Fiche d’investigation de fonctionnalité

Fonctionnalité : Moteur de recherche (barre de recherche et filtres)	**Fonctionnalité #3**
Problématique : Comment offrir aux utilisateurs un moteur de recherche rapide et performant qui puisse filtrer les recettes en fonction de plusieurs critères (ingrédients, appareils, ustensiles) tout en offrant une expérience fluide ?	

Option 1 : Utilisation des méthodes de tableau

Dans cette option, nous utilisons des méthodes telles que .filter() et .forEach() pour parcourir les recettes et appliquer les filtres. L’avantage est que ces méthodes sont lisibles, concises, et bien optimisées pour la gestion des tableaux.

Avantages	**Inconvénients**
⊕ Plus simple à écrire et à maintenir ⊕ Lisibilité accrue pour les autres développeurs ⊕ Méthodes modernes et optimisées pour la gestion de données massives	⊖ Peut être plus lent que les boucles natives dans certains cas (surtout sur des grandes bases de données) ⊖ Limité aux fonctionnalités natives des méthodes, moins flexible pour certaines logiques complexes

Performance :

Les méthodes .filter() et .forEach() sont optimisées pour des parcours de tableau, mais peuvent devenir plus lentes avec des données massives.

Option 2 : Utilisation des boucles natives (for, while)

Dans cette option, nous utilisons des boucles natives for et while pour parcourir les recettes et appliquer les filtres. L’avantage est que nous avons un contrôle total sur la logique de parcours, ce qui peut potentiellement améliorer les performances sur des bases de données importantes.

Avantages	**Inconvénients**
⊕ Contrôle total sur le parcours des données ⊕ Peut être optimisé spécifiquement pour des cas particuliers ⊕ Potentiellement plus rapide sur de grands tableaux	⊖ Code plus long et moins lisible ⊖ Risque accru de bugs dans les boucles (boucles infinies, erreurs de logique) ⊖ Plus difficile à maintenir et à lire

Performance :

Les boucles natives peuvent être plus rapides pour des très grandes bases de données en évitant la surcharge des méthodes intégrées.

Solution retenue :

L’approche utilisant les **méthodes de tableau** a été retenue pour les cas d’utilisation où la base de données est de taille modérée à grande. La lisibilité et la facilité de maintenance sont des facteurs clés. Cependant, pour des bases de données massives, les **boucles natives** seront explorées en option d'optimisation si les performances deviennent un enjeu majeur.

Annexe :

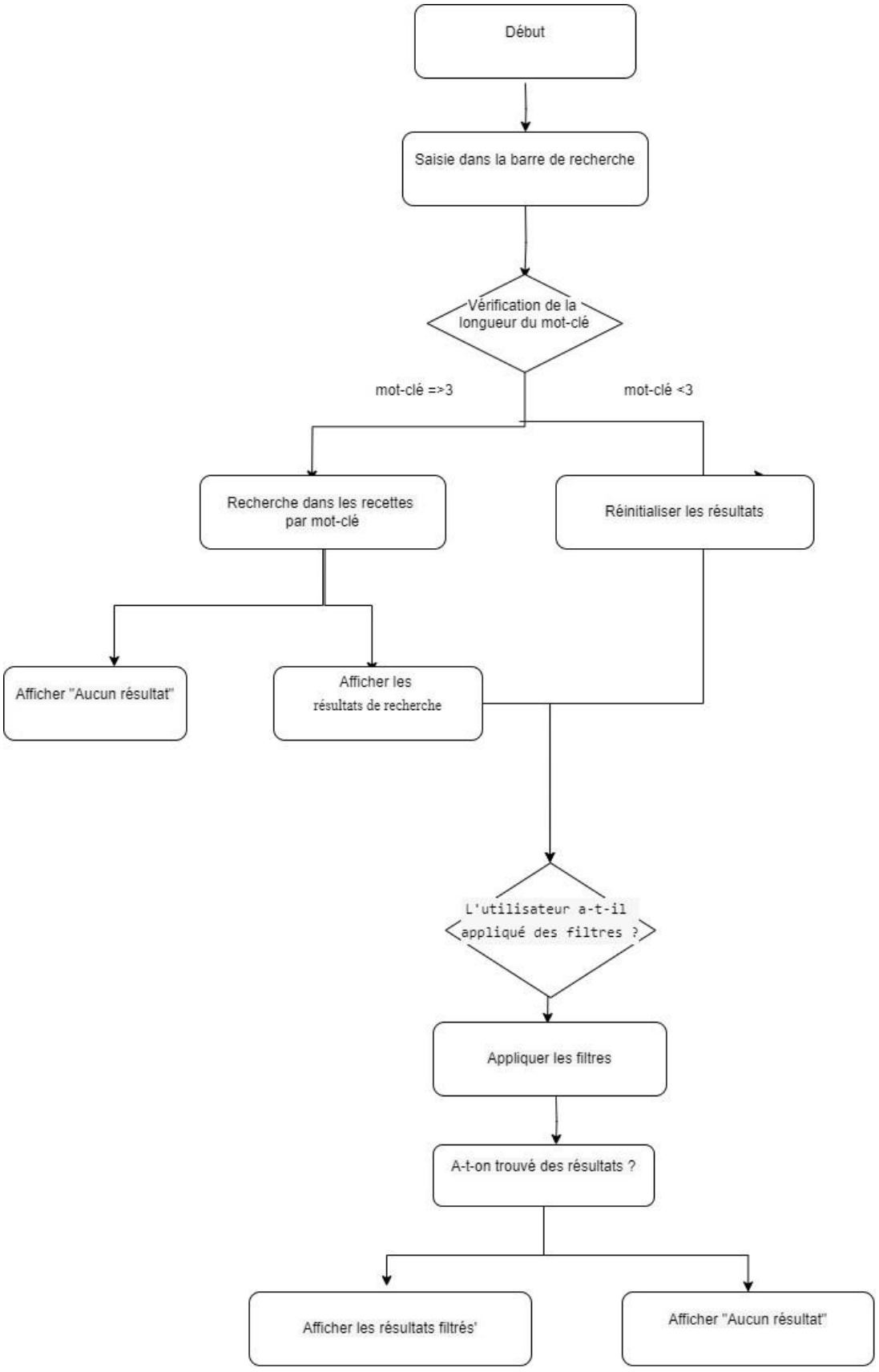


Figure 1 - Diagramme du flux de recherche -1-

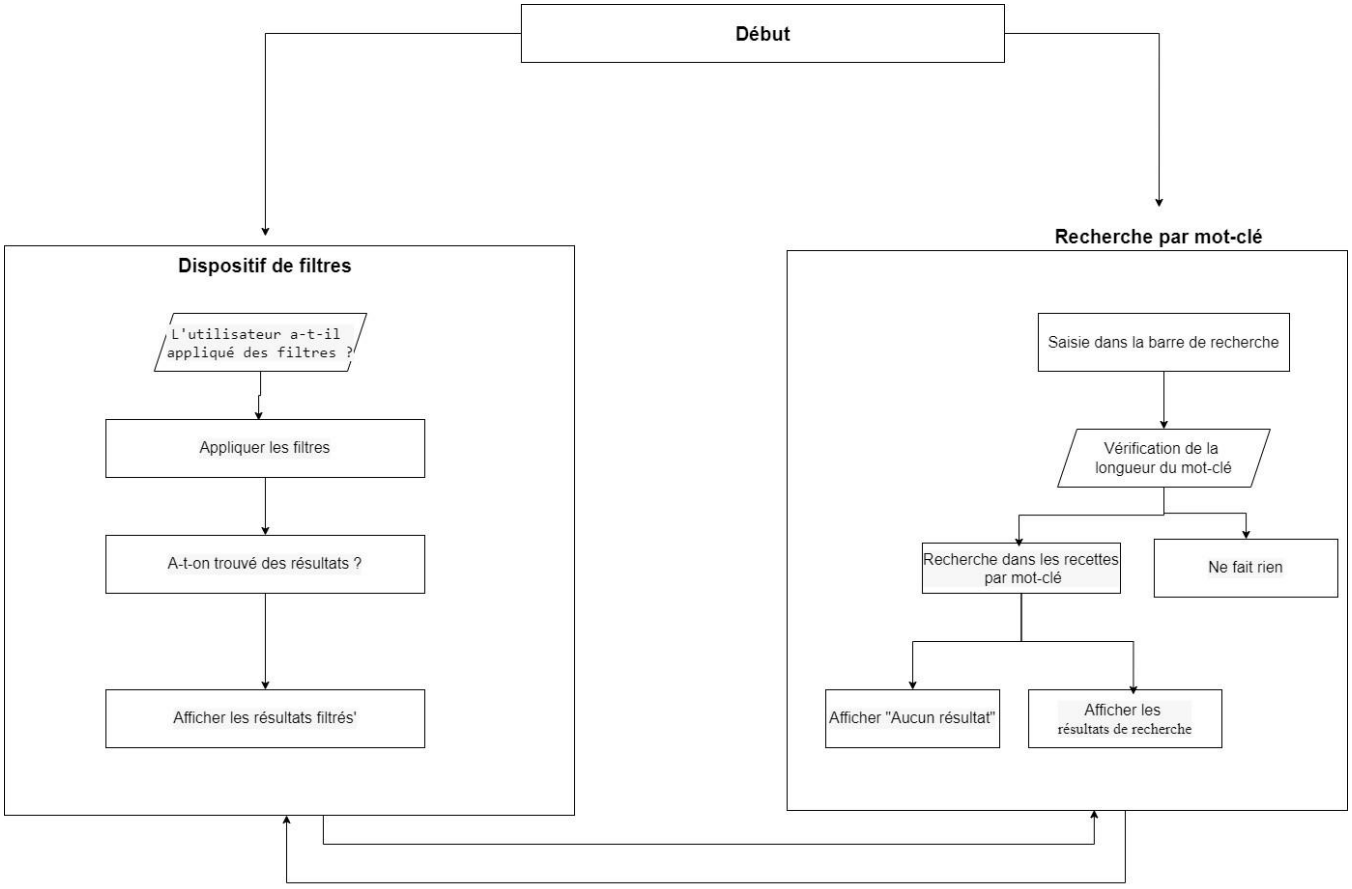


Figure 2 - Diagramme du flux de recherche -2-

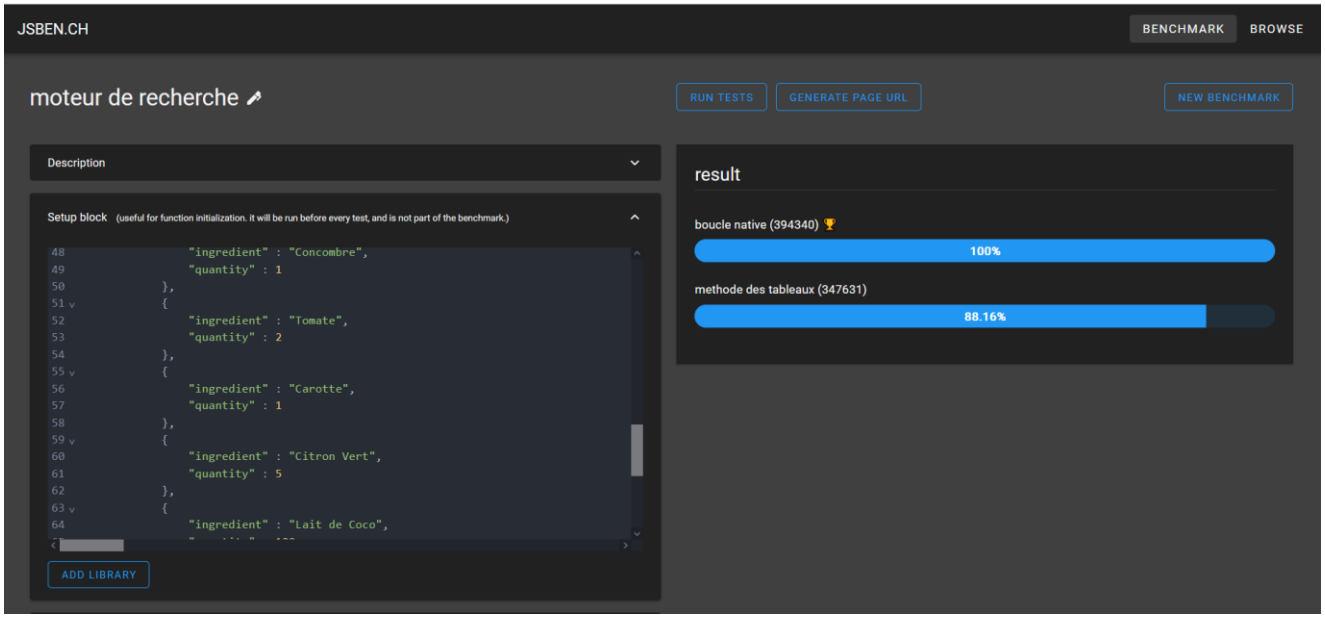
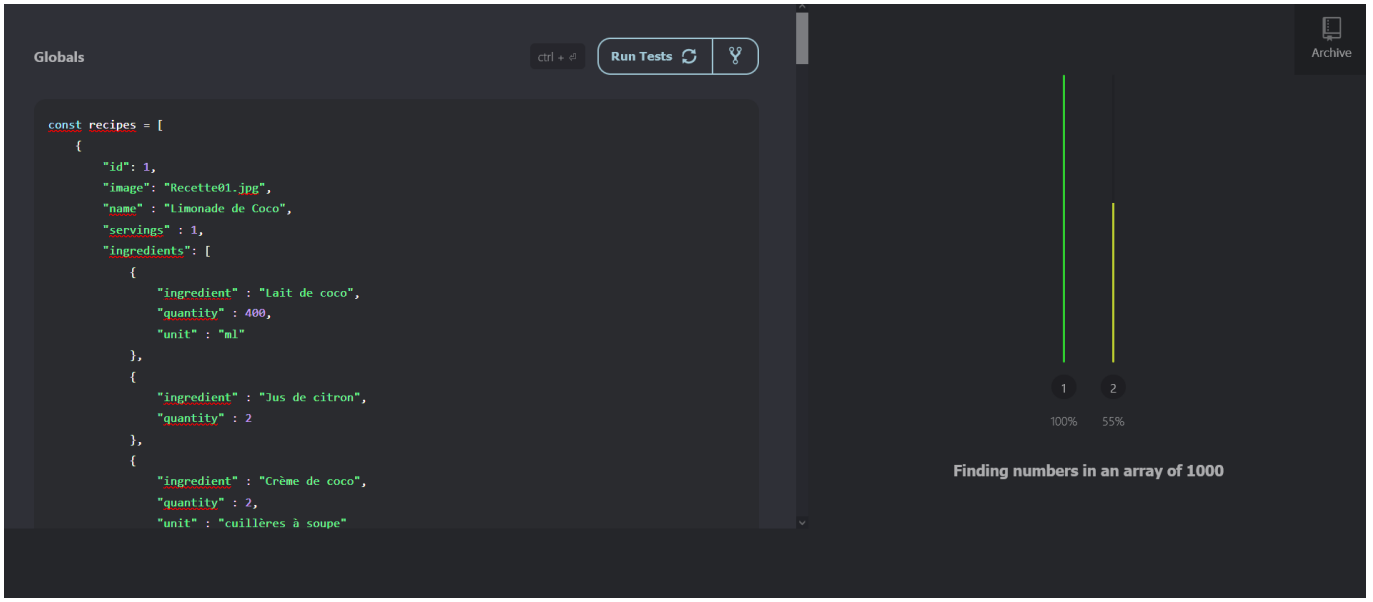


Figure 3 – jsben.ch teste performance



1- Boucles native
2- Méthode des tableuax

Figure 4 – Perflink teste performance