BUDOWA OPROGRAMOWANIA W ERZE AI · 4.11.2026

Wdrażanie agentic coding w projektach typu brownfield

MATEUSZ GOSTAŃSKI · STAFF SOFTWARE ENGINEER

grixu.dev



Tutorial vs brownfield

TUTORIAL 1 repo · świeży projekt · jedna osoba decyduje o wszystkim



KOD

pisany przez wiele osób

każda z innymi nawykami; to, co działa, jest kompromisem



LUDZIE

autorzy, których już nie ma

odeszli albo prowadzą dziś inne zespoły



ZESPOŁY

zależne części jednego produktu

każdy zespół z własnym rytmem i priorytetami



STACK

wiele repo i języków

TypeScript i Python, frontend, backend, E2E, Terraform

Jak w takim środowisku wdrożyć ADLC?

ZESPÓŁ

Kim jestem i gdzie pracuję



Mateusz Gostański

ROLA W ZESPOLE
Tech Lead

STANOWISKO
Staff Software Engineer

alokai Console

4 DEVÓW · 1 PO · 1 UI DESIGNER

NASZE MONOREPO · TYPESCRIPT

aplikacja
frontend, API, testy E2E

usługi AI
agent do poprawek sklepów, proxy LLM

dokumentacja
docs, ADR i Terraform tej usługi

SĄSIEDNIE REPO

platforma chmurowa Alokai
PYTHON

infrastruktura jako kod
TERRAFORM

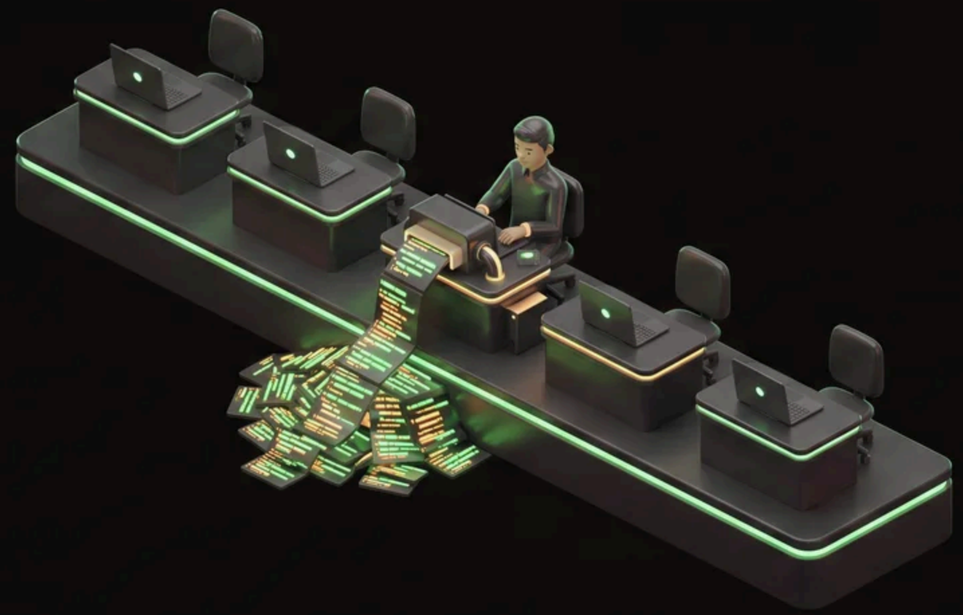
wdrożenia GitOps
ARGOCD

Nasz kod i to, co obok

```
~/vsf/
├─ console-monorepo/                    nasz zespół
├─ cloud-nucleus/                       platforma chmurowa · Python
├─ cloud-monorepo/                      Helm · konfiguracje
├─ cloud-terraform-internal-services/   usługi wewnętrzne · Terraform
├─ terraform-modules/                   wspólne moduły
├─ vsf-hosting-platform-iac/            hosting klientów · Terraform
├─ vsf-hosting-platform-argocd/         wdrożenia GitOps · ArgoCD
└─ console-customer-docs/               dokumentacja dla klientów
```

Poziom wyżej: repo innych zespołów, od których zależy nasz kod.

ADLC



Wszyscy zaczynają od kodu

Pierwszy boom prawie zawsze jest w Implement.
Reszta cyklu zostaje taka, jak była.

PLAN — DESIGN — **IMPLEMENT** — REVIEW — TEST — MAINTAIN — RETIRE

2025 H1 → 2026 Q3 · CI SAMI LUDZIE

Co się zmieniło

~70 → ~134

ZADAŃ „GOTOWE”
/ MIES.

