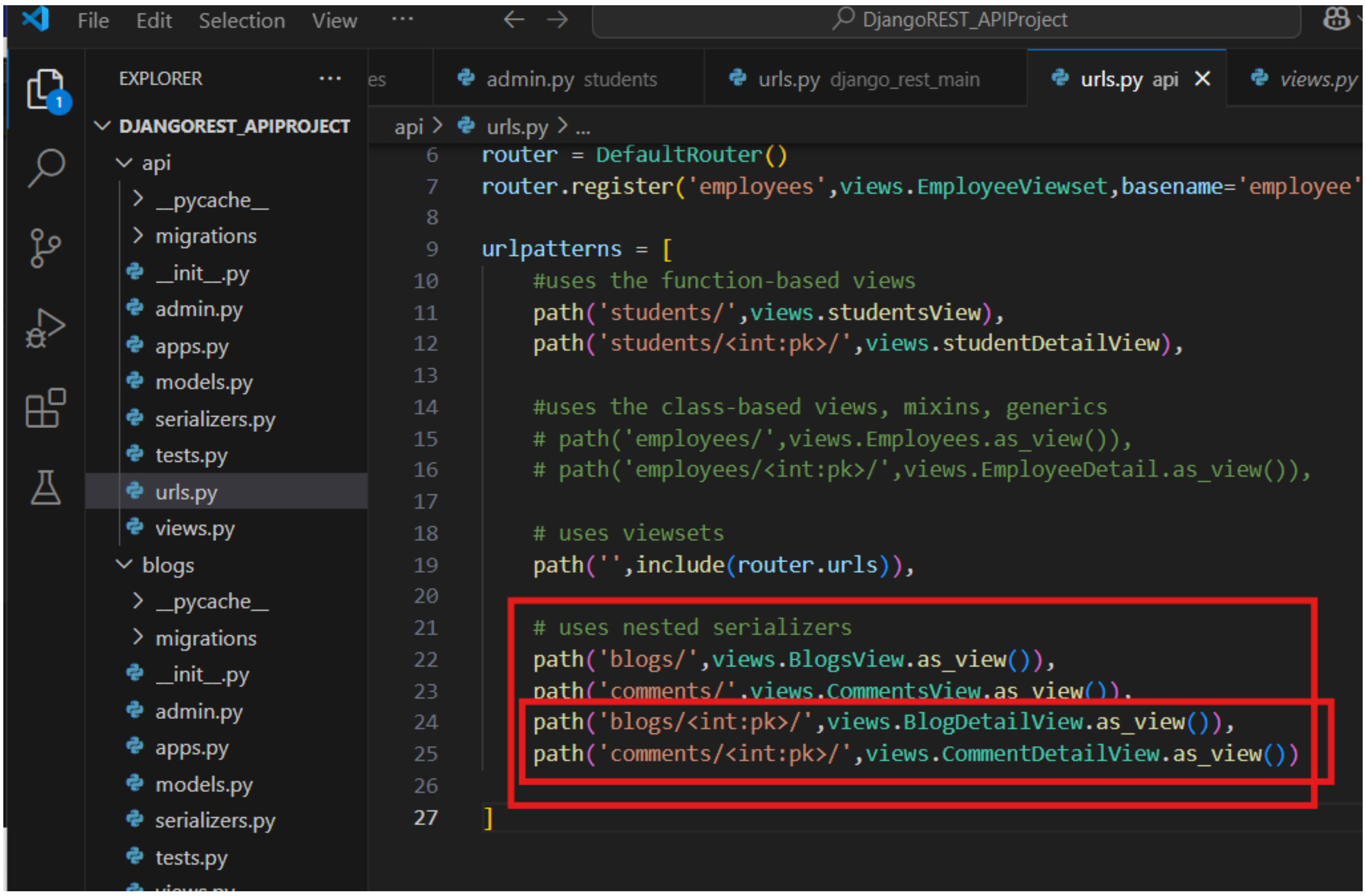
Topic: 13. Nested Serializers for Related Models Part 2

Speaker: Personal | Notebook: API Development using Django Framework

---

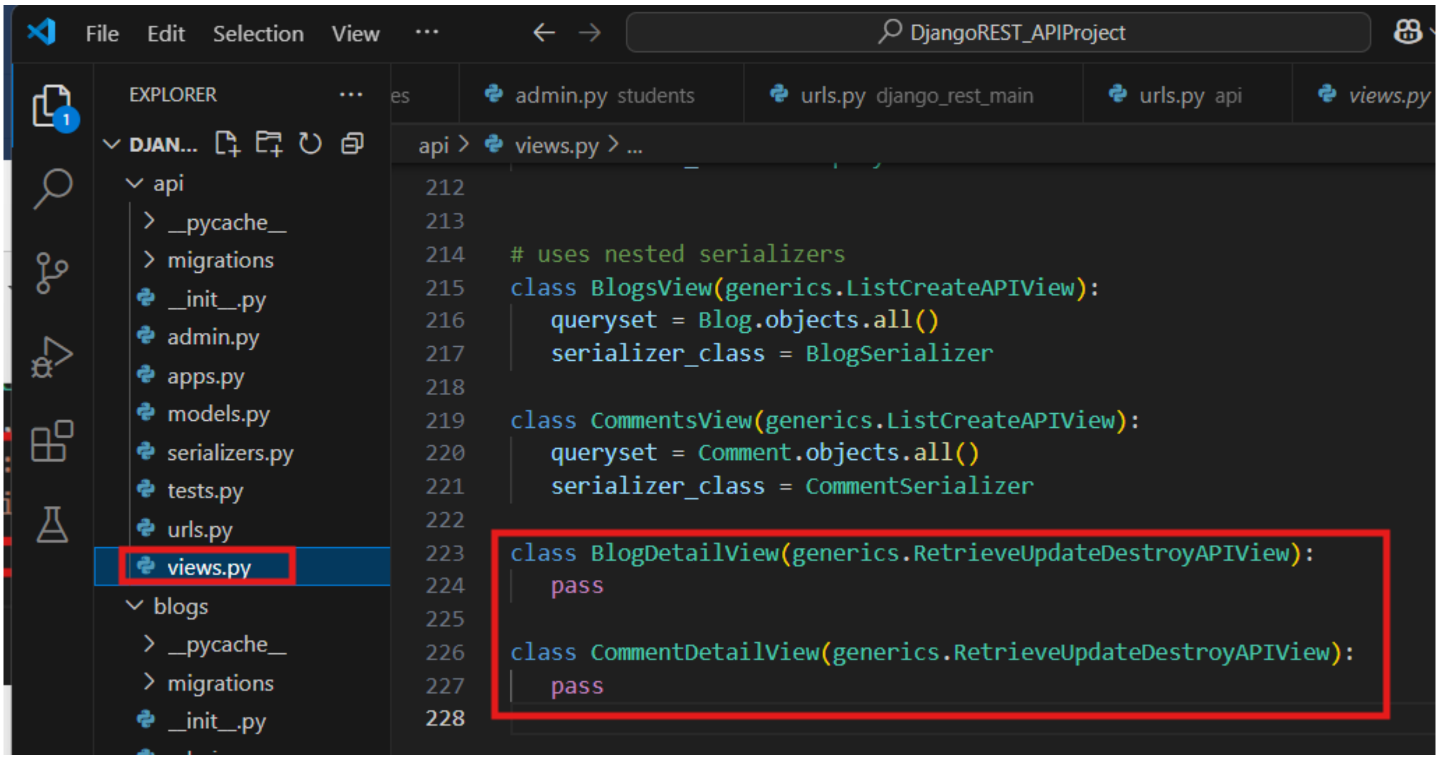
In this post, we use the primary key to allow other CRUD operations.

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



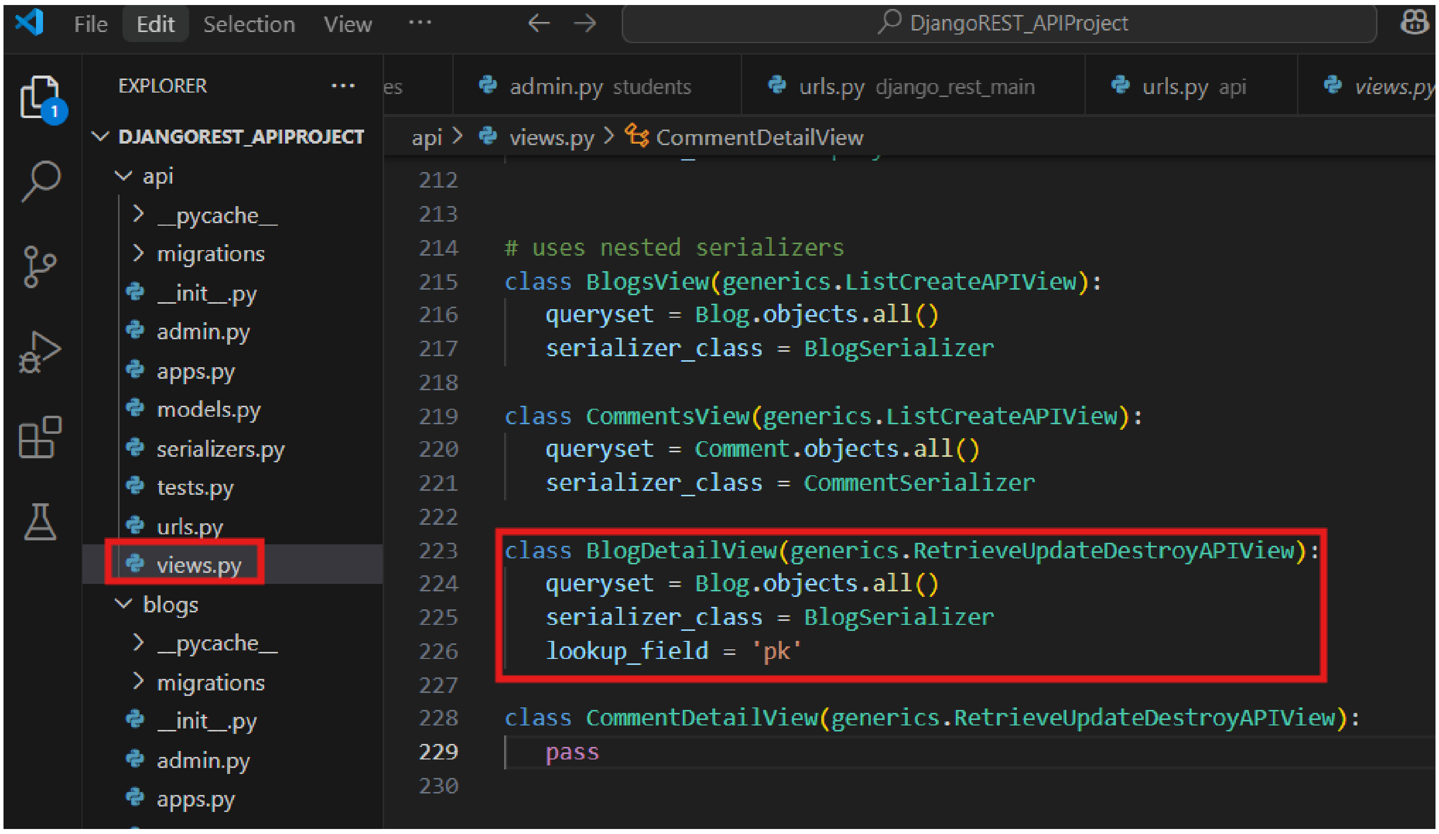
```
router = DefaultRouter()
router.register('employees',views.EmployeeViewset,basename='employee'

urlpatterns = [
    #uses the function-based views
    path('students/',views.studentsView),
    path('students/<int:pk>/',views.studentDetailView),

    #uses the class-based views, mixins, generics
    # path('employees/',views.Employees.as_view()),
    # path('employees/<int:pk>/',views.EmployeeDetail.as_view()),

    # uses viewsets
    path('',include(router.urls)),

    # uses nested serializers
    path('blogs/',views.BlogsView.as_view()),
    path('comments/',views.CommentsView.as_view()),
    path('blogs/<int:pk>/',views.BlogDetailView.as_view()),
    path('comments/<int:pk>/',views.CommentDetailView.as_view())

]
```

2. Update the API\VIEWS.PY



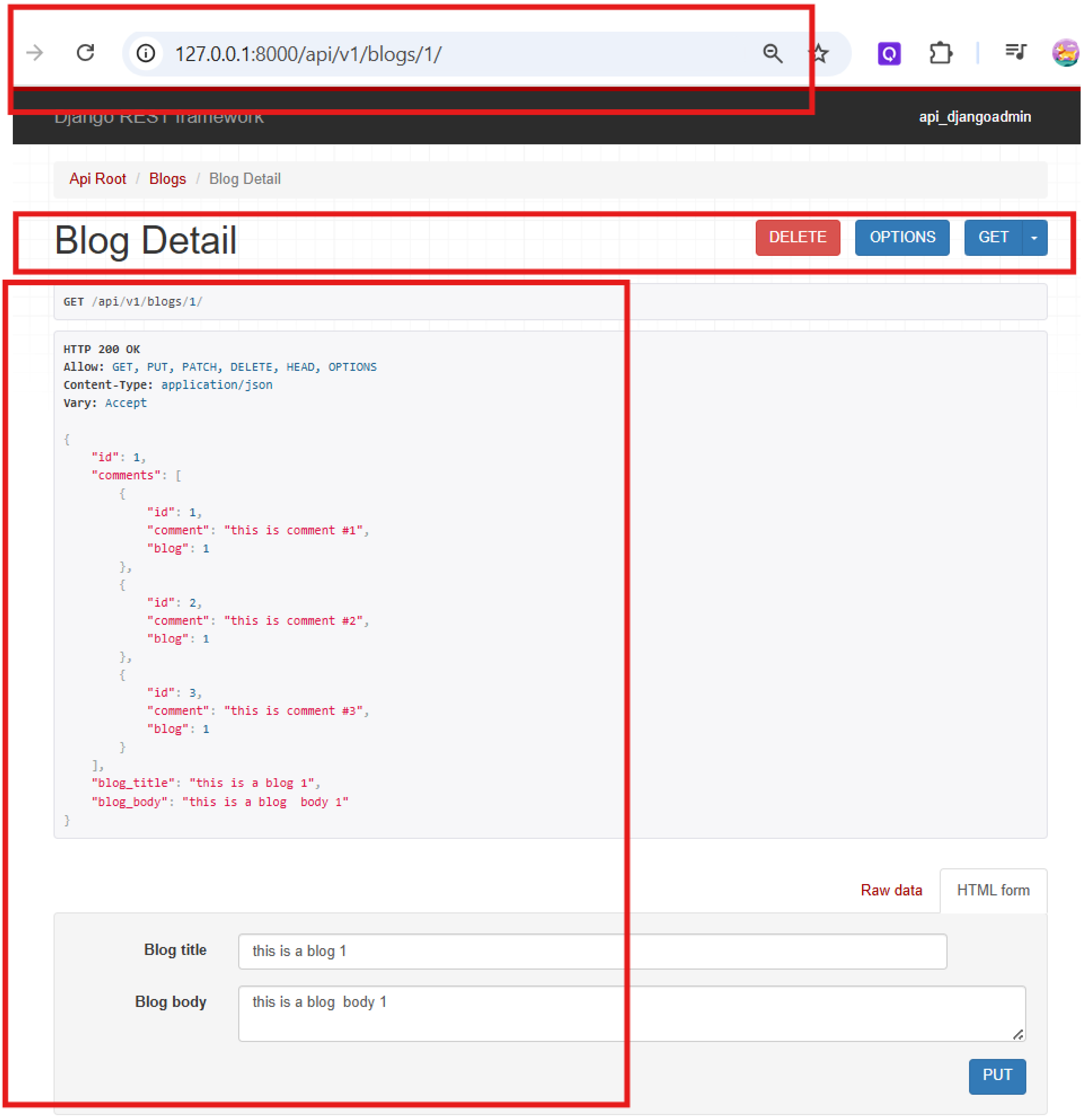
```
# uses nested serializers
class BlogsView(generics.ListCreateAPIView):
    queryset = Blog.objects.all()
    serializer_class = BlogSerializer

class CommentsView(generics.ListCreateAPIView):
    queryset = Comment.objects.all()
    serializer_class = CommentSerializer

class BlogDetailView(generics.RetrieveUpdateDestroyAPIView):
    pass

class CommentDetailView(generics.RetrieveUpdateDestroyAPIView):
    pass
```

Then update as:



```
# uses nested serializers
class BlogsView(generics.ListCreateAPIView):
    queryset = Blog.objects.all()
    serializer_class = BlogSerializer

class CommentsView(generics.ListCreateAPIView):
    queryset = Comment.objects.all()
    serializer_class = CommentSerializer

class BlogDetailView(generics.RetrieveUpdateDestroyAPIView):
    queryset = Blog.objects.all()
    serializer_class = BlogSerializer
    lookup_field = 'pk'

class CommentDetailView(generics.RetrieveUpdateDestroyAPIView):
    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

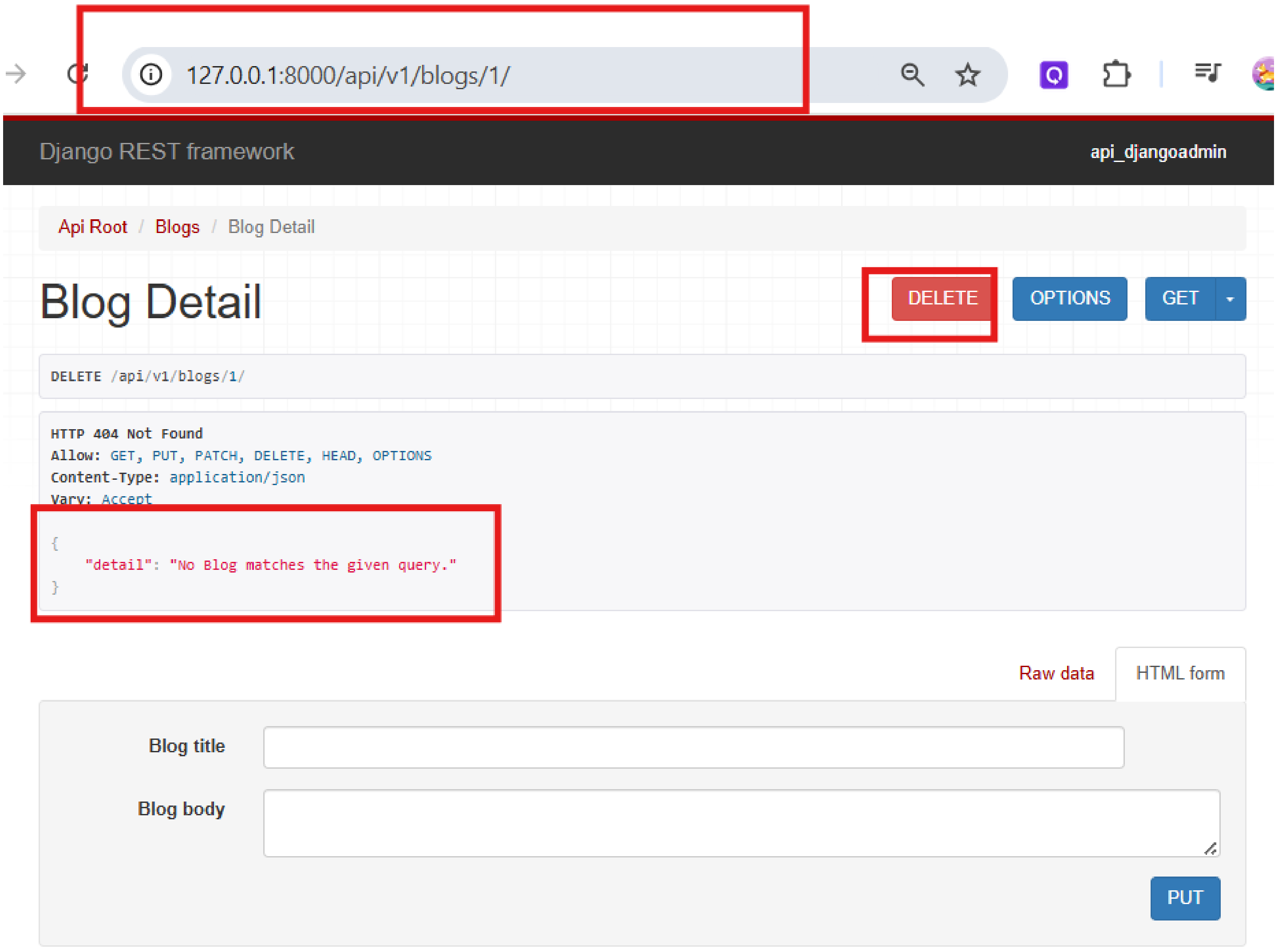
Blog Detail

```
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"
}
```

Blog title	this is a blog 1
Blog body	this is a blog body 1

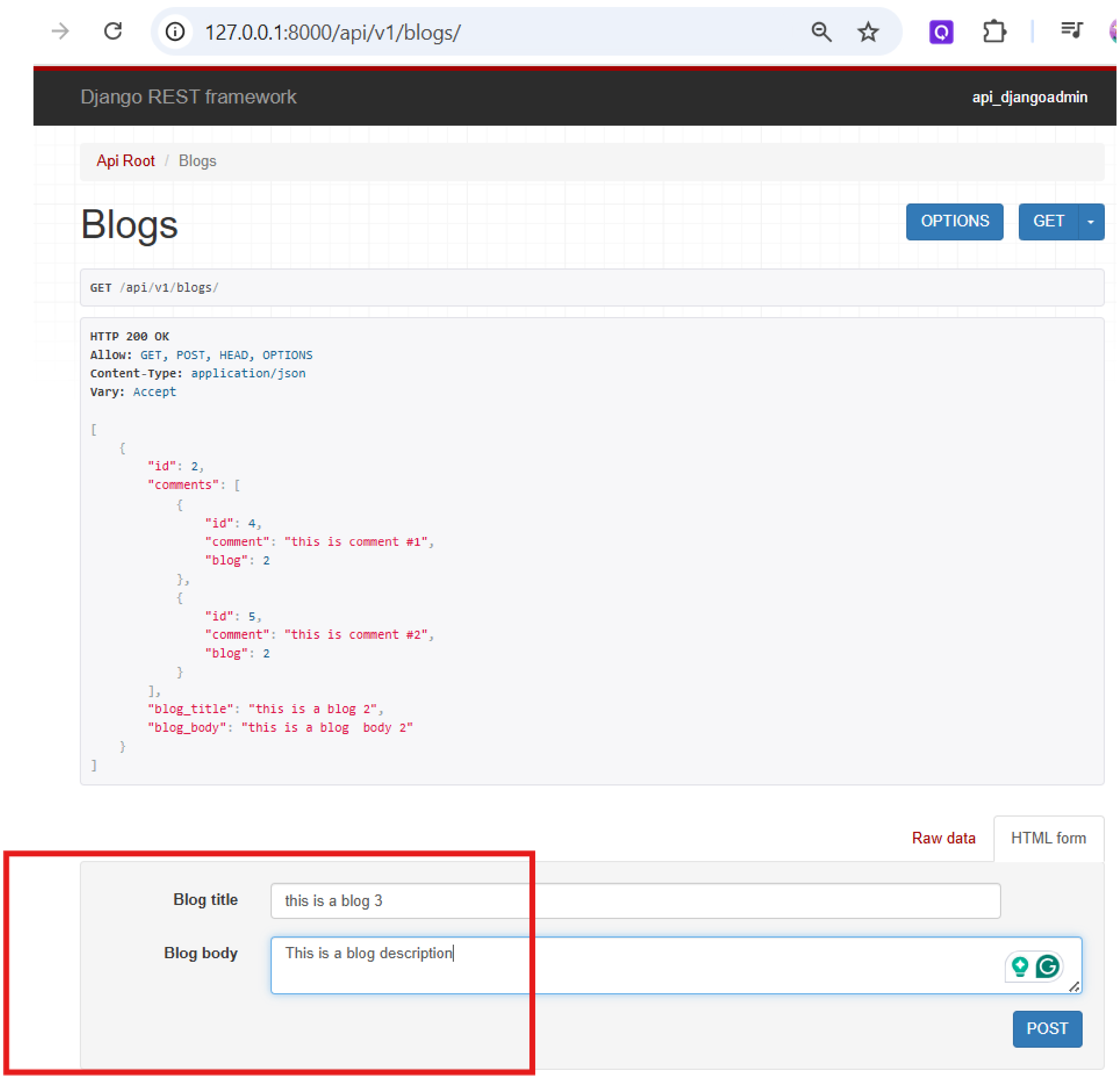
Now, you can update and delete this record.



```
DELETE /api/v1/blogs/1/
```

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

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

You can also add a new post here:

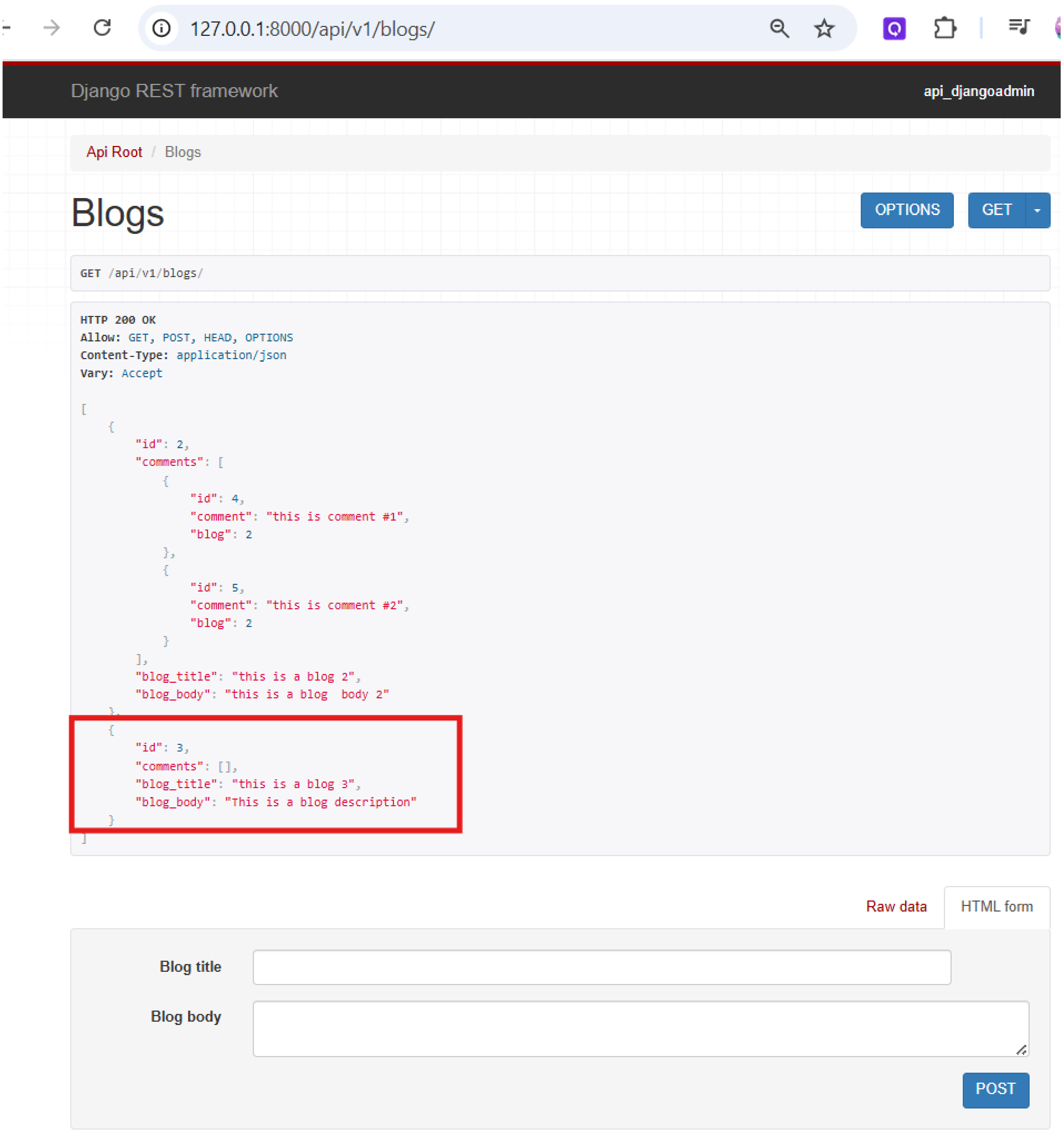


```
GET /api/v1/blogs/
```

```
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"
    }
]
```

Blog title	this is a blog 3
Blog body	This is a blog description

Then you can reload your page:

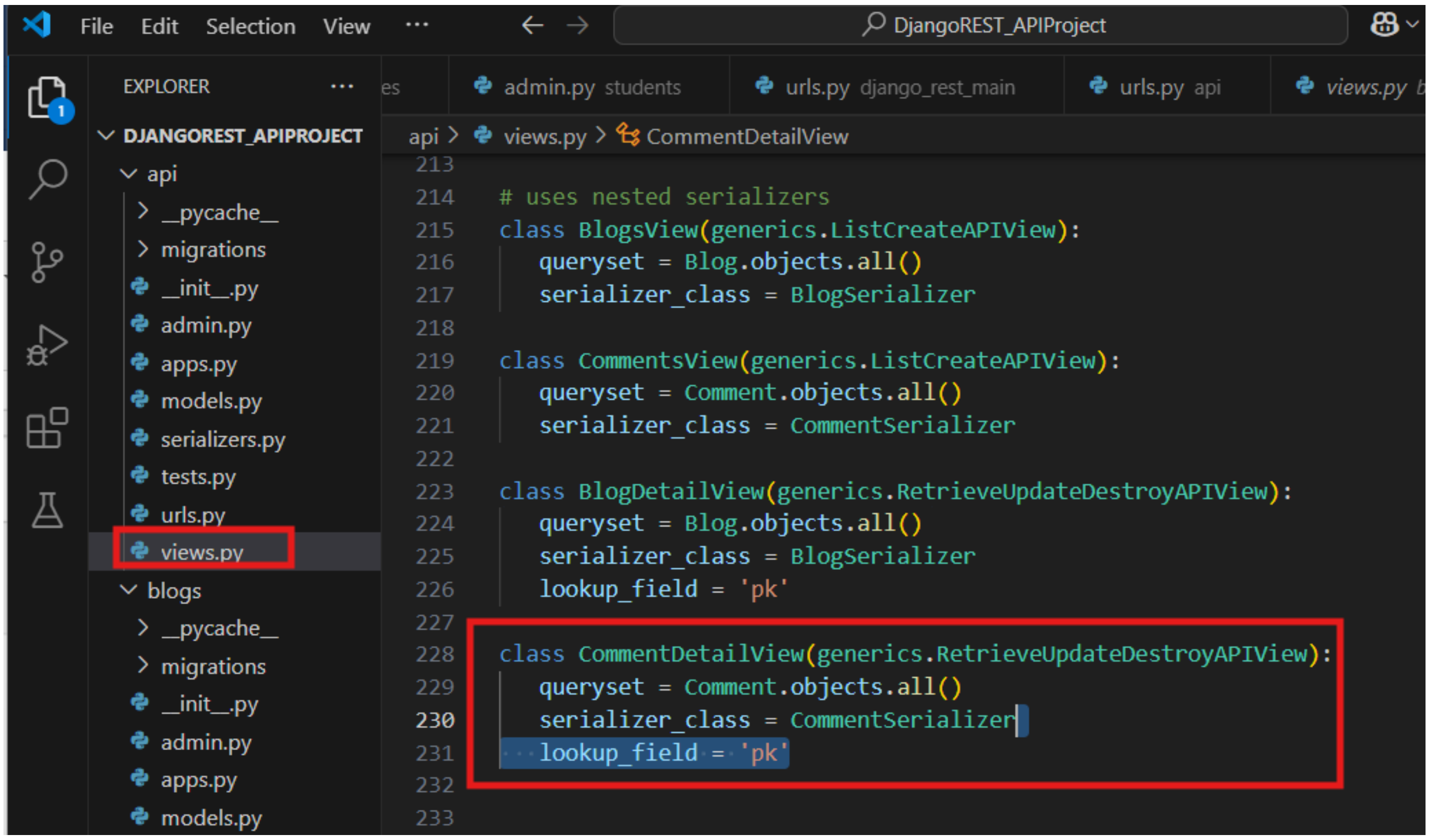


Blogs

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

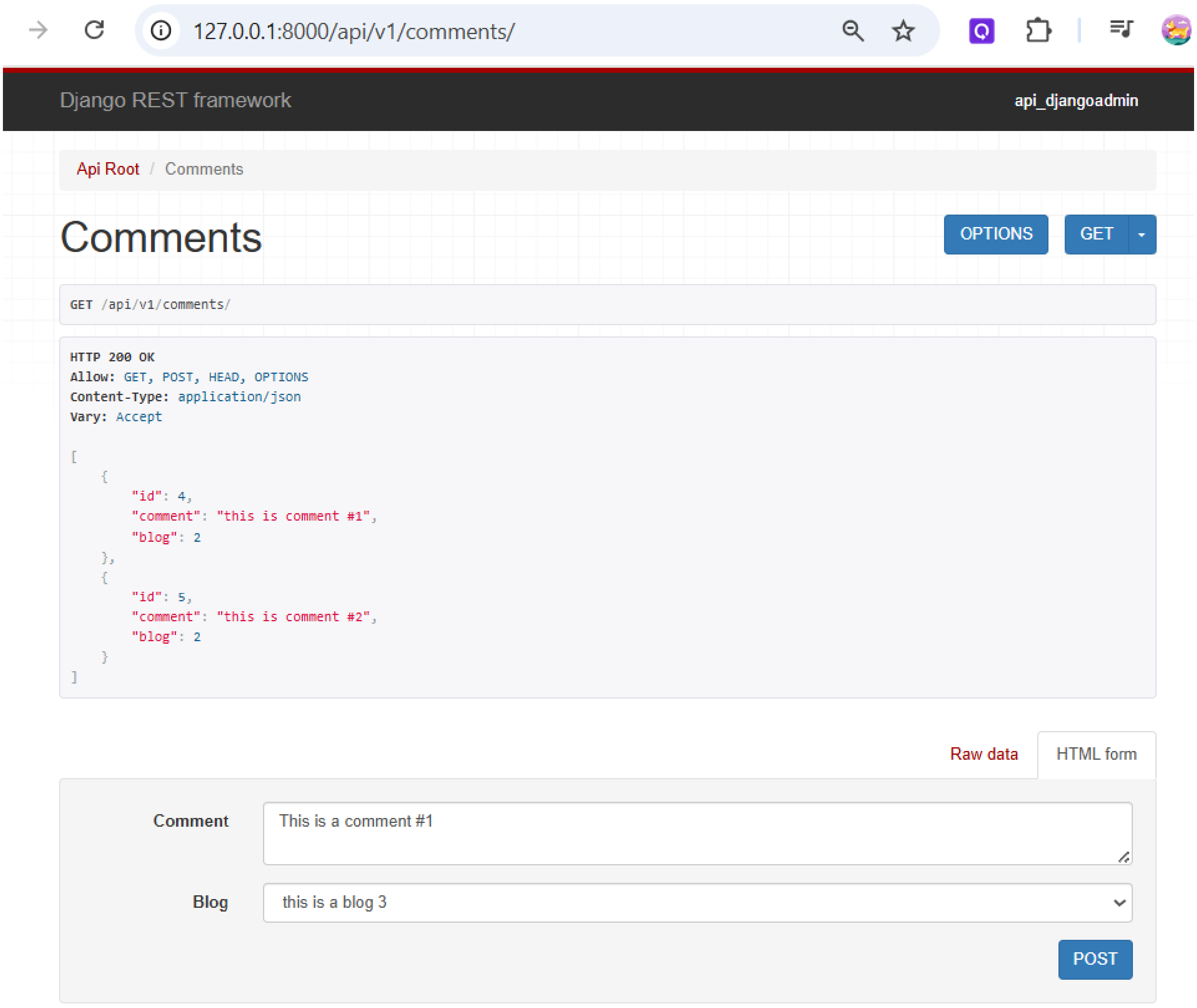
```
HTTP 200 OK
Allow: GET, POST, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

[
    {
        "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"
    }
]
```

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



```python

# uses nested serializers
class BlogsView(generics.ListCreateAPIView):
    queryset = Blog.objects.all()
    serializer_class = BlogSerializer

class CommentsView(generics.ListCreateAPIView):
    queryset = Comment.objects.all()
    serializer_class = CommentSerializer

class BlogDetailView(generics.RetrieveUpdateDestroyAPIView):
    queryset = Blog.objects.all()
    serializer_class = BlogSerializer
    lookup_field = 'pk'

class CommentDetailView(generics.RetrieveUpdateDestroyAPIView):
    queryset = Comment.objects.all()
    serializer_class = CommentSerializer
    lookup_field = 'pk'

```

5. To view the COMMENTS path:



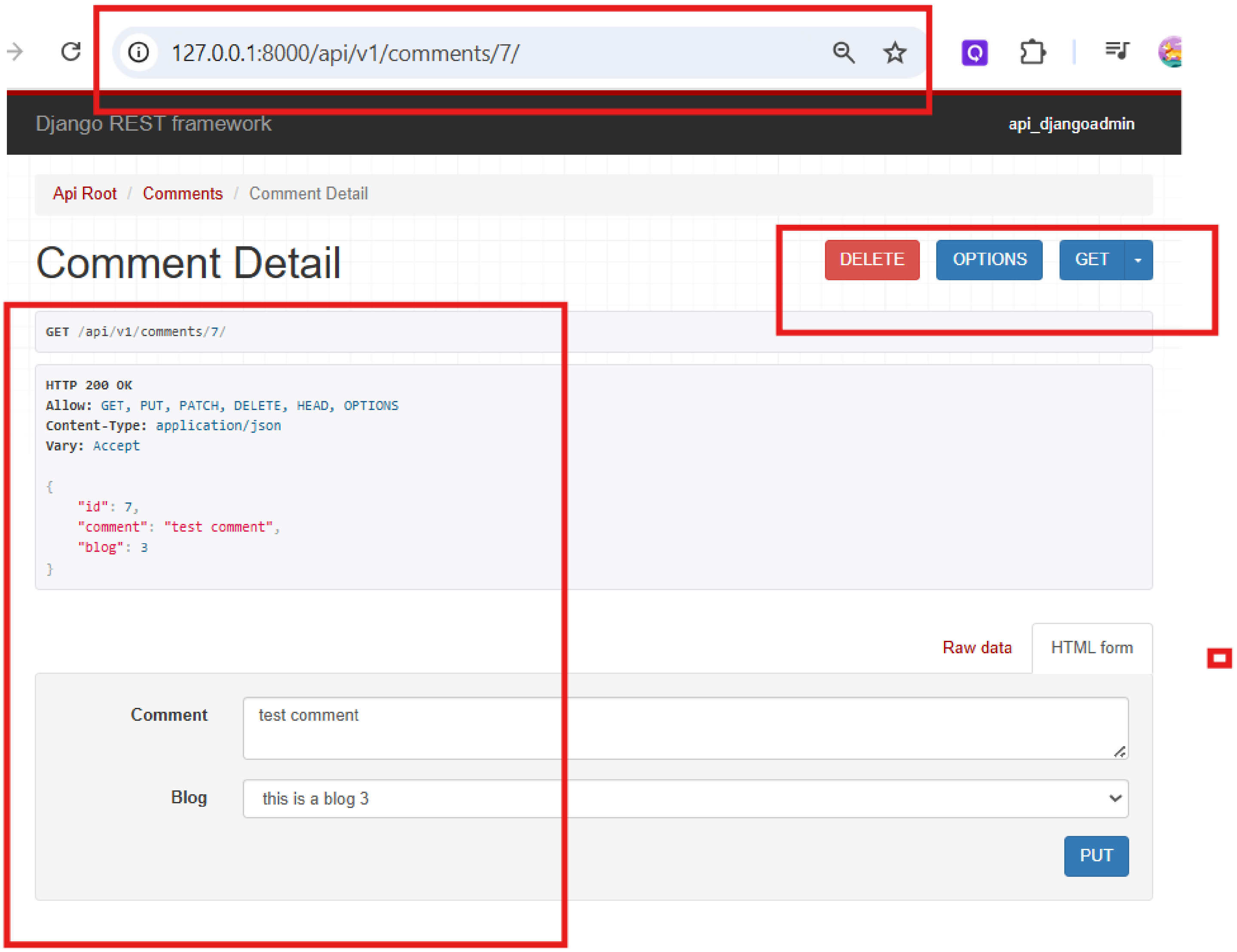
Comments

```
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
    }
]
```

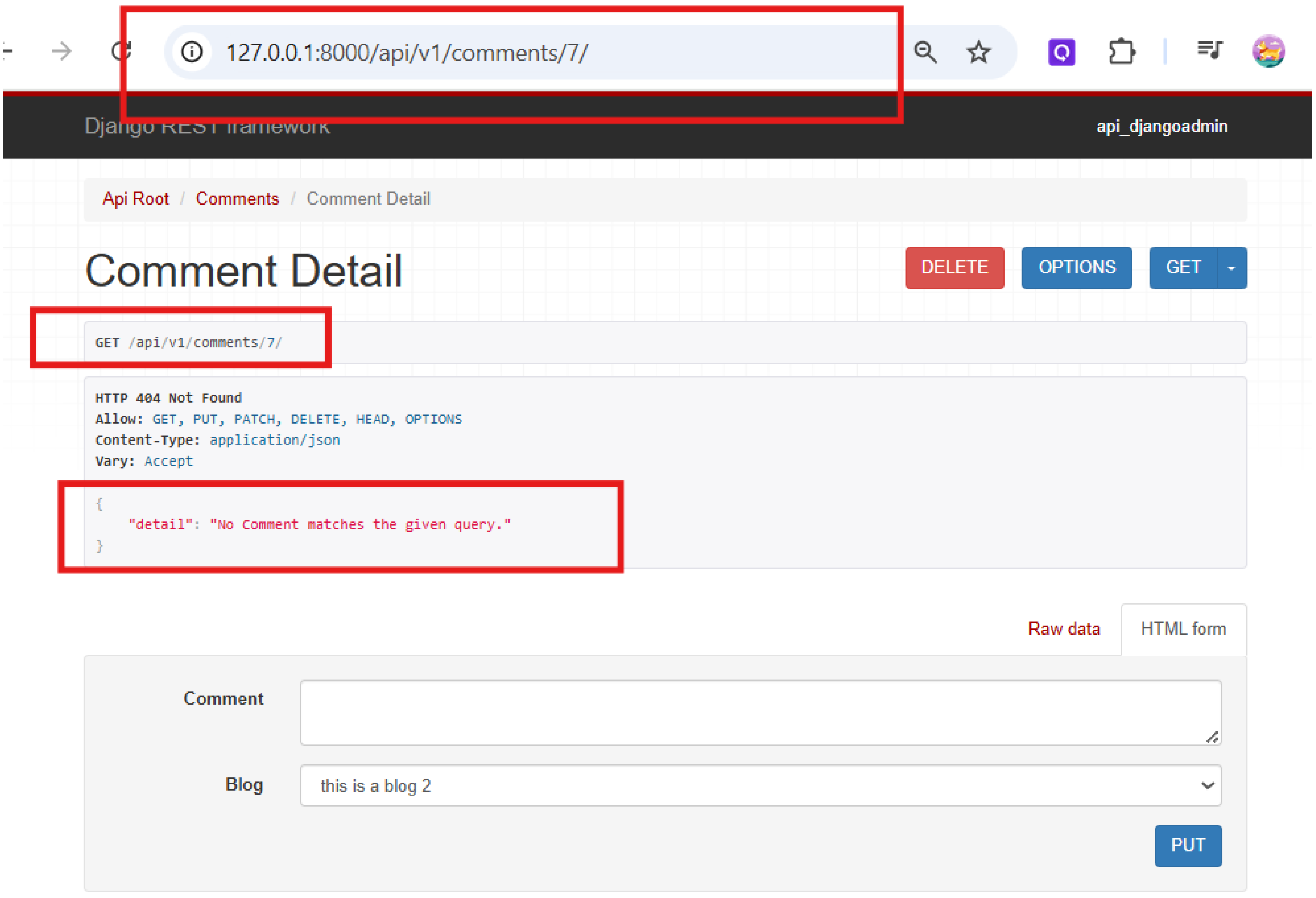
Comment	This is a comment #1
Blog	this is a blog 3

6. To delete the specific comment:



```
GET /api/v1/comments/7/

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

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

7. View the deleted record:


127.0.0.1:8000/api/v1/comments/7/
Django REST framework
api_djangoadmin
Api Root / Comments / Comment Detail

Comment Detail

```
GET /api/v1/comments/7/
```

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

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

Comment	
Blog	this is a blog 2

8.