Fiche d’investigation de fonctionnalité

Fonctionnalité : Rechercher / Filtrer les recettes	Fonctionnalité #2
Problématique : Afin de se démarquer des autres sites proposant le même service, nous devons proposer un algorithme de recherche rapide, fluide et performant à l’utilisateur afin qu’il puisse accéder aux recettes facilement et rapidement.	

Option 1 : Utilisation des boucles natives

Dans cette option, le but principal est d’utiliser les boucles « for », « while » proposées par le langage JavaScript. Ces boucles permettent d’effectuer la même action plusieurs fois de suite.

Avantages	**Inconvénients**
• Facile à mettre en place • Déterminer la façon dont on parcourt le tableau (i++, i--, etc.) • Parcourir un nombre précis d’éléments • possibilité de « break » • Compatibilité(Internet explorer) • Rapidité • Utilisation de générateurs possible	• Lecture du code (besoin d’itérateurs) • Maintenabilité

Nombre de champs minimum à remplir pour une recherche : 1
Champ de recherche global : Rechercher un ingrédient, appareil, ustensile ou une recette
Nombre de champs maximum à remplir pour une recherche : 4
Champ de recherche global : Rechercher un ingrédient, appareil, ustensile ou une recette
Champs de recherche spécifiques :

- Rechercher un ingrédient
- Rechercher un appareil
- Rechercher un ustensile

Option 2 : Utiliser la programmation fonctionnelle

Dans cette option on utilise la programmation fonctionnelle proposée par JavaScript. On utilisera donc des boucles « forEach » et les fonctions telles que « filter », « map » etc … pourront être utilisées

Avantages	**Inconvénients**
• Solution Moderne • Bonne lisibilité du code • Maintenabilité	• Solution possiblement plus lente • Compatibilité (IE9) • Obligation de parcourir tous les éléments d’un tableau • impossible d’afficher les résultats immédiatement

Nombre de champs minimum à remplir pour une recherche : 1
Champ de recherche global : Rechercher un ingrédient, appareil, ustensile ou une recette
Nombre de champs maximum à remplir pour une recherche : 4
Champ de recherche global : Rechercher un ingrédient, appareil, ustensile ou une recette
Champs de recherche spécifiques :

- Rechercher un ingrédient
- Rechercher un appareil
- Rechercher un ustensile

Solution retenue:

La **programmation native** semble être la solution la plus appropriée vis à vis des demandes techniques du client.

Dans un premier temps, le client souhaite que les recherches soient effectuées le plus rapidement possible. Or les résultats du benchmark ont montré que les boucles for sont plus rapides que les boucles foreach.

De plus, avec la programmation native, il est possible d’afficher les résultats dès que possible grâce aux générateurs (yield). Il est impossible d’utiliser ces générateurs avec la programmation fonctionnelle puisque la méthode filter() retourne un tableau avec les éléments répondant à la condition passée. Or il est obligatoire de boucler sur toutes les recettes afin d’obtenir ces mêmes résultats.

