

## Fiche d'investigation de fonctionnalité

|                                                                                                                                             |                          |
|---------------------------------------------------------------------------------------------------------------------------------------------|--------------------------|
| <b>Fonctionnalité :</b> Filtrer les recettes                                                                                                | <b>Fonctionnalité #1</b> |
| <b>Problématique :</b> L'utilisateur doit pouvoir accéder rapidement à une recette correspondant à ses besoin dans les recettes déjà reçues |                          |

**Option 1 : Programmation impérative**

Développer une version de la fonction filterRecipes() en utilisant les boucles natives (while, for...)

**Avantages**

- ⊕ Bien lisible (Les instructions sont lues et exécutées de haut en bas)

**Inconvénients**

- ⊖ Le code devient très rapidement volumineux et perd en clarté
- ⊖ Il y a beaucoup de répétitions dans le code

**Option 2 : Programmation fonctionnelle**

Développer une version en programmation fonctionnelle de la fonction filterRecipes() avec les méthodes de l'objet array (foreach, filter, map, reduce)

**Avantages**

- ⊕ Code optimisé
- ⊕ Facilement maintenable et testable
- ⊕ Réutiliser facilement le code

**Inconvénients**

- ⊖ Assez complexe à appréhender.

**Solution retenue :**

Nous avons donc retenu l'approche "Programmation fonctionnelle". La raison est qu'elle permet d'avoir un code optimisé et organiser en cas de travail en équipe ou départ du développeur. Et elle est plus performante que l'approche "Programmation impérative".

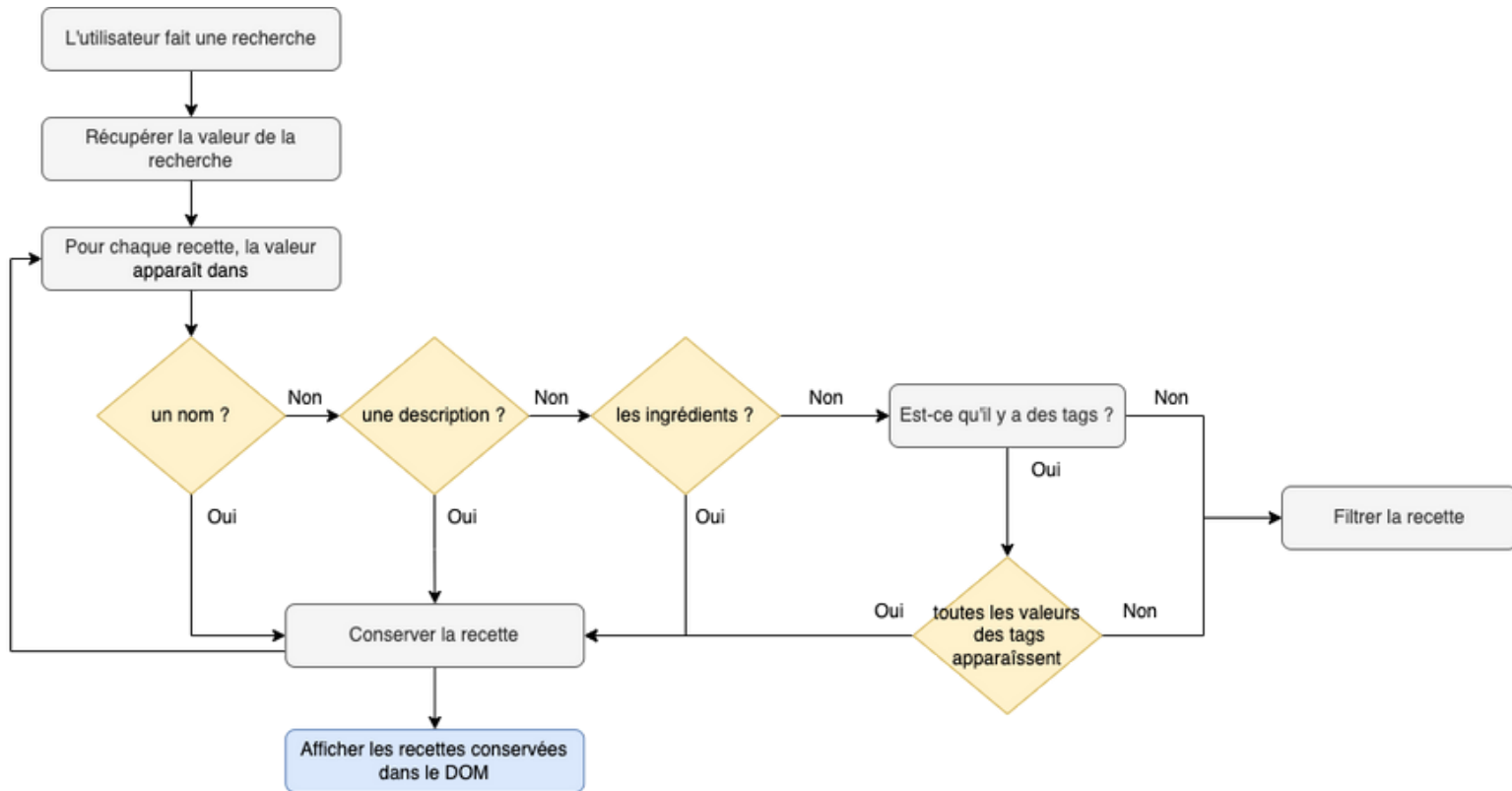


Figure 1 - Diagramme d'activité `filterRecipes()`