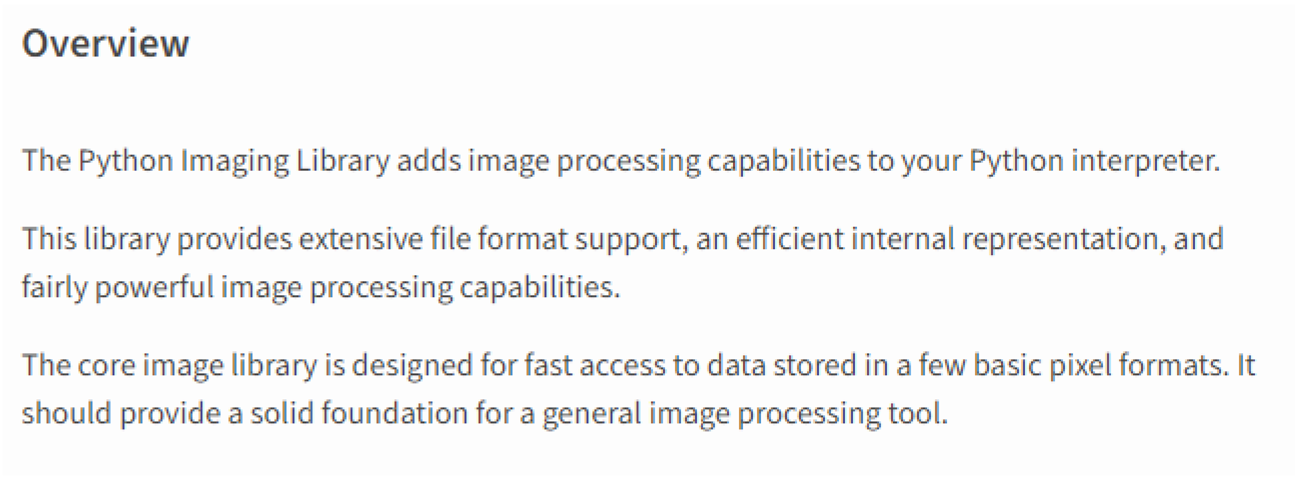
Topic: Image Compression 25: Setup, Model & Logic

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



Overview

The Python Imaging Library adds image processing capabilities to your Python interpreter.

This library provides extensive file format support, an efficient internal representation, and fairly powerful image processing capabilities.

The core image library is designed for fast access to data stored in a few basic pixel formats. It should provide a solid foundation for a general image processing tool.

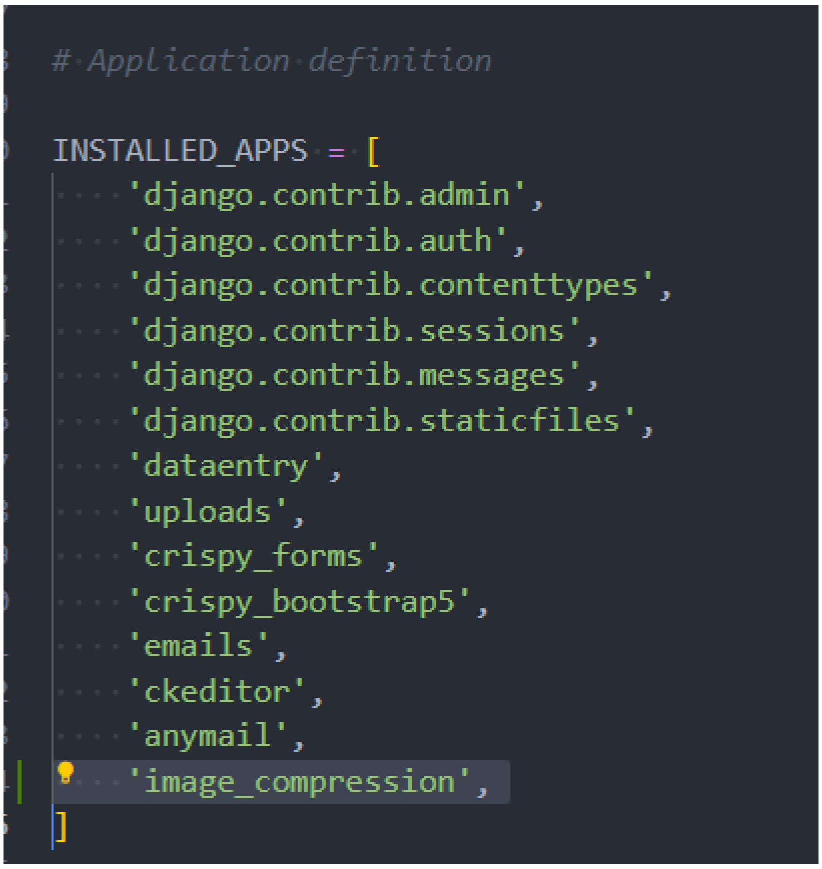
1. Go to Pillow documentation and install the Django package in the Djngo server terminal and in Celery terminal

```
$ pip install pillow
```

2. In the Django server, we create a new app IMAGE_COMPRESSION

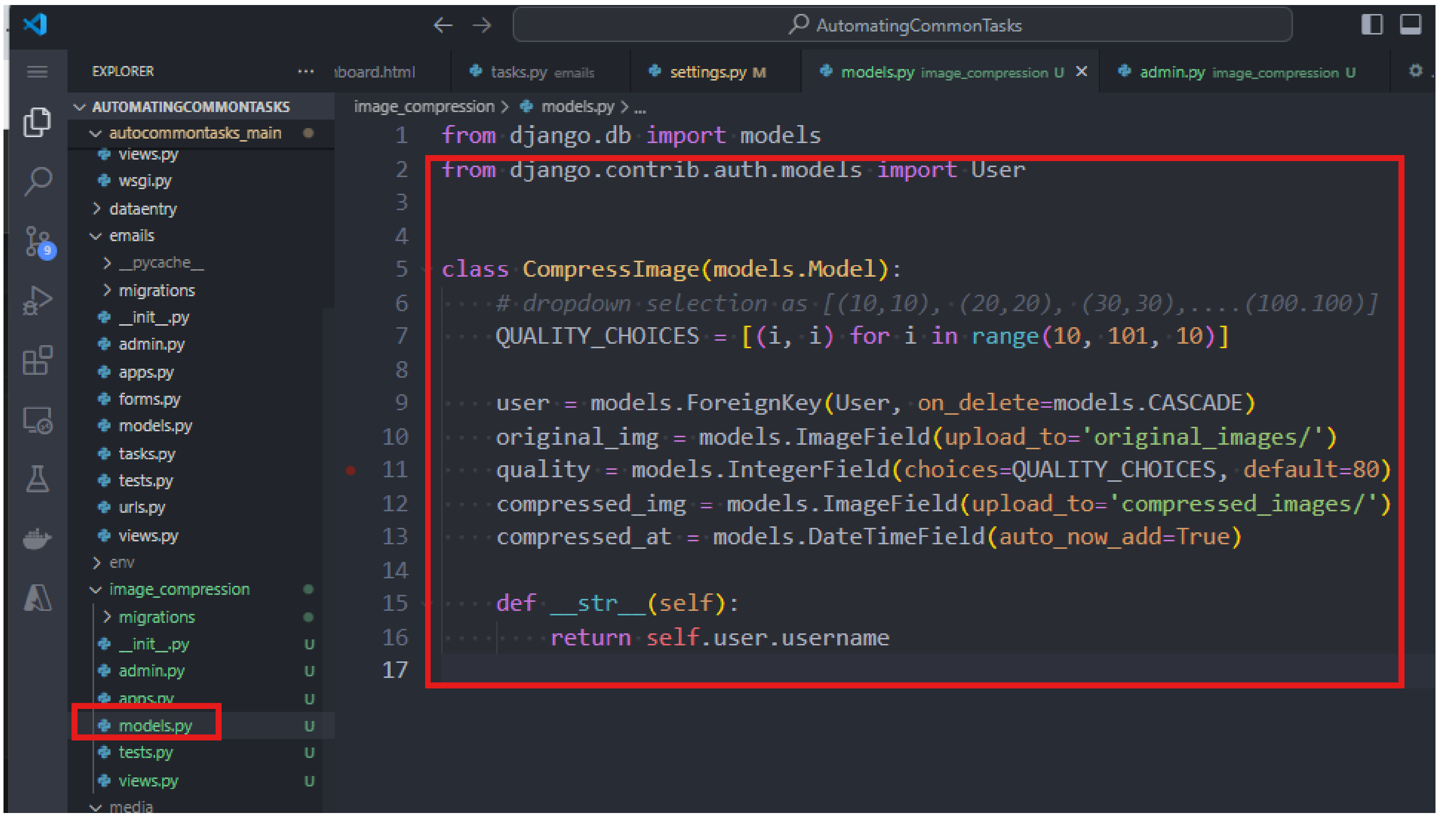
```
$ python manage.py startapp image_compression
```

3. Register the new app in SETTINGS.PY INSTALLED_APPS



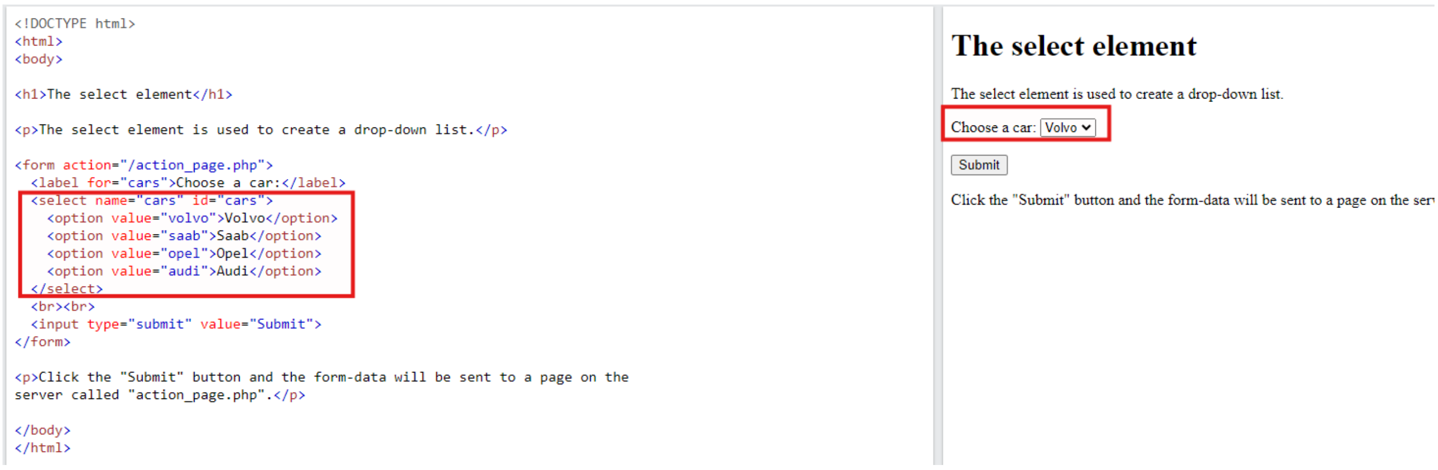
```python
# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'dataentry',
    'uploads',
    'crispy_forms',
    'crispy_bootstrap5',
    'emails',
    'ckeditor',
    'anymail',
    'image_compression',
]
```

