DRF Filtering — A to Z (Complete Guide)
এখানে আমি তোমাকে **Filtering এর সবকিছু** একদম A → Z শিখিয়ে দেবো — Basic, Advanced, Search, Ordering, Custom Filter — সবকিছু + Example সহ।

✅ 1. Filtering কী?

Filtering মানে হচ্ছে query param দিয়ে ডেটা ফিল্টার করা:

```
/employees/?department=IT
/employees/?is_active=true
```

✅ 2. DRF-এ Filtering করার ৪টা Method আছে

Method	Description
1. Manual Filtering	নিজে queryset ফিল্টার করা
2. `DjangoFilterBackend`	Most powerful filtering
3. `SearchFilter`	Full text search
4. `OrderingFilter`	Order by any field

🎯 A → Manual Filtering (Basic)

views.py

```python
from rest_framework import viewsets
from employees.models import Employee
from employees.serializers import EmployeeSerializer

class EmployeeViewSet(viewsets.ModelViewSet):
    queryset = Employee.objects.all()
    serializer_class = EmployeeSerializer

    def get_queryset(self):
        qs = super().get_queryset()

        dept = self.request.query_params.get("department")
        active = self.request.query_params.get("is_active")

        if dept:
            qs = qs.filter(department=dept)

        if active:
            qs = qs.filter(is_active=active.lower() == "true")

        return qs
```

🎯 B → `DjangoFilterBackend` (Most Used Filtering)

1️⃣ Install

```
pip install django-filter
```

2️⃣ settings.py

```
REST_FRAMEWORK = {
    'DEFAULT_FILTER_BACKENDS': [
        'django_filters.rest_framework.DjangoFilterBackend'
    ]
}
```

3️⃣ views.py

```
from django_filters.rest_framework import DjangoFilterBackend

class EmployeeViewSet(viewsets.ModelViewSet):
    queryset = Employee.objects.all()
    serializer_class = EmployeeSerializer
    filter_backends = [DjangoFilterBackend]
    filterset_fields = ['department', 'is_active', 'position']
```

👉 Now you can filter

```
/employees/?department=HR
/employees/?is_active=true
/employees/?position=Manager
```

🎯 C → Advanced Filtering with Custom FilterSet

filters.py

```
import django_filters
from employees.models import Employee

class EmployeeFilter(django_filters.FilterSet):
    start_date = django_filters.DateFilter(field_name="date_join",
lookup_expr="gte")
    end_date = django_filters.DateFilter(field_name="date_join",
lookup_expr="lte")
    name = django_filters.CharFilter(field_name="first_name",
lookup_expr="icontains")

    class Meta:
        model = Employee
        fields = ['department', 'is_active']
```

views.py

```
class EmployeeViewSet(viewsets.ModelViewSet):
    queryset = Employee.objects.all()
    serializer_class = EmployeeSerializer
    filter_backends = [DjangoFilterBackend]
    filterset_class = EmployeeFilter
```

Now you can do:

```
/employees/?start_date=2024-01-01
/employees/?end_date=2024-12-31
/employees/?name=At
/employees/?department=IT&is_active=true
```

🎯 D → Search Filtering (SearchFilter)

views.py

```
from rest_framework.filters import SearchFilter

class EmployeeViewSet(viewsets.ModelViewSet):
    queryset = Employee.objects.all()
    serializer_class = EmployeeSerializer
    filter_backends = [SearchFilter]
    search_fields = ['first_name', 'last_name', 'email', 'phone']
```

Now you can search:

```
/employees/?search=rahman
/employees/?search=018
```

🎯 E → Ordering Filter

views.py

```python
from rest_framework.filters import OrderingFilter

class EmployeeViewSet(viewsets.ModelViewSet):
    queryset = Employee.objects.all()
    serializer_class = EmployeeSerializer
    filter_backends = [OrderingFilter]
    ordering_fields = ['first_name', 'date_join']
```

Example:

```
/employees/?ordering=date_join
/employees/?ordering=-first_name
```

🎯 F → Combine All Filters (Best Practice)

```python
from rest_framework.filters import SearchFilter, OrderingFilter
from django_filters.rest_framework import DjangoFilterBackend

class EmployeeViewSet(viewsets.ModelViewSet):
    queryset = Employee.objects.all()
    serializer_class = EmployeeSerializer
    filter_backends = [DjangoFilterBackend, SearchFilter, OrderingFilter]

    filterset_fields = ['department', 'is_active', 'position']
    search_fields = ['first_name', 'last_name', 'email']
    ordering_fields = ['date_join', 'first_name']
```

