Sure! I'll create a Django app with a Product model for perfumes, including fields for name, price, and image. The Django admin will allow you to upload images for each perfume.

Here's the breakdown:

1. **Model** (models.py): Defines the perfume product with an image field.
2. **Admin Configuration** (admin.py): Enables image uploads.
3. **Templates/View** (views.py & templates/products.html): Displays the products.

**1. Install Pillow (for ImageField)**

Django requires Pillow for image handling. Install it using:

pip install pillow

**2. Define the Model (models.py)**

from django.db import models

class Product(models.Model):

name = models.CharField(max\_length=255)

price = models.DecimalField(max\_digits=10, decimal\_places=2)

image = models.ImageField(upload\_to='product\_images/') # Uploads to MEDIA\_ROOT/product\_images/

def \_\_str\_\_(self):

return self.name

**3. Register in Admin (admin.py)**

from django.contrib import admin

from .models import Product

class ProductAdmin(admin.ModelAdmin):

list\_display = ('name', 'price', 'image\_preview')

readonly\_fields = ('image\_preview',)

def image\_preview(self, obj):

return f'<img src="{obj.image.url}" width="100px" />' if obj.image else "No Image"

image\_preview.allow\_tags = True

image\_preview.short\_description = "Image Preview"

admin.site.register(Product, ProductAdmin)

**4. Configure Media Settings (settings.py)**

import os

MEDIA\_URL = '/media/'

MEDIA\_ROOT = os.path.join(BASE\_DIR, 'media')

**5. Serve Media Files in urls.py**

from django.conf import settings

from django.conf.urls.static import static

from django.urls import path

from .views import product\_list

urlpatterns = [

path('products/', product\_list, name='product\_list'),

] + static(settings.MEDIA\_URL, document\_root=settings.MEDIA\_ROOT)

**6. Create View to Display Products (views.py)**

from django.shortcuts import render

from .models import Product

def product\_list(request):

products = Product.objects.all()

return render(request, 'products.html', {'products': products})

**7. Create a Template to Display Perfumes (templates/products.html)**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Perfume Collection</title>

<style>

.product {

border: 1px solid #ddd;

padding: 10px;

margin: 10px;

text-align: center;

}

.product img {

max-width: 150px;

height: auto;

}

</style>

</head>

<body>

<h1>Perfume Collection</h1>

<div style="display: flex; flex-wrap: wrap;">

{% for product in products %}

<div class="product">

<img src="{{ product.image.url }}" alt="{{ product.name }}">

<h2>{{ product.name }}</h2>

<p>${{ product.price }}</p>

</div>

{% endfor %}

</div>

</body>

</html>

**8. Upload Images via Django Admin**

1. Run **migrations**:
2. python manage.py makemigrations
3. python manage.py migrate
4. Create a superuser:
5. python manage.py createsuperuser
6. Start the server:
7. python manage.py runserver
8. Log in to /admin/, add perfumes with images, and visit /products/ to see them.

Now, Django will handle image uploads, and the perfume collection will be displayed dynamically! Let me know if you need tweaks. 🚀

It looks like you are encountering an error because BASE\_DIR is not defined in your manage.py file. BASE\_DIR is usually defined in the settings.py file, but it seems like you might be referencing it incorrectly in your manage.py file.

In a typical Django project, BASE\_DIR is defined in settings.py as follows:

import os

BASE\_DIR = os.path.dirname(os.path.dirname(os.path.abspath(\_\_file\_\_)))

This defines BASE\_DIR as the directory containing your Django project, and it's used to create paths to other directories like MEDIA\_ROOT.

**How to Fix the Error**

To resolve the error, follow these steps:

1. **Check if BASE\_DIR is defined correctly in settings.py**:

In your settings.py file, ensure that you have the following lines near the top:

import os

BASE\_DIR = os.path.dirname(os.path.dirname(os.path.abspath(\_\_file\_\_)))

1. **Fix the MEDIA\_ROOT definition in settings.py**:

Once BASE\_DIR is defined in settings.py, you should use it to define MEDIA\_ROOT as follows:

MEDIA\_URL = '/media/'

MEDIA\_ROOT = os.path.join(BASE\_DIR, 'media')

This will correctly set the MEDIA\_ROOT directory to a media folder inside your project directory.

1. **Ensure you are using settings.py for your configurations**:

Make sure that all configurations, including MEDIA\_ROOT and MEDIA\_URL, are in settings.py and not directly in manage.py.

**Check the File Structure:**

Make sure your file structure looks something like this:

scents\_website/

│

├── manage.py

├── scents\_website/

│ └── settings.py

└── media/ (this is where your uploaded images will be stored)

**Re-run Migrations:**

Once you've fixed the issue, you can run the migration command again:

python manage.py makemigrations

python manage.py migrate

This should fix the issue and allow Django to handle the media files properly.