

电商平台后台管理系统

目录

- 1 绪论 1
- 2 需求分析 2
 - 2.1 需求分析 2
 - 2.2 模块结构 2
- 3 系统设计 3
 - 3.1 开发技术与工具 3
 - 3.2 系统架构 3
 - 3.3 数据库设计 3
- 4 环境搭建与配置文件 7
 - 4.1 环境搭建 7
 - 4.2 配置文件 8
 - 4.3 系统目录结构 9
- 5 详细设计 10
 - 5.1 商品信息模块 10
 - 5.2 商品类型模块 10
 - 5.3 订单信息模块 11
 - 5.4 订单信息模块 12
 - 5.5 订单明细模块 12
 - 5.6 客户管理模块 13
 - 5.7 管理员管理模块 13
 - 5.8 菜单管理模块 14
- 6 使用方法说明 16
 - 6.1 系统登录与退出 16
 - 6.2 商品管理 17
 - 6.3 订单管理 21
 - 6.4 客户管理 24
 - 6.5 管理员管理 25
 - 6.6 菜单管理 27
 - 6.7 数据监控 29

1 绪论

一个电商网站，既在于它的网站前台设计制作的精良优美，更在于电商平台后台的实用性和便捷性。拥有一个功能齐全的后台管理系统，这方便我们查询商品信息、订单信息、客户信息等信息。电商网站运营过程中，从客户商品的浏览和下单到付款成功后，商品的数量也会随着客户的购买而减少，这需要获取商品的库存量、是否在售等信息；下单成功后，后台管理人员需要随时了解订单的状态等信息，这些都需要一个管理系统来管理这些信息。所以，开发一个电商平台管理系统是很有必要的，这会大大提高我们的工作效率和工作进度。

2 需求分析

2.1 需求分析

1. 商品管理：包括商品添加、商品删除、商品查询、商品修改和商品下架功能。
2. 商品类型管理：包括商品类型添加、商品类型删除和商品类型修改功能。
3. 订单管理：包括订单创建、订单查询、订单删除和查看订单明细功能。
4. 客户管理：包括查询客户、停用或启动客户功能。
5. 管理员管理：包括添加管理员、删除管理员和修改管理员权限功能。
6. 菜单管理：包括创建菜单、删除菜单、修改菜单功能页面和禁用或启动菜单功能。
7. 数据监控：包括 Druid 界面，可以查看 SQL/ URL /方法 的请求次数，耗时等等统计信息功能。

2.2 模块结构

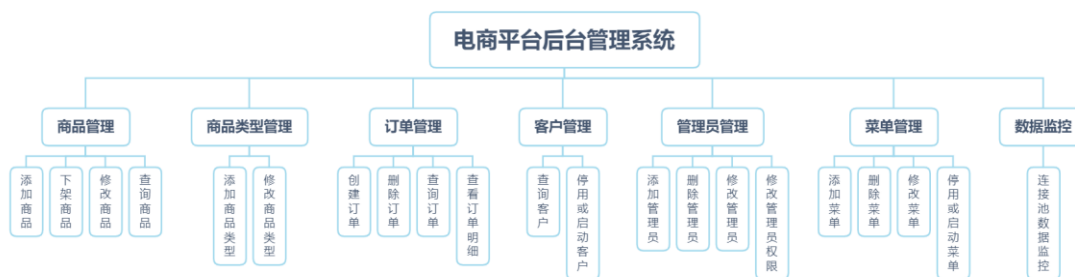


图 3.1 系统模块结构图

3 系统设计

3.1 开发技术与工具

1. 开发技术:

- 后端: SSM(Spring、SpringMVC、Mybatis)框架
- 前端: Bootstrap、Layui、Vue、Axios、JQuery
- 数据库: MySQL
- 项目管理工具: Maven

2. 开发工具:

- IntelliJ IDEA
- Navicat

3.2 系统架构

该项目采用的是 BS 架构, 即浏览器和服务器架构模式。在这种架构下, 用户通过访问 WEB 浏览器, 浏览器发送用户请求到 WEB 服务器, 服务器通过访问数据库, 数据库通过 SQL 查询返回结果给服务器, 接着响应给浏览器, 最后返回给用户显示数据。

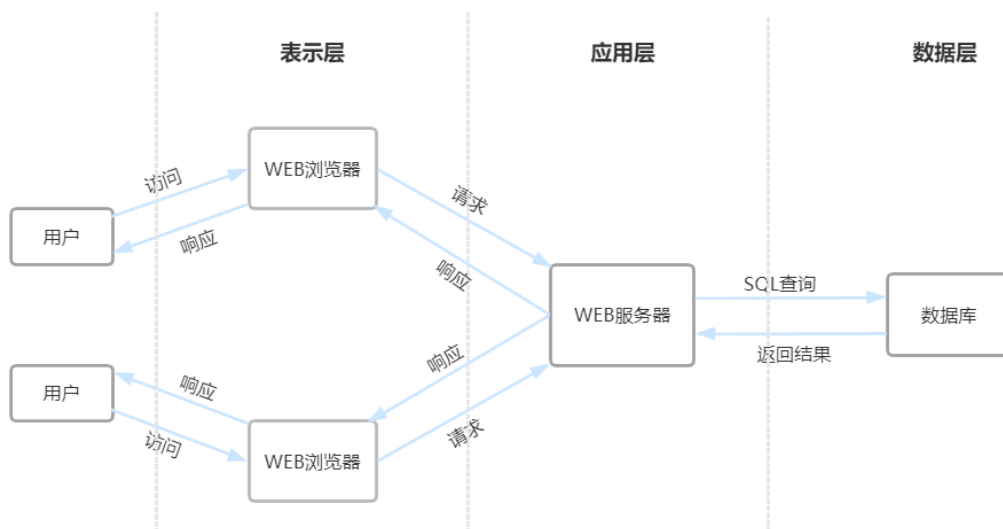


图 3.2 BS 架构图

3.3 数据库设计

1. 数据库表

表 3.1 管理员信息表 admin_info

字段名	类型	主外键	备注
id	int	PK	管理员 id

admin_name	varchar(20)	管理员登录账号
admin_pwd	varchar(20)	管理员登录密码
status	char(1)	管理员状态
create_time	datetime	创建时间
update_time	datetime	更新时间

表 3.2 系统功能表 function

字段名	类型	主外键	备注
id	int	PK	系统功能 id
parent_id	int		父结点 id
fc_name	varchar(20)		系统功能名称
fc_url	varchar(50)		功能页面或 url
status	char(1)		系统功能状态
create_time	datetime		创建时间
update_time	datetime		更新时间

表 3.3 商品类型表 product_type

字段名	类型	主外键	备注
id	int	PK	商品类型 id
type_name	varchar(20)		商品类型名称
create_time	datetime		创建时间
update_time	datetime		更新时间

表 3.4 商品信息表 product_info

字段名	类型	主外键	备注
id	int	PK	商品信息 id
pd_code	varchar(20)		商品编号
pd_name	varchar(20)		商品名称
tid	int	FK	商品类别 id
brand	varchar(20)		商品品牌
image	varchar(255)		商品图片 url
price	varchar(20)		商品价格
num	int		商品数量
remark	varchar(255)		商品描述
status	char(1)		商品状态

create_time	datetime	创建时间
update_time	datetime	更新时间

表 3.5 客户信息表 user_info

字段名	类型	主外键	备注
id	int	PK	客户信息 id
user_name	varchar(20)		用户登录账号
user_pwd	varchar(20)		用户登录密码
real_name	varchar(20)		真实姓名
sex	char(1)		性别
address	varchar(255)		地址
email	varchar(50)		邮箱
status	char(1)		用户状态
create_time	datetime		创建时间
update_time	datetime		更新时间

表 3.6 订单信息表 order_info

字段名	类型	主外键	备注
id	int	PK	订单信息 id
uid	int	FK	客户 id
order_price	decimal(8, 2)		订单金额
status	char(1)		订单状态
create_time	datetime		创建时间
update_time	datetime		更新时间

表 3.7 订单明细表 order_detail

字段名	类型	主外键	备注
id	int	PK	订单明细 id
oid	int	FK	订单信息 id
pid	int	FK	商品信息 id
num	int		数量
create_time	datetime		创建时间
update_time	datetime		更新时间

表 3.8 管理员权限表 permission

字段名	类型	主外键	备注
id	int	PK	权限 id
aid	int	FK	管理员 id
fid	int	FK	功能菜单 id

2. 数据库 ER 图

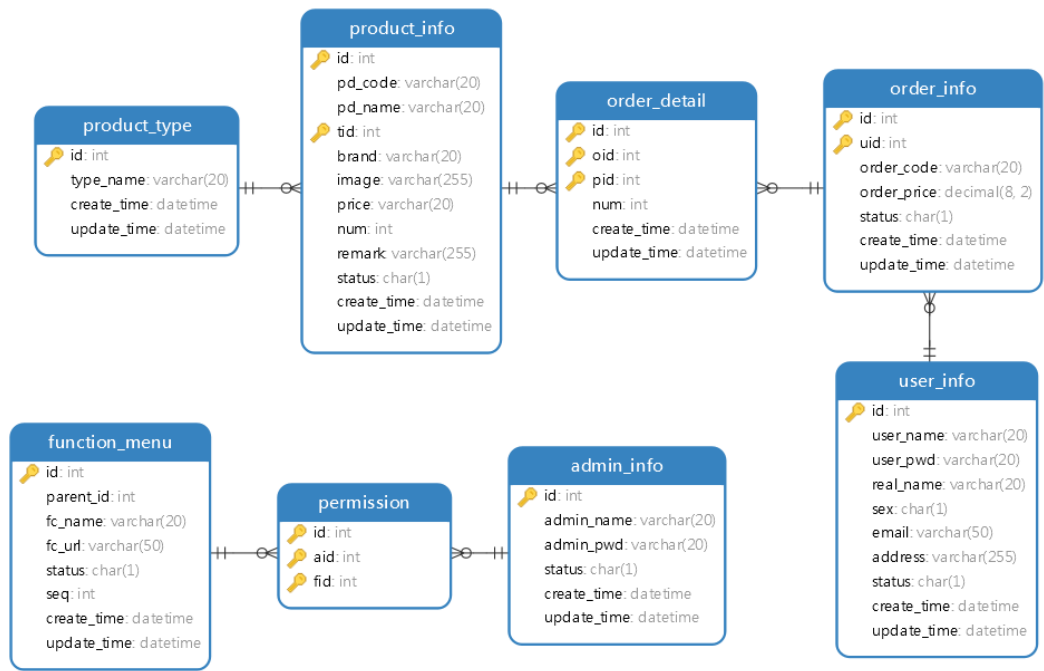


图 3.1 ER 图

4 环境搭建与配置文件

4.1 环境搭建

1. 创建项目

打开 IntelliJ IDEA，点击 File，选择 New Project，选择创建 Maven 项目，勾选 Create from archetype，选择 maven-archetype-webapp，点击 Next，填写项目名称和路径，最后点击 Finish，即可创建项目。

