



Benchhttp

Tests end-to-end de la performance d'endpoints HTTP

Grégory Albouy, Clara Gonnou, Damien Mathieu, Alex Mongeot, Thomas Moreira, Kérian Pelat

Au cours du cycle de vie d'un produit, il faudra sur le back-end ...

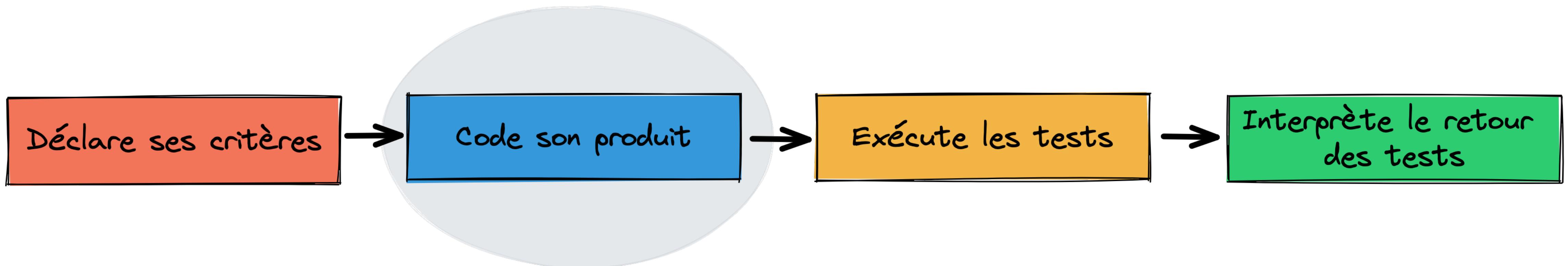
- 📦 intégrer une nouvelle **feature**
 - risques d'introduire des **régressions de performance**
- 🔧 faire du **refactoring**
- ✍️ réécrire après un **changement de spécifications**
 - nouveaux **standards** à atteindre

Quels besoins adresser ?

Spécifications du problème

- pouvoir **évaluer la performance d'endpoints HTTP**
- pouvoir **analyser** les résultats avec des **statistiques adaptées**
- pouvoir **définir** ce qui passe et ne passe pas un **test d'acceptation**
- pouvoir être **déclaratif à haut niveau**, ne pas toucher au code source

User journey : le flux de travail qu'un utilisateur veut suivre

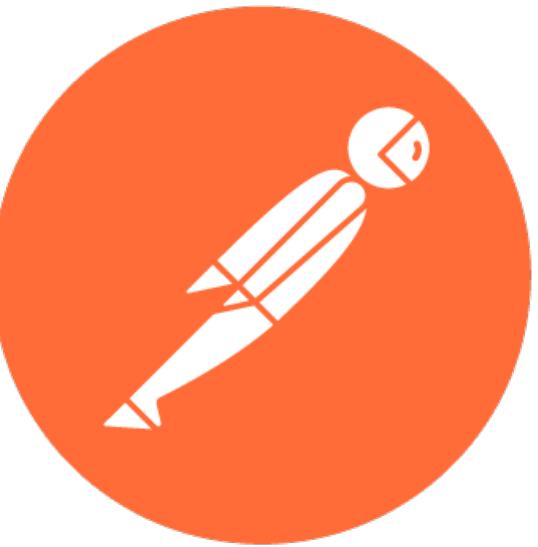


Quelles solutions aujourd'hui ?

Outils généralistes pour tester des APIs web

Puissants mais ne couvrent pas exactement nos besoins

- Postman ou Insomnia
- sont à exécuter **manuellement**
- testent le **contenu** des réponses



Outils spécialisés sur les tests de performance

Correspondent mais complexes et/ou invasifs dans le workflow

- **Gatling** ou **Blackfire**
- flexibles mais **complexes** à configurer
- configuration via du **code source**
- nécessite un compte utilisateur



The logo for Benchhttp features a yellow lightning bolt icon positioned to the left of the word "Benchhttp". The word is written in a bold, sans-serif font, also in yellow. The entire logo is set against a solid black background.

Benchhttp

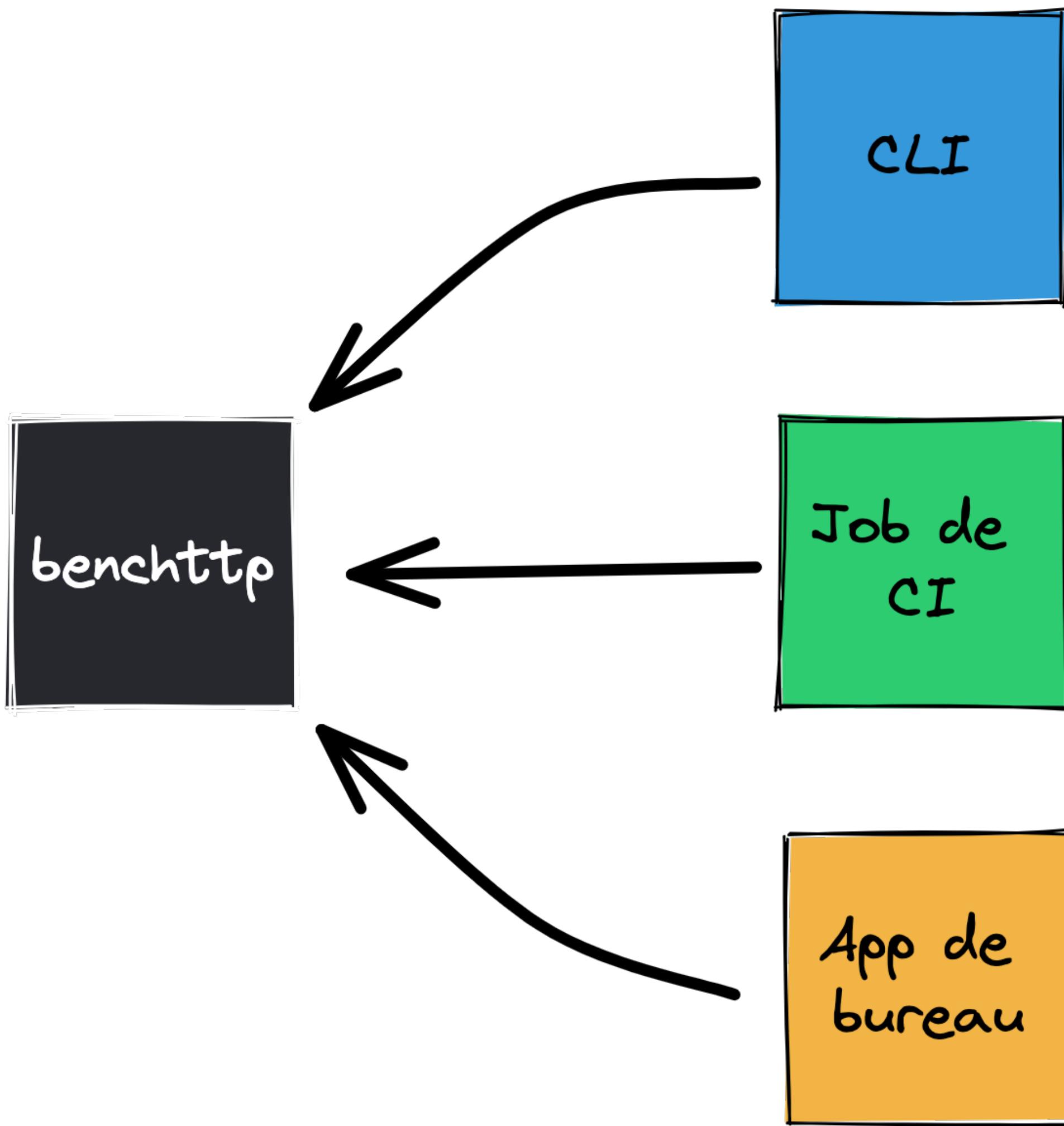
Benchttp : la solution et les cas d'utilisation

