

# BÀI 1: Template Inheritance



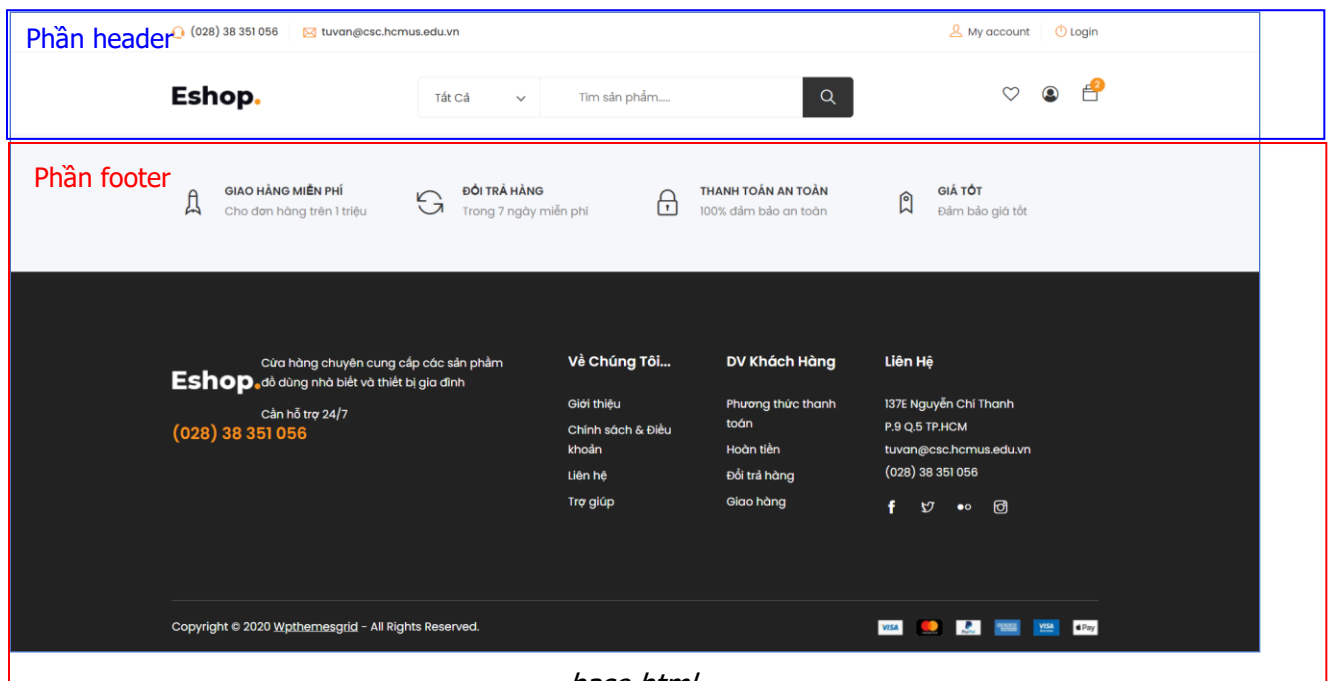
Mục tiêu chính:

- Tạo template inheritance
- Xây dựng ứng dụng web với template inheritance

## 1.1. Xây dựng website chuẩn Django Template/ Template Inheritance từ template mẫu được cung cấp

### ✓ Yêu cầu:

- Cung cấp bộ template **eshop**. Dựa vào bộ template này hãy xây dựng một web project có tên là **MyStore** (Cửa hàng online) > Trong đó tạo một web app có tên là **store**
- Vận dụng các kiến thức của module 2 để xây dựng các web page chuẩn Django template.
- Tạo file **base.html** chứa phần chung của các trang web (header và footer) > Áp dụng template inheritance để đưa phần chung base.html vào các trang và tháo bỏ các phần code thừa ở các trang có kế thừa base.html.
- Với base.html: Sẽ chứa tất cả phần import của các thư viện bootstrap, javascript,... phục vụ cho template của ứng dụng, và phần nội dung chung là header và footer như sau:



*base.html*

- Với các trang html khác, chèn base.html vào và hiệu chỉnh nội dung phù hợp từng trang (phần riêng của mỗi trang là phần khung màu xanh lá cây) như sau:



[\*index.html\*](#)



shop.html

(028) 38 351 056

tuvan@csc.hcmus.edu.vn

My account

Login

Eshop.

