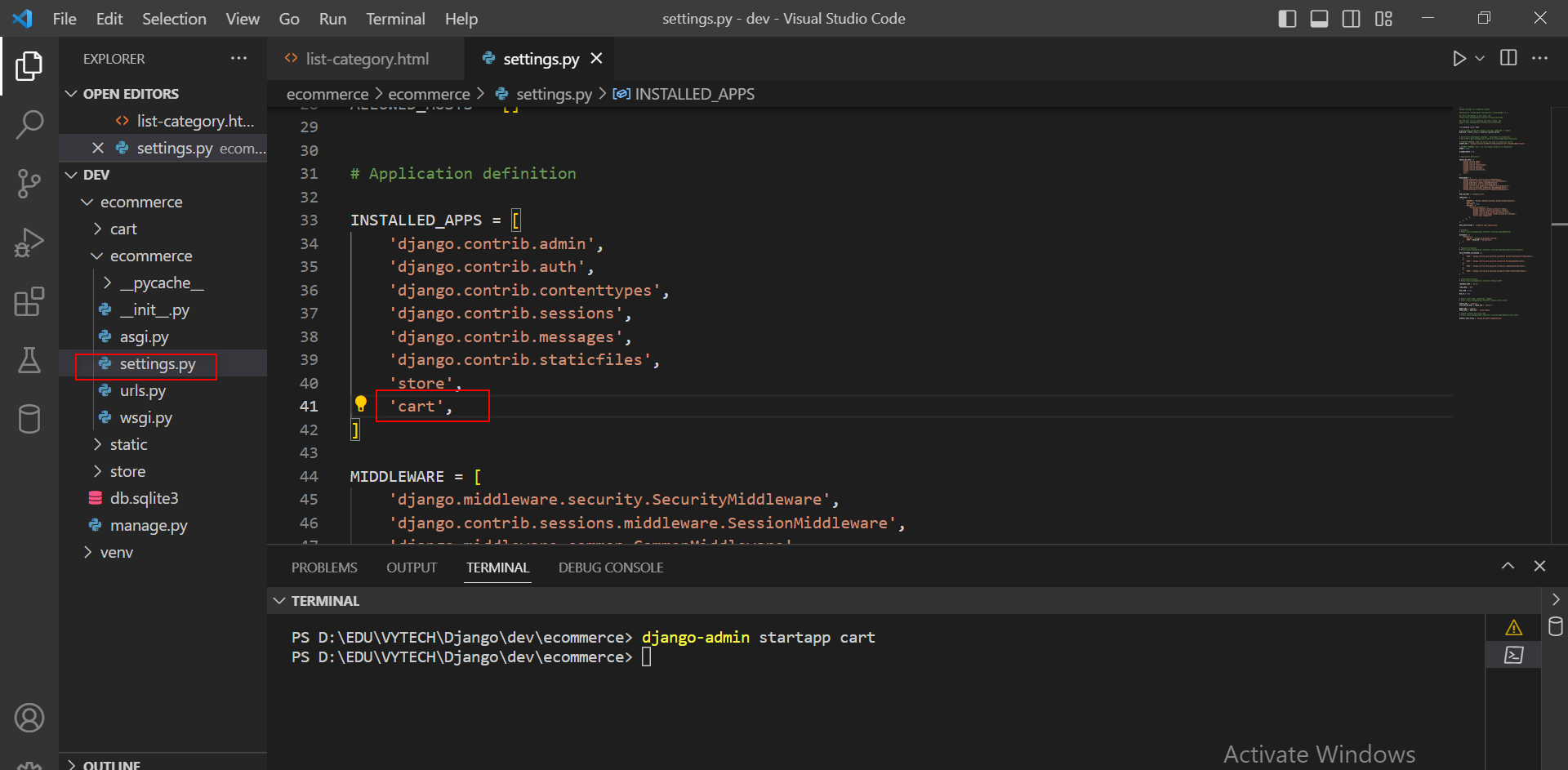
**CART FUNCTIONALITY INITIAL BUILD-UP**

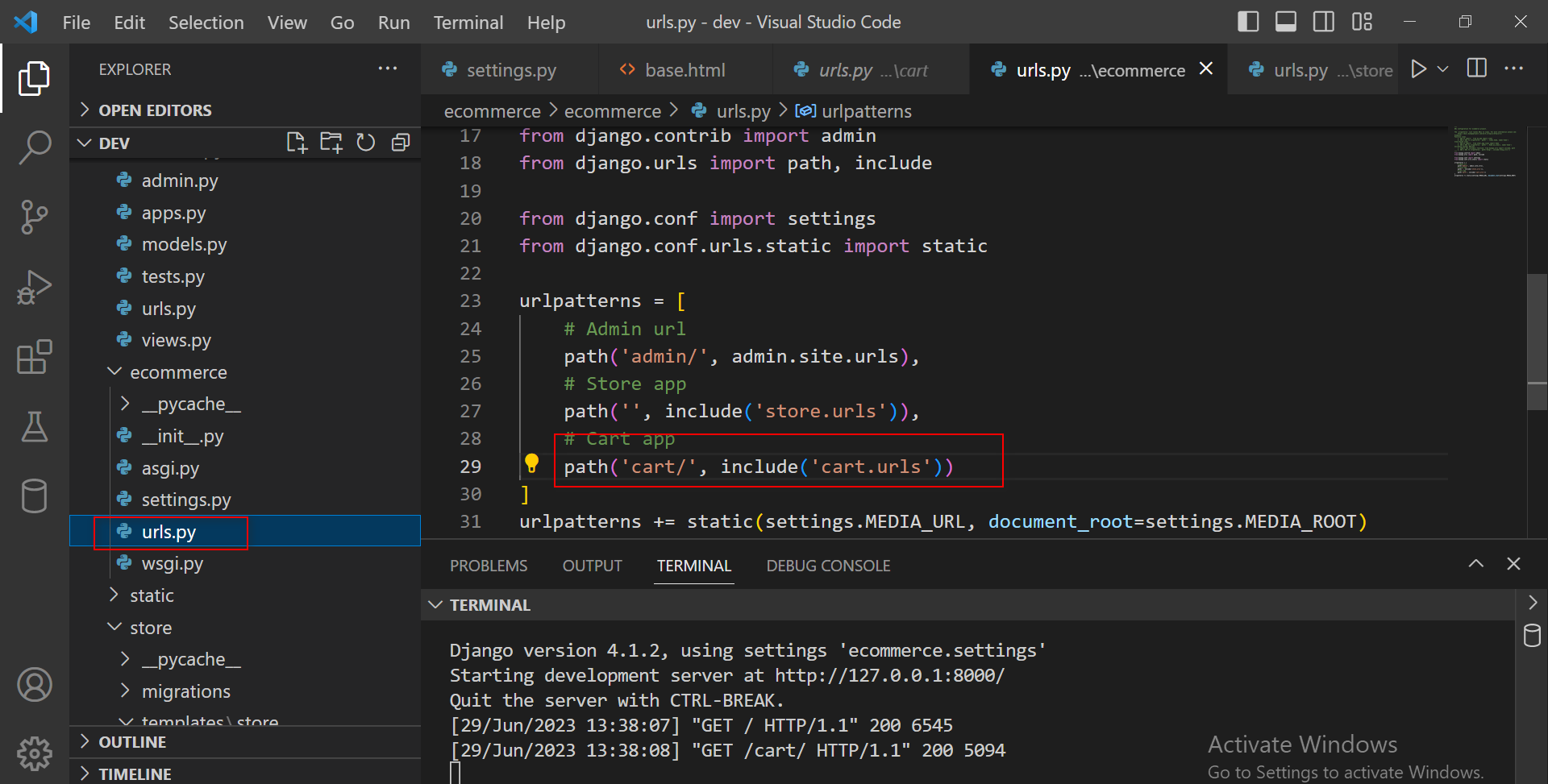
**1. Create Django app cart**



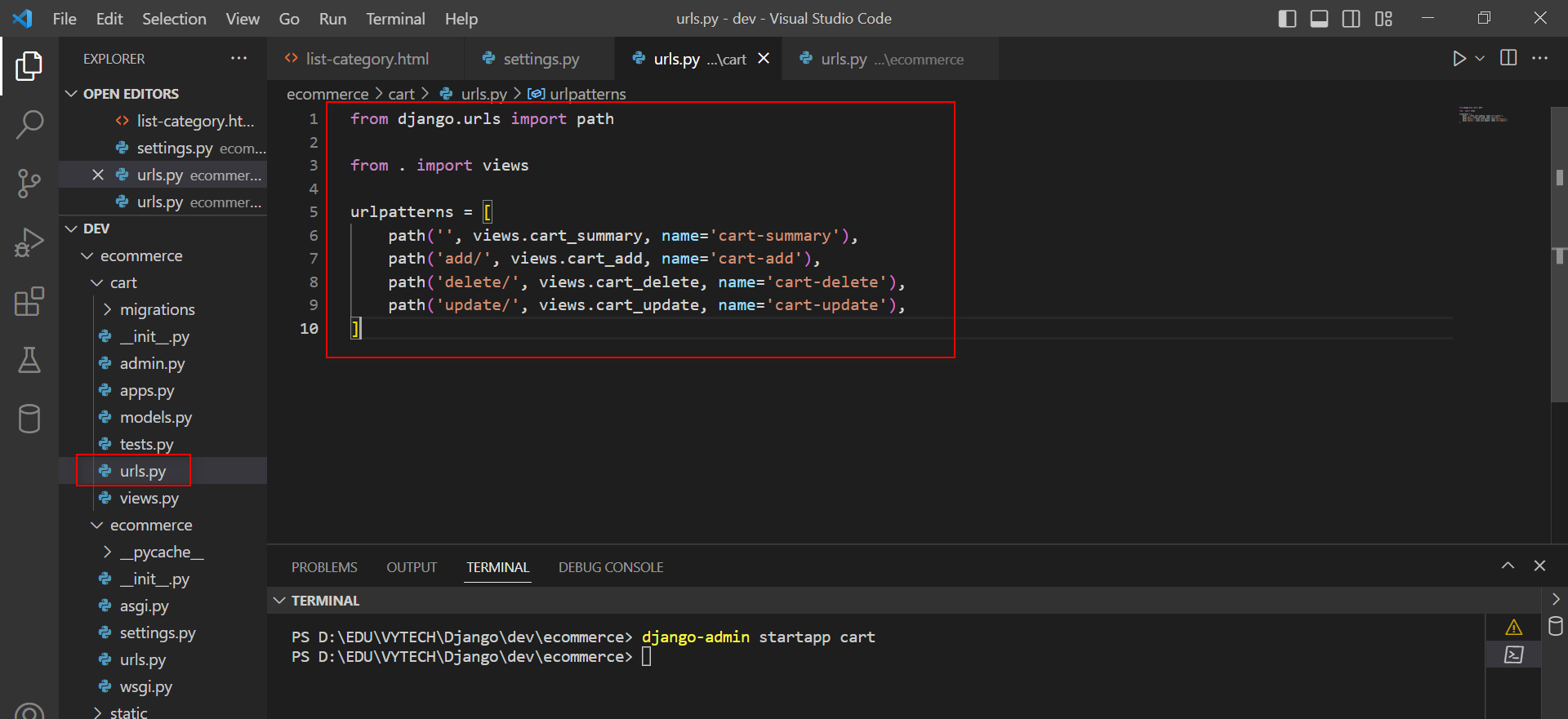
**Add to setting**



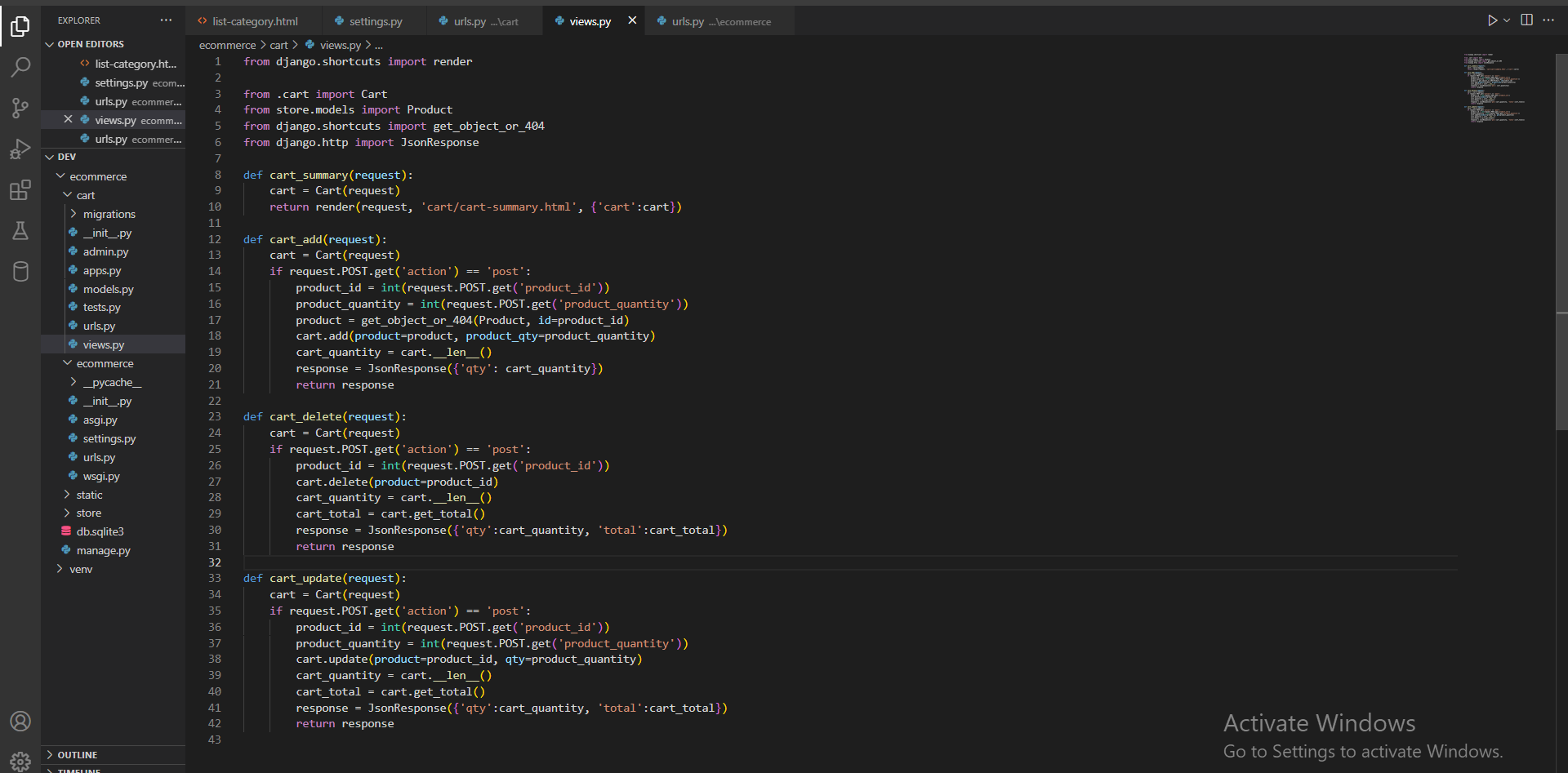
