ŚCIĄGA LARAVEL

Deployer dla Laravela

Jeden build w CI, smoke test kandydata, jeden rollback bez człowieka. Deployer 8, jedna VM, Nginx + PHP-FPM.

TL;DR

- Deployer zostaje od katalogów release'ów, plików współdzielonych, blokady, atomowego przełączenia i sprzątania.
- Na serwer leci **tarball zbudowany w CI**, sprawdzany przez SHA-256. Bez Gita, Composera i Node'a na serwerze.
- Migracje tylko z `DEPLOY_ALLOW_MIGRATIONS=true` i tylko addytywne.
- Smoke test nowego release'u **przed** `deploy:symlink`, readiness **po** nim.
- Readiness po przełączeniu nie przechodzi: **jeden** automatyczny rollback, potem deploy kończy się błędem.

Co robi standardowa recepta

`recipe/laravel.php`, Deployer 8 (v8.0.5, stan na 2026-10-27).

```
deploy:prepare       # info, setup, lock, release, update_code,
                     # env, shared, writable
deploy:vendors       # composer install na serwerze
artisan:storage:link
artisan:optimize
artisan:migrate      # przy każdym deployu, bezwarunkowo
deploy:publish       # symlink, unlock, cleanup, success
artisan:reload
```

Ponowna definicja taska o tej samej nazwie podmienia jego ciało. Zostaw grupy, podmień `deploy:update_code` i `deploy:vendors`, napisz własną listę `deploy`.

Katalogi i ustawienia

```
/var/www/starter/
├── .dep/                  # blokada, log release'ów, pliki stanu
├── current -> releases/42
├── releases/  38/ … 42/
└── shared/    .env  storage/
```

```
set('keep_releases', 5);
set('shared_files', ['.env']);
set('shared_dirs', ['storage']);
set('writable_dirs', ['bootstrap/cache', 'storage']);
set('writable_mode', 'acl');
```

Host, użytkownik, ścieżka, artefakt i jego SHA-256 przychodzą ze zmiennych środowiskowych (`DEPLOY_HOST` , `DEPLOY_USER` , `DEPLOY_PATH` , `DEPLOY_ARTIFACT` , `DEPLOY_ARTIFACT_SHA256`). Brak którejś = recepta nie wystartuje.

Upload artefaktu

```
task('deploy:update_code', function (): void {
    $artifactPath = get('artifact_path');

    if (! is_file($artifactPath)) {
        throw new \RuntimeException(sprintf('Brak artefaktu release: %s.', $artifactPath));
    }

    upload($artifactPath, '{{release_path}}/release.tar.gz');
    run('test "$(sha256sum {{release_path}}/release.tar.gz | cut -d " " -f 1)" = "
{{artifact_sha256}}"');
    run('tar -xzf {{release_path}}/release.tar.gz -C {{release_path}} --no-same-owner');
    run('rm {{release_path}}/release.tar.gz');
    run('printf "%s\n" "{{release_name}}" > {{release_path}}/RELEASE');
});
```

CI pakuje `git archive` + `vendor/` (`--no-dev --classmap-authoritative`) + `public/build` + `bootstrap/ssr`. Suma kontrolna jest porównywana na serwerze, po uploadzie. `deploy:vendors` robi tylko `test -f {{release_path}}/vendor/autoload.php`.

Przed przełączeniem

```
task('deploy:migrate', function (): void {
    if (getenv('DEPLOY_ALLOW_MIGRATIONS') !== 'true') {
        writeln('<comment>Migracje pominięte: DEPLOY_ALLOW_MIGRATIONS=true jest wymagane.</comment>');

        return;
    }

    run('cd {{release_path}} && php artisan migrate --force --no-interaction');
});
```

Tylko expand/contract. Schemat nigdy nie jest cofany: zła migracja dostaje forward fix.

```
run('test -f {{release_path}}/public/build/manifest.json');
run('test -f {{release_path}}/bootstrap/ssr/app.js');
run('cd {{release_path}} && php artisan about --only=environment --no-interaction');
run('cd {{release_path}} && php artisan ops:readiness --no-interaction');
```

`deploy:smoke` : bootuje framework z cache'owanym configiem release'u. `ops:readiness` to komenda startera (baza, Redis, storage).

