**摘要** 现如今人们的生活水平的快速提高，越来越多的人为了陶冶情操，排解内心的孤单和寂寞，都会选择去养一只宠物，而养宠物则需要一些宠物日常用品。随着电商时代到来，网络购物已经步入了寻常百姓家，为了迎合时代的发展，满足人们足不出户就能购买商品，给用户的日常生活带来了更多的便利和更好的体验。它采用ThinkPHP框架及MVC开发模式作为主要实现方式，完成了用户登录，商品管理，权限管理等模块。系统界面友好，操作简单、易于维护扩展。

**关键词** 宠物用品；网上商城；ThinkPHP

**ABSTRACT** Nowadays, people's living standards are rapidly improving. More and more people choose to raise a pet in order to cultivate their sentiments and resolve their inner loneliness and loneliness. Pets need some pet daily necessities. With the advent of the e-commerce era, online shopping has entered the homes of ordinary people. In order to cater to the development of the times, people can buy goods without leaving home, bringing more convenience and better to the daily life of users. Experience. It adopts ThinkPHP framework and MVC development mode as the main implementation method, and completes user login, commodity management, and rights management modules. The system interface is friendly, easy to operate, easy to maintain and expand.

**KEY WORDS** Pet supplies; online store; ThinkPHP

**目 录**

[第1章 前 言 1](#_Toc3804156)

[1.1 课题的背景 1](#_Toc3804157)

[1.2 课题研究目的与意义 1](#_Toc3804158)

[1.3 本文的组织结构 2](#_Toc3804159)

[第2章 相关技术和开发运行环境 3](#_Toc3804160)

[2.1 PHP语言简介 3](#_Toc3804161)

[2.2 ThinkPHP框架简介 3](#_Toc3804162)

[2.3 MVC架构简介 3](#_Toc3804163)

[2.4 CSS样式表 4](#_Toc3804164)

[2.5 JavaScript脚本语言 5](#_Toc3804165)

[2.6系统运行环境 5](#_Toc3804166)

[第3章 系统需求分析和设计 6](#_Toc3804167)

[3.1 系统需求分析 6](#_Toc3804168)

[3.2 系统功能结构图 7](#_Toc3804169)

[3.3 购物流程图 9](#_Toc3804170)

[3.4数据库设计 11](#_Toc3804171)

[3.4.1 数据库分析 11](#_Toc3804172)

[3.4.2数据库E-R图 11](#_Toc3804173)

[3.4.3 数据库结构表 13](#_Toc3804174)

[3.5系统模块设计 16](#_Toc3804175)

[3.5.1 系统架构 16](#_Toc3804176)

[3.5.2 目录结构 16](#_Toc3804177)

[3.5.3 模块设计 17](#_Toc3804178)

[3.5.4 模型设计 17](#_Toc3804179)

[3.5.5 模板设计 17](#_Toc3804180)

[第4章 系统的实现 46](#_Toc3804181)

[4.1 系统前台功能模块的实现 46](#_Toc3804182)

[4.1.1 前台首页的布局 46](#_Toc3804183)

[4.1.2 登录管理 48](#_Toc3804184)

[4.1.3 注册管理 49](#_Toc3804185)

[4.1.4 分类展示 51](#_Toc3804186)

[4.1.5 搜索管理 53](#_Toc3804187)

[4.1.6 购物车管理 55](#_Toc3804188)

[4.2 系统后台功能模块的实现 57](#_Toc3804189)

[4.2.1管理员登录模块 57](#_Toc3804190)

[4.2.2后台首页的布局 58](#_Toc3804191)

[4.2.3权限管理模块 58](#_Toc3804192)

[4.2.4商品管理模块 60](#_Toc3804193)

[第5章 总结和展望 63](#_Toc3804194)

[5.1总结 63](#_Toc3804195)

[5.2 展望 63](#_Toc3804196)

[致 谢 64](#_Toc3804197)

[参考文献 65](#_Toc3804198)

# 前 言

## 课题的背景

随着社会不断的发展，人们的消费水平地提高，人们的物质需求已达到满足，为满足人们的精神需求，饲养宠物已逐渐成为人们的一大喜好。随着拥有宠物的消费者地增加，对于宠物日常用品的需求也随之增加。

而如今世界已处于互联网时代，互联网时代的来临催生了网络媒体、远程医疗、网络娱乐、互联网农业、电子商务、社交网络、网络教育、物联网以及互联网金融一系列新兴产业。在这一新兴事物的影响下，人们已经开始习惯足不出户在网上购买商品，传统店铺销售已不如以前。为了顺应潮流，PetMall宠物网上商城正是为此而开发。

## 1.2 课题研究目的与意义

我国的宠物行业属于朝阳产业，相关数据表明，当一个国家的人均 GDP 在 3000 至 8000 美元之间，宠物行业就会迅速发展。以目前的情况来看，我国的很多城市市区人均 GDP 已经超过了 3000美元，宠物行业已经发展到一定的规模，从事宠物行业的人超过 6万，宠物行业拥有的固定资产超过 20 亿元，一年创造的产值超过60 亿元。在北京登记注册的宠物狗数量超过 50 万只，上海则超过 70 万只。通过研究数据，可以发现国内的宠物市场以纯种狗和猫为主，经济涨幅超过两成。在宠物花费上，上海每年养狗的费用超过 6 亿元，每个家庭每个月的宠物花费大概在 300 元; 在青岛，养宠物的花费在几百块至几千块，而且这部分的花费仍然在持续增加。按照这样的速度发展下去，我国将拥有超过 1． 5 亿只的宠物，宠物行业的产值将超过 400 亿元。[1]

在宠物销售量持续增加的同时，与宠物相关的产品也得到了热销，例如宠物服装、宠物医疗等，销售宠物周边也成为做生意的门道。宠物食品包括饲料、鱼虾罐头、饼干、悠闲食品等。国产饲料的价钱比较便宜，一般的饲料每公斤 12 元，进口宠物食品的价钱是国产饲料的数倍。比起宠物养殖，销售宠物食品以及其他用品的形式更为灵活，在宠物经济中占据较大的比例。在国际市场，已经有企业将注意力放在生产和销售宠物服饰、香水、玩具等，甚至奢侈品行业的巨头也为宠物设计一系列的产品。[1]

因而，本课题的主要目标就是为方便消费者更好的购买宠物日常商品，促进传统实体店进行转型。

## 1.3 本文的组织结构

表1-1 本文的组织结构

|  |  |
| --- | --- |
| **章节** | **主 要 内 容** |
| 第一章 前言 | 本章简单介绍了时代已处于互联网时代，阐述了新时代背景下的宠物经济发展，并由此提出了项目的背景、目标。 |
| 第二章 相关技术和开发运行环境 | 本章主要阐述系统开发所用到的关键技术和开发运行环境。 |
| 第三章 PetMall网上商城系统的设计 | 本章分析了网站系统的需求，详细规划了网站的结构与功能，并进行了数据库的设计。 |
| 第四章 PetMall网上商城系统的实现 | 本章根据第三章的设计内容，给出网站系统的几个重要模块的实现过程。 |
| 第五章 总结与展望 | 本章是对所做工作的总结与展望，为后续的再开发奠定基础。 |