50 → 249 linii

MEDIANA PR · ≈5×
Q1 2025 → Q3 2026

~700 → ~3 900

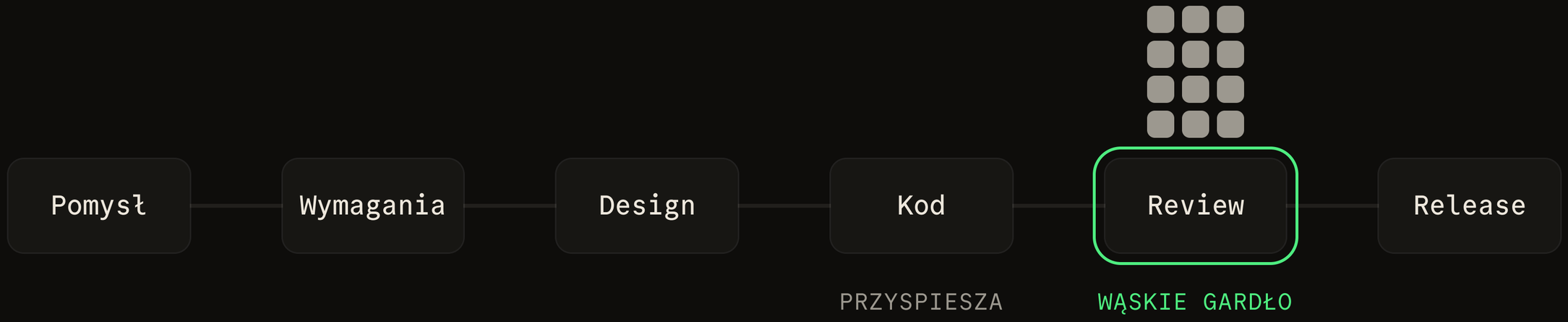
LINII ZMIAN NA
REVIEWERA / TYDZ.
30 DNI: 2025 → 2026

~5 → ~2,3 mies.

ZAPAS BACKLOGU

PRZEPŁYW PRACY

Kod przyspieszył. Korek się przeniósł.



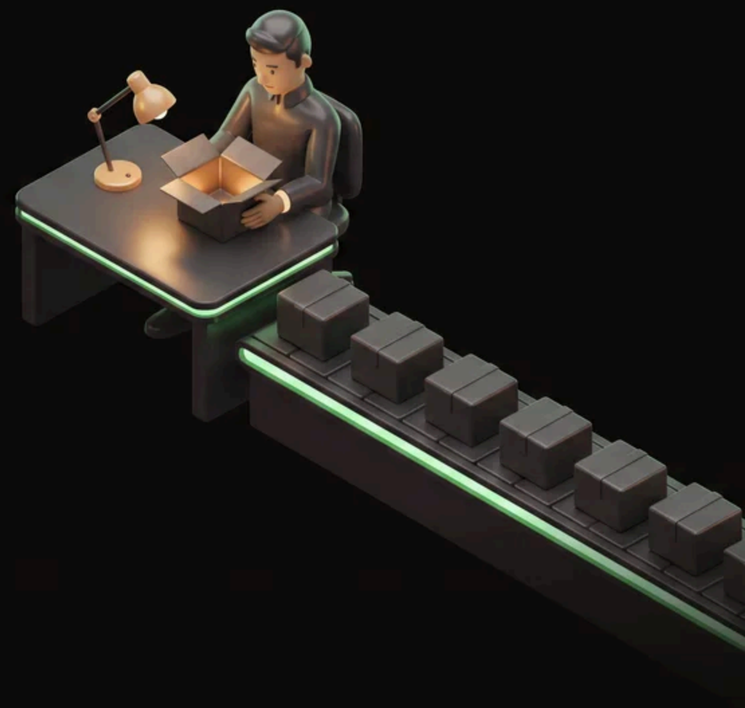
GDYBYM ZACZYNAŁ OD NOWA

Od czego zacząłbym dziś

1	**Review**	oceniamy ryzyko i wiemy, gdzie patrzeć
2	**PRD → spec → testy**	wiemy, co ma powstać, zanim powstanie kod
3	**Ludzie**	uczymy i oddajemy im narzędzia
4	**Codzienność**	automatyzujemy to, co się powtarza

KROK

1



Jak odblokować review?

Ci sami ludzie, ~5× więcej linii do przeczytania.

PLAN — DESIGN — IMPLEMENT — **REVIEW** — TEST — MAINTAIN — RETIRE

MINDSHIFT

Nie czytamy już każdej linii. Nie każdy PR zasługuje na to samo.

Oceniamy ryzyko tego, czego nie przeczytamy, i wiemy, na czym się skupić.



