admin.py

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

Settings:

Results:

All clear, no errors found

apps.py

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

Settings:

Results:

All clear, no errors found

models.py

```
1   from django.db import models
2   from django.contrib.auth.models import User
3   from posts.models import Post
4
5
6   class Like(models.Model):
7       """
8       Like model, related to User and Post
9       Uses 'unique_together' to prevent duplicate likes by the same user
10      """
11      owner = models.ForeignKey(User, on_delete=models.CASCADE)
12      post = models.ForeignKey(
13          Post, on_delete=models.CASCADE, related_name='likes'
14      )
15      created_at = models.DateTimeField(auto_now_add=True)
16
17      class Meta:
18          ordering = ['-created_at']
19          unique_together = ['owner', 'post']
20
21      def __str__(self):
22          return f'{self.owner} likes {self.post}'
23
```

Settings:

Results:

All clear, no errors found

serializers.py

```python
from django.db import IntegrityError
from rest_framework import serializers
from .models import Like


class LikeSerializer(serializers.ModelSerializer):
    """
    Serializer for Like model
    Create method ensures no duplicate likes
    """
    owner = serializers.ReadOnlyField(source='owner.username')

    class Meta:
        model = Like
        fields = [
            'id', 'created_at', 'owner', 'post'
        ]

    def create(self, validated_data):
        try:
            return super().create(validated_data)
        except IntegrityError:
            raise serializers.ValidationError({'detail': 'possible duplicate'})
```

Settings:

Results:

All clear, no errors found

urls.py

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

urlpatterns = [
    path('likes/', views.LikeList.as_view()),
    path('likes/<int:pk>/', views.LikeDetail.as_view()),
]
```

Settings:

Results:

All clear, no errors found

views.py

```python
from rest_framework import generics, permissions
from drf_postcards.permissions import IsOwnerOrReadOnly
from .models import Like
from .serializers import LikeSerializer


class LikeList(generics.ListCreateAPIView):
    """List all likes or like a post if logged in"""
    permission_classes = [permissions.IsAuthenticatedOrReadOnly]
    serializer_class = LikeSerializer
    queryset = Like.objects.all()

    def perform_create(self, serializer):
        serializer.save(owner=self.request.user)


class LikeDetail(generics.RetrieveDestroyAPIView):
    """Retrieve a like or unlike a post"""
    permission_classes = [IsOwnerOrReadOnly]
    serializer_class = LikeSerializer
    queryset = Like.objects.all()
```

Settings:

Results:

All clear, no errors found