Déclaration de Travail d'Architecture

Nom du projet :	Plateforme Geo-Aware Food Sourcing
Préparé par :	HEDI DHIB
N° de version :	1.0
Date de version :	19/08/2025
Titre :	Déclaration de Travail d'Architecture
Revu par :	Vincent
Date de révision :	21/08/2025
Liste de distribution :	Architect, Dev team, CPO, CIO, CEO
De :	HEDI DHIB
Date :	19/08/2025
Email :	Hedi.dhib@gmail.com
Pour Action :	Révision
Date de rendu :	21/8/2025
Types d'action :	Approbation, Révision, Information, Classement, Action requise, Participation à une réunion, Autre (à spécifier)
Historique de versions :	Voir Github

Table des Matières :

7. Risques et facteurs de réduction
    a. Analyse des risques
    b. Hypothèses
8. Critères d'acceptation et procédures
    a. Métriques et KPIs
    b. Procédure d'acceptation
9. Approbations signées
    a. Date de signature

1. Objet de ce document

Ce document constitue la « Déclaration de travail d'architecture » pour le « Geo-Aware Food Sourcing » de Foosus .

La Déclaration de travail d'architecture définit le périmètre, les objectifs, l'approche, les rôles et les livrables liés à la mission d'architecture d'entreprise menée dans le cadre de la refonte de la plateforme.

La Déclaration de travail d'architecture constitue habituellement le document qui permet de mesurer la réussite de l'exécution du projet d'architecture et peut former la base de l'accord contractuel entre le fournisseur et le consommateur de services d'architecture. En général, toutes les informations de ce document doivent se situer à un haut niveau.

Cette démarche vise à garantir une évolution cohérente, scalable, sécurisée et responsable du système d'information, tout en soutenant la croissance de Foosus et sa capacité d'innovation rapide.

2. Déclaration de travail d'architecture

a. Requête du projet et contexte

Foosus SAS, start-up spécialisée dans la mise en relation entre consommateurs et producteurs locaux, fait face à un double défi : une **croissance rapide** de son activité et une **accumulation importante de dette technique** issue de choix technologiques initiaux non scalables.

Dans ce contexte, l'entreprise souhaite entamer une **refonte en profondeur de son système d'information**, en s'appuyant sur une approche d'architecture d'entreprise pour garantir la cohérence, la maintenabilité et l'évolutivité de la plateforme.

La solution actuelle, historiquement monolithique, a montré ses limites : fragmentation des services, absence de standardisation technique, ralentissements en production, difficulté d'intégration continue, et faible résilience face aux montées en charge. Ces contraintes freinent l'innovation produit, complexifient le déploiement de nouvelles fonctionnalités et réduisent la qualité de l'expérience utilisateur.

La requête formulée par les parties prenantes (Direction, Produit, Tech) consiste à **concevoir une nouvelle architecture cible** capable de supporter l'ambition stratégique de Foosus sur les plans suivants :

- **Extension géographique** à l'échelle nationale (et à terme européenne),
- **Scalabilité horizontale** pour absorber une montée en charge progressive jusqu'à 1 million d'utilisateurs,
- **Réduction de la dette technique** via une gouvernance architecturale structurée et un refactoring progressif,

- **Soutien à l'innovation rapide**, avec un time-to-market réduit pour les évolutions produit,
- **Haute disponibilité et sécurité by design**, en cohérence avec les standards de la foodtech et les exigences RGPD.

Cette initiative de refonte se veut aussi un **levier de transformation organisationnelle**, en alignant les pratiques d'ingénierie logicielle, de produit et de gouvernance avec les meilleures pratiques du marché, notamment via l'adoption de **microservices**, du **DevOps**, du **CI/CD**, et d'une **architecture centrée sur les APIs**.
La mission d'architecture portera donc sur :

- La **définition de l'état cible de l'architecture**,
- La **planification de la trajectoire de migration**,
- L'**accompagnement des équipes techniques et produit** dans la mise en œuvre,
- Et l'**instauration d'un cadre de gouvernance technique durable**.

b. Description du projet et périmètre

Le projet consiste en la **refonte complète de la plateforme numérique Foosus**, avec pour objectif principal de concevoir et mettre en œuvre une **architecture moderne, modulaire et évolutive**, répondant aux besoins métiers actuels et futurs.

Description du projet :
La solution cible devra reposer sur une **architecture orientée microservices**, **cloud-native**, résiliente, sécurisée, et hautement automatisée. Elle devra permettre :

- Une **expérience utilisateur unifiée** sur web et mobile (approche mobile-first),
- Une **interopérabilité native** avec des systèmes tiers (paiement, cartographie, CRM, ERP),
- Une **prise en charge de la géolocalisation en temps réel**,
- Des **déploiements continus (CI/CD)** avec faible temps d'indisponibilité,
- Une **séparation claire des domaines métier**, facilitant l'évolutivité et la maintenabilité.

Ce projet comprend également une **phase transitoire critique** durant laquelle la nouvelle architecture coexistera avec l'ancienne. Un mécanisme de migration progressive (strangling pattern, proxies, découplage via APIs) sera mis en œuvre pour assurer la continuité des opérations sans interruption de service.

Périmètre du projet :

Périmètre fonctionnel :

- Refonte des parcours utilisateurs (inscription, commande, mise en relation),
- Intégration d'un moteur de recommandation basé sur la géolocalisation,
- Back-office modulable pour producteurs, clients et équipes support,
- Portail d'administration centralisé (sécurité, rôles, statistiques, monitoring).

Périmètre technique :

- Couches front-end (web et mobile),
- Couches back-end microservices,

- API Gateway, gestion des identités (IAM),
- Infrastructure cloud (IaaS/PaaS), observabilité (logs, traces, métriques),
- Pipelines CI/CD, tests automatisés, sandbox d'intégration.

Périmètre organisationnel :

- Collaboration inter-équipes Produit, Tech, Design, Support,
- Mise en place d'un référentiel technique et de bonnes pratiques partagées,
- Mise en œuvre d'un processus d'architecture gouverné (via un Comité d'Architecture).

Hors périmètre (phase ultérieure ou dépendance externe) :

- Intégration avec les partenaires logistiques tiers (hors MVP),
- Implémentation de l'IA pour la prévision des stocks (phase R&D),
- Connecteurs CRM/ERP externes avancés (intégration post-refonte).

c. Vue d'ensemble

La refonte de la plateforme Foosus s'inscrit dans une démarche globale de **transformation numérique** visant à aligner le système d'information avec la stratégie de croissance rapide de l'entreprise, centrée sur l'alimentation locale et la mise en relation géociblée entre consommateurs et producteurs.

Cette transformation s'articule autour de trois axes majeurs :

1. Transformation Technologique

La plateforme existante sera progressivement remplacée par un **système modulaire orienté microservices**, hébergé dans une **infrastructure cloud-native**, et piloté par une **architecture orientée événements (EDA)**. L'objectif est d'assurer :

- La **scalabilité horizontale** et l'élasticité des ressources.
- La **résilience** face aux pannes.
- L'**automatisation complète du cycle de vie applicatif**, via une chaîne CI/CD robuste.

2. Transformation Organisationnelle

L'approche architecturale proposée favorisera une **plus grande autonomie des équipes** grâce à la responsabilisation autour de **domaines fonctionnels bien délimités** (approche DDD), tout en instaurant une gouvernance technique forte et documentée :

- Mise en place d'un **Comité d'Architecture** régulier.
- Adoption de standards techniques transverses (API, sécurité, qualité).
- Collaboration étroite entre les équipes produit, développement et direction.

