**北京理工大学**

**继续教育暨现代远程教育学院**

毕业设计（论文）

**毕业论文题目： 基于Python语言Django框架的商品管理系统**

**指导教师姓名：** **吴素研**

**类　别： 夜大．专升本**

**专 业： 计算机科学与技术**

**班 级： 2016年计算机科学与技术A班（012）**

**姓 名： 何慧君**

**2017年 5月20 日**

# 摘要

本文是以学习和研究网页设计与Web系统开发为目标，在课堂学习的基础上，综合运用HTML、CSS、JS、B/S系统等知识，探索利用目前流行的Python语言，开发出可用的Web系统。经过学习发现，可以使用Django框架，来快速开发Web系统。

本论文将会简要阐述Pyhton的部署，Django安装、配置，以及商品管理系统主要功能的实现，以及遇到的各种问题。

关键词：Python，Django，商品管理

**目录**

[摘要 II](#_Toc483718007)

[前言 1](#_Toc483718008)

[1 绪论 2](#_Toc483718009)

[2 Django 3](#_Toc483718010)

[2.1 Django特点 3](#_Toc483718011)

[2.2 Django工作方式 3](#_Toc483718012)

[2.3 开发环境 4](#_Toc483718013)

[2.4 安装Django 4](#_Toc483718014)

[2.5 Bootstrap安装 5](#_Toc483718015)

[3 主要功能 6](#_Toc483718016)

[3.1 功能设计 6](#_Toc483718017)

[3.2 主要界面 7](#_Toc483718018)

[3.2.1 首页 7](#_Toc483718019)

[3.2.2 所有商品（分页） 8](#_Toc483718020)

[3.2.3 商品详情 8](#_Toc483718021)

[3.2.4 管理员登录页 9](#_Toc483718022)

[3.2.5 商品管理页 9](#_Toc483718023)

[3.2.6 商品添加页面 10](#_Toc483718024)

[4 代码实现 11](#_Toc483718025)

[4.1 创建项目 11](#_Toc483718026)

[4.2 设置数据库 11](#_Toc483718027)

[4.3 设置国际化 12](#_Toc483718028)

[4.4 配置静态文件 12](#_Toc483718029)

[4.5 创建Django app 12](#_Toc483718030)

[4.6 创建models 13](#_Toc483718031)

[4.7 运行程序 14](#_Toc483718032)

[4.8 同步数据库 15](#_Toc483718033)

[4.9 设置Admin 15](#_Toc483718034)

[4.10 创建超级用户 16](#_Toc483718035)

[4.11 编写template 16](#_Toc483718036)

[4.12 编写首页和商品展示页（模板） 16](#_Toc483718037)

[4.13 完善Views 16](#_Toc483718038)

[4.14 实现分页（模板） 17](#_Toc483718039)

[4.15 完善url 18](#_Toc483718040)

[5 结论 19](#_Toc483718041)

[致谢 20](#_Toc483718042)

[参考文献 21](#_Toc483718043)

[源代码 22](#_Toc483718044)

[1. commodity/apps.py 22](#_Toc483718045)

[2. commodity/models.py 22](#_Toc483718046)

[3. commodity/views.py 23](#_Toc483718047)

[4. commodity/admin.py 25](#_Toc483718048)

[5. commodity/templates/base.html 26](#_Toc483718049)

[6. commodity/templates/index.html 29](#_Toc483718050)

[7. commodity/templates/list.html 31](#_Toc483718051)

[8. commodity/templates/post.html 34](#_Toc483718052)

[9. WebStore/urls.py 36](#_Toc483718053)

[10. WebStore/settings.py 37](#_Toc483718054)

# 前言

应用程序有两种模式C/S、B/S。C/S是客户端/服务器端程序，也就是说这类程序一般独立运行。而B/S就是浏览器端/服务器端应用程序，这类应用程序一般借助Chrome等浏览器来运行。WEB应用程序一般是B/S模式。Web应用程序首先是“应用程序”，和用标准的程序语言，如C、C++等编写出来的程序没有什么本质上的不同。然而Web应用程序又有自己独特的地方，就是它是基于Web的，而不是采用传统方法运行的。换句话说，它是典型的浏览器/服务器架构的产物。B/S结构能够很好地应用在广域网上，成为越来越多的企业的选择。

一个Web应用程序是由完成特定任务的各种Web组件（web components）构成的并通过Web将服务展示给外界。在实际应用中，Web应用程序是由多个模型(model)－视图(view)－控制器(controller)等组织起来的代码组成，这些组件相互协调为用户提供一组完整的服务。

Web应用程序对企业的重要用途是对数据进行处理，管理信息系统（Management Information System,简称MIS）就是这种架构最典型的应用。MIS可以应用于局域网，也可以应用于广域网。基于Internet的MIS系统以其成本低廉、维护简便、覆盖范围广、功能易实现等诸多特性，得到越来越多的应用。

本文就是尝试利用Python语言实现商品管理系统，增强对Web应用系统需求、设计、开发的知识和能力。

# 绪论

Python 已经有将近 30 年的历史，在过去 30 年中，Python 在运维工程师和数据科学家群体中受到广泛欢迎，然而却极少有企业将 Python 作为生产环境的首选语言。在最近几年，这一情况有所改变。随着云计算、大数据以及人工智能技术的快速发展，Python 及其开发生态环境正在受到越来越多的关注。

互联网时代来临后，Python被用来在Web开发领域进行尝试，涌现出了一批基于 Python 开发一些WEB的网站，还有不少大型的、基于Python 的网站，比如 Youtube、豆瓣等网站。使得一些Web开源框架迅速成长，如Django、Flask等，为程序员高效开发Web程序提供了巨大的帮助。

进入了云计算时代，基于过去一段时间 Python 在系统管理工具的积累，以及其本身具备了非常好的系统集成能力，Python 在云计算领域可以说大放异彩。最著名的是Python开发的Openstack。不仅在私有云领域，在公有云领域，包括 AWS，包括 Google 云，当这些公有云提供出 SDK 的时候，它们首选的技术路线依然是 Python。

最近两年又火起来的人工智能领域，Python靠着过去多年在科学计算等方面的积累出现了大爆发。比如图像识别用的都是 Python OpenCV库。在深度学习领域几乎没有任何其他语言可以跟 Python 相提并论的，比如 Caffe，Theano，TesnorFlow，Keras 这些非常流行的深度学习框架，都是以 Python 为主要开发语言。事实证明了在深度学习领域目前 Python 是处于非常主导的地位。

如上所述，为了能跟上人工智能的潮流，从用户体验角度，从开发者角度来讲，Python 是更好的语言，也是更好的接口语言，值得我们学习和掌握它。另一方面，考虑到可以用 Python 集成各种各样的服务，这样能有效降低成本，同时也能够减轻自己开发团队的压力，让开发团队能够减少一些学习成本。

# Django

[Django](http://www.jianshu.com/p/d15188a74104)是Python中目前风靡的Web Framework,框架能够帮助我们把程序的整体架构搭建好, 而我们所需要做的工作就是填写逻辑, 而框架能够在合适的时候调用你写的逻辑, 而不需要我们自己去调用逻辑, 让Web开发变的更敏捷.

Django是一个高级Python Web框架, 鼓励快速,简洁, 以程序设计的思想进行开发. 通过使用这个框架, 可以减少很多开发麻烦, 使你更专注于编写自己的app, 而不需要重复造轮子. Django免费并且开源。

## Django特点

1. 完全免费并开源源代码。
2. 快速高效开发。
3. 使用MTV架构(熟悉Web开发的应该会说是MVC架构)。
4. 强大的可扩展性。

## Django工作方式

用户在浏览器中输入URL后的回车, 浏览器会对URL进行检查, 首先判断协议,如果是http就按照 Web 来处理, 然互调用DNS查询, 将域名转换为IP地址, 然后经过网络传输到达对应Web服务器, 服务器对url进行解析后, 调用View中的逻辑(MTV中的V), 其中又涉及到Model(MTV中的M), 与数据库的进行交互, 将数据发到Template(MTV中的T)进行渲染, 然后发送到浏览器中, 浏览器以合适的方式呈现给用户。

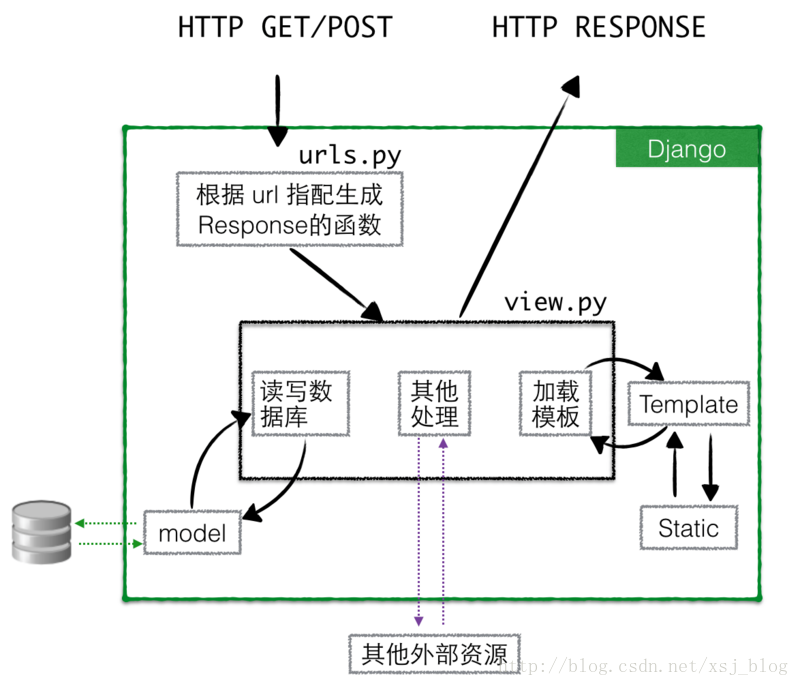


图2-1工作方式

## 开发环境

本文使用的主要开发环境为：

1. 操作系统：macOS Sierra 10.12.4
2. 开发语言：Python 3.6.1
3. Web框架：Django1.11.1
4. 数据库：SQLite3
5. 前端框架：Bootstrap3.3.7
6. 开发IDE：Pycharm CE 2017.1
7. 虚拟环境：Anaconda 4.3.17
8. 版本管理：GitHub

## 安装Django

使用终端安装最新版的Django：

conda install Django

## Bootstrap安装

Bootstrap，来自 Twitter，是目前最受欢迎的前端框架。Bootstrap 是基于 HTML、CSS、JAVASCRIPT 的，它简洁灵活，使得 Web 开发更加快捷。

从官网下载所需资源，稍后复制到开发目录中：

http://getbootstrap.com/getting-started/#download

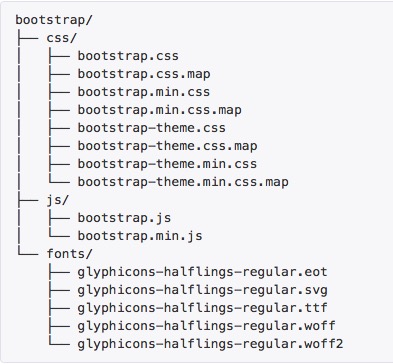
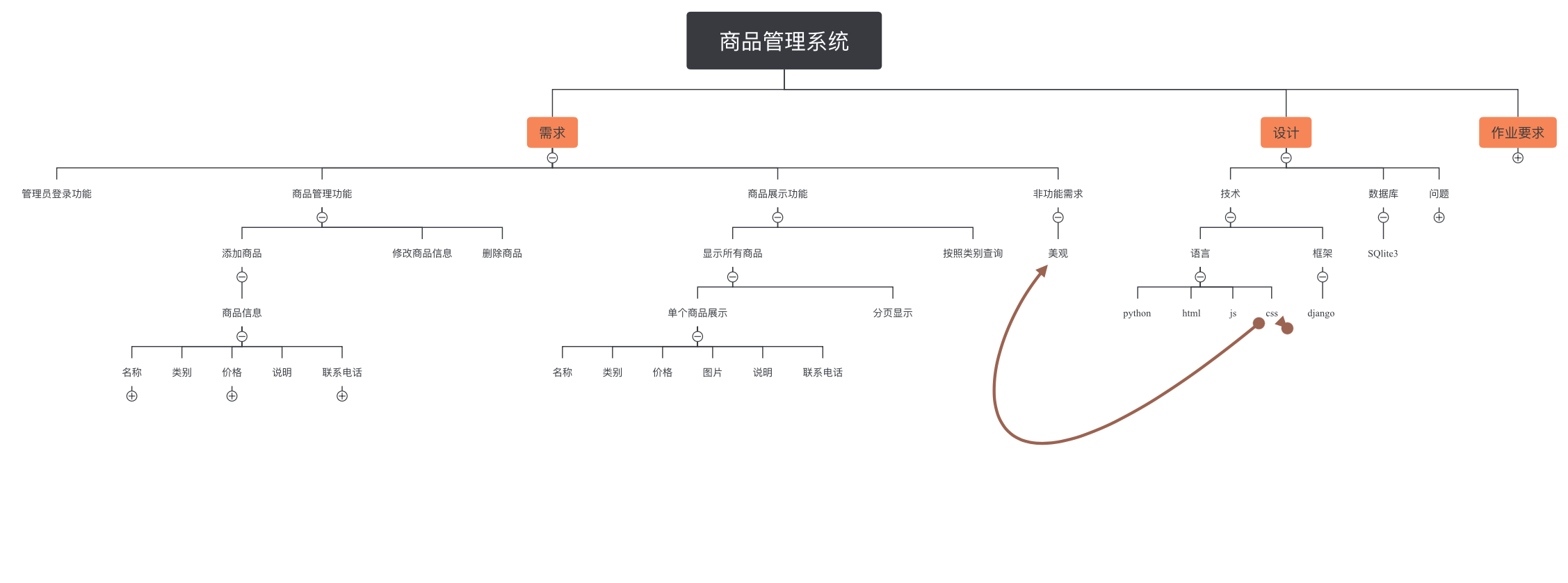


