存储的工具，通过本次毕业设计的制作，其中用到了表关联、模糊查询、动态sql等等，让我对于sql语句的使用更加的熟练。

第三，对于Ajax、Jquery的使用，项目中大多数地方使用到了ajax来与后台进行交互，对返回的结果进行处理，比如：数字分页，让我对Ajax、Jquery的使用更加的熟练。

第四，在注册功能中对于第三方接口的使用，能够熟练的使用Java发送邮件用来做校验。

在此次毕业设计的过程中，毫无疑问这次的毕业设计教会了我很多东西，极大的提升了自我学习能力。首先对Java面向对象的理解的更加深入，其次是对Java常用框架有了更加深入的理解，最后就是不能吃毕业设计锻炼了个人的耐心，因为项目开发的周期比较长，而且是独自开发，这在之前是没有过的，最后对于此次毕业设计的制作加大了我对开发的兴趣，在以后一定会更加努力的。

参考文献

1. 张永强.计算机软件Java编程特点及其技术分析[J]. 计算机产品与流通.2019-01-23
2. 王越.Java编程语言在计算机软件开发中的应用[J]. 电子技术与软件工程.2019-08-18
3. 贾晓芳,沈泽刚. Java Web应用开发中的常见乱码形式及解决方法[J]. 软件导刊.2017-04-20
4. 唐权. SSM框架在JavaEE企业级项目开发中 的应用与实践. 软件导刊.2017-03-25
5. [郑智方](https://kns.cnki.net/kns/popup/knetsearchNew.aspx?sdb=CJFQ&sfield=%e4%bd%9c%e8%80%85&skey=%e9%83%91%e6%99%ba%e6%96%b9&scode=43673021&acode=43673021" \t "https://kns.cnki.net/kns/brief/knet).[MySQL的重要性以及步入云的应用实例](https://kns.cnki.net/kns/detail/detail.aspx?QueryID=6&CurRec=30&recid=&FileName=WXXJ202001138&DbName=CJFDLAST2020&DbCode=CJFQ&yx=&pr=&URLID=&bsm=QKS0504;" \t "https://kns.cnki.net/kns/brief/_blank). [计算机产品与流通](https://kns.cnki.net/kns/NaviBridge.aspx?bt=1&DBCode=CJFD&BaseID=WXXJ&UnitCode=&NaviLink=%e8%ae%a1%e7%ae%97%e6%9c%ba%e4%ba%a7%e5%93%81%e4%b8%8e%e6%b5%81%e9%80%9a" \t "https://kns.cnki.net/kns/brief/_blank), 2020-01-15 .
6. [邹青松](https://kns.cnki.net/kns/popup/knetsearchNew.aspx?sdb=CJFQ&sfield=%e4%bd%9c%e8%80%85&skey=%e9%82%b9%e9%9d%92%e6%9d%be&scode=43770216&acode=43770216" \t "https://kns.cnki.net/kns/brief/knet). [计算机软件Java编程特点及其技术运用](https://kns.cnki.net/kns/detail/detail.aspx?QueryID=9&CurRec=30&recid=&FileName=HELJ202006026&DbName=CJFDAUTO&DbCode=CJFQ&yx=&pr=&URLID=&bsm=QKK0104;" \t "https://kns.cnki.net/kns/brief/_blank). [黑龙江科学](https://kns.cnki.net/kns/NaviBridge.aspx?bt=1&DBCode=CJFD&BaseID=HELJ&UnitCode=&NaviLink=%e9%bb%91%e9%be%99%e6%b1%9f%e7%a7%91%e5%ad%a6" \t "https://kns.cnki.net/kns/brief/_blank),2020-03-23 .
7. [官亚芬](https://kns.cnki.net/kns/popup/knetsearchNew.aspx?sdb=CJFQ&sfield=%e4%bd%9c%e8%80%85&skey=%e5%ae%98%e4%ba%9a%e8%8a%ac&scode=40451506&acode=40451506" \t "https://kns.cnki.net/kns/brief/knet). [计算机软件开发中JAVA编程语言及其实际应用](https://kns.cnki.net/kns/detail/detail.aspx?QueryID=31&CurRec=103&recid=&FileName=TXWL202004082&DbName=CJFDAUTO&DbCode=CJFQ&yx=&pr=&URLID=&bsm=QK0103;" \t "https://kns.cnki.net/kns/brief/_blank). [中国新通信](https://kns.cnki.net/kns/NaviBridge.aspx?bt=1&DBCode=CJFD&BaseID=TXWL&UnitCode=&NaviLink=%e4%b8%ad%e5%9b%bd%e6%96%b0%e9%80%9a%e4%bf%a1" \t "https://kns.cnki.net/kns/brief/_blank), 2020-02-20.
8. [谢懿](https://kns.cnki.net/kns/popup/knetsearchNew.aspx?sdb=CJFQ&sfield=%e4%bd%9c%e8%80%85&skey=%e8%b0%a2%e6%87%bf&scode=15709918&acode=15709918" \t "https://kns.cnki.net/kns/brief/knet). [计算机软件JAVA编程特点及其技术分析](https://kns.cnki.net/kns/detail/detail.aspx?QueryID=48&CurRec=211&recid=&FileName=NJCM202001158&DbName=CJFDAUTO&DbCode=CJFQ&yx=&pr=CJFT2020;&URLID=&bsm=QK0604;T05;" \t "https://kns.cnki.net/kns/brief/_blank). [农家参谋](https://kns.cnki.net/kns/NaviBridge.aspx?bt=1&DBCode=CJFD&BaseID=NJCM&UnitCode=&NaviLink=%e5%86%9c%e5%ae%b6%e5%8f%82%e8%b0%8b" \t "https://kns.cnki.net/kns/brief/_blank).2020-01-05 .
9. [余涛](https://kns.cnki.net/kns/popup/knetsearchNew.aspx?sdb=CJFQ&sfield=%e4%bd%9c%e8%80%85&skey=%e4%bd%99%e6%b6%9b&scode=44133136&acode=44133136" \t "https://kns.cnki.net/kns/brief/knet). [计算机软件开发中Java编程语言的应用研究](https://kns.cnki.net/kns/detail/detail.aspx?QueryID=48&CurRec=224&recid=&FileName=CXJL202001070&DbName=CJFDAUTO&DbCode=CJFQ&yx=&pr=&URLID=&bsm=QK0202;" \t "https://kns.cnki.net/kns/brief/_blank). [信息记录材料](https://kns.cnki.net/kns/NaviBridge.aspx?bt=1&DBCode=CJFD&BaseID=CXJL&UnitCode=&NaviLink=%e4%bf%a1%e6%81%af%e8%ae%b0%e5%bd%95%e6%9d%90%e6%96%99" \t "https://kns.cnki.net/kns/brief/_blank).2020-01-01 .
10. 单东林, 张晓菲, 魏然. 锋利的jQuery [M]. 北京:人民邮电出版社, 2012.
11. [刘焱](https://kns.cnki.net/kns/popup/knetsearchNew.aspx?sdb=CJFD&sfield=%e4%bd%9c%e8%80%85&skey=%e5%88%98%e7%84%b1&scode=32447766&acode=32447766" \t "https://kns.cnki.net/kns/brief/knet). [jQuery在平台中的应用开发研究](https://kns.cnki.net/kns/detail/detail.aspx?QueryID=68&CurRec=42&recid=&FileName=XXXT201901064&DbName=CJFDLAST2019&DbCode=CJFD&yx=&pr=&bsm=QK0203;" \t "https://kns.cnki.net/kns/brief/_blank)]. [信息系统工程](https://kns.cnki.net/kns/NaviBridge.aspx?bt=1&DBCode=CJFD&BaseID=XXXT&UnitCode=&NaviLink=%e4%bf%a1%e6%81%af%e7%b3%bb%e7%bb%9f%e5%b7%a5%e7%a8%8b" \t "https://kns.cnki.net/kns/brief/_blank). 2019-01-20 .