💡 tester pendant la **phase de développement**

✓ faire des **tests d'acceptation** et de **non-régression en CI**

📊 visualiser les résultats de manière **intelligible** via des **statistiques adaptées**

1 moteur → 3 cas d'utilisation → 3 applications

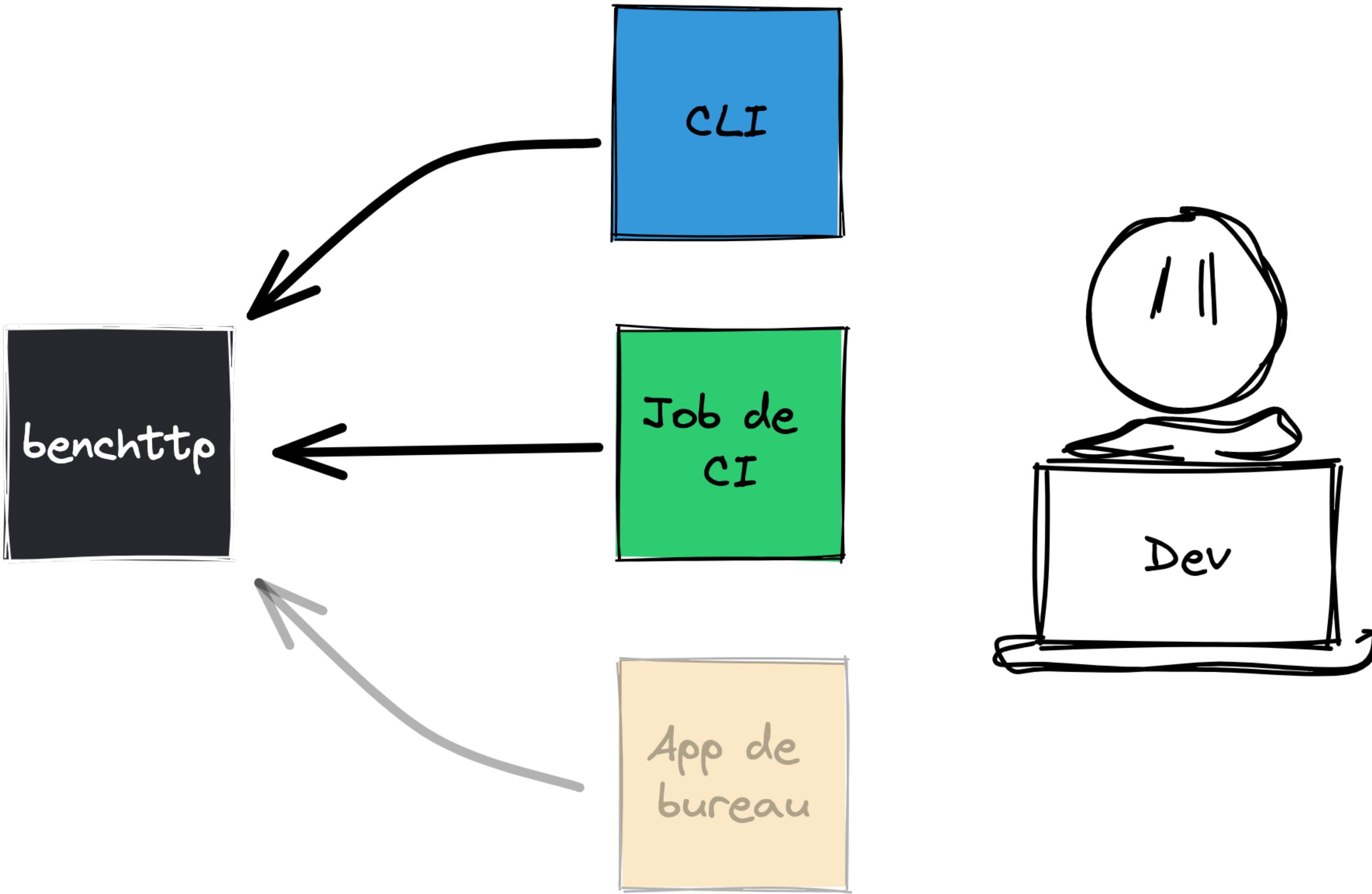


configurer et tester
feedback loop pendant le développement

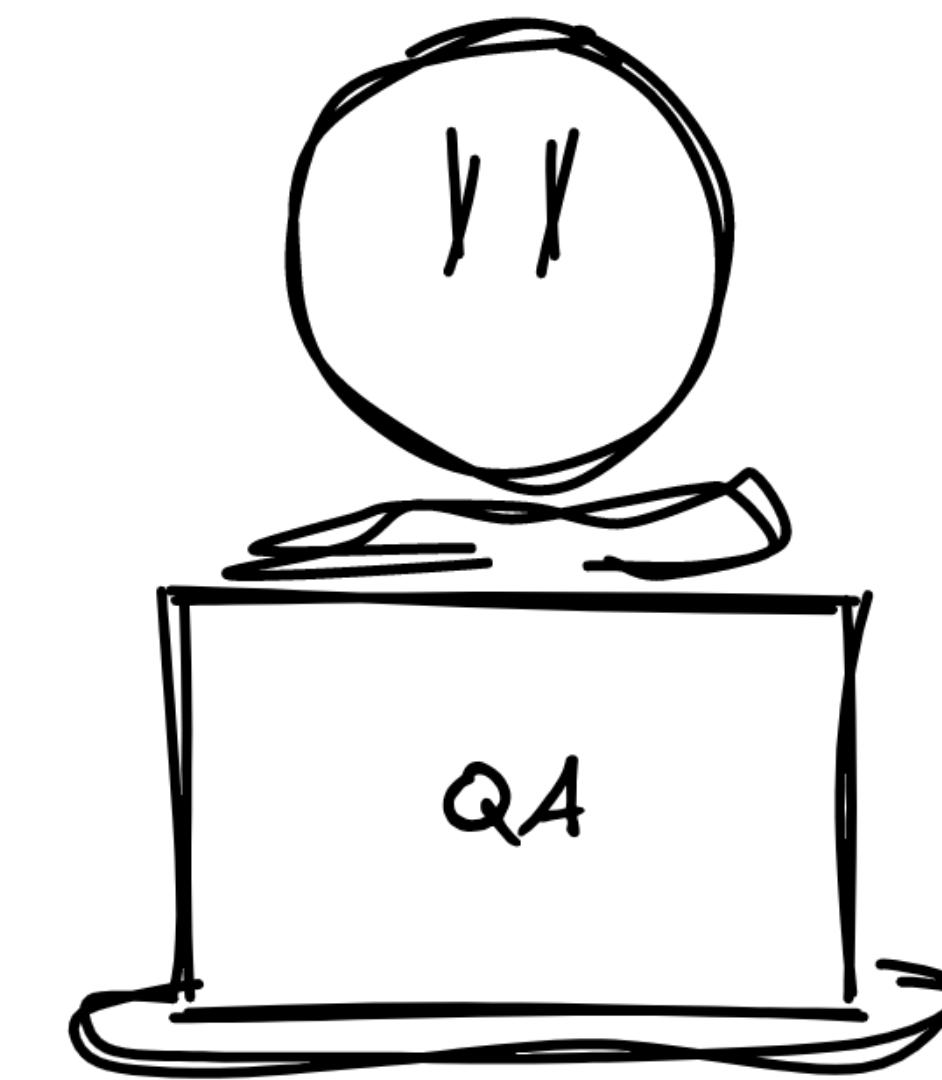
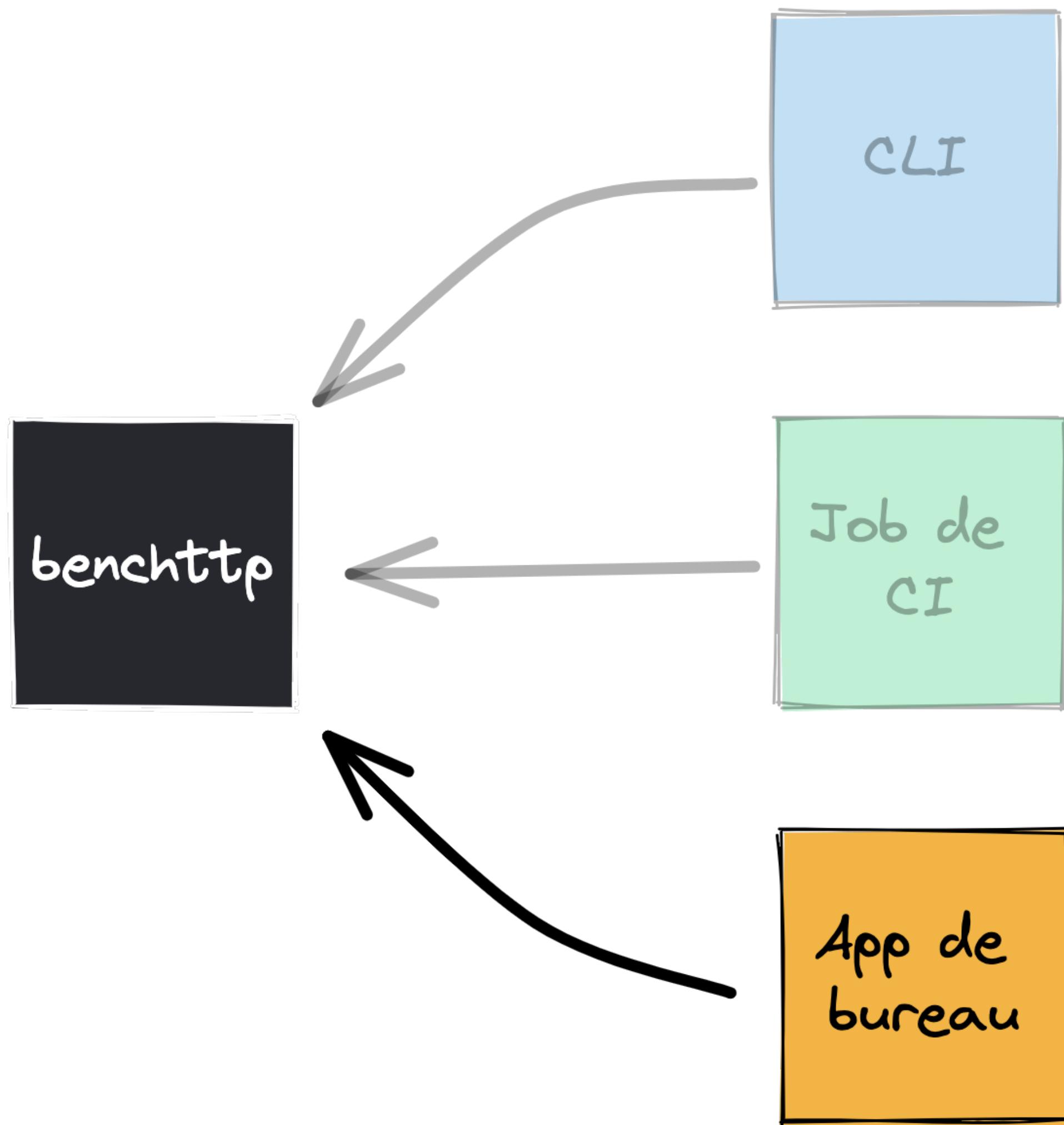
configurer, tester et sécuriser
tests d'acceptation et de non-régression automatiques

configurer, tester et visualiser/explorier
données statistiques et détails des tests

