POC Reporting

14 juin 2024

Sommaire

Objet de ce document

Le reporting de la POC (Preuve de concept) est un document indiquant les choix technologiques pour le développement de l'application ainsi que les résultats obtenus par la POC pour valider si celle-ci est en accord avec les objectifs initiaux.

Historique du document

Date de modification	Objet	Par
01/05/2024	Création du document	Mathieu Pavy

Technologies

Front end

L'application dite "client" à été développée en utilisant la technologie Javascript, plus précisément le framework Angular.
Angular est un framework construit par Google afin de créer des applications mono page (Single Page Application). L'utilisation de ce framework donne un cadre strict pour le développement en plus de fournir une architecture de base opérationnelle, cela permet d'unifier les architectures de développement.
Angular est aujourd'hui très populaire pour sa fiabilité ainsi que les performances.

Back end

L'application back-end à été développée en utilisant la technologie Java (v17), avec Spring Boot à l'appui comme framework.
Spring Boot fournit un cadre permettant la création d'application serveur, il est aussi reconnu pour sa fiabilité, ses performances et sa rapidité pour le développement d'application. Pour ce projet une API (Application Programming Interface) à été développée.

Authentification

Afin de pouvoir faire communiquer le front-end avec le back-end, un JWT (Json Web Token) à été implémenté pour sécuriser les échanges à chaque requête. Ce token permet de vérifier l'identité de l'appelant à chacune des requêtes.

Rappel des exigences de la POC

- Fournir une API RESTful qui renvoie le lieu où se rendre :
  - La technologie Java est imposée par le consortium,
  - L'API doit pouvoir s'inscrire à terme dans une architecture microservice ;
- Fournir une interface graphique qui consomme l'API :
  - Une simple page permettant de sélectionner une spécialité et de saisir la localisation est suffisante,
  - Le consortium impose d'utiliser l'un des frameworks Javascript/Typescript courant du marché : Angular, React, VueJS ;
- S'assurer que toutes les données du patient sont correctement protégées ;
- S'assurer que votre PoC est entièrement validée avec des tests reflétant la pyramide de tests (tests unitaires, d'intégration et E2E) :
  - L'API doit être éprouvée avec des tests de stress pour garantir la continuité de l'activité en cas de pic d'utilisation ;
- S'assurer que la PoC peut être facilement intégrée dans le développement futur : rendre le code facilement partageable, fournir des pipelines d'intégration et de livraison continue (CI/CD) ;
- S'assurer que les équipes de développement chargées de cette PoC sont en mesure de l'utiliser comme un bloc de construction pour d'autres modules ou tout du moins comme un modèle à suivre ;
- Le code est versionné à l'aide d'un workflow Git adapté ;
- La documentation technique de la PoC sera formalisée dans un fichier readme.md respectant le format markdown et contiendra au minimum :
  - les instructions pour l'exécution des tests,
  - le fonctionnement du pipeline,
  - le workflow Git retenu (ce dernier sera détaillé pour qu'il soit réutilisable par les équipes) ;

CI/CD

La CI/CD (Continous integration / Continuous delivery) est une pipeline permettant l'intégration et le déploiement en continu. Dans le cadre de la POC, Jenkins à été retenu pour la création de la pipeline. Voici le détail du fonctionnement de la pipeline :

- Récupération du code depuis le repository (pull)
- Exécutions des tests unitaires (back-end et front-end)
- Si aucune erreur n'est rencontré durant les tests :
    - Copies des sources dans une image Docker
    - Build de l'application dans le conteneur
    - Le conteneur est prêt pour une utilisation en production

Dans le cadre de la POC, la pipeline ne propose pas de déploiement en continu.

Il faudrait aussi ajouter un déclencheur pour "jouer" la pipeline. Très largement, la pipeline s'exécute au moment de compléter une Pull Request.
Cela apporte plusieurs avantages :

- Impossible de publier l'application en pushant la branche main
- On peut imposer une relecture du code par d'autres développeurs.
- Cela permet d'avoir un processus standard dans l'utilisation des branches, PR, merge..

Tests

Tests unitaires

Les tests unitaires visent à tester une “unité” de code, ces tests permettant de vérifier qu’une partie du code respecte bien les exigences attendues.
Ils sont joués automatiquement durant la pipeline afin de s’assurer qu’aucune régression n’est détectée.

Tests fonctionnels

Les tests fonctionnels permettent quant à eux de contrôler une fonctionnalité entière. Les tests sont également exécutés lors du pipeline.

Test de charge

JMeter à été utilisé pour réaliser le test de charge. Ce test permet de contrôler la capacité de l’application pour pouvoir répondre aux clients.
Voici les critères attendues pour valider les performances de l’application :

Critère	**Valeur**
Temps de réponse	<200 millisecondes, avec plus de 800 requêtes par seconde.

