电商平台后台管理系统

目录

[1 绪论](#_bookmark1) [1](#_bookmark2)

[2 需求分析](#_bookmark3) [2](#_bookmark4)

[2.1 需求分析](#_bookmark5) [2](#_bookmark6)

[2.2 模块结构](#_bookmark7) [2](#_bookmark8)

[3 系统设计](#_bookmark9) [3](#_bookmark10)

[3.1 开发技术与工具](#_bookmark11) [3](#_bookmark12)

[3.2 系统架构](#_bookmark13) [3](#_bookmark14)

[3.3 数据库设计](#_bookmark15) [3](#_bookmark16)

[4 环境搭建与配置文件](#_bookmark17) [7](#_bookmark18)

[4.1 环境搭建](#_bookmark19) [7](#_bookmark20)

[4.2 配置文件](#_bookmark21) [8](#_bookmark22)

[4.3 系统目录结构](#_bookmark23) [9](#_bookmark24)

[5 详细设计](#_bookmark25) [10](#_bookmark26)

[5.1 商品信息模块](#_bookmark27) [10](#_bookmark28)

[5.2 商品类型模块](#_bookmark29) [10](#_bookmark30)

[5.3 订单信息模块](#_bookmark31) [11](#_bookmark32)

[5.4 订单信息模块](#_bookmark33) [12](#_bookmark34)

[5.5 订单明细模块](#_bookmark35) [12](#_bookmark36)

[5.6 客户管理模块](#_bookmark37) [13](#_bookmark38)

[5.7 管理员管理模块](#_bookmark39) [13](#_bookmark40)

[5.8 菜单管理模块](#_bookmark41) [14](#_bookmark42)

[6 使用方法说明](#_bookmark43) [16](#_bookmark44)

[6.1 系统登录与退出](#_bookmark45) [16](#_bookmark46)

[6.2 商品管理](#_bookmark47) [17](#_bookmark48)

[6.3 订单管理](#_bookmark49) [21](#_bookmark50)

[6.4 客户管理](#_bookmark51) [24](#_bookmark52)

[6.5 管理员管理](#_bookmark53) [25](#_bookmark54)

[6.6 菜单管理](#_bookmark55) [27](#_bookmark56)

[6.7 数据监控](#_bookmark57) [29](#_bookmark58)

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的了解订单 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。

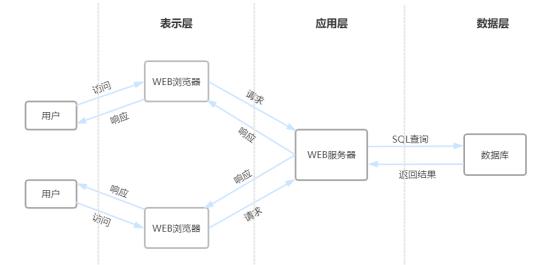


图 3.2 BS 架构图

3.3 数据库设计

1.数据库表

表 3.1 管理员信息表 admin\_info

主外键



PK

管理员 id

字段名

类型

备注

int

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

主外键



PK

系统功能名称 功能页面或 url 系统功能状态 创建时间

更新时间

id parent\_id fc\_name fc\_url status create\_time update\_time

int

int varchar(20) varchar(50) char(1) datetime datetime

系统功能 id 父结点 id

字段名

类型

备注

表 3.3 商品类型表 product\_type

主外键



PK

商品类型 id 商品类型名称 创建时间

更新时间

id type\_name create\_time update\_time

int varchar(20) datetime datetime

字段名

类型

备注

表 3.4 商品信息表 product\_info

主外键



int

varchar(20)

varchar(20)

int

varchar(20)

varchar(255)

varchar(20)

int

varchar(255)

char(1)

商品信息 id 商品编号 商品名称 商品类别 id 商品品牌

商品图片 url

商品价格

商品数量

商品描述

商品状态

id pd\_code pd\_name tid brand image price num remark status

字段名

类型

备注

PK

FK

create\_time update\_time

创建时间 更新时间

datetime datetime

表 3.5 客户信息表 user\_info

主外键



int

varchar(20)

varchar(20)

varchar(20)

char(1)

varchar(255)

varchar(50)

char(1)

datetime

datetime

客户信息 id 用户登录账号 用户登录密码 真实姓名

性别

地址

邮箱

用户状态

创建时间

更新时间

id user\_name user\_pwd real\_name sex address email status create\_time update\_time

