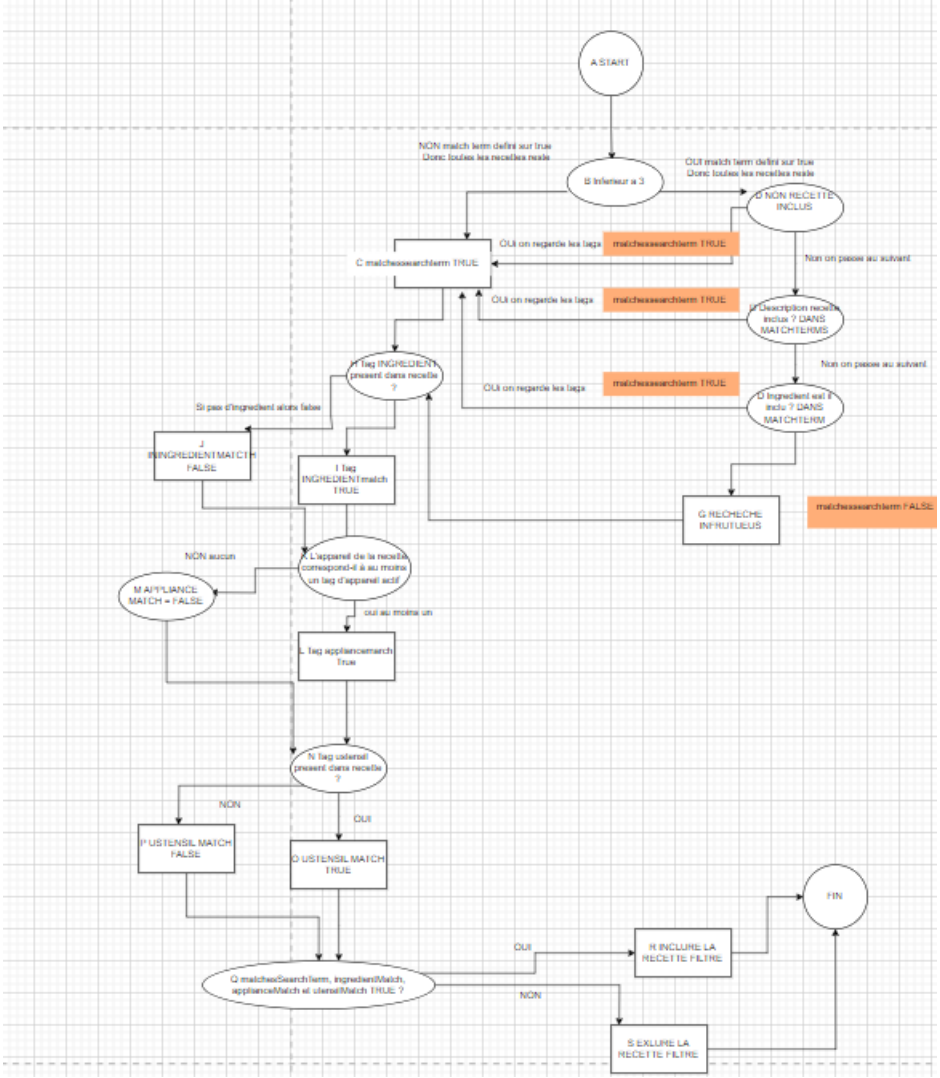
Choix de l’algorithme : Search and Tag

1 : Algorythme 1 Search and Tag :



I Effet : Filtre les recettes en utilisant la méthode filter de JavaScript, qui est chaînée avec d'autres méthodes comme includes et some.

II Fonctionnement :

Vérifie si le terme de recherche correspond à la recette (nom, description, ou ingrédients).

