# 001 Introduction

# 002 Walkthrough

# 003 Requirement

Tool

Braintree : 支付

Sublime

# 004 Setup a previous project

从github repository拿到trydjango的git地址<git@github.com:navicester/trydjango18course.git>

或者<https://github.com/codingforentrepreneurs/ecommerce-2>

克隆基础项目

>**git clone git@github.com:navicester/trydjango18course.git**

**virtualenv .** 试了一下不工作，还是回到父目录执行

安装request组件

>**pip install requests[security]** 有时安装不成功

**>pip freeze**

cffi==1.8.3

cryptography==1.5.1

enum34==1.1.6

idna==2.1

ipaddress==1.0.17

ndg-httpsclient==0.4.2

pyasn1==0.1.9

pycparser==2.14

pyOpenSSL==16.1.0

requests==2.11.1

six==1.10.0

>**pip install -r** **requirements.txt**

更新django到1.8版本

**>pip install django==1.8.4**

**>pip install django-registration-redux --upgrade**

更新安装组件

**>pip freeze > requirements.txt**

删除老的数据库

**>rm db.sqlite3**

数据库迁移

**>python manage.py migrate**

创建超级用户

**>python manage.py createsuperuser**

# 005 Trydjano18 to ecommerce

将文件里的trydjango18全部替换成ecommerce2，并添加新的git repository

Trydjango18目录改成ecommerce2

>rm -rf .git 切记

git init

git remote add origin git@github.com:navicester/ecommerce2course.git

git add .

git push -u origin master

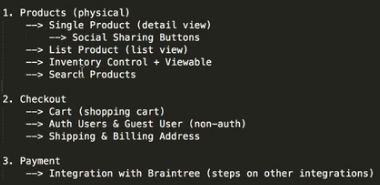
如果撤销commit

git reset --hard <commit\_id>

git push origin HEAD –force

其他:  
    根据–soft –mixed –hard，会对working tree和index和HEAD进行重置:  
    git reset –mixed：此为默认方式，不带任何参数的git reset，即时这种方式，它回退到某个版本，只保留源码，回退commit和index信息  
    git reset –soft：回退到某个版本，只回退了commit的信息，不会恢复到index file一级。如果还要提交，直接commit即可  
    git reset –hard：彻底回退到某个版本，本地的源码也会变为上一个版本的内容  
    HEAD 最近一个提交  
    HEAD^ 上一次  
    <commit\_id> 每次commit的SHA1值. 可以用git log 看到,也可以在页面上commit标签页里找到.

# 006 Project Roadmap



# 007 Product App

添加Product application

* 创建app python manage.py startapp \*app\_name\*
* 定义Product类 (**Model**)
* 在settings里的INSTALLED\_APPS添加这个新加的APP
* 添加Admin接口

之后会实现各个子功能

* DetaiView
* 008 定义ProductDetailView (**View**)
* 009 添加ProductDetailView入口 (**url**)
* 010 实现模板 (**template**)
* ListView
* 011 添加ProductListView (**View**), 模板(**template**), 入口 (**url**)
* 012 模板里添加直接访问Model实例的链接 (**get\_absolute\_url**)
* 013 定制Model queryset (**Model Manager**)
* Variation
* 014 定义Variation (**model**)，添加admin，更新Product模板添加Variation选项(**template**)
* 015 Product保存时，如果没有型号，则添加一个默认型号 (**Post Save**)
* Product View Layout
* 016 分为两列，左边(col-sm-8)显示title，description，右边显示Variation
* 017 图片上传功能 (**image**)
* 018 搜索功能 (**search**)
* 019 库存显示功能，应用(**ModelFormset**)功能
* 020 库存登录才能访问，(**LoginRequiredMixin**)
* 021 消息 (**Message**)
* 022 分享
* 023 动态更新价格 (**Jquery**)
* 024 仅一个型号时的价格显示 (**\_Set.first**)
* Category

025 定义Category，并为Product定义元素Categories (**m2m**)和Default Category

026 Detail View，

027 显示相关产品 (同种Category) (**Distinct**)

**>****python manage.py startapp product**

在products.models添加product类

**from** django**.**db **import** models

# Create your models here.

**class** **Product(**models**.**Model**):**

title **=** models**.**CharField**(**max\_length**=**120**)**

description **=** models**.**TextField**(**blank**=True,** null**=True)**

price **=** models**.**DecimalField**(**decimal\_places**=**2**,** max\_digits**=**20**)**

active **=** models**.**BooleanField**(**default**=True)**

**def** \_\_unicode\_\_**(**self**):** #def \_\_str\_\_(self):

**return** self**.**title

settings里添加products

INSTALLED\_APPS **=** **(**

'django.contrib.admin'**,**

'django.contrib.auth'**,**

'django.contrib.contenttypes'**,**

'django.contrib.sessions'**,**

'django.contrib.messages'**,**

'django.contrib.sites'**,**

'django.contrib.staticfiles'**,**

'newsletter'**,**

'products'**,**

'crispy\_forms'**,**

'registration'**,**

**)**

**>python manage.py makemigrations**

**>python manage.py migrate**

添加admin接口. products.admin.py

**from** **.**models **import** Product

# Register your models here.

admin**.**site**.**register**(**Product**)**

# 008 Product Detail View

参考

<https://docs.djangoproject.com/en/1.8/ref/class-based-views/generic-display/#detailview>

**Generic display views**

* **DetailView**
* **ListView**

**classdjango.views.generic.detail.DetailView**

While this view is executing, **self.object** will contain the object that the view is operating upon.

This view inherits methods and attributes from the following views:

* [**django.views.generic.detail.SingleObjectTemplateResponseMixin**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectTemplateResponseMixin)
* [**django.views.generic.base.TemplateResponseMixin**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-simple/#django.views.generic.base.TemplateResponseMixin)
* **django.views.generic.detail.BaseDetailView**
* [**django.views.generic.detail.SingleObjectMixin**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin)
* [**django.views.generic.base.View**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/base/#django.views.generic.base.View)

**Method Flowchart**

1. [**dispatch()**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/base/#django.views.generic.base.View.dispatch)
2. [**http\_method\_not\_allowed()**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/base/#django.views.generic.base.View.http_method_not_allowed)
3. [**get\_template\_names()**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-simple/#django.views.generic.base.TemplateResponseMixin.get_template_names)
4. [**get\_slug\_field()**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin.get_slug_field)
5. [**get\_queryset()**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin.get_queryset)
6. [**get\_object()**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin.get_object)
7. [**get\_context\_object\_name()**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin.get_context_object_name)
8. [**get\_context\_data()**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin.get_context_data)
9. **get()**
10. [**render\_to\_response()**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-simple/#django.views.generic.base.TemplateResponseMixin.render_to_response)

**classdjango.views.generic.base.View**

**dispatch(request, \*args, \*\*kwargs)**

The **view** part of the view – the method that accepts a **request** argument plus arguments, and returns a HTTP response.

The default implementation will inspect the HTTP method and attempt to delegate to a method that matches the HTTP method; a **GET** will be delegated to **get()**, a **POST** to **post()**, and so on.

By default, a **HEAD** request will be delegated to **get()**. If you need to handle **HEAD** requests in a different way than**GET**, you can override the **head()** method. See [Supporting other HTTP methods](https://docs.djangoproject.com/en/1.8/topics/class-based-views/#supporting-other-http-methods) for an example.

**http\_method\_not\_allowed(request, \*args, \*\*kwargs)**

If the view was called with a HTTP method it doesn’t support, this method is called instead.

The default implementation returns **HttpResponseNotAllowed** with a list of allowed methods in plain text.

**classdjango.views.generic.base.TemplateResponseMixin**

**get\_template\_names()**

Returns a list of template names to search for when rendering the template.

If [**template\_name**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-simple/#django.views.generic.base.TemplateResponseMixin.template_name) is specified, the default implementation will return a list containing [**template\_name**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-simple/#django.views.generic.base.TemplateResponseMixin.template_name) (if it is specified).

**render\_to\_response(context, \*\*response\_kwargs)**[**¶**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-simple/#django.views.generic.base.TemplateResponseMixin.render_to_response)

Returns a **self.response\_class** instance.

If any keyword arguments are provided, they will be passed to the constructor of the response class.

Calls [**get\_template\_names()**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-simple/#django.views.generic.base.TemplateResponseMixin.get_template_names) to obtain the list of template names that will be searched looking for an existent template.

**classdjango.views.generic.detail.SingleObjectMixin**

**get\_slug\_field()**[**¶**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin.get_slug_field)

Returns the name of a slug field to be used to look up by slug. By default this simply returns the value of [**slug\_field**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin.slug_field).

**get\_queryset()**[**¶**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin.get_queryset)

Returns the queryset that will be used to retrieve the object that this view will display. By default, [**get\_queryset()**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin.get_queryset)returns the value of the [**queryset**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin.queryset) attribute if it is set, otherwise it constructs a [**QuerySet**](https://docs.djangoproject.com/en/1.8/ref/models/querysets/#django.db.models.query.QuerySet) by calling the **all()**method on the [**model**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin.model) attribute’s default manager.

**get\_object(queryset=None)**[**¶**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin.get_object)

Returns the single object that this view will display. If **queryset** is provided, that queryset will be used as the source of objects; otherwise, [**get\_queryset()**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin.get_queryset) will be used. **get\_object()** looks for a [**pk\_url\_kwarg**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin.pk_url_kwarg) argument in the arguments to the view; if this argument is found, this method performs a primary-key based lookup using that value. If this argument is not found, it looks for a [**slug\_url\_kwarg**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin.slug_url_kwarg) argument, and performs a slug lookup using the [**slug\_field**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin.slug_field).

**get\_context\_object\_name(obj)**[**¶**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin.get_context_object_name)

Return the context variable name that will be used to contain the data that this view is manipulating. If[**context\_object\_name**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin.context_object_name) is not set, the context name will be constructed from the **model\_name** of the model that the queryset is composed from. For example, the model **Article** would have context object named **'article'**.

**get\_context\_data(\*\*kwargs)**[**¶**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin.get_context_data)

Returns context data for displaying the list of objects.

The base implementation of this method requires that the **self.object** attribute be set by the view (even if **None**). Be sure to do this if you are using this mixin without one of the built-in views that does so.

It returns a dictionary with these contents:

* **object**: The object that this view is displaying (**self.object**).
* **context\_object\_name**: **self.object** will also be stored under the name returned by[**get\_context\_object\_name()**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin.get_context_object_name), which defaults to the lowercased version of the model name.

**classdjango.views.generic.list.ListView**[**¶**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/generic-display/#django.views.generic.list.ListView)

A page representing a list of objects.

While this view is executing, **self.object\_list** will contain the list of objects (usually, but not necessarily a queryset) that the view is operating upon.

**Ancestors (MRO)**

This view inherits methods and attributes from the following views:

* [**django.views.generic.list.MultipleObjectTemplateResponseMixin**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-multiple-object/#django.views.generic.list.MultipleObjectTemplateResponseMixin)
* [**django.views.generic.base.TemplateResponseMixin**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-simple/#django.views.generic.base.TemplateResponseMixin)
* [**django.views.generic.list.BaseListView**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/generic-display/#django.views.generic.list.BaseListView)
* [**django.views.generic.list.MultipleObjectMixin**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-multiple-object/#django.views.generic.list.MultipleObjectMixin)
* [**django.views.generic.base.View**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/base/#django.views.generic.base.View)

**Method Flowchart**

1. [**dispatch()**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/base/#django.views.generic.base.View.dispatch)
2. [**http\_method\_not\_allowed()**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/base/#django.views.generic.base.View.http_method_not_allowed)
3. [**get\_template\_names()**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-simple/#django.views.generic.base.TemplateResponseMixin.get_template_names)
4. [**get\_queryset()**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-multiple-object/#django.views.generic.list.MultipleObjectMixin.get_queryset)
5. [**get\_context\_object\_name()**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-multiple-object/#django.views.generic.list.MultipleObjectMixin.get_context_object_name)
6. [**get\_context\_data()**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-multiple-object/#django.views.generic.list.MultipleObjectMixin.get_context_data)
7. **get()**
8. [**render\_to\_response()**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-simple/#django.views.generic.base.TemplateResponseMixin.render_to_response)

创建Product Detail View

* 从DetailView继承，并设置model
* 指定template/context, 并渲染render

更新products.views，创建ProductDetailView，并指定template和context （测试用）

**from** django**.**views**.**generic**.**detail **import** DetailView

**from** **.**models **import** Product

# Create your views here.

**class** **ProductDetailView(**DetailView**):**

model **=** Product

**def** product\_detail\_view\_func**(**request**,** id**):**

product\_instance **=** Product**.**objects**.**get**(**id**=**id**)**

template **=** "products/product\_detail.html"

context **=** **{**

"object"**:** product\_instance

**}**

**return** render**(**request**,** template**,** context**)**

<http://127.0.0.1:8000/products/1/>

# 009 url with django app

实现DetailView的显示，其通过测试函数分解其功能

<https://github.com/codingforentrepreneurs/Guides/blob/master/all/common_url_regex.md>

在ecommerce2.url添加入口

url**(**r'^products/'**,** include**(**'products.urls'**)),**

新建文件products.url

其中, as\_view将class based view转换为function based view

P<id>这个跟view里面的参数是一致的

**from** django**.**conf **import** settings

**from** django**.**conf**.**urls **import** include**,** url

**from** django**.**conf**.**urls**.**static **import** static

**from** django**.**contrib **import** admin

**from** **.**views **import** ProductDetailView

urlpatterns **=** **[**

# Examples:

url**(**r'^(?P<pk>\d+)/$'**,** ProductDetailView**.**as\_view**(),** name**=**'product\_detail'**),**

#url(r'^(?P<id>\d+)', 'products.views.product\_detail\_view\_func', name='product\_detail\_function'),

**]**

# 010 Add html templates

创建products/templates/products/product\_detail.htnl

DetailView的默认名字是template\_name = "<appname>/<modelname>\_detail.html"

可以通过更新这个名字来修改template位置

<http://127.0.0.1:8000/products/1/>

可以访问刚刚创建的product

如果访问的数据越界了，需要做一些处理

**from** django**.**shortcuts **import** render**,** get\_object\_or\_404

**from** django**.**http **import** Http404

**def** product\_detail\_view\_func**(**request**,** id**):**

#product\_instance = Product.objects.get(id=id)

product\_instance **=** get\_object\_or\_404**(**Product**,** id**=**id**)**

**try:**

product\_instance **=** Product**.**objects**.**get**(**id**=**id**)**

**except** Product**.**DoesNotExist**:**

**raise** Http404

**except:**

**raise** Http404

template **=** "products/product\_detail.html"

context **=** **{**

"object"**:** product\_instance

**}**

**return** render**(**request**,** template**,** context**)**

# 011 ListView

<https://docs.djangoproject.com/en/1.8/ref/class-based-views/generic-display/#listview>

ProductListView实现

* 添加入口 (url), \*\*ListView.as\_view()
* 添加ProductListView，从ListView继承
* 添加模板

更新products.urls

**from** **.**views **import** ProductDetailView**,** ProductListView

urlpatterns **=** **[**

# Examples:

url**(**r'^$'**,** ProductListView**.**as\_view**(),** name**=**'products'**),**

**]**

<http://127.0.0.1:8000/products/>

更新products.views，添加listview的处理

**from** django**.**views**.**generic**.**list **import** ListView

**from** django**.**utils **import** timezone

**class** **ProductListView(**ListView**):**

model **=** Product

**def** get\_context\_data**(**self**,** **\***args**,** **\*\***kwargs**):**

context **=** super**(**ProductListView**,** self**).**get\_context\_data**(\***args**,** **\*\***kwargs**)**

context**[**"now"**]** **=** timezone**.**now**()**

**return** context

添加products/templates/products/product\_list.html

**{% extends "base.html" %}**

**{% block content %}**

<table>

**{% for object in object\_list %}**

<tr>

<td>**{{object.title}}**</td>

</tr>

**{% endfor %}**

</table>

**{% endblock %}**

# 012 Using Links for Model Instance

实现：模板里添加直接访问Model实例的链接 (get\_absolute\_url)

Products/templates/products/product\_list.html

<td><a href=**"/products/{{object.pk}}/"**>**{{object.title}}**</a></td>

<td><a href=**"{% url 'product\_detail' pk=object.pk %}"**>**{{object.title}}**</a></td>

可以用下面的方式

<td><a href=**"{{ object.get\_absolute\_url }}"**>**{{object.title}}**</a></td>

更新models

**from** django**.**core**.**urlresolvers **import** reverse

**class** **Product(**models**.**Model**):**

**def** get\_absolute\_url**(**self**):**

**return** reverse**(**"product\_detail"**,** kwargs**={**"pk"**:** self**.**pk**})**

# 013 Model Manager

实现：定制化model queryset, 过滤model里面的一些objects

<https://docs.djangoproject.com/en/1.10/topics/db/managers/>

Model里添加Model Manager

class ProductQuerySet(models.query.QuerySet):

def active(self):

return self.filter(active=True)

class ProductManager(models.Manager):

def get\_queryset(self):

return ProductQuerySet(self.model, using=self.\_db)

def all(self, \*args, \*\*kwargs):

return self.get\_queryset().active()

**class** **Product(**models**.**Model**):**

title **=** models**.**CharField**(**max\_length**=**120**)**

description **=** models**.**TextField**(**blank**=True,** null**=True)**

price **=** models**.**DecimalField**(**decimal\_places**=**2**,** max\_digits**=**20**)**

active **=** models**.**BooleanField**(**default**=True)**

objects **=** ProductManager**()**

view里更新queryset

**class** **ProductListView(**ListView**):**

model **=** Product

queryset **=** Product**.**objects**.**all**()**

# 014 Product Variation

实现：添加产品型号

* 定义Variation (model)
* 添加admin接口
* 更新模板，添加Variation选项

添加类variation，文件products.models

class Variation(models.Model):

product = models.ForeignKey(Product)

title = models.CharField(max\_length=120)

price = models.DecimalField(decimal\_places=2, max\_digits=20)

sale\_price = models.DecimalField(decimal\_places=2, max\_digits=20, null=True, blank=True)

active = models.BooleanField(default=True)

inventory = models.IntegerField(null=True, blank=True) #refer none == unlimited amount

def \_\_unicode\_\_(self):

return self.title

def get\_price(self):

if self.sale\_price is not None:

return self.sale\_price

else:

return self.price

def get\_absolute\_url(self):

return self.product.get\_absolute\_url()

添加admin接口

**from** **.**models **import** Variation

admin**.**site**.**register**(**Variation**)**

更新ProductDetail

<select class='form-control'>

{% for vari\_obj in object.variation\_set.all %}

<option value = "{{vari\_obj.id}}">

{{vari\_obj}}

</option>

{% endfor %}

</select>

更新一些css符合commerce的风格

# 015 Post Save Signal for Variation

<https://docs.djangoproject.com/en/1.8/ref/signals/#post-save>

实现：Post Save，新建一个product时，如果没有variation，那么创建一个default的variation

如果产品保存是没有型号variation，则自动创建一个

在model添加post save处理

新建一个product时，如果没有variation，那么创建一个default的variation

**def** product\_post\_saved\_receiver**(**sender**,** instance**,** created**,** **\***args**,** **\*\***kwargs**):**

product **=** instance

variations **=** product**.**variation\_set**.**all**()**

**if** variations**.**count**()** **==** 0**:**

new\_var **=** Variation**()**

new\_var**.**product **=** product

new\_var**.**title **=** "Default"

new\_var**.**price **=** product**.**price

new\_var**.**save**()**

post\_save**.**connect**(**product\_post\_saved\_receiver**,** sender**=**Product**)**

在template product\_detail.html过滤掉默认创建的variation

**{% if object.variation\_set.count > 0 %}**

<select class=**'form-control'**>

**{% for vari\_obj in object.variation\_set.all %}**

<option value = **"{{vari\_obj.id}}"**>

**{{vari\_obj}}**

</option>

**{% endfor %}**

</select>

**{% endif %}**

<br/>

<a href=**"#"**>**Add to Cart**</a>

# 016 Project Detail Layout

实现：分为两栏，左边是描述，右边是Variation

<div class=**"row"**>

<div class=**"****col-sm-8"**>

<h2>**{{object.title}}**</h2>

<p class=**"lead"**>

**{{object.description}}**

</p>

</div>

<div class=**"col-sm-4"**>

<h3>**{{object.price}}**</h3>

**{% if object.variation\_set.count > 1 %}**

<select class=**'form-control'**>

**{% for vari\_obj in object.variation\_set.all %}**

<option value = **"{{vari\_obj.id}}"**>

**{{vari\_obj}}**

</option>

**{% endfor %}**

</select>

**{% endif %}**

<br/>

<hr/>

<h4>**Related Products**</h4>

<a href=**"#"**>**Add to Cart**</a>

</div>

**{% endblock %}**

# 017 Image Uploads

<https://github.com/codingforentrepreneurs/Guides/blob/master/all/imagefield_and_pillow.md>

实现：图片上传功能

pillow是python image库

>**pip install pillow**

创建ProductImage类 (product.models)

其中，存放路径在MEDIA\_ROOT = os.path.join(os.path.dirname(BASE\_DIR), "static\_in\_env", "media\_root")

子目录可以自定义

这儿有2个文件iphone\_cover.jpg，mp3\_player.jpg分别传给Product里面的iPhone Cover和MP3 Player

**from** django**.**utils**.**text **import** slugify

**def** image\_upload\_to**(**instance**,** filename**):**

title **=** instance**.**product**.**title

slug **=** slugify**(**title**)**

basename**,** file\_extension **=** filename**.**split**(**"."**)**

new\_filename **=** "%s-%s.%s" **%(**slug**,** instance**.**id**,** file\_extension**)**

**return** "products/%s/%s" **%(**slug**,** new\_filename**)**

比如上传的文件名名为iphone\_cover.jpg，为iPhone Cover model添加ProductImage

则title, slug, basename, file\_extension, new\_filename的值分别如下：

iPhone Cover iphone-cover iphone\_cover jpg iphone-cover-2.jpg

如果是第一次创建ProductImage，instance**.**id为None

MP3 Player mp3-player mp3\_player jpg mp3-player-None.jpg

MP3 Player mp3\_player.jpg

同样的名字，如果做第二次修改

MP3 Player mp3-player mp3\_player jpg mp3-player-3.jpg

MP3 Player mp3\_player.jpg

**Currently:**[products/mp3-player/mp3-player-3.jpg](http://127.0.0.1:8001/media/products/mp3-player/mp3-player-3_7KveE47.jpg)

同样的名字，如果继续覆盖，文件不会被覆盖，而是增加随机数重新拷贝一个

MP3 Player mp3-player mp3\_player jpg mp3-player-3.jpg

MP3 Player mp3\_player.jpg

**Currently:**[products/mp3-player/mp3-player-3\_7KveE47.jpg](http://127.0.0.1:8001/media/products/mp3-player/mp3-player-3_7KveE47.jpg)

入参instance和filename分别为

iPhone Cover iphone\_cover.jpg

filename为上传文件的文件名

**class** **ProductImage(**models**.**Model**):**

product **=** models**.**ForeignKey**(**Product**)**

image **=** models**.**ImageField**(**upload\_to**=**image\_upload\_to**)**

**def** \_\_unicode\_\_**(**self**):**

**return** self**.**product**.**title

>python manage.py makemigrations

>python manage.py migrate

创建admin接口

**from** **.**models **import** Product**,**Variation**,**ProductImage

admin**.**site**.**register**(**ProductImage**)**

在product\_detail\_view添加图片显示

下面这两个的显示分别如下

{{ img.image.file }}

{{ img.image.url }}

D:\virtualenv\ecommerce-ws\src\static\_in\_env\media\_root\products\mp3-player\mp3-player-None.jpg

/media/products/mp3-player/mp3-player-None.jpg

MEDIA\_URL = '/media/'

MEDIA\_ROOT = os.path.join(os.path.dirname(BASE\_DIR), "static\_in\_env", "media\_root")

