Fiche d'investigation de fonctionnalité

Fonctionnalité: Moteur de recherche principal	**Fonctionnalité #2**
Problématique : Trouver l'algorithme de recherche principal le plus performant. Cet algorithme doit retourner une liste de recette à partir d'une valeur (mots ou groupe de lettres saisis par l'utilisateur dans la recherche principale) Pour chaque recette retournée la valeur est soit comprise dans son titre, soit ses ingrédients ou bien sa description.	

Option 1: Les méthodes de l'objet array (cf Figure) Les méthodes de l'objet array sont conçues pour parcourir une liste et transformer chaque membre de cette liste et renvoyer une nouvelle liste ou appliquer une opération à chaque membre de la liste.	
Avantages • Facile à maintenir • Facile à lire • Ils correspondent mieux aux paradigmes de la programmation fonctionnelle • Possibilité d'utiliser le chaînage de méthodes pour empiler les méthodes les unes après les autres.	**Inconvénients** • Transforme l'array d'origine • Peut être difficile à appréhender pour un débutant

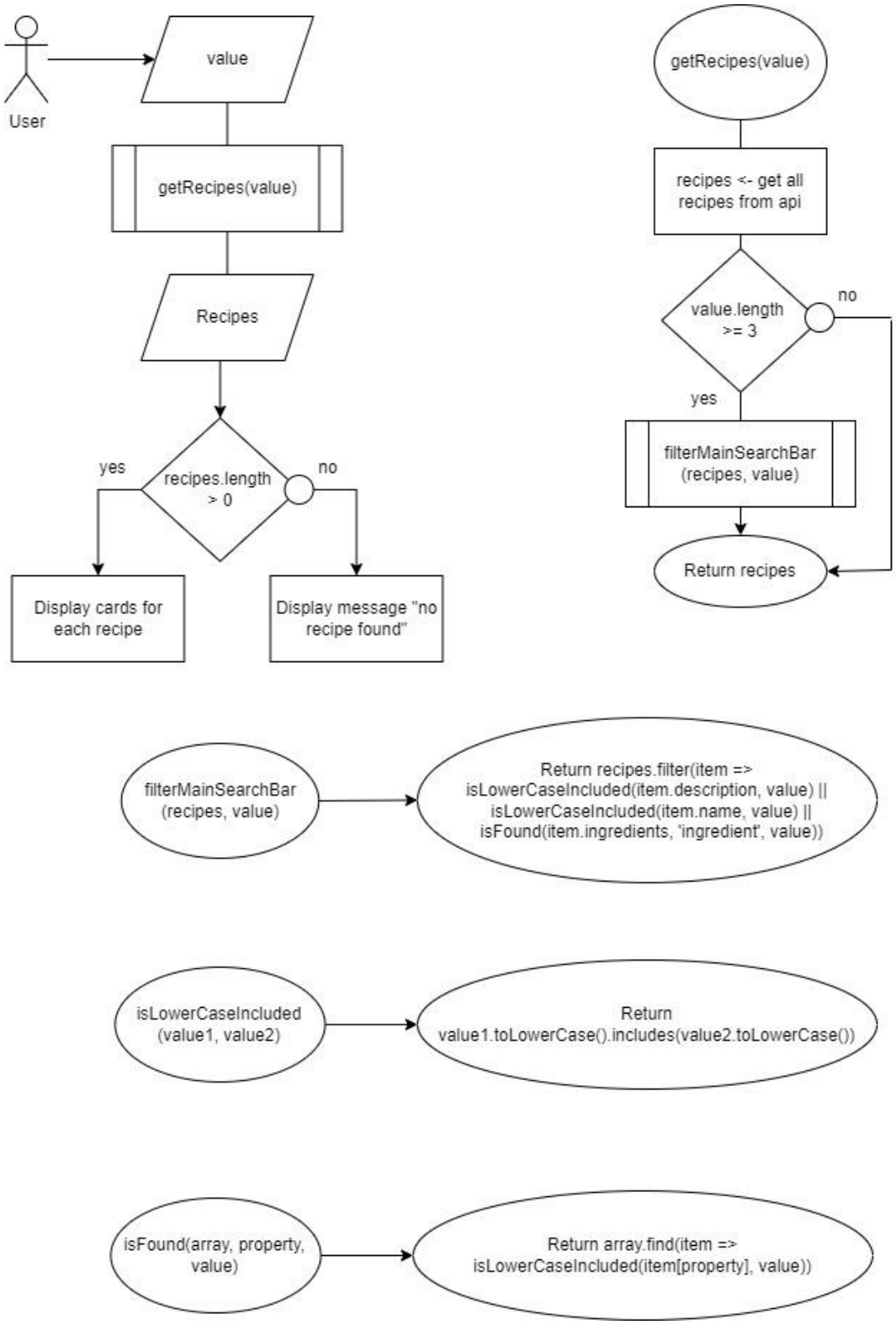
Option 2: Les boucles natives (cf Figure) Elles sont spécialement conçues pour itérer sur une liste tant qu'une condition est vraie	
Avantages • Facile à écrire et à comprendre pour un débutant	**Inconvénients** • Nécessite parfois beaucoup de mémoire • Plus de ligne de code