2. 添加依赖 jar 包

在 pom.xml 文件中，依次引入如下依赖 jar 包：

Spring 框架：

- spring-context
- spring-core
- spring-beans
- spring-aop
- spring-expression

SpringMVC 框架：

- spring-web
- spring-webmvc

数据库驱动：

- mysql-connector-java

数据库连接池：

- druid

持久化层：

- mybatis
- mybatis-spring

日志框架：

- logback-core
- logback-classic
- slf4j-api

数据校验：

- validation-api
- hibernate-validator

Json 工具包：

- jackson-core-asl
- jackson-mapper-asl

- jackson-datatype-jsr310

文件上传:

- commons-fileupload
- commons-io

Servlet:

- servlet-api

测试:

- junit

其它:

- Lombok

3. 创建系统目录结构

在项目目录下, 依次创建如下表 4.1 的系统目录结构。

表 4.1 系统目录表

系统目录	说明
com.hsb.eshop.common	用于存放公共实体类
com.hsb.eshop.controller	用于存放控制类
com.hsb.eshop.dao	用于存放数据库访问层接口
com.hsb.eshop.interceptor	用于存放拦截器类
com.hsb.eshop.pojo	用于存放实体类
com.hsb.eshop.service	用于存放业务逻辑层接口和实现类
mapper	用于存放 Mybatis 映射文件
static	用于存放静态文件
views	用于存放视图文件

4.2 配置文件

在项目中, 依次创建如下表 4.2 的配置文件, 并写上相应的配置信息。

表 4.2 配置文件表

配置文件	说明
applicationContext.xml	Spring 框架的配置文件
jdbc.properties	存储数据库连接的属性文件
logback.xml	Logback 日志框架配置文件
mybatis-config.xml	Mybatis 配置文件
spring-servlet.xml	SpringMVC 框架配置文件
web.xml	webapp 配置文件
pom.xml	Maven 配置文件

4.3 系统目录结构

项目最终的目录结构如图 4.1 所示。

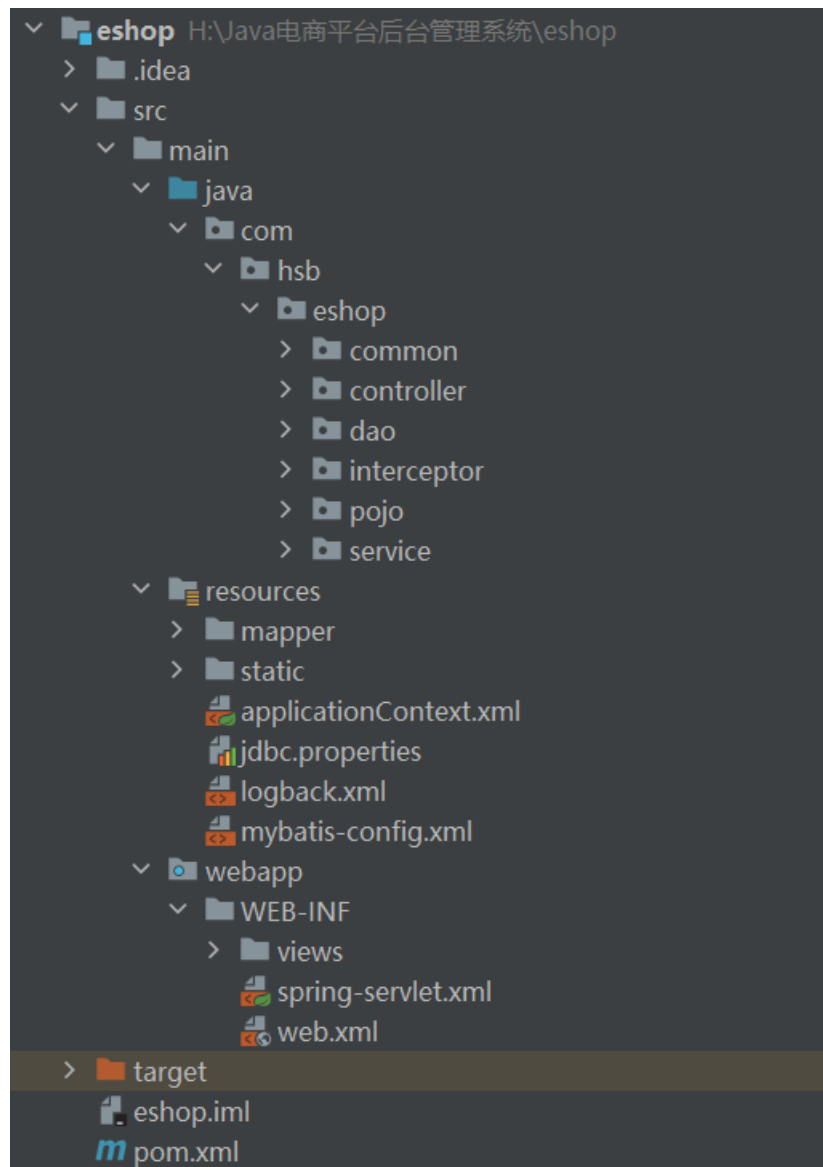


图 4.1 系统目录结构

5 详细设计

5.1 商品信息模块

1. 功能设计

- 添加商品信息功能
- 删除商品信息功能
- 按 id 查询商品信息功能
- 查询所有商品功能
- 按条件查询商品信息功能
- 修改商品信息功能

2. 接口设计

接口名称	URL	接口说明
添加商品接口	/productInfo/insert	post 请求
删除商品接口	/productInfo/delete/{id}	根据 id 删除商品
查询商品接口	/productInfo/query/{id}	根据 id 查询商品
查询所有商品接口	/productInfo/queryAlls	查询所有商品
按条件查询商品接口	/productInfo/queryByCondition	按条件查询商品
修改商品接口	/productInfo/update	根据 id 修改商品
查询 table 数据接口	/productInfo/queryAll	返回表格插件数据

3. 类设计

ProductInfo: 商品信息实体类, 封装商品信息。

ProductInfoDao: 商品信息数据持久化类, 负责与数据库联络, 实现对商品信息的增删查改操作。

ProductInfoService: 商品信息业务逻辑类, 定义商品信息持久化类相应的接口。

ProductInfoServiceImpl: 商品信息业务逻辑实现类, 实现商品信息业务逻辑类的接口。

ProductInfoController: 商品信息业务模块流程控制类, 负责实现相关的 HTTP 请求, 返回 json 数据。

5.2 商品类型模块

1. 功能设计

- 添加商品类型功能
- 删除商品类型功能
- 查询商品类型功能

- 修改商品类型功能

2. 接口设计

接口名称	URL	接口说明
添加商品类型接口	/productType/insert	post 请求
删除商品类型接口	/productType/delete/{id}	根据 id 删除商品类型
查询商品类型接口	/productType/queryAll	根据 id 查询商品类型
修改商品类型接口	/productType/update	根据 id 修改商品类型

3. 类设计

ProductType: 商品类型实体类, 封装商品类型信息。

ProductTypeDao: 商品类型数据持久化类, 负责与数据库联络, 实现对商品类型的增删查改操作。

ProductTypeService: 商品类型业务逻辑类, 定义商品类型持久化类相应的接口。

ProductTypeServiceImpl: 商品类型业务逻辑实现类, 实现商品类型业务逻辑类的接口。

ProductTypeController: 商品类型业务模块流程控制类, 负责实现相关的 HTTP 请求, 返回 json 数据。

5.3 订单信息模块

1. 功能设计

- 查询订单信息功能
- 删除商品类型功能
- 按条件查询订单信息功能

2. 接口设计

接口名称	URL	接口说明
查询订单信息接口	/orderInfo/queryAll	查询所有订单信息
删除订单信息接口	/orderInfo/delete/{id}	根据 id 删除订单信息
按条件查询订单接口	/orderInfo/query	根据条件查询订单

3. 类设计

OrderInfo: 订单信息实体类, 封装订单信息。

OrderInfoDao: 订单信息数据持久化类, 负责与数据库联络, 实现对订单信息的增删查改操作。

OrderInfoService: 订单信息业务逻辑类, 定义订单信息持久化类相应的接口。

OrderInfoServiceImpl: 订单信息业务逻辑实现类, 实现订单信息业务逻辑类的接口。

OrderInfoController: 订单信息业务模块流程控制类, 负责实现相关的 HTTP 请求, 返回 json 数据。

5.4 订单信息模块

1. 功能设计

- 查询订单信息功能
- 删除商品类型功能
- 按条件查询订单信息功能
- 保存订单信息功能

2. 接口设计

接口名称	URL	接口说明
查询订单信息接口	/orderInfo/queryAll	查询所有订单信息
删除订单信息接口	/orderInfo/delete/{id}	根据 id 删除订单信息
按条件查询订单接口	/orderInfo/query	根据条件查询订单
保存订单信息接口	/orderInfo/saveInfo	保存订单信息

3. 类设计

OrderInfo: 订单信息实体类, 封装订单信息。

OrderInfoDao: 订单信息数据持久化类, 负责与数据库联络, 实现对订单信息的增删查改操作。

OrderInfoService: 订单信息业务逻辑类, 定义订单信息持久化类相应的接口。

OrderInfoServiceImpl: 订单信息业务逻辑实现类, 实现订单信息业务逻辑类的接口。

OrderInfoController: 订单信息业务模块流程控制类, 负责实现相关的 HTTP 请求, 返回 json 数据。

5.5 订单明细模块

1. 功能设计

- 添加订单明细功能
- 删除订单明细功能
- 查询订单明细功能

2. 接口设计

接口名称	URL	接口说明
添加订单明细接口	/orderDetail/addDetail	添加订单明细
删除订单明细接口	/orderDetail/delete/{id}	根据 id 删除订单明细
查询订单明细接口	/orderDetail/queryDetail/{id}	根据 id 查询订单明细

3. 类设计

OrderDetail: 订单明细实体类, 封装订单明细信息。

OrderDetailDao: 订单明细数据持久化类, 负责与数据库联络, 实现对订单明细的增删查改操作。

OrderDetailsService: 订单明细业务逻辑类, 定义订单明细持久化类相应的接口。

OrderServiceImpl: 订单明细业务逻辑实现类, 实现订单明细业务逻辑类的接口。

OrderDetailController: 订单明细业务模块流程控制类, 负责实现相关的 HTTP 请求, 返回 json 数据。

5.6 客户管理模块

1. 功能设计

- 查询所有客户信息功能
- 按名称查询客户功能
- 修改客户状态功能

2. 接口设计

接口名称	URL	接口说明
查询所有客户接口	/userInfo/queryAll	查询所有客户信息
按名称查询客户接口	/userInfo/query/{name}	按名称查询客户信息
修改客户状态接口	/userInfo/updateStatus/{id}/{status}	根据 id 修改客户状态