3. Transformation Métier

Le système cible permettra de **prototyper rapidement des fonctionnalités innovantes**, d'adapter les expériences utilisateurs en fonction des contextes géographiques, et de **fournir une plateforme de services numériques réutilisables** (moteur de recherche, modules de paiement, gestion de la relation client, etc.).

Cette refonte représente une **opportunité de remettre à plat les fondations techniques** tout en accompagnant l'évolution des modèles économiques de Foosus (B2C, B2B2C,

marketplace, etc.), et en offrant une base pérenne pour l'expansion nationale puis européenne.

d. Alignement stratégique

La Déclaration de Travail d'Architecture s'inscrit dans une logique d'**alignement étroit entre les objectifs d'entreprise de Foosus** et les transformations technologiques à mettre en œuvre. L'architecture cible ne doit pas uniquement répondre à des critères techniques, mais être un **levier de création de valeur business**, en cohérence avec la stratégie de croissance de l'entreprise.

Objectifs stratégiques de Foosus

Objectif stratégique	Impact attendu	Contribution de l'architecture
Accélération de la croissance utilisateur	+10% par jour et 1M utilisateurs visés à 18 mois	Mise à l'échelle horizontale via microservices, services découplés, tolérance à la charge
Réduction de la dette technique	Stabilité, maintenabilité, amélioration du TTM (Time to Market)	Refonte modulaire, standardisation, documentation, gestion centralisée des dépendances
Innovation produit continue	Capacité à tester et déployer de nouvelles fonctionnalités rapidement	Intégration CI/CD, architecture hexagonale, séparation stricte des couches métier/technique
Expérience utilisateur géo-personnalisée	Fidélisation, engagement utilisateur, valeur perçue	Architecture API-first, intégration de moteurs de recommandation, optimisation des performances
Conformité, sécurité et souveraineté des données	Alignement RGPD, auditabilité, image de marque	Gouvernance des données, chiffrement, IAM (gestion des identités et des accès), traçabilité

Enjeux transverses :

- **Soutenir les campagnes marketing** sans risque de surcharge de la plateforme,
- **Ouvrir l'écosystème Foosus** à des partenaires via des APIs sécurisées (place de marché, logistique, analytique),
- **Permettre la prise de décision pilotée par la donnée**, en intégrant des couches d'observabilité et de reporting dès la conception.

L'architecture proposée ne se contente pas de soutenir la stratégie de Foosus : elle en devient un **accélérateur** et un **facteur de différenciation sur le marché**.

3. Objectifs et périmètre

a. Objectifs

Les objectifs de ce travail d'architecture sont à la fois **stratégiques, techniques et organisationnels**. Ils visent à garantir que la future plateforme Foosus soit un levier de croissance, d'innovation et de différenciation sur un marché en forte mutation.
L'architecture définie devra répondre aux objectifs business suivants :

Objectif Business	Notes
Favoriser la mise en relation locale entre producteurs et consommateurs	En exploitant la **géolocalisation**, les **distances** et des mécanismes de recherche contextuelle pour renforcer la proximité
Améliorer la fiabilité et la performance de la plateforme	Grâce à une infrastructure résiliente, à la tolérance aux pannes, à l'observabilité native et à la scalabilité automatique Pour soutenir efficacement les **campagnes marketing**, absorber les **pics de charge** (événements, saisonnalité, promotions)
Permettre une expérimentation produit rapide	Via une **architecture modulaire**, des environnements isolés (feature toggles, dark launches) et des **déploiements continus sans interruption de service**
Réduire la dette technique	En instaurant des **standards de développement**, une documentation vivante, des référentiels de bonnes pratiques et un **cadre de gouvernance** technique clair

Synthèse :
Ces objectifs ne sont pas cloisonnés : ils sont **interdépendants** et structurent le socle du futur système d'information. Leur atteinte permettra à Foosus de :

- Mieux servir les utilisateurs finaux,
- Outiller ses équipes produit & tech pour innover en confiance,
- Offrir une plateforme plus fiable, maintenable et extensible dans le temps.

L'architecture devra donc être pensée comme un **vecteur de transformation continue**, en assurant l'agilité nécessaire à une start-up en forte croissance, tout en apportant la robustesse nécessaire à une montée en charge progressive.

b. Périmètre

Le périmètre du travail d'architecture couvre un ensemble d'éléments techniques, fonctionnels et organisationnels nécessaires à la transformation de la plateforme Foosus. Il est structuré selon trois axes principaux : technique, organisationnel et temporel.

1. Périmètre technique
Le travail d'architecture englobe :

- La **couche back-end** : découplage du monolithe existant vers des microservices métiers indépendants, exposés via des APIs REST et/ou événementielles.
- La **couche front-end** : refonte en approche mobile-first, SPA (Single Page Application) ou micro-frontends si pertinent.
- Les **APIs internes et publiques** : normalisation des contrats d'échange (OpenAPI), sécurisation, documentation.
- La **gestion de la base de données** : choix des moteurs, partitionnement logique, stratégie de migration des données existantes.
- La **chaîne CI/CD** : pipelines automatisés, stratégie de tests, rollback et canary release.
- L'**infrastructure cloud-native** : conteneurisation (Docker), orchestration (Kubernetes ou équivalent), sécurité, observabilité (logs, métriques, traces).

- La **gouvernance des identités** : gestion des accès utilisateurs, SSO, OAuth2, RBAC.

2. Périmètre organisationnel
L'approche d'architecture impliquera les équipes suivantes :

- **Équipe Produit** : pour alignement des besoins fonctionnels, priorisation et suivi des cas d'usage.
- **Équipe Technique (dev + ops)** : pour co-construction des solutions, choix technologiques, pilotage des POCs.
- **Équipe Design (UX/UI)** : pour l'intégration des contraintes UX dans l'architecture (performance, accessibilité).
- **Direction & parties prenantes métiers** : pour validation stratégique et arbitrage du périmètre.

La gouvernance inclura également :

- Un **référentiel technique commun** (conventions de code, bonnes pratiques),
- Des **rituels de validation d'architecture** (ex : Architecture Decision Records),
- Une **documentation continue** pour faciliter l'onboarding et la maintenabilité.

3. Périmètre temporel
Le projet est organisé en **plusieurs phases itératives**. Le présent travail d'architecture couvre en particulier la **Phase 1**, d'une durée estimée à **deux trimestres**, avec pour livrables :

- L'état de l'existant (as-is),
- L'état cible (to-be),
- La feuille de route de transformation,
- Les premières briques de mise en œuvre (MVP),
- La stratégie de cohabitation avec le legacy.

4. Hors périmètre
Les éléments suivants ne sont pas inclus dans cette phase, sauf en tant que dépendances identifiées :

- L'intégration complète avec des systèmes CRM/ERP (connecteurs en phase ultérieure),
- La mise en œuvre d'algorithmes d'IA/ML (prévision, recommandations),
- Le remplacement total de la solution legacy (sera traité par étapes),
- Les développements métier très spécifiques hors MVP.

c. Parties prenantes, préoccupations, et visions

Un projet d'architecture d'entreprise implique une diversité d'acteurs, chacun porteur de **préoccupations spécifiques** liées à son rôle dans l'organisation. Le rôle de l'architecture est d'**écouter ces attentes, les documenter**, puis d'y répondre à travers des **visions techniques concrètes**, des artefacts partagés, et des pratiques de gouvernance.