Solution retenue:
Bien que les boucles natives sont plus rapides dans certains cas (les navigateurs et les moteurs JS les ont optimisées pour qu'elles le soient).
Comparativement le gain de rapidité reste minime (1% Chrome, 8% Firefox selon tests JSBEN.CH)
La solution retenue dans un premier temps sont les méthodes de l'objet array notamment pour sa maintenabilité et sa lisibilité. Certaines optimisations doivent faire partie des bonnes pratiques générales. Si les performances sont importantes et que les tableaux utilisés sont très volumineux, il est préférable d'écrire un test pour vérifier cela puis arbitrer entre l'usage de l'option 1 ou l'option 2 en sachant qu'il s'agit actuellement d'une solution temporaire puisqu'à terme l'algorithme sera côté backend.

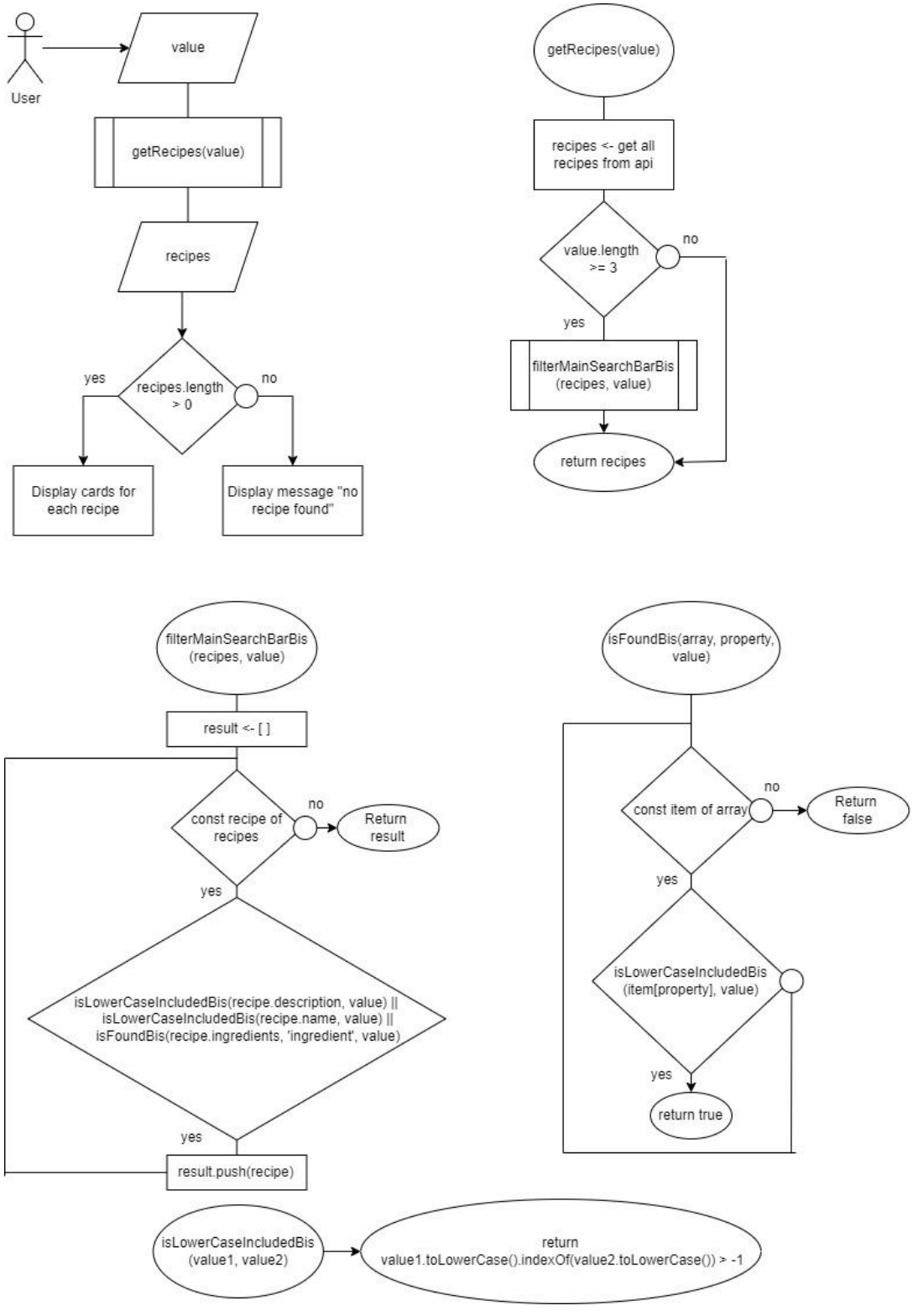
Références:

- https://troumad.developpez.com/C/algorigrammes/
- https://medium.com/@gabriellegianna92/loops-vs-array-methods-26999051ba45#:~:text=For%20loops%20are%20a%20looping,operation%20to%20each%20list%20member.

Annexes



Algorigramme option 1 (norme ISO 5807)



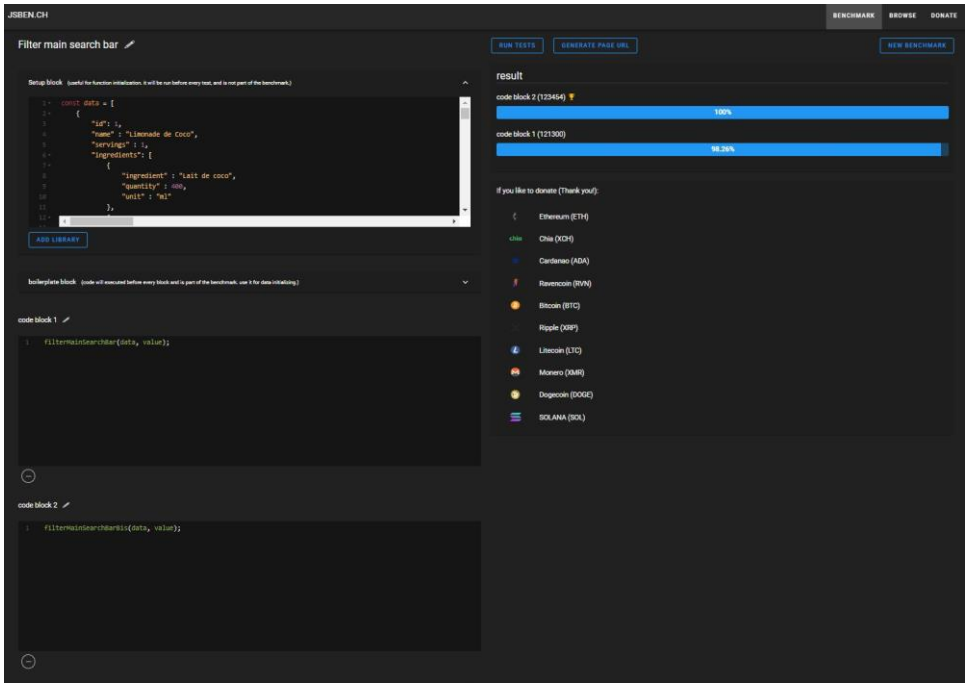
Algorigramme option 2 (norme ISO 5807)

Benchmark JSBEN.CH

Lien vers le test: https://jsben.ch/qEwVA

Résultats:

- Chrome:



- Firefox: