Despliegue api rest laravel 10

Tabla de contenidos

1 Despliegue de API REST en Laravel 10 con Sail en Producción

Vamos a desplegar tu API REST en Laravel 10 con Docker y Sail en un servidor de producción. Te explicaré cada paso detalladamente.

1.1 Configuración del Servidor

Necesitas un VPS o servidor dedicado con un sistema operativo Linux (ej. Ubuntu 22.04). Lo primero es actualizarlo y preparar el entorno.

En el Servidor (host)

Conéctate vía SSH:

```
ssh usuario@tu-servidor
```

Actualiza el sistema:

```
sudo apt update && sudo apt upgrade -y
```

Instala los paquetes básicos:

```
sudo apt install -y curl git unzip ufw
```

1.2 Instalar Docker y Docker Compose

Sail funciona sobre Docker, así que necesitas instalarlo en el servidor.

En el Servidor (host)

Ejecuta:

```
sudo apt install -y docker.io docker-compose
sudo systemctl enable --now docker
```

Verifica que Docker funciona:

```
docker --version
docker-compose --version
```

Añade tu usuario al grupo Docker (opcional, para no usar 'sudo' en cada comando):

```
sudo usermod -aG docker $(whoami)
```

(Sal de la sesión y vuelve a entrar para que surta efecto)

1.3 Subir el Código de la API

Tienes dos opciones: subir el código desde tu máquina o clonar el repositorio en el servidor.

Opción 1: Subir desde tu máquina

En tu máquina local:

```
scp -r tu-api usuario@tu-servidor:/var/www/
```

Opción 2: Clonar directamente en el servidor

En el servidor:

```
cd /var/www/
git clone https://github.com/tu-usuario/tu-api.git
```

Asegúrate de que los permisos sean correctos:

```
sudo chown -R $USER:www-data /var/www/tu-api
sudo chmod -R 775 /var/www/tu-api
```

1.4 Configurar Laravel y Entorno

Dentro del Proyecto en el Servidor

Accede a la carpeta del proyecto:

```
cd /var/www/tu-api
```

Copia el archivo de entorno y edítalo:

```
cp .env.example .env
nano .env
```

Cambia las siguientes variables:

```
APP_ENV=production
APP_DEBUG=false
APP_URL=https://tu-dominio.com

DB_CONNECTION=mysql
DB_HOST=mysql
DB_PORT=3306
DB_DATABASE=nombre_bd
DB_USERNAME=usuario_bd
DB_PASSWORD=clave_secreta
```

1.5 Configurar Sail para Producción

(En '/var/www/tu-api' en el servidor)

Levanta los contenedores:

```
./vendor/bin/sail up -d
```

Ejecuta las migraciones:

```
./vendor/bin/sail artisan migrate --force
```

Genera la clave de la aplicación:

```
./vendor/bin/sail artisan key:generate
```

1.6 Optimización de Laravel

Dentro del contenedor de Laravel:

```
./vendor/bin/sail artisan config:cache
./vendor/bin/sail artisan route:cache
./vendor/bin/sail artisan view:cache
```

1.7 Configurar Nginx como Proxy Reverso

Sail ya tiene Nginx dentro del contenedor, pero necesitas un **proxy inverso** para que la API sea accesible desde el exterior.

En el Servidor (host)

Instala Nginx:

```
sudo apt install -y nginx
```

Crea una configuración para Laravel:

```
sudo nano /etc/nginx/sites-available/tu-api
```

Pega este contenido:

```
server {
    listen 80;
    server_name tu-dominio.com;

    location / {
        proxy_pass http://127.0.0.1:80;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }
}
```

Activa el sitio y reinicia Nginx:

```
sudo ln -s /etc/nginx/sites-available/tu-api /etc/nginx/sites-enabled/
sudo systemctl restart nginx
```

1.8 Configurar HTTPS con Let's Encrypt

Ejecuta:

```
sudo apt install certbot python3-certbot-nginx
sudo certbot --nginx -d tu-dominio.com
```

Esto generará y configurará automáticamente el certificado SSL.

1.9 Configurar Supervisor para las Colas (Opcional)

Si tu aplicación usa colas, instala 'supervisor' y configúralo.

En el Servidor (host)

```
sudo apt install -y supervisor
```

Crea el archivo:

```
sudo nano /etc/supervisor/conf.d/laravel-worker.conf
```

Agrega esto:

```
[program:laravel-worker]
process_name=%(program_name)s_%(process_num)02d
command=docker exec -i tu-api-app sail artisan queue:work --tries=3
autostart=true
autorestart=true
numprocs=1
redirect_stderr=true
stdout_logfile=/var/log/laravel-worker.log
```

Carga la configuración:

```
sudo supervisorctl reread
sudo supervisorctl update
sudo supervisorctl start laravel-worker
```

1.10 Seguridad y Firewall

Habilita el firewall y solo permite SSH, HTTP y HTTPS:

```
sudo ufw allow OpenSSH
sudo ufw allow 80
sudo ufw allow 443
sudo ufw enable
```

1.11 Últimos Pasos y Pruebas

En el Servidor (host)

Verifica que todo está corriendo:

```
docker ps
```

Si ves los contenedores en ejecución, ¡vas bien!

Prueba que la API responde:

```
curl -I https://tu-dominio.com/api/endpoint
```

Si necesitas ver logs:

```
./vendor/bin/sail artisan logs
```

1.12 Resumen Final

- **Servidor**: Instalamos Docker, Docker Compose y Nginx.
- **Proyecto**: Subimos Laravel y configuramos ‘.env‘.
- **Sail**: Levantamos los contenedores (‘sail up -d‘).
- **Nginx**: Configuramos un proxy inverso.
- **SSL**: Instalamos Let’s Encrypt.
- **Supervisor**: Configuramos workers si usamos colas.
- **Seguridad**: Configuramos firewall y permisos.

¡Tu API REST en Laravel 10 con Sail ya está en producción!