
takima
Je délègue tous mes tests à une IA
Valentin Dumas

Les Tests

Important

Vital

Long

“

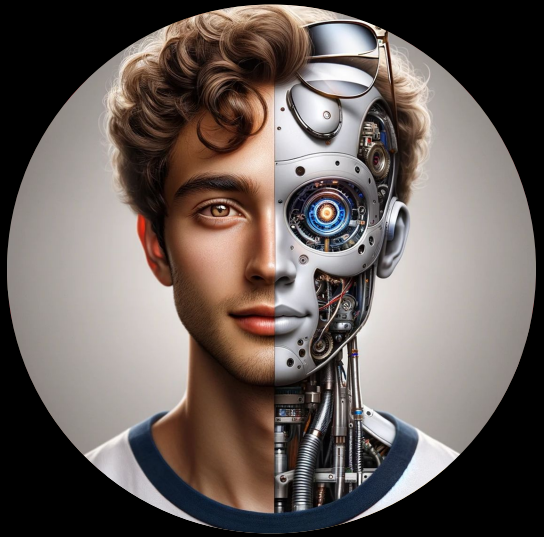
Je passe 25% de mon temps à faire du test

”

“

Vais-je pouvoir gagner 25% de productivité ?

”



Valentin Dumas

Dev Fullstack *Augmenté*

takima


Focus du jour

Focus du jour

~~*tests e2e, tests de charge, tests ui*~~

TU **TI**


Les solutions sur étagère

Je délègue mes tests à une IA

Solutions sur étagère

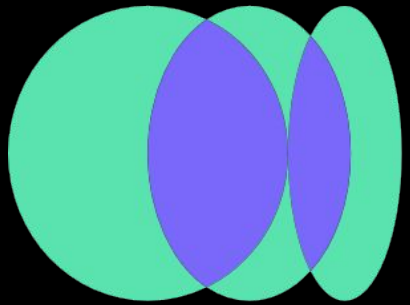
ChatGPT

Gemini

Github Copilot

freedev.ai

Cody AI



qodo

Qodo Gen

Codiumate : Comment ça fonctionne ?

“Generate tests”

“Refacto this class”

CONTEXTE PROJET

qodo

Codo Gen

GPT


Live Démo

Live Démo

Génération TI sur contrôleur

“ Generate integration tests for endpoints using rest template. Write whole class ”

```
@SpringBootTest(webEnvironment = SpringBootTest.WebEnvironment.RANDOM_PORT)
Codiumate: Add more tests
public class BankAccountControllerIT {
    @LocalServerPort
    private int port;
    @Autowired
    private TestRestTemplate restTemplate;
    @Autowired
    private JdbcTemplate jdbcTemplate;
    private String baseUrl;

    @BeforeEach
    public void setUp() {
        this.baseUrl = "http://localhost:" + port + "/accounts";
        jdbcTemplate.execute( sql: "TRUNCATE TABLE accounts");
    }
```

Live Démo

Génération TI sur contrôleur

“ Generate integration tests for endpoints using rest template. Write whole class ”

```
@Test
Codiumate: Add more tests
public void testGetAccountNotFound() {
    ResponseEntity<BankAccount> response = restTemplate.getForEntity(
            url: baseUrl + "/nonexistent", BankAccount.class);
    assertThat(response.getStatusCode()).isEqualTo(HttpStatus.NOT_FOUND);
}

@Test
Codiumate: Add more tests
public void testWithdrawNotFound() {
    String accountNumber = "nonexistent";
    double amount = 50.0;
    ResponseEntity<BankAccount> response = restTemplate.postForEntity(
            url: baseUrl + "/" + accountNumber + "/withdraw?amount=" + amount,
    assertThat(response.getStatusCode()).isEqualTo(HttpStatus.NOT_FOUND);
}
```


Bilan des courses

Bilan des courses

Qu’est-ce qu’on a fait ?

→ Génération TU

~~1 - 2 h~~

15 - 20 %

du temps

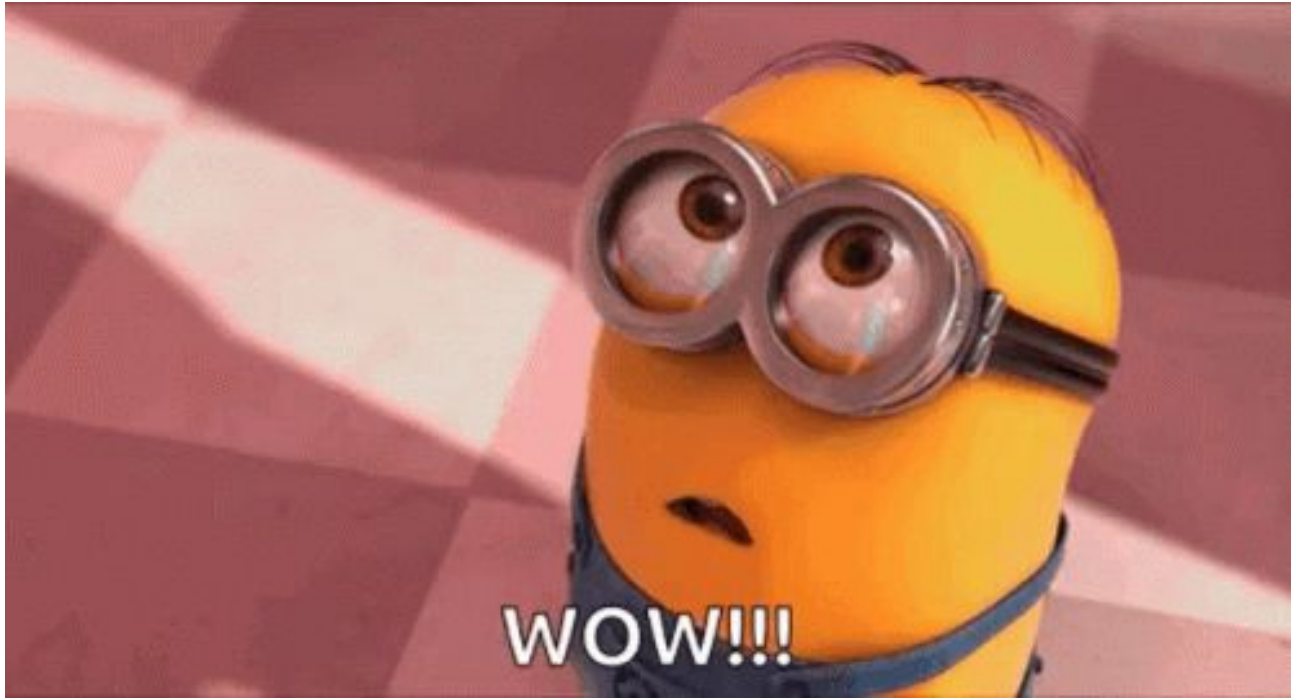
GLOBAL

→ Génération TI sur service

1 - 30 min

→ Génération TI sur contrôleur

“


WOW!!!

”

Ce qui est super

- → **Expérience Développeur**
- → Use-case **TU** bien adressé
- → **Fonctionnel** avec des librairies (Spring, AssertJ, URLs)
- → L'IA respecte les **conventions** et **sait mocker**
- → **Factorise** correctement

Les limites du système

Pour l'expérience du développeur

→ ça mériterait d'aller vers l'**autocomplétion** (IntelliJ)

→ pas d'option de **self-hosting** (LLM)

Je délègue mes tests à une IA

Les limites de Qodo Gen

Technique

Propriété Intellectuelle

Confidentialité

Comment je l'utilise au quotidien

C'est mon meilleur **générateur** de **boilerplate**

- → Pour un nouveau projet / service
- → Pour une nouvelle classe de test

C'est un **vrai assistant** pour les **TU / TI**

- → Pour le refactoring / use-case complexes
- → Suggestions de code

“

Oui

”

“

On peut au moins gagner 15% de notre temps !

”

“

On peut déjà intégrer des IAs génératives pour gagner du temps sur le test

”

“

Cependant..

”

Ces approches ne remplacent aucunement la nécessité de bien appréhender leur “architecture”

(enfin, pour l’instant)

“

L’archi c’est pour moi,

Le boilerplate pour L’IA

”

Merci de votre attention

Valentin Dumas

Dev Fullstack *Augmenté*

takima