# 第2章 相关技术和开发运行环境

本章主要介绍了在项目开发过程中需要使用到的相关技术和开发工具，并且简单介绍了本网站系统的开发运行环境。

## 2.1 PHP语言简介

PHP是一种广泛使用的服务器端编程语言，用于开发动态网页。它是一种开源的、跨平台的、独立于架构的、解释的、面向对象的、快速的、简单易学的、健壮的 、安全的、性能优越的用于Web服务器端动态网页开发的语言。目前已有超过上千万的网站和数万家公司，以及数百万的程序开发人员在使用它。它是目前动态网页开发中使用最为广泛的语言之一。PHP语言以其优异的性能、强大的数据库操作功能迅速成为主流的Web服务器端开发语言[2]。

## 2.2 ThinkPHP框架简介

ThinkPHP是一款由国人开发基于MVC机构的轻巧型开发开源框架，提供了基础web开发功能，其内置了很多安全处理机制，开发人员可以使用ThinkPHP开发出强壮而又高效的PHP应用，大大简化了企业级应用开发。ThinkPHP的MVC模式吸收了所有主流框架的特点提供了层次分明、设计合理的MVC分层设计。ThinkPHP很好内置了许多安全处理机制，帮助发者在应用层方面尽量减少安全漏洞[3]。

## 2.3 MVC架构简介

MVC是一种先进的开发模式，能够解决团队开发之间协同配合的问题，使得网站各部件以更高的效率运行。MVC模式将网站分为3大部分，分别为模型、视图、控制器[3]。MVC的结构如图2-1所示。

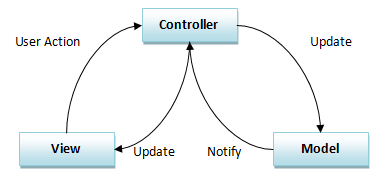


图2-1 MVC模型

模型（Model）： 用于封装与web应用与业务逻辑有关的数据以及对其的处理方法。模型有对数据库直接访问的权利，及模型不依赖视图和控制器，模型不关心自身会被如何显示。

视图（View）：对控制器转过来的数据进行有目的的显示。在这一层中一般没有业务逻辑。

控制器（Controller）：控制器起到组合模型和视图的组织作用，用于控制web应用的流程，对用户的事件或数据库反馈的数据进行响应。

## 2.4 CSS样式表

CSS(Cascading Style Sheet，简称CSS)中文译为层叠式表, 又称串样式列表、级联样式表、串接样式表、阶层式样式表，一种用来为结构化文档（HTML页面或XML应用）添加样式，并允许将样式信息与文档分离的一种计算机语言，由W3C定义和维护。CSS的引入就是为了使得HTML语言能够更好地适应页面的美工设计。它以HTML用语言为基础，提供了丰富的格式化功能，如字体、颜色、背景和整体排版等，并且页面设计者可以针对各种可实话浏览器设置不同的样式风格。

CSS样式一般是作用在DIV上的，可以为每个DIV区域定义独特的CSS，这样我们就可以将网页中指定的DIV部分变成我们所需要的风格、样式，而网页又是由多个DIV构成，从而达到梅花页面的作用[4]。

## 2.5 JavaScript脚本语言

JavaScrip是面向Web的编程语言。绝大多数现代网站都使用了JavaScript,并且所以的现代网站都使用了JavaScript，并且所以的现代Web浏览器一一基于桌面系统、游戏机、平板电脑和智能手机的浏览器一一均包含了JavaScript解释器。这使得JavaScript能够称得上史上使用最广泛的编程语言。JavaScript也是前端开发工程师必须掌握的三种技能之一：描述网页内容的HTML、描述网页样式的CSS以及描述网页行为的JavaScript.[5]。

## 2.6系统运行环境

WampServer是Windows Apache Mysql PHP集成安装环境，即Windows下的Apache+Mysql/MariaDB+Perl/PHP/Python，一组常用来搭建动态网站或者服务器的[开源软件](https://baike.baidu.com/item/%E5%BC%80%E6%BA%90%E8%BD%AF%E4%BB%B6/8105369" \t "_blank)，本身都是各自独立的程序，但是因为常被放在一起使用，拥有了越来越高的兼容度，可以让开发者一次安装并设置好开发环境。本课题系统的运行将采直接采用wampserver。其版本为WampServer 3.1.4，包含版本：Apache 2.4.4 ，Mysql 5.6.12， PHP 5.4.16。

# 第3章 系统需求分析和设计

## 3.1 系统需求分析

关于本系统需求，包括包含以下几个方面：角色，界面，功能，安全与保密性等。

本系统的用户角色：消费者，管理员。

界面：美观，大气，友好，用户体验好。

功能：本系统功能分为前台和后台管理。

前台功能主要功能有以下几个：

1. 浏览商品
2. 注册，登陆功能
3. 商品加入购物车
4. 下订单
5. 用户个人中心

后台网站管理主要功能有以下一些以下主要功能：

1. 用户管理
2. 会员列表
3. 修改密码
4. 禁用用户
5. 商品分类管理
6. 商品分类列表
7. 添加商品分类
8. 商品内容管理
9. 商品列表
10. 添加商品
11. 商品模型
12. 编辑商品信息
13. 商品模型
14. 订单管理
15. 订单列表
16. 修改订单状态

针对该公司的需求，本文3.2节对系统进行了分模块的设计说明。

## 3.2 系统功能结构图

为了能更好的说明网站的结构，下面给出本电子商务网站的前台功能模块结构图和后台功能模块结构图。

PetMall网上商城系统前台包含一个电子商务网站所应具有的一些基本功能，其设计能满足用户正常浏览商品，加入购物车，购物商品，同时用户可以在用户中心中查看订单和修改个人信息等进行一些基本操作。PetMall网上商城系统前台管理模块的功能设计，前台功能模块结构图，如图3-1所示。

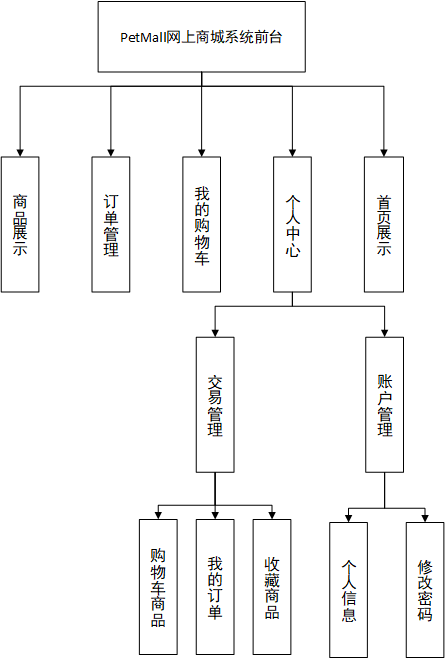


图3-1 前台功能模块结构图

与系统前台功能相对应，PetMall网上商城系统的后台包含用户管理，商品管理，订单管理等一些基本功能。管理员可以删除用户，添加新商品，回复用户留言等进行些基本的操作。PetMall网上商城系统后台管理模块的功能设计，如图3-2所示。

