**MARKET Iniciamos con el market**

1. **Inicio (core/home.html)** muestre algo diferente si está logueado o no.
2. Mostrar productos destacados (un carrusel, cards, etc.).
3. Redirigir a login si intentan comprar/vender sin estar autenticados.
4. Integrar tu app market para listar, crear, comprar o intercambiar.

**🔹 Paso 1: Inicio con contenido dinámico**

En tu core/home.html podés hacer algo más atractivo con cards/carousel.  
Ejemplo básico con **cards** que, si no está logueado, mandan a login:

{% extends "base.html" %}

{% load socialaccount %}

{% block title %}Inicio • Mercadito{% endblock %}

{% block content %}

<div class="home-container">

<h1>Bienvenido al Mercadito</h1>

{% if not user.is\_authenticated %}

<p>Iniciá sesión para comprar, vender o intercambiar productos</p>

<div class="social-login">

<a href="{% provider\_login\_url 'google' %}" class="btn-social google">

<img src="{% static 'img/google.svg' %}" alt="Google" class="social-logo">

</a>

<a href="{% provider\_login\_url 'github' %}" class="btn-social github">

<img src="{% static 'img/github.svg' %}" alt="GitHub" class="social-logo">

</a>

</div>

{% else %}

<h2>Últimos productos</h2>

<div class="card-container">

{% for product in products %}

<div class="card">

<h3>{{ product.title }}</h3>

<p>{{ product.description|truncatewords:15 }}</p>

<p><strong>${{ product.price }}</strong></p>

<a href="{% url 'product-list' %}">Ver más</a>

</div>

{% empty %}

<p>No hay productos publicados todavía.</p>

{% endfor %}

</div>

{% endif %}

</div>

{% endblock %}

👉 Notá que estoy usando products, así que tenemos que pasarlos desde la vista de home.

**🔹 Paso 2: Vista de inicio (core/views.py)**

Creamos una vista home que trae algunos productos para mostrar en portada:

from django.shortcuts import render

from market.models import Product

def home(request):

products = Product.objects.filter(active=True).order\_by("-created\_at")[:6] # últimos 6

return render(request, "core/home.html", {"products": products})

Y en tus urls.py principales:

from django.contrib import admin

from django.urls import path, include

from core.views import home

urlpatterns = [

path("admin/", admin.site.urls),

path("", home, name="home"), # inicio

path("market/", include("market.urls")), # mercado

path("accounts/", include("allauth.urls")), # login/signup

]

**🔹 Paso 3: Productos en market**

Ya tenés product\_list, lo dejamos así por ahora:

market/views.py:

from django.shortcuts import render

from .models import Product

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

@login\_required

def product\_list(request):

products = Product.objects.filter(active=True).order\_by("-created\_at")

return render(request, "market/product\_list.html", {"products": products})

Y creamos un template simple market/product\_list.html:

{% extends "base.html" %}

{% block title %}Productos • Mercadito{% endblock %}

{% block content %}

<h2>Todos los productos</h2>

<div class="card-container">

{% for product in products %}

<div class="card">

<h3>{{ product.title }}</h3>

<p>{{ product.description }}</p>

<p><strong>${{ product.price }}</strong></p>

{% if user.is\_authenticated %}

<button>Comprar / Truequear</button>

{% else %}

<a href="{% url 'account\_login' %}">Inicia sesión para comprar</a>

{% endif %}

</div>

{% empty %}

<p>No hay productos disponibles.</p>

{% endfor %}

</div>

{% endblock %}

**🔹 Paso 4: Crear productos**

Para que un usuario pueda **vender/publicar**, necesitás un ProductForm y una vista protegida con login\_required.

market/forms.py:

from django import forms

from .models import Product

class ProductForm(forms.ModelForm):

class Meta:

model = Product

fields = ["title", "description", "marca", "price"]

market/views.py:

from django.shortcuts import render, redirect

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

from .forms import ProductForm

@login\_required

def product\_create(request):

if request.method == "POST":

form = ProductForm(request.POST)

if form.is\_valid():

product = form.save(commit=False)

product.seller = request.user

product.save()

return redirect("product-list")

else:

form = ProductForm()

return render(request, "market/product\_form.html", {"form": form})

market/urls.py:

from django.urls import path

from .views import product\_list, product\_create

urlpatterns = [

path("lista/", product\_list, name="product-list"),

path("nuevo/", product\_create, name="product-create"),

]

**🔹 Paso 5: Template para crear producto**

market/product\_form.html:

{% extends "base.html" %}

{% block title %}Nuevo producto{% endblock %}

{% block content %}

<h2>Publicar producto</h2>

<form method="post">

{% csrf\_token %}

{{ form.as\_p }}

<button type="submit" class="btn-primary">Publicar</button>

</form>

{% endblock %}

**🔹 Paso 6: Mejorar producto**

**Mejorar el modelo Product**

Actualmente tu modelo solo tiene título, descripción, marca y precio. Vamos a agregar:

* **stock**: para controlar la cantidad disponible.
* **active**: ya lo tenés, sirve para "desactivar" productos sin borrarlos.
* **imagen**: opcional, para mostrar la foto en la card.
* **fecha de modificación**: para saber cuándo se actualizó.

from django.conf import settings

from django.db import models

class Product(models.Model):

seller = models.ForeignKey(settings.AUTH\_USER\_MODEL, on\_delete=models.CASCADE, related\_name="products")

title = models.CharField(max\_length=200)

description = models.TextField(blank=True)

marca = models.CharField(max\_length=100, blank=True, default="Generico")

price = models.DecimalField(max\_digits=12, decimal\_places=2)

stock = models.PositiveIntegerField(default=1) # nuevo campo

image = models.ImageField(upload\_to="product\_images/", blank=True, null=True) # opcional

active = models.BooleanField(default=True)

created\_at = models.DateTimeField(auto\_now\_add=True)

updated\_at = models.DateTimeField(auto\_now=True) # fecha de última modificación

def \_\_str\_\_(self):

return self.title

def is\_available(self):

return self.active and self.stock > 0

**Tip**: is\_available() sirve para mostrar “No disponible” si no hay stock o está desactivado.

**2️⃣ Formularios (market/forms.py)**

Incluí el stock y la imagen:

from django import forms

from .models import Product

class ProductForm(forms.ModelForm):

class Meta:

model = Product

fields = ["title", "description", "marca", "price", "stock", "image"]

widgets = {

"description": forms.Textarea(attrs={"rows": 3}),

}

**3️⃣ Crear / Editar / Eliminar productos**

**Crear producto (ya lo tenés)**

market/views.py:

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

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

from .forms import ProductForm

from .models import Product

@login\_required

def product\_create(request):

if request.method == "POST":

form = ProductForm(request.POST, request.FILES)

if form.is\_valid():

product = form.save(commit=False)

product.seller = request.user

product.save()

return redirect("product-list")

else:

form = ProductForm()

return render(request, "market/product\_form.html", {"form": form})

**Editar producto**

@login\_required

def product\_edit(request, pk):

product = get\_object\_or\_404(Product, pk=pk, seller=request.user)

if request.method == "POST":

form = ProductForm(request.POST, request.FILES, instance=product)

if form.is\_valid():

form.save()

return redirect("product-list")

else:

form = ProductForm(instance=product)

return render(request, "market/product\_form.html", {"form": form})

**Eliminar producto**

@login\_required

def product\_delete(request, pk):

product = get\_object\_or\_404(Product, pk=pk, seller=request.user)

if request.method == "POST":

product.active = False

product.save()

return redirect("product-list")

return render(request, "market/product\_confirm\_delete.html", {"product": product})

**4️⃣ Templates**

**Formulario (product\_form.html)**

Ya te lo armé antes, solo asegurate de que soporte imágenes:

<form method="post" enctype="multipart/form-data">

{% csrf\_token %}

{{ form.as\_p }}

<button type="submit" class="btn-primary">Guardar</button>

</form>

**Confirmar eliminación (product\_confirm\_delete.html)**

{% extends "base.html" %}

{% block title %}Eliminar producto{% endblock %}

{% block content %}

<h2>Eliminar producto</h2>

<p>¿Seguro querés eliminar "{{ product.title }}"?</p>

<form method="post">

{% csrf\_token %}

<button type="submit" class="btn-primary">Sí, eliminar</button>

<a href="{% url 'product-list' %}" class="btn-secondary">Cancelar</a>

</form>

{% endblock %}

**Mostrar stock en cards (product\_list.html o home.html)**

<p>

{% if product.is\_available %}

<strong>Stock: {{ product.stock }}</strong>

{% else %}

<span style="color:red;">No disponible</span>

{% endif %}

</p>

**5️⃣ Cosas adicionales a tener en cuenta**

1. **Archivos de imágenes**: Asegurate de tener MEDIA\_URL y MEDIA\_ROOT configurados en settings.py y de servirlos en urls.py durante desarrollo.

# settings.py

MEDIA\_URL = "/media/"

MEDIA\_ROOT = BASE\_DIR / "media"

# urls.py (principal)

from django.conf import settings

from django.conf.urls.static import static

urlpatterns = [

# ... tus urls

] + static(settings.MEDIA\_URL, document\_root=settings.MEDIA\_ROOT)

1. **Botones de comprar / truequear**: Cuando el producto se compre, restar 1 del stock y marcar “No disponible” si llega a 0.
2. **Validaciones**: Prevenir que el stock sea negativo al crear o editar producto.
3. **Usuarios vs productos**: Cada usuario solo debería poder **editar o eliminar sus propios productos**.