

Coder plus vite avec l'IA, livrer mieux en équipe : du code au merge

Florian Bruniaux

Founding Engineer / Harness Engineer @ Méthode Aristote

Ce qu'on va voir

Comment organiser les vérifications, du code produit au merge.

- | | | |
|----|----------------------|---|
| 01 | Vérifier tôt | Tester en local et préparer une PR claire. |
| 02 | Organiser la review | Faire travailler les bots et trier leurs retours. |
| 03 | Décider du merge | Corriger, vérifier et savoir quand s'arrêter. |
| 04 | Améliorer le harnais | Apprendre des problèmes et voir si l'équipe gagne du temps. |



DEV WITH AI · 12 OCTOBRE 2026

Dev
With
AI

Florian Bruniaux

Depuis 2017, du NLP à l'IA générative.

AUJOURD'HUI

AI Founding Engineer

Méthode Aristote · EdTech + AI

Spécialiste du **context** & **harness** engineering

Geek et passionné d'open source

Open source depuis début 2026



RTK : core contributor Q1 2026

Une idée, une soirée, un projet.

~15

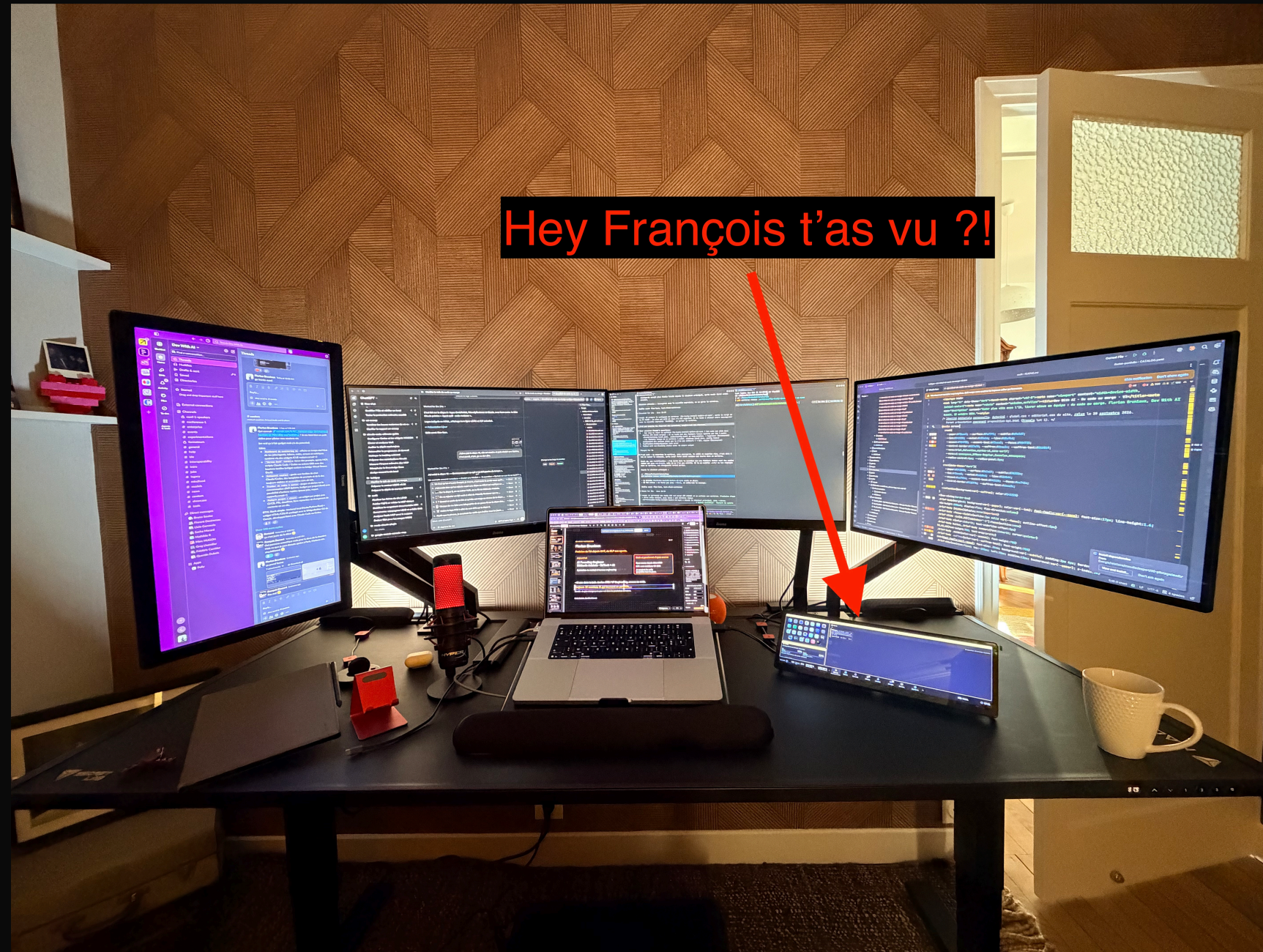
ans dans la tech

Ancien CTO / VP Engineering,
revenu au code.

30

**sessions IA qui tournent
toujours en parallèle.**

Un vrai nerd...



Hey François t'as vu ?!

XENEON Edge

Le petit écran à droite
de mon portable.



[Découvrir le XENEON Edge](#)

Quelques projets et ressources

Claude Code Ultimate Guide

cc.bruniaux.com

Claude Cowork Guide

cowork.bruniaux.com

ccboard

ccboard.bruniaux.com

CC Copilot Bridge

ccbridge.bruniaux.com

StarMapper

starmapper.bruniaux.com

Les petits nouveaux

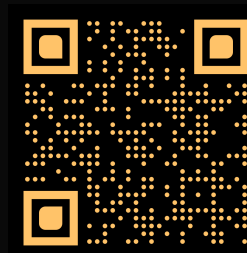
Search Console MCP

search-console.bruniaux.com

wt-sweep