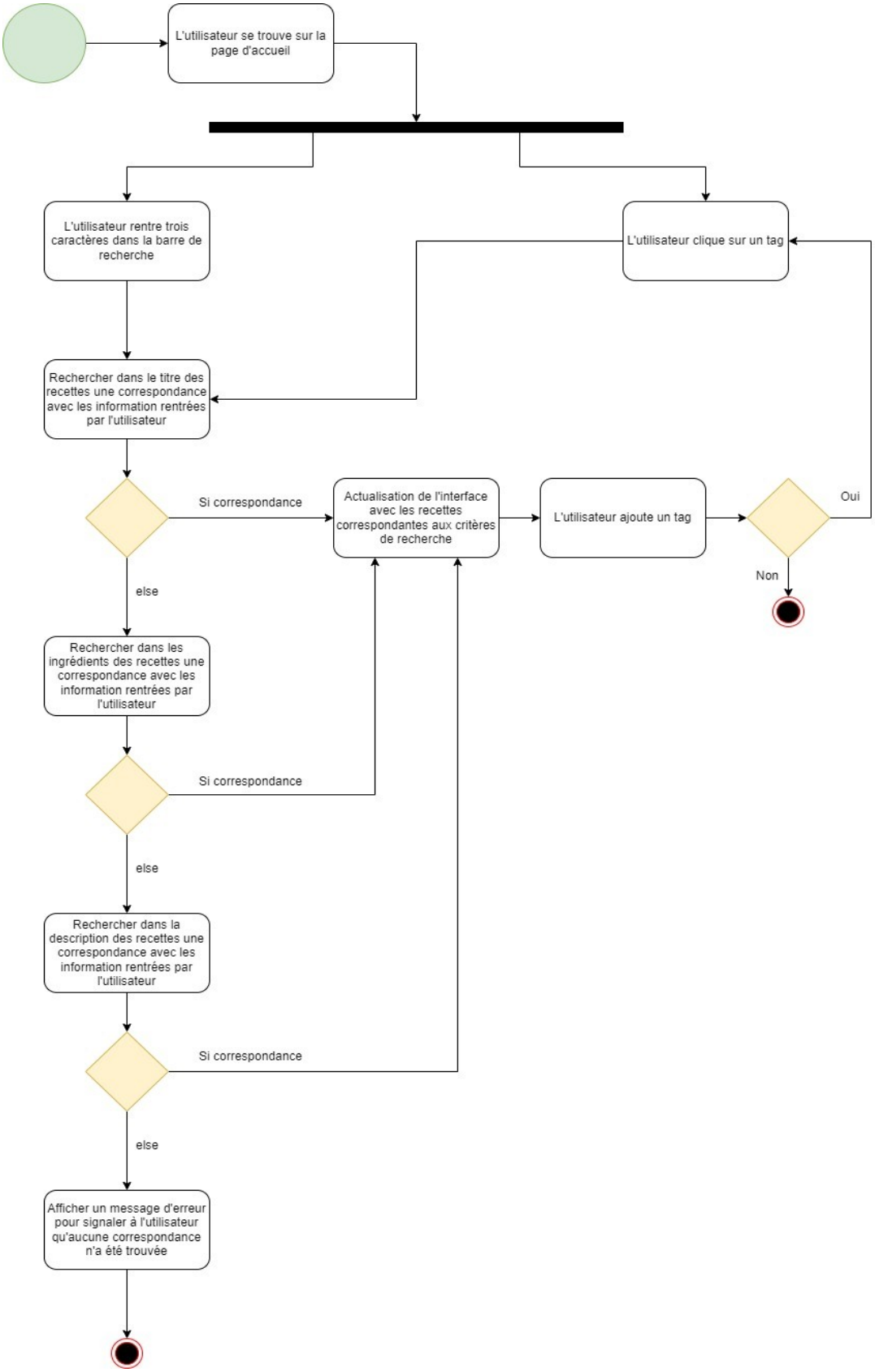

Figure 1: Option 1 - Algorigramme Programmation Native

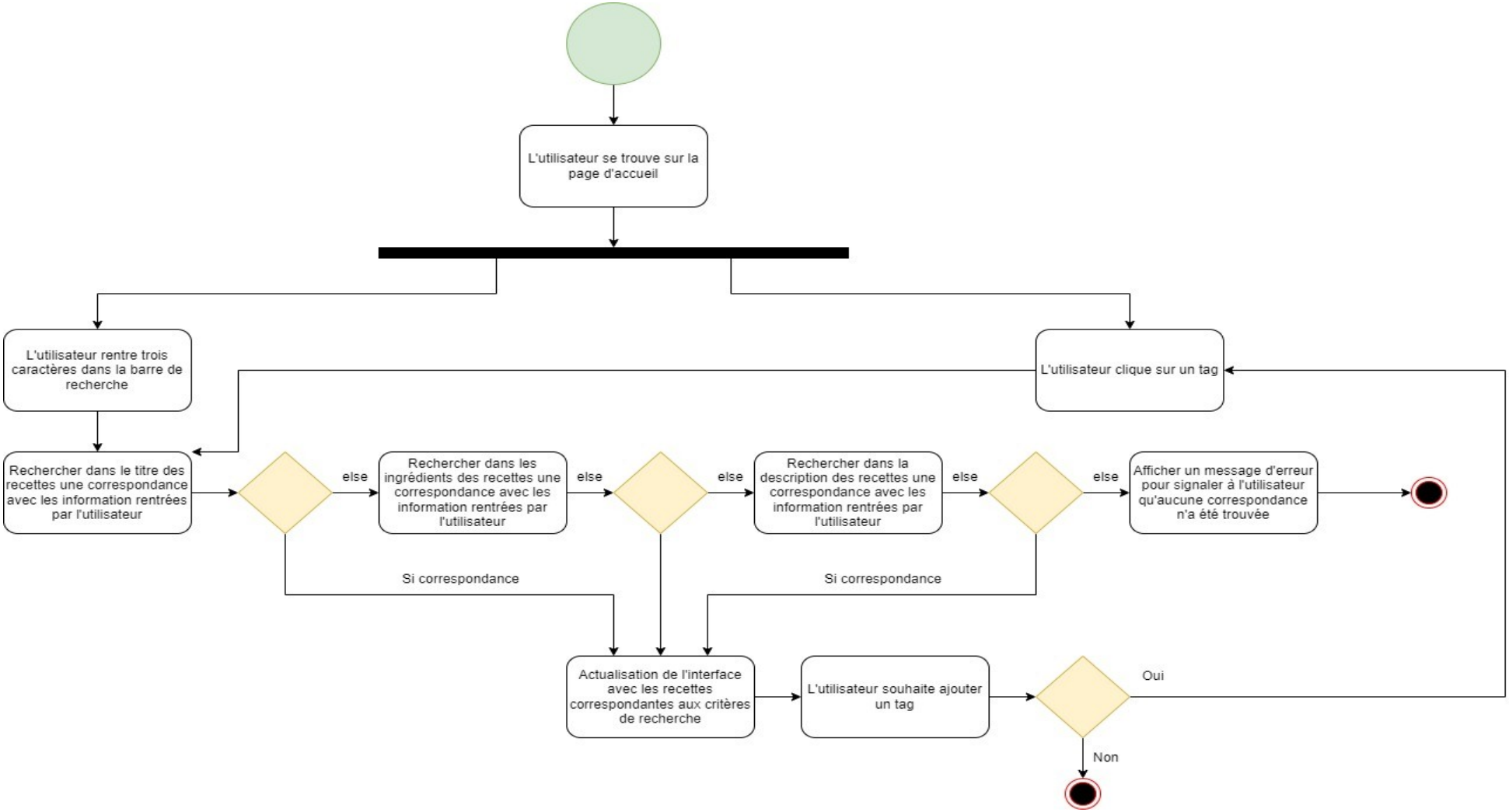

Figure 2: Option 2 : Algorigramme Programmation Fonctionnelle