

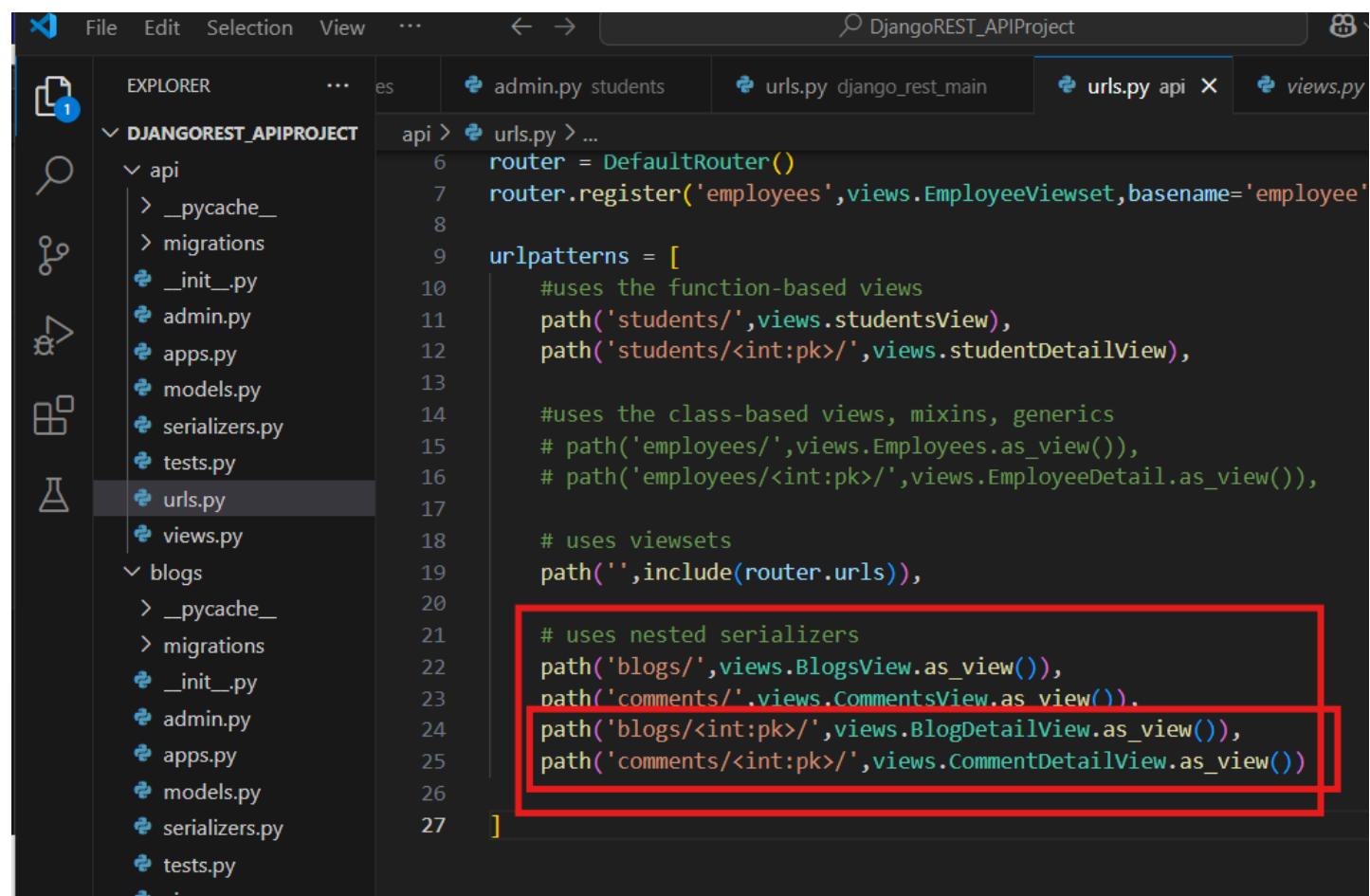
## Topic: 13. Nested Serializers for Related Models Part 2

Speaker: Personal / Notebook: API Development using Django Framework



In this post, we use the primary key to allow other CRUD operations.