字段名

类型

备注

PK

表 3.6 订单信息表 order\_info

主外键



PK

FK

订单金额 订单状态 创建时间 更新时间

int

int decimal(8,2) char(1) datetime datetime

id

uid order\_price status create\_time update\_time

订单信息 id 客户 id

字段名

类型

备注

表 3.7 订单明细表 order\_detail

主外键



PK

FK

FK

create\_time

update\_time

订单明细 id 订单信息 id 商品信息 id 数量

创建时间

更新时间

int

int

int

int

datetime datetime

id oid pid num

字段名

类型

备注

表 3.8 管理员权限表 permission

主外键



PK

FK

FK

权限 id

管理员 id 功能菜单 id

id aid fid

int int int

字段名

类型

备注

2.数据库 ER 图

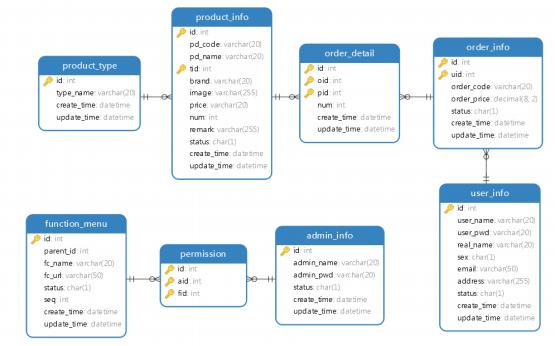


图 3.1 ER 图

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 上 Create from archetype，选择 maven-archetype-webapp，点击 Next，填写项目名 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-apo

⚫ 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



com.hsb.eshop.common

com.hsb.eshop.controller

com.hsb.eshop.dao

com.hsb.eshop.interceptor

com.hsb.eshop.pojo

com.hsb.eshop.service

mapper

static

views

用于存放公共实体类

用于存放控制类

用于存放数据库访问层接口

用于存放拦截器类

用于存放实体类

用于存放业务逻辑层接口和实现类 用于存放 Mybatis 映射文件

用于存放静态文件

用于存放视图文件

说明

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



applicationContext.xml

jdbc.properties

logback.xml

mybatis-config.xml

spring-servlet.xml

web.xml

pom.xml

Spring 框架的配置文件 存储数据库连接的属性文件 Logback 日志框架配置文件 Mybatis 配置文件 SpringMVC 框架配置文件 webapp 配置文件

Maven 配置文件

说明

4.3 系统目录结构

项目最终的目录结构如图 4.1 所示。

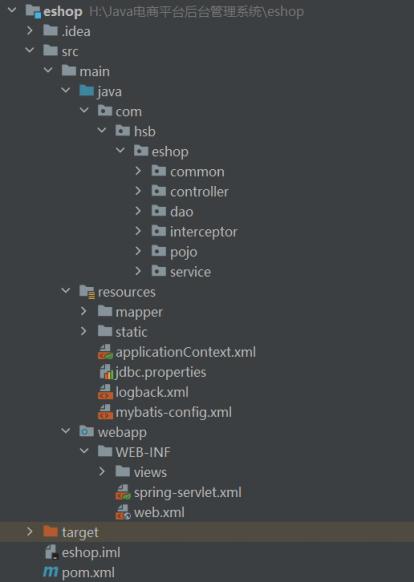


图 4.1 系统目录结构

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



添加商品接口

删除商品接口

查询商品接口

查询所有商品接口

按条件查询商品接口

修改商品接口

查询 table 数据接口

/productInfo/insert /productInfo/delete/{id} /productInfo/query/{id} /productInfo/queryAlls /productInfo/queryByCondition /productInfo/update /productInfo/queryAll

post 请求

根据 id 删除商品 根据 id 查询商品 查询所有商品

按条件查询商品 根据 id 修改商品 返回表格插件数据

接口说明

URL

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



添加商品类型接口

删除商品类型接口

查询商品类型接口

修改商品类型接口

/productType/insert /productType/delete/{id} /productType/queryAll /productType/update

