

DJANGO REST FRAMEWORK

PROYECTO BIBLIOTECA

Presentado por:

1152358 - Israel Bulla Rey

Capturas del código

Modelos

```
1  from django.db import models
2  from .validators import *
3  from django.core.validators import MinValueValidator, MaxValueValidator
4
5  # Create your models here.
6  class Autor(models.Model):
7      nombre = models.CharField(max_length=100, validators=[validate_blank])
8      nacionalidad = models.CharField(max_length=100)
9
10     def __str__(self):
11         return self.nombre
12
13     class Libro(models.Model):
14         titulo = models.CharField(max_length=100)
15         autor = models.ForeignKey(Autor, on_delete=models.CASCADE, related_name='libros')
16         fecha_publicacion = models.DateField()
17         resumen = models.TextField(validators=[validate_character])
18
19         def __str__(self):
20             return self.titulo
21
22     class Resena(models.Model):
23         libro = models.ForeignKey(Libro, on_delete=models.CASCADE, related_name='resenas')
24         texto = models.TextField()
25         calificacion = models.IntegerField(validators=[validate_calification])
26         fecha = models.DateTimeField(auto_now_add=True)
27         rating = models.FloatField(default=0, validators=[MinValueValidator(0.0), MaxValueValidator(5.0)])
28
29         def __str__(self):
30             return f"{self.libro.titulo} - {self.calificacion}/5"
```

Serializadores

```
from rest_framework import serializers
from .models import *

class AutorSerializer(serializers.ModelSerializer):
    class Meta:
        model = Autor
        fields = '__all__'

class LibroSerializer(serializers.ModelSerializer):
    recent_reviews = serializers.SerializerMethodField()
    author_name = serializers.ReadOnlyField(source='autor.nombre')

    def get_recent_reviews(self, obj):
        reviews = obj.resenas.order_by('-fecha')[:5]
        return ResenaSerializer(reviews, many=True).data

    class Meta:
        model = Libro
        fields = '__all__'

class ResenaSerializer(serializers.ModelSerializer):
    class Meta:
        model = Resena
        fields = '__all__'
```

ViewSets

Inicio hasta AutorViewSet

```
from .models import *
from .serializers import *
from rest_framework import viewsets
from django.db.models import Avg
from rest_framework.decorators import action
from rest_framework.response import Response
from django_filters.rest_framework import DjangoFilterBackend
from rest_framework.filters import OrderingFilter

class AutorViewSet(viewsets.ModelViewSet):
    queryset = Autor.objects.all()
    serializer_class = AutorSerializer
    filter_backends = [DjangoFilterBackend, OrderingFilter]
    #filterset_fields = ['nombre', 'nacionalidad']
    ordering_fields = ['nombre', 'nacionalidad']

    def get_queryset(self):
        queryset = Autor.objects.all()
        nombreautor = self.request.query_params.get('nombre')
        nacionalidad = self.request.query_params.get('nacionalidad')
        if nombreautor:
            queryset = queryset.filter(nombre__icontains=nombreautor)
        if nacionalidad:
            queryset = queryset.filter(nacionalidad__icontains=nacionalidad)
        return queryset

    def perform_create(self, serializer):
        autor = serializer.save()
        print(f"Autor creado: {autor.nombre}")
```

LibroViewSet

```
class LibroViewSet(viewsets.ModelViewSet):
    queryset = Libro.objects.all()
    serializer_class = LibroSerializer
    filter_backends = [DjangoFilterBackend, OrderingFilter]
    filterset_fields = ['autor', 'fecha_publicacion']
    ordering_fields = ['titulo', 'fecha_publicacion']

    def get_queryset(self):
        if self.request.query_params.get('reciente'):
            return Libro.objects.order_by('-fecha_publicacion')[:10]
        return Libro.objects.all()

    @action(detail=True, methods=['get'])
    def rating_aprox(self, request, pk=None):
        libro = self.get_object()
        aprox = libro.resenas.aggregate(rating_aprox=Avg('rating'))['rating_aprox']
        return Response({'rating_aprox': aprox})
```

ResenaViewSet

```
class ResenaViewSet(viewsets.ModelViewSet):
    queryset = Resena.objects.all()
    serializer_class = ResenaSerializer
    filter_backends = [DjangoFilterBackend, OrderingFilter]
    filterset_fields = ['calificacion', 'rating']
    ordering_fields = ['calificacion', 'rating']

    def get_queryset(self):
        if self.request.query_params.get('reciente'):
            return Resena.objects.order_by('-fecha')
        return Resena.objects.all()

    def perform_create(self, serializer):
        resena = serializer.save()
        print(f"Reseña creada para el libro: {resena.libro.titulo}")
```

