Fiche d'investigation de fonctionnalité

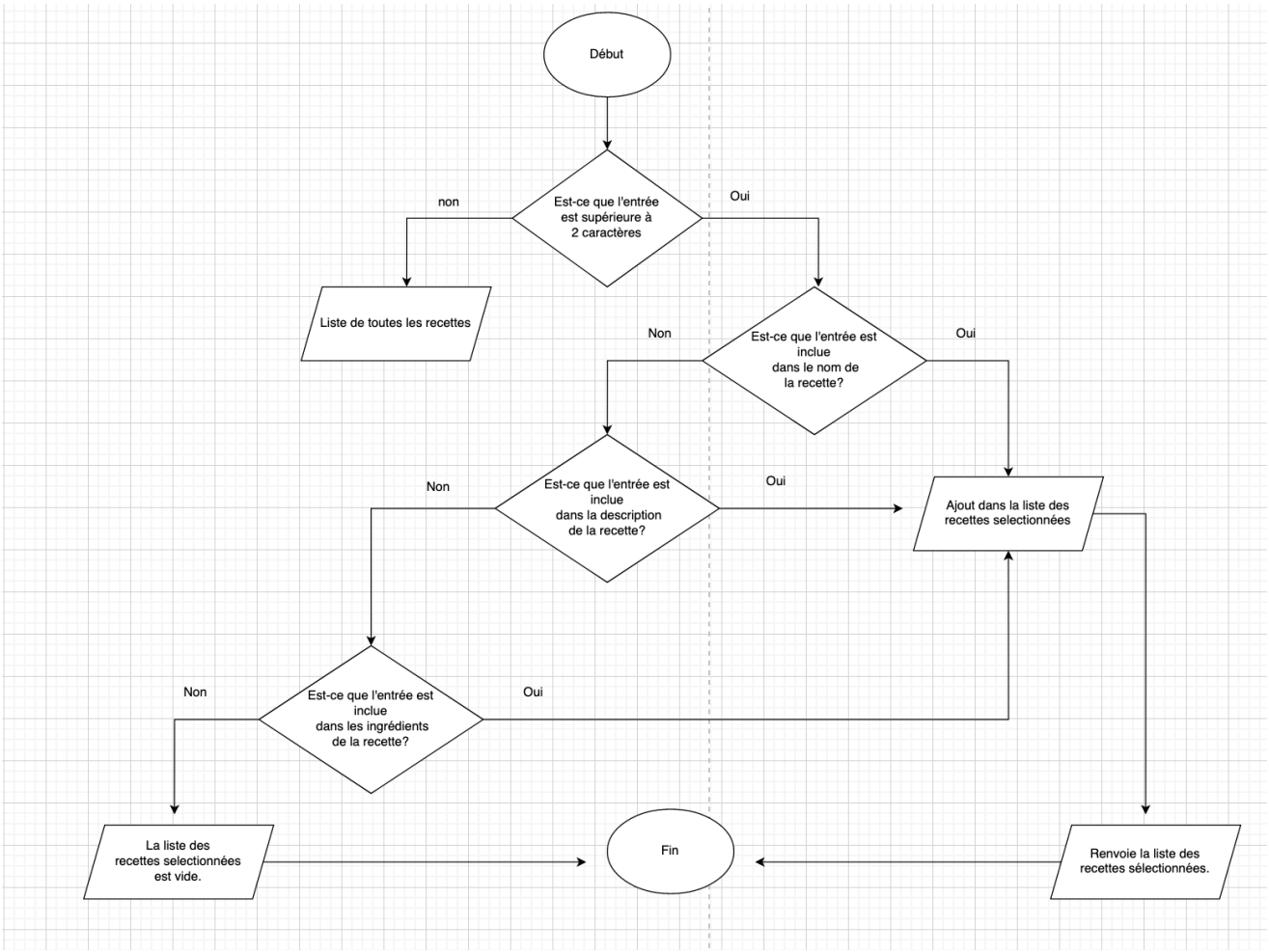
Fonctionnalité : Rechercher des recettes	**Fonctionnalité #1**
Problématique : Filtrer les recettes en fonction du besoin d'un utilisateur. L'utilisateur doit pouvoir saisir une recette dans le champ de recherche et y accéder rapidement dans l'interface.	

