Geração de Documentação:

O Django Rest Framework (DRF) pode gerar **documentação automática** da API, semelhante ao que o FastAPI faz. No DRF, a documentação é gerada com base nas **views, serializers, permissões e rotas** já configuradas. Para isso, existem pacotes que trabalham em conjunto com o DRF para criar interfaces amigáveis e interativas.

Aqui estão algumas opções populares:

**1. DRF Browsable API**

* O DRF já possui uma interface básica integrada, chamada **Browsable API**, que pode servir como uma documentação simples.
* A interface é acessada diretamente no navegador e mostra os endpoints disponíveis, os métodos permitidos (GET, POST, etc.) e exemplos de entrada e saída.

**Configuração básica**: Certifique-se de que a BrowsableAPIRenderer está habilitada no DEFAULT\_RENDERER\_CLASSES no arquivo settings.py:

REST\_FRAMEWORK = {

'DEFAULT\_RENDERER\_CLASSES': [

'rest\_framework.renderers.JSONRenderer',

'rest\_framework.renderers.BrowsableAPIRenderer',

],

}

**Limitação**: A interface não é tão detalhada quanto as documentações geradas por pacotes como **Swagger**.

**2. DRF + drf-spectacular (Recomendado)**

* O pacote **drf-spectacular** é uma das melhores opções para gerar documentação automática no DRF. Ele cria documentação no formato **OpenAPI 3.0** (usado pelo Swagger e Redoc).

**Instalação:**

pip install drf-spectacular

**Configuração:**

No settings.py, adicione o gerador de esquemas:

REST\_FRAMEWORK = {

'DEFAULT\_SCHEMA\_CLASS': 'drf\_spectacular.openapi.AutoSchema',

}

**URLs para a documentação:**

No arquivo urls.py, inclua as rotas para acessar a documentação:

from django.urls import path

from drf\_spectacular.views import SpectacularAPIView, SpectacularSwaggerView, SpectacularRedocView

urlpatterns = [

path('api/schema/', SpectacularAPIView.as\_view(), name='schema'), # Esquema OpenAPI

path('api/docs/swagger/', SpectacularSwaggerView.as\_view(url\_name='schema'), name='swagger-ui'), # Swagger UI

path('api/docs/redoc/', SpectacularRedocView.as\_view(url\_name='schema'), name='redoc'), # Redoc

]

**Acessando a documentação:**

* **Esquema OpenAPI (JSON)**: /api/schema/
* **Swagger UI**: /api/docs/swagger/
* **Redoc**: /api/docs/redoc/

**3. DRF + drf-yasg**

Outra opção popular é o **drf-yasg**, que também gera documentação no formato OpenAPI.

**Instalação:**

pip install drf-yasg

**Configuração:**

No arquivo urls.py, configure as rotas para o Swagger e Redoc:

from django.urls import path, re\_path

from rest\_framework import permissions

from drf\_yasg.views import get\_schema\_view

from drf\_yasg import openapi

schema\_view = get\_schema\_view(

openapi.Info(

title="API Documentation",

default\_version='v1',

description="Descrição da sua API",

terms\_of\_service="https://www.google.com/policies/terms/",

contact=openapi.Contact(email="suporte@suaapi.com"),

license=openapi.License(name="MIT License"),

),

public=True,

permission\_classes=[permissions.AllowAny],

)

urlpatterns = [

re\_path(r'^swagger(?P<format>\.json|\.yaml)$', schema\_view.without\_ui(cache\_timeout=0), name='schema-json'),

path('swagger/', schema\_view.with\_ui('swagger', cache\_timeout=0), name='schema-swagger-ui'),

path('redoc/', schema\_view.with\_ui('redoc', cache\_timeout=0), name='schema-redoc'),

]

**Acessando a documentação:**

* **Swagger UI**: /swagger/
* **Redoc**: /redoc/
* **Comparação entre drf-spectacular e drf-yasg**

| **Recurso** | **drf-spectacular** | **drf-yasg** |
| --- | --- | --- |
| **Suporte OpenAPI** | OpenAPI 3.0 | OpenAPI 2.0/3.0 |
| **Configuração mínima** | Sim | Sim |
| **Manutenção ativa** | Sim | Sim, mas menos frequente |
| **Performance** | Melhor para APIs grandes | Bom, mas pode ser mais lento em APIs grandes |

* Para projetos novos, o **drf-spectacular** é geralmente recomendado devido ao suporte nativo para OpenAPI 3.0 e melhor performance.

**Resumo**

* O DRF pode gerar documentação automática da API.
* Use pacotes como **drf-spectacular** ou **drf-yasg** para gerar interfaces interativas no estilo Swagger ou Redoc.
* A documentação gerada facilita o consumo da API e melhora a experiência de desenvolvedores.

Essas ferramentas fazem o DRF se equiparar ao FastAPI na criação de documentações robustas e interativas!