图2-2 BootStrap目录

# 主要功能

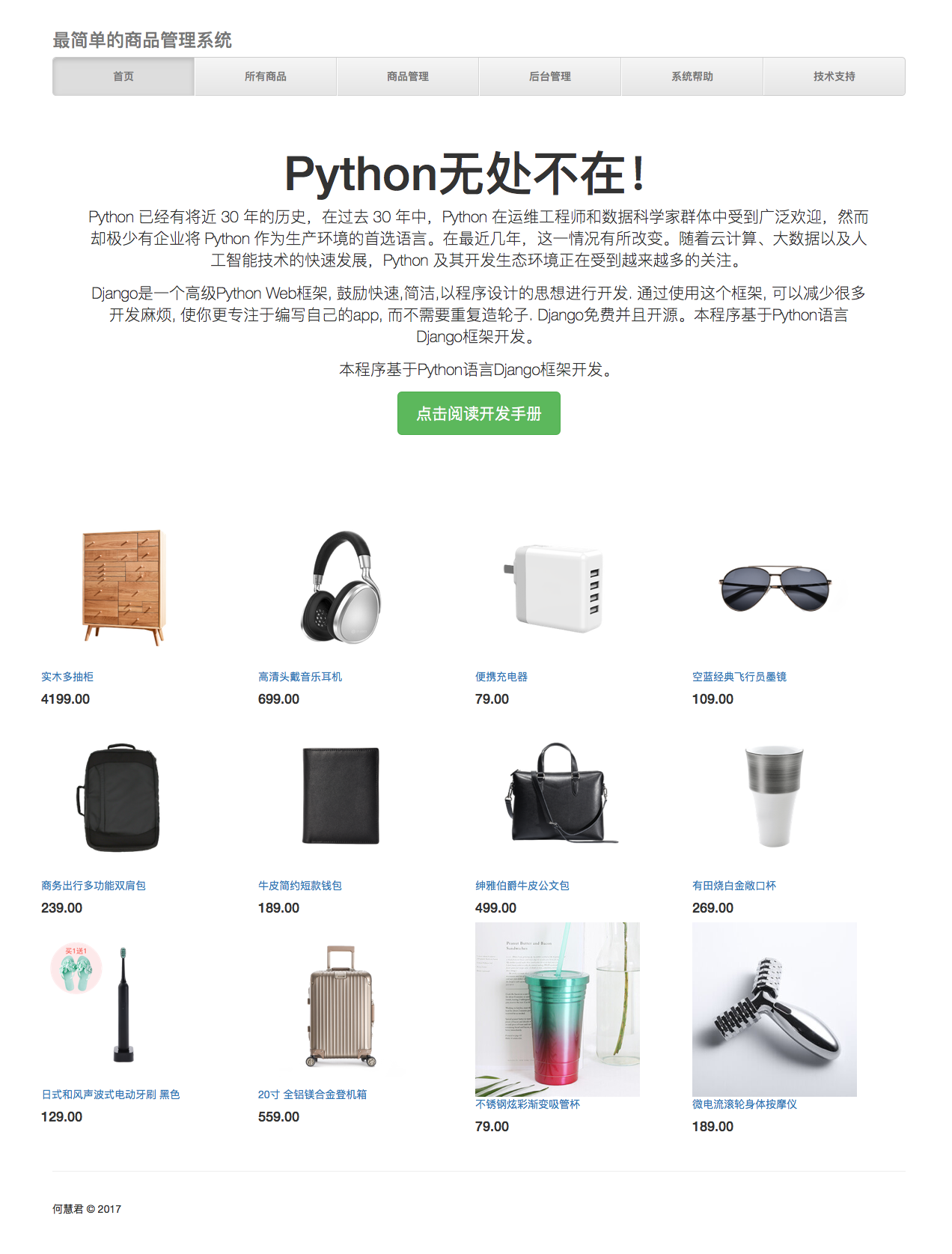
## 功能设计

主要功能和设计考虑如图3-1所示：

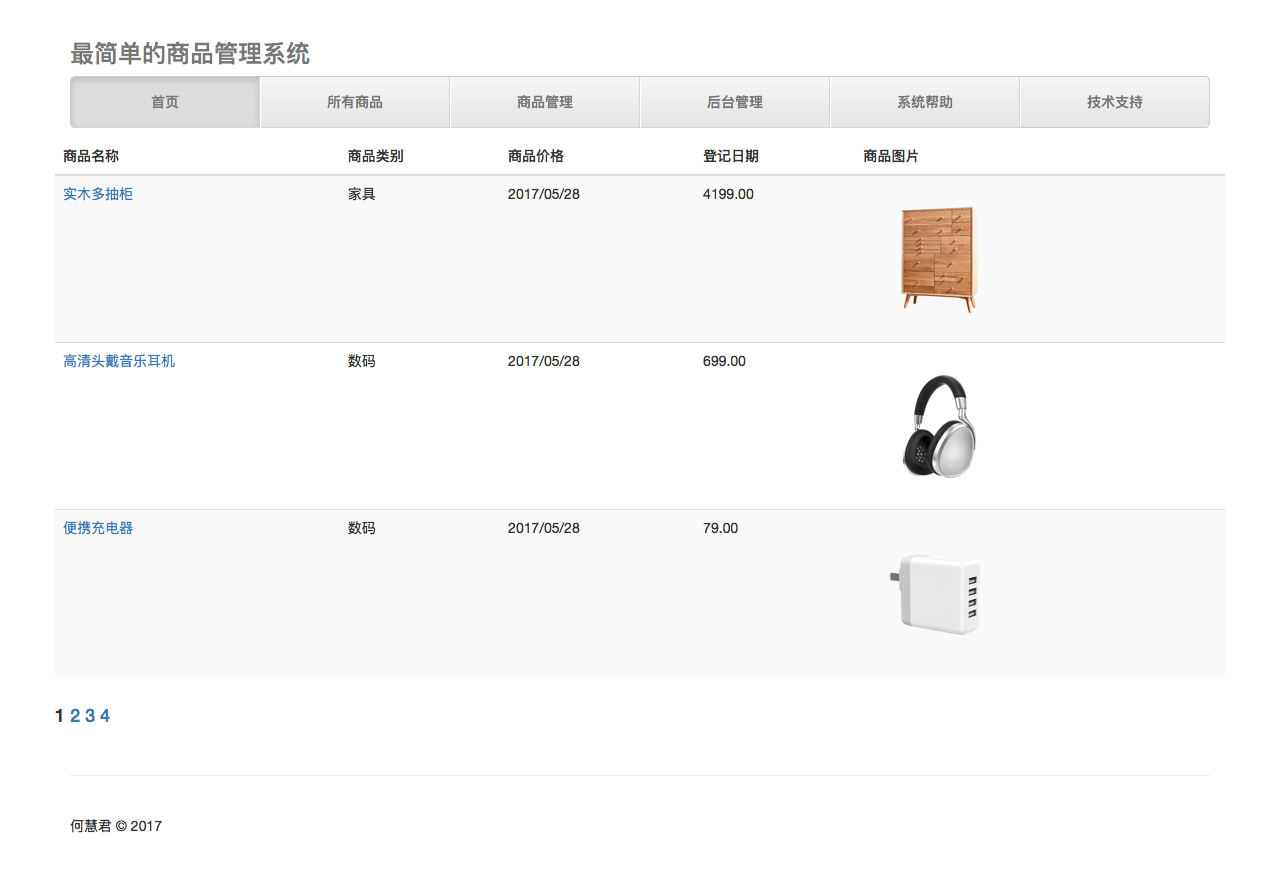


## 主要界面

### 首页



### 所有商品（分页）



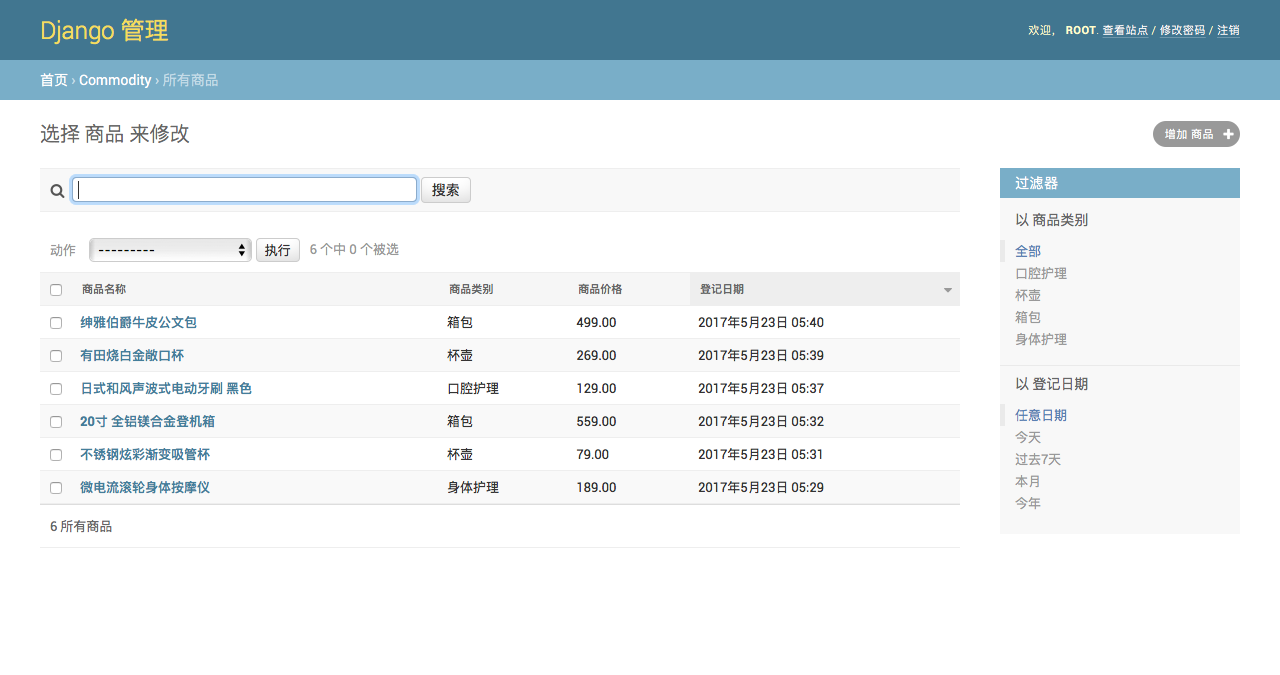
### 商品详情



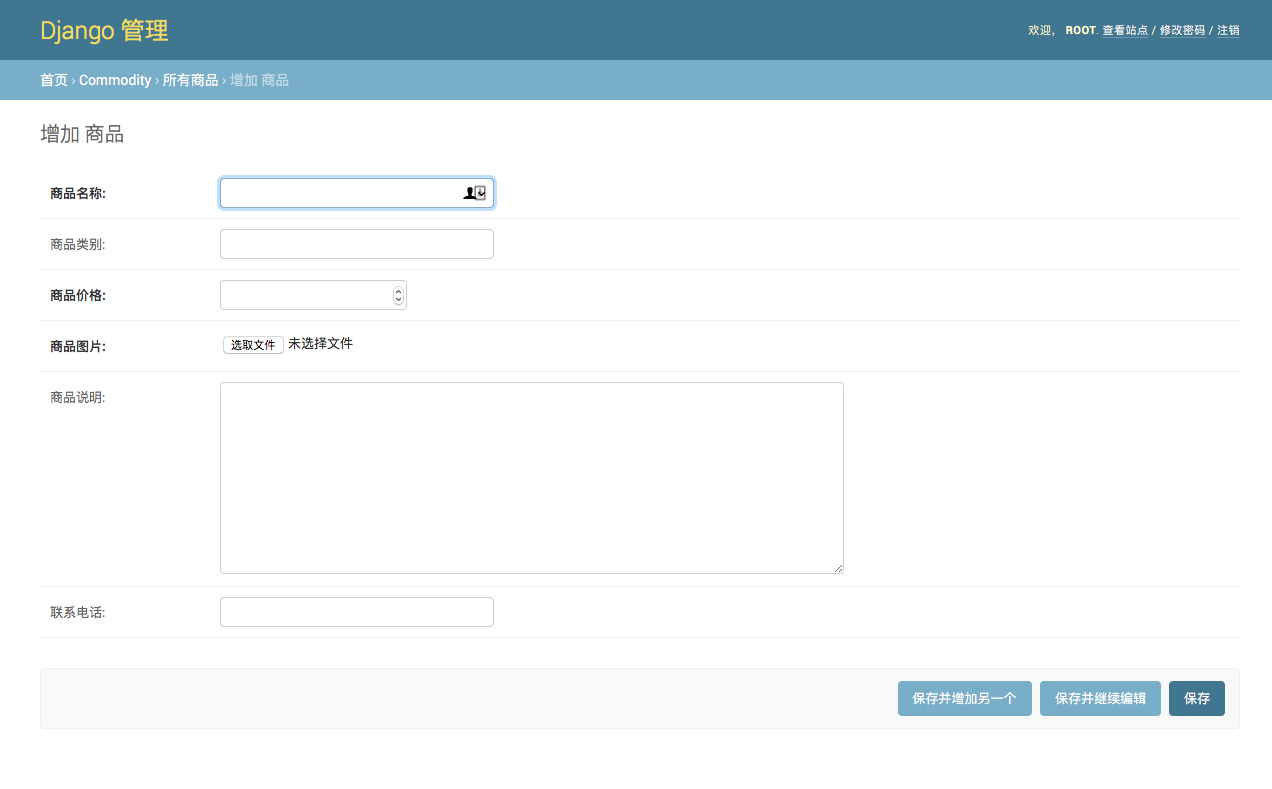
### 管理员登录页



### 商品管理页



### 商品添加页面



# 代码实现

从现在开始正式的进入代码阶段，简述开发过程。

## 创建项目

我们创建一个名为WebStore的Django项目，创建项目的指令[[1]](#footnote-1)（终端）如下:

django-admin.py startproject WebStore

文件结构如下：

WebStore

├── manage.py

└── WebStore

├── \_\_init\_\_.py

├── settings.py

├── urls.py

└── wsgi.py

## 设置数据库

Django项目建成后, 就可以设置数据库了。本文默认使用SQLite数据库, 在WebStore/WebStore/setting.py中可以查看和修改数据库设置:

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.sqlite3',

'NAME': os.path.join(BASE\_DIR, 'db.sqlite3'),

}

}

还可以设置其他数据库, 如MySQL, PostgreSQL, 现在为了便于发布和提供老师审核, 使用默认数据库设置。

## 设置国际化

Django 支持国际化，多语言。Django的国际化是默认开启的，如果不需要国际化支持，可以在设置文件中设置 USE\_I18N = False，那么Django会进行一些优化，不加载国际化支持机制。在WebStore/WebStore/setting.py修改默认的语言和时区。

LANGUAGE\_CODE = 'zh-hans'

TIME\_ZONE = 'Asia/Shanghai'