urls

```
"""
URL configuration for biblioteca_project project.

The `urlpatterns` list routes URLs to views. For more information please see:
|   https://docs.djangoproject.com/en/5.2/topics/http/urls/
Examples:
Function views
    1. Add an import:  from my_app import views
    2. Add a URL to urlpatterns:  path('', views.home, name='home')
Class-based views
    1. Add an import:  from other_app.views import Home
    2. Add a URL to urlpatterns:  path('', Home.as_view(), name='home')
Including another URLconf
    1. Import the include() function: from django.urls import include, path
    2. Add a URL to urlpatterns:  path('blog/', include('blog.urls'))
"""
from django.contrib import admin
from biblioteca.modelviewsets import *
from rest_framework.routers import DefaultRouter
from django.urls import path, include

router = DefaultRouter()
router.register('libros', LibroViewSet)
router.register('autores', AutorViewSet)
router.register('resenas', ResenaViewSet)

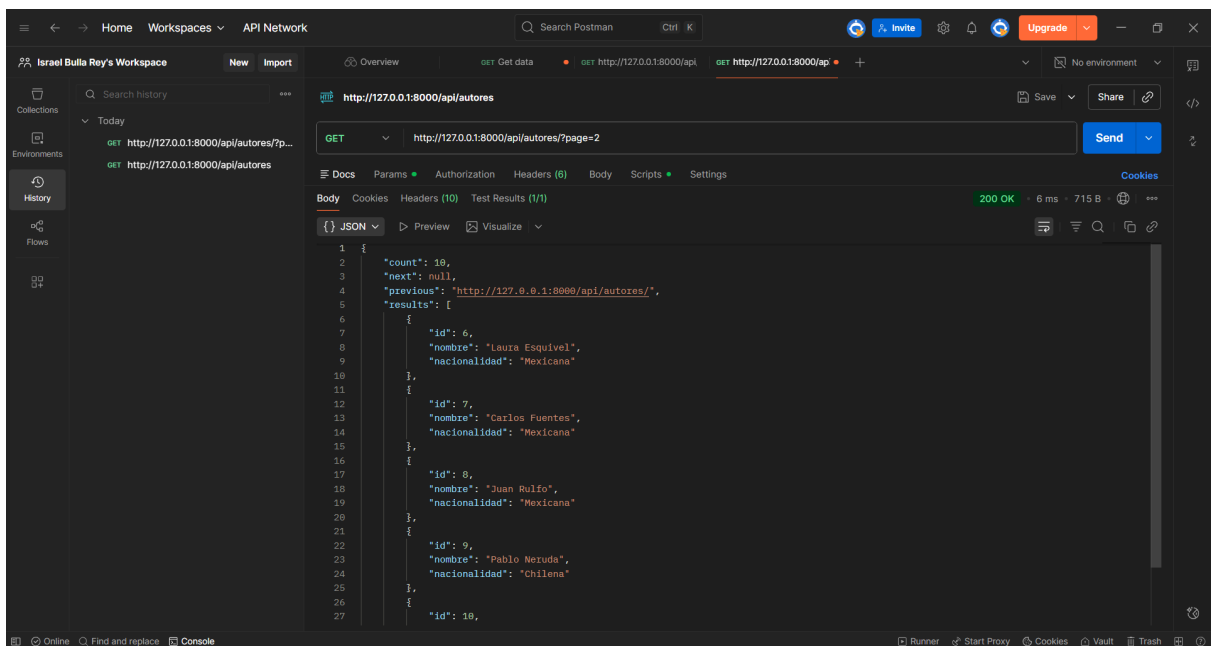
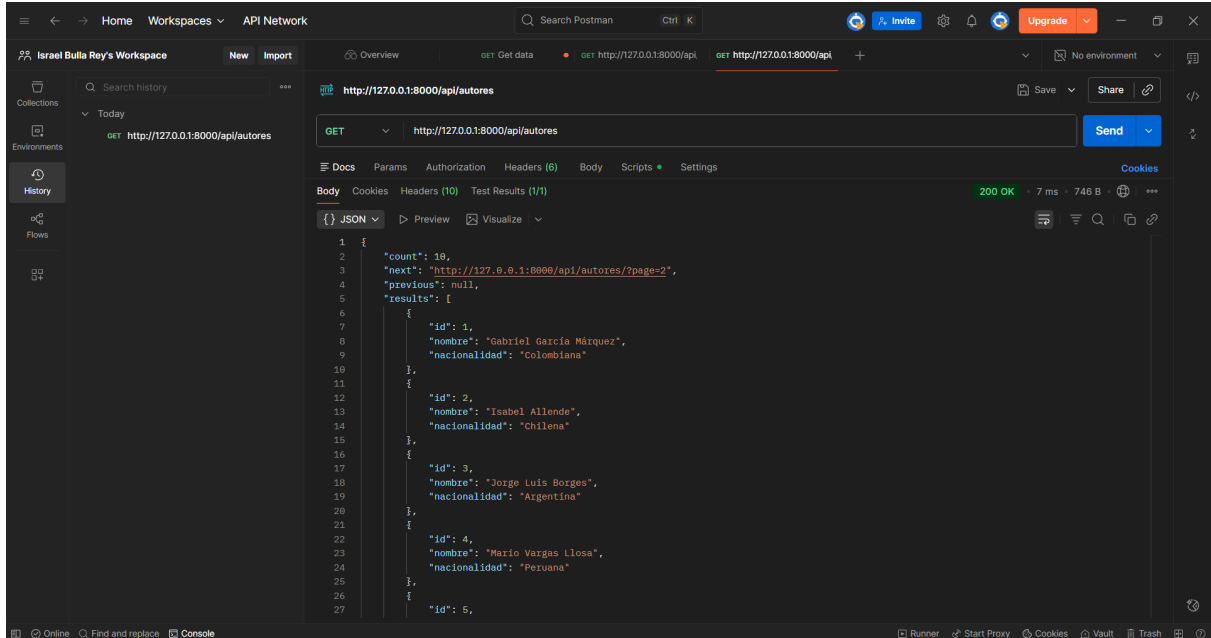
urlpatterns = [
    path('admin/', admin.site.urls),
    path('api/', include(router.urls)),
]
```