REVIEW

Cztery warstwy review

1	**Agentowe review**	self-review agenta, reguły i drugi agent w CI
2	**Dowody**	zrzut albo nagranie: tak to działa
3	**Ocena ryzyka**	gdzie w zmianie leży ryzyko i na czym się skupić
4	**Zrozumienie**	człowiek rozumie, co i dlaczego robi zmiana

WARSTWA 1 · AGENTOWE REVIEW

Wprowadź self-review

- agent sprawdza własną zmianę, zanim zobaczy ją człowiek
- plugin `code-review` do Claude Code: 7 lens, każda patrzy pod innym kątem

Claude Code · console

```
> /code-review:start-cr

* Dispatching 7 lenses in parallel.
  └ comments              simplicity & types
    readability & tests   security
    naming & module       performance
    objects & patterns    spec (skipped: no spec)

✓ All seven lenses delivered.
```

- `security` · … `javascript:` targets also serialise to origin `"null"`, so the equality passes and the value reaches `res.redirect(302, …)` with session cookies already written

WARSTWA 1 · AGENTOWE REVIEW

Uwaga z review staje się regułą

1. „Please use `Promise.all`” — 3 razy w review
2. kolega zapisuje to jako regułę
3. reguły ostrzegają albo blokują agenta

Claude Code · console — A BLOCKING RULE IN TESTS

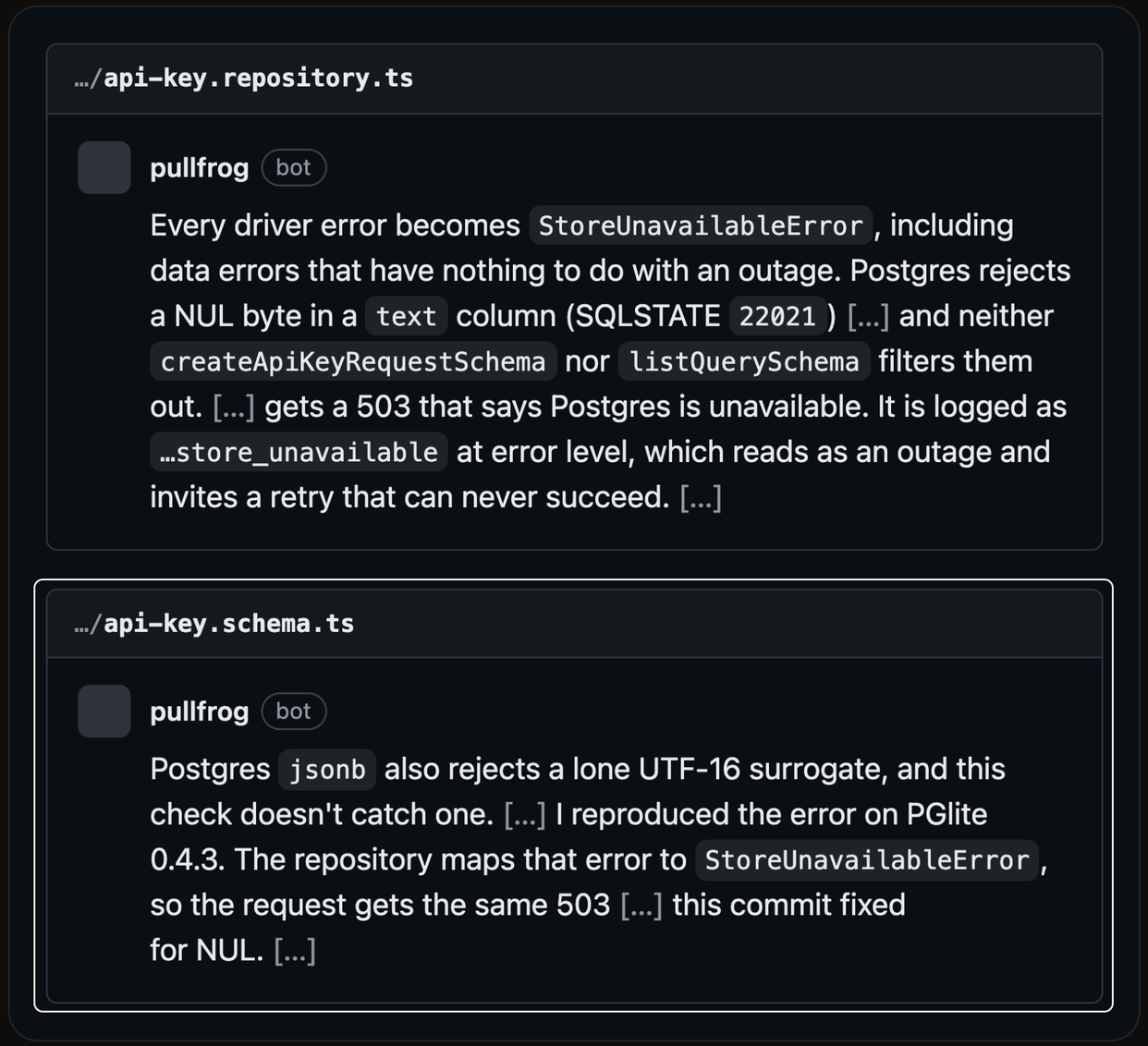
- Edit `cloud.controller.spec.ts`
  - `+ expect(…).toHaveBeenCalledWith(…)`
  - `block-detailed-error-logging-tests`
    `Detailed error/logging test blocked`
- The hook blocks assertions on the reporter’s arguments, so I only check that `warn` was called.
- Edit `cloud.controller.spec.ts`
  - `+ expect(…).toHaveBeenCalledTimes(1)`