Synthèse des parties prenantes et de leurs préoccupations

Partie prenante	Préoccupations	Vision architecturale apportée
CIO (Directeur des Systèmes d'Information)	Sécurité, conformité RGPD, scalabilité, pérennité	Architecture cible documentée, principes directeurs, stratégie cloud et sécurité, gouvernance centralisée
CPO (Chief Product Officer)	Innovation rapide, time-to-market, alignement UX/business	Architecture modulaire, découplée, permettant l'expérimentation continue, feature toggles, déploiements fréquents
Dev Team (Tech Leads, Développeurs, Ops)	Clarté des responsabilités, standards, documentation technique, dette technique	Règles de développement, référentiel de bonnes pratiques, documentation centralisée, support à l'outillage CI/CD
Design Team (UX/UI)	Performance perçue, cohérence des parcours, accessibilité	Intégration des contraintes UX dès la modélisation, compatibilité mobile-first, API performantes
Équipe Business / Marketing	KPIs, stabilité, support aux campagnes, compréhension des flux utilisateurs	Schémas de flux, SLA, visualisation de parcours, traçabilité, monitoring temps réel
Support client & opérations	Outils back-office, supervision des incidents, gestion des utilisateurs	Interface d'administration centralisée, observabilité native, plan de continuité, rôles et permissions (RBAC)
Direction Générale	Alignement stratégique, maîtrise des risques, visibilité sur l'avancement	Feuille de route architecture, gestion des risques, indicateurs de pilotage, planning de transformation

Vision architecturale partagée :
L'objectif est de construire une vision **commune et compréhensible par l'ensemble des parties prenantes**, en s'appuyant sur des outils tels que :

- Diagrammes de flux,
- Architecture cible commentée,
- Matrices de décisions (ADR),
- Démonstrateurs (Proof of Concepts),
- Documentation vivante (wiki, GitHub, Notion).

Cela permettra d'**aligner les attentes métiers et les choix techniques** tout au long du cycle de transformation, et de garantir une **cohérence durable** de l'architecture dans le temps.

d. Approche managériale

La transformation de l'architecture de la plateforme Foosus nécessite une **approche managériale structurée, collaborative et itérative**, afin d'assurer la cohérence des décisions, l'implication des équipes, la gestion des risques et le respect des délais.
Cette approche s'appuie sur quatre piliers fondamentaux :

1. Gouvernance de l'architecture :
Une **cellule d'architecture transverse** sera constituée dès l'amorçage du projet. Elle regroupera :

- L'**Architecte Logiciel Principal** (promoteur de l'architecture),
- Des représentants de l'équipe produit, technique et sécurité,
- Un sponsor stratégique (CIO/CPO).

Ses rôles :

- Piloter les travaux d'architecture (vision, documents, référentiels),
- Arbitrer les décisions techniques structurantes,
- Valider les propositions issues des équipes (patterns, choix technos),
- Maintenir une **documentation d'architecture vivante et traçable**.

Un **Comité d'Architecture mensuel** assurera la validation des évolutions majeures et la résolution des points bloquants.

2. Collaboration inter-équipes
L'architecture ne sera pas conçue « en silo » mais dans une **logique de collaboration étroite avec les équipes de delivery** :

- **Ateliers multi-profils** à chaque phase clé (vision, conception, POC),
- **Partage des décisions via ADRs (Architecture Decision Records)**,
- **Pair design / pair review** pour les composants critiques,
- Utilisation d'un **espace partagé (Git, Notion, ou autre)** pour la transparence documentaire.

Cette approche permet d'**accroître l'appropriation des choix d'architecture** par les développeurs et de faciliter leur implémentation continue.

3. Méthodologie projet & pilotage agile
Le pilotage du projet suit une **méthodologie Agile Scrum**, avec :

- Des **sprints de 2 à 3 semaines** incluant des travaux d'architecture,
- L'intégration de tâches d'architecture dans le backlog produit (via une étiquette dédiée),
- Des **rituels synchrones** : daily, sprint reviews, démos techniques, post-mortem en cas d'échec d'une initiative technique,
- Des **OKR d'équipe alignés avec les objectifs d'architecture** (résilience, testabilité, modularité).

L'architecte agit en **facilitateur transverse**, garant de la cohérence entre les domaines fonctionnels, la roadmap produit et la réalité des capacités techniques.

4. Suivi, traçabilité et capitalisation
Pour garantir la qualité et la transparence, l'approche managériale intègre :

- Un **système de suivi des décisions techniques**, versionné et révisable,
- Des **indicateurs de pilotage** : avancement des livrables d'architecture, couverture documentaire, adoption des patterns,
- Une **stratégie de montée en compétence** (ateliers, tech talks, mentoring) autour des principes d'architecture adoptés.

Cette approche managériale vise à créer un **écosystème propice à la transformation durable**, en s'assurant que l'architecture serve les objectifs business tout en renforçant la maturité technique de l'organisation.

e. Procédures de changement de périmètre

Dans tout projet d'architecture, il est essentiel d'anticiper la possibilité de changements de périmètre liés à :

- L'évolution des priorités business ou réglementaires,
- La découverte de nouvelles contraintes techniques,
- Des arbitrages organisationnels ou budgétaires.

Le cadre de gestion du périmètre mis en place sur le projet Foosus repose sur les principes de **traçabilité, agilité maîtrisée et gouvernance partagée**.

1. Identification et typologie des changements

Les changements de périmètre peuvent concerner :

- **Le périmètre fonctionnel** (ajout/suppression de cas d'usage, changement d'objectifs),
- **Le périmètre technique** (changement d'approche techno, révision d'un pattern architectural),
- **Le périmètre organisationnel** (évolution des responsabilités, des ressources ou des dépendances inter-équipes),
- **Le périmètre temporel** (glissement de planning, réorganisation des phases, modification du séquencement).

2. Procédure de gestion des changements

Un processus léger mais formalisé sera appliqué :

Étape	Description	Responsable
1. Détection	Toute partie prenante peut signaler un changement de périmètre via GitHub, Notion, ou réunion projet	Tous
2. Qualification	Analyse de l'impact (coût, délai, scope, valeur), catégorisation critique/majeur/mineur	Architecte + PO
3. Recommandation	Proposition de scénario de traitement (acceptation, rejet, escalade, itération)	Architecte
4. Validation	Arbitrage par le **Comité de Pilotage** ou **Comité d'Architecture** selon l'impact	CIO / CPO / Sponsor
5. Documentation	Enregistrement du changement approuvé dans le registre de décisions techniques (ADR, changelog, roadmap)	Architecte
6. Communication	Notification à toutes les équipes impactées + mise à jour des livrables si nécessaire	Scrum Master Chef de projet

3. Outils supports

- **Notion / Confluence** : formulaire de changement ou backlog "Architecture Requests" dédié.

- **GitHub / GitLab** : tickets techniques liés au périmètre (étiquettes "scope-change").
- **Fichiers ADR (Architecture Decision Records)** : pour tracer tout changement technique significatif.
- **Revue régulière** en comité pour synchroniser les décisions sur la roadmap produit.

4. Principes directeurs

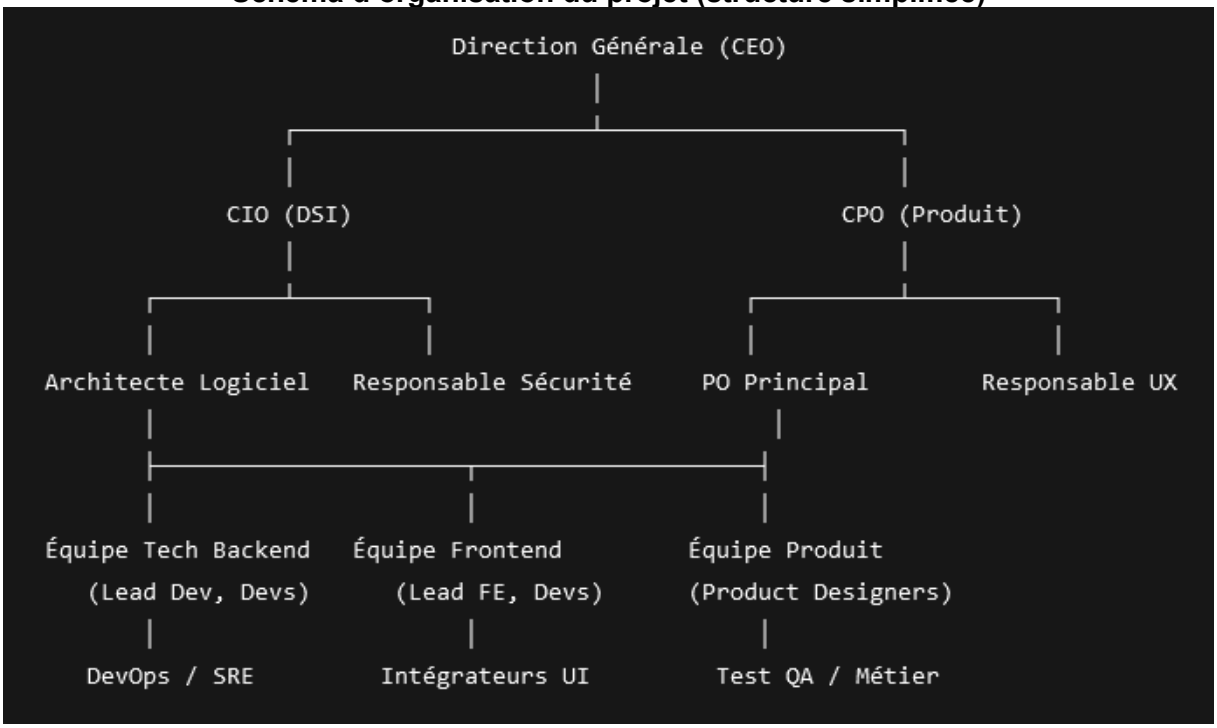
- **Pas de changement silencieux** : tout changement impactant le périmètre doit être formalisé, même mineur.
- **Équilibre valeur / coût** : chaque changement est évalué en fonction de sa valeur business et de son coût d'adaptation.
- **Agilité encadrée** : la procédure est conçue pour ne pas freiner l'innovation, tout en évitant les dérives incontrôlées.
- **Transparence** : tous les changements approuvés sont visibles, historisés et accessibles à toutes les parties prenantes.

4. Rôles et responsabilités

a. Structure de gouvernance

La gouvernance du projet d'architecture repose sur une **structure hybride, transverse et collaborative**, conçue pour garantir la cohérence des décisions tout en favorisant l'agilité des équipes. Elle articule les responsabilités stratégiques, tactiques et opérationnelles autour d'un **pilotage architectural centralisé**, tout en maintenant une forte proximité avec les équipes de delivery.

Schéma d'organisation du projet (structure simplifiée)



Rôles clés et articulation :

Rôle	Responsabilités principales	Lien hiérarchique / fonctionnel
Architecte Logiciel	Porte la vision d'architecture, formalise les états cibles, arbitre les choix technos, pilote les décisions structurantes	Fonctionnel avec CIO, CPO, équipes techniques
CIO (Directeur SI)	Garant de la stratégie IT, du budget, de la sécurité, et de la scalabilité	Hiérarchique sur l'architecte et les équipes techniques
CPO (Chief Product Officer)	Porte la stratégie produit, priorise les besoins métiers, pilote les roadmaps	Fonctionnel sur le backlog architecture / produit
PO Principal	Interface produit/technique, collecte des besoins, rédige les user stories et alimente le backlog	Coordination avec le CPO et l'architecte
Responsable Sécurité	Garantit la conformité RGPD, la sécurité applicative et des données, et les accès IAM	Fonctionnel sur toute l'architecture technique
Responsable UX / UI	Pilote l'expérience utilisateur, la cohérence graphique et l'accessibilité	Transverse avec PO et équipes frontend
Leads Dev / Tech Leads	Supervisent la mise en œuvre technique, challengent et appliquent les décisions d'architecture, pilotent les chantiers	Fonctionnel avec l'architecte et les équipes de dev
DevOps / SRE	Gèrent les pipelines CI/CD, l'infrastructure cloud, l'automatisation, le monitoring et la résilience	En support transverse à tous les streams
QA / Test	Définissent les critères d'acceptation techniques, valident la robustesse des livrables	Coordination avec PO, Devs, et SRE

Gouvernance des décisions :

- **Comité d'Architecture** (bi-mensuel) : Architecte, CIO, PO, Lead Dev, Responsable sécurité.
- **Comité de Pilotage stratégique** (mensuel) : CIO, CPO, CEO.
- **Revue d'architecture continue** : via pull requests, ADRs et ateliers avec les développeurs.

Cette organisation permet de **concilier la rigueur des décisions d'architecture** avec la **souplesse du delivery agile**, tout en assurant une **clarté des rôles** et une **répartition équilibrée des responsabilités**.

b. Process du projet

Le projet de refonte architecturale de la plateforme Foosus est structuré autour d'un **cadre de pilotage agile, rigoureux et collaboratif**, intégrant les meilleures pratiques en matière de gestion de projet, de gouvernance technique et de contrôle qualité.
L'objectif est de **sécuriser la mise en œuvre progressive de l'architecture cible**, tout en assurant une coordination fluide entre les différentes équipes impliquées (Produit, Tech, Sécurité, Direction).

Type de réunion	Périodicité	Objectifs	Participants
Daily Stand-up	Quotidien	Synchronisation des équipes, gestion des blocages	Équipes tech & produit
Sprint Planning	Toutes les 2 à 3 semaines	Planification des tâches, alignement backlog / roadmap	PO, Devs, Architecte
Sprint Review & Démo	Fin de sprint	Démonstration des livrables, retours métier, validation	PO, CPO, Architecte, stakeholders
Architecture Review Board (ARB)	Bi-mensuel	Arbitrage des décisions structurantes, revue des propositions, suivi roadmap archi	Architecte, CIO, PO, Lead Dev, Sécurité
Comité de Pilotage stratégique	Mensuel	Suivi des KPIs, avancement, validation des jalons, arbitrages de périmètre	CEO, CIO, CPO, Architecte
Post-Mortem / RCA	À la suite d'un incident critique	Analyse des causes racines, actions correctives	Tech, SRE, QA, Architecte

2. Référentiel documentaire

- **Outils utilisés** :
  - *Notion* ou *Confluence* : documentation métier, vision produit, backlog architectural
  - *GitHub/GitLab* : tickets, documentation technique, ADRs (Architecture Decision Records)
  - *Miro* / *FigJam* : schémas d'architecture, parcours UX
  - *Drive partagé / Git (versionné)* : livrables, documents TOGAF, artefacts officiels
- **Typologie de documents à maintenir** :
  - Vision d'architecture (as-is / to-be)
  - Roadmap technique et feuille de route de migration
  - Standards de développement et d'intégration
  - Contrats d'architecture (avec parties prenantes)
  - Journal des décisions (ADRs)
  - Suivi des risques et hypothèses
  - Registre des changements de périmètre

3. Management de la configuration

- Utilisation de **Git** pour le versionnement de tous les artefacts d'architecture (modèles, contrats, documents, diagrammes).
- Mise en place d'une **branche "architecture"** dans le dépôt, avec politique de **pull request + review obligatoire** pour chaque contribution.
- Suivi des versions des livrables via un **changelog structuré** (versioning sémantique : MAJEUR.MINEUR.PATCH).
- Historique complet des décisions techniques via **ADRs horodatés**.