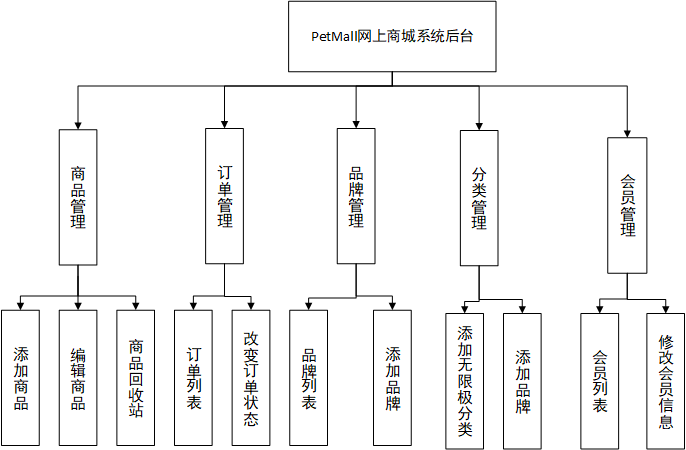


图3-2 后台功能模块结构图

## 3.3 购物流程图

所谓电子商务，实质就是建立一个虚拟的购物超市，首先客户应该在网页中选购自己需要的商品并将商品放入购物车中，选购好商品后就可以到收银台填写订单信息，提交订单和付款等操作。PetMall网上商城系统的购物流程图，图3-3所示。

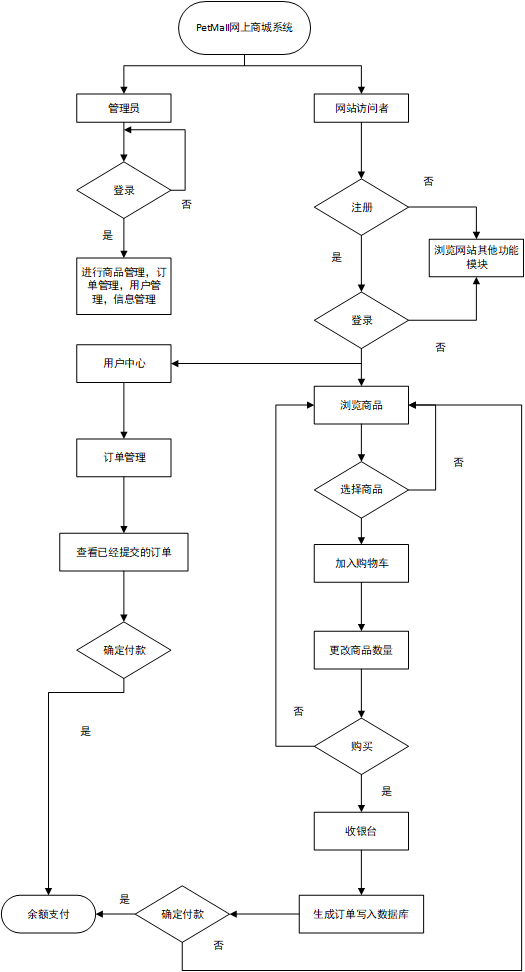


图3-3 购物流程图

## 3.4数据库设计

## 3.4.1 数据库分析

因为本项目需要充分考虑到成本及用户需求等问题，MySQL 是最流行的关系型数据库管理系统，而且成本低，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)应用软件之一。所以本系统采用MySQL作为数据库。

## 3.4.2数据库E-R图

根据以上对系统的需求分析，规划出本购物系统的实体关系E-R图。表示方法为：

实体：用矩形表示，矩形框内标注实体名

属性：用椭圆表示，用横线将其与实体连接起来。

联系：用棱形表示（1:n ,1:1，m:n）。

1、用户信息实体

用户信息实体包括用户ID(Uid),用户昵称（user\_alias）,用户邮箱(user\_email),用户密码（user\_pwd）,注册时间（reg\_time）,用户手机（user\_phone）。

用户信息实体的E-R图,如图3-4所示。

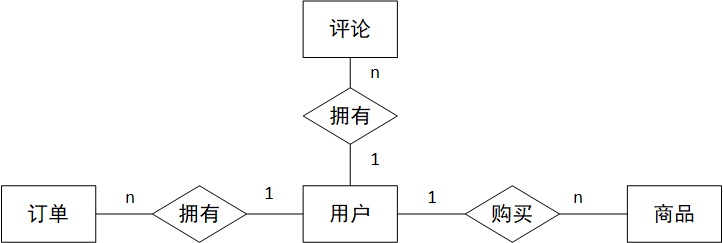


图3-4 用户信息实体E-R图

2、管理员信息实体

管理员信息实体包括管理员ID(id),管理员名（admin\_name）,密码(pwd)，权限（admin\_level）。管理员信息实体的E-R图,如图3-5所示。

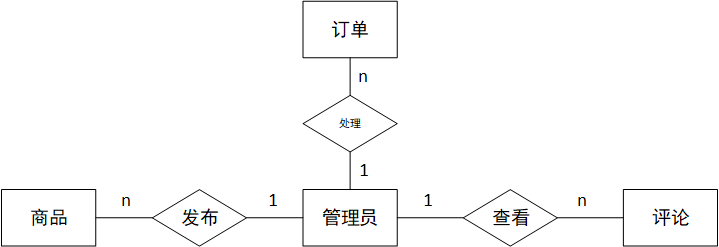


图3-5 管理员信息实体E-R图

3、商品信息实体

商品信息实体包括编号(id),商品名称(goods\_name),商品所属分类(type\_id),商品存货(goods\_number),商品品牌(brand\_id),商品现价(goods\_price),上架状态（is\_delete）,图片链接地址(goods\_img),商品描述（goods\_desc）等。商品信息实体E-R图如图3-6所示。

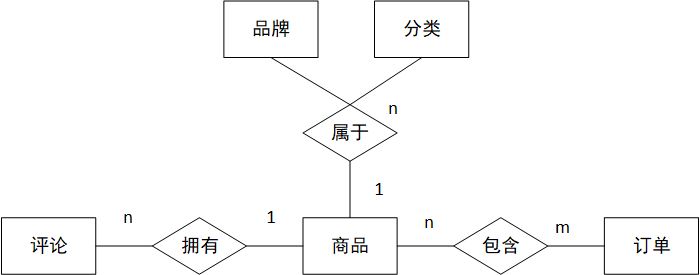


图3-6 商品信息实体E-R图

4、商品订单实体

商品订单实体包括订单ID(id),订单号(acti),下单时间(create\_time),下单人(user\_Uid),订单状态(action\_status),商品名称(goods\_id),订单地址（user\_address）。商品订单实体E-R图,如图3-7所示。



图3-7 商品订单实体E-R图

## 3.4.3 数据库结构表

根据在数据库设计中给出的数据库实体E-R图，设计数据表结构。数据库名称为petmall，包含20张数据表，表前缀为pet\_ 。具体表格清单如表3-1所示。

表3-1 数据库中的数据表

|  |  |  |
| --- | --- | --- |
| **表名** | **备注** | **用途** |
| pet\_user\_address | 地址信息表 | 存储用户地址的详细信息 |
| pet\_goods\_collect | 收藏商品表 | 存储用户收藏的商品信息 |
| pet\_comment | 用户评论表 | 存储用户的评论信息 |
| pet\_order | 订单信息表 | 存储用户订单信息 |
| pet\_admin | 管理员信息表 | 存储管理员信息 |
| pet\_auth\_group | 管理员角色表 | 存储管理员角色拥有哪些权限 |
| pet\_auth\_group\_access | 管理员归属组表 | 存储管理员位于哪个用户组 |
| pet\_auth\_rule | 权限表 | 存储管理员各项权限 |
| pet\_brand | 商品品牌表 | 存储商品品牌信息 |
| pet\_cart | 用户购物车表 | 存储用户购物车信息 |
| pet\_good | 商品表 | 存储商品的详细信息 |
| pet\_goods\_type | 商品模型表 | 存储商品模型的信息 |
| pet\_good\_images | 商品相册表 | 存储商品相册 |
| pet\_good\_type | 商品分类表 | 存储商品分的信息 |
| pet\_order\_good | 订单商品表 | 存储订单里的商品 |
| pet\_spec | 商品各项规格类型表 | 存储商品规格类型信息 |
| pet\_spec\_goods\_price | 商品与规格映射表 | 存储商品与规格映射关系 |
| pet\_spec\_item | 商品规格项表 | 存储商品各项规格项 |
| pet\_user | 用户表 | 存储用户 |
| pet\_user\_details | 用户详细信息表 | 存储用户详细信息 |