4. Create the model in MODELS.PY



```python
from django.db import models
from django.contrib.auth.models import User


class CompressImage(models.Model):
    # dropdown selection as [(10,10), (20,20), (30,30),....(100.100)]
    QUALITY_CHOICES = [(i, i) for i in range(10, 101, 10)]

    user = models.ForeignKey(User, on_delete=models.CASCADE)
    original_img = models.ImageField(upload_to='original_images/')
    quality = models.IntegerField(choices=QUALITY_CHOICES, default=80)
    compressed_img = models.ImageField(upload_to='compressed_images/')
    compressed_at = models.DateTimeField(auto_now_add=True)

    def __str__(self):
        return self.user.username

```

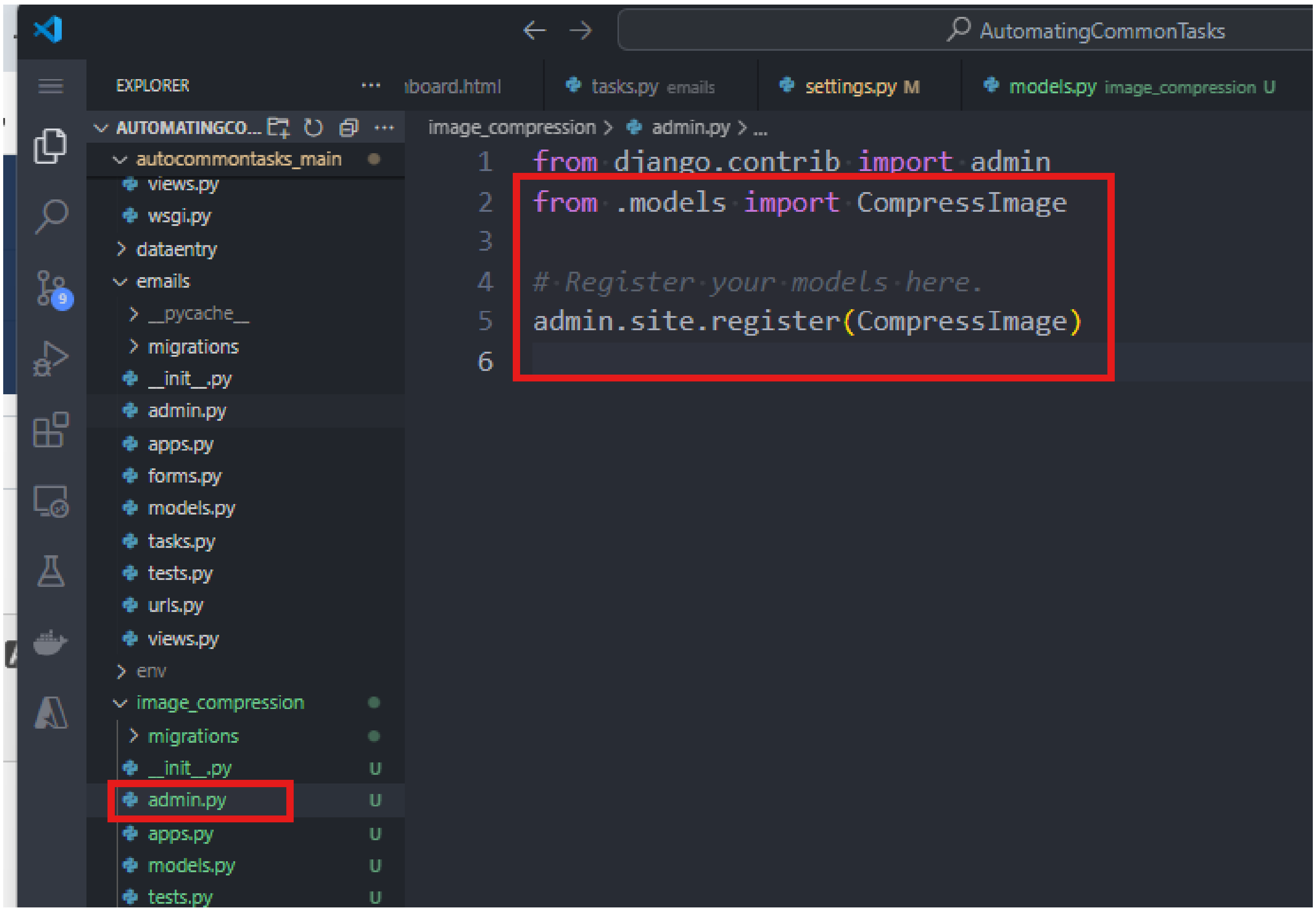
To make the QUALITY field a drop-down option, we use the SELECT TAG. Where we see the VALUE AND THE LABEL (which is displayed as a user option in the dropdown)



```
<!DOCTYPE html>
<html>
<body>

<h1>The select element</h1>

<p>The select element is used to create a drop-down list.</p>

<form action="/action_page.php">
  <label for="cars">Choose a car:</label>
  <select name="cars" id="cars">
    <option value="volvo">Volvo</option>
    <option value="saab">Saab</option>
    <option value="opel">Opel</option>
    <option value="audi">Audi</option>
  </select>
  <br><br>
  <input type="submit" value="Submit">
</form>

<p>Click the "Submit" button and the form-data will be sent to a page on the
server called "action_page.php".</p>

</body>
</html>
```

The select element
The select element is used to create a drop-down list.
Choose a car: Volvo
Submit
Click the "Submit" button and the form-data will be sent to a page on the ser

5. Register the model for our ADMIN panel. Update ADMIN.PY:

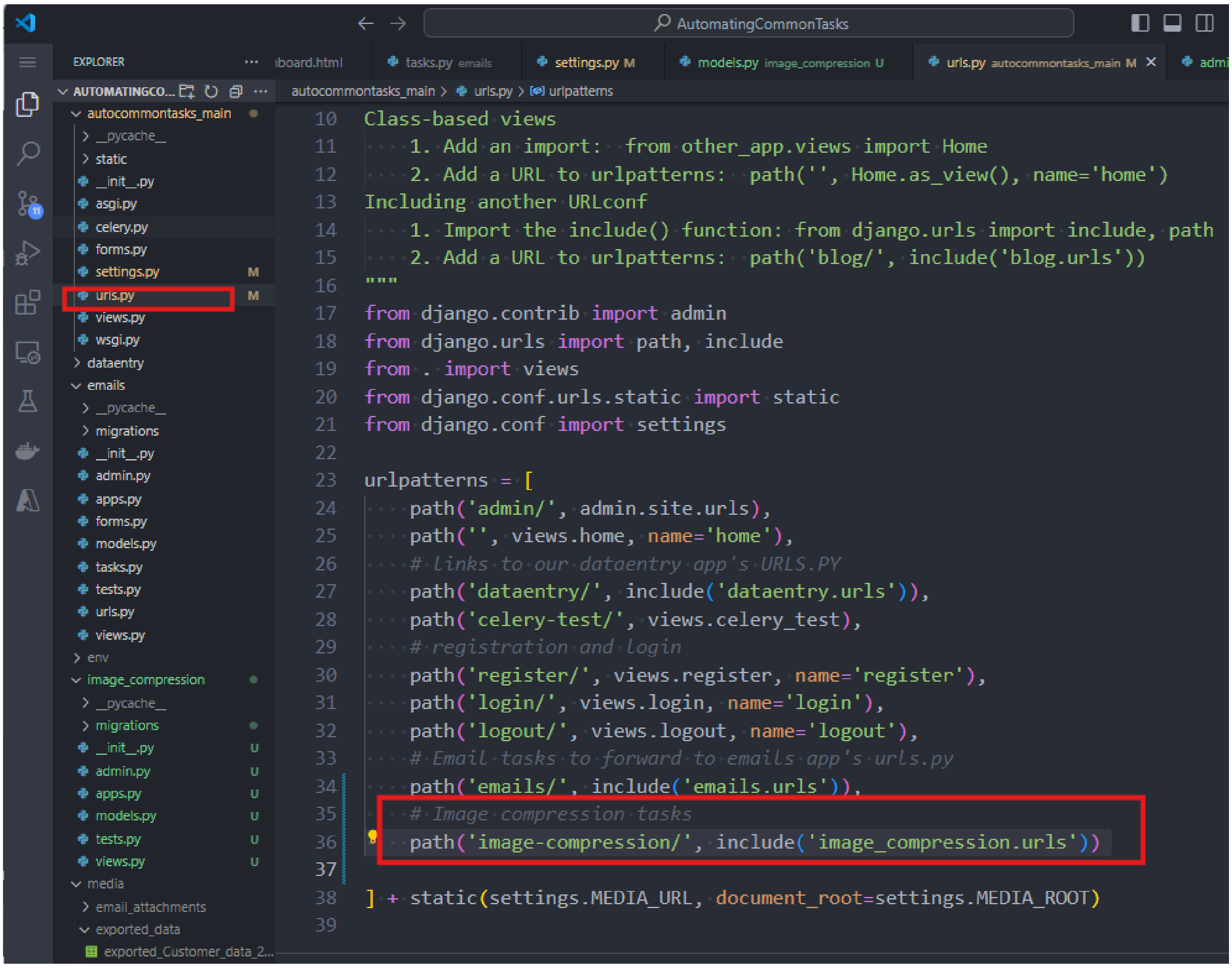


```python
from django.contrib import admin
from .models import CompressImage

# Register your models here.
admin.site.register(CompressImage)
```

