Spécification des Conditions requises pour l'Architecture

Gosme Anthony

Version 1.0

Table des matières

1 INFORMATION SUR LE DOCUMENT

Nom du projet	*Nouvelle architecture de commerce en ligne V2*
Préparé par	*Anthony Gosme, Architecte solution*
Version	*1.0*
Titre	*Spécification des conditions requises pour l'architecture*
Courriel	*anthonygosme@ocr.com*
Actions	*Approbation, Révision, Information, Classement, Action requise, Participation à une réunion, autre (à spécifier)*

1.1 OBJET DE CE DOCUMENT

La spécification des Conditions requises pour l'architecture fournit un ensemble de déclarations quantitatives qui dessinent ce que doit faire un projet d'implémentation afin d'être conforme à l'architecture.

Une spécification des Conditions requises pour l'architecture constitue généralement un composant majeur du contrat d'implémentation, ou du contrat pour une définition de l'architecture plus détaillée.

Comme mentionné ci-dessus, la spécification des Conditions requises pour l'architecture accompagne le document de définition de l'architecture, avec un objectif complémentaire : le document de définition de l'architecture fournit une vision qualitative de la solution et tache de communiquer l'intention de l'architecte.

La spécification des Conditions requises pour l'architecture fournit une vision quantitative de la solution, énumérant des critères mesurables qui doivent être remplis durant l'implémentation de l'architecture.

2 Mesure du succès

Réf.	Métrique	Valeur initiale	Valeur cible
KPI1	Adhésions journalières utilisateurs		+10%
KPI2	Adhésions journalières producteurs	1,4 / mois	> 4 / mois
KPI3	Déploiement de mise à jour	3,5 / semaine	> 1 / semaine
KPI4	Incident de production	> 25 / semaine	< 1 / mois

3 Conditions requises pour l'architecture

Réf.	Qualité architecturale	Besoin non fonctionnel	Solution
Q1	Disponibilité	Mise à jour service sans interruption	Helm & Kubernetes
Q2	Disponibilité	Mise à jour BDD sans interruption	MongoDb managed multi region AWS
Q3	Évolutivité	Version de taille réduite	Layers docker
Q4	Performance	Performance similaire dans différentes régions	AWS multirégion
Q5	Qualité	Fiabilité nouvelle version	DevOps
Q6	Qualité	Problème intégrations	Microservice + DevOps + CI/CD
Q7	Qualité	Tests et qualification rapides	Env. Dev + test + qualification + CI/CD
Q8	Évolutivité	Architecture évolutive	Microservice
Q9	Disponibilité	Accessibilité téléphone et browser	Ionic
Q10	Performance	Comptabilité faible bande passante	Application simple page : Angular
Q11	Disponibilité	Géolocalisation des ressources	AWS CloudFront, DNS
Q12	Évolutivité	COTS open source et dev spécifiques	
Q13	Évolutivité	Pile technologique unique	Docker + OpenJDK+ Spring Boot + Angular

4 ACCORD DE NIVEAU DE SERVICE

4.1 ACCORD BUSINESS SLA

Les SLA feront l'objet de plans de tests

Réf.	Objectif	SLO
SLA1	Disponibilité	99.7% -> 2 h
SLA2	Réponse support	98% niveau 1 -> 3 h 98% niveau 2 -> 8 h 98% niveau 3 -> 3 jours 98% niveau 4 -> 35h
SLA3	Temps de réponse max (3G+ - 3.6Mb/s)	99% -> 4 secondes 80% -> 1.5 secondes
SLA4	Temps entre incidents	1 mois
SLA5	Temps de perte de données après incident RPO (recovery point objectif)	10 min
SLA6	Temps d'indisponibilité de la plateforme après incident (RTO recovery time objectif)	1 h

4.1 OBJECTIFS APPLICATIF

Réf.	Objectif	
OA1	Les mises à jour applicatives et données se font sans arrêt des services	98% sans arrêts des services
OLA	Qualité de l'application	Sonarqube - note A
SLA3		99% -> 4 secondes 80% -> 1.5 secondes
SLA4	Temps entre incidents	1 mois
SLA5	Temps de perte de données après incident RPO (recovery point objectif)	10 min
SLA6	Temps d'indisponibilité de la plateforme après incident (RTO recovery time objectif)	1 h

5 Implémentation

5.1 LIGNES DIRECTRICES POUR L'IMPLÉMENTATION

Réf.	Ligne directrice
LD1	Utiliser l'Open source
LD2	Utiliser des solutions de Cloud natives et cloud SaaS
LD3	Backlog fonctionnel - Product
LD4	Backlog architecture - Architecture
LD5	Microservices
LD6	Validation UX CX
LD7	Validation QA
LD8	Framework Scrum
LD9	Durabilité des informations
LD10	Principe de conception SOLID
LD11	Garantir l'intégrité des données stockées
LD12	Pipeline delivery avec environnement séparé DEV, QA, PROD
LD13	Utilisation de design patterns (ex. : singleton, decorateur, factory, strategy, observer, builder)
LD14	Single Sign On

