Guide d'étapes clés : Développez une solution en microservices pour votre client

Comment utiliser ce document ?

Ce guide vous propose un découpage du projet en étapes. Vous pouvez suivre ces étapes selon vos besoins. Dans chacune, vous trouverez :

- des recommandations pour compléter la mission ;
- les points de vigilance à garder en tête ;
- une estimation de votre avancement sur l'ensemble du projet (attention, celui-ci peut varier d'un apprenant à l'autre).

Suivre ce guide vous permettra ainsi :

- d'organiser votre temps ;
- de gagner en autonomie ;
- d'utiliser les cours et ressources de façon efficace ;
- de mobiliser une méthodologie professionnelle que vous pourrez réutiliser.

Gardez en tête que votre progression sur les étapes n'est qu'une estimation, et qu'elle sera différente selon votre vitesse de progression.

Recommandations générales

Ce projet reprend essentiellement des notions techniques déjà vues. Les nouveautés sont l'utilisation d'une base de données NoSQL, de Spring Cloud Gateway et Docker. En suivant le découpage des sprints, les différents sous-projets sont peu complexes.

Chaque sprint amène à créer un microservice back. Le microservice du front sera commun aux trois sprints et évoluera au fur et à mesure de l'ajout des besoins. L'interface doit être la plus sobre possible. Le microservice gateway est également commun aux trois sprints. Bien qu'elles répondent au besoin, les bases de données seront simples. À noter que l'authentification se fera par

Spring Security, mais qu'il n'y a pas besoin de mettre en place d'inscription ou de gestion de droits.

Étape 1 : Prenez en main le projet et le Sprint 1

5 % de progression

Avant de démarrer cette étape, je dois avoir :

- lu l'énoncé du projet.

Une fois cette étape terminée, je devrais avoir :

- compris le besoin du client ;
- créé le premier projet Spring Boot et sa structure en couches pour le microservice back patient.

Recommandations :

- Utilisez *start.spring.io* et choisir les bonnes dépendances ;
- Mettez en place le repository GitHub.

Point de vigilance :

- Attention à ne pas ajouter des éléments non demandés.

Étape 2 : Créez le microservice patient, la gateway et la première partie de l'interface utilisateur du Sprint 1

30% de progression

Avant de démarrer cette étape, je dois avoir :

- compris les besoins clients ;
- créé la structure du projet.

Une fois cette étape terminée, je devrais avoir :

- terminé le microservice back de gestion du patient ;
- implémenté un microservice avec Spring Cloud Gateway ;
- mis en place un microservice à part entière pour le front.

Recommandations :

- Le besoin est simple, la seule nouveauté est l'utilisation de Spring Cloud Gateway. Son utilisation va vous permettre de simplifier la gestion des accès aux microservices ;

- La mise en place de la structure de la base de données peut se faire directement via l'ORM ;
- Pensez à ajouter les données patients tests fournis. Cela vous permettra de tester avec Postman que les endpoints retournent bien ce qui est attendu ;
- Niveau interface utilisateur, il suffit juste qu'elle soit fonctionnelle pour la démo. Restez sobre. Par exemple, vous pouvez avoir une page listant tous les patients, et une page affichant le détail d'un patient ;
- Spring Security doit servir à contrôler les accès via authentification. Ne pas oublier d'encrypter les mots de passe. N'hésitez pas à utiliser une méthode d'authentification HTTP basique et des utilisateurs en mémoire.

Points de vigilance :

- N'ajoutez pas de fonctionnalités non demandées ;
- Le front ne doit pas être incorporé dans le microservice back, il doit être indépendant. Il sera mis à jour par les sprints suivants.

Ressources :

- Le cours Utilisez Spring Data pour interagir avec vos bases de données ;
- La documentation Spring Cloud Gateway.

Étape 3 : Créez une interface pour les notes du médecin avec une base de données NoSQL du Sprint 2

60 % de progression

Avant de démarrer cette étape, je dois avoir :

- relu les informations du Sprint 2 dans le scénario.

Une fois cette étape terminée, je devrais avoir :

- terminé le microservice back de gestion des notes ;
- mis en place la base de données MongoDB ;
- mis à jour le microservice gérant le front ;
- mis à jour la configuration Spring Cloud Gateway du microservice de gateway.