以下是列出几个重要的数据表的详细设置：

商品分类表（pet\_good\_type）字段详细设置，如表3-2所示。

表3-2 pet\_good\_type数据表结构

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段** | **类型** | **属性** | **约束条件** | **说明** |
| id | int | 无符号、非空、自增 | 主键 | 分类id |
| type\_name | char(30) | varchar(45) |  | 分类名字 |
| pid | varchar(45) | 非空 |  | 父id |
| parent\_id\_path | varchar(45) | 非空 |  | 分类族谱 |
| type\_order | tinyint(4) | 非空 |  | 分类排序 |
| is\_show | tinyint(4) | 非空 |  | 是否显示 |

订单表（pet\_order）字段详细设置，如表3-3所示。

表3-3 pet\_order数据表结构

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段** | **类型** | **属性** | **约束条件** | **说明** |
| id | int | 无符号、非空、自增 |  | 订单id |
| acti | varchar(45) | 非空 |  | 订单交易号 |
| user\_Uid | int | 非空 |  | 用户id |
| admin\_id | varchar(45) | 非空 |  | 操作管理员id |
| address\_id | int(11) | 非空 |  | 用户地址id |
| action\_status | tinyint(4) | 非空 |  | 订单状态 |
| pay\_status | tinyint(4) | 非空 |  | 付款状态 |
| shipping\_status | tinyint(4) | 非空 |  | 配送状态 |
| order\_count | decimal(10,2) | 非空 |  | 订单金额 |
| action\_time | varchar(45) | 非空 |  | 操作时间 |
| order\_message | varchar(150) | 非空 |  | 买家留言 |
| express\_number | varchar(45) | 非空 |  | 快递单号 |
| create\_time | varchar(45) | 非空 |  | 订单创建时间 |
| comeback\_reason | varchar(45) | 非空 |  | 退货原因 |
| is\_delete | varchar(45) | 非空 |  | 订单是否删除 |
| user\_address | varchar(245) | 非空 |  | 用户地址 |

商品规格项表（pet\_spec\_item）字段详细设置，如表3-4所示。

表3-4 pet\_spec\_item数据表结构

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段** | **类型** | **属性** | **约束条件** | **说明** |
| id | int | 无符号、非空、自增 | 主键 | 规格项id |
| spec\_id | int | 非空 |  | 规格类型id |
| item | varchar(54) | 非空 |  | 规格项（如SMXL） |

管理员角色表（pet\_auth\_group）字段详细设置，如表3-5所示。

表3-5 pet\_auth\_group数据表结构

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段** | **类型** | **属性** | **约束条件** | **说明** |
| id | int | 无符号、非空、自增 | 主键 | 角色id |
| title | varchar(45) | 非空 |  | 角色名 |
| status | tinyint | 非空 |  | 角色状态 |
| rules | text | 非空 |  | 角色权限 |

## 3.5系统模块设计

## 3.5.1 系统架构

PetMall网上商城采用ThinkPHP5.0框架作为系统架构。ThinkPHP框架的设计采用MVC设计模式，Controller对应MVC控制器，Model对应MVC模型，View对应MVC视图。相对应地，规划设计控制器、模型和模板成为代码开发阶段的主要工作[5]。

## 3.5.2 目录结构

系统采用单一入口模式进行项目部署和访问，一个统一的入口完成需求的所有功能。目录结构如下表所示：

表3-6 目录结构说明

|  |  |
| --- | --- |
| **文件/目录** | **说明** |
| Application | 应用目录 |
| ThinkPHP | 框架目录 |
| public | 资源文件目录，存放图片、CSS、JS和其它静态文件 |
| README.md | README文件 |
| public/index.php | 入口文件 |

## 3.5.3 模块设计

针对ThinkPHP的模块而言，就是控制器的设计。一个完整的ThinkPHP应用基于模块/控制器/操作设计，并且，如果有需要的话，可以支持多入口文件和多级控制器。ThinkPHP新版采用模块化的架构设计思想，对目录结构规范做了调整，可以支持多模块应用的创建，让应用的扩展更加方便。所以，实现系统功能，就是设计模块和编写方法。

## 3.5.4 模型设计

每个模型类对应一张数据表，主要用来实现以下两点功能。

* 自动验证：自动验证是ThinkPHP模型层提供的一种数据验证方法，可以在使用create创建数据对象的时候自动进行数据验证。
* 自动完成：自动完成是ThinkPHP提供用来完成数据自动处理和过滤的方法，使用create方法创建数据对象的时候会自动完成数据处理。

## 3.5.5 模板设计

模板作为渲染前的初级样式，由静态页面转换而成。模板设计的内容包括以下几项。

* 为对应的模块方法创建相应的模板文件。
* 将html代码替换成可行的模板标签或php代码。
* 对公共部分进行模板布局，即抽去所以页面的头部和尾部形成独立文件。