Przełączenie i restart

```
root /var/www/starter/current/public;

location = /index.php {
    include fastcgi_params;
    fastcgi_pass unix:/run/php/starter.sock;
    fastcgi_param SCRIPT_FILENAME $realpath_root/index.php;
    fastcgi_param DOCUMENT_ROOT $realpath_root;
}
```

`deploy:symlink` = `mv -T release current` (atomowo). `$realpath_root` pilnuje OPcache bez przeładowywania PHP-FPM.

```
run('cd {{current_path}} && php artisan horizon:terminate --no-interaction');
run('cd {{current_path}} && php artisan schedule:interrupt --no-interaction');
run('cd {{current_path}} && (php artisan inertia:stop-ssr --no-interaction || true)');
```

Unity systemd z `WorkingDirectory=…/current` i `Restart=always` wstają na nowym release. Joby w kolejce muszą działać między kolejnymi release'ami.

Jeden rollback, nigdy dwa

- `deploy:remember_healthy`: `readlink -f current` do `.dep/healthy_release` przed przełączeniem.
- `deploy:smoke:live`: `/health/ready` przez `curl --resolve host:443:127.0.0.1`, 3 próby, timeout 5 s, co 5 s.
- Nie przechodzi: powrót, zapis `BAD_RELEASE`, restart workerów, ponowna kontrola.
- Stary release też chory: zapis `.dep/deploy_blocked`. Każdy kolejny deploy staje, dopóki człowiek nie usunie pliku.
- Każda gałąź rzuca wyjątek, więc run workflow jest czerwony.

```
run('cd {{deploy_path}} && ln -sfn '.escapeshellarg($healthy).' current.rollback && mv -T
current.rollback current');
```

Atomowo, jak przełączenie w przód. Wbudowany task `rollback` używa bezpośrednio `ln -nfs` i sprawdza `BAD_RELEASE`, zanim wybierze kandydata.

Lista tasków

```
'deploy:check_blocked',
'deploy:prepare',
'deploy:vendors',
'artisan:storage:link',
'artisan:config:cache',
'artisan:route:cache',
'artisan:view:cache',
'artisan:event:cache',
'deploy:migrate',
'deploy:smoke',
'deploy:remember_healthy',
```

Przed przełączeniem: może się wywalić bez wiedzy użytkowników.

```
'deploy:symlink',
'deploy:restart_workers',
'deploy:smoke:live',
'deploy:unlock',
'deploy:cleanup',
'deploy:success',

after('deploy:failed', 'deploy:unlock');
```

Po przełączeniu: objęte rollbackiem. Kolejność sprawdzisz przez `dep tree deploy`.

Z GitHub Actions

```
export DEPLOY_ARTIFACT_SHA256="$(cut -d ' ' -f 1 .release/starter.tar.gz.sha256)"
vendor/bin/dep deploy "$DEPLOY_ENVIRONMENT" --file=deploy.php --revision="$GITHUB_SHA" --ansi
```

- Sprawdź SHA-256 artefaktu i to, czy commit z manifestu równa się `GITHUB_SHA`.
- Tylko `workflow_dispatch`; `concurrency: deploy-${{ inputs.environment }}`, `cancel-in-progress: false`.
- Klucz SSH jako sekret environmentu, przypięty `known_hosts`.
- Required reviewers na `production` (sprawdź plan GitHuba dla prywatnych repo).
- `deployer/deployer: ^8.0` w `require-dev`.

Kiedy tego nie używać

- **Kontenery:** artefaktem jest obraz, przełączenie i rollback robi orkiestrator.
- **Kilka serwerów webowych za load balancerem:** przełączenie przestaje być atomowe między hostami.
- **Platforma zarządzana** (Laravel Cloud, Forge z Envoyerem) już to robi.
- **Migracje nieaddytywne:** żadne narzędzie nie da deployu bez przestoju. Zaplanuj okno serwisowe.

PODSUMOWANIE

Ściąga w skrócie

- → Build w CI, upload tarballa, SHA-256 sprawdzane na serwerze
- → Migracje tylko na wyraźną flagę i tylko expand/contract
- → Smoke test kandydata przed `deploy:symlink`
- → `$realpath_root` zamiast przeładowywania PHP-FPM
- → Restart Horizona, schedulera i SSR po przełączeniu
- → Jeden automatyczny rollback, potem blokada i głośny błąd

Przydało się?

Zapisz tę ściągę na później.

Pełny artykuł: dominik-dev.pl

Udostępnij • Skomentuj