USE\_I18N = True

## 配置静态文件

静态文件是指网站中的js,css,图片，视频等文件，通常在WebStore/WebStore/setting.py中已经设置好了。在WebStore根目录下创建static文件夹，在static下再创建upload、css、js、image等目录。

STATIC\_URL = '/static/'

STATICFILES\_DIRS = [

os.path.join(BASE\_DIR, "static"), '/Users/alexmac/Library/Mobile\ Documents/com\~apple\~CloudDocs/DEV/WebStore/static/',#物理路径

]

## 创建Django app

在Django中的app被认为是一个功能模块, 与其他的web框架可能有很大的区别, 将不通功能放在不同的app中, 方便代码的复用。在WebStore目录下，终端输入创建app指令：

python manage.py startapp commodity

WebStore根目录下，文件结构如下：

── commodity

│ ├── \_\_init\_\_.py

│ ├── admin.py

│ ├── migrations

│ │ └── \_\_init\_\_.py

│ ├── models.py

│ ├── tests.py

│ └── views.py

├─db.sqlite3

├─manage.py

├─WebStore

├── \_\_init\_\_.py

├── settings.py

├── urls.py

必须在WebStore/WebStore/setting.py下添加新建app

INSTALLED\_APPS = (

...

'commodity', #这里填写的是app的名称

)

## 创建models

在WebStore/commodity/models.py下编写如下程序:

class Commondity(models.Model):  
 CommodityName = models.CharField('商品名称', max\_length=100)  
 CommodityCategory = models.CharField('商品类别', max\_length=50, blank=True)  
 CommodityPrice = models.DecimalField('商品价格', max\_digits=11, decimal\_places=2)  
 CommondityImage = models.ImageField('商品图片', upload\_to='static', default='static/upload/None/no-img.jpg')  
 CommodityDateTime = models.DateTimeField('登记日期', auto\_now\_add=True)  
 CommodityContent = models.TextField('商品说明', blank=True, null=True)  
 CommondityContactMobile = models.CharField('联系电话', max\_length=11, blank=True, null=True)  
  
 def \_\_str\_\_(self):  
 return self.CommodityName # 一般系统默认使用<Commodity: Commodity object> 来表示对象, 通过这个函数可以告诉系统使用CommodityName字段来表示这个对象  
  
 class Meta:  
 verbose\_name = '商品' #给模型起个更好听的名字，这儿相当于进行了汉化。  
 verbose\_name\_plural = '所有商品'  
 # 按时间下降排序  
 ordering = ['-CommodityDateTime']

## 运行程序

在WebStore根目录下，用终端命令行输入以下指令：

$ python manage.py runserver #启动Django中的开发服务器

启动浏览器, 输入<http://127.0.0.1:8000/>, 就可以访问网站了。

## 同步数据库

在WebStore根目录下，用终端命令行输入以下指令：

python manage.py migrate #同步在model中建立的数据库

如果对model进行了修改，需要先执行一次makemigrations命令，再执行migrae。

python manage.py makemigrations #先检查更新

## 设置Admin

Django有一个优秀的特性, 内置了Django admin后台管理界面, 方便管理者进行添加和删除网站的内容. 新建的项目系统已经为我们设置好了后台管理功能，可以在WebStore/WebStore/setting.py中查看。

在WebStore/commodity/admin.py中增加代码，让后台管理界面能对“商品”信息进行管理。默认管理界面中仅显示上面这是的“CommodityName”，为更方便的在后台管理信息，需要增加一些一些代码。具体的如下：

from django.contrib import admin

from commodity.models import Commondity

class AdminCommondity(admin.ModelAdmin):

list\_display = ('CommodityName','CommodityCategory','CommodityPrice','CommodityDateTime')#后台显示的列表内容

search\_fields = ('CommodityName', 'CommodityCategory',)#从哪些字段中搜索

list\_filter = ('CommodityCategory','CommodityDateTime',)#筛选器

admin.site.register(Commondity,AdminCommondity)

## 创建超级用户

初次登陆后台管理界面，需要使用如下命令账号创建超级用户：

python manage.py createsuperuser

按照提示输入用户名、邮箱、密码，创建第一个超级用户。

## 编写template

在commodity目录下创建template目录，在template下增加base.html,做为本文程序的基础模版。模版样式采用Bootstrap样式表。

## 编写首页和商品展示页（模板）

用{% extends "base.html" %}引入模版页面，增加需要展示商品的相关代码。

首页为index.html。<代码见附录>

商品展示页为post.html。<代码见附录>

## 完善Views

通常访问网页程序的逻辑是：request进来->从服务器获取数据->处理数据->把网页呈现出来，Django也是按照这个流程来处理信息的，它使用url和views来处理:

1. url设置相当于客户端向服务器发出request请求的入口, 并用来指明要调用的程序逻辑.
2. views用来处理程序逻辑, 然后呈现到template(一般为GET方法, POST方法略有不同)。
3. template一般为html+CSS的形式, 主要是呈现给用户的表现形式。

Django中views里面的代码就是一个一个函数逻辑, 处理客户端(浏览器)发送的HTTPRequest, 然后返回HTTPResponse。我们在WebStore/commodity/views.py中编写简单的逻辑：

from django.http import HttpResponse

from commodity.models import Commondity

from datetime import datetime

from django.http import Http404

from django.shortcuts import render

from django.core.paginator import PageNotAnInteger, Paginato

def list(request):#系统默认的Paginator

limit = 3 # 每页显示的记录数

post\_list = Commondity.objects.all()#获取全部对象

paginator = Paginator(post\_list, limit) # 实例化一个分页对象

page = request.GET.get('page') # 获取页码

try:

post\_list = paginator.page(page) # 获取某页对应的记录

except PageNotAnInteger: # 如果页码不是个整数

post\_list = paginator.page(1) # 取第一页的记录

except EmptyPage: # 如果页码太大，没有相应的记录

post\_list = paginator.page(paginator.num\_pages) # 取最后一页的记录

return render(request,'list.html', {'post\_list': post\_list})

def index(request):

post\_index = Commondity.objects.all()[:8]#获取全部对象,仅显示前8条数据

return render(request,'index.html',{'post\_index':post\_index})

def detail(request, id):

try:

post = Commondity.objects.get(id=str(id))

except Commondity.DoesNotExist:

raise Http404

return render(request, 'post.html', {'post': post})

## 实现分页（模板）

用{% extends "base.html" %}引入模版页面，增加需要展示商品的相关代码。

在views.py中使用pagination，实现简单分页。<代码见附录>

商品展示页为post.html。<代码见附录>

## 完善url

如何使这个逻辑在http请求进入时, 被调用呢, 这里需要WebStore/WebStore/urls.py中进行url设置，设置了访问主页（list.html）和商品详情页面（post.html）。代码如下：

from django.conf.urls import url,include

from django.contrib import admin

from commodity import views #1.8以上新的写法

urlpatterns = (

url(r'^admin/', admin.site.urls),

url(r'^$', views.index,name='index'), # alex：1.8以上新的写法

url( r'^(?P<id>\d+)/$',views.detail, name='detail'),

url(r'^list/$',views.list),

)

# 结论

随着互联网、HTML5、移动设备的快速发展，Web应用的形式、服务方式等迭代更新很快，掌握基本的开发技术、设计思路等，不仅可以增强个人职业能力，还可以帮助个人或公司创造更有意义的产品或服务。

