

Fonctionnalité : Recherche de recettes + tags	Fonctionnalité n°1
Problématique : accéder rapidement grâce à la fonctionnalité de recherche à une recette correspond au besoin de l'utilisateur que ce soit liés aux titre, ingrédients ou à sa description.	

Algorithme 1 : Recherche de recette + tags	
Dans cette option, le code est composé de boucle et de méthode avancées de tri et de recherche dans le tableau. Afin de traiter de la manière la plus rapide les recettes rechercher.	
Avantages <ul style="list-style-type: none"> • Ecriture du code simple et compréhensible • Simplicité du recherche dans un tableau 	Inconvénients <ul style="list-style-type: none"> • Complexité de l'utilisation des méthodes de recherche avancées
Nombre de champs minimum à remplir : 0 Nombre de champs optionnelles : 4 zone de recherche, zone de texte dans liste d'ingrédients, zone de texte dans liste d'appareils et zone de texte dans liste de ustensiles. Nombre de tags : 3 tags (ingrédients, appareils, ustensiles)	

Algorithme 2 : Recherche de recette + tags	
Dans cette option, le code est composé de boucle et de méthode native de javascript de tri et de recherche dans le tableau. Afin de traiter de la manière la plus rapide les recettes rechercher.	
Avantages <ul style="list-style-type: none"> • Utilisation de la connaissance de bases de javascript • Localiser les problèmes facilement 	Inconvénients <ul style="list-style-type: none"> • Code plus complexe • Exécution lente du code
Nombre de champs minimum à remplir : 0 Nombre de champs optionnelles : 4 zone de recherche, zone de texte dans liste d'ingrédients, zone de texte dans liste d'appareils et zone de texte dans liste de ustensiles. Nombre de tags : 3 tags (ingrédients, appareils, ustensiles)	

Solution retenue : Nous avons retenu le premier algorithme avec usage de boucle et méthode avancées puisqu'ils permet une exécution plus rapide et un code facile à comprendre

<pre>forEach, filter function finished 698 k ops/s ± 2.21% Fastest</pre>	<pre>list.forEach(recipe => { recipe.ingredients.forEach(ingredient=>{ if (ingredient.ingredient.toLowerCase() === searchelement) { liste.push(recipe); } }); }); break; case "description": liste = list.filter(item => item.description.toLowerCase().indexOf(searchelement) > -1); break; case "name": liste = list.filter(item => item.name.toLowerCase().indexOf(searchelement) > -1); break; default : console.log("data not found"); } return liste; } searchBy (recipes,'name','limonade')</pre>
<pre>for loop finished 684 k ops/s ± 0.65% 2 % slower</pre>	<pre>case "description": for(let i=0; i<list.length;i++){ if(list[i].description.toLowerCase().indexOf(searchelement) > -1){ liste.push(list[i]); } } break; case "name": for(let i=0; i<list.length;i++){ if(list[i].name.toLowerCase().indexOf(searchelement) > -1){ liste.push(list[i]); } } break; default : console.log("data not found"); } return liste; } searchBy (recipes,'name','limonade')</pre>