**2. Configurate basic urls and views for ours cart**



**Add content for urls**



**Add content for views**



def cart\_summary(request):

    return render(request, 'cart/cart-summary.html')

def cart\_add(request):

    pass

def cart\_delete(request):

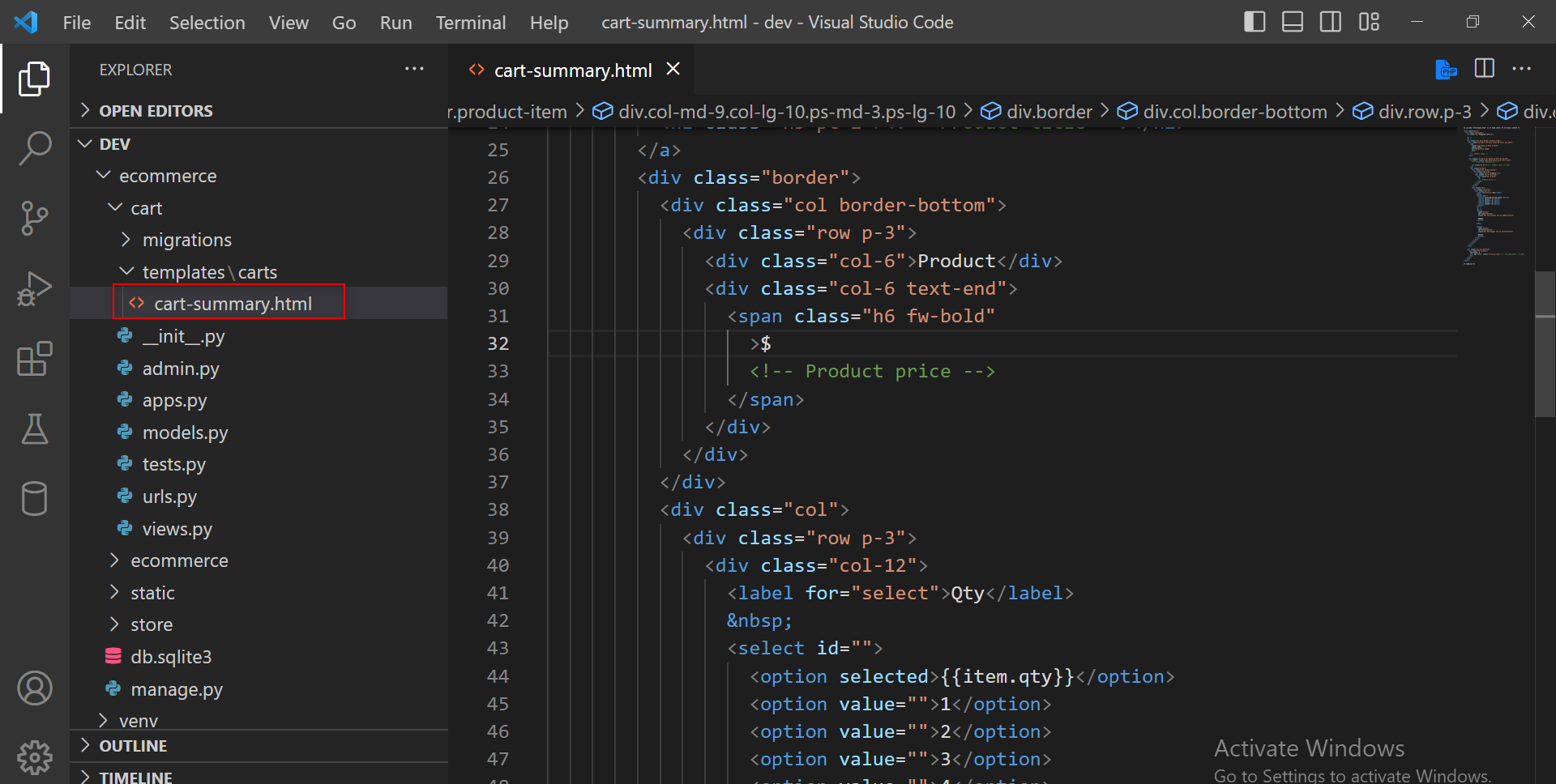
    pass

def cart\_update(request):