Utilisateurs cibles

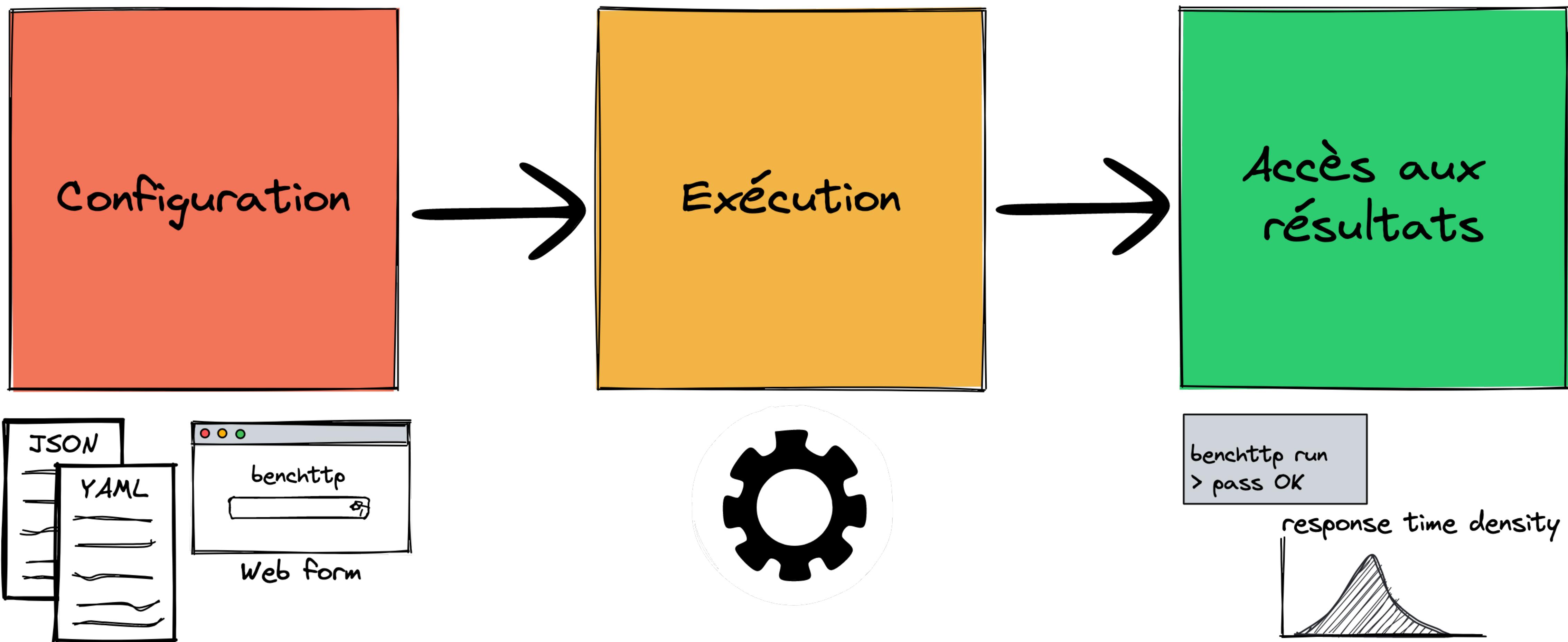


Utilisateurs cibles

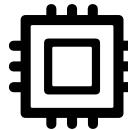


Périmètre du POC commercial : un parcours d'exécution complet

Un parcours d'exécution complet



Les implications techniques du POC

 moteur **opérationnel à 100%**

 gestion des **erreurs prévisibles**

 UX **claire**

 minimum de supposition sur le contexte d'utilisation : **non-opinionated**

→ la persistance de la donnée générée est considérée hors scope !

Démo live



Démo → [benchhttp/cli](#)

Démo → [benchhttp/action](#)

Démos à [github.com/benchhttp/cobaye/pulls](#)

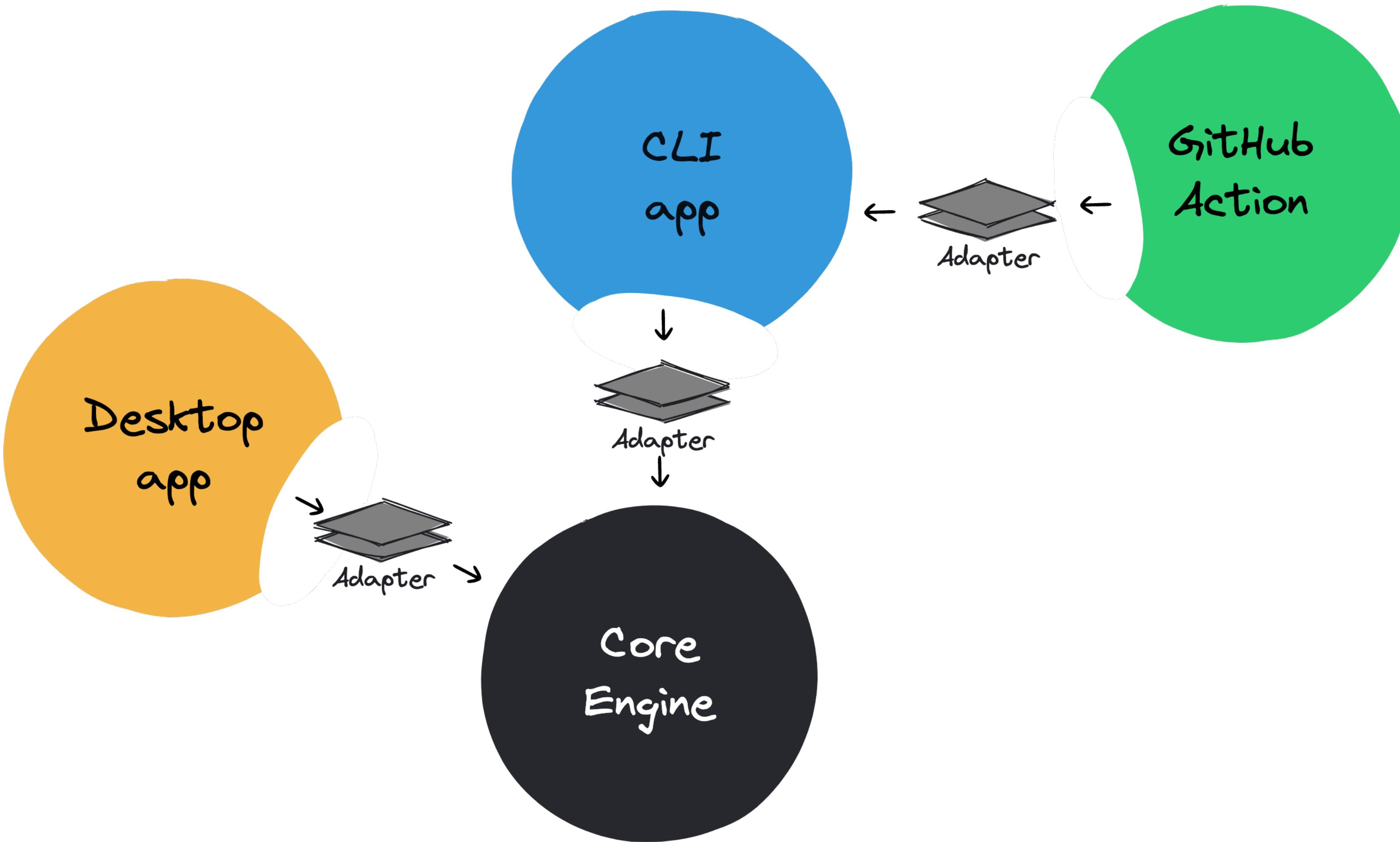
pass ✓ fail ✘

Démo → [benchhttp/desktop](#)

Technique



Architecture globale de la solution



**Choix technique :
exécution entièrement locale**

Exécution entièrement locale due à des problématiques métier

⟳ intégration au flow de **développement**

☰ besoins de **scalabilité imprévisibles**

✗ failles de **sécurité** et enjeux de **responsabilité**

- élimine les coûts d'infrastructure et de déploiement
- délègue les enjeux de sécurité et la responsabilité

**Challenge technique :
contrainte incontournable**

Challenge technique: les restrictions CORS

- ✖ Access to XMLHttpRequest at 'http://localhost:5000/global_config' [step1:1](#)
from origin '<http://localhost:8080>' has been blocked by CORS policy:
Response to preflight request doesn't pass access control check: No 'Access-Control-Allow-Origin' header is present on the requested resource.

Challenge technique : éliminer les restrictions CORS

**impossible de faire des requêtes depuis un navigateur vers un serveur qui
n'accepte pas l'origine de Benchtp**



Challenge technique : éliminer les restrictions CORS

impossible de faire des requêtes depuis un navigateur vers un serveur qui
n'accepte pas l'origine de Benchhttp

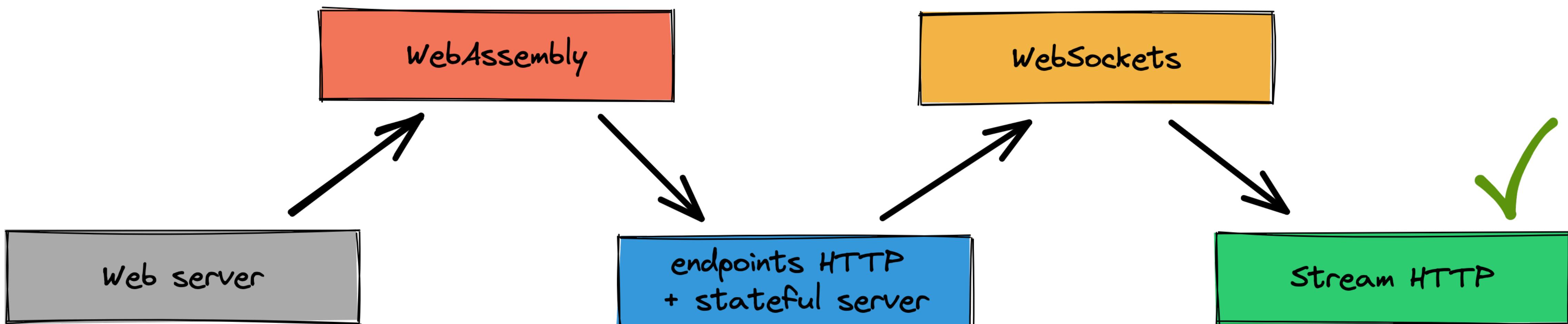


→ Notre utilisateur !

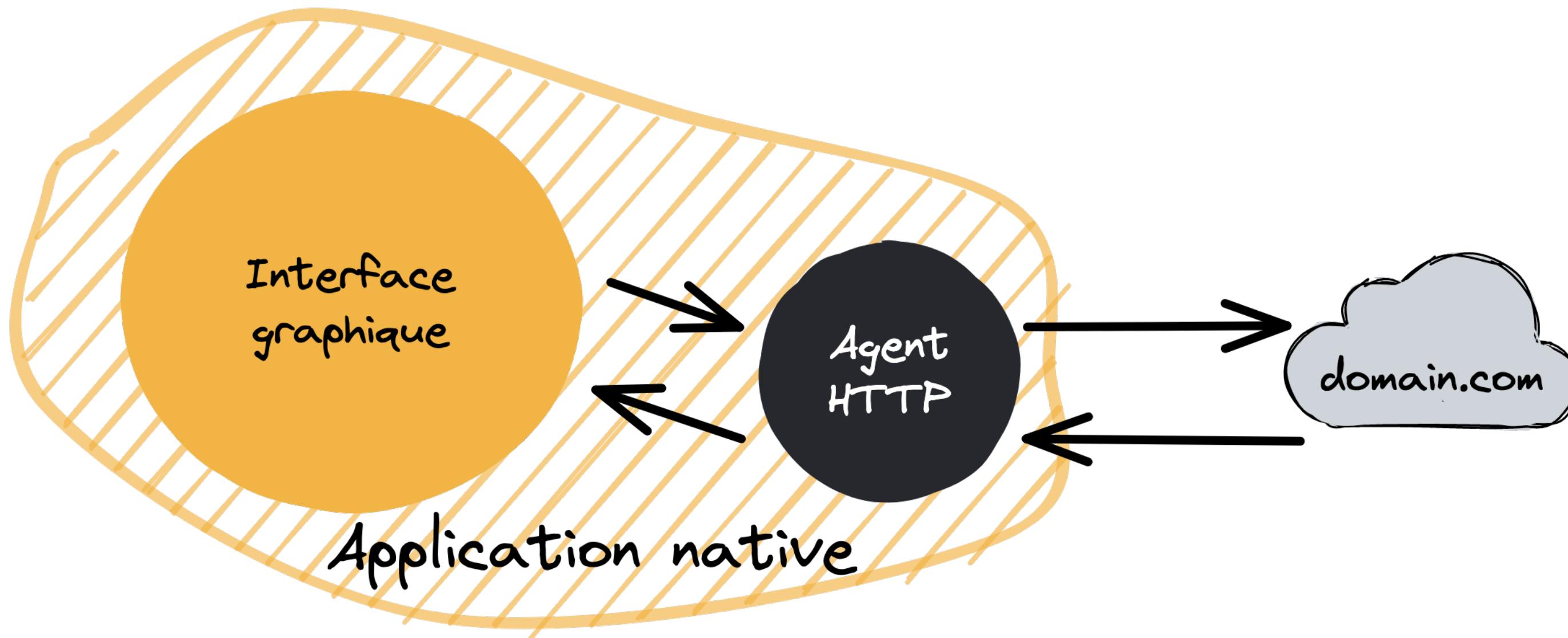
→ un agent HTTP doit faire les requêtes localement !

Comment faire communiquer l'application front-end avec l'agent HTTP local ?

→ POC successifs → affiner la solution



Conclusion technique : app desktop + serveur HTTP embarqué



→ installable sans aucune dépendance et multiplateforme

Roadmap

benchhttp@next

- améliorer l'UX sur le desktop
- récupérer les résultats de précédentes exécutions par configuration
- comparer différentes exécutions liées à une configuration donnée
- pouvoir configurer des niveaux de严重性 dans la suite de tests (ex. : `warn`)
- profiler l'utilisation hardware
- importer/exporter des configurations dans l'app desktop

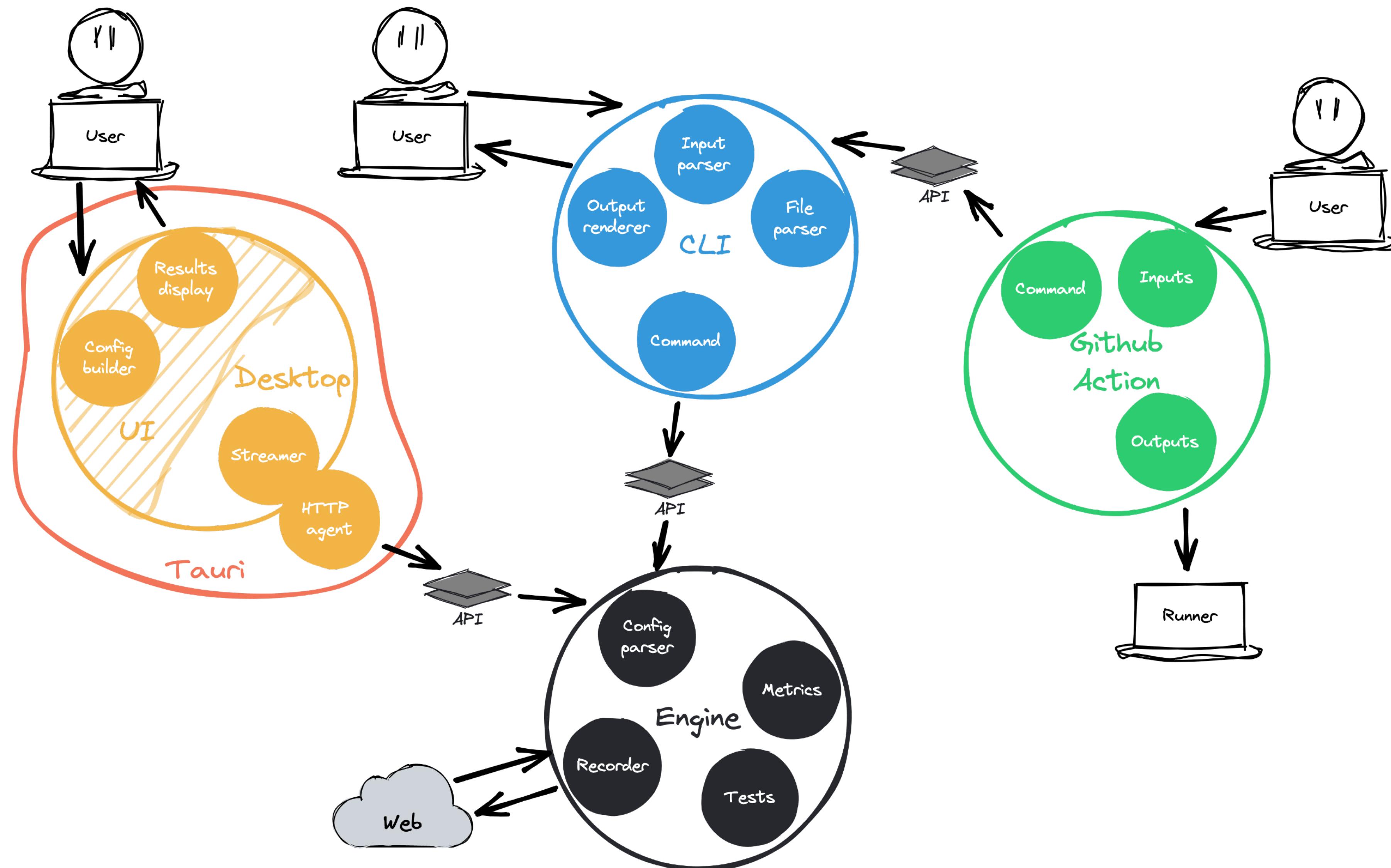
Annexes



Annexe :

