Fiche d’investigation de fonctionnalité

Fonctionnalité : Recherche	**Fonctionnalité #2**
Problématique : Permettre à l’utilisateur de trouver rapidement des recettes en fonction de mots-clés et de filtres avancés (ingrédients, ustensiles, appareils), tout en gardant l’interface simple et fluide.	

Option 1 : Recherche classique via la barre de recherche

L’utilisateur saisit 3 caractères ou plus, et un tri est appliqué immédiatement en passant par les noms de recettes, leurs description et leurs ingrédients.

Avantages	**Inconvénients**
Résultats pertinents dès la saisis	La recherche n’est pas optimale et il peut rester de nombreux résultats

Comportement :

- Saisie de 3 caractères minimum
- Résultats peu pertinents si le mot-clé est vague
- Aucune mise à jour dynamique des filtres

Option 2 : Recherche de recettes avec les filtres avancés

L’utilisateur ajoute un filtre Ingrédients, Appareils et/ou Ustensils. Cette approche est simple mais peut apporter de nombreux résultats. Les filtres sont mis à jour après chaque ajout de filtre afin qu’il ne reste que des options d’affinages.

Avantages	**Inconvénients**
⊕ Facile à implémenter ⊕ Pas de surcharge visuelle	⊖ Peu de personnalisation ⊖ Le nombre de recette retourné peut être conséquent

Comportement :

- Choisit un filtre dans l’un des trois menus déroulants
- Met à jour les menus déroulants et les recettes proposés

Option 3 : Système de recherche dynamique avec filtres avancés (cf Figure)

L'utilisateur commence à taper dans la barre de recherche.
À partir de 3 caractères, le système filtre dynamiquement les recettes par titre, ingrédients, description.
Les filtres avancés (ingrédients, appareils, ustensiles) sont mis à jour en fonction des résultats obtenus.
Si aucune recette ne correspond, un message explicite est affiché.

Avantages	**Inconvénients**
⊕ Résultats pertinents dès la saisie ⊕ Mécanisme de filtrage enrichi et personnalisé ⊕ Tags interactifs pour affiner la recherche ⊕ Expérience utilisateur fluide et moderne	⊖ Complexité technique plus élevée ⊖ Doit gérer les cas de non-correspondance

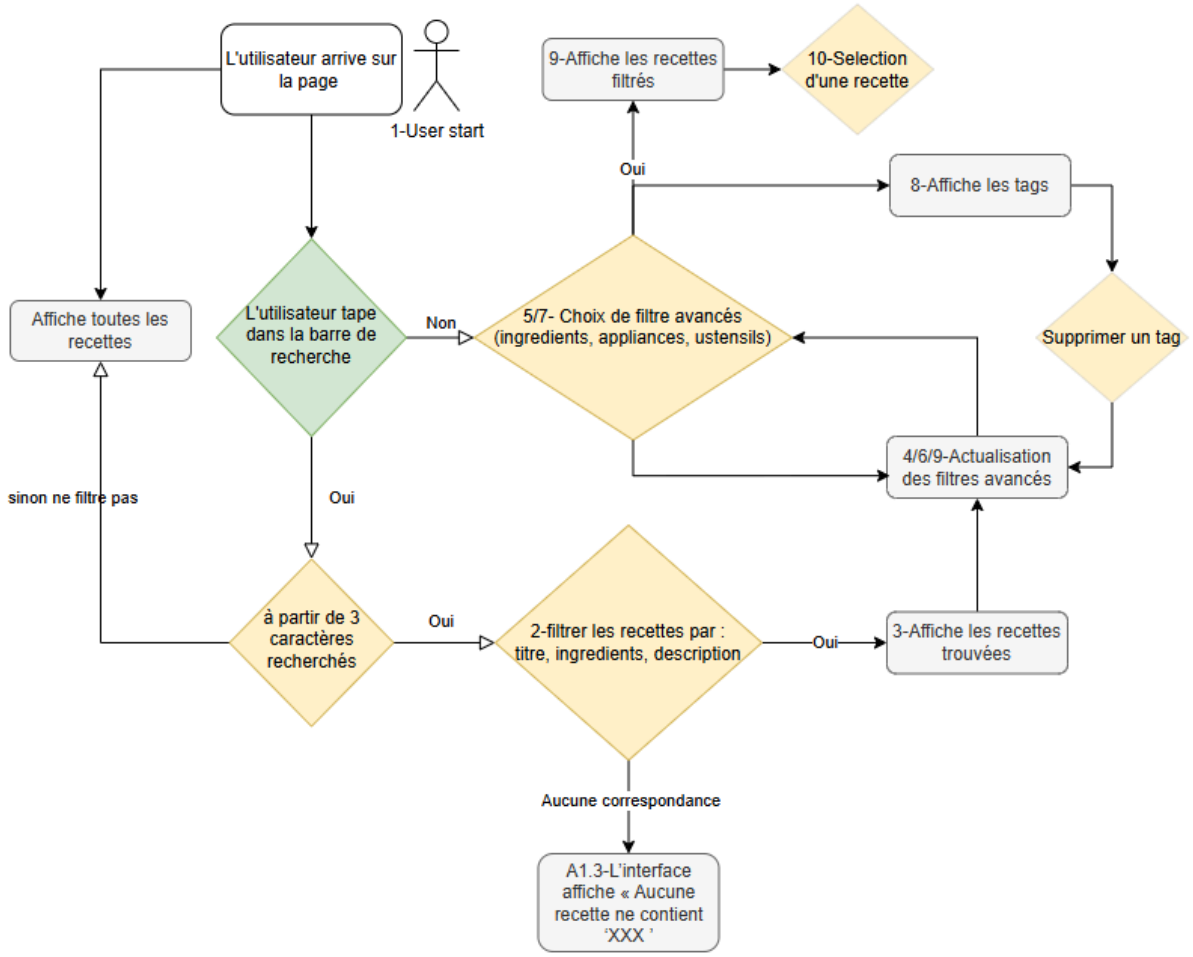
Comportement :