Tất Cả







Tìm sản phẩm...

3

Home

Shop

Liên Hệ

SẢN PHẨM	TÊN	ĐƠN GIÁ	SỐ LƯỢNG	THÀNH TIỀN	
	<b>Women Dress</b> Maboriosam in a tonto nesciung eget distingy magndapibus.	\$110.00	<div>- 1 +</div>	\$220.88	
	<b>Women Dress</b> Maboriosam in a tonto nesciung eget distingy magndapibus.	\$110.00	<div>- 2 +</div>	\$220.88	
	<b>Women Dress</b> Maboriosam in a tonto nesciung eget distingy magndapibus.	\$110.00	<div>- 3 +</div>	\$220.88	

Nhập mã giảm giá

ÁP DỤNG

☐ Giao hàng (+20.000đ)

Tạm tính\$330.00


Giao hàngMiễn phí


Tiết kiệm\$20.00


Tổng đơn hàng\$310.00


THANH TOÁN

TIẾP TỤC MUA SẮM

 **GIAO HÀNG MIỄN PHÍ**  
Cho đơn hàng trên 1 triệu

 **ĐỔI TRẢ HÀNG**  
Trong 7 ngày miễn phí

 **THANH TOÁN AN TOÀN**  
100% đảm bảo an toàn

 **GIÁ TỐT**  
Đảm bảo giá tốt

Eshop.

Cửa hàng chuyên cung cấp các sản phẩm đồ dùng nhà bếp và thiết bị gia đình

Cần hỗ trợ 24/7  
(028) 38 351 056

Về Chúng Tôi...

Giới thiệu

Chính sách & Điều khoản

Liên hệ

Trợ giúp

DV Khách Hàng

Phương thức thanh toán

Hoàn tiền

Đổi trả hàng

Giao hàng





Liên Hệ

137E Nguyễn Chí Thanh







P.9 Q.5 TP.HCM

tuvan@csc.hcmus.edu.vn

(028) 38 351 056

Copyright © 2020 Wpthemesgrid - All Rights Reserved.

cart.html





Liên lạc

Gửi Email

Tên \*

Subject \*

Email \*

Phone \*

Nội dung \*

GỬI



Điện thoại:

(028) 38 351 056



Email:

[tuvan@csc.hcmus.edu.vn](mailto:tuvan@csc.hcmus.edu.vn)



Địa chỉ:

602, hẻm Nguyễn Văn Bền, Phường 4,  
TP.HCM



GIAO HÀNG MIỄN PHÍ

Cho đơn hàng trên 1 triệu



ĐỔI TRẢ HÀNG

Trong 7 ngày miễn phí



THANH TOÁN AN TOÀN

100% đảm bảo an toàn



GIÁ TỐT

Đảm bảo giá tốt

Eshop.

Cửa hàng chuyên cung cấp các sản phẩm đồ dùng nhà  
biết và thiết bị gia đình

Cần hỗ trợ 24/7

(028) 38 351 056

Về Chúng Tôi...

Giới thiệu

Chính sách & Điều

khảo

Liên hệ

Trợ giúp

DV Khách Hàng

Phương thức thanh  
toán

Hoàn tiền

Đổi trả hàng

Giao hàng

Liên Hệ

137E Nguyễn Chí Thanh

P.9 Q.5 TP.HCM

[tuvan@csc.hcmus.edu.vn](mailto:tuvan@csc.hcmus.edu.vn)

(028) 38 351 056



[contact.html](#)

### ✓ Hướng dẫn

- Tạo project MyStore
- Trong **MyStore** tạo web app **store**
- Vận dụng kiến thức bài "Mapping URL, Template, Static File, View" đã học ở module 2 để cấu hình và tạo các trang web theo chuẩn Django Template (index.html, shop.html, cart.html, checkout.html, contact.html)
- Trong templates/store của app store tạo trang base.html (có thể copy từ trang contact.html và đổi tên thành base.html). Để lại phần html chung là header và footer. Phần nội dung riêng xóa đi và thay vào đó là phần code (để cho biết đây là phần riêng, mỗi trang sẽ khác nhau):

```
{% block body_block %}
    <!-- Anything outside of this will be inherited if you extend! -->
</block>
{% endblock %}
```

- Mở các trang html đang có, chèn vào đầu trang dòng khai báo sử dụng template inheritance trong base.html, sau đó ở phần block riêng giữ lại phần code riêng của từng trang, phần code đã có trong base.html thì xóa bỏ:

```
{% extends "store/base.html" %}
{% load static %}
{% block body_block %}
    <!-- Code of specific page -->
{% endblock %}
```

- Sau khi thực hiện xong ở tất cả các trang thì chạy thử để kiểm tra.

## 1.2. Tạo các model và import dữ liệu

### ✓ Yêu cầu: Tạo các model cần thiết cho store và nhập/import dữ liệu được cung cấp

- Tạo model **Category** để lưu trữ thông tin ngành hàng, (tương ứng với table **category**). Các thông tin gồm có pk (khóa chính, tự động), name.
- Một Category có thể có một hoặc nhiều SubCategory.
- Tạo model **SubCategory** để lưu trữ thông tin loại hàng, (tương ứng với table **subcategory**). Các thông tin gồm có: pk (khóa chính, tự động), category (khóa ngoại), name, image
- Một SubCategory có thể có một hoặc nhiều Product.
- Tạo model **Product** để lưu trữ thông tin các hàng hóa (tương ứng với table **product**). Các thông tin gồm có: pk (khóa chính, tự động), subcategory (khóa ngoại), name, price, price\_origin, image, content, public\_day (lấy ngày giờ hiện hành của hệ thống), viewed.
- Nhập dữ liệu cho Category và SubCategory (bằng DB Browser SQLite)



- Import dữ liệu **store\_product.csv** vào cho bảng Product (bảng DB Browser SQLite) trong CSDL của ứng dụng hoặc sử dụng database **db.sqlite3** được cung cấp để chọn và tự export bảng cần dùng rồi import vào CSDL của ứng dụng.

#### ✓ Hướng dẫn

- Tiếp tục sử dụng MyStore project ở bài trên.
- Trong webapp store, tạo **models.py**. Trong models.py, lần lượt tạo các model Category, SubCategory và Product:
- Sau khi tạo xong cần **python manage.py makemigrations store** và **python manage.py migrate**

```
models.py > ...
from django.db import models
import datetime
from ckeditor_uploader.fields import RichTextUploadingField

# Create your models here.

class Category(models.Model):
    name = models.CharField(max_length=150, unique=True)

    def __str__(self):
        return self.name

class SubCategory(models.Model):
    category = models.ForeignKey(Category, on_delete=models.PROTECT)
    name = models.CharField(max_length=150, unique=True)
    image = models.ImageField(upload_to="store/images", default="store/images/default.png")

    def __str__(self):
        return self.name

class Product(models.Model):
    subcategory = models.ForeignKey(SubCategory, on_delete=models.PROTECT)
    name = models.CharField(max_length=250)
    price = models.FloatField(default=0.0)
    price_origin = models.FloatField(null=True)
    image = models.ImageField(upload_to="store/images", default="store/images/default.png")
    content = RichTextUploadingField(blank=True, null=True)
    public_day = models.DateTimeField(default=datetime.datetime.now)
    viewed = models.IntegerField(default=0)

    def __str__(self):
        return self.name
```

- Sau khi tạo xong cần **python manage.py makemigrations store** và **python manage.py migrate**
- Tạo và cấu hình cho thư mục **media** để lưu trữ hình ảnh của project MyStore. Trong media tạo hai thư mục là **store** và **uploads**. Hình ảnh được upload của SubCategory và Product sẽ được lưu trữ trong media\store\images (cách tạo và cấu hình cho thư mục media làm tương tự như bài "Django MTV" đã học ở module 2).