# 018 Search Query

实现：这一节实现搜索功能

如果想要搜索的话，一般我们会用

127.0.0.1:8000/products/?q=

distinct符合两个q的object不会重复显示

products/views.py

**from** django**.**db**.**models **import** Q

**class** **ProductListView(**FilterMixin**,** ListView**):**

model **=** Product

queryset **=** Product**.**objects**.**all**()**

**def** get\_context\_data**(**self**,** **\***args**,** **\*\***kwargs**):**

context **=** super**(**ProductListView**,** self**).**get\_context\_data**(\***args**,** **\*\***kwargs**)**

context**[**"now"**]** **=** timezone**.**now**()**

context**[**"query"**]** **=** self**.**request**.**GET**.**get**(**"q"**)** #None

**return** context

**def** get\_queryset**(**self**,** **\***args**,** **\*\***kwargs**):**

qs **=** super**(**ProductListView**,** self**).**get\_queryset**(\***args**,** **\*\***kwargs**)**

query **=** self**.**request**.**GET**.**get**(**"q"**)**

**if** query**:**

qs **=** self**.**model**.**objects**.**filter**(**

Q**(**title\_\_icontains**=**query**)** **|**

Q**(**description\_\_icontains**=**query**)**

**)**

**try:**

qs2 **=** self**.**model**.**objects**.**filter**(**

Q**(**price**=**query**)**

**)**

qs **=** **(**qs **|** qs2**).**distinct**()**

**except:**

**pass**

**return** qs

添加navbar

修改navbar.html来增加搜索

注释掉about和contact不再使用

去掉默认的submit按钮，功能跟回车一样

添加method为get，action指向product url

<!-- <li><a href="{% url 'about' %}">About</a></li>

<li><a href="{% url 'contact' %}">Contact</a></li> -->

<form class=**"navbar-form navbar-left"** method=**"GET"** role=**"search"** action=**'{% url "products" %}'**>

<div class=**"form-group"**>

<input type=**"text"** class=**"form-control"** placeholder=**"Search"** name=**"q"'**>

</div>

<!-- <button type="submit" class="btn btn-default">Submit</button> -->

</form>

# 019 Formset for Inventory

实现：库存视图

* 定义Variation库存ModelForm
* 定义Variation Model
* 定义VariationListView视图（从ListView继承），使用到Formset
* 添加variation入口

<https://docs.djangoproject.com/en/1.8/topics/forms/modelforms/#modelform-factory-function>

127.0.0.1:8000/products/1/inventory/

添加forms.py

ViarationInventoryForm

**from** django **import** forms

**from** django**.**forms**.**models **import** modelformset\_factory

**from** **.**models **import** Variation

**class** **VariationInventoryForm(**forms**.**ModelForm**):**

**class** **Meta:**

model **=** Variation

fields **=** **[**

"price"**,**

"sale\_price"**,**

"inventory"**,**

"active"**,**

**]**

VariationInventoryFormSet **=** modelformset\_factory**(**Variation**,** form**=**VariationInventoryForm**,** extra**=**0**)**

form.instance是自带的参数

**class** **BaseModelForm(**BaseForm**):**

**def** \_\_init\_\_**(**self**,** data**=None,** files**=None,** auto\_id**=**'id\_%s'**,** prefix**=None,**

initial**=None,** error\_class**=**ErrorList**,** label\_suffix**=None,**

empty\_permitted**=False,** instance**=None):**

在model中增加variation定义

在views.py增加VariationList object

**from** django**.**contrib **import** messages

**from** django**.**shortcuts **import** render**,** get\_object\_or\_404**,** redirect

**class** **VariationListView(**ListView**):**

model **=** Variation

queryset **=** Variation**.**objects**.**all**()**

**def** get\_context\_data**(**self**,** **\***args**,** **\*\***kwargs**):**

context **=** super**(**VariationListView**,** self**).**get\_context\_data**(\***args**,** **\*\***kwargs**)**

context**[**"formset"**]** **=** VariationInventoryFormSet**(**queryset**=**self**.**get\_queryset**()) 增加formset context**

**return** context

**def** get\_queryset**(**self**,** **\***args**,** **\*\***kwargs**):**

product\_pk **=** self**.**kwargs**.**get**(**"pk"**)**

**if** product\_pk**:**

product **=** get\_object\_or\_404**(**Product**,** pk**=**product\_pk**)**

queryset **=** Variation**.**objects**.**filter**(**product**=**product**) 过滤variation为对应与该product的**

**return** queryset

**def** post**(**self**,** request**,** **\***args**,** **\*\***kwargs**):**

formset **=** VariationInventoryFormSet**(**request**.**POST**,** request**.**FILES**)**

**if** formset**.**is\_valid**():**

formset**.**save**(**commit**=False)**

**for** form **in** formset**:**

new\_item **=** form**.**save**(**commit**=False)**

#if new\_item.title:

product\_pk **=** self**.**kwargs**.**get**(**"pk"**)**

product **=** get\_object\_or\_404**(**Product**,** pk**=**product\_pk**)**

new\_item**.**product **=** product

new\_item**.**save**()**

messages**.**success**(**request**,** "Your inventory and pricing has been updated."**) 抛出消息**

**return** redirect**(**"products"**)**

**raise** Http404

增加variation url

url**(**r'^(?P<pk>\d+)/inventory/$'**,** VariationListView**.**as\_view**(),** name**=**'product\_inventory'**),**

添加products/templates/products/variation\_list.html

#template\_name = "<appname>/<modelname>\_list.html"

**{% extends "base.html" %}**

**{% block content %}**

<table class=**'table'**>

**{% for object in object\_list %}**

<tr>

<td><a href=**'{{ object.get\_absolute\_url }}'**>**{{ object.title }}**</a></td><td>**{{ object.inventory }}**</td>

</tr>

**{% endfor %}**

</table>

<form method=**"POST"** action=**""**> **{% csrf\_token %}**

**{{ formset.management\_form }}**

**{% for form in formset %}**

**{{ form.instance.product.title }}**

**{{ form.instance.title }}**

**{{ form.as\_p }}**

**{% endfor %}**

<input type=**"submit"** value=**'Update'** class=**'btn'** />

</form>

**{% endblock %}**

另外：可以通过下面方法添加inventory超链接（不在原始文档里）

**class** **Product(**models**.**Model**):**

**def** get\_inventory\_url**(**self**):**

**return** reverse**(**"product\_inventory"**,** kwargs**={**"pk"**:** self**.**pk**})**

<a href=**"{{object.get\_inventory\_url}}"**>**Inventory**</a>

# 020 Login Required Mixins

实现：库存需登录才能访问

创建文件products/mixins.py

**from** django**.**contrib**.**admin**.**views**.**decorators **import** staff\_member\_required

**from** django**.**contrib**.**auth**.**decorators **import** login\_required

**from** django**.**utils**.**decorators **import** method\_decorator

**from** django**.**http **import** Http404

**class** **StaffRequired****Mixin(**object**):**

@classmethod

**def** as\_view**(**self**,** **\***args**,** **\*\***kwargs**):**

view **=** super**(**StaffRequiredMixin**,** self**).**as\_view**(\***args**,** **\*\***kwargs**)**

**return** login\_required**(**view**)**

@method\_decorator**(**login\_required**)**

**def** dispatch**(**self**,** request**,** **\***args**,** **\*\***kwargs**):**

**if** request**.**user**.**is\_staff**:**

**return** super**(**StaffRequiredMixin**,** self**).**dispatch**(**request**,** **\***args**,** **\*\***kwargs**)**

**else:**

**raise** Http404

**class** **LoginRequiredMixin(**object**):**

@classmethod

**def** as\_view**(**self**,** **\***args**,** **\*\***kwargs**):**

view **=** super**(**LoginRequiredMixin**,** self**).**as\_view**(\***args**,** **\*\***kwargs**)**

**return** login\_required**(**view**)**

@method\_decorator**(**login\_required**)**

**def** dispatch**(**self**,** request**,** **\***args**,** **\*\***kwargs**):**

**return** super**(**LoginRequiredMixin**,** self**).**dispatch**(**request**,** **\***args**,** **\*\***kwargs**)**

更新views.py

**from** **.**mixins **import** StaffRequiredMixin

**class** **VariationListView(**StaffRequiredMixin**,** ListView**):**

# 021 Django Message

<https://docs.djangoproject.com/en/1.8/ref/contrib/messages/#using-messages-in-views-and-templates>

**Displaying messages**[¶](https://docs.djangoproject.com/en/1.8/ref/contrib/messages/#displaying-messages)

**get\_messages(request)**[**[source]**](https://docs.djangoproject.com/en/1.8/_modules/django/contrib/messages/api/#get_messages)[**¶**](https://docs.djangoproject.com/en/1.8/ref/contrib/messages/#django.contrib.messages.get_messages)

**In your template**, use something like:

{% **if** messages %}

<ul class="messages">

{% **for** message **in** messages %}

<li{% **if** message.tags %} class="{{ message.tags }}"{% endif %}>{{ message }}</li>

{% endfor %}

</ul>

{% endif %}

将这一段拷贝到 base.html

通过<http://getbootstrap.com/components/#alerts>

修饰

**{% if messages %}**

<div class=**'container'**>

<div class=**'alert alert-success text-center alert-dismissible'**>

<button type=**"button"** class=**"close"** data-dismiss=**"alert"** aria-label=**"Close"**><span aria-hidden=**"true"**>&times;</span></button>

**{% for message in messages %}**

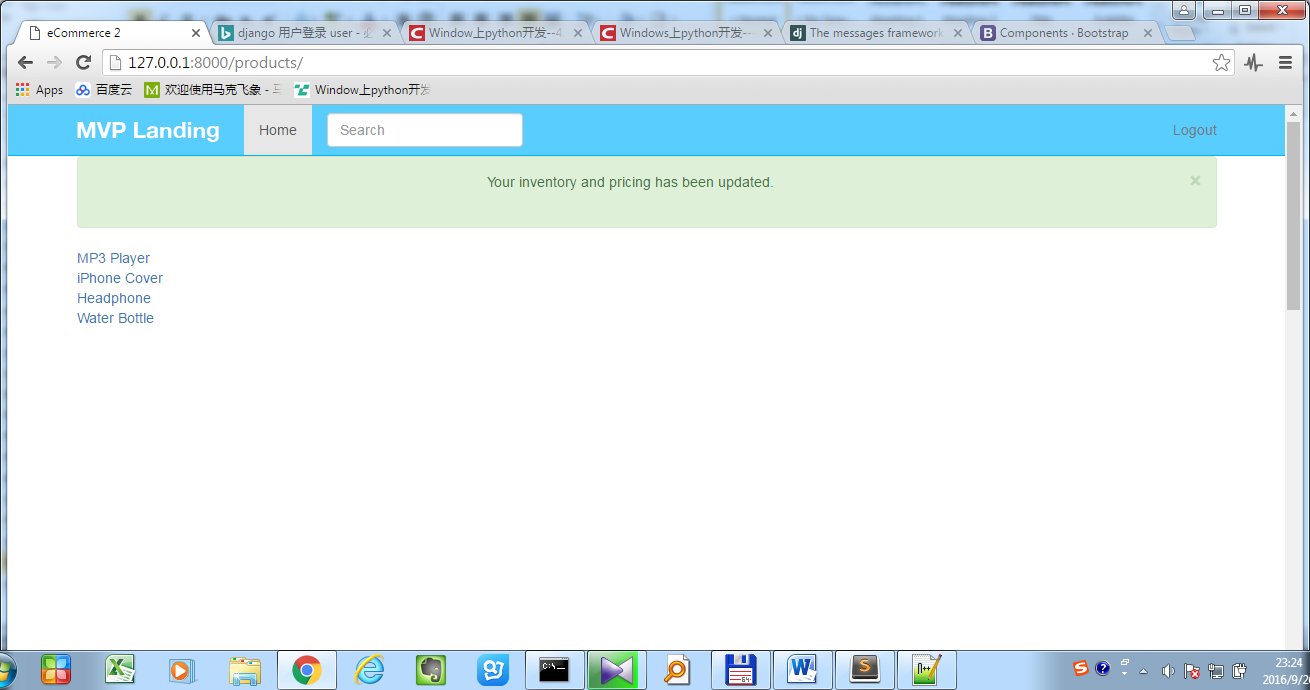
<p{% if message.tags %} class=**"{{ message.tags }}"**{% endif %}>**{{ message }}**</p><br/>

**{% endfor %}**

</div>

</div>

**{% endif %}**



# 022 Social Share

<https://github.com/codingforentrepreneurs/Guides/blob/master/all/social_share_links.md>

<p>

Share<br/>

<a href="https://www.facebook.com/sharer/sharer.php?u={{ request.build\_absolute\_uri }}">

<i class="fa fa-facebook-square fa-3x"></i></a>

<a href="#">

<i class="fa fa-twitter-square fa-3x"></i></a>

</p>

用法参考lwc

# 023 Dynamic Update Price with jQuery

包含图片和文字的两种更新方式

文字: text或者html

图片:

在base.html

<script>

$**(**document**).**ready**(function(){**

**{%** block jquery **%}**

**{%** endblock **%}**

**});**

</script>

Product\_detail.html

<script>

**{%** block jquery **%}**

**function** setPrice**(){**

**var** price **=** $**(**".variation\_select option:selected"**).**attr**(**"data-price"**)**

**var** sale\_price **=** $**(**".variation\_select option:selected"**).**attr**(**"data-sale-price"**)**

**if** **(**sale\_price **!=** "" **&&** sale\_price **!=** "None" **&&** sale\_price **!=** null **)** **{**

$**(**"#price"**).**html**(**"<h3>" **+** sale\_price **+** " <small style='color:red;text-decoration:line-through;'>Original Price:" **+** price **+** "</small></h3>"**);**

**}** **else** **{**

$**(**"#price"**).**html**(**price**);**

**}**

**}**

setPrice**()**

$**(**".variation\_select"**).**change**(function(){**

setPrice**();**

// var img = $(".variation\_select option:selected").attr("data-img")

// $("img").attr("src", img);

**})**

**{%** endblock **%}**

</script>

<div class=**"col-sm-8"**>

<h2>**{{object.title}}**</h2>

**{% if object.productimage\_set.count > 0 %}**

<div>

**{% for img in object.productimage\_set.all %}**

<!-- {{ img.image.file }}

{{ img.image.url }} -->

<img id=**'img'** class= **'img-responsive'** src=**"{{img.image.url}}"**/>

**{% endfor %}**

</div>

**{% endif %}**

<p class=**"lead"**>

**{{object.description}}**

</p>

</div>

<div class=**"col-sm-4"**>

<h3 id=**'price'**>**{{object.price}}**</h3>

**{% if object.variation\_set.count > 1 %}**

<select class=**'form-control variation\_select'**>

**{% for vari\_obj in object.variation\_set.all %}**

<!-- <option data-img="http://www.spirit1059.com/pics/Feeds/Articles/2015611/118317/Beach.jpg" data-price="{{ vari\_obj.price }}" value="{{ vari\_obj.id }}">{{ vari\_obj }}</option> -->

<option data-sale-price=**"{{ vari\_obj.sale\_price }}"** data-price=**"{{ vari\_obj.price }}"** value=**"{{ vari\_obj.id }}"**>**{{ vari\_obj }}**</option>

**{% endfor %}**

</select>

**{% endif %}**

将price的style移到custom.css

style='color:red;text-decoration:line-through;'

**.**og-price **{**

**color:red;**

**text-decoration:line-through;**

**}**

# 024 Single Variation Price

实现：只有一个型号时的价格显示

**{% if object.variation\_set.count > 1 %}**

<h3 id=**'price'**>**{{ object.variation\_set.first.price }}**</h3>

<select class=**'form-control variation\_select'**>

**{% for vari\_obj in object.variation\_set.all %}**

<option data-sale-price=**"{{ vari\_obj.sale\_price }}"** data-price=**"{{ vari\_obj.price }}"** value=**"{{ vari\_obj.id }}"**>**{{ vari\_obj }}**</option>

**{% endfor %}**

</select>

**{% else %}**

<input type=**"hidden"** name=**'item'** value=**'{{ object.variation\_set.first.id }}'** />

<h3 id=**'price'**>**{% if object.variation\_set.first.sale\_price %}**

**{{ object.variation\_set.first.sale\_price }}**

<small class=**'og-price'**>**{{ object.variation\_set.first.price }}**</small>

**{% else %}**

**{{ object.variation\_set.first.price }}**

**{% endif %}**

</h3>

**{% endif %}**

# 025 Product Categories

**实现： 产品类**

定义产品类型，其中包含

* title
* Slug
* description
* active
* timestamp

在Product Model里面添加元素categories和default，由于Category是在Product后面定义，所以如果要引用该类，需用引号括起来’Category’

Product和Category是多对多的关系，但是每个Product有唯一的默认类

**class** **Product(**models**.**Model**):**

title **=** models**.**CharField**(**max\_length**=**120**)**

description **=** models**.**TextField**(**blank**=True,** null**=True)**

price **=** models**.**DecimalField**(**decimal\_places**=**2**,** max\_digits**=**20**)**

active **=** models**.**BooleanField**(**default**=True)**

categories **=** models**.**ManyToManyField**(**'Category'**,** blank**=True)**

default **=** models**.**ForeignKey**(**'Category'**,** related\_name**=**'default\_category'**,** null**=True,** blank**=True)**

class Category(models.Model):

title = models.CharField(max\_length=120, unique=True)

slug = models.SlugField(unique=True)

description = models.TextField(null=True, blank=True)

active = models.BooleanField(default=True)

timestamp = models.DateTimeField(auto\_now\_add=True, auto\_now=False)

def \_\_unicode\_\_(self):

return self.title

在Admin中注册Category

**from** **.**models **import** Product**,** Variation**,** ProductImage**,** Category

admin**.**site**.**register**(**Category**)**

# 026 Category Detail View

**实现**：

- 添加特有的url, url\_categories.py，并在url文件里添加该入口

- 在views添加CategoryListView

- 添加template （包括定义路径和添加模板文件）ListView的默认名字是template\_name = "<appname>/<modelname>\_list.html", 这儿修改为”products/product\_list.html”

- 在views添加CategoryDetailView, 同时增加url入口

- 添加category\_detail.html

urls.py

url**(**r'^categories/'**,** include**(**'products.urls\_categories'**)),**

products/urls\_categories.py

**from** django**.**conf **import** settings

**from** django**.**conf**.**urls **import** include**,** url

**from** django**.**conf**.**urls**.**static **import** static

**from** django**.**contrib **import** admin

**from** **.**views **import** CategoryListView**,** CategoryDetailView

urlpatterns **=** **[**

# Examples:

# url(r'^$', 'newsletter.views.home', name='home'),

url**(**r'^$'**,** CategoryListView**.**as\_view**(),** name**=**'categories'**),**

url**(**r'^(?P<slug>[\w-]+)/$'**,** CategoryDetailView**.**as\_view**(),** name**=**'category\_detail'**),**

#url(r'^(?P<id>\d+)', 'products.views.product\_detail\_view\_func', name='product\_detail\_function'),

slug的设置可参考

<https://github.com/codingforentrepreneurs/Guides/blob/master/all/common_url_regex.md>#slugs

products/views.py

**from** **.**models **import** Product**,** Variation**,** Category

class CategoryListView(ListView):

model = Category

queryset = Category.objects.all()

template\_name = "products/product\_list.html"

class CategoryDetailView(DetailView):

model = Category

def get\_context\_data(self, \*args, \*\*kwargs):

context = super(CategoryDetailView, self).get\_context\_data(\*args, \*\*kwargs)

obj = self.get\_object()

product\_set = obj.product\_set.all()

default\_products = obj.default\_category.all()

products = ( product\_set | default\_products ).distinct()

context["products"] = products

return context

products/models.py

class Category(models.Model):

title = models.CharField(max\_length=120, unique=True)

slug = models.SlugField(unique=True)

description = models.TextField(null=True, blank=True)

active = models.BooleanField(default=True)

timestamp = models.DateTimeField(auto\_now\_add=True, auto\_now=False)

def \_\_unicode\_\_(self):

return self.title

def get\_absolute\_url(self):

return reverse("category\_detail", kwargs={"slug": self.slug })

products/templates/products/category\_detail.html

{% extends "base.html" %}

{% block content %}

<h3>{{ object }}</h3>

<div class = 'table'>

{% for product in object.product\_set.all %}

<tr>

<td>

<a href='{{ product.get\_absolute\_url }}'>{{product.title}}</a>

</td>

</tr>

{% endfor %}

</div>

{% endblock %}

# 027 Related Products

实现：显示相关产品

- 修改ProductManager，增加related相关的信息

- 需改ProductDetailView，覆盖

models.py

class Product(models.Model):

class Meta:

ordering = ["-title"]

class ProductManager(models.Manager):

def get\_related(self, instance):

products\_one = self.get\_queryset().filter(categories\_\_in=instance.categories.all())

products\_two = self.get\_queryset().filter(default=instance.default)

qs = (products\_one | products\_two).exclude(id=instance.id).distinct()

return qs

views.py

class ProductDetailView(DetailView):

model = Product

#template\_name = "product.html"

#template\_name = "<appname>/<modelname>\_detail.html"

def get\_context\_data(self, \*args, \*\*kwargs):

context = super(ProductDetailView, self).get\_context\_data(\*args, \*\*kwargs)

instance = self.get\_object()

context["related"] = Product.objects.get\_related(instance).order\_by("?")[:6]

return context

products/template/product\_detail.html

<h4>Related Products</h4>

{% for product in related %}

<a href='{{ product.get\_absolute\_url }}'>{{product.title}}</a>

{% endfor %}

# 028 Distinct Radom QuerySets

- 修改view的context

- 修改template显示，增加related图形

views.py

**import** random

**class** **ProductDetailView(**DetailView**):**

model **=** Product

#template\_name = "product.html"

#template\_name = "<appname>/<modelname>\_detail.html"

**def** get\_context\_data**(**self**,** **\***args**,** **\*\***kwargs**):**

context **=** super**(**ProductDetailView**,** self**).**get\_context\_data**(\***args**,** **\*\***kwargs**)**

instance **=** self**.**get\_object**()**

#order\_by("-title")

context**[**"related"**]** **=** sorted**(**Product**.**objects**.**get\_related**(**instance**)[:**6**],** key**=** **lambda** x**:** random**.**random**())**

**return** context

在models.py

**class** **Product(**models**.**Model**):**

**def** get\_image\_url**(**self**):**

img **=** self**.**productimage\_set**.**first**() # return obj or None**

**if** img**:**

**return** img**.**image**.**url

**return** img #None

product\_detail.html

修改product image显示方法

<!-- {% if object.productimage\_set.count > 0 %}

<div>

{% for img in object.productimage\_set.all %}

<img id='img' class= 'img-responsive' src="{{img.image.url}}"/>

{% endfor %}

</div>

{% endif %} -->

**{% if object.get\_image\_url %}**

<div>

<img id=**'img'** class= **'img-responsive'** src=**"{{object.****get\_image\_url}}"**/>

</div>

**{% endif %}**

<h4>Related Products</h4>

{% for product in related %}

<!-- <a href='{{ product.get\_absolute\_url }}'>{{product.title}}</a> -->

{% if object.get\_image\_url %}

<img id='img' class= 'img-responsive' src="{{object.get\_image\_url}}"/><br>

{% else %}

<a href='{{ product.get\_absolute\_url }}'>{{product.title}}</a>

{% endif %}

{% endfor %}

# 029 Improve UI

实现：UI美化

<http://getbootstrap.com/components/#thumbnails>

用thumbnail美化product\_detail.html

<hr/>

<h4>**Related Products**</h4>

<div class=**'row'**>

**{% for product in related %}**

<div class=**'col-xs-6'**>

<div class=**'thumbnail text-center'**>

<h4><a href=**'{{ product.get\_absolute\_url }}'**>**{{product.title}}**</a></h4>

**{% if product.get\_image\_url %}**

<a href=**'{{ product.get\_absolute\_url }}'**><img id=**'img'** class= **'img-responsive'** src=**"{{product.get\_image\_url}}"**/></a><br>

**{% endif %}**

</div>

</div>

**{% cycle '' '**</div><div class=**"row"**>**'%}**

**{% endfor %}**

</div>

美化product\_list.html

<h1>All Products <small><a href="{% url 'categories' %}">Categories</a></small></h1>