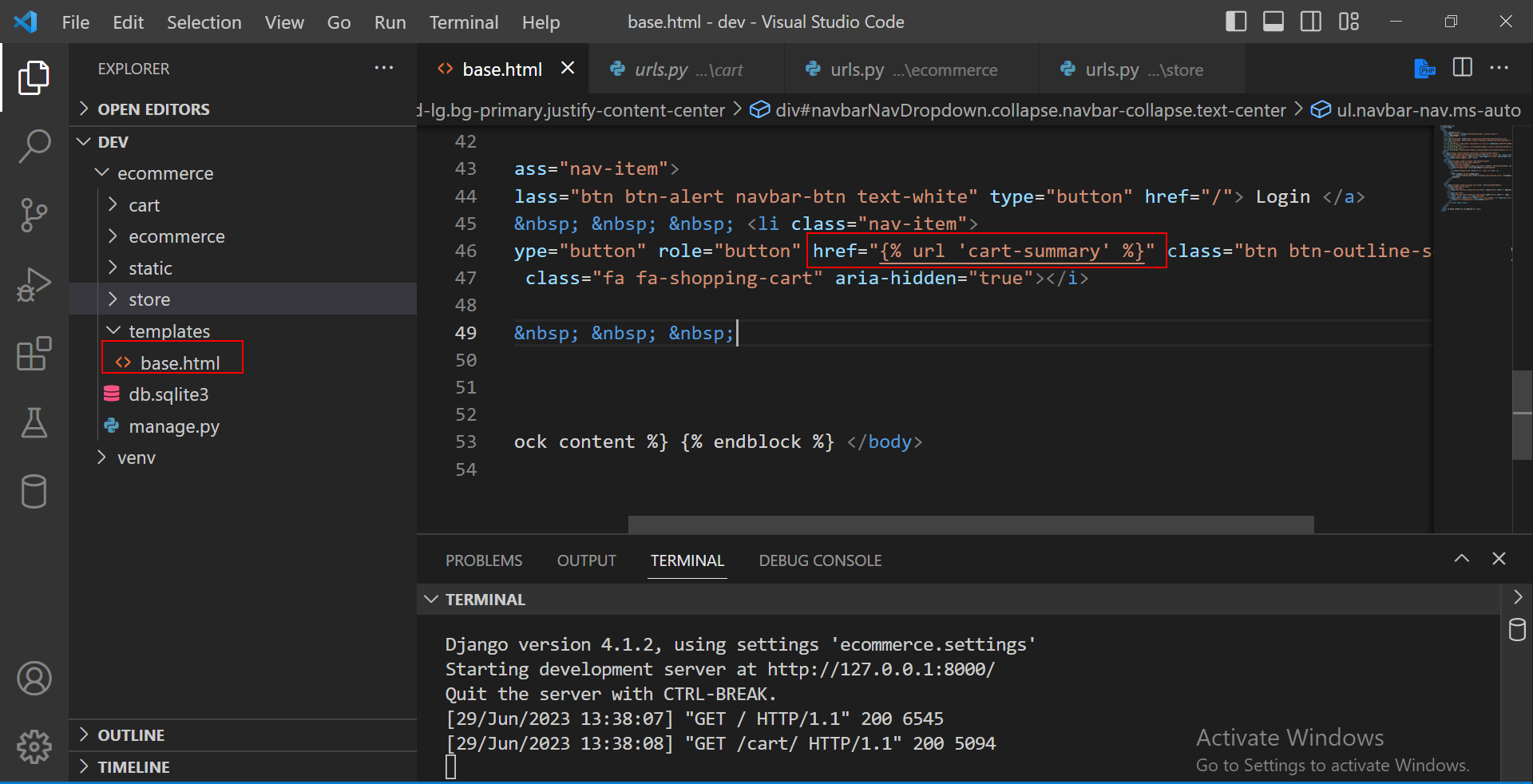
    pass

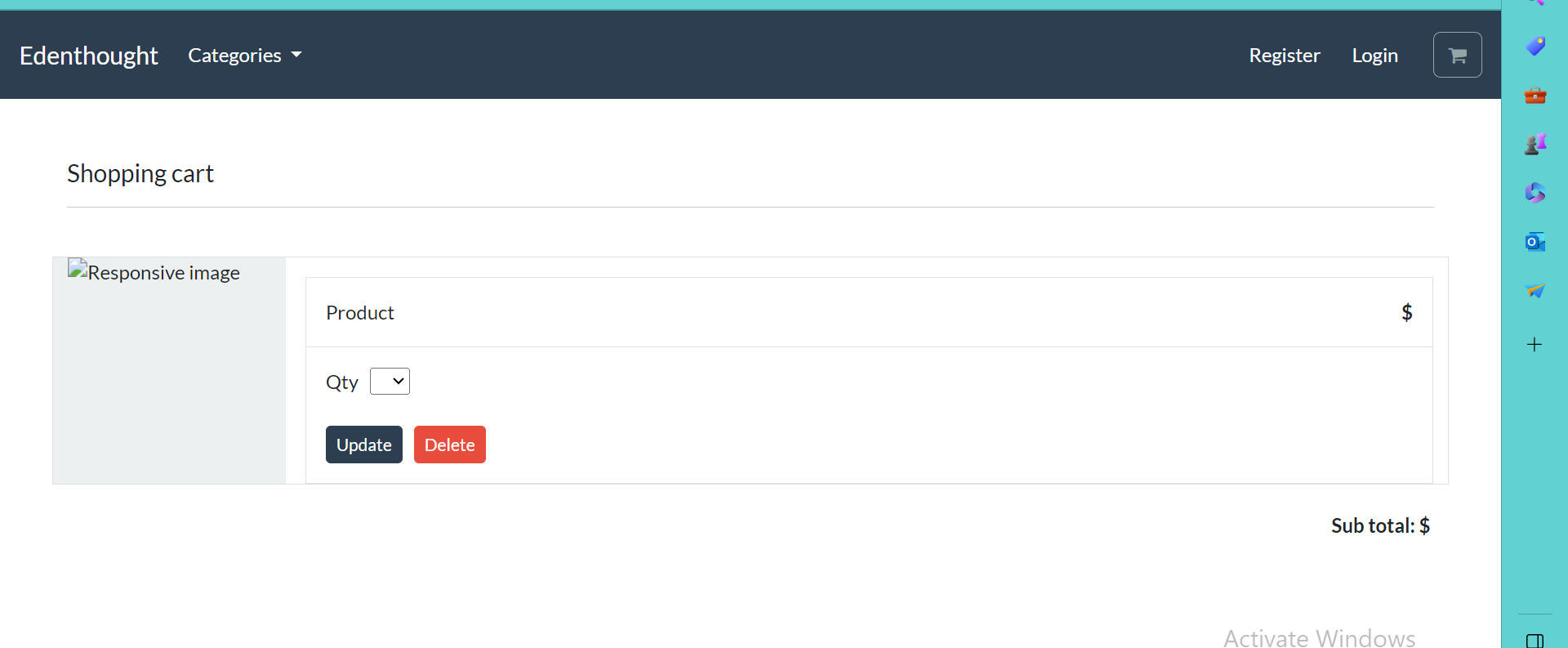
**3. Build the cart summary template**

**Create templates and carts folder, add file cart-summary.html**



**Add cart summary to file base**



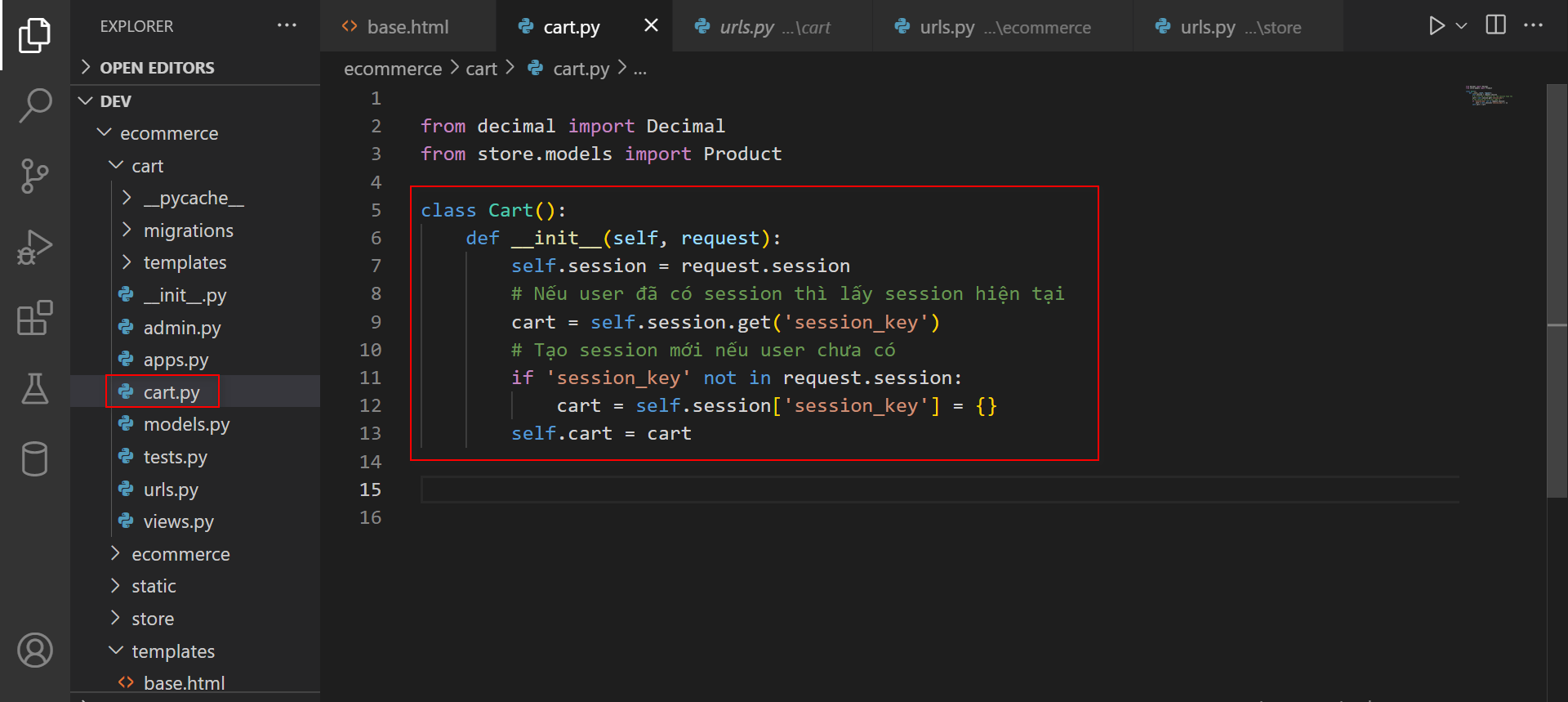


**4. Cart functionally – Sessions**

**Create a Session**

**Create file cart.py**

**Add content to file**



class Cart():

    def \_\_init\_\_(self, request):

        self.session = request.session

        # Nếu user đã có session thì lấy session hiện tại

        cart = self.session.get('session\_key')

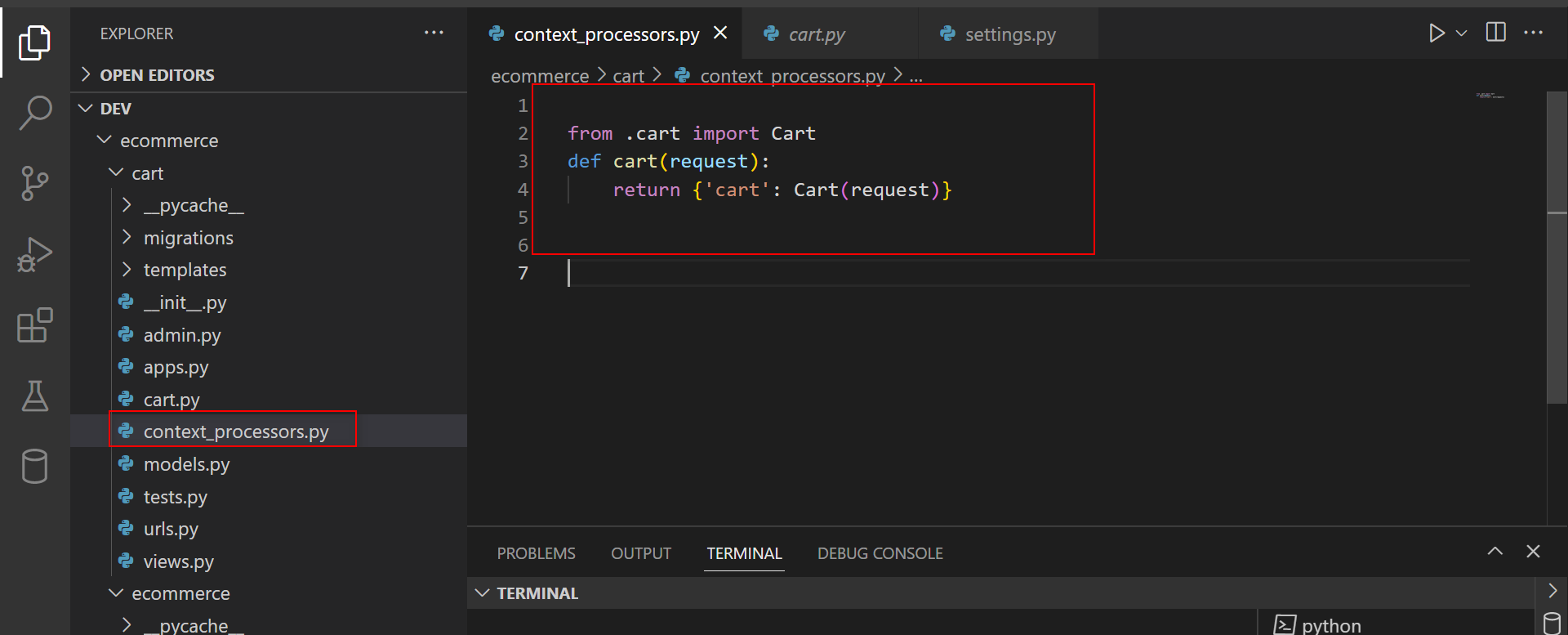
        # Tạo session mới nếu user chưa có