- Nhập dữ liệu cho Category và SubCategory (đăng nhập vào admin để nhập liệu) và import dữ liệu cho Product (từ tập tin được cung cấp store\_product.csv)

- Dữ liệu trong Category:

Table: store\_category

	id	name
	Filter	Filter
1	1	Thiết bị gia đình
2	2	Đồ dùng nhà bếp

- Dữ liệu trong SubCategory

Table: store\_subcategory

	id	name	category_id	image
	Filter	Filter	Filter	Filter
1	1	Bàn Ủi - Bàn Là	1	store/images/ban_ui.jpg
2	2	Cây Nước Nóng Lạnh	1	store/images/cay_nuoc_nong_lanh.png
3	3	Máy Hút Bụi	1	store/images/may_hut_bui.jpg
4	4	Quạt Điện	1	store/images/quat_dien.jpg
5	5	Máy Lọc Nước	1	store/images/may_loc_nuoc.jfif
6	6	Bình Thủy Điện	2	store/images/binh_thuy_dien.jpg
7	7	Bếp Các Loại	2	store/images/bep_cac_loai.jpg
8	8	Lò Vi Sóng	2	store/images/lo-vi-song.jpg
9	9	Bình Đun Siêu Tốc	2	store/images/am-dun-sieu-toc-bang-dien-cao-cap.jpg
10	10	Nồi Điện Các Loại	2	store/images/noi_dien.jpg

*Chú ý: Các hình ảnh upload cho SubCategory được cung cấp sẵn trong thư mục store\_images, HV khi tạo mới SubCategory sẽ upload hình ảnh tương ứng như bảng trên.*

- Dữ liệu trong Product: import từ tập tin store\_product.csv, sau đó hiệu chỉnh lại các giá trị nếu cần (ví dụ như có thể thiết lập lại public\_day = ngày giờ hệ thống bằng cách thực thi truy vấn update trong shell)
- Vận dụng kiến thức bài "Lập trình hướng đối tượng và Models" đã học trong module 2 để thực hiện công các việc này.

### 1.3. Hiển thị dữ liệu lên trang index.html

- ✓ **Yêu cầu:** Hiển thị dữ liệu lên trang index.html như sau:

- Danh sách 20 product mới nhất lên Banner, mỗi product kèm theo hình ảnh, tên, giá gốc, giá. Trên hình ảnh và tên là link (href="{% url 'store:product.html' product.pk %}") dẫn tới product tương ứng sẽ hiển ở trang product.html.

- Danh sách subcategory theo từng category, mỗi subcategory kèm theo hình ảnh và tên. Trên hình ảnh và tên là link  
(href="{% url 'store:shop.html' subcategory.id%}") dẫn tới trang shop.html theo subcategory.



## Trang chủ (index.html)

### ✓ Hướng dẫn

- Tiếp tục sử dụng MyStore ở bài tập trên
- Toàn bộ hình ảnh cho ứng dụng được cung cấp trong 2 thư mục là static\_images (đưa vào thư mục static\images của app) và stote\_images (đưa vào thư mục store\images trong media)
- Trong store > views.py > cập nhật function **index()**: đọc dữ liệu từ các table và gửi ra template index.html

```
subcategory_list = models.SubCategory.objects.all()

def index(request):
    tbgd = models.SubCategory.objects.filter(category=1)
    ddnb = models.SubCategory.objects.filter(category=2)
    product_list = models.Product.objects.order_by("-public_day")
    twenty_newest = product_list[:20]

    return render(request, "store/index.html",
                  {
                      'twenty_newest': twenty_newest,
                      'subcategories': subcategory_list,
                      'tbgd': tbgd,
                      'ddnb': ddnb
                  })
```