通过此次商品管理系统的开发，熟悉了HTML、SQL、JS、Python等编程语言，熟悉了MVC架构，更是了解到有很多优秀的程序员，开发了不用重复发明轮子的“框架”，降低了开发的难度，让开发人员能更加分工明确、专注于解决实际问题。

本文不仅记录了一个小程序的开发过程，还可以作为入门教程，供更多初学者参考。为此本文和源码已经发布到了GitHub上，地址：

<https://github.com/hehuijun/webstore2>

并在阿里云上搭建了实际运行环境（由于刚提交的ICP备案信息，可能在6月10日前无法访问），地址：

<http://webstore.hehuijun.com>

# 致谢

转眼间，本学期即将结束了。在这几个月的时间里，继续学习计算机相关课程，收获颇丰。在学习和课程中得到了老师和同学们很多指导和帮助。由于时间太紧，本计划将使用Django开发出更完整的一个知识管理系统（简易blog），发布到云主机上，很遗憾没有实现。但是不会放弃，将继续利用以后的时间，持续学习和完善，完成个人平台的搭建。

首先我要感谢我的指导教师吴素研老师。她治学严谨、知识渊博，鼓励我们对网页编程等相关内容，按照自己的兴趣学习，以便有更深入的理解，确实受益匪浅。

我还要感谢班主任景老师，是她的谆谆教导，让我打定主意，安心学习，今年的课程顺利学完。

此外，我还要感谢同学们，给予很多启发和帮助。

# 参考文献

Django官方网站：<https://docs.djangoproject.com/en/1.11/>

Django基础教程：<http://code.ziqiangxuetang.com/django/django-tutorial.html>

# 源代码

## commodity/apps.py

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | 1 | from django.apps import AppConfig | | 2 |  | | 3 |  | | 4 | *class* CommodityConfig(*AppConfig*): | | 5 | name = 'commodity' | |  |

## commodity/models.py

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | 1 | from django.db import models | | 2 |  | | 3 | # Create your models here. | | 4 | *class* Commondity(*models.Model*): | | 5 | CommodityName = models.CharField('商品名称', *max\_length*=100) | | 6 | CommodityCategory = models.CharField('商品类别', *max\_length*=50, *blank*=True) | | 7 | CommodityPrice = models.DecimalField('商品价格', *max\_digits*=11, *decimal\_places*=2) | | 8 | CommondityImage = models.ImageField('商品图片', *upload\_to*='static', *default*='static/no-img.jpg') | | 9 | CommodityDateTime = models.DateTimeField('登记日期', *auto\_now\_add*=True) | | 10 | CommodityContent = models.TextField('商品说明', *blank*=True, *null*=True) | | 11 | CommondityContactMobile = models.CharField('联系电话', *max\_length*=11, *blank*=True, *null*=True) | | 12 |  | | 13 | *def* \_\_str\_\_(*self*): | | 14 | return self.CommodityName  # 一般系统默认使用<Commodity: Commodity object> 来表示对象, 通过这个函数可以告诉系统使用CommodityName字段来表示这个对象 | | 15 |  | | 16 | *class* Meta: | | 17 | verbose\_name = '商品' #给模型起个更好听的名字，这儿相当于进行了汉化。 | | 18 | verbose\_name\_plural = '所有商品' | | 19 | # 按时间下降排序 | | 20 | ordering = ['-CommodityDateTime'] | |  |

## commodity/views.py

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | 1 | from django.http import HttpResponse | | 2 | from commodity.models import Commondity | | 3 | from datetime import datetime | | 4 | from django.http import Http404 | | 5 | from django.shortcuts import render | | 6 | from django.core.paginator import PageNotAnInteger, Paginator | | 7 |  | | 8 |  | | 9 | # Create your views here. | | 10 |  | | 11 | # def list(request):#没有分页 | | 12 | #     post\_list = Commondity.objects.all()#获取全部对象 | | 13 | #     return render(request,'list.html',{'post\_list':post\_list}) | | 14 |  | | 15 | *def* list(*request*):#系统默认的Paginator | | 16 | limit = 3  # 每页显示的记录数 | | 17 | post\_list = Commondity.objects.all()#获取全部对象 | | 18 | paginator = Paginator(post\_list, limit)  # 实例化一个分页对象 | | 19 | page = request.GET.get('page')  # 获取页码 | | 20 | try: | | 21 | post\_list = paginator.page(page)  # 获取某页对应的记录 | | 22 | except PageNotAnInteger:  # 如果页码不是个整数 | | 23 | post\_list = paginator.page(1)  # 取第一页的记录 | | 24 | except EmptyPage:  # 如果页码太大，没有相应的记录 | | 25 | post\_list = paginator.page(paginator.num\_pages)  # 取最后一页的记录 | | 26 | return render(request,'list.html', {'post\_list': post\_list}) | | 27 | *def* index(*request*): | | 28 | |  | | --- | | post\_index = Commondity.objects.all()[:8]#获取全部对象,仅显示前8条数据 | | | 29 | return render(request,'index.html',{'post\_index':post\_index}) | | 30 |  | | 31 | *def* detail(*request*, *id*): | | 32 | try: | | 33 | post = Commondity.objects.get(*id*=*str*(id)) | | 34 | except Commondity.DoesNotExist: | | 35 | raise Http404 | | 36 | return render(request, 'post.html', {'post': post}) | | 37 |  | |  |

## commodity/admin.py

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | 1 | from django.contrib import admin | | 2 | from commodity.models import Commondity | | 3 |  | | 4 | # Register your models here. | | 5 | *class* AdminCommondity(*admin.ModelAdmin*): | | 6 | list\_display = ('CommodityName','CommodityCategory','CommodityPrice','CommodityDateTime')#后台显示的列表内容 | | 7 | search\_fields = ('CommodityName', 'CommodityCategory',)#从哪些字段中搜索 | | 8 | list\_filter = ('CommodityCategory','CommodityDateTime',)#筛选器 | | 9 |  | | 10 | admin.site.register(Commondity,AdminCommondity) | | 11 |  | |  |

## commodity/templates/base.html

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | 1 | <!DOCTYPE html> | | 2 |  | | 3 | <html lang="zh-CN"> | | 4 | <head> | | 5 | <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"> | | 6 |  | | 7 | <meta http-equiv="X-UA-Compatible" content="IE=edge"> | | 8 | <meta name="viewport" content="width=device-width, initial-scale=1"> | | 9 | <!-- 上述3个meta标签\*必须\*放在最前面，任何其他内容都\*必须\*跟随其后！ --> | | 10 |  | | 11 | <meta name="description" content="商品管理系统"> | | 12 | <meta name="author" content="何慧君"> | | 13 | <!--link rel="icon" href="http://v3.bootcss.com/favicon.ico"--> | | 14 |  | | 15 | <title>最简单的商品管理系统</title> | | 16 |  | | 17 | <!-- Bootstrap core CSS --> | | 18 | <link href="../static/css/bootstrap.min.css" rel="stylesheet"> | | 19 |  | | 20 | <!-- IE10 viewport hack for Surface/desktop Windows 8 bug --> | | 21 | <link href="../static/css/ie10-viewport-bug-workaround.css" rel="stylesheet"> | | 22 |  | | 23 | <!-- Custom styles for this template --> | | 24 | <link href="../static/css/justified-nav.css" rel="stylesheet"> | | 25 |  | | 26 | <!-- Just for debugging purposes. Don't actually copy these 2 lines! --> | | 27 | <!--[if lt IE 9]> | | 28 | <script src="../static/js/ie8-responsive-file-warning.js"></script><![endif]--> | | 29 | <script src="../static/css/ie-emulation-modes-warning.js"></script> | | 30 |  | | 31 | <!-- HTML5 shim and Respond.js for IE8 support of HTML5 elements and media queries --> | | 32 | <!--[if lt IE 9]> | | 33 | <script src="https://cdn.bootcss.com/html5shiv/3.7.3/html5shiv.min.js"></script> | | 34 | <script src="https://cdn.bootcss.com/respond.js/1.4.2/respond.min.js"></script> | | 35 | <![endif]--> | | 36 | </head> | | 37 |  | | 38 | <body> | | 39 |  | | 40 | <div class="container"> | | 41 |  | | 42 | <!-- The justified navigation menu is meant for single line per list item. | | 43 | Multiple lines will require custom code not provided by Bootstrap. --> | | 44 | <div class="masthead"> | | 45 | <h3 class="text-muted">最简单的商品管理系统</h3> | | 46 | <nav> | | 47 | <ul class="nav nav-justified"> | | 48 | <li class="active"><a href="{% url " home" %}">首页</a></li> | | 49 | <li><a href="/admin/commodity/" target="\_blank">商品管理</a></li> | | 50 | <li><a href="/admin" target="\_blank">后台管理</a></li> | | 51 | <li><a href="/static/readme.pdf" target="\_blank">系统帮助</a></li> | | 52 | <li><a href="mailto:he.huijun@gmail.com">技术支持</a></li> | | 53 | </ul> | | 54 | </nav> | | 55 | </div> | | 56 |  | | 57 |  | | 58 | <!-- Example row of columns --> | | 59 |  | | 60 |  | | 61 | <div class="row"> | | 62 | {% block content %} | | 63 | {% endblock %} | | 64 |  | | 65 |  | | 66 | </div> <!-- /container --> | | 67 |  | | 68 |  | | 69 | <!-- IE10 viewport hack for Surface/desktop Windows 8 bug --> | | 70 |  | | 71 | <script src="../static/css/ie10-viewport-bug-workaround.js"></script> | | 72 | <!-- Site footer --> | | 73 | <footer class="footer"> | | 74 | <p>何慧君 © 2017</p> | | 75 | </footer> | | 76 |  | | 77 | </body> | | 78 | </html> | |  |

## commodity/templates/index.html

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | 1 | <!--index.html--> | | 2 | {% extends "base.html" %} | | 3 | {% block content %} | | 4 | <!-- Jumbotron --> | | 5 | <div class="jumbotron"> | | 6 | <h1>Python无处不在！</h1> | | 7 | <p class="lead">Python 已经有将近 30 年的历史，在过去 30 年中，Python 在运维工程师和数据科学家群体中受到广泛欢迎，然而却极少有企业将 Python | | 8 | 作为生产环境的首选语言。在最近几年，这一情况有所改变。随着云计算、大数据以及人工智能技术的快速发展，Python 及其开发生态环境正在受到越来越多的关注。</p> | | 9 | <p class="lead">Django是一个高级Python Web框架, 鼓励快速,简洁,以程序设计的思想进行开发. 通过使用这个框架, 可以减少很多开发麻烦, 使你更专注于编写自己的app, 而不需要重复造轮子. | | 10 | Django免费并且开源。本程序基于Python语言Django框架开发。</p> | | 11 | <p class="lead">本程序基于Python语言Django框架开发。</p> | | 12 | <p><a class="btn btn-lg btn-success" href="/static/readme.pdf" role="button" target="\_blank">点击阅读开发手册</a></p> | | 13 | </div> | | 14 | <div> | | 15 | {% for post in post\_index %} | | 16 | <div style="*width*:290px" class="pull-left"> | | 17 | <div class="display: block"> | | 18 | <a href="{% url 'detail' id=post.id %}"> | | 19 | <img class="img-responsive" src="../{{ post.CommondityImage.url}}" alt="{{ post.CommodityName }}" height="220" width="220"/> | | 20 | </a> | | 21 | </div> | | 22 | <div> | | 23 | <div> | | 24 | <a class="display: block" href="{% url 'detail' id=post.id %}" class="center-block"> {{post.CommodityName }}</a> | | 25 | </div> | | 26 | <div class="display: block"><h4> | | 27 | {{ post.CommodityPrice }}</h4> | | 28 | </div> | | 29 | </div> | | 30 | </div> | | 31 | {% endfor %} | | 32 | </div> | | 33 | {% endblock %} | |  |

## commodity/templates/list.html

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | 1 | <!--list.html--> | | 2 | {% extends "base.html" %} | | 3 |  | | 4 | {% block content %} | | 5 |  | | 6 | <div> | | 7 | <p></p> | | 8 | </div> | | 9 | <div> | | 10 | <p></p> | | 11 | </div> | | 12 | <div class="table-responsive"> | | 13 | <table class="table table-striped"> | | 14 | <thead> | | 15 | <tr> | | 16 | <th>商品名称</th> | | 17 | <th>商品类别</th> | | 18 | <th>商品价格</th> | | 19 | <th>登记日期</th> | | 20 | <th>商品图片</th> | | 21 | </tr> | | 22 | </thead> | | 23 | <tbody class="table-hover"> | | 24 | {% for post in post\_list %} | | 25 | <tr> | | 26 | <td><a href="{% url "detail" id=post.id %}"> {{ post.CommodityName }}</a></td> | | 27 | <td>{{ post.CommodityCategory }}</td> | | 28 | <td>{{ post.CommodityDateTime |date:"Y/m/d"}}</td> | | 29 | <td>{{ post.CommodityPrice }}</td> | | 30 | <td><img src="../{{ post.CommondityImage.url }}" alt="{{ post.CommodityName }}" height="150" width="150"></td> | | 31 | </tr> | | 32 | {% endfor %} | | 33 | </tbody> | | 34 | </table> | | 35 | </div> | | 36 | <div float="right"> | | 37 | <h4> | | 38 | {% for page\_number in post\_list.paginator.page\_range %} | | 39 | {% ifequal page\_number  post\_list.number %} | | 40 | {{ page\_number }} | | 41 | {% else %} | | 42 | <a href="?page={{ page\_number }}">{{ page\_number }}</a> | | 43 | {% endifequal %} | | 44 | {% endfor %} | | 45 | </h4> | | 46 | </div> | | 47 | {% endblock %} | |  |