🎯 G → Filtering with Relations (ForeignKey)

Example: Filter all employees in a specific company

```
/employees/?company__name=Google
/employees/?department__name=Marketing
```

🎯 H → Custom Filtering Logic (Advanced)

```
def get_queryset(self):
    qs = super().get_queryset()
    min_salary = self.request.query_params.get("min_salary")

    if min_salary:
        qs = qs.filter(salary__gte=min_salary)

    return qs
```

🎯 I → URL Filtering Example

```
/employees/?department=IT&is_active=true&ordering=-date_join&search=atik
```

🟢 প্রয়োজনে তুমি যেকোনো প্রজেক্টের জন্য —

আমি চাইলে **complete filtering system** setup করে দিতে পারবো (models + views + URLs + advanced filters)।

চাও কি তোমার Employee project-এর উপর **ready-made filtering code**?

DRF Searching করার পুরো A → Z Guide (সবচেয়ে সহজ ভাষায় + Example সহ)

তুমি চেয়েছো **searching korer upay** → আমি এখন DRF এ SearchFilter দিয়ে কীভাবে search কাজ করে সেটা পুরো গাইড দিচ্ছি।

✅ 1. DRF SearchFilter কী?

SearchFilter ব্যবহার করে তুমি query parameter দিয়ে “LIKE search” করতে পারো।

উদাহরণ:

```
/employees/?search=atik
```

এটা first_name, last_name, email—যে field গুলো search_fields-এ দেওয়া আছে সেগুলোতে match খুঁজবে।

✅ 2. Basic Search Filter Setup

views.py

```python
from rest_framework import viewsets
from rest_framework.filters import SearchFilter
from employees.models import Employee
from employees.serializers import EmployeeSerializer

class EmployeeViewSet(viewsets.ModelViewSet):
    queryset = Employee.objects.all()
    serializer_class = EmployeeSerializer
    filter_backends = [SearchFilter]
    search_fields = ['first_name', 'last_name', 'email']
```

🔍 3. এখন তুমি search করতে পারো:

```
/employees/?search=rahman
/employees/?search=ati
/employees/?search=018
```

ডাটার যে fields এ search_fields এ লিখেছো সেগুলোতে LIKE search করবে।

🔥 4. Advanced Searching (Prefix Operators)

DRF search_fields-এ special operators ব্যবহার করে আরো powerful search করা যায়।

🔵 4.1 Starts With Search (^ prefix)

Only শুরু মিললে match হবে:

```
search_fields = ['^first_name']
```

```
/employees/?search=a
```

→ "Atik", "Arif", "Amin" এসব আসবে
→ "Fatik" আসবে না

🔵 4.2 Ends With Search ($ prefix)

```
search_fields = ['$email']
```

```
/employees/?search=gmail.com
```

→ শুধু gmail.com দিয়ে শেষ হওয়া email গুলো আসবে

🔵 4.3 Exact Match (= prefix)

```
search_fields = ['=phone']
```

```
/employees/?search=018123456
```

Exact match না হলে আসবে না।

🔵 4.4 Contains Search (default: no prefix)

ইটিই default behaviour.

```
search_fields = ['first_name']
```

```
/employees/?search=ik
```

→ "Atik", "Shafik", "Manik" সব আসবে

🟣 5. Search with Relations (ForeignKey/OneToMany)

Example:

Employee → Department (FK)

views.py

```
search_fields = [
    'first_name',
    'department__name'
]
```

Now you can search:

```
/employees/?search=HR
/employees/?search=Marketing
```

🟢 6. Combine Search + Filter + Ordering

```
from rest_framework.filters import SearchFilter, OrderingFilter
from django_filters.rest_framework import DjangoFilterBackend

class EmployeeViewSet(viewsets.ModelViewSet):
    queryset = Employee.objects.all()
    serializer_class = EmployeeSerializer
    filter_backends = [DjangoFilterBackend, SearchFilter, OrderingFilter]

    filterset_fields = ['department', 'is_active']
    search_fields = ['first_name', 'last_name', 'email',
'department__name']
    ordering_fields = ['first_name', 'date_join']
```

🎯 7. Example Query (Real Project Level)

```
/employees/?search=atik&department=HR&ordering=-date_join
```

এখানে 3টা কাজ একসাথে হচ্ছে:

1. নাম/email দিয়ে search
2. department দিয়ে filter
3. সর্বশেষ join করা employee আগে দেখাবে

⭐ চাইলে তোমার Employee Project এর জন্য —

আমি full **Search + Filter + Ordering** ready-made production code দিয়ে দেবো।
বললেই দিবো!