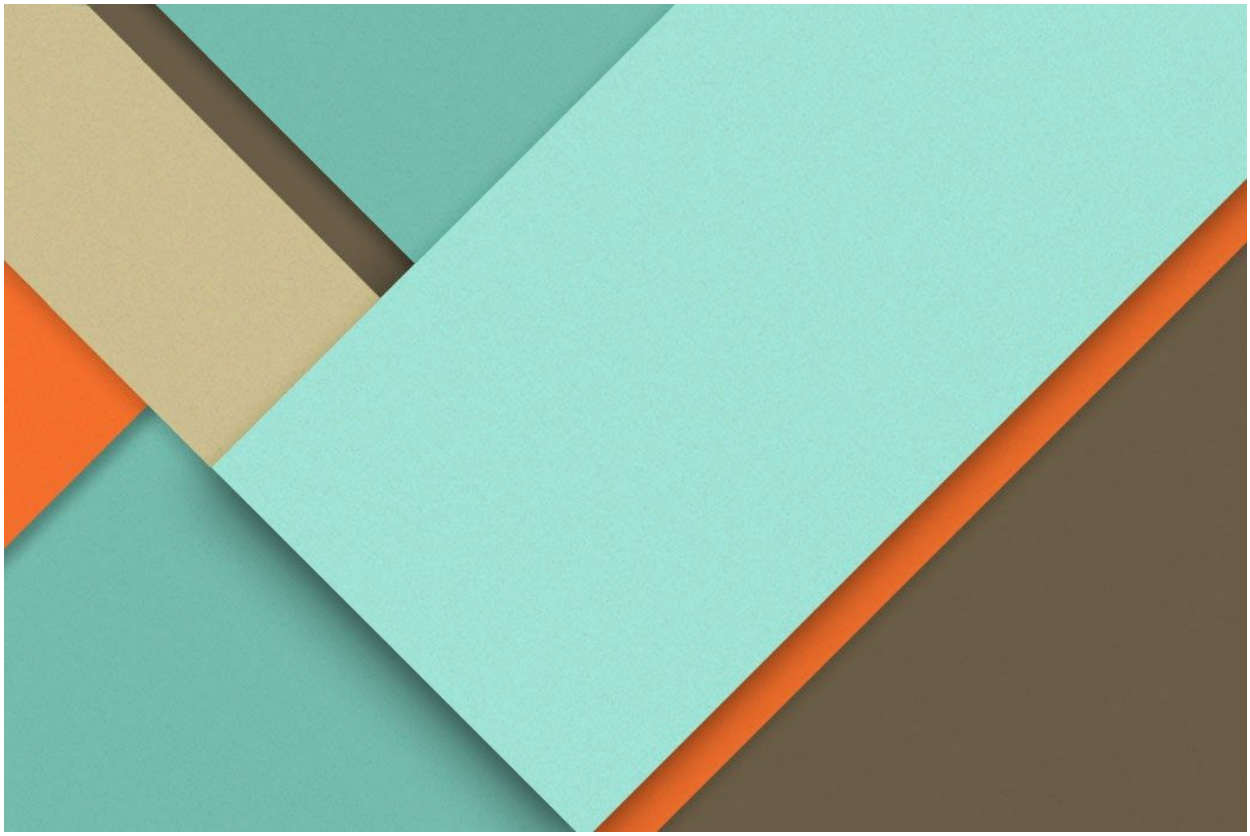


In Fact

04/03/2020

BETTAYEB, BUI, SIMON, VALENCA

Vue d'ensemble

Le but de ce projet était de concevoir et mettre en place un jobboard qui permettrait de centraliser les plateformes de recrutement des entreprises. En effet, lors d'une phase de recherche d'emploi, il est fastidieux de se créer un compte sur chacun des sites de candidatures des entreprises, c'est là qu'Infact intervient et permets aux chercheurs d'emploi de s'inscrire sur un seul site et suivre leur candidatures et aux entreprises d'avoir accès à un plus grand nombre de potentiels candidats car nombreux sont ceux qui sont découragés à la vue du nombre de sites différents sur lesquels il est nécessaire de s'inscrire.

La page d'accueil du site est le formulaire de recherche qui permets d'envoyer des requêtes et d'ajouter des filtres tels que : la distance, la date de parution, le salaire etc...
Le bandeau permets de gérer la partie connexion/inscriptions, il est possible de cliquer sur les deux supérieur boutons qui feront apparaître les différents formulaires.
Une fois inscrit/connecté en tant que particulier il est possible de modifier son profil, d'ajouter son cv. En tant qu'entreprise, il est possible d'ajouter un logo en plus des informations.

En tant que particulier il est possible en cliquant sur le bouton "mes contributions" d'accéder au candidatures effectuées. Dans une version future il sera aussi possible de suivre l'avancement du processus de recrutement.
En tant qu'entreprise, une liste des offres postées sera disponible et une liste des candidatures par offre est aussi accessible ainsi qu'une gestion de l'avancement du processus de recrutement.

En cliquant sur une offre en tant que particulier il est possible de postuler ce qui transmettra le profil et le cv à l'entreprise.

Objectifs

Les objectifs d'inFact sont les suivants

1. Pour un particulier :
    a. Trouver des offres d'emploi à partir de certains critères : la distance, la date de parution etc...
    b. Pouvoir postuler à des offres directement sur le site
    c. Pouvoir suivre l'avancement des candidatures
    d. Gérer mon compte
    e. Déposer mon CV
2. Pour une entreprise
    a. Poster des offres d'emploi
    b. Suivre les candidatures
    c. faire avancer les offres dans le processus de recrutement

Déroulement et organisation

La répartition du travail était la suivante :

- Back :
    - Jonathan Valenca
    - Youssef Bettayeb
- Front :
    - Emeric Bui
    - Corentin Simon

Dès que la repartition des taches était décidée 2 trellos ont été mis en place.
Nous avons ensuite décidé de l'architecture du projet :

Notre back & front seraient gérés par le même serveur sous nodeJS, notre base de données en PostgreSQL. Pour simuler une connexion au back le temps du développement nous avons décidé d'utiliser Mountebank.

PARTIE FRONT - MOUNTEBANK & WEBPACK

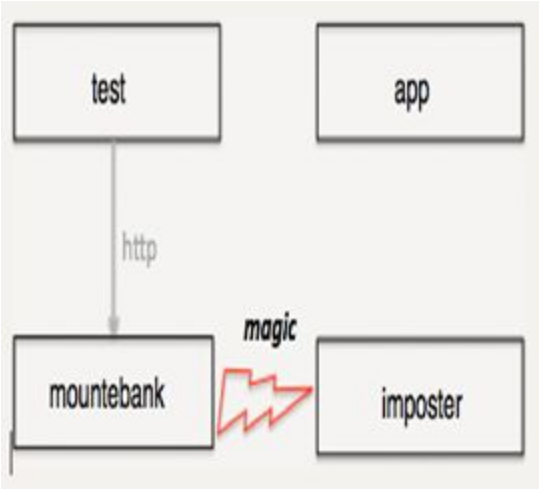
Afin de travailler de côté pour la partie FRONT, nous avons utilisé Mountebank et Webpack Config pour pouvoir assurer le travail indépendamment du BACK. Mountebank nous permet de simuler une connexion au BACK, en envoyant des requêtes REST à une url http et simuler la réponse d'une api. Nous allons expliquer comment mountebank fonctionne ci-dessous.

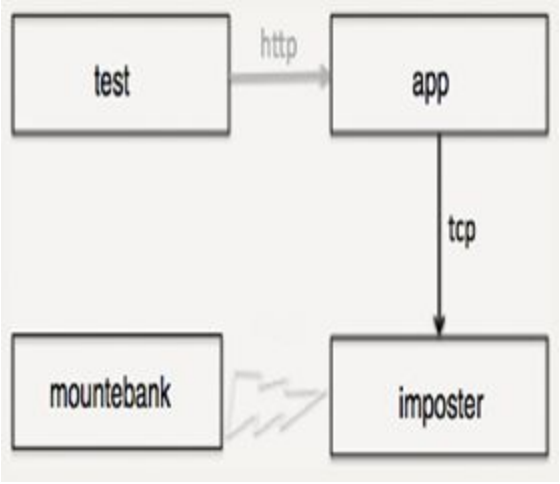
Comment fonctionne mountebank ?

Mountebank est un outil open source à fournir des doubles tests multiprotocoles et multiplateformes. Il suffit de pointer votre application en cours de test vers mountebank au lieu de la dépendance réelle, et de tester comme vous le feriez avec des souches et des maquettes traditionnelles.

Mountebank utilise des imposteurs pour faire office de doubleurs de tests à la demande. Votre test est communiqué à mountebank via http en utilisant l'api via le stub qui vérifier les attentes simulées. Dans le cas d'utilisation typique, chaque test démarrera un imposteur pendant la configuration du test et arrêtera un imposteur pendant le démontage du test.

Mountebank emploie plusieurs types d'imposteurs, chacun répondant à un protocole spécifique (GET/POST/PUT/DELETE – Requête REST) et à des paramètres définis dans l'imposteur (chemin, body, etc...). En général, votre test indiquera à l'imposteur le port sur lequel il doit se lier, et l'imposteur ouvrira le socket correspondant.





Pourquoi utilisons-nous Webpack ?

Dans la programmation modulaire, les développeurs divisent les programmes en morceaux discrets de fonctionnalités appelés modules.Chaque module a une surface plus petite qu'un programme complet, ce qui rend la vérification, le débogage et les tests triviaux. Des modules bien écrits fournissent des abstractions solides et des limites d'encapsulation, de sorte que chaque module a une conception cohérente et un objectif clair au sein de l'application globale.

Node.js a soutenu la programmation modulaire presque depuis sa création. Sur le web, la prise en charge des modules a été lente à arriver. Il existe de nombreux outils qui prennent en charge le JavaScript modulaire sur le web, avec toute une série d'avantages et de limites. Webpack s'appuie sur les enseignements tirés de ces systèmes et applique le concept de modules à n'importe quel fichier de votre projet.

Webpack apporte à ce niveau des fonctionnalités intéressantes :

- Disposer de toutes les ressources statiques (CSS, images, fontes) en tant que module,
- Intégrer et consommer des bibliothèques tierces très simplement en tant que module,
- Séparer votre build en plusieurs morceaux, chargés à la demande,
- Garder un chargement initial très rapide si besoin avec la commande watch ou build,

Webpack s'occupe uniquement de ressources JavaScript. Webpack propose un système de load qui permet de transformer tout et n'importe quoi en JavaScript. Ainsi, tout est consommable en tant que module. Webpack prend en charge la fonction require() qui est très utile pour notre architecture de projet et accéder à nos entités , API et autres modules.

PARTIE BACK - NodeJS & PostGreSQL

Notre Back est géré par NodeJS, nous utilisons postgreSQL pour la gestion de la base de données. Celle ci stockera les offres d'emplois, les inscriptions, les candidatures, les profils ainsi que les différents documents. Pour ce qui est des utilitaires de "géographie" nous avons décidé d'utiliser Open Street Map et Nominatim qui sont des API qui permettent de récupérer des adresses à partir de longitudes et latitudes et inversement, ce qui nous permets ensuite de calculer la distance entre 2 points et filtrer les résultats de recherche selon ces critères. Les designs patterns qui ont été retenus sont : L'entity et le repository pour les différentes classes/tables. Le factory sera utilisé pour la session client.