Option 1 : programmation fonctionnelle (cf figure annexe 1). Utilisation des fonctions de prototype array. Ici, emploi de la méthode filter de l'objet array qui permet de filtrer les recettes en fonction du terme saisi par l'utilisateur dans le champ de recherche. Pour cela, l'algorithme va vérifier s'il y a des correspondances entre la saisie de l'utilisateur et le nom de la recette ou ses ingrédients ou sa description. Si la recette correspond à cette condition elle est ajoutée à un tableau qui permettra ensuite d'afficher les recettes mis à jour par l'algorithme.	
Avantages : - Code plus maintenable et robuste. - Code plus court et plus rapide.	**Inconvénients :** - Code moins compréhensible.
Entrer plus de 2 caractères dans la barre de recherche principale	

Option 2 : programmation native (cf figure annexe 1).. Utilisation des boucles (while, for…) Dans cet algorithme, utilisation de la boucle for qui parcourt le tableau général des recettes en vérifiant s'il existe une correspondance entre la saisie entrée par l'utilisateur et le nom de la recette ou sa description ou ses ingrédients. Si la recette correspond à cette condition elle est ajoutée à un tableau qui permettra ensuite d'afficher les recettes mis à jour par l'algorithme.	
Avantages : Le code est plus facile à comprendre.	**Inconvénients :** - Le code est moins maintenable, moins robuste. - L'exécution du code est plus lente.
Entrer plus de 2 caractères dans la barre de recherche principale	

Solution retenue :
Il apparaît que le temps d'exécution de l'algorithme en programmation native augmente clairement avec le nombre de données (cf annexe2).
Le choix se porte donc sur l'option 1, la programmation fonctionnelle.

Annexe 1


Début
Est-ce que l'entrée est supérieure à 2 caractères
non
Oui
Liste de toutes les recettes
Est-ce que l'entrée est inclue dans le nom de la recette?
Non
Oui
Est-ce que l'entrée est inclue dans la description de la recette?
Non
Oui
Ajout dans la liste des recettes selectionnées
Est-ce que l'entrée est inclue dans les ingrédients de la recette?
Non
Oui
La liste des recettes selectionnées est vide.
Fin
Renvoie la liste des recettes sélectionnées.