Evidencias del uso de la API

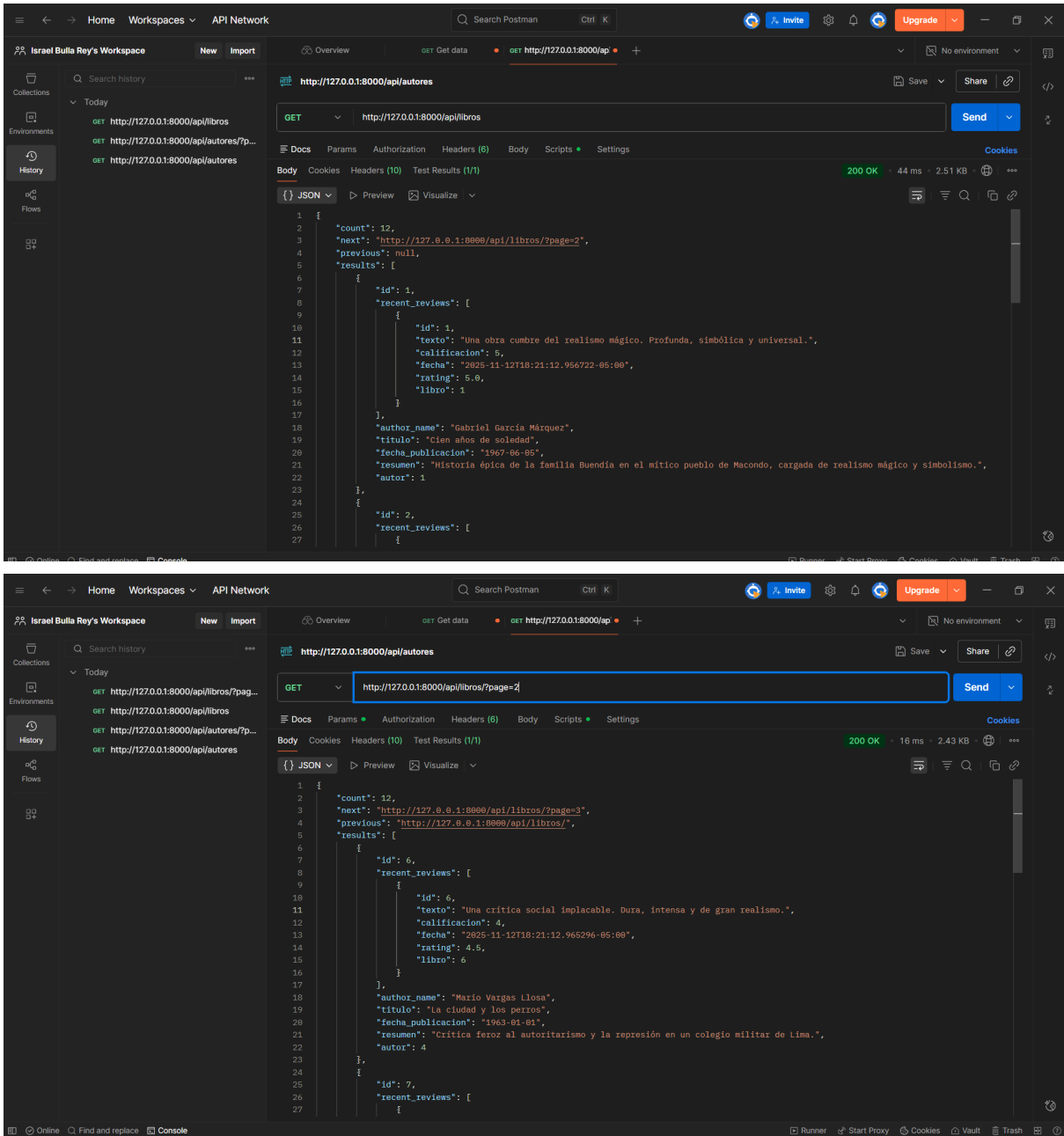
Comprobado mediante el uso de Postman

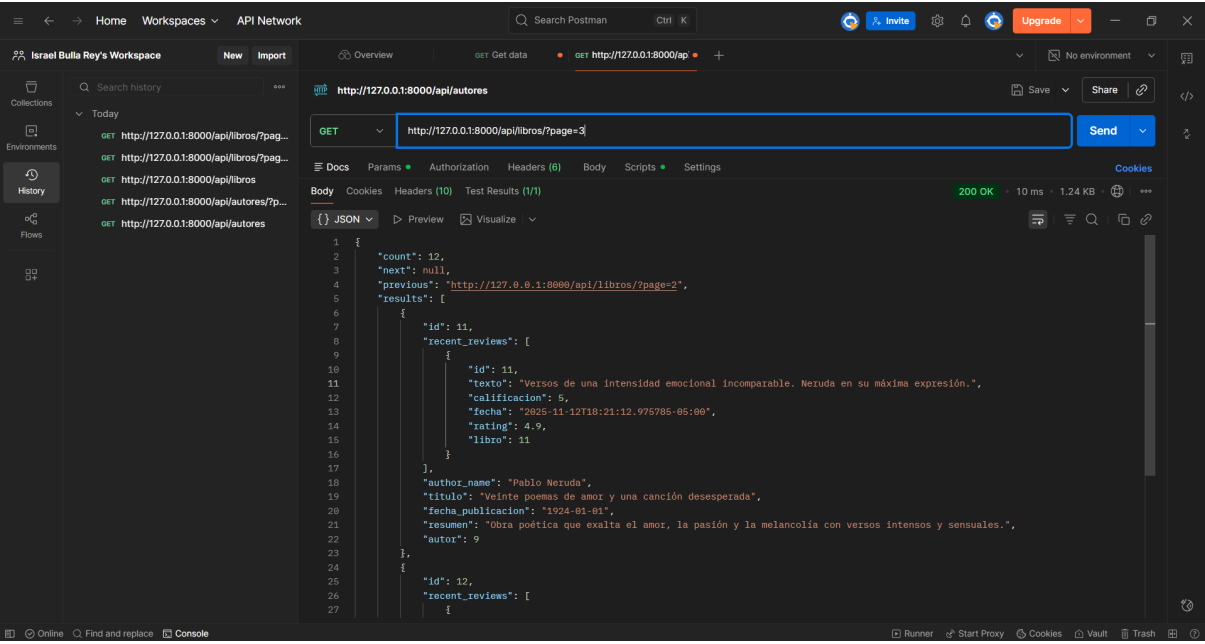
Mostrar autores:

Solo se muestran 5 por página.