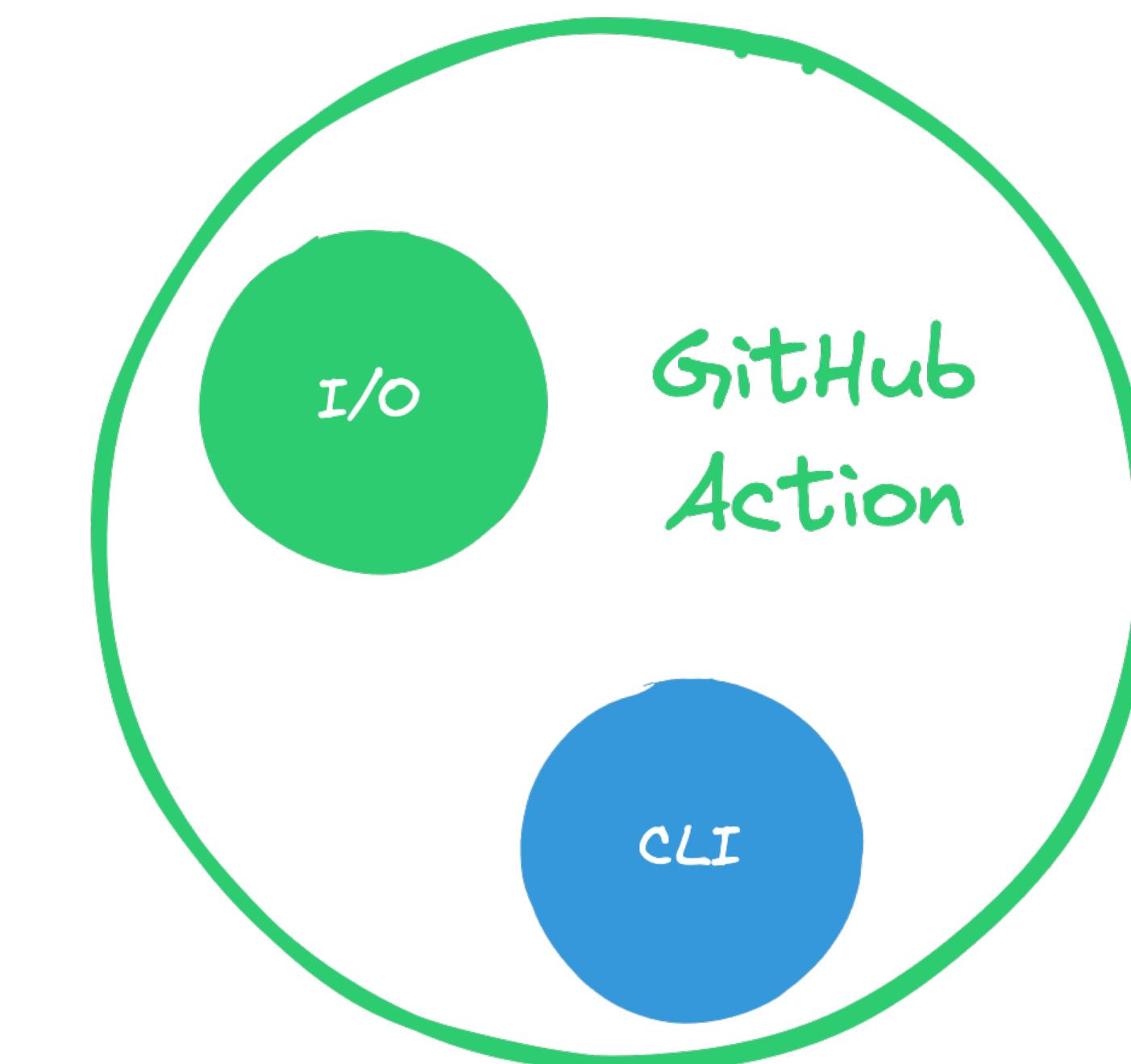
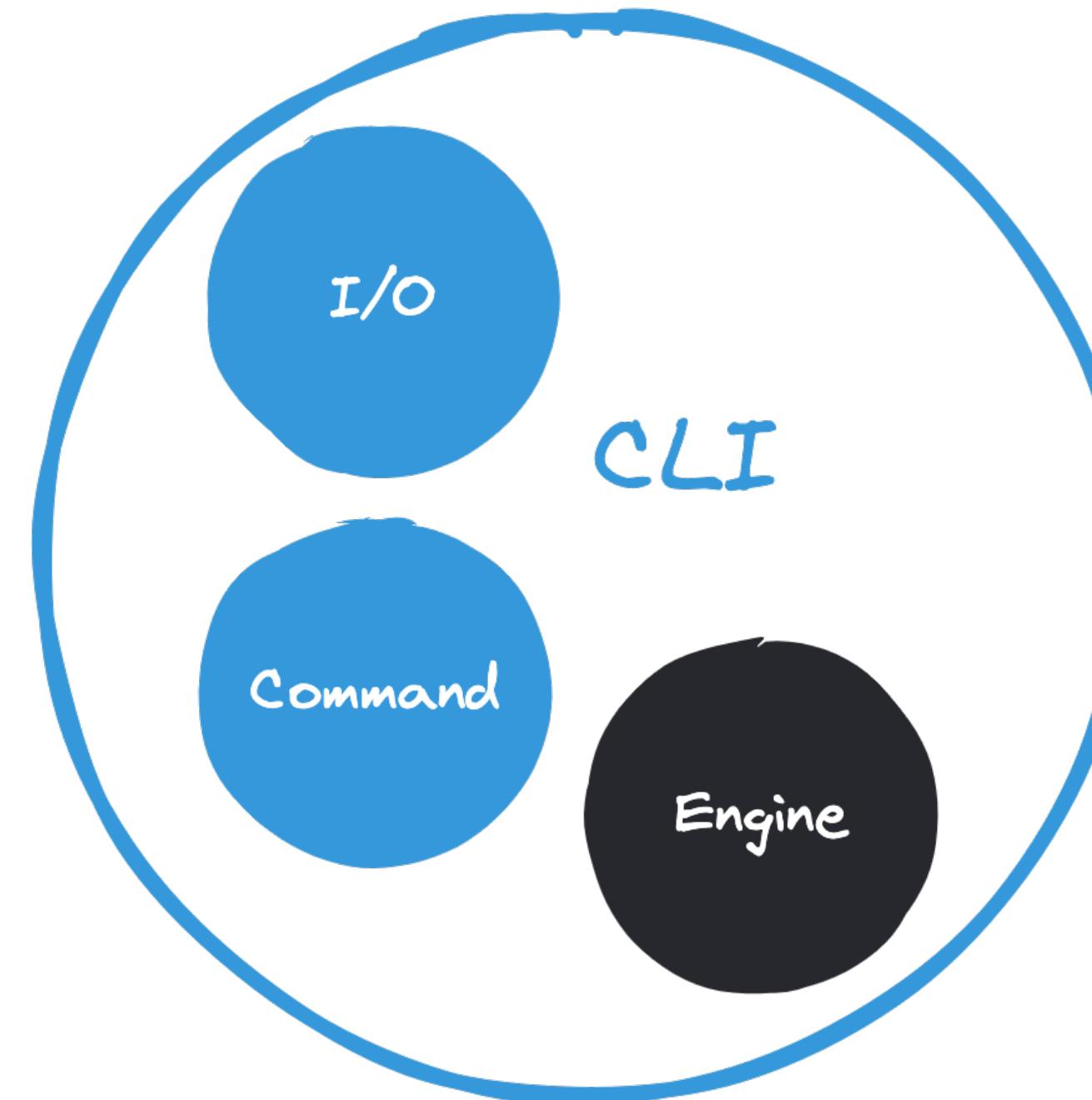