3. 类设计

UserInfo: 客户信息实体类, 封装客户信息。

UserInfoDao: 客户信息数据持久化类, 负责与数据库联络, 实现对客户信息的增删查改操作。

UserInfoService: 客户信息业务逻辑类, 定义客户信息持久化类相应的接口。

UserInfoServiceImpl: 客户信息业务逻辑实现类, 实现客户信息业务逻辑类的接口。

UserInfoController: 客户信息业务模块流程控制类, 负责实现相关的 HTTP 请求, 返回 json 数据。

5.7 管理员管理模块

1. 功能设计

- 添加管理员功能

- 查询所有管理员功能
- 删除管理员功能
- 修改管理员状态功能
- 修改管理员信息功能

2. 接口设计

接口名称	URL	接口说明
添加管理员接口	/adminInfo/insert	post 请求
查询管理员接口	/adminInfo/queryAll	查询所有管理员信息
删除管理员接口	/adminInfo/delete/{id}	根据 id 删除管理员
修改客户状态接口	/adminInfo/updateStatus/{id} /{status}	根据 id 修改管理员状态
修改管理员信息接口	/adminInfo/update/{id}	根据 id 修改管理员

3. 类设计

AdminInfo: 管理员信息实体类, 封装管理员信息。

AdminInfoDao: 管理员信息数据持久化类, 负责与数据库联络, 实现对管理员信息的增删查改操作。

AdminInfoService: 管理员信息业务逻辑类, 定义管理员信息持久化类相应的接口。

AdminInfoServiceImpl: 管理员信息业务逻辑实现类, 实现管理员信息业务逻辑类的接口。

AdminInfoController: 管理员信息业务模块流程控制类, 负责实现相关的 HTTP 请求, 返回 json 数据。

5.8 菜单管理模块

1. 功能设计

- 添加菜单功能
- 删除菜单功能
- 查询所有菜单功能
- 修改菜单状态功能
- 修改菜单信息功能

2. 接口设计

接口名称	URL	接口说明
添加菜单接口	/functionMenu/insert	添加菜单信息
删除菜单接口	/functionMenu/delete/{id}	根据 id 删除菜单
修改菜单状态接口	/functionMenu/updateStatus/{id}/{status}	根据 id 修改菜单状态

查询所有菜单接口	/functionMenu/queryAll	查询所有菜单信息
修改菜单信息接口	/functionMenu/update/{id}	根据 id 修改菜单信息

3. 类设计

FunctionMenu: 菜单信息实体类, 封装菜单信息。

FunctionMenuDao: 菜单信息数据持久化类, 负责与数据库联络, 实现对菜单信息的增删查改操作。

FunctionMenuService: 菜单信息业务逻辑类, 定义菜单信息持久化类相应的接口。

FunctionMenuServiceImpl: 菜单信息业务逻辑实现类, 实现菜单信息业务逻辑类的接口。

FunctionMenuController: 菜单信息业务模块流程控制类, 负责实现相关的 HTTP 请求, 返回 json 数据。

6 使用方法说明

6.1 系统登录与退出

管理员通过登录名和密码进行系统的登录，若表单为空或登录名或密码不正确，将会有相应的提示信息；登录成功后，页面跳转到系统的首页；点击页面的右上角的下拉菜单，选择点击退出即可退出系统。

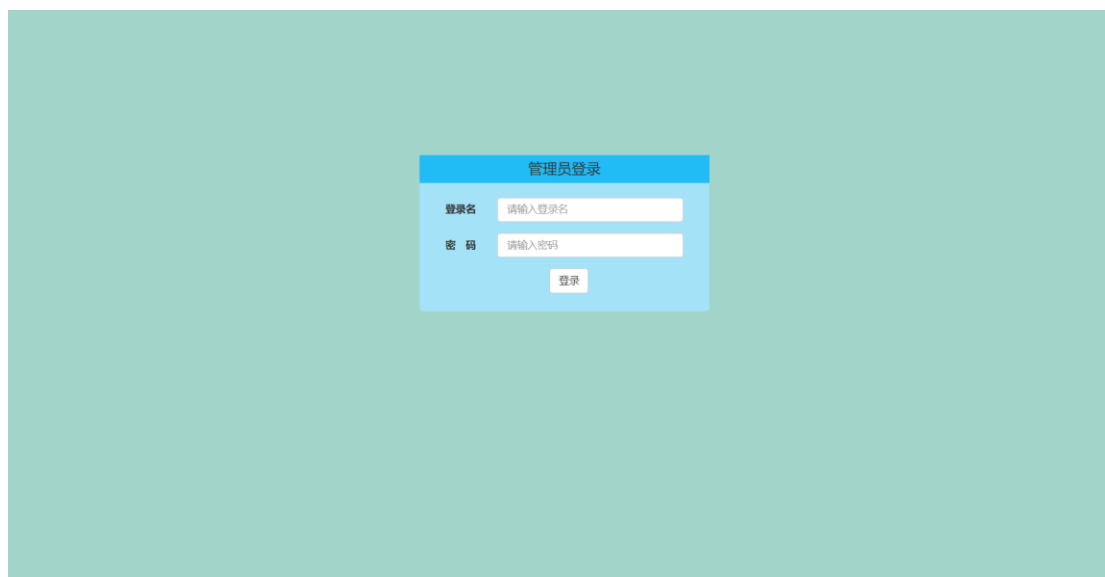


图 6.1.1 登录界面



图 6.1.2 错误信息提示



图 6.1.3 登录成功页面

6.2 商品管理

1. 商品列表

分页展示商品信息，根据自己的需求可以改变每页展示的记录数目；可以根据商品的编号、名称、类型、品牌和价格进行商品的搜索；选中表格的商品可以实现商品的修改和删除操作；添加商品时，若表单为空或不符合要求，将会有相应的提示信息；每当操作完成会有相应的回显提示信息。