Recommandations :

- La base de données NoSQL est une nouveauté. Il est important de comprendre la différence entre une base de données relationnelle et une base de données NoSQL ;
- L'utilisation de Spring Data MongoDB rendra la communication avec la base de données simple ;
- Il faut identifier comment les patients pourront être retrouvés à partir des notes les concernant ;

- Ce sprint sera très similaire au dernier, mais l'accent sera mis sur la gestion des notes non structurées fournies par les prestataires médicaux ;
- Comme pour le microservice back patient, vous utiliserez Spring Security pour sécuriser les accès.

Points de vigilance :

- Il faudra utiliser MongoDB. Ne pas oublier la dépendance correspondante dans le pom.xml ;
- Pensez à ajouter les données notes tests fournies. Cela vous permettra de tester avec Postman que les endpoints retournent bien ce qui est attendu ;
- Il faudra intégrer à l'interface la gestion des notes. Il n'y a pas de contrainte sur l'interface. Une possibilité est d'afficher les notes concernant un patient sur la page affichant les informations du patient (développée au sprint 1).

Ressources :

- La documentation Spring Data MongoDB (rédigée en anglais) ;
- Le cours Utilisez Spring Data MongoDB pour interagir avec des bases de données NoSQL.

Étape 4 : Incorporez le rapport d'évaluation du diabète au projet existant pour le Sprint 3

80 % de progression

Avant de démarrer cette étape, je dois avoir :

- relu les informations du Sprint 3 dans le scénario.

Une fois cette étape terminée, je devrais avoir :

- terminé le microservice back de gestion de l'évaluation des risques de diabète ;
- mis à jour le microservice gérant le front ;
- mis à jour la configuration Spring Cloud Gateway du microservice de gateway.

Recommandations :

- Ici, il n'y a pas de base de données à créer. Mais ce microservice va interroger les deux autres afin d'obtenir des données concernant un patient en particulier, qu'il traitera avant de renvoyer l'assessment (c'est-à-dire, le résultat de l'évaluation des risques) ;
- L'algorithme d'assessment est simple, mais il faut y incorporer toutes les règles fournies ;
- Le microservice front doit être mis à jour. Par exemple, l'assessment peut être affiché sur la page qui affiche les informations du patient (cf sprint 1) ;

- Comme pour les autres microservices back, vous utiliserez Spring Security pour sécuriser les accès.

Points de vigilance :
- L'algorithme d'assessment est simple, mais il faut y incorporer toutes les règles fournies ;
- Il faut faire attention à la casse des notes ;
- Il faut que le endpoint respecte le format de renvoi attendu.

Étape 5 : Dockerisez les microservices

85 % de progression

Avant de démarrer cette étape, je dois avoir :
- fini les trois sprints.

Une fois cette étape terminée, je devrais avoir :
- un fichier docker-compose qui permet de lancer tous les microservices ;
- l'application fonctionnelle.

Recommandations :
- Chaque microservice doit avoir son propre dockerfile, mais l'ensemble du projet doit avoir un seul fichier docker-compose.yml ;
- Ce fichier doit contenir toutes les infos pour le déploiement de l'ensemble du projet ;
- L'utilisation de Docker Desktop est recommandée.

Points de vigilance :
- La configuration peut être complexe, il faut être vigilant sur la cohérence des appellations, notamment par rapport aux noms donnés aux microservices ;
- L'énoncé ne demande pas explicitement que les bases de données soient conteneurisées, mais ce n'est pas interdit de le faire.

Ressources :
- Le chapitre Créez votre premier Dockerfile du cours Optimisez votre déploiement en créant des conteneurs avec Docker ;
- La documentation Containerize an application de Docker (rédigée en anglais).

Étape 6 : Recherchez le Green Code

100 % de progression

Avant de démarrer cette étape, je dois avoir :

- fini tout le code et la conteneurisation.

Une fois cette étape terminée, je devrais avoir :

- identifié les enjeux du Green Code ;
- listé des recommandations d'amélioration « Green » du projet dans le fichier readme.md du repository.

Recommandations :

- Il n'est pas demandé d'appliquer au projet les principes du Green Code, mais bien de comprendre les enjeux et d'être capable de les expliquer :
  - Quel est l'objectif du green code?
  - Comment identifier les parties d'un code qui consomment de la mémoire inutilement?
- Même s'il s'agit de pistes d'améliorations, il est important d'avoir un œil critique sur un projet en cours ;
- Regardez les ressources, mais faites aussi des recherches vous-même.

Ressources :

- Le chapitre Réduisez l'empreinte écologique de votre site web du cours Appliquez les principes du Green IT dans votre entreprise ;
- L'article How Green Are Java Best Coding Practices? de Scitepress ;
- L'article Green code : écrivez du code vert ! de l'Institut du numérique responsable.

Projet terminé !