Plan de test

1. Périmètre des tests

Les tests du projet sont utilisés pour vérifier avant tout le bon fonctionnement des vues.
On peut distinguer les tests en quatre catégories :

- Les tests simples de retour d'un code réponse HTML. Cela concerne toutes les vues du projet : on vérifie via des tests unitaires (unittest) que les codes HTML renvoyés lors des tests sont ceux attendus, cela permet de vérifier le bon fonctionnement du site et de la gestion des erreurs.
- Les tests visant à évaluer les interactions entre les vues et la base de données et le système d'authentification, également réalisé avec unittest. Sont testés les processus d'authentification (login/logout/register) et la vue de comparaison.
- Les tests visant le bon fonctionnement de la partie script de l'application, qui concerne l'algorithme de comparaison, effectués également avec unittest.
- Les tests fonctionnels, via Selenium, permettent non seulement de tester que l'ajout aux favoris fonctionne correctement mais permet également de vérifier une fois de plus le processus d'authentification et l'association d'un profil à l'utilisateur.

  La combinaison de tests unitaires et fonctionnels permet en théorie d'ajouter de nouvelles vues au besoin avec un effort minimal pour l'adaptation des tests à ces nouvelles vues.

2. Limites des tests

Les tests évaluent la totalité des vues du site mais ne couvrent pas les performances du projet. Les fonctions de management ne sont également pas couvertes ainsi que les fonctions principales assurées par Django (la sécurité notamment).

3. Outils de test

Les tests unitaires sont effectués via unittest, la librairie de test par défaut de Django. La compatibilité des deux outils est particulièrement utile d'autant que Django est capable de créer et détruire une base de données spécifiquement utilisée pour les tests. Ce qui facilite fortement les tests sur la base de données.
Selenium est un excellent outil de test pour les opérations plus complexes, pour simuler un parcours que l'utilisateur est susceptible de prendre et qu'il est important de tester et maintenir fonctionnel.

4. Exécution des tests

Les tests sont exécutés simultanément grâce à la commande 'test' de Django(manage.py). Cela permet une vue d'ensemble sur l'état de tous les tests. La possibilité d'activer ou non la verbosité des tests est un atout lors de la construction de ces derniers pour corriger les éventuelles erreurs.