4. Assurance qualité

- **Critères d'acceptation définis en amont** pour chaque livrable d'architecture.
- **Relecture croisée des documents** par les membres du Comité d'Architecture.
- **Validation formelle des livrables majeurs** (état cible, normes, interfaces critiques) par le CIO/CPO.
- Intégration d'**indicateurs qualité** dans la documentation : couverture, niveau de formalisation, taux d'adoption.

5. Procédure en cas d'escalade

Type d'escalade	Déclencheur	Canal	Responsable
Opérationnelle	Blocage technique ou dépendance bloquante	Slack #ops-escalade + réunion ad hoc	Lead Dev / PO
Stratégique	Décision d'impact majeur (budget, scope, roadmap)	Mail formel + Comité de pilotage	Architecte / CIO
Conflit décisionnel	Désaccord sur une approche ou une norme	Revue ARB + fiche ADR avec options	Architecte + CPO/CIO

6. Procédure de changement

Les changements sont encadrés par le processus défini dans la section **3.e.** (identification, qualification, validation, documentation, communication), avec usage systématique de tickets et ADRs.

Cette organisation garantit une **exécution contrôlée, transparente et réactive du projet**, capable de s'adapter aux aléas tout en préservant l'intégrité de la vision d'architecture définie.

c. Rôles et responsabilités (RACI)

Cette matrice RACI définit les responsabilités de chaque acteur impliqué dans les activités clés du travail d'architecture. Elle permet de **clarifier les rôles**, **fluidifier les interactions** entre équipes, et **éviter les zones de flou décisionnel**.

Légende :
R = Responsable (réalise l'activité)
A = Approbateur (valide formellement)
C = Consulté (donne un avis)
I = Informé (tient à jour, reçoit les décisions)

Matrice RACI – Activités d'architecture Foosus

Activité	Responsable (R)	Approbateur (A)	Consulté (C)	Informé (I)
Vision d'architecture	Architecte	CIO	Dev Team	CPO
Documentation technique	Architecte	Lead Dev	Dev Team	PO
Alignement business	Architecte	CPO	CIO	CEO
Définition de l'état cible	Architecte	CIO / CPO	Lead Dev, PO	Dev Team

Activité	Responsable (R)	Approbateur (A)	Consulté (C)	Informé (I)
Gouvernance de sécurité	Responsable sécurité	CIO	Architecte	Dev Team
Réalisation de la roadmap technique	Architecte	CIO	PO, Lead Dev	Dev Team
Standardisation et bonnes pratiques	Architecte	Lead Dev	Dev Team	PO
Validation des choix technologiques	Architecte	CIO	Lead Dev	PO
Décisions de migration technique	Architecte	CIO	DevOps, Lead Dev	PO
Suivi des risques architecture	Architecte	CIO	CPO	CEO
Gestion des changements de périmètre	PO / Architecte	CPO / CIO	Lead Dev, CEO	Dev Team
Publication des ADR (Architecture Decision Records)	Architecte	-	Lead Dev, PO	Toute l'équipe
Mise à jour des documents TOGAF	Architecte	CIO	PO	Comité d'architecture
Revue architecture (ARB)	Architecte	CIO / CPO	Lead Dev, Sécurité	CEO
Formation & onboarding architecture	Architecte	-	Lead Dev	PO, Dev Team

Cette structuration RACI permet d'ancrer la **responsabilité de la vision chez l'architecte**, tout en assurant un **partage continu avec les équipes techniques, produit et métier**. Elle facilite aussi la remontée des alertes, la co-construction, et la validation hiérarchique des décisions critiques.

5. Approche architecturale

a. Process d'architecture

Le projet de refonte de la plateforme Foosus s'appuie sur le cadre TOGAF ADM (Architecture Development Method), reconnu pour sa capacité à **structurer la conception, la planification, la mise en œuvre et la gouvernance d'une architecture d'entreprise**.

Application du cycle ADM au projet Foosus :
L'intégralité du cycle ADM a été analysée. Certaines phases seront pleinement activées, d'autres adaptées au contexte de Foosus, selon une approche **pragmatique, itérative et orientée valeur**.

Phase	Entrée / Sortie	Application au projet Foosus
Préliminaire	Contexte, objectifs, contraintes, parties prenantes	Réalisée partiellement : nécessite finalisation de la charte d'architecture, définition du scope initial et des outils de gouvernance
A — Vision de l'architecture	État initial + vision cible	Déjà amorcée. À consolider avec les priorités stratégiques (scalabilité, modularité, sécurité, time-to-market)
B — Architecture business	Besoins métiers, use cases, processus	À formaliser en lien avec le Product Owner : cartographie des processus cibles, matrices objectifs-capacités
C — Architecture des SI	Modèles d'information et d'applications	À définir : découpage en microservices, APIs, mapping des entités métier. Couvre à la fois l'état de référence et la cible
D — Architecture technologique	Infrastructures, performances, sécurité, outils	Axée cloud-native, automatisation CI/CD, observabilité, sécurité Zero Trust. À spécifier pour chaque domaine technologique
E — Opportunités et solutions	Modèles de solutions, quick wins	Conçue en mode itératif. Identification des MVPs et prototypes à fort impact. Intégration d'approches "Proof of Value"
F — Planification de migration	Séquençage, cohabitation legacy / cible	À construire en collaboration avec les équipes techniques : découpage par lot, stratégie Strangler Fig, pilotage par domaine fonctionnel
G — Gouvernance de l'implémentation	Cadre de suivi et de conformité	À établir via le Comité d'Architecture : validation des livrables, suivi des écarts, mise en conformité des implémentations
H — Management du changement	Plans de communication et d'adhésion	Sensibilisation des équipes, ateliers de montée en compétence, implication progressive des leads. Gestion active des freins
Management des exigences *(en continu)*	Consolidation et traçabilité des besoins	Piloté via backlog partagé entre produit et architecture. Intégration dans les user stories techniques et documentation vivante

Notes supplémentaires sur l'approche itérative :

- L'ensemble du cycle ADM sera **piloté de manière incrémentale**, avec des **releases d'architecture successives** alignées sur la roadmap produit.
- Les phases **E à G** seront **revues à chaque itération majeure** (release trimestrielle ou MVP critique).
- Des **revues intermédiaires d'architecture (Architecture Review Sessions)** auront lieu à chaque fin de sprint technique comportant des changements structurants.
- La gestion des artefacts ADM s'appuiera sur des outils de collaboration versionnés (Git, Notion, Confluence) pour assurer **la traçabilité, la collaboration et l'auditabilité**.

b. Contenu de l'architecture

Le cadre TOGAF ACF (Architecture Content Framework) permet de **structurer les artefacts d'architecture** selon une classification cohérente, facilitant la documentation, le partage et la gouvernance. Tous les artefacts TOGAF ne sont pas nécessaires à chaque projet ; seule une sélection ciblée est ici activée.

Le tableau ci-dessous précise les **zones de contenu pertinentes pour le projet Foosus**, ainsi que les **sous-catégories concernées** et leur rôle dans la réponse aux préoccupations des parties prenantes.

Vue d'ensemble du contenu d'architecture (ACF)

Zone de contenu	Entrée / Sortie	Notes / Sous-catégories couvertes
1. Principes, Vision, Conditions requises	Vision d'architecture, objectifs, exigences réglementaires, contraintes	- Principes directeurs d'architecture (scalabilité, sécurité, modularité) - Catalogue des exigences non-fonctionnelles (disponibilité, résilience, temps de réponse) - Cartographie des préoccupations des parties prenantes
2. Architecture Business	Processus métier, cas d'usage, capacités attendues	- Cartographie des processus cibles (onboarding, commande, mise en relation) - Use Cases métiers illustrés - Matrice objectifs / capacités - Modèle de services
3. Architecture des SI — Données	Entités métier, dictionnaire de données, règles de gestion	- Modèle conceptuel de données (MCD) - Stratégie de gouvernance des données (qualité, accessibilité, traçabilité) - Plan de migration des données existantes
4. Architecture des SI — Applications	Applications, microservices, APIs, échanges inter-systèmes	- Cartographie applicative cible (microservices, composants back/front) - Design des APIs (OpenAPI, contrats) - Catalogue des services applicatifs exposés - Modèle de découpage par domaine fonctionnel (DDD)
5. Architecture Technologique	Infrastructure, sécurité, performances	- Infrastructure cible cloud-native (IaaS/PaaS) - Patrons d'infrastructure (CI/CD, haute dispo, observabilité) - Plan de gestion des identités et des accès (IAM, RBAC, SSO) - Diagrammes d'architecture physique
6. Réalisation de l'architecture	Plans, roadmaps, backlog technique, ADRs	- Feuille de route d'implémentation (par vague ou domaine) - Planning de migration progressif (Strangler Pattern) - Registre des décisions techniques (Architecture Decision Records) - Mécanisme de suivi de la conformité des développements

Alignement avec les préoccupations des parties prenantes

- **CIO** : visualisation des architectures cible (cloud, sécurité), garantie de conformité RGPD, cycle de validation des choix.
- **CPO** : cartographie des processus et capacités métiers, logique modulaire et réutilisable.
- **Tech Leads & Dev** : référentiels techniques accessibles, documentation des APIs, standards partagés.

- **UX / Produit** : prise en compte des flux utilisateurs, performance perçue, cohérence fonctionnelle.
- **Direction** : roadmap lisible, liens clairs entre architecture et stratégie.

Formats et outils de production des artefacts

- **MCD / MLD / Cas d'usage** : structurés sous format UML, MerMEID, ou DBML
- **Diagrammes techniques** : C4 Model, ArchiMate, diagrammes PlantUML
- **Contrats APIs** : OpenAPI 3.x, versionnés dans Git
- **Documents TOGAF** : en Markdown (docs-as-code), versionnés avec Git + changelog
- **Visualisation dynamique** : Miro / Lucidchart / Structurizr

c. Méthodologies pertinentes et normes de l'industrie

L'approche architecturale retenue pour le projet Foosus s'appuie sur un **ensemble de standards reconnus** du domaine de l'architecture d'entreprise, combinés à des **méthodes modernes issues de l'ingénierie logicielle agile et du DevOps**.

L'objectif est de garantir un équilibre entre **rigueur méthodologique**, **adaptabilité aux contraintes de terrain**, et **alignement stratégique avec les besoins business**.

Méthodologies et cadres utilisés

Cadre / Norme	Utilisation dans le projet
TOGAF ADM & ACF	Cadre de structuration des phases, artefacts et rôles d'architecture
Architecture Modulaire orientée Domaines (DDD)	Structuration des microservices, découpage logique, autonomie des équipes
Architecture centrée événements (EDA)	Pilotage de la communication inter-composants via événements asynchrones
C4 Model	Représentation multi-niveaux des vues logicielles (contextuelle, logique, composant, code)
12-Factor App	Conformité des services cloud-native (stateless, config externe, build-release-run)
DevOps & GitOps	Intégration CI/CD, automatisation des déploiements, suivi par configuration déclarative
Normes de sécurité (OWASP, RGPD, Zero Trust)	Protection applicative, gestion des identités, traçabilité, auditabilité
ADR (Architecture Decision Records)	Historique des décisions structurantes, versionnées, argumentées
API-First / OpenAPI 3.x	Standardisation des interfaces, documentation automatisée, interopérabilité

Caractéristiques du travail d'architecture (cadre TOGAF étendu)

Dimension	Choix appliqué dans le projet Foosus
Niveau de détail	*Segment* + *Capacité* : découpage centré sur les domaines métiers et leurs besoins techniques associés
Période de couverture	Phase 1 : 6 mois (2 trimestres), avec projection cible sur 12-18 mois dans la roadmap
Sujets traités	Architecture métier, données, applicative, technologique, sécurité, migration
Niveau d'abstraction	Mix : • *Architecture cible concrète* (composants, schémas, APIs) • *Référentiels abstraits* (principes, standards, règles de gouvernance)
Ligne de base vs cible	Documentation des deux : • *As-Is* = état hérité du legacy (technique, dette, dépendances) • *To-Be* = microservices, cloud, CI/CD, découplage par domaine
Approche itérative	Oui, via des *cycles courts de validation* à chaque sprint ou MVP, intégration de feedback en continu
Partitionnement	Oui, architecture partitionnée par domaine fonctionnel (produit, commande, utilisateur, catalogue, paiement…) Interopérabilité entre domaines par APIs standardisées

Soutien au Continuum de l'entreprise :
L'architecture conçue soutient activement le **Continuum d'Entreprise**, c'est-à-dire la **capacité de Foosus à intégrer, faire évoluer et aligner ses différentes capacités métiers et technologiques** sur le long terme. Pour cela :

- Les artefacts TOGAF sont **versionnés**, **documentés** et **accessibles** via des outils collaboratifs ;
- Les décisions structurantes sont **justifiées et historisées** (via ADRs) ;
- L'architecture cible est **vivante**, **modulaire**, **interopérable** et capable d'accompagner l'évolution du modèle économique de l'entreprise (B2C → B2B2C, partenariats, nouvelles verticales métier).

6. Plan de travail

a. Analyse de l'existant (as-is)

i. Activités

- Cartographie du SI actuel (fonctionnelle, technique, applicative)
- Identification des composants legacy à remplacer ou à interfacer
- Analyse de la dette technique (code, infra, dépendances)
- Revue des processus critiques et des flux de données existants
- Recueil des irritants techniques et métiers auprès des équipes

ii. Livrables

- Diagrammes d'architecture existante (C4 - niveau 1 à 3)
- Inventaire des composants legacy et des dépendances
- Rapport de dette technique priorisée
- Cartographie des processus métier clés
- Synthèse des irritants techniques

b. Définition de l'architecture cible (to-be)

i. Activités

- Définition des principes directeurs et objectifs non-fonctionnels
- Conception de l'architecture cible (composants, domaines, flux)
- Définition du modèle de gouvernance (règles, standards)
- Déclinaison en vues métier, applicative, données, technique
- Choix des outils, patterns et technologies recommandées
- Préparation des transitions (interfaces, découplages, migrations)

ii. Livrables

- Vision cible d'architecture (C4 niveau 1 à 4, ArchiMate ou équivalent)
- Catalogue des microservices cibles et domaines fonctionnels
- Registre des principes d'architecture et standards techniques
- Dossiers d'architecture applicative, data et infra
- Modèles de gouvernance (RACI, processus, qualité, sécurité)
- Documentation des choix (ADRs)

c. Roadmap et plan de migration

i. Activités

- Identification des quick wins, MVPs techniques ou métiers
- Définition des lots de transformation (par domaine ou use case)
- Élaboration d'une stratégie de migration incrémentale (strangler pattern)
- Planification des phases, itérations, releases et jalons
- Définition des critères d'entrée/sortie pour chaque étape

ii. Livrables

- Feuille de route de mise en œuvre de l'architecture cible
- Matrice priorisée des chantiers techniques
- Planning de migration (Gantt ou roadmap synthétique)
- Cartes d'impact / dépendances entre composants
- Scénarios de coexistence et de décommissionnement du legacy

d. Gouvernance, pilotage et accompagnement

i. Activités

- Mise en place du Comité d'Architecture (fréquence, rituels, rôles)
- Formalisation des processus d'escalade et de décision
- Mise en œuvre du suivi des décisions (ADRs, changelog)
- Coaching et sensibilisation des équipes (tech talks, pair review)
- Animation des ateliers de validation croisée
- Évaluation continue des risques et des écarts

ii. Livrables

- Charte de gouvernance technique et Comité d'Architecture
- Registre des décisions (ADRs + changelog)
- Plan de formation et d'accompagnement des équipes
- Rapports de suivi des risques et des ajustements

- Tableaux de bord de pilotage architectural

e. Plan de communication

Le plan de communication vise à garantir que **toutes les parties prenantes disposent de l'information nécessaire, au bon moment et sous le bon format**, pour participer aux décisions, contribuer aux activités d'architecture et suivre l'évolution du projet.
La communication est structurée autour de **rituels synchrones**, **canaux asynchrones**, et d'un **référentiel documentaire centralisé**.

Objectif de communication	Public cible	Message clé / contenu	Canal	Fréquence	Responsable
Aligner les parties prenantes sur la vision d'architecture	CIO, CPO, CEO, Product	Vision as-is / to-be, roadmap, priorités stratégiques	Comité de pilotage, Note de cadrage (PDF + présentation)	Mensuel / jalons clés	Architecte
Partager les décisions d'architecture	Équipes Tech (Lead Dev, DevOps, Dev), PO	ADRs, standards, composants retenus, règles de gouvernance	GitHub (docs/adr), Slack #architecture, Revue ARB	À chaque décision majeure (bi-mensuel)	Architecte
Diffuser les documents TOGAF et livrables d'architecture	Toutes les équipes projet, futurs contributeurs	Livrables formels : modèle cible, cartographies, référentiels	Notion / Git (dépôt architecture), Newsletter interne	Trimestriel + mises à jour incrémentales	Architecte + Référent Technique
Sensibiliser aux enjeux de migration et aux changements	Devs, QA, UX, Produit	Problématiques de coexistence, dépendances legacy, plan de transition	Slack, démo sprint, Tech Talk	À chaque vague de migration (sprint review)	Architecte + PO
Informer sur l'avancement global du chantier architecture	Direction, PO, Product Designers	Indicateurs clés, risques, jalons atteints, MVP livrés	Tableau de bord, Reporting Notion, Réunion CoPil	Mensuel	Architecte + PM
Former et faire monter en compétence les équipes	Développeurs, Intégrateurs, Product Designers	Prise en main des patterns, outils, standards, bonnes pratiques	Tech Talks, Wiki interne, Vidéos asynchrones	1 à 2 fois / mois	Architecte + Lead Dev
Collecter du feedback	Dev Team,	Retour	Formulaire	À chaque fin	Architecte + Scrum

Objectif de communication	Public cible	Message clé / contenu	Canal	Fréquence	Responsable
sur l'adoption des standards	PO, QA	d'expérience, écueils d'implémentation, suggestions	Notion, Slack thread, rétrospectives	de sprint ou itération critique	Master

Livrables associés au plan de communication

- **Calendrier de diffusion** des messages clés (aligné avec la roadmap technique)
- **Templates standardisés** de compte-rendu, note de décision, newsletter interne
- **Playbook de communication technique** : comment écrire, valider, et publier un ADR ou un document d'architecture
- **Kit d'embarquement architecture** : pack PDF / site interne avec les principaux modèles, principes, diagrammes, ADRs à jour

Intégration dans les sprints

Le plan de communication est **intégré dans le backlog Agile**, sous forme :

- d'**user stories** (ex. : "En tant que développeur, je veux accéder à une documentation claire des microservices pour intégrer mon module plus vite"),
- de **tâches techniques** (publier un ADR, mettre à jour un modèle, animer un tech talk),
- et de **Definition of Done** : aucun changement structurant ne peut être clôturé sans documentation + communication associée.

f. Évènements

Événement	Fréquence	Objectif	Audience
Architecture Review Board (ARB)	Bi-mensuel	Revue et validation des décisions techniques clés	Architecte, CIO, PO, Lead Dev
Comité de Pilotage stratégique	Mensuel	Suivi macro du projet, arbitrages stratégiques	CEO, CIO, CPO, Architecte
Sprint Review & Démo technique	Fin de sprint	Présentation des évolutions structurelles et techniques	Équipes produit, tech, stakeholders
Ateliers d'architecture collaboratifs	À chaque lot fonctionnel / composant clé	Co-construction des modèles, validation des patterns	Architecte, Dev Team, PO, Sécurité
Tech Talks / Formations internes	Mensuel	Partage de bonnes pratiques, montée en compétence	Développeurs, PO, QA, Lead UX

g. Canaux

Canal	Usage principal
Slack (#architecture, #ops, #tech)	Communication rapide, escalades, discussions techniques
GitHub / GitLab	Suivi des tickets, pull requests, ADRs, changelog
Notion / Confluence	Référentiel documentaire structuré (TOGAF, décisions, visions, modèles)
Email	Communications formelles, convocations, validations
Meet / Teams / Zoom	Réunions, ateliers, démonstrations, support transversal

h. Formats

Type de contenu	Format utilisé
Modèles d'architecture	Diagrammes C4 (Structurizr, PlantUML), ArchiMate
Décisions techniques	Fiches ADR en markdown dans Git
Documents TOGAF	Markdown versionnés dans dépôt Git, PDF pour diffusion externe
Feuilles de route & plannings	Roadmaps visuelles (Miro, FigJam, Gantt), Tableaux de bord
Comptes-rendus	Templates standardisés (action / décision / prochaine étape) dans Notion

i. Contenu

- **Vision d'architecture (as-is / to-be)** : visualisation, principes, objectifs
- **Cartographies** : SI, données, processus, microservices, APIs
- **Référentiel de standards** : sécurité, développement, qualité, performance
- **Feuille de route** : phasage, priorisation, dépendances
- **Décisions structurantes** : ADRs, validations, arbitrages
- **Rapports de risques et indicateurs de conformité**

j. Durée et effort

- Le **plan de communication est actif tout au long du projet**, dès la phase préliminaire jusqu'au déploiement de la dernière vague.
- La charge associée à la communication représente en moyenne **15 à 20 % du temps de l'architecte**, et **5 à 10 % pour les profils contributeurs** (PO, Lead Dev, QA).
- Cette charge est planifiée dans les sprints et rattachée à des **stories ou tâches identifiées**.

k. Collaboration

La collaboration s'appuie sur :

- Une **documentation en mode "docs-as-code"**, versionnée, traçable et partagée ;
- Des **ateliers multi-profils** favorisant la co-construction ;
- Un usage transversal de Git, Slack, Notion, et Miro pour briser les silos ;
- Un **calendrier commun d'activités d'architecture**, visible par tous ;

- Une approche **feedback-driven**, avec recueil régulier des remarques des équipes pour améliorer la communication et les livrables.

I. Plan et calendrier du projet

Le projet suit une organisation en **vagues successives**, avec un rythme moyen de **1 vague par trimestre**, découpée en :

Phase	Durée estimée	Livrables principaux
Phase 0 – Cadrage & Préliminaire	2 semaines	Vision initiale, plan de travail, gouvernance
Phase 1 – Diagnostic & Architecture cible	1 mois	Modèles TOGAF, vision to-be, standards, ADRs
Phase 2 – MVP architecture & lot prioritaire	1 à 2 mois	Implémentation d'un domaine pilote, validation des patterns
Phase 3 – Migration progressive	3 mois	Déploiements itératifs, documentation vivante, accompagnement
Phase 4 – Gouvernance & pérennisation	continu	Contrats d'architecture, automatisation, indicateurs de conformité

7. Risques et facteurs de réduction

a. Analyse des risques

L'analyse des risques permet d'**anticiper les obstacles potentiels** à la réussite du travail d'architecture et de **prévoir les actions de réduction ou de mitigation** adaptées. Elle est évolutive et sera **mise à jour régulièrement** par l'architecte au fil des itérations.

