Projet 2 : Indexation des CVs dans Elasticsearch

Introduction :

L'indexation des données est une pratique courante dans le développement des systèmes d'information complexes afin de pouvoir fournir des applications fluides et performantes. C'est une technique adoptée par tout SGBD digne de ce nom pour optimiser le temps de réponse des requêtes de fetch et de recherche d'une complexité linéaire à un une complexité logarithmique. Cependant, manipuler des indexes exigeait toujours une expertise pointue sur les base de données ainsi qu'une connaissance profonde quant à leur fonctionnement, ce qui n'était pas à la portée de tout le monde. Or pour la culture entreprise qui est pleinement orientée par la génération de la valeur consommable, compétence veut souvent dire plus de dépense, de même que problème de performance a pour conséquence une qualité de delivery moindre. Heureusement pour l'industrie du software, Elasticsearch a fait son apparition pour répondre à ces besoins en apportant une solution aux problèmes de performance en général et ceux de search en particulier. A présent, la solution Elasticsearch est devenue puissante à tel point qu'elle est parvenue à banaliser les moteurs de recherche.

Objectif du projet :

Le but de ce projet est de sortir l'étudiant de sa zone de confort en le challengeant avec une problématique réelle à laquelle il peut faire face dans une entreprise, tout en gardant les mêmes contraintes qui peuvent exister dans un environnement professionnel. Nous pouvons citer à titre d'exemple le facteur de temps limité, la pression pour fournir au moins une solution fonctionnelle mais surtout une grande autonomie (l'entreprise favorise les personnes autonomes, après tout l'étudiant d'aujourd'hui sera le cadre de demain !).

Contexte fonctionnel :

Ces dernières années, une entreprise lambda a été préoccupée par une problématique majeure : le recrutement des candidats et le suivi de ses collaborateurs. Face à ces besoins insistants, l'équipe IT de l'entreprise a développé une solution de recrutement et d'onboarding baptisée LHOM (Lambda Hiring & Onboarding Master).

Aujourd'hui, LHOM est un logiciel qui répond aux besoins principaux de l'entreprise et à ceux exprimés par ses clients. Suite au succès de la version initiale, les équipes de l'entreprise ont pris pour objectif le développement de celle-ci en y apportant des améliorations. Étant donné que la base de l'appli contient un nombre important de candidats, la recherche d'un ensemble de candidats partageant les mêmes éléments (ou la recherche d'un candidat en particulier) est devenue fastidieuse. Par conséquent, une des fonctionnalités attendues dans le radar de livraison est une recherche avancée de candidats.

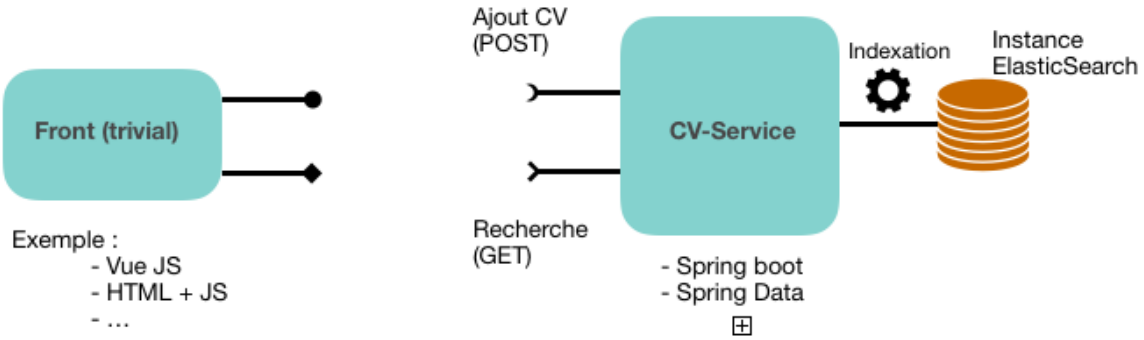
Fonctionnellement parlant, la feature consiste à donner aux utilisateurs la possibilité d'effectuer des recherches complexes et variées sur la liste regroupant les candidats enregistrés dans le système, en se basant sur des filtres ou des tags (Ex: quels sont les candidats qui font du JAVA et du Node en même temps ?).