12. [刘志洋](https://kns.cnki.net/kns/popup/knetsearchNew.aspx?sdb=CJFQ&sfield=%e4%bd%9c%e8%80%85&skey=%e5%88%98%e5%bf%97%e6%b4%8b&scode=43928081&acode=43928081" \t "https://kns.cnki.net/kns/brief/knet). [ajax技术在web程序开发中的运用探讨](https://kns.cnki.net/kns/detail/detail.aspx?QueryID=73&CurRec=1&recid=&FileName=GXFZ202002079&DbName=CJFDLAST2020&DbCode=CJFQ&yx=&pr=&URLID=&bsm=QK0203;" \t "https://kns.cnki.net/kns/brief/_blank). [轻纺工业与技术](https://kns.cnki.net/kns/NaviBridge.aspx?bt=1&DBCode=CJFD&BaseID=GXFZ&UnitCode=&NaviLink=%e8%bd%bb%e7%ba%ba%e5%b7%a5%e4%b8%9a%e4%b8%8e%e6%8a%80%e6%9c%af" \t "https://kns.cnki.net/kns/brief/_blank).2020-02-25.
13. 徐雯, 高建华. 基于Spring MVC及Mybatis的Web应用框架研究[J]. 微型电脑应用, 2012, 28(7):1-4.
14. [林萍](https://kns.cnki.net/kns/popup/knetsearchNew.aspx?sdb=CJFQ&sfield=%e4%bd%9c%e8%80%85&skey=%e6%9e%97%e8%90%8d&scode=15006022&acode=15006022" \t "https://kns.cnki.net/kns/brief/knet), [朱婵](https://kns.cnki.net/kns/popup/knetsearchNew.aspx?sdb=CJFQ&sfield=%e4%bd%9c%e8%80%85&skey=%e6%9c%b1%e5%a9%b5&scode=11489954&acode=11489954" \t "https://kns.cnki.net/kns/brief/knet).[基于Ajax技术和JAVAEE的分页查询优化](https://kns.cnki.net/kns/detail/detail.aspx?QueryID=73&CurRec=51&recid=&FileName=XTYY201708031&DbName=CJFDLAST2017&DbCode=CJFQ&yx=&pr=&URLID=&bsm=QK0203;" \t "https://kns.cnki.net/kns/brief/_blank). [计算机系统应用](https://kns.cnki.net/kns/NaviBridge.aspx?bt=1&DBCode=CJFD&BaseID=XTYY&UnitCode=&NaviLink=%e8%ae%a1%e7%ae%97%e6%9c%ba%e7%b3%bb%e7%bb%9f%e5%ba%94%e7%94%a8" \t "https://kns.cnki.net/kns/brief/_blank), 2017-08-15.
15. [温立辉](https://kns.cnki.net/kns/popup/knetsearchNew.aspx?sdb=CJFQ&sfield=%e4%bd%9c%e8%80%85&skey=%e6%b8%a9%e7%ab%8b%e8%be%89&scode=36379631&acode=36379631" \t "https://kns.cnki.net/kns/brief/knet).[AJAX异步交互技术浅析](https://kns.cnki.net/kns/detail/detail.aspx?QueryID=73&CurRec=65&recid=&FileName=SDGJ201704189&DbName=CJFDLAST2017&DbCode=CJFQ&yx=&pr=&URLID=&bsm=QK0204;" \t "https://kns.cnki.net/kns/brief/_blank). [山东工业技术](https://kns.cnki.net/kns/NaviBridge.aspx?bt=1&DBCode=CJFD&BaseID=SDGJ&UnitCode=&NaviLink=%e5%b1%b1%e4%b8%9c%e5%b7%a5%e4%b8%9a%e6%8a%80%e6%9c%af" \t "https://kns.cnki.net/kns/brief/_blank),2017-02-15 .
16. 许家珆, 白忠建, 吴磊. 软件工程:理论与实践(第2版)[M]. 北京:高等教育出版社.2009.
17. [邱吉雨](https://kns.cnki.net/kns/popup/knetsearchNew.aspx?sdb=CJFQ&sfield=%e4%bd%9c%e8%80%85&skey=%e9%82%b1%e5%90%89%e9%9b%a8&scode=40048474&acode=40048474" \t "https://kns.cnki.net/kns/brief/knet). [基于Java开发Web项目的核心技术](https://kns.cnki.net/kns/detail/detail.aspx?QueryID=90&CurRec=6&recid=&FileName=DYXU201807026&DbName=CJFDLAST2018&DbCode=CJFQ&yx=&pr=&URLID=&bsm=QKK0203;" \t "https://kns.cnki.net/kns/brief/_blank). [电子元器件与信息技术](https://kns.cnki.net/kns/NaviBridge.aspx?bt=1&DBCode=CJFD&BaseID=DYXU&UnitCode=&NaviLink=%e7%94%b5%e5%ad%90%e5%85%83%e5%99%a8%e4%bb%b6%e4%b8%8e%e4%bf%a1%e6%81%af%e6%8a%80%e6%9c%af" \t "https://kns.cnki.net/kns/brief/_blank), 2018-07-20 .
18. [李鹏博](https://kns.cnki.net/kns/popup/knetsearchNew.aspx?sdb=CJFQ&sfield=%e4%bd%9c%e8%80%85&skey=%e6%9d%8e%e9%b9%8f%e5%8d%9a&scode=35388771&acode=35388771" \t "https://kns.cnki.net/kns/brief/knet),[于立婷](https://kns.cnki.net/kns/popup/knetsearchNew.aspx?sdb=CJFQ&sfield=%e4%bd%9c%e8%80%85&skey=%e4%ba%8e%e7%ab%8b%e5%a9%b7&scode=34210028&acode=34210028" \t "https://kns.cnki.net/kns/brief/knet),[王天琪](https://kns.cnki.net/kns/popup/knetsearchNew.aspx?sdb=CJFQ&sfield=%e4%bd%9c%e8%80%85&skey=%e7%8e%8b%e5%a4%a9%e7%90%aa&scode=35388772&acode=35388772" \t "https://kns.cnki.net/kns/brief/knet). [Java web软件框架技术探析](https://kns.cnki.net/kns/detail/detail.aspx?QueryID=90&CurRec=21&recid=&FileName=TXSJ201614208&DbName=CJFDLAST2016&DbCode=CJFQ&yx=&pr=&URLID=&bsm=QK0203;QS0103;" \t "https://kns.cnki.net/kns/brief/_blank). [通讯世界](https://kns.cnki.net/kns/NaviBridge.aspx?bt=1&DBCode=CJFD&BaseID=TXSJ&UnitCode=&NaviLink=%e9%80%9a%e8%ae%af%e4%b8%96%e7%95%8c" \t "https://kns.cnki.net/kns/brief/_blank), 2016-07-25 .
19. [范成臣](https://kns.cnki.net/kns/popup/knetsearchNew.aspx?sdb=CJFQ&sfield=%e4%bd%9c%e8%80%85&skey=%e8%8c%83%e6%88%90%e8%87%a3&scode=30306160&acode=30306160" \t "https://kns.cnki.net/kns/brief/knet).[浅析在JAVA WEB应用程序中使用Servlet技术](https://kns.cnki.net/kns/detail/detail.aspx?QueryID=90&CurRec=70&recid=&FileName=SZJT201309090&DbName=CJFDHIS2&DbCode=CJFQ&yx=&pr=&URLID=&bsm=QK0203;" \t "https://kns.cnki.net/kns/brief/_blank). [数字技术与应用](https://kns.cnki.net/kns/NaviBridge.aspx?bt=1&DBCode=CJFD&BaseID=SZJT&UnitCode=&NaviLink=%e6%95%b0%e5%ad%97%e6%8a%80%e6%9c%af%e4%b8%8e%e5%ba%94%e7%94%a8" \t "https://kns.cnki.net/kns/brief/_blank).2013-09-15 .
20. 于亚芳,郭磊.JavaEE框架技术课程建设研究[J].电脑知识与技术.2018-10-19.

