



Grille de revue des tests générés par l'IA

7 points pour relire un test Playwright écrit par un assistant (Copilot, Claude Code, Cursor) avant de le fusionner

Un test qui passe n'est pas forcément un bon test. Cochez chaque point pour chaque test généré ; un point non coché bloque la fusion, ou se justifie par écrit.

☐ 1. L'assertion vérifie un résultat métier

L'IA vérifie souvent qu'une action a eu lieu, pas ce que le métier attend.

AVANT

```
await page.getByRole('button', { name: 'Valider la commande' }).click();
await expect(page).toHaveURL(/\/commande/);
```

APRÈS

```
await expect(page.getByRole('heading', { name: 'Commande confirmée' })).toBeVisible();
await expect(page.getTestId('montant-total')).toHaveText('49,90 €');
```

Si la fonctionnalité renvoyait un mauvais résultat, ce test échouerait-il ?

☐ 2. Des locators accessibles, pas de CSS ou XPath fragile

AVANT

```
page.locator('div.form > div:nth-child(2) > input')
```

APRÈS

```
page.getByLabel('Adresse e-mail')
page.getByRole('button', { name: 'Se connecter' })
```

Le test casserait-il si on ajoutait une simple balise autour du formulaire ?

☐ 3. Des données de test dédiées, recrées à chaque exécution

Pas de compte ou de commande trouvés pendant l'exploration et partagés entre tests : créer les données (par l'API) dans un `beforeEach`.

Ce test passerait-il seul, ou en parallèle avec les autres ?

☐ 4. Les secrets restent hors du prompt et du code

AVANT

```
.fill('MotDePasse123!')
```

APRÈS

```
.fill(process.env.TEST_USER_PASSWORD!)
```

Exploration avec Playwright MCP : fichier `dotenv` passé avec `--secrets`.

Que verrait un inconnu qui lirait ce fichier ou l'historique de l'assistant ?

☐ 5. Jamais sur la production

Environnement de test, profil de navigateur isolé (`--isolated`), `baseUrl` lue depuis une variable d'environnement.

Sur quelle URL l'assistant a-t-il réellement travaillé, et avec quel compte ?

☐ 6. Une exploration limitée à un parcours

Une consigne précise (une page, un parcours, un résultat attendu) coûte moins de jetons et donne de meilleurs tests que « teste tout le site ».

La consigne tenait-elle en un parcours, avec un résultat attendu ?

☐ 7. Le test a échoué au moins une fois, volontairement

Fausser une fois le résultat attendu (ou casser la fonctionnalité) prouve que le test détecte une régression ; puis tout remettre en place.

Ai-je vu ce test en rouge au moins une fois, pour la bonne raison ?

Les 5 défauts à repérer en premier

- **Sélecteurs fragiles** : CSS ou XPath au lieu de `getByRole`, `getByLabel`, `getByTestId`.
- **Assertions vides** : l'action est vérifiée, pas son résultat.
- **Attentes fixes** : `waitForTimeout` au lieu d'une assertion qui réessaie (`toBeVisible`).
- **Données codées en dur** : comptes et montants recopiés depuis l'exploration.

- **Seul le cas nominal** : aucun cas d'erreur, de champ vide ou de valeur limite.