6. Make the necessary model migrations

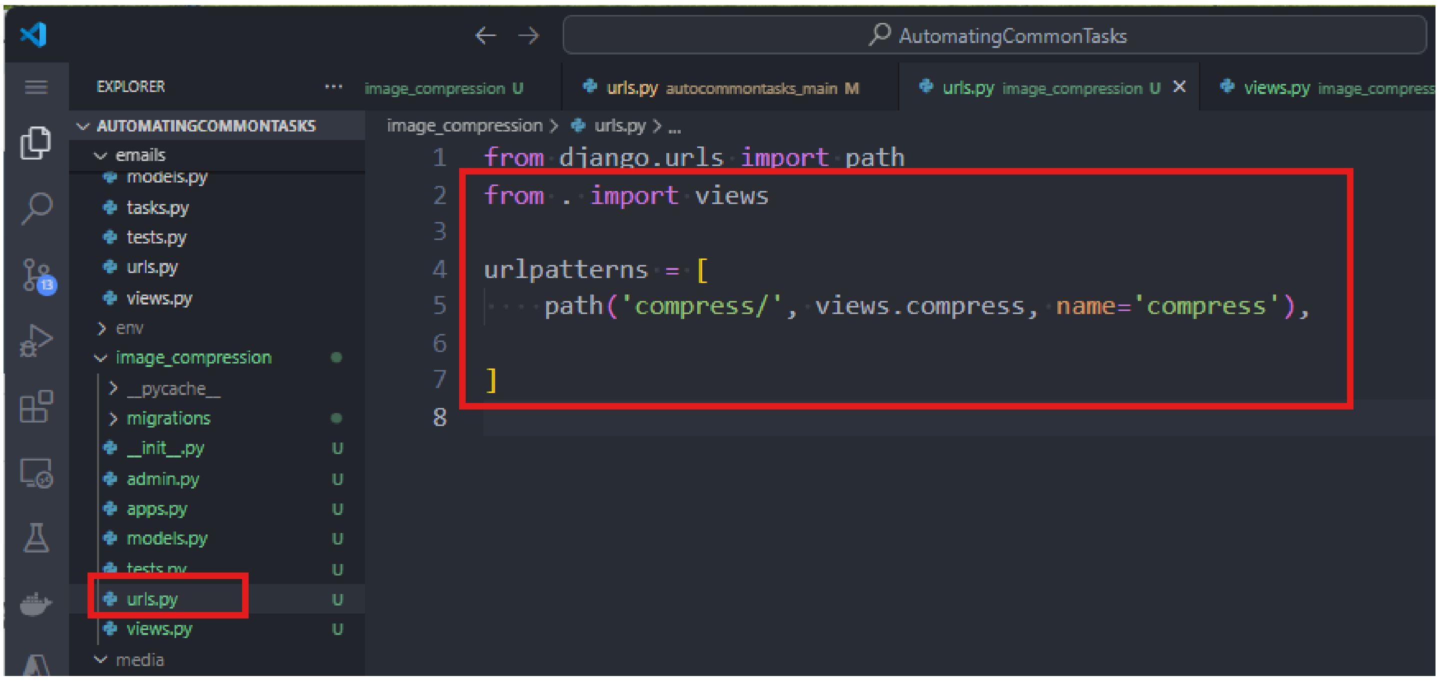
```
$ python manage.py makemigrations

$ python manage.py migrate
```

7. Create the new URL pattern. Since we have a new app, we need to create a new pattern in our main project's URLS.PY.



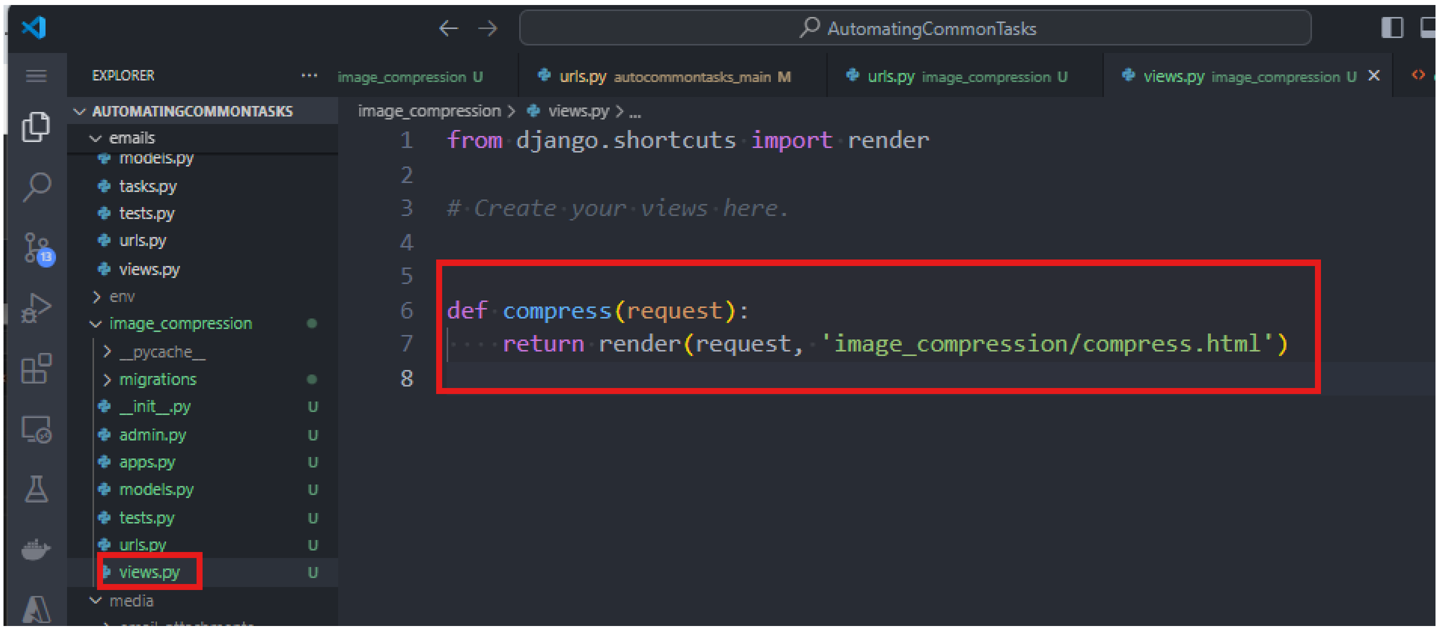
```python
Class-based views
    1. Add an import:  from other_app.views import Home
    2. Add a URL to urlpatterns:  path('', Home.as_view(), name='home')
Including another URLconf
    1. Import the include() function: from django.urls import include, path
    2. Add a URL to urlpatterns:  path('blog/', include('blog.urls'))
"""
from django.contrib import admin
from django.urls import path, include
from . import views
from django.conf.urls.static import static
from django.conf import settings

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', views.home, name='home'),
    # links to our dataentry app's URLS.PY
    path('dataentry/', include('dataentry.urls')),
    path('celery-test/', views.celery_test),
    # registration and login
    path('register/', views.register, name='register'),
    path('login/', views.login, name='login'),
    path('logout/', views.logout, name='logout'),
    # Email tasks to forward to emails app's urls.py
    path('emails/', include('emails.urls')),
    # Image compression tasks
    path('image-compression/', include('image_compression.urls'))

] + static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

Then, create a new URLS.PY file in our new app for image-compression-related URL paths.



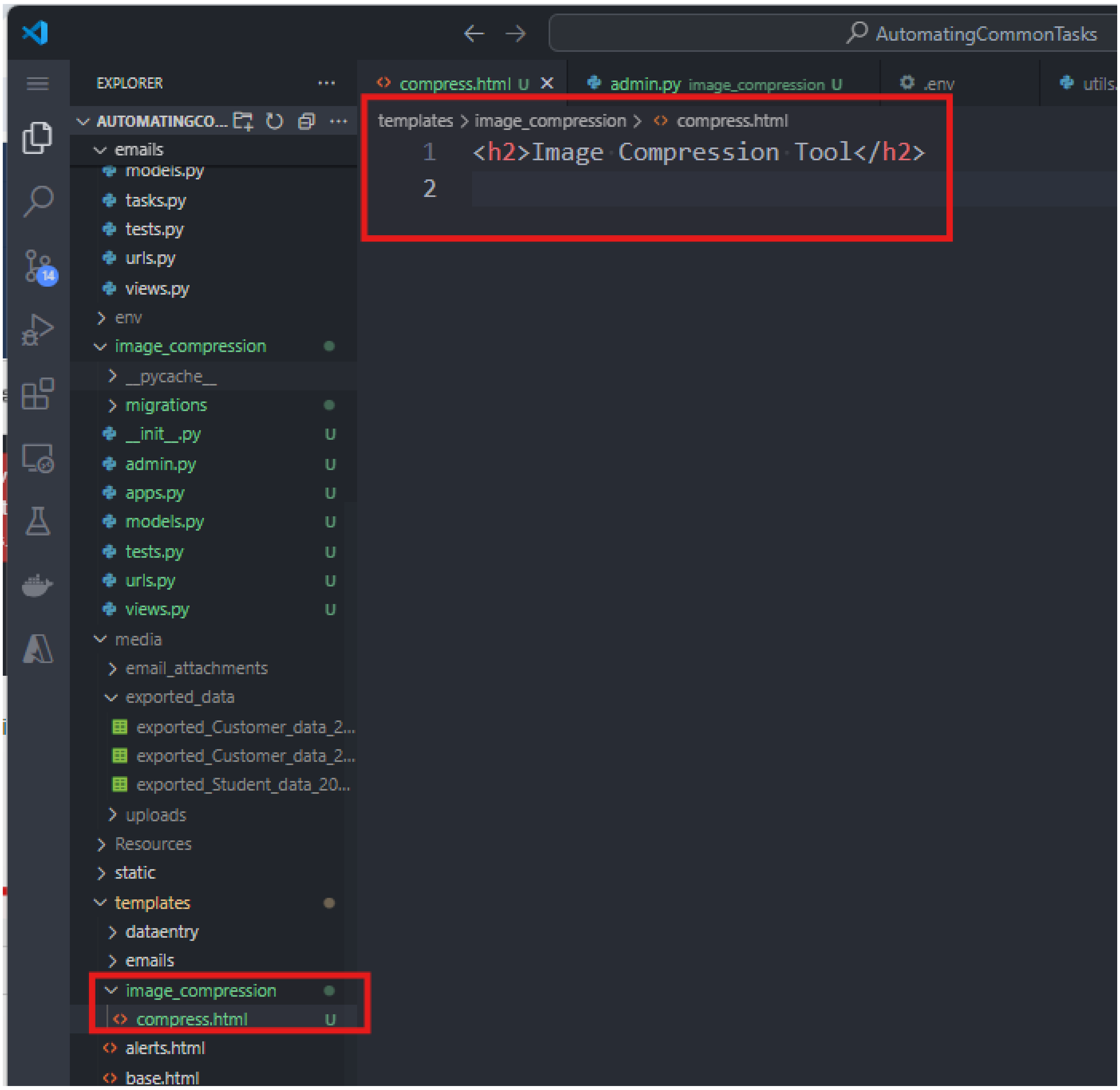
```python
from django.urls import path
from . import views

urlpatterns = [
    path('compress/', views.compress, name='compress'),

]
```

8. Create the function in the VIEWS.PY



```
from django.shortcuts import render

# Create your views here.

def compress(request):
    return render(request, 'image_compression/compress.html')
```

9. Create a new FOLDER called IMAGE_COMPRESSION, and in this folder, create a new file, COMPRESS.HTML



