Fiche d’investigation de fonctionnalité

Fonctionnalité : Recherche de recettes	**Fonctionnalité #2**

Problématique :

Permet de trier parmi les recettes disponibles en prenant en compte plusieurs paramètres :

- Un champ de recherche principal (optionnel)
- Trois champs de sélection avancés (ingrédients, appareils et ustensiles. Optionnels)

La recherche renverra les recettes dont le nom, la description ou les ingrédients contiennent la recherche via la barre de recherche principale, et dont l’appareil, les ingrédients et les ustensiles correspondent aux menus de sélection avancés (tags).

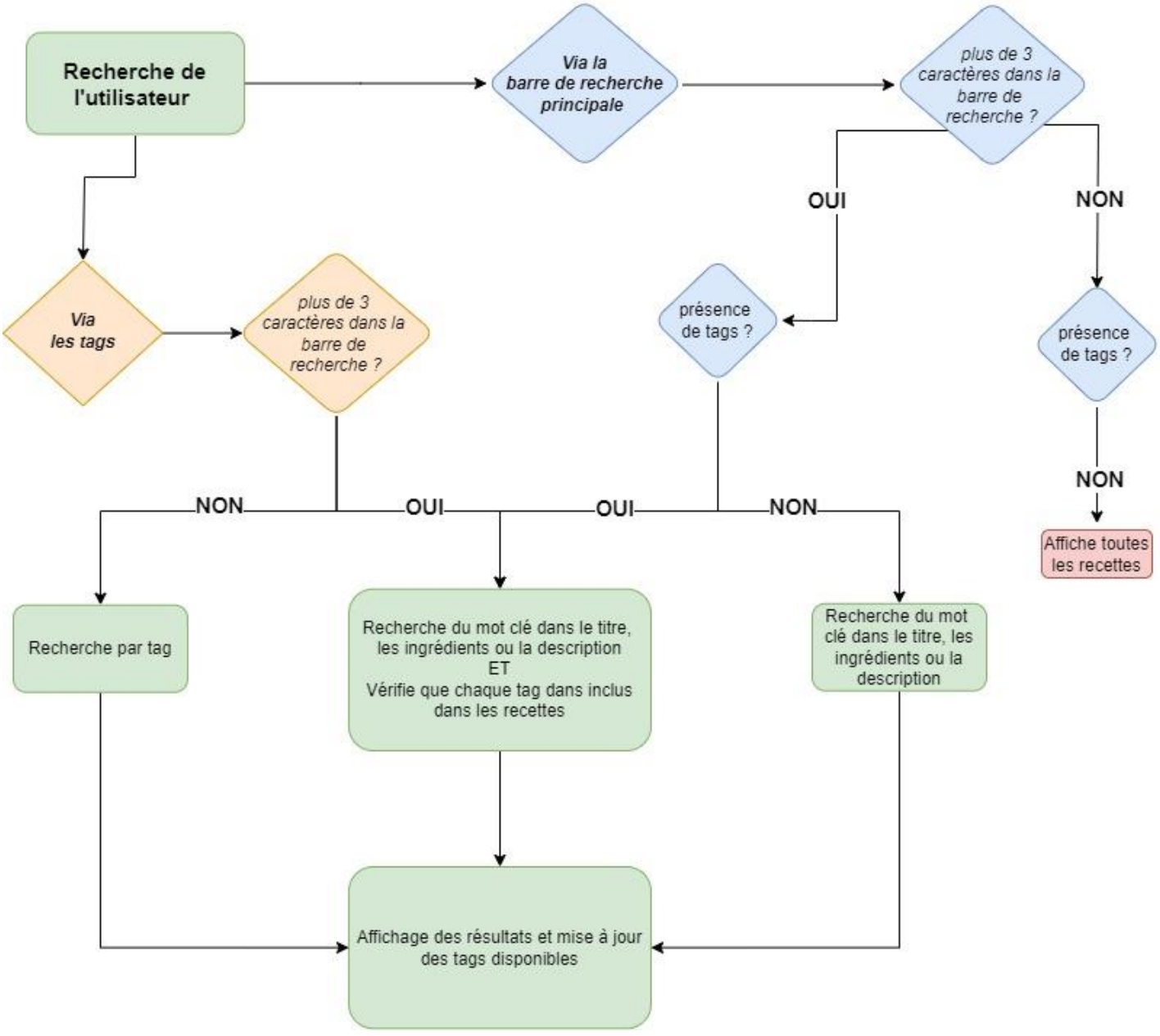
Afin de fournir la meilleure expérience utilisateur, j’ai testé deux algorithmes de recherche différent :