致　谢

伴随着六月份的到来，我们即将要对四年的大学美好生活画上一个圆满的句号。回望这四年，大一入学似乎就在昨天，可转眼我们就要离开生活了四年的地方。在这段时光里，我们欢笑过、挥泪过、奋斗过、拼搏过，大学的时光是充实而有丰富的，我们既有满满当当的课程，又有轻松愉快的课下生活。和蔼可亲的老师们、奋发向上的同学们、会帮互助的室友们，这些都是我们人生中积累的财富。

首先，感谢我的父母对于我在经济上的支持。其次，感谢我的老师、同学、室友们对于我在学习、生活上的帮助与鼓励。再次，感谢软件学院对于我在学习环境、师资力量上的提供，如果没没有学院设备的提供，我们无法享受到这么舒服安逸的学习生活，没有老师的智慧，我们不可能学到重组的知识去面对以后的职业生涯。最后，感谢河南科技大学提供给我读大学的机会。正因为有了他们我们才能走向人生的下一个阶段。

附　录

项目核心代码

/\*

\* @author 李自强

\* @since 2019年3月10日

\* \*/

@Controller

public class userController {

private final static Logger logger = LoggerFactory.getLogger(userController.class);

@Resource(name = "userService")

userService us;

@RequestMapping("topreindex")

public String toindex(HttpServletRequest request,HttpSession session,HttpServletResponse response) throws Exception {

return "prestageHtml/index";

}

@RequestMapping("prelogin")

public String login(HttpServletResponse response) {

response.setContentType("text/html;charset=utf-8");

return "prestageHtml/login";

}

@RequestMapping("tuichu")

public String tuichu(HttpSession session) {

session.removeAttribute("success");

session.removeAttribute("user");

return "redirect:/goodsindex";

}

@RequestMapping("doprelogin")

public ModelAndView doprelogin(Users u, HttpServletResponse response, HttpSession session) throws IOException {

response.setContentType("text/html;charset=utf-8");

PrintWriter out = response.getWriter();

ModelAndView mv = new ModelAndView();

Users user = us.selectUser(u);

if (user != null) {

if (session.getAttribute("success").equals("1")) {

session.setAttribute("user", user);

session.setAttribute("username", user.getUsername());

return new ModelAndView(new RedirectView("goodsindex"));

} else {

out.print("<script>alert('验证码不正确')</script>");

mv.setViewName("prestageHtml/login");

}

}

else{

mv.setViewName("prestageHtml/login");

out.println("<script>alert('用户密码不匹配')</script>");

}

return mv;

}

@RequestMapping("preregister")

public String register() {

return "prestageHtml/register";

}

@RequestMapping("userManager")

public ModelAndView userManager(HttpSession session,Integer pageNow) {

ModelAndView mv = new ModelAndView();

if(pageNow==null)

{

pageNow=1;

}

List<Users> users=us.userPage((pageNow-1)\*6);

System.out.println(users);

mv.addObject("userList", users);

mv.setViewName("sufstageHtml/user/userManager");

return mv;

}

@RequestMapping("dopreregister")

public ModelAndView doregister(Users u, HttpServletResponse response, HttpSession session) throws IOException {

response.setContentType("text/html;charset=utf-8");

ModelAndView mv = new ModelAndView();

u.setRegtime(new Date());

PrintWriter out = response.getWriter();

String pwd = MD5Utils.MD5Encode(u.getUserpwd(), "utf-8");

u.setUserpwd(pwd);

int a = us.insert(u);

if (a > 0) {

if (session.getAttribute("success").equals("1")) {

out.print("<script>alert('注册成功')</script>");

session.setAttribute("user", u);

mv.setViewName("redirect:/prelogin");

} else {

out.print("<script>alert('验证码不正确')</script>");

mv.setViewName("redirect:/preregister");

}

} else {

out.print("<script>alert('注册失败')</script>");

mv.setViewName("prestageHtml/register");

}

return mv;

}

/\*\*

\* 生成验证码

\*/

@RequestMapping(value = "getVerify")

public void getVerify(HttpServletRequest request, HttpServletResponse response) {

try {

response.setContentType("image/jpeg");// 设置相应类型,告诉浏览器输出的内容为图片

response.setHeader("Pragma", "No-cache");// 设置响应头信息，告诉浏览器不要缓存此内容

response.setHeader("Cache-Control", "no-cache");

response.setDateHeader("Expire", 0);

RandomValidateCodeUtil randomValidateCode = new RandomValidateCodeUtil();

randomValidateCode.getRandcode(request, response);// 输出验证码图片方法

} catch (Exception e) {

logger.error("获取验证码失败>>>> ", e);

}

}

/\*\*

\* 校验验证码

\*/

@RequestMapping(value = "checkVerify", method = RequestMethod.POST)

@ResponseBody

public Map<String, Object> checkVerify(@RequestParam String verifyInput, HttpSession session) {

Map<String, Object> map = new HashMap<String, Object>();

try {

// 从session中获取随机数

String inputStr = verifyInput;

String random = (String) session.getAttribute("RANDOMVALIDATECODEKEY");

System.out.println(random + inputStr);

if (random == null) {

map.put("success", false);

session.setAttribute("success", "0");

return map;

}

if (random.equals(inputStr)) {

map.put("success", true);

session.setAttribute("success", "1");

return map;

} else {

map.put("success", false);

session.setAttribute("success", "0");

return map;

}

} catch (Exception e) {

logger.error("验证码校验失败", e);

map.put("success", false);

session.setAttribute("success", "0");

return map;

}

}

@RequestMapping("checkUname")

@ResponseBody

public Map<String, Object> checkUname(String username) {

Map<String, Object> map = new HashMap<String, Object>();

Users u=us.checkUname(username);

System.out.println(u);

if(u!=null)

{

map.put("checkUn", true);

}

else

{

map.put("checkUn", false);

}

return map;

}

}

@RestController

public class AlipayController {

@Resource(name="ShoppingCarService")

ShoppingCarService scs;

@RequestMapping(value="zhifu1")//改编所用 value=“zhifu1”为真正用法

public ModelAndView zhifu(HttpServletRequest request,String GoodsDetailId) {

int goodsdetailid =Integer.parseInt(GoodsDetailId);

Goodsdetail goodsDetail= scs.getColor(goodsdetailid);//改编所用 此处注释为真正用法

ModelAndView mv = new ModelAndView();

mv.addObject("goodsDetail",goodsDetail);//改编所用 此处注释为真正用法

mv.setViewName("prestageHtml/zhifu");//改编所用 此处prestageHtml/zhifu为真正用法

return mv;

}

@RequestMapping(value = "goAlipay", produces = "text/html; charset=UTF-8")

@ResponseBody

public String goAlipay(HttpServletRequest request,String goodsDetailId) throws Exception {

int goodsdetailid =Integer.parseInt(goodsDetailId);

//获得初始化的AlipayClient

AlipayClient alipayClient = new DefaultAlipayClient(AlipayConfig.URL, AlipayConfig.APPID, AlipayConfig.RSA\_PRIVATE\_KEY, "json", AlipayConfig.CHARSET, AlipayConfig.ALIPAY\_PUBLIC\_KEY, AlipayConfig.SIGNTYPE);

//设置请求参数

AlipayTradePagePayRequest alipayRequest = new AlipayTradePagePayRequest();

alipayRequest.setReturnUrl(AlipayConfig.return\_url);

alipayRequest.setNotifyUrl(AlipayConfig.notify\_url);

//商户订单号，商户网站订单系统中唯一订单号，必填

SimpleDateFormat tempDate=new SimpleDateFormat("yyyyMMddHHmmss");

String datetime =tempDate.format(new java.util.Date());

String out\_trade\_no =datetime;

//付款金额

float total\_amount =goodsDetail.getGoodsPrice();

//订单名称，必填

String subject = "巨哥";

//商品描述，可空

String body = "用户订购商品个数：" + "10";

// 该笔订单允许的最晚付款时间，逾期将关闭交易。取值范围：1m～15d。m-分钟，h-小时，d-天，1c-当天（1c-当天的情况下，无论交易何时创建，都在0点关闭）。 该参数数值不接受小数点， 如 1.5h，可转换为 90m。

String timeout\_express = "1c";

alipayRequest.setBizContent("{\"out\_trade\_no\":\""+ out\_trade\_no +"\","

+ "\"total\_amount\":\""+ total\_amount +"\","

+ "\"subject\":\""+ subject +"\","

+ "\"body\":\""+ body +"\","

+ "\"timeout\_express\":\""+ timeout\_express +"\","

+ "\"product\_code\":\"FAST\_INSTANT\_TRADE\_PAY\"}");

//请求

String result = alipayClient.pageExecute(alipayRequest).getBody();

return result;

}

/\*\*

\* @Description: 支付宝同步通知页面

\*/

@RequestMapping(value = "alipayReturnNotice")

public ModelAndView alipayReturnNotice(HttpServletRequest request, HttpServletRequest response) throws Exception {

//获取支付宝GET过来反馈信息

Map<String,String> params = new HashMap<String,String>();

Map<String,String[]> requestParams = request.getParameterMap();

for (Iterator<String> iter = requestParams.keySet().iterator(); iter.hasNext();) {

String name = (String) iter.next();

String[] values = (String[]) requestParams.get(name);

String valueStr = "";

for (int i = 0; i < values.length; i++) {

valueStr = (i == values.length - 1) ? valueStr + values[i]

: valueStr + values[i] + ",";

}

//乱码解决，这段代码在出现乱码时使用

valueStr = new String(valueStr.getBytes("ISO-8859-1"), "utf-8");

params.put(name, valueStr);

}

//调用SDK验证签名

boolean signVerified = AlipaySignature.rsaCheckV1(params, AlipayConfig.ALIPAY\_PUBLIC\_KEY, AlipayConfig.CHARSET, AlipayConfig.SIGNTYPE);

ModelAndView mv = new ModelAndView("alipaySuccess");

//——请在这里编写您的程序（以下代码仅作参考）——

if(signVerified) {

//商户订单号

String out\_trade\_no = new String(request.getParameter("out\_trade\_no").getBytes("ISO-8859-1"),"UTF-8");

//支付宝交易号

String trade\_no = new String(request.getParameter("trade\_no").getBytes("ISO-8859-1"),"UTF-8");

//付款金额

String total\_amount = new String(request.getParameter("total\_amount").getBytes("ISO-8859-1"),"UTF-8");

}

// 修改订单状态为支付成功，已付款; 同时新增支付流水

return mv;

}

/\*\*

\* @Description: 支付宝异步 通知页面

\*/

@RequestMapping(value = "alipayNotifyNotice")

@ResponseBody

public String alipayNotifyNotice(HttpServletRequest request, HttpServletRequest response) throws Exception {

//获取支付宝POST过来反馈信息

Map<String,String> params = new HashMap<String,String>();

Map<String,String[]> requestParams = request.getParameterMap();

for (Iterator<String> iter = requestParams.keySet().iterator(); iter.hasNext();) {

String name = (String) iter.next();

String[] values = (String[]) requestParams.get(name);

String valueStr = "";

for (int i = 0; i < values.length; i++) {

valueStr = (i == values.length - 1) ? valueStr + values[i]

: valueStr + values[i] + ",";

}

//乱码解决，这段代码在出现乱码时使用

/\*valueStr = new String(valueStr.getBytes("ISO-8859-1"), "utf-8");\*/

params.put(name, valueStr);

}

//调用SDK验证签名

boolean signVerified = AlipaySignature.rsaCheckV1(params, AlipayConfig.ALIPAY\_PUBLIC\_KEY, AlipayConfig.CHARSET, AlipayConfig.SIGNTYPE);

//——请在这里编写您的程序（以下代码仅作参考）——

/\* 实际验证过程建议商户务必添加以下校验：

1、需要验证该通知数据中的out\_trade\_no是否为商户系统中创建的订单号，

2、判断total\_amount是否确实为该订单的实际金额（即商户订单创建时的金额），

3、校验通知中的seller\_id（或者seller\_email) 是否为out\_trade\_no这笔单据的对应的操作方（有的时候，一个商户可能有多个seller\_id/seller\_email）

4、验证app\_id是否为该商户本身。

\*/

//验证成功

if(signVerified) {

//商户订单号

String out\_trade\_no = new String(request.getParameter("out\_trade\_no").getBytes("ISO-8859-1"),"UTF-8");

//支付宝交易号

String trade\_no = new String(request.getParameter("trade\_no").getBytes("ISO-8859-1"),"UTF-8");

//交易状态

String trade\_status = new String(request.getParameter("trade\_status").getBytes("ISO-8859-1"),"UTF-8");

//付款金额

String total\_amount = new String(request.getParameter("total\_amount").getBytes("ISO-8859-1"),"UTF-8");

}

return "success";

}

}

英文资料翻译

**Spring Boot 研究和应用(译文)**

Yonghe Wang, Jinsong Zhang, Anming Deng,zhixun Zhou

摘要：Spring Boot 是一个用于简化、加速 Spring 开发的框架，文章对 Spring Boot 的开发做了简要介绍，之后对整合传统

Spring 的开发做了探索性研究。

关键词：Spring Boot；开发；整合传统 Spring

1. SpringBoot简介

Spring Boot 是在 2013 年推出的新项目，主要用来简化Spring 开发框架的开发、配置、调试、部署工作，同时在项目内集成了大量易于使用且实用的基础框架。

在 Spring Boot 中集成的基础框架，是在开发中经常需要使用的框架，如内嵌容器（Tomcat、Jettty、Undertow）、日志框架、JMS 框架、持久化框架、流行的NOSQL 数据库（Cassandra、MongoDB）、缓存框架等。在传统的Spring 框架中开发，用户需要自行编写XML 文件，并在 Maven、Gradle 中加入相应的依赖包，在响应的代码中添加功能代码，才能使用。同时对于引入多个依赖包时，包之间的版本调整也是个困 难的问题，经常需要对依赖包的版本冲突进行处理。调整依赖包的版本以及冲突问题是一个费时费力、重复、繁琐的工作。 如果使用Spring Boot 开发，仅需在 Maven、Gradle 配置文件中加上少量的配置，即可在代码中使用所需的框架，让原本 的配置简化到几乎是零代码、零 XML 配置，同时依赖包的版本问题也由 Spring Boot 轻松解决。

Spring Boot 对目前的的主流构建工具 Maven、Gradle 都提供了良好的支持，对其他构建工具也提供了支持，如Ant，但是相对 Maven、Gradle 来说，支持的力度相对要少些，所以需要尽量使用 Maven、Gradle 来构建 Spring Boot 程序。本文的构建环境是 Maven，使用 Gradle 可以参考 Spring Boot 的官方 Reference 文档。

1. 简单示例 Hello World 开发对比

A. 使用传统Spring 开发Hello World 程序

使用传统Spring 开发一个Hello World 的Web 程序，构建开发工程是一个比较繁琐的过程，典型的工程，需要修改、配 置一下 4 个文件：Hello.java、Application.xml、servlet-hel- lo.xml、web.xml，即使在IDE 运行、调试，还是需要配置好J2EE容器才能进行调试工作。如果要进行运维、部署工作，还需要用IDE 将应用打包为war 文件，之后还要进行 J2EE 容器的配置、部署，再将 war 文件部署到容器中运行，才能看到运行效果。

1. 使用 Spring Boot 开发Hello World 程序

使用Spring Boot 开发Hello World 程序，所需的工作量将大大减少。仅仅需要修改、配置 2 个文件：Hello.java、pom.xml，借助 Spring Boot 框架内的内嵌容器，在 IDE 即可轻松启动Web 程序。

pom.xml 核心内容如下：

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>1.3.5.RELEASE</version>

</parent>

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

</dependencies> Hello.java 代码： package hello;

import org.springframework.boot.SpringApplication; import org.springframework.boot.autoconfigure. SpringBo-

otApplication;

import org.springframework.web.bind.annotation. Reques- tMapping;

import org.springframework.web.bind.annotation. Respon- seBody;

@SpringBootApplication public class Hello { @RequestMapping("/test/v1") @ResponseBody

String home() {

return "Hello World";

}

public static void main(String[] args) throws Exception { SpringApplication.run(Hello.class, args);

}

}

从 pom.xml 和 Hello.java 的代码可见，使用 Spring Boot， 代码、配置均得到了极大的简化，仅仅十多行代码加上 Maven 配置，即可完成一个 Web 程序。

同时这个 web 程序可以轻松打包为单一的可运行 jar 文件 hello.jar，在命令行使用如下命令：

java -jar hello.jar

就能不部署 J2EE 容器的情况下，启动一个 Web 应用。这种把Web 应用打包为单一可运行jar 包同时内嵌J2EE容器的方式，主要目的是简化应用的部署、配置工作。在实际运维中这种打包方式可以很好地配合Docker 类的虚拟化台达到简单、快速部署的目标。

1. SpringBoot的主要功能特性

作为一个简化Spring开发、调试、部署的框架，Spring Boot提供了许多好用的特性，这里仅介绍开发中主要特性。其他的诸多实用特性，比如：日志集成、安全技术、NoSQL 技术、消息（JMS）、邮件发送、JTA 分布式事务集成、JMX 的监控和管理技术集成、测试功能集成、使用条件的自动配置技术、WebSo- ckets 开发等不在本文中介绍，读者在本文的基础上，可以自行参考第六章中给出的资源进行扩展学习。

Spring Boot 的提供微服务框架开发功能，除了将 Web 程序打包为 war 文件的部署方式。还可以将 Web 程序编译为可独立运行的jar 包，同时还可以内嵌 J2EE 容器，使得 Web 程序也可以独立于容器外运行。

1. SpringApplication

SpringApplication 类提供了一种从main()方法启动Spring 应用的便捷方式。如下面的 SpringApplication 类代码所示， Hello 类在的 main () 方法中，通过调用 SpringgApplication.run 这个静态方法启动应用本身。

SpringApplication 类代码：

package hello;

import org.springframework.boot.SpringApplication; import org.springframework.boot.autoconfigure. SpringBo-

otApplication;

import org.springframework.web.bind.annotation. Reques- tMapping;

import org.springframework.web.bind.annotation. Respon- seBody;

@SpringBootApplication public class Hello {

@RequestMapping("/test/v1") @ResponseBody

String home() {

return "Hello World";

}

public static void main(String[] args) throws Exception { SpringApplication.run(Hello.class, args);

}

}

上述代码说明了，SpringApplication 可以简单地、很好地启动一个 Spring 应用。对于 Web 应用，Spring Boot 可以通过打包工具将 Tomcat、Jetty 等容器嵌入到可执行的 jar 包内，使jar 包成为独立运行的 Web 应用。

B.外化配置

Spring开发框架中允许将程序的运行参数放置到properties 文件内，在Spring Boot 中，这些可配置的参数的来源扩大了，不在局限于properties 文件，可以来自其他外部环境（比如启动命令行、系统环境变量），这个功能称为外化配置（externalize）。

除了使用 properties 配置文件，还可以使用 YAML 配置文件，配置文件中参数可以系统环境变量、命令行参数等外部环境。命令行参数；

来自于 java:comp/env 的 JNDI 属性；

Java 系统属性（System.getProperties()）；

操作系统环境变量；

只有在random.\*里包含的属性会产生一个RandomValu- ePropertySource；

在打包的 jar 外的应用程序配置文件（application.prop- erties，包含 YAML 和profile 变量）；

在打包的 jar 内的应用程序配置文件（application.prop- erties，包含 YAML 和profile 变量）；

在@Configuration 类上的@PropertySource 注解；

默认属性（使用 SpringApplication.setDefaultProperties

指定）。

下面是一个具体的示例，从外部环境中获取 db.port 的属性（数据库端口）：

代码示例：获取外部资源参数

public class Application { @Value("${dp.port}") private String extResource;

public static void main(String[] args) throws Exception {

SpringApplication.run(Application.class, args);

}

}

Spring Boot 的配置文件有 2 种格式，分别是 properties和YAML。

1. Application 属性文件

SpringApplication会从以下位置加载application.properties

文件，并把它们添加到 Spring Environment 中：

当前目录下的一个/config 子目录；

当前目录；

一个 classpath 下的/config 包；

classpath 根路径（root）。

这个列表是按优先级排序的（列表中位置高的将覆盖位 置低的）。

注：你可以使用 YAML（'.yml'）文件替代'.properties'。

如果不喜欢将 application.properties 作为配置文件名，你可以通过指定spring.config.name环境属性来切换其他的名称。你也可以使用spring.config.location环境属性来引用一个明确的路径（目录位置或文件路径列表以逗号分割）。

1. YAML

YAML 是JSON 的一个超集，也是一种方便的定义层次配置数据的格式。无论你何时将 SnakeYAML 库放到classpath 下， SpringApplication类都会自动支持YAML 作为properties的替换。

从下面的样例代码可知YAML易于人工书写、观察，缺点 是 YAML 文件不能通过@PropertySource 注解加载。

YAML 样例（代码来自 <http://docs.spring.io/>）： nvironments:

dev:

url: [http://dev.bar.com](http://dev.bar.com/) name: Developer Setup

prod:

url: [http://foo.bar.com](http://foo.bar.com/) name: My Cool App对应的 Properties (代码来自 <http://docs.spring.io/>） environments.dev.url=[http://dev.bar.com](http://dev.bar.com/) environments.dev.name=Developer Setup environments.prod.url=[http://foo.bar.com](http://foo.bar.com/) environments.prod.name=My Cool App

1. Profiles

在 Spring Boot 可以通过@Profiles 来限制某些配置类的加载。如图 10 所示，代码中的配置，仅在环境的 Profile 为pro- duction 才被加载，如果Profile 为其他值，此配置类则不会被加载。

通过环境变量控制@Profiles，从而可以灵活控制程序的 加载。

Profiles 示 例 ： @Configuration @Profile("production")

public class ProductionConfiguration {

// ...

}

如下面的代码，在 Application.properties 中的激活 Profile

为：dev, hsqldb。 spring.profiles.active=dev,hsqldb

1. 开发web应用

在传统的 spring MVC、RESTful 开发，需要手工处理XML，进行许多配置，在 Spring Boot，仅需在 maven 配置添加一项：spring-boot-starter-web，如图 1 所示。

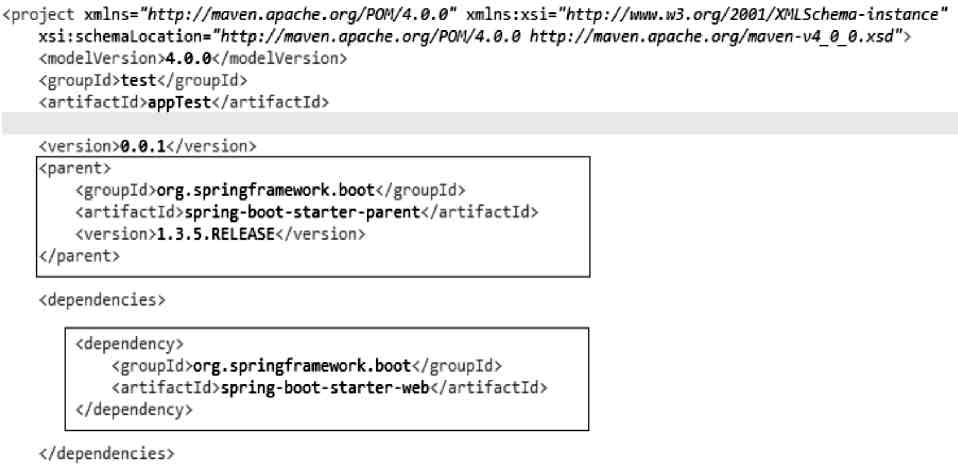


图 1 Spring Boot Web 程序的Maven 配置文件

e)使用JPA操作数据库

在Spring Boot 使用JSP 操作数据库，操作十分简单，仅需要在 Maven 配置加入 pom 项目：spring-boot-starter-data-jpa， 如图 2 所示。

然后定义需要的实体类：Customer，代码如下：

@Entity

public class Customer implements Serializable { @Id

@GeneratedValue private Long id;

@Column(nullable = false)

private String name; @Column(nullable = false) private Integer age;

// ... additional members, often include @OneToMany ma-

ppings

protected Customer() {

// no-args constructor required by JPA spec

// this one is protected since it shouldn't be used directly

}

public Customer(String name, int age) { this.name = name;

this.age = age;

}

public String getName() { return this.name;

}

// ... etc

}

根据业务需要的创建数据接口：CustomerRepository，代码如下：

代码示例，CustomerRepository：

public interface CustomerRepository extends Repository< Customer, Long> {

Page<Customer> findAll(Pageable pageable);

Customer findByNameAndCountryAllIgnoringCase(String name, String country);

}

如图 3 所示，在配置类Application 中对数据接口Custom- erRepository 使用@Bean 注解配置后，Spring Boot 会自动创建接口 CustomerRepository 的实现，在零编码的情况下，就能使用 CustomerRepository 接口。

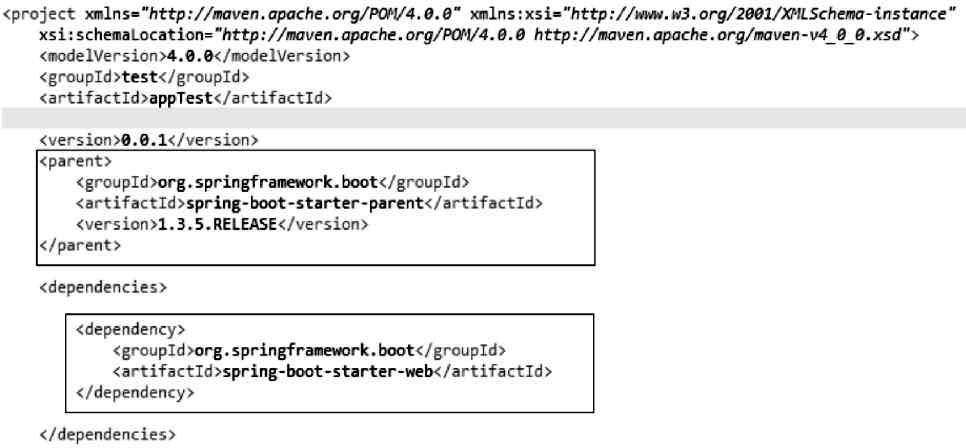


图 2 增加 pom 项：spring-boot-starter-data-jpa

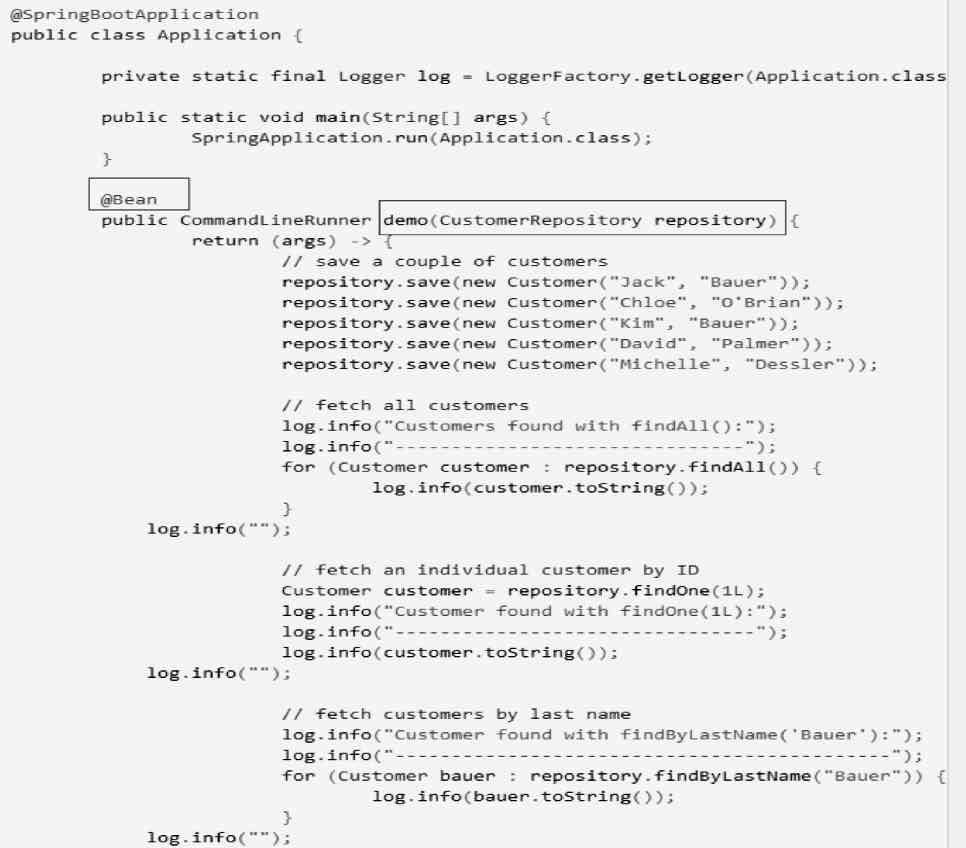


图 3 图 3 实际配置类中零编码的情况下即可使用接口

CustomerRepository

（图 3 来自 <https://spring.io/guides/gs/accessing-data-jpa/）>

C.整合传统的基于XML配置的Spring旧代码

如果要把早期的基于 XML 配置的旧代码整合到 Spring Boot 新代码中，在条件许可的情况下，尽量对旧代码进行改造， 这样可以避免在实际运行中出现问题。

如果旧代码改造成本较高，也可以考虑使用@ImportRe- source 注解引入传统的 XML 配置，XML 配置中的 bean 也能正常使用。

在@ImportResource 注解接入传统的 Spring XML 配置文件，可以解决大部分 bean 的的创建文件，实际使用时仍然存在少量的代码会出现问题，这时需要进行大量、细致 的测试，才能找出有问题的代码并修正问题，这样也能解决问题。

1. SpringBoot的那些坑
2. JSP调试问题

使用内嵌容器时，仅在使用 Tomcat 可以支持JSP（需要增加配置），其他容器不支持 JSP。打包为 war 文件部署到标准容器时，不存在调试问题，所以仅会影响 JSP 调试。目前使用 JSP 开发的的情况越来越少，所以这个问题对开发的影响较小。

1. 注解自动扫描

Spring Boot 仅仅扫描本身工程的源代码内的spring 配置类，不会扫描引用的 jar 包内的进行 spring 配置类对象。对于引用的 jar 包的 spring 配置对象需要用@import 注解进行。

1. 整合传统Spring 的 XML 配置冲突问题

@ImportResource 注解导入 XML 配置，如果在 XML 使用了<context:component-scan>的自动扫描功能，会引发Spring 启动冲突。

解决方法：

在<context: component-scan>加入<exclude-filter>字段排除冲突的配置类（@Configuration），如：

<context: exclude-filter type="assignable" expression=" 类的全路径"/>

**Spring Boot Research and Application (原文)**

Yonghe Wang, Jinsong Zhang, Anming Deng,zhixun Zhou

Abstract: Spring Boot is a framework for simplifying and accelerating the development of Spring. The article briefly introduces the development of Spring Boot, and then integrates the tradition The development of Spring has done exploratory research.

Key words: Spring Boot;Development;Integrating traditional Spring

I.  INTRODUCTION OF SPRINGBOOT

Spring Boot is a new project launched in 2013, which is mainly used to simplify the development, configuration, debugging, and deployment of the Spring development framework. At the same time, it integrates a large number of easy-to-use and practical basic frameworks within the project.

The basic framework integrated in Spring Boot is one that is often used in development, such as embedded containers (Tomcat, Jettty, Undertow), logging framework, JMS framework, persistence framework, popular NOSQL database (Cassandra, MongoDB), caching framework, etc.In a traditional Spring framework, users need to write their own XML files, add corresponding dependency packages in Maven and Gradle, and add functional code in the responding code to use them.Versioning between packages is also difficult when multiple dependent packages are introduced, and versioning conflicts between dependent packages often need to be addressed.Tweaking versions of dependencies and conflicting issues is a time-consuming, repetitive, and tedious task.If you use Spring Boot for development, you only need to add a small amount of configuration in Maven and Gradle configuration files to use the required framework in the code, so that the original configuration is simplified to almost zero code and zero XML configuration. Meanwhile, the version problem of dependent package is easily solved by Spring Boot.

