Fonctionnalité : Recherche avancée dans les recettes	fonctionnalité #2
Problématique : Comment optimiser l'algorithme de recherche des recettes afin d'offrir une expérience utilisateur fluide et rapide, tout en garantissant un code maintenable et performant	

Implémentation 1 : Approche classique avec boucles natives

- **Description :**
  - **Utilisation de boucles for, while et for...of.**
  - **Filtrage des recettes basé sur les critères suivants :**
    - **Correspondance avec le champ de recherche principal (nom, description, ou ingrédients).**
    - **Correspondance avec les tags actifs (ingrédients, appareils, ustensiles).**

Avantages	**Inconvénients**
Contrôle précis sur chaque étape de l'algorithme. Possibilité d'optimisation manuelle spécifique.	Code plus verbeux et complexe. Risque accru d'erreurs de duplication de logique.

Implémentation 2 : Approche fonctionnelle avec méthodes de tableau

- **Description :**
  - **Utilisation des méthodes de tableau modernes telles que filter, map, some, et every.**
  - **Utilisation de fonctions fléchées pour une syntaxe concise.**
  - **Application des critères de recherche pour filtrer les recettes.**

Avantages	**Inconvénients**
• Code plus lisible et facile à maintenir. • Moins de duplication de logique.	• Moins de contrôle direct sur les performances. • Dépendance à l'implémentation interne de JavaScript.

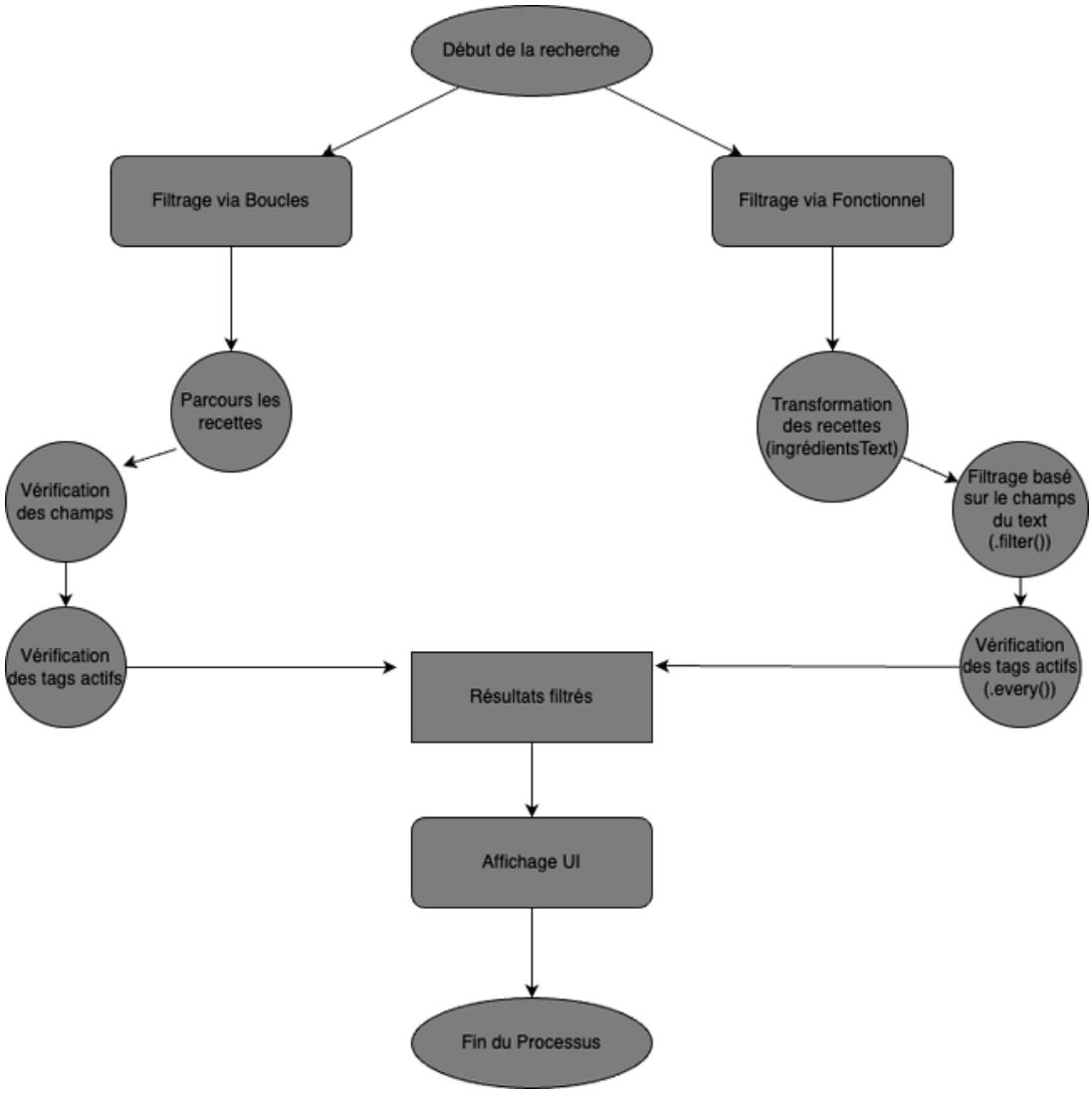
3. Résultats des tests de performance

Méthodologie :

- **Tests réalisés avec l'outil Jsben.ch.**
- **Nombre de recettes : 50 (données JSON fournies).**
- **Critères d'évaluation :**
  - **Temps d'exécution (opérations par seconde).**

Résultats :

Implémentation	**Opérations/seconde**	**Différence (%)**
Boucles natives	**115,458,73**	**+0.9%**
Méthodes fonctionnelles	**114,442,44**	**-0.9%**



4. Recommandation finale

- **Choix préféré : Approche fonctionnelle.**
  - **Justification :**
    - **Le code est plus lisible et maintenable, ce qui facilite son évolution future.**
    - **La différence de performance est négligeable pour un site de taille modérée.**
  - **Impact :**
    - **Réduction des coûts de maintenance.**
    - **Amélioration de la collaboration entre les développeurs.**

5. Conclusion

Les deux implémentations remplissent les objectifs de la fonctionnalité. Toutefois, pour garantir une meilleure lisibilité et facilité de maintenance, l'approche fonctionnelle est recommandée pour ce projet.