```
<h2>Image Compression Tool</h2>
```

10. Update the HOME.HTML to call this new web page.

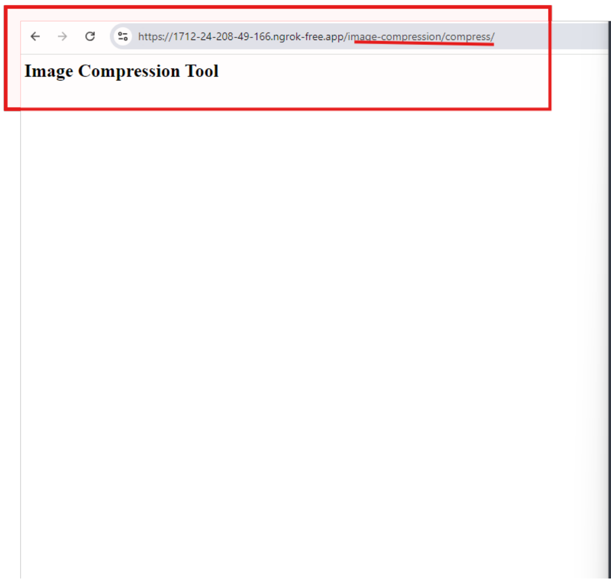
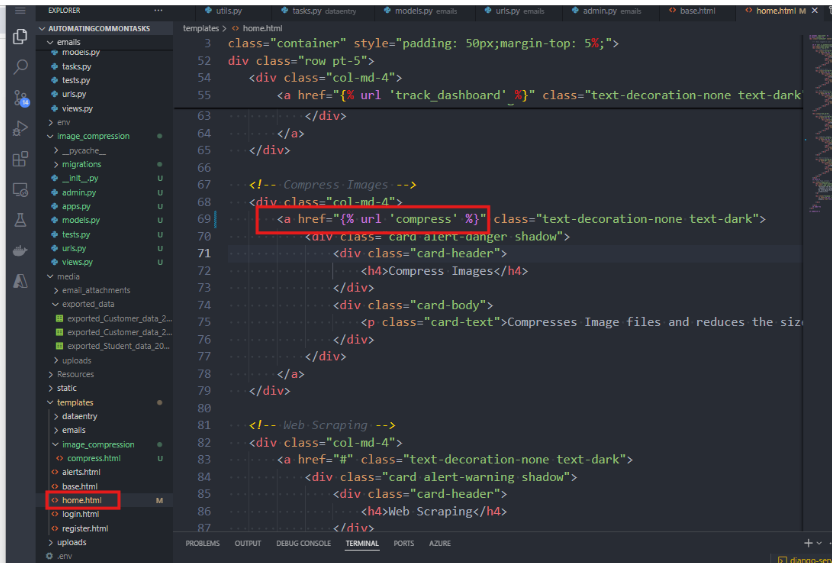
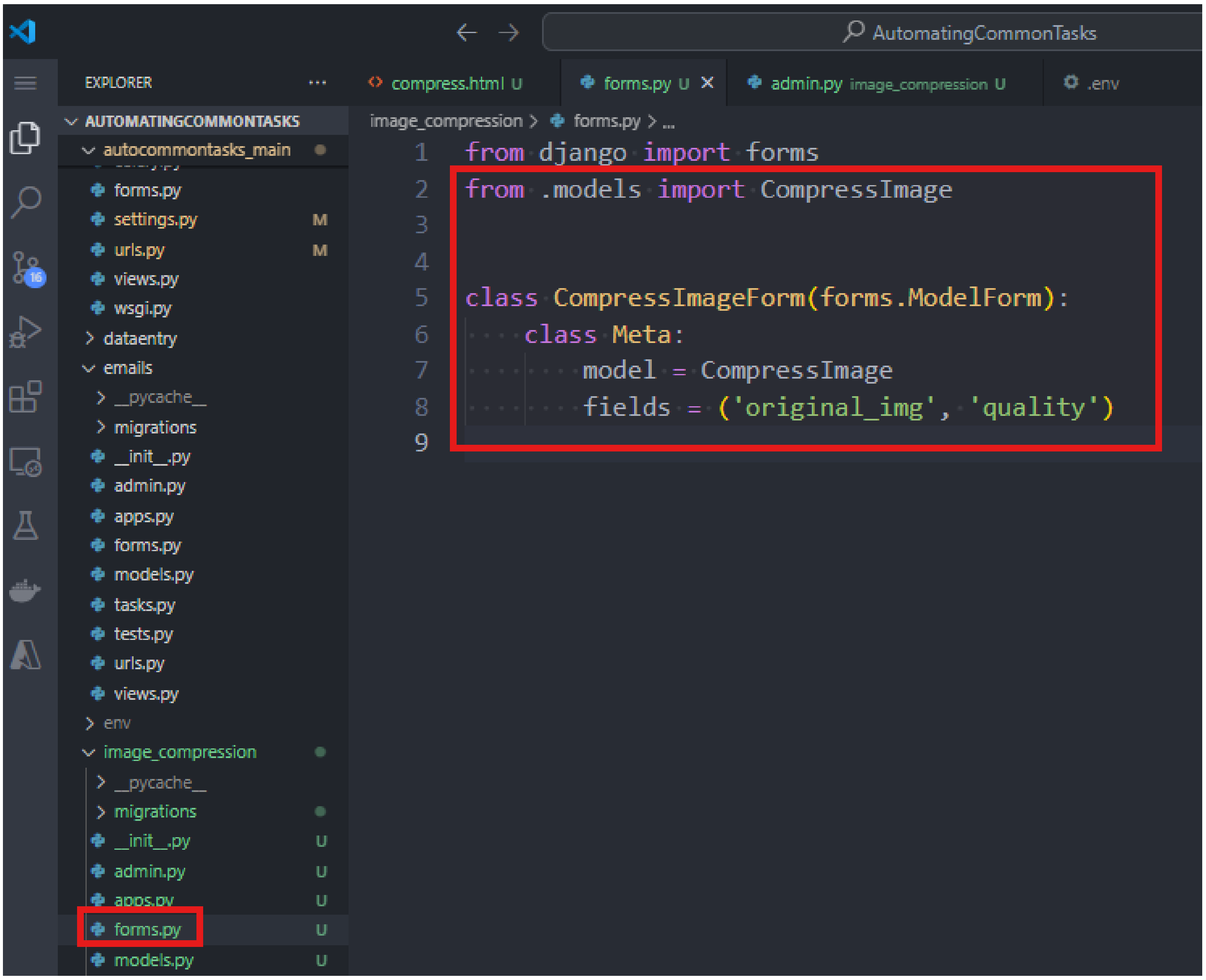

https://1712-24-208-49-166.ngrok-free.app/image-compression/compress/

Image Compression Tool


templates > home.html

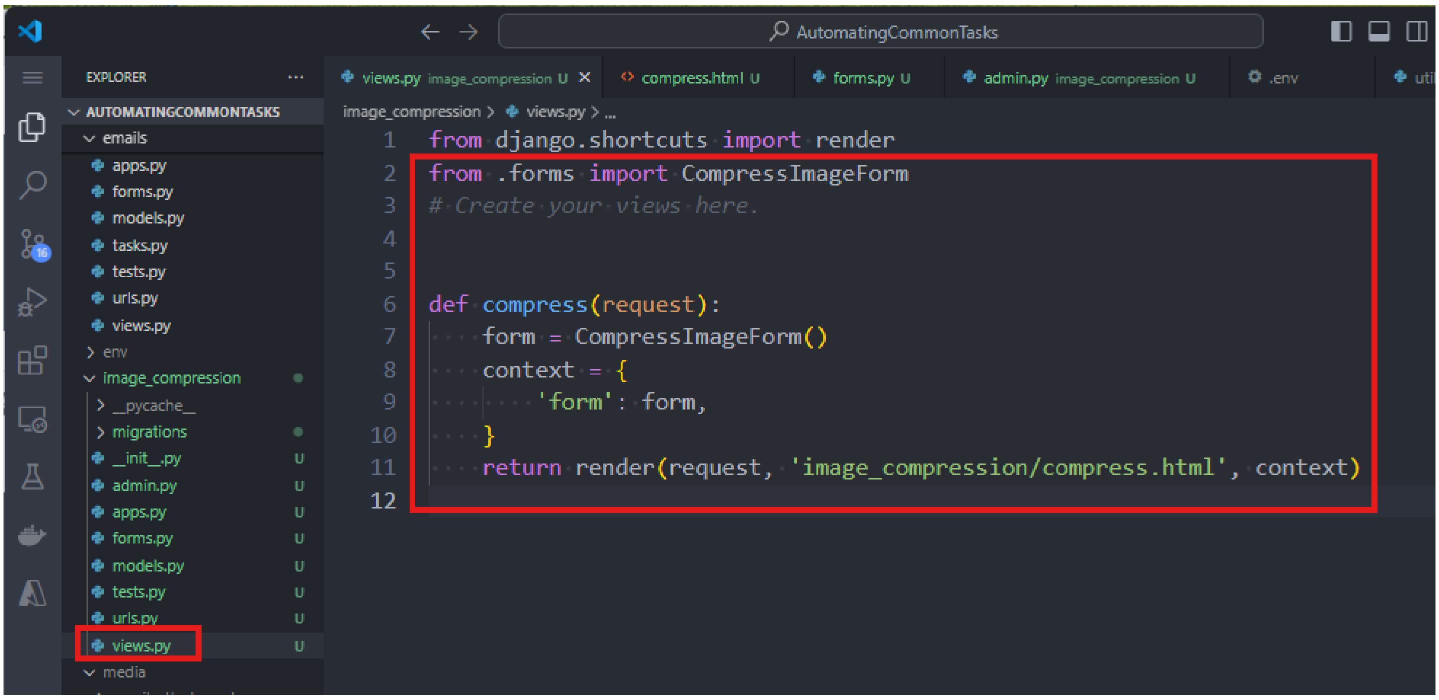
```html
class="container" style="padding: 50px;margin-top: 5%;">
div class="row pt-5">
  <div class="col-md-4">
    <a href="{% url 'track_dashboard' %}" class="text-decoration-none text-dark
        </div>
    </a>
  </div>

  <!-- Compress Images -->
  <div class="col-md-4">
    <a href="{% url 'compress' %}" class="text-decoration-none text-dark">
      <div class="card alert-danger shadow">
        <div class="card-header">
          <h4>Compress Images</h4>
        </div>
        <div class="card-body">
          <p class="card-text">Compresses Image files and reduces the siz
        </div>
      </div>
    </a>
  </div>

