

Fiche d'investigation de fonctionnalité

Fonctionnalité Recherche	Fonctionnalité #1
Problématique Rechercher facilement et rapidement une recette par une recherche globale et une recherche par Critère.	

Option 1 : Recherche par filtre sur des elements de tableaux Dans cette option, nous utilisons les methodes des tableaux « filter ».	
Avantages <ul style="list-style-type: none"> ⊕Simplicité ⊕Lisibilité ⊕Maintenabilité ⊕Optimisé pour les performances par le moteur javascript 	Inconvénients <ul style="list-style-type: none"> ⊖Allocation memoire supplémentaire pour stocker le Nouveau tableau filtré ⊖ Inneficace si le tableau est trop volumineux
Nécessite au minimum 3 caractères dans la recherche principale pour déclencher la recherche	

Option 2 : Recherche par Boucle (itération sur des elements de tableaux) Dans cette option, nous utilisons de boucles,	
Avantages <ul style="list-style-type: none"> ⊕Meilleur performance en cas de verification multiple. Dans notre cas : recherche dans le nom, ensuite dans la description et les ingrédients 	Inconvénients <ul style="list-style-type: none"> ⊖Complexité accrue (nécessite plus de lignes de code) : Peut introduire des bugs plus facilement ⊖Lisibilité réduite : l'intention peut ne pas etre lisible au Premier coup d'oeil
Nécessite au minimum 3 caractères dans la recherche principale pour déclencher la recherche	

Solution retenue : Les performances étant meilleure, j'ai retenue l'option1 aussi pour sa maintenabilité et sa meilleure lisibilité pour les développeurs. Résultat JSBENCH https://jsben.ch/KgePo pour l' algorithme de recherche principale 50 recettes : 5712 ops par secondes pour l'algo avec la methode filter contre 4785 pour la boucle « for of ».
--

Test bench de l'algorithme de recherche principale

1^{er} test : Recherche avec les mots clés : « tomate citron » et 50 recettes

JSBEN.CH

BENCHMARKBROWSEDONATE

comparaison entre l'algorithme utilisant filter et l'algorithme utilisant les boucles

RUN TESTSGENERATE PAGE URLNEW BENCHMARK

Setup block (useful for function initialization. it will be run before every test, and is not part of the benchmark.)

```
1768      "unit": "grammes"
1769    }
1770  ],
1771  "time": 60,
1772  "description": "Préparer la frangipane : Mélanger le sucre la poudre d'a
1773  "appliance": "Four",
1774  "ustensils": ["rouleau à pâtisserie", "fouet"]
1775  }
1776  ];
1777
1778  const searchText = "tomate citron";
1779
```

ADD LIBRARY

boilerplate block (code will executed before every block and is part of the benchmark. use it for data initializing.)

algo utilisant la methode filter

```
1▼ function filterRecipes(recipes, searchText) {
2▼   if (searchText.length <= 2) {
3     return recipes;
4   }
5
6   const searchTextArray = searchText.split(" ");
7
8▼   return recipes.filter((recipe) => {
9     console.log(recipe);
10▼    return searchTextArray.every((text) => {
11      return (
12        recipe.name.toLowerCase().includes(text) ||
13        recipe.description.toLowerCase().includes(text) ||
14
```

result

algo utilisant la methode filter (6390) 🏆

100%

algo utilisant les boucles (for of) (5949)

93.1%

If you like to donate (Thank you!):

Ethereum (ETH)

Chia (XCH)

Cardano (ADA)

Ravencoin (RVN)

Bitcoin (BTC)

Ripple (XRP)

Litecoin (LTC)

Monero (XMR)

Dogecoin (DOGE)

SOLANA (SOL)

Test bench de l'algorithme de recherche principal

2ème test : Recherche avec les mots clés : «fdsfsd fddfsdf» et 50 recettes

JSBEN.CH

BENCHMARKBROWSEDONATE

comparaison entre l'algorithme utilisant filter et l'algorithme utilisant les boucles

RUN TESTSGENERATE PAGE URLNEW BENCHMARK

Setup block (useful for function initialization, it will be run before every test, and is not part of the benchmark.)

```
1 // @ts-nocheck
1769   "unit": "grammes"
1770   },
1771   "time": 60,
1772   "description": "Préparer la frangipane : Mélanger le sucre la poudre d'â
1773   "appliance": "Four",
1774   "ustensils": ["rouleau à pâtisserie", "fouet"]
1775 }
1776 ];
1777
1778 const searchText = "fdsfsd fddfsdf";
1779
```

ADD LIBRARY

boilerplate block (code will be executed before every block and is part of the benchmark. use it for data initializing)

algo utilisant la methode filter

```
1 function filterRecipes(recipes, searchText) {
2   if (searchText.length <= 2) {
3     return recipes;
4   }
5
6   const searchTextArray = searchText.split(" ");
7
8   return recipes.filter((recipe) => {
9     console.log(recipe);
10    return searchTextArray.every((text) => {
11      return (
12        recipe.name.toLowerCase().includes(text) ||
13        recipe.description.toLowerCase().includes(text) ||
14
```

result

algo utilisant la methode filter (6539) 🏆

100%

algo utilisant les boucles (for of) (5495)

