题目：超市订单管理系统

# 1 系统需求分析

超市订单管理系统是一个专为连锁店、超市等商业场所提供订单管理平台的系统。该系统的目标是建立一个订单管理平台，为需要合理规划超市供应链、供应商以及工作人员提供的便捷的平台。该系统的主要业务需求包括记录并维护某超市的供应商信息，以及该超市与供应商之间的交易订单信息，包括三种角色，系统管理员经理，普通员工。

## 1.1 系统功能分析

本系统主要的功能是实现超市订单管理功能，以便为超市、连锁店提供以及其他负责人提供订单详情、联系方式等，系统的主要功能有以下五个方面：

（1）登录/注销：

管理员可以在网站上登录浏览，离开时注销并退出。

（2）订单管理：

管理员可以浏览所有订单信息，并且通过点击查看了解订单详情信息。

（3）供应商管理：

管理员可以在网站浏览所有供应商信息，并在在与其他供应商达成合作之后，添加相关供应商信息，并且通过点击查看了解他们的联系方式等。

（4）用户管理：

管理员可以管理所有超市员工用户，对用户进行增删改查，对于离职或其他原因的未工作用户给予注销管理。

（5）密码修改：

管理员可对自己的账号密码进行修改，填写对应之前的正确密码以及新密码之后，即完成相关修改密码操作。

（6）搜索功能：

在以上管理界面中，均允许了管理员根据关键字进行搜索，要求搜索框中输入的字段必须完全包含在物品名称中，否则无法查询。

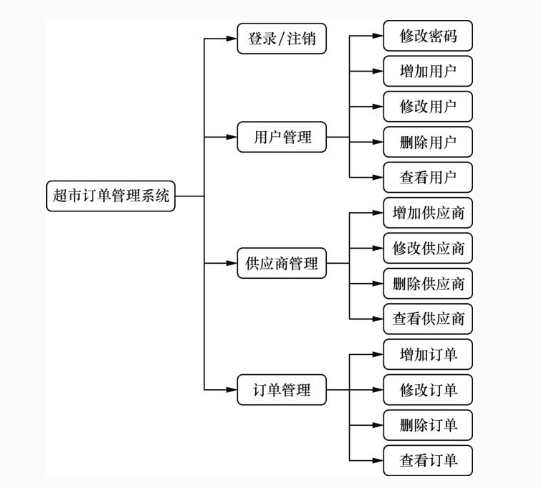


图1.1 超市订单管理系统

## 1.2 系统功能需求

根据系统功能要求，该超市订单管理系统以管理员为中心的用户角色，可以将系统分解成几个模块来分别设计应用程序界面，如图 1.1所示。

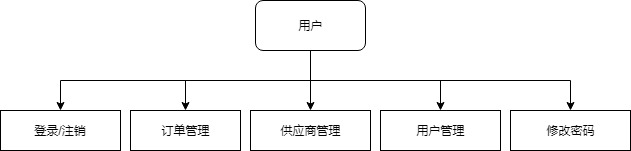


图1.2 用户功能模块结构图

## 1.3 系统性能需求

超市订单管理系统的开发是在Window10平台上，以SSM为架构，采用MySQL 作为数据库管理系统管理后台数据库。本系统是超市信息管理建设中必不可少的一部分，它实现了现代管理信息系统的大部分功能需要。使用本系统可以使超市管理更加方便快捷，合理的页面设计也使得这个用户充分享受到基于Internet管理信息系统的优越。本系统开发说明：

* 功能完备

在开发初期，查看了大量关于电子商务，管理信息系统，J2EE等方面的资料，同时借鉴了很多其他电子商务网站和管理信息的流程。经过总结，确定了满足需求分析的基本模块。系统总体设计上实现了整个系统模块的划分，系统主要包含5大模块，分别是：订单管理信息，供应商管理，用户管理，修改密码，登陆退出系统，基本上实现了综合管理系统的所有功能。

* 界面友好

系统用户登陆到管理页面后，每页有导航和引领的作用。系统具有自适应的能力，同时导航条方便快捷的引导用户进行各种合理的操作。

* 管理科学

本系统一开始就从管理学的角度做出了详细细致的考虑，后来有参考了电子商务管理等，最后才做出了系统总体设计，因此可以讲该系统是较为科学的。

系统的性能需求主要表现在数据库中的各个表需要频繁地被插入、删除以及更新。对于用户来说，系统地响应时间不宜太长，否则会降低用户体验。为此要求我们建立良好的表结构，加上足够的存储空间以及硬件性能。

# 2 可行性分析

## 2.1 研究前提

随着我国经济情况的日新月异，飞速发展，涌现出许许多多的超市和便利店。越来越多的人喜欢到超市购物，超市里销售的商品也呈现出多种多样的变化趋势。我们开发一个超市订单管理系统，它可以对仓储各环节实施全过程控制管理，对整个进货、退货、盘点等各个环节的规范化作业，控制整个过程的正常运行。去掉了手工书写票据和送到机房输入的步骤，解决库房信息陈旧滞后的弊病，方便了仓库管理人员对物品的放置和调配，提高了工作效率。

该系统容易被接受，具有简单易学性，便于管理等功能，是对超市订单管理的一种有效工具。

## 2.2 设计要求

### 2.2.1 安全性

超市订单管理增强对产品规范的审计，重点确定该项目中需要审计的产品。买家只能针对卖家允许公开的信息进行查阅。买家只享受对自己账号内数据的查阅权，与定后处理权，订货支付权，申请退货权，不允许偷窥其他人。卖家只能针对买家允许公开的信息进行查阅。卖家只享受对自己账号内数据的查阅权，发货权，退款相应处理权，不允许偷窥其他人。

### 2.2.2 系统性能

管理员登录查看超市供应商与超市员工用户管理，可以进行增、删、改、查等操作。超市订单系统可以使超市的管理趋于正规化、现代化和系统化。本项目的产品可以达到以下目标：

（1）提高工作效率，减少返工。

（2）业务流程的流水线化。

（3）符合相关标准和规则。

（4）与目前的应用产品相比较，提高了可用性或减少了失效程度。

### 2.2.3 可扩展性

所有信息呈现，操作完全由打开的网页呈现并完成。本系统所占有的是超市市场，它追求的是简单、易学、易用，能够更好地解决管理人员的负担，能够辅助超市有效的管理物品。对于订单管理系统的用户，可满足对订单管理的需求，且此种需求被接受并且满足，其系统便可以推广。

# 3 数据库设计

## 3.1 数据库需求分析

经过对超市管理系统的调查分析，得出用户的需求大致如下：

（1）管理员可以在系统中对订单、供应商以及用户进行增、删、改、查的处理。

（2）管理员需要输入账号密码登录，并且可以增添新的管理员。

如下是利用数据流图方法对数据库做需求分析：

第一步：由用户的需求，可以得到顶层数据流图如图3.1.1所示。

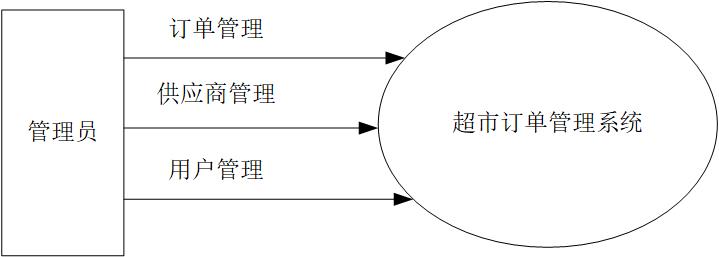


图3.1.1 超市订单管理系统的顶层数据流图

第二步：超市订单管理系统的第1层数据流图如图3.1.2所示。



图3.1.2 超市订单管理系统的第1层数据流图

第三步：超市订单管理系统的第2层数据库流图——订单管理的细化数据流图如图3.1.3所示。

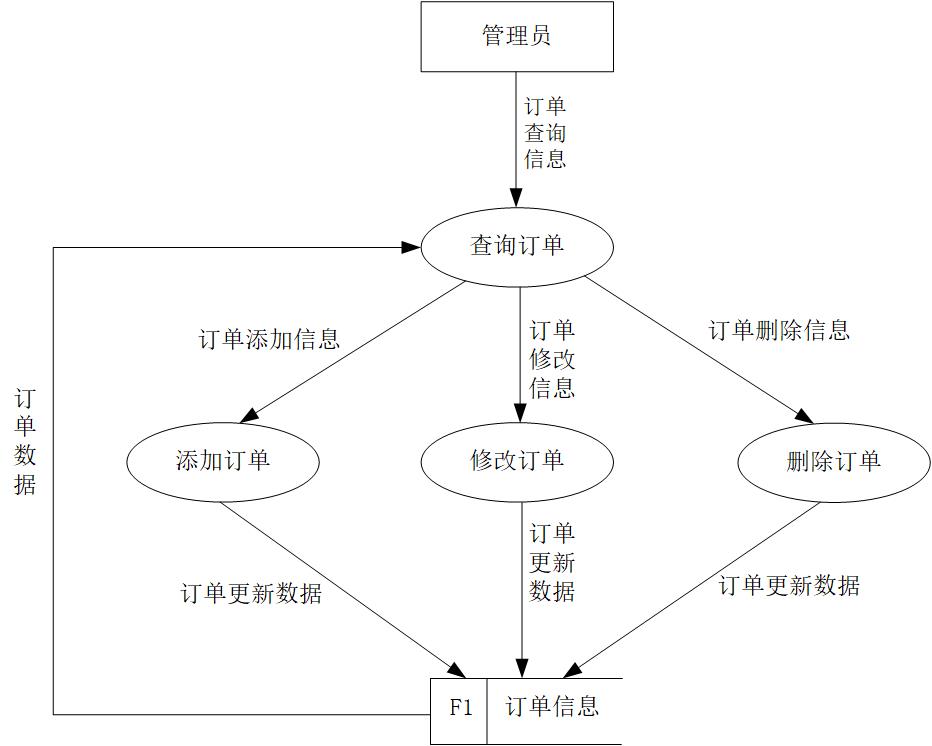


图3.1.3 超市订单管理系统的第2层数据库流图——订单管理的细化数据流图

第四步：超市订单管理系统的第2层数据流库——供应商管理的细化数据流图如图3.1.4所示。

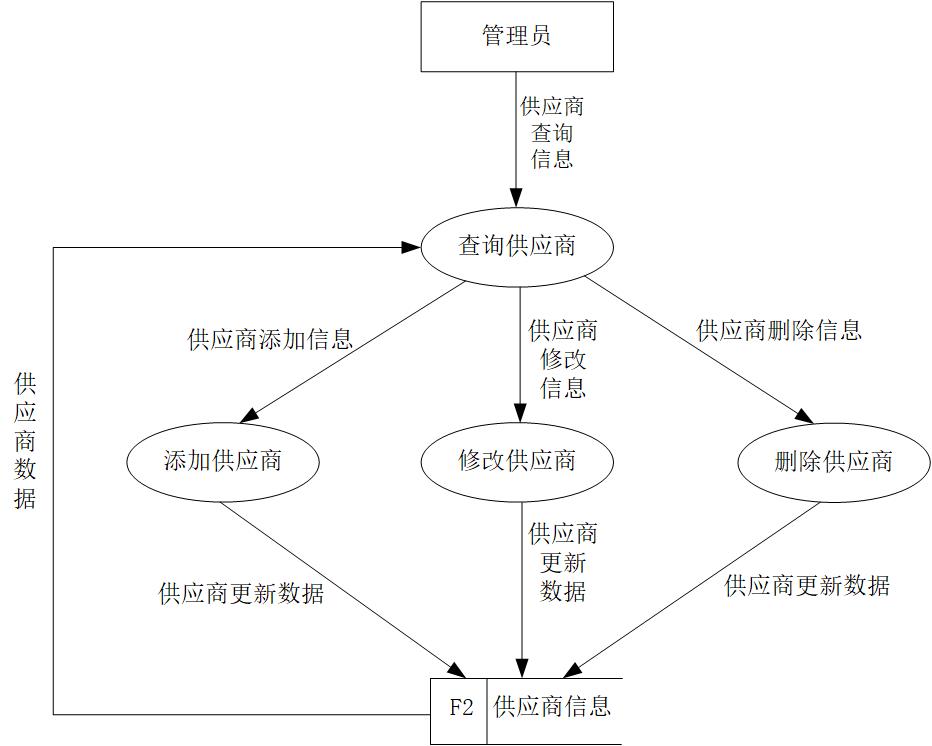


图3.1.4 超市订单管理系统的第2层数据库流图——供应商管理的细化数据流图

第五步超市订单管理系统的第2层数据流库——用户管理的细化数据流图如图3.1.5所示。

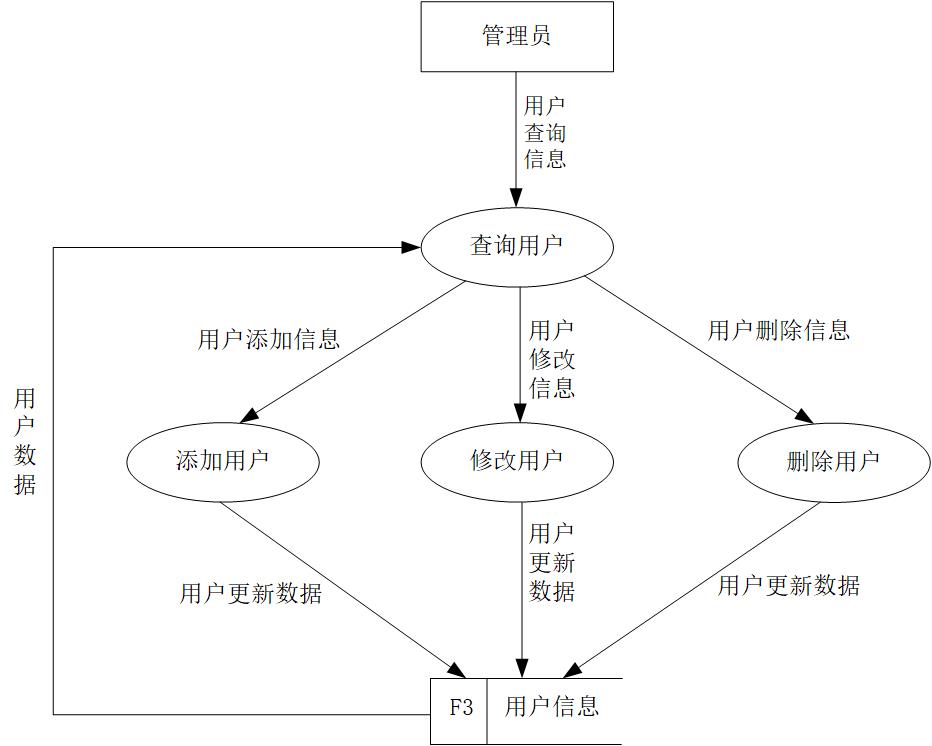


图3.1.5 超市订单管理系统的第2层数据库流图——用户管理的细化数据流图

根据如上的数据流程图，可以列出以下记录超市订单管理所需的数据项和数据结构：

·管理员：管理员ID、管理员姓名、管理员密码、管理员性别、管理员角色、管理员出生日期、管理员电话、管理员住址。

·订单：订单编码、商品名称、供应商名称、订单金额、是否付款。

·供应商：供应商编码、供应商名称、联系人、联系电话、微信。

## 3.2 数据库概念结构设计

本系统一共有用户、供应商、订单、角色、地址这五个基本实体。

管理员可以对应多个订单，而一个订单只能对应于一个管理员。管理员可以管理多个供应商，而一个供应商只能对应于一个管理员。一个供应商可以对应多条订单，但一条订单只能对应于一个供应商。此外，有一个用户对应一个角色，一个角色对应多个用户；一个地址对应多个订单，一个订单对应一个地址。数据库表之间的关系如下：



