YA HE TERMINADO. Ahora te paso las reglas para construir el generador:

1. El nombre del archivo que contiene el generador es la concatenación de del nombre del modelo y la palabra views, en notación Snake Case en letras minúsculas. A continuación te paso ejemplos con el nombre del modelo y el respectivo nombre del archivo:

Actividad: actividad\_views.py

TipoVenta: tipo\_venta\_views.py

TipoDocumentoIdentidad: tipo\_documento\_identidad\_views.py

2. En la primera línea de código existe un comentario:

# neumatic\apps\maestros\views\actividad\_views.py

En este caso, la ruta relativa: "neumatic\apps\maestros\views\" será un argumento, que se concatenará al nombre del archivo, explicado en el punto 1.

3. En la tercera línea, la instrucción:

from django.urls import reverse\_lazy

es la misma en todos los casos

4. En la cuarta línea, la instrucción:

from ..models.base\_models import Actividad

Se construye con dos argumentos:

"..models.base\_models" que es la ruta del archivo donde está el modelo

"Actividad" que es el nombre del modelo

5. En la quinta línea, la instrucción:

from ..forms.actividad\_forms import ActividadForm

Se construye con dos argumentos:

"..forms.actividad\_forms" que es la ruta del archivo donde está el formulario asociado al modelo Actividad

"ActividadForm" que es el nombre del formulario asociado al modelo Actividad

El nombre del formulario es la concatenación del nombre del modelo con la palabra Form

6. La clase: class ConfigViews():

Siempre tendrá ese nombre, pero su contenido solo tendrá dos argumentos variables:

# Modelo

model = Actividad

# Formulario asociado al modelo

form\_class = ActividadForm

model: que se le asignará el nombre del modelo

form\_class: que se le asignará el nombre del formulario asociado al modelo

Una variante adicional será el valor de model\_string, que por defecto es model.\_\_name\_\_.lower(), sin embargo se le puede asignar otro valor.

Para que model\_string tome el valor model.\_\_name\_\_.lower()

Asignaremos el valor "default", en los casos que el modelo sea de una palabra, por ejemplo:

Si el modelo es Actividad se asigna model.\_\_name\_\_.lower()

Si el modelo es Cliente se asigna model.\_\_name\_\_.lower()

En los modelos con dos palabras:

Si el modelo es TipoVenta se asigna tipo\_venta

Si el modelo es TipoCompra se asigna tipo\_compra

En los modelos con tres palabras:

Si el modelo es TipoDocumentoIdentidad se asigna tipo\_documento\_identidad

Si el modelo es TipoDocumentoVenta se asigna tipo\_documento\_venta

Si el modelo es TipoDocumentoCompra se asigna tipo\_documento\_compra

En resumen, debemos pasar del formato UpperCamelCase a Snake Case en minúsculas

El resto de las instrucciones que siguen después de la asignación a model\_string, son las mismas para todas los casos con otros modelos.

7. la clase: class DataViewList():

Los valores de cada uno de los atributos de esta clase son variables y dependen de la estructura de su modelo respectivo.

8. El contenido de las clases:

class ActividadCreateView(MaestroCreateView):

class ActividadUpdateView(MaestroUpdateView):

class ActividadDeleteView (MaestroDeleteView):

Es el mismo para todos los modelos de la aplicación maestros. Sin embargo, hay una observación importante al momento de identificar cada una de las clases: