

Django Filters & Search

Django Development II

Azimjon Pulatov | PDP Online

REST API - Searching, Sorting, Filtering

Products Merchandising



Sort Order

None

▼

Sort

Search by SKU or name

Search

Add or Remove Products

Visible items: 5

12

▼

 per page


<

1

 of 1

>

×




ZAINO TRAVEL

SKU: 206001984

In Stock 75,00 €

MANUAL

×



ZAINO MONVISO 3


SKU: 206001905

In Stock 110,00 €

MANUAL

⬆

×



ZAINO FREEWAY


SKU: 206001902

In Stock 95,00 €

MANUAL

⬆

×



ZAINO MONVISO 1


SKU: 206001903

In Stock 85,00 €

MANUAL

⬆

×



ZAINO MONVISO 2

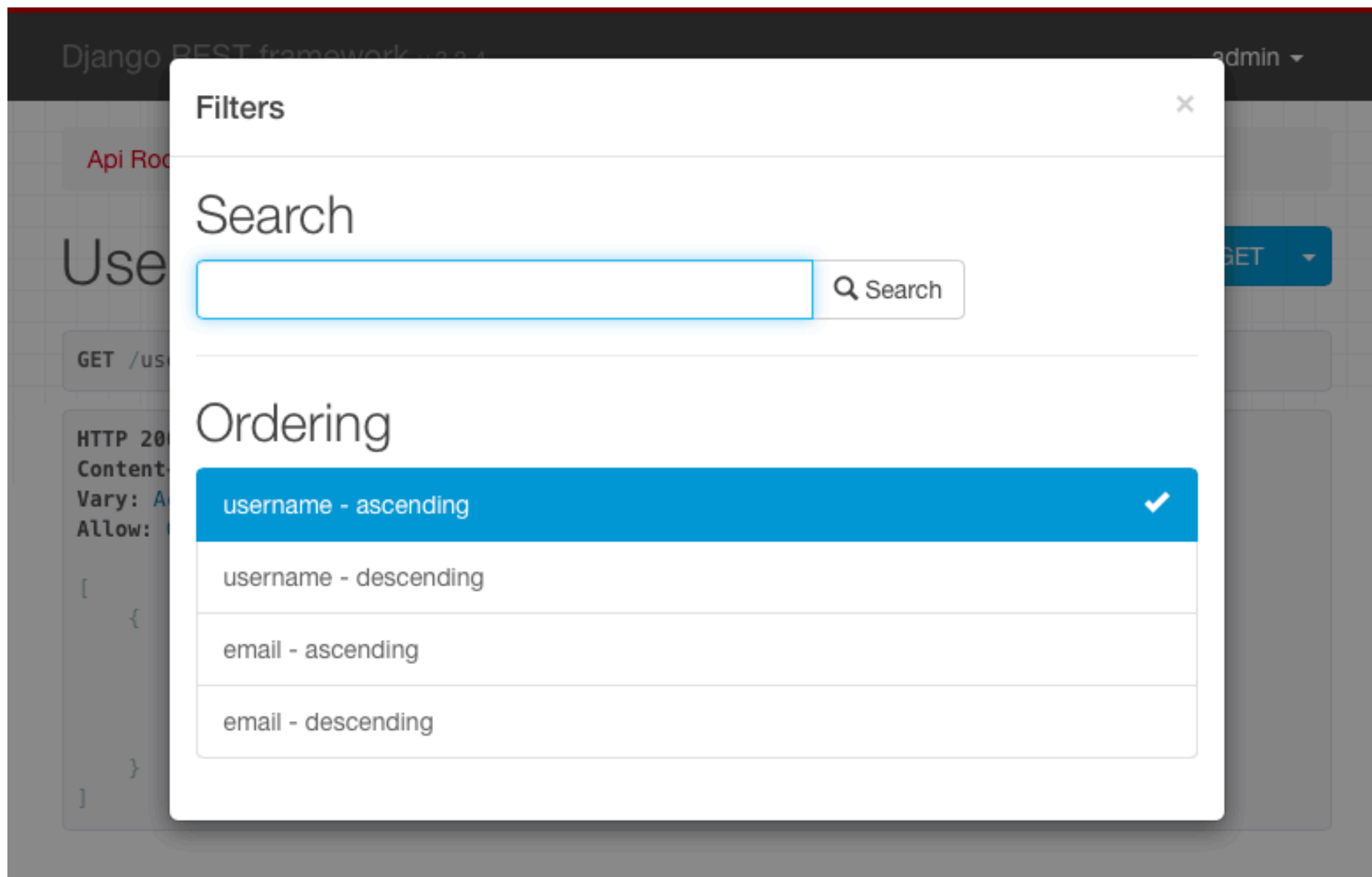
SKU: 206001904

In Stock 99,00 €

MANUAL

⬆

DRF - Search & Ordering



Filtering QuerySet

```
from rest_framework import generics

class ToDoList(generics.ListAPIView):
    serializer_class = ToDoSerializer

    def get_queryset(self):
        """
        This view should return a list of all the todos
        for the currently authenticated user.
        """
        user = self.request.user
        return ToDo.objects.filter(user=user)
```

Filtering QuerySet with Query Parameter

```
class SongList(generics.ListAPIView):
    serializer_class = SongSerializer

    def get_queryset(self):
        """
        Optionally restricts the returned purchases to a given user,
        by filtering against a `username` query parameter in the URL.
        """
        queryset = Song.objects.all()
        query = self.request.query_params.get('search')
        if query is not None:
            queryset = queryset.filter(name__icontains=query)
        return queryset
```


DRF Search Parameter

`http://example.com/api/users?search=russell`

```
from rest_framework import filters

class UserListView(generics.ListAPIView):
    queryset = User.objects.all()
    serializer_class = UserSerializer
    filter_backends = [filters.SearchFilter]
    search_fields = ['username', 'email']
```

DRF Filter Parameter

http://example.com/api/products?category=clothing&in_stock=True

```
class ProductList(generics.ListAPIView):  
    queryset = Product.objects.all()  
    serializer_class = ProductSerializer  
    filter_backends = [DjangoFilterBackend]  
    filterset_fields = ['category', 'in_stock']
```

Custom Search Parameter

- '^' Starts-with search.
- '=' Exact matches.
- '@' Full-text search. (Currently only supported Django's **PostgreSQL backend**.)
- '\$' Regex search.

For example:

```
search_fields = ['=username', '=email']
```


DRF Ordering Parameter

`http://example.com/api/users?ordering=-username`

```
class UserListView(generics.ListAPIView):  
    queryset = User.objects.all()  
    serializer_class = UserSerializer  
    filter_backends = [filters.OrderingFilter]  
    ordering_fields = ['username', 'email']
```



compture



 All

 Images

 News

 Videos

 Maps

 More

Settings

Tools

About 197,000 results (0.58 seconds)

Did you mean: **computer**

[w] [en.wiktionary.org](https://en.wiktionary.org/wiki/compture) > wiki > compture ▼

