





## Modelos y Apps con las que trabaja el Proyecto

Por Nombre Por Fecha Por Stok → Notar que se puede ordenar por fecha, stock y nombre

| COD          | Nombres     | Precio Venta | Precio Compra | Stok | Num Ventas | Acciones  |
|--------------|-------------|--------------|---------------|------|------------|---|
| 750521000050 | Yogurt Toni | 2,30         | 1,00          | 35   | 27         |   |
| 702011075291 | Milows      | 2,30         | 1,50          | 21   | 29         |   |

View

```
def get_queryset(self):
    kword = self.request.GET.get("kword", '')
    order = self.request.GET.get("order", '')
    queryset = Product.objects.buscar_producto(kword, order)
    return queryset
```

Manager

```
def buscar_producto(self, kword, order):
    consulta = self.filter(
        Q(name__icontains=kword) | Q(barcode=kword)
    )
    # verificamos en que orden se solicita
    if order == 'date':
        # ordenar por fecha
        return consulta.order_by('created')
    elif order == 'name':
        # ordenar por nombre
        return consulta.order_by('name')
    elif order == 'stok':
        return consulta.order_by('count')
    else:
        return consulta.order_by('-created')
```

## Registrar Producto - Modulo Almacén

```
class ProductCreateView(AlmacenPermisoMixin, CreateView):
    template_name = "producto/form_producto.html"
    form_class = ProductForm
    success_url = reverse_lazy('producto_app:producto-lista')
```

```
# validations
def clean_barcode(self):
    barcode = self.cleaned_data['barcode']
    if len(barcode) < 11:
        raise forms.ValidationError('Ingrese un codigo de barras correcto')

    return barcode

def clean_purchase_price(self):
    purchase_price = self.cleaned_data['purchase_price']
    if not purchase_price > 0:
        raise forms.ValidationError('Ingrese un precio compra mayor a cero')

    return purchase_price

def clean_sale_price(self):
    sale_price = self.cleaned_data['sale_price']
    purchase_price = self.cleaned_data.get('purchase_price')
    if not sale_price >= purchase_price:
        raise forms.ValidationError('El precio de venta debe ser mayor o igual que el precio de compra')

    return sale_price
```

## Registrar Producto - Modulo Almacén

```
class ProductUpdateView(AlmacenPermisoMixin, UpdateView):
    template_name = "producto/form_producto.html"
    model = Product
    form_class = ProductForm
    success_url = reverse_lazy('producto_app:producto-lista')
```

Se maneja igual que el create ya que trabaja con el mismo formulario, sin embargo el HTML se cambia dependiendo si es para crear o actualizar ya que muestra un boton de eliminar en el actualizar

Guardar

Eliminar

```
<div class="cell medium-6">
  <button type="submit" class="success button" style="width: 200px;">Guardar</button>
  {% if product %}
  <a href="{% url 'producto_app:producto-delete' product.id %}" class="alert button clear">Eliminar</a>
  {% endif %}
</div>
```

Recordando que cuando se trabaja con update pues se puede acceder al objeto, entonces he usado éso para saber si hay un objeto y en caso de haberlo mostrar el botón