        if 'session\_key' not in request.session:

            cart = self.session['session\_key'] = {}

        self.cart = cart

**Create context processor in cart folder**



from .cart import Cart

def cart(request):

    return {'cart': Cart(request)}

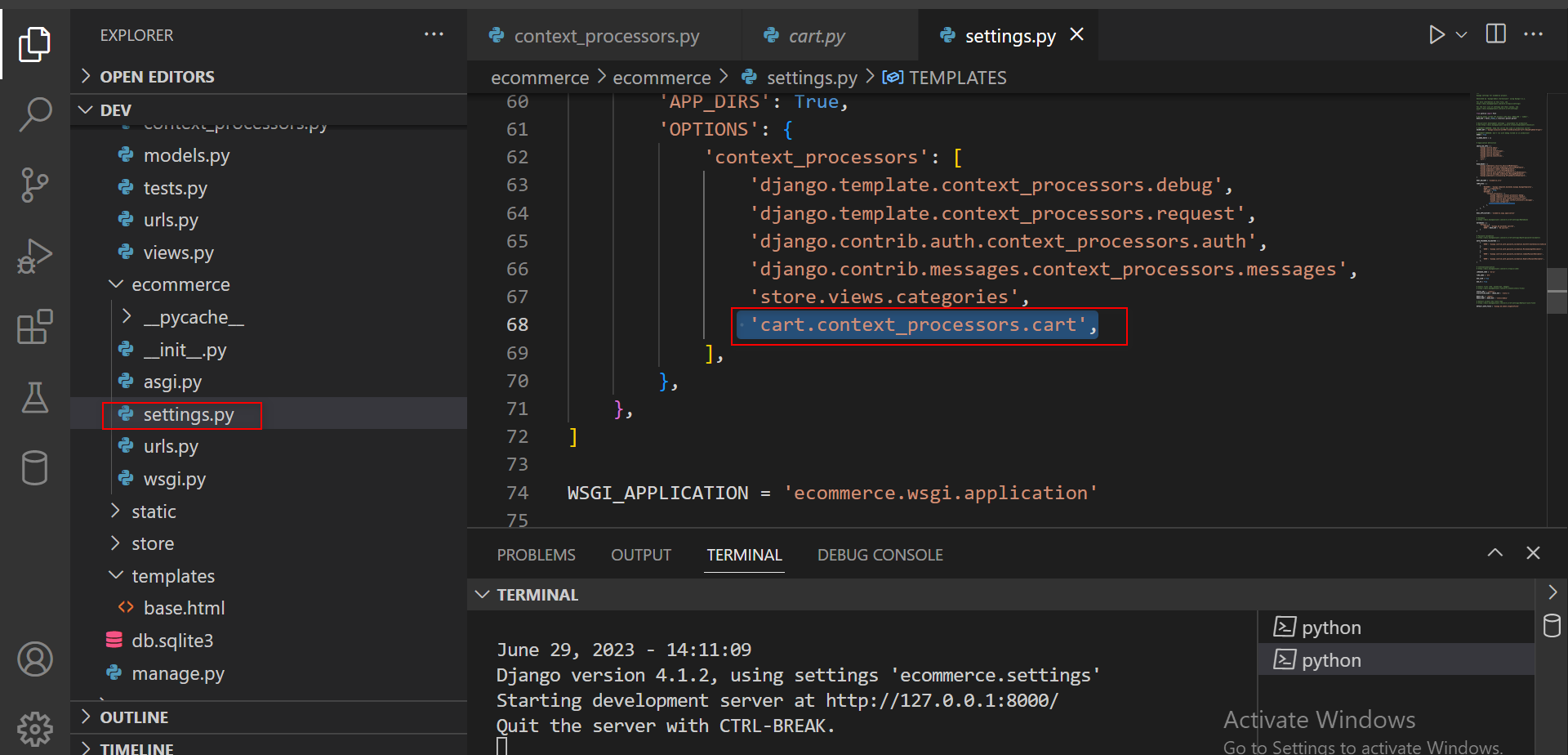
Trong Python, context\_processors là một phần quan trọng trong Django framework, giúp bạn truyền các biến context (bối cảnh) cho tất cả các template trong ứng dụng web của bạn. Context\_processors cho phép bạn định nghĩa một tập hợp các tham số chung hay biến cho mọi template mà bạn render.

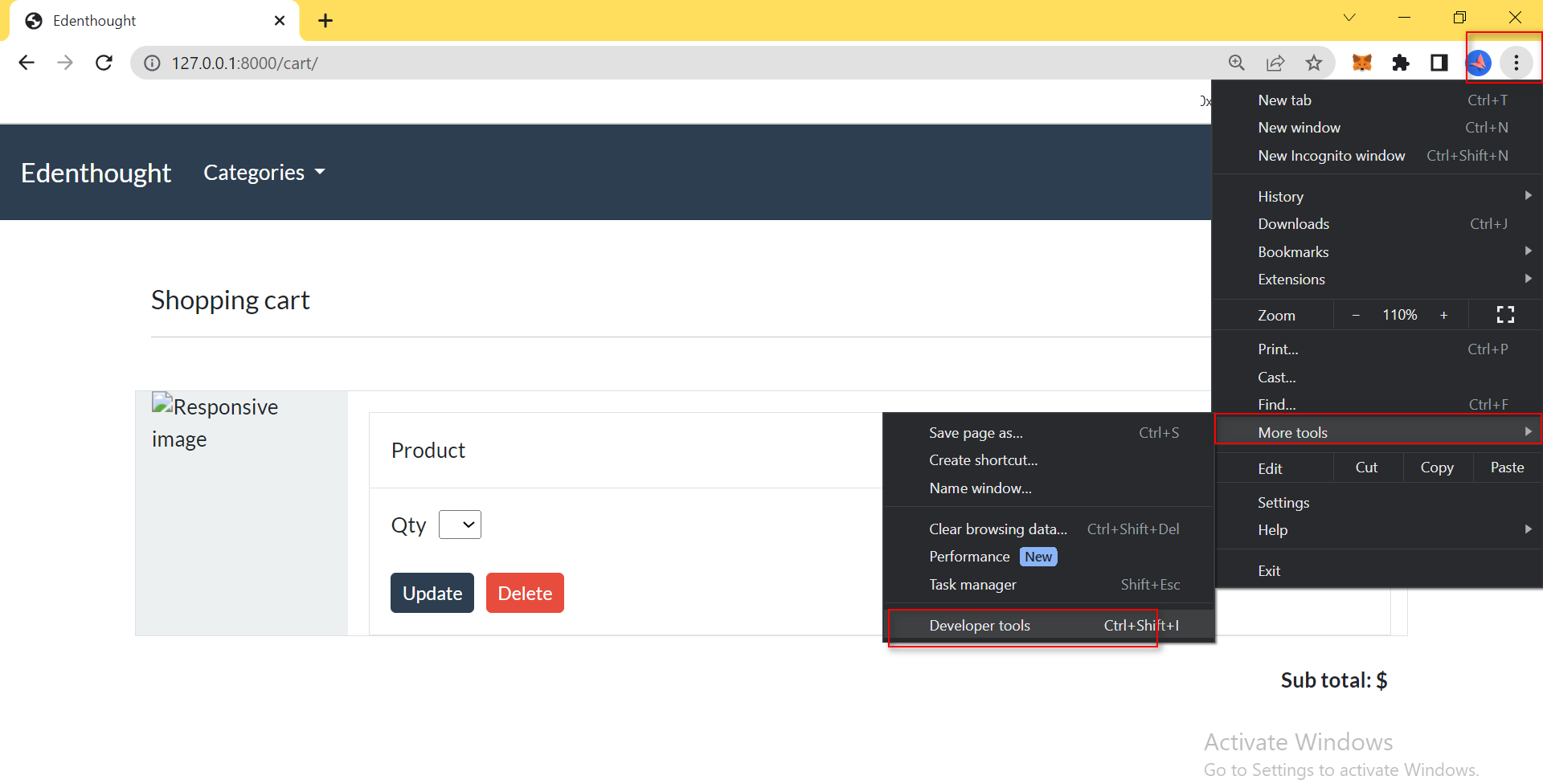
Để hiểu rõ hơn về context\_processors, hãy xem xét quá trình hoạt động của Django khi hiển thị một template:

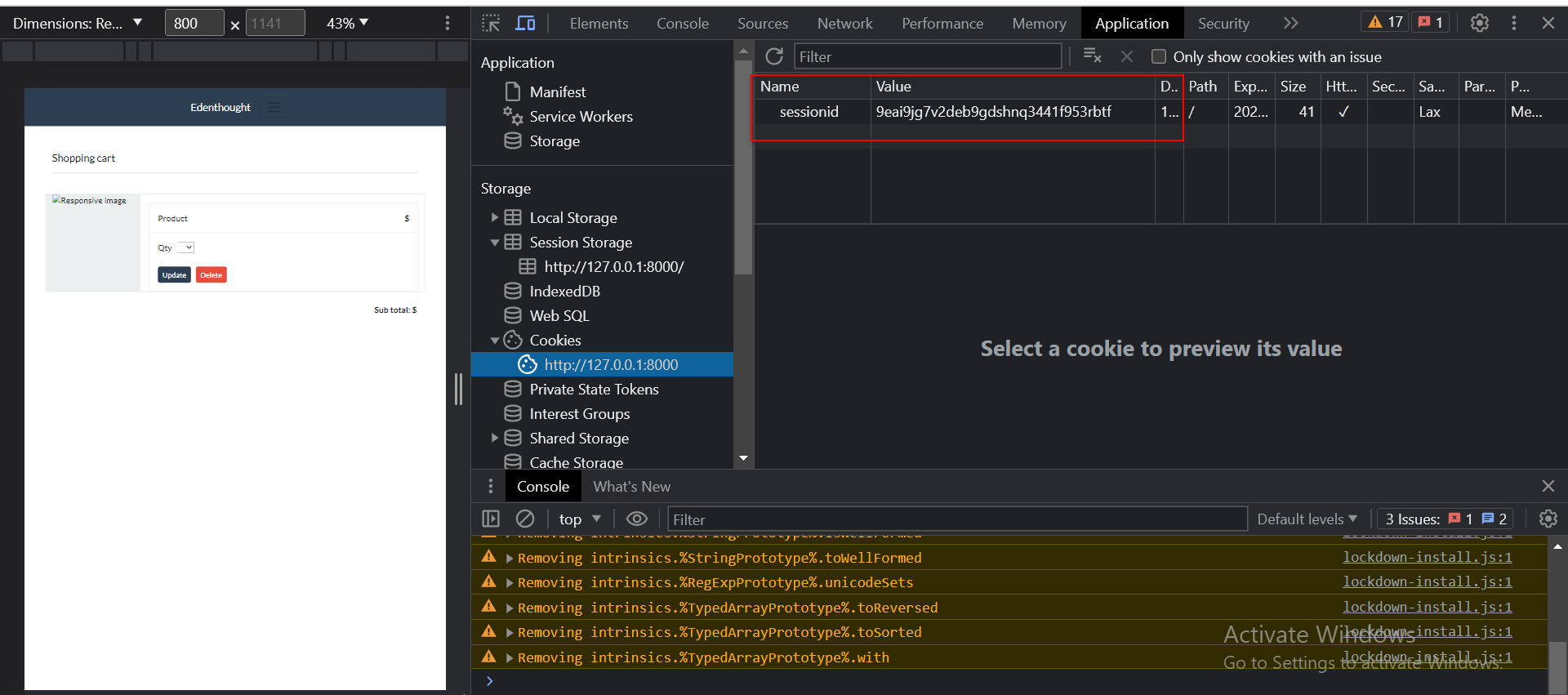
* Người dùng truy cập một URL trên ứng dụng web của bạn.
* Django nhận được yêu cầu và xác định view (chế độ xem) tương ứng để xử lý yêu cầu đó.
* View xử lý yêu cầu và chuẩn bị dữ liệu để truyền cho template.
* Template nhận dữ liệu từ view và render (hiển thị) nó để tạo trang HTML.
* Trang HTML được trả về cho người dùng.

Trong quá trình này, context\_processors giúp bạn thêm các biến vào bối cảnh (context) của mỗi template trước khi nó được render. Bất kỳ biến nào được thêm vào context thông qua context\_processors sẽ có sẵn trong mọi template mà bạn render.

**Add content to file setting.py**



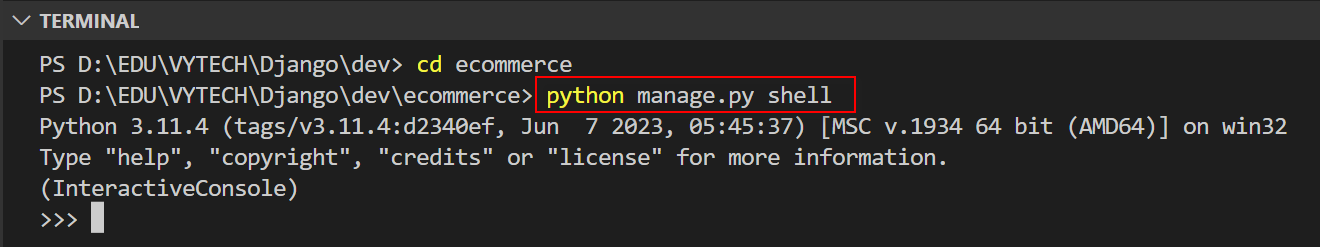




**Test Session:**

**Run command**

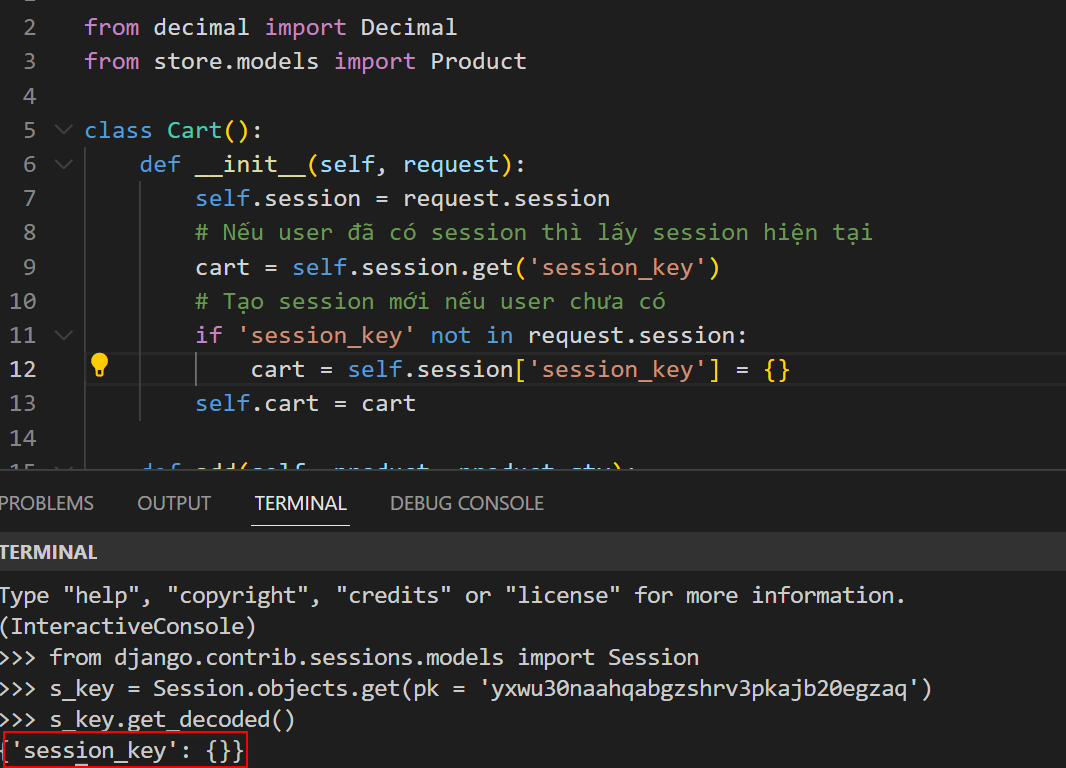
**python manage.py shell**



**from django.contrib.sessions.models import Session**

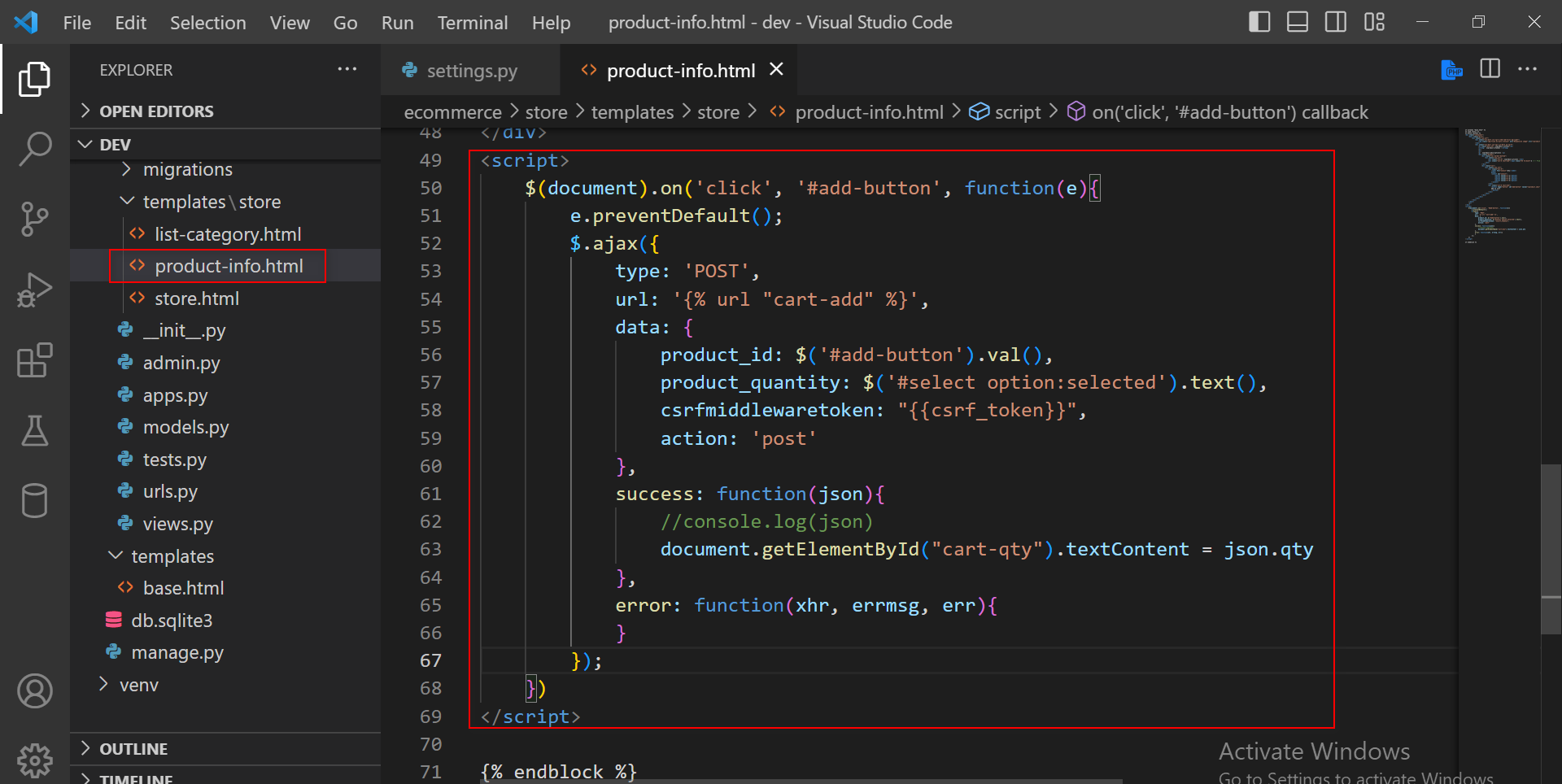
**s\_key = Session.objects.get(pk = '** **mno18bix56ddhs9o53l7g11yzgas47ch')**

