Développez un algorithme de recherche en JavaScript

90 heures

Mis à jour le mercredi 18 janvier 2023

Le nommage des livrables à déposer sur la plateforme a été changé et des indications sur les temps de soutenance ont été ajoutées le 16/03/2022.

Scénario

Vous êtes freelance et vous venez d'être missionné par l'entreprise "Les petits plats" en temps que Développeur Front-end pour une mission de 3 mois.

Logo de l'entreprise Les Petits Plats

Après avoir édité des livres de cuisine pendant plusieurs années, l'entreprise a décidé de se lancer dans un nouveau projet : réaliser son propre site de recettes de cuisine à l'instar de Marmiton ou 750g.

Sandra, cheffe de projet, est en charge de la digitalisation de l'entreprise avec la création de ce site web. Pour l'instant, elle travaille uniquement avec des freelances comme vous, avant de créer une équipe en interne pour gérer ce projet. Afin de s'assurer que vous avez tout ce qu'il vous faut à disposition, elle vous envoie un mail :

> **De** : Sandra
> **À** : Moi
> **Objet** : Détails algo de recherche

> Bonjour,
>
> Je suis ravie de t'avoir dans l'équipe pour cette nouvelle étape du projet.
>
> Comme tu le sais, les sites offrant des recettes de cuisine sont nombreux et l'équipe a pensé que l'un des éléments qui peuvent faire la différence sur notre site est la fluidité du moteur de recherche. L'équipe Back-end n'étant pas encore formée, nous disposons uniquement d'un fichier JavaScript contenant un tableau JSON de 50 recettes.
>
> Ta première mission sera donc d'implémenter la fonctionnalité de recherche. Tu trouveras ici la description du cas d'utilisation de recherche. C'est ce document qui te servira de référence pour tout le développement de cette fonctionnalité. En plus de ça, voici la maquette de la page sur Figma, assure-toi de bien respecter le design à la lettre.
>
> Ce que l'on veut avant tout c'est quelque chose de performant car nos utilisateurs veulent une recherche rapide, presque instantanée ! Ton travail sera transmis au Back-end dans un second temps pour être adapté par leurs soins. C'est pourquoi il faudra que tu leur transmettes un document expliquant bien ton travail. Je te laisse voir comment procéder en détail directement avec Jean-Baptiste.
>
> Sandra

En fin de matinée, vous recevez une notification Slack de Jean-Baptiste, votre Lead Developer :

JB	Salut ! Comme tu as pu le voir, la recherche est une fonctionnalité très importante pour l'équipe et on compte sur toi pour la développer d'une manière optimale. Dans notre équipe, pour tout algorithme important qu'on développe, on a pour habitude d'en faire deux implémentations différentes pour pouvoir comparer leurs performances et choisir la meilleure. Il faudra donc que tu fasses de même ! Pour ça il faudra que tu crées un document de comparaison qu'on appelle "fiche d'investigation de fonctionnalité". Nous avons récemment fait ça pour la fonctionnalité "connexion / inscription" dont voici le résultat. Donc réutilise directement le même modèle de document.
Vous	Très bien merci. Du coup, tu me conseilles de procéder comment ?
JB	Alors ce que je te conseille c'est de commencer par implémenter ton interface comme ça tu es débarrassé. Comme pour le reste du site, tu peux utiliser Bootstrap si tu veux, mais pas d'autre librairie. Veille à bien écrire du code qui passe avec succès le validateur W3C. Puis, côté algorithmes tu peux procéder en 3 étapes. D'abord, planifie les 2 versions de la fonctionnalité que tu veux tester. Puisque tu vas traiter beaucoup de tableaux, ce serait intéressant de faire une version utilisant les boucles natives (while, for...) et une version en programmation

	fonctionnelle avec les méthodes de l'objet array (foreach, filter, map, reduce). Pour ça, commence à remplir le document d'investigation de fonctionnalité autant que tu peux pour bien décrire les deux implémentations que tu veux comparer. Ces deux implémentations doivent se focaliser uniquement sur le champ de recherche principal. N'oublie pas de faire un schéma, ou "algorigramme", pour chacune des propositions (les deux implémentations peuvent avoir le même algorigramme) afin qu'on comprenne bien l'enchaînement des étapes de chacun des algorithmes, cela sera surtout utile à l'équipe Back-end. Tu peux te baser sur les schémas présents dans la fiche d'investigation de la fonctionnalité de Connexion/Inscription mais utilise le formatage que tu veux. Moi j'utilise draw.io pour faire mes schémas, c'est très pratique et gratuit.
Vous	Ok et ensuite je les implémente ?
JB	Exactement, deuxième étape : tu les implémentes tous les deux. Pour ça, utilise 2 branches différentes sur Git afin qu'on conserve bien le code séparé pour chacun. Pour ton implémentation, toutes les infos techniques sont sur le document du cas d'utilisation que t'a envoyé Sandra. Pour les recherches par tag, tu pourras utiliser une seule et même version de la recherche pour les 2 branches.
Vous	Et comment je choisis la meilleure version du code du coup ?
JB	Ça c'est ta troisième et dernière étape. Pour choisir le meilleur algorithme, il faut que tu testes leur performance. Pour ça, tu peux utiliser l'outil de comparaison de performance que tu veux, personnellement j'utilise Jsben.ch pour ce genre d'analyse. Il te donnera le nombre d'opérations par seconde réalisées par chaque script et te permet donc de voir en un clin d'œil quel script est le plus performant. Tu peux tester uniquement la recherche principale (pas besoin d'utiliser les filtres). Ajoute ensuite les résultats à la fiche d'investigation de fonctionnalité que tu auras rédigée. N'oublie pas de terminer le document par la recommandation d'algorithme à garder suite à ton analyse et tests.
Vous	Parfait, merci pour tes conseils JB. Je me lance !

Ça y est, vous avez toutes les informations nécessaires pour démarrer votre travail. C'est parti !

Livrables

1. Une **fiche d'investigation** de fonctionnalité sur l'algorithme de recherche (format PDF). Vous y intégrerez le choix de l'algorithme définitif en comparaison à l'autre

algorithme de recherche développé.
2. Un fichier au format TXT contenant le lien vers votre **code** sur GitHub comprenant les deux branches avec les deux solutions différentes pour la recherche.

Pour faciliter votre passage devant le jury, déposez sur la plateforme, dans un dossier zip nommé "***Titre_du_projet_nom_prénom***", avec tous les livrables du projet comme suit : **Nom_Prénom_n° du livrable_nom du livrable__date de démarrage du projet**. Cela donnera :

- *Nom_Prénom_1_fiche_investigation_mmaaaa ;*
- *Nom_Prénom_2_code_mmaaaa.*

Par exemple, le premier livrable peut être nommé comme suit *: Dupont_Jean_1_fiche_investigation_012022.*

Soutenance

Durant la présentation orale, votre interlocuteur jouera le rôle de Jean-Baptiste, le lead dev du projet :

- **Présentation des livrables (15 minutes)**
  - Durant 5 minutes vous présenterez comment vous avez décomposé la problématique à l'aide du document d'investigation.
  - Ensuite vous aurez entre 10 et 15 minutes pour montrer le rendu de l'application avec la version du moteur de recherche la plus performante, en justifiant avec des chiffres pourquoi cette version est meilleure que l'autre version.
- **Discussion (10 minutes)**
  - Votre interlocuteur vous questionnera sur vos choix, tant sur la recherche que sur la façon dont vous avez implémenté l'interface (avec ou sans Bootstrap). L'évaluateur questionnera vos choix, soyez donc prêt à défendre votre travail.
- **Debrief (5 minutes)**
  - À la fin de la session, l'évaluateur quittera le rôle de Jean-Baptiste pour que vous puissiez débriefer ensemble.

Votre présentation devrait durer 15 minutes (+/- 5 minutes). Puisque le respect de la durée des présentations est important en milieu professionnel, les présentations en dessous de 10 minutes ou au-dessus de 20 minutes peuvent être refusées.

Compétences évaluées

- Analyser un problème informatique
- Développer un algorithme pour résoudre un problème