# 第4章 系统的实现

## 4.1 系统前台功能模块的实现

## 4.1.1 前台首页的布局

商城首页采用了css+div的布局。商城首页，如图4-1所示。

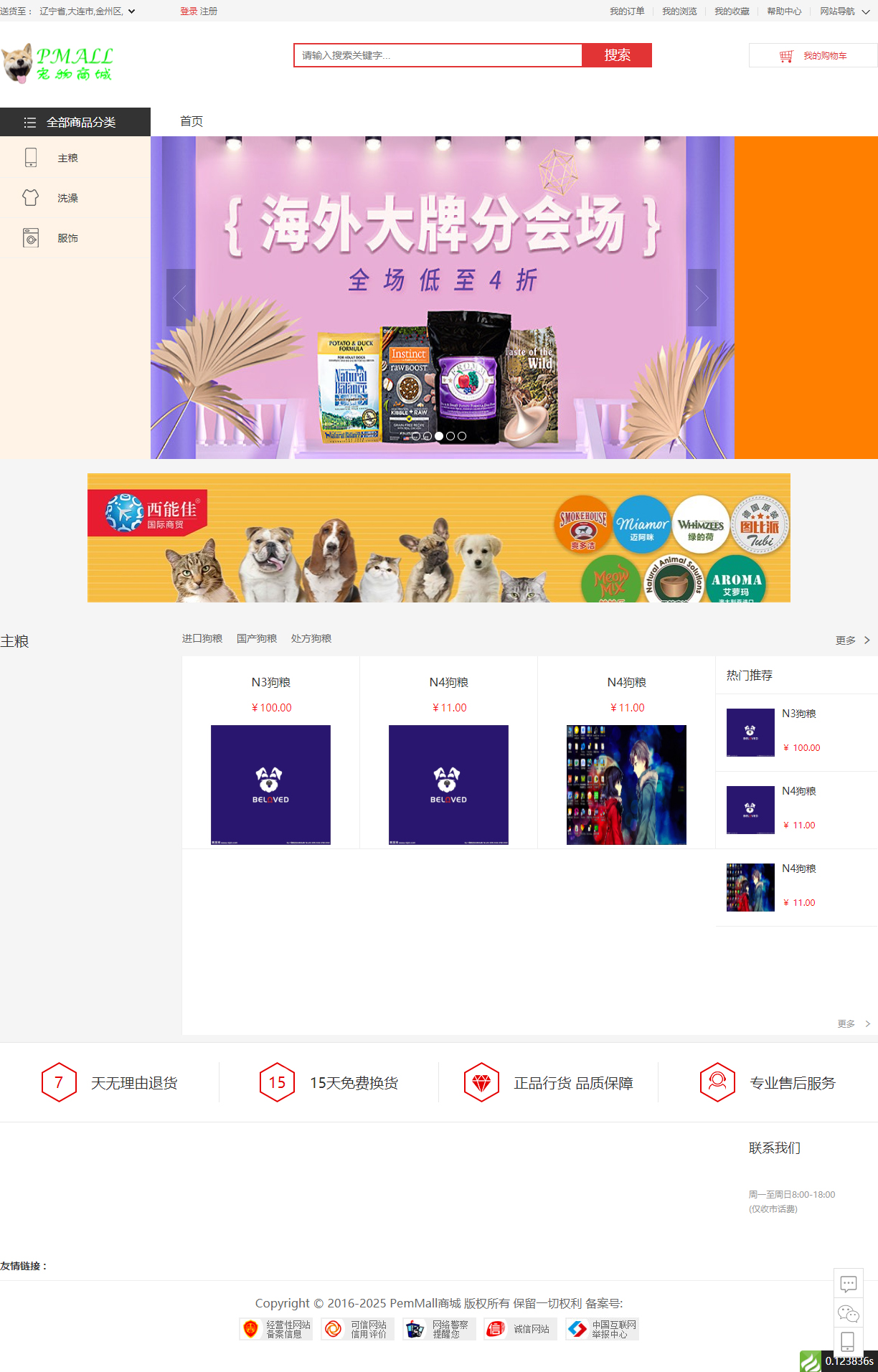


图4-1 前台首页示意图

## 4.1.2 登录管理

用户登录管理模块的实现主要使用两个文件，一个是填写登录信息的视图文件：Application/index /View/User/login.html,另一个是接收登录信息写入数据库的实现文件：Application/index/Controller/User.php。用户登录页如图4-2所示。



图4-2 登录页面

登录功能的文字描述：

一、检查验证码：

当验证码的输入框失去焦点时发生blue事件，利用jq检查是否填写验证码，如果没有填写，则提示“验证码不能为空”。如果已经填写则利用ajax请求将验证码发给user控制器checkCaptcha方法进行验证，如果错误则进行提示“验证码不正确！”。

二、回车触发提交

1、按回车键，触发登录按钮的click事件，利用html5自带的属性，把邮箱input标签的type属性设置为"email"，但触发登录按钮时自动验证邮箱格式是否正确。

2、用post方法提交用户名和密码到对应控制器的checklogin方法

四、ajax验证是否存在该用户

用ajax方法提交用户名到对应控制器的login方法

五、login方法：

查询用户：通过input()助手函数来获取前台提交过来的用户名和密码，在控制器使用Db类查询数据库，验证用户名和密码是否正确，如果用户名和密码正确而且用户没被禁用，就创建session，然后跳转到首页。

其对应的User控制器实现脚本的核心代码如下：

1. **public** **function** login() {
2. $redirst = input("redirst") ? input("redirst") : url("index/index/index");
3. **if** (request()->isPost()) {
4. $data = request()->param();
5. $map['user\_email'] = strtolower($data['email']);
6. $map['user\_pwd'] = md5(md5($data['password']));
7. $rst = Db::name('user')
8. ->alias('a')
9. ->join('user\_details b', 'a.Uid = b.user\_Uid')
10. ->where($map)
11. ->find();
12. **if** ($rst['is\_delete'] == 1) {
13. $this->error('该用户已被禁用');
14. } **elseif** ($rst['is\_delete'] == 0) {
15. **if** (isset($data['holdStatus']) && $data['holdStatus'] == 1) {
16. //使用cookie实现自动登录
17. setcookie("user\_email", $map['user\_email'], time() + 3600, '/');
18. setcookie("user\_pwd", md5($data['password']), time() + 3600, '/');
20. }
21. session\_start();
22. $\_SESSION['alias'] = $rst['user\_alias'];
23. $\_SESSION['user'] = $rst['user\_email'];
24. $\_SESSION['uid'] = $rst['Uid'];
25. $this->success('登录成功', $data['redirst']);
27. } **else** {
28. $this->error('账号或密码错误');
29. }
30. }
31. $this->assign('redirst', $redirst);
32. **return** $this->fetch();
33. }

## 4.1.3 注册管理

用户注册管理模块的实现主要使用两个文件，一个是填写注册信息的视图文件：Application/Index/View/User/register.html,另一个是接收注册信息的实现文件：Application/Index/Controller/User.php。注册页面，如图4-3所示。



图4-3 注册页面

注册功能的文字描述：

一、验证输入登录名的格式是否等于邮箱

二、验证输入的邮箱是否已被注册

当邮箱的输入框失去焦点时发生blue事件

通过ajax发送邮箱值到后台checkEmail方法进行检查是否已被注册。

三、利用jq检查两次密码输入是否一致。

四、验证输入的验证码

利用ajax发送到后台checkCaptcha方法检查验证码

五、回车触发提交

1、按回车键，触发登录按钮的click事件

2、用post方法提交用户名和密码到对应控制器的register方法

六、register方法：

通过input助手函数来获取前台提交过来的数据， insert方法写入数据库

其对应的User控制器实现脚本的核心代码如下：

1. //用户进行注册
2. **public** **function** register() {
3. $userData['user\_email'] = strtolower(input('email'));
4. $userData['user\_pwd'] = md5(md5(input('post.pwd')));
5. $userData['is\_delete'] = 0;
6. $userData['reg\_time'] = date('Y-m-d H:i:s', time());
8. $rst = Db::name('user')->where('user\_email', '=', $userData['user\_email'])->find();
9. //判断是否重复注册
10. **if** ($rst) {
11. $this->error('该用户已经注册过了');
12. }
14. $status = false;
16. Db::startTrans();
17. try {
18. Db::name('user')->insert($userData);
19. $userId = Db::name('user')->getLastInsID();
20. Db::name('user\_details')->data(['user\_Uid' => $userId, 'user\_alias' => '用户' . time() . rand(0, 10000)])->insert();
21. Db::commit();
22. $status = true;
24. } catch (\Exception $e) {
26. Db::rollback();
28. }
30. **if** ($status) {
31. $this->success('注册成功', url('index/user/login'));
32. } **else** {
33. $this->error('注册失败', url('index/user/reg'));
34. }
36. }

## 4.1.4 分类展示

分类展示模块的实现主要使用两个文件，一个是显示分类展示的视图文件：Application/Home/View/List/index.html,另一个是实现分类展示的文件：Application/Home/Controller/ListController.class.php。分类展示页面，如图4-4所示。