Spring Boot provides good support for the current mainstream build tools Maven and Gradle, as well as for other build tools, such as Ant, but the support is relatively less than Maven and Gradle, so it is necessary to use Maven and Gradle to build the Spring Boot program as much as possible.The build environment for this article is Maven, and you can use Gradle to refer to the official Reference documentation of Spring Boot.

II. A SIMPLE EXAMPLE OF A HELLO WORLD DEVELOPMENT CPMPARISON

1. Develop a Hello World program using traditional Spring

Using traditional Spring to develop a Hello World Web program, the construction of the development project is a relatively tedious process, a typical project, need to modify and configure four files: hello.java, application-xml, servlet-hel-lo.xml, web.xml, even in the IDE run, debugging, or need to configure the J2EE container in order to debug.If you want to do operations and deployment, you also need to package the application as a war file with the IDE, and then you need to configure and deploy the J2EE container, and then deploy the war file to the container to run, so that you can see the operation effect.

1. Develop a Hello World program using Spring Boot

The amount of work required to develop a Hello World program using Spring Boot will be greatly reduced.You only need to modify and configure two files: hello.java and pom.xml. With the help of the built-in container in the Spring Boot framework, you can easily launch the Web program from the IDE.

The core contents of pom. XML are as follows:

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>1.3.5.RELEASE</version>

</parent>

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

</dependencies> Hello.java 代码： package hello;

import org.springframework.boot.SpringApplication; import org.springframework.boot.autoconfigure. SpringBo-

otApplication;

import org.springframework.web.bind.annotation. Reques- tMapping;

import org.springframework.web.bind.annotation. Respon- seBody;

@SpringBootApplication public class Hello { @RequestMapping("/test/v1") @ResponseBody

String home() {

return "Hello World";

}

public static void main(String[] args) throws Exception { SpringApplication.run(Hello.class, args);

}

}

The amount of work required to develop a Hello World program using Spring Boot will be greatly reduced.Only two files need to be modified and configured: it can be seen from the code of pom.xml and hello.java that the code and configuration are greatly simplified by using Spring Boot. A Web program can be completed with just over ten lines of code and Maven configuration.Meanwhile, this web application can be easily packaged as a single runnable jar file hello.jar, using the following command on the command line:

java -jar hello.jar

You can launch a Web application without deploying a J2EE container.The main purpose of packaging Web applications as a single, runnable jar with a J2EE container embedded in it is to simplify the deployment and configuration of the application.In the actual operation and maintenance, this packaging method can well cooperate with the virtual platform of Docker class to achieve the goal of simple and rapid deployment.

III. MAIN FEATURES OF SPRINGBOOT

As a framework that simplifies Spring development, debugging, and deployment, Spring Boot provides many useful features, but here are just some of the main features in development.Other many useful features, such as: log integration, security technology, no technology, message (JMS), email, JTA distributed transaction integration, JMX monitoring and management of technology integration, testing, functional integration, conditions of use of automatic configuration technology, WebSo - ckets development is not described in this article, readers in this paper, on the basis of reference to be given in chapter 6 the extension of the resources of the study.

Spring Boot provides microservices framework development capabilities, except for the deployment of a Web application packaged as a war file.You can also compile your Web application as a stand-alone jar, and you can also embed your J2EE container so that your Web application can also run independently of the container.

A.SpringApplication

Spring Boot provides microservices framework development capabilities, except for the deployment of a Web application packaged as a war file.The Web Spring application class also provides a convenient way to launch a Spring application from the main() method.As shown in the SpringApplication class code below, the Hello class starts the application itself in the main () method by calling the static method springgapplication.run.

SpringApplication class code:

package hello;

import org.springframework.boot.SpringApplication; import org.springframework.boot.autoconfigure. SpringBo-

otApplication;

import org.springframework.web.bind.annotation. Reques- tMapping;

import org.springframework.web.bind.annotation. Respon- seBody;

@SpringBootApplication public class Hello {

@RequestMapping("/test/v1") @ResponseBody

String home() {

return "Hello World";

}

public static void main(String[] args) throws Exception { SpringApplication.run(Hello.class, args);

}

}

The code above illustrates that Spring application can easily and nicely launch a Spring application.For Web applications, Spring Boot can use packaging tools to embed containers such as Tomcat, Jetty, and so on into executable jars to make the jars a stand-alone Web application.

B. Externalized configuration

The Spring development framework allows you to place the running parameters of a program into a properties file. In Spring Boot, these configurable parameters can be sourced from other external environments (such as starting the command line, system environment variables) rather than from a properties file. This is called externalize.

In addition to using the properties profile, you can also use the YAML profile, where the parameters can be external environments such as system environment variables, command-line parameters, and so on.Command line arguments;

JNDI properties from Java :comp/env;

Java System properties (system.getproperties ());

Operating system environment variables;

Only the properties contained in random.\* will produce a randomvalu-epropertysource;

Application profile (application.prop-erties, including YAML and profile variables) outside of the packaged jar;

The application configuration file (application.prop-erties, including the YAML and profile variables) inside the packaged jar;

@propertysource annotation on the @configuration class;

Default attributes (using SpringApplication setDefaultProperties

Specified).

Here is a concrete example of getting the db.port property (database port) from an external environment:

Code example: get external resource parameters

Public class Application {@value ("${dp.port}") private String extResource;

Public static void main(String[] args) throws Exception {

SpringApplication. Run (Application class, args);

}

}

The configuration files for Spring Boot come in two formats, propertieshe and YAML.

1. Application properties file

SpringApplication loads application.properties from the following locations

Files and add them to the Spring Environment:

A /config subdirectory in the current directory;

Current directory;

A /config package under classpath;

The classpath root path.

The list is sorted by priority (the higher in the list overrides the lower).

Note: you can use the YAML ('.yml') file instead of '.properties'.

If you don't like application.properties as the configuration file name, you can switch other names by specifying the spring.config.name environment property.You can also use the spring.config.location environment attribute to refer to an explicit path (the list of directory locations or file paths is separated by commas).

1. YAML

YAML is a superset of JSON and a convenient format for defining hierarchical configuration data.Whenever you place the SnakeYAML library under the classpath, the SpringApplication class automatically supports YAML as a replacement for properties.

As you can see from the sample code below, YAML is easy to write and observe manually. The drawback is that the YAML file cannot be loaded with the @propertysource annotation.

Sample YAML (code from http://docs.spring.io/) : nvironments:

Dev:

Url: http://dev.bar.com name: Developer Setup

The prod.

Url: http://foo.bar.com the name:My Cool App corresponding to the Properties (code from http://docs.spring.io/) environments.dev.url=http://dev.bar.com environments. Dev. Name =Developer Setup environments.prod.url=http://foo.bar.comEnvironments. Prod. Name = My Cool App

1. Profiles

In Spring Boot, you can limit the loading of certain configuration classes by @profiles.As shown in figure 10, the configuration in the code is loaded only if the Profile of the environment is pro-duction, and the configuration class is not loaded if the Profile is any other value.

With the environment variable control @profiles, you can flexibly control the loading of the program.

Profiles example: @configuration @profile ("production")

Public class ProductionConfiguration {

/ /...

}

The following code activates the Profile in application.properties

Is: dev, HSQLDB.Spring. Profiles. The active = dev, HSQLDB

1. Develop web applications

In traditional spring MVC, RESTful development, you need to manually process XML and do many configurations. In spring Boot, you only need to add one item in maven configuration: spring-boot-starter-web, as shown in figure 1.

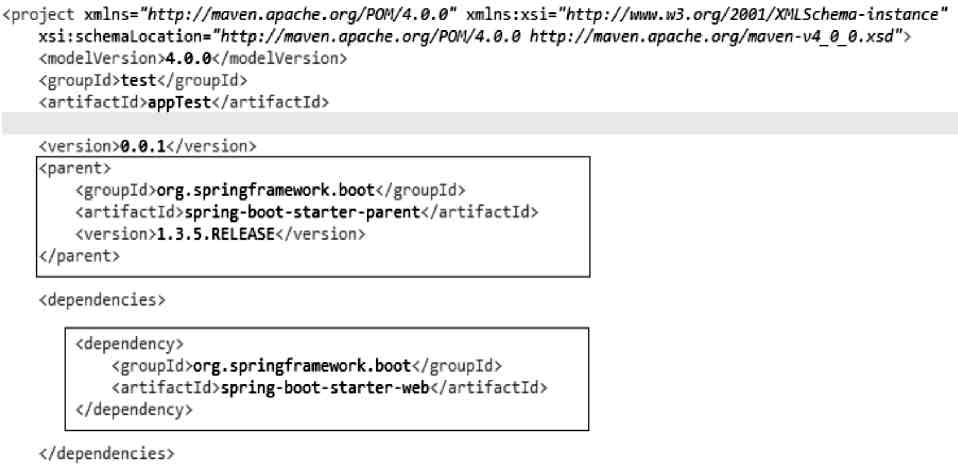


Figure 1. Maven configuration file for the Spring Boot Web program

e)Use JPA to manipulate the database

Using JSPS to operate the database in Spring Boot is simple, just add the pom project in the Maven configuration: spring-boot-starter-data-jpa, as shown in figure 2.

Then define the required entity class: Customer, and the code is as follows:

@ the Entity

Public class Customer implements Serializable {@id

@ GeneratedValue private Long id;

@ the Column (nullable = false)

Private String name;Column(nullable = false) private Integer age;

/ /...Additional members, often include @onetomany ma-

ppings

Protected the Customer () {

// no-args constructor required by JPA spec

// this one is protected since it shouldn't be used directly

}

Public Customer(String name, int age) {this.name = name;

Enclosing the age = age;

}

Public String getName() {return this.name;

}

/ /...The etc

}

Create the data interface according to the business needs: CustomerRepository, the code is as follows:

Code sample, CustomerRepository:

Public interface CustomerRepository extends Repository< Customer, Long> {

Page < Customer > the.findall (Pageable Pageable);

The Customer findByNameAndCountryAllIgnoringCase (String name, String country);

}

As shown in figure 3, after the @bean annotation configuration of the data interface custom-errepository in the configuration class Application, Spring Boot automatically creates an implementation of the interface CustomerRepository, which can be used with zero encoding.

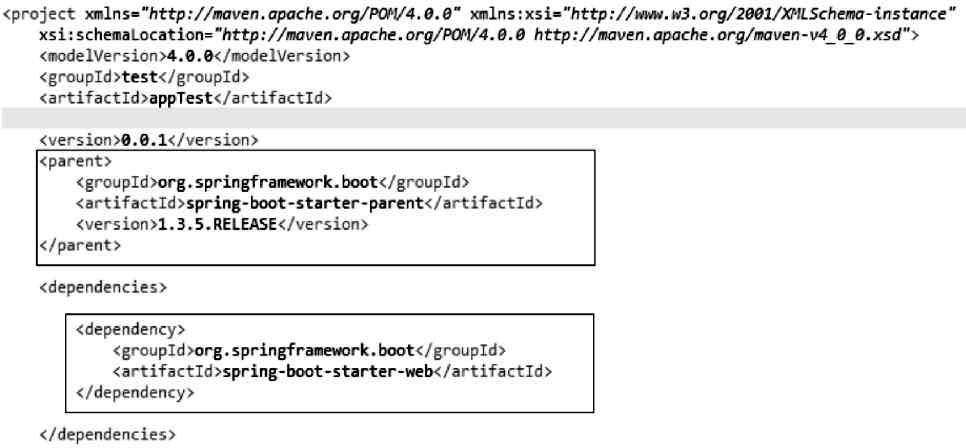


Figure 2 adds the pom entry: spring-boot-starter-data-jpa

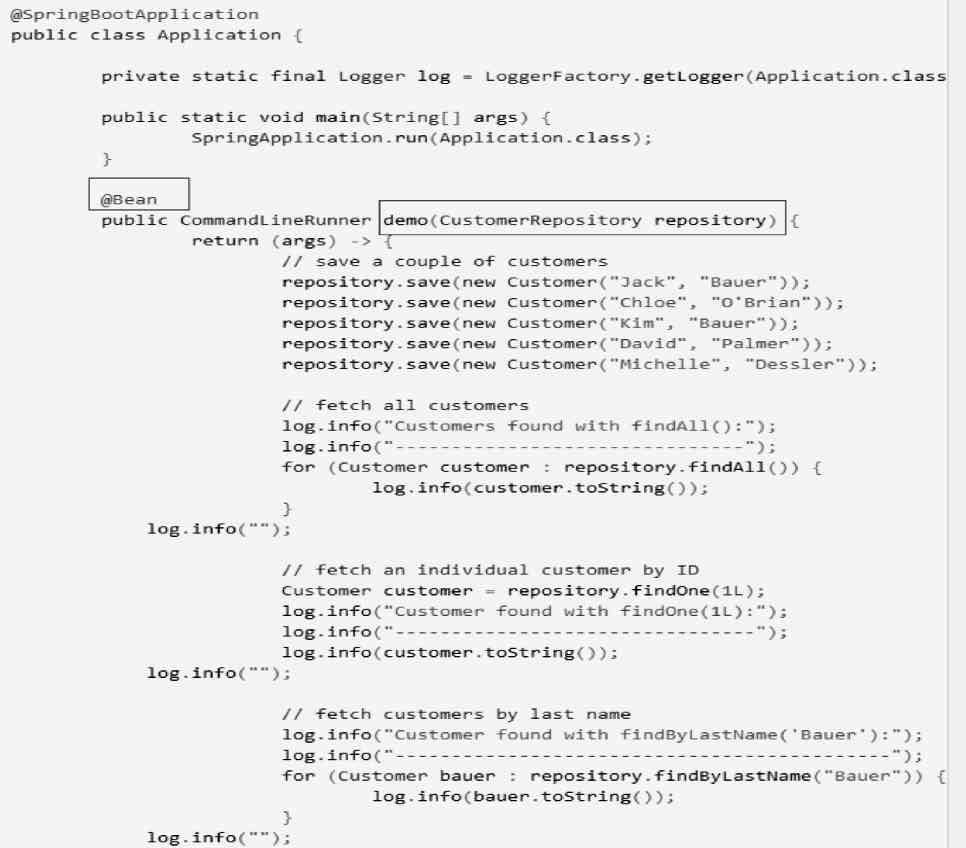


Figure 3 figure 3 can be used with zero encoding in the actual configuration class

CustomerRepository

(FIG. 3 from https://spring.io/guides/gs/accessing-data-jpa/)

C.Integrate traditional Spring old code based on XML configuration

If you want to incorporate old code from earlier XML configurations into your new Spring Boot code, try to rework the old code as much as you can to avoid problems in the real world.

If the old code is expensive to retrofit, you can also consider using the @importre-source annotation to introduce the traditional XML configuration, and beans in the XML configuration will work just fine.

Accessing the traditional Spring XML configuration file with the @importresource annotation solves the problem of creating most of the beans, but there is still a small amount of code that can cause problems when it is actually used.

IV. THE PITS IN SPRINGBOOT

Debug the problem

With embedded containers, JSPS are supported only when Tomcat is used (additional configuration is required), other containers do not support JSPS.When packaged as a war file and deployed to a standard container, there is no debugging problem, so only JSP debugging is affected.Currently, JSP development is less and less common, so this issue has less impact on development.

A. automatic annotation scanning

Spring Boot simply scans the Spring configuration classes in the source code of the project itself, not the Spring configuration class objects in the referenced jar.The spring configuration object for the referenced jar package needs to be annotated with the @import annotation.

B. Integrate traditional Spring's XML configuration conflicts

The @importresource annotation imports the XML configuration, causing a Spring startup conflict if the automatic scan function of <context:component-scan> is used in the XML.

Solutions:

Insert the Configuration class (@configuration) in the <context: component-scan> to exclude conflicts in the <exclude-filter> field, such as:

<context: exclude-filter type="assignable" expression=" full path of class "/>