- Affiche toutes les recettes si la recherche est vide
- Met à jour les filtres selon les résultats
- Affiche un message du type *« Aucune recette ne contient XXX »* si aucune correspondance

Solution retenue :

Nous avons retenu l'approche Recherche dynamique avec filtres avancés.
Elle permet une expérience de recherche plus intuitive, avec des résultats pertinents et filtrables en temps réel.
Elle offre également une gestion claire des cas de non-correspondance tout en affichant un nombre réduit de champs à remplir.

Annexes


L'utilisateur arrive sur la page
1-User start
9-Affiche les recettes filtrés
10-Selection d'une recette
Oui
8-Affiche les tags
Affiche toutes les recettes
L'utilisateur tape dans la barre de recherche
Non
5/7- Choix de filtre avancés (ingredients, appliances, ustensils)
Supprimer un tag
4/6/9-Actualisation des filtres avancés
sinon ne filtre pas
Oui
à partir de 3 caractères recherchés
Oui
2-filtrer les recettes par : titre, ingredients, description
Oui
3-Affiche les recettes trouvées
Aucune correspondance
A1.3-L'interface affiche « Aucune recette ne contient 'XXX '

Annexes 2

Tableau de comparaison JSbench :

n° essai	méthode fonctionnelle	méthode native
1	7 385 928 (100%)	7 365 645 (99.73%)
2	7 792 044 (100%)	7 754 275 (99.52%)
3	7 843 638 (100%)	7 786 577 (99.27%)
4	7 822 786 (100%)	7 793 218 (99.62%)
5	7 816 535 (100%)	7 750 968 (99.16%)
6	7 875 262 (100%)	7 798 625 (99.03%)
7	7 853 359 (100%)	7 843 050 (99.87%)
8	7 866 132 (100%)	7 846 043 (99.74%)
9	7 842 304 (100%)	7 820 527 (99.72%)
10	7 722 456 (100%)	7 639 447 (98.93%)

D'après les résultats de JSbench, la méthode fonctionnelle est la meilleure des deux méthodes en terme de nombre d'opérations par secondes. En moyenne, la méthode native est plus lente de 0.54%.
De plus, la méthode fonctionnelle est bien plus simple à lire et comprendre, ce qui permettra une meilleure maintenabilité du code dans le temps.

Annexes 3

Sécurités

Nous allons ici voir ensemble les différentes actions mises en place afin de sécuriser le site face aux injections de html dans le formulaire.

1. Échapper les caractères spéciaux

Lorsqu'on affiche une entrée utilisateur dans le DOM, il est recommandé d'utiliser la propriété ".textContent" plutôt que ".innerHTML", car ce dernier interprétera les éléments qui lui sont donnés comme du HTML.

2. Utilisation des attributs HTML natifs pour la validation

Les attributs comme required, pattern, maxlength, … permettent d'éviter des soumissions avec des données malformées.

3. Eviter l'utilisation de fonctions dangereuses

Des fonctions comme :
- eval()
- innerHTML
- document.write()
- setTimeout(string)

4. Content security policy (CSP)

Même si c'est plutôt une configuration serveur, on peut s'assurer que la page utilise une CSP correcte pour empêcher l'exécution de scripts non autorisés