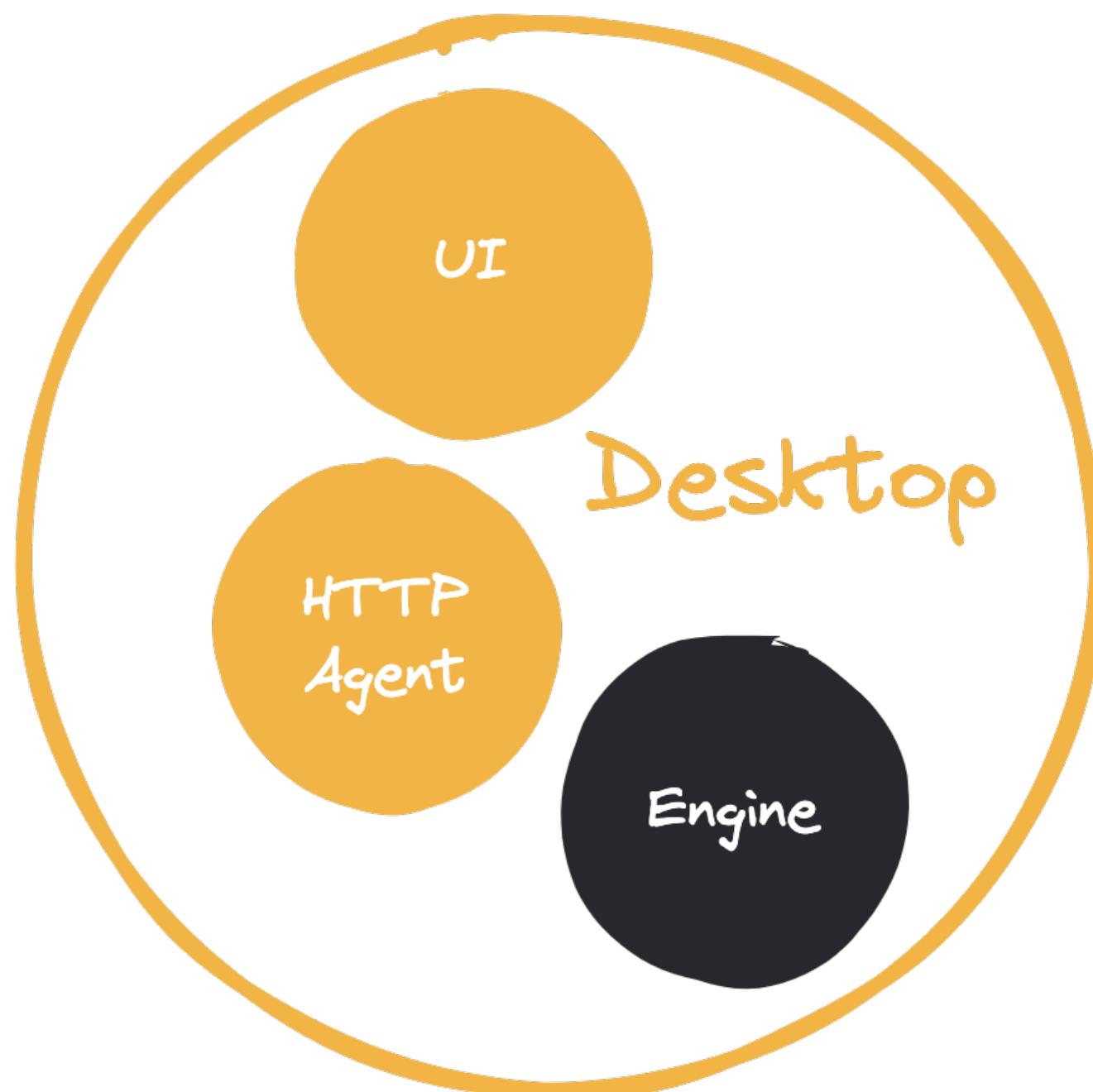
Architecture détaillée

Modularité interne



Functional core / imperative shell

Chaque app embarque le noyau dont elle dépend



Annexe : Stack technique

Core et CLI

Performance et portabilité

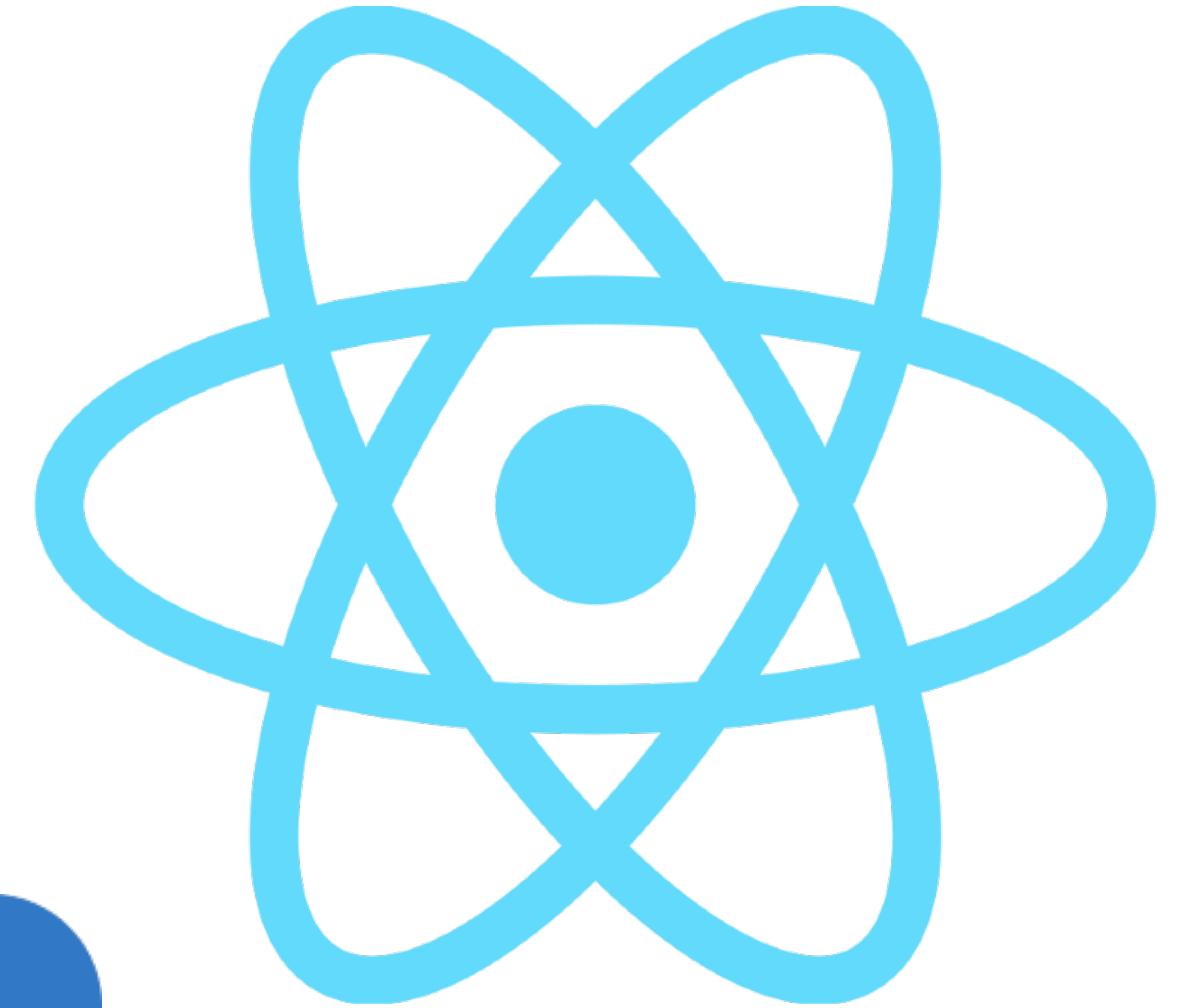
- performant sur les cas d'usage
- accès bas niveau sur le protocol
HTTP et hardware
- programmation concurrente incluse
- portatif via la compilation en binaire



Front-end : creation d'interfaces

Outils solides et écosystème riche

- compétences de l'équipe
- écosystème vaste, activement maintenu, largement éprouvé
- portabilité conséquente
(navigateur, app bureau)



Construction de l'application desktop

Compatible et transparent

- compatible React pour l'interface
- compile cross-platform
- embarquement d'un executable binaire via bindings Rust



Annexe :

Travail en collaboration et tooling

Travail en collaboration et tooling

- Développement
- spécifications et priorités définies sur GitHub Project et issues
- maquettage sur Figma
- PR liée à une issue, review requise
- **CI** : lint, test et build sur tout les dépôts GitHub
- **CI** : calcul de la couverture par les tests
- **CD** : versionnage, build et publication des exécutables
- **Communication**
- Discord dédié
- Notification auto sur Discord via webhook pour les événements GitHub

Travail en collaboration et tooling