5.2 SPÉCIFICATION POUR L'IMPLÉMENTATION

Réf.	Implémentation	Technologies
SPI1	Stack unifié pour les microservices	OpenJDK, Spring Boot, Angular, AWS MongoDB
SPI2	Stack unifié pour les applications smartphone	Ionic
SPI3	Stack unifié de déploiement	Docker, OpenJDK, Helm, Kubernetes
SPI4	Test unitaire	Junit, mockito,
SPI5	Performance, résilience	Jmeter
SPI6	Test Fonctionnel et IHM	Selenium, Cucumber+Gherkins
SPI7	Test de qualité de code	SonarCube, SonarLint
SPI8	CI/CD – Devops	AWS Codepipeline, CodeDeploy, CodeCommit, CodeBuild
SPI9	Dépôt Git	Repository de code et de configuration des services
SPI10	Infra As Code	AWS CloudFormation
SPI11	Log centralisé	AWS CloudWatch
SPI12	Piste d'audit	AWS CloudTrail
SPI13	Monitoring production	ELK, Metricbeats, Jolokia
SPI14	Authentification basée sur rôles	AWS IAM
SPI15	Répartition de charge	AWS ELB
SPI16	Pare-feu applicatif WAF	AWS WAF
SPI17	CDN	AWS Cloudfront
SPI18	Élasticité des services en production	Kubernetes & AWS auto scaling

5.3 STANDARD POUR L'IMPLÉMENTATION

Réf.	Intérêt	Norme
STD1	Sécurisation des échanges	TLS
STD2	Chiffrement des données	AES-256
STD3	Encodage utf8	UTF8

5.4 CONDITION REQUISE D'INTERPOLARITÉ

Réf.	Intérêt	Norme
INT1	Protocole d'échange	REST
INT2	Format de données échangé	JSON
INT3	Spécification d'interfaces	OPENAPI
INT4	Couplage faible des sous-systèmes	HTTP REST API
INT5	Point accès service	API Gateway
INT6	Bus d'échange de données	Amazon Simple Queue Service

5.5 CONDITIONS REQUISES POUR LE MANAGEMENT DU SERVICE IT

Réf.	Aspects	Implémenter	Explication
CRM1	Dev	ITIL, JIRA	Bonne pratique de management de service IT
CRM2	Dev	Test d'intégration	Test technique
CRM3	Dev	Scrum & Kanban	Gestion Backlog ave Product Owner
CRM4	Devops & QA	K8S et Helm	Déploiement, orchestration et MAJ des services
CRM5	Devops & QA	QA	Rôle et pratique de qualification logicielle
CRM6	Devops & QA	Test fonctionnel	Test des fonctionnalité
CRM7	Devops & QA	Smoke Test & Health check	Test basique sur environnement de production
CRM8	Devops & QA	Environnement de qualification	Pour jouer les tests fournis par l'équipe R&D et ceux de QA
CRM9	Production	SLA	Recueillir les métriques
CRM10	Production	Monitoring ELK	Monitoring de qualité de service
CRM11	Production	Support client	Recueil et traitement des retours client
CRM12	Production	Erreurs connues	Base de données d'erreurs connue
CRM13	Architecture	Répertoire de document d'architecture	MAJ des infos d'architecture
CRM14	Architecture	Togaf	Cadre d'architecturre
CRM13		Backlog	Gestion Backlog ave Product Owner
CRM14			

6 CONTRAINTES

Réf.	Contrainte
C1	Le projet initial est approuvé pour un coût de 50 000 USD (45 190 €) et une période de 6 mois est prévue pour définir l'architecture et préparer un projet de suivi afin de développer un prototype.
C2	L'architecture doit permettre d'obtenir le meilleur rapport qualité-coût.
C3	L'architecture peut inclure de nouveaux composants personnalisés ou des composants du commerce pour favoriser la flexibilité, la stabilité et l'extensibilité.
C4	Les solutions open source sont préférables aux solutions payantes.
C5	Le support continu des composants doit être pris en compte lors de leur sélection ou lors des prises de décision de création ou d'achat.
C6	Toutes les solutions du commerce ou open sources doivent, dans la mesure du possible, faire partie d'une même pile technologique afin de réduire les coûts de maintenance et de support continus.
C7	Évolutivité de la solution en utilisant les microservices

7 Hypothèses

Réf.	Hypothèse
H1	La nouvelle plateforme sera totalement indépendante de l'ancienne
H2	La bascule du service vers la nouvelle plateforme impactera le service pendant 24 heures
H3	Montée en charge progressive de la nouvelle plateforme
H4	Arrêt de développement de fonctionnalité sur l'ancienne plateforme