2021

Stratégie de test

Karl Menino

Version : 1.0

12/10/2021

OPENCLASSROOMS

Table des matières

Table des matières

1. HISTORIQUE DES REVISIONS

Version	Description	Date	Validateur
1.0		12/10/2021	Menino karl

Ce document décrit la stratégie de tests.

Karl menino,

Architecte Logiciel,

2. Stratégie de tests

A. Test unitaire

Pour l'approche général que nous allons mettre en place pour nos tests, nous allons utiliser la méthode F.I.R.S.T.

F pour fast (rapides). Nous devons faire des tests unitaires qui ne test qu'une seule classe et qui doit être rapide (Càd quelques millisecondes). Cela afin de pouvoir faire des centaines, voir des milliers, de test par seconde. En cas de ralentissement du au matériel ou au réseau, nous utiliserons un Mock.

I pour isoler et indépendant. Nous devons avoir les tests les plus indépendants possible, donc qu'aucuns tests ne dépendent du résultat d'un autre test. Le but étant que nous voulons rendre notre test reproductible et éviter d'avoir des effets de bord. Les tests peuvent facilement échouer pour des raisons externes quand ils sont liés par une dépendance partagée. Contrairement au code de l'application, lorsque nous testons, il peut être acceptable de nous répéter, si cela préserve l'isolement de nos tests. D'ailleurs, Les principes AAA ou GIVEN/WHEN/THEN peuvent nous aider ici : chaque test organise sa propre classe de sous-tests, et ne fait qu'une assertion par test. Cela garantit que nos tests utilisent des données séparées.

R pour répétable. Si nous écrivons un test qui nous donne la confiance nécessaire en notre code, il doit nous dire la même chose, peu importe où, ou combien de fois, nous l'exécutons. Nos tests doivent aussi être reproductibles quel que soit l'environnement d'exécution.

S pour self-validating (auto validation). Cela signifie en fait que l'exécution de nos tests ne laisse aucun doute sur leur succès ou leur échec. En utilisant un Framework de test comme JUnit, des bibliothèques d'assertions, et en écrivant des tests spécifiques, nous pouvons garantir qu'en cas d'échec d'un test, vous aurez des rapports clairs et sans ambiguïté qui vous diront exactement ce qui a réussi ou échoué.

T pour thorough (approfondi). Nos tests devraient être écrits au plus près possible du moment où nous écrivons le code. En écrivant nos tests quand nous écrivons votre code, le test et le code peuvent être conçus pour respecter le F.I.R.S.T. Notre code doit être testé largement pour des cas négatifs et positifs. Étant donné que la meilleure manière d'écrire des tests approfondis est de s'assurer d'avoir écrit du code largement testable, ces deux aspects tendent vers le même résultat. Pour piloter la conception de notre code avec le TDD, et établir la base de notre pyramide de tests, nous devons nous poser certaines des questions suivantes :

- Est-ce que j'ai un test de scénario nominal pour chaque cas que j'ai codé ?
- Est-ce que j'ai pensé aux scénarios alternatifs et cas limites ? Et si la date de naissance d'un vampire était après sa mort, en raison de sa résurrection ? Est-ce que mon système pourrait le gérer ?
- Est-ce que chacune de mes exceptions lancées est testée ?
- Est-ce qu'il existe un scénario où, sans changer le type de données utilisé dans mon test, je peux causer un comportement inattendu ? Que se passera-t-il si je passe une chaîne nulle ou vide ?
- Est-ce que j'ai pensé à la sécurité en priorité ? Malheureusement, ce point est toujours le dernier de la liste. Est-ce que ce code peut uniquement être exécuté par les utilisateurs qui ont le droit de le faire ? Et si ce n'est pas le cas ?

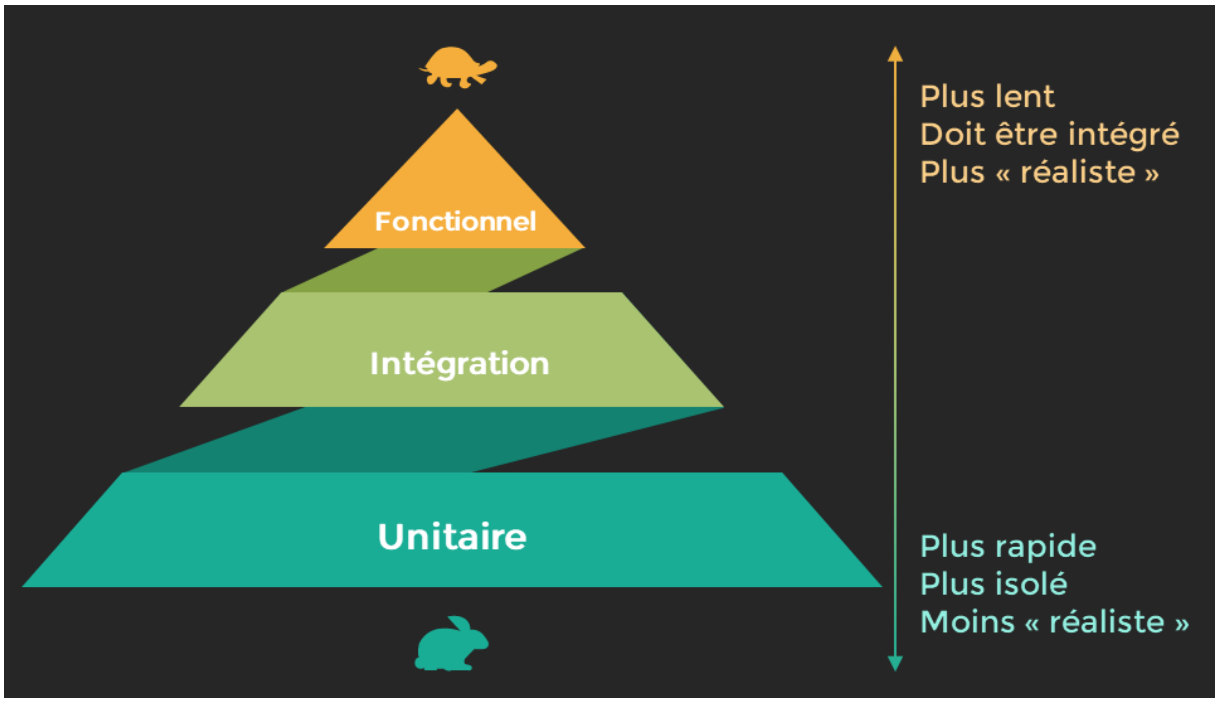
B. Nom de nos tests

Les conventions de nommage sont indissociables du F.I.R.S.T. et sont extrêmement importantes pour construire du code lisible. Nous avons décidé

d'utiliser le Camel case pour chaque élément à décrire lier par des caractères Under score

C. Test d'intégration et fonctionnel

Voici le design pattern concernant nos tests :



Nous pouvons remarquer que le milieu et le sommet de la pyramide est composer de nos tests fonctionnels et d'intégration. Cela veut dire que généralement, il y aura moins de test d'intégration que de test unitaire et moins de test fonctionnel que de test d'intégration.

Il existe deux types de test d'intégration :

- **Les tests d'intégration composants** : ils permettent de vérifier si plusieurs unités de code fonctionnent bien ensemble, dans un environnement de test assez proche du test unitaire, c'est-à-dire de manière isolée, sans lien avec des composants extérieurs et ne permettant pas le démarrage d'une vraie application ;
- **Les tests d'intégration système** : ils permettent de vérifier le fonctionnement de plusieurs unités de code au sein d'une configuration d'application, avec éventuellement des liens avec des composants extérieurs comme une base de données, des fichiers, ou des API en réseau.