图3.2.1 数据库E-R图分析

（1）用户：主键ID、用户编码、用户名称、用户密码、性别、出生日期、手机、地址、用户角色、创建者、创建时间、更新者、更新时间、用户头像、工作照。

（2）账单：订单编号、订单编码、商品名称、商品描述、商品单位、商品数量、商品总额、是否支付、创建者、创建时间、更新者、更新时间、供应商ID。

（3）供应商：供应商ID、供货商编码、供货商名称、供应商详细描述、供应商联系人、联系电话、地址、微信、创建者、创建时间、更新时间、更新者、营业执照、组织机构代码证。

（4）地址：主键ID、联系人姓名、收货地址明细、邮编、联系人电话、创建者、创建日期、修改者、修改时间、用户ID。

（5）角色：角色编号、角色编码、角色名称、创建者、创建时间、修改者、修改时间。

## 3.3 数据库逻辑结构设计

将概念结构设计中的各个模型转化为DBMS支持的表结构，同时保持不会出现插入异常、删除异常和修改异常，表结构应该做到符合3NF。根据系统 E-R 图，需要设计4个数据表来存放信息。在本系统中，一共有五个实体，实体转化为数据库模型为如下所示：

1、一对多联系转化为一个关系模式

用户--订单(用户编号,订单编号)

供货商--订单(供货商编号,订单编号)

用户--身份（用户编号，身份编号）

用户--地址（用户编号）

2、利用以上关系模式得到的所有数据表如下所示：

（1）用户表（smbms\_user）：

·数据项：主键ID、用户编码、用户名称、用户密码、性别、出生日期、手机、地址、用户角色、创建者、创建时间、更新者、更新时间、用户头像、工作照。

·说明：用户ID是唯一的用户标识，使此表的主键。如表3.3.1所示。

表3.3.1 用户表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 数据长度 | 可否为空 | 备注 |
| Id | bigint | 20 | Not null | 主键ID |
| userCode | varchar | 15 | Not null | 用户编码 |
| userName | varchar | 15 | Not null | 用户名称 |
| userPassword | varchar | 15 | Not null | 用户密码 |
| gender | int | 10 |  | 性别 |
| birthday | date |  |  | 出生日期 |
| phone | varchar | 15 |  | 手机 |
| address | varchar | 30 |  | 地址 |
| userRole | int | 10 |  | 用户角色 |
| createdBy | bigint | 20 |  | 创建者 |
| creationDate | datetime |  |  | 创建时间 |
| modifyBy | bigint | 20 |  | 更新者 |
| modifyDate | datetime |  |  | 更新时间 |
| idPicPath | varchar | 300 |  | 用户头像 |
| workPicPath | varchar | 300 |  | 工作照 |

（2）供应商表（smbms\_provider）：

·数据项：供应商ID、供货商编码、供货商名称、供应商详细描述、供应商联系人、联系电话、地址、微信、创建者、创建时间、更新时间、更新者、营业执照、组织机构代码证。

·说明：这张表标识的是超市管理信息系统中商品供应商的信息列表，供应商ID是该表的主键。

·编号方法：商品供应商ID采用自动生成方式，如表3.3.2所示。

表3.3.2 供货商表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 数据长度 | 可否为空 | 备注 |
| Id | Bigint | 20 | Not null | 供货商ID（主键） |
| proCode | Varchar | 20 | Not null | 供货商编码 |
| proName | varchar | 20 | Not null | 供货商名称 |
| ProDesc | varchar | 50 |  | 供应商详细描述 |
| proContact | varchar | 20 | Not null | 供货商联系人 |
| proPhone | Varchar | 20 | Not null | 联系电话 |
| ProAddress | Varchar | 50 | Not null | 供货商地址 |
| proFax | varchar | 20 |  | 微信 |
| CreateBy | bigint | 20 |  | 创建者 |
| CreatationDate | datetime |  |  | 创建时间 |
| modifyDate | datetime |  |  | 更新时间 |
| modifyBy | bigint | 20 |  | 更新者 |
| companyLicPicPath | varchar | 300 |  | 营业执照 |
| orgCodePicPath | varchar | 300 |  | 组织机构代码证 |

（3）订单表（smbms\_bill）：

·数据项：订单编号、订单编码、商品名称、商品描述、商品单位、商品数量、商品总额、是否支付、创建者、创建时间、更新者、更新时间、供应商ID。

·说明：这张表标识的是超市管理信息系统订单信息列表，订单ID是该表的主键。

·编号方法：订单ID采用自动生成方式，供应商ID与供应商表中供应商ID一一对应，如表3.3.3所示。

表3.3.3 订单表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 数据长度 | 可否为空 | 备注 |
| Id | bigint | 20 | Not null | 订单ID（主键） |
| billCode | varchar | 20 | Not null | 订单编码 |
| ProductName | Varchar | 20 | Not null | 商品名称 |
| ProductDescent | Varchar | 50 | Not null | 商品描述 |
| ProductUnit | Varchar | 10 | Not null | 商品单位 |
| ProductCount | Decimal | 20,2 | Not null | 商品数量 |
| totalPrice | Decimal | 20,2 | Not null | 商品总额 |
| isPayment | int | 10 | Not null | 是否支付 |
| createdBy | bigint | 20 |  | 创建者 |
| creationDate | Datetime |  |  | 创建时间 |
| modifyBy | bigint | 20 |  | 更新者 |
| modifyDate | datetime |  |  | 更新时间 |
| providerID | Int | 20 |  | 供应商ID |

（4）身份表（smbms\_role）：

·数据项：角色编号、角色编码、角色名称、创建者、创建时间、修改者、修改时间。

·说明：这张表标识的是超市订单管理信息系统中用户身份列表，身份编号是该表的主键。

·编号方法：用户身份编号与用户表中的员工身份编号一一对应，如表3.3.4所示。

表3.3.4 身份表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 数据长度 | 可否为空 | 备注 |
| Id | bigint | 20 | Not null | 角色ID（主键） |
| RoleCode | varchar | 15 | Not null | 角色编码 |
| roleName | Varchar | 15 | Not null | 角色名称 |
| createdBy | bigint | 20 |  | 创建者 |
| creationDate | datetime |  |  | 创建时间 |
| modifyBy | bigint | 20 |  | 修改者 |
| modifyDate | datetime |  |  | 修改时间 |

（5）地址表（smbms\_address）：

* 数据项：主键ID、联系人姓名、收货地址明细、邮编、联系人电话、创建者、创建日期、修改者、修改时间、用户ID。
* 编号方法：用户ID与用户表中的用户ID一一对应，如表3.3.5所示。

表3.3.5 地址表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 数据长度 | 可否为空 | 备注 |
| Id | bigint | 20 | Not null | 主键ID（主键） |
| Contact | varchar | 15 | Not null | 联系人姓名 |
| addressDesce | Varchar | 50 | Not null | 收货地址明细 |
| postcode | Varchar | 15 |  | 邮编 |
| Tel | Varchar | 20 | Not null | 联系人电话 |
| createdBy | bigint | 20 |  | 创建者 |
| creationDate | Datetime |  |  | 创建时间 |
| modifyBy | bigint | 20 |  | 修改者 |
| modifyDate | datetime |  |  | 修改时间 |
| userID | Bigint | 20 |  | 用户ID |

数据库连接利用了SSM框架的底层的MyBatis，建立了实体类与MySQL之间映射关系，从而实现数据持久化、封装数据库连接等操作。

## 3.4 数据库物理结构设计

1、选择关系模式的存取方式

对数据库逻辑结构设计中建立的表结构，供应商表的供应商编号属性唯一决定每一个供应商元组，所以对供应商表建立以供应商编号为主关键字的索引。同理，对管理员关系模式、订单关系模式也采用类似的索引存取方法。

2、数据表存储结构设计

本系统的所有数据表均存放在物理磁盘中。用户表、供应商表和订单表的结构是相对稳定的，表中的已有记录是要长期保存的，在此基础上系统会相应用户的操作对数据表进行增、删、改、查等操作。

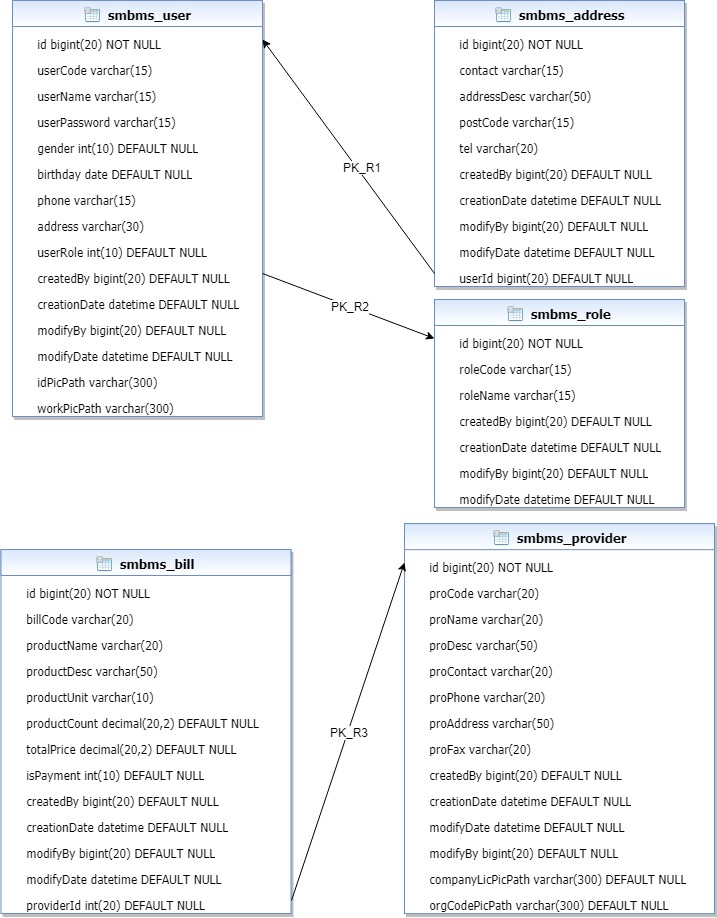


图3.4.1 超市订单管理系统数据库表之间的关系

## 3.5 数据库的建立

### 3.5.1 数据库的建立

（1）创建数据库

create database smbms;

USE smbms;

（2）创建表smbms\_address

DROP TABLE IF EXISTS `smbms\_address`;

CREATE TABLE `smbms\_address` (

`id` bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '主键ID',

`contact` varchar(15) COLLATE utf8\_unicode\_ci NOT NULL COMMENT '联系人姓名',

`addressDesc` varchar(50) COLLATE utf8\_unicode\_ci NOT NULL COMMENT '收货地址明细',

`postCode` varchar(15) COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '邮编',

`tel` varchar(20) COLLATE utf8\_unicode\_ci NOT NULL COMMENT '联系人电话',

`createdBy` bigint(20) DEFAULT NULL COMMENT '创建者',

`creationDate` datetime DEFAULT NULL COMMENT '创建时间',

`modifyBy` bigint(20) DEFAULT NULL COMMENT '修改者',

`modifyDate` datetime DEFAULT NULL COMMENT '修改时间',

`userId` bigint(20) DEFAULT NULL COMMENT '用户ID',

PRIMARY KEY (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=1 DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;

（3）创建表smbms\_bill

DROP TABLE IF EXISTS `smbms\_bill`;

CREATE TABLE `smbms\_bill` (

`id` bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '主键ID',

`billCode` varchar(20) COLLATE utf8\_unicode\_ci NOT NULL COMMENT '账单编码',

`productName` varchar(20) COLLATE utf8\_unicode\_ci NOT NULL COMMENT '商品名称',

`productDesc` varchar(50) COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '商品描述',

`productUnit` varchar(10) COLLATE utf8\_unicode\_ci NOT NULL COMMENT '商品单位',

`productCount` decimal(20,2) NOT NULL COMMENT '商品数量',

`totalPrice` decimal(20,2) NOT NULL COMMENT '商品总额',

`isPayment` int(10) NOT NULL COMMENT '是否支付（1：未支付 2：已支付）',

`createdBy` bigint(20) DEFAULT NULL COMMENT '创建者（userId）',

`creationDate` datetime DEFAULT NULL COMMENT '创建时间',

`modifyBy` bigint(20) DEFAULT NULL COMMENT '更新者（userId）',

`modifyDate` datetime DEFAULT NULL COMMENT '更新时间',

`providerId` int(20) DEFAULT NULL COMMENT '供应商ID',

PRIMARY KEY (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=1 DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;

（4）创建表smbms\_provider

DROP TABLE IF EXISTS `smbms\_provider`;

CREATE TABLE `smbms\_provider` (

`id` bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '主键ID',

`proCode` varchar(20) COLLATE utf8\_unicode\_ci NOT NULL COMMENT '供应商编码',

`proName` varchar(20) COLLATE utf8\_unicode\_ci NOT NULL COMMENT '供应商名称',

`proDesc` varchar(50) COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '供应商详细描述',

`proContact` varchar(20) COLLATE utf8\_unicode\_ci NOT NULL COMMENT '供应商联系人',

`proPhone` varchar(20) COLLATE utf8\_unicode\_ci NOT NULL COMMENT '联系电话',

`proAddress` varchar(50) COLLATE utf8\_unicode\_ci NOT NULL COMMENT '地址',

`proFax` varchar(20) COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '微信',

`createdBy` bigint(20) DEFAULT NULL COMMENT '创建者（userId）',

`creationDate` datetime DEFAULT NULL COMMENT '创建时间',

`modifyDate` datetime DEFAULT NULL COMMENT '更新时间',

`modifyBy` bigint(20) DEFAULT NULL COMMENT '更新者（userId）',

`companyLicPicPath` varchar(300) DEFAULT NULL COMMENT '营业执照',

`orgCodePicPath` varchar(300) DEFAULT NULL COMMENT '组织机构代码证',

PRIMARY KEY (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=1 DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;

（5）创建表smbms\_role

DROP TABLE IF EXISTS `smbms\_role`;

CREATE TABLE `smbms\_role` (

`id` bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '主键ID',

`roleCode` varchar(15) COLLATE utf8\_unicode\_ci NOT NULL COMMENT '角色编码',

`roleName` varchar(15) COLLATE utf8\_unicode\_ci NOT NULL COMMENT '角色名称',

`createdBy` bigint(20) DEFAULT NULL COMMENT '创建者',

`creationDate` datetime DEFAULT NULL COMMENT '创建时间',

`modifyBy` bigint(20) DEFAULT NULL COMMENT '修改者',

`modifyDate` datetime DEFAULT NULL COMMENT '修改时间',

PRIMARY KEY (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=1 DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;

（6）创建表smbms\_user

DROP TABLE IF EXISTS `smbms\_user`;

CREATE TABLE `smbms\_user` (

`id` bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '主键ID',

`userCode` varchar(15) COLLATE utf8\_unicode\_ci NOT NULL COMMENT '用户编码',

`userName` varchar(15) COLLATE utf8\_unicode\_ci NOT NULL COMMENT '用户名称',

`userPassword` varchar(15) COLLATE utf8\_unicode\_ci NOT NULL COMMENT '用户密码',

`gender` int(10) DEFAULT 2 COMMENT '性别（1:女、 2:男）',

`birthday` date DEFAULT NULL COMMENT '出生日期',

`phone` varchar(15) COLLATE utf8\_unicode\_ci NOT NULL COMMENT '手机',

`address` varchar(30) COLLATE utf8\_unicode\_ci NOT NULL COMMENT '地址',

`userRole` int(10) DEFAULT NULL COMMENT '用户角色（取自角色表-角色id）',

`createdBy` bigint(20) DEFAULT NULL COMMENT '创建者（userId）',

`creationDate` datetime DEFAULT NULL COMMENT '创建时间',

`modifyBy` bigint(20) DEFAULT NULL COMMENT '更新者（userId）',

`modifyDate` datetime DEFAULT NULL COMMENT '更新时间',

`idPicPath` varchar(300) COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '用户头像',

`workPicPath` varchar(300) COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '工作照',

PRIMARY KEY (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=11 DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;

### 3.5.2 初始数据的输入

数据表创建成功后，数据库中还没有实际的数据。为了保证外部键能使用，数据需要提前输入，如用户编码、用户姓名、订单名称和供应商等等。具体插入语句如下：

(1)向smbms\_address表插入数据

insert into `smbms\_address`(`id`,`contact`,`addressDesc`,`postCode`,`tel`,`createdBy`,`creationDate`,`modifyBy`,`modifyDate`,`userId`) values

(1,'王丽','北京市东城区东交民巷44号','100010','13678789999',1,'2020-04-13 00:00:00',NULL,NULL,1),

(2,'张红丽','北京市海淀区丹棱街3号','100000','18567672312',1,'2020-04-13 00:00:00',NULL,NULL,1),

(3,'任志强','北京市东城区美术馆后街23号','100021','13387906742',1,'2020-04-13 00:00:00',NULL,NULL,1),

(4,'曹颖','北京市朝阳区朝阳门南大街14号','100053','13568902323',1,'2020-04-13 00:00:00',NULL,NULL,2),

(5,'李慧','北京市西城区三里河路南三巷3号','100032','18032356666',1,'2020-04-13 00:00:00',NULL,NULL,3),

(6,'王国强','北京市顺义区高丽营镇金马工业区18号','100061','13787882222',1,'2020-04-13 00:00:00',NULL,NULL,3);

(2)向smbms\_bill表插入数据

insert into `smbms\_bill`(`id`,`billCode`,`productName`,`productDesc`,`productUnit`,`productCount`,`totalPrice`,`isPayment`,`createdBy`,`creationDate`,`modifyBy`,`modifyDate`,`providerId`) values

(1,'BILL2016\_001','洗发水、护发素','日用品-洗发、护发','瓶','500.00','25000.00',2,1,'2020-06-14 13:02:03',NULL,NULL,13),

(2,'BILL2016\_002','香皂、肥皂、药皂','日用品-皂类','块','1000.00','10000.00',2,1,'2020-03-23 04:20:40',NULL,NULL,13),

(3,'BILL2016\_003','大豆油','食品-食用油','斤','300.00','5890.00',2,1,'2020-05-14 13:02:03',NULL,NULL,6),

(4,'BILL2016\_004','橄榄油','食品-进口食用油','斤','200.00','9800.00',2,1,'2020-04-10 03:12:13',NULL,NULL,7),

(5,'BILL2016\_005','洗洁精','日用品-厨房清洁','瓶','500.00','7000.00',2,1,'2020-05-14 13:02:03',NULL,NULL,9),

(6,'BILL2016\_006','美国大杏仁','食品-坚果','袋','300.00','5000.00',2,1,'2020-04-14 06:08:09',NULL,NULL,4),

(7,'BILL2016\_007','沐浴液、精油','日用品-沐浴类','瓶','500.00','23000.00',1,1,'2020-07-01 10:10:22',NULL,NULL,14),

(8,'BILL2016\_008','不锈钢盘碗','日用品-厨房用具','个','600.00','6000.00',2,1,'2020-04-14 05:12:13',NULL,NULL,14),

(9,'BILL2016\_009','塑料杯','日用品-杯子','个','350.00','1750.00',2,1,'2020-02-04 11:40:20',NULL,NULL,14),

(10,'BILL2016\_010','豆瓣酱','食品-调料','瓶','200.00','2000.00',2,1,'2020-01-29 05:07:03',NULL,NULL,8),

(11,'BILL2016\_011','海之蓝','饮料-国酒','瓶','50.00','10000.00',1,1,'2020-04-14 16:16:00',NULL,NULL,1),

(12,'BILL2016\_012','芝华士','饮料-洋酒','瓶','20.00','6000.00',1,1,'2020-06-09 17:00:00',NULL,NULL,1),

(13,'BILL2016\_013','长城红葡萄酒','饮料-红酒','瓶','60.00','800.00',2,1,'2020-04-14 15:23:00',NULL,NULL,1),

(14,'BILL2016\_014','泰国香米','食品-大米','斤','400.00','5000.00',2,1,'2020-05-09 15:20:00',NULL,NULL,3),

(15,'BILL2016\_015','东北大米','食品-大米','斤','600.00','4000.00',2,1,'2020-05-14 14:00:00',NULL,NULL,3),

(16,'BILL2016\_016','可口可乐','饮料','瓶','2000.00','6000.00',2,1,'2020-03-27 13:03:01',NULL,NULL,2),

(17,'BILL2016\_017','脉动','饮料','瓶','1500.00','4500.00',2,1,'2020-05-10 12:00:00',NULL,NULL,2),

(18,'BILL2016\_018','哇哈哈','饮料','瓶','2000.00','4000.00',2,1,'2020-06-24 15:12:03',NULL,NULL,2);

(3)向smbms\_provider表插入数据

insert into `smbms\_provider`(`id`,`proCode`,`proName`,`proDesc`,`proContact`,`proPhone`,`proAddress`,`proFax`,`createdBy`,`creationDate`,`modifyDate`,`modifyBy`) values

(1,'BJ\_GYS001','北京三木堂商贸有限公司','长期合作伙伴，主营产品:茅台、五粮液、郎酒、酒鬼酒、泸州老窖、赖茅酒、法国红酒等','张国强','13566667777','北京市丰台区育芳园北路','010-58858787',1,'2020-03-21 16:52:07',NULL,NULL),

(2,'HB\_GYS001','石家庄帅益食品贸易有限公司','长期合作伙伴，主营产品:饮料、水饮料、植物蛋白饮料、休闲食品、果汁饮料、功能饮料等','王军','13309094212','河北省石家庄新华区','0311-67738876',1,'2020-04-13 04:20:40',NULL,NULL),

(3,'GZ\_GYS001','深圳市泰香米业有限公司','初次合作伙伴，主营产品：良记金轮米,龙轮香米等','郑程瀚','13402013312','广东省深圳市福田区深南大道6006华丰大厦','0755-67776212',1,'2020-03-21 16:56:07',NULL,NULL),

(4,'GZ\_GYS002','深圳市喜来客商贸有限公司','长期合作伙伴，主营产品：坚果炒货.果脯蜜饯.天然花茶.营养豆豆.特色美食.进口食品.海味零食.肉脯肉','林妮','18599897645','广东省深圳市福龙工业区B2栋3楼西','0755-67772341',1,'2020-03-22 16:52:07',NULL,NULL),

(5,'JS\_GYS001','兴化佳美调味品厂','长期合作伙伴，主营产品：天然香辛料、鸡精、复合调味料','徐国洋','13754444221','江苏省兴化市林湖工业区','0523-21299098',1,'2020-02-22 16:52:07',NULL,NULL),

(6,'BJ\_GYS002','北京纳福尔食用油有限公司','长期合作伙伴，主营产品：山茶油、大豆油、花生油、橄榄油等','马莺','13422235678','北京市朝阳区珠江帝景1号楼','010-588634233',1,'2020-03-21 17:52:07',NULL,NULL),

(7,'BJ\_GYS003','北京国粮食用油有限公司','初次合作伙伴，主营产品：花生油、大豆油、小磨油等','王驰','13344441135','北京大兴青云店开发区','010-588134111',1,'2020-04-13 00:00:00',NULL,NULL),

(8,'ZJ\_GYS001','慈溪市广和绿色食品厂','长期合作伙伴，主营产品：豆瓣酱、黄豆酱、甜面酱，辣椒，大蒜等农产品','薛圣丹','18099953223','浙江省宁波市慈溪周巷小安村','0574-34449090',1,'2020-01-21 06:02:07',NULL,NULL),

(9,'GX\_GYS001','优百商贸有限公司','长期合作伙伴，主营产品：日化产品','李立国','13323566543','广西南宁市秀厢大道42-1号','0771-98861134',1,'2020-03-21 19:52:07',NULL,NULL),

(10,'JS\_GYS002','南京火头军信息技术有限公司','长期合作伙伴，主营产品：不锈钢厨具等','陈女士','13098992113','江苏省南京市浦口区浦口大道1号新城总部大厦A座903室','025-86223345',1,'2020-03-25 16:52:07',NULL,NULL),

(11,'GZ\_GYS003','广州市白云区美星五金制品厂','长期合作伙伴，主营产品：海绵床垫、坐垫、靠垫、海绵枕头、头枕等','梁天','13562276775','广州市白云区钟落潭镇福龙路20号','020-85542231',1,'2020-01-21 06:12:17',NULL,NULL),

(12,'BJ\_GYS004','北京隆盛日化科技','长期合作伙伴，主营产品：日化环保清洗剂，家居洗涤专卖、洗涤用品网、墙体除霉剂、墙面霉菌清除剂等','孙欣','13689865678','北京市大兴区旧宫','010-35576786',1,'2020-01-21 12:51:11',NULL,NULL),

(13,'SD\_GYS001','山东豪克华光联合发展有限公司','长期合作伙伴，主营产品：洗衣皂、洗衣粉、洗衣液、洗洁精、消杀类、香皂等','吴洪转','13245468787','山东济阳济北工业区仁和街21号','0531-53362445',1,'2020-01-28 10:52:07',NULL,NULL),

(14,'JS\_GYS003','无锡喜源坤商行','长期合作伙伴，主营产品：日化品批销','周一清','18567674532','江苏无锡盛岸西路','0510-32274422',1,'2020-04-23 11:11:11',NULL,NULL),

(15,'ZJ\_GYS002','乐摆日用品厂','长期合作伙伴，主营产品：各种中、高档塑料杯，塑料乐扣水杯（密封杯）、保鲜杯（保鲜盒）、广告杯、礼品杯','王世杰','13212331567','浙江省金华市义乌市义东路','0579-34452321',1,'2020-06-22 10:01:30',NULL,NULL);

(4)向smbms\_role表插入数据

insert into `smbms\_role`(`id`,`roleCode`,`roleName`,`createdBy`,`creationDate`,`modifyBy`,`modifyDate`) values

(1,'SMBMS\_ADMIN','系统管理员',1,'2020-01-01 00:00:00',NULL,NULL),

(2,'SMBMS\_MANAGER','经理',1,'2020-02-02 00:01:00',NULL,NULL),

(3,'SMBMS\_EMPLOYEE','普通员工',1,'2020-02-03 00:00:00',NULL,NULL);

(5)向smbms\_user表插入数据

insert into `smbms\_user`(`id`,`userCode`,`userName`,`userPassword`,`gender`,`birthday`,`phone`,`address`,`userRole`,`createdBy`,`creationDate`,`modifyBy`,`modifyDate`) values

(1,'admin','系统管理员','1234567',1,'1983-10-10','13688889999','山东省日照市东港区成府路207号',1,1,'2020-03-21 16:52:07',NULL,NULL),

(2,'liming','李明','0000000',2,'1983-12-10','13688884457','山东省日照市东港区前门东大街9号',2,1,'2020-03-01 00:00:00',NULL,NULL),

(5,'hanlubiao','韩路彪','0000000',2,'2001-06-05','18567542321','山东省日照市东港区北辰中心12号',2,1,'2020-02-11 19:52:09',NULL,NULL),

(6,'zhanghua','张华','0000000',1,'1980-06-15','13544561111','山东省日照市东港区学院路61号',3,1,'2020-02-11 10:51:17',NULL,NULL),

(7,'wangyang','王洋','0000000',2,'2001-12-31','13444561124','山东省青岛市三二二区西二旗辉煌国际16层',3,1,'2020-06-11 19:09:07',NULL,NULL),

(8,'zhaoyan','赵燕','0000000',1,'1999-03-07','18098764545','山东省青岛市东科区回龙观小区10号楼',3,1,'2020-04-21 13:54:07',NULL,NULL),

(10,'sunlei','孙磊','0000000',2,'1998-01-04','13387676765','山东省日照市朝阳区管庄新月小区12楼',3,1,'2020-05-06 10:52:07',NULL,NULL),

(11,'sunxing','孙兴','0000000',2,'1997-03-12','13367890900','北京市朝阳区建国门南大街10号',3,1,'2020-01-09 16:51:17',NULL,NULL),

(12,'zhangchen','张晨','0000000',1,'1986-03-28','18098765434','朝阳区管庄路口北柏林爱乐三期13号楼',3,1,'2019-06-09 05:52:37',1,'2020-04-14 14:15:36'),

(13,'dengchao','邓超','0000000',2,'1981-11-04','13689674534','北京市海淀区北航家属院10号楼',3,1,'2020-07-01 08:02:47',NULL,NULL),

(14,'yangguo','杨过','0000000',2,'1989-01-01','13388886623','北京市朝阳区北苑家园茉莉园20号楼',3,1,'2020-02-01 03:52:07',NULL,NULL),

(15,'zhaomin','赵敏','0000000',1,'1989-12-04','18099897657','山东省临沂市昌平区天通苑3区12号楼',2,1,'2020-01-12 12:02:12',NULL,NULL);

此外，本系统中所用到的用户性别和用户身份代码如表3.5.1至表3.5.2所示。

表3.5.1 用户性别代码表

|  |  |
| --- | --- |
| 代码 | 说明 |
| 1 | 女 |
| 2 | 男 |

表3.5.2 用户身份代码

|  |  |
| --- | --- |
| 代码 | 说明 |
| 1 | 系统管理员 |
| 2 | 经理 |
| 3 | 普通员工 |

# 4 各功能模块的设计与实现

## 4.1 系统开发条件

### 4.1.1 开发语言

系统使用的开发语言是Java。Java具有简单性、面向对象、[分布式](https://baike.baidu.com/item/%E5%88%86%E5%B8%83%E5%BC%8F/19276232)、[健壮性](https://baike.baidu.com/item/%E5%81%A5%E5%A3%AE%E6%80%A7/4430133)、[安全性](https://baike.baidu.com/item/%E5%AE%89%E5%85%A8%E6%80%A7/7664678)、平台独立与可移植性、[多线程](https://baike.baidu.com/item/%E5%A4%9A%E7%BA%BF%E7%A8%8B/1190404)等特点。Java可以编写[桌面应用程序](https://baike.baidu.com/item/%E6%A1%8C%E9%9D%A2%E5%BA%94%E7%94%A8%E7%A8%8B%E5%BA%8F/2331979)、[Web应用程序](https://baike.baidu.com/item/Web%E5%BA%94%E7%94%A8%E7%A8%8B%E5%BA%8F)、[分布式系统](https://baike.baidu.com/item/%E5%88%86%E5%B8%83%E5%BC%8F%E7%B3%BB%E7%BB%9F/4905336)应用程序等。正是因为Java语言拥有如此诸多的优秀特性，所以我们选择了它作为开发超市订单管理系统，使得整个开发、调试过程更加高效。

### 4.1.2 开发框架

超市订单管理系统以SSM架构作为支撑，分为表现层、业务层和持久层三层，实现后台数据更新。该架构由Spring MVC、Spring和MyBatis三个开源框架整合而成，用于开发结构合理，性能优越，代码健壮的应用程序。

****

图4.1.1 SSM架构示意图

### 4.1.3 前端框架

由于本系统是Web应用，所以使用了HTML5+CSS3+JavaScript的方式实现前端页面。实现过程中参考了Bootstrap前端开发框架。Bootstrap是Twitter退出的一个用于前端开发的开源工具包。在设计前端页面时，参考了Bootstrap的相关开源代码。

### 4.1.4 集成开发环境

编程所使用的集成开发环境是Eclipse,是著名的跨平台的自由集成开发环境（IDE）。Eclipse 是一个开放源代码的、基于Java的可扩展开发平台。就其本身而言，它只是一个框架和一组服务，用于通过插件组件构建开发环境。本次系统便选用了Eclipse作为开发平台。

### 4.1.5 Web应用服务器

Tomcat由[Apache](https://baike.baidu.com/item/Apache/6265)、Sun和其他一些公司及个人共同开发而成。由于有了Sun的参与和支持，最新的Servlet和JSP规范可以在Tomcat中得到体现。因为Tomcat技术先进、性能稳定，因而成为目前比较流行的Web应用服务器。本次系统选用的便是Tomcat作为应用服务器。

### 4.1.6 数据库管理系统

本系统使用的数据库管理系统是MySQL Community。MySQL是一个关系型数据库管理系统，由瑞典MySQL AB公司开发。在WEB应用方面，MySQL是最好的RDBMS (Relational Database Management System，关系数据库管理系统)应用软件。

系统中的数据库以及数据库中的所有关系模式都使用MySQL进行处理。

## 4.2 用户界面设计

完成数据库创建和功能说明以后，我们进行下一步工作，即设计用户界面，完成了系统要求的 5 项主要功能。

我们把超市订单管理系统的窗体分成5个主要部分，如下图所示。

1. 订单管理

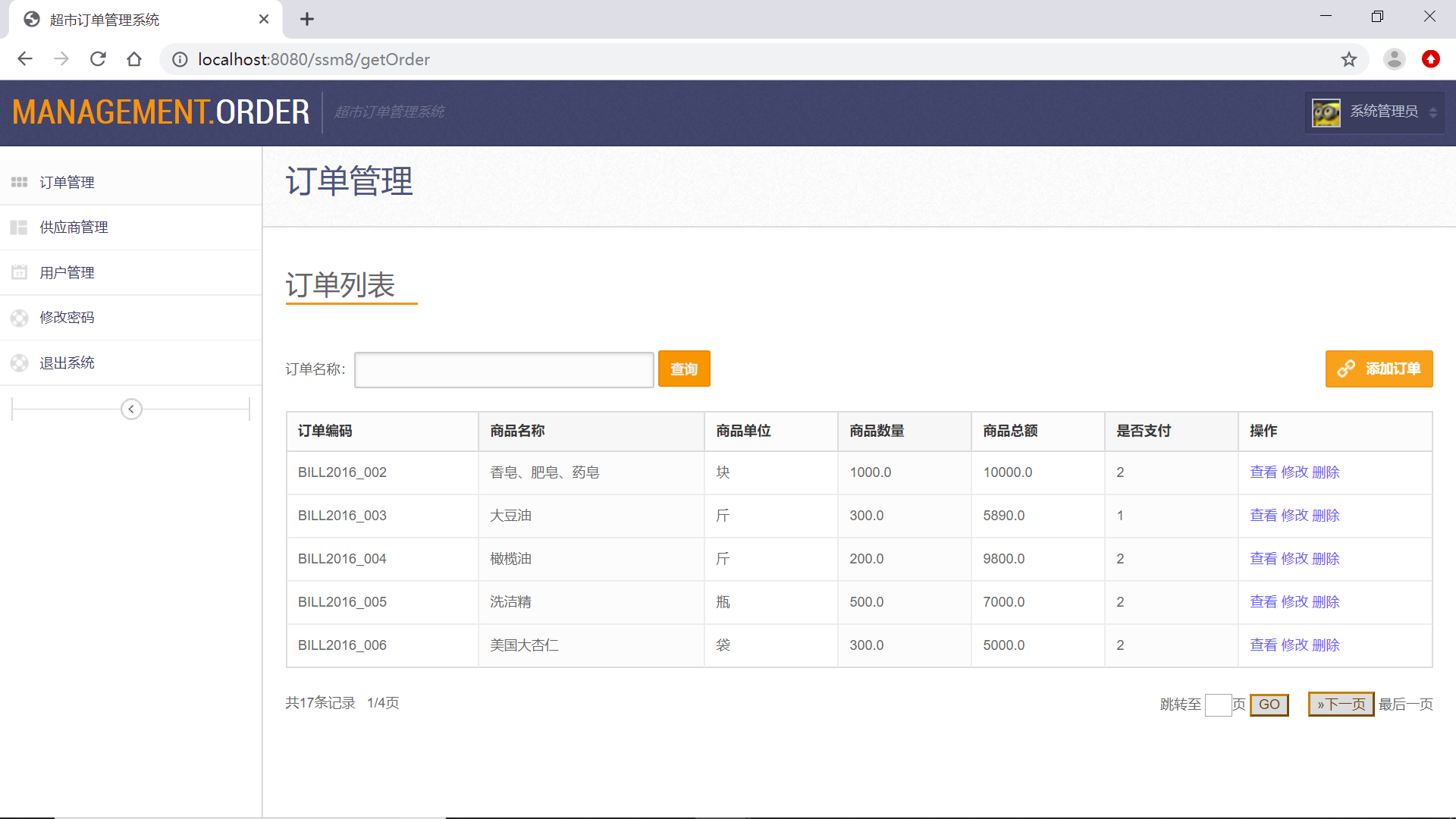


图 4.2.1 订单管理界面

1. 供应商管理

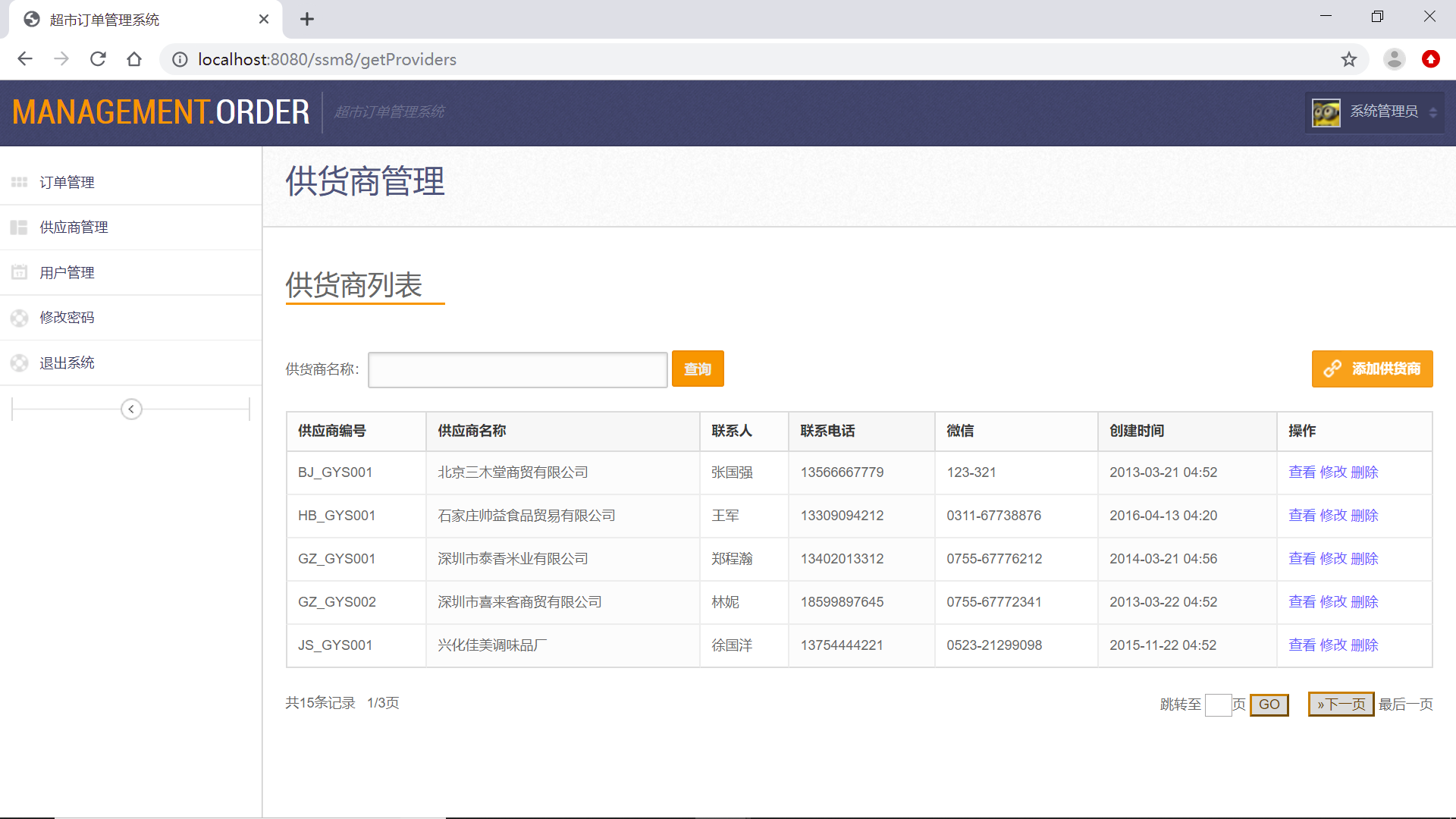


图 4.2.2 供应商管理界面

1. 用户管理

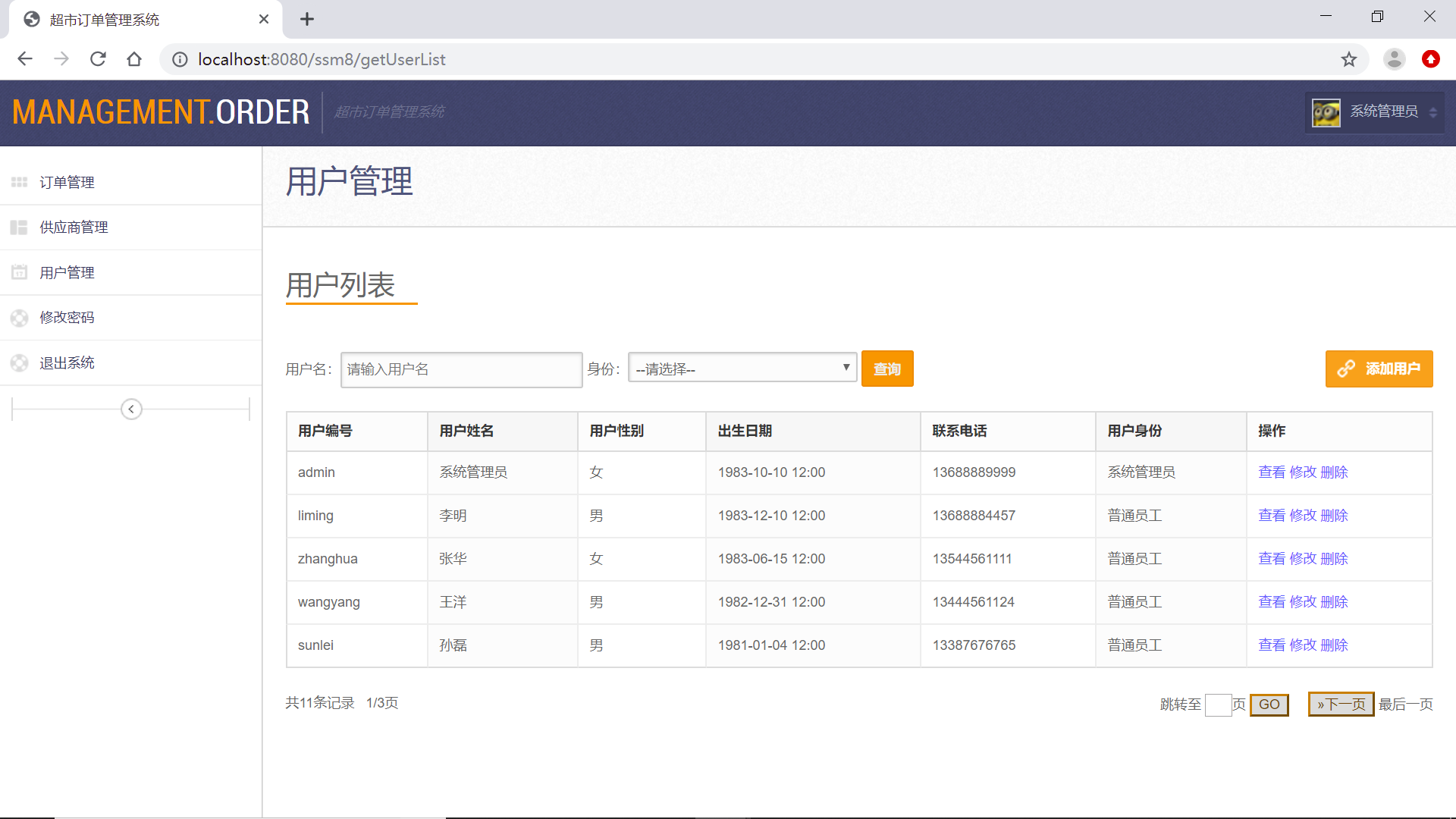


图 4.2.3 用户管理界面

1. 修改密码

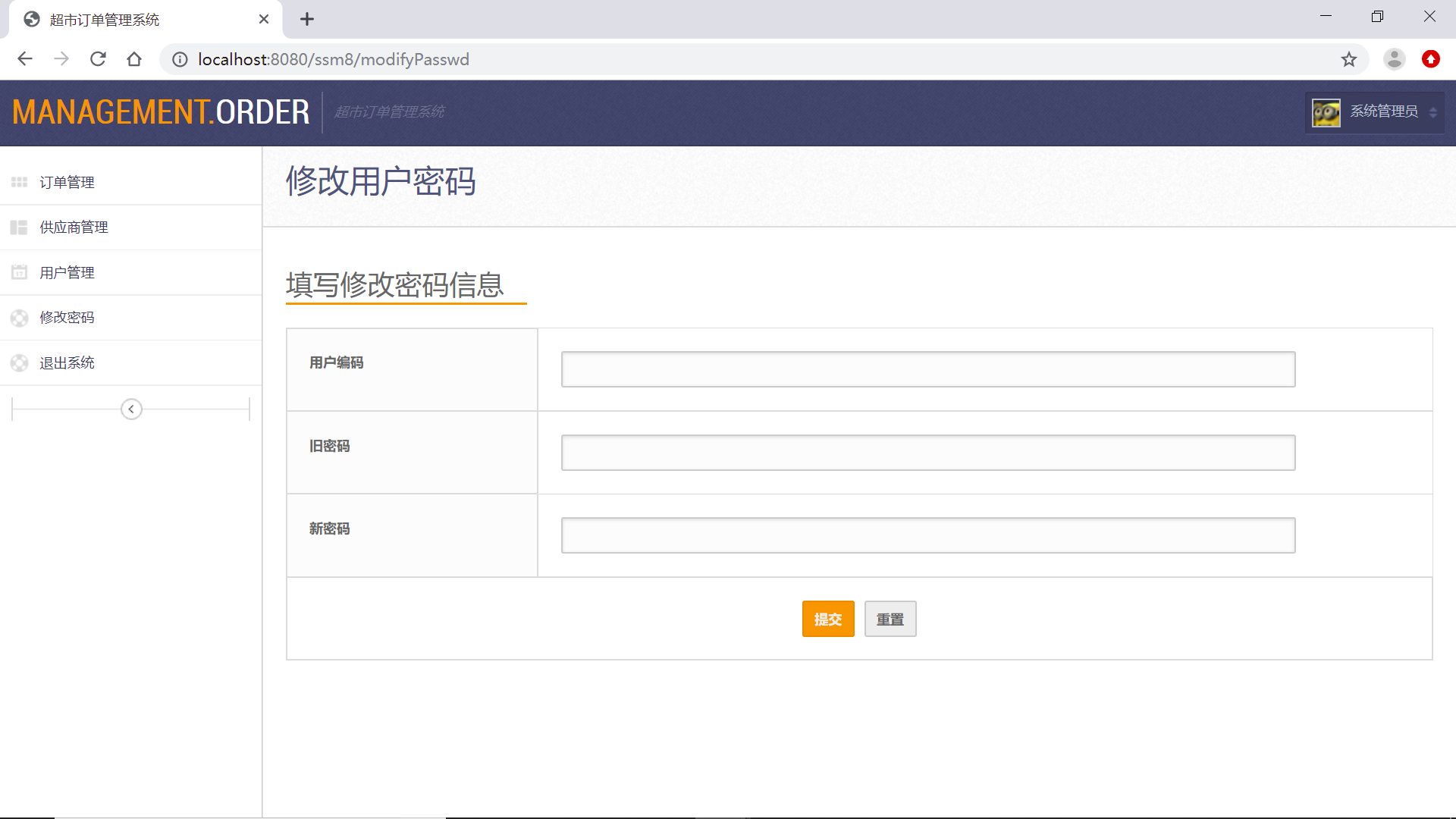


图 4.2.4 修改密码界面

1. 登录注销

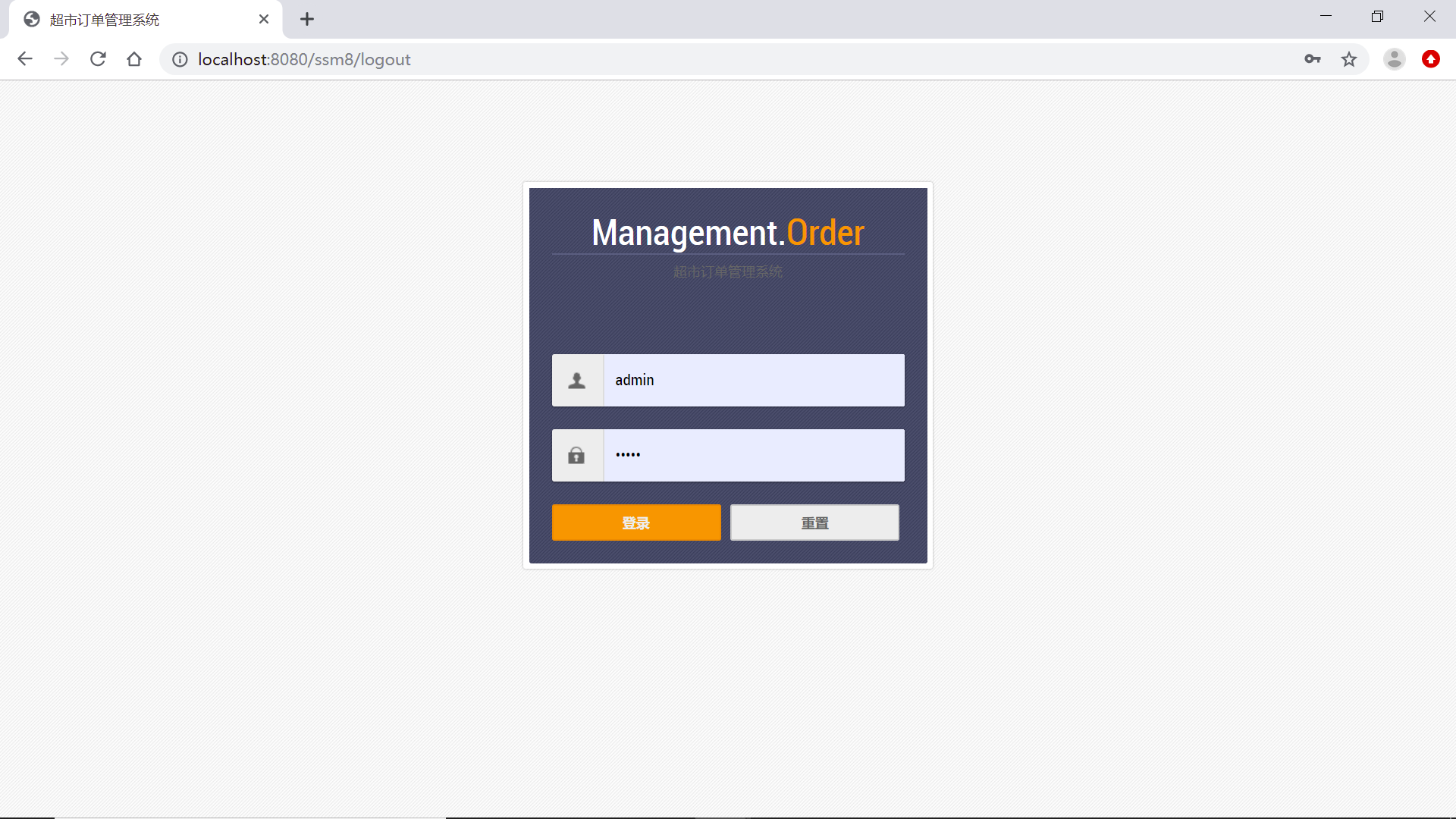


图 4.2.5 登录界面

## 4.3 功能模块说明

1、订单信息添加、查询、修改与删除

（1）订单信息查看：为了对订单浏览信息，能够实现浏览的功能是十分必要的。管理员输入需要搜索的相应信息，点击查看按钮后系统将寻找到的数据展示到网页中。

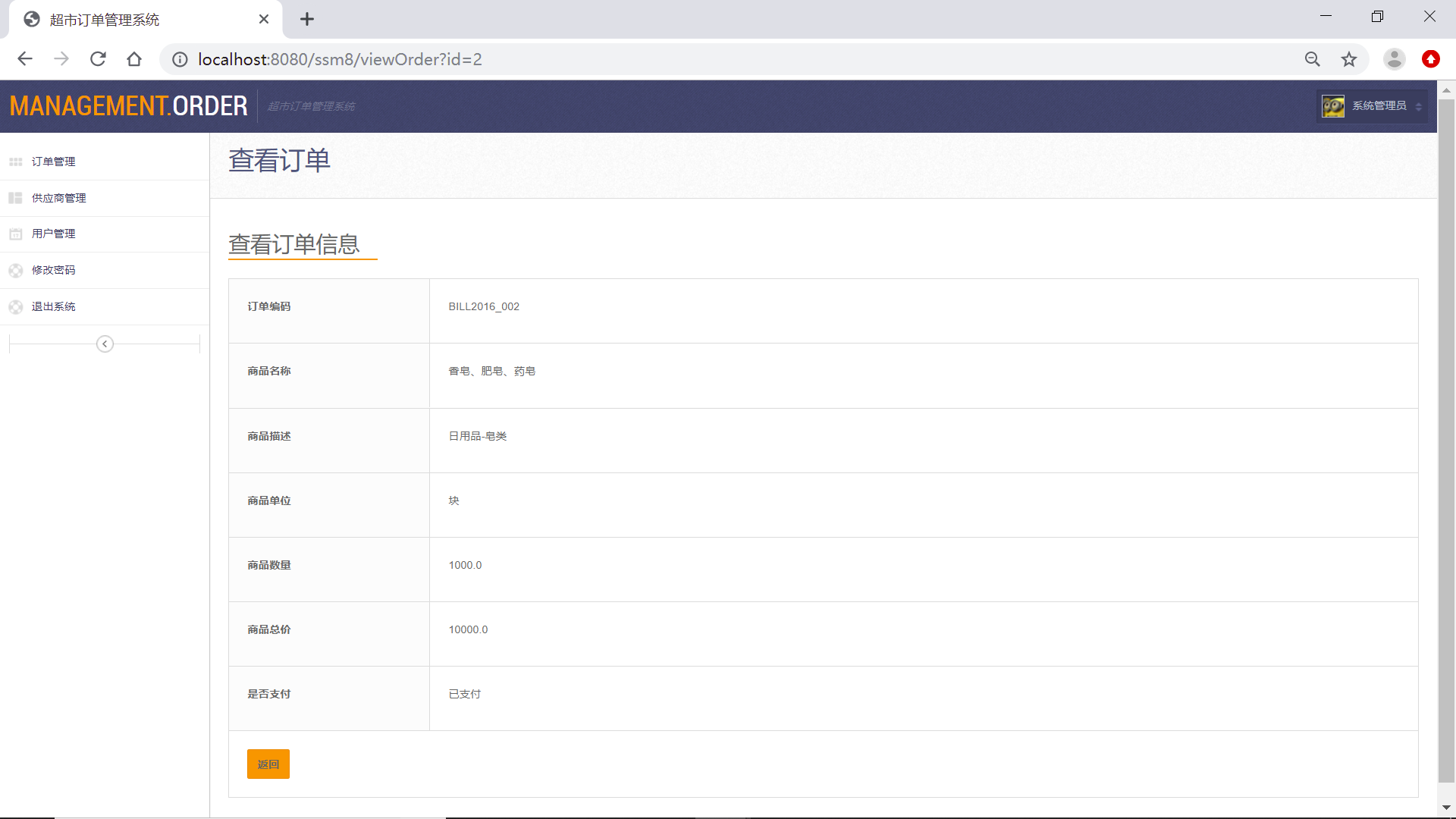


图4.3.1 订单查看界面

（2）订单信息添加：作为超市订单管理系统，订单信息的管理是很重要的。每当采购部门增加新的订单时，订单信息就要增加。超市也可能因为其它原因增加订单信息，订单添加模块都可以做出快捷的解决方案。管理员输入相应的信息，点击提交后系统将数据保存到数据库中。

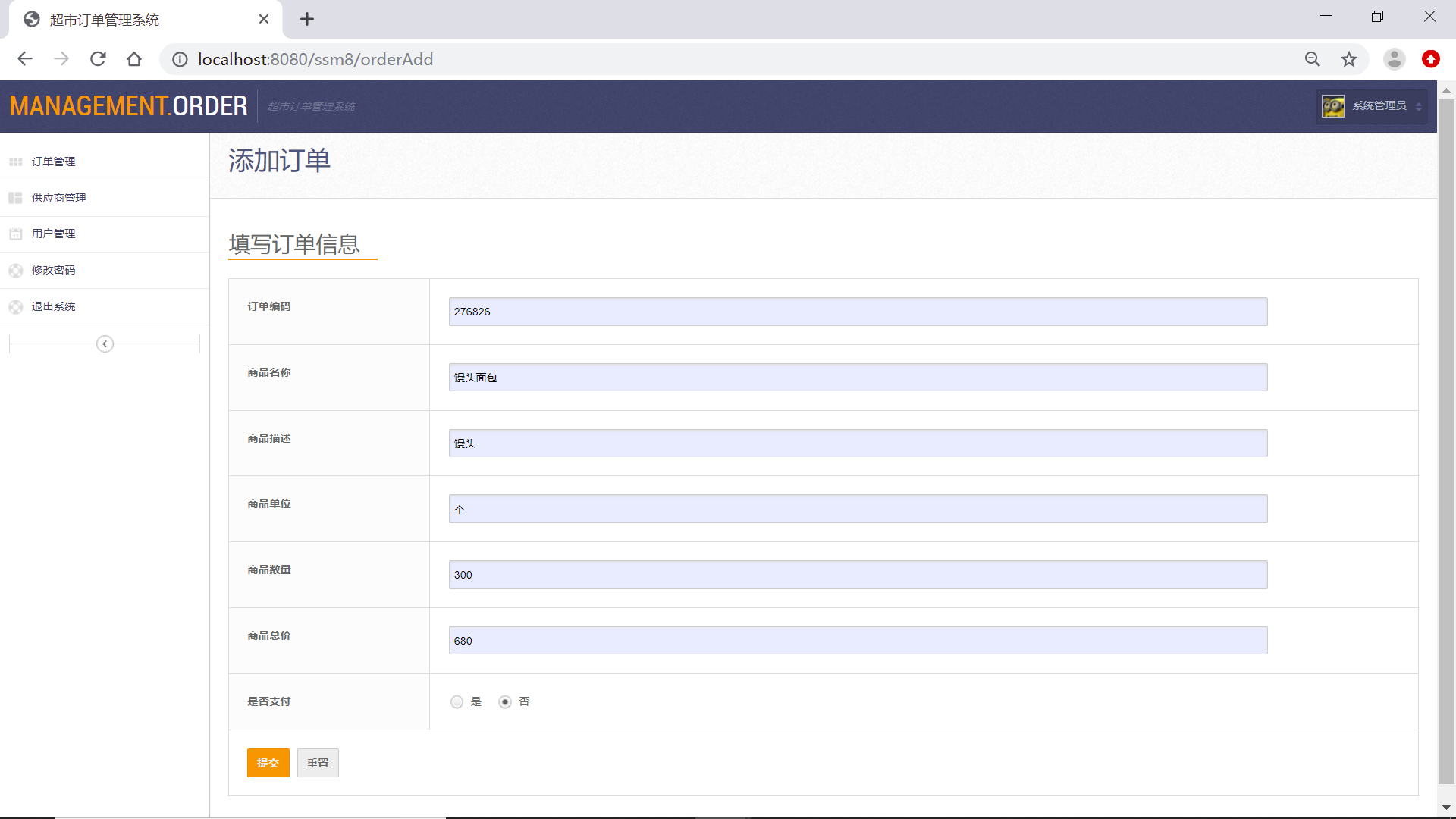


图4.3.2 订单添加界面

（3）订单信息修改：根据订单编号可以查询订单详细信息，然后修改订单的所有信息。系统从数据库中读取数据并显示到页面上，管理员修改数据后，点击修改按钮，系统将更新表中的数据。

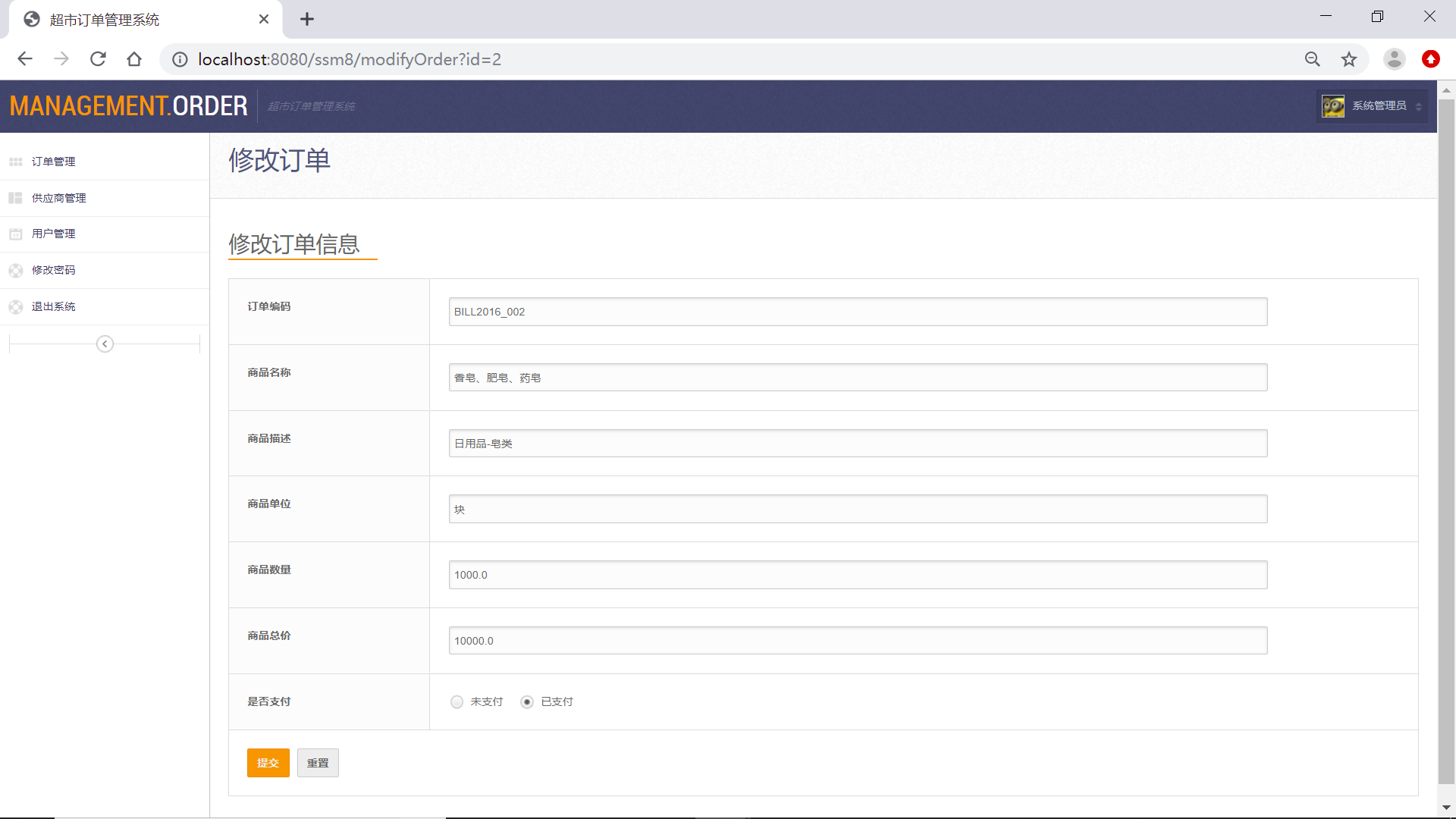


图4.3.3 订单修改界面

（4）订单信息删除：根据订单编号可以删除该订单的信息。管理员选择需要删除订单名称并点击删除按钮，系统将从数据库中删除相应数据。

（5）订单信息查询：在成千上万种商品种，如果人为寻找某一个商品肯定是不可能的，只有通过商品信息查询模块才能为用户或管理人员解决这个难题。根据订单名称可以查询该订单的信息。管理员输入订单名称并点击查询按钮，系统将从数据库中查询相应的数据并显示到页面上。

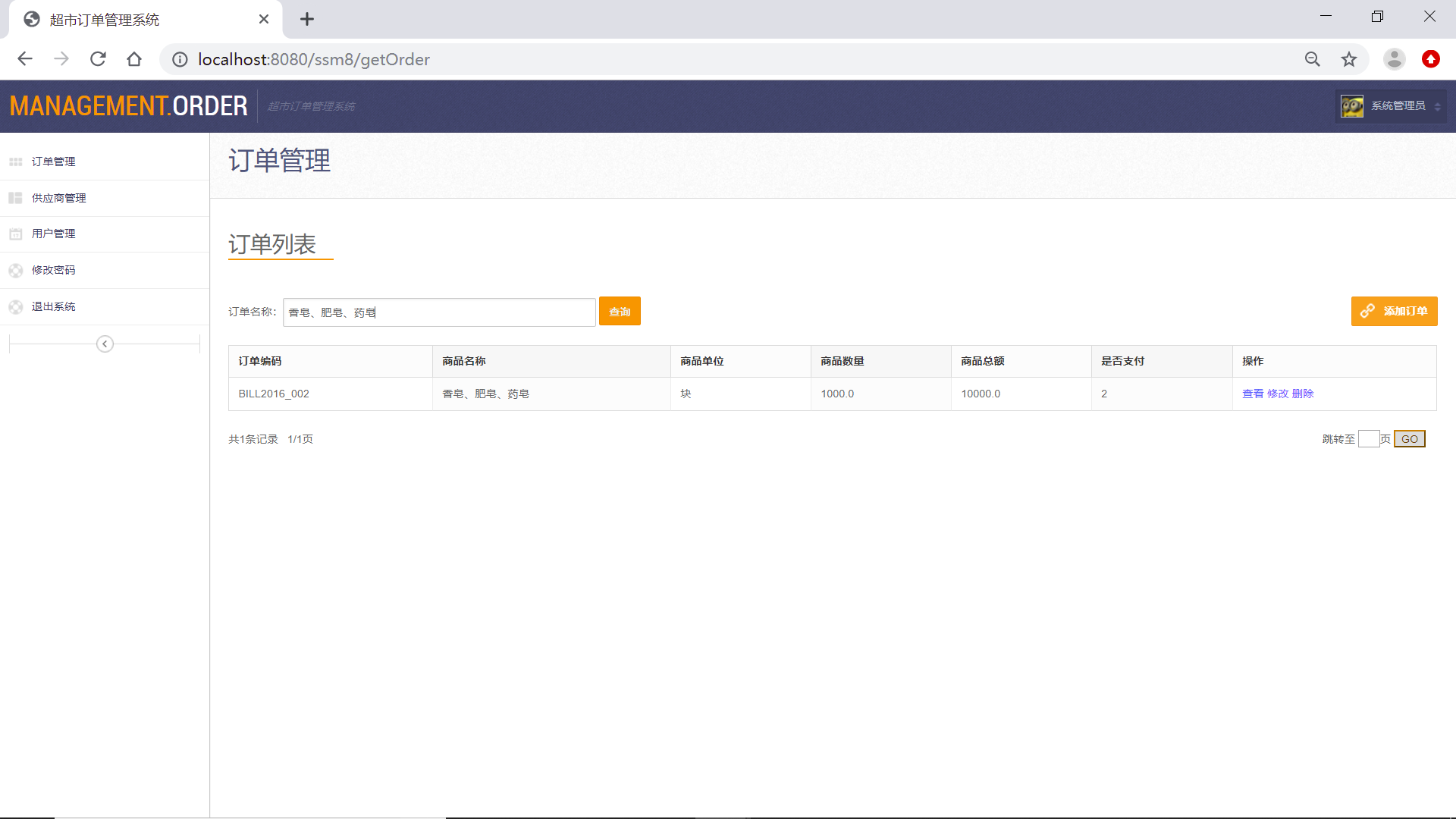


图4.3.4 订单查询界面

1. 供应商信息添加、查询、修改与删除

（1）供应商查询界面：供应商查询界面提供了供应商的信息，可根据供应商名称的关键字进行筛选查询，并提供了添加供应商、查看供应商详细信息、修改供应商信息、删除供应商的功能。

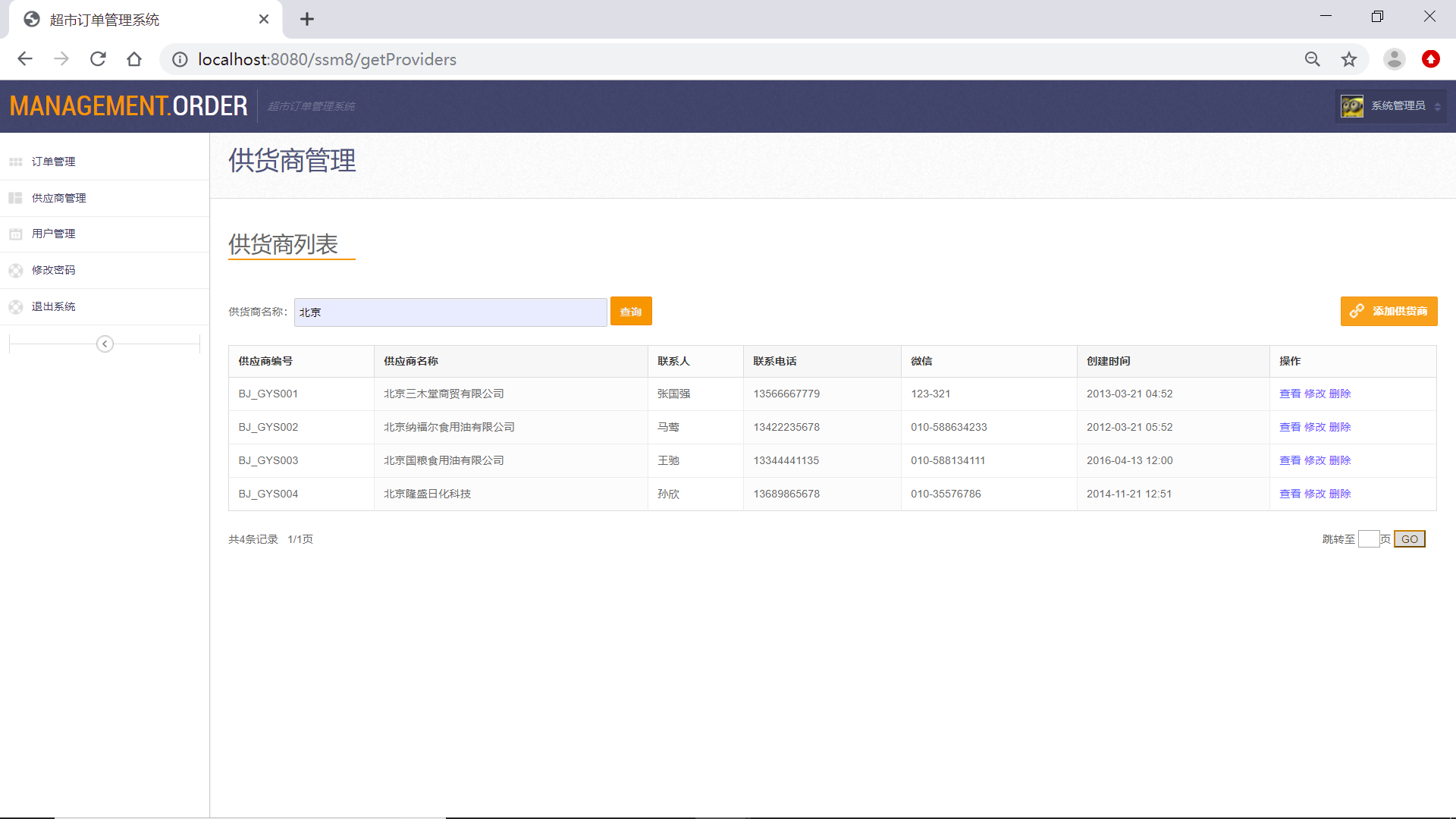


图4.3.5 供应商查询界面

（2）供应商查看详情界面：在供应商查询界面点击具体供应商操作列表的查看按钮，可以查看供应商的具体信息，包括：供货商编码、供货商名称、详细描述、联系人、联系电话、地址、微信。



图4.3.6 供应商查看界面

（3）供应商修改页面：若供应商信息变动，管理员可通过供应商信息修改功能对供应商信息进行更新，更新后的数据将保存到数据库中。

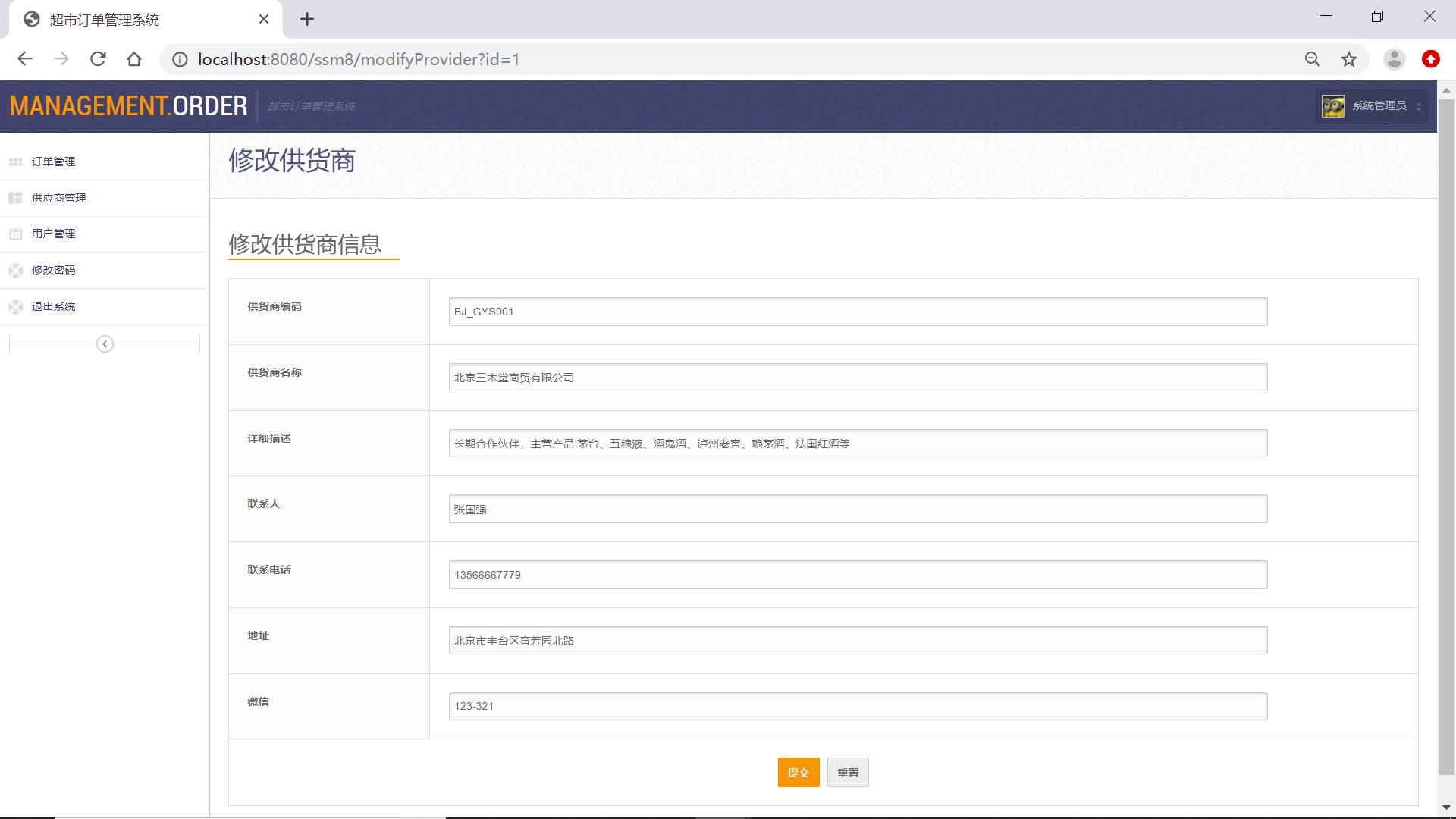


图4.3.7 供应商修改界面

（4）商品供应商信息删除：企业倒闭或者经营策略的改变，当它对超市商品的供应没有作用时，商品供应商厂家信息的删除是正常的。管理员输入供应商名称查询数据表中的数据并显示到页面上，点击删除后系统将表中的相应数据删除。

（5）供应商添加界面：与供应商达成交易后，管理员在供应商添加页面填写供应商具体信息，填写完毕点击提交，添加后的数据将保存到数据库中。



图4.3.8 供应商添加界面

1. 用户信息添加、查询、修改与删除

（1）用户管理页面，通过输入用户名和身份查询用户。当不记得用户名的具体名字时，只输入用户名的其中一个字，会检索出所有带这个字的用户，方便管理员查询管理。点击右边链接添加用户，会连接到相关网页添加用户信息。点击操作里的查看、修改等可以进行相应的改、删、查操作。

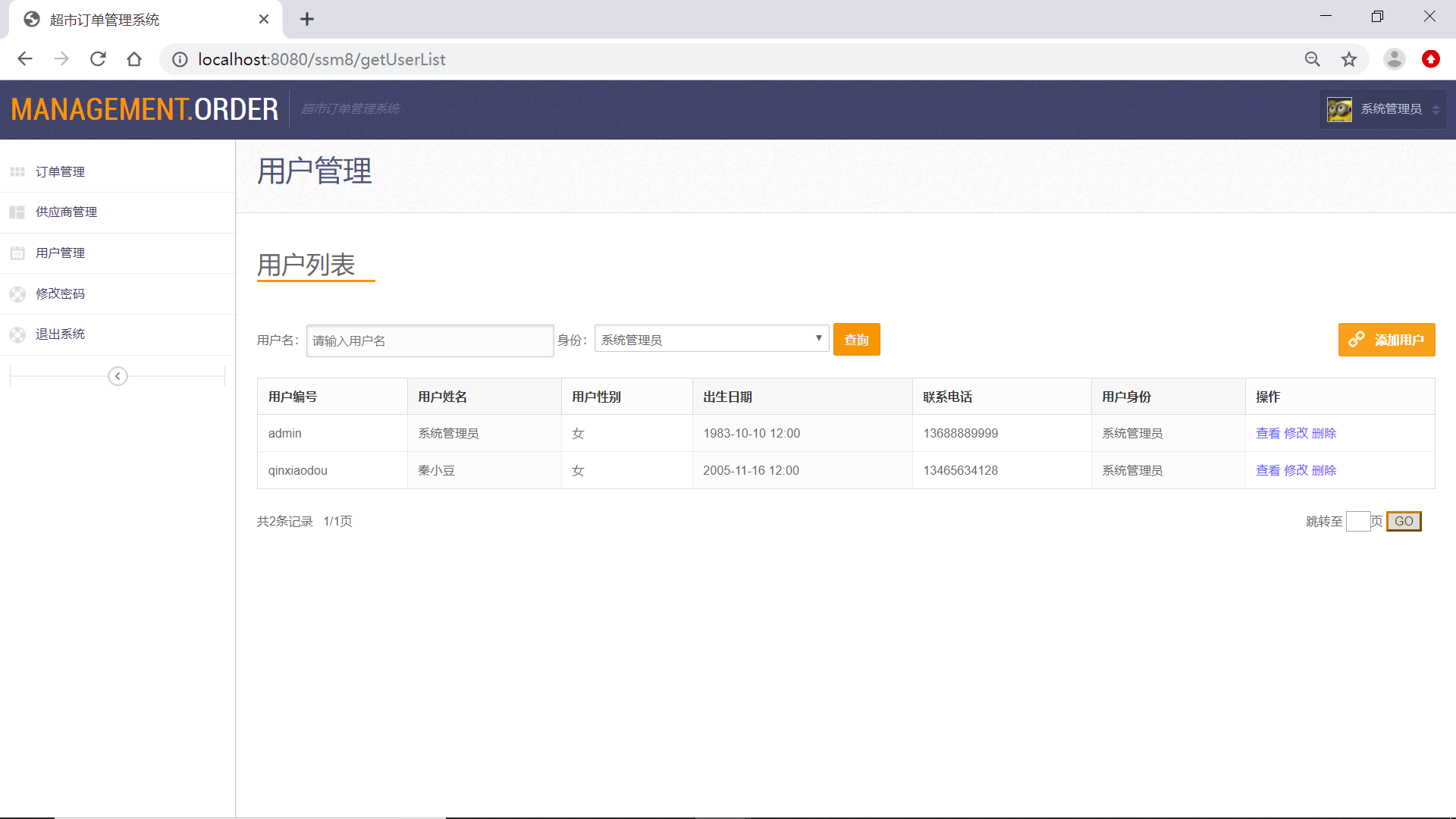


图4.3.9 用户查询界面

（2）用户信息删除：当企业员工离职时，或者经过一段时间后，会发现用户表中一些信息时无用的，用户删除模块可以解决这样的问题。

（3）添加用户信息：填写用户相关信息，下面有两个按钮，可以选择重置或者提交。

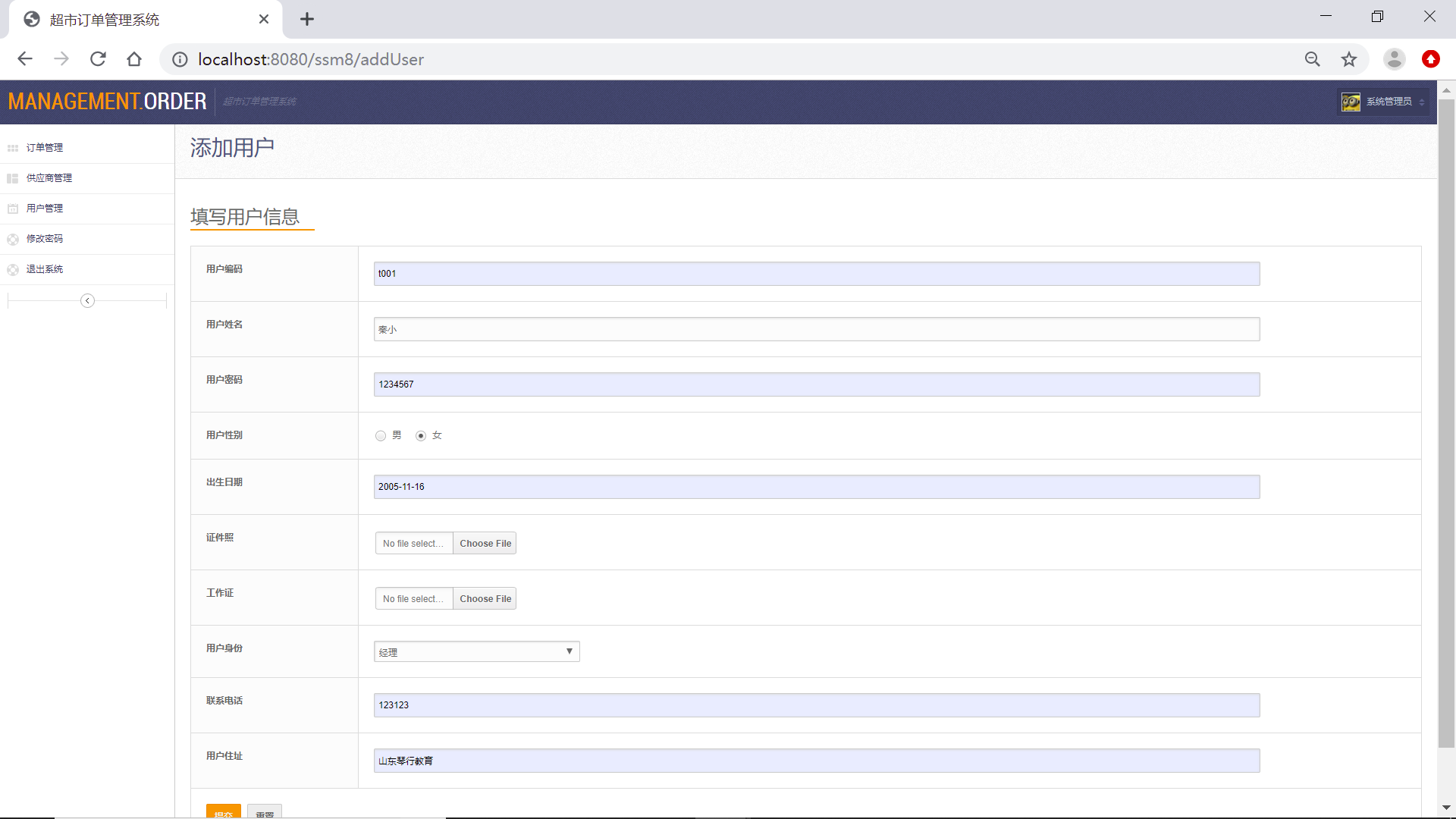


图4.3.10 用户添加界面

1. 修改密码

为了系统的安全，用户的应该只有用户个人才能修改，这不仅保证了整个公司的利益也保护了个人隐私。用户在输入相应的用户编号，填写旧密码以及新密码后，点击提交，重置密码成功。发现输入错误时，可以手动删除或者点击重置按钮，重新填写。

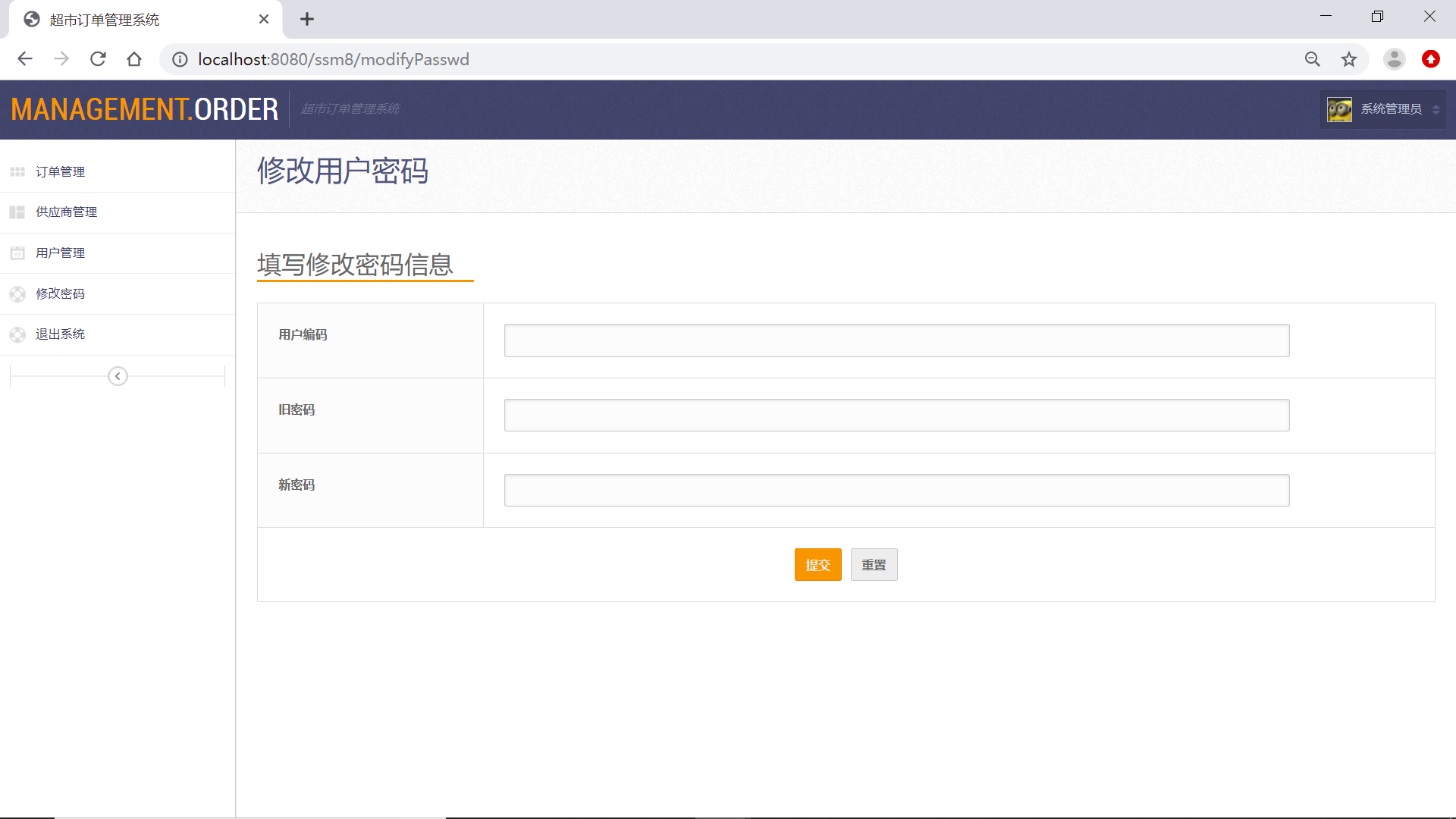


图4.3.13 修改密码界面

修改用户密码成功后，会弹出修改用户密码成功页面，如图4.3.14所示。

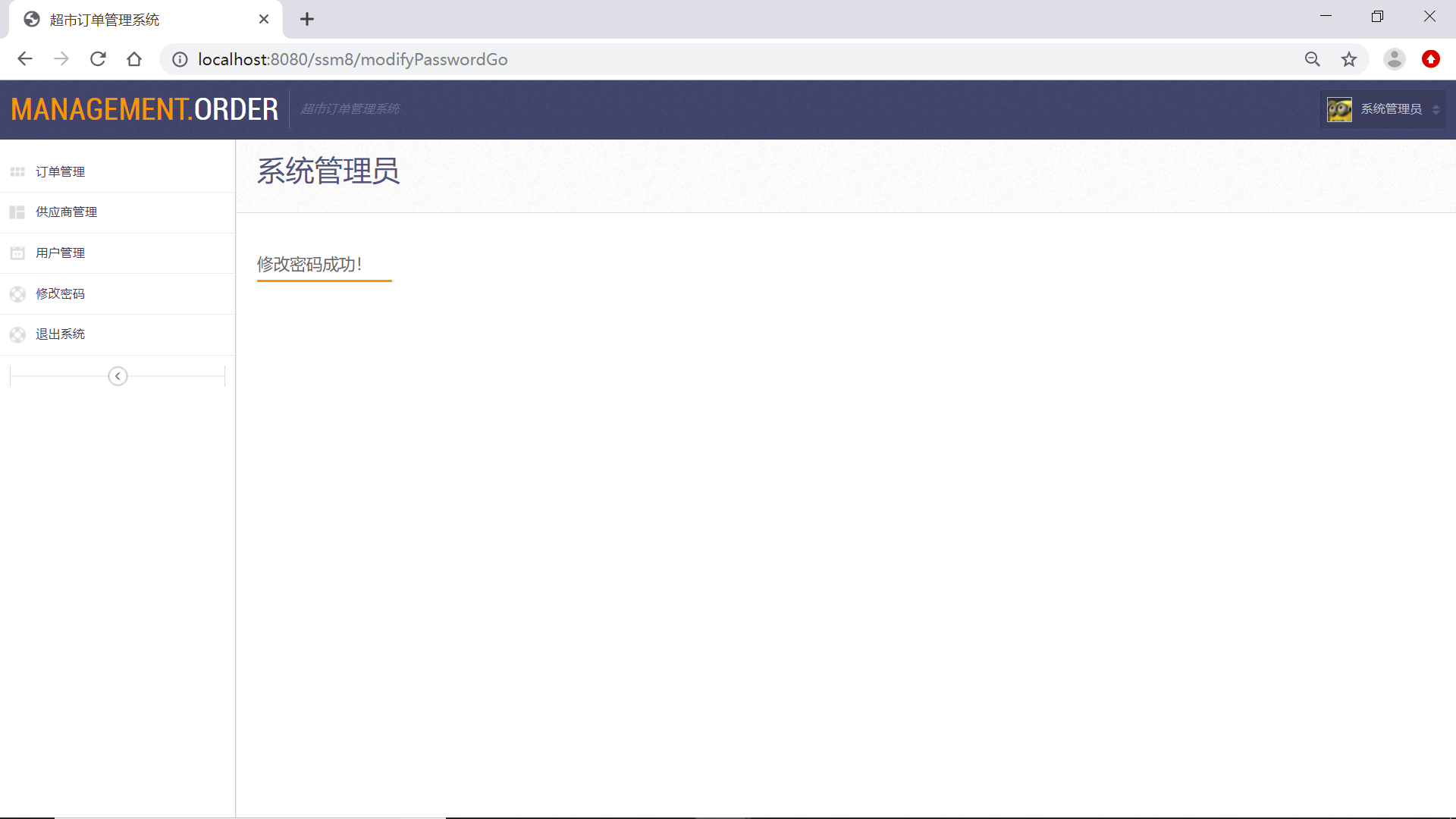


图4.3.14 修改密码成功界面

1. 登录/注销

输入用户名以及用户密码登录进入超市订单管理界面，可以查看管理信息。管理员可以对相关数据进行增、改、查等操作，也可以注销退出系统。

# 5 实训总结

## 5.1 所遇困难

在实现本系统时遇到的困难主要体现在两个方面，一是系统的前端页面的设计，二是怎样Web与数据库实现交互。

系统前端页面的设计困难的解决是通过参考著名的前端框架Bootstrap实现的。Bootstrap框架提供了许多精美的组建、布局，还开放了源代码供参考。在此基础上我们还加入了一些利用JavaScript代码实现的美化效果，使得前端设计更加美观。