  <!-- Web Scraping -->
  <div class="col-md-4">
    <a href="#" class="text-decoration-none text-dark">
      <div class="card alert-warning shadow">
        <div class="card-header">
          <h4>Web Scraping</h4>
        </div>
```

11. Create a FORMS.PY


image_compression > forms.py > ...

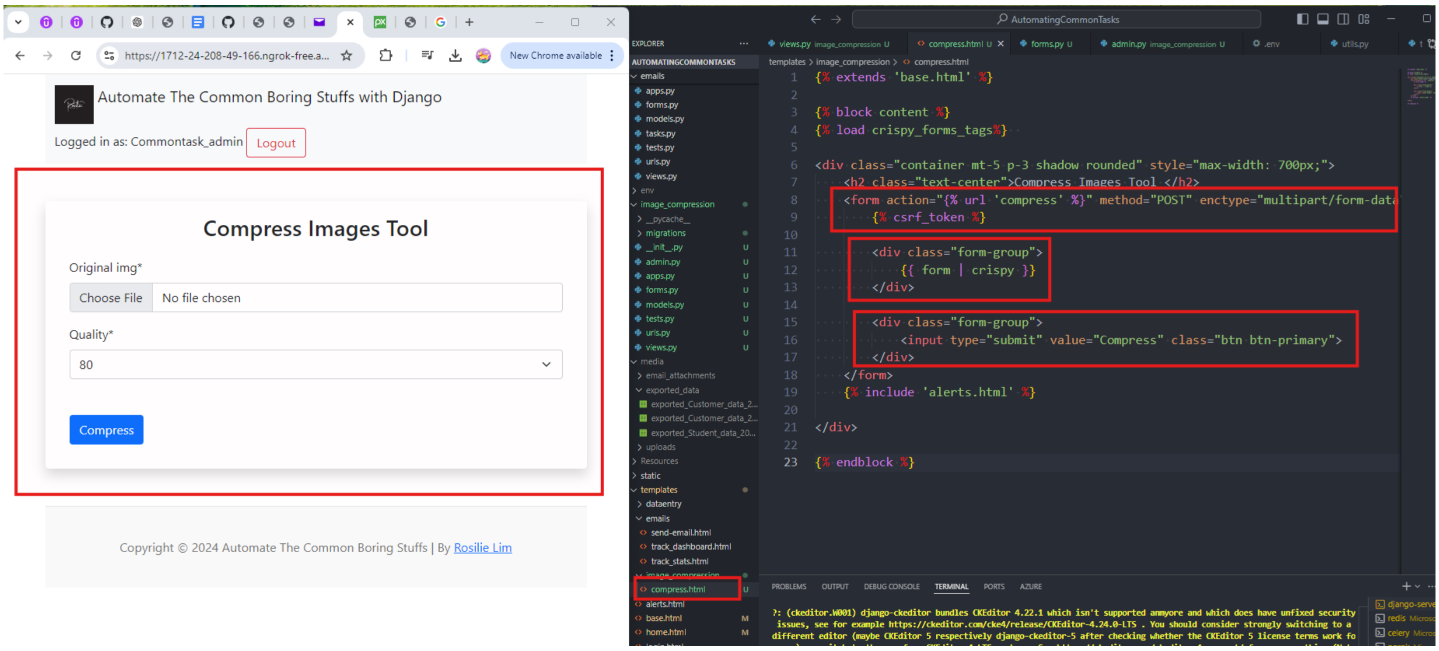
```python
from django import forms
from .models import CompressImage


class CompressImageForm(forms.ModelForm):
    class Meta:
        model = CompressImage
        fields = ('original_img', 'quality')
```

12. Call our form using our VIEWS.PY:



```python
from django.shortcuts import render
from .forms import CompressImageForm
# Create your views here.


def compress(request):
    form = CompressImageForm()
    context = {
        'form': form,
    }
    return render(request, 'image_compression/compress.html', context)
```

13. In the COMPRESS.HTML, update as:



Automate The Common Boring Stuffs with Django

Logged in as: Commontask_admin Logout

Compress Images Tool

Original img*

Choose File No file chosen

Quality*

80

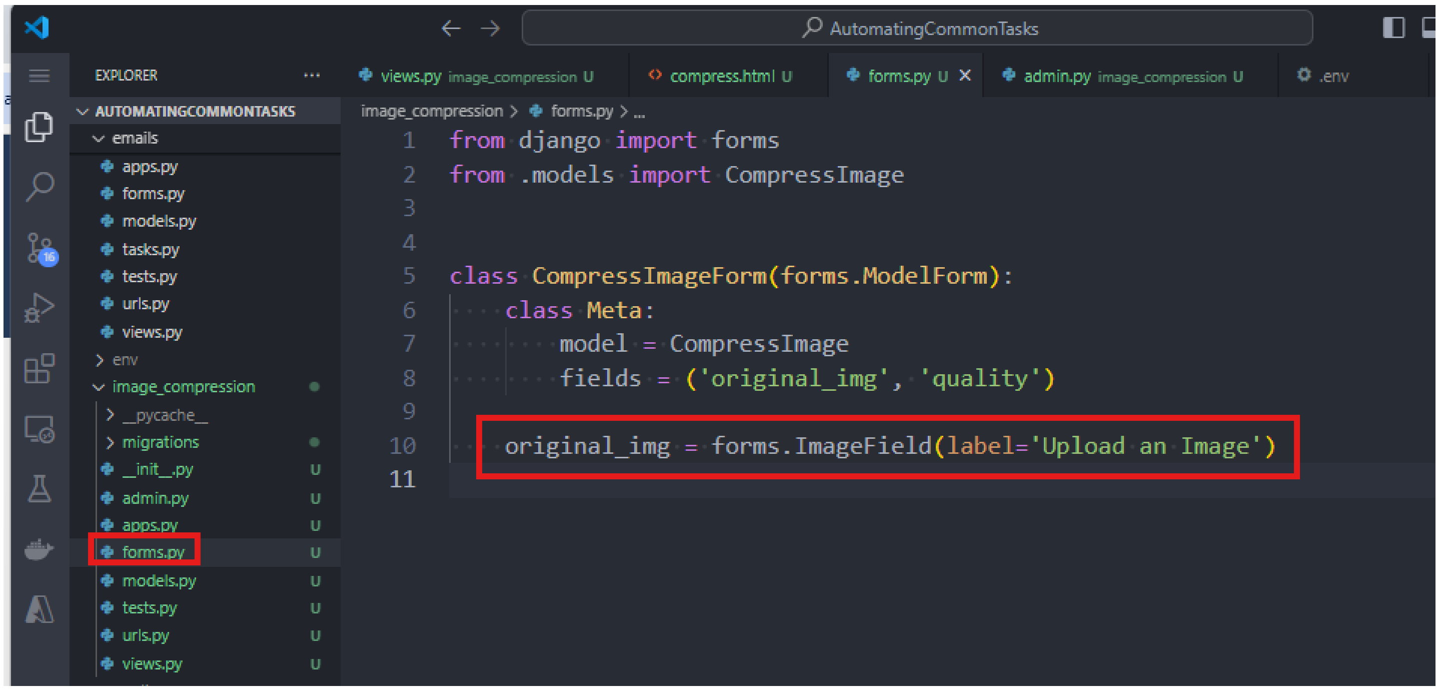
Compress

Copyright © 2024 Automate The Common Boring Stuffs | By Rosilie Lim

```html
{% extends 'base.html' %}

{% block content %}
{% load crispy_forms_tags%}

<div class="container mt-5 p-3 shadow rounded" style="max-width: 700px;">
    <h2 class="text-center">Compress Images Tool </h2>
    <form action="{% url 'compress' %}" method="POST" enctype="multipart/form-data
        {% csrf_token %}

        <div class="form-group">
            {{ form | crispy }}
        </div>

        <div class="form-group">
            <input type="submit" value="Compress" class="btn btn-primary">
        </div>
    </form>
    {% include 'alerts.html' %}

</div>

{% endblock %}
```

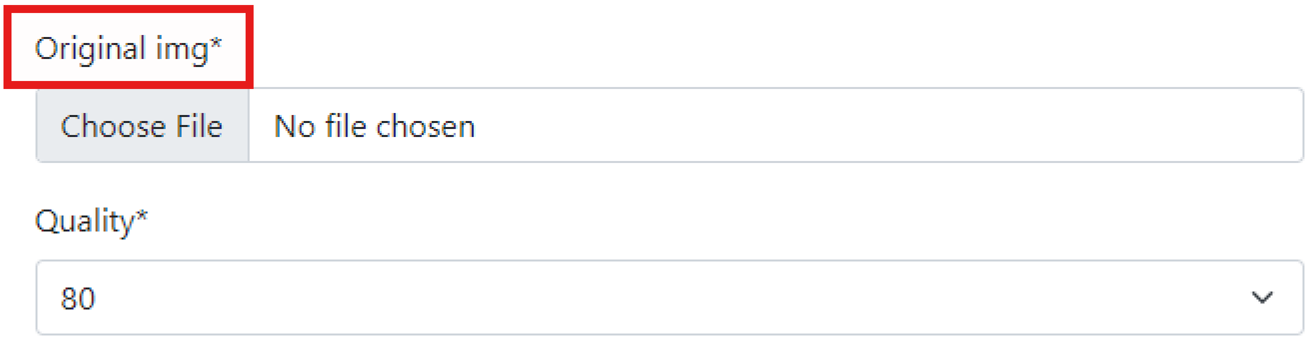
```
?: (ckeditor.W001) django-ckeditor bundles CKEditor 4.22.1 which isn't supported anmyore and which does have unfixed security
 issues, see for example https://ckeditor.com/cke4/release/CKEditor-4.24.0-LTS . You should consider strongly switching to a
different editor (maybe CKEditor 5 respectively django-ckeditor-5 after checking whether the CKEditor 5 license terms work fo
```

14. We can change the label of the field on the form, so update FORMS.PY and add the line:

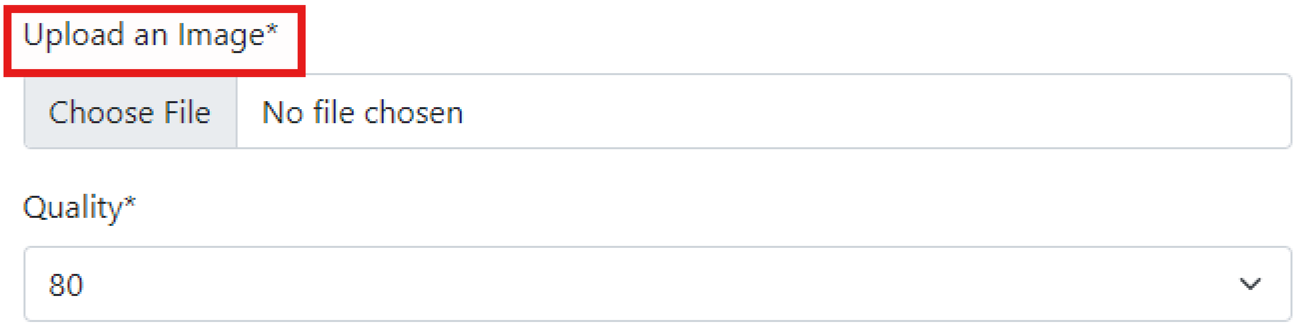


```python
from django import forms
from .models import CompressImage


class CompressImageForm(forms.ModelForm):
    class Meta:
        model = CompressImage
        fields = ('original_img', 'quality')

    original_img = forms.ImageField(label='Upload an Image')
```

FROM:

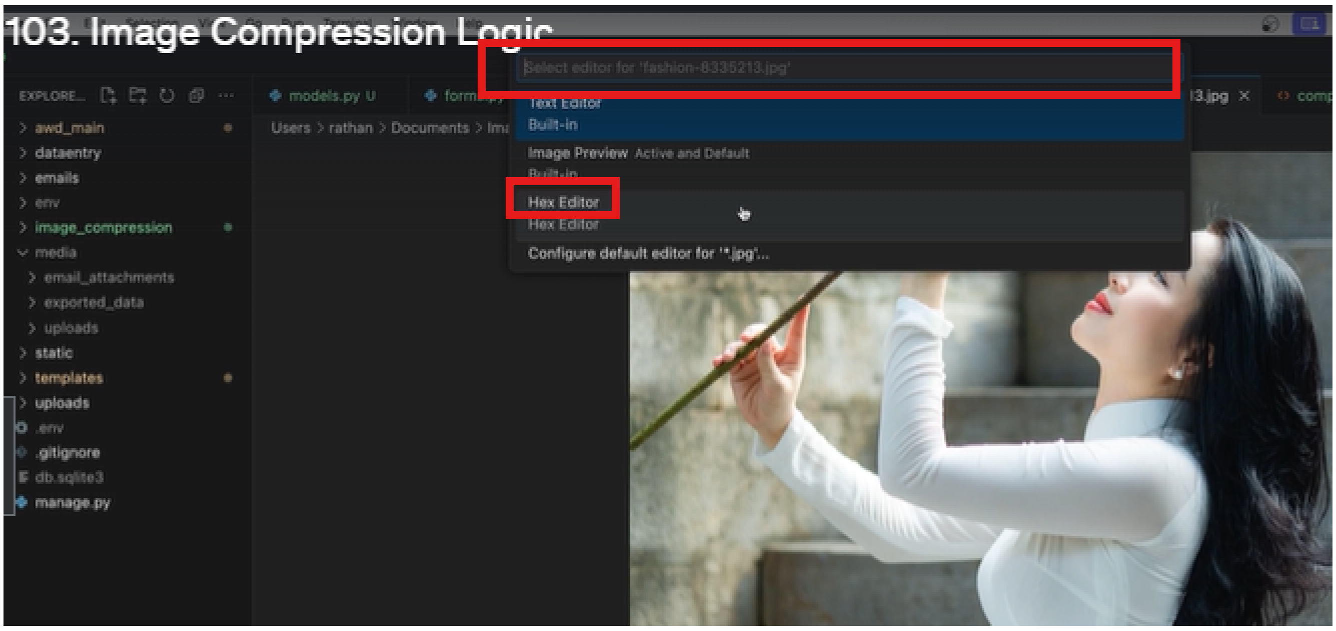


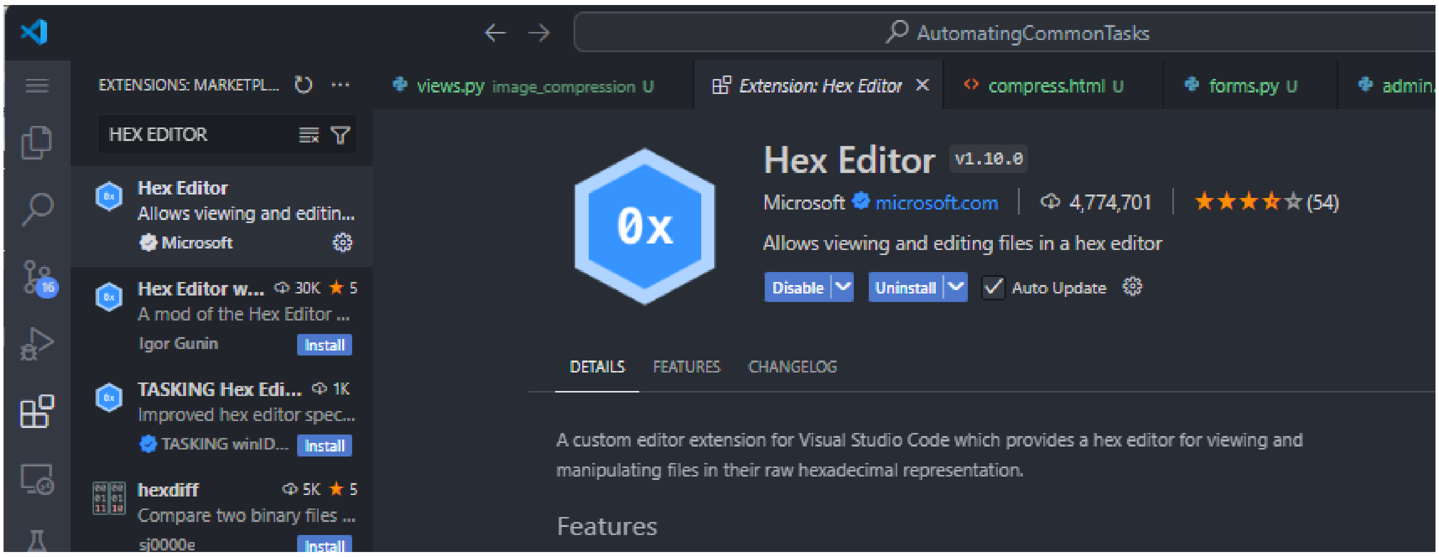
TO:



15. When we use io.BytesIO, we get bytes value of the image. To see what is the visual representation of these bytes, you can add the EXTENSION 'HEX EDITOR'.

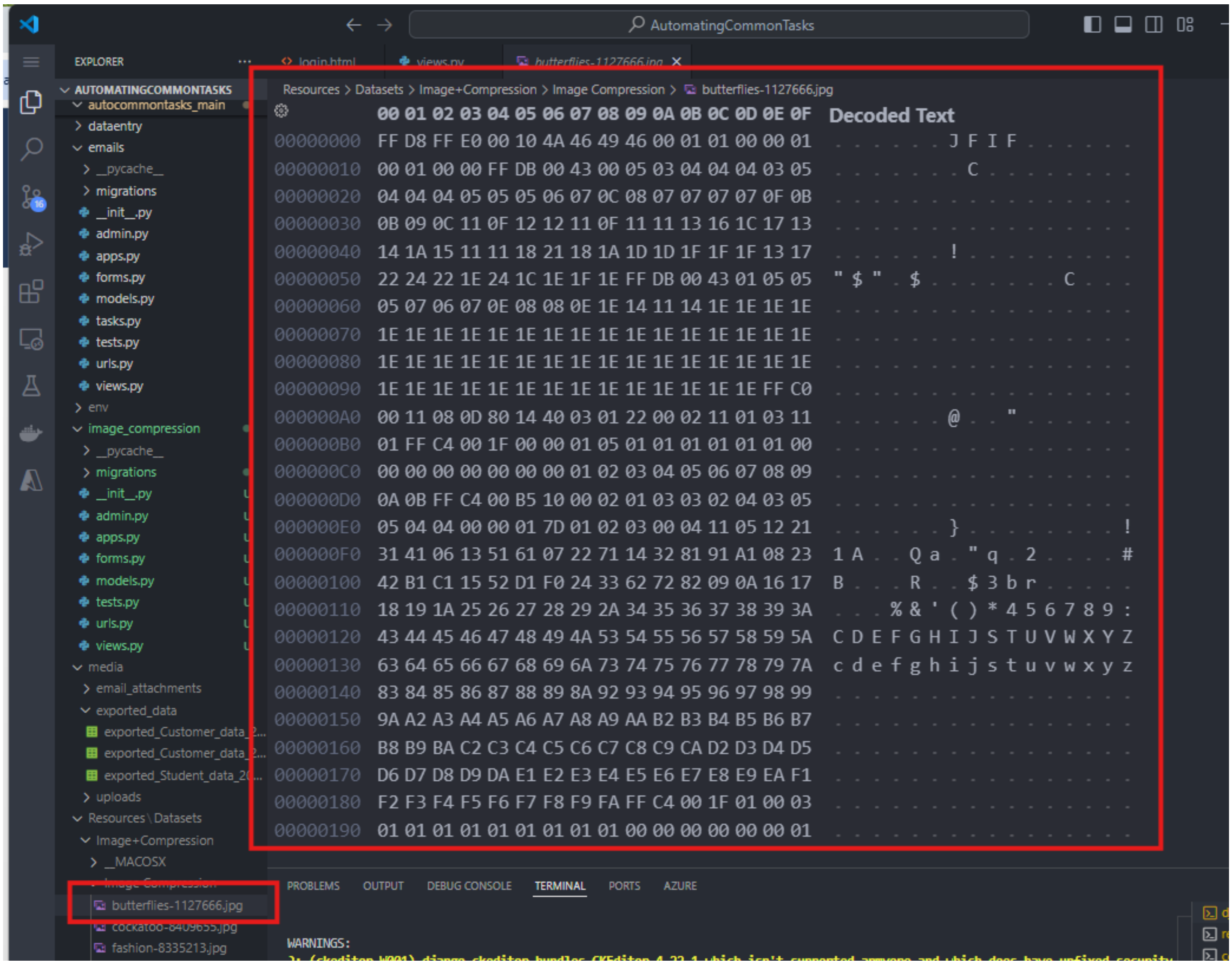
Now open an image file. On the tab of this image, right-click, select 'REOPEN EDITOR WITH', and select HEXEDITOR and you will see the HEX value of the file.


103. Image Compression Logic



A custom editor extension for Visual Studio Code which provides a hex editor for viewing and manipulating files in their raw hexadecimal representation.

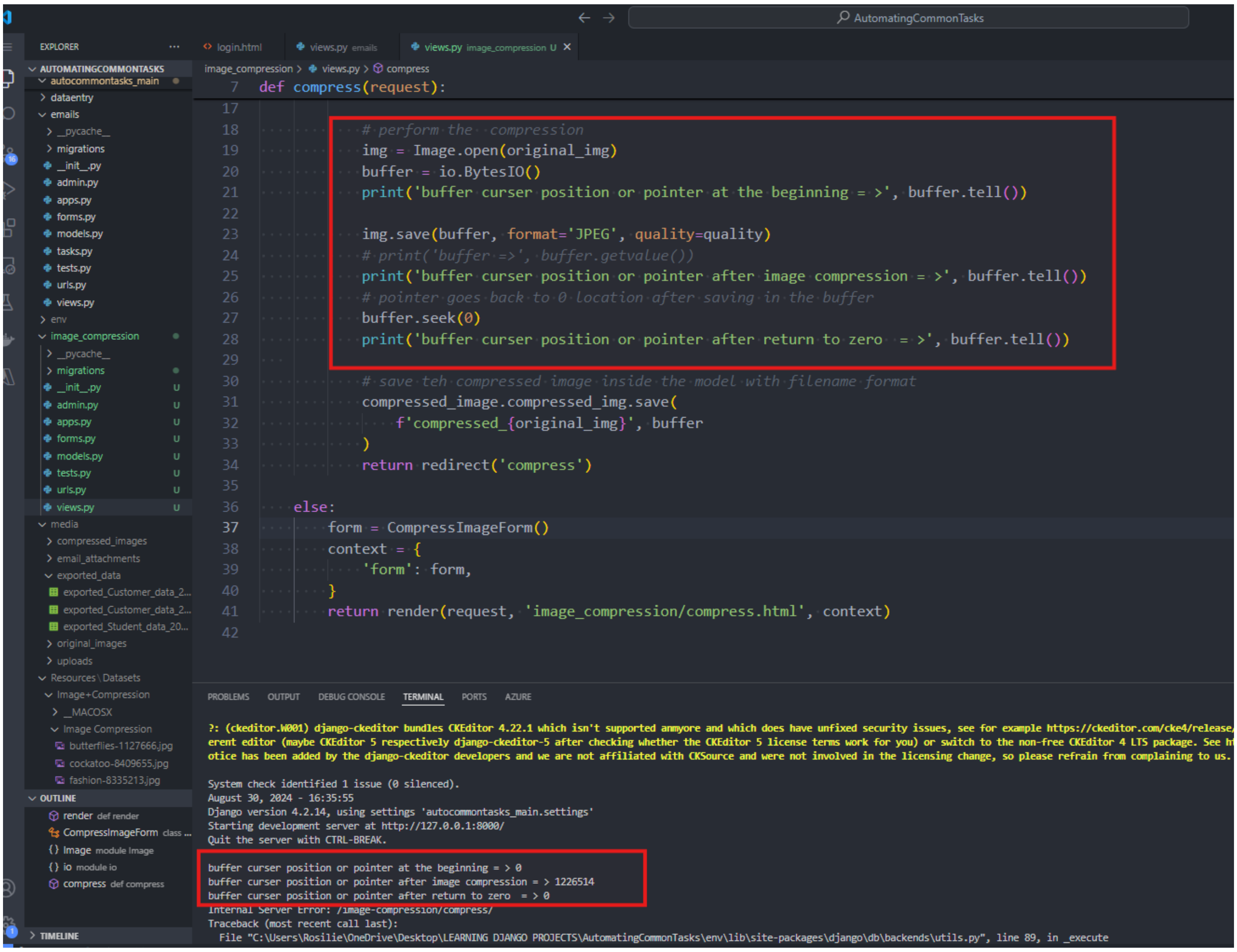
Features