Les requêtes HTTP et les routes sont gérées dans le fichier controller.js, le dossier controller gère les requêtes webservices qui font appel au repository.

Du côté de la base de données toutes nos requêtes sont transactionnelles de part la nature de postgreSQL et l'utilisation du contexte async ce qui assure une atomicité lors de l'ajout d'une offre, d'une candidature ou autre...

Toutes nos fonctions ont été testées unitairement via jest, à l'exception de certaines fonctions ou celà n'était pas pertinent et ou nous avons mit en place des tests API via postman.

Problématiques Organisationnelles

La principale problématique organisationnelle à laquelle nous avons fait face est la répartition des tâches entre les différents membres du groupe. Nous avons fait le choix d'assigner deux personnes à la partie back du produit et deux personnes à la partie front. De ce fait, nous pouvions nous appuyer sur notre binôme en cas de blocages.

Nous avons également eu des difficultés à concilier nos agendas respectifs pour faire des points réguliers sur l'avancée du projet et les possibles évolutions à développer.

Problématiques Techniques

Nous avons fait face à diverses problématiques techniques au cours du projet. La première était la configuration, en effet nous avons décidé de fonctionner avec NodeJS, il était donc nécessaire de modifier le docker-compose.yml fourni afin d'inclure nodeJS et mountebank. Ces deux étapes nous ont coûté beaucoup de temps car nous avions pas assez d'expérience sur les technologies.
Une autre problématique que nous avons rencontrée était l'obligation d'une synchronisation, en effet le plugin postgreSQL pour Nodejs n'utilisait que des fonctions appels asynchrones il a donc imposé la synchronisation à travers notre back end, le contexte async était donc nécessaire nous n'avions pas énormément d'expérience sur les requêtes asynchrones cela a donc quelque peu ralenti le temps de développement car il était nécessaire de se renseigner sur le sujet. Un autre exemple de fonctions problématique serait la fonction qui s'occupait de retourner une adresse à partir d'une latitude et longitude, il a été nécessaire d'apprendre à utiliser et résoudre les promesses pour pouvoir utiliser l'API d'open Street Map correctement.

Lors des premiers brainstormings nous avions décidé de mettre en place plusieurs patterns de développement pour le back-end (Repository,Entity,Factory et Hydrator), au fur et à mesure du développement nous nous sommes rendus compte que ces derniers n'étaient pas tous pertinents et réalisables en nodeJS.

Enfin la dernière difficulté qui nous a le plus coûté était la connexion entre le back et le front, nous avons retardée la connexion et nous nous y sommes pris quelque peu en retard, nous avons pas eu assez de temps pour mettre en place la connexion, ainsi cette dernière n'est pas effective, le front fonctionne très bien avec mountebank, le back fonctionne très bien de son côté mais la connexion n'est pas faite. Ceci est un point à travailler pour le prochain projet et il ne faudra pas se contenter de travailler avec une maquette comme mountebank et tester la connexion beaucoup plus tôt.

Améliorations futures

Les amélioration possibles de notre projet, au delà de la connexion back/front, auraient pû être l'ajout d'un processus de recrutement. En marquant une candidature avec différentes étapes, il aurait été possible et pertinent de mettre en place un suivi de l'avancement du processus de recrutement. Ce qui aurait permis à l'employeur de garder les candidats à jour sur le recrutement, on aurait pu imaginer plusieurs étapes exemple :

- Consultation du dossier
- Entretien téléphonique RH
- Présentation au référent technique
- Entretien
- Proposition d'embauche
- Rejet de candidature

Une autre piste d'amélioration aurait été l'implémentation d'un système de "notation" des entreprises, ou les candidats pourraient laisser des avis sur les entreprises ou des informations un peu plus précises sur l'entreprise, un exemple serait la présence ou non d'une cantine, d'un CE etc... Ces critères sont parfois importants pour un candidat et peuvent motiver ou non une candidature.

.