

```

1  """ core/api_views.py """
2
3  from rest_framework.generics import ListAPIView,
4  • RetrieveUpdateDestroyAPIView
5  from django_filters.rest_framework import DjangoFilterBackend
6  from rest_framework.filters import SearchFilter
7  from rest_framework.pagination import LimitOffsetPagination
8  from rest_framework.permissions import IsAuthenticatedOrReadOnly
9  from rest_framework.authentication import SessionAuthentication,
10 • BasicAuthentication, TokenAuthentication
11 from rest_framework import authentication
12 from rest_framework import exceptions
13
14 from core.serializers import RestaurantSerializer
15 from core.models import Restaurant
16
17 class RestaurantsPagination(LimitOffsetPagination):
18     default_limit = 10
19     max_limit = 100
20
21 class RestaurantList(ListAPIView):
22     queryset = Restaurant.objects.all()
23     serializer_class = RestaurantSerializer
24     filter_backends = (DjangoFilterBackend, SearchFilter)
25     filter_fields = ('category',)
26     search_fields = ('name', 'description', 'category', 'address')
27     pagination_class = RestaurantsPagination
28
29     def get_queryset(self):
30         queryset = Restaurant.objects.all()
31         return queryset
32
33 class RestaurantRetrieveUpdateDestroy(RetrieveUpdateDestroyAPIView):
34     authentication_classes = [SessionAuthentication,
35     • BasicAuthentication, TokenAuthentication]
36     permission_classes = [IsAuthenticatedOrReadOnly]
37     queryset = Restaurant.objects.all()
38     lookup_field = 'id'
39     serializer_class = RestaurantSerializer
40
41     def update(self, request, *args, **kwargs):
42         response = super().update(request, *args, **kwargs)
43         if response.status_code == 200:
44             from django.core.cache import cache
45             restaurant = response.data
46             cache.set('restaurant_data_{}'.format(restaurant['id']),
47             • restaurant)
48             return response

```

```

47  """ core/models.py """
48
49  from django.db import models
50
51  class Restaurant(models.Model):
52      id = models.AutoField(primary_key=True)
53      name = models.CharField(max_length=200)
54      description = models.TextField()
55      category = models.CharField(max_length=50)
56      address = models.TextField()
57      phone = models.CharField(max_length=30)
58      website = models.URLField(max_length=200)
59      created_at = models.DateTimeField('Created at',
    •      auto_now_add=True)
60
61  """ core/serializers.py """
62
63  from rest_framework import serializers
64
65  from core.models import Restaurant
66
67  class RestaurantSerializer(serializers.ModelSerializer):
68      class Meta:
69          model = Restaurant
70          fields = (
71              'id', 'name', 'description', 'category', 'address',
    •      'phone', 'website'
72          )
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94

```

```

95  """ core/tests.py """
96
97  from rest_framework.test import APITestCase
98  from core.models import Restaurant
99
100 from django.contrib.auth.models import User
101
102 class AuthenticationTestCase(APITestCase):
103     def setUp(self):
104         user = User.objects.create(username='admin')
105         user.set_password('123456')
106         user.save()
107         self.login_url = '/api/v1/rest-auth/login/'
108
109     def test_login_sucess(self):
110         response = self.client.post(self.login_url, {'username':
111     • 'admin', 'password': '123456'})
112         self.assertEqual(response.status_code, 200)
113
114     def test_login_fail(self):
115         response = self.client.post(self.login_url, {'username':
116     • 'other', 'password': 'abcdef'})
117         self.assertEqual(response.status_code, 400)
118
119     def test_logout(self):
120         self.client.login(username='admin', password='123456')
121         response = self.client.post('/api/v1/rest-auth/logout/')
122         self.assertEqual(response.status_code, 200)
123
124 class RestaurantListTestCase(APITestCase):
125     def test_list_restaurants(self):
126         restaurants_count = Restaurant.objects.count()
127         response = self.client.get('/api/v1/restaurants/')
128         self.assertIsNone(response.data['next'])
129         self.assertIsNone(response.data['previous'])
130         self.assertEqual(response.data['count'], restaurants_count)
131         self.assertEqual(len(response.data['results']),
132     • restaurants_count)
133
134 class RestaurantUpdateTestCase(APITestCase):
135     def setUp(self):
136         restaurant = Restaurant(name='Restaurant 1',
137     description='hipster, vegan',
138     address='street 11',
139     phone='+49 30 0000001')
140         restaurant.save()
141
142     def test_update_restaurants(self):
143         user = User(username='admin', password='123456')
144         user.save()