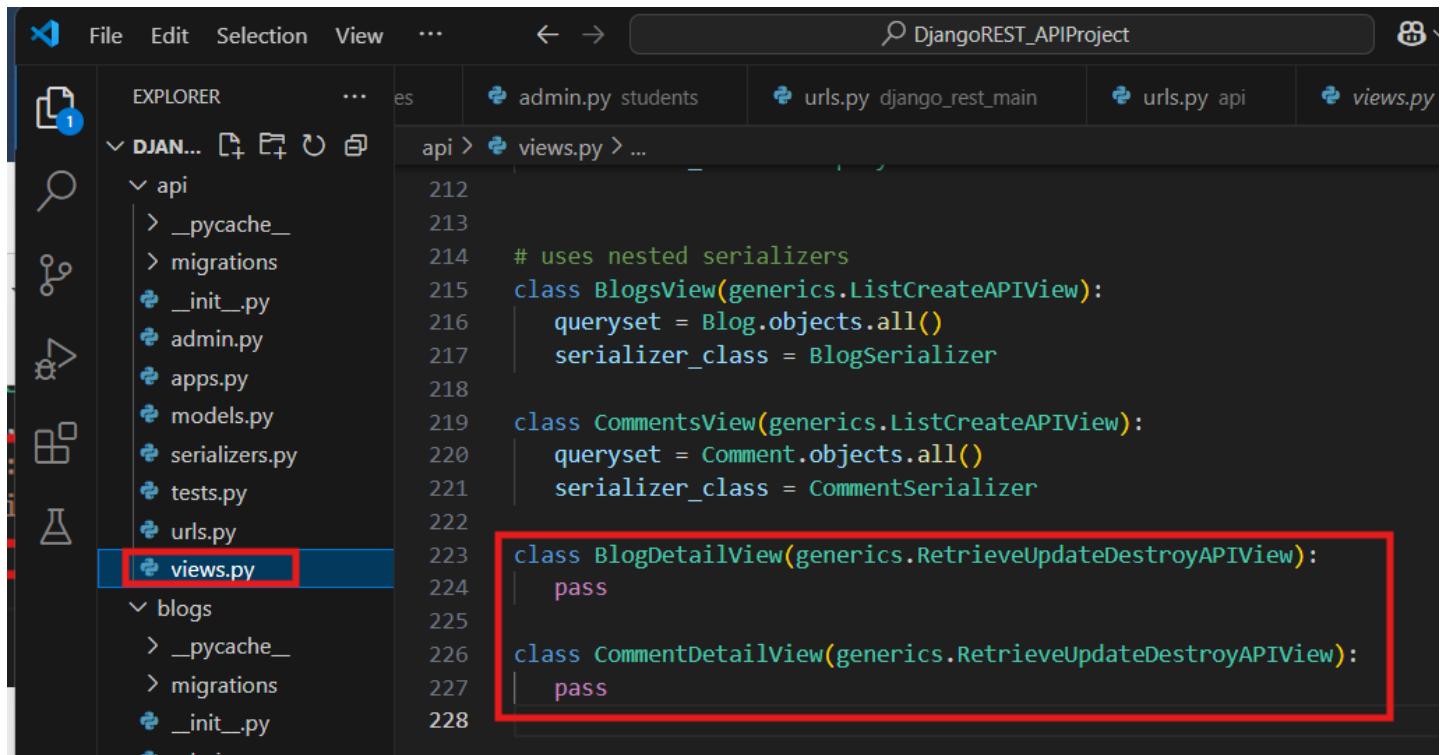
1. So we update the API\URLS.PY to include the PK-based operations.



```
File Edit Selection View ... ← → DjangoREST_APIProject
EXPLORER ... admin.py students urls.py django_rest_main urls.py api X views.py
DJANGOREST_APIPROJECT
  api
    __pycache__
    migrations
    __init__.py
    admin.py
    apps.py
    models.py
    serializers.py
    tests.py
    urls.py
    views.py
  blogs
    __pycache__
    migrations
    __init__.py
    admin.py
    apps.py
    models.py
    serializers.py
    tests.py
    views.py

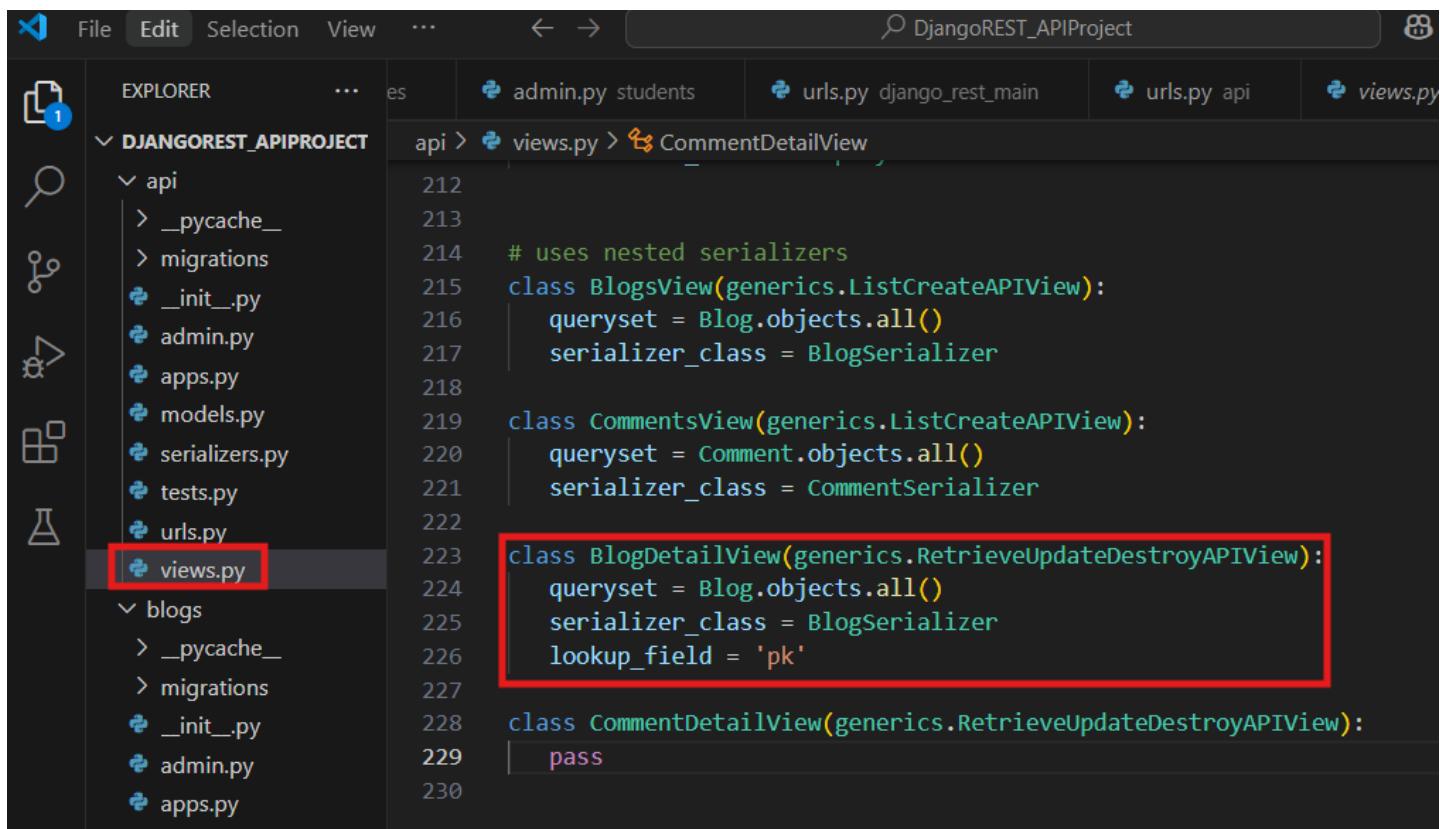
api > urls.py > ...
6   router = DefaultRouter()
7   router.register('employees',views.EmployeeViewSet, basename='employee')
8
9   urlpatterns = [
10      #uses the function-based views
11      path('students/',views.studentsView),
12      path('students/<int:pk>',views.studentDetailView),
13
14      #uses the class-based views, mixins, generics
15      # path('employees/',views.Employees.as_view()),
16      # path('employees/<int:pk>',views.EmployeeDetail.as_view()),
17
18      # uses viewsets
19      path('',include(router.urls)),
20
21      # uses nested serializers
22      path('blogs/',views.BlogsView.as_view()),
23      path('comments/.views.CommentsView.as_view()'),
24      path('blogs/<int:pk>',views.BlogDetailView.as_view()),
25      path('comments/<int:pk>',views.CommentDetailView.as_view())
26
27 ]
```

2. Update the API\VIEWS.PY



```
121
122
123
124 # uses nested serializers
125 class BlogsView(generics.ListCreateAPIView):
126     queryset = Blog.objects.all()
127     serializer_class = BlogSerializer
128
129 class CommentsView(generics.ListCreateAPIView):
130     queryset = Comment.objects.all()
131     serializer_class = CommentSerializer
132
133 class BlogDetailView(generics.RetrieveUpdateDestroyAPIView):
134     pass
135
136 class CommentDetailView(generics.RetrieveUpdateDestroyAPIView):
137     pass
```

Then update as:



```
121
122
123
124 # uses nested serializers
125 class BlogsView(generics.ListCreateAPIView):
126     queryset = Blog.objects.all()
127     serializer_class = BlogSerializer
128
129 class CommentsView(generics.ListCreateAPIView):
130     queryset = Comment.objects.all()
131     serializer_class = CommentSerializer
132
133 class BlogDetailView(generics.RetrieveUpdateDestroyAPIView):
134     queryset = Blog.objects.all()
135     serializer_class = BlogSerializer
136     lookup_field = 'pk'
137
138 class CommentDetailView(generics.RetrieveUpdateDestroyAPIView):
139     pass
```

3. Now view the specific path of blog post # 1:

127.0.0.1:8000/api/v1/blogs/1/

Django REST Framework api\_djangoadmin

Api Root / Blogs / Blog Detail

## Blog Detail

DELETE OPTIONS GET

```
GET /api/v1/blogs/1/
```

HTTP 200 OK

Allow: GET, PUT, PATCH, DELETE, HEAD, OPTIONS

Content-Type: application/json

Vary: Accept

```
{
    "id": 1,
    "comments": [
        {
            "id": 1,
            "comment": "this is comment #1",
            "blog": 1
        },
        {
            "id": 2,
            "comment": "this is comment #2",
            "blog": 1
        },
        {
            "id": 3,
            "comment": "this is comment #3",
            "blog": 1
        }
    ],
    "blog_title": "this is a blog 1",
    "blog_body": "this is a blog body 1"
}
```

Raw data HTML form

Blog title: this is a blog 1

Blog body: this is a blog body 1

PUT

Now, you can update and delete this record.

→ C 127.0.0.1:8000/api/v1/blogs/1/      

Django REST framework api\_djangoadmin

Api Root / Blogs / Blog Detail

## Blog Detail

[DELETE /api/v1/blogs/1/](#) [OPTIONS](#) [GET](#) ▾

HTTP 404 Not Found  
Allow: GET, PUT, PATCH, DELETE, HEAD, OPTIONS  
Content-Type: application/json  
Vary: Accept

```
{ "detail": "No Blog matches the given query." }
```

Raw data [HTML form](#)

Blog title

Blog body

[PUT](#)

You can also add a new post here:

Api Root / Blogs

# Blogs

[OPTIONS](#)[GET](#)[GET /api/v1/blogs/](#)

HTTP 200 OK  
Allow: GET, POST, HEAD, OPTIONS  
Content-Type: application/json  
Vary: Accept

```
[  
  {  
    "id": 2,  
    "comments": [  
      {  
        "id": 4,  
        "comment": "this is comment #1",  
        "blog": 2  
      },  
      {  
        "id": 5,  
        "comment": "this is comment #2",  
        "blog": 2  
      }  
    ],  
    "blog_title": "this is a blog 2",  
    "blog_body": "this is a blog body 2"  
  }  
]
```

[Raw data](#)[HTML form](#)

Blog title

Blog body

[POST](#)

Then you can reload your page:

127.0.0.1:8000/api/v1/blogs/

## Django REST framework

api\_djangoadmin

Api Root / Blogs

# Blogs

OPTIONS GET

GET /api/v1/blogs/

HTTP 200 OK

Allow: GET, POST, HEAD, OPTIONS

Content-Type: application/json

Vary: Accept

```
[{"id": 2, "comments": [{"id": 4, "comment": "this is comment #1", "blog": 2}, {"id": 5, "comment": "this is comment #2", "blog": 2}], "blog_title": "this is a blog 2", "blog_body": "this is a blog body 2"}, {"id": 3, "comments": [], "blog_title": "this is a blog 3", "blog_body": "This is a blog description"}]
```