- Kiểm tra xem đã khai báo url của index() hay chưa, nếu chưa thì khai báo trong urls.py của store.
- Trong template index.html: đọc và hiển thị các nội dung theo yêu cầu.
  - Ví dụ: Hiển thị danh sách 20 product mới nhất:

```

<div class="row">
  <div class="col-12">
    {% if twenty_newest %}
    <div class="owl-carousel popular-slider">
      <!-- Start Single Product -->
      {% for product in twenty_newest %}
      <div class="single-product">
        <div class="product-img">
          <a href="{% url 'store:product.html' product.pk %}">
            
            
            <span class="out-of-stock">New</span>
          </a>
          <div class="button-head">
            <div class="product-action-2">
              <a title="Add to cart" href="#">Chọn mua</a>
            </div>
          </div>
        </div>
        <div class="product-content">
          <h3><a href="{% url 'store:product.html' product.pk %}">{{product.name}}</a></h3>
          <div class="product-price">
            <span class="old">{% load humanize %}{{product.price_origin|floatformat:0|intcomma}} đ</span>
            <span>{% load humanize %}{{product.price|floatformat:0|intcomma}} đ</span>
          </div>
        </div>
      </div>
      {% endfor %}
      <!-- End Single Product -->
    </div>
    {% endif %}
  </div>
</div>

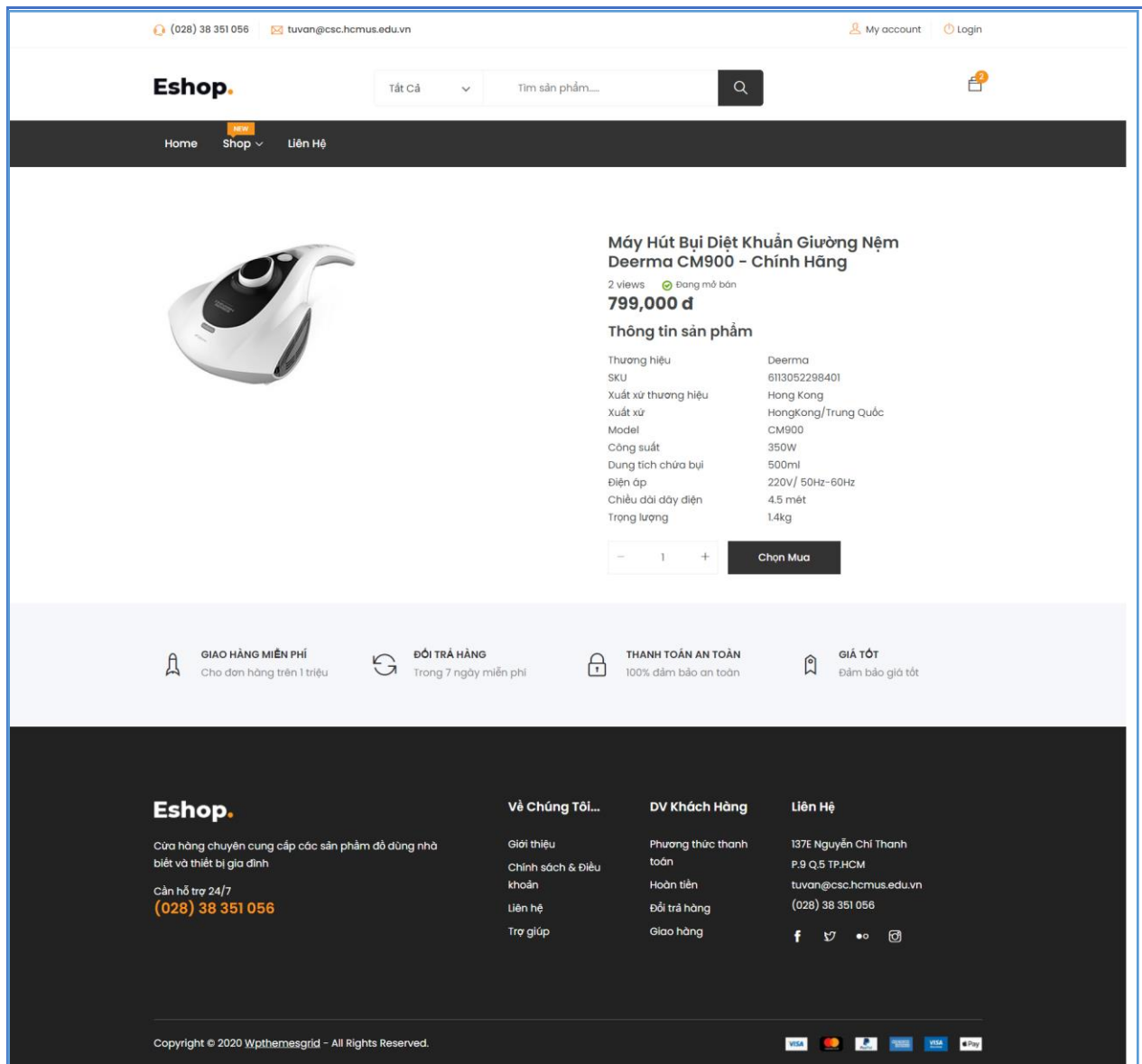
```