- Via la boucle native for ... of
- Via la méthode reduce()

Option 1 : Boucle native for...of	
Avantages : • Simple à comprendre et à mettre en place • Lisible et maintenable • Très performant si le site comporte beaucoup de recettes (à partir de 200)	**Inconvénients :** • Moins performant si pas beaucoup de recettes (moins de 200)

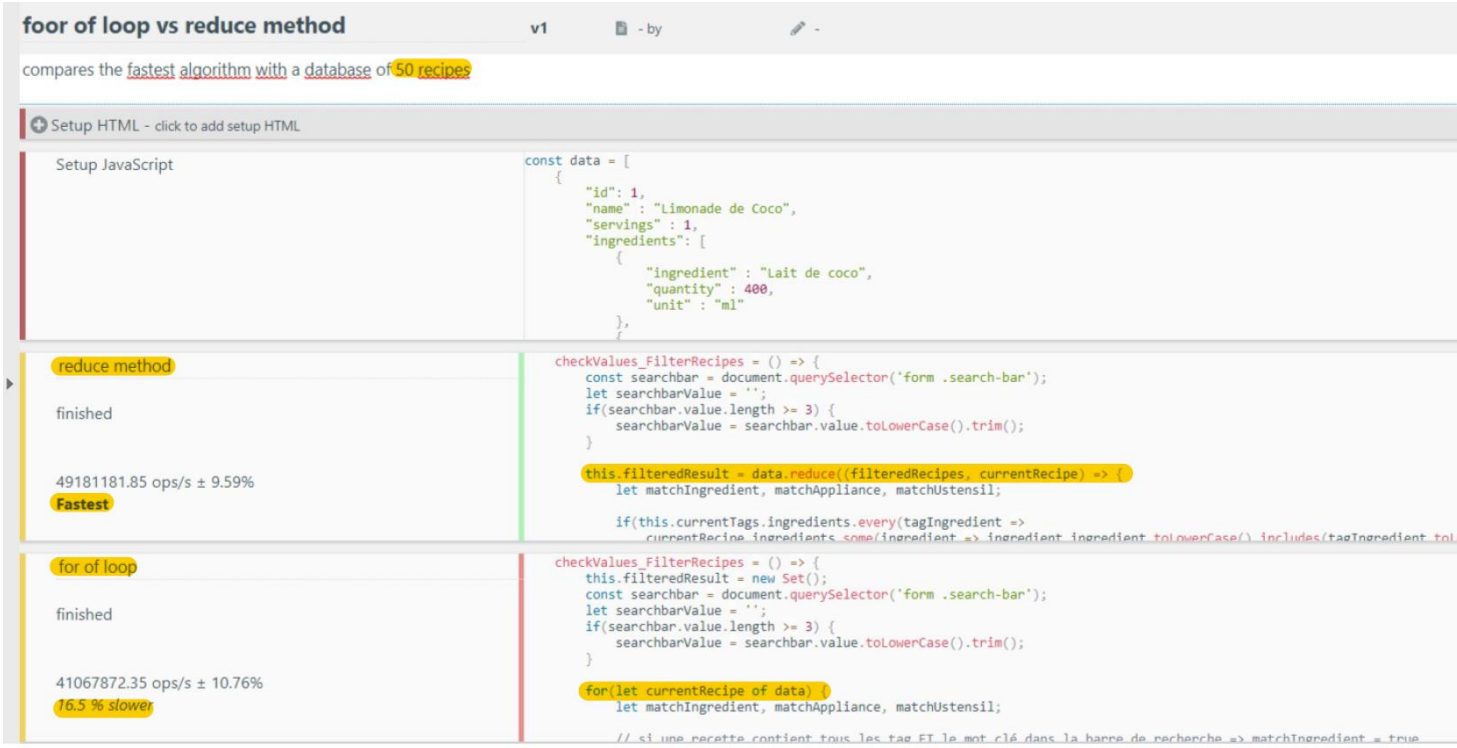
Option 2 : Méthode de l’objet Array reduce	
Avantages : • Globalement simple à comprendre et à mettre en place • Lisible et maintenable • Performant si le site comporte peu de recettes.	**Inconvénients :** • Moins facile à mettre en place et à comprendre pour certains (nouveauté de ES6) • Problème de scalabilité

Algorigramme :



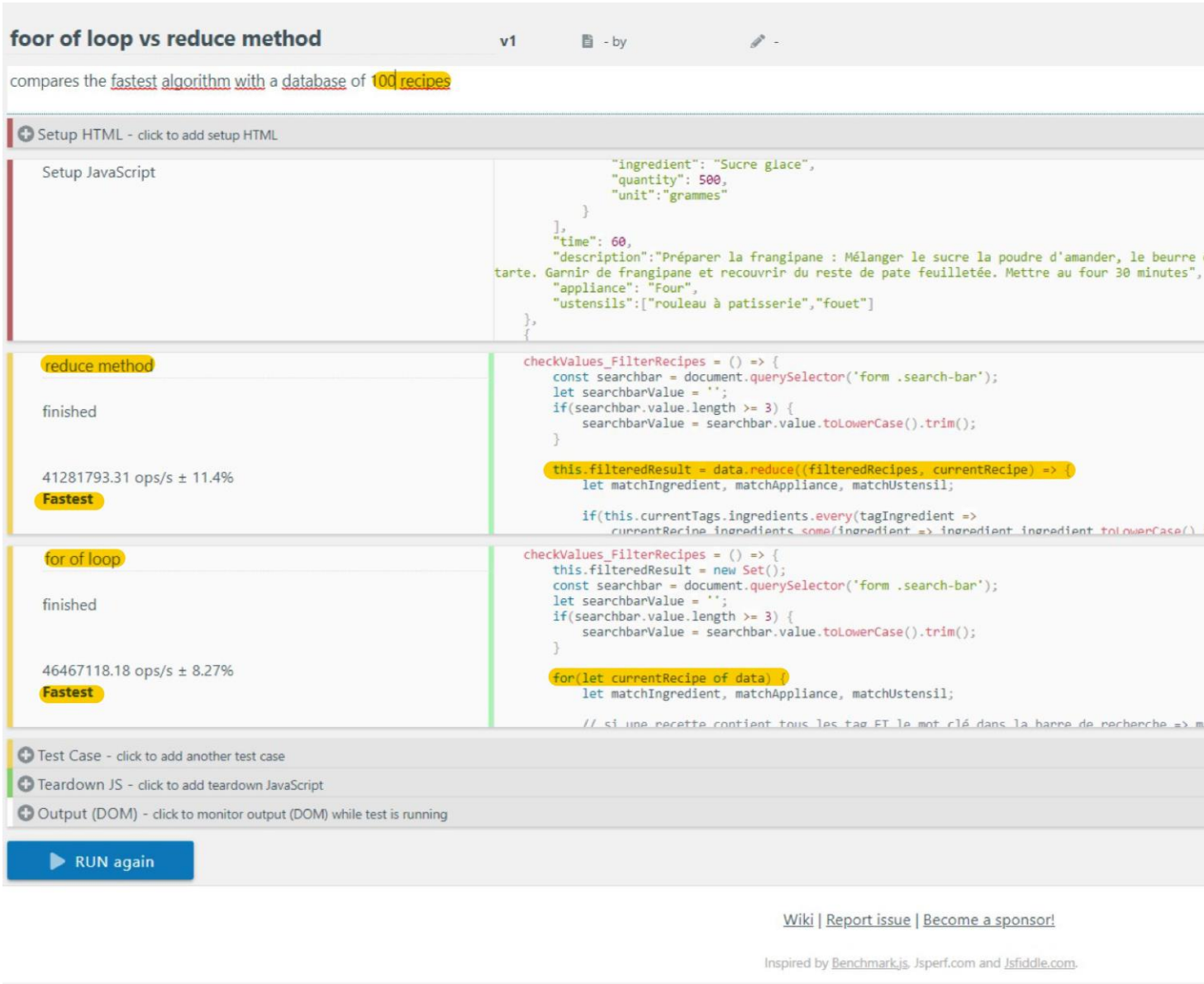
Comparaison entre for...of et reduce

Sur une base de 50 recettes :



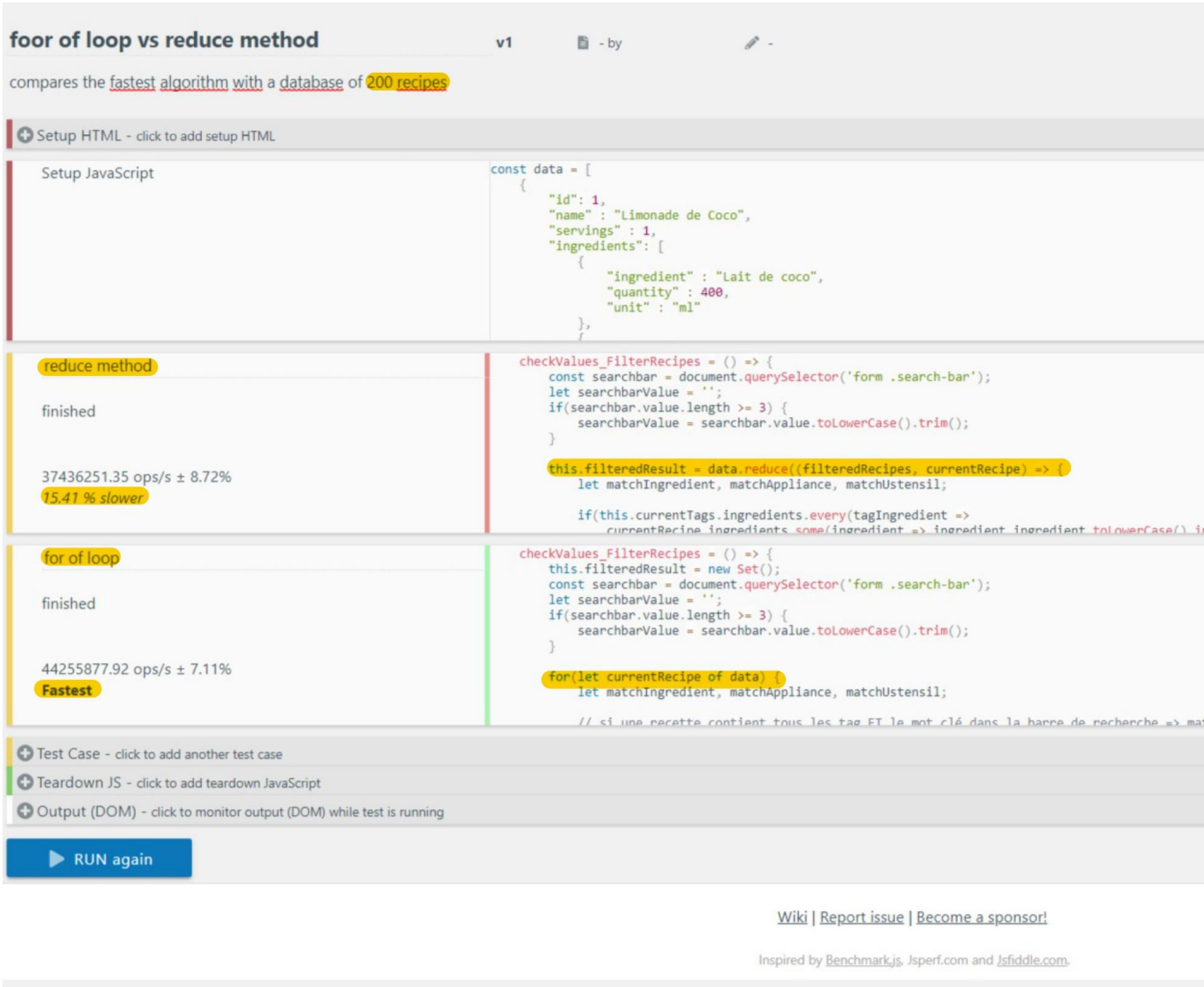
Résultat : reduce est 16% plus rapide

100 recettes :



Résultat : les 2 méthodes sont quasiment équivalentes

200 recettes :



Résultat : la boucle for est 35% plus rapide

Conclusion :

J'ai choisi de retenir l’option 1 : la boucle native for of. En effet, malgré qu’en l’état actuel du site et de ses 50 recettes la méthode reduce est un poil plus rapide que la boucle for, si la base de données venait à s’agrandir et contenir plus de recettes, il serait beaucoup plus efficace d’utiliser la boucle for que la méthode reduce.