wt-sweep.bruniaux.com

github.com/FlorianBruniaux



florian.bruniaux.com



Les agents se multiplient. Qui relit derrière ?

Quand les PR arrivent plus vite qu'on ne les vérifie...

EN PARALLÈLE



Produire plus de code ne suffit pas à livrer plus vite.

Organiser les vérifications avec le harnais



4 · COORDINATION

Files · budgets · responsabilités

3 · EXÉCUTION

Le goulot se déplace ici



SUR LE POSTE

Tests et review locale



DANS LA CI

Checks et previews



SUR LA PR

Reviews et décision

Permissions · traces · coûts



2 · REPO

Instructions et tests versionnés



1 · POSTE

Outils et configuration

Vérifier tôt, valider dans la CI



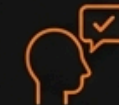
**Découvrir une erreur plus tard
peut multiplier les coûts et les délais**



Runs supplémentaires



Attente



Review à reprendre

Le local prépare la PR. La CI requise reste obligatoire.

Une PR lisible par l'équipe et les agents

TL;DR

Après un `git pull` qui modifie `pnpm-lock.yaml` ou `pnpm-workspace.yaml`, les hooks `post-merge` et `post-rewrite` échouaient sur `ai:sync`, parce que pnpm refuse tout script tant que `node_modules` n'est pas à jour. Les deux hooks lancent désormais `pnpm install --frozen-lockfile` d'abord, comme le fait déjà `post-checkout`.

▶  Modélisation

▶  Métier : la règle qui change

▶  Business : ce que ça change si c'est faux

▶  Tech : ce qui change dans le code

▶  Tests : ce qui est vérifié, ce qui ne l'est pas

▶  Doc : ce qui doit suivre

▶  Checklist

▶  AI et détails

▶  Local harness details

Pour l'humain

Comprendre le problème et le changement.

Pour les agents

Mon skill lit « AI et détails » : une section optimisée pour l'IA.

Pour tous

Tests effectués, limites et points ouverts.

Qu'est-ce que ce changement peut casser ?

Une vérification précise, des résultats observables.

CE QUI CHANGE



L'IA PEUT VÉRIFIER



L'HUMAIN PEUT AUSSI

**Un bouton
ou une interface**

Inspecter le DOM, tester le clic.
Contrôler le rendu sur mobile.

Vérifier la lisibilité
et le parcours réel.

**Un composant
partagé**

Écrire / lancer des tests E2E
sur les pages concernées.

Jouer les parcours sensibles.
Repérer les cas oubliés.

**Les droits
d'accès (RBAC)**

Tester les rôles : accès autorisé
et refusé, côté UI et API.

Confirmer qui doit avoir accès.
Revoir les cas à risque.

Des rôles complémentaires, pas exclusifs. Les tests habituels restent requis.

Des labels GitHub pour orienter la review

Qui doit relire cette PR ?

EXEMPLES DE LABELS



Review AI

Review confiée à l'IA

Changement borné, connu et vérifiable par les tests.

Dans un périmètre autorisé par l'équipe.



Review Human

Relecture humaine requise

Droits d'accès, données privées, calculs, paiements ou conséquences mal connues.

En cas de doute, on demande une relecture humaine.

POUR L'HUMAIN : LES PR À RELIRE

is:pr is:open label:"Review Human"

POUR L'IA : LES PR À TRAITER

is:pr is:open label:"Review AI"

Un repère pour l'humain. Un filtre pour l'IA.

Les contrôles requis restent obligatoires.

3 lecteurs / 1 review à consolider

Même version du code · Même base de comparaison



CodeRabbit

Service externe



FriendlyReviewer

de Marc Hugon · service externe



Claude

Agent natif · code-reviewer

/tech:pr-feedbacks

Mon skill pour traiter tous les retours de la PR

1 · Récupérer

Tous les feedbacks
de la PR

→ 2 · Valider

Lire, recouper et vérifier
dans le code actuel

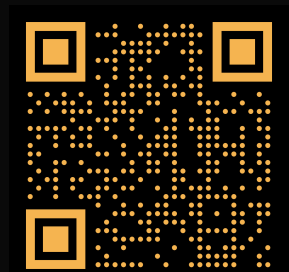
→ 3 · Corriger

Lancer les correctifs
multi-agents / multi-LLM

Après les correctifs : relancer les tests et les reviews concernés.

Pour l'humain : désaccords, limites et décisions à prendre.

friendlyreviewer.fr



Un bon diagnostic ne garantit pas une bonne correction

Le bot a raison sur le problème. Sa solution reste à vérifier.

EXEMPLE SIMPLIFIÉ : 2 CAS, 2 ERREURS ATTENDUES

1 · PROBLÈME REPÉRÉ

Le test accepte toute erreur.

- Document absent → une erreur
- Accès interdit → une erreur

Il peut passer pour la mauvaise raison.

2 · SUGGESTION DU BOT

Attendre le même message dans les 2 cas.

- Document absent → « Refusé »
- Accès interdit → « Refusé »

Plus précis, mais toujours incorrect.

3 · CORRECTION À FAIRE

Vérifier l'erreur propre à chaque cas.

- Document absent → « Introuvable »
- Accès interdit → « Refusé »

Adapter les assertions, puis retester.

Vérifier le problème signalé. Puis vérifier la correction proposée.

Quand l'agent doit-il arrêter de corriger ?