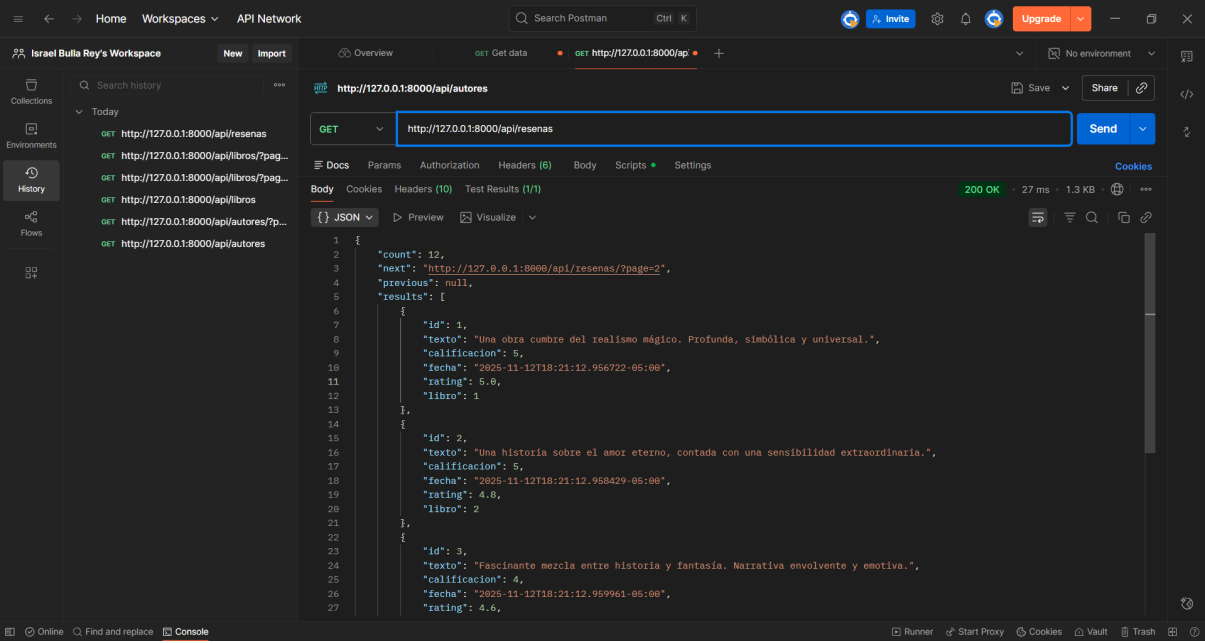


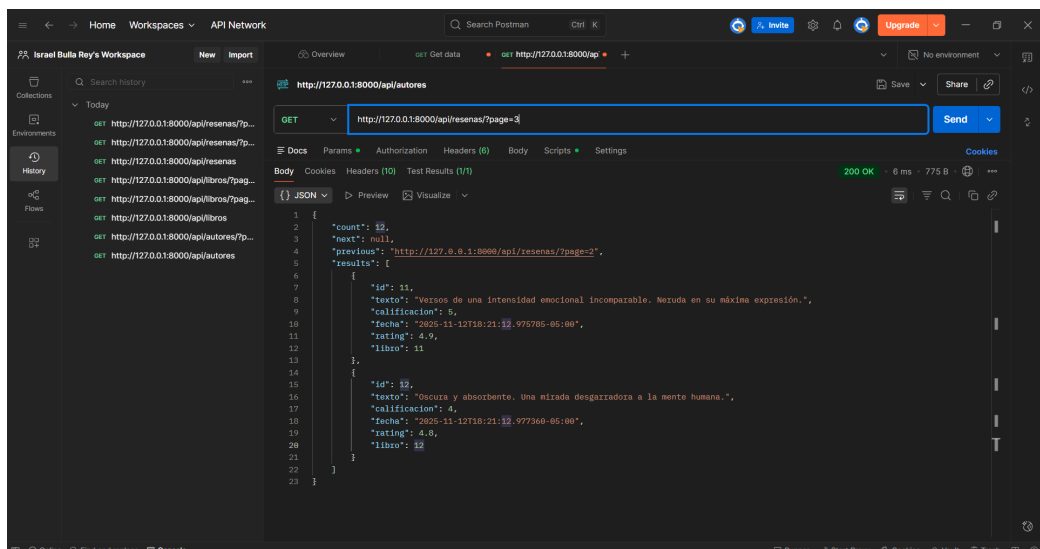
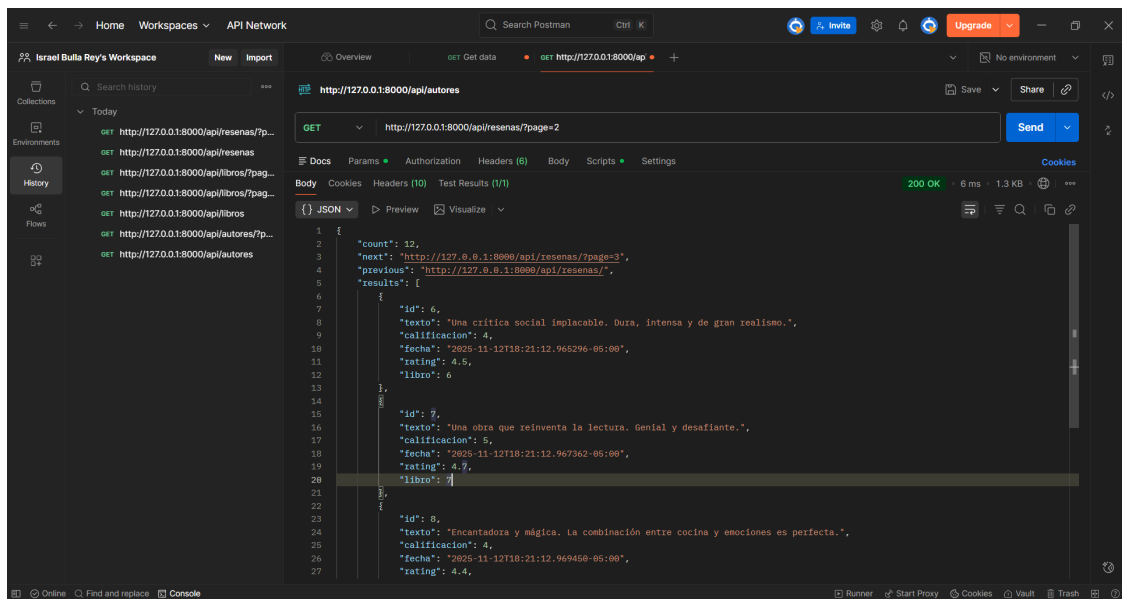
Mostrar libros:



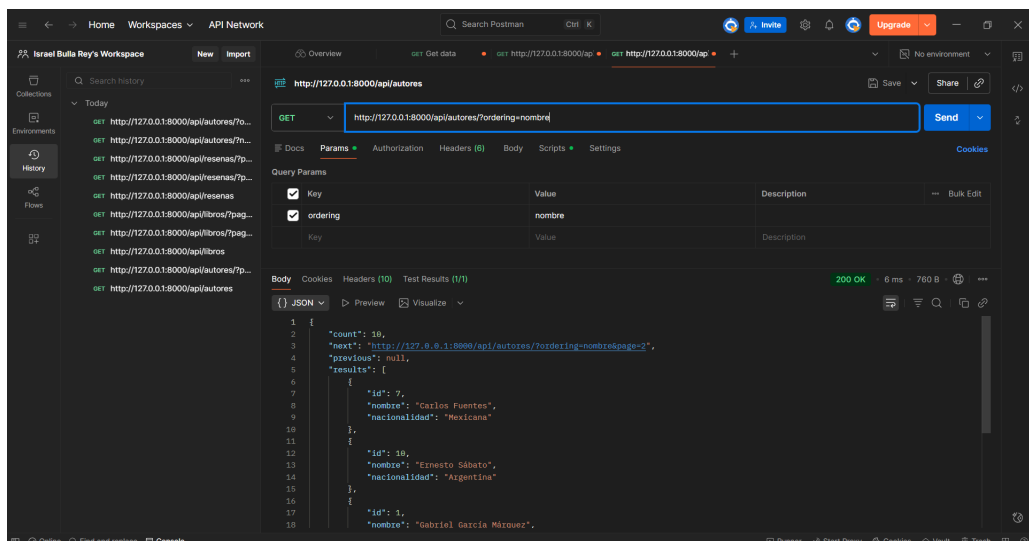


Mostrar reseñas:

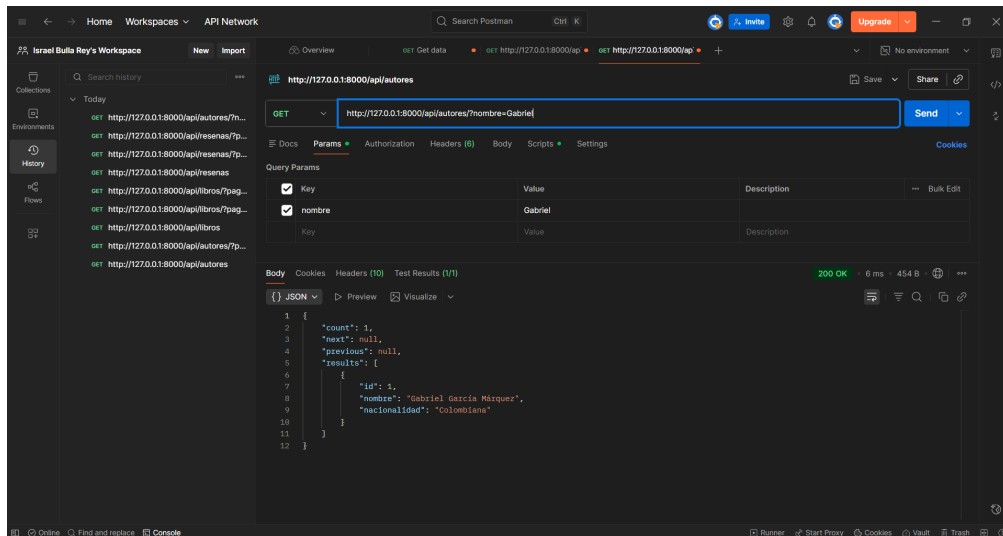




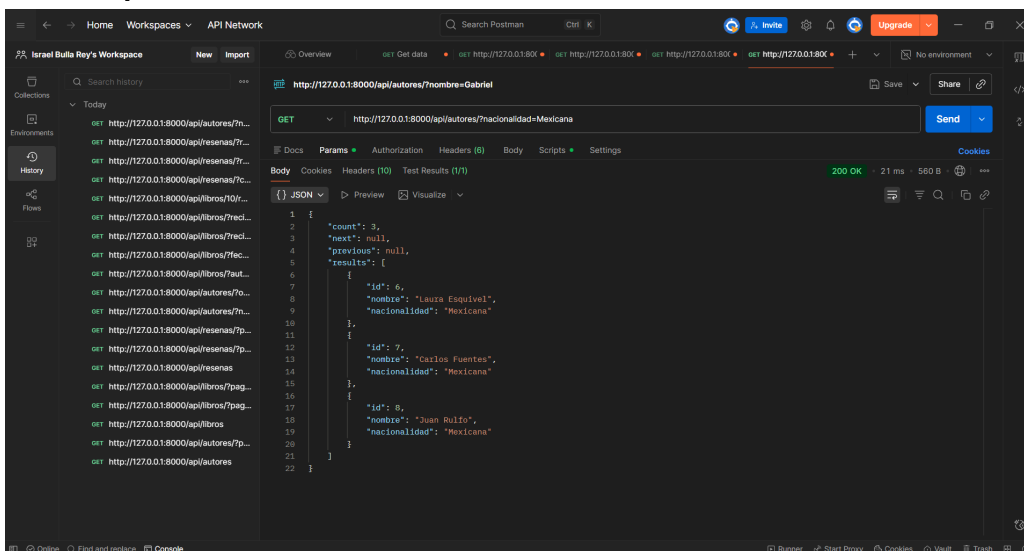
Ordenar por nombre los autores:



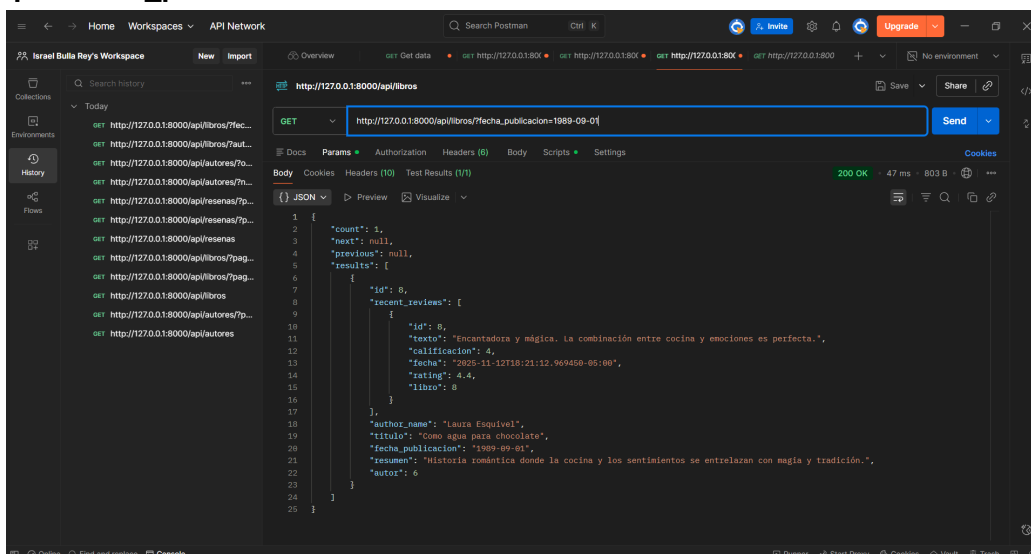
Filtrar autores por nombre:



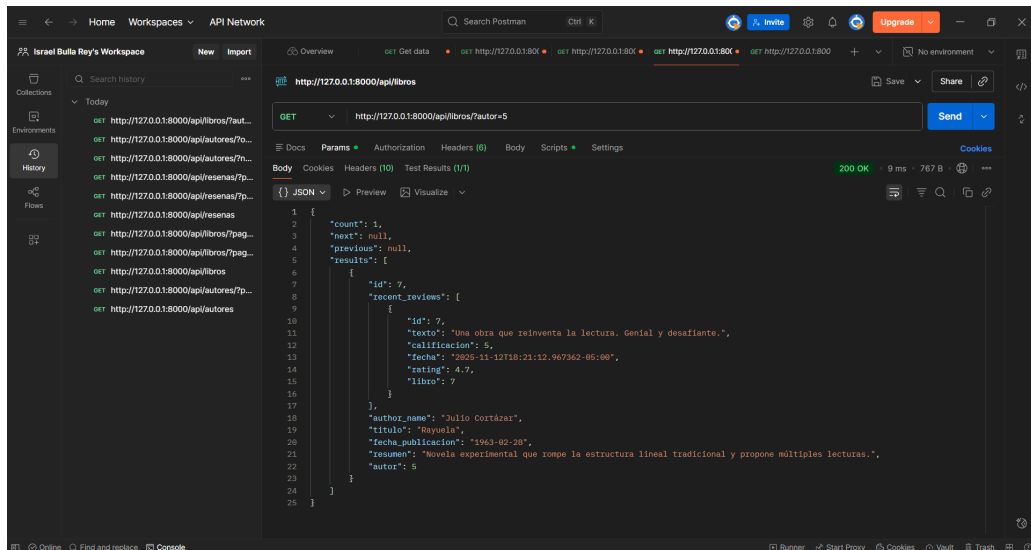
Filtrar autores por su nacionalidad:



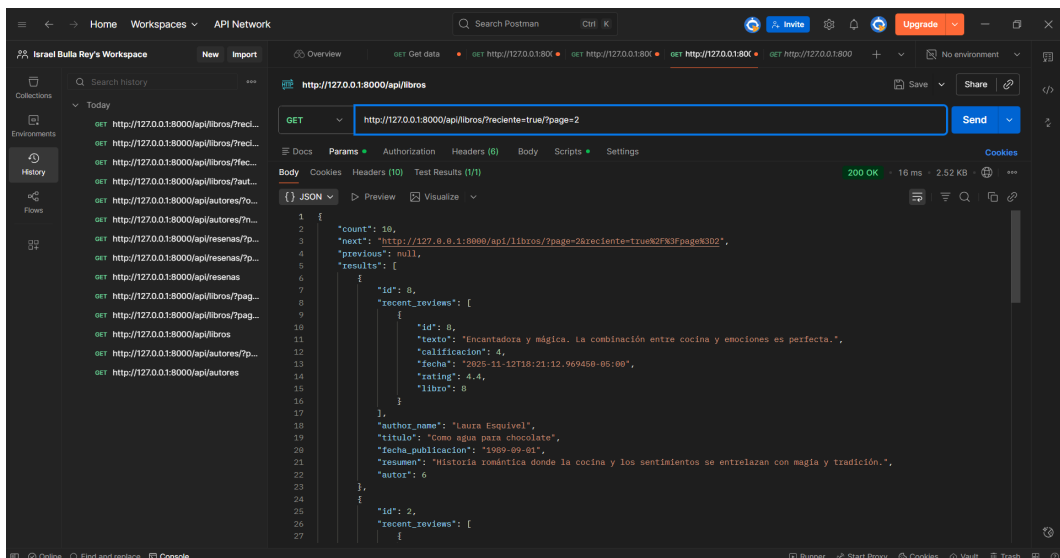
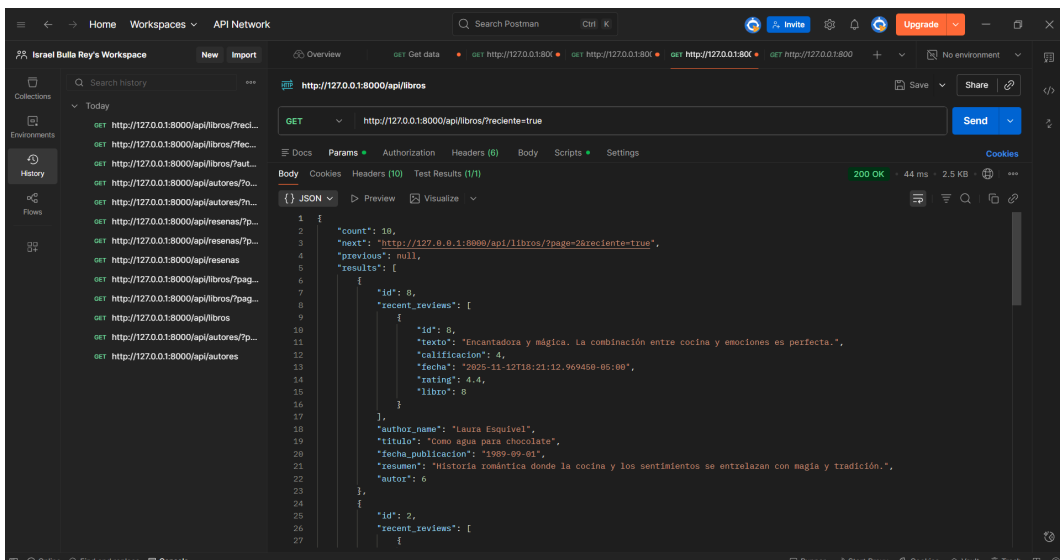
Filtrar por fecha_publicacion los libros:



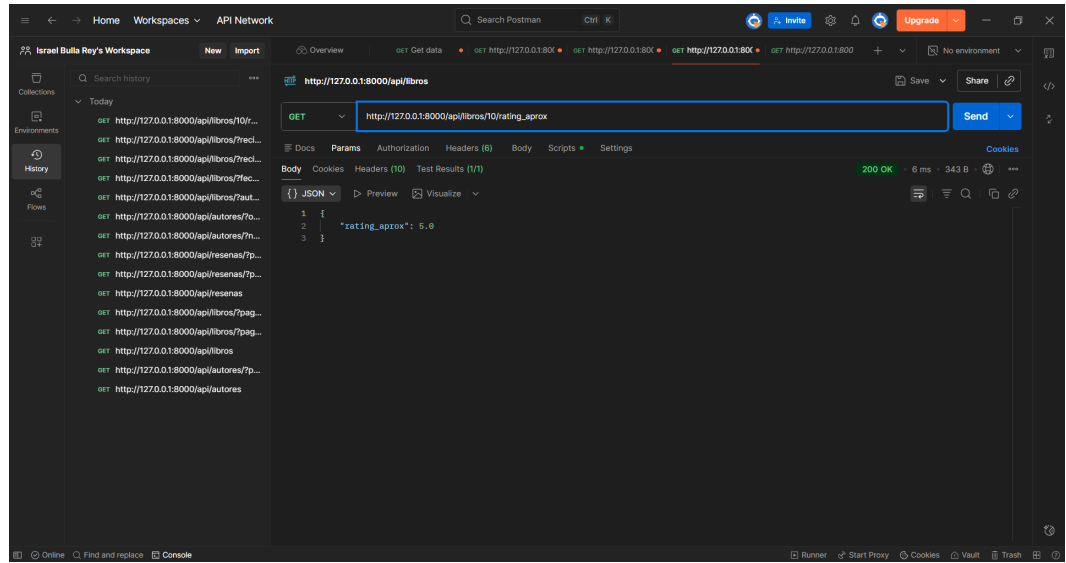
Filtrar libros por el autor (ID)