✓ fixed in 11 s

WARSTWA 1 · AGENTOWE REVIEW

Drugi agent sprawdza PR w CI

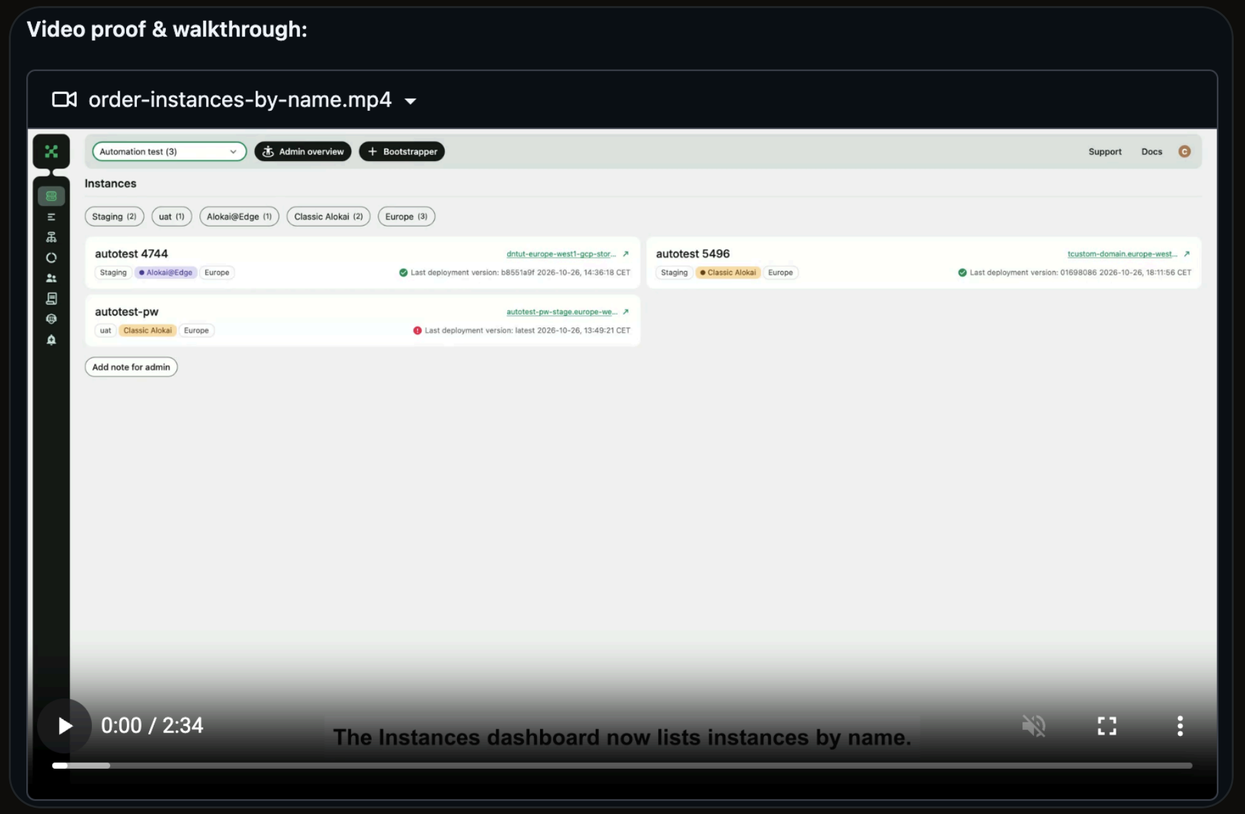
- pullfrog znajduje błąd, który udaje awarię bazy
- potem ten sam błąd w poprawce od Claude’a
- PR-y otwieramy wcześnie: zanim zajrzy człowiek, pracują już boty, CI i `pr-shepherd`



WARSTWA 2 · DOWODY

PR z nagraniem zmiany

- PR pokazuje, jak zmiana działa: zrzut albo nagranie z napisami
- reviewer widzi zachowanie, zanim zajrzy w kod
- nagranie robi skill `walkthrough-video`: ze specyfikacji albo z samej zmiany

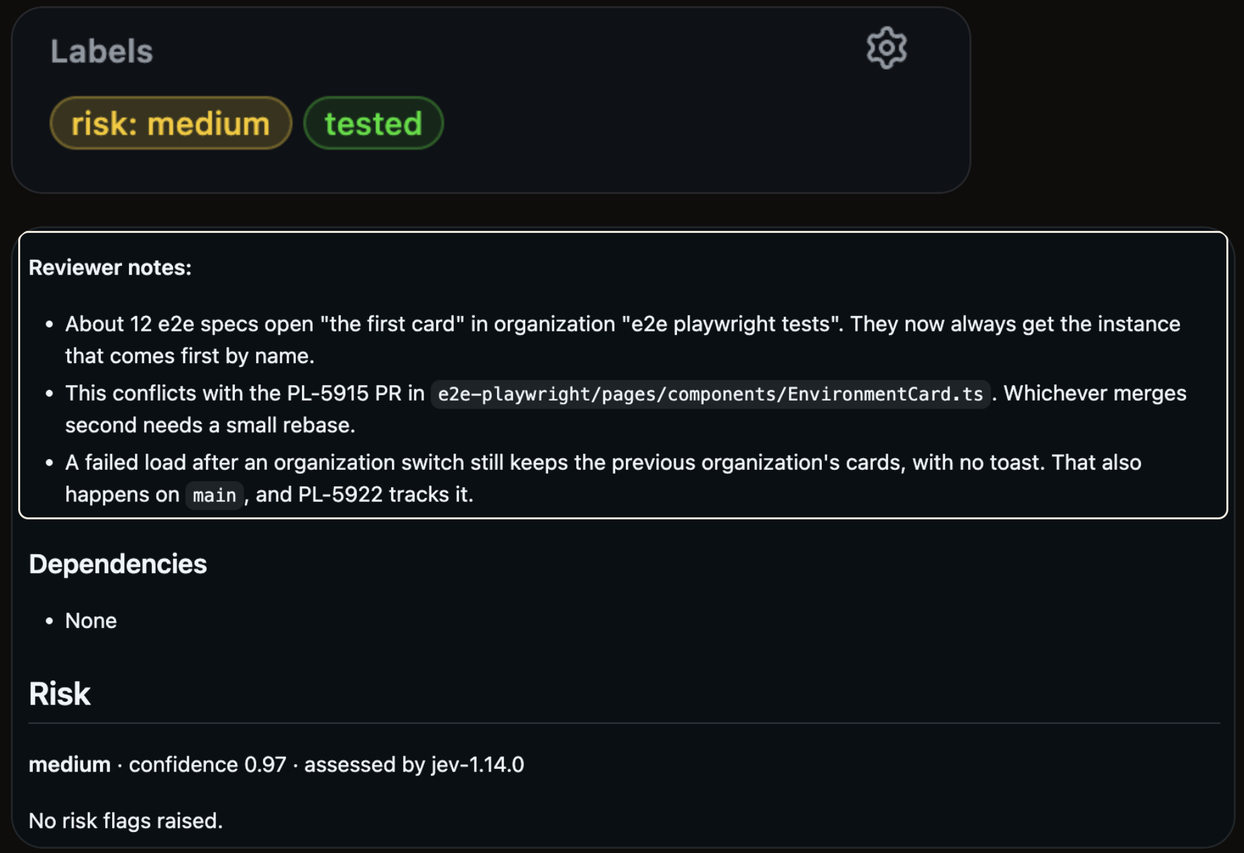


PR · NAGRANIE 2:34 Z NAPISAMI

WARSTWA 3 · OCENA RYZYKA

Najpierw ryzyko, potem kod

- PR dostaje ocenę ryzyka i etykietę
- uwagi dla reviewera mówią, gdzie patrzeć
- uważnie czytam tylko tam, gdzie ryzyko jest wysokie



WARSTWA 4 · ZROZUMIENIE

PR opowiedziany jak historia

- `/code-story` opowiada PR w 8 rozdziałach
- każdy rozdział: pytanie → narracja → kod
- każde „dlaczego” ma źródło

SKĄD POCHODZI „DLACZEGO” W TEJ HISTORII

- **Opis PR #4268** — cel zmiany, lista API, zależność od #4244 i Cerbos 0.56, polecenie liczenia wierszy `status = 3` przed wdrożeniem, flaga `role_management`.
- **Komunikaty commitów** — powód flagi `assignable: false` (`f39e47261`), podział plików-łączników (`7195eb9a0`), właściciel zachowuje admina (`e78aed9a7`), zajęcie zaproszenia przed odczytem ról (`965c266d2`), przeładowanie po samym kasowaniu (`53b0e7523`).
- **Komentarze w kodzie** — limit przed blokadą, odmowa przy padniętym Redis, dzierżawa krótsza niż odstęp, porównanie tylko treści polityki.
- **Przegląd pullfrog** (na `a95b0f0` , starszym niż ta gałąź) — wszystkie ramki „Uwaga”.

- **Moje wnioski, nie zapisane w repo** — dlaczego strażnik musi czytać uprawnienia poza Cerbosem (Cerbos nie odpowiada na pytanie o podzbiór). Kod tego nie mówi.

KONIEC HISTORII · ŹRÓDŁA

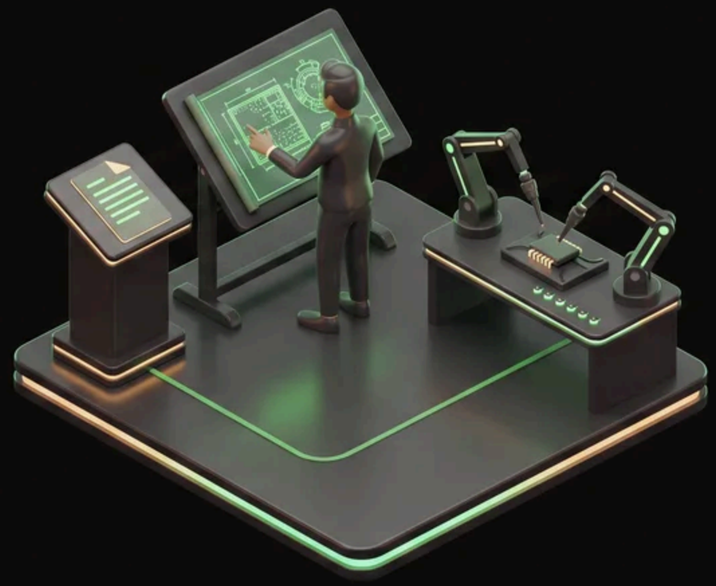
PRZEPŁYW PRACY

Review się odblokowało. Teraz brakuje pracy.



KROK

2



Od PRD do testów

Gotowej pracy zaczyna brakować. Zaczynamy od wymagań.

PLAN — DESIGN — IMPLEMENT — REVIEW — TEST — MAINTAIN — RETIRE

KROK 2 · WYMAGANIA

PRD bez spotkań

Claude Code · console

> /prd-init

└ 8 questions → 9 decisions · 3 for other roles

✓ 28 min · 9 decisions

… /PRD-ROUND · 3 DAYS LATER

> /prd-round

- Overhead threshold for 99% of requests: keep 800 ms? Today LiteLLM handles 99% of requests within 14 ms …

  └ "p99 < 800 ms" → data → "≤ 150 ms"

BEZ SPOTKANIA

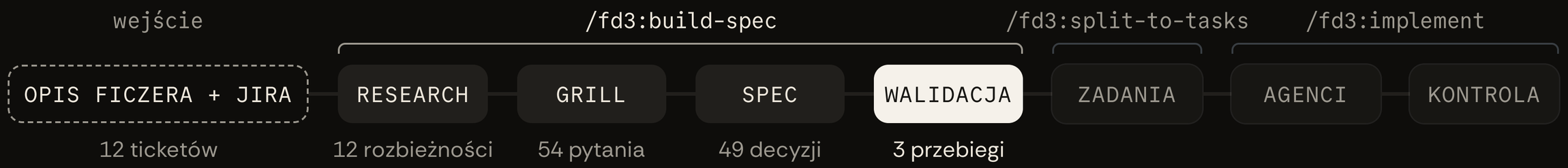
Inny dev odpowiada na otwarte pytanie commitem i zatwierdza 2 decyzje.

FORMAT DOKUMENTU

Stały szkielet, ale luźny — za ciasny nie pasuje do różnych przypadków.

KROK 2 · SPECYFIKACJA

Ze specyfikacją błędy wychodzą przed kodem



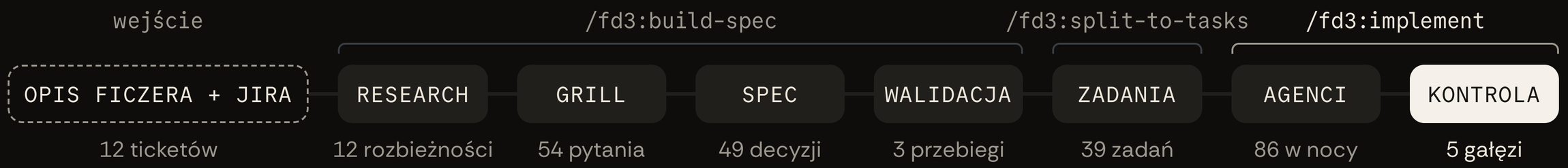
Claude Code · fd3 — ONE SESSION, CONDENSED

> 8. Użyj firecrawl, API key przekazany przez TF

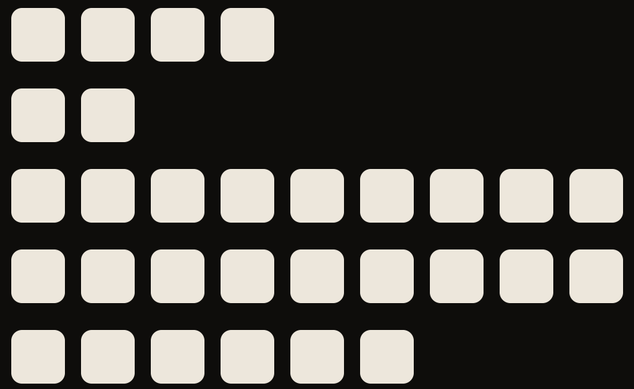
✓ D21 · Documentation lookup uses firecrawl, with the API key delivered through Terraform. `WebSearch` is not available. ← Q8

- **fd3:write-spec** SPEC.md · 658 lines · 49 decisions
- **fd3:validate-spec** passes 1, 2, 3
- The second validation pass found 3 errors the first one let through.

Zadania, noc agentów i poranna kontrola



30 dla agentów, na 5 gałęziach



- ✓ sprawdzone
- ✓ sprawdzone
- ✓ sprawdzone
- ✓ sprawdzone
- ✓ sprawdzone

9 tylko dla człowieka



Naprawa: 14 agentów, 12 min. **Sprawdzone, nie przepisane z raportu.**

Spec mówi, co sprawdzić

1 CO SIĘ ZMIENIŁO

```
  AC-3 admin widzi listę ról
+ AC-4 admin tworzy własną rolę
+ AC-5 członek bez uprawnień → 403
+ AC-6 awaria usługi → komunikat
```

kryteria ze spec albo `git diff`

2 SCENARIUSZE

API · curl

- AC-4 POST /roles
- AC-5 403

UI · przeglądarka

- AC-4 formularz roli

Błędy · WireMock

AC-6 503 · timeout · zły JSON

- zmienia dane

3 TY DECYDUJESZ

Czy testy mogą zmieniać dane?

- ✓ utworzenie roli
- ✗ ~~usunięcie roli~~
- ✓ seed użytkownika testowego

4 BŁĘDY ZALEŻNOŚCI



czy aplikacja dobrze obsłuży każdy z tych błędów?

5 RAPORT

69

SPRAWDZEŃ

62 / 1 / 6

OK / BŁĄD / ZABLOKOWANE

każdy błąd z dowodem

KROK

3



Ucz ludzi

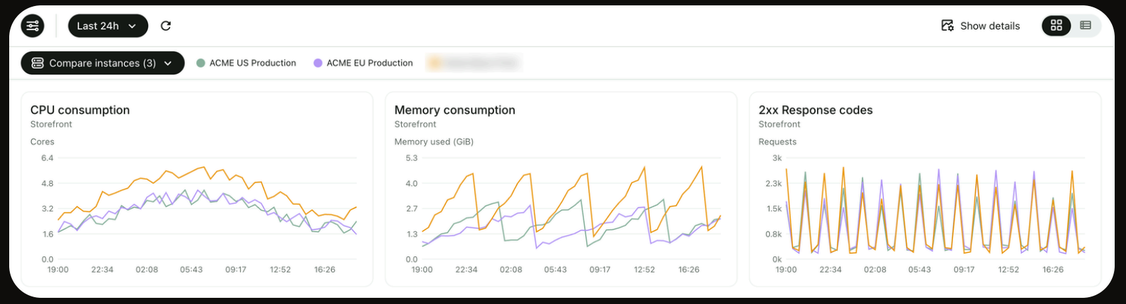
Pozostałe fazy ADLC nie wdrożą się same.

KROK 3 · DESIGN

Skille do designu napisał designer

Nie ja — ja tylko go wdrożyłem. Dziś, jak sam szacuje, ~70% czasu spędza w Claude Code, ~30% w Figmie.

1	**Wejście**	szkic low-fi i draft w Figmie
2	**Prototyp**	`/ui-design` : klikalna makieta w repo
3	**Weryfikacja**	agent porównuje zrzuty z Figmą
4	**Iteracje**	komentarze → poprawki → v2
5	**Przekazanie**	do Figmy, a makieta zasila spec



