Fiche d'investigation de fonctionnalité

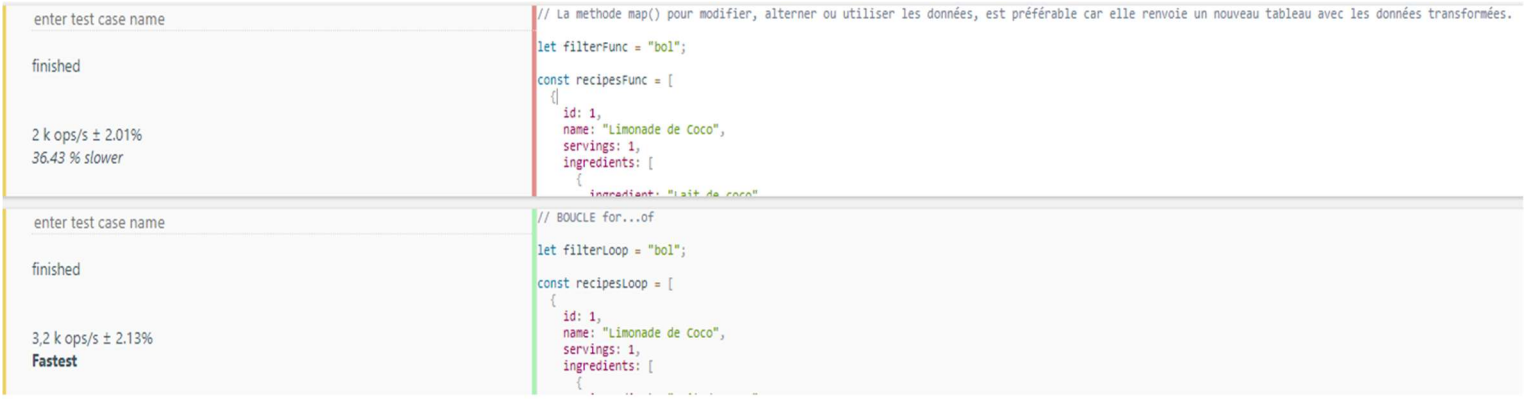
Fonctionnalité : Filtrer les recettes dans l'interface utilisateur	Fonctionnalité n°1
Problématique : Permettre de trier parmi les recettes disponibles via un champ de recherche et des menus de sélections d'ingrédients, d'appareils ou d'ustensiles. La recherche renverra les recettes dont le nom, la description ou les ingrédients sont contenues dans la recherche, ainsi l'appareil, les ingrédients et les ustensiles qui correspondent aux menus de sélections.	
Nombre de champ : 1 champ de recherche (optionnel) **Nombre de sélecteur** : 3 sélecteurs (ingrédients, appareils, ustensiles) (optionnel) **Nombre de champ minimum** : 0 champ et 0 sélecteur (renvoie alors toutes les recettes)	

Option 1 : Boucle avancée (map(), forEach(), filter() …) Dans cette option, le code est composé de boucle et de méthodes de tableaux avancés.	
Avantages : - Ecriture du code plus simple - Meilleur maintenabilité du code - Compatible avec tous les navigateurs - Utilisation et pratique de manipulation de tableau avancé - Meilleure compréhension du code	**Inconvénients :** - Nécessite la connaissance et l'utilisation des méthodes de manipulation des tableaux pour construire l'algorithme

Option 2 : Boucle native (for … of) Dans cette option, le code est composé de boucle et de méthodes natives de Javascript.	
Avantages : - Utilisation des connaissances de bases d'itération de Javascript	**Inconvénients :** - Rend la compréhension de code plus complexe - Réduit l'efficacité que permettent certaines méthodes avancées - Rend moins maintenable le code et l'utilisation par d'autres développeurs

Solution retenue :
J'ai retenue l'option 1 pour différentes raisons, malgré des résultats moins performants (cf. image Jsbench) :

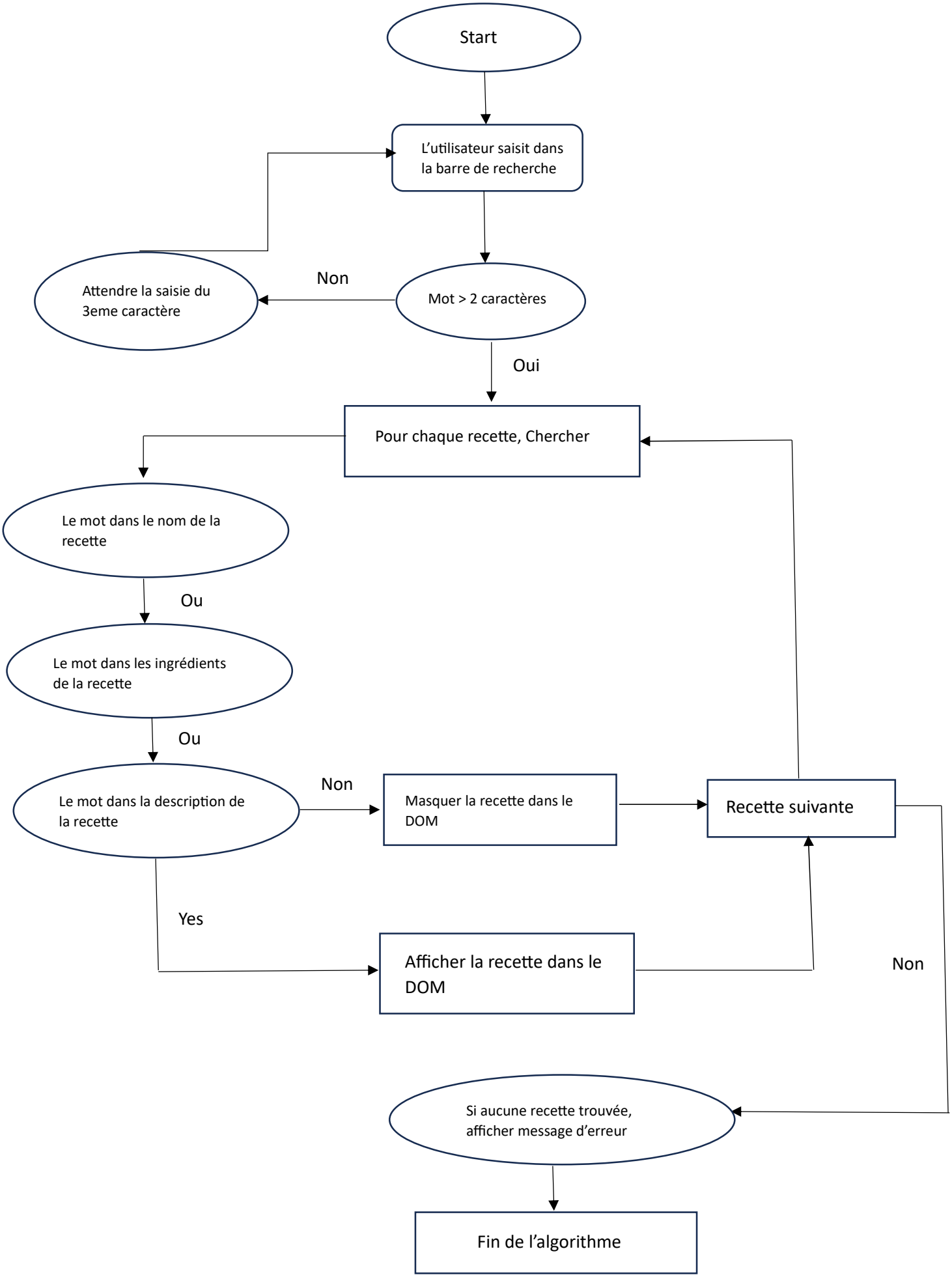
- Le code est plus simple à lire
- Le code est plus maintenable et plus facilement réutilisable
- Les méthodes ('map()', ' filter()', 'forEach()') sont plus explicites et permettent une meilleure manipulation des tableaux



Test 1 : Méthodes map(), filter() → 2k ops/s

Test 2 : Boucle for…of → 3.2k ops/s

Option 1, méthode map()



Option 2, boucle native (for…of)