





Views.py



CI Python Linter

```
285 obj = get_object_or_404(TrafficRule, id=id)
286
287 if request.method == "POST":
288     # Delete the object
289     obj.delete()
290     # After deleting, redirect to the home page
291     return redirect('traffic_rules')
292     context["PAGENUMBER"] = PAGENUMBER
293     return render(request, "delete_trafficrule_view.html", context)
294 else:
295     return redirect('login')
296
297
298 # update view for RoadSign
299 def update_trafficRule_view(request, id):
300     if not isinstance(request.user, AnonymousUser) and request.user.is_staff:
301
302         context = {}
303
304         # Fetch the object related to the passed id
305         obj = get_object_or_404(TrafficRule, id=id)
306
307         # Pass the object as an instance in the form
308         form = TrafficRuleForm(request.POST or None, instance=obj)
309         # Save the data from the form and redirect to detail_view
310         if form.is_valid():
311             form.save()
312             return redirect('traffic_rules')
313
314         # Add form dictionary to context
315         context["form"] = form
316         context["PAGENUMBER"] = PAGENUMBER
317         return render(request, "update_trafficrule_view.html", context)
318     else:
319         return redirect('login')
320
```

Settings:

Results:

All clear, no errors found

models.py



CI Python Linter

```
36 ~ def __str__(self):
37 ~     return self.question.question_text
38 ~
39 ~
40 ~ class TrafficRule(models.Model):
41 ~     title = models.CharField(max_length=400, validators=[
42 ~         MinLengthValidator(5), MaxLengthValidator(400)])
43 ~     sub_title = models.CharField(max_length=400, validators=[
44 ~         MinLengthValidator(5),
45 ~         MaxLengthValidator(400)])
46 ~     sub_text = models.TextField(max_length=500, validators=[
47 ~         MinLengthValidator(5),
48 ~         MaxLengthValidator(500)])
49 ~     image_link = models.CharField(max_length=500, validators=[
50 ~         MinLengthValidator(5),
51 ~         MaxLengthValidator(500)])
52 ~
53 ~ def __str__(self):
54 ~     return self.title
55 ~
56 ~
57 ~ class RoadSign(models.Model):
58 ~     title = models.CharField(max_length=400, validators=[
59 ~         MinLengthValidator(5),
60 ~         MaxLengthValidator(400)], blank=False)
61 ~     description = models.TextField(max_length=800, validators=[
62 ~         MinLengthValidator(10),
63 ~         MaxLengthValidator(800)], blank=False)
64 ~     image_link = models.CharField(max_length=500, validators=[
65 ~         MinLengthValidator(10),
66 ~         MaxLengthValidator(500)],
67 ~         blank=False)
68 ~
69 ~ def __str__(self):
70 ~     return self.title
71 ~
```

Settings:



Results:

All clear, no errors found

urls.py



CI Python Linter

```
1 from django.urls import path, include
2 from . import views
3
4
5 urlpatterns = [
6     path('traffic_rule_question_detail/<int:question_id>/',
7         views.traffic_rule_question_detail,
8         name='traffic_rule_question_detail'),
9     path("traffic_rules", views.traffic_rules, name="traffic_rules"),
10    path("", views.index, name="home"),
11    path('start_quiz', views.start_quiz, name='start_quiz'),
12    path('continue_quiz', views.continue_quiz, name='continue_quiz'),
13    path('next_question/<int:current_question_id>/',
14        views.next_question, name='next_question'),
15    path('previous_question/<int:current_question_id>/',
16        views.previous_question, name='previous_question'),
17    path('road_signs',
18        views.road_signs_page, name='road_signs'),
19    path('road_signs/create',
20        views.create_roadSign_view, name='road_signs/create'),
21    path('<int:id>/delete/',
22        views.delete_roadSign_view,
23        name='roadsign_delete'),
24    path('update/<int:id>/',
25        views.update_roadSign_view,
26        name='update_roadSign_view'),
27    path('traffic_rule/create',
28        views.create_trafficrule_view, name='traffic_rule/create'),
29    path('delete/<int:id>/', views.delete_trafficrule_view,
30        name='trafficrule_delete'),
31    path('update/trafficrule/<int:id>/', views.update_trafficRule_view,
32        name='update_trafficRule_view'),
33 ]
34
```