L'énoncé du projet :

L'énoncé du projet est concis mais peut s'avérer challengeant en même temps. Il s'agit de réaliser un POC (Proof Of Concept) sur cette fonctionnalité de recherche avancée qui sera potentiellement ajoutée au projet LHOM (Lambda Hiring and Onboarding Master) le POC aura pour seul objectif d'étudier la faisabilité et la complexité de la tâche).
Techniquement parlant, La feature consiste globalement à indexer des CVs issus de plusieurs formats dans Elasticsearch afin d'être en mesure de les requêter par la suite en bénéficiant de la puissance d'Elasticsearch. On peut imaginer le projet sous forme d'un micro-service (CV-Service) qui détient une instance Elasticsearch et expose une API REST de deux endpoints :

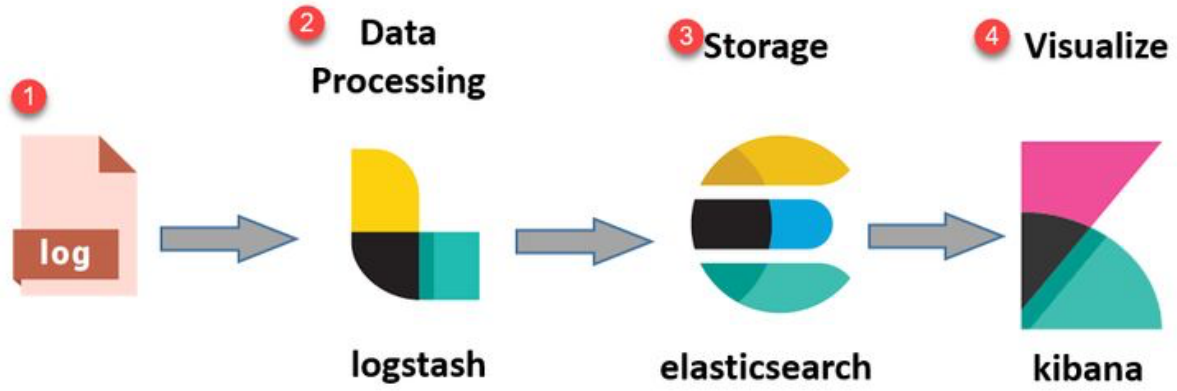
- L'un indexe les données du CV dans l'instance Elasticsearch lors de son ajout;
- Le deuxième permet la recherche en se basant sur des termes.

Lors de l'ajout d'un CV via un client http, celui-ci doit être parsé et son contenu doit être indexé dans l'instance Elasticsearch. Un utilisateur a la possibilité d'uploader des CVs sous différents formats : PDF ou Word. Une fois la CVthèque indexée , nous nous appuierons sur l'endpoint de recherche pour réaliser des recherches variées en un temps raisonnable.
Le backend est à écrire avec le framework Spring, utilisant deux modules primordiaux de Spring : Spring boot et Spring Data Elasticsearch (ou bien le client Java high level de Elastic). Le front n'est pas nécessaire (à part si vous voulez vous challenger vous-même). L'interaction avec notre backend s'effectuera donc avec un client http (Postman à titre d'exemple) . Le schéma ci-dessous fournit une vue d'oiseau de la solution.



Une application qui ne génère pas de log aura beaucoup de problèmes à surmonter les différents bugs qui surgissent lors de son utilisation. Vous devez par conséquent ajouter des logs au fur et à mesure là où vous en ressentez la pertinence.
Dans le but de bien observer nos logs, nous ferons appel à la fameuse stack de Elastic nommée ELK : initialement, votre application utilisera la console

pour logger (printer les logs), l'idée est de configurer votre application pour qu'elle envoie ses logs à Logstash (qui les indexe à son tour dans Elasticsearch) afin qu'on soit en mesure de les consulter depuis Kibana (non depuis la console).
Cette configuration n'est valide que pour l'environnement de Prod. Cela veut dire que vous devez gérer deux configurations (en vous basant sur les profiles de Spring), une dite Dev qui logue sur la console, et une autre Prod qui logue sur Kibana.



Répartition des groupes

Les groupes doivent être constitués de 2 à 4 personnes maximum. Le deadline pour rendre le projet est le 07/11/2021 à 23h59.

Modalités de rendu :

- Un lien Github du projet est à rendre par chaque équipe.
- Le nom et prénom de chaque membre du groupe doit figurer dans le readme du projet.
- Les projets seront communiqués au responsable du cours (Bin) deux jours après la deadline (J+2), qui prendra la responsabilité de relancer (cinq jours après) ceux qui n'auront pas encore envoyé leur projet (à savoir est ce qu'ils veulent abandonner ou continuer).

Modalités d'évaluation :

Sauf cas extrême, les membres d'une même équipe obtiendront une appréciation identique. L'évaluation prendra en considération les aspects suivants :

- La qualité du code;
- La pertinence et la qualité du livrable (est ce que ça fonctionne?);
- Le respect des deadlines.

Indice : pour obtenir une note moyenne, votre système doit au moins répondre à la question suivante : Quels sont les profils qui font du Java?

Je serai disponible pour les questions sur Slack. Bon courage pour tout le monde (^_^) .