

## Fiche d'investigation de fonctionnalité

Fonctionnalité : Algorithme de recherche	Fonctionnalité #1
<p>Problématique : Créé un algorithme de recherche principal performant. cette algorithme a pour fonction de retourner une liste de recette à partir d'un valeur (mot ou groupe de lettres saisis par l'utilisateur dans la barre de recherche principale)</p> <p>Pour chaque recette retourner la valeur est soit comprise dans son titre, soit ses ingrédients ou bien sa description.</p>	

### Option 1 : Les Méthodes de l'objet Array

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 lire
- tien sur une ligne de code
- 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

Elles sont spécialement conçues pour itérer sur une liste tant qu'une condition est vraie

#### Avantages

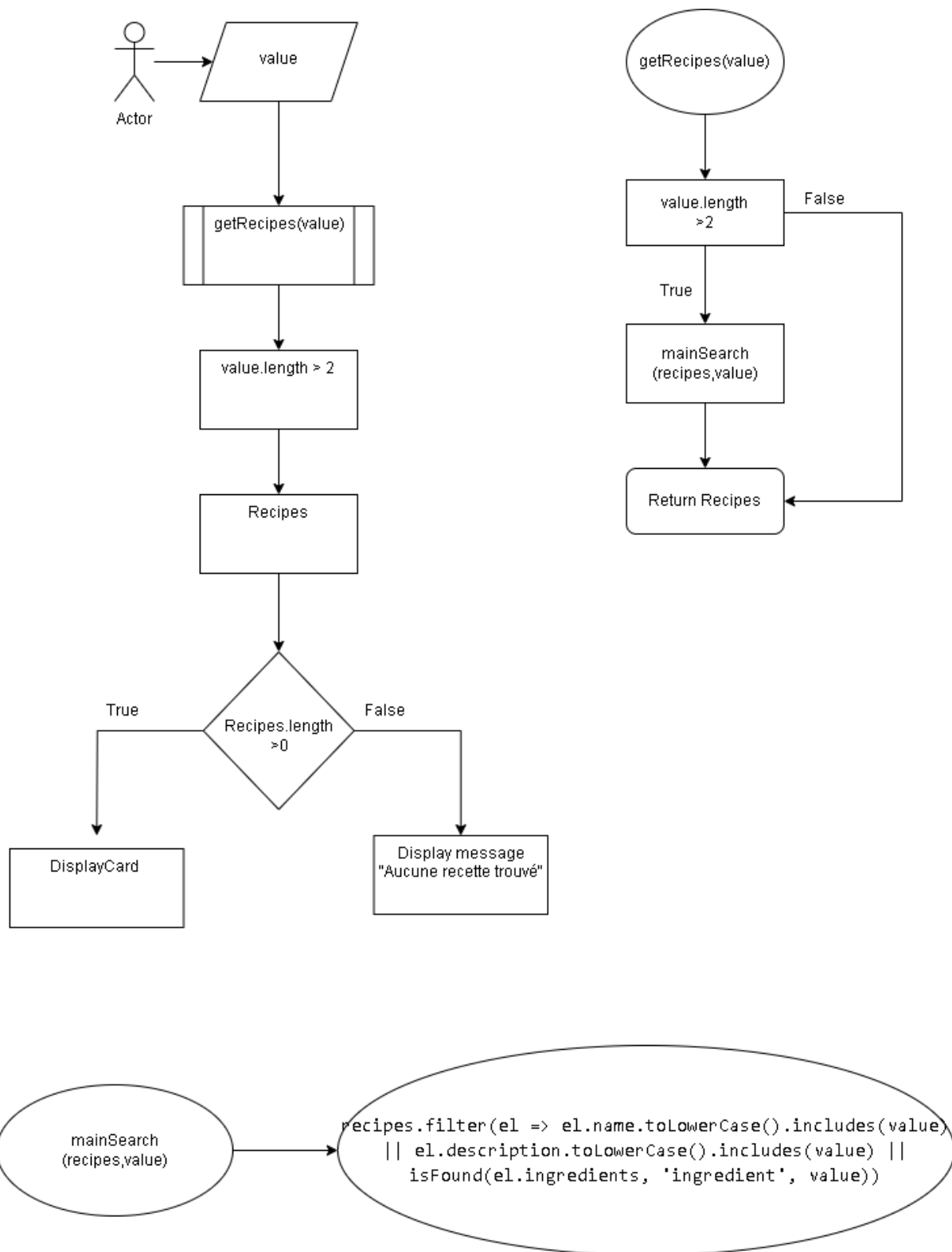
- facile à comprendre
- plus performant

#### Inconvénients

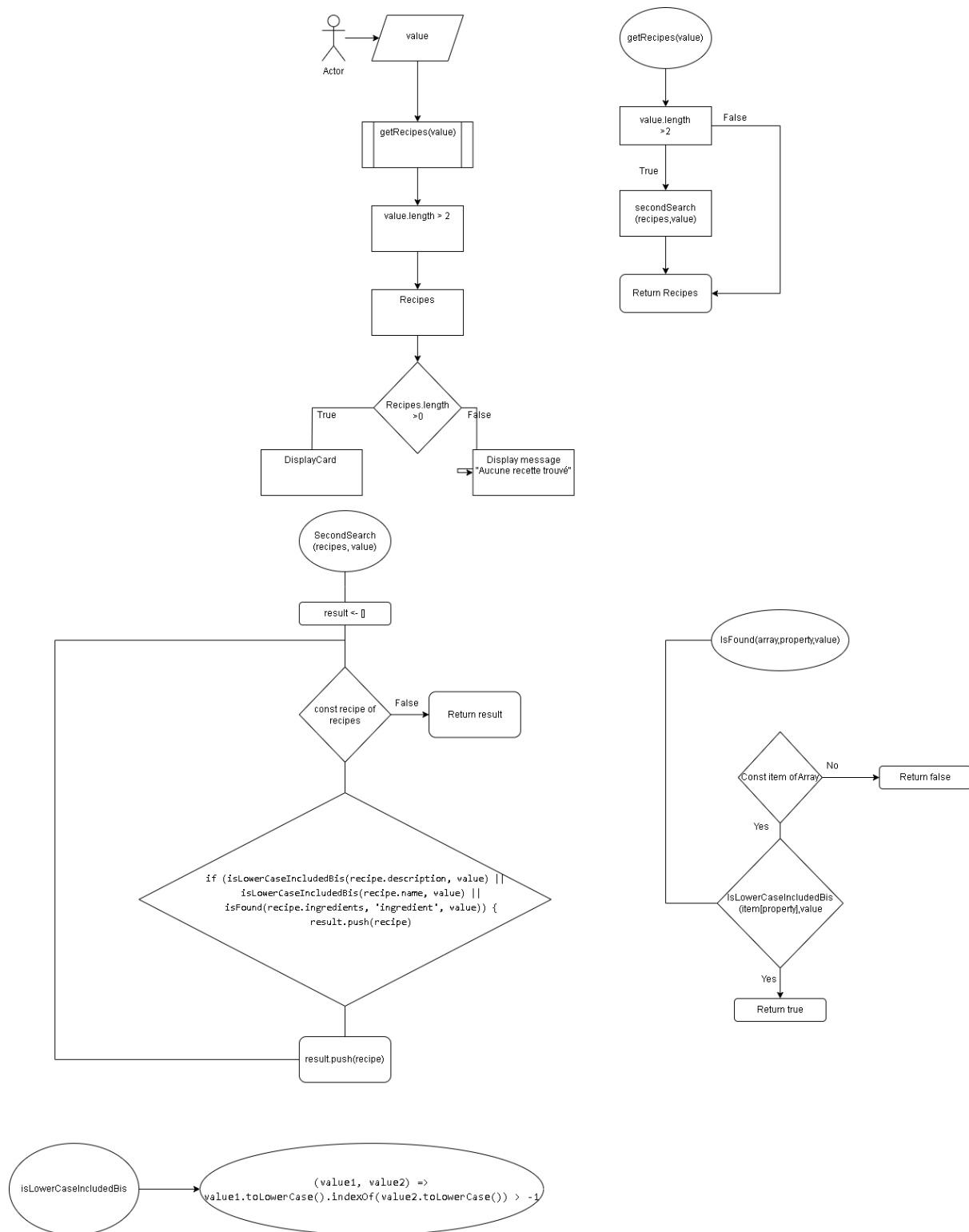
- plus de ligne de code
- Peut être difficile à appréhender pour un débutant

**Solution retenue :** La solution retenu a été l'Option 2 Les boucles Natives car , plus performant au bench.js (39 226 pour la solution boucle native contre 35870 pour la méthode de l'objet Array , malgré que le code est plus long et peut paraître être compliquée à appréhender au premier abord, elle reste assez simple.