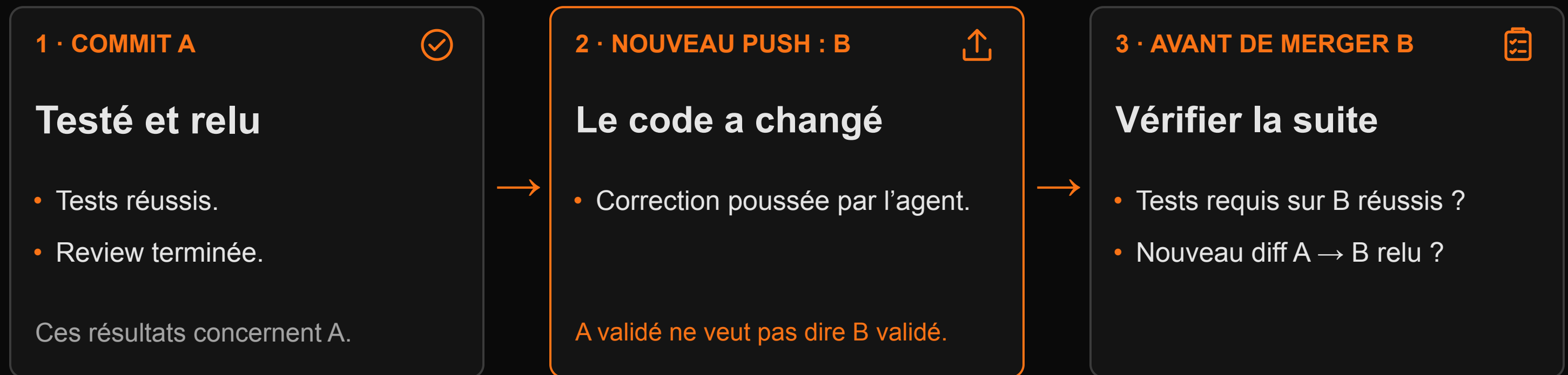
La CI est rouge. Faut-il modifier le code ?



Une CI rouge ne veut pas toujours dire un bug dans le code.

Un nouveau push change ce qu'il reste à vérifier

Les tests et la review portent sur une version précise du code.

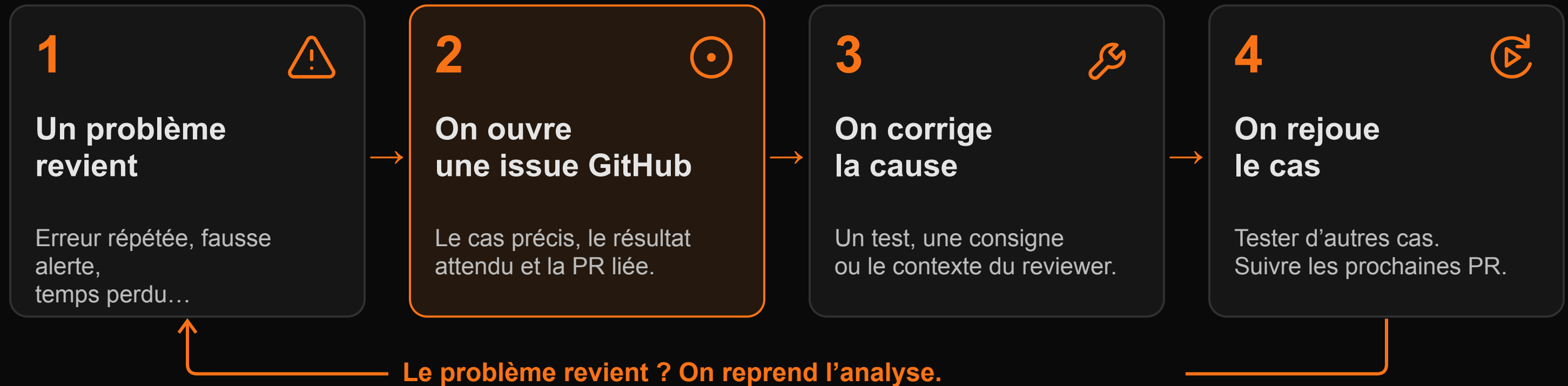


Après chaque push : identifier ce qu'il faut retester et relire.

Un test en cours, sauté ou annulé n'est pas un test réussi.

