Les petits plats

Fiche d'investigation de fonctionnalité

Problématique

Freelance missionné (pour 3 mois) par l'entreprise "Les petits plats" en temps que Développeur Front-end pour développer et intégrer 2 algorithmes de recherches parmis des recettes en fournissant une métrique comparant les performances des algorithmes produits.

Mission

- Accéder rapidement à une recette correspondant à un besoin de l'utilisateur dans les recettes déjà reçues.
- Implémenter la fonctionnalité de recherche.
- Comparatif de perf des algo.

Attendus

- Implémentation de la maquette.
- Créer 2 algos de recherche.
- Accéder rapidement à une recette correspondant à un besoin de l'utilisateur dans les recettes déjà reçues.
- Les résultats de recherche sont actualisés, ainsi que les éléments disponibles dans les champs de recherche avancée.

Fonctionnalité : Filtrer les recettes depuis l'interface utilisateur.
Problématique : Mise à jour rapide avec filtrage en fonction de plusieurs paramètres combinés.
D'après les scénarii (nominal ou alternatifs) obtenir une liste de recettes triées selon la correspondance entre saisie ou le choix d'un tag avec les recettes (cad : le nom,la description, les ingrédients ou les ingrédients, les appareils et ustensils)

Option 1 : Algo 1 (**for** loop)	
Avantages: utilisation familière	Inconvénients : peu explicite, ES6 "négligée"
Conditions : Dans cette version, la boucle exclut les éléments qui remplissent les conditions imposées par les if. Un tableau est généré (clone de la liste complète des recettes) en clonant la source de data. La fonction supprime chaque recette dont les conditions sont évaluées.	

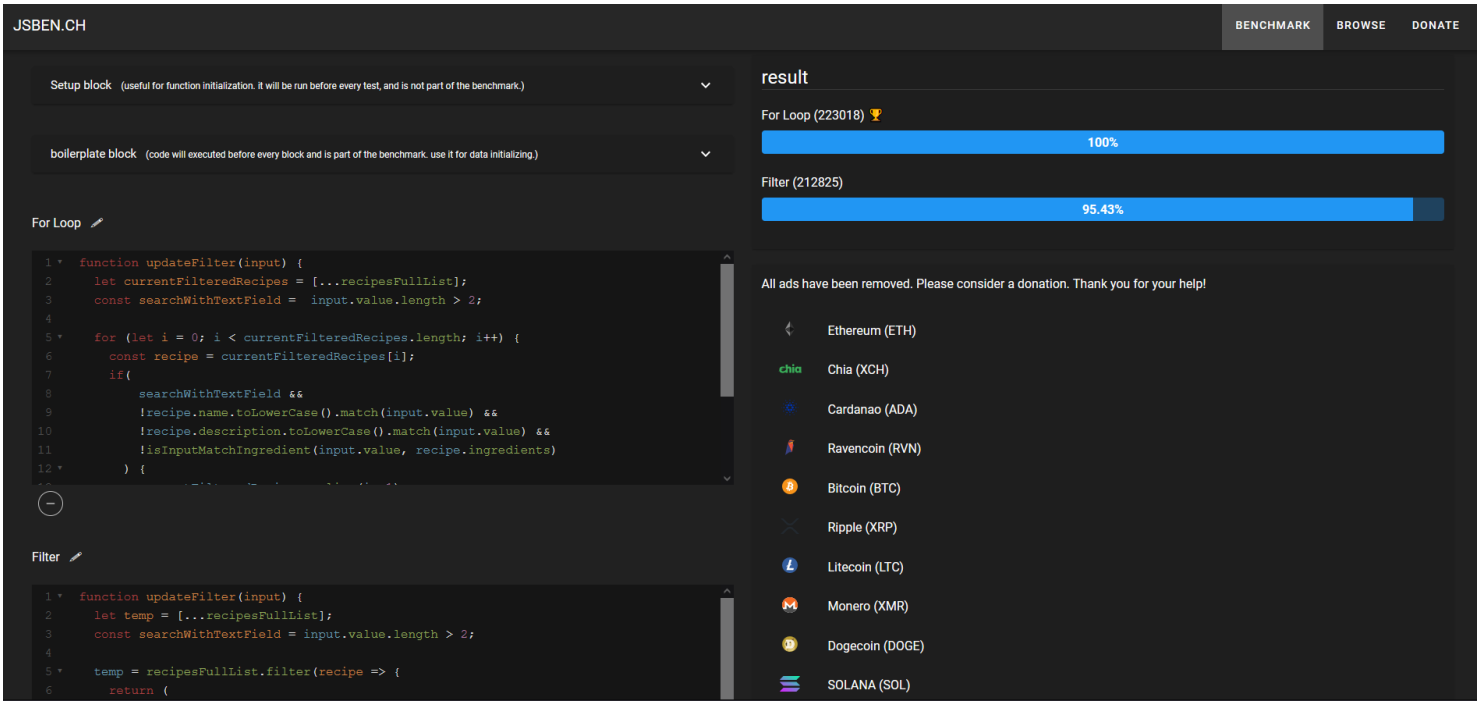
Option 2 : Algo 2 (**filter** iterator)	
Avantages : objectif explicite, chainable	Inconvénients : …
Conditions : Dans cette version, la boucle inclut les éléments qui remplissent les conditions imposées par les	

if.
Un tableau est généré (clone de la liste complète des recettes) en clonant la source de data.
La fonction stocke chaque recette dont les conditions sont évaluées.

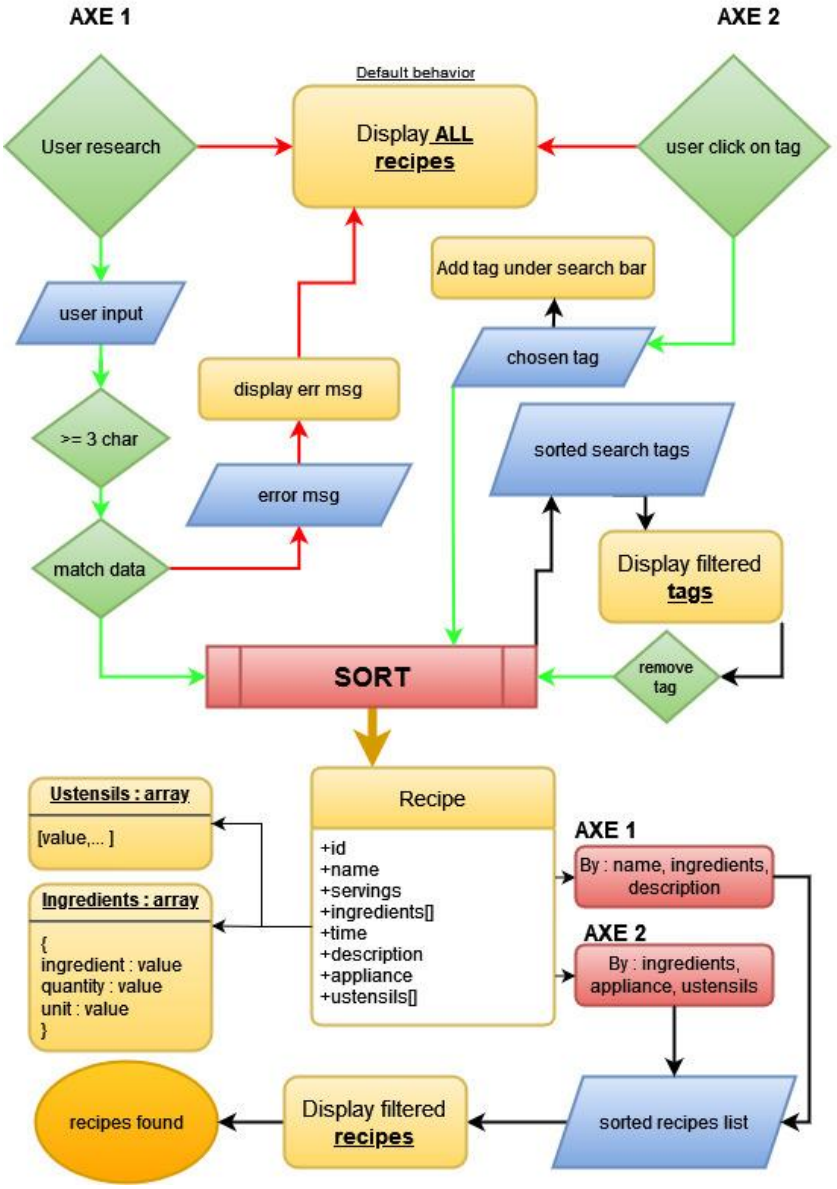
Solution retenue : Option 2

Malgré la faible différence de performance (screenshot en annexe) dans le contexte, l'utilisation de filter() offre une meilleure lisibilité du code et une évolutivité plus grande notamment car elle est chainable et renvoie directement un array.

Annexe 1 :



Annexe 2 :


AXE 1
AXE 2
Default behavior
User research
Display ALL recipes
user click on tag
Add tag under search bar
user input
chosen tag
display err msg
>= 3 char
sorted search tags
error msg
match data
Display filtered tags
SORT
remove tag
Ustensils : array
[value,...]
Recipe
+id
+name
+servings
+ingredients[]
+time
+description
+appliance
+ustensils[]
AXE 1
By : name, ingredients, description
Ingredients : array
{
ingredient : value
quantity : value
unit : value
}
AXE 2
By : ingredients, appliance, ustensils
recipes found
Display filtered recipes
sorted recipes list