...

- Làm tương tự cho các nội dung còn lại
- Chạy thử và xem kết quả

## 1.4. Hiển thị dữ liệu lên trang product.html

- ✓ **Yêu cầu:** Hiển thị dữ liệu lên trang product.html như sau:
  - Từ các trang khác khi người dùng click vào link sản phẩm bất kỳ thì dẫn đến trang product.html và hiển thị thông tin chi tiết của sản phẩm đó.
  - Tăng lượt view cho sản phẩm thêm 1 lần.



### Sản phẩm (product.html)

#### ✓ Hướng dẫn

- Tiếp tục sử dụng MyStore ở bài tập trên
- Trong store > views.py > tạo function **product\_detail()**: đọc dữ liệu từ table Product theo pk và gửi ra store/product.html

```
from django.db.models import Count, F, Value
def product_detail(request, pk):
    product_select = models.Product.objects.get(pk=pk)
    # khi người dùng chọn xem 1 sản phẩm > tăng view của sản phẩm thêm 1
    models.Product.objects.filter(pk=product_select.pk).update(viewed=F('viewed') + 1)
```

```
product_select.refresh_from_db()
```

```
return render(request, "store/product.html",
               {'product': product_select,
                'subcategories': subcategory_list,
               })
```

- Khai báo url của product\_detail() trong urls.py của store
- Trong template product.html: đọc và hiển thị các nội dung theo yêu cầu.

```
<div class="row">
  <div class="col-lg-6 col-md-12 col-sm-12 col-xs-12">
    
  </div>
  <div class="col-lg-6 col-md-12 col-sm-12 col-xs-12">
    <div class="quickview-content">
      <h2>{{product.name}}</h2>
      <div class="quickview-rating-review">
        <div class="quickview-rating-wrap">
          {{product.viewed}} views
        </div>
        <div class="quickview-stock">
          <span><i class="fa fa-check-circle-o"></i> Đang mở bán</span>
        </div>
      </div>
      <h4>{% load humanize %}{{product.price|floatformat:0|intcomma}} đ</h4>
      <h3>Thông tin sản phẩm</h3>
      <div class="quickview-peragraph">
        <p>{{product.content|safe}}</p>
      </div>
      <div class="quantity">
        <!-- Input Order -->
        <div class="input-group">
          <div class="button minus">
            <button type="button" class="btn btn-primary btn-number disabled="disabled"
                  data-type="minus" data-field="quant[1]">
              <i class="ti-minus"></i>
            </button>
          </div>
          <input type="text" name="quant[1]" class="input-number" data-min="1" data-max="1000"
                 value="1">
          <div class="button plus">
            <button type="button" class="btn btn-primary btn-number" data-type="plus"
                  data-field="quant[1]">
              <i class="ti-plus"></i>
            </button>
          </div>
        </div>
        <!-- End Input Order -->
      </div>
      <div class="add-to-cart">
        <a href="#" class="btn">Chọn mua</a>
      </div>
    </div>
  </div>
</div>
```