图 6.2.1 商品列表



图 6.2.2 搜索商品

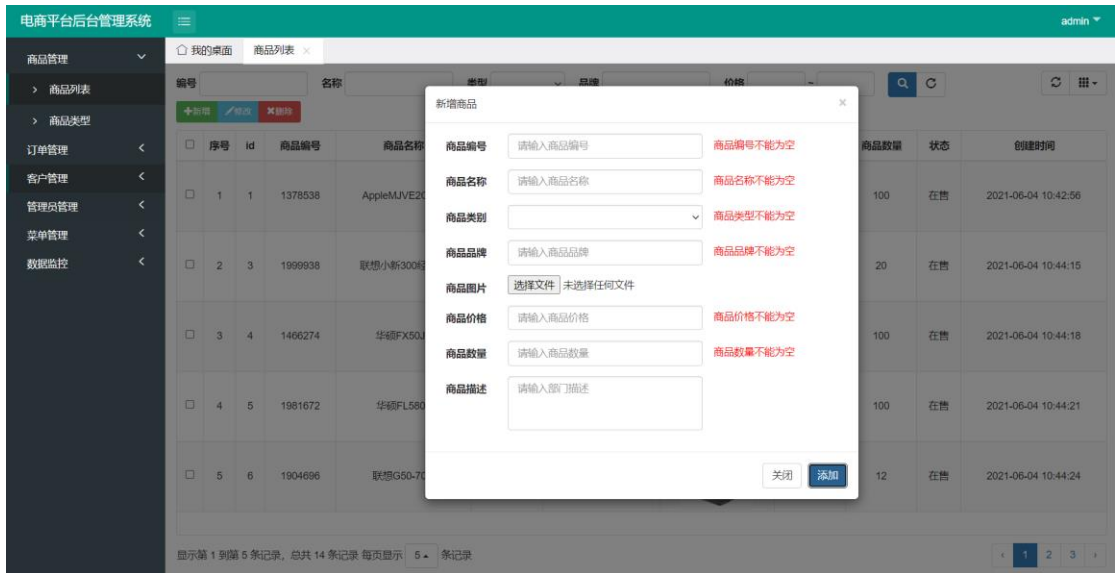


图 6.2.3 添加商品

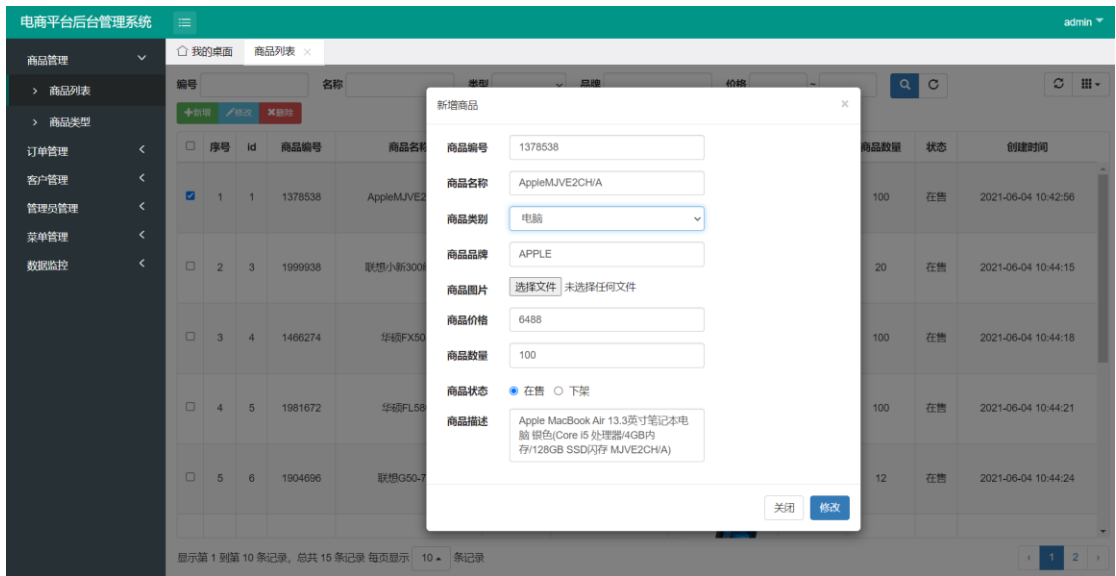


图 6.2.4 修改商品

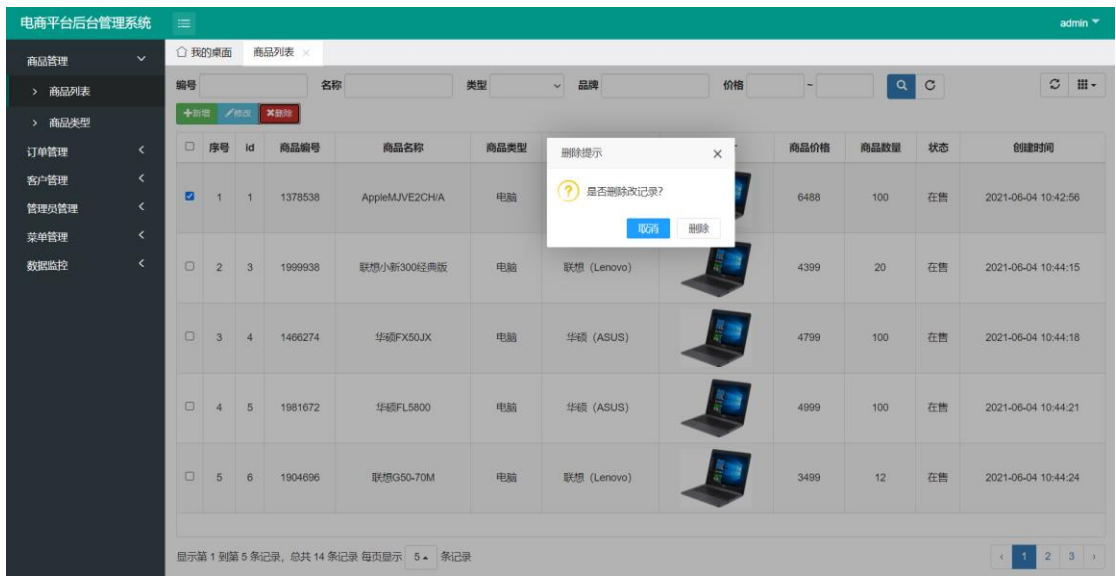


图 6.2.5 删除商品

2. 商品类型

显示商品的类型信息，选中表格的类型可以实现商品的修改和删除功能；点击新增，可以添加商品类型，每当操作完成会有相应的回显提示信息。。

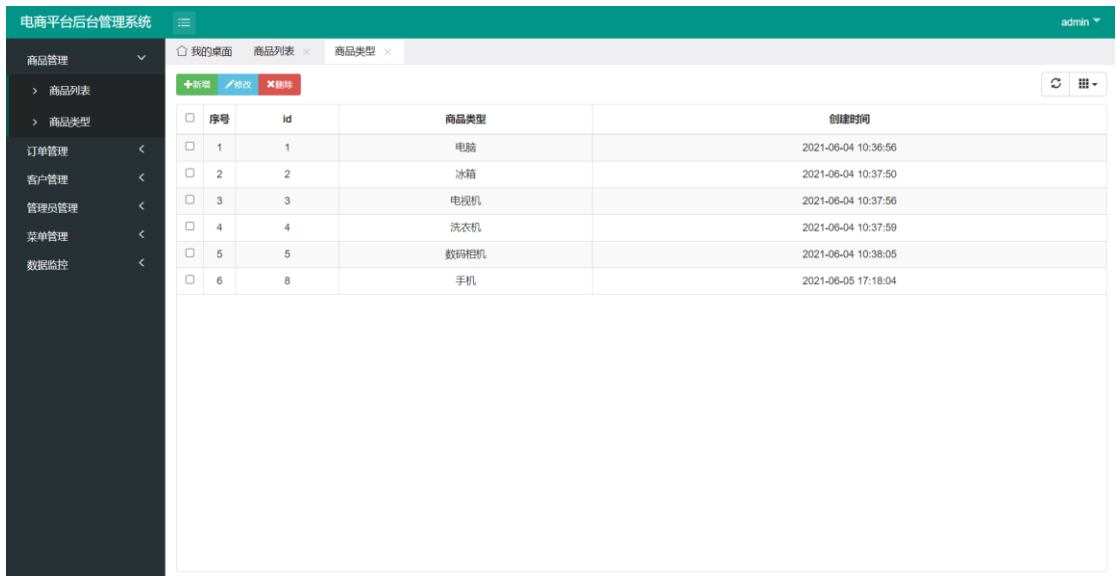


图 6.2.6 商品类型

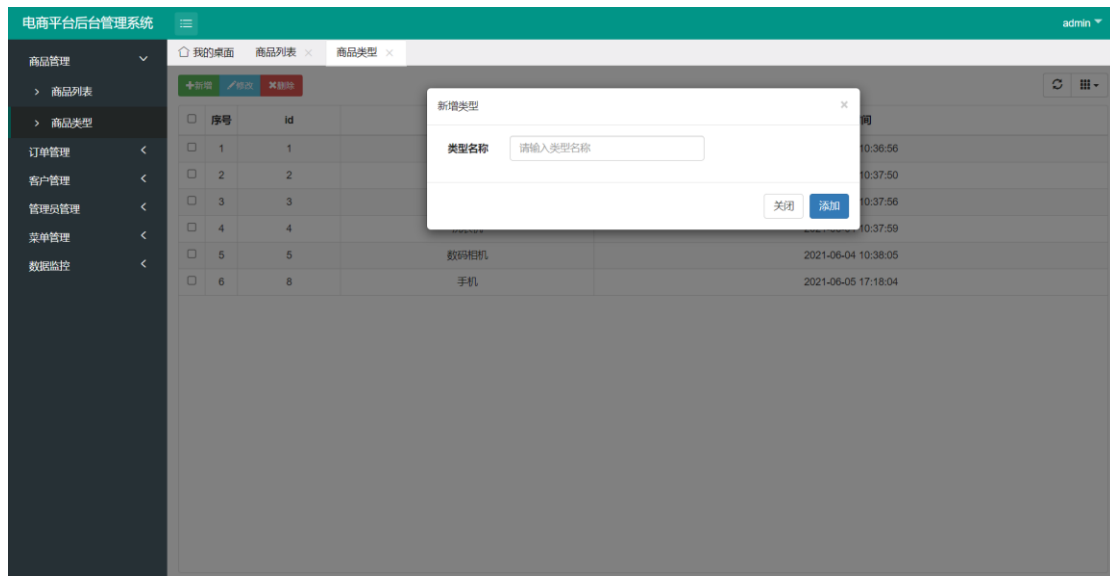


图 6.2.7 添加商品类型

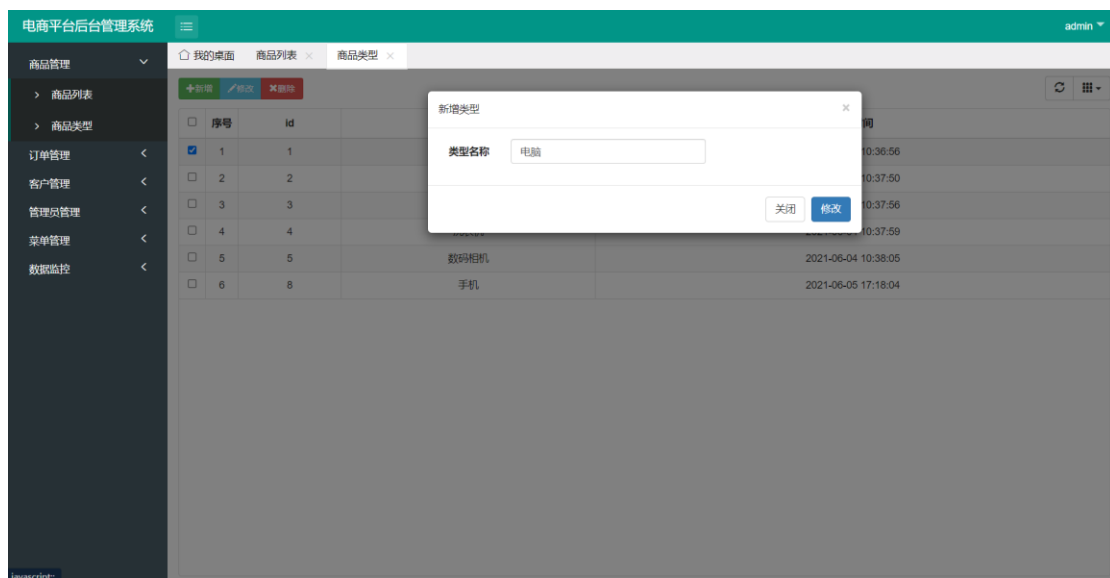


图 6.2.8 修改商品类型

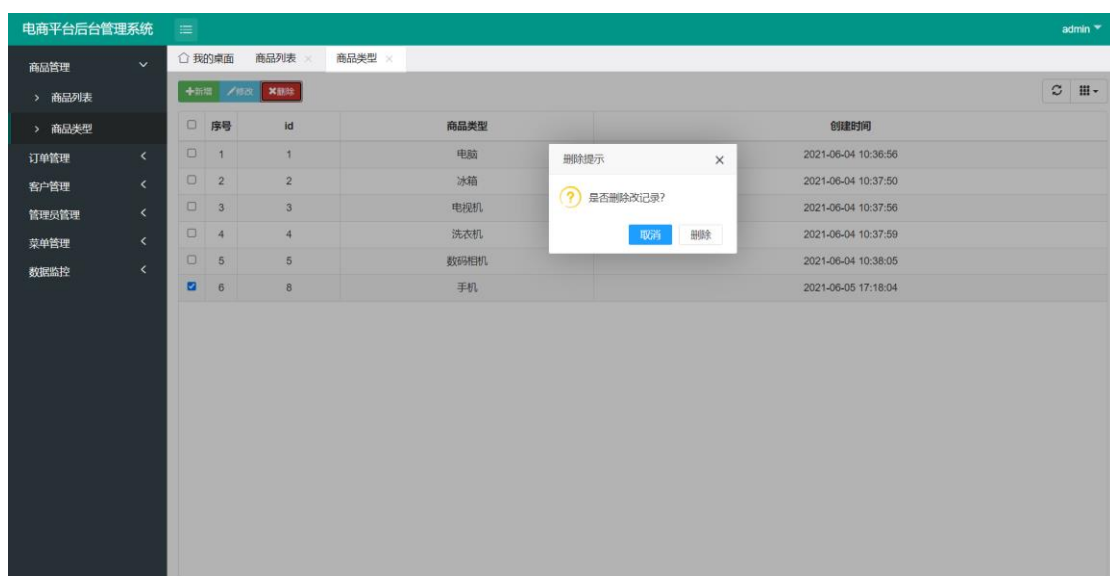
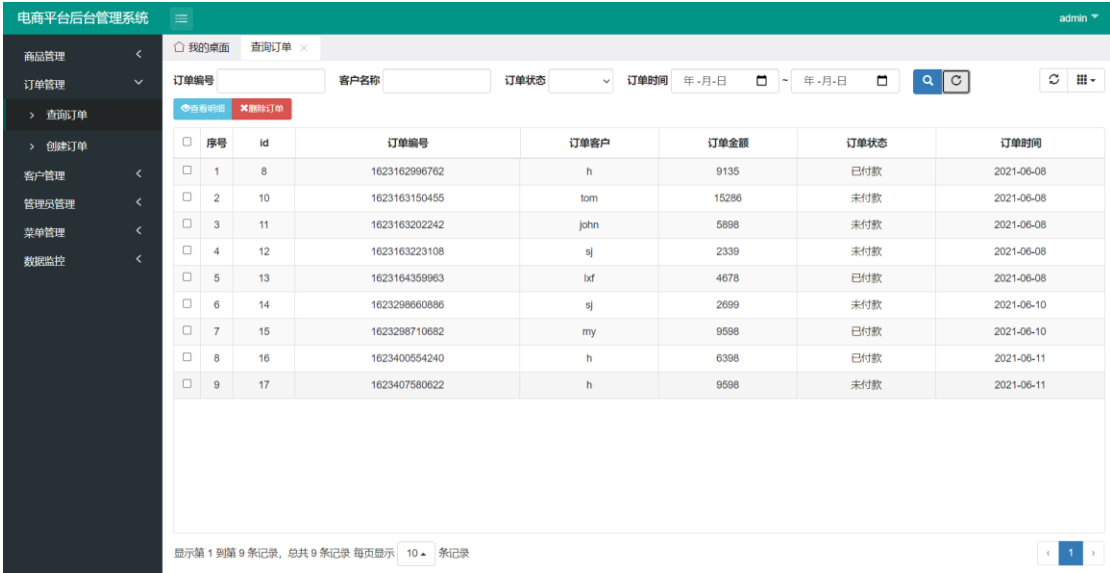


