

admin.py



CI Python Linter

```
1 from django.contrib import admin
2 from .models import Post
3
4 admin.site.register(Post)
5
```

Settings:

Results:

All clear, no errors found

apps.py



CI Python Linter

```
1 from django.apps import AppConfig
2
3
4 class PostsConfig(AppConfig):
5     default_auto_field = 'django.db.models.BigAutoField'
6     name = 'posts'
7
```


Settings:

Results:

All clear, no errors found



models.py



CI Python Linter

```
19 HOLIDAY_TYPE_CHOICES = (
20     ("solo", "Solo"),
21     ("couples", "Couples"),
22     ("group", "Group"),
23     ("girls", "Girls"),
24     ("guys", "Guys"),
25     ("work", "Work"),
26     ("family", "Family")
27 )
28 owner = models.ForeignKey(User, on_delete=models.CASCADE)
29 created_at = models.DateTimeField(auto_now_add=True)
30 updated_at = models.DateTimeField(auto_now=True)
31 title = models.CharField(max_length=255)
32 content = models.TextField(blank=True)
33 image = models.ImageField(
34     upload_to='images/', default='../default_post_ziaq3n', blank=True
35 )
36 continent = models.CharField(
37     max_length=2,
38     choices=CONTINENT_CHOICES,
39     default="EU"
40 )
41 holiday_type = models.CharField(
42     max_length=10,
43     choices=HOLIDAY_TYPE_CHOICES,
44     default="solo",
45     blank=True
46 )
47
48 class Meta:
49     ordering = ['-created_at']
50
51 def __str__(self):
52     return f'{self.id} {self.title}'
53
```

Settings:

Results:

All clear, no errors found

serializers.py

code
institute

CI Python Linter

```
27 def validate_image(self, value):
28     if value.size > 2 * 1024 * 1024:
29         raise serializers.ValidationError('Image must be smaller than 2MB')
30     if value.image.height > 4096:
31         raise serializers.ValidationError(
32             'Image height must be less than 4096px'
33         )
34     if value.image.width > 4096:
35         raise serializers.ValidationError(
36             'Image width must be less than 4096px'
37         )
38     return value
39
40 def get_is_owner(self, obj):
41     request = self.context['request']
42     return request.user == obj.owner
43
44 def get_like_id(self, obj):
45     user = self.context['request'].user
46     if user.is_authenticated:
47         like = Like.objects.filter(
48             owner=user, post=obj
49         ).first()
50         return like.id if like else None
51     return None
52
53 class Meta:
54     model = Post
55     fields = [
56         'id', 'owner', 'created_at', 'updated_at', 'title',
57         'content', 'image', 'continent', 'is_owner', 'profile_id',
58         'profile_image', 'like_id', 'comments_count', 'likes_count',
59         'holiday_type', 'continent_display', 'holiday_type_display'
60     ]
61
```

Settings:

☒ ☐ ☐

Results:

All clear, no errors found

urls.py

code
institute

CI Python Linter

```
1 from django.urls import path
2 from posts import views
3
4 urlpatterns = [
5     path('posts/', views.PostList.as_view()),
6     path('posts/<int:pk>/', views.PostDetail.as_view()),
7     path('get_continents/', views.GetContinentsView.as_view()),
8     path('get_holiday_types/', views.GetHolidayTypesView.as_view()),
9 ]
10
```

Settings:

☒ ☐ ☐

Results:

All clear, no errors found

views.py

code
institute

CI Python Linter

```
41     'comments__created_at'
42 ]
43
44 def perform_create(self, serializer):
45     print("Received data:", self.request.data)
46     serializer.save(owner=self.request.user)
47
48 class PostDetail(generics.RetrieveUpdateDestroyAPIView):
49     """Retrieve a specific post or edit or delete it if the post owner"""
50     permission_classes = [IsOwnerOrReadOnly]
51     serializer_class = PostSerializer
52     queryset = Post.objects.annotate(
53         comments_count=Count('comment', distinct=True),
54         likes_count=Count('likes', distinct=True)
55     ).order_by('-created_at')
56
57 class GetContinentsView(View):
58     """Retrieves the list of continents set in the Post model"""
59
60     def get(self, request, *args, **kwargs):
61         choices = [{'value': value, 'display': display}
62                     for value, display in Post.CONTINENT_CHOICES]
63         return JsonResponse(choices, safe=False)
64
65 class GetHolidayTypesView(View):
66     """Retrieves the list of holiday types set in the Post model"""
67
68     def get(self, request, *args, **kwargs):
69         choices = [{'value': value, 'display': display}
70                     for value, display in Post.HOLIDAY_TYPE_CHOICES]
71         return JsonResponse(choices, safe=False)
72
73
74
75
```

Settings:

☒ ☐ ☐

Results:

All clear, no errors found