```

```
142 self.client.force_authenticate(user=user)
143 restaurant = Restaurant.objects.first()
144 response = self.client.patch(
145     '/api/v1/restaurants/{}/'.format(restaurant.id),
146     {
147         'name': 'New Restaurant',
148         'description': 'Awesome Restaurant',
149         'address': 'street 1000',
150     },
151     format='json',
152 )
153 updated = Restaurant.objects.get(id=restaurant.id)
154 self.assertEqual(updated.name, 'New Restaurant')
155
156 def test_update_restaurants_not_authenticated(self):
157     restaurant = Restaurant.objects.first()
158     response = self.client.patch(
159         '/api/v1/restaurants/{}/'.format(restaurant.id),
160         {
161             'name': 'New Restaurant',
162             'description': 'Awesome Restaurant',
163             'address': 'street 1000',
164         },
165         format='json',
166     )
167
168     self.assertEqual(response.status_code, 403)
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
```

```

192 """ restolist/urls.py """
193
194 from django.contrib import admin
195 from django.urls import include, path
196
197 import core.api_views
198
199 urlpatterns = [
200     path('api/v1/restaurants/',
201         core.api_views.RestaurantList.as_view()),
202     path('api/v1/restaurants/<int:id>',
203         core.api_views.RestaurantRetrieveUpdateDestroy.as_view()
204     ),
205     path('api/v1/rest-auth/', include('rest_auth.urls')),
206     path('admin/', admin.site.urls),
207 ]
208
209 """ restolist/settings.py """
210
211 import os
212
213 BASE_DIR =
214     • os.path.dirname(os.path.dirname(os.path.abspath(__file__)))
215
216 SECRET_KEY = 'v6b_fsjws8pa*z!j14c_@x8t7ax)o$xgt7edj^v+4-_78j9e#a'
217
218 DEBUG = True
219
220 ALLOWED_HOSTS = []
221
222 INSTALLED_APPS = [
223     'django.contrib.admin',
224     'django.contrib.auth',
225     'django.contrib.contenttypes',
226     'django.contrib.sessions',
227     'django.contrib.messages',
228     'django.contrib.staticfiles',
229     'django.contrib.sites',
230     'rest_framework',
231     'rest_framework.authtoken',
232     'rest_auth',
233     'allauth',
234     'allauth.account',
235     'rest_auth.registration',
236     'django_filters',
237     'corsheaders',
238     'core'
239

```

```

239     ]
240
241     MIDDLEWARE = [
242         'corsheaders.middleware.CorsMiddleware',
243         'django.middleware.common.CommonMiddleware',
244         'django.middleware.security.SecurityMiddleware',
245         'django.contrib.sessions.middleware.SessionMiddleware',
246         'django.middleware.common.CommonMiddleware',
247         'django.middleware.csrf.CsrfViewMiddleware',
248         'django.contrib.auth.middleware.AuthenticationMiddleware',
249         'django.contrib.auth.middleware.RemoteUserMiddleware',
250         'django.contrib.messages.middleware.MessageMiddleware',
251         'django.middleware.clickjacking.XFrameOptionsMiddleware',
252     ]
253
254     CORS_ORIGIN_WHITELIST = (
255         'http://localhost:3000',
256     )
257
258     AUTHENTICATION_BACKENDS = (
259         'django.contrib.auth.backends.RemoteUserBackend',
260         'allauth.account.auth_backends.AuthenticationBackend'
261     )
262
263     ROOT_URLCONF = 'restolist.urls'
264
265     TEMPLATES = [
266         {
267             'BACKEND':
268             • 'django.template.backends.django.DjangoTemplates',
269             'DIRS': [],
270             'APP_DIRS': True,
271             'OPTIONS': {
272                 'context_processors': [
273                     'django.template.context_processors.debug',
274                     'django.template.context_processors.request',
275                     'django.contrib.auth.context_processors.auth',
276                     'django.contrib.messages.context_processors.messages
277                     • ',
278                 ],
279             },
280         },
281     ],
282
283     WSGI_APPLICATION = 'restolist.wsgi.application'
284
285     DATABASES = {
286         'default': {
287             'ENGINE': 'django.db.backends.sqlite3',
288             'NAME': os.path.join(BASE_DIR, 'db.sqlite3'),
289         },
290     }

```

```
287         },
288     }
289
290     AUTH_PASSWORD_VALIDATORS = [
291         {
292             'NAME':
293             • 'django.contrib.auth.password_validation.UserAttributeSimila
294             • rityValidator',
295         },
296         {
297             'NAME':
298             • 'django.contrib.auth.password_validation.MinimumLengthValida
299             • tor',
300         },
301         {
302             'NAME':
303             • 'django.contrib.auth.password_validation.CommonPasswordValid
304             • ator',
305         },
306         {
307             'NAME':
308             • 'django.contrib.auth.password_validation.NumericPasswordVali
309             • dator',
310         },
311     ]
312
313     LANGUAGE_CODE = 'en-us'
314
315     TIME_ZONE = 'UTC'
316
317     USE_I18N = True
318
319     USE_L10N = True
320
321     USE_TZ = True
322
323     STATIC_URL = '/static/'
324
```