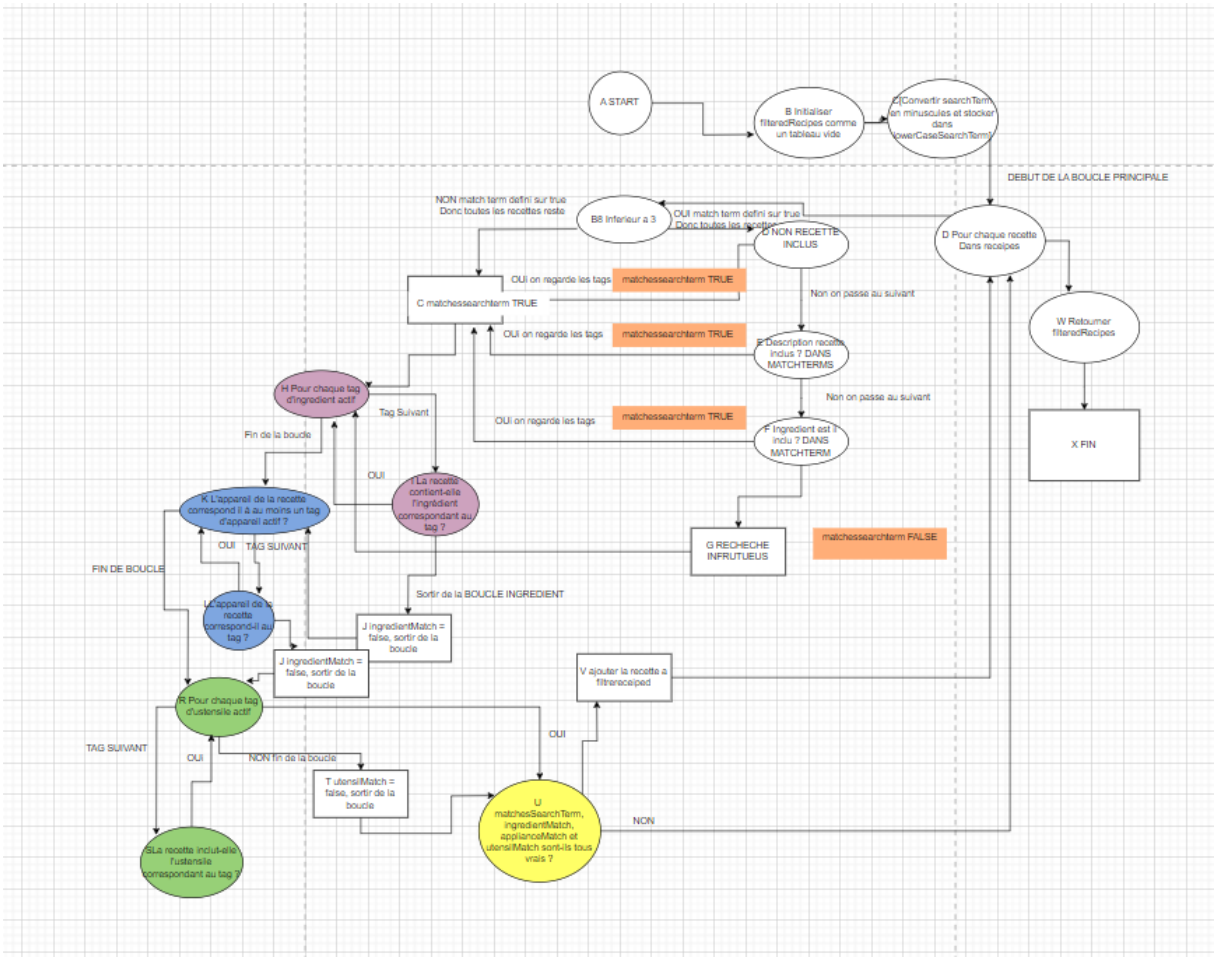
Vérifie la présence de tous les tags d'ingrédients dans la recette.

Vérifie la correspondance du tag d'appareil avec l'appareil de la recette.

Vérifie la présence de tous les tags d'ustensiles dans la recette.

III Efficacité : Cette méthode est généralement plus concise et peut être plus lisible. Elle utilise des fonctions intégrées de haut niveau qui sont optimisées et peut être plus rapide sur les petits ensembles de données. Cependant, pour les gros ensembles de données, l'utilisation répétée de some et every peut être coûteuse en performance car chaque recette est vérifiée à plusieurs reprises pour chaque tag.

