Déclaration de Travail d’Architecture

Projet : ***Concevez une nouvelle architecture afin de soutenir le développement de votre entreprise***

Client : FOOSUS

Table des Matières

1. Déclaration de travail d’architecture
2. Objectifs et périmètre
3. Rôles et responsabilités
4. Approche architecturale
5. Plan de travail
6. Risques et facteurs de réduction
7. Critères d’acceptation et procédures
8. Approbations signées

Information sur le document

Nom du projet	***Concevez une nouvelle architecture afin de soutenir le développement de votre entreprise***

Préparé par :	Henrick AGNAME
N° de version du document :	0.1
Titre :	*Déclaration de travail d'architecture*
Date de version du document :	15/06/2023
Revu par :	Henrick AGNAME
Historique de versions du document	Voir git

Déclaration de travail d'architecture

Requête du projet et contexte

Les principaux objectifs de l'entreprise en matière d'architecture sont les suivants.
- Tirer parti de la géolocalisation pour relier des fournisseurs et des consommateurs et pour proposer des produits disponibles à proximité des lieux de résidence de ces derniers.
- L'architecture devra être évolutive pour permettre à nos services de se déployer sur diverses régions à travers des villes et des pays donnés.
- Notre solution doit être disponible pour nos fournisseurs et nos consommateurs, où qu'ils se trouvent. Cette solution doit être utilisable avec des appareils mobiles et fixes. Elle doit tenir compte des contraintes de bande passante pour les réseaux cellulaires et les connexions Internet haut débit.
- Elle doit pouvoir prendre en charge différents types d'utilisateurs (par exemple, fournisseurs, back-office, consommateurs) avec des fonctionnalités et des services spécifiques pour ces catégories

Description du projet et périmètre

Vue d'ensemble

Alignement stratégique

Objectifs et périmètre

Objectifs

Les objectifs business de ce travail d’architecture sont les suivants :

Objectif Business	***Notes***
Maintien de la solution existante pendant le processus de réalisation de la nouvelle solution	Eviter la mise hors service de la plateforme et d'optimiser le taux d'adhésion des fournisseurs et des consommateurs.
Augmentation des campagne marketing	Mettre en place des campagnes marketing de grande envergure ciblées par zone géographique tout en supportant la charge.
Élasticité de la pile technologique	Le système doit être capable de répondre aux besoins et aux exigences de la clientèle en constante évolution.
Reduction des pannes lié au flux de connexion et aux demandes de service	Absorber les pics d'utilisateurs et le lancement de campagnes marketing
Amélioration de l’accès au système en cas de surcharge	En cas de surcharge du système, les utilisateurs connectés doivent avoir une accessibilité réduite ou dégradée, ce qui signifie que les services devraient fonctionner plus lentement, être moins réactifs ou présenter des limitations temporaires.
Une plateforme géographiquement disponible	Les particularités locales peuvent inclure les différences culturelles, linguistiques, réglementaires ou autres spécificités liées à une région ou un pays. La plateforme devra permettre une personnalisation en fonction des besoins spécifiques de chaque marché local pour répondre aux

	exigences des utilisateurs. De plus, les facteurs de réseau (connexion lente, haut débit) ne devraient pas entraîner une interruption de l'accès aux différents services.
Une disponibilité accrue	La plateforme devra rester disponible lors de la livraison de nouvelles versions ou lors de la modification du schéma de base de données.
Une expérience utilisateur commune et unique	Tous les utilisateurs devront bénéficier d'une expérience utilisateur identique, peu importe leur localisation géographique.
Réduction des pannes liées à l'intégration et aux tests de nouvelles versions.	Le temps nécessaire pour examiner et tester chaque nouvelle version dans un environnement de production devra être accéléré avant sa mise en production, afin de garantir une meilleure qualité et performance du produit final.
Améliorer les produits en fonction des besoins des utilisateurs et optimiser leur satisfaction tout en respectant les contraintes budgétaires.	Mettre en place un système de suivi et d'analyse de l'utilisation des solutions par les utilisateurs afin de mieux comprendre leurs besoins, ce qui permettra de faire évoluer les produits et de mieux répondre aux attentes des utilisateurs. Il convient également de mettre en place une plateforme de tests pour les nouveaux produits, permettant de recueillir des retours d'expérience proches de ceux des utilisateurs finaux, et ainsi d'améliorer les produits avant leur lancement sur le marché.
Nombre d’adhésion utilisateurs par jour	Augmentation de 10%
Adhésion de producteurs alimentaires	Passer de 1,4/mois à 4/mois
Délai moyen de parution	Réduit de 3,5 semaines à moins d’une semaine

Taux d’incidents de production P1	Pour commencer : réduit de >25/mois à moins de 1/mois.

Périmètre

Parties prenantes, préoccupations, et visions

Le tableau suivant montre les parties prenantes qui utilisent ce document, leurs préoccupations, et la façon dont le travail d’architecture répondra à ces préoccupations par l’expression de plusieurs visions.