post 请求

根据 id 删除商品类型 根据 id 查询商品类型 根据 id 修改商品类型

接口说明

URL

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



查询订单信息接口

删除订单信息接口

按条件查询订单接口

/orderInfo/queryAll /orderInfo/delete/{id} /orderInfo/query

查询所有订单信息 根据 id 删除订单信息 根据条件查询订单

接口说明

URL

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



查询订单信息接口

删除订单信息接口

按条件查询订单接口

保存订单信息接口

/orderInfo/queryAll /orderInfo/delete/{id} /orderInfo/query /orderInfo/saveInfo

查询所有订单信息 根据 id 删除订单信息 根据条件查询订单 保存订单信息

接口说明

URL

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



添加订单明细接口

删除订单明细接口

查询订单明细接口

/orderDetail/addDetail /orderDetail/delete/{id} /orderDetail/queryDetail/{id}

添加订单明细

根据 id 删除订单明细 根据 id 查询订单明细

接口说明

URL

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。

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

OrderDetailServiceImpl：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



查询所有客户接口

按名称查询客户接口

修改客户状态接口

/userInfo/queryAll /userInfo/query/{name} /userInfo/updateStatus/{id}/ {status}

查询所有客户信息 按名称查询客户信息 根据 id 修改客户状态

接口说明

URL

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



添加管理员接口

查询管理员接口

删除管理员接口

修改客户状态接口

修改管理员信息接口

/adminInfo/insert /adminInfo/queryAll /adminInfo/delete/{id} /adminInfo/updateStatus/{id} /{status}

/adminInfo/update/{id}

post 请求

查询所有管理员信息 根据 id 删除管理员 根据 id 修改管理员状 态

根据 id 修改管理员

接口说明

URL

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



添加菜单接口

删除菜单接口

修改菜单状态接口

/functionMenu/insert /functionMenu/delete/{id} /functionMenu/updateStatus/{ id}/{status}

添加菜单信息

根据 id 删除菜单

根据 id 修改菜单状态

接口说明

URL

/functionMenu/queryAll /functionMenu/update/{id}

查询所有菜单信息 根据 id 修改菜单信息

查询所有菜单接口 修改菜单信息接口

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。

9 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

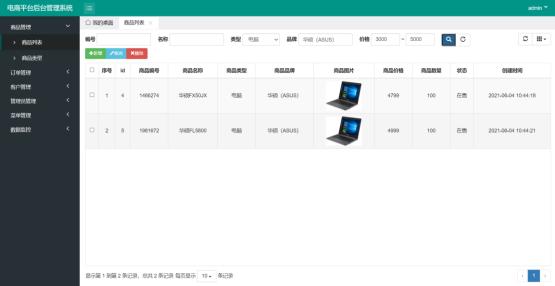


图 6.2.2 搜索商品

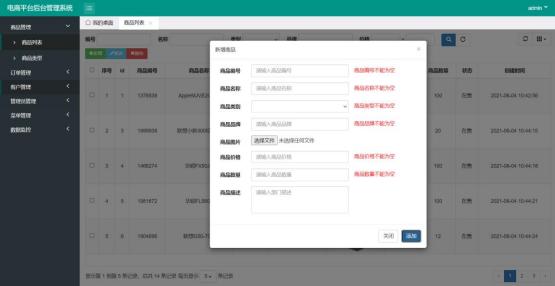


图 6.2.3 添加商品

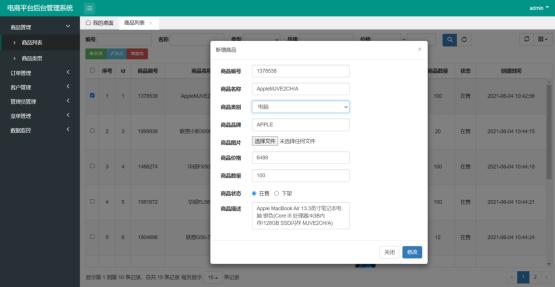


图 6.2.4 修改商品



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。。

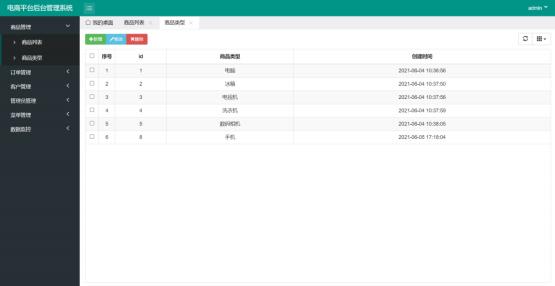


图 6.2.6 商品类型

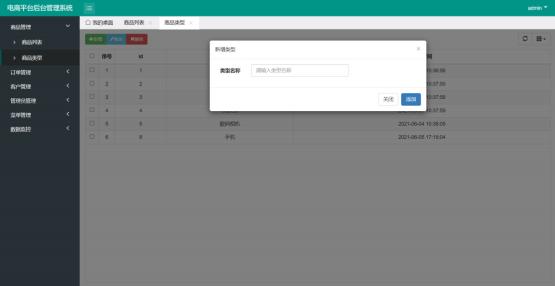


图 6.2.7 添加商品类型

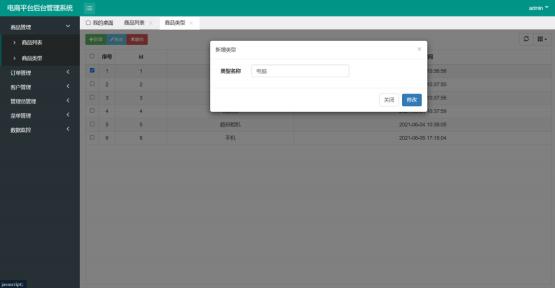


图 6.2.8 修改商品类型

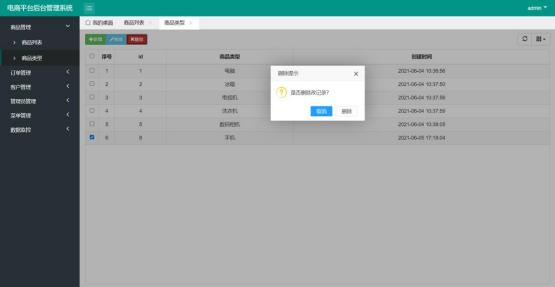


图 6.2.9 删除商品类型

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。