2 Algorythme 2 Boucle



I Effet : Filtre les recettes en utilisant une boucle for classique et des structures de contrôle conditionnelles.

II Fonctionnement :

Passe en revue chaque recette individuellement.

Utilise une boucle interne pour vérifier la correspondance du terme de recherche dans les ingrédients si nécessaire.

Vérifie chaque tag d'ingrédient, d'appareil et d'ustensile pour chaque recette.

III Efficacité : Cette méthode offre un contrôle plus fin sur le processus de filtrage et peut être plus efficace si le terme de recherche est couramment trouvé sans avoir

Pourquoi avoir choisi l'algorythme Search and tag ?

Formule utiliser dans chrome pour tester les recherche

Formule applique :

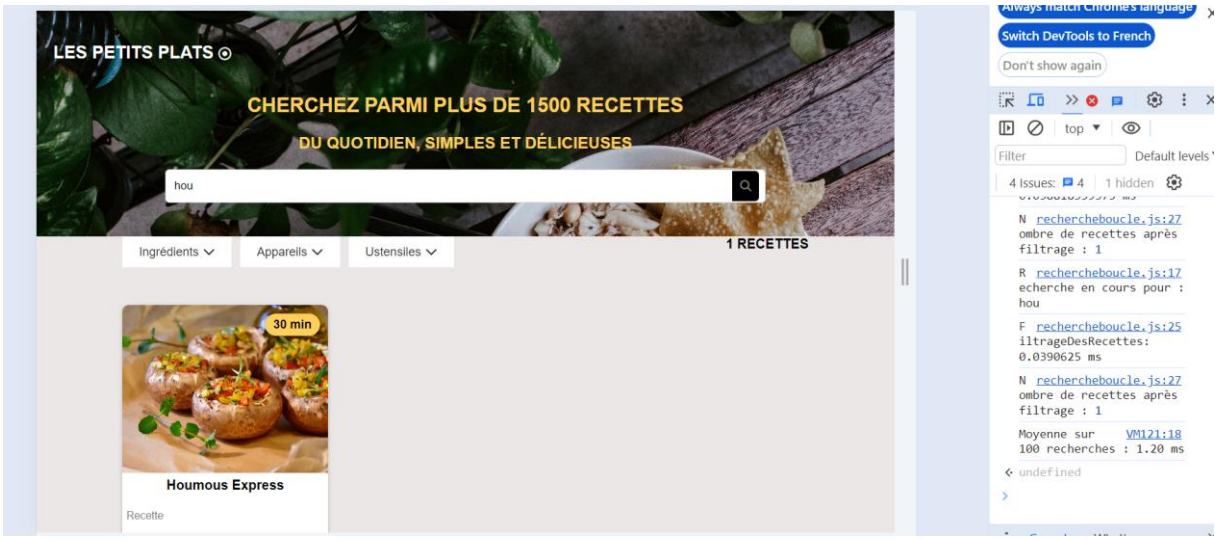
```
function testSearchPerformance() {
    const searchInput = document.querySelector('.search-container input');
    const searchTerm = "hou";

    const times = [];

    for (let i = 0; i < 100; i++) {
      searchInput.value = searchTerm;

      const start = performance.now();
      searchInput.dispatchEvent(new Event('input'));
      const end = performance.now();

      times.push(end - start);
    }

    const average = times.reduce((a, b) => a + b) / times.length;
    console.log(`Moyenne sur 100 recherches : ${average.toFixed(2)} ms`);
  }

  testSearchPerformance();
```

1 Test algorythme filtre100 fois

Entre 0.9 et 1.3

2 test algorythme boucle 100 fois

Entre 1.2 et 1.5



Même si les 2 font très bien le travail la fonction filtre un est plus en adéquation avec le codage moderne et performe plus pour le green code. Il est egalemetn legerement plus rapide

Cependant en fonction du nombre de donnée récolté et de recette il sera peut être intéressant de retester notamment la vitesse du site avec la totalité des recettes.