**s\_key.get\_decoded()**



**5. Cart functionality Add**

**a. Ajax integration add**



    $(document).on('click', '#add-button', function(e){

        e.preventDefault();

        $.ajax({

            type: 'POST',

            url: '{% url "cart-add" %}',

            data: {

                product\_id: $('#add-button').val(),

                product\_quantity: $('#select option:selected').text(),

                csrfmiddlewaretoken: "{{csrf\_token}}",

                action: 'post'

            },

            success: function(json){

                document.getElementById("cart-qty").textContent = json.qty

            },

            error: function(xhr, errmsg, err){

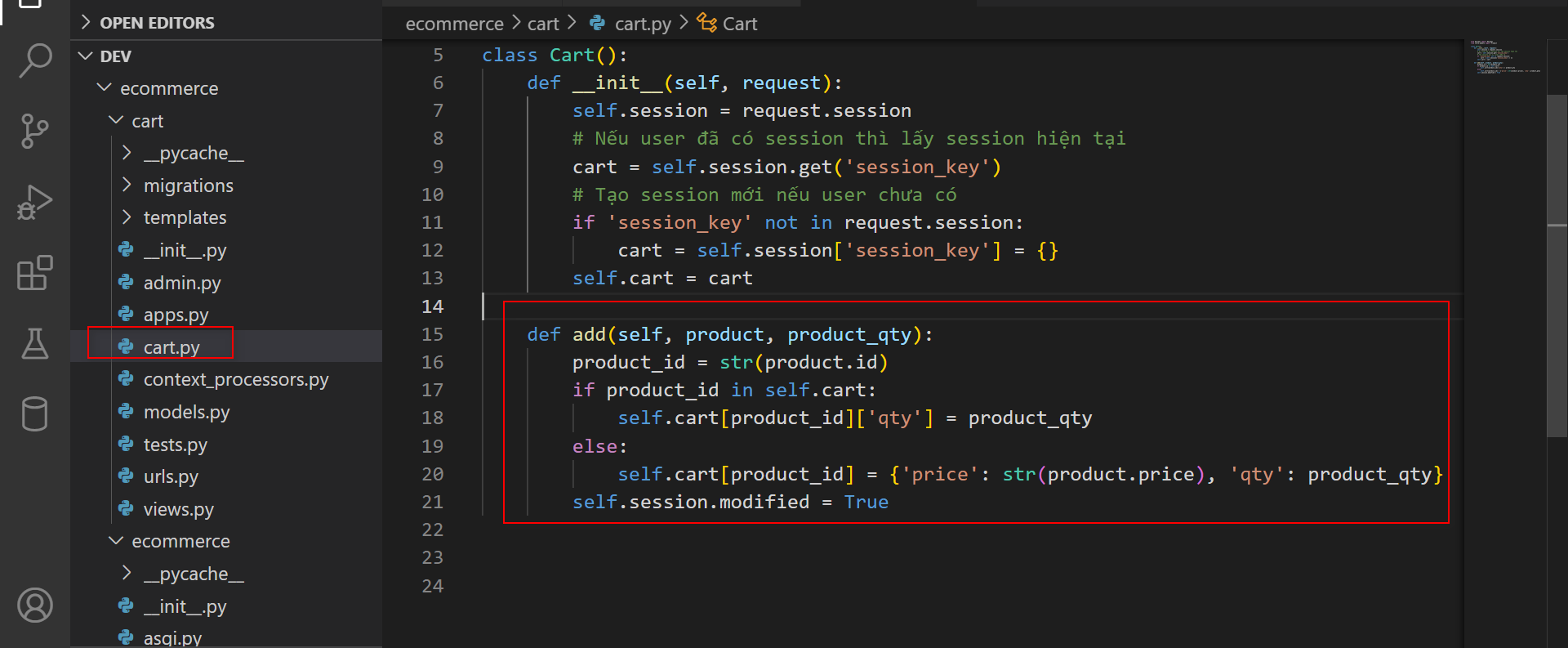
            }

        });

    })

**b. Add functionality Part1**

**Add function add in cart.py**



    def add(self, product, product\_qty):

        product\_id = str(product.id)

        if product\_id in self.cart:

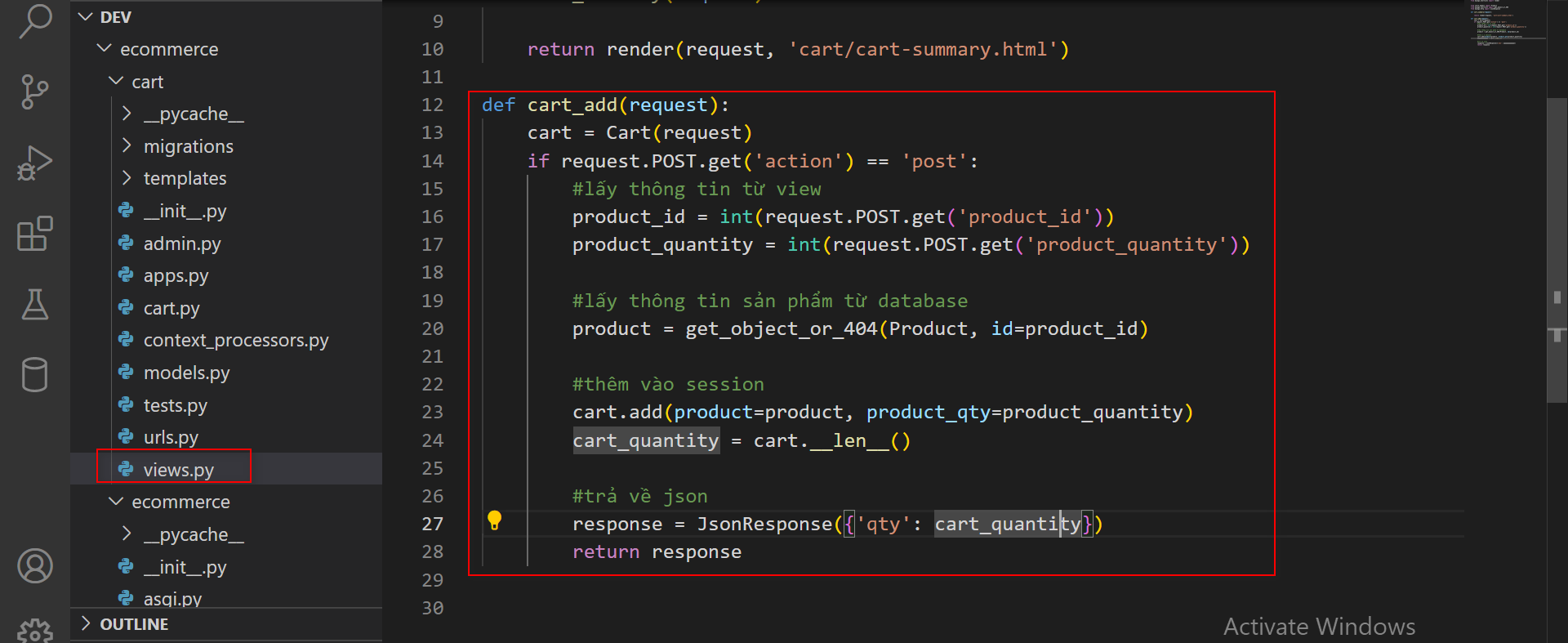
            self.cart[product\_id]['qty'] = product\_qty

        else:

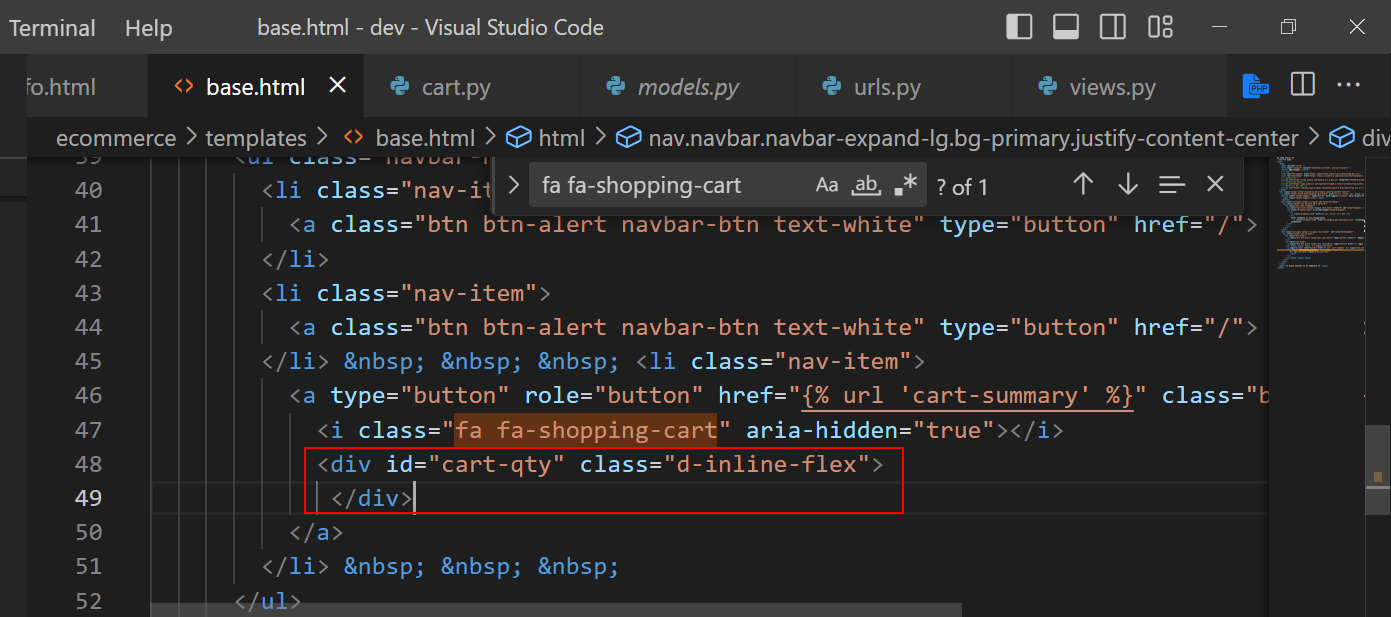
            self.cart[product\_id] = {'price': str(product.price), 'qty': product\_qty}

        self.session.modified = True

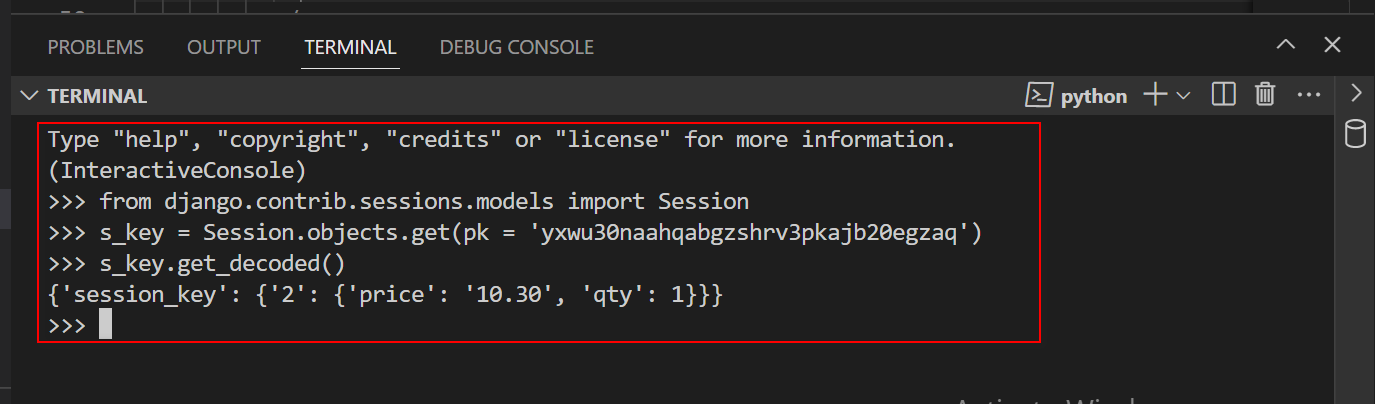
**c. Create function add\_cart in views.py**