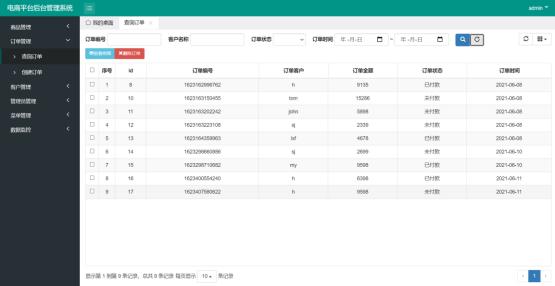


图 6.3.1 订单列表

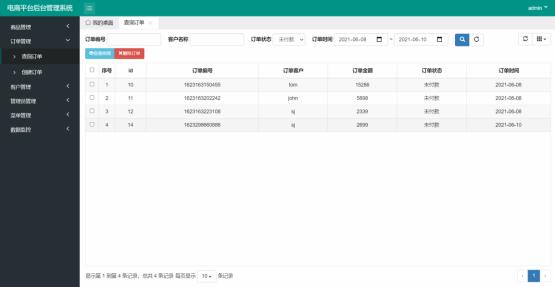


图 6.3.2 查找订单

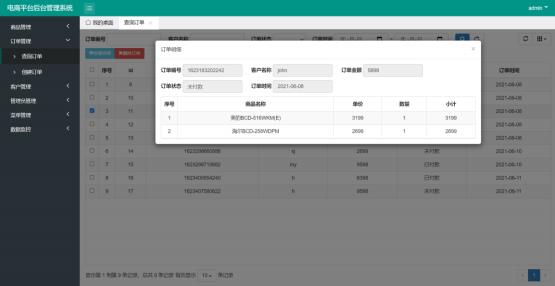


图 6.3.3 查看订单明细

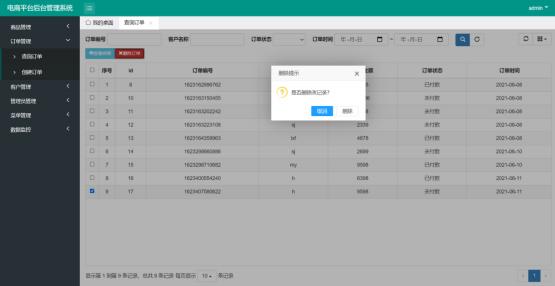


图 6.3.4 删除订单

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。

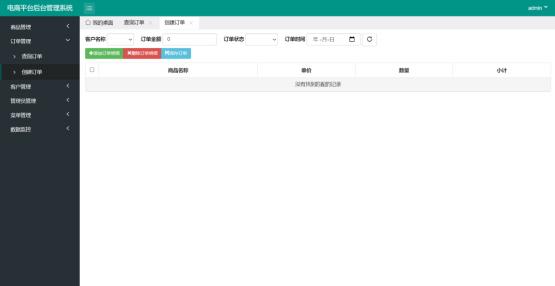


图 6.3.5 创建订单

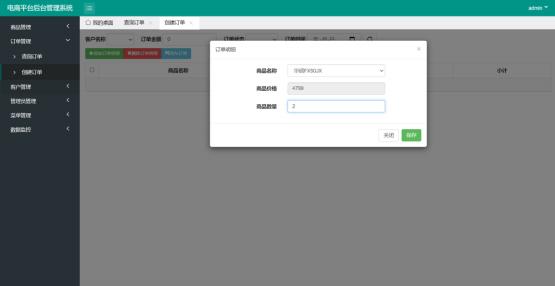


图 6.3.6 添加订单明细

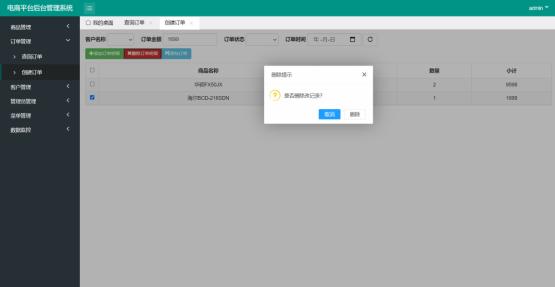


图 6.3.6 删除订单明细

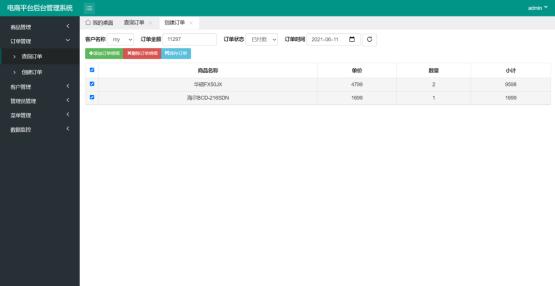


图 6.3.6 保存订单

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。

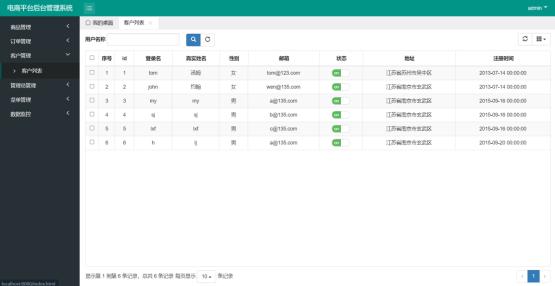


图 6.4. 1 客户列表



图 6.4.2 查询客户

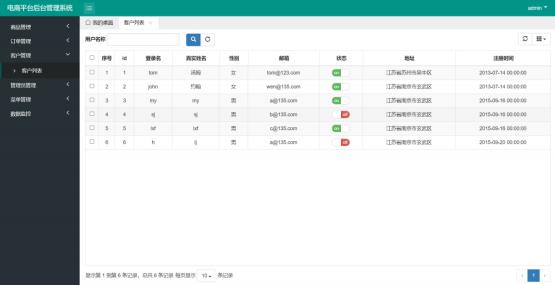


图 6.4.3 停用客户

