ŚCIĄGA LARAVEL

Bramki jakości i lefthook

Hooki narzędzi AI, git hooki, CI. Gwarancję daje tylko CI. GitLab CI, lefthook, gitleaks i hooki Claude Code dla Laravela w Dockerze.

Trzy poziomy

- **Hook narzędzia AI:** działa w sesji jednego agenta. Pomija go każdy, kto nie używa tego agenta. Daje szybką informację zwrotną w trakcie edycji.
- **Git hook:** działa tam, gdzie go zainstalowano. Pomija go `git commit --no-verify` albo brak instalacji. Poprawia styl, zanim powstanie commit.
- **CI:** działa przy każdym pushu, na serwerze. Nikt go nie pominie, jeśli gałąź jest chroniona. **To jest gwarancja.**
- Ważna reguła najpierw ląduje w CI. Hooki tylko obniżają koszt jej spełnienia.

CI sprawdza, nigdy nie poprawia

```
-    - composer larastan || true
-    - composer pint --test || true
+    - composer larastan
+    - ./vendor/bin/pint --test
-    - npm run format || true
-    - npm run types || true
-    - npm run lint || true
+    - npm run format:check
+    - npm run types
+    - npm run lint:check
```

`composer pint` nie istniał: `Command "pint" is not defined.`, kod wyjścia 1, a job zielony przez 17 miesięcy.

```
"format": "prettier --write resources/",
"format:check": "prettier --check resources/",
"lint": "eslint . --fix",
"lint:check": "eslint .",
```

`--write` i `--fix` w CI poprawiają checkout, który zaraz jest wyrzucany. ESLint przy ostrzeżeniach kończy się kodem 0: zostają widoczne, blokują tylko błędy.

Jedna komenda, te same kroki co CI

```
# Wszystkie bramki w trybie sprawdzania, te same kroki co w CI
check: setup-test-db
	docker compose exec -T app composer larastan
	docker compose exec -T app vendor/bin/pint --test
	docker compose exec -T app npm run format:check
	docker compose exec -T app npm run types
	docker compose exec -T app npm run lint:check
	docker compose exec -T app php artisan test --compact --parallel
```

Wpisz `make check` do `CLAUDE.md` i `AGENTS.md`, żeby agent uruchamiał bramkę z CI, zanim powie, że skończył.

lefthook.yml

```
pre-commit:
  commands:
    pint:
      glob: "*.php"
      run: docker compose exec -T app vendor/bin/pint {staged_files}
      priority: 1
      stage_fixed: true
    eslint:
      glob: "*.{js,jsx,ts,tsx}"
      run: docker compose exec -T app npx eslint --fix {staged_files}
      priority: 2
      stage_fixed: true
```

```
    prettier:
      glob: "resources/**/*.{js,jsx,ts,tsx,css}"
      run: docker compose exec -T app npx prettier --write --ignore-unknown {staged_files}
      priority: 3
      stage_fixed: true
    gitleaks:
      run: gitleaks git --pre-commit --staged --redact --verbose --no-banner
      priority: 4
```

Tylko pliki w stage'u. Testy zostają w `make check` i CI.

Cztery pułapki lefthooka

- `stage_fixed: true`, **nie** `git add .` Dodaje do stage'a tylko pliki, które przeszły `glob`, i działa wyłącznie w `pre-commit`.
- `docker compose exec -T`. Hooki działają bez terminala. `{staged_files}` są względne wobec katalogu głównego repo, który jest też katalogiem roboczym kontenera.
- **Kolejność, nie równoległość.** Domyślnie komendy idą po kolei, `priority` ustala kolejność. Przy `parallel: true` `eslint --fix` i `prettier --write` ścigają się o ten sam plik.
- **Globy gobwas.** `**` to jeden lub więcej katalogów. Potrzebujesz też zera? `glob_matcher: doublestar`.

```
$lefthook install   # istniejący hook przemianowuje na pre-commit.old
```

gitleaks: najpierw skan historii

```
docker run --rm -v "$PWD":/repo -w /repo ghcr.io/gitleaks/gitleaks:v8.30.1 \
  git --redact --no-banner .
```

578 commitów, `leaks found: 476`, wszystkie `generic-api-key`: 472 UUID-y `translation_key`, 4 fałszywe `STRIPE_SECRET` w dokumentacji skilli.

```
[extend]
useDefault = true

[[allowlists]]
targetRules = ["generic-api-key"]
regexTarget = "line"
regexes = ['''translation_key"?:\s*"?[0-9a-f]{8}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{12}''']

[[allowlists]]
targetRules = ["generic-api-key"]
paths = ['''^\.(claude|agents|cursor|github)/skills/.*\.md$''']
```

`.gitleaks.toml`: skan historii zwraca `no leaks found`, a fałszywy klucz AWS nadal jest łapany (kod wyjścia 1).

gitleaks w GitLab CI

```
secrets:gitleaks:
  stage: test
  image:
    name: ghcr.io/gitleaks/gitleaks:v8.30.1
    entrypoint: [""]
  variables:
    GIT_DEPTH: 0
  cache: []
  script:
    - gitleaks git --redact --no-banner --verbose .
```

`GIT_DEPTH: 0` : nowe projekty w GitLabie klonują się z głębokością 20, więc gitleaks widziałby tylko 20 commitów.

Claude Code: Pint po każdej edycji

```
"PostToolUse": [{
    "matcher": "Edit|Write",
    "hooks": [{
        "type": "command",
        "command": "\"$CLAUDE_PROJECT_DIR\"/.claude/hooks/pint-on-edit.sh"
    }]
}]
```

Wewnątrz `"hooks"` w `.claude/settings.json`.

```
file=$(jq -r '.tool_input.file_path // empty')
[[ "$file" == *.php ]] || exit 0

relative=${file#"$CLAUDE_PROJECT_DIR"/}
cd "$CLAUDE_PROJECT_DIR" || exit 0

if ! docker compose exec -T app vendor/bin/pint "$relative" > /dev/null; then
    echo "Pint nie mógł sformatować $relative" >&2
    exit 2
fi
```

Kod 2 w `PostToolUse` nie cofnie edycji, ale odsyła stderr do agenta w tej samej turze. Nie pomoże przy Codexie, Cursorze ani kodzie pisanym ręcznie.

Kiedy tego nie stosować

- **Brak CI:** najpierw job w CI. Hook bez bramki po stronie serwera przerzuca problem na tego, kto zapomniał go zainstalować.
- **Testy w pre-commicie:** hook z 217 testami przy każdym commicie uczy `--no-verify`. Testy idą do `make check` i CI.
- **Reguła, której nikt nie naprawi:** zostaw ostrzeżenia jako ostrzeżenia, zamiast od razu farbować pierwszy uczciwy pipeline na czerwono.
- **Jednoplikowe skrypty i repo na chwilę:** trzy poziomy kosztują więcej, niż oszczędzają.

PODSUMOWANIE

Ściąga w skrócie

- → Gwarancję daje tylko CI, hooki ułatwiają przejście
- → Bez `|| true`, w CI tylko skrypty sprawdzające
- → `make check` robi dokładnie to, co CI
- → lefthook: pliki w stage'u, `stage_fixed`, `priority`, `exec -T`
- → Przeskanuj całą historię, zanim gitleaks trafi do CI
- → Hooki AI dają agentowi informację zwrotną, niczego nie gwarantują

Przydało się?

Zapisz tę ściągę na później.

Pełny artykuł: dominik-dev.pl

Udostępnij • Skomentuj