图 6.2.9 删除商品类型

6.3 订单管理

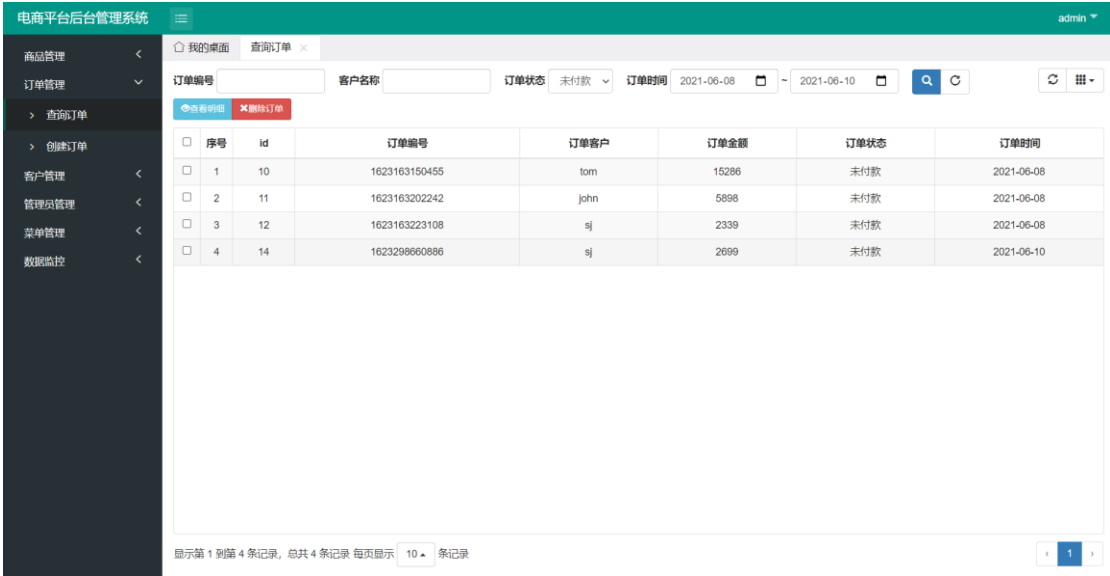
1. 查询订单

分页显示订单信息，可以改变每页显示的记录数目；可以根据订单的编号、客户名称、订单状态和订单时间来查找相应的订单信息；选中表格的复选框可以查询订单明细和删除订单；每当操作完成会有相应的回显提示信息。



序号	id	订单编号	订单客户	订单金额	订单状态	订单时间
1	8	1623162996762	h	9135	已付款	2021-06-08
2	10	1623163150455	tom	15286	未付款	2021-06-08
3	11	1623163202242	john	5898	未付款	2021-06-08
4	12	1623163223108	sj	2339	未付款	2021-06-08
5	13	1623164359963	lxf	4678	已付款	2021-06-08
6	14	1623298660886	sj	2699	未付款	2021-06-10
7	15	1623298710682	my	9598	已付款	2021-06-10
8	16	1623400554240	h	6398	已付款	2021-06-11
9	17	1623407580622	h	9598	未付款	2021-06-11

图 6.3.1 订单列表



序号	id	订单编号	订单客户	订单金额	订单状态	订单时间
1	10	1623163150455	tom	15286	未付款	2021-06-08
2	11	1623163202242	john	5898	未付款	2021-06-08
3	12	1623163223108	sj	2339	未付款	2021-06-08
4	14	1623298660886	sj	2699	未付款	2021-06-10

图 6.3.2 查找订单

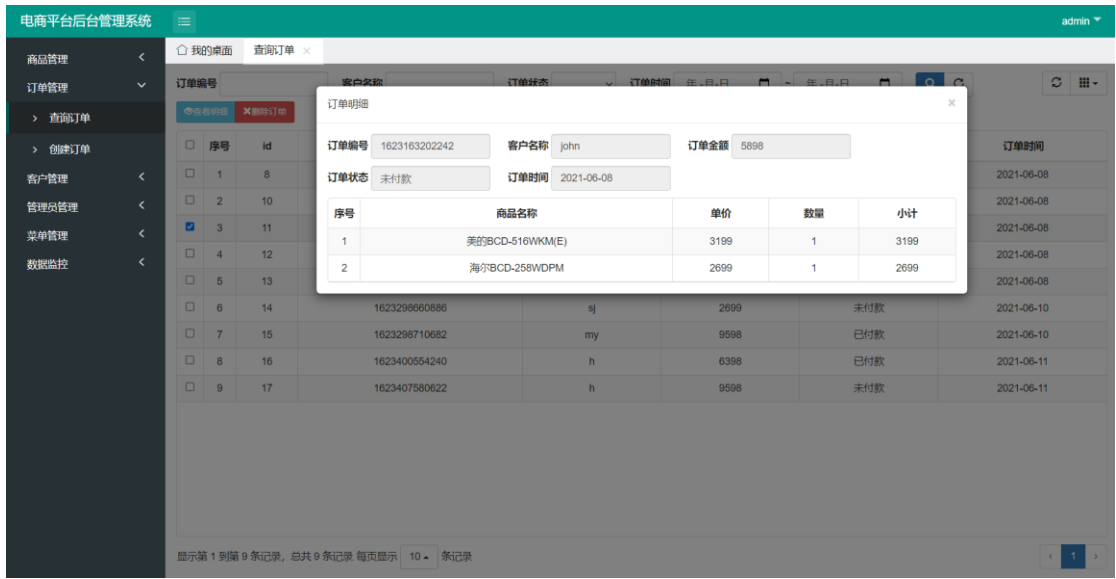


图 6.3.3 查看订单明细

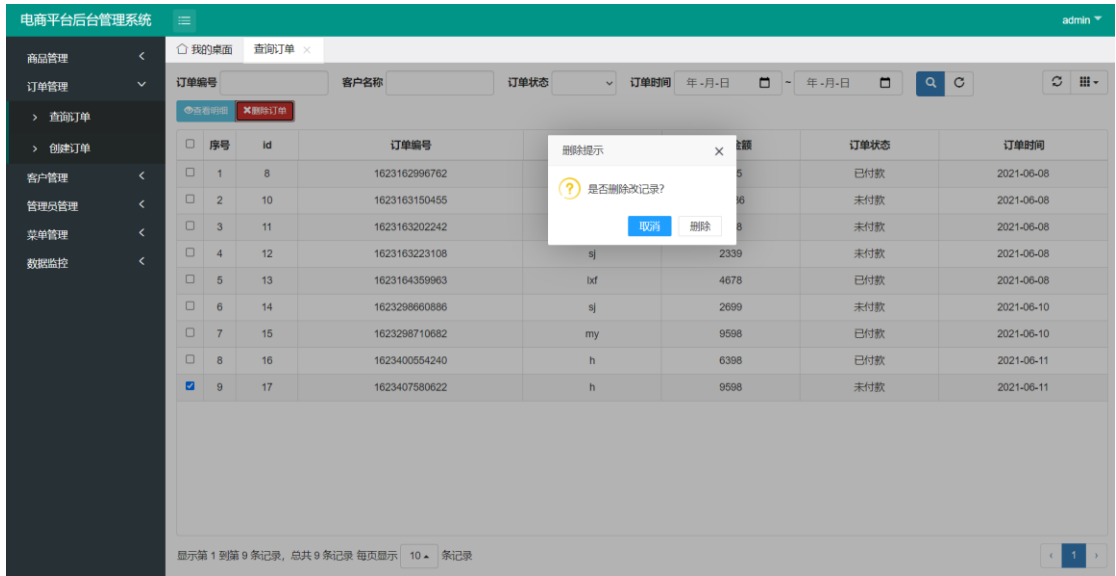


图 6.3.4 删除订单

2. 创建订单

创建订单信息，点击添加订单明细按钮可以添加选择商品以及数量，添加完成后，在表格显示选择的商品信息；选中表格的复选框，点击删除按钮可以删除该商品订单明细；添加商品明细完成后，填上表单信息，勾选商品的订单明细，订单金额将会自动统计，最后点击保存订单按钮，可以保存该订单的信息。