## Algorithme (Méthode Object Array)



## Algorithme (Boucle Native )



## Benchmark JSBEN.CH

lien vers le test : <https://jsben.ch/zJ02l>

### Résultats sur Navigateur Chrome :

JSBEN.CH

BENCHMARKBROWSEDONATE

Test algorithme de recherche Ali El baamrani

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 const isFound = (array, property, value) => array.find(item => isLowerCaseIncluded(item, value, property))
2 const isLowerCaseIncluded = (value1, value2) => value1.toLowerCase().includes(value2.toLowerCase())
3
4 const mainSearch = (recipes, value) => recipes.filter(el => el.name.toLowerCase().includes(value.toLowerCase()))
5
6 const isLowerCaseIncludedBis = (value1, value2) => value1.toLowerCase().indexOf(value2.toLowerCase()) !== -1
7
8 const isFoundBis = (array, property, value) => {
9   for (const item of array) {
10     if (isLowerCaseIncluded(item, value, property)) {
11       return true
12     }
13   }
14   return false
15 }
```

ADD LIBRARY

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

```
1 const recipes = [
2   {
3     "id": 1,
4     "name": "Limonade de Coco",
5     "servings": 1,
6     "ingredients": [
7       {
8         "ingredient": "Lait de coco",
9         "quantity": 400,
10        "unit": "ml"
11      }
12    ]
13  },
14  {
15    "id": 2,
16    "name": "Smoothie de Fruits",
17    "servings": 1,
18    "ingredients": [
19      {
20        "ingredient": "Banane",
21        "quantity": 1,
22        "unit": "fruit"
23      },
24      {
25        "ingredient": "Framboise",
26        "quantity": 100,
27        "unit": "g"
28      },
29      {
30        "ingredient": "Yaourt grec",
31        "quantity": 100,
32        "unit": "g"
33      }
34    ]
35  },
36  {
37    "id": 3,
38    "name": "Salade de Pois Chiches",
39    "servings": 1,
40    "ingredients": [
41      {
42        "ingredient": "Pois chiches",
43        "quantity": 1,
44        "unit": "can"
45      },
46      {
47        "ingredient": "Tomate",
48        "quantity": 1,
49        "unit": "fruit"
50      },
51      {
52        "ingredient": "Ciboule",
53        "quantity": 1,
54        "unit": "stalk"
55      },
56      {
57        "ingredient": "Huile d'olive",
58        "quantity": 2,
59        "unit": "tbsp"
60      },
61      {
62        "ingredient": "Sel",
63        "quantity": 1,
64        "unit": "pinch"
65      }
66    ]
67  },
68  {
69    "id": 4,
70    "name": "Tarte aux Pommes",
71    "servings": 1,
72    "ingredients": [
73      {
74        "ingredient": "Pâte à tarte",
75        "quantity": 1,
76        "unit": "sheet"
77      },
78      {
79        "ingredient": "Pommes",
80        "quantity": 2,
81        "unit": "fruit"
82      },
83      {
84        "ingredient": "Sucre",
85        "quantity": 100,
86        "unit": "g"
87      },
88      {
89        "ingredient": "Cannelle",
90        "quantity": 1,
91        "unit": "tsp"
92      }
93    ]
94  },
95  {
96    "id": 5,
97    "name": "Gâteau au Chocolat",
98    "servings": 1,
99    "ingredients": [
100     {
101       "ingredient": "Beurre",
102       "quantity": 100,
103       "unit": "g"
104     },
105     {
106       "ingredient": "Sucre",
107       "quantity": 100,
108       "unit": "g"
109     },
110     {
111       "ingredient": "Œufs",
112       "quantity": 2,
113       "unit": "egg"
114     },
115     {
116       "ingredient": "Farine",
117       "quantity": 100,
118       "unit": "g"
119     },
120     {
121       "ingredient": "Cacao",
122       "quantity": 50,
123       "unit": "g"
124     }
125    ]
126  }
127 ]
```

result

Boucle Native (39226) 100%

Méthode Object Array (35870) 91.44%

If you like to donate (Thank you!):

Ethereum (ETH)

Chia (XCH)

Cardano (ADA)

Ravencoin (RVN)

### Résultats sur Navigateur Firefox :

JSBEN.CH

BENCHMARKBROWSEDONATE

Test algorithme de recherche Ali El baamrani

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 const isFound = (array, property, value) => array.find(item => isLowerCaseIncluded(item, value, property))
2 const isLowerCaseIncluded = (value1, value2) => value1.toLowerCase().includes(value2.toLowerCase())
3
4 const mainSearch = (recipes, value) => recipes.filter(el => el.name.toLowerCase().includes(value.toLowerCase()))
5
6 const isLowerCaseIncludedBis = (value1, value2) => value1.toLowerCase().indexOf(value2.toLowerCase()) !== -1
7
8 const isFoundBis = (array, property, value) => {
9   for (const item of array) {
10     if (isLowerCaseIncluded(item, value, property)) {
11       return true
12     }
13   }
14   return false
15 }
```

ADD LIBRARY

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

```
1 const recipes = [
2   {
3     "id": 1,
4     "name": "Limonade de Coco",
5     "servings": 1,
6     "ingredients": [
7       {
8         "ingredient": "Lait de coco",
9         "quantity": 400,
10        "unit": "ml"
11      }
12    ]
13  },
14  {
15    "id": 2,
16    "name": "Smoothie de Fruits",
17    "servings": 1,
18    "ingredients": [
19      {
20        "ingredient": "Banane",
21        "quantity": 1,
22        "unit": "fruit"
23      },
24      {
25        "ingredient": "Framboise",
26        "quantity": 100,
27        "unit": "g"
28      },
29      {
30        "ingredient": "Yaourt grec",
31        "quantity": 100,
32        "unit": "g"
33      }
34    ]
35  },
36  {
37    "id": 3,
38    "name": "Salade de Pois Chiches",
39    "servings": 1,
40    "ingredients": [
41      {
42        "ingredient": "Pois chiches",
43        "quantity": 1,
44        "unit": "can"
45      },
46      {
47        "ingredient": "Tomate",
48        "quantity": 1,
49        "unit": "fruit"
50      },
51      {
52        "ingredient": "Ciboule",
53        "quantity": 1,
54        "unit": "stalk"
55      },
56      {
57        "ingredient": "Huile d'olive",
58        "quantity": 2,
59        "unit": "tbsp"
60      },
61      {
62        "ingredient": "Sel",
63        "quantity": 1,
64        "unit": "pinch"
65      }
66    ]
67  },
68  {
69    "id": 4,
70    "name": "Tarte aux Pommes",
71    "servings": 1,
72    "ingredients": [
73      {
74        "ingredient": "Pâte à tarte",
75        "quantity": 1,
76        "unit": "sheet"
77      },
78      {
79        "ingredient": "Pommes",
80        "quantity": 2,
81        "unit": "fruit"
82      },
83      {
84        "ingredient": "Sucre",
85        "quantity": 100,
86        "unit": "g"
87      },
88      {
89        "ingredient": "Cannelle",
90        "quantity": 1,
91        "unit": "tsp"
92      }
93    ]
94  },
95  {
96    "id": 5,
97    "name": "Gâteau au Chocolat",
98    "servings": 1,
99    "ingredients": [
100     {
101       "ingredient": "Beurre",
102       "quantity": 100,
103       "unit": "g"
104     },
105     {
106       "ingredient": "Sucre",
107       "quantity": 100,
108       "unit": "g"
109     },
110     {
111       "ingredient": "Œufs",
112       "quantity": 2,
113       "unit": "egg"
114     },
115     {
116       "ingredient": "Farine",
117       "quantity": 100,
118       "unit": "g"
119     },
120     {
121       "ingredient": "Cacao",
122       "quantity": 50,
123       "unit": "g"
124     }
125    ]
126  }
127 ]
```

result

Boucle Native (43802) 100%

Méthode Object Array (38582) 88.08%

If you like to donate (Thank you!):

Ethereum (ETH)

Chia (XCH)

Cardano (ADA)

Ravencoin (RVN)

Bitcoin (BTC)

Ripple (XRP)

Litecoin (LTC)

Monero (XMR)