







Fiche d'investigation de fonctionnalité

Fonctionnalité: Algorithme de recherche par mot	Fonctionnalité #1
Problématique: Le moteur de recherche doit être le plus rapide possible pour une bonne expérience utilisateur	

Option 1: Algorithme de recherche avec boucles natives «for» (cf Figure 1).

Dans cette option, lorsque l'utilisateur se sert de la barre de recherche et qu'il tape 3 lettres ou plus, on trie l'array de toutes les recettes avec une loop for.






Pour chaque recette, si aucune correspondance n'est trouvée dans le titre, la description ou les ingrédients, on supprime la recette de l'array.

Avantages	Inconvénients <ul style="list-style-type: none">- Pas assez rapide- Beaucoup de lignes de code- Pas assez lisible
Performance avec  JSBench.me  JSBEN.CH	
Recherche du mot «coco» dans 50 recettes  525463.41 ops/s \pm 0.16%  72.3% 51.77% slower	

Option 2: Algorithme de recherche méthode de l'objet array «filter» (cf Figure 2).

Dans cette option, lorsque l'utilisateur se sert de la barre de recherche et qu'il tape 3 lettres ou plus, on retourne un nouvel array trié des recettes avec la méthode filter.

L'array filtré contient toutes les recettes dont le nom, la description, et les ingrédients contiennent le mot recherché.

Avantages <ul style="list-style-type: none">- Rapide- Peu de lignes de code- Lisible	Inconvénients
Performance avec  JSBench.me  JSBEN.CH	
Recherche du mot «coco» dans 50 recettes  1089440.16 ops/s \pm 0.46%  100%  Fastest	

Solution retenue:

La solution retenue est l'algorithme utilisant la méthode filter() de l'objet array.

Cette méthode est plus rapide, plus lisible, plus facilement maintenable et utilise moins de ligne de code.

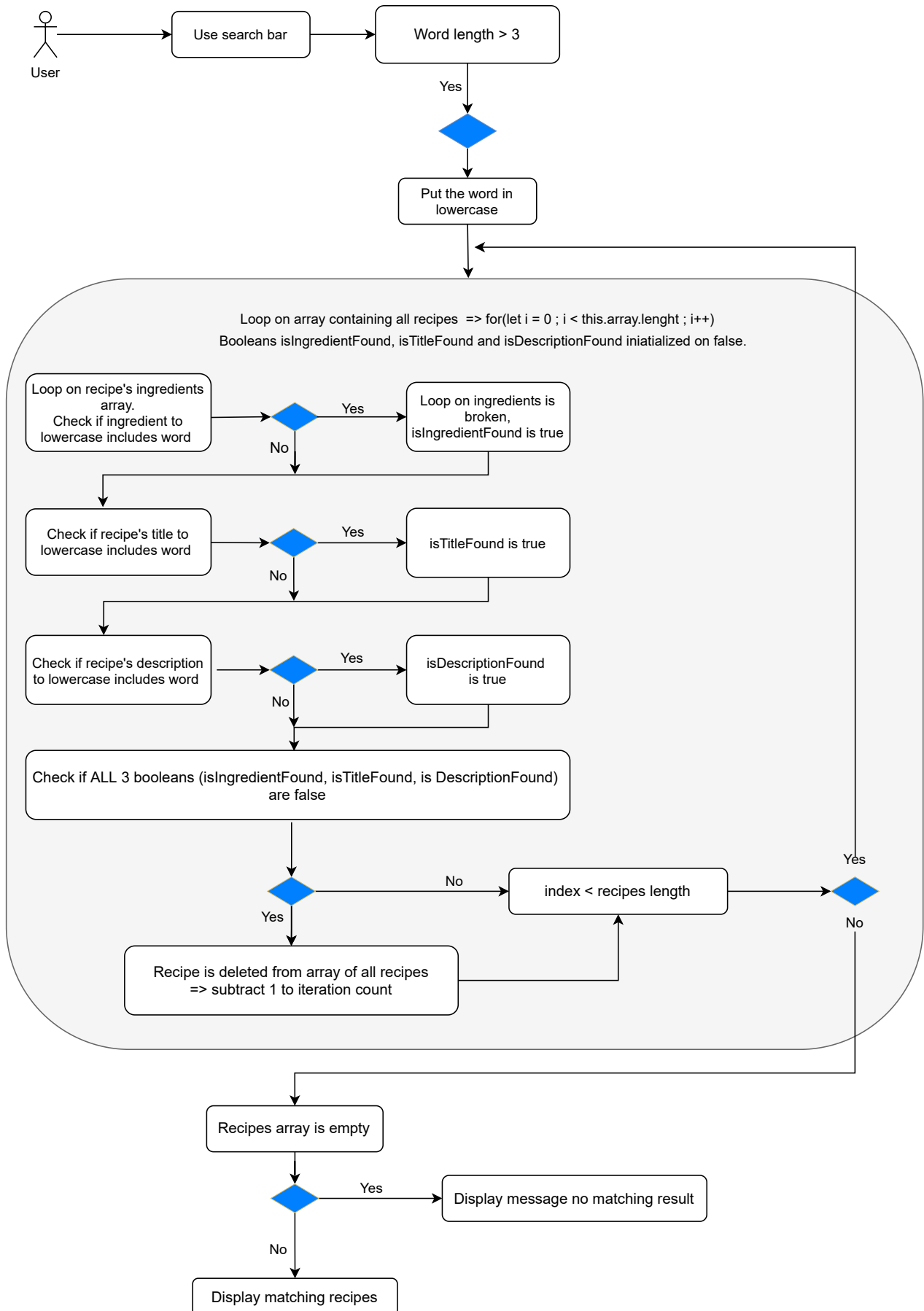


Figure 1: Algorithme de recherche avec boucles natives «for»

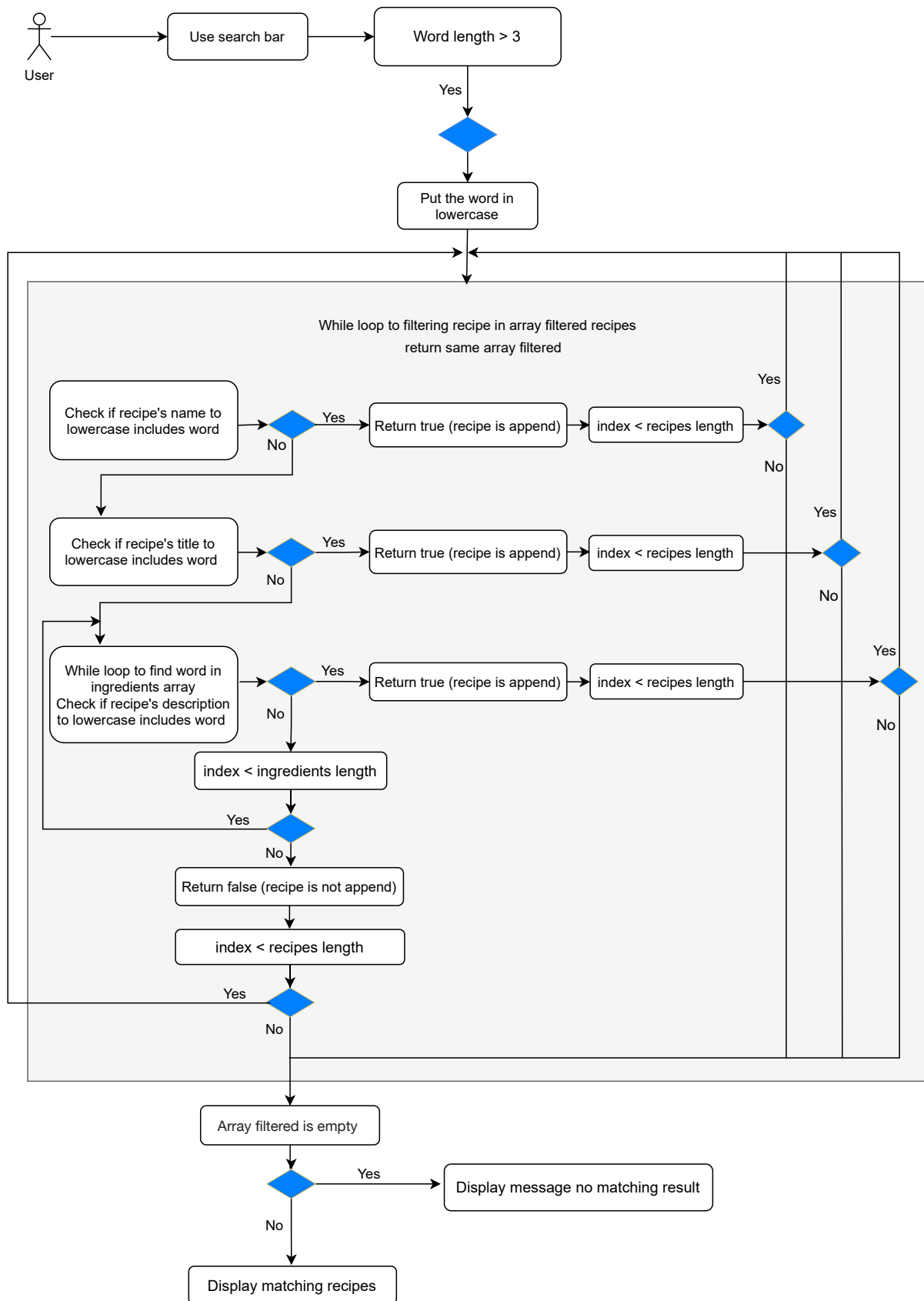


Figure 2: Algorithme de recherche avec méthode de l'objet array «filter()»