Quand un problème revient, on améliore le harnais

Une issue pour garder la trace. Un changement à vérifier.



EXEMPLE

L'agent oublie
le cas « liste vide ».



Test de régression
+ consigne précisée si besoin.



On vérifie ce cas
et les autres comportements.

Chez Aristote, on livre plus vite

Une feature en quelques heures. Une épique en quelques jours.



Comparer des PR similaires, avant et après



Moins de temps humain ? Lire et trier les commentaires



Moins d'attente ? De la demande de review au merge



Moins de reprises ? Corrections et bugs après livraison



À quel coût ? Modèles et CI, runs ratés compris

Qu'est-ce qu'on peut faire mieux ?

Repartez d'une PR récente. Choisissez 1 amélioration à essayer.

REVENIR SUR UNE PR

-  Qu'est-ce qui a mal marché ?
-  Qu'est-ce qui a pris le plus de temps ?
-  Qu'est-ce qu'on change sur la prochaine PR ?

1 Essayer 1 changement



Choisissez une action concrète.
Testez-la sur la prochaine PR.

2 Regarder si ça aide



Moins d'attente ? Moins de reprises ?
Sans laisser passer plus de bugs.

Gardez ce qui aide l'équipe. Ajustez ou abandonnez le reste.

Le harnais, du besoin à la livraison

Le harnais **évolue** avec le code, les outils et l'équipe. Il demande des **hard skills** et de l'**organisation**.

LE CADRE COMMUN

Instructions et règles · skills et outils · permissions et budgets

 SUR LE POSTE

 SUR LA PR ET DANS LA CI

 AVEC L'ÉQUIPE

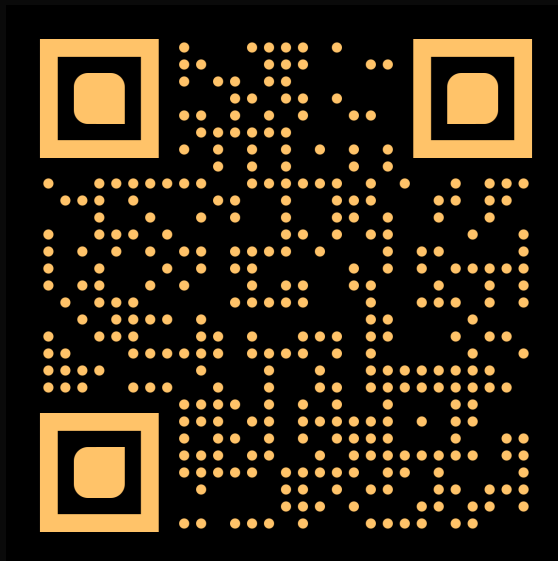


Corriger, puis relancer les tests et les reviews concernés

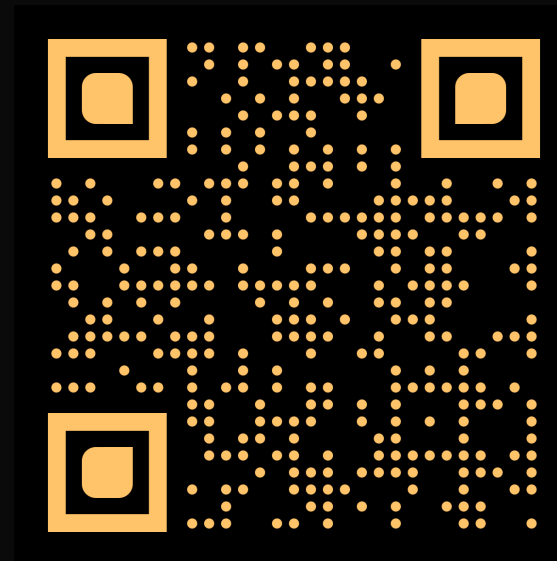
APRÈS LIVRAISON

Bugs, retours, temps et coûts → améliorer les règles, tests et outils

Questions ?



[Portfolio](#)



[GitHub](#)