

Fiche d'investigation de fonctionnalité

Fonctionnalité : Moteur de recherche

Problématique : Afin de se différencier des concurrents sur le marché, le site « les petits plats » souhaite mettre en place un moteur de recherche fluide et rapide.

Option 1 : Algorithme basé sur des boucles natives :

Cette option utilise les boucles « for et while » pour parcourir toutes les données.

Avantages :

- Compatibilité avec les anciens navigateurs

Inconvénients :

- Lisibilité du code,

- Risques d'erreurs accidentelles dues à la complexité du code,

- Implémentation de nouveau filtre plus lente.

Option 2 : Algorithme basé sur la programmation fonctionnelle

Cette option utilise les standards contemporains de Javascript basé sur les méthodes de l'objet Array « foreach, filter, map et reduce ».

Avantages :

- Lisibilité du code,

- Moins de risques d'erreurs accidentelles,

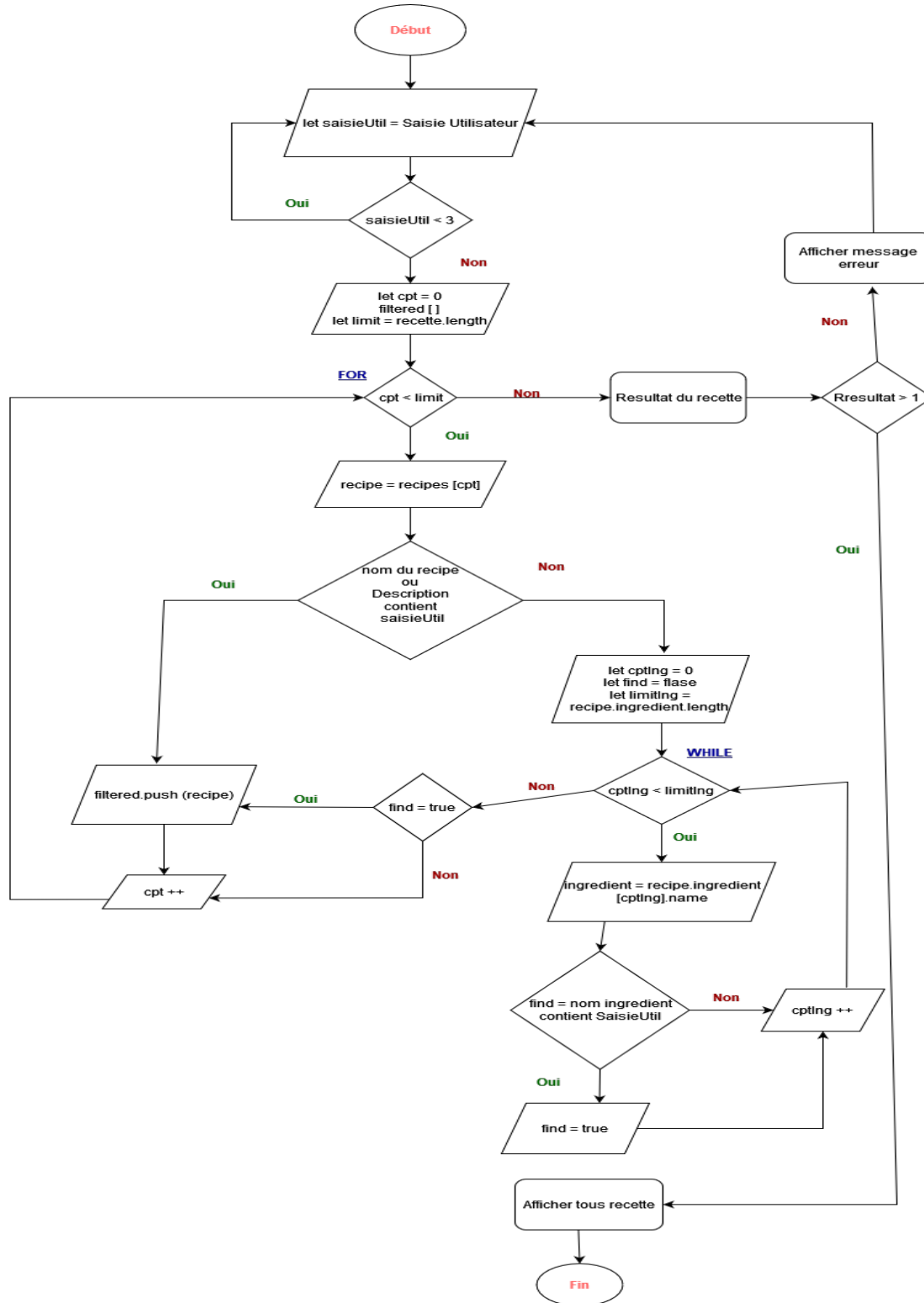
- Implémentations de nouveau filtre rapide

Inconvénients :

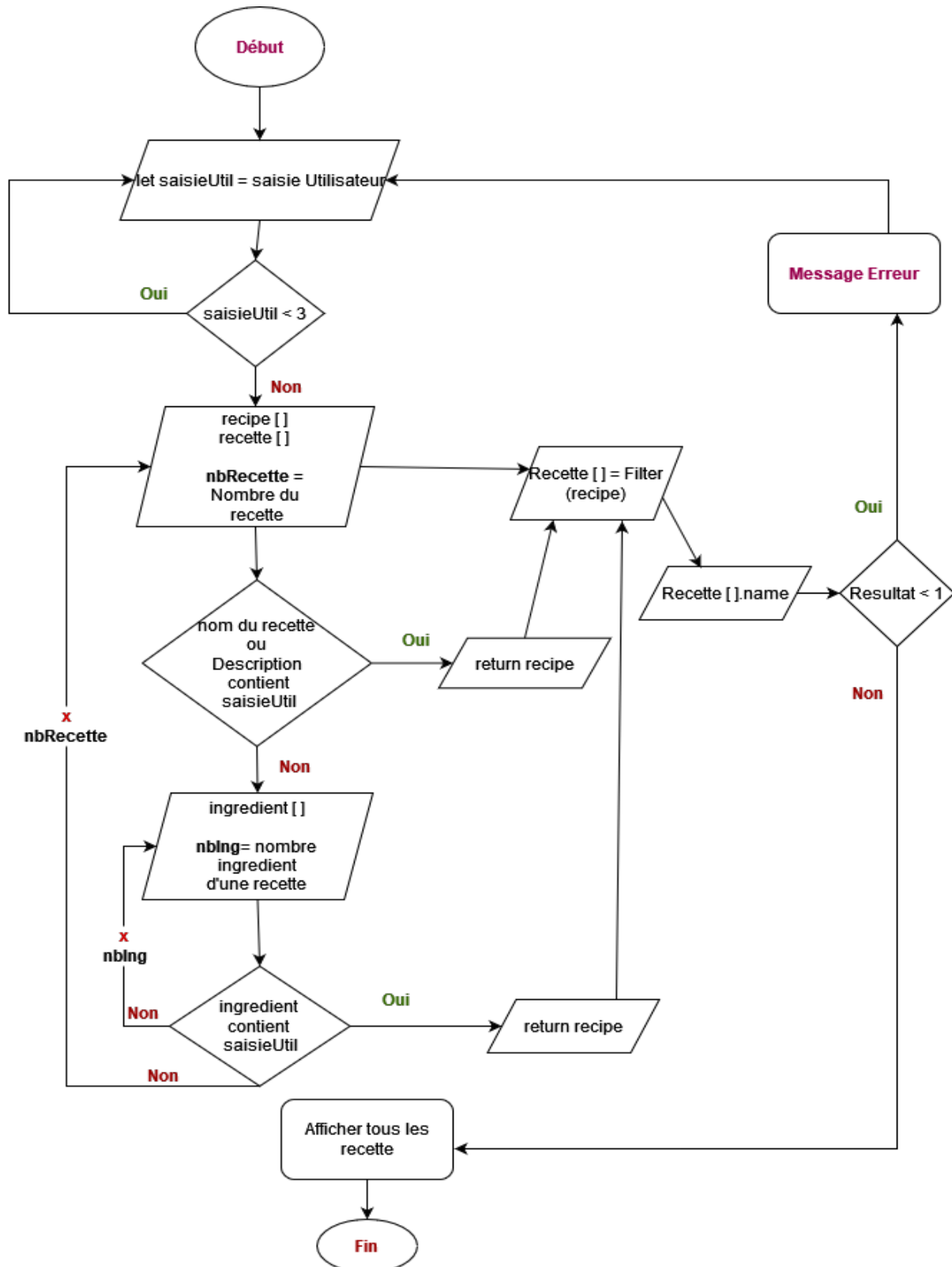
- Compatibilité avec les anciens navigateurs.

Solution aux tests réalisés entre les 2 méthodes sur jsben.ch, l'option 2 a été retenue. Elle s'est démarquée par sa rapidité (entre 16 à 26% plus rapide que l'option 1. Et il est clair qu'elle se démarque également par sa maintenabilité et la facilité d'implémenter de nouvelle option de recherche par le futur

Algorithme Option 1 : Native



Algorithme Option 1 : Fonctionnelle





Fiche d'investigation fonctionnalité

Résultats des tests

JSBEN.CH

BENCHMARKBROWSEDONATE

no title (put title and/or keywords here, which describes your test)

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.)

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

Algo Native

```
1765 let finding = recipe.ingredients.length;
1766 while (cptCh < finding) {
1767   const ingredient = recipe.ingredients[cptCh].name;
1768   if (cherche = ingredient.includes(saisieUtil)) {
1769     filtered.push(recipe);
1770     break;
1771   }
1772   cptCh++;
1773 }
1774 }
1775 }
1776 }
```

Algo Fonctionnel

```
1749 data.forEach((recipe) => recette.push(new Recipe(recipe))); //prendre tous les donné et mettre dans nouvelle tab
1750
1751 const saisieUtil = 'coco'; //variable saisie utilisateur 'coco'
1752 recette = recette.filter((recipe) => { //prendre tous les recette si...
1753   if (recipe.name.includes(saisieUtil)) return true; //si le saisie est dans le nom
1754   if (recipe.description.toLowerCase().includes(saisieUtil)) return true; // si le saisie est dans description
1755   else if (!recipe.ingredients.find((ingredient) => ingredient.name.includes(saisieUtil))) return true; //dans
1756
1757   return false;
1758 });
```

result

Algo Fonctionnel (23821) 🏆

100%

Algo Native (19759)

82.95%

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)



Fiche d'investigation fonctionnalité