Preuve de Concept

Projet : Création d'une PoC

Historique

Date	Version	Commentaires
01/02/2023	0.1	Création du document
08/02/2023	0.2	Contraintes et hypothèses de la PoC
15/02/2023	0.3	Architecture de la PoC et architecture cible
22/02/2023	1.0	Finalisation du document

Parrains

Parrain	Rôle et organisation	Résultats
Kara Trace	CIO, Ursa Major Health	Intégration des systèmes de médecine générale en temps quasi réel avec des prestataires de soins de santé.
Anika Hansen	PDG, Jupiter Scheduling Inc	Architecture de domaine consolidée avec des événements sécurisés circulant en temps réel entre les systèmes
Équipe d'intégration des systèmes de santé du Royaume-Uni	Emergency Expert Systems	Enrichissement des données et accès aux données anonymisées des hôpitaux et des patients
Chris Pike	Architecte métier principal, Schedule Shed Réductions de pannes.	Flux d'événements normalisés et infrastructure de sécurité pour réduire la responsabilité de l'organisation causée par des erreurs de planification

Contraintes

Contraintes	Solutions trouvées
Les systèmes et processus existants ne doivent pas être significativement entravés pendant les phases du projet.	Développement d'un POC n'utilisant pas les services existants
La feuille de route architecturale et les orientations alimentant les exigences fonctionnelles doivent respecter le Cadre des normes numériques et technologiques du NHS	Respect des normes et contraintes de sécurité et de gestion des données NHS (notamment liés au Cloud)
Les données réelles des patients doivent à tout moment rester conformes aux réglementations européennes, notamment le RGPD.	Plusieurs audits seront effectués et une plusieurs tâches seront planifiées par le DPO dès les premiers sprints regroupant, notamment : - Attention particulière aux différentes failles de sécurité (XSS, DDOS..); - Formation en sécurité pour les acteurs du projet (développement et maintenance); - Analyses O'Wasp; - Cryptage des données;
Les phases initiales du projet devraient viser la création de modules de construction réutilisables ou de modèles pour des modules de construction futurs qui respectent les meilleures pratiques convenues.	Développement d'un code réfléchi pour sa réutilisabilité, attention particulière à la dette technique, utilisation de patterns architecturaux connus pour leur évolutivité (microservices, orchestrations des conteneurs, cloud..);

Objectifs de la PoC

Nous rappelons ici les objectifs de la PoC définis de le document d'Énoncé sommaire des travaux (Statement of Architecture Work).
L'intérêt principal est de valider la faisabilité de la solution en utilisant si possible des modules réutilisables pour le développement de la solution et pour l'architecture. Les points suivants doivent être implémentés :

- Une stratégie de test pour la validation de principe;
- Un plan de test;
- Une pipeline CI/CD réutilisable et fonctionnelle;

Risques

Risque	Stratégie d'atténuation	Implémentation dans le PoC
Le système d'intervention d'urgence en temps réel n'est pas adapté aux incidents	Test de performance précoce d'une preuve de concept représentative	Oui
Le système d'intervention d'urgence en temps réel gère la latence concernant la disponibilité des lits des hôpitaux du réseau	Test de performance précoce d'une preuve de concept représentative	Oui
Le système d'intervention d'urgence en temps réel n'offre pas de solution lorsqu'il n'y a pas de lits d'hôpital disponibles pour la spécialisation requise	Attribution à l'hôpital le plus proche disposant de lits	Oui
Le système d'intervention d'urgence en temps réel ne répond pas dans les 200 millisecondes à la demande de lits	Validation de principe pour vérifier qu'un paramètre d'« urgence » est en mesure de fournir au système de réponse le nom d'un hôpital disposant d'un lit en moins de 200 nanosecondes, pendant un pic d'activité	A tester dans des conditions vraisemblables
Le système d'intervention d'urgence ne peut pas être interfacé par d'autres systèmes	Utiliser OpenAPI pour définir les contrats de service. Inclure cela dans la première preuve de concept - personnaliser par la suite en tant que solution building blocks	Oui

Exigences de la PoC

Le tableau suivant présente les différentes exigences définies dans le document d'Hypothèse de développement de la preuve de concept et en quelle mesure la solution développée a été en mesure de répondre à ces exigences.