<div class=**'row'**>

<!-- <table> -->

**{% for product in object\_list %}**

<div class=**'col-xs-4'**>

<div class=**'thumbnail text-center'**>

<h4><a href=**'{{ product.get\_absolute\_url }}'**>**{{product.title}}**</a></h4>

**{% if product.get\_image\_url %}**

<a href=**'{{ product.get\_absolute\_url }}'**><img id=**'img'** class= **'img-responsive'** src=**"{{product.get\_image\_url}}"**/></a><br>

**{% endif %}**

</div>

</div>

**{% cycle '' '' '**</div><div class=**"row"**>**'%}**

<!-- <tr>

<td><a href="{{ object.get\_absolute\_url }}">{{object.title}}</a></td>

</tr> -->

**{% endfor %}**

<!-- </table>-->

</div>

美化category\_detail.html

<div class = **'row'**>

**{% for product in products %}**

<div class=**'col-xs-4'**>

<div class=**'thumbnail text-center'**>

<h4><a href=**'{{ product.get\_absolute\_url }}'**>**{{product.title}}**</a></h4>

**{% if product.get\_image\_url %}**

<a href=**'{{ product.get\_absolute\_url }}'**><img id=**'img'** class= **'img-responsive'** src=**"{{product.get\_image\_url}}"**/></a><br>

**{% endif %}**

</div>

</div>

**{% cycle '' '' '**</div><div class=**"row"**>**'%}**

**{% endfor %}**

</div>

# 030 Django Template Include with Variable

实现：将Thumbnail功能移到专门的问题, include+with

- 将thumbnail功能移到专门的文件

- include thumbnail并附带相应的参数

修改models.py

**from** django**.**utils**.**safestring **import** mark\_safe

class Variation(models.Model):

def get\_html\_price(self):

if self.sale\_price is not None:

html\_text = "<span class='sale-price'>%s</span> <span class='og-price'>%s</span>" %(self.sale\_price, self.price)

else:

html\_text = "<span class='price'>%s</span>" %(self.price)

return mark\_safe(html\_text)

添加product\_thumbnail.html

<div class='thumbnail text-center'>

<h4><a href='{{ product.get\_absolute\_url }}'>{{product.title}}</a></h4>

{% if product.get\_image\_url %}

<a href='{{ product.get\_absolute\_url }}'><img id='img' class= 'img-responsive' src="{{product.get\_image\_url}}"/></a><br>

{% endif %}

{% if price == "True" %}

{{ product.variation\_set.first.get\_html\_price }}

{% endif %}

</div>

product\_list.html

<div class=**'row'**>

**{% for product in object\_list %}**

<div class=**'col-xs-4'**>

**{% include 'products/product\_thumbnail.html' with product=product price='True' %}**

</div>

**{% cycle '' '' '**</div><div class=**"row"**>**'%}**

**{% endfor %}**

</div>

product\_detail.html

<h4>Related Products</h4>

<div class='row'>

{% for product in related %}

<div class='col-xs-6'>

{% include "products/product\_thumbnail.html" with product=product price="True" %}

</div>

{% cycle '' '</div><div class="row">'%}

{% endfor %}

</div>

category\_detail.html

<div class = **'row'**>

**{% for product in products %}**

<div class=**'col-xs-4'**>

**{% include "products/product\_thumbnail.html" with product=product price="True" %}**

</div>

**{% cycle '' '' '**</div><div class=**"row"**>**'%}**

**{% endfor %}**

</div>

# 031 Featured Product on Homepage

实现功能

- 在首页增加产品显示

- jumbotron改为产品展示

- 增加FeatureProduct类

# modified: newsletter/views.py

# modified: products/admin.py

# modified: products/models.py

# modified: templates/home.html

newsletter/views.py

**from** products**.**models **import** ProductFeatured

**def** home**(**request**):**

title **=** 'Sign Up now'

featured\_image **=** ProductFeatured**.**objects**.**filter**(**active**=True).**order\_by**(**"?"**).**first**()**

form **=** SignUpForm**(**request**.**POST **or** **None)**

context **=** **{**

"title"**:** title**,**

"form"**:** form**,**

"featured\_image"**:**featured\_image**,**

**}**

products/admin.py

**from** **.**models **import** Product**,**Variation**,**ProductImage**,**Category**,**ProductFeatured

admin**.**site**.**register**(**ProductFeatured**)**

products/models.py

**def** image\_upload\_to\_featured**(**instance**,** filename**):**

title **=** instance**.**product**.**title

slug **=** slugify**(**title**)**

basename**,** file\_extension **=** filename**.**split**(**"."**)**

new\_filename **=** "%s-%s.%s" **%(**slug**,** instance**.**id**,** file\_extension**)**

**return** "products/%s/featured/%s" **%(**slug**,** new\_filename**)**

**class** **ProductFeatured(**models**.**Model**):**

product **=** models**.**ForeignKey**(**Product**)**

image **=** models**.**ImageField**(**upload\_to**=**image\_upload\_to\_featured**)**

title **=** models**.**CharField**(**max\_length**=**120**,** null**=True,** blank**=True)**

text **=** models**.**CharField**(**max\_length**=**220**,** null**=True,** blank**=True)**

text\_right **=** models**.**BooleanField**(**default**=False)**

text\_css\_color **=** models**.**CharField**(**max\_length**=**6**,** null**=True,** blank**=True)**

show\_price **=** models**.**BooleanField**(**default**=False)**

make\_image\_background **=** models**.**BooleanField**(**default**=False)**

active **=** models**.**BooleanField**(**default**=True)**

**def** \_\_unicode\_\_**(**self**):**

**return** self**.**product**.**title

templates/home.html

{% block jumbotron %}

{% if not request.user.is\_authenticated%}

{% if featured\_image %}

<div class="jumbotron">

<div class="container">

<div class="row">

<div class='{% if featured\_image.make\_image\_background %} col-sm-12 {% else %} col-sm-6 {% endif %} text-center {% if featured\_image.text\_right %} pull-right {% endif %}'>

<h1>{{ featured\_image.product.title }}</h1>

<p>{{ featured\_image.product.description }}</p>

<p>

<a class="btn btn-lg btn-primary" href="{{ featured\_image.product.get\_absolute\_url }}" role="button">More Details &raquo;</a>

</p>

</div>

{% if not featured\_image.make\_image\_background %}

<div class='col-sm-6'>

<img src="{{ featured\_image.image.url }}" class="img-responsive" />

</div>

{% endif %}

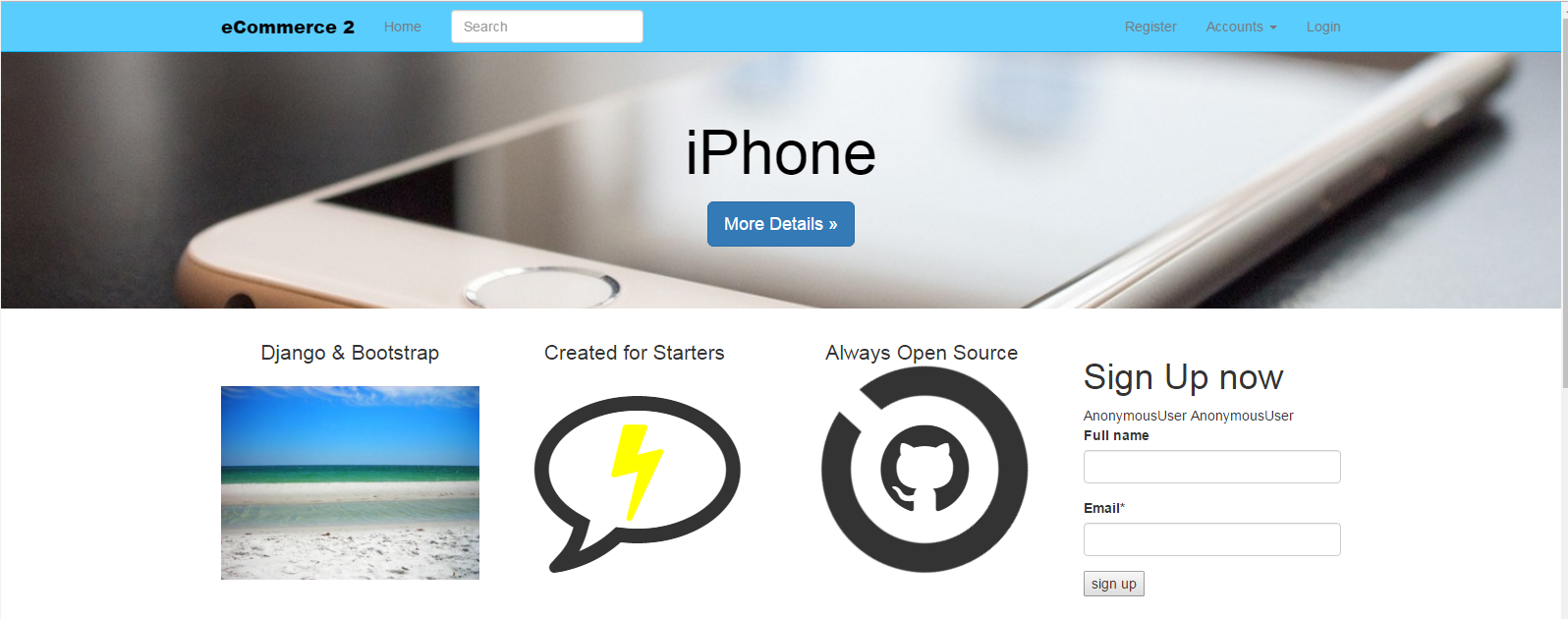
</div>

</div>

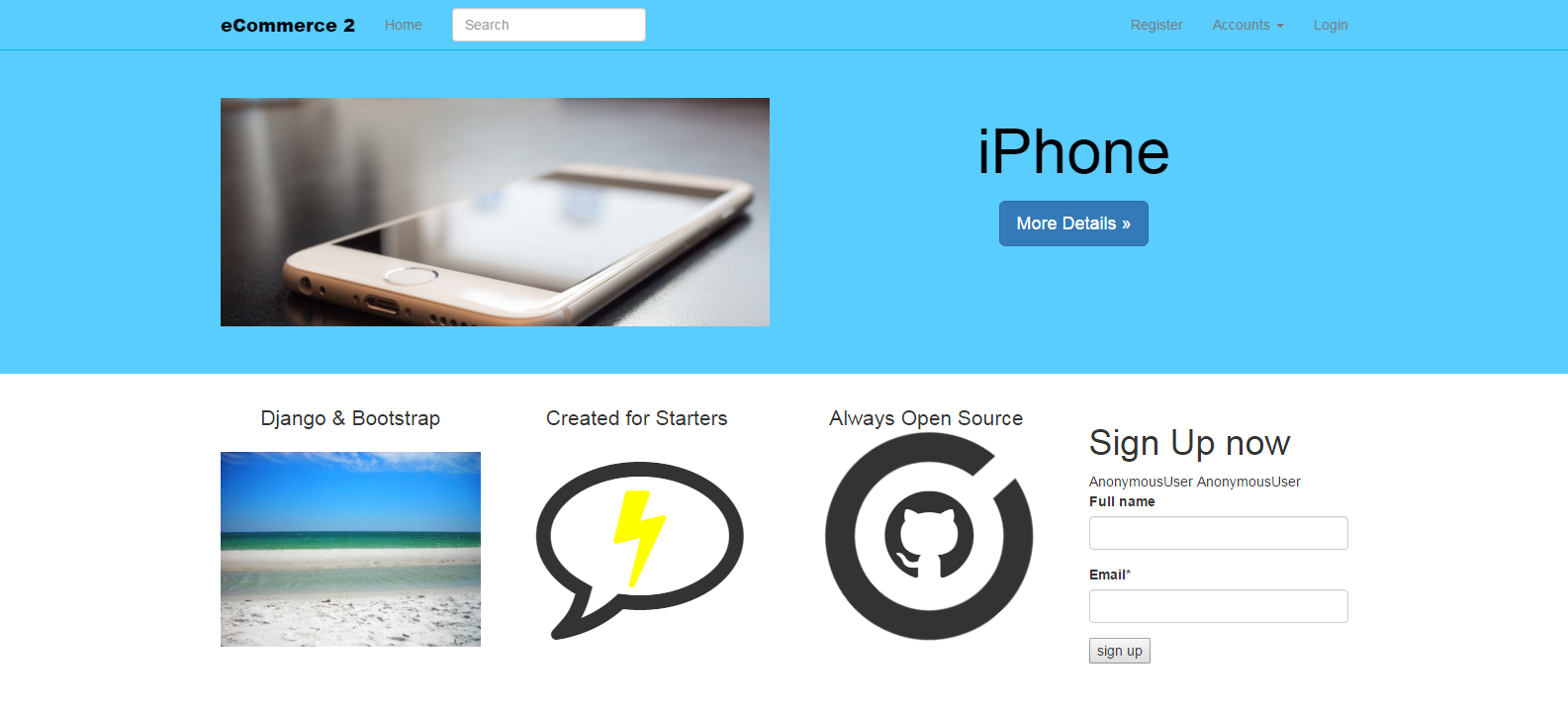
</div>

{% else %}

<div class="jumbotron">



不带背景图的



# 032 Login as Dropdown Menu

修改navbar.html

<ul class=**"nav navbar-nav navbar-right"**>

**{% if request.user.is\_authenticated %}**

<li><a href=**"{% url 'auth\_logout' %}"**>**Logout**</a></li>

**{% else %}**

<li><a href=**"{% url 'registration\_register' %}"**>**Register**</a></li>

**{% if not request.user.is\_authenticated and not "/accounts/login" in request.get\_full\_path %}**

<li class=**"dropdown"**>

<a href=**"#"** class=**"dropdown-toggle"** data-toggle=**"dropdown"** role=**"button"** aria-haspopup=**"true"** aria-expanded=**"false"**>**Accounts** <span class=**"caret"**></span></a>

<ul class=**"dropdown-menu"**>

<form class=**'navbar-form'** method=**'POST'** action=**'{% url "auth\_login" %}'**>**{% csrf\_token %}**

<div class=**'form-group'**>

<input type=**'text'** class=**'form-control'** name=**'username'** placeholder=**'Username'** />

</div>

<div class=**'form-group'**>

<input type=**'password'** class=**'form-control'** name=**'password'** placeholder=**'Password'** />

</div>

<button type=**'submit'** class=**'btn btn-default'**>**Login**</button>

</form>

<p class=**"text-center"**><a href=**"{% url 'auth\_login' %}"**>**Forget Password**</a></p>

</ul>

</li>

**{% endif %}**

<li><a href=**"{% url 'auth\_login' %}"**>**Login**</a></li>

**{% endif %}**

</ul>

# 033 Shopping cart icon

登录界面移到下拉菜单后为cart腾出了位置

从下面网页获取icon

<http://fontawesome.io/icon/shopping-cart/>

# modified: static\_in\_pro/our\_static/css/custom.css

# modified: templates/navbar.html

static\_in\_pro/our\_static/css/custom.css

**.**fa-navbar-cart**{**

**font-size: 18px;**

**vertical-align: middle;**

templates/navbar.html

<ul class="nav navbar-nav navbar-right">

<li><a href="#"><i class="fa fa-shopping-cart fa-navbar-cart"></i> <span class='badge'>0</span></a></li>

{% if request.user.is\_authenticated %}

<li><a href="{% url 'auth\_logout' %}">Logout</a></li>



# 034 Product Lists on Homepage

col\_class\_set OK

col-class-set KO

# modified: newsletter/views.py

# modified: products/templates/products/product\_list.html

# modified: templates/home.html

#

# new : products/templates/products/products.html

newsletter/views.py

**def** home**(**request**):**

title **=** 'Sign Up now'

featured\_image **=** ProductFeatured**.**objects**.**filter**(**active**=True).**order\_by**(**"?"**).**first**()**

products **=** Product**.**objects**.**all**().**order\_by**(**'?'**)[:**6**]**

products2 **=** Product**.**objects**.**all**().**order\_by**(**'?'**)[:**6**]**

form **=** SignUpForm**(**request**.**POST **or** **None)**

context **=** **{**

"title"**:** title**,**

"form"**:** form**,**

"featured\_image"**:**featured\_image**,**

"products"**:**products**,**

"products2"**:**products2**,**

**}**

products/templates/products/product\_list.html

{% extends "base.html" %}

{% block content %}

<h1>All Products <small><a href="{% url 'categories' %}">Categories</a></small></h1>

<div class='row'>

{% include 'products/products.html' with object\_list=object\_list %}

</div>

{% endblock %}

templates/home.html

{% block content %}

<div class="row">

<h3>Recommended Products</h3>

{% include 'products/products.html' with object\_list=products col\_class\_set="col-sm-2" %}

</div>

<div class="row">

<h3>Featured Products</h3>

{% include 'products/products.html' with object\_list=products2 %}

</div>

products/templates/products/products.html

{% for product in object\_list %}

<div class='col-xs-4 {{col\_class\_set}}'>

{% include 'products/product\_thumbnail.html' with product=product price='True' %}

</div>

{% if not col\_class\_set %}

{% cycle '' '' '</div><div class="row">'%}

{% endif %}

{% endfor %}

# 035 Product Editing with django Inline

<https://docs.djangoproject.com/en/1.8/ref/contrib/admin/#inlinemodeladmin-objects>

# modified: products/admin.py

class ProductImageInline(admin.TabularInline):

model = ProductImage

extra = 0

max\_num = 10

class VariationInline(admin.TabularInline):

model = Variation

extra = 0

max\_num = 10

class ProductAdmin(admin.ModelAdmin):

list\_display = ['\_\_unicode\_\_', 'price']

inlines = [

ProductImageInline,

VariationInline,

]

class Meta:

model = Product

admin**.**site**.**register**(**Product**,** ProductAdmin**)**

# 036 Carts App

python manage.py startapp carts

# modified: carts/models.py

# modified: ecommerce2/settings/base.py

carts/models.py

**from** django**.**conf **import** settings

**from** django**.**db **import** models

**from** products**.**models **import** Variation

# Create your models here.

**class** **CartItem(**models**.**Model**):**

item **=** models**.**ForeignKey**(**Variation**)**

quantity **=** models**.**PositiveIntegerField**(**default**=**1**)**

**def** \_\_unicode\_\_**(**self**):**

**return** self**.**item**.**title

**class** **Cart(**models**.**Model**):**

user **=** models**.**ForeignKey**(**settings**.**AUTH\_USER\_MODEL**,** null**=True,** blank**=True)**

items **=** models**.**ManyToManyField**(**CartItem**)**

timestamp **=** models**.**DateTimeField**(**auto\_now\_add**=True,** auto\_now**=False)**

updated **=** models**.**DateTimeField**(**auto\_now\_add**=False,** auto\_now**=True)**

**def** \_\_unicode\_\_**(**self**):**

**return** str**(**self**.**id**)**

ecommerce2/settings/base.py

INSTALLED\_APPS **=** **(**

'django.contrib.admin'**,**

'django.contrib.auth'**,**

'django.contrib.contenttypes'**,**

'django.contrib.sessions'**,**

'django.contrib.messages'**,**

'django.contrib.sites'**,**

'django.contrib.staticfiles'**,**

'newsletter'**,**

'products'**,**

'carts'**,**

'crispy\_forms'**,**

'registration'**,**

**)**

# 037 ManyToMany through

[https://docs.djangoproject.com/en/1.8/topics/db/models/#extra-fields-on-many-to-many-relationships](https://docs.djangoproject.com/en/1.8/topics/db/models/" \l "extra-fields-on-many-to-many-relationships)

**Extra fields on many-to-many relationships**[**¶**](https://docs.djangoproject.com/en/1.8/topics/db/models/#extra-fields-on-many-to-many-relationships)

When you’re only dealing with simple many-to-many relationships such as mixing and matching pizzas and toppings, a standard [**ManyToManyField**](https://docs.djangoproject.com/en/1.8/ref/models/fields/#django.db.models.ManyToManyField) is all you need. However, sometimes you may need to associate data with the relationship between two models.

For example, consider the case of an application tracking the musical groups which musicians belong to. There is a many-to-many relationship between a person and the groups of which they are a member, so you could use a[**ManyToManyField**](https://docs.djangoproject.com/en/1.8/ref/models/fields/#django.db.models.ManyToManyField) to represent this relationship. However, there is a lot of detail about the membership that you might want to collect, such as the date at which the person joined the group.

For these situations, Django allows you to specify the model that will be used to govern the many-to-many relationship. You can then put extra fields on the intermediate model. The intermediate model is associated with the[**ManyToManyField**](https://docs.djangoproject.com/en/1.8/ref/models/fields/#django.db.models.ManyToManyField) using the [**through**](https://docs.djangoproject.com/en/1.8/ref/models/fields/#django.db.models.ManyToManyField.through) argument to point to the model that will act as an intermediary. For our musician example, the code would look something like this:

**from** **django.db** **import** models

**class** **Person**(models.Model):

name = models.CharField(max\_length=128)

**def** \_\_str\_\_(self): *# \_\_unicode\_\_ on Python 2*

**return** self.name

**class** **Group**(models.Model):

name = models.CharField(max\_length=128)

members = models.ManyToManyField(Person, through='Membership')

**def** \_\_str\_\_(self): *# \_\_unicode\_\_ on Python 2*

**return** self.name

**class** **Membership**(models.Model):

person = models.ForeignKey(Person)

group = models.ForeignKey(Group)

date\_joined = models.DateField()

invite\_reason = models.CharField(max\_length=64)

When you set up the intermediary model, you explicitly specify foreign keys to the models that are involved in the many-to-many relationship. This explicit declaration defines how the two models are related.

There are a few restrictions on the intermediate model:

* Your intermediate model must contain one - and only one - foreign key to the source model (this would be **Group** in our example), or you must explicitly specify the foreign keys Django should use for the relationship using[**ManyToManyField.through\_fields**](https://docs.djangoproject.com/en/1.8/ref/models/fields/#django.db.models.ManyToManyField.through_fields). If you have more than one foreign key and **through\_fields** is not specified, a validation error will be raised. A similar restriction applies to the foreign key to the target model (this would be**Person** in our example).
* For a model which has a many-to-many relationship to itself through an intermediary model, two foreign keys to the same model are permitted, but they will be treated as the two (different) sides of the many-to-many relationship. If there are more than two foreign keys though, you must also specify **through\_fields** as above, or a validation error will be raised.
* When defining a many-to-many relationship from a model to itself, using an intermediary model, you must use[**symmetrical=False**](https://docs.djangoproject.com/en/1.8/ref/models/fields/#django.db.models.ManyToManyField.symmetrical) (see [the model field reference](https://docs.djangoproject.com/en/1.8/ref/models/fields/#manytomany-arguments)).

Now that you have set up your [**ManyToManyField**](https://docs.djangoproject.com/en/1.8/ref/models/fields/#django.db.models.ManyToManyField) to use your intermediary model (**Membership**, in this case), you’re ready to start creating some many-to-many relationships. You do this by creating instances of the intermediate model:

**>>>** ringo = Person.objects.create(name="Ringo Starr")

**>>>** paul = Person.objects.create(name="Paul McCartney")

**>>>** beatles = Group.objects.create(name="The Beatles")

**>>>** m1 = Membership(person=ringo, group=beatles,

**...**  date\_joined=date(1962, 8, 16),

**...**  invite\_reason="Needed a new drummer.")

**>>>** m1.save()

**>>>** beatles.members.all()

[<Person: Ringo Starr>]

**>>>** ringo.group\_set.all()

[<Group: The Beatles>]

**>>>** m2 = Membership.objects.create(person=paul, group=beatles,

**...**  date\_joined=date(1960, 8, 1),

**...**  invite\_reason="Wanted to form a band.")

**>>>** beatles.members.all()

[<Person: Ringo Starr>, <Person: Paul McCartney>]

Unlike normal many-to-many fields, you can’t use **add**, **create**, or assignment (i.e., **beatles.members = [...]**) to create relationships:

*# THIS WILL NOT WORK*

>>> beatles.members.add(john)

*# NEITHER WILL THIS*

>>> beatles.members.create(name="George Harrison")

*# AND NEITHER WILL THIS*

>>> beatles.members = [john, paul, ringo, george]

Why? You can’t just create a relationship between a **Person** and a **Group** - you need to specify all the detail for the relationship required by the **Membership** model. The simple **add**, **create** and assignment calls don’t provide a way to specify this extra detail. As a result, they are disabled for many-to-many relationships that use an intermediate model. The only way to create this type of relationship is to create instances of the intermediate model.

