Tutorial: Integración de Celery y RabbitMQ en el Proyecto Django 'Compushop'

1. Instalación de WSL

Para ejecutar Docker Desktop en Windows es necesario tener WSL2 actualizado: - Abre PowerShell como administrador. - Ejecuta: **wsl --update** - Verifica que tu distribución esté correctamente instalada: **wsl --list --verbose** Si da error, considera reinstalar la distribución (como Ubuntu).

2. Instalación de Docker Desktop

1. Descarga Docker desde: https://www.docker.com/products/docker-desktop/ 2. Instálalo y asegúrate de activar la opción “Usar WSL2”. 3. Verifica instalación:
```
docker version
```

3. Ejecución de RabbitMQ en Docker

Usamos la imagen oficial con la interfaz de administración incluida:
```
docker run -d --hostname rabbit-host --name rabbitmq \ -p 5672:5672 -p 15672:15672 rabbitmq:3.13.1-management
```

- 5672 es el puerto del protocolo AMQP usado por Celery. - 15672 es para acceder al panel web en http://localhost:15672 (usuario: guest / contraseña: guest).

4. Configuración de Celery en Django

1. Instala Celery con pip:
```
pip install celery
```

2. Estructura de archivos usada en el proyecto:
```
compushop/ ■■■ compushop/ ■ ■■■ __init__.py # Importa celery_app ■ ■■■ celery.py # Configura Celery ■ ■■■ settings.py # Configura broker ■■■ ordenes/ ■ ■■■ tasks.py # Define tareas Celery ...
```

5. Código clave de integración

Archivo celery.py:
```
from celery import Celery import os os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'compushop.settings') app = Celery('compushop')
app.config_from_object('django.conf:settings', namespace='CELERY')
app.autodiscover_tasks()
```

Archivo __init__.py de compushop:
```
from .celery import app as celery_app __all__ = ['celery_app']
```

Archivo settings.py (añadir):
```
CELERY_BROKER_URL = 'amqp://localhost'
```

Archivo tasks.py en la app 'ordenes':
```
from celery import shared_task from .models import Orden @shared_task def task_orden_creada(orden_id): orden = Orden.objects.get(id=orden_id) print(f'[*] Orden procesada: {orden}')
```

6. Ejecución del worker de Celery

Ejecutar desde la raíz del proyecto:
```
celery -A compushop worker --loglevel=info
```

Este comando lanza el worker que escucha tareas enviadas desde Django.

7. Disparo de la tarea

Desde la vista donde se crea la orden:
```
task_orden_creada.delay(orden.id)
```

Esto envía la tarea al worker de Celery mediante RabbitMQ.

¡Listo! Ahora tienes una integración completa entre Celery, RabbitMQ, Docker y Django.