图4-4 分类展示页面

分类展示功能的文字描述：

实例化Cate模型，通过input助手函数来获取前台提交的数据，select方法来查询数据，assign方法来赋值变量，fetch方法来输出变量

其对应的Good控制器实现脚本的核心代码如下：

1. /\*\*
2. \* 商品列表页
3. \*/
4. **public** **function** goodsList() {
6. $filter\_param = **array**(); // 帅选数组
7. $id = input('get.id/d', 1); // 当前分类id
8. $brand\_id = input('get.brand\_id', 0);
9. $spec = input('get.spec', 0); // 规格
10. $attr = input('get.attr', ''); // 属性
11. $sort = input('get.sort', 'sort'); // 排序
12. $sort\_asc = input('get.sort\_asc', 'desc'); // 排序
13. $price = input('get.price', ''); // 价钱
14. $start\_price = trim(input('post.start\_price', '0')); // 输入框价钱
15. $end\_price = trim(input('post.end\_price', '0')); // 输入框价钱
16. **if** ($start\_price && $end\_price) {
17. $price = $start\_price . '-' . $end\_price;
18. }
19. // 分类菜单显示
20. $goodsCate = model('GoodsCategory')->where("id", $id)->find(); // 当前分类
21. //($goodsCate['level'] == 1) && header('Location:'.U('Home/Channel/index',array('cat\_id'=>$id))); //一级分类跳转至大分类馆
22. $cateArr = $goodsLogic->get\_goods\_cate($goodsCate);
23. ……

## 4.1.5 搜索管理

搜索模块的实现主要使用两个文件，一个是显示搜索管理的视图文件：Application/Index/View/good/search.html,另一个是实现分类展示的文件：Application/Index/Controller/Good.php。搜索页面，如图4-5所示。

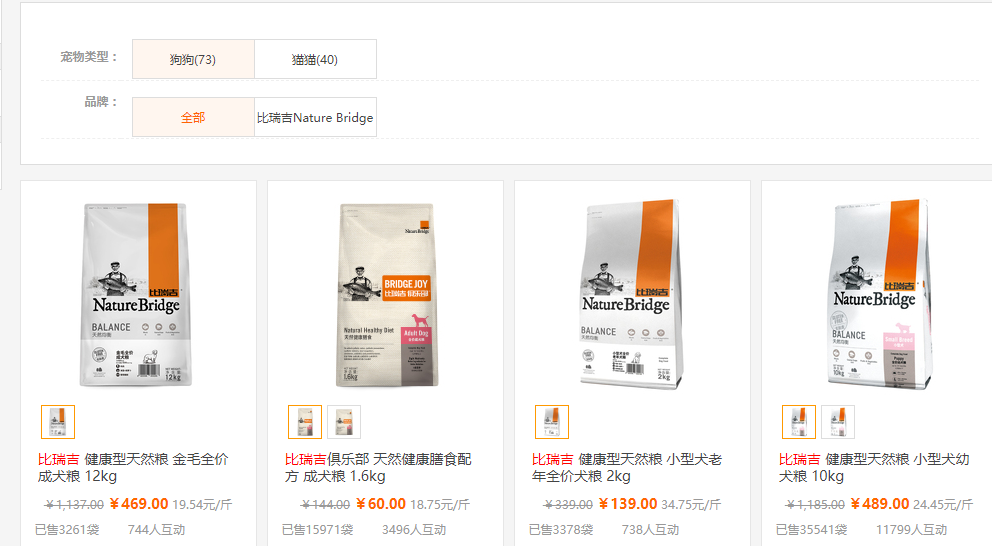


图4-5 搜索页面

搜索功能的文字描述：

1、接收前台首页搜索框传过来的关键词

2、查询商品数据表

3、以传过来的关键词为条件查出相关数据总条数赋值给数组$goodsHaveSearchWord

4、实例化分页类，为搜索到的数据创建分页

5、以限制条件查询每页应显示的查询结果

6、把post传过来的关键词分配到前台搜索页面模板

7、向搜索页面的分页按钮分配数据

8、向搜索页面分配搜索结果的显示数据

其对应的SearchController控制器实现脚本的核心代码如下：

1. /\*\*
2. \* 商品搜索列表页
3. \*/
4. **public** **function** search() {
5. $filter\_param = **array**(); // 帅选数组
6. $id = I('get.id/d', 0); // 当前分类id
7. $brand\_id = I('brand\_id', 0);
8. $sort = I('sort', 'sort'); // 排序
9. $sort\_asc = I('sort\_asc', 'desc'); // 排序
10. $price = I('price', ''); // 价钱
11. $start\_price = trim(I('start\_price', '0')); // 输入框价钱
12. $end\_price = trim(I('end\_price', '0')); // 输入框价钱
13. **if** ($start\_price && $end\_price) {
14. $price = $start\_price . '-' . $end\_price;
15. }
16. // 如果输入框有价钱 则使用输入框的价钱
17. $q = urldecode(trim(I('q', ''))); // 关键字搜索
18. empty**empty**($q) && $this->error('请输入搜索词');
19. $id && ($filter\_param['id'] = $id); //加入帅选条件中
20. $brand\_id && ($filter\_param['brand\_id'] = $brand\_id); //加入帅选条件中
21. $price && ($filter\_param['price'] = $price); //加入帅选条件中
22. $q && ($\_GET['q'] = $filter\_param['q'] = $q); //加入帅选条件中
23. $goodsLogic = **new** GoodsLogic(); // 前台商品操作逻辑类
24. $SearchWordLogic = **new** SearchWordLogic();
25. $where = $SearchWordLogic->getSearchWordWhere($q);
26. $where['is\_on\_sale'] = 1;
27. Db::name('search\_word')->where('keywords', $q)->setInc('search\_num');
28. $goodsHaveSearchWord = Db::name('goods')->where($where)->count();
29. **if** ($goodsHaveSearchWord) {
30. $SearchWordIsHave = Db::name('search\_word')->where('keywords', $q)->find();
31. ……
32. }

## 4.1.6 购物车管理

购物车管理模块的实现主要使用两个文件，一个是购物车管理的视图文件：Application/Index/View/cart/index.html,另一个是实现购物车管理的文件：Application/Index/Controller/Cart.php。购物车页面，如图4-6所示。