PROTOTYP W REPO · METRIC COMPARISON

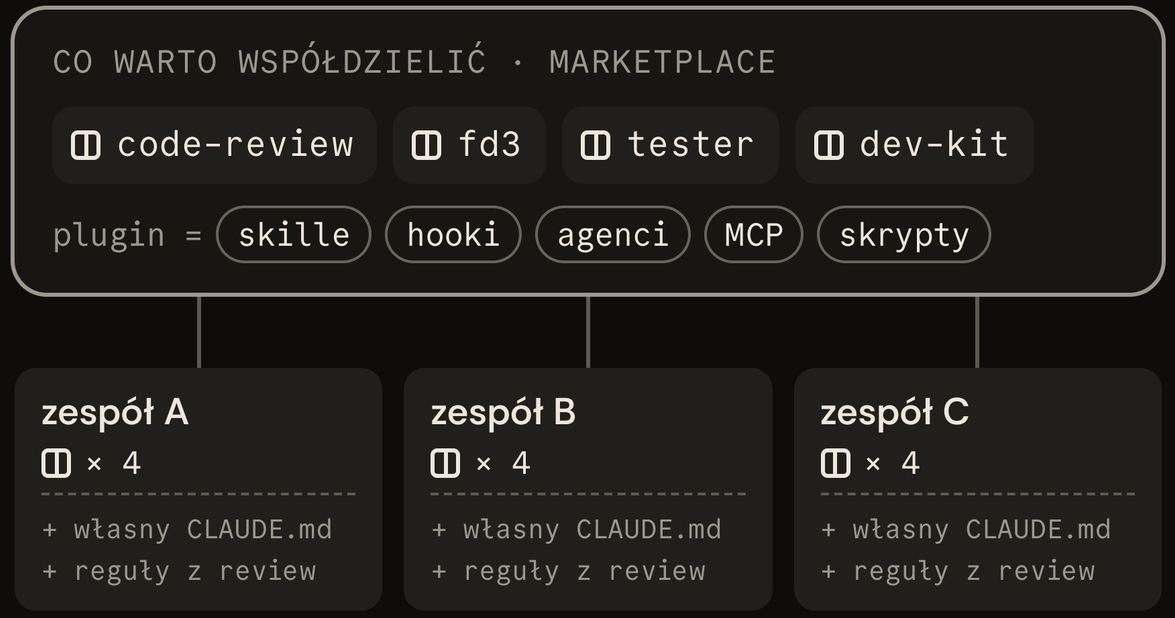
KROK 3 · ENABLEMENT

Ucz ludzi i daj im się bawić

JAK TO WPROWADZAM W ZESPOLE

1	**Pokazuj**	czym sam się bawisz
2	**Ucz**	narzędzi, nie tylko ich konfiguracji
3	**Daj czas**	na eksperymenty
4	**Zostaw miejsce**	nie łataj wszystkiego sam, niech zrobi to ktoś z zespołu

JAK TO ROZLAĆ NA INNE ZESPOŁY



Najwięcej dało nam ADLC tam, gdzie ludzie sami zbudowali swoje narzędzia.

KROK

4

Codzienność



Fazy, o których mało się mówi, a które też są częścią ADLC.

PLAN — DESIGN — IMPLEMENT — REVIEW — TEST — **MAINTAIN** — RETIRE

Agenci poza fazą Implement

PLAN — DESIGN — IMPLEMENT — REVIEW — TEST — MAINTAIN — RETIRE

	MAINTAIN	RETIRE
ZADANIE	`/dev-kit:dep-upgrade-check` PR z aktualizacją simple-git 3 → 4	`/codebase-design` znajdź kod do usunięcia
CO ZROBIŁ AGENT	znalazł przyczynę w kodzie biblioteki, poprawka w 1 linii	3 audyty naraz, potem usuwanie w 6 etapach
WYNIK	2,5 min DO WERDYKTU	-1 174 LINII · 133 PLIKI

Skill w pętli prowadzi PR, aż będzie gotowy

Claude Code · cc-toolkit

- @pullfrog Not changing this. `ScheduleWakeup` does take `stop: true`: the … docs say … "…which cancels the pending wakeup immediately"
- Fixed and pushed two, rejected two with reasons. All four threads are resolved.

… ITERATION 4 · 5 MIN LATER

- All 3 checks are green and no threads are open. The `pr-shepherd` exit condition is met, stopping the loop.
- `ScheduleWakeup stop: true`
- ✓ The `pr-shepherd` loop finished after four iterations.

PĘTLA KOŃCZY SIĘ, GDY

CI jest zielone i wszystkie komentarze z review są rozwiązane.

W PRACY · 6 PR-ÓW NARAZ · 29 MIN

10/11

KOMENTARZY BOTA
Z REVIEW ZAMKNIĘTYCH
BEZ MNIE

KROK 4 · UTRZYMANIE

Wpisz przegląd narzędzi AI w kalendarz



GDY COŚ SIĘ WYDARZY

1. NOWY MODEL
   `/doctor prompt-audit`
2. AKTUALIZACJA NARZĘDZIA
   przeczytaj changelog, porównaj ze swoją wersją
3. REFAKTOR
   popraw fakty w CLAUDE.md, najlepiej generuj je automatycznie

STAŁE TERMINY

- CO MIESIĄC
  przejrzyj spis skilli: skąd są i w jakiej wersji
- CO ≤ 6 MIES.
  pełny przegląd, nawet gdy nic się nie wydarzyło

ZAMKNIĘCIE

Co zabrać ze sobą



1

Oceniaj PR po dowodach i ryzyku

`code-review`

`dev-kit:pr-open`

`walkthrough-video`



2

Przejdź na Spec-Driven Development

`fd3` `tester`



3

Daj ludziom czas i miejsce na własne narzędzia



4

Dostrzegaj codzienne przeszkody i usuwaj je z pomocą agentów

`dev-kit:ci-fix`

`dev-kit:pr-shepherd`

Buduj swój proces ADLC.

BUDOWA OPROGRAMOWANIA W ERZE AI · 4.11.2026

Dziękuję

Wszystkie pluginy z prezentacji są do pobrania

grixu.dev

Obserwuj

LINKEDIN

in/mateusz-gostanski

Pobierz

PLUGINY DO CLAUDE CODE

github.com/grixu/cc-toolkit



MATEUSZ GOSTAŃSKI · STAFF SOFTWARE ENGINEER