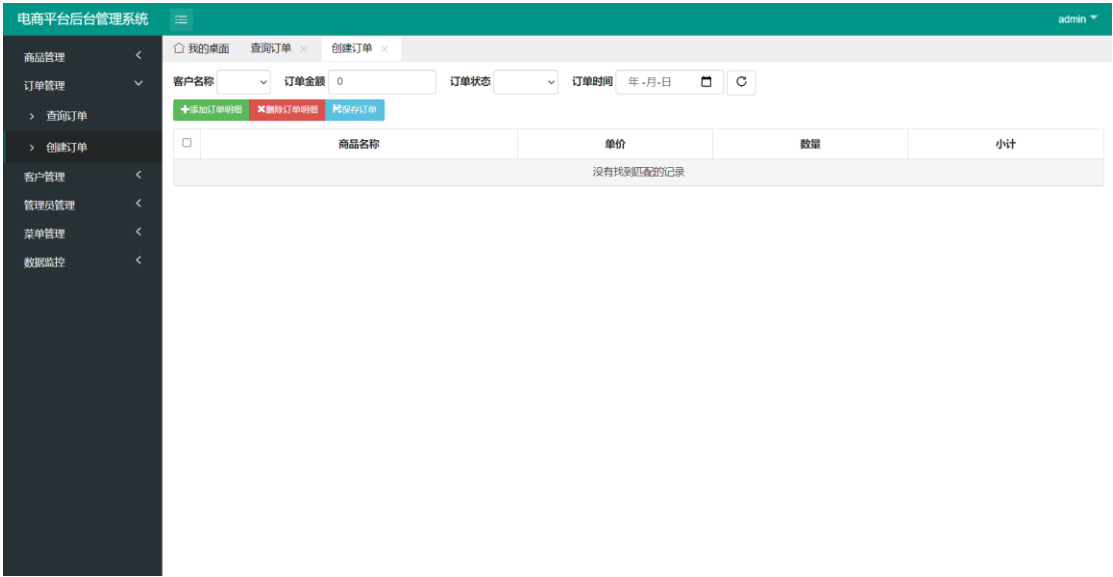


图 6.3.5 创建订单

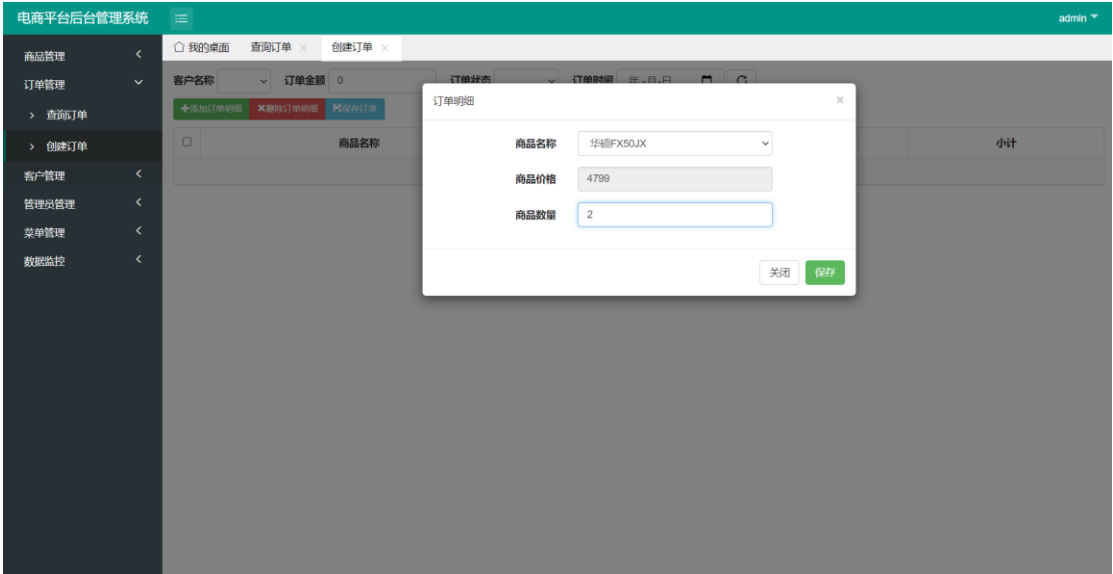


图 6.3.6 添加订单明细

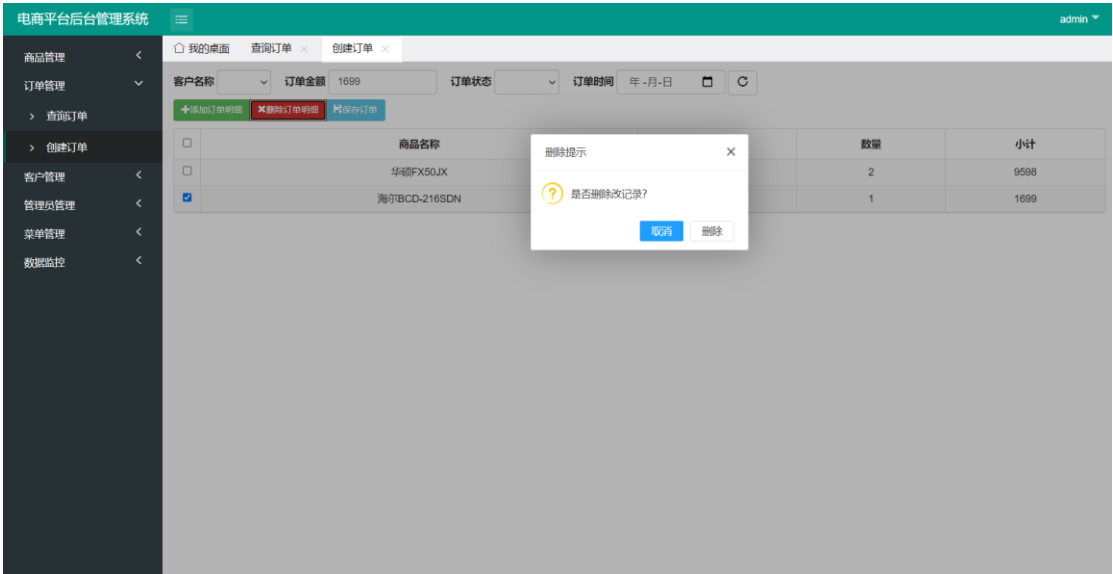


图 6.3.6 删除订单明细

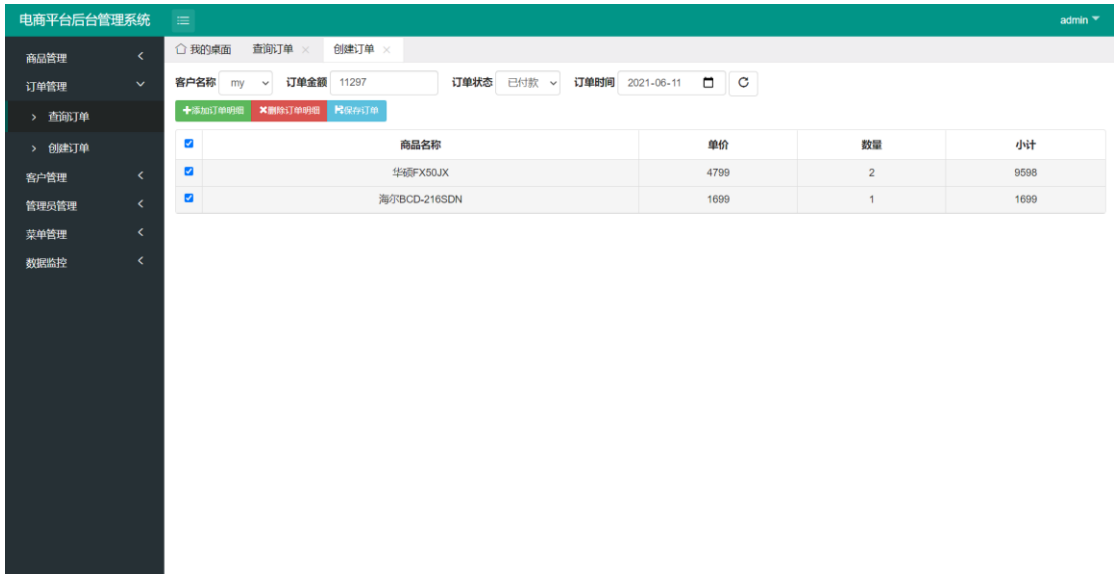


图 6.3.6 保存订单

6.4 客户管理

分页展示客户信息，可以改变每页显示的记录数目；可以根据客户名称查询客户；点击表格状态栏的开关按钮，可以改变客户的启动和停用状态。

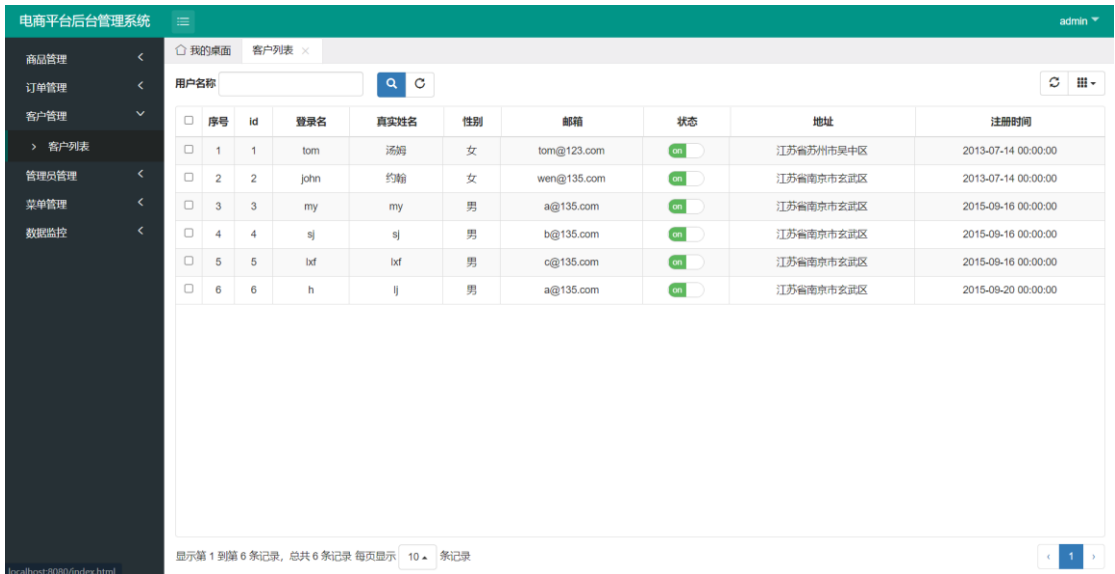


图 6.4.1 客户列表

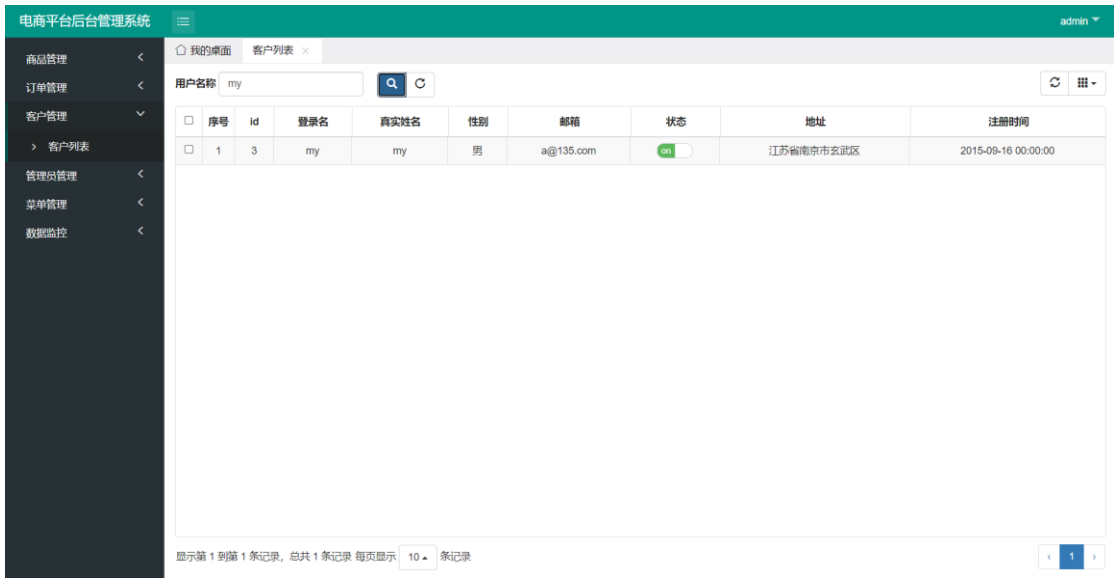


图 6.4.2 查询客户

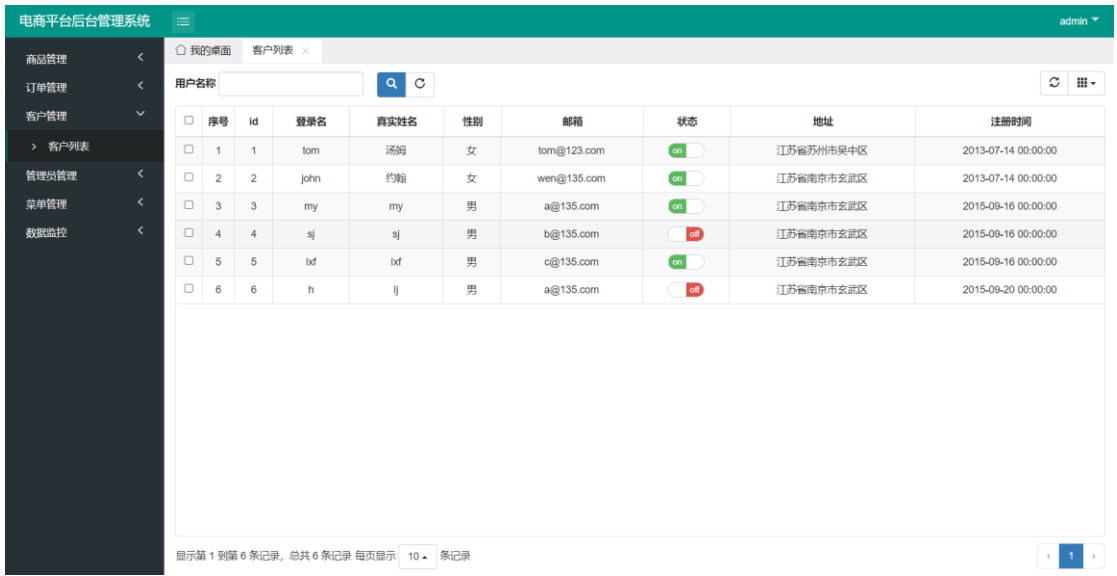


图 6.4.3 停用客户

6.5 管理员管理

展示管理员信息，点击新增按钮可以添加管理员，可以选中相应的权限功能；选中表格的复选框可以进行修改和删除管理员操作，其中，修改操作可以修改管理员的登录名、密码和相应的菜单权限；点击表格状态栏的开关按钮，可以改变管理员的启动和停用状态。