The [**remove()**](https://docs.djangoproject.com/en/1.8/ref/models/relations/#django.db.models.fields.related.RelatedManager.remove) method is disabled for similar reasons. However, the [**clear()**](https://docs.djangoproject.com/en/1.8/ref/models/relations/#django.db.models.fields.related.RelatedManager.clear) method can be used to remove all many-to-many relationships for an instance:

**>>>** *# Beatles have broken up*

**>>>** beatles.members.clear()

**>>>** *# Note that this deletes the intermediate model instances*

**>>>** Membership.objects.all()

[]

Once you have established the many-to-many relationships by creating instances of your intermediate model, you can issue queries. Just as with normal many-to-many relationships, you can query using the attributes of the many-to-many-related model:

*# Find all the groups with a member whose name starts with 'Paul'*

>>> Group.objects.filter(members\_\_name\_\_startswith='Paul')

[<Group: The Beatles>]

As you are using an intermediate model, you can also query on its attributes:

*# Find all the members of the Beatles that joined after 1 Jan 1961*

>>> Person.objects.filter(

... group\_\_name='The Beatles',

... membership\_\_date\_joined\_\_gt=date(1961,1,1))

[<Person: Ringo Starr]

If you need to access a membership’s information you may do so by directly querying the **Membership** model:

**>>>** ringos\_membership = Membership.objects.get(group=beatles, person=ringo)

**>>>** ringos\_membership.date\_joined

datetime.date(1962, 8, 16)

**>>>** ringos\_membership.invite\_reason

'Needed a new drummer.'

Another way to access the same information is by querying the [many-to-many reverse relationship](https://docs.djangoproject.com/en/1.8/topics/db/queries/#m2m-reverse-relationships) from a **Person** object:

**>>>** ringos\_membership = ringo.membership\_set.get(group=beatles)

**>>>** ringos\_membership.date\_joined

datetime.date(1962, 8, 16)

**>>>** ringos\_membership.invite\_reason

'Needed a new drummer.'

一直报错，没法解决

self.apply\_migration(states[migration], migration, fake=fake, fake\_initial=f

ake\_initial)

File "D:\virtualenv\ecommerce-ws\lib\site-packages\django\db\migrations\execut

or.py", line 148, in apply\_migration

state = migration.apply(state, schema\_editor)

File "D:\virtualenv\ecommerce-ws\lib\site-packages\django\db\migrations\migrat

ion.py", line 115, in apply

operation.database\_forwards(self.app\_label, schema\_editor, old\_state, projec

t\_state)

File "D:\virtualenv\ecommerce-ws\lib\site-packages\django\db\migrations\operat

ions\fields.py", line 201, in database\_forwards

schema\_editor.alter\_field(from\_model, from\_field, to\_field)

File "D:\virtualenv\ecommerce-ws\lib\site-packages\django\db\backends\base\sch

ema.py", line 480, in alter\_field

"through= on M2M fields)" % (old\_field, new\_field)

ValueError: Cannot alter field carts.Cart.items into carts.Cart.items - they are

not compatible types (you cannot alter to or from M2M fields, or add or remove

through= on M2M fields)

最后暴力解决

删除migrations目录，并且在db里删除carts相对表

# modified: carts/admin.py

# modified: carts/models.py

carts/models.py

**from** products**.**models **import** Variation

# Create your models here.

**class** **CartItem(**models**.**Model**):**

cart **=** models**.**ForeignKey**(**"Cart"**)** #cart class is below

item **=** models**.**ForeignKey**(**Variation**)**

quantity **=** models**.**PositiveIntegerField**(**default**=**1**)**

**def** \_\_unicode\_\_**(**self**):**

**return** self**.**item**.**title

**class** **Cart(**models**.**Model**):**

user **=** models**.**ForeignKey**(**settings**.**AUTH\_USER\_MODEL**,** null**=True,** blank**=True)**

items **=** models**.**ManyToManyField**(**Variation**,** through**=**CartItem**)**

timestamp **=** models**.**DateTimeField**(**auto\_now\_add**=True,** auto\_now**=False)**

updated **=** models**.**DateTimeField**(**auto\_now\_add**=False,** auto\_now**=True)**

**def** \_\_unicode\_\_**(**self**):**

**return** str**(**self**.**id**)**

carts/admin.py

**from** **.**models **import** Cart**,** CartItem

**class** **CartItemInline(**admin**.**TabularInline**):**

model **=** CartItem

**class** **CartAdmin(**admin**.**ModelAdmin**):**

inlines **=** **[**

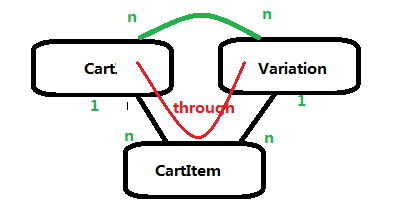
CartItemInline

**]**

**class** **Meta:**

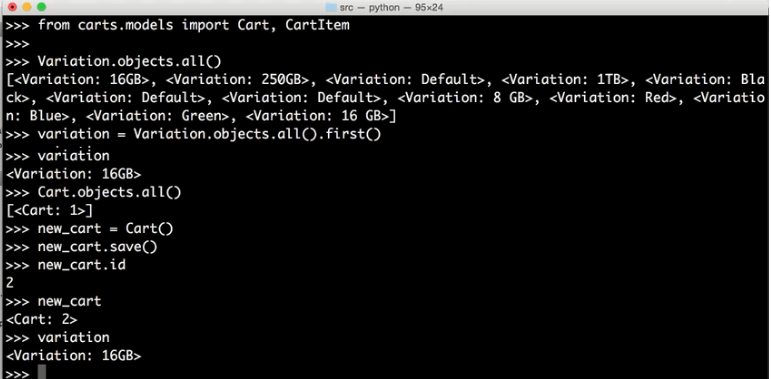
model **=** Cart

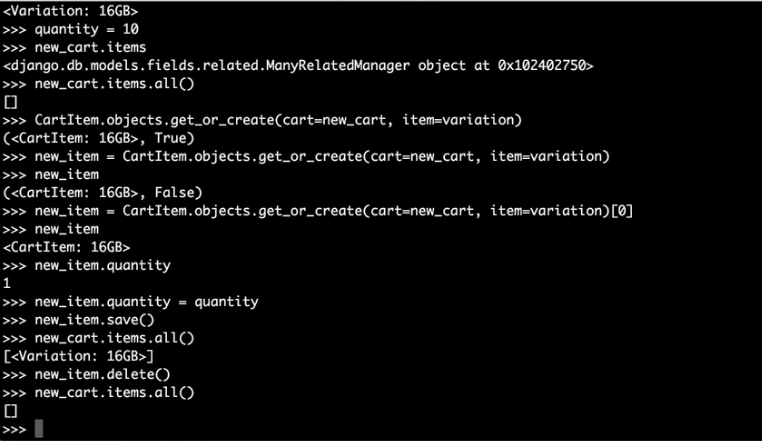
admin**.**site**.**register**(**Cart**,** CartAdmin**)**

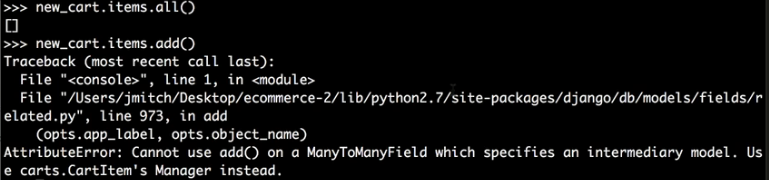


# 038 ManyToMany Through using the Python Shell

python manage.py shell







# 039 Add, Update, Remove Carts in CBV

<https://docs.djangoproject.com/en/1.8/ref/class-based-views/base/#view>

# modified: carts/views.py

# modified: ecommerce2/urls.py

carts/views.py

**from** django**.**views**.**generic**.**base **import** View

**from** django**.**http **import** HttpResponseRedirect

**from** django**.**shortcuts **import** render**,** get\_object\_or\_404

**from** products**.**models **import** Variation

**from** **.**models **import** Cart**,** CartItem

# Create your views here.

**class** **CartView(**View**):**

**def** get**(**self**,** request**,** **\***args**,** **\*\***kwargs**):**

item\_id **=** request**.**GET**.**get**(**'item'**)**

delete\_item **=** request**.**GET**.**get**(**'delete'**)**

**if** item\_id**:**

item\_instance **=** get\_object\_or\_404**(**Variation**,** id**=**item\_id**)**

qty**=**request**.**GET**.**get**(**'qty'**)**

cart **=** Cart**.**objects**.**all**().**first**()**

cart\_item **=** CartItem**.**objects**.**get\_or\_create**(**cart**=**cart**,** item**=**item\_instance**)[**0**]**

**if** delete\_item**:**

cart\_item**.**delete**()**

**else:**

cart\_item**.**quantity **=** qty

cart\_item**.**save**()**

**return** HttpResponseRedirect**(**'/'**)**

ecommerce2/urls.py

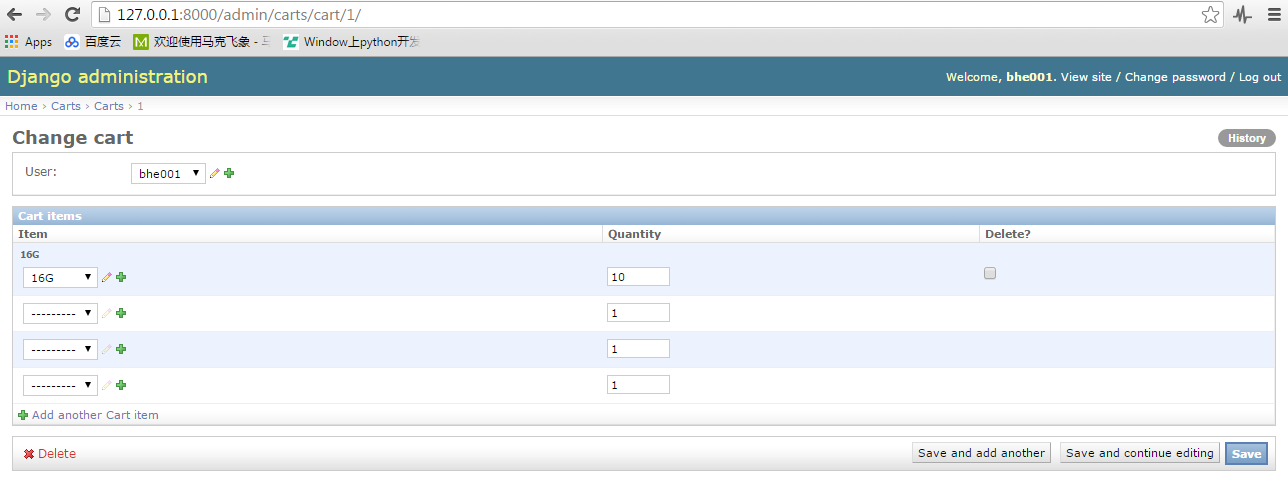
**from** carts**.**views **import** CartView

url**(**r'^cart/$'**,** CartView**.**as\_view**(),** name**=**'cart'**),**

看上去as\_view会触发view.get的调用

<http://127.0.0.1:8000/cart/?item=1&qty=10>

<http://127.0.0.1:8000/cart/?item=1&qty=10&delete=True>



# 040 Django sessions

<https://docs.djangoproject.com/en/1.8/topics/http/sessions/#module-django.contrib.sessions>

class CartView(View):

def get(self, request, \*args, \*\*kwargs):

request.session.set\_expiry(0)

cart\_id = request.session.get("cart\_id")

if cart\_id == None:

cart = Cart()

cart.save()

cart\_id = cart.id

request.session["cart\_id"] = cart\_id

cart = Cart.objects.get(id=cart\_id)

if request.user.is\_authenticated():

cart.user = request.user

cart.save()

item\_id = request.GET.get('item')

delete\_item = request.GET.get('delete')

if item\_id:

item\_instance = get\_object\_or\_404(Variation, id=item\_id)

qty=request.GET.get('qty')

#cart = Cart.objects.all().first()

cart\_item = CartItem.objects.get\_or\_create(cart=cart, item=item\_instance)[0]

if delete\_item:

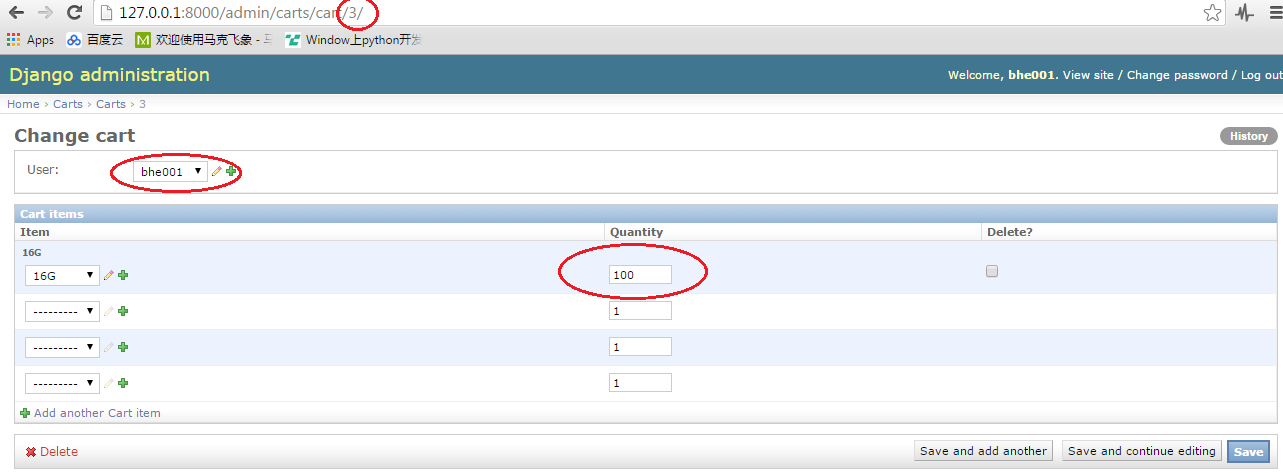
cart\_item.delete()

else:

cart\_item.quantity = qty

cart\_item.save()

return HttpResponseRedirect('/')



<http://127.0.0.1:8000/cart/?item=1&qty=100>

# 041 Render Cart View

<https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin>

# modified: carts/views.py

# added : carts/templates/

carts/views.py

**from** django**.**views**.**generic**.**detail **import** SingleObjectMixin

**class** **CartView(**SingleObjectMixin**,** View**):**

model **=** Cart

template\_name **=** "carts/view.html"

**def** get\_object**(**self**,** **\***args**,** **\*\***kwargs**):**

self**.**request**.**session**.**set\_expiry**(**0**)**

cart\_id **=** self**.**request**.**session**.**get**(**"cart\_id"**)**

**if** cart\_id **==** **None:**

cart **=** Cart**()**

cart**.**save**()**

cart\_id **=** cart**.**id

self**.**request**.**session**[**"cart\_id"**]** **=** cart\_id

cart **=** Cart**.**objects**.**get**(**id**=**cart\_id**)**

**if** self**.**request**.**user**.**is\_authenticated**():**

cart**.**user **=** self**.**request**.**user

cart**.**save**()**

**return** cart

**def** get**(**self**,** request**,** **\***args**,** **\*\***kwargs**):**

cart **=** self**.**get\_object**()**

item\_id **=** request**.**GET**.**get**(**'item'**)**

delete\_item **=** request**.**GET**.**get**(**'delete'**)**

**if** item\_id**:**

item\_instance **=** get\_object\_or\_404**(**Variation**,** id**=**item\_id**)**

qty**=**request**.**GET**.**get**(**'qty'**)**

#cart = Cart.objects.all().first()

cart\_item **=** CartItem**.**objects**.**get\_or\_create**(**cart**=**cart**,** item**=**item\_instance**)[**0**]**

**if** delete\_item**:**

cart\_item**.**delete**()**

**else:**

cart\_item**.**quantity **=** qty

cart\_item**.**save**()**

#context = self.get\_context\_data(request, \*args, \*\*kwargs)

context**={**

"object"**:**self**.**get\_object**()**

**}**

template **=** self**.**template\_name

**return** render**(**request**,** template**,** context**)**

carts/templates/view.html

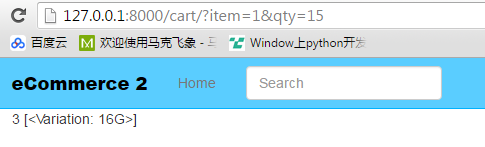
{% extends "base.html" %}

{% block content %}

{{ object }}

{{ object.items.all }}

{% endblock %}



# 042 Add to cart from product view

# modified: carts/views.py

# modified: products/templates/products/product\_detail.html

carts/views.py

class CartView(SingleObjectMixin, View):

delete\_item = request.GET.get('delete')

if item\_id:

item\_instance = get\_object\_or\_404(Variation, id=item\_id)

- qty=request.GET.get('qty')

+ qty=request.GET.get('qty', 1)

+ try:

+ if int(qty) < 1:

+ delete\_item = True

+ except:

+ raise Http404

#cart = Cart.objects.all().first()

products/templates/products/product\_detail.html

<div class="col-sm-4">

+ <form id='add-form' method='GET' action="{% url 'cart' %}">

{% if object.variation\_set.count > 1 %}

<h3 id='price'>{{ object.variation\_set.first.price }}</h3>

- <select class='form-control variation\_select'>

+ <select name='item' class='form-control variation\_select'>

{% for vari\_obj in object.variation\_set.all %}

<option data-sale-price="{{ vari\_obj.sale\_price }}" data-price="{{ vari\_obj.price }}" value="{{ vari\_obj.id }}">{{ vari\_obj }}</option>

{% endfor %}

</select>

{% else %}

+ <input type="hidden" name='item' value='{{ object.variation\_set.first.id }}' />

<h3 id='price'>{% if object.variation\_set.first.sale\_price %}

{{ object.variation\_set.first.sale\_price }}

<small class='og-price'>{{ object.variation\_set.first.price }}</small>

{% else %}

{{ object.variation\_set.first.price }}

{% endif %}

</h3>

{% endif %}

+ <br/>

+ <input class='form-control' type='number' name='qty' value='1' />

<br/>

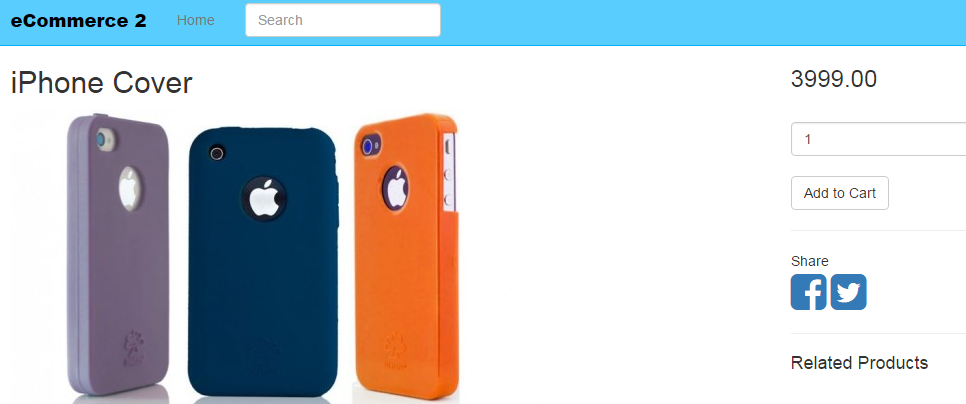
- <a href="#">Add to Cart</a>

+ <input id='submit-btn' type='submit' value='Add to Cart' class='btn btn-default' />

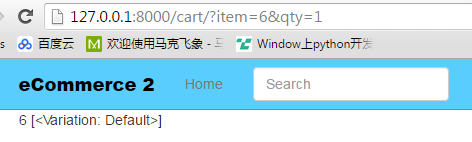
+ </form>

<p>

Share<br/>



执行上面的Add to Cart会跳转到下面地址



# 043 Format Cart Remove Items

# modified: carts/models.py

# modified: carts/templates/carts/view.html

# modified: products/models.py

carts/models.py

class CartItem(models.Model):

def \_\_unicode\_\_(self):

return self.item.title

+ def remove(self):

+ return self.item.remove\_from\_cart()

carts/templates/carts/view.html

{% extends "base.html" %}

{% block content %}

-{{ object }}

-{{ object.items.all }}

+<table class='table'>

+{% for item in object.cartitem\_set.all %}

+ <tr id='item-{{ item.item.id }}'>

+ <td>{{ item.item.get\_title }}</td>

+ <td>{{ item.quantity }}</td>

+ <td>{{ item.line\_total }}</td>

+ <td class='text-right' ><a href='{{ item.remove }}'>X</a>

+ </tr>

+{% endfor %}

+</table>

{% endblock %}

products/models.py

class Variation(models.Model):

def get\_absolute\_url(self):

return self.product.get\_absolute\_url()

+ def add\_to\_cart(self):

+ return "%s?item=%s&qty=1" %(reverse("cart"), self.id) #here cart is url name

+

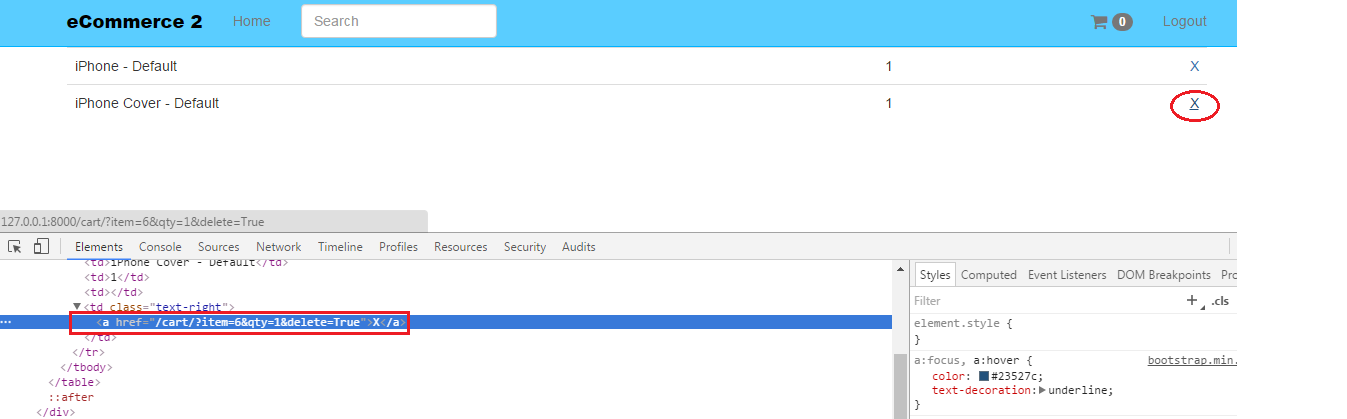
+ def remove\_from\_cart(self):

+ return "%s?item=%s&qty=1&delete=True" %(reverse("cart"), self.id)

+

+ def get\_title(self):

+ return "%s - %s" %(self.product.title, self.title)



# 044 Update Cart Item quantity in Cart

这个章节有一个view含多个form的，可以看看怎么处理的

html里添加q参数，传递到url，然后在view处理。这些GET参数并不影响model

# modified: carts/templates/carts/view.html

----------------------- carts/templates/carts/view.html -----------------------

{% extends "base.html" %}

+<script>

+{% block jquery %}

+$(".item-qty").change(function(){

+ $(this).next(".btn-update").fadeIn();

+});

+

+{% endblock %}

+</script>

{% block content %}

<table class='table'>

{% for item in object.cartitem\_set.all %}

<tr id='item-{{ item.item.id }}'>

+ <form action="." method="GET">

<td>{{ item.item.get\_title }}</td>

+ <input type='hidden' name='item' value='{{ item.item.id }}' />

- <td>{{ item.quantity }}</td>

+ <td><input type='number' class='item-qty' name='qty' value='{{ item.quantity }}' /><input type='submit' class='btn-update btn btn-link' value='Update item' style='display:none;'/></td>

