# Installing and Configuring TINYMCE in Django

installation of tinymce

pip install django-tinymce4-lite.

settings.py

INSTALLED\_APPS = [

...

'tinymce',

]

TINYMCE\_DEFAULT\_CONFIG = {

'height': 360,

'width': 1120,

'cleanup\_on\_startup': True,

'custom\_undo\_redo\_levels': 20,

'selector': 'textarea',

'theme': 'modern',

'plugins': '''

textcolor save link image media preview codesample contextmenu

table code lists fullscreen insertdatetime nonbreaking

contextmenu directionality searchreplace wordcount visualblocks

visualchars code fullscreen autolink lists charmap print hr

anchor pagebreak

''',

'toolbar1': '''

fullscreen preview bold italic underline | fontselect,

fontsizeselect | forecolor backcolor | alignleft alignright |

aligncenter alignjustify | bullist numlist table |

| link image media | codesample | hr visualchars |

''',

'contextmenu': 'formats | link image',

'remove\_linebreaks': False

}

urls.py

urlpatterns = [

path('tinymce/', include('tinymce.urls')),

]

forms.py

from .models import BlogPost

from django import forms

from tinymce import TinyMCE

class TinyMCEWidget(TinyMCE):

def use\_required\_attribute(self, \*args):

return False

class BlogForm(forms.ModelForm):

title = forms.CharField()

content = forms.CharField(widget=TinyMCEWidget(

attrs={'required': False, 'cols': 20, 'rows': 10})

)

image = forms.ImageField()

class Meta:

model = BlogPost

fields = [

'title',

'content',

'image'

]

views.py

class PostCreateView(SuccessMessageMixin, CreateView):

model = BlogPost

form\_class = BlogForm

template\_name = 'blogs/blog-form.html'

success\_message = 'Post added successfully!'