Tests end-to-end

Les tests permettent de tester l’interface graphique. Il est possible d'automatiser ces tests afin de contrôler qu’aucun changement visuel inattendu modifie l’apparence de l’application. Selenium a été choisi pour effectuer les tests e2e.

Sécurité

Plusieurs actions sont à mettre en œuvre afin que l'application soit sécurisée, en effet il convient de mettre en place plusieurs niveaux de sécurité pour minimiser les failles dans l'application.

JWT

Un Json Web Token est mis en place pour vérifier l'identité de l'utilisateur pour accéder à une ressource. Ce token permet de contrôler son niveau d'autorisation afin qu'il ne soit pas capable d'accéder à une ressource nécessitant un rôle plus important que le sien.

Refresh Token

En complément avec le JWT, un refresh token est un indispensable pour la sécurité de l'application. Celui-ci permet d'actualiser le jwt courant pour en prolonger la durée sans demander à l'utilisateur de se reconnecter, mais il est également possible de révoquer un token pour interdire l'accès à un utilisateur.

HTTPS

Pour sécuriser la communication entre l'application front-end et le serveur, il est primordial de mettre en place une communication via un protocole sécurisé. Celui-ci permet de chiffrer la communication entre les différentes entités afin qu'un éventuel attaquant ne puisse être capable de lire les données.

Réseau

La sécurité sera également à mettre en place au niveau du réseau, par l'installation d'un VPN (Virtual Private Network).
Physique
Il est indispensable aujourd'hui d'avoir une application qui fonctionne dans un lieu physique sécurisé. Il y a différents moyens à mettre en oeuvre pour cela :

- Sécuriser l'accès au site (badge, porte..)
- Personnel de sécurité
- Sécuriser le WIFI, voir le désactiver pour n'utiliser que du filaire;

Disponibilité

Afin de garantir que l'application soit disponible 24h/24 et 7j/7.
Une installation dans le cloud peut être une solution intéressante. En effet les clouds aujourd'hui présentent plusieurs méthodes afin de garantir la haute disponibilité d'une application.
Il est alors possible de répliquer l'application pour des montés de charges afin que celle-ci ne tombe pas en cas de surutilisation (scalabilité).
La réplication se fait aussi sur différents points géographique. Cela permet d'avoir un temps de réponse le plus faible possible, sans contrainte géographique. Cela prévient aussi des risques (séismes, incendie, inondation), dans ce cas une autre instance prendrait la relève pour assurer la disponibilité de l'application.

Résultats obtenus

Exigences de la POC	Statut
La technologie Java est imposée,	✔
L'API doit pouvoir s'inscrire à terme dans une architecture microservice	✔
Le consortium impose d'utiliser l'un des frameworks Javascript/Typescript courant du marché : Angular, React, VueJS ;	✔
Toutes les données du patient sont correctement protégées	✔
Tests unitaires, d'intégration et E2E	✔
Rendre le code facilement partageable, fournir des pipelines d'intégration et de livraison continue (CI/CD)	✔
Le code est versionné	✔
Une documentation est existante	✔

Recommandations pour la production

S'agissant d'une POC, tous les éléments n'ont pas étés délivrés pour permettre l'utilisation de l'application en production, il y reste des ajustements pour avoir une version utilisable en production :

- Utilisation d'une vraie base de données (SQL, NoSQL)
- Mise en place d'une phase de déploiement sur la pipeline
- Utilisation d'une version de préproduction et de production
- Mise en place d'une documentation de l'API
- Ajouter des tests (unitaires, fonctionnels et E2E)
- Mise en place d'un trigger permettant de lancer un build lors d'un nouveau push
- Hasher le mot de passe utilisateur

Mise en place de l'application

Il est nécessaire d'avoir installer sur la machine serveur les outils permettant de déploiement de l'application à savoir :

- AngularCLI
- Maven
- Docker
- Jenkins
- JMeter

Pour le premier lancement, il faut avoir créer les premiers conteneurs de l'application, en jouant le script de déploiement `first_deploy.bat`

Pour la suite, il faut dans l'interface de Jenkins le script de pipeline présent dans le repository JenkinsFile.

Il suffira alors de `Lancer un build` pour exécuter la pipeline. La pipeline se chargera de récupérer le code du repository sur Github, lancera les tests et créera les nouvelles images et conteneurs.

Données utilisées dans le cadre de la POC

Pour se connecter à l'application, un compte à été créé dont voici les identifiants :
Login : dev@dev.fr
Password: sLNdHqYXd5DQgXGSmH7G

Il y a 4 spécialisations qui sont utilisables :

- Cardiologie
- Immunologie
- Neuropathologie
- Diagnostic

Il y a 8 villes de disponibles :

- Paris
- Toulouse
- Lyon
- Marseille
- Lille
- Strasbourg
- Biarritz

- Brest