<td>{{ item.line\_total }}</td>

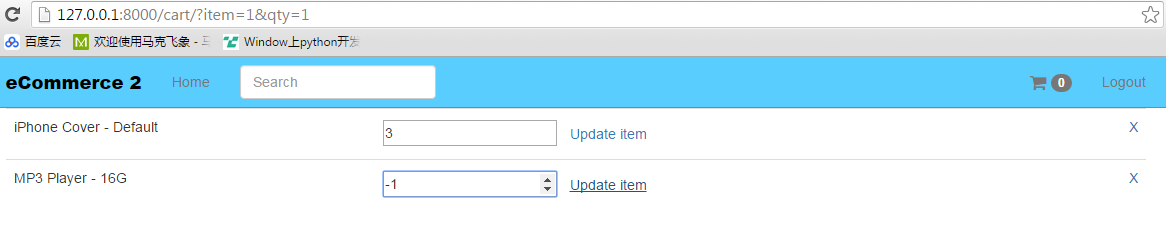
<td class='text-right' ><a href='{{ item.remove }}'>X</a></td>

+ </form>

</tr>

{% endfor %}

</table>



调节数字，后面的“update Item”会fade in

更新“update Item”

- 如果quantity > 0, 数字更新并且“update item”消失

- 如果quantity < 0, 该行被删除

# 045 Cart Item Line Total

执行update item之后，价格会自动更新

<https://docs.djangoproject.com/en/1.8/ref/signals/#pre-save>

之前操作将migration目录删掉了，migrate不起作用

- 添加migrations目录，并添加\_\_init\_\_.py文件

- 执行makemigrations，这时候会生成0001\_initial.py,这个是基于当前修改的文件

- 执行migrate

- 修改CartItem

- 执行makemigrations，migrate，生成0002\_cartitem\_line\_item\_total.py文件

# modified: carts/models.py

# modified: carts/templates/carts/view.html

------------------------------- carts/models.py -------------------------------

+from decimal import Decimal

from django.conf import settings

from django.db import models

+from django.db.models.signals import pre\_save

class CartItem(models.Model):

cart = models.ForeignKey("Cart") #cart class is below

item = models.ForeignKey(Variation)

quantity = models.PositiveIntegerField(default=1)

+ line\_item\_total = models.DecimalField(max\_digits=10, decimal\_places=2)

+def cart\_item\_pre\_save\_receiver(sender, instance, \*args, \*\*kwargs):

+ qty = instance.quantity

+ if qty >= 1:

+ price = instance.item.get\_price()

+ line\_item\_total = Decimal(qty) \* Decimal(price)

+ instance.line\_item\_total = line\_item\_total

+

+pre\_save.connect(cart\_item\_pre\_save\_receiver, sender=CartItem)

----------------------- carts/templates/carts/view.html -----------------------

{% block content %}

<table class='table'>

{% for item in object.cartitem\_set.all %}

<tr id='item-{{ item.item.id }}'>

<form action="." method="GET">

<td>{{ item.item.get\_title }}</td>

<input type='hidden' name='item' value='{{ item.item.id }}' />

<td><input type='number' class='item-qty' name='qty' value='{{ item.quantity }}' /><input type='submit' class='btn-update btn btn-link' value='Update item' style='display:none;'/></td>

- <td>{{ item.line\_total }}</td>

+ <td>{{ item.line\_item\_total }}</td>

<td class='text-right' ><a href='{{ item.remove }}'>X</a></td>

</form>

# 046 Cart Subtotal

该章节：增加总价功能

# modified: carts/models.py

# modified: carts/templates/carts/view.html

------------------------------- carts/models.py -------------------------------

-from django.db.models.signals import pre\_save

+from django.db.models.signals import pre\_save, post\_save, post\_delete

class CartItem(models.Model):

+def cart\_item\_post\_save\_receiver(sender, instance, \*args, \*\*kwargs):

+ instance.cart.update\_subtotal()

+post\_save.connect(cart\_item\_post\_save\_receiver, sender=CartItem)

+post\_delete.connect(cart\_item\_post\_save\_receiver, sender=CartItem)

class Cart(models.Model):

user = models.ForeignKey(settings.AUTH\_USER\_MODEL, null=True, blank=True)

items = models.ManyToManyField(Variation, through=CartItem)

timestamp = models.DateTimeField(auto\_now\_add=True, auto\_now=False)

updated = models.DateTimeField(auto\_now\_add=False, auto\_now=True)

+ subtotal = models.DecimalField(max\_digits=50, decimal\_places=2)

+ def update\_subtotal(self):

+ print "updating..."

+ subtotal = 0

+ items = self.cartitem\_set.all()

+ for item in items:

+ subtotal += item.line\_item\_total

+ self.subtotal = "%.2f" %(subtotal)

+ self.save()

----------------------- carts/templates/carts/view.html -----------------------

{% block content %}

<table class='table'>

{% for item in object.cartitem\_set.all %}

<tr id='item-{{ item.item.id }}'>

<form action="." method="GET">

<td>{{ item.item.get\_title }}</td>

<input type='hidden' name='item' value='{{ item.item.id }}' />

<td><input type='number' class='item-qty' name='qty' value='{{ item.quantity }}' /><input type='submit' class='btn-update btn btn-link' value='Update item' style='display:none;'/></td>

<td>{{ item.line\_item\_total }}</td>

<td class='text-right' ><a href='{{ item.remove }}'>X</a></td>

</form>

</tr>

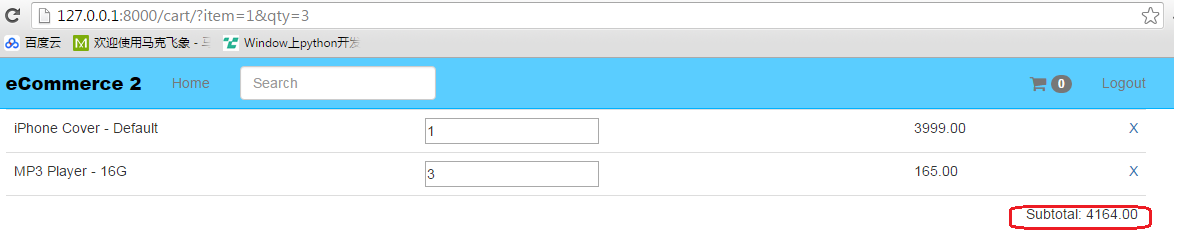
{% endfor %}

+ <tr>

+ <td colspan='4' class='text-right'>Subtotal: <span id='subtotal'>{{ object.subtotal }}</span></td>

+ </tr>

colspan表示第四列



# 047 Ajax Part1

Ajax = Asynchronous Javascript and XML

<https://docs.djangoproject.com/en/1.8/ref/request-response/#django.http.JsonResponse>

功能1 add to cart

# modified: carts/views.py

# modified: products/templates/products/product\_detail.html

-------------------------------- carts/views.py --------------------------------

-from django.http import HttpResponseRedirect

+from django.http import HttpResponseRedirect, Http404, JsonResponse

class CartView(SingleObjectMixin, View):

def get(self, request, \*args, \*\*kwargs):

cart = self.get\_object()

item\_id = request.GET.get('item')

- delete\_item = request.GET.get('delete')

+ delete\_item = request.GET.get('delete', False)

+ item\_added = False

class CartView(SingleObjectMixin, View):

def get(self, request, \*args, \*\*kwargs):

+ if request.is\_ajax():

+ data = {

+ "deleted": delete\_item,

+ "item\_added": item\_added,

+ }

+ return JsonResponse(data)

--------------- products/templates/products/product\_detail.html ---------------

+// ajax add to cart

+$("#submit-btn").click(function(event){

+ event.preventDefault();

+ var formData = $("#add-form").serialize();

+ console.log(formData);

+ $.ajax({

+ type: "GET", // "POST"

+ url: "{% url 'cart' %}",

+ data: formData,

+ success: function(data) {

+ console.log(data)

+ $("#jquery-message").text("Added " + data.item\_added + " Deleted " + data.deleted)

+ },

+ error: function(response, error) {

+ // console.log(response)

+ // console.log(error)

+ $("#add-form").submit()

+ }

+ })

+

+ // $("#add-form").submit() # it will go to the next page

})

<div class="col-sm-4">

+ <p id='jquery-message' class='lead'>

- <form method='GET' action="{% url 'cart' %}">

+ <form **id='add-form'** method='GET' action="{% url 'cart' %}"> <!-- add form for AJAX -->

{% if object.variation\_set.count > 1 %}

<h3 id='price'>{{ object.variation\_set.first.price }}</h3>

<select name='item' class='form-control variation\_select'>

<br/>

<input class='form-control' type='number' name='qty' value='1' />

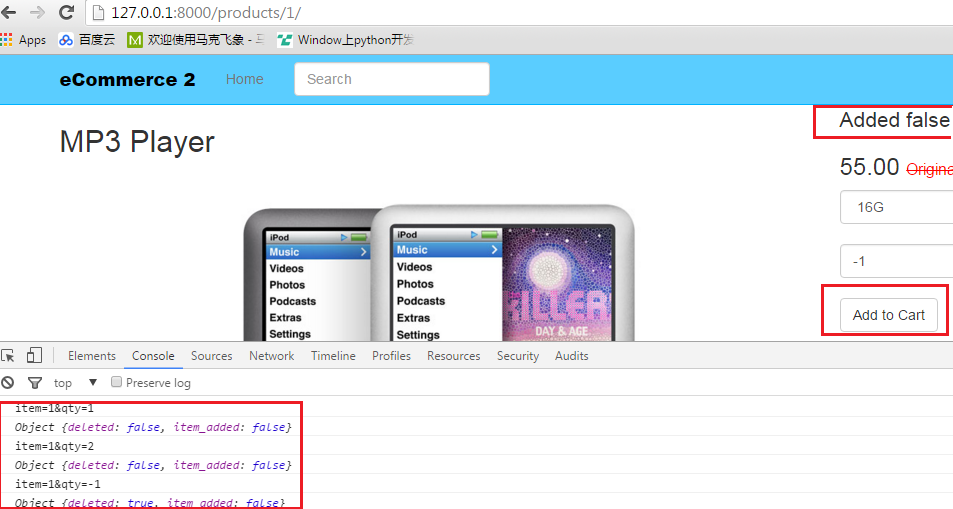
<br/>

- <input type='submit' value='Add to Cart' class='btn btn-default' />

+ <input **id='submit-btn'** type='submit' value='Add to Cart' class='btn btn-default' /> <!-- submit-btn for AJAX -->

</form>

执行add to cart, 页面不会跳转，console.log()可打印相应的log



# 048 Ajax Part2

调节数量，total和subtotal会自动调整

# modified: carts/templates/carts/view.html

# modified: carts/views.py

-------------------------------- carts/views.py --------------------------------

class CartView(SingleObjectMixin, View):

else:

cart\_item.quantity = qty

cart\_item.save()

+ if not request.is\_ajax():

+ return HttpResponseRedirect(reverse("cart"))

+ #return cart\_item.cart.get\_absolute\_url()

if request.is\_ajax():

+ try:

+ total = cart\_item.line\_item\_total

+ except:

+ total = None

+ try:

+ subtotal = cart\_item.cart.subtotal

+ except:

+ subtotal = None

data = {

"deleted": delete\_item,

"item\_added": item\_added,

+ "line\_total": total,

+ "subtotal": subtotal,

}

return JsonResponse(data)

----------------------- carts/templates/carts/view.html -----------------------

<script>

{% block jquery %}

$(".item-qty").change(function(){

- $(this).next(".btn-update").fadeIn();

+ // $(this).next(".btn-update").fadeIn();

+ // event.preventDefault();

+ var item = $(this).prev("input[type='hidden']").val();

+ var qty = $(this).val()

+ var data = {

+ item: item,

+ qty: qty

+ }

+ console.log(data);

+ $.ajax({

+ type: "GET", // "POST"

+ url: "{% url 'cart' %}",

+ data: data,

+ success: function(data) {

+ $("#jquery-message").text("Added " + data.item\_added + " Deleted " + data.deleted)

+ if (data.deleted){

+ $("#item-"+item).fadeOut();

+ $("#subtotal").text(data.subtotal);

+ } else {

+ $("#item-line-total-"+item).text(data.line\_total);

+ $("#subtotal").text(data.subtotal);

+ }

+ },

+ error: function(response, error) {

+ // console.log(response)

+ // console.log(error)

+ $("#add-form").submit()

+ }

+ })

});

<table class='table'>

{% for item in object.cartitem\_set.all %}

- <tr>

+ <tr **id='item-{{ item.item.id }}'**> <!-- id for AJAX-->

- <form action="." method="GET">

- <input type='hidden' name='item' value='{{ item.item.id }}' />

- <td><input type='number' class='item-qty' name='qty' value='{{ item.quantity }}' /><input type='submit' class='btn-update btn btn-link' value='Update item' style='display:none;'/></td>

- </form>

<td>{{ item.item.get\_title }}</td>

+ <td><form action="." method="GET"><input type='hidden' name='item' value='{{ item.item.id }}' /><input type='number' class='item-qty' name='qty' value='{{ item.quantity }}' /><input type='submit' class='btn-update btn btn-link' value='Update item' style='display:none;'/></form></td>

- <td>{{ item.line\_item\_total }}</td>

+ <td **id='item-line-total-{{ item.item.id }}**'>{{ item.line\_item\_total }}</td>

<td class='text-right' ><a href='{{ item.remove }}'>X</a></td>

</tr>

{% endfor %}

<tr>

- <td colspan='4' class='text-right'>Subtotal: {{ object.subtotal }}</td> <!-- id for AJAX-->

+ <td id='subtotal' colspan='4' class='text-right'>Subtotal: {{ object.subtotal }}</td> <!-- id for AJAX-->

</tr>

这边添加了三个id，为了jquery操作使用

<tr id='item-{{ item.item.id }}'>

<td **id='item-line-total-{{ item.item.id }}**'>

<td id='subtotal'

# 049 Jquery Flash Message

<http://getbootstrap.com/components/#alerts>

jquery语句include的template不能有空格回车，必须连在一起否则html显示会有问题

传递参数时用下列语句

var template = "{% include 'alert.html' with message='" + message + "' %}"

# modified: carts/templates/carts/view.html

# modified: products/templates/products/product\_detail.html

# modified: static\_in\_pro/our\_static/css/custom.css

# modified: templates/javascript.html

#

# static\_in\_pro/our\_static/js/custom.js

# templates/alert.html

----------------------- carts/templates/carts/view.html -----------------------

$(".item-qty").change(function(){

success: function(data) {

$("#jquery-message").text("Added " + data.item\_added + " Deleted " + data.deleted)

if (data.deleted){

$("#item-"+item).fadeOut();

$("#subtotal").text(data.subtotal);

} else {

$("#item-line-total-"+item).text(data.line\_total);

$("#subtotal").text(data.subtotal);

}

+ showFlashMessage("cart updated");

},

--------------- products/templates/products/product\_detail.html ---------------

$("#submit-btn").click(function(event){

url: "{% url 'cart' %}",

data: formData,

success: function(data) {

- console.log(data)

- $("#jquery-message").text("Added " + data.item\_added + " Deleted " + data.deleted)

+ //console.log(data)

+ //$("#jquery-message").text("Added " + data.item\_added + " Deleted " + data.deleted)

+ showFlashMessage("product added");

},

------------------- static\_in\_pro/our\_static/css/custom.css -------------------

+.container-alert-flash {

+ position: absolute;

+ top: 10px;

+ z-index: 50000;

+ width: 100%;

+ right: 0%;

+ margin: 0px;

+ display:none;

+}

-------------------------- templates/javascript.html --------------------------

+ <script src="{% static 'js/custom.js' %}"></script>

static\_in\_pro/our\_static/js/custom.js

function showFlashMessage(message) {

// var template = "{% include 'alert.html' with message='" + message + "' %}"

var template = "<div class='container container-alert-flash'>" +

"<div class='col-sm-3 col-sm-offset-8'> " +

"<div class='alert alert-success alert-dismissible' role='alert'>" +

"<button type='button' class='close' data-dismiss='alert' aria-label='Close'>" +

"<span aria-hidden='true'>&times;</span></button>"

+ message + "</div></div></div>"

$("body").append(template);

$(".container-alert-flash").fadeIn();

setTimeout(function(){

$(".container-alert-flash").fadeOut();

}, 1800);

}

# 050 Display Message from Server

# modified: carts/templates/carts/view.html

# modified: carts/views.py

# modified: products/templates/products/product\_detail.html

----------------------- carts/templates/carts/view.html -----------------------

$(".item-qty").change(function(){

$("#item-line-total-"+item).text(data.line\_total);

$("#subtotal").text(data.subtotal);

}

+ var message = ""

+ if (data.item\_added) {

+ message = "New item added"

+ } else if (data.deleted){

+ message = "Item removed"

+ } else {

+ message = "Item updated"

+ }

- showFlashMessage("cart updated");

+ showFlashMessage(message);

},

-------------------------------- carts/views.py --------------------------------

+from django.core.urlresolvers import reverse

class CartView(SingleObjectMixin, View):

cart = self.get\_object()

item\_id = request.GET.get('item')

delete\_item = request.GET.get('delete', False)

+ flash\_message = ""

item\_added = False

if item\_id:

- cart\_item = CartItem.objects.get\_or\_create(cart=cart, item=item\_instance)[0]

+ cart\_item, **created** = CartItem.objects.get\_or\_create(cart=cart, item=item\_instance)

+ if created:

+ flash\_message = "Successfully added to the cart"

+ item\_added = True

if delete\_item:

+ flash\_message = "Item removed successfully."

cart\_item.delete()

else:

+ if not created:

+ flash\_message = "Quantity has been updated successfully."

cart\_item.quantity = qty

cart\_item.save()

data = {

"deleted": delete\_item,

"item\_added": item\_added,

"line\_total": total,

"subtotal": subtotal,

+ "flash\_message": flash\_message,

}

return JsonResponse(data)

--------------- products/templates/products/product\_detail.html ---------------

$("#submit-btn").click(function(event){

success: function(data) {

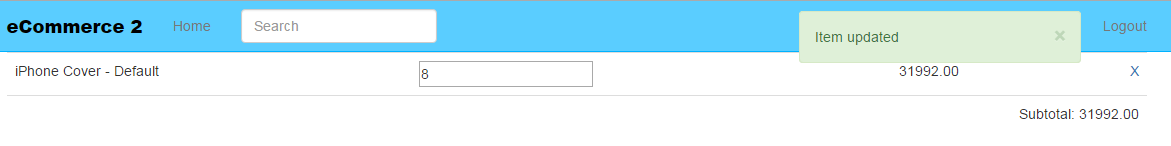
//console.log(data)

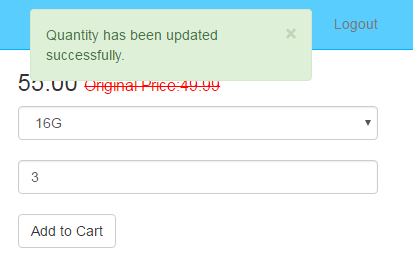
//$("#jquery-message").text("Added " + data.item\_added + " Deleted " + data.deleted)

- showFlashMessage("product added");

+ showFlashMessage(data.flash\_message);

},





# 087 Empty Cart

# modified: carts/templates/carts/view.html

# modified: carts/views.py

# carts/templates/carts/empty\_cart.html

carts/models.py

-from django.db.models.signals import pre\_save, post\_save

+from django.db.models.signals import pre\_save, post\_save, post\_delete

+post\_delete.connect(cart\_item\_post\_save\_receiver, sender=CartItem)

----------------------- carts/templates/carts/view.html -----------------------

$(".item-qty").change(function(){

$("#item-line-total-"+item).text(data.line\_total);

$("#subtotal").text(data.subtotal);

}

+ if (data.total\_items == 0 ) {

+ $(".table").fadeOut()

+ var template = "{% include 'carts/empty\_cart.html' %}";

+ $(".main-content").html(template);

+ }

var message = ""

{% block content %}

+<div class='row main-content'>

+{% if object.cartitem\_set.count < 1 %}

+ {% include "carts/empty\_cart.html" %}

+{% else %}

<table class='table'>

{% for item in object.cartitem\_set.all %}

<tr id='item-{{ item.item.id }}'> <!-- id for AJAX-->

+{% endif %}

+</div>

-------------------------------- carts/views.py --------------------------------

class CartView(SingleObjectMixin, View):

subtotal = cart\_item.cart.subtotal

except:

subtotal = None

+ try:

+ total\_items = cart\_item.cart.items.count()

+ except:

+ total\_items = 0

data = {

"deleted": delete\_item,

"item\_added": item\_added,

"line\_total": total,

"subtotal": subtotal,

"flash\_message": flash\_message,

+ "total\_items": total\_items

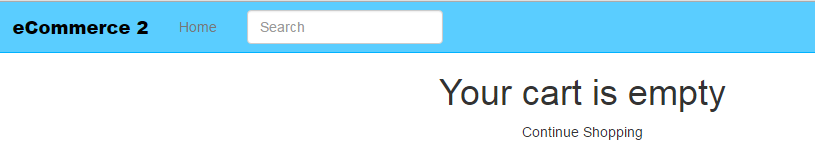
}

return JsonResponse(data)

carts/templates/carts/empty\_cart.html

再次强调，js里面include html时应该escape空格

**<!-- MAKE SURE YOUR DATA IS ESCAPED -->**<div class=**'col-sm-6 col-sm-offset-3 text-center'**><h1>**Your cart is empty**</h1><p>**Continue Shopping**</p></div>



# 088 Cart Count in Navbar

执行一次ajax，类似于form的submit

不把count保存到session，显示会一直闪（数字会返回为0）

# modified: carts/models.py

# modified: carts/templates/carts/view.html

# modified: carts/views.py

# modified: ecommerce2/urls.py

# modified: products/templates/products/product\_detail.html

# modified: templates/javascript.html

# modified: templates/navbar.html

------------------------------- carts/models.py -------------------------------

class Cart(models.Model):

items = models.ManyToManyField(Variation, through=CartItem)

timestamp = models.DateTimeField(auto\_now\_add=True, auto\_now=False)

updated = models.DateTimeField(auto\_now\_add=False, auto\_now=True)

- subtotal = models.DecimalField(max\_digits=50, decimal\_places=2)

+ subtotal = models.DecimalField(max\_digits=50, decimal\_places=2, default=25.00)

----------------------- carts/templates/carts/view.html -----------------------

$(".item-qty").change(function(){

message = "Item updated"

}

showFlashMessage(message);

+ updateCartItemCount();

},

-------------------------------- carts/views.py --------------------------------

# Create your views here.

+class ItemCountView(View):

+ def get(self, request, \*args, \*\*kwargs):

+ if request.is\_ajax():

+ cart\_id = self.request.session.get("cart\_id")

+ if cart\_id == None:

+ count = 0

+ else:

+ cart = Cart.objects.get(id=cart\_id)

+ count = cart.items.count()

+ request.session["cart\_item\_count"] = count

+ return JsonResponse({"count": count})

+ else:

+ raise Http404

------------------------------ ecommerce2/urls.py ------------------------------

-from carts.views import CartView

+from carts.views import CartView, ItemCountView

urlpatterns = [

# Examples:

url(r'^products/', include('products.urls')),

url(r'^categories/', include('products.urls\_categories')),

+ url(r'^cart/count/$', ItemCountView.as\_view(), name='item\_count'),

]

--------------- products/templates/products/product\_detail.html ---------------

$("#submit-btn").click(function(event){

//console.log(data)

//$("#jquery-message").text("Added " + data.item\_added + " Deleted " + data.deleted)

showFlashMessage(data.flash\_message);

+ updateCartItemCount();

},

-------------------------- templates/javascript.html --------------------------

+ <script>

+

+ function updateCartItemCount(){

+ var badge = $("#cart-count-badge");

+

+ $.ajax({

+ type: "GET",

+ url: "{% url 'item\_count' %}",

+ success: function(data){

+ badge.text(data.count);