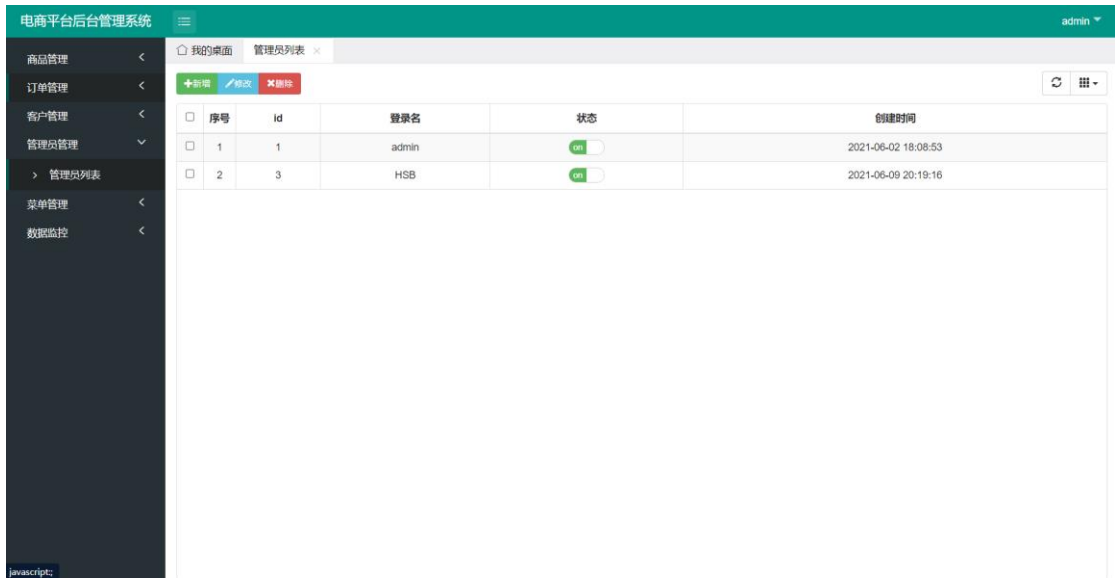


图 6.5.1 管理员列表

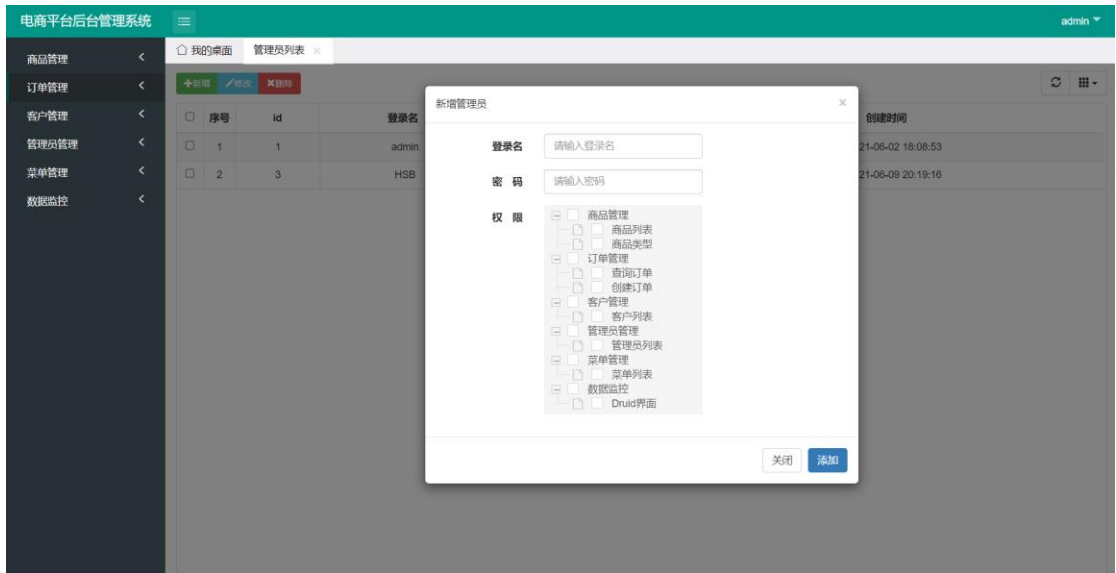


图 6.5.2 添加管理员

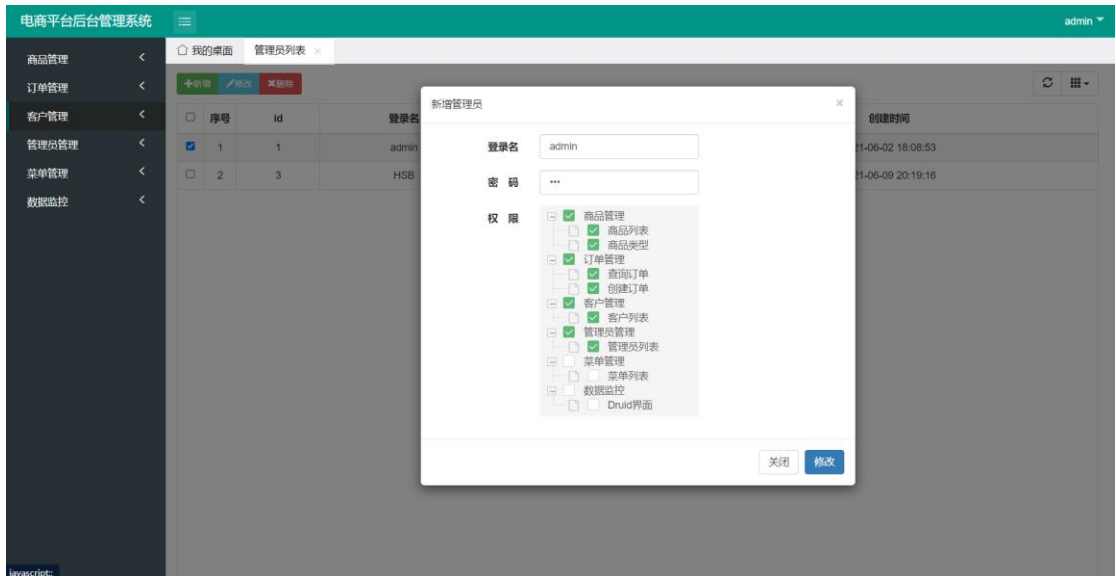


图 6.5.3 修改管理员

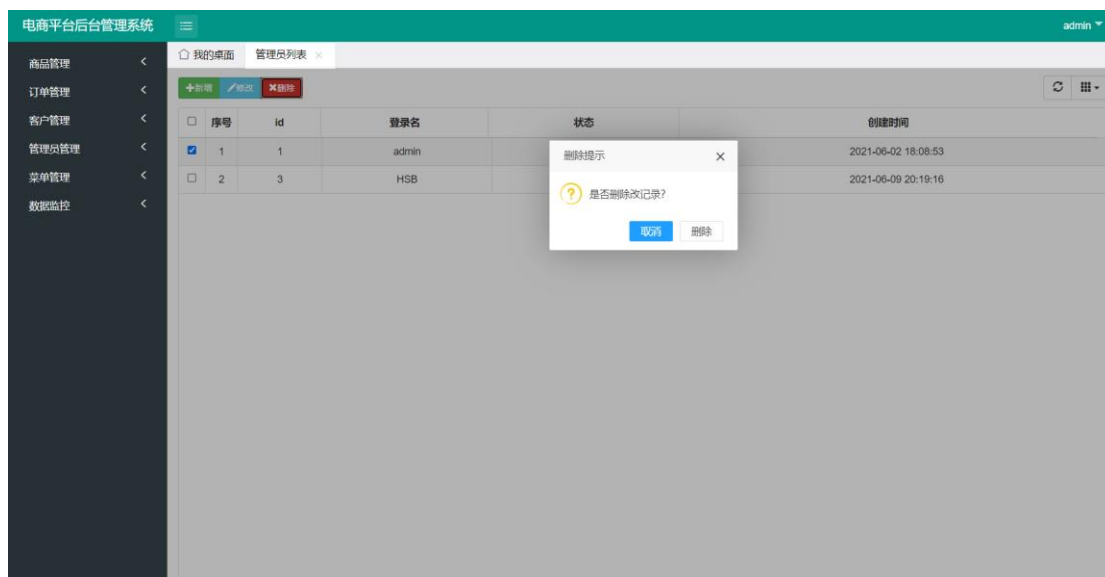


图 6.5.4 删除管理员

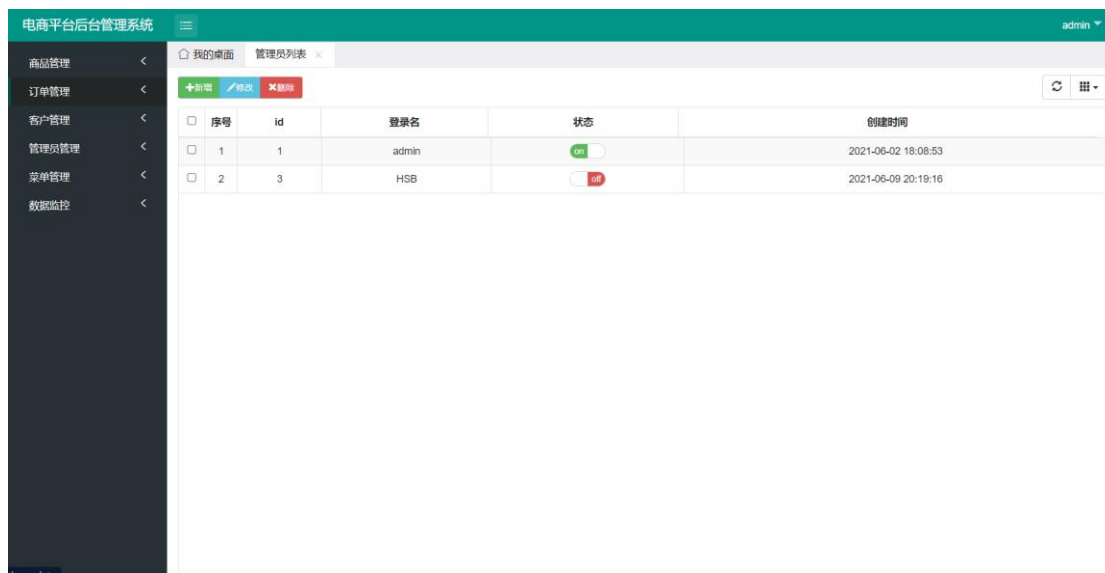


图 6.5.5 改变管理员状态

6.6 菜单管理

展示菜单功能信息，点击新增按钮可以添加新菜单；选中表格的复选框可以进行菜单的修改和删除操作；点击表格状态栏的开关按钮，可以改变菜单的启动和停用状态，若菜单处于停用状态，则页面左边的菜单栏将不会显示该菜单及相应的功能。