Partie prenante	Préoccupation	Vision
Natasha Jarson, CIO Daniel Antony, CPO Jo Kumar CFO	Innover dans le périmètre d’une Architecture d’Entreprise	Il sera nécessaire d'établir des processus de travail clairs et de capitaliser sur les données et les ressources pour soutenir l'innovation au sein de l'architecture d'entreprise et assurer la réussite à long terme de l'entreprise.
CMO, Natasha Jarson CIO, Daniel Antony CPO, Jo Kumar CFO	Soutenir l’innovation technique rapide et l’expérimentation	Mise en place d'une architecture micro-service avec des services à responsabilités uniques, standardisation des processus (pratiques agiles), mise en place de l'intégration continue et du déploiement continu, et choix d'un stack technique optimal pour un hébergement cloud.
CMO, CPO, Jack Harkness COO	Visibilité de la plateforme	• Visibilité sur l'utilisation des logiciels : mettre en place des outils de suivi et d'analyse pour comprendre comment les utilisateurs interagissent avec les logiciels, afin d'identifier les points

(Directeur des opérations)		d'amélioration et d'optimiser l'expérience utilisateur. • Capacité d'inversion des décisions d'architecture avec un coût minimal : mettre en place une architecture micro services qui permet de remplacer ou de modifier un composant sans impacter l'ensemble du système. • Réplication sur une plateforme pour tester de nouveaux produits : mettre en place un environnement de test qui reproduit l'infrastructure de production et permet d'expérimenter de nouvelles fonctionnalités de produits ou de services en toute sécurité. Cela servira à valider les innovations avant de les déployer sur la plateforme principale, garantissant ainsi leur compatibilité avec les objectifs commerciaux fondamentaux
Ash Callum, CEO Jo Kumar CFO	Taux d'inscription de nouveaux utilisateurs	• Le nouveau système devra être disponible partout dans le monde et fournir une expérience utilisateur similaire, peu importe la localisation de l'utilisateur. • Le nouveau système devra fournir une fonctionnalité de géolocalisation. • Le nouveau système devra être scalable et s'adapter à l'évolution de la base de clientèle (mise à l'échelle).
TODO	Améliorer la réputation de Foosus sur le marché grâce à	• Mise en place de tests automatisés (tests unitaires, d'intégration et de bout en bout). • Automatisation du processus de

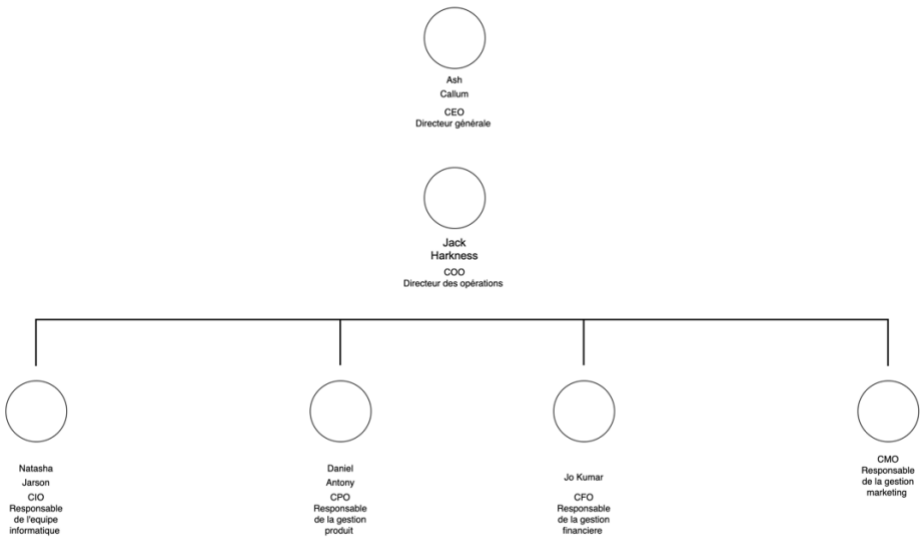
	la stabilité.	déploiement et d'intégration de nouvelles fonctionnalités et correctifs pour minimiser les erreurs humaines. • Tester les nouvelles versions du logiciel avec un petit groupe d'utilisateurs avant de les déployer sur l'ensemble des utilisateurs. • Mettre en place un processus de gestion des incidents pour traiter rapidement les problèmes qui se produisent, ce qui inclut la détection, la communication et la résolution des incidents, ainsi qu'une capitalisation de ces données pour éviter que des problèmes similaires ne se reproduisent. • Mise à l'échelle horizontale de l'architecture afin de supporter la charge due à l'augmentation du nombre d'utilisateurs et de zones géographiques sans impacter le service. • Mise en place d'un déploiement séquentiel par région, ce qui permettra de détecter puis résoudre d'éventuels problèmes potentiels sans affecter l'ensemble des utilisateurs.

Approche managériale

Procédures de changement de périmètre

Rôles et responsabilités

Structure de gouvernance


Ash
Callum
CEO
Directeur générale
Jack
Harkness
COO
Directeur des opérations
Natasha
Jarson
CIO
Responsable
de l'equipe
informatique
Daniel
Antony
CPO
Responsable
de la gestion
produit
Jo Kumar
CFO
Responsable
de la gestion
financiere
CMO
Responsable
de la gestion
marketing

PROCESS DU PROJET

- Réunion régulière : Il sera nécessaire de mettre en place des réunions hebdomadaires avec l'ensemble de l'équipe pour discuter de l'avancement du projet, des problèmes rencontrés et des solutions potentielles. Cela permettra de maintenir une communication ouverte avec les membres de l'équipe.

- Comité de pilotage : Mise en place d'un comité de pilotage composé de toutes les parties prenantes clés. Le comité se réunira régulièrement, soit une fois par mois, pour évaluer l'état du projet, prendre des décisions stratégiques et valider les orientations à suivre.

- Répertoire de documents : Le répertoire de documents existant dans lequel figurent tous les documents liés au projet devra être fourni régulièrement de manière claire pour faciliter la navigation.

- Gestion de la configuration : Mise en place d'un système de gestion de configuration pour assurer le contrôle des versions par service et des documents associés. Chaque service sera indépendant et pourra être testé et déployé de manière isolée. Il sera plus facile de suivre les modifications et les mises à jour spécifiques à chaque service. De plus, les équipes pourront travailler sur des services individuels sans interférer avec le travail des autres, ce qui réduira les conflits et les problèmes de coordination.

- Assurance qualité : Mise en place d'un processus d'assurance qualité pour garantir que le logiciel développé répond aux normes de qualité et aux exigences des clients. Il sera nécessaire de mettre en place une planification de tests, la réalisation de revues de code et la réalisation de revues de code.

- Procédure en cas d'escalade : Mettre en place un processus clair pour signaler et résoudre les problèmes qui ne peuvent pas être résolus au niveau de l'équipe. Cela inclura la désignation d'une personne responsable de l'escalade ainsi que la communication des problèmes au comité de pilotage ou à la direction.

- Procédure en cas de changement : Mise en place d'un processus de gestion des changements pour traiter les demandes de modification de la portée, des exigences ou des délais du projet. Cela inclura l'évaluation des impacts potentiels, la prise de décision quant à l'acceptation ou au refus des changements et la communication des changements aux parties concernées.

Rôles et responsabilités (RACI)

Objectif/Taches	CIO	CPO	CFO	CMO	COO	CEO	Architecte	Développeurs	DBA	Testeurs	Equipes UX	Equipes CX
Innover dans le périmètre d'une Architecture d'Entreprise												
Réunion régulière	I	I	I	I	I	A	I	R	R	R	R	R
Comité de pilotage	R	R	R	R	R	A	I	I	I	I	I	I
Répertoire de documents	I	I	I	I	I	A	R	R	R	R	R	R
Management de la configuration	I	I	I	I	I	A	R	R	R	R	R	R
Assurance qualité	I	I	I	I	I	A	C	R	R	R	R	R
Procédure en cas d'escalade	I	I	I	I	I	A	R	R	R	R	R	R
Procédure en cas de changement	I	I	I	I	I	A	C	R	R	R	R	R
Soutenir l'innovation technique rapide et l'expérimentation												
Mise en place d'une infrastructure runtime	R	I	I	I	I	A	A	I	A	I	I	I
Hébergement sur le cloud	R	I	I	I	A	A	A	I	I	I	I	I
Extention base de données	R	I	I	I	A	A	A	I	A	I	I	I
Développement d'interface front	I	A	I	I	I	A	R	R	I	C	C	C
Interface de recherche	I	A	I	I	I	A	R	R	I	C	C	C
Interface de commande	I	A	I	I	I	A	R	R	I	C	C	C
interface d'inventaire	I	A	I	I	I	A	R	R	I	C	C	C
Orchestrat et développement de services	R	A	I	I	I	A	R	R	C	C	C	C
Système d'inventaire	R	A	I	I	I	A	R	R	C	C	C	C
Système de comande	R	A	I	I	I	A	R	R	C	C	C	C
Système de recherche	R	A	I	I	I	A	R	R	C	C	C	C
Système de facturation	I	A	R	I	I	A	R	R	C	C	C	C
Système de mail	I	A	I	I	R	A	R	R	C	C	C	C
Securisation des transaction	R	A	I	I	I	A	R	R	C	C	C	C
Système d'analyse de données	R	A	C	C	I	A	R	R	C	C	C	C

Approche architecturale

Process d'architecture

La méthode de développement d'architecture TOGAF (ou ADM pour « Architecture Development Method ») décrit une méthodologie des meilleures pratiques pour le développement architectural. Néanmoins, toutes les phases ne sont pas également pertinentes pour chaque projet. Le tableau ci-dessous décrit l'utilisation de l'ADM pour ce projet spécifique.

Phase	**Entrée/Sortie**	**Notes**
Préliminaire	Besoins et objectifs business / Énoncé du périmètre de l'architecture	Définir le périmètre et les objectifs du projet en se basant sur les éléments fournis
A —Vision de l'architecture	Énoncé du périmètre de l'architecture / Vision de l'architecture	Prendre en compte les objectifs business et les notes pour définir la vision de l'architecture
B —Architecture business	Vision de l'architecture / Modèle d'architecture	Identifier les processus métier et les objectifs business

	business Roadmap d'architecture business	
C — Architecture des systèmes d'information	Modèle d'architecture business / Modèle d'architecture des systèmes d'information Roadmap d'architecture des systèmes d'information	Définir les systèmes d'information en se basant sur les notes fournis
D — Architecture technologique	Modèle d'architecture des systèmes d'information / Modèle d'architecture technologique / Roadmap d'architecture technologique	Identifier les technologies et les infrastructures nécessaires en se basant sur les notes fournis
E —Opportunités et solutions	Modèle d'architecture technologique / Portefeuille de projets Évaluation des solutions	Identifier les opportunités et les solutions pour combler les écarts entre l'état actuel et l'état cible
F —Planning de migration	Portefeuille de projets / Plan de migration	Établir un plan de migration pour passer de l'état actuel à l'état cible

G —Gouvernance de l’implémentation	Plan de migration / Processus de gouvernance	Mettre en place un processus de gouvernance pour superviser et guider l'implémentation des projets
H —Management du changement d’architecture	Processus de gouvernance / Plan de gestion du changement	Gérer les aspects humains et organisationnels du changement
Management des conditions requises	Besoins et objectifs business / Mise à jour des besoins et objectifs business	S'assurer que les besoins et les objectifs business
<<Fournir toutes notes supplémentaires sur les phases, les itérations, etc.>>		

Contenu de l’architecture

Le cadre de contenu d’architecture TOGAF (ou ACF pour « Architecture Content Framework ») fournit une catégorisation des meilleures pratiques pour le contenu de l’architecture. Néanmoins, tous les éléments ne sont pas également pertinents pour chaque projet. Le tableau ci-dessous décrit les zones de contenu pertinentes pour ce projet spécifique.

Zone de contenu	**Entrée/Sortie**	**Notes**
Principes, Vision, et Conditions requises de l’Architecture	Entré	Périmètre du projet, objectifs business, objectifs techniques, taux d'inscription de nouveaux utilisateurs, amélioration de la réputation de Foosus, visibilité de la plateforme, soutien à l'innovation technique rapide et à l'expérimentation, contraintes

		budgétaires.
Architecture Business	Sortie	Modélisation des processus métier, identification des acteurs et des rôles, cartographie des processus aux services et applications, analyse des besoins et des exigences fonctionnelles des utilisateurs, définition des objectifs de performance.
Architecture des systèmes d’information — Données	Sortie	Modèles de données, schémas de base de données, intégrité des données, gestion des données, stockage des données, métadonnées, contrôle d'accès aux données, flux de données entre les applications et services.
Architecture des systèmes d’information — Applications	Sortie	Inventaire des applications, description des fonctionnalités, interfaces entre les applications, intégration des applications, architecture micro-services, intégration continue et déploiement continu, choix du stack technique pour l'hébergement cloud.
Architecture technologique	Sortie	Infrastructure, réseau, sécurité, standards et protocoles, middleware, plateformes, choix des technologies pour le développement d'applications, déploiement et gestion des ressources, élasticité de la pile technologique, capacité à supporter des pics d'utilisateurs et des campagnes marketing.
Réalisation de l’architecture	Sortie	Plan de mise en œuvre, plan de migration, gouvernance de l'implémentation, gestion du changement d'architecture, surveillance et ajustements, tests et validation des nouvelles

		fonctionnalités, gestion des incidents, déploiement séquentiel par région, amélioration des produits en fonction des besoins des utilisateurs, suivi et analyse de l'utilisation des solutions par les utilisateurs.

Méthodologies pertinentes et normes de l’industrie

Méthodologies pertinentes et normes de l’industrie :

1. **TOGAF (The Open Group Architecture Framework)** : Utilisation de TOGAF comme cadre de référence pour la méthodologie d'architecture d'entreprise. Il fournit une approche structurée pour le développement, la maintenance et la gouvernance de l'architecture d'entreprise.

2. **ITIL (Information Technology Infrastructure Library)** : Utilisation des meilleures pratiques ITIL pour la gestion des services informatiques. ITIL permet de garantir l'alignement des services informatiques sur les besoins de l'entreprise et d'améliorer la qualité du service.

3. **Agile/Scrum** : Utilisation des méthodologies agiles, telles que Scrum, pour favoriser l'innovation, la collaboration et la flexibilité dans le développement de projets.

4. **DevOps** : Application des principes DevOps pour faciliter l'intégration continue et le déploiement continu des services et des applications, améliorant ainsi l'efficacité opérationnelle et la qualité des livrables.

5. **ISO/IEC 27001:** Il s'agit d'une norme internationale pour la gestion de la sécurité de l'information (ISMS). Elle établit les exigences pour la mise en place, la maintenance et l'amélioration continue d'un système de gestion de la sécurité de l'information.

6. **NIST (National Institute of Standards and Technology) Cybersecurity Framework:** Ce cadre fournit des lignes directrices pour la gestion des risques liés à la cybersécurité et la protection des infrastructures critiques. Il vise à aider les organisations à mieux comprendre, gérer et réduire les risques liés à la cybersécurité**.**

7. **OWASP (Open Web Application Security Project):** L'OWASP est une organisation à but non lucratif qui vise à améliorer la sécurité des logiciels. Ils fournissent des outils, des ressources et des bonnes pratiques pour aider les organisations à sécuriser leurs applications web**.**

8. **GDPR (General Data Protection Regulation):** Le GDPR est un règlement de l'Union européenne qui vise à protéger les données personnelles des citoyens de l'UE. Il impose des exigences strictes en matière de protection des données et de notification en cas de violation de données.

9. **HIPAA (Health Insurance Portability and Accountability Act):** Pour les organisations du secteur de la santé, la conformité à la loi américaine HIPAA est essentielle pour protéger les informations de santé et les données des patients.

Soutien au Continuum de l’entreprise :

- **Niveau de détail** : Stratégique pour une vision globale et à long terme, segment pour des domaines spécifiques, et capacité pour des fonctionnalités individuelles.
- **Période** : L'architecture couvre une période de temps à moyen et long terme, selon les objectifs de l'entreprise et les priorités des parties prenantes.
- **Sujet** : Le domaine de sujet couvert inclut les domaines business, systèmes d'information (données et applications) et technologique.
- **Niveau d’abstraction :** L'architecture comprend des représentations concrètes des solutions et une architecture de référence plus abstraite pour faciliter la compréhension et la communication entre les parties prenantes.
- **Ligne de base vs cible** : L'accent est mis sur la documentation de la ligne de base actuelle et la proposition d'une future architecture cible. Ces activités seront abordées en séquence, en commençant par la documentation de la ligne de base et en passant ensuite à l'élaboration de l'architecture cible.
- **Itération** : L'itération est utilisée dans l'ADM pour permettre un développement progressif et une amélioration continue de l'architecture.
- **Partitionnement** : Les relations avec d'autres travaux d'architecture au sein d'un environnement partitionné seront prises en compte pour garantir la cohérence et l'intégration entre les différentes parties de l'architecture d'entreprise.

Plan de travail

Hébergement sur le cloud

Activités

- Choix du fournisseur de services cloud
- Planification de l’infrastructure
- Mise en place des services cloud

Livrables

Critères d'évaluation :

1. Coût
2. Maturité des services offerts
3. Écosystème et intégration avec les outils existants
4. Régions et zones de disponibilité
5. Conformité et sécurité

Amazone Web Services (AWS)

Mise en place de L’infrastructure runtime

Activités

- Concevoir l'infrastructure runtime
- Identifier les ressources nécessaires pour l'infrastructure runtime
- Mettre en place l'infrastructure runtime

Livrables

Environnement de développement :

- Système d'exploitation : Windows
- Serveur d'application : Apache Tomcat
- Base de données : Oracle Database
- Langages de programmation : Java, JavaScript, HTML, CSS
- Outils de développement : IntliJI

Environnement de test :

- Système d'exploitation : unix
- Serveur d'application : Apache Tomcat
- Base de données : Oracle Database
- Langages de programmation : Java, JavaScript, HTML, CSS
- Outils de test : Junit

Environnement de production :

- Système d'exploitation : unix
- Serveur d'application : Apache Tomcat
- Load balancer : Nginx
- Base de données : Oracle Database
- Langages de programmation : Java, JavaScript, HTML, CSS
- Outils de surveillance : Nagios, Zabbix, Splunk

Système d’exploitation Windows :

- Méthodes de sécurisation :
    - Configuration des politiques de sécurité et des pare-feux
    - Installation des mises à jour de sécurité régulières
    - Utilisation de comptes d'utilisateur avec des permissions limitées
    - Utilisation d'un logiciel antivirus
- Déploiement automatique sans risque d'interruption :
    - Utilisation de l'outil Ansible pour la configuration et l'automatisation du système d'exploitation
    - Utilisation de techniques de déploiement progressif pour minimiser les risques d'interruption

Serveur d’application Apache Tomcat :

- Méthodes de sécurisation :
    - Configuration des politiques de sécurité et des pare-feux
    - Utilisation de certificats SSL pour les connexions HTTPS
    - Utilisation de mots de passe forts et de techniques d'authentification avancées
- Déploiement automatique sans risque d'interruption :
    - Utilisation de l'outil Ansible pour la configuration et l'automatisation des serveurs d'application
    - Utilisation de techniques de déploiement progressif pour minimiser les risques d'interruption

Load balancer Nginx :

- Méthodes de sécurisation :
    - Configuration des politiques de sécurité et des pare-feux
    - Utilisation de certificats SSL pour les connexions HTTPS
    - Utilisation de mots de passe forts et de techniques d'authentification avancées
    - Utilisation de l'authentification à deux facteurs
    - Configuration des règles de gestion de trafic pour minimiser les attaques DDoS
- Déploiement automatique sans risque d'interruption :
    - Utilisation de l'outil Ansible pour la configuration et l'automatisation du load balancer
    - Utilisation de techniques de déploiement progressif pour minimiser les risques d'interruption

Base de données Oracle:

- Méthodes de sécurisation :
    - Configuration des politiques de sécurité et des pare-feux
    - Utilisation de mots de passe forts et de techniques d'authentification avancées
    - Utilisation de l'authentification à deux facteurs
    - Utilisation de la cryptographie pour chiffrer les données sensibles
    - Configuration de sauvegardes régulières et de restaurations de données
- Déploiement automatique sans risque d'interruption :
    - Utilisation de l'outil Ansible pour la configuration et l'automatisation des clusters de base de données
    - Utilisation de techniques de déploiement progressif pour minimiser

Extension base de données

Activités

- Analyse des besoins
- Conception de la nouvelle base de données : Concevoir les modifications de la structure de la base de données, y compris les nouvelles tables, les relations, les index et les contraintes.
- Développement : Créer et modifier les requêtes SQL, les procédures stockées, les fonctions et les déclencheurs pour implémenter les modifications de la base de données.
- Valider les modifications de la base de données dans un environnement de test
- Planifier et exécuter la migration des données existantes vers la nouvelle structure de base de données en préservant l'intégrité des données
- Déploiement : Mettre en œuvre les modifications de la base de données dans l'environnement de production en minimisant les interruptions de service pour les utilisateurs finaux.
- Sauvegarde et restauration : Mettre en place des processus de sauvegarde et de restauration pour protéger les données et faciliter la récupération en cas de problèmes.
- Maintenance et optimisation

Livrables

1. Oracle Database
2. Oracle SQL Developer :
    - Gestion de bases de données (concevoir, développer, tester et déployer des objets de base de données, exécuter des requêtes SQL et d'administrer la base de données.)
3. Langage SQL (Structured Query Language) pour interagir avec la base de données (procédures stockées, des fonctions, des déclencheurs)
4. Environnement de développement intégré (IDE) : Oracle SQL Developer
5. Hibernate ORM (mappage object relatonnel) pour faciliter le développement de la couche d’accès aux données et la communication entre l’application Java et la base de données.
6. Respect des norme JPA (Java persistence API)
7. Mise en place d’un pilote JDBC pour la communication entre hibernate et la base de données
8. Outils de versionnage et de gestion des sources (GIT) pour gerer et suivre les modifications apportées au code SQL, aux scripts de gestions de la base de données.
9. Cluster 3 nœud

Développement d'interfaces front

Activités

- Analyse des besoins
- Conception des interface utilisateur (UI)
- Conception de l’expérience Utilisateur (UX)
- Développement du code Front-end (Framework)
- Intégration avec les services back end (Api)
- Tests: (test unitaires, test integrations, test multi-plateformes)
- Optimisation des performances
- Deploiement et maintenance

Livrables

Front-end Web :

1. Framework JavaScript (Angular)
2. HTML5, CCS3
3. Librairie UI/UX : Bootstrap
4. Gestion de l’etat des données : Bibliothèque NgRx
5. Gestion des formulaires : Angular Reactive forms
6. Gestion des tests Jasmine

Front-end Mobile :

1. Framework Ionic (Angular)
2. HTML5, CCS3
3. Librarie UI/UX : Bootstrap
4. Gestion de l’état des données : Bibliothèque NgRx
5. Gestion des formulaires : Angular Reactive forms
6. Gestion des tests Jasmine

API Gateway

Activités

- Conception et définition des spécifications :
  a. Déterminer les exigences en matière d'authentification, d'autorisation et de sécurité.
- Sélection et configuration de la technologie
- Développement
- Tests :
  a. Réaliser des tests unitaires, des tests d'intégration et des tests de charge pour s'assurer que l'API Gateway fonctionne correctement et répond aux exigences de performance.
  b. Valider les mécanismes d'authentification et d'autorisation, le routage des requêtes, la gestion des erreurs et autres fonctionnalités.
- Documentation :
  a. Documenter les points de terminaison, les paramètres, les codes de réponse, les

formats de données et les exemples d'utilisation de l'API Gateway.
    b. Rédiger des guides pour les développeurs et les opérateurs sur la façon de déployer, configurer et surveiller l'API Gateway.
- Déploiement
- Intégration continue et déploiement continu
- Surveillance et maintenance

Livrables

1. Langage de programmation : Java
2. Version : Java 11 (ou une version ultérieure stable)
3. Spring Cloud Gateway
4. Eureka (Spring Cloud Netflix) : pour l'enregistrement et la disponibilité des services (annuaire de microservice)
5. Spring Security : pour l'authentification et d'autorisation
6. Certificat SSL/TLS
7. Traçabilité et surveillance :
8. Sleuth (Spring Cloud) : Un outil de traçabilité pour suivre les requêtes à travers les micro services.
9. Zipkin : Un système de traçage distribué pour visualiser et analyser les performances et la latence des requêtes.
10. Prometheus : Un système de surveillance et d'alerte pour collecter et stocker les métriques.
11. Grafana : Un outil d'analyse et de visualisation pour afficher les métriques collectées par Prometheus.
12. Conteneurisation et orchestration :
13. Docker : l'API Gateway.
14. Kubernetes : Pour déployer, gérer et orchestrer du conteneurs Docker.
15. Communication :
16. API rest
17. Protocole HTTPS JSON Web Tokens (JWT)
18. Redis: cache distribué
19.

Système de gestion des utilisateurs

Activités

- Analyse des besoins utilisateurs :

    - Identification des catégories d’utilisateurs
- Conception de l’architecture des comptes utilisateurs
- Développement des interfaces utilisateurs
- Sécurité et confidentialité
- Formation et documentation
- Test et assurance qualité
- Support et maintenance

Livrables

1. Langage de programmation : Java
2. Framework : Spring Boot
3. Librairies pour la sécurité : JSON Web Token (JWT)
4. Base de données : Oracle database
5. ORM (Object-Relational Mapping) : Hibernate
6. Service Discovery : Kubernetes
7. Enregistrement et localisions d’instance instances micro-services.
8. Containerisation : Docker
9. Kubernetes
10. Automatisation du déploiement, la gestion et l'évolutivité des services
11. Système de contrôle de version GIT

Système d'inventaire

Activités

- Définition des objectifs et des exigences
- Planification du projet
- Conception du système
- Développement du logiciel
- Test du système
- Déploiement Prod
- Formation des utilisateurs
- Support et maintenance
- Documentation

Livrables

1. Langage de programmation : Java
2. Framework back-end : Spring Boot
3. Base de données : Oracle

4. ORM (Object Relational Mapping) : Hibernate
5. Gestion des dépendances : Maven (gestion des bibliotèques et des modules)
6. Intégration continue: Jenkins
7. Kubernetes (déployer et gérer)
8. Conteneurisation : Docker
9. Contrôle de version : Git

Système de commande

Activités

- Définition des objectifs et des exigences
- Planification du projet
- Conception du système
- Développement du logiciel
- Test du système
- Déploiement Prod
- Formation des utilisateurs
- Support et maintenance
- Documentation

Livrables

1. Langage de programmation : Java
2. Framework back-end : Spring Boot
3. Base de données : Oracle
4. ORM (Object Relational Mapping) : Hibernate
5. Gestion des dépendances : Maven (gestion des bibliotèques et des modules)
6. Intégration continue/déploiement continu (CI/CD) : Jenkins
7. Hébergement : Kubernetes (déployer et gérer)
8. Conteneurisation : Docker
9. Contrôle de version : Git

Système de mail

Activités

- Définition des objectifs et des exigences
- Planification du projet
- Conception du système
- Développement du logiciel
- Test du système

- Déploiement Prod
- Formation des utilisateurs
- Support et maintenance
- Documentation

Livrables

1. Langage de programmation : Java
2. Framework back-end : Spring Boot
3. Bibliothèque pour l'envoi de courriels : JavaMail API avec Spring Mail
4. Serveur de messagerie : SMTP (protocoles de serveur de messagerie)
5. Gestion des dépendances : Maven
6. Kubernetes (pour déployer et gérer)
7. Conteneurisation : Docker
8. Contrôle de version : Git

Système d'analyse de données

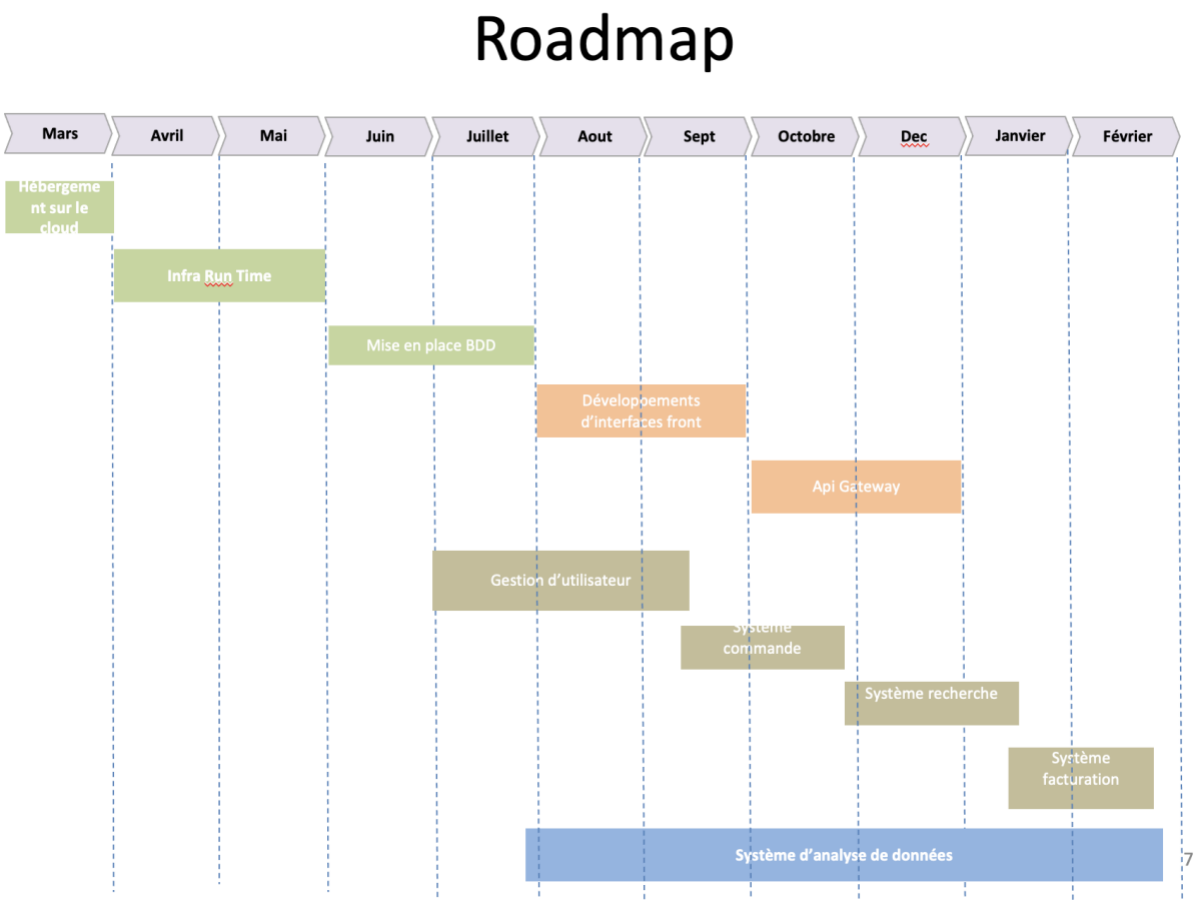
Activités

- Définition des objectifs et des exigences
- Planification du projet
- Conception du système
- Développement du logiciel
- Test du système
- Déploiement Prod
- Formation des utilisateurs
- Support et maintenance
- Documentation

Livrables

1. Langage de programmation : Java
2. Framework back-end : Spring Boot
3. Collecte de données : Elasticsearch et Kibana (ELK Stack)
    - Utilisez des outils de collecte de données et de monitoring pour suivre les habitudes des utilisateurs et les performances des microservices.
4. Base de données : Elasticsearch.
5. Gestion des dépendances : Maven (Java)
6. Hébergement : Kubernetes
7. Conteneurisation : Docker

Plan et calendrier du projet



Risques et facteurs de réduction

Analyse des risques

ID	Risque	Gravité	Probabilité	Facteur de réduction	Propriétaire
1	Mauvaise configuration de l'infrastructure runtime	3	2	Documentation, revue de pairs	Ingénieur DevOps
1	Manque de ressources pour	2	2	Planification des ressources(plan	Ingénieur DevOps

	l'infrastructure runtime			de charge)	
1	Erreurs humaines lors de la configuration	3	2	Formation, vérification des pairs	Ingénieur DevOps
2	Indisponibilité du service cloud	4	2	Choix d'un fournisseur fiable	Ingénieur DevOps
2	Coûts élevés de l'hébergement cloud	2	3	Surveillance des coûts, optimisation	Ingénieur DevOps
2	Violation de la conformité réglementaire	4	2	Veille réglementaire, audits	Responsable conformité
3	Problèmes de performance de la base de données	3	3	Surveillance, optimisation	DBA
3	Sécurité insuffisante de la base de données	4	3	Audit de sécurité, chiffrement	DBA
3	Perte de données due à des défaillances matérielles	4	1	Sauvegardes régulières, RAID	DBA /SYS
4	Incompatibilité entre front-end et back-end	3	2	Communication, tests d'intégration	Dev. front-end

4	Bugs et erreurs d'affichage sur l'interface front	2	3	Tests, revue de code	Dev. front-end
4	Épuisement professionnel des développeurs front-end	3	2	Gestion du temps, soutien de l'équipe	Chef de projet
5	Inefficacité de l'interface de recherche	2	2	Tests utilisateurs, UX design	Dev. back-end
5	Mauvaise intégration avec Elasticsearch	3	2	Tests, documentation	Dev. back-end / SYS
5	Délais de livraison non respectés	3	3	Planification, suivi de projet	Chef de projet
6	Erreurs dans l'interface de commande	4	2	Tests utilisateurs, UX design	Dev. back-end
6	Problèmes d'intégration avec le système de paiement	4	2	Tests, collaboration avec le fournisseur	Dev. back-end
6	Fraudes et chargebacks	3	2	Mesures de securité, suivi des transactions	Dev. back-end
7	Incohérences de données dans	3	2	Validation des données, tests	Dev. back-end

	l'interface d'inventaire				
7	Mauvaise gestion des stocks	3	3	Surveillance, alertes	Dev. back-end
7	Erreurs humaines dans la saisie d'inventaire	2	3	Formation, vérification des pairs	Dev. back-end
8	Difficultés d'orchestration des services	3	3	Automatisation, monitoring	Ingénieur DevOps
8	Mauvaise gestion des ressources	2	3	Surveillance, optimisation	Ingénieur DevOps
8	Communication insuffisante entre les équipes	2	3	Réunions régulières, outils de communication	Chef de projet
9	Pannes matérielles affectant le projet	3	2	Matériel de secours, maintenance préventive	Responsable IT
10	Turnover du personnel affectant le projet / Ramp up	3	2	Gestion des ressources humaines, formation	Chef de projet

Hypothèses

Le tableau ci-dessous résume les hypothèses pour cette Déclaration de travail d’architecture :

ID	Hypothèse	Impact	Propriétaire
1	L'équipe de développement a les compétences nécessaires pour la mise en œuvre des technologies proposées.	Si ce n'est pas le cas, cela pourrait entraîner des retards ou une augmentation des coûts due à la nécessité de formation ou de recrutement.	Chef de projet
2	Les services d'hébergement cloud seront disponibles avec une fiabilité et une performance suffisante. /taux de disponibilité (SLA) 99%.	Si ce n'est pas le cas, cela pourrait affecter la disponibilité et la performance du système.	Ingénieur DevOps
3	Les utilisateurs interagiront avec le système de la manière prévue.	Si ce n'est pas le cas, cela pourrait nécessiter des ajustements dans le design du système ou dans la formation des utilisateurs.	UX Designer
4	Les réglementations et exigences en matière de sécurité seront respectées tout au long du projet. (Secure coding, RGPD).	Si ce n'est pas le cas, cela pourrait entraîner des problèmes de conformité ou de sécurité.	Responsable IT

Critères d’acceptation et procédures

Métriques et KPIs

De plus, les métriques suivantes seront utilisées pour déterminer le succès de ce travail d’architecture :

Métrique	Technique de mesure	Valeur cible	Justification	Notes supplémentaires

Nombre de bugs par release	Suivi des bugs dans un outil de gestion de bugs/JIRA	< 5	Un nombre faible de bugs indique une bonne qualité du code	Les bugs critiques doivent être traités en priorité
Temps de réponse moyen des services	Surveillance des temps de réponse à l'aide d'outils de surveillance du réseau	< 200 ms	Une réponse rapide améliore l'expérience utilisateur	Ceci est une moyenne, certains services peuvent avoir des temps de réponse plus longs
Disponibilité du système	Surveillance de la disponibilité du système avec des outils de surveillance du réseau	99.9%	Une haute disponibilité est essentielle pour l'expérience utilisateur et la satisfaction du client	Cela équivaut à environ 9 heures de temps d'arrêt non planifié par an
Taux de succès des transactions	Surveillance des transactions réussies et échouées	> 98%	Un taux de succès élevé indique que le système est fiable et performant	Les transactions échouées doivent être analysées pour identifier et résoudre les problèmes
Satisfaction des utilisateurs	Enquêtes de satisfaction des utilisateurs, feedback des utilisateurs	Score moyen > 4 sur 5	La satisfaction des utilisateurs est un indicateur clé de la qualité du système	Les retours négatifs doivent être analysés et abordés
Nombre d'adhésions d'utilisateurs par jour	Suivi des inscriptions d'utilisateurs dans la base de données du système	Augmentation de 10%	L'augmentation des adhésions d'utilisateurs indique une croissance de l'audience et une adoption accrue du système	Cela pourrait nécessiter des campagnes de marketing ou des améliorations de l'UX pour stimuler l'adhésion
Adhésion de	Suivi des	Passer de	L'augmentation des	Cela pourrait

producteurs alimentaires	inscriptions des producteurs dans la base de données du système	1,4/mois à 4/mois	adhésions de producteurs améliore la variété des produits disponibles et la valeur du système pour les utilisateurs	nécessiter des efforts de recrutement et de partenariat
Délai moyen de parution*	Suivi du temps entre la soumission et la publication des produits	Réduit de 3,5 semaines à moins d'une semaine	Un délai de parution plus court permet une rotation plus rapide des produits et une meilleure réactivité aux préférences des utilisateurs	Cela pourrait nécessiter des améliorations du processus ou de l'automatisation
Taux d'incidents de production P1	Suivi des incidents P1 dans un outil de gestion des incidents	Pour commencer : réduit de >25/mois à moins de 1/mois	Un faible taux d'incidents P1 indique une stabilité et une fiabilité élevées du système	Les incidents P1 doivent être traités en priorité et analysés pour éviter leur récurrence

Procédure d’acceptation

1. **Vérification des livrables** : À chaque étape importante du projet, l'équipe du projet devra présenter le livrable correspondant au propriétaire ou au responsable du projet pour vérification. Cela pourrait inclure, par exemple, une démonstration des nouvelles fonctionnalités du système, une révision du code, ou une présentation des résultats d'un cycle de tests.

2. **Feedback et corrections** : Le propriétaire ou le responsable du projet fournira un feedback sur le livrable, y compris tout problème ou préoccupation qu'il pourrait avoir. L'équipe du projet fera alors les corrections nécessaires.

3. **Validation finale** : Une fois que toutes les corrections ont été faites, le propriétaire ou le responsable du projet procédera à une validation finale du livrable. Cela pourrait impliquer une vérification finale des fonctionnalités, une évaluation de la performance du système, ou une revue de la documentation du projet.

4. **Acceptation formelle** : Si le livrable satisfait à tous les critères d'acceptation, le propriétaire ou le responsable du projet fournira une acceptation formelle du livrable. Cela pourrait être fait par le biais d'une signature électronique, d'un e-mail formel, ou d'un autre moyen convenu entre toutes les parties.

5. **Suivi des acceptations** : Toutes les acceptations seront suivies et documentées pour référence future. Cela pourrait être fait sur le dépôt github.

Approbations signées

15/06/2023