Annexe 2

Test sur 1000 recettes

Console.time

For

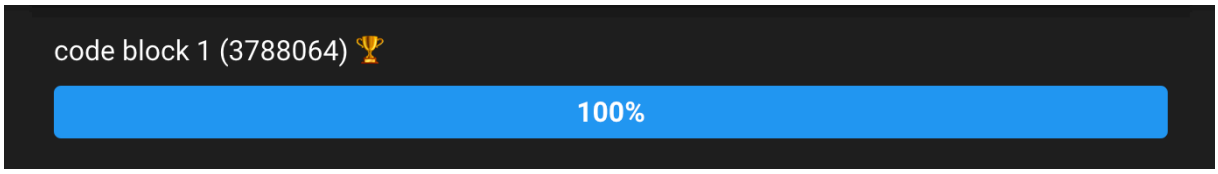
Temps de l'algo: 0.010009765625 ms

Filter

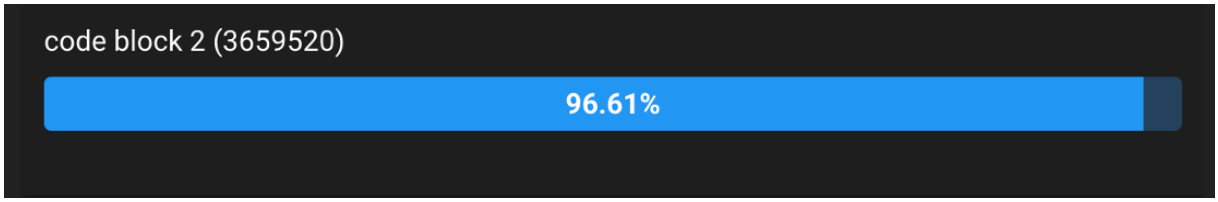
Temps de l'algo: 0.010986328125 ms

JSBench

For



Filter



JSBench.me

For

Filter

Test sur 4000 recettes

Console.time

For

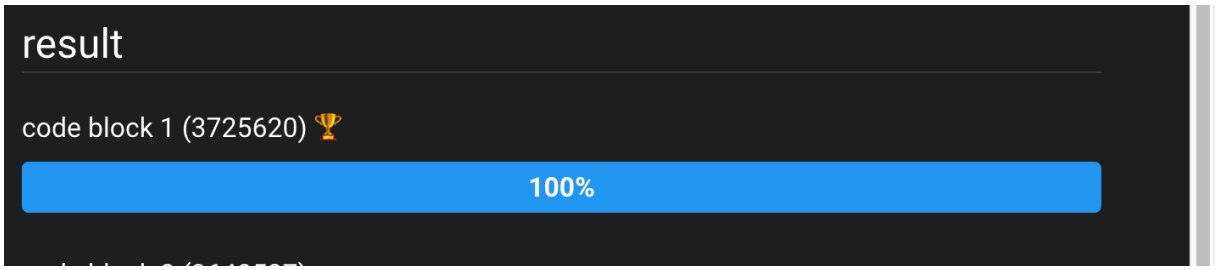
Temps de l'algo: 0.012939453125 ms

Filter

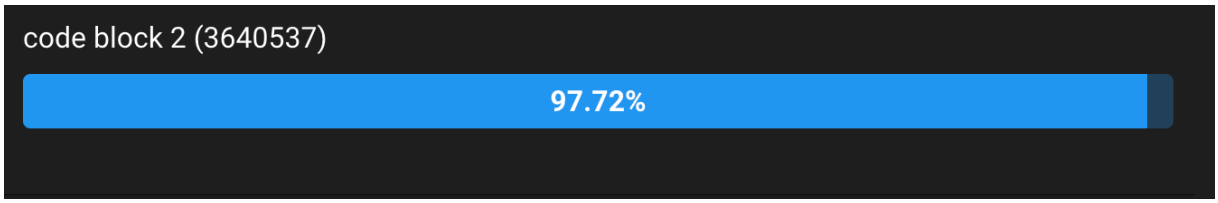
Temps de l'algo: 0.010986328125 ms

JSBench

For



Filter



JSBench.me

For

Filter

Test sur 8000 recettes

Console.time

For

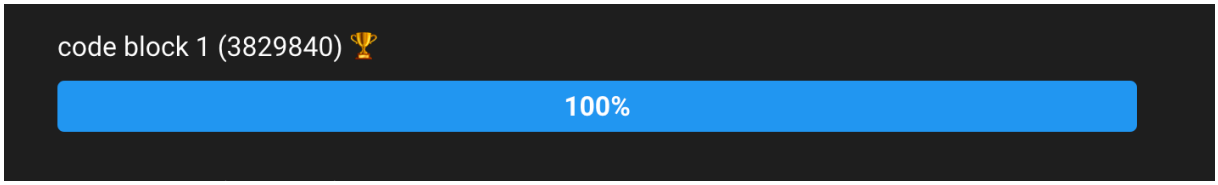
Temps de l'algo: 0.01416015625 ms

Filter

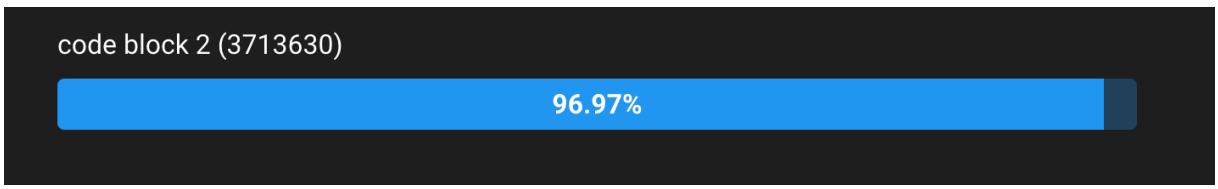
Temps de l'algo: 0.010009765625 ms

JSBench

For



Filter



JSBench.me

For

Filter

Bilan des tests

1000 recettes

	For	Filter	Bilan
Console.time	Temps de l'algo: 0.010009765625 ms	Temps de l'algo: 0.010986328125 ms	For est plus rapide de 0.0009765625 ms.
JSBench	100%	96,61%	For est de 3,39% plus rapide
JSBench.me	Même performance	Même performance	Pas de différence

4000 recettes

	For	Filter	Bilan
Console.time	Temps de l'algo: 0.012939453125 ms	Temps de l'algo: 0.010986328125 ms	Filter est plus rapide de 0.001953125 ms
JSBench	100%	97,72%	For est de 2.28% plus rapide
JSBench.me	Même performance	Même performance	Pas de différence

8000 recettes

	For	Filter	Bilan
Console.time	Temps de l'algo: 0.01416015625 ms	Temps de l'algo: 0.010009765625 ms	Filter est plus rapide de 0.04150390625 ms
JSBench	100%	96,97%	For est de 3.03% plus rapide.
JSBench.me	Près de 4% plus lente qu'avec le filter		Filter est plus rapide

D'après le console.time, le for prend plus de temps si on augmente les données alors que le filtre ne change pas ou très peu quel que soit le nombre de données.

Le for augmente clairement avec le nombre de données.

Il apparaît que pour les autres outils, JSBench et JSBench.me, le for semble plus performant. Cependant, ces outils n'ont pas l'air de prendre en compte le nombre de données.