**d. Add div to base.html**



**Test a session**



**e. Add content to base.html**

              {% with qty\_amount=cart|length %}

                {% if qty\_amount > 0 %}

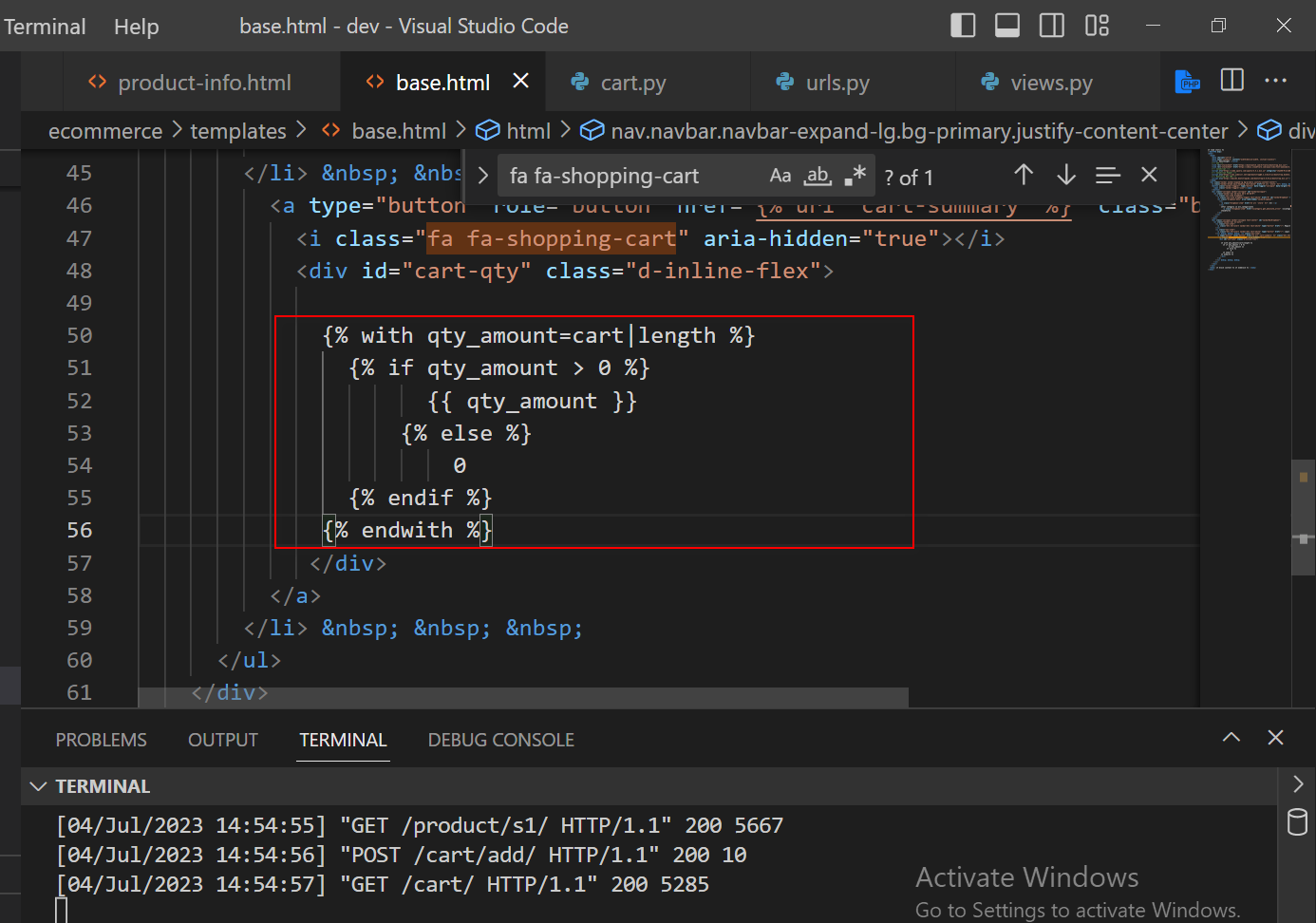
                      {{ qty\_amount }}

                    {% else %}

                        0

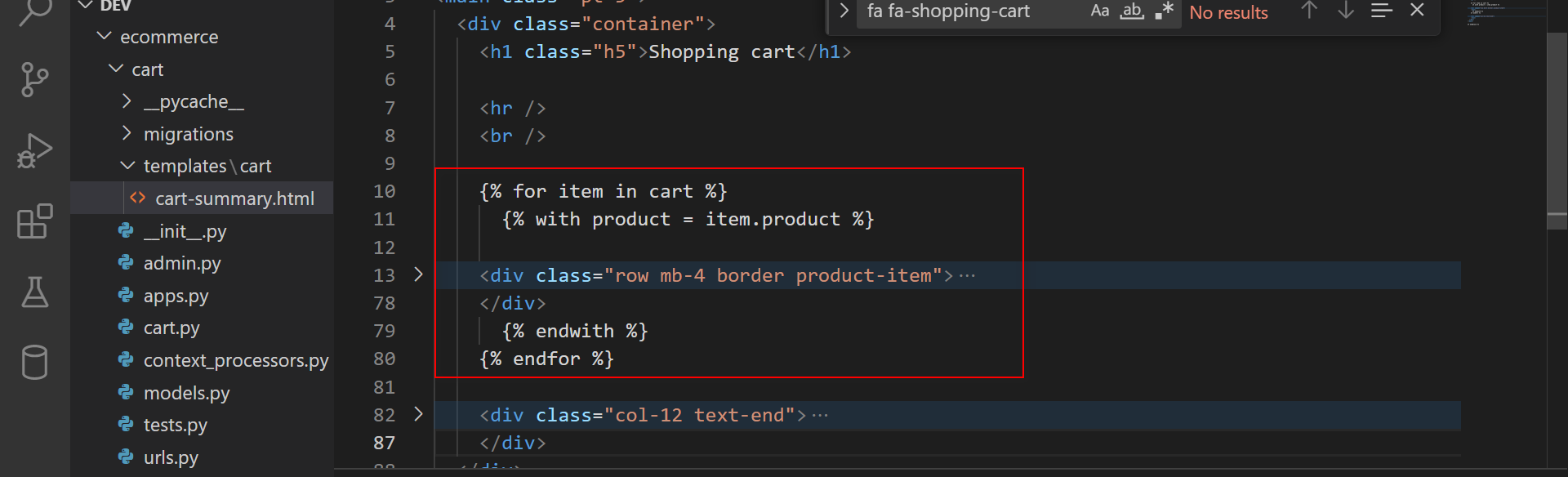
                {% endif %}

              {% endwith %}



**6. Update the cart summary template**

**a. Edit cart-summary.html**



**b. add 2 function to views.py**

    def \_\_iter\_\_(self):

        all\_product\_ids = self.cart.keys()

        products = Product.objects.filter(id\_\_in=all\_product\_ids)

        cart = self.cart.copy()

        for product in products:

            cart[str(product.id)]['product'] = product

        for item in cart.values():

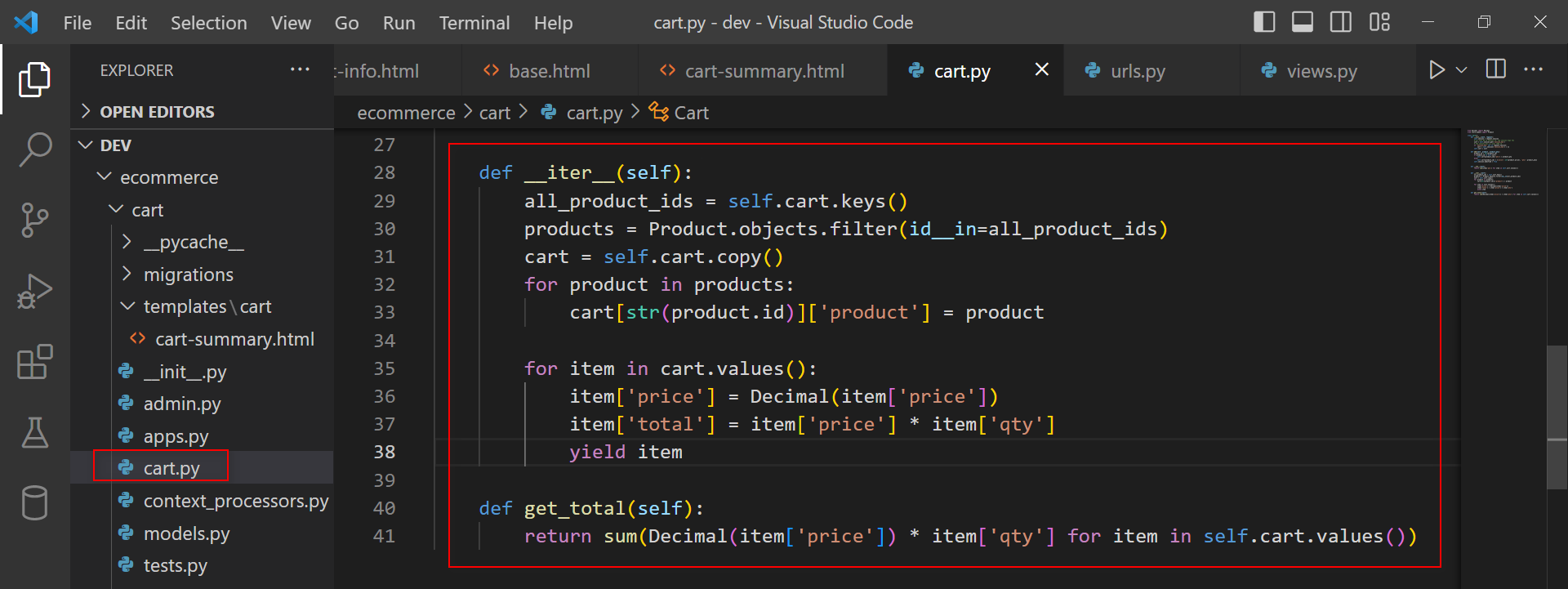
            item['price'] = Decimal(item['price'])

            item['total'] = item['price'] \* item['qty']

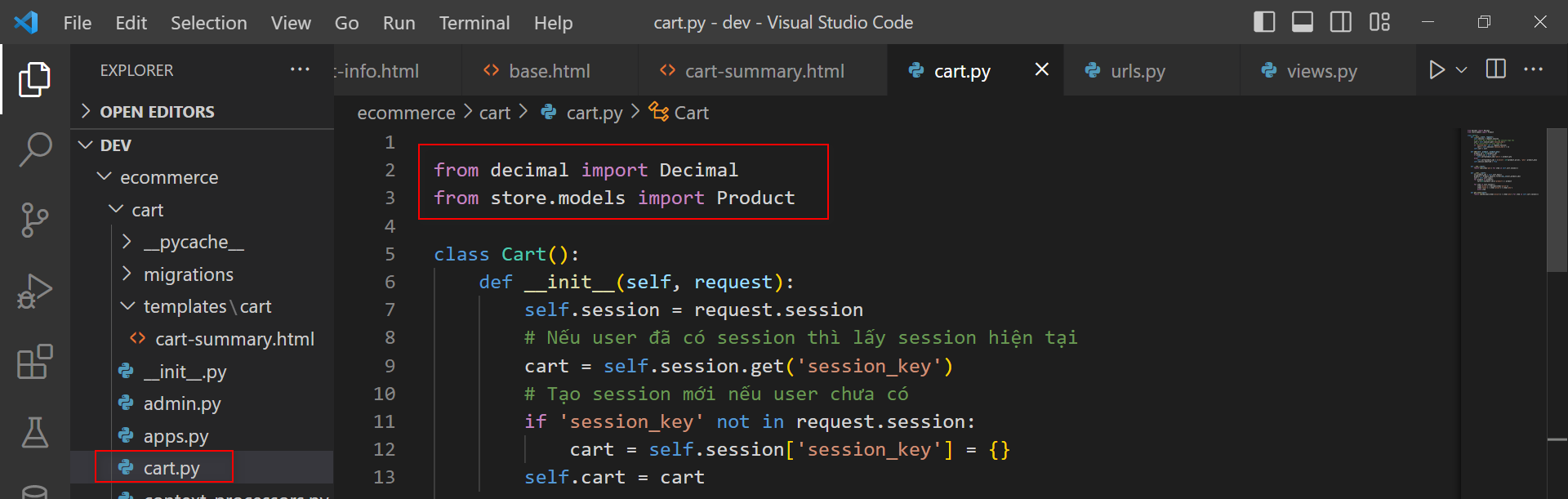
            yield item

    def get\_total(self):

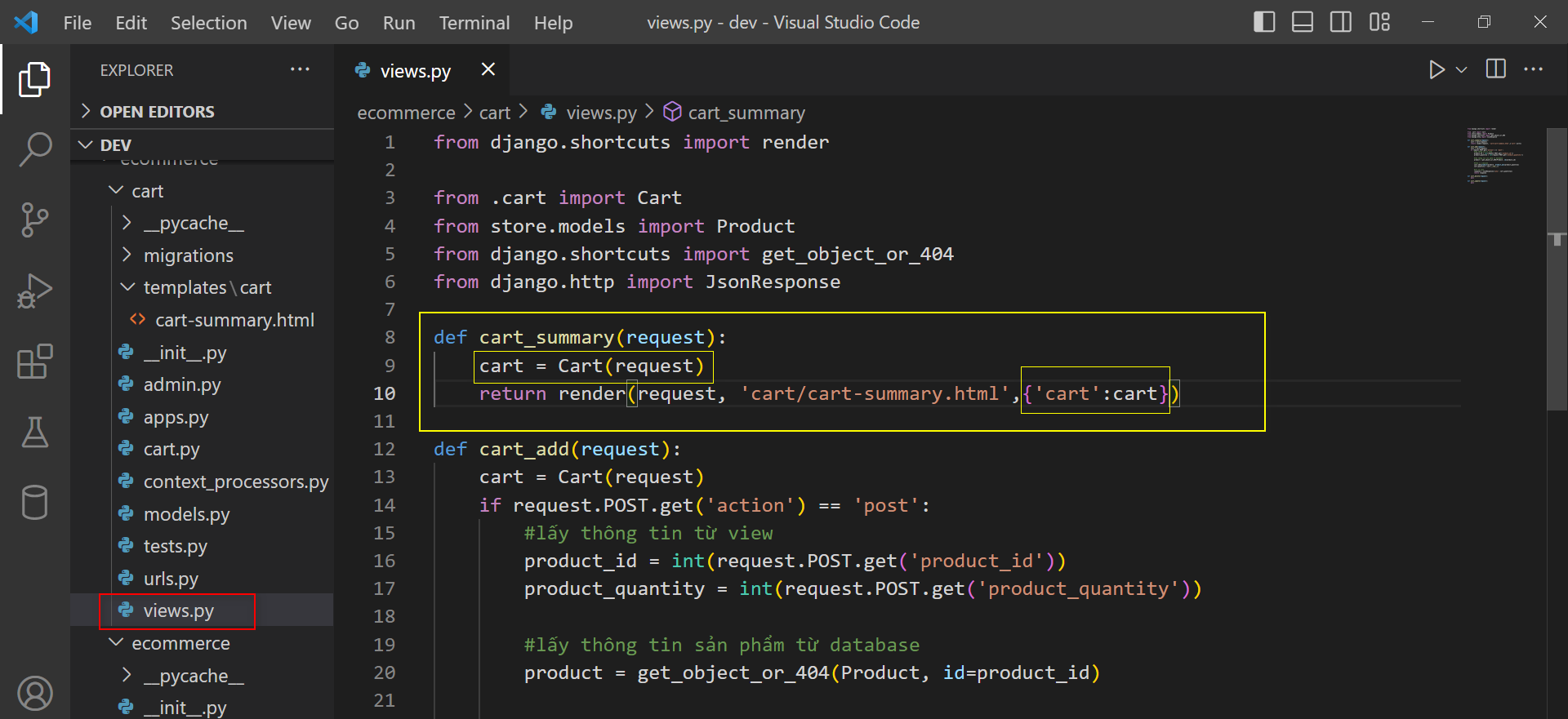
        return sum(Decimal(item['price']) \* item['qty'] for item in self.cart.values())

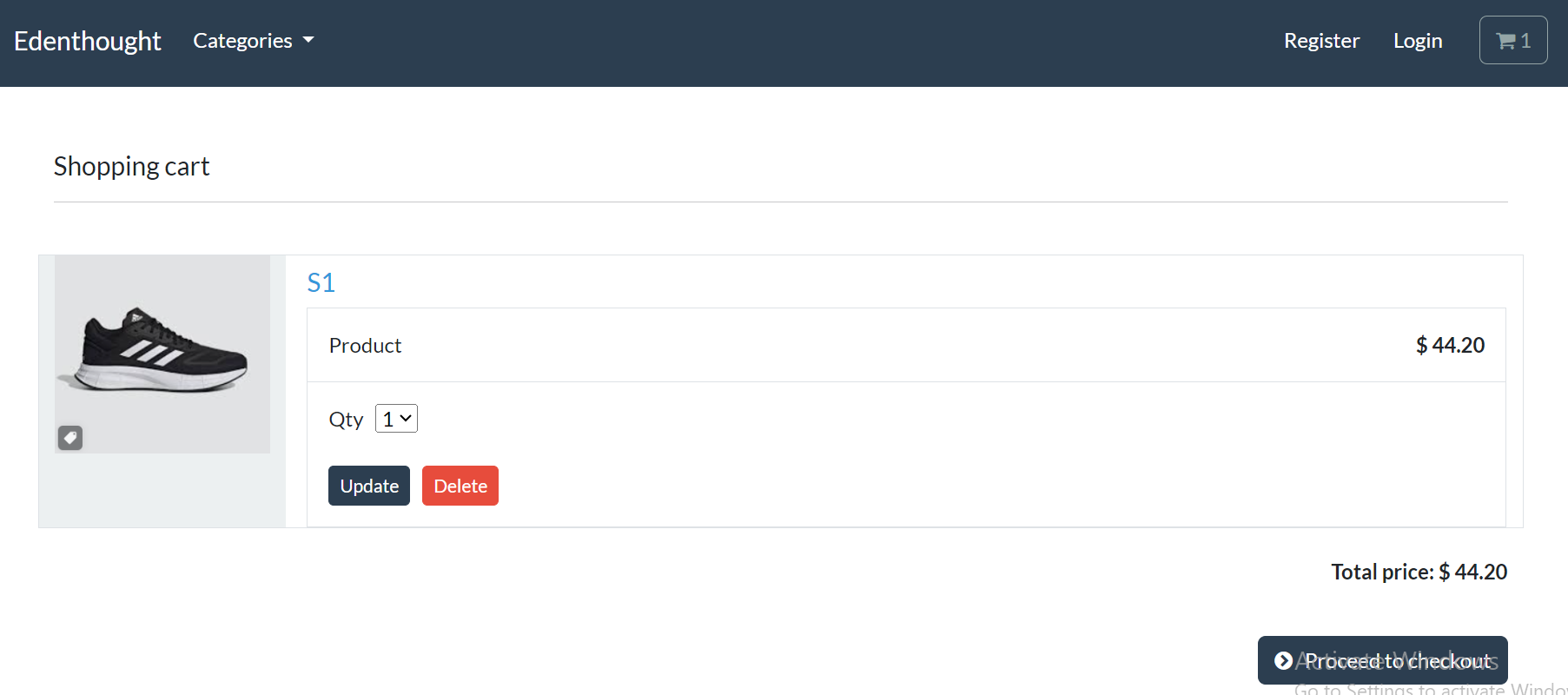


**c. Add import**



**d. Edit function cart\_summary**





**7. the cart summary template**

Add function delete\_cart to view.py

def cart\_delete(request):

    cart = Cart(request)

    if request.POST.get('action') == 'post':

        product\_id = int(request.POST.get('product\_id'))

        cart.delete(product=product\_id)

        cart\_quantity = cart.\_\_len\_\_()

        cart\_total = cart.get\_total()

        response = JsonResponse({'qty':cart\_quantity, 'total':cart\_total})

        return response

