Les petits plats

Fiche d’investigation de fonctionnalité

Fonctionnalité : Recherche principale	Fonctionnalité de recherche générale #1
Problématique : Afin de pouvoir donner la meilleure expérience de recherche aux utilisateurs , nous cherchons à rendre le plus fluide possible les séquences de requêtes de la barre principale dans le but d’afficher les recettes le plus rapidement possible de manière pertinente. Le but de cette fonctionnalité est de se démarquer de la concurrence pour proposer aux utilisateurs une séquence de recherche très réactive et pertinente. En effet il est important d’avoir une recherche principale très performante pour offrir aux usagers le meilleur outil possible pour leur utilisation	

Option 1 :

On effectue un algorithme utilisant la programmation fonctionnelle avec les méthodes de l’objet array (foreach , filter , map , reduce etc…)

On utilise la méthode filter pour générer un tableau de recettes

Avantages :	**Inconvénients :**
- Recherche rapide, fluide et pertinente - Moins de répétitions , moins de ligne de code - Code plus maintenable - Plus d’operazione dans un temps imparti	- Lisibilité du code moins claire

Nombre de caractère a remplir pour déclencher la recherche : 3

Option 2 :

Utilisation de boucles natives (while , for etc…)
Utiliser une boucle pour l’ensemble des recettes pour voir s’il y a un match avec la valeur de recherche

Avantages :	**Inconvénients :**
- Lisibilité du code plus claire	- Moins de calcul dans un temps imparti - Boucle moins performante - Plus de ligne de code

Nombre de caractères a remplir pour déclencher la recherche : 3

Solution retenue :

En testant les 2 algorithmes dans un benchmark nous nous sommes rendu compte que la solution la plus rapide et pertinente était l’option 1 du filter qui effectue plus d’opérations dans un temps imparti

Annexes

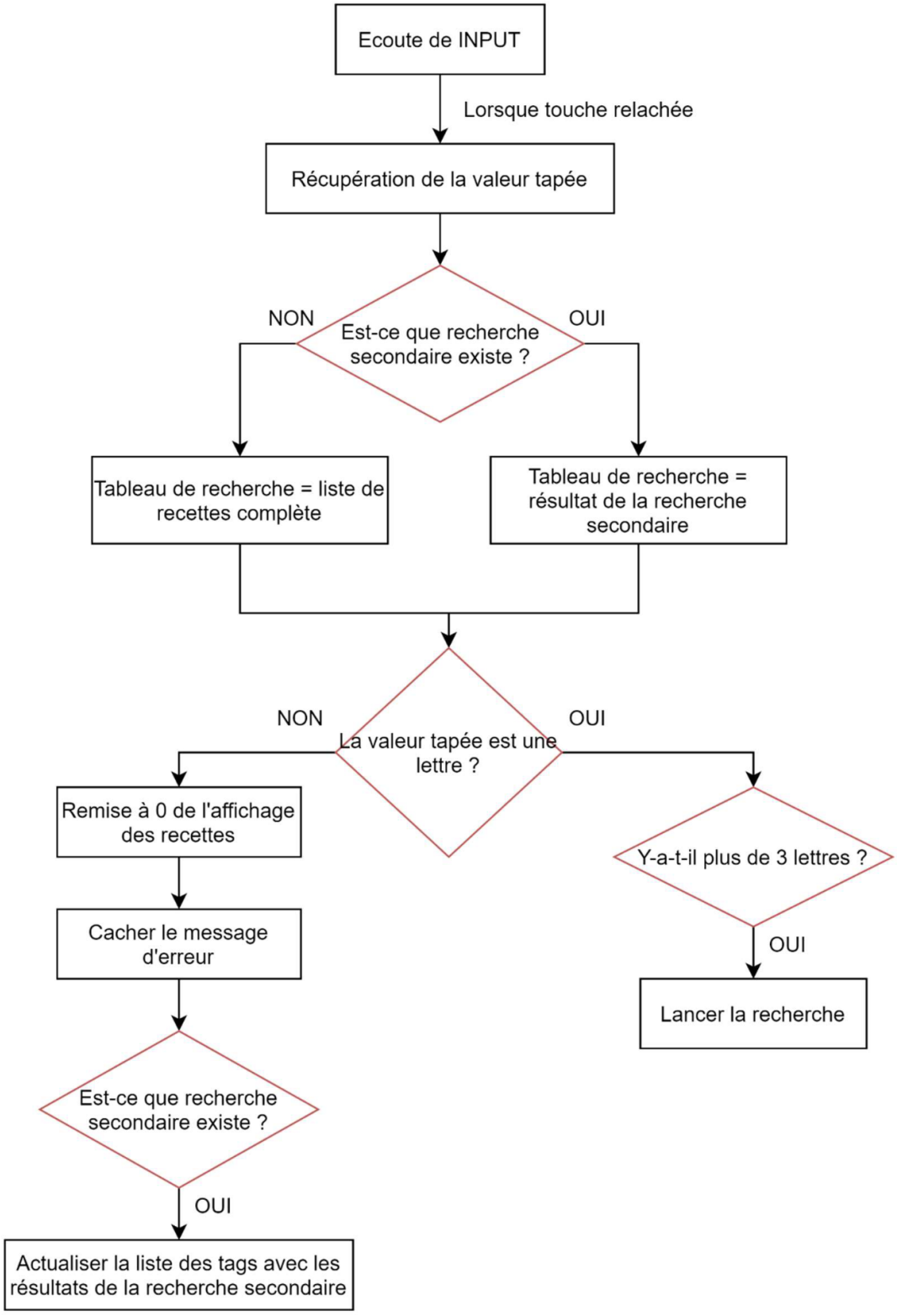

Figure 1 - Diagramme d'activité commune aux deux recherches

Les petits plats

Analyse des Performances JS

1. Benchmark JSBen.ch

Lien : https://jsben.ch/H5ULD

On test les performances JS suivant plusieurs scénarios. Les nombres du tableau indiquent le nombre total d'opérations/secondes . Plus la valeur est grande plus l'algorithme est performant car il y plus d'opérations par seconde dans un temps imparti :

	Algo 1 Filter	Algo 2 For
Scenario 1 (occurence: ex:Poulet) data :50	207559	196506
Scenario 2 (occurence: ex: « ») data :50	1561076	332013
Scenario 3 (occurence: ex:Poulet) data :500	16896	16007
Scenario 4 (occurence: ex:Poulet) data :1000	9894	9670
Scenario 5 (occurence: ex:Poulet) data :5000	6821	6612

On remarque ici que l'algorithme avec Filter est plus performant que celui avec les boucles FOR.

Par exemple on obtient pour le scénario 1 avec 50 données :