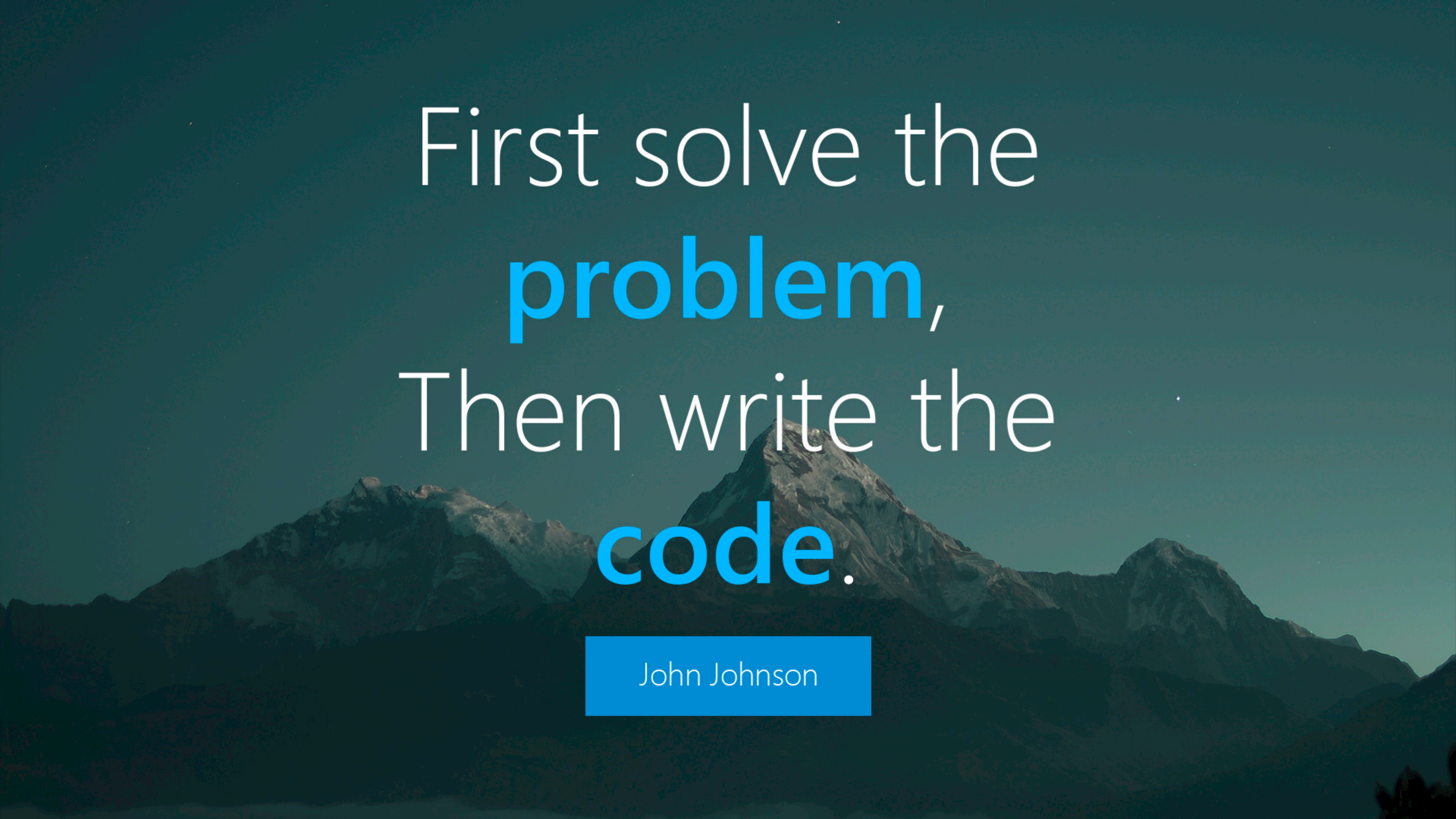
compture - Wiktionary

Wiktionary. Search. **compture**. Language; Watch · Edit. LatinEdit. ParticipleEdit. **cōmptūre**.

vocative masculine singular of cōmptūrus. Retrieved from ...

Similarity Search - PG Trigram

```
>>> from django.contrib.postgres.search import TrigramSimilarity
>>> Author.objects.create(name='Katy Stevens')
>>> Author.objects.create(name='Stephen Keats')
>>> test = 'Katie Stephens'
>>> Author.objects.annotate(
...     similarity=TrigramSimilarity('name', test),
... ).filter(similarity__gt=0.3).order_by('-similarity')
[<Author: Katy Stevens>, <Author: Stephen Keats>]
```

First solve the
problem,
Then write the
code.

John Johnson