84.03%

If you like to donate (Thank you!):

Ethereum (ETH)

Chia (XCH)

Cardano (ADA)

Ravencoin (RVN)

Bitcoin (BTC)

Ripple (XRP)

Litecoin (LTC)

Monero (XMR)

Dogecoin (DOGE)

SOLANA (SOL)

Test bench de l'algorithme de recherche principal

3ème test : Recherche avec les mots clés : «poulet coco» et 100 recettes

JSBEN.CH

BENCHMARKBROWSEDONATE

comparaison entre l'algorithme utilisant filter et l'algorithme utilisant les boucles

Setup block (useful for function initialization, it will be run before every test, and is not part of the benchmark.)

```
3538 },
3539 {
3540   "ingredient": "Sucre glace",
3541   "quantity": 500,
3542   "unit": "grammes"
3543 }
3544 ],
3545 "time": 60,
3546 "description": "Préparer la frangipane : Mélanger le sucre la poudre d'a
3547 "appliance": "Four",
3548 "ustensils": ["rouleau à pâtisserie", "fouet"]
3549 }
3550
```

ADD LIBRARY

boilerplate block (code will executed before every block and is part of the benchmark. use it for data initializing)

algo utilisant la methode filter

```
1 function filterRecipes(recipes, searchText) {
2   if (searchText.length <= 2) {
3     return recipes;
4   }
5
6   const searchTextArray = searchText.split(" ");
7
8   return recipes.filter((recipe) => {
9     console.log(recipe);
10    return searchTextArray.every((text) => {
11      return (
12        recipe.name.toLowerCase().includes(text) ||
13        recipe.description.toLowerCase().includes(text) ||
14        recipe.ingredients.toLowerCase().includes(text) ||
15        recipe.appliance.toLowerCase().includes(text) ||
16        recipe.ustensils.toLowerCase().includes(text)
17      );
18    });
19  });
20 }
```

result

algo utilisant la methode filter (3520) 🏆

100%

algo utilisant les boucles (for of) (2702)

76.76%

If you like to donate (Thank you!):

Ethereum (ETH)

Chia (XCH)

Cardano (ADA)

Ravencoin (RVN)

Bitcoin (BTC)

Ripple (XRP)

Litecoin (LTC)

Monero (XMR)

Dogecoin (DOGE)

SOLANA (SOL)

Annexes

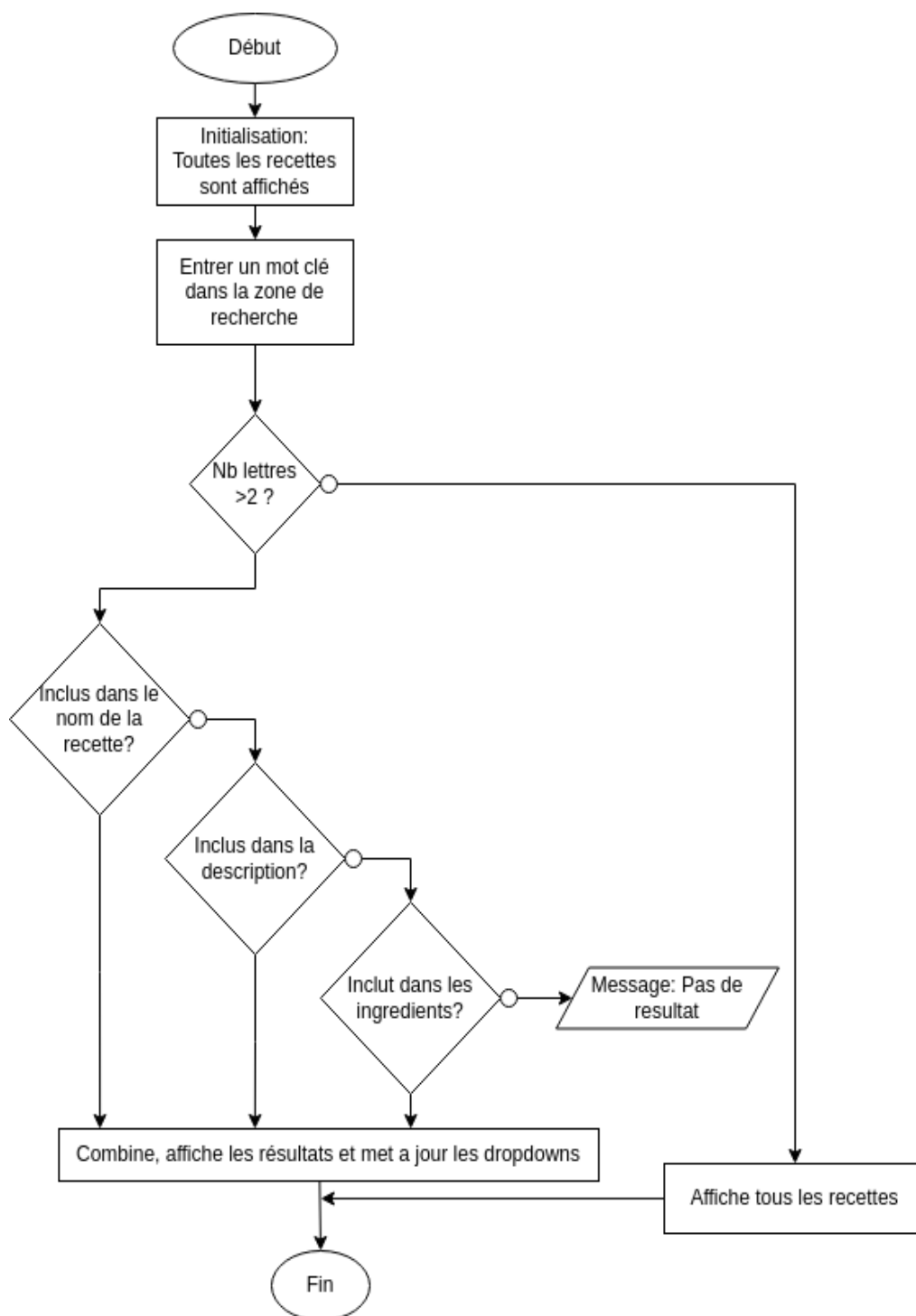


Figure 1 - Diagramme d'activité de la recherche principale

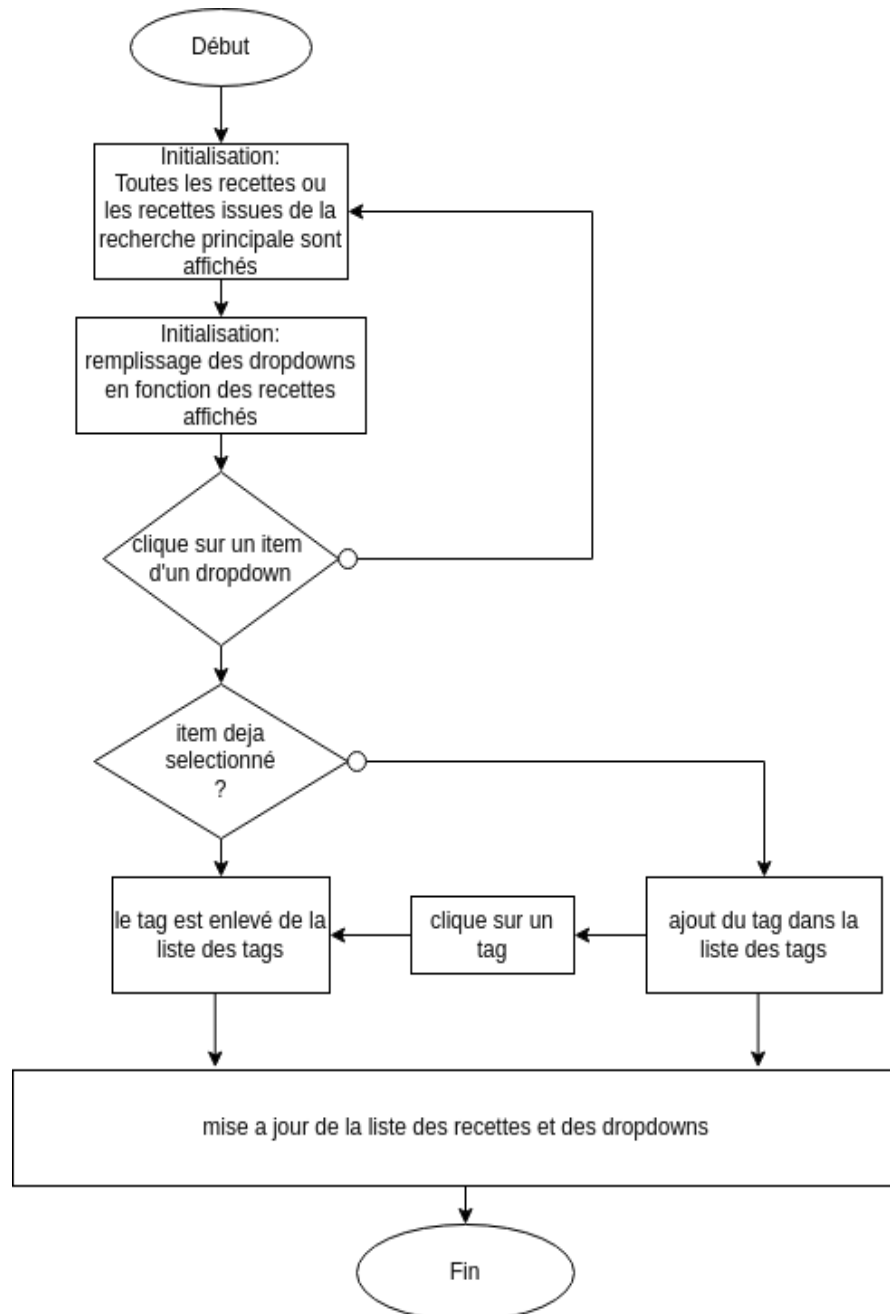


Figure 2 : Diagramme d'activité de la recherche par critère