Laravel

Cosas de Laravel y los paquetes asociados a este ecosistema

Paquetes

Tips

Tips

Laravel Horizon + Supervisor en MacOs con Laravel Herd + Brew

Escenario

El indicado en el título, fue probado con Laravel 12 y para una app en la que se necesita trabajar de forma constante desde un Mac.

> Yo para emular ciertas cosas, lo uso en desarrollo, pero cuando termino, desactivo las tareas mmediante configuración (`.env`)

Instalar Supervisor

Necesitamos instalar **Supervisor**

```
brew install supervisor
```

En Mac con brew, veremos que el fichero de configuración es `/opt/homebrew/etc/supervisord.conf` el cual, al final hace un include `files = /opt/homebrew/etc/supervisor.d/*.ini`

Dicho esto, tenemos claro que el fichero de configuración para nuestro despliegue (o los diferentes despliegues) sera incluido en ese directorio, como por ejemplo para una de mis apps, `/opt/homebrew/etc/supervisor.d/swissknife.ini`

Ejemplo básico para una app sencilla, teniendo en cuenta que quiero especificar indubitadamente la versión de php

```
[program:swissknife]
process_name=%(program_name)s
command="/Users/abkrim/Library/Application Support/Herd/bin/php"
/Users/abkrim/SitesLR12/swissknife/artisan horizon
autostart=true
autorestart=true
user=abkrim
```

```
numprocs=1
redirect_stderr=true
stdout_logfile=/Users/abkrim/SitesLR12/swissknife/storage/logs/horizon.log
stderr_logfile=/Users/abkrim/SitesLR12/swissknife/storage/logs/horizon.err.log
stopwaitsecs=3600
```

Para iniciarlo y activarlo al reinicio

```
brew services start supervisor
```

Debug de problemas

A veces, puede que hagamos cambios en el fichero de configuración y no consigamos que funcione.

Para ello paramos el servicio de brew y lo cargamos de forma manual

```
brew services stop supervisor
supervisord -n -c  /opt/homebrew/etc/supervisord.conf

Error: %(process_num) must be present within process_name when numprocs > 1 in section
'program:swissknife' (file: '/opt/homebrew/etc/supervisor.d/swissknife.ini')
For help, use /opt/homebrew/bin/supervisord -h
```

Mira, un error de configuración que deja el servicio de nuestra app, sin cargarse.

Con eso podemos corregir el tema, y volver a cargar de forma automática nuestro supervisor.

```
supervisord -n -c  /opt/homebrew/etc/supervisord.conf

2025-02-28 12:20:38,341 WARN For [program:swissknife], redirect_stderr=true but stderr_logfile
has also been set to a filename, the filename has been ignored
2025-02-28 12:20:38,342 INFO Included extra file
"/opt/homebrew/etc/supervisor.d/swissknife.ini" during parsing
2025-02-28 12:20:38,353 INFO RPC interface 'supervisor' initialized
2025-02-28 12:20:38,353 CRIT Server 'unix_http_server' running without any HTTP authentication
checking
2025-02-28 12:20:38,353 INFO supervisord started with pid 65213
2025-02-28 12:20:39,361 INFO spawned: 'swissknife' with pid 65218
2025-02-28 12:20:40,735 INFO success: swissknife entered RUNNING state, process has stayed up
for > than 1 seconds (startsecs)
```

Matamos el proceso, y ya podemos reiniciar con brew.

```
brew services start supervisor
==> Successfully started `supervisor` (label: homebrew.mxcl.supervisor)
```

Y a disfrutar de Horizon+Laravel

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Tips

Media Library Filament de Ralph J. Smit + Spatie Media Library + Livewire

Introducción

A menudo, es más efectivo seguir un enfoque lógico al resolver problemas, pero en ocasiones nos encontramos con obstáculos. En esos momentos, recurrimos a Internet en busca de respuestas. También consultamos a inteligencias artificiales como ChatGPT, Claude, Gemini o DeepSeek, pero si no encuentran soluciones exactas, pueden llevarnos por caminos erróneos, generando respuestas confusas.

En mi caso, persistí en resolver un problema y me di cuenta de que me faltaba un elemento por revisar: Livewire.

Error de tamaño en la subida de imágenes

En mi aplicación de desarrollo, comencé a utilizar Media Library Filament, que se basa en Spatie Media Library.

Dado que manejo archivos grandes de imágenes, lo primero que hice fue buscar los límites de tamaño permitidos.

Lo habitual es verificar:

- PHP (*php.ini*)
- Media Library (*config/media-library.php*)

Sin embargo, a pesar de haber configurado un límite de 30 MB en todas las configuraciones, continuaba recibiendo el siguiente error:

```
The uploads.2f138bcc-28d3-40b2-a941-a516d750c1d7 field must not be greater than 12288 kilobytes.
```

Ralphjsmit Media Library Manager Livewire Issue

Era evidente que el problema era de configuración, pero ¿dónde estaba el error?

Todas las respuestas de inteligencia artificial y los tutoriales de Laracast no ofrecían nada nuevo, ya que solo repetían lo que ya había verificado:

En el *php.ini*, teníamos `upload_max_filesize=64M` y en *config/media-library.php* `'max_file_size' => 1024 * 1024 * 30, // 30MB`.

Sin embargo, a menudo olvido que muchos paquetes dependen de otros, como Filepond y Livewire... ¿Oh, mencioné Livewire?

Es fundamental revisar su configuración, que efectivamente existe. Puedes publicarla usando el siguiente comando:

```
php artisan livewire:publish --config
```

En la configuración, encontramos lo siguiente:

```
'temporary_file_upload' => [
    'disk' => null,        // Ejemplo: 'local', 's3'              | Predeterminado: 'default'
    'rules' => null,       // Ejemplo: ['file', 'mimes:png,jpg']  | Predeterminado:
['required', 'file', 'max:12288'] (12MB)
    'directory' => null,   // Ejemplo: 'tmp'                      | Predeterminado: 'livewire-
tmp'
    'middleware' => null,  // Ejemplo: 'throttle:5,1'             | Predeterminado:
'throttle:60,1'
    'preview_mimes' => [   // Tipos de archivos admitidos para URLs pre-firmadas temporales...
        'png', 'gif', 'bmp', 'svg', 'wav', 'mp4',
        'mov', 'avi', 'wmv', 'mp3', 'm4a',
        'jpg', 'jpeg', 'mpga', 'webp', 'wma',
    ],
    'max_upload_time' => 5, // Duración máxima (en minutos) antes de que una carga sea
invalidada...
    'cleanup' => true,      // ¿Debería limpiar cargas temporales más antiguas de 24 horas?
],
```

Me había pasado por alto revisar la configuración de Livewire.

Para corregir el límite de tamaño de carga, puedes modificar la regla correspondiente:

```
'rules' => 'file|max:30720', // Ejemplo: ['file', 'mimes:png,jpg']  | Predeterminado:
['required', 'file', 'max:12288'] (12MB)
```

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Tips

Laravel Boost - Problemas de Rendimiento y Seguridad

Problema

Laravel Boost es una herramienta MCP (Model Context Protocol) extremadamente útil para desarrollo asistido por IA, pero puede causar problemas graves si no se configura correctamente:

Problemas en Desarrollo

Rendimiento degradado

- Múltiples instancias del proceso `php artisan boost:mcp` ejecutándose simultáneamente
- El Browser Logs Watcher genera logs masivos (>1MB por llamada)
- Bloquea peticiones web causando timeouts
- Síntomas: páginas que tardan 30+ segundos en cargar o errores 502 de nginx

Consumo excesivo de recursos

- Procesos PHP acumulándose en memoria
- Logs creciendo sin control
- Sistema ralentizado general

Problemas en Producción (CRÍTICOS)

Exposición de información sensible

Laravel Boost expone información que NO debe estar disponible en producción:

- Credenciales de base de datos
- Tokens y secrets del archivo .env
- Estructura completa de la base de datos
- Rutas y configuración de la aplicación
- Logs con información sensible
- Permite ejecutar queries SQL arbitrarias

Riesgo de seguridad

Si Laravel Boost está activo en producción, un atacante podría:

- Leer variables de entorno (DB_PASSWORD, API_KEYS, etc.)
- Explorar la estructura de la base de datos
- Ejecutar consultas de solo lectura
- Acceder a logs con información de usuarios

Solución

1. Publicar Configuración

```
php artisan vendor:publish --tag=boost-config
```

Esto crea `config/boost.php` con opciones configurables.