- Chạy thử và xem kết quả



## 1.5. Hiển thị dữ liệu lên trang shop.html

- ✓ **Yêu cầu:** Hiển thị dữ liệu lên trang shop.html như sau:
  - Danh sách các subcategory hiển thị trên menu trái kèm theo link (`href="{% url 'store:shop.html' subcategory.id%}"`) để khi người dùng nhấn vào sẽ hiển thị danh sách các product của subcategory này ở phần trung tâm của trang.
  - 3 product mới nhất hiển thị trên menu trái (dưới danh sách subcategory và khoảng giá). Trên tên là link (`href="{% url 'store:product.html' product.pk %}"`) dẫn tới product tương ứng sẽ hiển ở trang product.html.
  - Danh sách product theo subcategory sắp giảm dần theo `public_day` (sản phẩm mới hiển thị trước), hiển thị ở phần trung tâm của trang, có phân trang (9 sản phẩm/ 1 trang), mỗi product kèm theo hình ảnh, tên, giá. Trên hình ảnh và tên là link (`href="{% url 'store:product.html' product.pk %}"`) dẫn tới product tương ứng sẽ hiển ở trang product.html.



- Tiếp tục sử dụng MyStore ở bài tập trên.
- Trong store > views.py > tạo function **shop()**: đọc dữ liệu từ các table cần thiết theo pk (là khóa của subcategory) và gửi ra store/shop.html

```
def shop(request, pk):
    product_list = []
    subcategory_name = ''
    if pk != 0:
        product_list = models.Product.objects.filter(
            subcategory=pk).order_by("-public_day")
        selected_sub = models.SubCategory.objects.get(pk = pk)
        subcategory_name = selected_sub.name
    else:
        product_list = models.Product.objects.order_by("-public_day")

    three_newest = product_list[:3]

    page = request.GET.get('page', 1) # trang bat dau
    paginator = Paginator(product_list, 9) # so product/trang

    try:
        products = paginator.page(page)
    except PageNotAnInteger:
        products = paginator.page(1)
    except EmptyPage:
        products = paginator.page(paginator.num_pages)

    return render(request, "store/shop.html",
        {'three_newest': three_newest,
         'subcategories': subcategory_list,
         'products': products,
         'pk': pk,
         'subcategories': subcategory_list,
         'subcategory_name': subcategory_name,
        })
```

- Khai báo url của shop() trong urls.py của store
- Trong template shop.html: đọc và hiển thị các nội dung theo yêu cầu.  
Ví dụ: hiển thị 3 sản phẩm mới nhất

```

<div class="single-widget recent-post">
  <h3 class="title">Hàng mới về</h3>
  {% if three_newest %}
  {%for product in three_newest%}
  <!-- Single Post -->
  <div class="single-post first">
    <div class="image">
      
    </div>
    <div class="content">
      <h5><a href="{% url 'store:product.html' product.pk %}">{{product.name}}</a></h5>
      <p class="price">{% load humanize %}{{product.price|floatformat:0|intcomma}} đ</p>
    </div>
    <!-- End Single Post -->
  </div>
  {% endfor %}
  {% endif %}
</div>

```

- Chạy thử và xem kết quả