id	父结点id	菜单名称	功能页面	状态	创建时间
16	0	数据监控		on	2021-06-09 19:19:32
11	0	菜单管理		on	2021-06-02 20:43:26
9	0	管理员管理		on	2021-06-02 19:58:24
3	0	客户管理		on	2021-06-02 19:11:22
2	0	订单管理		on	2021-06-02 19:10:56
1	0	商品管理		on	2021-06-02 19:10:36
4	1	商品列表	product_page	on	2021-06-02 19:12:24
5	1	商品类型	productType_page	on	2021-06-02 19:12:49
6	2	查询订单	orderInfo_page	on	2021-06-02 19:13:05
7	2	创建订单	orderCreate_page	on	2021-06-02 19:13:20
8	3	客户列表	user_page	on	2021-06-02 19:13:36
10	9	管理员列表	admin_page	on	2021-06-02 19:59:24
12	11	菜单列表	menu_page	on	2021-06-02 21:01:18
17	16	Druid界面	druid	on	2021-06-09 19:27:22

图 6.6.1 菜单列表

新增菜单

菜单名称: 请输入菜单名称

上一级菜单: 选择菜单

功能页面: 请输入页面URL

关闭 添加

图 6.6.2 添加菜单

新增菜单

菜单名称: 商品类型

上一级菜单: 商品管理

功能页面: productType_page

关闭 修改

图 6.6.3 修改菜单

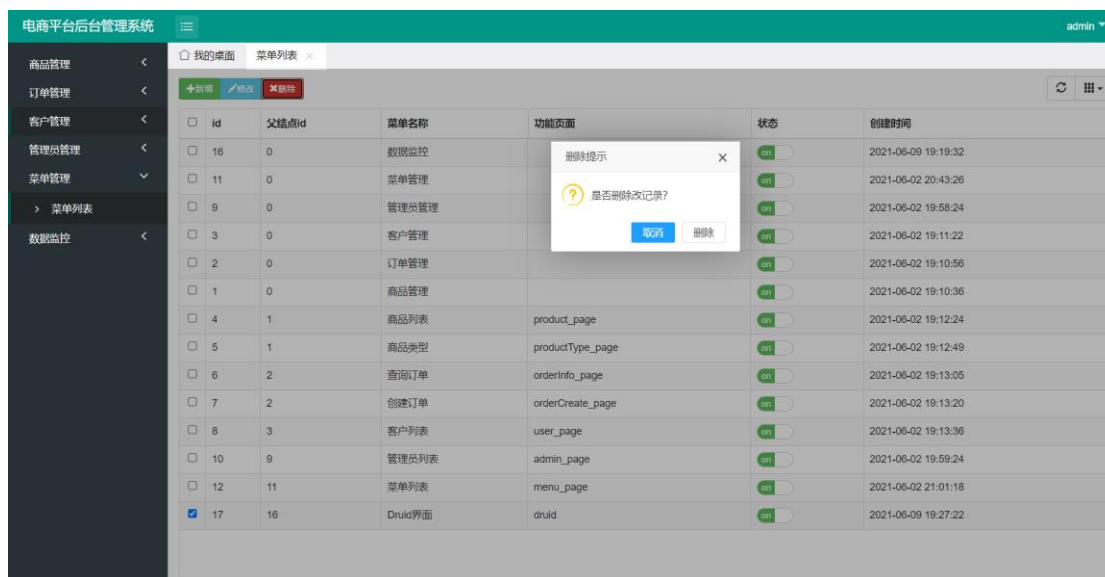


图 6.6.4 删除菜单

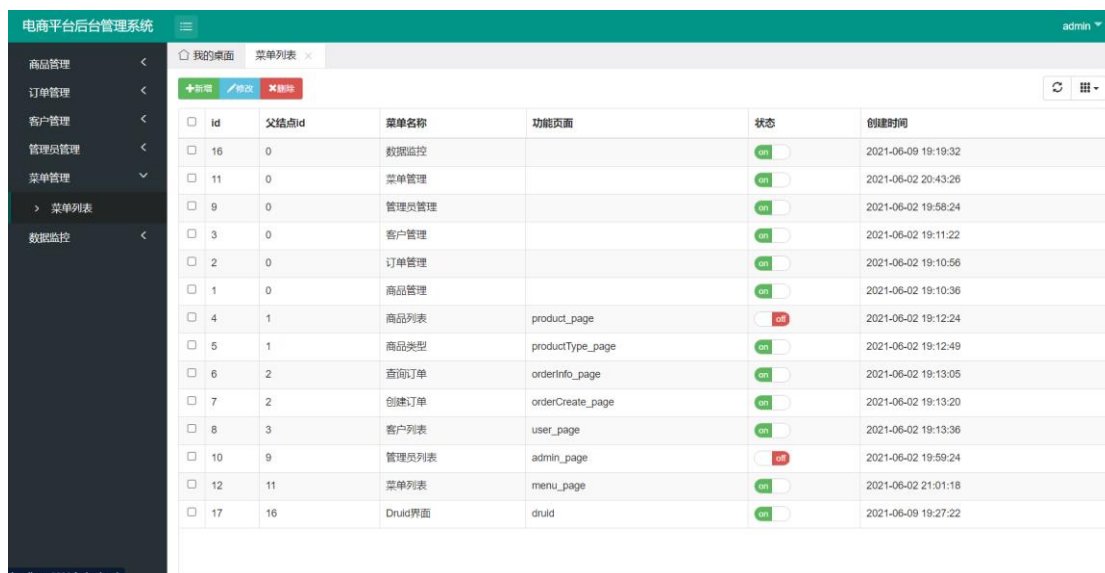


图 6.6.5 改变菜单状态

6.7 数据监控

本项目使用的是阿里巴巴的 Druid 连接池，Druid 连接池为监控而生，内置强大的监控功能，监控特性不影响性能。Druid 提供了监控页面，可以查看 SQL/URL/方法 的请求次数，耗时等等统计信息。

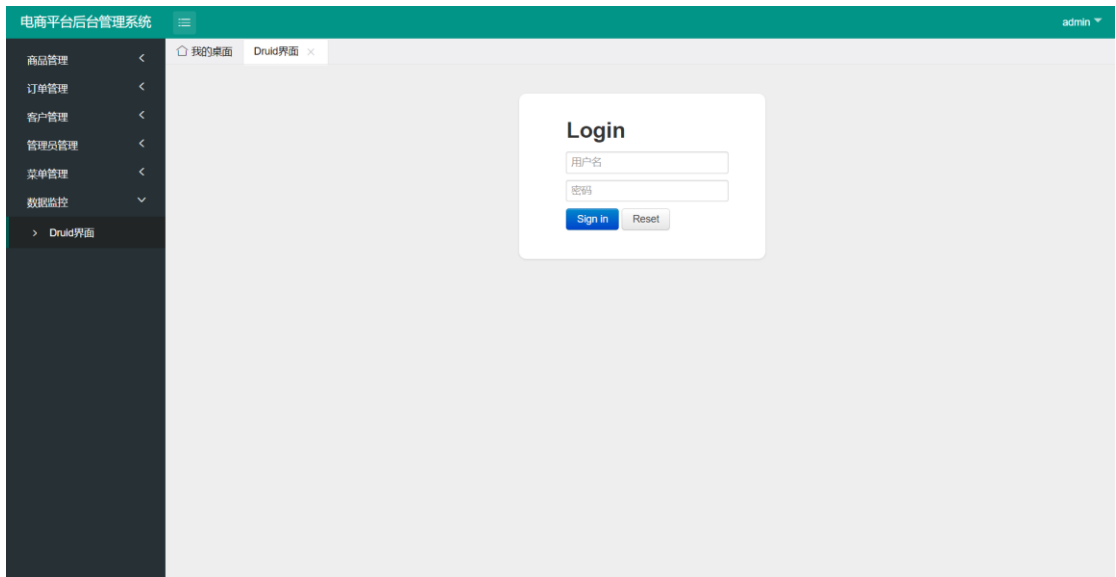


图 6.7.1 Druid 登录界面

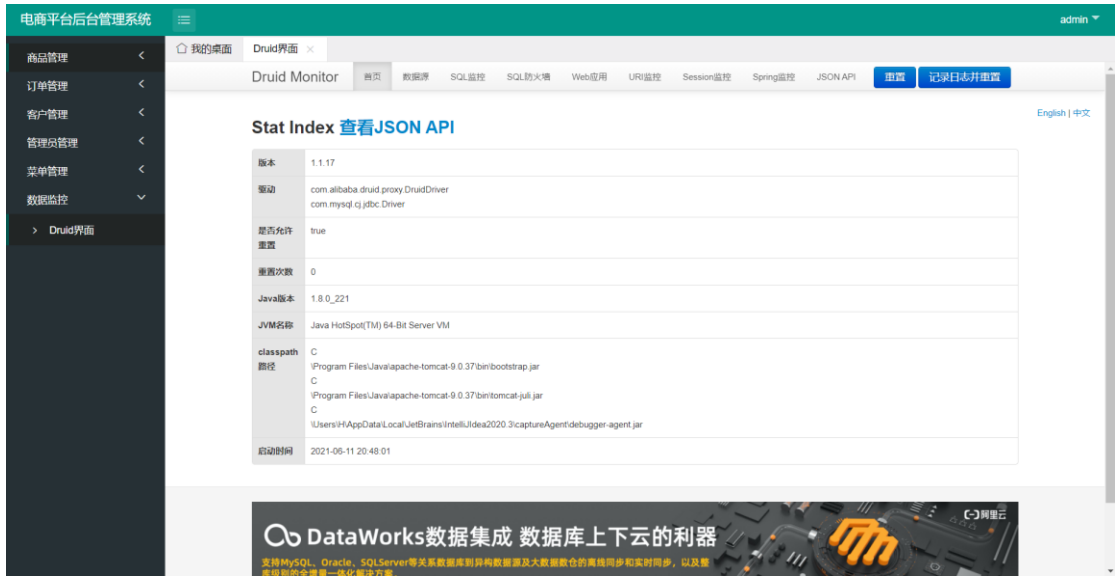


图 6.7.2 Druid 首页