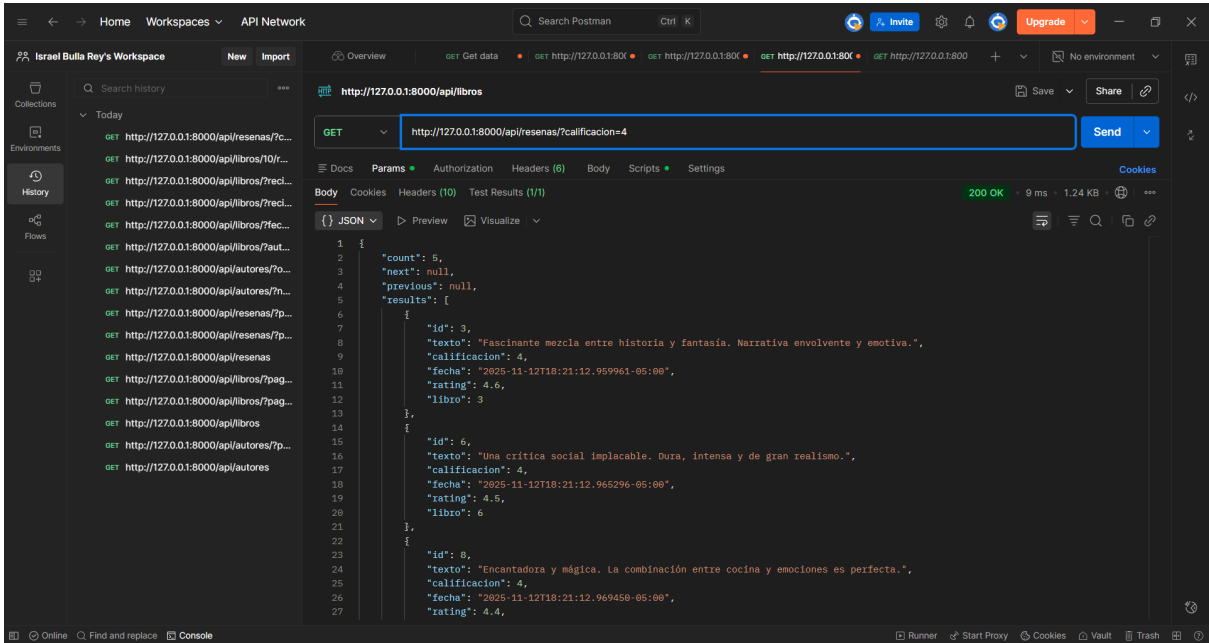
Mostrar los libros más recientes por fecha de publicación:



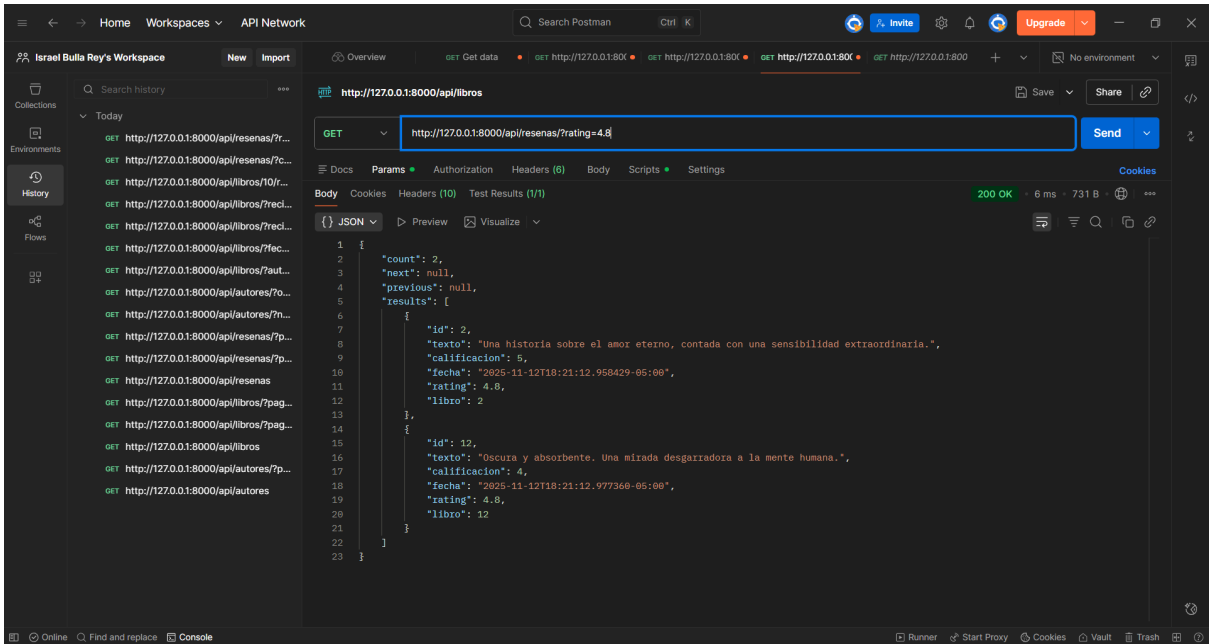
Mostrar el rating_aprox de un libro:



Filtrar las reseñas de libros con calificación 4:



Filtrar las reseñas de libros con rating 4.8:



Mostrar las últimas reseñas de personas:

