Hachelef Asma, Blandin Pernelle, Demirtay Emrecan, Gautier--Gocko Erwan

Rapport de tests SAÉ 2.01

-Développement d'une application pour Livre Express-

Année universitaire 2024-2025

Texte repris en partie de notre rapport final

Les tests unitaires permettent de vérifier que des fonctions retournent bien le résultat voulu, selon des paramètres définis au préalable. Ils permettent de vérifier le bon fonctionnement de notre programme, malgré les changements apportés au fur et à mesure du temps.
Pour leur mise en place, nous avons suivi la méthode TDD, *Test Driven-Development*, vu lors des cours de qualité de développement durant le deuxième semestre. Nous avons tout d'abord réalisé les tests unitaires avant de développer les fonctions nécessaires dans la mesure du possible.
Certaines fonctions ou classes ne sont pas testables, car elles demandent une entrée ou affichent des informations à l'utilisateur. De plus, nous ne pouvons pas tester les classes en relation directe avec la base de données, ce qui correspond à un certain nombre de fonctions de notre modèle. Notre programme permet d'ajouter des livres et des clients dans la base de données, ce qui pourrait fausser directement nos tests unitaires si nous récupérons les données dans celle-ci.

Pour nous aider à l'écriture des tests unitaires, nous avons utilisé l'outil Coverage, inclus dans Visual Studio Code. Il nous permet de voir l'ensemble des méthodes qui doivent être testées et leur avancement, comme le montre la figure n°6.
De plus, dans le cas où une fonction contiendrait des conditions spécifiques et qu'elles ne sont pas testées, Coverage nous le signale, permettant ainsi d'améliorer la pertinence de nos tests unitaires.

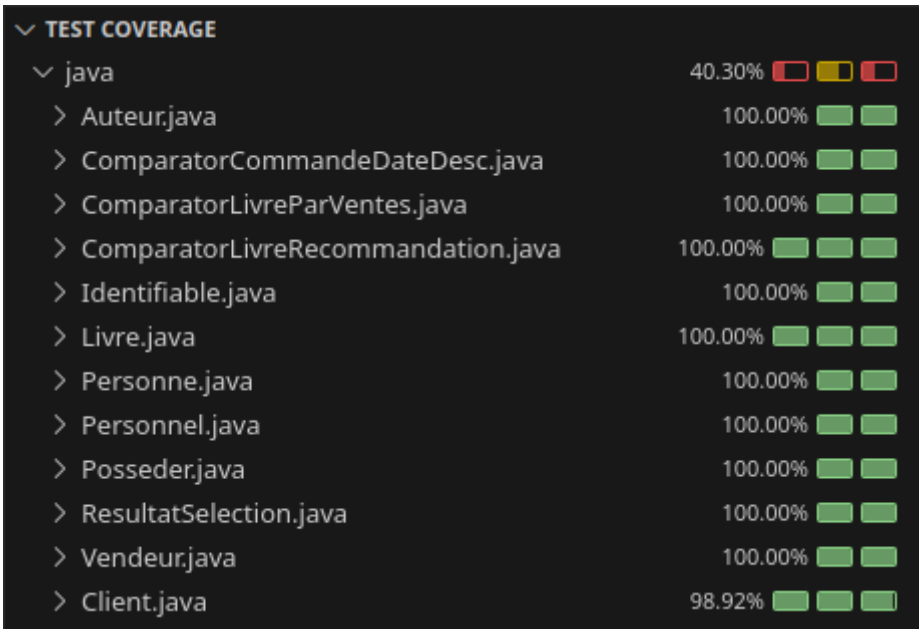

Figure n°1 : L'outil Coverage dans Visual Studio Code