Tutorial Django Inicial – Pt 3
Jordi Virgili Gomà

Índice

Django REST framework es una biblioteca de Python que permite crear APIs RESTful sobre el framework Django.

Paso 1: Instalar Django REST framework

Puede hacerse usando pip, el administrador de paquetes estándar de Python. Simplemente ejecute el siguiente comando (O con el asistente de PyCharm):

pip install `django-rest-framework`

Agrega Django REST framework a INSTALLED_APPS en tu archivo settings.py de Django:

```
INSTALLED_APPS = [
    # ...
    'rest_framework',
]
```

Paso 2: preparar el sistema para aceptar autenticaciones (Oauth2)

Para configurar Django REST framework para usar OAuth2 como método de autenticación instalar las siguientes dependencias:

```
pip install django-oauth-toolkit django-rest-framework
```

Configurar las librerias para nuestro proyecto implica agregar las siguientes líneas en el archivo settings.py:

```
INSTALLED_APPS = [
    #...
    'oauth2_provider',
    'rest_framework',
]

# Configuración de OAuth2
AUTHENTICATION_BACKENDS = (
    'oauth2_provider.backends.OAuth2Backend',
    'django.contrib.auth.backends.ModelBackend',
)

REST_FRAMEWORK = {
    # Use Django's standard django.contrib.auth permissions,
    # or allow read-only access for unauthenticated users.
    'DEFAULT_SCHEMA_CLASS': 'rest_framework.schemas.coreapi.AutoSchema',

    'DEFAULT_PERMISSION_CLASSES': [
        'rest_framework.permissions.DjangoModelPermissionsOrAnonReadOnly',
        'rest_framework.permissions.IsAuthenticated',
    ],
    'DEFAULT_AUTHENTICATION_CLASSES': (
```

```
        'oauth2_provider.contrib.rest_framework.OAuth2Authentication',
        'rest_framework.authentication.SessionAuthentication',
    ),
}

OAUTH2_PROVIDER = {
    'ACCESS_TOKEN_EXPIRE_SECONDS': 60 * 60 * 24,
    'REFRESH_TOKEN_EXPIRE_SECONDS': 60 * 60 * 24 * 30,
    # this is the list of available scopes
    'SCOPES': {'read': 'Read scope', 'write': 'Write scope', 'groups': 'Access to
your groups'}
}
```

Recordemos que, al instalar nuevas APPS, se debe realizar un makemigrations i migrate para propagar los cambios a la base de datos.

Luego se deben configurar las vistas que manejan las solicitudes de autorización y acceso. Para ello, agregar las siguientes líneas al archivo urls.py:

```
from django.urls import path
from oauth2_provider.views import TokenView, AuthorizationView

urlpatterns = [
    path('api/token/', TokenView.as_view(), name='token'),
    path('o/', include('oauth2_provider.urls', namespace='oauth2_provider'))


]
```

Tener en cuenta que, se deben configurar las vistas de la API RESTful para que utilicen OAuth2 como método de autenticación. Para ello:

```
from rest_framework.authentication import SessionAuthentication,
BasicAuthentication
from rest_framework.permissions import IsAuthenticated

class MiVistaXYZ(APIView):
    authentication_classes = [SessionAuthentication, BasicAuthentication]
    permission_classes = [IsAuthenticated]
```

O en su defecto podemos utilizar anotaciones con el estilo:

```
@api_view(['GET', 'POST'])

@permission_classes([IsAuthenticated])
```

Paso 3: Registrar aplicación

Para demostrar que somos dignos (es coña, es para demostrar que quien te valida la identidad está autorizado), necesitamos registrar la aplicación. Eso se puede hacer con el asistente de la url

‘http://127.0.0.1:8000/o/applications/register/’ dónde crearemos una relación de tipo Confidential y owner-password based. Hemos de anotar el client_id y secret para usarlo en las peticiones posteriores (ANTES DEL HASH FINAL).

Paso 4: Controlador información HTTP (sin token)

Un serializador en Django REST framework es una clase que convierte un objeto en formato Python (como una instancia de un modelo de Django) en un formato que se puede enviar a través de la http (como JSON o XML).

1. Para crear un serializador, cree un archivo serializers.py en su aplicación y defina una clase de serializador que herede de la clase Serializer de Django REST framework. Por ejemplo:

```
from rest_framework import serializers
from .models import Libro

class LibroSerializer(serializers.ModelSerializer):
    class Meta:
        model = Libro
        fields = '__all__'
```

Este serializador de ejemplo convierte una instancia de la clase Libro en un formato JSON que incluye todos los campos del modelo (especificados por el atributo fields de la clase Meta).

2. Crear vistas para la API:

En Django REST framework, una vista es una función que toma una solicitud HTTP y devuelve una respuesta HTTP (habitualmente JSON). Para crear una vista, define una función en views.py que devuelva una respuesta a partir de una solicitud. Por ejemplo:

```
from rest_framework import generics
from .serializers import LibroSerializer
from .models import Libro

class APILibroList(generics.ListCreateAPIView):
    queryset = Libro.objects.all()
    serializer_class = LibroSerializer
```

Este ejemplo define una vista de lista/creación que devuelve todas las instancias de Libro y permite a su vez crear nuevas instancias. La vista usa el serializador LibroSerializer para convertir las instancias en formato JSON.

3. Configurar las URL correspondientes a la API:

En Django, las URL se definen en el archivo urls.py de la aplicación. Para agregar las URL de la API, importar as vistas y agregar patrones de URL al archivo urls.py. Por ejemplo:

```
from django.urls import path
from .views import APILibroList

urlpatterns = [
    path('api/librolist/', APILibroList.as_view(), name='api-list'),
]
```

Este ejemplo agrega un patrón de URL para la vista APILibroList que se puede acceder en la ruta /api/librolist/. El nombre api-list' es opcional y se puede usar para hacer referencia a esta URL en otras partes de la aplicación.

Paso 5: pedir token

Para obtener un token nuevo utilizando OAuth2 y curl, se envia una solicitud POST a la URL de la vista TokenView configurada en el proyecto Django:

```
curl -X POST -d
"grant_type=password&username=<username>&password=<password>&client_id=<client_id>
&client_secret=<client_secret>" http://localhost:8000/o/token/
```

Esto se puede convertir a cualquier lenguaje usando Postman.