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

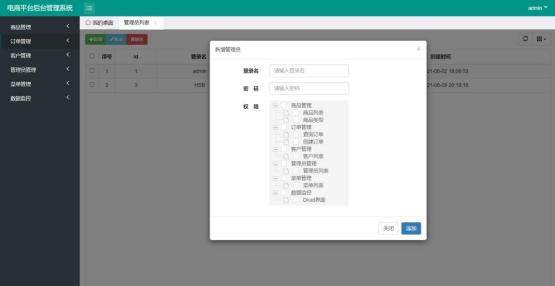


图 6.5.2 添加管理员

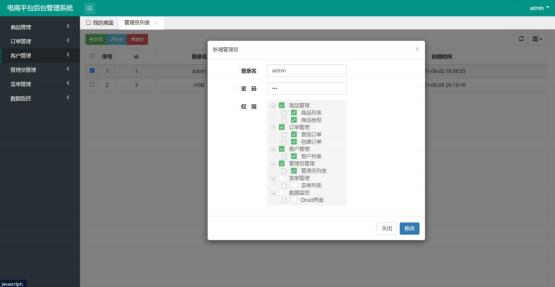


图 6.5.3 修改管理员

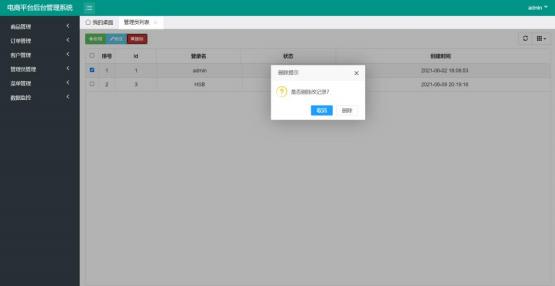


图 6.5.4 删除管理员

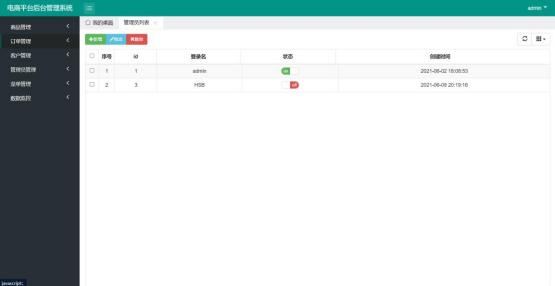


图 6.5.5 改变管理员状态

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。



图 6.6. 1 菜单列表

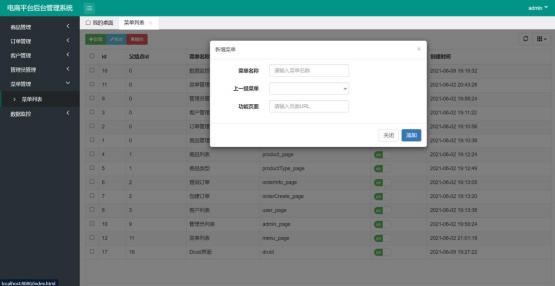


图 6.6.2 添加菜单

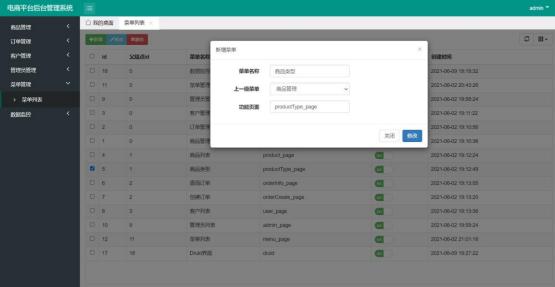


图 6.6.3 修改菜单

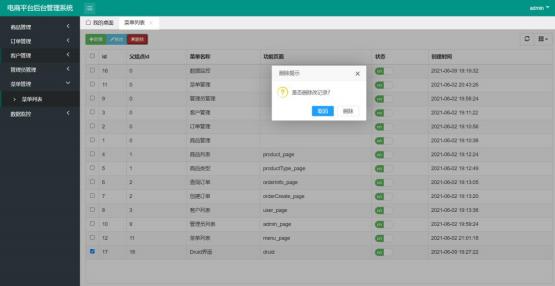


图 6.6.4 删除菜单

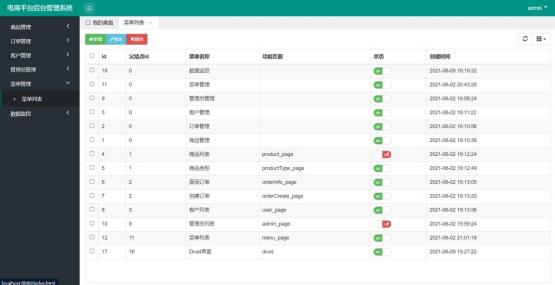


图 6.6.5 改变菜单状态

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。

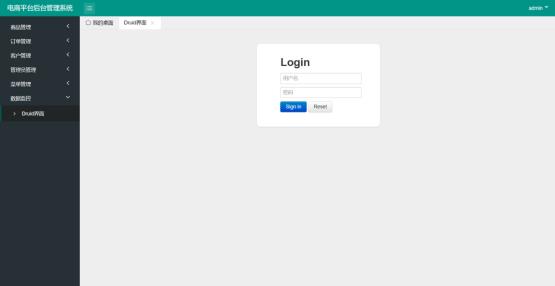


图 6.7. 1 Druid 登录界面

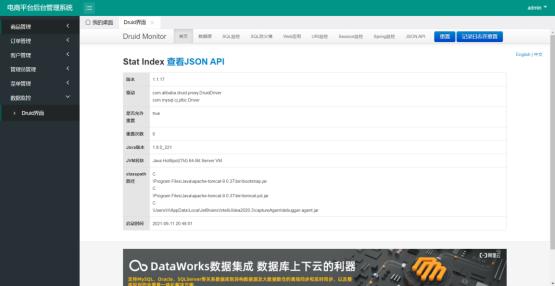


图 6.7.2 Druid 首页