+ console.log(data.count);

+ },

+ error: function(response, error) {

+ console.log(response)

+ console.log(error);

+ }

+ })

+

+ }

+ $(document).ready(function(){

+ updateCartItemCount()

+ })

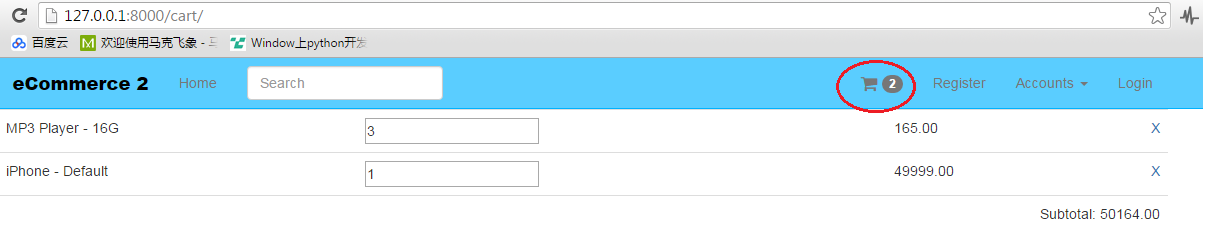
+

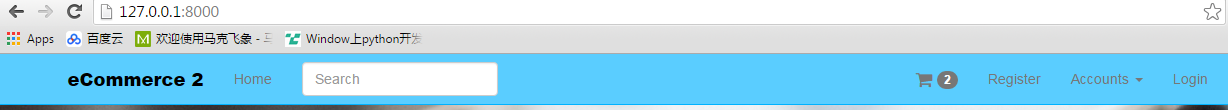
+ </script>

---------------------------- templates/navbar.html ----------------------------

- <li><a href="#"><i class="fa fa-shopping-cart fa-navbar-cart"></i> <span class='badge'>0</span></a></li>

+ <li><a **href='{% url "cart" %}'**><i class="fa fa-shopping-cart fa-navbar-cart"></i> <span **id="cart-count-badge"** class='badge'>**{{ request.session.cart\_item\_count }}**</span></a></li>





# 089 Cart Total Taxes

# modified: carts/models.py

# modified: carts/templates/carts/view.html

# modified: carts/views.py

------------------------------- carts/models.py -------------------------------

class Cart(models.Model):

timestamp = models.DateTimeField(auto\_now\_add=True, auto\_now=False)

updated = models.DateTimeField(auto\_now\_add=False, auto\_now=True)

subtotal = models.DecimalField(max\_digits=50, decimal\_places=2, default=25.00)

+ tax\_percentage = models.DecimalField(max\_digits=10, decimal\_places=5, default=0.085)

+ tax\_total = models.DecimalField(max\_digits=50, decimal\_places=2, default=25.00)

+ total = models.DecimalField(max\_digits=50, decimal\_places=2, default=25.00)

+ # discounts

+ # shipping

+def do\_tax\_and\_total\_receiver(sender, instance, \*args, \*\*kwargs):

+ subtotal = Decimal(instance.subtotal)

+ tax\_total = round(subtotal \* Decimal(instance.tax\_percentage), 2) #8.5%

+ print instance.tax\_percentage

+ total = round(subtotal + Decimal(tax\_total), 2)

+ instance.tax\_total = "%.2f" %(tax\_total)

+ instance.total = "%.2f" %(total)

+ #instance.save()

+pre\_save.connect(do\_tax\_and\_total\_receiver, sender=Cart)

----------------------- carts/templates/carts/view.html -----------------------

$(".item-qty").change(function(){

if (data.deleted){

$("#item-"+item).fadeOut();

$("#subtotal").text(data.subtotal);

+ $("#taxtotal").text(data.tax\_total);

+ $("#carttotal").text(data.cart\_total);

} else {

$("#item-line-total-"+item).text(data.line\_total);

$("#subtotal").text(data.subtotal);

+ $("#taxtotal").text(data.tax\_total);

+ $("#carttotal").text(data.cart\_total);

}

<table class='table'>

+ <tr>

+ <td colspan='4' class='text-right'>Tax (Estimated): <span id='taxtotal'>{{ object.tax\_total }}</span></td>

+ </tr>

+

+ <tr>

+ <td colspan='4' class='text-right'>Total: <span id='carttotal'>{{ object.total }}</span></td>

+ </tr>

-------------------------------- carts/views.py --------------------------------

class CartView(SingleObjectMixin, View):

cart\_id = self.request.session.get("cart\_id")

if cart\_id == None:

cart = Cart()

+ cart.tax\_percentage = 0.075

cart.save()

cart\_id = cart.id

self.request.session["cart\_id"] = cart\_id

+ try:

+ cart\_total = cart\_item.cart.total

+ except:

+ cart\_total = None

+ try:

+ tax\_total = cart\_item.cart.tax\_total

+ except:

+ tax\_total = None

data = {

"deleted": delete\_item,

"item\_added": item\_added,

"line\_total": total,

"subtotal": subtotal,

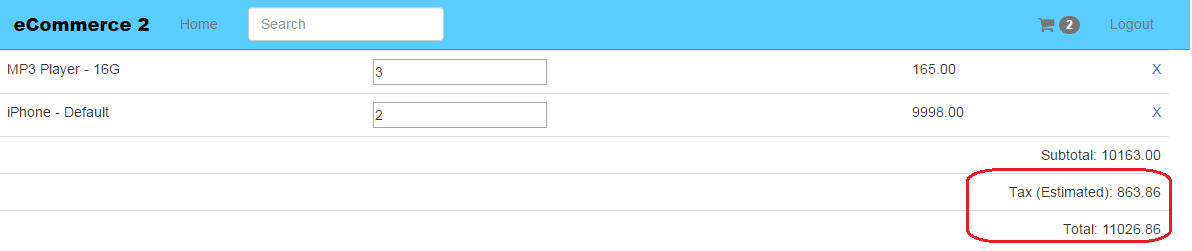
+ "cart\_total": cart\_total,

+ "tax\_total": tax\_total,

"flash\_message": flash\_message,

"total\_items": total\_items

}



# 090 Checkout View

request.build\_absolute\_uri？

# modified: carts/views.py

# modified: ecommerce2/urls.py

# carts/templates/carts/checkout\_view.html

-------------------------------- carts/views.py --------------------------------

-from django.shortcuts import render, get\_object\_or\_404

-from django.views.generic.detail import SingleObjectMixin

+from django.shortcuts import render, get\_object\_or\_404, redirect

+from django.views.generic.detail import SingleObjectMixin, DetailView

+from django.contrib.auth.forms import AuthenticationForm

+class CheckoutView(DetailView):

+ model = Cart

+ template\_name = "carts/checkout\_view.html"

+

+ def get\_object(self, \*args, \*\*kwargs):

+ cart\_id = self.request.session.get("cart\_id")

+ if cart\_id == None:

+ return redirect("cart")

+ cart = Cart.objects.get(id=cart\_id)

+ return cart

+

+ def get\_context\_data(self, \*args, \*\*kwargs):

+ context = super(CheckoutView, self).get\_context\_data(\*args, \*\*kwargs)

+ user\_can\_continue = False

+ if not self.request.user.is\_authenticated():

+ context["login\_form"] = AuthenticationForm()

+ context["next\_url"] = self.request.build\_absolute\_uri()

+

+ if self.request.user.is\_authenticated():

+ user\_can\_continue = True

+

+ context["user\_can\_continue"] = user\_can\_continue

+

+ return context

------------------------------ ecommerce2/urls.py ------------------------------

-from carts.views import CartView, ItemCountView

+from carts.views import CartView, ItemCountView, CheckoutView

+ url(r'^checkout/$', CheckoutView.as\_view(), name='checkout'),

carts/templates/carts/checkout\_view.html

{% extends "base.html" %}

{% load i18n %}

{% load crispy\_forms\_tags %}

{% block content %}

{% if not user\_can\_continue %}

<div class='col-sm-6'>

<p class='lead'>Continue as Guest</p>

</div>

<div class='col-sm-6'>

<p class='lead'>Login to Continue</p>

<form method='POST' action="{% url 'auth\_login' %}"> {% csrf\_token %}

{{ login\_form|crispy }}

<input type='hidden' name='next' value='{{ next\_url }}' />

<input type='submit' class='btn btn-success' value='Login' />

</form>

<p class='text-center'>

<p>{% trans "Forgot password" %}? <a href="{% url 'auth\_password\_reset' %}">{% trans "Reset it" %}</a>!</p>

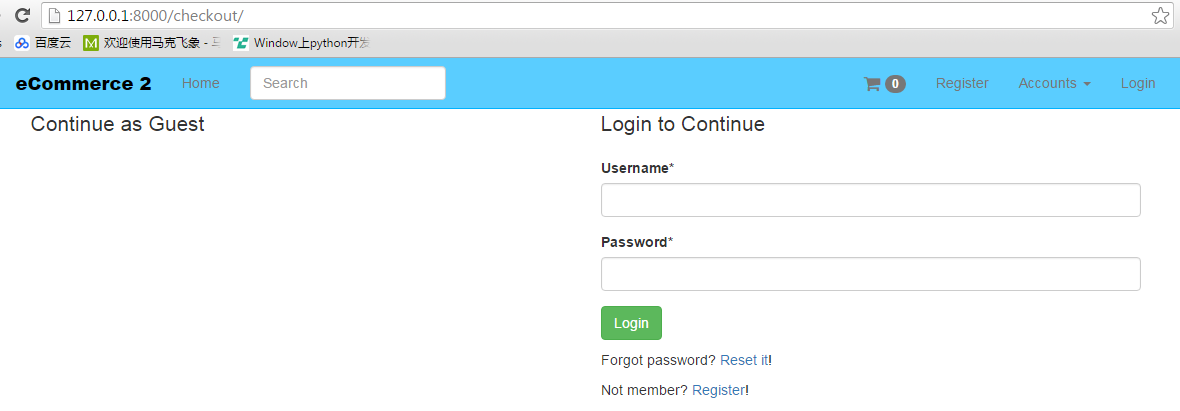
<p>{% trans "Not member" %}? <a href="{% url 'registration\_register' %}">{% trans "Register" %}</a>!</p>

</p>

</div>

{% endif %}

{% endblock %}



# 091 Orders App

# modified: ecommerce2/settings/base.py

# orders/

<https://docs.djangoproject.com/en/1.8/topics/db/models/#one-to-one-relationships>

------------------------- ecommerce2/settings/base.py -------------------------

INSTALLED\_APPS **=** **(**

'newsletter'**,**

'products'**,**

'carts'**,**

**+** 'orders'**,**

'crispy\_forms'**,**

'registration'**,**

**)**

orders/

**from** django**.**conf **import** settings

**from** django**.**db **import** models

# Create your models here.

**class** **UserCheckout(**models**.**Model**):**

user **=** models**.**OneToOneField**(**settings**.**AUTH\_USER\_MODEL**,** null**=True,** blank**=True)** #not required

email **=** models**.**EmailField**(**unique**=True)** #--> required

**def** \_\_unicode\_\_**(**self**):** #def \_\_str\_\_(self):

**return** self**.**email

# 092 Checkout from FromMixin

<https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-editing/#formmixin>

FormMixin[¶](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-editing/#formmixin)

get\_success\_url()[¶](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-editing/#django.views.generic.edit.FormMixin.get_success_url)

Determine the URL to redirect to when the form is successfully validated. Returns [success\_url](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-editing/#django.views.generic.edit.FormMixin.success_url) by default.

<https://docs.djangoproject.com/en/1.8/ref/class-based-views/generic-display/#detailview>

**classdjango.views.generic.detail.DetailView**[**¶**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/generic-display/#django.views.generic.detail.DetailView)

**get\_object(queryset=None)**[**¶**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin.get_object)

Returns the single object that this view will display. If **queryset** is provided, that queryset will be used as the source of objects; otherwise, [**get\_queryset()**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin.get_queryset) will be used. **get\_object()** looks for a [**pk\_url\_kwarg**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin.pk_url_kwarg)argument in the arguments to the view; if this argument is found, this method performs a primary-key based lookup using that value. If this argument is not found, it looks for a [**slug\_url\_kwarg**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin.slug_url_kwarg) argument, and performs a slug lookup using the [**slug\_field**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin.slug_field).

**get\_context\_data(\*\*kwargs)**[**¶**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin.get_context_data)

Returns context data for displaying the list of objects.

The base implementation of this method requires that the **self.object** attribute be set by the view (even if**None**). Be sure to do this if you are using this mixin without one of the built-in views that does so.

It returns a dictionary with these contents:

* **object**: The object that this view is displaying (**self.object**).
* **context\_object\_name**: **self.object** will also be stored under the name returned by[**get\_context\_object\_name()**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/#django.views.generic.detail.SingleObjectMixin.get_context_object_name), which defaults to the lowercased version of the model name.

**Using SingleObjectMixin with View**[**¶**](https://docs.djangoproject.com/en/1.8/topics/class-based-views/mixins/#using-singleobjectmixin-with-view)

If we want to write a simple class-based view that responds only to **POST**, we’ll subclass [**View**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/base/#django.views.generic.base.View) and write a **post()**method in the subclass. However if we want our processing to work on a particular object, identified from the URL, we’ll want the functionality provided by **[SingleObjectMixin](https://docs.djangoproject.com/en/1.8/ref/class-based-views/mixins-single-object/" \l "django.views.generic.detail.SingleObjectMixin" \o "django.views.generic.detail.SingleObjectMixin)**.

We’ll demonstrate this with the **Author** model we used in the [generic class-based views introduction](https://docs.djangoproject.com/en/1.8/topics/class-based-views/generic-display/).

# modified: carts/templates/carts/checkout\_view.html

# modified: carts/views.py

# modified: orders/forms.py

**-------------------** carts**/**templates**/**carts**/**checkout\_view**.**html **-------------------**

**{%** block content **%}-**

**+<**div **class=**'col-sm-6'**>**

**+** **<**form method**=**'POST' action**=**''**>{%** csrf\_token **%}**

**+** **{{** form**|**crispy **}}**

**+** **<**input type**=**'submit' **class=**'btn btn-success' value**=**'Continue as Guest' **/>**

**+** **</**form**>**

**+</**div**>**

**--------------------------------** carts**/**views**.**py **--------------------------------**

**+from** django**.**views**.**generic**.**edit **import** FormMixin

**+from** orders**.**forms **import** GuestCheckoutForm

**-class** **CheckoutView(**DetailView**):**

**+class** **CheckoutView(**FormMixin**,** DetailView**):**

model **=** Cart

template\_name **=** "carts/checkout\_view.html"

**+** form\_class **=** GuestCheckoutForm

context**[**"form"**]** **=** self**.**get\_form**()**

**+** **def** post**(**self**,** request**,** **\***args**,** **\*\***kwargs**):**

**+** form **=** self**.**get\_form**()**

**+** **if** form**.**is\_valid**():**

**+** **print** form**.**cleaned\_data**.**get**(**"email"**)**

**+** **return** self**.**form\_valid**(**form**)**

**+** **else:**

**+** **return** self**.**form\_invalid**(**form**)**

**+**

**+** **def** get\_success\_url**(**self**):**

**+** **return** reverse**(**"checkout"**)**

**-------------------------------** orders**/**forms**.**py **-------------------------------**

**+from** django **import** forms

**+**

**+class** **GuestCheckoutForm(**forms**.**Form**):**

**+** email **=** forms**.**EmailField**()**

**+** email2 **=** forms**.**EmailField**(**label**=**'Verify Email'**)**

**+**

**+** **def** clean\_email2**(**self**):**

**+** email **=** self**.**cleaned\_data**.**get**(**"email"**)**

**+** email2 **=** self**.**cleaned\_data**.**get**(**"email2"**)**

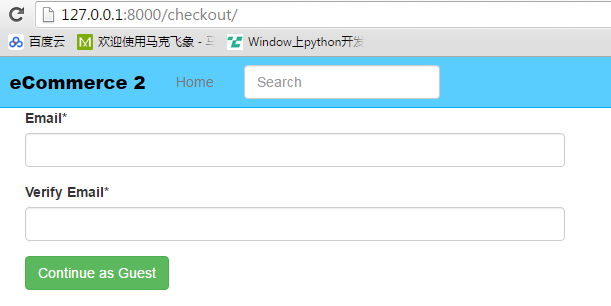
**+**

**+** **if** email **==** email2**:**

**+** **return** email2

**+** **else:**

**+** **raise** forms**.**ValidationError**(**"Please confirm emails are the same"**)**



# 093 Save Guest Info

保存Guest的信息

# modified: carts/templates/carts/checkout\_view.html

# modified: carts/views.py

# modified: orders/admin.py

# modified: orders/forms.py

# modified: templates/navbar.html

------------------- carts/templates/carts/checkout\_view.html -------------------

**{% block content %}**

-<div class='col-sm-6'>

- <form method='POST' action=''>{% csrf\_token %}

- {{ form|crispy }}

- <input type='submit' class='btn btn-success' value='Continue as Guest' />

- </form>

-</div>

{% if not user\_can\_continue %}

<div class='col-sm-6'>

<p class='lead'>Continue as Guest</p>

+ <form method='POST' action=''>{% csrf\_token %} <!-- this POST will go to checkout view -->

+ {{ form|crispy }}

+ <input type='submit' class='btn btn-success' value='Continue as Guest' />

+ </form>

</div>

<div class='col-sm-6'>

-------------------------------- carts/views.py --------------------------------

**+from** orders**.**models **import** UserCheckout

**class** **CheckoutView(**FormMixin**,** DetailView**):**

**def** get\_context\_data**(**self**,** **\***args**,** **\*\***kwargs**):**

context **=** super**(**CheckoutView**,** self**).**get\_context\_data**(\***args**,** **\*\***kwargs**)**

user\_can\_continue **=** **False**

**+** user\_check\_id **=** self**.**request**.**session**.**get**(**"user\_checkout\_id"**)**

**-** **if** **not** self**.**request**.**user**.**is\_authenticated**():**

**+** **if** **not** self**.**request**.**user**.**is\_authenticated**()** **or** user\_check\_id **==** **None:** #or request.user.is\_guest:

context**[**"login\_form"**]** **=** AuthenticationForm**()**

context**[**"next\_url"**]** **=** self**.**request**.**build\_absolute\_uri**()**

**-** **if** self**.**request**.**user**.**is\_authenticated**():**

**+** **elif** self**.**request**.**user**.**is\_authenticated**()** **or** user\_check\_id **!=** **None:**

user\_can\_continue **=** **True**

**+** **else:**

**+** **pass**

**def** post**(**self**,** request**,** **\***args**,** **\*\***kwargs**):**

**+** self**.**object **=** self**.**get\_object**()**

form **=** self**.**get\_form**()**

**if** form**.**is\_valid**():**

**-** **print** form**.**cleaned\_data**.**get**(**"email"**)**

**+** email **=** form**.**cleaned\_data**.**get**(**"email"**)**

**+** user\_checkout**,** created **=** UserCheckout**.**objects**.**get\_or\_create**(**email**=**email**)**

**+** request**.**session**[**"user\_checkout\_id"**]** **=** user\_checkout**.**id

**return** self**.**form\_valid**(**form**)**

**else:**

**return** self**.**form\_invalid**(**form**)**

------------------------------- orders/admin.py -------------------------------

**+from** **.**models **import** UserCheckout

**+**admin**.**site**.**register**(**UserCheckout**)**

------------------------------- orders/forms.py -------------------------------

**+from** django**.**contrib**.**auth **import** get\_user\_model

**+**User **=** get\_user\_model**()**

**class** **GuestCheckoutForm(**forms**.**Form**):**

**def** clean\_email2(self):

email2 **=** self**.**cleaned\_data**.**get**(**"email2"**)**

**if** email **==** email2**:**

**+** user\_exists **=** User**.**objects**.**filter**(**email**=**email**).**count**()**

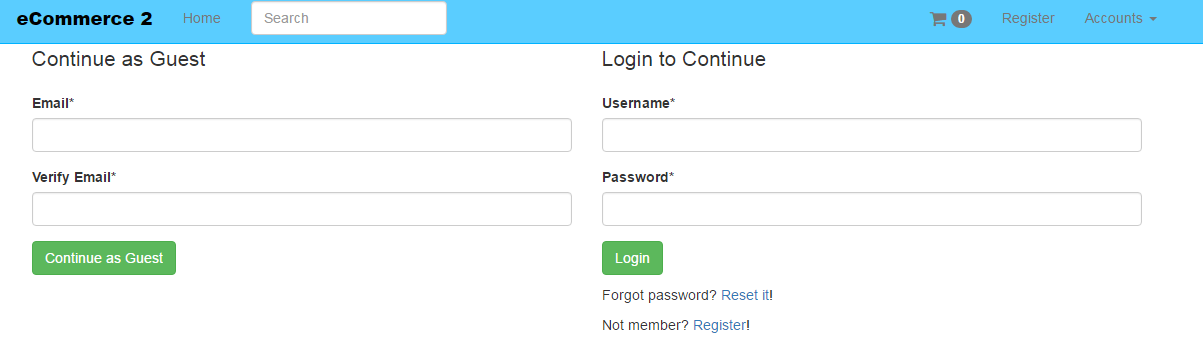
**+** **if** user\_exists **!=** 0**:**

**+** **raise** forms**.**ValidationError**(**"This User already exists. Please login instead."**)**

**return** email2

**else:**

**raise** forms**.**ValidationError**(**"Please confirm emails are the same"**)**



# 094 User Address Model

# modified: carts/views.py

# modified: orders/admin.py

# modified: orders/models.py

-------------------------------- carts/views.py --------------------------------

**class** **CheckoutView(**FormMixin**,** DetailView**):**

**def** get\_context\_data**(**self**,** **\***args**,** **\*\***kwargs**):**

**+** **if** self**.**request**.**user**.**is\_authenticated**():**

**+** user\_checkout**,** created **=** UserCheckout**.**objects**.**get\_or\_create**(**email**=**self**.**request**.**user**.**email**)**

**+** user\_checkout**.**user **=** self**.**request**.**user

**+** user\_checkout**.**save**()**

**+** self**.**request**.**session**[**"user\_checkout\_id"**]** **=** user\_checkout**.**id

**+**

context**[**"user\_can\_continue"**]** **=** user\_can\_continue

context**[**"form"**]** **=** self**.**get\_form**()**

**-------------------------------** orders**/**admin**.**py **-------------------------------**

**from** django**.**contrib **import** admin

# Register your models here.

**-from** **.**models **import** UserCheckout

**+from** **.**models **import** UserCheckout**,** UserAddress

**+**admin**.**site**.**register**(**UserAddress**)**

**-------------------------------** orders**/**models**.**py **-------------------------------**

**+**ADDRESS\_TYPE **=** **(**

**+** **(**'billing'**,** 'Billing'**),**

**+** **(**'shipping'**,** 'Shipping'**),**

**+)**

**+**

**+class** **UserAddress(**models**.**Model**):**

**+** user **=** models**.**ForeignKey**(**UserCheckout**)**

**+** type **=** models**.**CharField**(**max\_length**=**120**,** choices**=**ADDRESS\_TYPE**)**

**+** street **=** models**.**CharField**(**max\_length**=**120**)**

**+** city **=** models**.**CharField**(**max\_length**=**120**)**

**+** state **=** models**.**CharField**(**max\_length**=**120**)**

**+** zipcode **=** models**.**CharField**(**max\_length**=**120**)**

**+**

**+** **def** \_\_unicode\_\_**(**self**):**

**+** **return** self**.**street

# 095 User Address View

<https://docs.djangoproject.com/en/1.8/ref/class-based-views/generic-editing/#formview>

A view that displays a form. On error, redisplays the form with validation errors; on success, redirects to a new URL.