Settings:



Results:

All clear, no errors found

forms.py

CI Python Linter



```
26 # specify fields to be used
27 fields = [
28     "title",
29     "description",
30     "image_link",
31 ]
32
33
34 class TrafficRuleForm(forms.ModelForm):
35     title = forms.CharField(widget=forms.TextInput(attrs={
36         'placeholder': 'Title',
37         'style': 'width: 300px;',
38         'class': 'form-control'}))
39     sub_title = forms.CharField(widget=forms.TextInput(attrs={
40         'placeholder': 'Subtitle',
41         'style': 'width: 300px;',
42         'class': 'form-control'}))
43     sub_text = forms.CharField(widget=forms.Textarea(attrs={
44         'placeholder': 'Subtext',
45         'style': 'width: 300px;',
46         'class': 'form-control'}))
47     image_link = forms.URLField(widget=forms.URLInput(attrs={
48         'placeholder': 'Image link',
49         'style': 'width: 300px;',
50         'class': 'form-control'}))
51
52 class Meta:
53     model = TrafficRule
54
55     fields = [
56         "title",
57         "sub_title",
58         "sub_text",
59         "image_link"
60     ]
61
```

Settings:



Results:

All clear, no errors found

admin.py



CI Python Linter

```
1 from django.contrib import admin
2
3 from myapp.models import RoadSign, TrafficRule, TrafficRuleAnswer
4 from myapp.models import TrafficRuleChoice, TrafficRuleQuestion
5 # Register your models here.
6
7
8 @admin.register(RoadSign)
9 class RoadSignAdmin(admin.ModelAdmin):
10     list_display = ['title', 'description', 'image_link']
11
12
13 @admin.register(TrafficRule)
14 class TrafficRuleAdmin(admin.ModelAdmin):
15     list_display = ['title', 'sub_title', 'sub_text', 'image_link']
16
17
18 @admin.register(TrafficRuleQuestion)
19 class TrafficRuleQuestion(admin.ModelAdmin):
20     list_display = ['question_text', 'pub_date', 'is_saved', 'link']
21
22
23 @admin.register(TrafficRuleChoice)
24 class TrafficRuleChoice(admin.ModelAdmin):
25     list_display = ['question', 'choice_text', 'is_correct']
26
27
28 @admin.register(TrafficRuleAnswer)
29 class TrafficRuleAnswerAdmin(admin.ModelAdmin):
30     list_display = ['question', 'user', 'is_answered']
31
```

Settings:



Results:

All clear, no errors found

test_forms.py



CI Python Linter

```
1 from django.test import SimpleTestCase
2 from myapp.forms import RoadSignForm, TrafficRuleForm
3
4
5 class TestForms(SimpleTestCase):
6     def test_roadsign_form_valid_data(self):
7         form = RoadSignForm(data={
8             "title": "roadsign 1",
9             "description": "roadsign 1 description",
10            "image_link": "firstimagelink.png"
11        })
12
13        self.assertTrue(form.is_valid())
14
15     def test_roadsign_form_no_data(self):
16         form = RoadSignForm(data={})
17
18        self.assertFalse(form.is_valid())
19        self.assertEqual(len(form.errors), 3)
20
21     def test_trafficrule_form_valid_data(self):
22         form = TrafficRuleForm(data={
23             "title": "traffic rule 1",
24             "sub_title": "This is the first traffic rule description",
25             "sub_text": "This is the first traffic rule sub_text",
26             "image_link": "firstimagelinkfortrafficrule.png"
27        })
28        self.assertTrue(form.is_valid())
29
30     def test_trafficrule_form_no_data(self):
31         form = RoadSignForm(data={})
32
33        self.assertFalse(form.is_valid())
34        self.assertEqual(len(form.errors), 3)
35
```

Settings:



Results:

All clear, no errors found

test_models.py



CI Python Linter

```
28 - def test_road_sign_deletion(self):
29     w = self.create_roadsign()
30     initial_count = RoadSign.objects.count()
31     w.delete()
32     self.assertEqual(RoadSign.objects.count(), initial_count - 1)
33
34     def create_trafficrule(self, title="The first traffic rule title",
35                           sub_title="The first traffic rule sub title",
36                           sub_text="The first traffic rule sub text",
37                           image_link="The first traffic rule image link"):
38         return TrafficRule.objects.create(title=title,
39                                           sub_title=sub_title,
40                                           sub_text=sub_text,
41                                           image_link=image_link)
42
43 - def test_trafficrule_creation(self):
44     w = self.create_trafficrule()
45     self.assertEqual(w.title, "The first traffic rule title")
46
47 - def test_traffic_rule_deletion(self):
48     w = self.create_trafficrule()
49     initial_count = TrafficRule.objects.count()
50     w.delete()
51     self.assertEqual(TrafficRule.objects.count(), initial_count - 1)
52
53 - def test_traffic_rule_update(self):
54     w = self.create_trafficrule()
55     updated_title = "Updated traffic rule title"
56
57     w.title = updated_title
58     w.save()
59
60     updated_traffic_rule = TrafficRule.objects.get(pk=w.pk)
61
62     self.assertEqual(updated_traffic_rule.title, updated_title)
63
```

Settings:



Results:

All clear, no errors found

test_urls.py

CI Python Linter



```
33 - def test_traffic_rule_update_url_resolves(self):
34     url = reverse("update_trafficRule_view", args=[1])
35     self.assertEqual(resolve(url).func, update_trafficRule_view)
36
37 - def test_continue_quiz_url_resolves(self):
38     url = reverse("continue_quiz")
39     self.assertEqual(resolve(url).func, continue_quiz)
40
41 - def test_road_signs_page_url_resolves(self):
42     url = reverse("road_signs")
43     self.assertEqual(resolve(url).func, road_signs_page)
44
45 - def test_road_signs_create_url_resolves(self):
46     url = reverse("road_signs/create")
47     self.assertEqual(resolve(url).func, create_roadSign_view)
48
49 - def test_traffic_rule_question_detail_url_resolves(self):
50     url = reverse("traffic_rule_question_detail", args=[1])
51     self.assertEqual(resolve(url).func, traffic_rule_question_detail)
52
53 - def test_next_question_url_resolves(self):
54     url = reverse("next_question", args=[1])
55     self.assertEqual(resolve(url).func, next_question)
56
57 - def test_previous_question_url_resolves(self):
58     url = reverse("previous_question", args=[1])
59     self.assertEqual(resolve(url).func, previous_question)
60
61 - def test_delete_road_sign_url_resolves(self):
62     url = reverse("roadsign_delete", args=[1])
63     self.assertEqual(resolve(url).func, delete_roadSign_view)
64
65 - def test_update_road_sign_url_resolves(self):
66     url = reverse("update_roadSign_view", args=[1])
67     self.assertEqual(resolve(url).func, update_roadSign_view)
68
```

Settings:



Results:

All clear, no errors found

test_views.py

CI Python Linter



```
11     }
12     self.setup_user()
13
14 - def setup_user(self):
15     User.objects.create_user(**self.credentials)
16
17 - def test_traffic_rules_view_deny_anonymous(self):
18     response = self.client.get('/traffic_rules', follow=True)
19     self.assertRedirects(response, '/login/')
20
21 - def test_traffic_rules_view_load(self):
22     # Log in
23     response_login = self.client.post(
24         '/login/', self.credentials, follow=True)
25
26     self.assertTrue(response_login.context['user'].is_active)
27
28     response_traffic_rules = self.client.get('/traffic_rules', follow=True)
29
30     self.assertEqual(response_traffic_rules.status_code, 200)
31
32 - def test_road_signs_page_view_deny_anonymous(self):
33     response = self.client.get('/road_signs', follow=True)
34     self.assertRedirects(response, '/login/')
35
36 - def test_road_signs_view_load(self):
37     # Log in
38     response_login = self.client.post(
39         '/login/', self.credentials, follow=True)
40
41     self.assertTrue(response_login.context['user'].is_active)
42
43     response_road_signs = self.client.get('/road_signs', follow=True)
44
45     self.assertEqual(response_road_signs.status_code, 200)
46
```

Settings:



Results:

All clear, no errors found