HINTS

ADMINISTRAR USUARIOS EN DJANGO ADMIN

La gestión de usuarios en Django admin es un tema complicado. Si impone demasiados permisos, es posible que interfiera con las operaciones diarias. Si permite que los permisos se otorguen libremente sin supervisión, puede poner su sistema en riesgo.

Django proporciona un buen marco de autenticación, con una estrecha integración con el administrador de Django. Fuera de la caja, el administrador de Django no impone restricciones especiales al administrador del usuario. Esto puede conducir a escenarios peligrosos que podrían comprometer su sistema. ¿Sabía que los usuarios del personal que administran a otros usuarios en el administrador pueden editar sus propios permisos? ¿Sabías que también pueden convertirse en super usuarios?

PERMISOS DE MODELO

Estos son complicados. Si no se establecen permisos, entonces pone su sistema en riesgo de intrusos, fugas de datos y errores humanos. Pero si abusa de ellos o los usa demasiado, se corre el riesgo de interferir con las operaciones diarias.

Django viene con un sistema de autenticación incorporado, y este incluye usuarios, grupos y permisos.

Cuando se crea un modelo, Django generará automáticamente cuatro permisos predeterminados para las siguientes acciones:

- **add:** los usuarios con este permiso pueden agregar una instancia del modelo.
- **delete:** los usuarios con este permiso pueden eliminar una instancia del modelo.
- **change:** los usuarios con este permiso pueden actualizar una instancia del modelo.
- **view:** los usuarios con este permiso pueden ver instancias de este modelo. Este permiso era muy esperado, y finalmente se agregó en Django 2.1.

Los nombres de permisos siguen una convención de nomenclatura muy específica: `<app>.<action>_<modelname>`.

Desglose:

- `<app>` es el nombre de la aplicación. Por ejemplo, el modelo User se importa desde la aplicación `auth` (`django.contrib.auth`).
- `<action>` es una de las acciones anteriores (add, delete, change o view).
- `<modelname>` es el nombre del modelo, en minúsculas.

Conocer esta convención de nomenclatura puede ayudarlo a administrar los permisos más fácilmente. Por ejemplo: el nombre del permiso para cambiar un usuario es `auth.change_user`.

EVITAR QUE LOS NO SUPER USUARIOS CONCEDAN DERECHOS DE SUPER USUARIO

Super usuario es un permiso muy fuerte que no debe otorgarse a la ligera. Sin embargo, cualquier usuario con un permiso de cambio en el modelo `User`, puede convertir a cualquier usuario en super usuario, incluidos ellos mismos. Esto va en contra del propósito del sistema de permisos, por lo que se desea resolver.

Según el ejemplo anterior, para evitar que los no superusuarios se conviertan en superusuarios, se agrega la siguiente restricción:

```
from typing import Set

from django.contrib import admin
from django.contrib.auth.models import User
from django.contrib.auth.admin import UserAdmin

@admin.register(User)
class CustomUserAdmin(UserAdmin):
    def get_form(self, 2equest, obj=None, **kwargs):
        form = super().get_form(2equest, obj, **kwargs)
        is_superuser = 2equest.user.is_superuser
        disabled_fields = set()  # type: Set[str]

        if not is_superuser:
            disabled_fields |= {
                'username',
                'is_superuser',
            }

        for f in disabled_fields:
            if f in form.base_fields:
                form.base_fields[f].disabled = True

```

```
24          return form
```

Además del ejemplo anterior, realiza las siguientes adiciones:

1. Inicializaste un conjunto vacío `disabled_fields` que contendrá los campos para deshabilitarlos. Set es una estructura de datos que contiene valores únicos. Tiene sentido usar un conjunto en este caso, porque solo necesita deshabilitar un campo una vez. El operador |= se utiliza para realizar una actualización. Para obtener más información acerca de los conjuntos, consulte en: conjuntos en Python.
2. A continuación, si el usuario es un super usuario, agrega dos campos al conjunto (`username` del ejemplo anterior, y `is_superuser`). Evitarán que los no superusuarios se conviertan en superusuarios.
3. Por último, itera sobre los campos del conjunto, los marca como deshabilitados y devuelve el formulario.