Los mixins son métodos que le heredamos a una clase. Estas funcionalidades se heredan de primero en la clase.

class CategoryList(IsSuperuserMixin,ListView):

Todo lo que escribamos en el mixin se heredara en la clase hija, si existe algun metodo igual en la clase hija, se añadiran los datos del mixin

# Hereda de object que es la clase base de toda clase Python  
class IsSuperuserMixin(object):  
 def dispatch(self, request, \*args, \*\*kwargs):  
 if request.user.is\_superuser:  
 return super().dispatch(request, \*args, \*\*kwargs)  
 return redirect('index')  
  
 #Si tenemos el metodo get\_context\_data tambien tendra estos datos  
 def get\_context\_data(self, \*\*kwargs):  
 context = super().get\_context\_data(\*\*kwargs)  
 context['date\_now'] = datetime.now()  
 return context

# PermissionRequiredMixin

<https://docs.djangoproject.com/en/4.1/topics/auth/default/#the-permissionrequiredmixin-mixin>

Esta es una customizacion del PermisionRequiredMixin, la única diferencia es que esta personaliado para que te redireccione a donde tu quieras.

class ValidatePermissionRequiredMixin(object):  
 permission\_required = ''  
 url\_redirect = None  
  
 #Con esto convertimos los permisos en una lista en todos los casos  
 def get\_perms(self):  
 #Si es un str, convierte perms en una lista  
 if isinstance(self.permission\_required, str):  
 perms = (self.permission\_required,)  
 #Caso contrario, devuelve perms  
 else:  
 perms = self.permission\_required  
 return perms  
  
 def get\_url\_redirect(self):  
 if self.url\_redirect is None:  
 return reverse\_lazy('accounts:login')  
 return self.url\_redirect  
  
 def dispatch(self, request, \*args, \*\*kwargs):  
 #Validamos si el usuario tiene esa lista de permisos  
 if request.user.has\_perms(self.get\_perms()):  
 return super().dispatch(request, \*args, \*\*kwargs)  
 return redirect(self.get\_url\_redirect())

En la vista, debes pasar los premisos como una tuple o un text para un unico permiso. Tambien puedes pasarle el URL redirect

class CategoryList(ValidatePermissionRequiredMixin ,ListView):  
 permission\_required = 'erp.view\_category'

# PermissionRequiredMixin v2

1. 1 permiso a la vez

Este valida los permisos provenientes de un grupo

Asi funciuona cuandoenviamos un solo permiso de esta forma:

permission\_required = 'delete\_product'

def dispatch(self, request, \*args, \*\*kwargs):  
 #obtenemos el request actual  
 request = get\_current\_request()  
 #Verificamos que grupo este en request.session  
 if 'group' in request.session:  
 #group = request.session['group']  
 group = Group.objects.get(pk=1)  
 #Si el permiso requerido esta dentro de los permisos de ese grupo  
 if group.permissions.filter(codename=self.permission\_required):  
 return super().dispatch(request, \*args, \*\*kwargs)  
 #Validamos si el usuario tiene esa lista de permisos  
 # if request.user.has\_perms(self.get\_perms()):  
 messages.error(request,'No tienes los permisos para acceder a este modulo.')  
 return redirect(self.get\_url\_redirect())