Fiche d'investigation de fonctionnalité

Fonctionnalité : Recherche par mots clé en 2 étapes	**Fonctionnalité #2**
Problématique : Obtenir les meilleures performances possibles pour une recherche par mot clé, implémentée sur un champ de recherche principal, puis sur les résultats de cette recherche, par cat®gorie (recherche avancée)	

Option 1 : Utilisation d'un algorithme de recherche lin®aire
Recherche principale : La recherche s'accomplit en passant par chacune des recettes, en comparant le mot cl® avec le contenu de chaque recette, dans l'ordre o½elles sont r®cup®r®es de l'api.
Recherche avanc®e : s'ex®cute sur les r®sultats de la 1¯re recherche, mais en cherchant uniquement dans la cat®gorie d'o½vient le mot.

Avantages	**Inconvénients**
Data structures basiques (implémentation plus facile)	Moins bonnes performances

Option 2 : Utilisation d'un algorithme de recherche non lin®aire ('Trie tree')
Recherche principale : A réception des données api, chacune des recettes est mappée dans un arbre de tri. La recherche s'effectue dans cet arbre.

Recherche avancée : pas de différence avec le précédent algorithme

Avantages	**Inconvénients**
Meilleures performances si la taille des données croit	Au 1er chargement de la page la construction du tree prend quelques millisecondes (il est ensuite socké dans le local storage)

Nombre de characteres minimum à entrer dans le champ principal : 3

Nombre de characteres minimum à entrer dans le champ secondaire (catégories) : 3

COMPARAISONS CHIFFREES DES PERFORMANCES (utilisant l'api 'performance.now()')

		LINEAR SEARCH	**TRIE SEARCH**
recherche du mot 'amande'	**FIREFOX** **CHROME**	**1 millisecond** **0.0999 millisecond**	**1 millisecond** **0.0999 millisecond**

CONCLUSION
Les performances de recherche d'un même mot des 2 algorithmes sont très similaires, bien que leur complexité soit différente. Pour avoir une vraie idée des différences il faudrait tester leurs performances sur des sets de données très grands. La recherche non linéaire serait alors la plus performante, comme le montre l'analyse du Big O.

Solution retenue :
Recherche non linéaire par 'tree trie' (ou 'prefix tree),pour sa scalabilité