

Topic: Image Compression 26: Thumbnail, Size, Automatic Download

Speaker: Udemy Instructor Rathan Kumar | Notebook: Django: Automating Common Tasks



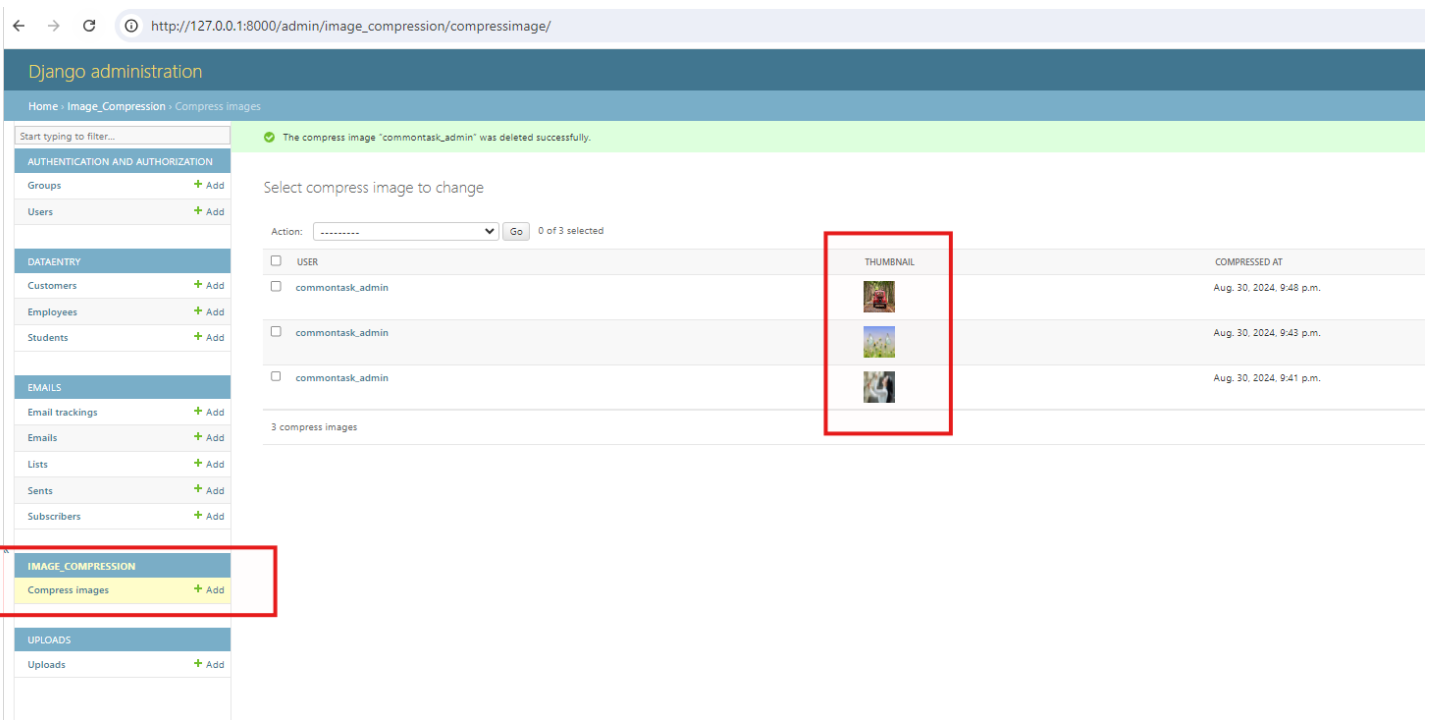
1. We need some thumbnails for our images on the admin dashboard. So, update the ADMIN.PY as:

```
AutomatingCommonTasks

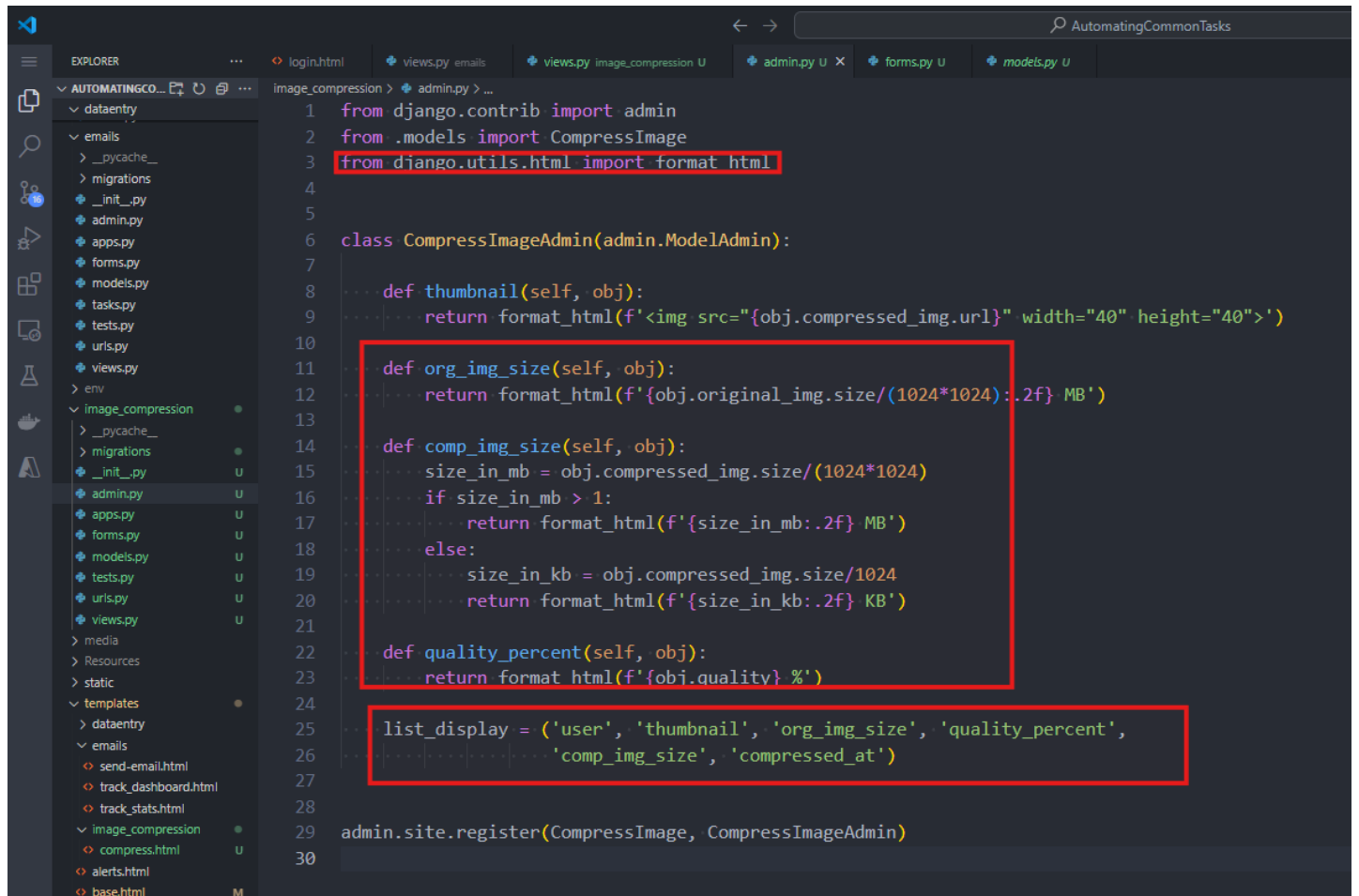
EXPLORER
AUTOMATINGCOMMONTASKS
  autocommontasks_main
    wsgi.py
    dataentry
      __pycache__
      management
      migrations
      __init__.py
      admin.py
      apps.py
      models.py
      tasks.py
      tests.py
      uris.py
      utils.py
      views.py
    emails
      __pycache__
      migrations
      __init__.py
      admin.py
      apps.py
      forms.py
      models.py
      tasks.py
      tests.py
      uris.py
      views.py
    env
    image_compression
      __pycache__
      migrations
      __init__.py
      admin.py
      apps.py
      forms.py

image_compression > admin.py > ...
1  from django.contrib import admin
2  from .models import CompressImage
3  from django.utils.html import format_html
4
5
6  class CompressImageAdmin(admin.ModelAdmin):
7      ... def thumbnail(self, obj):
8          ... return format_html(f'')
9
10     ... list_display = ('user', 'thumbnail', 'compressed_at')
11
12
13  admin.site.register(CompressImage, CompressImageAdmin)
14
```

2. Reload your admin panel:



3. To display the original and compressed sizes of the image, in the ADMIN.PY, we update:



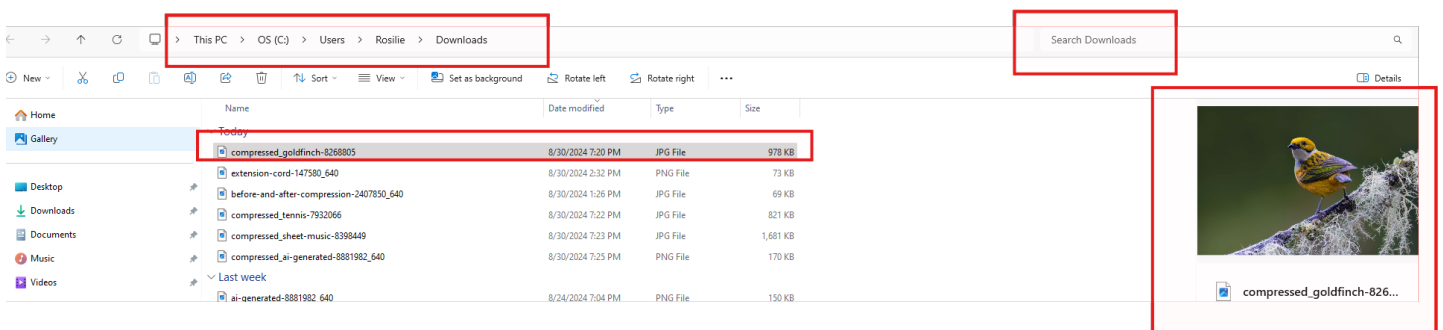
4. To allow the user to download automatically the compressed image, we update our VIEWS.PY

```

2 from .forms import CompressImageForm
3 from PIL import Image
4 import io
5 from django.contrib import messages
6 from django.http import HttpResponseRedirect
7
8
9 def compress(request):
10     user = request.user
11     if request.method == 'POST':
12         form = CompressImageForm(request.POST, request.FILES)
13         if form.is_valid():
14             original_img = form.cleaned_data['original_img']
15             quality = form.cleaned_data['quality']
16
17             # temporarily saves the form
18             compressed_image = form.save(commit=False)
19             compressed_image.user = user
20
21             # perform the compression
22             img = Image.open(original_img)
23
24             # set the image format based on the uploaded image's format
25             output_format = img.format
26
27             buffer = io.BytesIO()
28             img.save(buffer, format=output_format, quality=quality)
29             buffer.seek(0)
30
31             # save teh compressed image inside the model with filename format
32             compressed_image.compressed_img.save(
33                 f'compressed_{original_img}', buffer
34             )
35             # return redirect('compress')
36             # Automatically download the compressed file not as binary value but as formatted image
37             response = HttpResponseRedirect(
38                 buffer.getvalue(), content_type=f'image/{output_format.lower()}'
39             )
40             response['Content-Disposition'] = f'attachment;filename=compressed_{original_img}'
41             return response
42
43     else:
44         form = CompressImageForm()
45         context = {
46             'form': form,
47         }
48         return render(request, 'image_compression/compress.html', context)

```

5. Open your downloads folder for the images:



6. Push your changes to Github.