

admin.py



CI Python Linter

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

Settings:

 ☒ 

Results:

All clear, no errors found

apps.py



CI Python Linter

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


Settings:

 ☒ 

Results:

All clear, no errors found


models.py



CI Python Linter

```
12 TRAVEL_EXPERIENCE_CHOICES = (
13     ("newbie", "Newbie traveller"),
14     ("intermediate", "Intermediate traveller"),
15     ("expert", "Expert traveller")
16 )
17 owner = models.OneToOneField(User, on_delete=models.CASCADE)
18 created_at = models.DateTimeField(auto_now_add=True)
19 updated_at = models.DateTimeField(auto_now=True)
20 name = models.CharField(max_length=255, blank=True)
21 content = models.TextField(blank=True)
22 image = models.ImageField(
23     upload_to='images/', default='../default_profile_xecywf'
24 )
25 location = CountryField(blank=True)
26 favourite_country = CountryField(blank=True)
27 travel_experience = models.CharField(
28     max_length=20,
29     choices=TRAVEL_EXPERIENCE_CHOICES,
30     default="newbie"
31 )
32
33 class Meta:
34     ordering = ['-created_at']
35
36 def __str__(self):
37     return f"{self.owner}'s profile"
38
39
40 def create_profile(sender, instance, created, **kwargs):
41     if created:
42         Profile.objects.create(owner=instance)
43
44
45 post_save.connect(create_profile, sender=User)
46 |
```

Settings:

 ☒ 

Results:

All clear, no errors found

serializers.py

code
institute

CI Python Linter

```
17 posts_count = serializers.ReadOnlyField()
18 followers_count = serializers.ReadOnlyField()
19 following_count = serializers.ReadOnlyField()
20 recent_followers_count = serializers.ReadOnlyField()
21 location = CountryField(name_only=True, allow_blank=True, required=False)
22 favourite_country = CountryField(
23     name_only=True, allow_blank=True, required=False)
24 travel_experience = serializers.ChoiceField(
25     choices=Profile.TRAVEL_EXPERIENCE_CHOICES)
26 travel_experience_display = serializers.ReadOnlyField(
27     source='get_travel_experience_display')
28
29 def get_is_owner(self, obj):
30     request = self.context['request']
31     return request.user == obj.owner
32
33 def get_following_id(self, obj):
34     user = self.context['request'].user
35     if user.is_authenticated:
36         following = Follow.objects.filter(
37             owner=user, followed=obj.owner
38         ).first()
39         return following.id if following else None
40     return None
41
42 class Meta:
43     model = Profile
44     fields = [
45         'id', 'owner', 'created_at', 'updated_at', 'name',
46         'content', 'image', 'is_owner', 'following_id',
47         'posts_count', 'followers_count', 'following_count',
48         'recent_followers_count', 'location', 'favourite_country',
49         'travel_experience', 'travel_experience_display'
50     ]
51
```

Settings:

☒ ☐ ☐

Results:

All clear, no errors found

urls.py

code
institute

CI Python Linter

```
1 from django.urls import path
2 from profiles import views
3
4 urlpatterns = [
5     path('profiles/', views.ProfileList.as_view()),
6     path('profiles/<int:pk>', views.ProfileDetail.as_view()),
7     path('get_countries/', views.GetCountriesView.as_view()),
8     path('get_travel_experience_choices/',
9         views.GetTravelExperienceChoicesView.as_view()),
10 ]
11
```

Settings:

☒ ☐ ☐

Results:

All clear, no errors found

views.py

code
institute

CI Python Linter

```
38 ]
39
40 class ProfileDetail(generics.RetrieveUpdateAPIView):
41     """Retrieve a specific profile or edit it if the profile owner"""
42     serializer_class = ProfileSerializer
43     permission_classes = [IsOwnerOrReadOnly]
44     queryset = Profile.objects.annotate(
45         posts_count=Count('owner_post', distinct=True),
46         followers_count=Count('owner_followed', distinct=True),
47         following_count=Count('owner_following', distinct=True),
48         recent_followers_count=Count(
49             'owner_followed',
50             distinct=True,
51             filter=Q(created_at__gte=datetime.now().timedelta(days=7)))
52     ).order_by('-created_at')
53
54 class GetCountriesView(APIView):
55     """Retrieves the list of countries from django-countries"""
56     def get(self, request, *args, **kwargs):
57         country_list = [country.name for country in countries]
58         return JsonResponse(country_list, safe=False)
59
60 class GetTravelExperienceChoicesView(View):
61     """
62     Retrieves the list of travel experience choices set in the
63     Profile model
64     """
65     def get(self, request, *args, **kwargs):
66         choices = [{'value': value, 'display': display}
67                     for value, display in Profile.TRAVEL_EXPERIENCE_CHOICES]
68         return JsonResponse(choices, safe=False)
69
70
71
```

Settings:

☒ ☐ ☐

Results:

All clear, no errors found