```
          00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F   Decoded Text
00000000  FF D8 FF E0 00 10 4A 46 49 46 00 01 01 00 00 01   . . . . . . J F I F . . . . . .
00000010  00 01 00 00 FF DB 00 43 00 05 03 04 04 04 03 05   . . . . . . . C . . . . . . . .
00000020  04 04 04 05 05 05 06 07 0C 08 07 07 07 07 0F 0B   . . . . . . . . . . . . . . . .
00000030  0B 09 0C 11 0F 12 12 11 0F 11 11 13 16 1C 17 13   . . . . . . . . . . . . . . . .
00000040  14 1A 15 11 11 18 21 18 1A 1D 1D 1F 1F 1F 13 17   . . . . . . ! . . . . . . . . .
00000050  22 24 22 1E 24 1C 1E 1F 1E FF DB 00 43 01 05 05   " $ " . $ . . . . . . . . C . .
00000060  05 07 06 07 0E 08 08 0E 1E 14 11 14 1E 1E 1E 1E   . . . . . . . . . . . . . . . .
00000070  1E 1E 1E 1E 1E 1E 1E 1E 1E 1E 1E 1E 1E 1E 1E 1E   . . . . . . . . . . . . . . . .
00000080  1E 1E 1E 1E 1E 1E 1E 1E 1E 1E 1E 1E 1E 1E 1E 1E   . . . . . . . . . . . . . . . .
00000090  1E 1E 1E 1E 1E 1E 1E 1E 1E 1E 1E 1E 1E 1E FF C0   . . . . . . . . . . . . . . . .
000000A0  00 11 08 0D 80 14 40 03 01 22 00 02 11 01 03 11   . . . . . . @ . . " . . . . . .
000000B0  01 FF C4 00 1F 00 00 01 05 01 01 01 01 01 01 00   . . . . . . . . . . . . . . . .
000000C0  00 00 00 00 00 00 00 01 02 03 04 05 06 07 08 09   . . . . . . . . . . . . . . . .
000000D0  0A 0B FF C4 00 B5 10 00 02 01 03 03 02 04 03 05   . . . . . . . . . . . . . . . .
000000E0  05 04 04 00 00 01 7D 01 02 03 00 04 11 05 12 21   . . . . . . } . . . . . . . . !
000000F0  31 41 06 13 51 61 07 22 71 14 32 81 91 A1 08 23   1 A . . Q a . " q . 2 . . . . #
00000100  42 B1 C1 15 52 D1 F0 24 33 62 72 82 09 0A 16 17   B . . . R . . $ 3 b r . . . . .
00000110  18 19 1A 25 26 27 28 29 2A 34 35 36 37 38 39 3A   . . . % & ' ( ) * 4 5 6 7 8 9 :
00000120  43 44 45 46 47 48 49 4A 53 54 55 56 57 58 59 5A   C D E F G H I J S T U V W X Y Z
00000130  63 64 65 66 67 68 69 6A 73 74 75 76 77 78 79 7A   c d e f g h i j s t u v w x y z
00000140  83 84 85 86 87 88 89 8A 92 93 94 95 96 97 98 99   . . . . . . . . . . . . . . . .
00000150  9A A2 A3 A4 A5 A6 A7 A8 A9 AA B2 B3 B4 B5 B6 B7   . . . . . . . . . . . . . . . .
00000160  B8 B9 BA C2 C3 C4 C5 C6 C7 C8 C9 CA D2 D3 D4 D5   . . . . . . . . . . . . . . . .
00000170  D6 D7 D8 D9 DA E1 E2 E3 E4 E5 E6 E7 E8 E9 EA F1   . . . . . . . . . . . . . . . .
00000180  F2 F3 F4 F5 F6 F7 F8 F9 FA FF C4 00 1F 01 00 03   . . . . . . . . . . . . . . . .
00000190  01 01 01 01 01 01 01 01 01 00 00 00 00 00 00 01   . . . . . . . . . . . . . . . .
```

```
WARNINGS:
?: (ckeditor.W001) django-ckeditor bundles CKEditor 4.22.1 which isn't supported anymore and which does have unfixed security
```

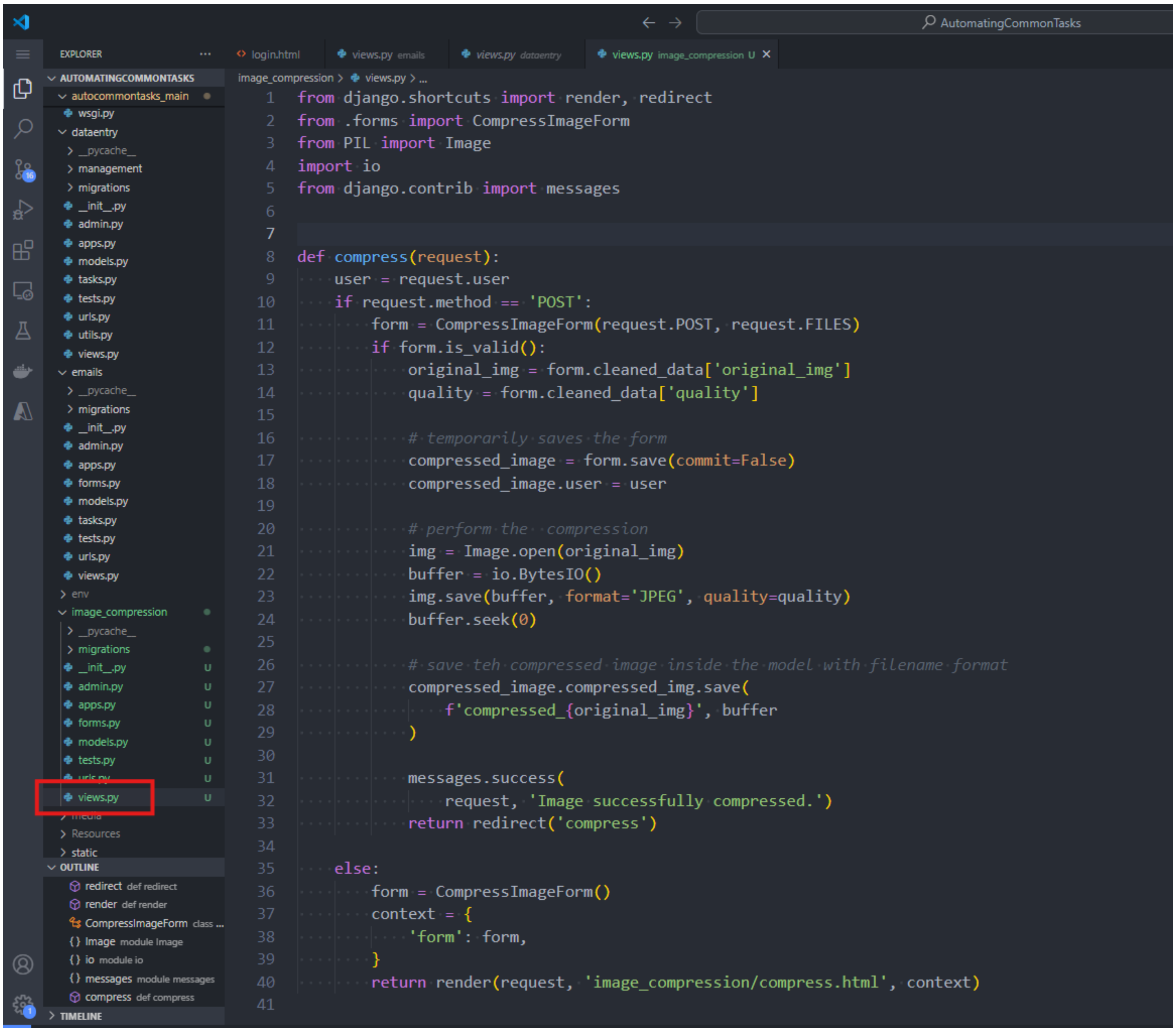
16. We use BUFFER.SEEK(0) to make sure that after we save, we set our curser position back to 0.



```python
def compress(request):
            # perform the  compression
            img = Image.open(original_img)
            buffer = io.BytesIO()
            print('buffer curser position or pointer at the beginning = >', buffer.tell())

            img.save(buffer, format='JPEG', quality=quality)
            # print('buffer =>', buffer.getvalue())
            print('buffer curser position or pointer after image compression = >', buffer.tell())
            # pointer goes back to 0 location after saving in the buffer
            buffer.seek(0)
            print('buffer curser position or pointer after return to zero  = >', buffer.tell())

            # save teh compressed image inside the model with filename format
            compressed_image.compressed_img.save(
                f'compressed_{original_img}', buffer
            )
            return redirect('compress')

    else:
        form = CompressImageForm()
        context = {
            'form': form,
        }
        return render(request, 'image_compression/compress.html', context)
```

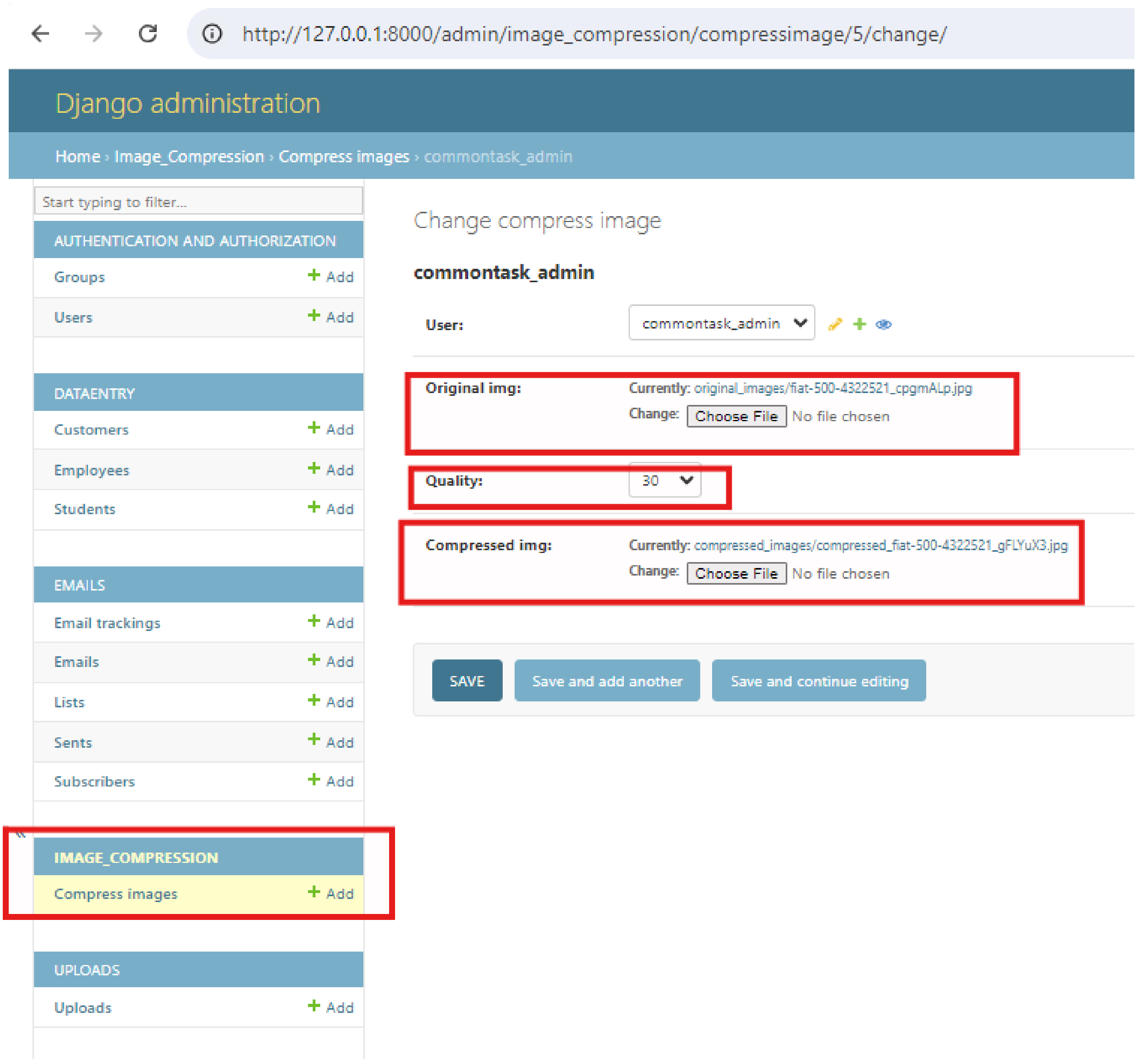
```
?: (ckeditor.W001) django-ckeditor bundles CKEditor 4.22.1 which isn't supported anmyore and which does have unfixed security issues, see for example https://ckeditor.com/cke4/release
erent editor (maybe CKEditor 5 respectively django-ckeditor-5 after checking whether the CKEditor 5 license terms work for you) or switch to the non-free CKEditor 4 LTS package. See ht
otice has been added by the django-ckeditor developers and we are not affiliated with CKSource and were not involved in the licensing change, so please refrain from complaining to us.

System check identified 1 issue (0 silenced).
August 30, 2024 - 16:35:55
Django version 4.2.14, using settings 'autocommontasks_main.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.

buffer curser position or pointer at the beginning = > 0
buffer curser position or pointer after image compression = > 1226514
buffer curser position or pointer after return to zero  = > 0
Internal Server Error: /image-compression/compress/
Traceback (most recent call last):
  File "C:\Users\Rosilie\OneDrive\Desktop\LEARNING DJANGO PROJECTS\AutomatingCommonTasks\env\lib\site-packages\django\db\backends\utils.py", line 89, in _execute
```

17. The VIEWS.PY shall be:



```python
from django.shortcuts import render, redirect
from .forms import CompressImageForm
from PIL import Image
import io
from django.contrib import messages


def compress(request):
    user = request.user
    if request.method == 'POST':
        form = CompressImageForm(request.POST, request.FILES)
        if form.is_valid():
            original_img = form.cleaned_data['original_img']
            quality = form.cleaned_data['quality']

            # temporarily saves the form
            compressed_image = form.save(commit=False)
            compressed_image.user = user

            # perform the  compression
            img = Image.open(original_img)
            buffer = io.BytesIO()
            img.save(buffer, format='JPEG', quality=quality)
            buffer.seek(0)

            # save teh compressed image inside the model with filename format
            compressed_image.compressed_img.save(
                f'compressed_{original_img}', buffer
            )

            messages.success(
                request, 'Image successfully compressed.')
            return redirect('compress')

    else:
        form = CompressImageForm()
        context = {
            'form': form,
        }
        return render(request, 'image_compression/compress.html', context)
```

18. Checking our ADMIN panel


http://127.0.0.1:8000/admin/image_compression/compressimage/5/change/

Django administration

Home › Image_Compression › Compress images › commontask_admin

Start typing to filter...

AUTHENTICATION AND AUTHORIZATION
- Groups + Add
- Users + Add

DATAENTRY
- Customers + Add
- Employees + Add
- Students + Add

EMAILS
- Email trackings + Add
- Emails + Add
- Lists + Add
- Sents + Add
- Subscribers + Add

IMAGE_COMPRESSION
- Compress images + Add

UPLOADS
- Uploads + Add

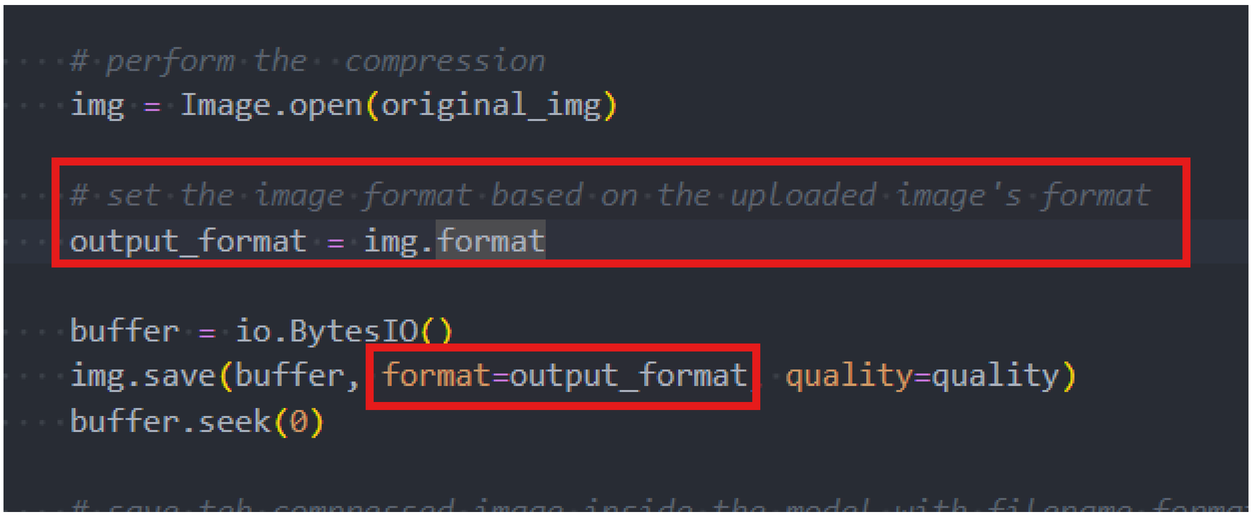
Change compress image

commontask_admin

User:	commontask_admin
Original img:	Currently: original_images/fiat-500-4322521_cpgmALp.jpg Change: Choose File No file chosen
Quality:	30
Compressed img:	Currently: compressed_images/compressed_fiat-500-4322521_gFLYuX3.jpg Change: Choose File No file chosen

SAVE | Save and add another | Save and continue editing

19. To set the image format to any format not just JPEG, we update our VIEWS.PY AS:


```
    # perform the  compression
    img = Image.open(original_img)

    # set the image format based on the uploaded image's format
    output_format = img.format

    buffer = io.BytesIO()
    img.save(buffer, format=output_format  quality=quality)
    buffer.seek(0)
```