# modified: ecommerce2/urls.py

# modified: orders/forms.py

# modified: orders/views.py

# orders/templates/orders/address\_select.html

**-------------------------------** orders**/**forms**.**py **-------------------------------**

**+class** **AddressForm(**forms**.**Form**):**

**+** billing\_address **=** forms**.**CharField**()**

**+** shipping\_address **=** forms**.**CharField**()**

**-------------------------------** orders**/**views**.**py **-------------------------------**

**+from** django**.**views**.**generic**.**edit **import** FormView

**+from** **.**forms **import** AddressForm

**+**

# Create your views here.

**+class** **AddressSelectFormView(**FormView**):**

**+** form\_class **=** AddressForm

**+** template\_name **=** "orders/address\_select.html"

**------------------------------** ecommerce2**/**urls**.**py **------------------------------**

**+from** orders**.**views **import** AddressSelectFormView

urlpatterns **=** **[**

**+** url**(**r'^checkout/address/$'**,** AddressSelectFormView**.**as\_view**(),** name**=**'order\_address'**),**

**]**

orders/templates/orders/address\_select.html

**{% extends "base.html" %}**

**{% load crispy\_forms\_tags %}**

**{% block content %}**

**{{ form|crispy }}**

**{% endblock %}**

# 096 Model Choice Field

# modified: orders/forms.py

# modified: orders/templates/orders/address\_select.html

# modified: orders/views.py

**-------------------------------** orders**/**forms**.**py **-------------------------------**

**+from** **.**models **import** UserAddress

**class** **AddressForm(**forms**.**Form**):**

**-** billing\_address **=** forms**.**CharField**()**

**-** shipping\_address **=** forms**.**CharField**()**

**+** billing\_address **=** forms**.**ModelChoiceField**(**

**+** queryset**=**UserAddress**.**objects**.**filter**(**type**=**"billing"**),**

**+** widget **=** forms**.**RadioSelect**,**

**+** empty\_label **=** **None** #not show enmpty

**+** **)**

**+** shipping\_address **=** forms**.**ModelChoiceField**(**

**+** queryset**=**UserAddress**.**objects**.**filter**(**type**=**"shipping"**),**

**+** widget **=** forms**.**RadioSelect**,**

**+** empty\_label **=** **None,**

**+**

**+** **)**

**-------------------------------** orders**/**views**.**py **-------------------------------**

**+from** **.**models **import** UserAddress**,** UserCheckout

# Create your views here.

**class** **AddressSelectFormView(**FormView**):**

form\_class **=** AddressForm

template\_name **=** "orders/address\_select.html"

**+** **def** get\_form**(**self**,** **\***args**,** **\*\***kwargs**):**

**+** form **=** super**(**AddressSelectFormView**,** self**).**get\_form**(\***args**,** **\*\***kwargs**)**

**+**

**+** form**.**fields**[**"billing\_address"**].**queryset **=** UserAddress**.**objects**.**filter**(**

**+** user\_\_email**=**self**.**request**.**user**.**email**,**

**+** type**=**'billing'**,**

**+** **)**

**+** form**.**fields**[**"shipping\_address"**].**queryset **=** UserAddress**.**objects**.**filter**(**

**+** user\_\_email**=**self**.**request**.**user**.**email**,**

**+** type**=**'shipping'**,**

**+** **)**

**+** **print** self**.**request**.**user**.**email

**+** **return** form

**+**

**+** **def** form\_valid**(**self**,** form**,** **\***args**,** **\*\***kwargs**):**

**+** form **=** super**(**AddressSelectFormView**,** self**).**form\_valid**(**form**,** **\***args**,** **\*\***kwargs**)**

**+** **return** form

**+**

**+** **def** get\_success\_url**(**self**,** **\***args**,** **\*\***kwargs**):**

**+** **return** "/checkout/"

或者 success\_url = '/checkout/'

**----------------- orders/templates/orders/address\_select.html -----------------**

**{% block content %}**

**+**<form method=**"POST"** action=**''**>**{% csrf\_token %}**

**{{ form|crispy }}**

**-**

**+**<input class=**'btn btn-default'** type=**'submit'** value=**'Select'**/>

**+**</form>

**{% endblock %}**

# 097 Order Model

创建Order Model

Address提交后回到Order

# modified: carts/views.py

# modified: db.sqlite3

# modified: orders/models.py

# modified: orders/views.py

**--------------------------------** carts**/**views**.**py **--------------------------------**

**-from** orders**.**models **import** UserCheckout

**+from** orders**.**models **import** UserCheckout**,** Order**,** UserAddress

**class** **CheckoutView(**FormMixin**,** DetailView**):**

**+** **def** get**(**self**,** request**,** **\***args**,** **\*\***kwargs**):**

**+** get\_data **=** super**(**CheckoutView**,** self**).**get**(**request**,** **\***args**,** **\*\***kwargs**)**

**+** cart **=** self**.**get\_object**()**

**+** user\_checkout\_id **=** request**.**session**.**get**(**"user\_checkout\_id"**)**

**+**

**+** **if** user\_checkout\_id **!=** **None:**

**+** user\_checkout **=** UserCheckout**.**objects**.**get**(**id**=**user\_checkout\_id**)**

**+** billing\_address\_id **=** self**.**request**.**session**[**"billing\_address"**]**

**+** shipping\_address\_id **=** self**.**request**.**session**[**"shipping\_address"**]**

**+**

**+** **if** billing\_address\_id **==** **None** **or** shipping\_address\_id **==** **None:**

**+** **return** redirect**(**"order\_address"**)**

**+** **else:**

**+** billing\_address **=** UserAddress**.**objects**.**get**(**id**=**billing\_address\_id**)**

**+** shipping\_address **=** UserAddress**.**objects**.**get**(**id**=**shipping\_address\_id**)**

**+**

**+** **try:**

**+** new\_order\_id **=** request**.**session**[**"order\_id"**]**

**+** new\_order **=** Order**.**objects**.**get**(**id**=**new\_order\_id**)**

**+** **except:**

**+** new\_order **=** Order**()**

**+** request**.**session**[**"order\_id"**]** **=** new\_order**.**id

**+**

**+** new\_order**.**cart **=** cart

**+** new\_order**.**user **=** user\_checkout

**+** new\_order**.**billing\_address **=** billing\_address

**+** new\_order**.**shipping\_address **=** shipping\_address

**+** new\_order**.**save**()**

**+** **return** get\_data

**-------------------------------** orders**/**models**.**py **-------------------------------**

**+from** carts**.**models **import** Cart

**+class** **Order(**models**.**Model**):**

**+** cart **=** models**.**ForeignKey**(**Cart**)**

**+** user **=** models**.**ForeignKey**(**UserCheckout**,** null**=True)**

**+** billing\_address **=** models**.**ForeignKey**(**UserAddress**,** related\_name**=**'billing\_address'**,** null**=True)**

**+** shipping\_address **=** models**.**ForeignKey**(**UserAddress**,** related\_name**=**'shipping\_address'**,** null**=True)**

**+** shipping\_total\_price **=** models**.**DecimalField**(**max\_digits**=**50**,** decimal\_places**=**2**,** default**=**5.99**)**

**+** order\_total **=** models**.**DecimalField**(**max\_digits**=**50**,** decimal\_places**=**2**,** **)**

**+**

**+** **def** \_\_unicode\_\_**(**self**):**

**+** **return** str**(**self**.**cart**.**id**)**

**-------------------------------** orders**/**views**.**py **-------------------------------**

**class** **AddressSelectFormView(**FormView**):**

**def** form\_valid**(**self**,** form**,** **\***args**,** **\*\***kwargs**):**

**-** form **=** super**(**AddressSelectFormView**,** self**).**form\_valid**(**form**,** **\***args**,** **\*\***kwargs**)**

**-** **return** form

**+** billing\_address **=** form**.**cleaned\_data**[**"billing\_address"**]**

**+** shipping\_address **=** form**.**cleaned\_data**[**"shipping\_address"**]**

**+** self**.**request**.**session**[**"billing\_address"**]** **=** billing\_address**.**id

**+** self**.**request**.**session**[**"shipping\_address"**]** **=** shipping\_address**.**id

**+** **return** super**(**AddressSelectFormView**,** self**).**form\_valid**(**form**,** **\***args**,** **\*\***kwargs**)**

# 098 Crafting the Order Part 1

解决保存时order\_total为空的问题 （pre\_save）

Cart view中将order获取整理到一个函数

**--------------------------------** carts**/**views**.**py **--------------------------------**

**class** **CheckoutView(**FormMixin**,** DetailView**):**

**+** **def** get\_order**(**self**,** **\***args**,** **\*\***kwargs**):**

**+** cart **=** self**.**get\_object**()**

**+** new\_order\_id **=** self**.**request**.**session**.**get**(**"order\_id"**)**

**+** **if** new\_order\_id **is** **None:**

**+** new\_order **=** Order**.**objects**.**create**(**cart**=**cart**)**

**+** self**.**request**.**session**[**"order\_id"**]** **=** new\_order**.**id

**+** **else:**

**+** new\_order **=** Order**.**objects**.**get**(**id**=**new\_order\_id**)**

**+** **return** new\_order

**def** get**(**self**,** request**,** **\***args**,** **\*\***kwargs**):**

get\_data **=** super**(**CheckoutView**,** self**).**get**(**request**,** **\***args**,** **\*\***kwargs**)**

cart **=** self**.**get\_object**()**

**+** new\_order **=** self**.**get\_order**()**

user\_checkout\_id **=** request**.**session**.**get**(**"user\_checkout\_id"**)**

**-** **try:**

**-** new\_order\_id **=** request**.**session**[**"order\_id"**]**

**-** new\_order **=** Order**.**objects**.**get**(**id**=**new\_order\_id**)**

**-** **except:**

**-** new\_order **=** Order**()**

**-** request**.**session**[**"order\_id"**]** **=** new\_order**.**id

**-**

**-** new\_order**.**cart **=** cart

new\_order**.**user **=** user\_checkout

new\_order**.**billing\_address **=** billing\_address

new\_order**.**shipping\_address **=** shipping\_address

**-------------------------------** orders**/**admin**.**py **-------------------------------**

# Register your models here.

**-from** **.**models **import** UserCheckout**,** UserAddress

**+from** **.**models **import** UserCheckout**,** UserAddress**,** Order

**+**admin**.**site**.**register**(**Order**)**

**-------------------------------** orders**/**models**.**py **-------------------------------**

**+from** decimal **import** Decimal

**+from** django**.**db**.**models**.**signals **import** pre\_save**,** post\_save

**+def** order\_pre\_save**(**sender**,** instance**,** **\***args**,** **\*\***kwargs**):**

**+** shipping\_total\_price **=** instance**.**shipping\_total\_price

**+** cart\_total **=** instance**.**cart**.**total

**+** order\_total **=** Decimal**(**shipping\_total\_price**)** **+** Decimal**(**cart\_total**)**

**+** instance**.**order\_total **=** order\_total

**-**

**+**pre\_save**.**connect**(**order\_pre\_save**,** sender**=**Order**)**

# 099 Crafting the Order Part 2

Checkout view中显示Order信息

# modified: carts/templates/carts/checkout\_view.html

# modified: carts/views.py

**--------------------------------** carts**/**views**.**py **--------------------------------**

**class** **CheckoutView(**FormMixin**,** DetailView**):**

template\_name **=** "carts/checkout\_view.html"

form\_class **=** GuestCheckoutForm

**+** context**[**"order"**]** **=** self**.**get\_order**()**

context**[**"user\_can\_continue"**]** **=** user\_can\_continue

context**[**"form"**]** **=** self**.**get\_form**()**

**------------------- carts/templates/carts/checkout\_view.html -------------------**

**+{% else %}**

**+**<div class=**'col-sm-4 col-sm-offset-4 text-center'**>

**+**<p class=**'lead'**>

**+Your Order**</p>

**+**

**+**<table class=**'table'**>

**+** <tr><td>**{{ order.cart.cartitem\_set.count }} Item(s):** <br/>**{% for i in order.cart.cartitem\_set.all %}**

**+** <b>**{{ i.item.get\_title }}**</b><br/>

**+ {% endfor %}**</td></tr>

**+** <tr><td>**Cart Subtotal: {{ order.cart.subtotal }}**</td></tr>

**+** <tr><td>**Tax: {{ order.cart.tax\_total }}**</td></tr>

**+** <tr><td>**Items Total: {{ order.cart.total }}**</td></tr>

**+**

**+** <tr><td>**Shipping: {{ order.shipping\_total\_price }}**</td></tr>

**+** <tr><td>**Order Total: {{ order.order\_total }}**</td></tr>

**+**</table>

**+**</div>

**{% endif %}**

# 100 Crafting the Order Part 3

A view that displays a form for creating an object, redisplaying the form with validation errors (if there are any) and saving the object.

<https://docs.djangoproject.com/en/1.8/ref/class-based-views/generic-editing/#createview>

A view that displays a form. On error, redisplays the form with validation errors; on success, redirects to a new URL.

<https://docs.djangoproject.com/en/1.8/ref/class-based-views/generic-editing/#formview>

解决Guest email为空的问题，因为Guest是没有User的 (user\_\_email**=**self**.**request**.**user**.**email -> user=user\_checkout)

Guest要解决地址不全的问题，比如shipping地址为空 (dispatch)

添加地址功能

**dispatch(request, \*args, \*\*kwargs)**[**¶**](https://docs.djangoproject.com/en/1.8/ref/class-based-views/base/#django.views.generic.base.View.dispatch)

The **view** part of the view – the method that accepts a **request** argument plus arguments, and returns a HTTP response.

The default implementation will inspect the HTTP method and attempt to delegate to a method that matches the HTTP method; a **GET** will be delegated to **get()**, a **POST** to **post()**, and so on.

By default, a **HEAD** request will be delegated to **get()**. If you need to handle **HEAD** requests in a different way than **GET**, you can override the **head()** method. See [Supporting other HTTP methods](https://docs.djangoproject.com/en/1.8/topics/class-based-views/#supporting-other-http-methods) for an example.

------------------------------ ecommerce2/urls.py ------------------------------

**-from** orders**.**views **import** AddressSelectFormView

**+from** orders**.**views **import** AddressSelectFormView**,** UserAddressCreateView

urlpatterns **=** **[**

**+** url**(**r'^checkout/address/add/$'**,** UserAddressCreateView**.**as\_view**(),** name**=**'user\_address\_create'**),**

**]**

------------------------------- orders/forms.py -------------------------------

**+class** **UserAddressForm(**forms**.**ModelForm**):**

**+** **class** **Meta:**

**+** model **=** UserAddress

**+** fields **=** **[**

**+** 'street'**,**

**+** 'city'**,**

**+** 'state'**,**

**+** 'zipcode'**,**

**+** 'type'

**+** **]**

------------------------------- orders/views.py -------------------------------

**+from** django**.**contrib **import** messages

**-from** django**.**shortcuts **import** render

**+from** django**.**shortcuts **import** render**,** redirect

**-from** django**.**views**.**generic**.**edit **import** FormView

**+from** django**.**views**.**generic**.**edit **import** FormView**,** CreateView

**-from** **.**forms **import** AddressForm

**+from** **.**forms **import** AddressForm**,** UserAddressForm

# Create your views here.

**+class** **UserAddressCreateView(**CreateView**):**

**+** form\_class **=** UserAddressForm

**+** template\_name **=** "forms.html"

**+** success\_url **=** "/checkout/address/"

**+**

**+** **def** get\_checkout\_user**(**self**):**

**+** user\_check\_id **=** self**.**request**.**session**.**get**(**"user\_checkout\_id"**)**

**+** user\_checkout **=** UserCheckout**.**objects**.**get**(**id**=**user\_check\_id**)**

**+** **return** user\_checkout

**+**

**+** **def** form\_valid**(**self**,** form**,** **\***args**,** **\*\***kwargs**):**

**+** form**.**instance**.**user **=** self**.**get\_checkout\_user**()**

**+** **return** super**(**UserAddressCreateView**,** self**).**form\_valid**(**form**,** **\***args**,** **\*\***kwargs**)**

**class** **AddressSelectFormView(**FormView**):**

form\_class **=** AddressForm

template\_name **=** "orders/address\_select.html"

**+** **def** dispatch**(**self**,** **\***args**,** **\*\***kwargs**):**

**+** b\_address**,** s\_address **=** self**.**get\_addresses**()**

**+** **if** b\_address**.**count**()** **==** 0**:**

**+** messages**.**success**(**self**.**request**,** "Please add a billing address before continuing"**)**

**+** **return** redirect**(**"user\_address\_create"**)**

**+** **elif** s\_address**.**count**()** **==** 0**:**

**+** messages**.**success**(**self**.**request**,** "Please add a shipping address before continuing"**)**

**+** **return** redirect**(**"user\_address\_create"**)**

**+** **else:**

**+** **return** super**(**AddressSelectFormView**,** self**).**dispatch**(\***args**,** **\*\***kwargs**)**

**+** **def** get\_addresses**(**self**,** **\***args**,** **\*\***kwargs**):**

**+** user\_check\_id **=** self**.**request**.**session**.**get**(**"user\_checkout\_id"**)**

**+** user\_checkout **=** UserCheckout**.**objects**.**get**(**id**=**user\_check\_id**)**

**+** b\_address **=** UserAddress**.**objects**.**filter**(**

**+** user**=**user\_checkout**,**

+ type**=**'billing'**,**

+ **)**

**+** s\_address **=** UserAddress**.**objects**.**filter**(**

**+** user**=**user\_checkout**,**

+ type**=**'shipping'**,**

+ **)**

**+** **return** b\_address**,** s\_address

**def** get\_form**(**self**,** **\***args**,** **\*\***kwargs**):**

form **=** super**(**AddressSelectFormView**,** self**).**get\_form**(\***args**,** **\*\***kwargs**)**

**-** form**.**fields**[**"billing\_address"**].**queryset **=** UserAddress**.**objects**.**filter**(**

**-** user\_\_email**=**self**.**request**.**user**.**email**,**

- type**=**'billing'**,**

- **)**

**-** form**.**fields**[**"shipping\_address"**].**queryset **=** UserAddress**.**objects**.**filter**(**

**-** user\_\_email**=**self**.**request**.**user**.**email**,**

- type**=**'shipping'**,**

- **)**

**+** b\_address**,** s\_address **=** self**.**get\_addresses**()**

**+**

**+** # if b\_address.count() == 0 or s\_address.count()==0:

**+** # return redirect("user\_address\_create")

**+** form**.**fields**[**"billing\_address"**].**queryset **=** b\_address

**+** form**.**fields**[**"shipping\_address"**].**queryset **=** s\_address

**+**

**return** form

# 101 Crafting the Order Part 4

解决问题：Guest用户时，submit之后无法显示checkout 🡪 修正user\_can\_continue

创建user address view

# modified: carts/templates/carts/checkout\_view.html

# modified: carts/templates/carts/view.html

# modified: carts/views.py

# modified: orders/models.py

# modified: orders/templates/orders/address\_select.html

-------------------------------- carts/views.py --------------------------------

**class** **CheckoutView(**FormMixin**,** DetailView**):**

context **=** super**(**CheckoutView**,** self**).**get\_context\_data**(\***args**,** **\*\***kwargs**)**

user\_can\_continue **=** **False**

user\_check\_id **=** self**.**request**.**session**.**get**(**"user\_checkout\_id"**)**

**-** **if** **not** self**.**request**.**user**.**is\_authenticated**()** **or** user\_check\_id **==** **None:** #or request.user.is\_guest:

**-** context**[**"login\_form"**]** **=** AuthenticationForm**()**

**-** context**[**"next\_url"**]** **=** self**.**request**.**build\_absolute\_uri**()**

**-** **elif** self**.**request**.**user**.**is\_authenticated**()** **or** user\_check\_id **!=** **None:**

**-** user\_can\_continue **=** **True**

**-** **else:**

**-** **pass**

**if** self**.**request**.**user**.**is\_authenticated**():**

**+** user\_can\_continue **=** **True**

user\_checkout**,** created **=** UserCheckout**.**objects**.**get\_or\_create**(**email**=**self**.**request**.**user**.**email**)**

user\_checkout**.**user **=** self**.**request**.**user

user\_checkout**.**save**()**

**+** **elif** **not** self**.**request**.**user**.**is\_authenticated**()** **or** user\_check\_id **==** **None:** #or request.user.is\_guest:

**+** context**[**"login\_form"**]** **=** AuthenticationForm**()**

**+** context**[**"next\_url"**]** **=** self**.**request**.**build\_absolute\_uri**()**

**+** **else:**

**+** **pass**

**+**

**+** **if** user\_check\_id **!=** **None:**

**+** user\_can\_continue **=** **True**

context**[**"order"**]** **=** self**.**get\_order**()**

context**[**"user\_can\_continue"**]** **=** user\_can\_continue

**if** user\_checkout\_id **!=** **None:**

user\_checkout **=** UserCheckout**.**objects**.**get**(**id**=**user\_checkout\_id**)**

**-** billing\_address\_id **=** self**.**request**.**session**[**"billing\_address"**]**

**-** shipping\_address\_id **=** self**.**request**.**session**[**"shipping\_address"**]**

**+** billing\_address\_id **=** self**.**request**.**session**.**get**(**"billing\_address"**)**

**+** shipping\_address\_id **=** self**.**request**.**session**.**get**(**"shipping\_address"**)**

**if** billing\_address\_id **==** **None** **or** shipping\_address\_id **==** **None:**

**return** redirect**(**"order\_address"**)**

------------------------------- orders/models.py -------------------------------

**class** **UserAddress(**models**.**Model**):**

**+** **def** get\_address**(**self**):**

**+** **return** "%s, %s, %s %s" **%(**self**.**street**,** self**.**city**,** self**.**state**,** self**.**zipcode**)**

------------------- carts/templates/carts/checkout\_view.html -------------------

**+**<div class=**'col-sm-4 col-sm-offset-4 text-center'**>

**+**<p><b>**Billing to**</b>

**+**<br/>

**+{{ order.billing\_address.get\_address }}**

**+**<p>

**+**<p><b>**Shipping to**</b>

**+**<br/>

**+{{ order.shipping\_address.get\_address }}**

**+**<p>

**+**<br>

**+**<p><b>**Complete Order**</b></p>

**+**

**{% endif %}**

----------------------- carts/templates/carts/view.html -----------------------

**+**<div class=**'col-sm-8 col-sm-offset-2'**>

**+**<h1>**Your cart**</h1>

**+**

<table class=**'table'**>

<tr>

<td colspan=**'4'** class=**'text-right'**>**Total:** <span id=**'carttotal'**>**{{ object.total }}**</span></td>

</tr>

**+**

**+** <tr>

**+** <td colspan=**'4'** class=**'text-right'**><a class=**'btn btn-warning'** href=**"{% url 'checkout' %}"**>**Checkout** </a></td>

**+** </tr>

**+**

</table>

**+**</div>

----------------- orders/templates/orders/address\_select.html -----------------

**{% block content %}**

**+**<div class=**'col-sm-4 col-sm-offset-4 text-center'**>

**+**<p class=**'lead'**>

**+Select your addresses for this order.**

**+**</p>

<form method=**"POST"** action=**''**>**{% csrf\_token %}**

**{{ form|crispy }}**

<input class=**'btn btn-default'** type=**'submit'** value=**'Select'**/>

**+**<a class=**'btn btn-link'** href=**'{% url "user\_address\_create" %}'**>**Add new**</a>

</form>

**+**</div>

**{% endblock %}**

# 102 Custom Mixin for Cart Order

解决选完address回到checkout view不能及时显示的问题 （需要refresh）

# modified: carts/templates/carts/checkout\_view.html

# modified: carts/views.py

# modified: orders/views.py

# orders/mixins.py

------------------- carts/templates/carts/checkout\_view.html -------------------