## commodity/templates/post.html

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | 1 | {% extends "base.html" %} | | 2 |  | | 3 | {% block content %} | | 4 | <div class="container"> | | 5 | <!-- Example row of columns --> | | 6 | <div class="row"> | | 7 | <div class="col-md-4"> | | 8 | <div> | | 9 | <p>&nbsp;</p> | | 10 | </div> | | 11 | <p><a class="btn btn-default" role="button">首页 </a> | | 12 | <a>&nbsp;>&nbsp;</a> <a class="btn btn-default" role="button">{{ post.CommodityCategory }} </a> | | 13 | <table width="700" border="0"> | | 14 | <tbody> | | 15 | <tr> | | 16 | <td width="277" rowspan="4"><img src="../{{ post.CommondityImage.url}}" height="330" width="330"/> | | 17 | </td> | | 18 | <td width="339" height="36"><h2>{{ post.CommodityName }}</h2> | | 19 | </td> | | 20 | </tr> | | 21 | <tr> | | 22 | <td height="41">¥：<a class="btn btn-primary btn-lg" role="button">{{ post.CommodityPrice }}</a> | | 23 | &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp; | | 24 | 登记日期：{{ post.CommodityDateTime|date:"Y/m/d" }} | | 25 | </td> | | 26 | </tr> | | 27 | <tr> | | 28 | <td height="167"><p>商品介绍：</p> | | 29 | {{ post.CommodityContent }}</td> | | 30 | </tr> | | 31 | <tr> | | 32 | <td height="51">联系电话：<h4>{{ post.CommondityContactMobile }}</h4></td> | | 33 | </tr> | | 34 | </tbody> | | 35 | </table> | | 36 |  | | 37 | </div> | | 38 | </div> | | 39 | </div> | | 40 |  | | 41 |  | | 42 | {% endblock %} | |  |

## WebStore/urls.py

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | 1 | from django.conf.urls import url,include | | 2 | from django.contrib import admin | | 3 | from commodity import views #alex：1.8以上新的写法 | | 4 |  | | 5 | urlpatterns = ( | | 6 | url(*r*'^admin/', admin.site.urls), | | 7 | url(*r*'^$', views.index,*name*='index'),  # alex：1.8以上新的写法 | | 8 | url( *r*'^(?P<id>\d+)/$',views.detail, *name*='detail'), | | 9 | url(*r*'^list/$',views.list), | | 10 | ) | |  |

## WebStore/settings.py

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | 1 | """ | | 2 | Django settings for WebStore project. | | 3 |  | | 4 | Generated by 'django-admin startproject' using Django 1.10.5. | | 5 |  | | 6 | For more information on this file, see | | 7 | https://docs.djangoproject.com/en/1.10/topics/settings/ | | 8 |  | | 9 | For the full list of settings and their values, see | | 10 | https://docs.djangoproject.com/en/1.10/ref/settings/ | | 11 | """ | | 12 |  | | 13 | import os | | 14 |  | | 15 | # Build paths inside the project like this: os.path.join(BASE\_DIR, ...) | | 16 | BASE\_DIR = os.path.dirname(os.path.dirname(os.path.abspath(\_\_file\_\_))) | | 17 |  | | 18 |  | | 19 | # Quick-start development settings - unsuitable for production | | 20 | # See https://docs.djangoproject.com/en/1.10/howto/deployment/checklist/ | | 21 |  | | 22 | # SECURITY WARNING: keep the secret key used in production secret! | | 23 | SECRET\_KEY = 'e=w9\_@l$jhia=d)b(ii=j$)09&!&\*22swf&3n0^az\_r+z(sii-' | | 24 |  | | 25 | # SECURITY WARNING: don't run with debug turned on in production! | | 26 | DEBUG = True | | 27 |  | | 28 | ALLOWED\_HOSTS = [] | | 29 |  | | 30 |  | | 31 | # Application definition | | 32 |  | | 33 | INSTALLED\_APPS = ( | | 34 | 'django.contrib.admin', | | 35 | 'django.contrib.auth', | | 36 | 'django.contrib.contenttypes', | | 37 | 'django.contrib.sessions', | | 38 | 'django.contrib.messages', | | 39 | 'django.contrib.staticfiles', | | 40 | 'commodity',  # 这里填写的是app的名 | | 41 | ) | | 42 |  | | 43 | MIDDLEWARE = [ | | 44 | 'django.middleware.security.SecurityMiddleware', | | 45 | 'django.contrib.sessions.middleware.SessionMiddleware', | | 46 | 'django.middleware.common.CommonMiddleware', | | 47 | 'django.middleware.csrf.CsrfViewMiddleware', | | 48 | 'django.contrib.auth.middleware.AuthenticationMiddleware', | | 49 | 'django.contrib.messages.middleware.MessageMiddleware', | | 50 | 'django.middleware.clickjacking.XFrameOptionsMiddleware', | | 51 | ] | | 52 |  | | 53 | ROOT\_URLCONF = 'WebStore.urls' | | 54 |  | | 55 | TEMPLATES = [ | | 56 | { | | 57 | 'BACKEND': 'django.template.backends.django.DjangoTemplates', | | 58 | 'DIRS': [], | | 59 | 'APP\_DIRS': True, | | 60 | 'OPTIONS': { | | 61 | 'context\_processors': [ | | 62 | 'django.template.context\_processors.debug', | | 63 | 'django.template.context\_processors.request', | | 64 | 'django.contrib.auth.context\_processors.auth', | | 65 | 'django.contrib.messages.context\_processors.messages', | | 66 | ], | | 67 | }, | | 68 | }, | | 69 | ] | | 70 |  | | 71 | WSGI\_APPLICATION = 'WebStore.wsgi.application' | | 72 |  | | 73 |  | | 74 | # Database | | 75 | # https://docs.djangoproject.com/en/1.10/ref/settings/#databases | | 76 |  | | 77 | DATABASES = { | | 78 | 'default': { | | 79 | 'ENGINE': 'django.db.backends.sqlite3', | | 80 | 'NAME': os.path.join(BASE\_DIR, 'db.sqlite3'), | | 81 | } | | 82 | } | | 83 |  | | 84 |  | | 85 | # Password validation | | 86 | # https://docs.djangoproject.com/en/1.10/ref/settings/#auth-password-validators | | 87 |  | | 88 | AUTH\_PASSWORD\_VALIDATORS = [ | | 89 | { | | 90 | 'NAME': 'django.contrib.auth.password\_validation.UserAttributeSimilarityValidator', | | 91 | }, | | 92 | { | | 93 | 'NAME': 'django.contrib.auth.password\_validation.MinimumLengthValidator', | | 94 | }, | | 95 | { | | 96 | 'NAME': 'django.contrib.auth.password\_validation.CommonPasswordValidator', | | 97 | }, | | 98 | { | | 99 | 'NAME': 'django.contrib.auth.password\_validation.NumericPasswordValidator', | | 100 | }, | | 101 | ] | | 102 |  | | 103 |  | | 104 | # Internationalization | | 105 | # https://docs.djangoproject.com/en/1.10/topics/i18n/ | | 106 |  | | 107 | #LANGUAGE\_CODE = 'en-us' | | 108 | LANGUAGE\_CODE = 'zh-hans' | | 109 |  | | 110 |  | | 111 | TIME\_ZONE = 'Asia/Shanghai' | | 112 |  | | 113 | USE\_I18N = True | | 114 |  | | 115 | USE\_L10N = True | | 116 |  | | 117 | USE\_TZ = True | | 118 |  | | 119 |  | | 120 | # Static files (CSS, JavaScript, Images) | | 121 | # https://docs.djangoproject.com/en/1.10/howto/static-files/ | | 122 |  | | 123 | STATIC\_URL = '/static/' | | 124 | STATICFILES\_DIRS = ( | | 125 | os.path.join(BASE\_DIR, "static"), | | 126 | '/Users/alexmac/Library/Mobile\ Documents/com\~apple\~CloudDocs/DEV/WebStore2/static/',  # 根据本地文件夹的不同而需要进行修改 | | 127 | ) | |  |

1. 在windows下可以使用django-admin startproject WebStore的命令，无py后缀。 [↑](#footnote-ref-1)