Exigences	Validation des exigences
Fournir une API RESTful qui tient les intervenants médicaux informés en temps	Validé par la PoC
Sécuriser les données des patients	A validé par des expertises en sécurité et par le DPO (O'Wasp et audit de sécurité conseillés)

Implémenter des tests unitaires, d'intégration, d'acceptation et stress automatisés	Validé par la PoC Nous conseillons cependant d'étendre les tests de stress et de mettre en place un environnement de test et de recette
Rendre la PoC facilement intégrable dans des développements futurs (accessibilité du code, CI/CD, documentation des tests)	Validé par la PoC
Proposer des Solutions Building Blocks réutilisables	Validé par la PoC

Méthodologie de la PoC

Le développement et la conception de la solution de PoC a été réalisée en suivant la méthodologie "Behavior-driven-development".
Le but de cette méthodologie est d'abord de réfléchir à quelle partie du processus et de la solution on souhaite développer, puis d'écrire différents tests qui couvrent les fonctionnalités prévues. Une fois les tests entièrement rédigés, on s'applique à rédiger le code afin que chaque test unitaire n'échoue pas.

Le choix d'une architecture microservice nous a poussé à d'abord déterminer les services que nous serions amenés à développer. Nous avons le choix des microservices suivants :

- Un service patient;
- Un service hôpital;
- Un service docteur;
- Un service urgence qui gère la création et la gestion d'urgences;
- Un service de rendez-vous qui gère la création et la gestion de rendez-vous ;

Nous avons donc écrit des tests qui, pour chaque "service" de notre application, permettent de :

- vérifier la possibilité d'ajouter un objet dans la base;
- vérifier la possibilité de supprimer cet objet de la base;
- vérifier la possibilité de récupérer les objets du service;
- vérifier la possibilité de modifier les objets existants du service;

Nous avons également décidé que la preuve de concept ne nécessitait pas forcément une architecture microservice en soit, et que développer un seule API serait plus simple, plus rapide et ne changerait que très peu le fonctionnement intrinsèque de l'application.

Pour finir, nous avons développé des tests d'intégration validant différents workflow complets, et avons ajouté des tests de stress pour valider le fonctionnement de l'application en conditions réelles.

Améliorations proposées

Nous pensons que les points suivants nécessitent une attention particulière lors des prochaines itérations de travail et de développement sur ce projet :

- Sécuriser le parc informatique lié au système, les données ainsi que la gestion des accès via un système de rôle;
- Mettre en place un système de redondance et de backup;
- Choisir une ou plusieurs solutions Cloud répondant aux exigences en matière de sécurité dûes au domaine métier (l'utilisation d'un Cloud privé est à étudier);
- Utiliser les Solutions Building Blocks existantes pour transformer les différents services en microservices;

Certains de ces points seront abordés dans le document d'Architecture Cible.

Key Performance Indicator

Le but de cette solution est de permettre en priorité d'améliorer la qualité des traitements des urgences et de sauver plus de vie. Le système doit être assez simple à utiliser pour gagner la confiance des utilisateurs.
Le tableau suivant présente les KPI choisies pour valider la réussite de ces objectifs.

KPI	**Niveau attendu**
Acheminement des urgences vers le bon hôpital	Plus de 90%
Le temps moyen de traitement d'une urgence	Réduction du temps de traitement à 12 minutes
Temps de réponse des requêtes aux services	Moins de 200 millisecondes avec une charge de travail allant jusqu'à 800 requêtes/seconde
Respect des normes	Documentation
Instruction pour mise en production	Fournies et validées
Réalisation du projet dans les délais souhaités	A déterminer

● d'améliorer la qualité des traitements d'urgence et de sauver plus de vies ;
● de gagner la confiance des utilisateurs quant à la simplicité d'un tel système.
Nous saurons que nous avons réussi quand nous verrons :
● que plus de 90 % des cas d'urgence sont acheminés vers l'hôpital compétent le plus proche du réseau ;

● que le temps moyen de traitement d’une urgence passe de 18,25 minutes (valeur actuelle) à 12,00 minutes (valeur souhaitée) ;
● que nous obtenons un temps de réponse de moins de 200 millisecondes avec une charge de travail allant jusqu'à 800 requêtes par seconde, par instance de service ;
● que la mise en œuvre explique les normes qu’elle respecte et pourquoi ;
● que les instructions pour mettre en production la PoC sont fournies ;
● que la mise en œuvre est terminée dans le délai imparti.

Signature

Responsable	Signature
Kara Trace	
Conseil d'administration de MedHead	