Les tests fonctionnels de bout en bout sont des tests qui partent de l'interface utilisateur pour obtenir un résultat selon un scénario prédéfini. Ils imitent l'utilisateur final de l'application. Un démarrage complet de l'application est donc nécessaire.

3. Résultats

A. Rapport SUREFIRE

Dernière publication: 2021-10-07 | Version: 0.0.1-SNAPSHOT

Produit par Maven

Surefire Report

Summary

[Summary] [Package List] [Test Cases]

Tests	Errors	Failures	Skipped	Success Rate	Time
43	0	0	0	100%	12,894

Note: failures are anticipated and checked for with assertions while errors are unanticipated.

Package List

[Summary] [Package List] [Test Cases]

Package	Tests	Errors	Failures	Skipped	Success Rate	Time
com.openclassroom.p11.dao	14	0	0	0	100%	1,749
com.openclassroom.p11.model	5	0	0	0	100%	0,066
com.openclassroom.p11	1	0	0	0	100%	0,002
com.openclassroom.p11.controller	3	0	0	0	100%	9,591
com.openclassroom.p11.manager	13	0	0	0	100%	0,68
com.openclassroom.p11.model.jsonModel	4	0	0	0	100%	0,048
com.openclassroom.p11.service	3	0	0	0	100%	0,758

Note: package statistics are not computed recursively, they only sum up all of its testsuites numbers.

com.openclassroom.p11.dao

	Class	Tests	Errors	Failures	Skipped	Success Rate	Time
	HistoriquePathologiesDaoTest	1	0	0	0	100%	1,551
	HopitalDaoTest	4	0	0	0	100%	0,068
	LitDaoTest	1	0	0	0	100%	0,031
	PatientDaoTest	4	0	0	0	100%	0,058
	SpecialiteDaoTest	4	0	0	0	100%	0,041

com.openclassroom.p11.model

	Class	Tests	Errors	Failures	Skipped	Success Rate	Time
	HistoriquePathologiesTest	1	0	0	0	100%	0,012
	HopitalTest	1	0	0	0	100%	0,025
	LitTest	1	0	0	0	100%	0,008
	PatientTest	1	0	0	0	100%	0,011
	SpecialiteTest	1	0	0	0	100%	0,01

com.openclassroom.p11

	Class	Tests	Errors	Failures	Skipped	Success Rate	Time
	P11ApplicationTests	1	0	0	0	100%	0,002

com.openclassroom.p11.controller

	Class	Tests	Errors	Failures	Skipped	Success Rate	Time
	ControllerApiRestTest	3	0	0	0	100%	9,591

com.openclassroom.p11.manager

	Class	Tests	Errors	Failures	Skipped	Success Rate	Time
	ApiManagerTest	2	0	0	0	100%	0,538
	HistoriquePathologiesManagerTest	1	0	0	0	100%	0,037
	HopitalManagerTest	2	0	0	0	100%	0,023
	PatientManagerTest	4	0	0	0	100%	0,042
	SpecialiteManagerTest	4	0	0	0	100%	0,04

com.openclassroom.p11.model.jsonModel

	Class	Tests	Errors	Failures	Skipped	Success Rate	Time
	InfoHopitalTest	1	0	0	0	100%	0,016
	InfoPatientTest	1	0	0	0	100%	0,01
	LocalisationPatientTest	1	0	0	0	100%	0,012
	ReponseRdvTest	1	0	0	0	100%	0,01

com.openclassroom.p11.service

	Class	Tests	Errors	Failures	Skipped	Success Rate	Time
	InfoHopitalProcheServiceTest	1	0	0	0	100%	0,225
	InfoPatientServiceTest	1	0	0	0	100%	0,023
	PriseRdvServiceTest	1	0	0	0	100%	0,51

Test Cases

[Summary] [Package List] [Test Cases]

ControllerApiRestTest

	priseRDV	2,485
	infoHopital	0,297
	patient	0,051

HistoriquePathologiesDaoTest

	saveAndDelete	0,05

HopitalDaoTest

	save	0,007
	findAllByLattitudeBetweenAndLongitudeBetween	0,01
	notFindAllByLattitudeBetweenAndLongitudeBetween	0,016
	delete	0,022

LitDaoTest

	saveAndDelete	0,022

PatientDaoTest

	save	0,007
	findPatientByNumero	0,008
	notFindPatientByNumero	0,008
	delete	0,022

SpecialiteDaoTest

	save	0,004
	findByNom	0,007
	notFindByNom	0,006
	delete	0,013

ApiManagerTest

	localiserPatientGps	0,049
	notLocaliserPatientGps	0,486

HistoriquePathologiesManagerTest

	save	0,024

HopitalManagerTest

	hopitalProche	0,005
	notHopitalProche	0,008

PatientManagerTest

	save	0,001
	findByNumber	0,005
	notFindPatientByNumero	0,013
	delete	0,012

SpecialiteManagerTest

	save	0,004
	findByName	0,006
	notFindByName	0,005
	delete	0,013

HistoriquePathologiesTest

	getterAndSetter	0,001

HopitalTest

	getterAndSetter	0,003

InfoHopitalTest

	getterAndSetter	0,002

InfoPatientTest

	getterAndSetter	0

LocalisationPatientTest

	getterAndSetter	0

ReponseRdvTest

	getterAndSetter	0

LitTest

	getterAndSetter	0

PatientTest

	getterAndSetter	0

SpecialiteTest

	getterAndSetter	0

P11ApplicationTests

	contextLoads	0

InfoHopitalProcheServiceTest

	infoHopital	0,214

InfoPatientServiceTest

	infoPatient	0,013

PriseRdvServiceTest

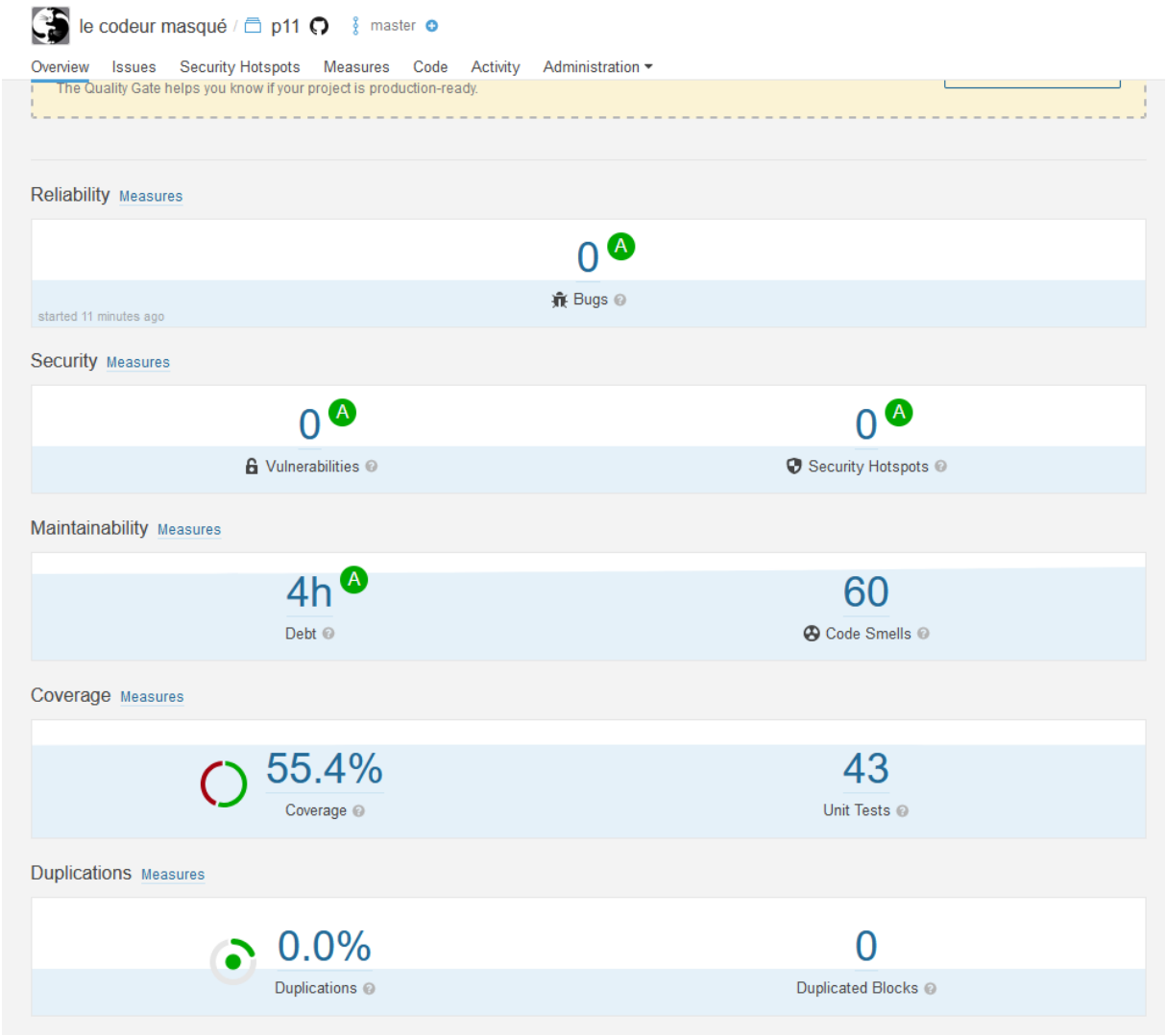
	priseRdv	0,503

B. Rapport JACOCO

p11

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
com.openclassroom.p11.model.jsonModel		29 %		13 %	76	119	0	21	12	54	0	4
com.openclassroom.p11.model		63 %		21 %	27	94	43	143	11	78	0	5
com.openclassroom.p11.controller		43 %		0 %	3	7	18	25	2	6	0	1
com.openclassroom.p11.filtreApi		55 %		50 %	2	4	6	12	0	2	0	1
com.openclassroom.p11.service		92 %		71 %	4	14	2	49	0	7	0	3
com.openclassroom.p11		37 %		n/a	1	2	2	3	1	2	0	1
com.openclassroom.p11.manager		100 %		n/a	0	14	0	36	0	14	0	5
com.openclassroom.p11.configuration		100 %		n/a	0	5	0	21	0	5	0	2
Total	767 of 1 783	56 %	146 of 182	19 %	113	259	71	310	26	168	0	22

C. Rapport SONNARCLOUD



4. Scénarios BDD

A. Trouver l'hôpital le plus proche

Scenario :

Given depuis ma localisation

When je recherche l'hôpital le plus proche en fonction de l'adresse, du rayon et de la pathologie

Then l'hôpital le plus proche mes proposés avec la spécialité demandée

B. Prendre Rendez vous

Scenario :

Given depuis ma localisation

When je recherche l'hôpital le plus proche en fonction de mes coordonnées et de la spécialité demandé

Then l'hôpital le plus proche mes proposés avec la spécialité demandée et mes données sont enregistré si je ne suis pas dans la base de données

C. Trouver les Informations patients

Scenario :

Given Je Veux les informations d'un patient

When je recherche les infos patientes en fonction de son numéro de sécurité sociale

Then l'application me renvoie l'historique des pathologie patients

D. Supprimez patient

Scenario :

Given un patient demande la suppression de ces informations en lien avec ces droits RGPD

When je supprime les infos patientes en fonction de son numéro de sécurité social

Then toutes les données patients sont supprimer.