实体Web与数据库交互的解决得益于SSM框架的三层Spring MVC、Spring和MyBatis，能够分离处理数据库与Web层的视图，从而达到交互的目的。

此外，在编写后端的时候，变量的大小写、系统配置也是困难重重。好在，在反复编写之后，迅速熟悉的技巧，能够让页面自由切换。系统配置更是反复在网上求证，得以解决。

## 5.2 实验心得

这一次合作开发超市订单管理系统，从开始选择课题的困惑到最终完成了一个我们还算满意的作品，使我学到了很多东西。从设计数据库到编写后台代码，链接数据库，在网页上显示，令人印象深刻。反复查阅资料，启动Tomcat到凌晨0点，都是藏着对这次项目的努力。其实，从一开始选择哪个题目是否用SSM框架来开发我一直也犹豫过，像国内势头正旺的ThinkPHP，易学易用，完善的中文开发文档，遇到问题或者bug可以非常容易的在中文社区得到解答。但是我最后选择了SSM框架，不仅仅因为它广泛，而是我希望能够挑战自己。经过这一个周的磨练，我最大的收获除了学到了真正可以应用的知识外，更重要的是学会了项目合作开发的经验。

# 6 附件

## 6.1 核心代码（一）

package cn.smbms.controller;

import java.util.Date;

import java.util.List;

import javax.servlet.http.HttpSession;

import org.apache.commons.lang3.StringUtils;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.RequestMapping;

import cn.smbms.pojo.Order;

import cn.smbms.pojo.User;

import cn.smbms.service.OrderService;

import cn.smbms.utils.Constants;

import cn.smbms.utils.PageHelper;

@Controller

public class OrderController {

//引入业务层的资源

@Autowired

OrderService orderService;

//添加订单界面

@RequestMapping("/orderAdd")

public String orderAdd() {

return "orderAdd";

}

//添加订单处理方法

@RequestMapping("/orderAddGo")

public String orderAddGo(Order order,HttpSession session) {

Order ord = new Order();

ord = order;

Integer oid=null;

oid=((User) session.getAttribute(Constants.USER\_SESSION)).getId();

if(null!=oid) {

ord.setCreatedBy(oid);

if(orderService.orderAdd(ord))

return "redirect:getOrder";

else

return "redirect:orderAdd";

}

else

return "redirect:login";

}

//获取订单列表的处理方法

@RequestMapping("/getOrder")

public String getOrderList(String orderName,Model md,String pageIndex) {

String strName="";

if(!StringUtils.isEmpty(orderName)) {

strName=orderName;

}

Integer currentPageNo=1;

Integer pageSize=Constants.pageSize;

if(!StringUtils.isEmpty(pageIndex)) {

currentPageNo=Integer.parseInt(pageIndex);

}

//根据条件获取总记录数

Integer totalCount=orderService.getOrderCount(strName);

//计算可以分几页

PageHelper pages=new PageHelper();

pages.setCurrentPageNo(currentPageNo);

pages.setPageSize(pageSize);

pages.setTotalCount(totalCount);

int totalPageCount = pages.getTotalPageCount();

List<Order> order=orderService.getOrderList(strName,(currentPageNo-1)\*pageSize,pageSize);

md.addAttribute("totalPageCount", totalPageCount);

md.addAttribute("totalCount", totalCount);

md.addAttribute("currentPageNo", currentPageNo);

md.addAttribute("order",order);

return "orderList";

}

//根据id删除订单信息

@RequestMapping("deleteOrderById")

public String deleteOrderById(String id,Model md) {

String msg="";

if(orderService.deleteOrderById(id)) {

msg="删除订单成功！";

}else

msg="删除订单失败！";

md.addAttribute("msg", msg);

return "redirect:getOrder";

}

//添加修改界面

@RequestMapping("/modifyOrder")

public String modifyOrder(String id,Model md) {

Order order=new Order();

order=orderService.getOrderById(id);

md.addAttribute("order", order);

return "modifyOrder";

}

//添加修改用户的执行路由信息

@RequestMapping("/modifyOrderGo")

public String modifyOrderGo(Order order,HttpSession session) {

Integer oid=((User)session.getAttribute(Constants.USER\_SESSION)).getId();

if(null!=order) {

order.setModifyBy(oid);

order.setModifyDate(new Date());

orderService.modifyOrder(order);

}

return "redirect:getOrder";

}

//根据ID查看订单

@RequestMapping("/viewOrder")

public String viewOrder(String id,Model md) {

Order order=new Order();

order=orderService.getOrderById(id);

md.addAttribute("order", order);

return "viewOrder";

}

}

## 6.2 核心代码（二）

package cn.smbms.controller;

import java.util.Date;

import java.util.List;

import javax.servlet.http.HttpSession;

import org.apache.commons.lang3.StringUtils;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.RequestMapping;

import cn.smbms.pojo.Provider;

import cn.smbms.pojo.User;

import cn.smbms.service.ProviderService;

import cn.smbms.utils.Constants;

import cn.smbms.utils.PageHelper;

@Controller

public class ProviderController {

//引入业务层的资源

@Autowired

ProviderService proService;

//添加供应商处理方法

@RequestMapping("/providerAddGo")

public String providerAddGo(Provider provider,HttpSession session) {

Provider pro=new Provider();

pro=provider;

Integer uid=null;

uid=((User) session.getAttribute(Constants.USER\_SESSION)).getId();

if(null!=uid) {

pro.setCreatedBy(uid);

if(proService.providerAdd(pro))

return "redirect:getProviders";

else

return "redirect:providerAdd";

}

else

return "redirect:login";

}

//添加供应商界面

@RequestMapping("/providerAdd")

public String providerAdd() {

return "providerAdd";

}

@RequestMapping("/getProviders")

public String getProviderList(String proName,Model md,String pageIndex) {

String strName="";

if(!StringUtils.isEmpty(proName)) {

strName=proName;

}

Integer currentPageNo=1;

Integer pageSize=Constants.pageSize;

if(!StringUtils.isEmpty(pageIndex)) {

currentPageNo=Integer.parseInt(pageIndex);

}

//根据条件获取总记录数

Integer totalCount=proService.getProviderCount(strName);

//计算可以分几页

PageHelper pages=new PageHelper();

pages.setCurrentPageNo(currentPageNo);

pages.setPageSize(pageSize);

pages.setTotalCount(totalCount);

int totalPageCount = pages.getTotalPageCount();

List<Provider> providers=proService.getProviderList(strName,(currentPageNo-1)\*pageSize,pageSize);

md.addAttribute("totalPageCount", totalPageCount);

md.addAttribute("totalCount", totalCount);

md.addAttribute("currentPageNo", currentPageNo);

md.addAttribute("providers",providers);

return "providerList";

}

//添加修改界面

@RequestMapping("/modifyProvider")

public String modifyProvider(String id,Model md) {

Provider provider=new Provider();

provider=proService.getProvider(id);

md.addAttribute("provider", provider);

return "modifyProvider";

}

//修改供应商

@RequestMapping("/modifyProviderGo")

public String modifyProviderGo(Provider provider,HttpSession session) {

Integer oid=((User)session.getAttribute(Constants.USER\_SESSION)).getId();

if(null!=provider) {

provider.setModifyBy(oid);

provider.setModifyDate(new Date());

proService.modifyProvider(provider);

}

return "redirect:getProviders";

}

//从ID获取某一个供应商信息

@RequestMapping("/viewProvider")

public String viewProvider(String id,Model md) {

Provider provider=new Provider();

provider=proService.getProvider(id);

md.addAttribute("provider", provider);

return "viewProvider";

}

//删除供应商

@RequestMapping("deleteProviderById")

public String deleteProviderById(String id,Model md) {

String msg="";

if(proService.deleteProviderById(id)) {

msg="删除订单成功！";

}else

msg="删除订单失败！";

md.addAttribute("msg", msg);

return "redirect:getProviders";

}

}

## 6.3 核心代码（三）

package cn.smbms.controller;

import java.util.Date;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpSession;

import org.apache.commons.lang3.StringUtils;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestParam;

import org.springframework.web.multipart.MultipartFile;

import cn.smbms.pojo.User;

import cn.smbms.service.UserService;

import cn.smbms.utils.Constants;

import cn.smbms.utils.PageHelper;

import cn.smbms.utils.UploadHelper;

@Controller

public class UserController {

@Resource

UserService userService;

//更据id删除用户信息

@RequestMapping("deleteUserById")

public String deleteUserById(String id,Model md) {

String msg="";

if(userService.deleteUserById(id)) {

msg="删除用户成功！";

}else

msg="删除用户失败！";

md.addAttribute("msg", msg);

return "redirect:getUserList";

}

//添加修改用户的执行路由信息

@RequestMapping("/modifyUserGo")

public String modifyUserGo(User user,HttpSession session) {

Integer uid=((User)session.getAttribute(Constants.USER\_SESSION)).getId();

if(null!=user) {

user.setModifyBy(uid);

user.setModifyDate(new Date());

userService.modifyUser(user);

}

return "redirect:getUserList";

}

//添加修改用户的界面路由

@RequestMapping("/modifyUser")

public String modifyUser(String id,Model md) {

User user=new User();

user=userService.getUserById(id);

md.addAttribute("user", user);

return "modifyUser";

}

//修改密码实现方法

@RequestMapping("/modifyPasswordGo")

public String modifyPasswordGo(String userCode,String userPassword,String newPassword,Model md) {

String msg="";

if(userService.modifyPasswd(userCode, userPassword, newPassword))

msg="修改密码成功！";

else

msg="修改密码失败！";

md.addAttribute("msg",msg);

return "modifyPasswordFinish";

}

//添加用户打开修改密码的界面

@RequestMapping("/modifyPasswd")

public String modifyPasswd() {

return "modifyPasswd";

}

//添加用户处理方法

@RequestMapping("addUserGo")

public String addUserGo(User user,HttpSession session,HttpServletRequest request,

@RequestParam(value ="midPicPath", required = false) MultipartFile attach,

@RequestParam(value ="mworkPicPath", required = false) MultipartFile attach1,

Model md) {

String strIdPicPath=null;

String strWorkPicPath=null;

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

String savePath= request.getSession().getServletContext().getRealPath("uploads");

msgMap=UploadHelper.saveFile(savePath, attach);

if(msgMap.get("flag")=="fail") {

md.addAttribute("error",msgMap.get("msg"));

return "redirect:addUser";

}else {

strIdPicPath=msgMap.get("url");

}

msgMap=UploadHelper.saveFile(savePath, attach1);

if(msgMap.get("flag")=="fail") {

md.addAttribute("error",msgMap.get("msg"));

return "redirect:addUser";

}else {

strWorkPicPath=msgMap.get("url");

}

User user1=new User();

user1=user;

Integer uid=null;

uid=((User) session.getAttribute(Constants.USER\_SESSION)).getId();

if(null!=uid) {

user1.setCreatedBy(uid);

user1.setIdPicPath(strIdPicPath);

user1.setWorkPicPath(strWorkPicPath);

if(userService.addUser(user1))

return "redirect:getUserList";

else {

md.addAttribute("error","您还没有登录，请登录！");

return "redirect:addUser";

}

}

else

return "redirect:login";

}

//添加用户界面

@RequestMapping("addUser")

public String addUser() {

return "addUser";

}

//请求登录界面

@RequestMapping("/login")

public String login() {

return "login";

}

//退出界面

@RequestMapping("/logout")

public String logout(HttpSession session){

session.removeAttribute("user");

return "login";

}

//请求处理程序

@RequestMapping("/loginGo")

public String loginGo(String userCode,String userPassword,Model md,HttpSession session) {

User user=new User();

user=userService.checkLogin(userCode, userPassword);

if(null!=user) {

md.addAttribute("user",user);

session.setAttribute(Constants.USER\_SESSION, user);

return "frame";

}else {

//md.addAttribute("msg","用户名或密码错误！");

throw new RuntimeException("用户名或密码错误！");

//return "login";

}

}

//异常处理

// @ExceptionHandler(value= {RuntimeException.class})

// public String handlerException(RuntimeException e,HttpServletRequest req) {

// req.setAttribute("e", e);

// return "error";

// }

@RequestMapping("/main")

public String main(HttpSession session) {

//检查是否登录

if(session.getAttribute("user")==null) {

return "redirect:login";

}

return "main";

}

//查询用户列表

@RequestMapping("/getUserList")

public String getUserList(String userName,String roleName,String pageIndex,Model md) {

//设置页面容量

int pageSize = Constants.pageSize;

//当前页码

int currentPageNo = 1;

if(pageIndex != null){

try{

currentPageNo = Integer.valueOf(pageIndex);

}catch(NumberFormatException e){

return "redirect:/user/syserror.html";

//response.sendRedirect("syserror.jsp");

}

}

//总数量（表）

if(StringUtils.isEmpty(userName))

userName="";

if(StringUtils.isEmpty(roleName))

roleName="";

int totalCount = userService.getUserCount(userName, roleName);

//总页数

PageHelper pages=new PageHelper();

pages.setCurrentPageNo(currentPageNo);

pages.setPageSize(pageSize);

pages.setTotalCount(totalCount);

int totalPageCount = pages.getTotalPageCount();

//控制首页和尾页

if(currentPageNo < 1){

currentPageNo = 1;

}else if(currentPageNo > totalPageCount){

currentPageNo = totalPageCount;

}

if(StringUtils.isEmpty(userName)) {

userName="";

}

List<User> users=userService.getUserList(userName,roleName,(currentPageNo-1)\*pageSize,pageSize);

md.addAttribute("userList", users);

md.addAttribute("totalPageCount", totalPageCount);

md.addAttribute("totalCount", totalCount);

md.addAttribute("currentPageNo", currentPageNo);

return "userList";

}

@RequestMapping("/viewUser")

public String viewUser(String id,Model md) {

User user=new User();

user=userService.getUserById(id);

md.addAttribute("user", user);

return "viewUser";

}

}

## 6.4 核心代码（四）

package cn.smbms.controller;

import javax.servlet.http.HttpSession;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.RequestParam;

import org.springframework.web.servlet.ModelAndView;

import cn.smbms.pojo.User;

@Controller

public class IndexController{

//用户登录处理界面

@RequestMapping("/userLoginGo")

public ModelAndView userLoginGo(User user,HttpSession session) {

ModelAndView mv =new ModelAndView();

mv.addObject("user",user);

session.setAttribute("userName","ddd");

mv.setViewName("success");

return mv;

}

//用户登录界面

@RequestMapping("/userLogin")

public String userLogin() {

return "userLogin";

}

@RequestMapping("/input")

public String input() {

return "input";

}

//@RequestMapping({"/index","/","/my"})

@RequestMapping(value="/index",method=RequestMethod.GET)

public String index(@RequestParam(value="username",required=false) String username,Model md) {

System.out.println("Hello SpringMVC");

md.addAttribute("username",username);

return "index";

}

@RequestMapping("/welcome")

public ModelAndView welcome(String username) {

ModelAndView mv=new ModelAndView();

mv.addObject("username",username);

mv.setViewName("welcome");

return mv;

}

}