图4-6 购物车页面

购物车功能的文字描述：

ajaxAddCart方法（添加购物车）：

1.在商品页面点击“加入购物车”按钮。

2.利用ajax发送商品的ID,购买数量，规格信息发送到ajaxAddCart方法。

3.在后台判断购买数量是否大于库存，如果没有则取用户SESSION的uid，然后将购物车的商品存入数据库。

其对应的Cart控制器实现脚本的核心代码如下：

1. /\*\*
2. \* ajax 将商品加入购物车
3. \*/
4. **function** ajaxAddCart() {
5. $data["good\_id"] = input("goods\_id/d"); // 商品id
6. $data["num"] = input("goods\_num/d"); // 商品数量
7. $data["item\_id"] = input("item\_id/d") ? input("item\_id/d") : 0; // 商品规格id
8. $data["user\_Uid"] = $\_SESSION['uid'];
9. $num["good\_id"] = input("goods\_id/d"); // 商品id
10. $num["item\_id"] = input("item\_id/d") ? input("item\_id/d") : 0; // 商品规格id
11. $num["user\_Uid"] = $\_SESSION['uid'];
12. **if** (empty**empty**($data["good\_id"])) {
13. $this->ajaxReturn(['status' => 0, 'msg' => '请选择要购买的商品', 'result' => '']);
14. }
15. **if** (empty**empty**($data["num"])) {
16. $this->ajaxReturn(['status' => 0, 'msg' => '购买商品数量不能为0', 'result' => '']);
17. }
18. **if** ($data["num"] > 200) {
19. $this->ajaxReturn(['status' => 0, 'msg' => '购买商品数量大于200', 'result' => '']);
20. }
21. /\* 获得该商品的库存 \*/
22. **if** ($data["item\_id"] != 0) {
23. $store\_count = Db::name("spec\_goods\_price")->where("item\_id", $data["item\_id"])->value('store\_count');
24. } **else** {
25. $store\_count = Db::name("good")->where("id", $data["good\_id"])->value('goods\_number');
26. }
27. $map = Db::name("cart")->where($num)->value('num');
28. try {
29. **if** (!$map) {
30. **if** ($data['num'] > $store\_count) {
31. $this->ajaxReturn(['status' => 0, 'msg' => '已大于库存，加入购物车失败']);
32. }
33. $data['create\_time'] = date('Y-m-d H:i:s', time());
34. // var\_dump($data['create\_time']);die;
35. $rst = Db::name("cart")->data($data)->insert();
37. } **else** {
38. **if** (($data['num'] + $map) > $store\_count) {
39. $this->ajaxReturn(['status' => 0, 'msg' => '已大于库存，加入购物车失败']);
40. }
41. $rst = Db::name("cart")->where($num)->setField('num', $data['num'] + $map);
42. }
43. **if** ($rst) {
44. $this->ajaxReturn(['status' => 1, 'msg' => '加入购物车成功']);
45. } **else** {
46. $this->ajaxReturn(['status' => 0, 'msg' => '加入购物车失败']);
47. }
48. } catch (TpshopException $t) {
49. $error = $t->getErrorArr();
50. $this->ajaxReturn($error);
51. }
52. }

## 4.2 系统后台功能模块的实现

系统后台的功能模块有，注册用户与权限管理模块、栏目分类管理模块、商品内容管理模块、服务管理模块和其他参数管理模块。其界面和功能与用户管理模块和商品管理模块大同小异，相关的核心代码也类似，只是对数据库表操作的sql语句略有不同。下面给出，管理员登陆模块，用户管理模块，商品管理模块的主要功能模块的实现过程。

## 4.2.1管理员登录模块

管理员登陆界面的实现主要对应三个php文件，一个是登录界面的视图文件：Application/Admin/view/Login/index.html,一个是对输入信息进行验证和生成验证码文件：Application/Admin/Controller/LoginController.class.php

管理员登录模块的文字描述:

checklogin方法：

查询用户：通过I来获取前台提交过来的数据，实例化User模型，查询用户名，如果存在用户名，检查密码是否正确；是否是工作人员；

如果正确，就创建session

其对应的LoginController控制器实现生成验证码的核心代码如下：

function code($i=1) {

$verify = new \Think\Verify();

$verify->fontSize = 30;

$verify->length = 3;

$verify->useNoise = false;

$verify->useCurve = false;

$verify -> entry($i);

}

## 4.2.2后台首页的布局

后台首页采用了css+div的布局。

(后台首页页面:Application/Admin/view/index/index.html)。后台首页，如图4-7所示。

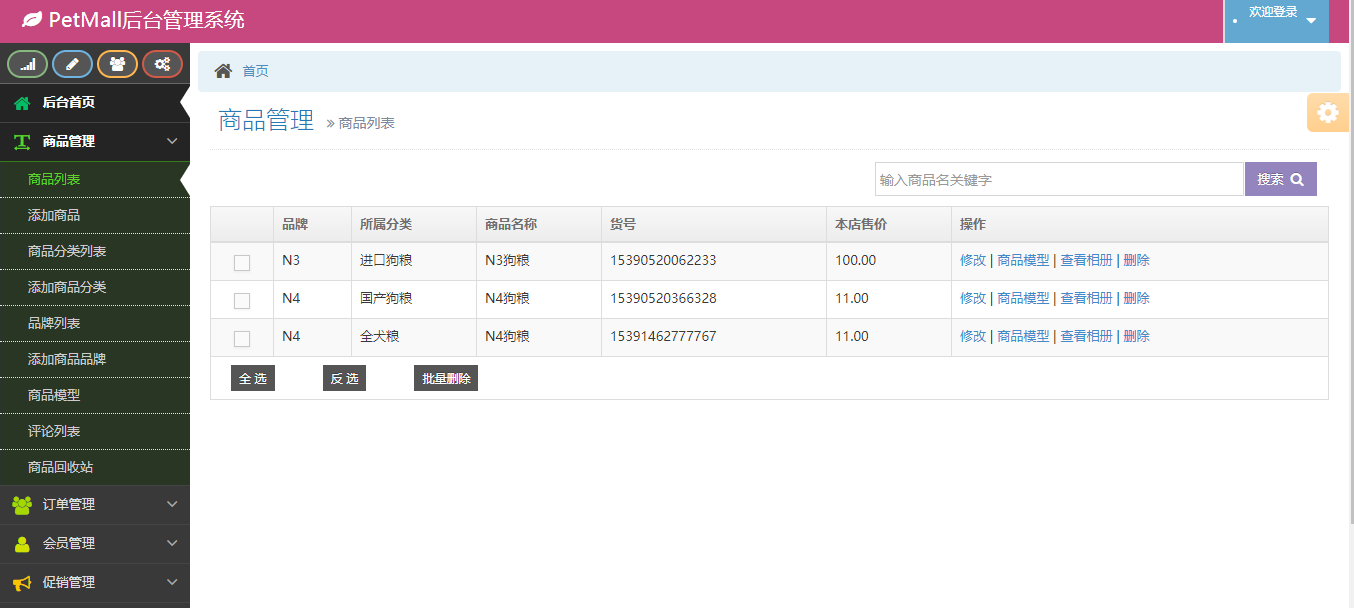


图4-7 后台首页

## 4.2.3权限管理模块

权限管理模块的实现主要使用两个文件，一个是权限管理的视图文件：Application/Admin/View/Role/index.html,另一个是实现权限管理模块的文件：Application/Admin/Controller/RoleController.class.php。权限管理页面，如图4-8所示。

图4-8 权限管理页面

权限管理功能的文字描述：

通过I来获取前台提交过了的数据，实例化Role模型，add方法插入数据

其对应的RoleController控制器实现脚本的核心代码如下：

public function add\_do(){//添加分组

// $group =I('post.group');

$node = I('post.node');

$b = I('post.status');

$a = I('post.title');

$db = M('role');

$data['status']=$b;

$data['name']=$a;

$r = $db->add($data);

if($r){

//转换二维数组添加

$arr =array();

foreach ($node as $v) {

$arr[]=array('role\_id'=>$r,'node\_id'=>$v);

}

$ar= M('access')->addAll($arr);

if($ar){

$this->success('添加管理组成功',U('index'),1);

}

}else{

$this->error('添加管理组失败',U('add'),1);

}

}

## 4.2.4商品管理模块

商品管理模块的实现主要使用两个文件，一个是商品管理的视图文件：application/admin/view/good/index.html,另一个是实现商品管理模块的文件：application/admin/controller/Good.php。商品管理页面，如图4-9所示。