2. Configurar Variables de Entorno

Añadir al archivo `.env`:

```
# Laravel Boost Configuration
# CRITICAL: MUST be false in production to prevent credential exposure
BOOST_ENABLED=true
BOOST_BROWSER_LOGS_WATCHER=false
```

Importante:

- `BOOST_ENABLED`: Activa/desactiva TODO Laravel Boost
- `BOOST_BROWSER_LOGS_WATCHER`: Monitoreo del navegador (causa problemas de rendimiento)

3. Modificar config/boost.php

```
<?php

return [
    // Se desactiva AUTOMÁTICAMENTE en producción
    'enabled' => env('BOOST_ENABLED', env('APP_ENV') !== 'production'),
```

```
    // Browser logs watcher DESACTIVADO por defecto
    'browser_logs_watcher' => env('BOOST_BROWSER_LOGS_WATCHER', false),
];
```

IMPORTANTE - Error Común:

NO uses `app()->isProduction()` en archivos de configuración:

```
// ❌ INCORRECTO - Causará error "Target class [env] does not exist"
'enabled' => env('BOOST_ENABLED', ! app()->isProduction()),

// ✅ CORRECTO - Usa solo env()
'enabled' => env('BOOST_ENABLED', env('APP_ENV') !== 'production'),
```

Razón: Los archivos de configuración se cargan ANTES de que la aplicación esté completamente inicializada. Solo puedes usar `env()` en estos archivos.

Esta configuración proporciona:

- **Fail-safe**: Se desactiva automáticamente si APP_ENV=production
- **Rendimiento**: Browser logs watcher desactivado por defecto
- **Seguridad**: Múltiples capas de protección

4. Configuración en Producción

En el servidor de producción, el `.env` DEBE tener:

```
APP_ENV=production
BOOST_ENABLED=false
```

NUNCA dejar Laravel Boost activo en producción.

Diagnóstico de Problemas

Si experimentas lentitud o errores 502:

1. Verificar procesos de Boost

```
ps aux | grep boost
```

Si ves múltiples instancias de `php artisan boost:mcp`, hay un problema.

2. Detener procesos problemáticos

```
pkill -f "boost:mcp"
pkill -f "boost:execute-tool"
```

3. Limpiar cache

```
php artisan cache:clear
php artisan config:clear
php artisan optimize:clear
```

4. Reiniciar servicios

Con Laravel Herd:

```
herd restart
```

Con otros entornos:

```
# Reiniciar PHP-FPM y nginx según tu configuración
```

Uso Recomendado

Desarrollo Normal

```
BOOST_ENABLED=true
BOOST_BROWSER_LOGS_WATCHER=false  # Mantener desactivado
```

Solo activa el browser logs watcher cuando necesites debug específico de frontend.

Debug Activo de Frontend

Si necesitas diagnosticar errores de JavaScript/navegador:

```
# Temporal - solo mientras debuggeas
BOOST_BROWSER_LOGS_WATCHER=true php artisan serve
```

Recuerda volver a `false` cuando termines.

Producción

```
APP_ENV=production
BOOST_ENABLED=false
```

Sin excepciones.

Beneficios de la Configuración Segura

- **Rendimiento**: Mejora de hasta 97% en tiempo de respuesta (de 30s a <1s)
- **Seguridad**: Protección automática contra exposición de credenciales
- **Estabilidad**: Previene acumulación de procesos y logs masivos
- **Fail-safe**: Desactivación automática en producción aunque olvides configurarlo

Referencias

- Laravel Boost Documentation
- Configuración recomendada basada en incidentes reales en entornos de desarrollo y producción

---

Última actualización: 2026-01-03

Tips

Skill básico para vigilancia y calidad de código

Este es un básico de los que uso para mis proyectos Laravel. Los hay mas duros, pero es em e vale

composer.json

```
"scripts": {
"test": [
    "@php artisan config:clear --ansi",
    "@php artisan test"
  ],
  "test:coverage": "vendor/bin/pest --coverage --coverage-clover=coverage.xml",
  "analyse": "vendor/bin/phpstan analyse",
  "insights": "vendor/bin/phpinsights --no-interaction",
  "lint": "vendor/bin/pint",
  "lint:check": "vendor/bin/pint --test",
  "check": "composer lint && composer test && composer analyse",
  "check-full": "composer lint && composer test && composer analyse && composer insights &&
composer coverage"
}
```

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.