diginamic FORMATION
« Votre passeport pour l'emploi numérique »

Types de tests

Sommaire

🧠 Pourquoi classifier les tests ?

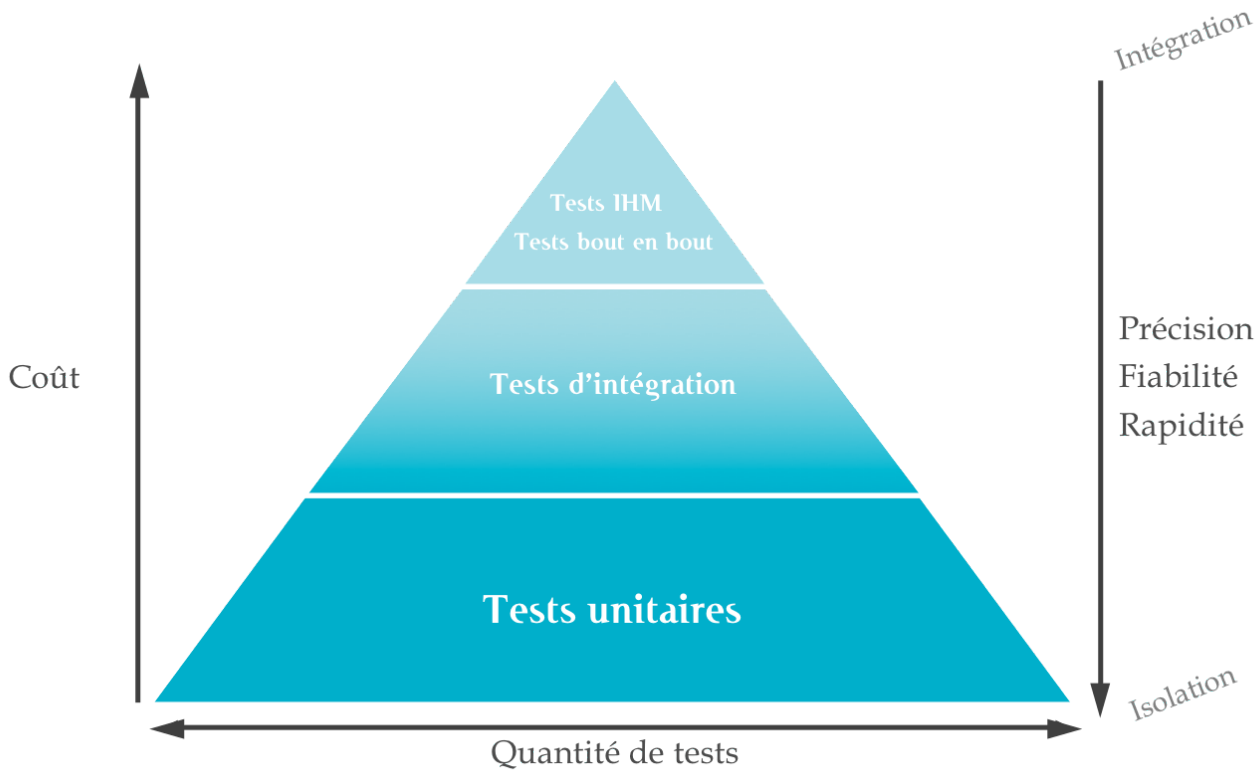
Tous les tests ne servent pas le même objectif. Certains valident le comportement d'une fonction isolée, d'autres valident l'interaction entre composants, voire l'expérience utilisateur globale.

La classification des tests permet de :

- structurer une stratégie de test efficace
- choisir le bon niveau de test selon le risque
- équilibrer qualité, coût et rapidité

🔺 La pyramide des tests

Une bonne stratégie de test s'appuie souvent sur une pyramide :



- **En bas**, les tests sont nombreux, rapides, peu coûteux : **tests unitaires**
- **Au milieu**, les tests sont moins nombreux mais plus riches : **tests d'intégration**
- **En haut**, les tests sont rares mais critiques : **tests end-to-end**

L'objectif est d'avoir une base solide et rapide (tests unitaires), soutenant quelques tests plus coûteux mais essentiels (intégration, E2E).

◇ Test unitaire

> 🔧 *Teste une brique de code isolée, comme une fonction ou une méthode.*

Objectifs

- Vérifier le comportement d'une fonction indépendamment du reste du système
- Travailler en isolation totale, sans base de données, API ou réseau
- Identifier précisément la source d'un bug

Caractéristiques

- Très rapides à exécuter
- Nombreux dans un projet bien structuré
- Ne dépendent d'aucun composant externe (mock si besoin)

Exemples typiques

- Vérifier qu'une fonction de calcul retourne le bon résultat
- Tester la logique métier d'une validation d'email

Avantages

- Faciles à écrire, à maintenir, à exécuter
- Premiers à s'exécuter en CI
- Excellents pour la couverture de code

◇ Test d’intégration

> 🔗 *Teste l’interaction entre plusieurs composants du système.*

Objectifs

- Vérifier que des composants communiquent correctement entre eux
- Détecter les erreurs liées à la configuration ou aux appels croisés

Caractéristiques

- Plus lents que les tests unitaires
- Peuvent impliquer une base de données, un serveur web, une authentification…
- Portent souvent sur des routes API, des appels HTTP, des transactions

Exemples typiques

- Tester qu’un utilisateur peut être enregistré via une requête POST
- Vérifier qu’une route FastAPI écrit bien dans la base de données

Avantages

- Très proches de l’usage réel
- Identifient des problèmes que les tests unitaires ne peuvent pas voir
- Essentiels pour valider l’architecture

◇ Test end-to-end (E2E)

> 🧍 *Teste l'application comme si on était un utilisateur réel.*

Objectifs

- Valider l'ensemble de la chaîne fonctionnelle
- Simuler un vrai comportement utilisateur sur l'interface finale

Caractéristiques

- Très coûteux en temps d'écriture et d'exécution
- Peuvent impliquer navigateur, base de données, back-end, authentification, etc.
- Parfois automatisés avec des outils comme Cypress, Playwright, Selenium

Exemples typiques

- Ouvrir un navigateur, se connecter à l'appli, remplir un formulaire, vérifier un affichage
- Tester un scénario complet (ex : "je m'inscris, je me connecte, je commande un produit")

Avantages

- Répliquent la réalité métier
- Captent des erreurs que tous les autres tests manquent
- Cruciaux pour la validation finale avant mise en production

◇ Autres types de tests (bonus)

Tests de performance

- Mesurent les temps de réponse, la charge supportée, la scalabilité
- Utiles pour les APIs ou les architectures distribuées

Tests de sécurité

- Vérifient qu'aucune faille évidente (ex : injection SQL, XSS) n'est exploitable
- Parfois semi-automatisés avec des outils spécialisés (OWASP ZAP, etc.)

Tests de non-régression

- Réexécutent d'anciens tests pour s'assurer que rien n'a été cassé par des modifications
- Souvent les mêmes que les tests unitaires/intégration, rejoués à chaque build

🧩 Synthèse comparative

Type de test	Vitesse ⚡	Portée 🔍	Coût 💰	Exemple typique
Unitaire	Très rapide	Très localisée	Très faible	Tester une fonction de validation
Intégration	Moyenne	Moyenne	Moyen	Tester un endpoint qui crée un objet
End-to-End (E2E)	Lente	Très large	Élevé	Simuler un parcours utilisateur complet