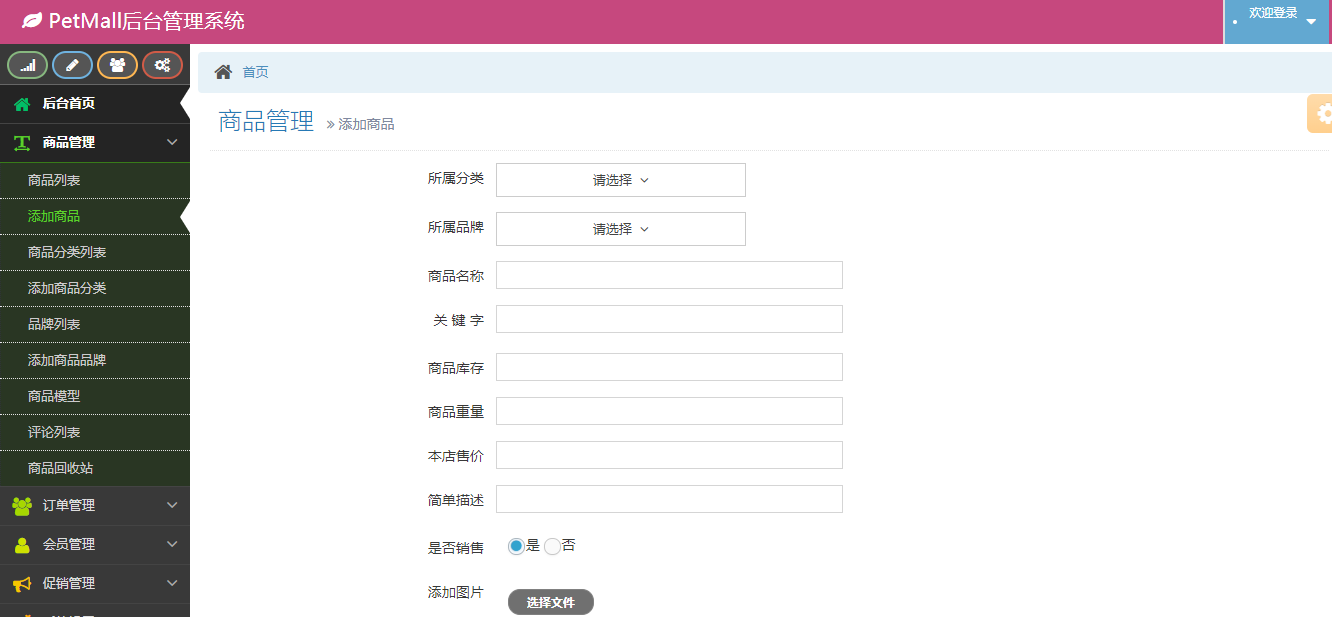


图4-9 商品管理页面

商品管理功能的文字描述：

添加商品的操作方法

1、填写商品的各项信息

2、上传图片时利用jq组件Huploadify，发送到后台myUpload方法，该方法负责保存图片，并返回该图片在服务器的路径。

3、各项参数填写好后，点击“提交”按钮。

4、通过post方法，将各项数据发送到addEditGood方法，将数据存入数据库。

其对应的Good控制器实现脚本的核心代码如下：

1. //添加或修改商品
2. **public** **function** addEditGood() {
4. **if** (request()->isPost()) {
5. $GoodModel = **new** GoodModel();
6. $GoodModel->data(request()->param());
7. **if** (request()->param('id')) {
8. $count = $GoodModel->isUpdate(true)->allowField(true)->save();
9. $info = '修改成功';
10. } **else** {
11. $data = request()->param();
12. $data['sn'] = time() . rand(1000, 10000);
13. $data['goods\_time'] = time();
14. $map['id'] = $data['type\_id'];
15. $data['good\_model'] = substr(Db::name('good\_type')->where($map)->value('parent\_id\_path'), 2, 1);
16. $count = $GoodModel->isUpdate(false)->allowField(true)->save($data);
18. $info = '添加成功';
19. }
20. **if** ($count) {
21. $this->success($info, 'index');
22. } **else** {
23. exit;
24. }
25. }
26. $data = model('good\_type')->getCategory();
27. $brand = Db::name('brand')->select();
28. $this->assign('brand', $brand);
29. $this->assign('type', $data);
30. **if** (request()->param('id')) {
31. $good\_data = Db::view('good')
32. ->view('good\_type', 'type\_name', 'good.type\_id = good\_type.id')
33. ->view('brand', 'brand\_name', 'brand.id=good.brand\_id')
34. ->where('good.id', request()->param('id'))
35. ->find();
37. $this->assign('iddata', $good\_data);
38. **return** $this->fetch("edit");
39. exit;
40. }
41. **return** $this->fetch('add');
42. }

# 第5章 总结和展望

## 5.1总结

论文分析了计算机网络的快速发展以及电子商务网站的现状，介绍了本项目的背景和研究价值，根据广州某公司实际的业务需求，提出了手工艺品网站的建设方案。 本文概述了手工艺品网上商城网站的主要任务和技术背景，分析了系统功能模块，并阐述了数据库设计的过程。然后，详细介绍了系统的具体设计与实现。最后，通过测试调试，系统基本能够满足需求分析和详细设计要求。

系统的基本功能已经实现，但是还是存在着许多的不足之处，需要对它进行进一步的完善和改进：

1. 平面设计水平不高，以至于界面比较呆板，不能让消费者有眼前一亮的感觉。
2. 数据库设计有很多冗余的地方，需进一步优化。
3. 对于网站的安全性与稳定性，需进一步优化。

## 5.2 展望

本系统还可以继续添加许多新鲜和使用的功能，但是本人的技术原因，一些功能暂时还不能实现，希望自己在将来可以不断地学习，解决掉着一些技术难题，争取把这个系统的不足完善上去。把它建设成一个强大的电子商务网站。

在本系统的设计和开发中涉及很多新的方法和技术，由于时间的关系，所以该系统还有许多不尽如人意的地方，这些都有待在以后的研究中进一步完善。

致 谢

经过两个月的努力，本系统已经开发成功。在本系统开发过程中，我要特别感谢我的指导老师覃忠台老师和我的同事孜孜不倦的指导和帮助。无论是在论文的选题、构思，还是在论文的研究方法以及成文定稿方面，我都得到了覃忠台老师悉心的教诲和无私的帮助。还有在我网站的设计和开发过程中遇到的很多技术问题都是在同事热情的指导和帮助下得以解决的。在此表示真诚的感谢和深深的谢意。

同时我还感谢母校的老师,谢谢他们为本系统提供的帮助.让我能够顺利地完成本系统的开发.

最后，向在百忙中抽出时间对本文进行评审并提出宝贵意见的各位专家表示衷心的感谢！

参考文献

1. 陈思含.浅谈新时代背景下的宠物经济发展[J].知识经济,2019(01):79+81.
2. 卫喆，陈争航.PHP5与MySQL5从入门到精通[M].北京:电子工业出版社，2008-8-1
3. 李开涌.[PHP MVC 开发实战](http://www.shangxueba.com/book/1790593.html)[M].北京：机械工业出版社,2013-07-01
4. 车开森林.浅谈CSS＋DIV在网页制作中的应用[J].计算机光盘软件与应用, 2012（2）.23-24
5. David Flanagan著；淘宝前端团队译.JavaScript权威指南（第6版）[M]. 北京：机械工业出版社，2011-04-01