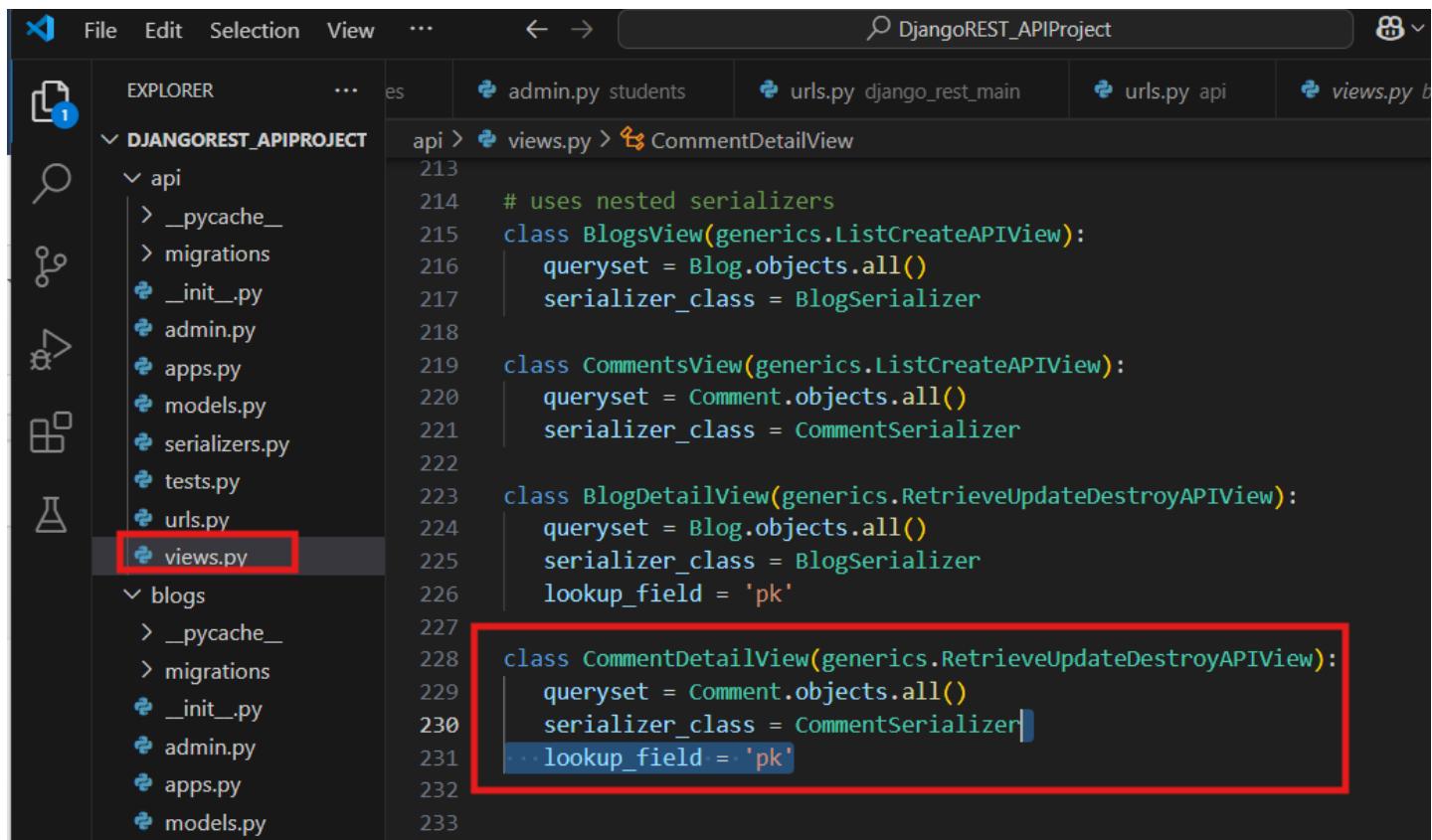
Raw data HTML form

Blog title

Blog body

POST

4. To allow CRUD operations on `Comment` model, then we update the `class CommentDetailView`:



The screenshot shows a code editor interface with the following details:

- File Bar:** File Edit Selection View ...
- Search Bar:** DjangoREST\_APIProject
- Left Sidebar:** EXPLORER, File Explorer, Search, Open, Find, Open in New Tab, Open in New Window, Open in New Editor, Open in New Editor (2), Open in New Editor (3), Open in New Editor (4), Open in New Editor (5), Open in New Editor (6), Open in New Editor (7), Open in New Editor (8), Open in New Editor (9), Open in New Editor (10), Open in New Editor (11), Open in New Editor (12), Open in New Editor (13), Open in New Editor (14), Open in New Editor (15), Open in New Editor (16), Open in New Editor (17), Open in New Editor (18), Open in New Editor (19), Open in New Editor (20), Open in New Editor (21), Open in New Editor (22), Open in New Editor (23), Open in New Editor (24), Open in New Editor (25), Open in New Editor (26), Open in New Editor (27), Open in New Editor (28), Open in New Editor (29), Open in New Editor (30), Open in New Editor (31), Open in New Editor (32), Open in New Editor (33), Open in New Editor (34), Open in New Editor (35), Open in New Editor (36), Open in New Editor (37), Open in New Editor (38), Open in New Editor (39), Open in New Editor (40), Open in New Editor (41), Open in New Editor (42), Open in New Editor (43), Open in New Editor (44), Open in New Editor (45), Open in New Editor (46), Open in New Editor (47), Open in New Editor (48), Open in New Editor (49), Open in New Editor (50), Open in New Editor (51), Open in New Editor (52), Open in New Editor (53), Open in New Editor (54), Open in New Editor (55), Open in New Editor (56), Open in New Editor (57), Open in New Editor (58), Open in New Editor (59), Open in New Editor (60), Open in New Editor (61), Open in New Editor (62), Open in New Editor (63), Open in New Editor (64), Open in New Editor (65), Open in New Editor (66), Open in New Editor (67), Open in New Editor (68), Open in New Editor (69), Open in New Editor (70), Open in New Editor (71), Open in New Editor (72), Open in New Editor (73), Open in New Editor (74), Open in New Editor (75), Open in New Editor (76), Open in New Editor (77), Open in New Editor (78), Open in New Editor (79), Open in New Editor (80), Open in New Editor (81), Open in New Editor (82), Open in New Editor (83), Open in New Editor (84), Open in New Editor (85), Open in New Editor (86), Open in New Editor (87), Open in New Editor (88), Open in New Editor (89), Open in New Editor (90), Open in New Editor (91), Open in New Editor (92), Open in New Editor (93), Open in New Editor (94), Open in New Editor (95), Open in New Editor (96), Open in New Editor (97), Open in New Editor (98), Open in New Editor (99), Open in New Editor (100), Open in New Editor (101), Open in New Editor (102), Open in New Editor (103), Open in New Editor (104), Open in New Editor (105), Open in New Editor (106), Open in New Editor (107), Open in New Editor (108), Open in New Editor (109), Open in New Editor (110), Open in New Editor (111), Open in New Editor (112), Open in New Editor (113), Open in New Editor (114), Open in New Editor (115), Open in New Editor (116), Open in New Editor (117), Open in New Editor (118), Open in New Editor (119), Open in New Editor (120), Open in New Editor (121), Open in New Editor (122), Open in New Editor (123), Open in New Editor (124), Open in New Editor (125), Open in New Editor (126), Open in New Editor (127), Open in New Editor (128), Open in New Editor (129), Open in New Editor (130), Open in New Editor (131), Open in New Editor (132), Open in New Editor (133), Open in New Editor (134), Open in New Editor (135), Open in New Editor (136), Open in New Editor (137), Open in New Editor (138), Open in New Editor (139), Open in New Editor (140), Open in New Editor (141), Open in New Editor (142), Open in New Editor (143), Open in New Editor (144), Open in New Editor (145), Open in New Editor (146), Open in New Editor (147), Open in New Editor (148), Open in New Editor (149), Open in New Editor (150), Open in New Editor (151), Open in New Editor (152), Open in New Editor (153), Open in New Editor (154), Open in New Editor (155), Open in New Editor (156), Open in New Editor (157), Open in New Editor (158), Open in New Editor (159), Open in New Editor (160), Open in New Editor (161), Open in New Editor (162), Open in New Editor (163), Open in New Editor (164), Open in New Editor (165), Open in New Editor (166), Open in New Editor (167), Open in New Editor (168), Open in New Editor (169), Open in New Editor (170), Open in New Editor (171), Open in New Editor (172), Open in New Editor (173), Open in New Editor (174), Open in New Editor (175), Open in New Editor (176), Open in New Editor (177), Open in New Editor (178), Open in New Editor (179), Open in New Editor (180), Open in New Editor (181), Open in New Editor (182), Open in New Editor (183), Open in New Editor (184), Open in New Editor (185), Open in New Editor (186), Open in New Editor (187), Open in New Editor (188), Open in New Editor (189), Open in New Editor (190), Open in New Editor (191), Open in New Editor (192), Open in New Editor (193), Open in New Editor (194), Open in New Editor (195), Open in New Editor (196), Open in New Editor (197), Open in New Editor (198), Open in New Editor (199), Open in New Editor (200), Open in New Editor (201), Open in New Editor (202), Open in New Editor (203), Open in New Editor (204), Open in New Editor (205), Open in New Editor (206), Open in New Editor (207), Open in New Editor (208), Open in New Editor (209), Open in New Editor (210), Open in New Editor (211), Open in New Editor (212), Open in New Editor (213), Open in New Editor (214), Open in New Editor (215), Open in New Editor (216), Open in New Editor (217), Open in New Editor (218), Open in New Editor (219), Open in New Editor (220), Open in New Editor (221), Open in New Editor (222), Open in New Editor (223), Open in New Editor (224), Open in New Editor (225), Open in New Editor (226), Open in New Editor (227), Open in New Editor (228), Open in New Editor (229), Open in New Editor (230), Open in New Editor (231), Open in New Editor (232), Open in New Editor (233)
- Code Editor:** The code editor shows the following Python code for the 'CommentDetailView' class:

```
213
214     # uses nested serializers
215     class BlogsView(generics.ListCreateAPIView):
216         queryset = Blog.objects.all()
217         serializer_class = BlogSerializer
218
219     class CommentsView(generics.ListCreateAPIView):
220         queryset = Comment.objects.all()
221         serializer_class = CommentSerializer
222
223     class BlogDetailView(generics.RetrieveUpdateDestroyAPIView):
224         queryset = Blog.objects.all()
225         serializer_class = BlogSerializer
226         lookup_field = 'pk'
227
228     class CommentDetailView(generics.RetrieveUpdateDestroyAPIView):
229         queryset = Comment.objects.all()
230         serializer_class = CommentSerializer
231         lookup_field = 'pk'
```

5. To view the COMMENTS path:

Api Root / Comments

# Comments

[OPTIONS](#)[GET](#) ▾

GET /api/v1/comments/

HTTP 200 OK  
Allow: GET, POST, HEAD, OPTIONS  
Content-Type: application/json  
Vary: Accept

```
[  
  {  
    "id": 4,  
    "comment": "this is comment #1",  
    "blog": 2  
  },  
  {  
    "id": 5,  
    "comment": "this is comment #2",  
    "blog": 2  
  }  
]
```

[Raw data](#)[HTML form](#)

Comment

This is a comment #1

Blog

this is a blog 3

[POST](#)

6. To delete the specific comment:

The screenshot shows a browser window displaying the Django REST framework API. The URL in the address bar is `127.0.0.1:8000/api/v1/comments/7/`. The page title is "Django REST framework" and the sub-page title is "Comment Detail".

On the right side of the page, there are three buttons: "DELETE" (red), "OPTIONS" (blue), and "GET" (blue). A red box highlights the "DELETE" button.

On the left side, there is a "Raw data" section containing the following JSON response:

```
HTTP 200 OK
Allow: GET, PUT, PATCH, DELETE, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "id": 7,
    "comment": "test comment",
    "blog": 3
}
```

Below this, there is a "HTML form" section with two fields:

- Comment:** A text input field containing "test comment".
- Blog:** A dropdown menu containing "this is a blog 3".

At the bottom right of the form is a "PUT" button.

7. View the deleted record:

The screenshot shows a browser window with the URL `127.0.0.1:8000/api/v1/comments/7/` highlighted with a red box. The page title is "Django REST framework" and the sub-page title is "api\_djangoadmin". The main content is titled "Comment Detail" with three buttons: "DELETE", "OPTIONS", and "GET". A red box highlights the "GET /api/v1/comments/7/" link. Below it, an "HTTP 404 Not Found" message is displayed, including headers: "Allow: GET, PUT, PATCH, DELETE, HEAD, OPTIONS", "Content-Type: application/json", and "Vary: Accept". The error message is: 

```
{ "detail": "No Comment matches the given query." }
```

. At the bottom, there are "Raw data" and "HTML form" tabs, and a "PUT" button.

8.