**{{ order.shipping\_address.get\_address }}**

<p>

**+** <a href=**'{% url "order\_address" %}'**>**Change an Address**</a>

**+**</p>

**+**<br/>

**+**<p>

<br>

<p><b>**Complete Order**</b></p>

-------------------------------- carts/views.py --------------------------------

**+from** orders**.**mixins **import** CartOrderMixin

**-class** **CheckoutView(**FormMixin**,** DetailView**):**

**+class** **CheckoutView(**CartOrderMixin**,** FormMixin**,** DetailView**):**

**-** **def** get\_order**(**self**,** **\***args**,** **\*\***kwargs**):**

**-** cart **=** self**.**get\_object**()**

**-** new\_order\_id **=** self**.**request**.**session**.**get**(**"order\_id"**)**

**-** **if** new\_order\_id **is** **None:**

**-** new\_order **=** Order**.**objects**.**create**(**cart**=**cart**)**

**-** self**.**request**.**session**[**"order\_id"**]** **=** new\_order**.**id

**-** **else:**

**-** new\_order **=** Order**.**objects**.**get**(**id**=**new\_order\_id**)**

**-** **return** new\_order

**def** get\_object**(**self**,** **\***args**,** **\*\***kwargs**):**

**-** cart\_id **=** self**.**request**.**session**.**get**(**"cart\_id"**)**

**-** **if** cart\_id **==** **None:**

**-** **return** **None**

**-** cart **=** Cart**.**objects**.**get**(**id**=**cart\_id**)**

**+** cart **=** self**.**get\_cart**()**

**+** **if** cart **==** **None:**

**+** **return** **None**

**return** cart

**def** get**(**self**,** request**,** **\***args**,** **\*\***kwargs**):**

get\_data **=** super**(**CheckoutView**,** self**).**get**(**request**,** **\***args**,** **\*\***kwargs**)**

cart **=** self**.**get\_object**()**

**+** **if** cart **==** **None:**

**+** **return** redirect**(**"cart"**)**

new\_order **=** self**.**get\_order**()**

user\_checkout\_id **=** request**.**session**.**get**(**"user\_checkout\_id"**)**

**if** user\_checkout\_id **!=** **None:**

user\_checkout **=** UserCheckout**.**objects**.**get**(**id**=**user\_checkout\_id**)**

**-** billing\_address\_id **=** self**.**request**.**session**.**get**(**"billing\_address"**)**

**-** shipping\_address\_id **=** self**.**request**.**session**.**get**(**"shipping\_address"**)**

**-**

**-** **if** billing\_address\_id **==** **None** **or** shipping\_address\_id **==** **None:**

**-** **return** redirect**(**"order\_address"**)**

**-** **else:**

**-** billing\_address **=** UserAddress**.**objects**.**get**(**id**=**billing\_address\_id**)**

**-** shipping\_address **=** UserAddress**.**objects**.**get**(**id**=**shipping\_address\_id**)**

**-**

**+** **if** new\_order**.**billing\_address **==** **None** **or** new\_order**.**shipping\_address **==** **None:**

**+** **return** redirect**(**"order\_address"**)**

new\_order**.**user **=** user\_checkout

**-** new\_order**.**billing\_address **=** billing\_address

**-** new\_order**.**shipping\_address **=** shipping\_address

new\_order**.**save**()**

**return** get\_data

orders/mixins.py

**from** carts**.**models **import** Cart

**from** **.**models **import** Order

**class** **CartOrderMixin(**object**):**

**def** get\_order**(**self**,** **\***args**,** **\*\***kwargs**):**

cart **=** self**.**get\_cart**()**

**if** cart **is** **None:**

**return** **None**

new\_order\_id **=** self**.**request**.**session**.**get**(**"order\_id"**)**

**if** new\_order\_id **is** **None:**

new\_order **=** Order**.**objects**.**create**(**cart**=**cart**)**

self**.**request**.**session**[**"order\_id"**]** **=** new\_order**.**id

**else:**

new\_order **=** Order**.**objects**.**get**(**id**=**new\_order\_id**)**

**return** new\_order

**def** get\_cart**(**self**,** **\***args**,** **\*\***kwargs**):**

cart\_id **=** self**.**request**.**session**.**get**(**"cart\_id"**)**

**if** cart\_id **==** **None:**

**return** **None**

cart **=** Cart**.**objects**.**get**(**id**=**cart\_id**)**

**if** cart**.**items**.**count**()** **<=** 0**:**

**return** **None**

**return** cart

# 103 Checkout Finalize View

# modified: carts/templates/carts/checkout\_view.html

# modified: carts/views.py

# modified: ecommerce2/urls.py

------------------------------ ecommerce2/urls.py ------------------------------

**-from** carts**.**views **import** CartView**,** ItemCountView**,** CheckoutView

**+from** carts**.**views **import** CartView**,** ItemCountView**,** CheckoutView**,** CheckoutFinalView

urlpatterns **=** **[**

**+** url**(**r'^checkout/final/$'**,** CheckoutFinalView**.**as\_view**(),** name**=**'checkout\_final'**),**

**]**

-------------------------------- carts/views.py --------------------------------

**+class** **CheckoutFinalView(**CartOrderMixin**,** View**):**

**+** **def** post**(**self**,** request**,** **\***args**,** **\*\***kwargs**):**

**+** order **=** self**.**get\_order**()**

**+** **if** request**.**POST**.**get**(**"payment\_token"**)** **==** "ABC"**:**

**+** **print** order**.**cart**.**items**.**all**()**

**+** **return** redirect**(**"checkout"**)**

**+**

**+** **def** get**(**self**,** request**,** **\***args**,** **\*\***kwargs**):**

**+** **return** redirect**(**"checkout"**)**

------------------- carts/templates/carts/checkout\_view.html -------------------

**-**<p><b>**Complete Order**</b></p>

**+**<form id=**"checkout"** method=**'POST'** action=**"{% url 'checkout\_final' %}"**>**{% csrf\_token %}**

**+**<input type=**'hidden'** name=**'payment\_token'** value=**'ABC'** />

**+**<p><b><button type=**'submit'**>**Complete Order**</button></b>

**+**</p>

**+**</form>

# 104 Order Status

Complete Order的处理

# modified: carts/views.py

# modified: orders/models.py

-------------------------------- carts/views.py --------------------------------

**class** **CheckoutView(**CartOrderMixin**,** FormMixin**,** DetailView**):**

**def** post**(**self**,** request**,** **\***args**,** **\*\***kwargs**):**

order **=** self**.**get\_order**()**

**if** request**.**POST**.**get**(**"payment\_token"**)** **==** "ABC"**:**

**-** **print** order**.**cart**.**items**.**all**()**

**+** order**.**mark\_completed**()**

**+** **del** request**.**session**[**"cart\_id"**]**

**+** **del** request**.**session**[**"order\_id"**]**

**return** redirect**(**"checkout"**)**

------------------------------- orders/models.py -------------------------------

**+**ORDER\_STATUS\_CHOICES **=** **(**

**+** **(**'created'**,** 'Created'**),**

**+** **(**'completed'**,** 'Completed'**),**

**+)**

**class** **Order(**models**.**Model**):**

**+** status **=** models**.**CharField**(**max\_length**=**120**,** choices**=**ORDER\_STATUS\_CHOICES**,** default**=**'created'**)**

cart **=** models**.**ForeignKey**(**Cart**)**

user **=** models**.**ForeignKey**(**UserCheckout**,** null**=True)**

**+** **def** mark\_completed**(**self**,** order\_id**=None):**

**+** self**.**status **=** "completed"

**+** self**.**save**()**

# 105 Order List

<https://docs.djangoproject.com/en/1.8/ref/class-based-views/generic-display/#listview>

# modified: ecommerce2/urls.py

# modified: orders/views.py

# orders/templates/orders/order\_list.html

------------------------------ ecommerce2/urls.py ------------------------------

**-from** orders**.**views **import** AddressSelectFormView**,** UserAddressCreateView

**+from** orders**.**views **import** AddressSelectFormView**,** UserAddressCreateView**,** OrderList

urlpatterns **=** **[**

url**(**r'^categories/'**,** include**(**'products.urls\_categories'**)),**

**+** url**(**r'^orders/$'**,** OrderList**.**as\_view**(),** name**=**'orders'**),**

------------------------------- orders/views.py -------------------------------

**+from** django**.**views**.**generic**.**list **import** ListView

**-from** **.**models **import** UserAddress**,** UserCheckout

**+from** **.**models **import** UserAddress**,** UserCheckout**,** Order

**+class** **OrderList(**ListView**):**

**+** queryset **=** Order**.**objects**.**all**()**

**+**

**+** **def** get\_queryset**(**self**):**

**+** user\_check\_id **=** self**.**request**.**session**.**get**(**"user\_checkout\_id"**)**

**+** user\_checkout **=** UserCheckout**.**objects**.**get**(**id**=**user\_check\_id**)**

**+** **return** super**(**OrderList**,** self**).**get\_queryset**().**filter**(**user**=**user\_checkout**)**

orders/templates/orders/order\_list.html

**{% extends "base.html" %}**

**{% block content %}**

**{% for object in object\_list %}**

<li>

**{{ object.id }}**

**{{ object.cart.items.all }}**

</li>

**{% endfor %}**

**{% endblock %}**

# 106 Login Required Mixin

# modified: orders/mixins.py

# modified: orders/views.py

------------------------------- orders/mixins.py -------------------------------

**+from** django**.**contrib**.**auth**.**decorators **import** login\_required

**+from** django**.**utils**.**decorators **import** method\_decorator

**+**

**+**

**+class** **LoginRequiredMixin(**object**):**

**+** @method\_decorator**(**login\_required**)**

**+** **def** dispatch**(**self**,** request**,** **\***args**,** **\*\***kwargs**):**

**+** **return** super**(**LoginRequiredMixin**,** self**).**dispatch**(**request**,\***args**,** **\*\***kwargs**)**

------------------------------- orders/views.py -------------------------------

**-from** **.**mixins **import** CartOrderMixin

**+from** **.**mixins **import** CartOrderMixin**,** LoginRequiredMixin

**-class** **OrderList(**ListView**):**

**+class** **OrderList(**LoginRequiredMixin**,** ListView**):**

queryset **=** Order**.**objects**.**all**()**

**def** get\_queryset**(**self**):**

**-** user\_check\_id **=** self**.**request**.**session**.**get**(**"user\_checkout\_id"**)**

**+** user\_check\_id **=** self**.**request**.**user**.**id

user\_checkout **=** UserCheckout**.**objects**.**get**(**id**=**user\_check\_id**)**

**return** super**(**OrderList**,** self**).**get\_queryset**().**filter**(**user**=**user\_checkout**)**

# 107 Order Detail

# modified: carts/templates/carts/checkout\_view.html

# modified: carts/views.py

# modified: ecommerce2/urls.py

# modified: orders/views.py

#

# orders/templates/orders/order\_detail.html

# orders/templates/orders/order\_summary\_short.html

------------------- carts/templates/carts/checkout\_view.html -------------------

<div class=**'col-sm-4 col-sm-offset-4 text-center'**>

**-**<p class=**'lead'**>

**-Order Summary**</p>

**-**<table class=**'table'**>

**-** <tr><td>**{{ order.cart.cartitem\_set.count }} Item(s):** <br/>**{% for i in order.cart.cartitem\_set.all %}**

**-** <b>**{{ i.item.get\_title }}**</b><br/>

**- {% endfor %}**</td></tr>

**-** <tr><td>**Cart Subtotal: {{ order.cart.subtotal }}**</td></tr>

**-** <tr><td>**Tax: {{ order.cart.tax\_total }}**</td></tr>

**-** <tr><td>**Items Total: {{ order.cart.total }}**</td></tr>

**-** <tr><td>**Shipping: {{ order.shipping\_total\_price }}**</td></tr>

**-** <tr><td>**Order Total: {{ order.order\_total }}**</td></tr>

**-**</table>

**-**</div>

**-**

**-**<div class=**'col-sm-4 col-sm-offset-4 text-center'**>

**-**

**-**<p><b>**Billing to**</b>

**-**<br/>

**-{{ order.billing\_address.get\_address }}**

**-**<p>

**-**<p><b>**Shipping to**</b>

**-**<br/>

**-{{ order.shipping\_address.get\_address }}**

**+{% include "orders/order\_summary\_short.html" with order=order %}**

-------------------------------- carts/views.py --------------------------------

**+from** django**.**contrib **import** messages

**class** **CheckoutFinalView(**CartOrderMixin**,** View**):**

order **=** self**.**get\_order**()**

**if** request**.**POST**.**get**(**"payment\_token"**)** **==** "ABC"**:**

order**.**mark\_completed**()**

**+** messages**.**success**(**request**,** "Thank you for your order."**)**

**del** request**.**session**[**"cart\_id"**]**

**del** request**.**session**[**"order\_id"**]**

**-** **return** redirect**(**"checkout"**)**

**+** **return** redirect**(**"order\_detail"**,** pk**=**order**.**pk**)**

------------------------------ ecommerce2/urls.py ------------------------------

**-from** orders**.**views **import** AddressSelectFormView**,** UserAddressCreateView**,** OrderList

**+from** orders**.**views **import** AddressSelectFormView**,** UserAddressCreateView**,** OrderList**,** OrderDetail

urlpatterns **=** **[**

**+** url**(**r'^orders/(?P<pk>\d+)/$'**,** OrderDetail**.**as\_view**(),** name**=**'order\_detail'**),**

------------------------------- orders/views.py -------------------------------

**+class** **OrderDetail(**DetailView**):**

**+** model **=** Order

**+**

**+** **def** dispatch**(**self**,** request**,** **\***args**,** **\*\***kwargs**):**

**+** **try:**

**+** user\_check\_id **=** self**.**request**.**session**.**get**(**"user\_checkout\_id"**)**

**+** user\_checkout **=** UserCheckout**.**objects**.**get**(**id**=**user\_check\_id**)**

**+** **except** UserCheckout**.**DoesNotExist**:**

**+** user\_checkout **=** UserCheckout**.**objects**.**get**(**user**=**request**.**user**)**

**+** **except:**

**+** user\_checkout **=** **None**

**+**

**+** obj **=** self**.**get\_object**()**

**+** **if** obj**.**user **==** user\_checkout **and** user\_checkout **is** **not** **None:**

**+** **return** super**(**OrderDetail**,** self**).**dispatch**(**request**,** **\***args**,** **\*\***kwargs**)**

**+** **else:**

**+** **raise** Http404

orders/templates/orders/order\_detail.html

**{% extends "base.html" %}**

**{% block content %}**

**{% include "orders/order\_summary\_short.html" with order=order %}**

<p><b>**Please print for your records**</b></p>

**{% endblock %}**

orders/templates/orders/order\_summary\_short.html

从checkout\_view.html移到这儿

# 111 UI Updates

# modified: orders/models.py

# modified: orders/templates/orders/order\_list.html

# modified: orders/views.py

# modified: templates/navbar.html

------------------------------- orders/models.py -------------------------------

**+from** django**.**core**.**urlresolvers **import** reverse

**class** **Order(**models**.**Model**):**

**+** **class** **Meta:**

**+** ordering **=** **[**'-id'**]**

**+**

**def** mark\_completed**(**self**,** order\_id**=None):**

self**.**status **=** "completed"

self**.**save**()**

**+** **def** get\_absolute\_url**(**self**):**

**+** **return** reverse**(**"order\_detail"**,** kwargs**={**"pk"**:** self**.**pk**})**

------------------- orders/templates/orders/order\_list.html -------------------

**{% block content %}**

**-{% for object in object\_list %}**

**-**<li>

**- {{ object.id }}**

**- {{ object.cart.items.all }}**

**-**</li>

**-{% endfor %}**

**+{% if object\_list|length** <= 0 %}

+ <div class=**'col-sm-6 col-sm-offset-3 text-center'**>

**+** <p class=**'lead'**>

**+ You do not currently have orders.**

**+** </p>

**+**

**+** </div>

**+**

**+{% else %}**

**+**

**+** <div class=**'col-sm-6 col-sm-offset-3'**>

**+** <h1>**Your Orders**</h1>

**+** <table class=**'table'**>

**+ {% for object in object\_list %}**

**+** <tr>

**+** <td>

**+** <a href=**'{{ object.get\_absolute\_url }}'**>**View #{% if object.order\_id %}{{ object.order\_id }}{% else %}{{ object.id }}{% endif %}**</a>

**+** </td>

**+** <td>

**+ Item Count: {{ object.cart.items.count }}**

**+** </td>

**+** </tr>

**+ {% endfor %}**

**+**

**+** </table>

**+** </div>

**+{% endif %}**

**{% endblock %}**

---------------------------- templates/navbar.html ----------------------------

**+** <li><a href=**"****{% url 'orders' %}"**>**Orders**</a></li>

# 113 Add On Advanced Filter for Product List

# modified: ecommerce2/settings/base.py

# modified: products/forms.py

# modified: products/templates/products/product\_list.html

# modified: products/views.py

# modified: requirements.txt

# modified: templates/navbar.html

------------------------- ecommerce2/settings/base.py -------------------------

INSTALLED\_APPS **=** **(**

'orders'**,**

'crispy\_forms'**,**

'registration'**,**

**+** 'django\_filters'

**)**

------------------------------ products/forms.py ------------------------------

**-from** **.**models **import** Variation

**+from** **.**models **import** Variation**,** Category

**+**

**+class** **ProductFilterForm(**forms**.**Form**):**

**+** q **=** forms**.**CharField**(**label**=**'Search'**,** required**=False)**

**+** category\_id **=** forms**.**ModelMultipleChoiceField**(**

**+** label**=**'Category'**,**

**+** queryset**=**Category**.**objects**.**all**(),**

**+** widget**=**forms**.**CheckboxSelectMultiple**,**

**+** required**=False)**

**+** # category\_title = forms.ChoiceField(

**+** # label='Category',

**+** # choices=CAT\_CHOICES,

**+** # widget=forms.CheckboxSelectMultiple,

**+** # required=False)

**+** max\_price **=** forms**.**DecimalField**(**decimal\_places**=**2**,** max\_digits**=**12**,** required**=False)**

**+** min\_price **=** forms**.**DecimalField**(**decimal\_places**=**2**,** max\_digits**=**12**,** required**=False)**

------------------------------ products/views.py ------------------------------

**-from** **.**forms **import** VariationInventoryFormSet

**+from** **.**forms **import** VariationInventoryFormSet**,** ProductFilterForm

**+from** django\_filters **import** FilterSet**,** CharFilter**,** NumberFilter

# Create your views here.

**+class** **ProductFilter(**FilterSet**):**

**+** title **=** CharFilter**(**name**=**'title'**,** lookup\_type**=**'icontains'**,** distinct**=True)**

**+** category **=** CharFilter**(**name**=**'categories\_\_title'**,** lookup\_type**=**'icontains'**,** distinct**=True)**

**+** category\_id **=** CharFilter**(**name**=**'categories\_\_id'**,** lookup\_type**=**'icontains'**,** distinct**=True)**

**+** min\_price **=** NumberFilter**(**name**=**'variation\_\_price'**,** lookup\_type**=**'gte'**,** distinct**=True)** # (some\_price\_\_gte=somequery)

**+** max\_price **=** NumberFilter**(**name**=**'variation\_\_price'**,** lookup\_type**=**'lte'**,** distinct**=True)**

**+** **class** **Meta:**

**+** model **=** Product

**+** fields **=** **[**

**+** 'min\_price'**,**

**+** 'max\_price'**,**

**+** 'category'**,**

**+** 'title'**,**

**+** 'description'**,**

**+** **]**

**+**

**+**

**+def** product\_list**(**request**):**

**+** qs **=** Product**.**objects**.**all**()**

**+** ordering **=** request**.**GET**.**get**(**"ordering"**)**

**+** **if** ordering**:**

**+** qs **=** Product**.**objects**.**all**().**order\_by**(**ordering**)**

**+** f **=** ProductFilter**(**request**.**GET**,** queryset**=**qs**)**

**+** **return** render**(**request**,** "products/product\_list.html"**,** **{**"object\_list"**:** f **})**

**+**

**+**

**+class** **FilterMixin(**object**):**

**+** filter\_class **=** **None**

**+** search\_ordering\_param **=** "ordering"

**+**

**+** **def** get\_queryset**(**self**,** **\***args**,** **\*\***kwargs**):**

**+** **try:**

**+** qs **=** super**(**FilterMixin**,** self**).**get\_queryset**(\***args**,** **\*\***kwargs**)**

**+** **return** qs

**+** **except:**

**+** **raise** ImproperlyConfigured**(**"You must have a queryset in order to use the FilterMixin"**)**

**+**

**+** **def** get\_context\_data**(**self**,** **\***args**,** **\*\***kwargs**):**

**+** context **=** super**(**FilterMixin**,** self**).**get\_context\_data**(\***args**,** **\*\***kwargs**)**

**+** qs **=** self**.**get\_queryset**()**

**+** ordering **=** self**.**request**.**GET**.**get**(**self**.**search\_ordering\_param**)**

**+** **if** ordering**:**

**+** qs **=** qs**.**order\_by**(**ordering**)**

**+** filter\_class **=** self**.**filter\_class

**+** **if** filter\_class**:**

**+** f **=** filter\_class**(**self**.**request**.**GET**,** queryset**=**qs**)**

**+** context**[**"object\_list"**]** **=** f

**+** **return** context

**+**

**class** **ProductListView(**FilterMixin**,** ListView**):**

model **=** Product

queryset **=** Product**.**objects**.**all**()**

**-**

**+** filter\_class **=** ProductFilter

**+**

**def** get\_context\_data**(**self**,** **\***args**,** **\*\***kwargs**):**

context **=** super**(**ProductListView**,** self**).**get\_context\_data**(\***args**,** **\*\***kwargs**)**

context**[**"now"**]** **=** timezone**.**now**()**

**+** context**[**"query"**]** **=** self**.**request**.**GET**.**get**(**"q"**)** #None

**+** context**[**"filter\_form"**]** **=** ProductFilterForm**(**data**=**self**.**request**.**GET **or** **None)**

**return** context

------------------------------- requirements.txt -------------------------------

**+**django**-**filter**==**0.15.3

---------------- products/templates/products/product\_list.html ----------------

**{% extends "base.html" %}**

**+{% load crispy\_forms\_tags %}**

**{% block content %}**

**+**<div class=**'col-sm-2'**>

**+** <form method=**"GET"** action=**"{% url 'products' %}"**>

**+ {{ filter\_form|crispy }}**

**+** <input type=**'hidden'** name=**'q'** value=**'{{ request.GET.q }}'** />

**+** <input type=**'submit'** value=**'Apply Filter'** class=**'btn btn-default'**>

**+** </form>

**+**

**+** <a href=**"{% url 'products' %}"**>**Clear Filters**</a>

</div>

**+**<div class=**'col-sm-10'**>

**+**

<h1>**All Products** <small><a href=**"{% url 'categories' %}"**>**Categories**</a></small></h1>

**+ {% if object\_list.count == 0 %}**

**+ Nothing found**

**+ {% else %}**

**{% include 'products/products.html' with object\_list=object\_list %}**

**+ {% endif %}**

**+**

**+**</div>

**{% endblock %}**

---------------------------- templates/navbar.html ----------------------------

<div class=**"form-group"**>

**-** <input type=**"text"** class=**"form-control"** placeholder=**"Search"** name=**"q"**>

**+** <input type=**"text"** class=**"form-control"** placeholder=**"Search"** name=**"q"** value=**'{{ request.GET.q }}'**>

</div>

# 校错

**Order List显示的不是当前用户的Order**

**class** **OrderList(**LoginRequiredMixin**,** ListView**):**

queryset **=** Order**.**objects**.**all**()**

**def** get\_queryset**(**self**):**

user\_check\_id **=** self**.**request**.**user**.**id

user\_checkout **=** UserCheckout**.**objects**.**get**(**id**=**user\_check\_id**)**

**return** super**(**OrderList**,** self**).**get\_queryset**().**filter**(**user**=**user\_checkout**)**

user\_checkout **=** UserCheckout**.**objects**.**get**(**id**=**user\_check\_id**)**

修改为

user\_checkout **=** UserCheckout**.**objects**.**get**(user\_**id**=**user\_check\_id**)**

因为UserCheckout和admin User id并不一致，根据admin User id 获取的用户并不是对应的UserCheckout的用户

**请推荐国内外都能高速访问的 bootstrap 和 jquery 的 CDN**

我一直用的http://www.bootcdn.cn, 貌似是和又拍云合作的项目   
速度挺不错的   
  
你说的几个库都有,试试速度吧:   
<link rel="stylesheet" href="http://cdn.bootcss.com/bootstrap/3.3.0/css/bootstrap.min.css">   
<link rel="stylesheet" href="http://cdn.bootcss.com/font-awesome/4.2.0/css/font-awesome.min.css">   
<script src="http://cdn.bootcss.com/jquery/2.1.3/jquery.min.js"></script>