Le tableau ci-dessous fournit une première évaluation structurée des risques.

ID	Risque	Gravité	Probabilité	Facteur de réduction	Propriétaire
R1	Forte dépendance au système legacy (monolithe) rendant difficile la migration	Haute	Élevée	Mise en place d'une stratégie de migration incrémentale (Strangler Pattern), découplage par proxy/API	Architecte
R2	Résistance au changement des équipes techniques ou produit	Moyenne	Moyenne	Sensibilisation via ateliers d'architecture, pair design, implication en amont dans les choix techniques	PO / Lead Dev
R3	Mauvaise adoption des standards ou patterns d'architecture proposés	Élevée	Moyenne	Publication d'un guide de bonnes pratiques, relectures de code, contrôle qualité, formation continue	Architecte / Lead Dev
R4	Retards dans la livraison des composants critiques ou MVPs techniques	Élevée	Moyenne	Roadmap réaliste avec marges, suivi itératif, intégration architecture dans backlog produit	PM / Architecte

ID	Risque	Gravité	Probabilité	Facteur de réduction	Propriétaire
R5	Difficulté à maintenir une documentation d'architecture à jour	Moyenne	Élevée	Intégration docs-as-code dans Git, responsabilisation de chaque équipe sur ses composants, validation en review	Architecte / Tech Leads
R6	Risque de dette technique créée dans la phase de transition	Haute	Moyenne	Checklists de qualité, contractualisation des composants par ADRs, validation systématique avant fusion	Architecte / QA
R7	Conflits entre roadmap produit et priorités architecturales	Moyenne	Moyenne	Alignement permanent avec le CPO, comité d'architecture pour arbitrage, pilotage par la valeur	Architecte / CPO

Note :

- Les **niveaux de gravité et de probabilité** sont évalués sur une échelle simple : *Faible / Moyenne / Élevée*.
- En cas de besoin, cette évaluation pourra être convertie dans un **modèle quantitatif matriciel**, avec pondération des risques et scoring de criticité (ex. : méthode AMDEC, ISO 31000).
- Le **registre des risques** est mis à jour dans Notion et versionné dans le dépôt Git, accessible à toutes les parties prenantes.

b. Hypothèses

Les hypothèses suivantes servent de **base de travail** à cette Déclaration de Travail d'Architecture. Elles doivent être **validées, surveillées ou ajustées** au cours du projet. Certaines peuvent faire l'objet de vérifications lors des phases d'analyse ou de planification.

ID	Hypothèse	Impact	Propriétaire
H1	Les équipes techniques sont disponibles pour participer aux ateliers d'architecture et aux revues régulières	Forte : un manque de disponibilité retarderait la co-construction des modèles et la validation des choix techniques	CTO / Engineering Manager
H2	Les briques existantes du système legacy sont suffisamment documentées pour permettre une cartographie fiable	Moyenne à forte : si la documentation est lacunaire, un audit approfondi sera nécessaire, rallongeant la phase d'analyse	Architecte
H3	L'infrastructure cloud choisie supportera les exigences techniques définies (scalabilité, observabilité, sécurité)	Moyenne : une limitation du provider pourrait contraindre les choix technologiques ou alourdir l'architecture	DevOps Lead
H4	Le Product Owner intégrera les exigences d'architecture dans le backlog produit (même sans livraison visible immédiate)	Haute : condition essentielle pour garantir la priorisation continue des fondations techniques	PO Principal
H5	Les outils de documentation, gestion des décisions et versioning (Notion, Git, Slack,	Faible à moyenne : en cas de refus ou d'alternatives imposées, la standardisation	Architecte / PMO

ID	Hypothèse	Impact	Propriétaire
	ADRs) sont disponibles et acceptés par les équipes	documentaire sera plus complexe	
H6	Les jalons de déploiement définis dans la roadmap produit sont compatibles avec la stratégie de migration technique par lots	Moyenne : un décalage pourrait générer une pression sur les équipes ou des livraisons incomplètes	CPO / Architecte

Remarques :

- Certaines hypothèses sont **techniques**, d'autres **organisationnelles** ou **méthodologiques**.
- Leur **validation explicite** est recommandée en début de phase d'architecture (préliminaire ou phase A du cycle TOGAF).
- Une hypothèse devenue invalide doit être **remplacée par une action corrective**, documentée dans le registre des risques ou modifiant la feuille de route.

8. Critères d'acceptation et procédures

a. Métriques et KPIs

L'évaluation du succès du travail d'architecture repose sur un ensemble de **métriques qualitatives et quantitatives**, conçues pour mesurer :

- la **valeur apportée aux équipes métier et tech**,
- le **niveau d'adoption des standards définis**,
- la **capacité de l'organisation à transformer sa plateforme de façon structurée**.

Métrique	Technique de mesure	Valeur cible	Justification	Notes supplémentaires
Taux d'adoption des composants cibles (microservices)	Suivi des déploiements dans les sprints / dashboards CI/CD	≥ 80 % des nouveaux développements utilisent les patterns cibles	Mesure la réussite de la migration vers l'architecture cible	Le ratio sera affiné selon le domaine métier (produit, user, commande…)
Taux de documentation des composants (docs-as-code)	Revue des PR + audit du dépôt Git sur les dossiers / README / ADR	100 % des composants critiques documentés	Mesure la qualité et la maintenabilité à long terme	L'absence de documentation bloque la validation ARB
Nombre de décisions structurantes enregistrées (ADRs)	Compte des ADR validés dans Git + changelog	≥ 20 décisions majeures documentées au T2	Traçabilité des choix d'architecture, base de référence partagée	Lié aux phases A à D du cycle TOGAF
Temps de traitement moyen d'une demande de changement (scope technique)	Durée entre ticket ouvert et validation ARB	≤ 5 jours ouvrés	Capacité à maintenir l'agilité et la gouvernance en parallèle	Concerne uniquement les demandes avec impact architectural

Métrique	Technique de mesure	Valeur cible	Justification	Notes supplémentaires
Niveau d'alignement des jalons d'architecture avec la roadmap produit	Matching entre roadmap produit et planning archi	≥ 90 % des jalons respectés ou réalignés en sprint planning	Évalue la synchronisation stratégique entre Tech et Produit	Mesuré mensuellement par le Comité de Pilotage
Réduction de la dette technique identifiée	Nombre d'éléments critiques du legacy migrés ou supprimés	≥ 60 % du backlog technique traité dans les 2 trimestres	Évalue l'impact concret de l'architecture sur la stabilité du SI	Basé sur les rapports d'audit de la phase 1

Notes supplémentaires :

- Ces KPIs sont mis à jour dans le **tableau de bord d'architecture** (Notion ou PowerBI, selon outils internes).
- Leurs valeurs cibles peuvent être ajustées par **le Comité d'Architecture** à chaque fin d'itération.
- En complément, des **sondages qualitatifs** peuvent être menés auprès des développeurs et PO pour mesurer la satisfaction liée à l'architecture (clarté des choix, facilité d'implémentation, autonomie).

b. Procédure d'acceptation

L'acceptation des livrables issus du travail d'architecture s'effectue selon une **procédure cadrée, itérative et traçable**, afin de garantir la **qualité, la cohérence, et l'adhésion de l'ensemble des parties prenantes**. Cette procédure est pilotée par l'Architecte en lien avec le Comité d'Architecture.

Étapes de la procédure d'acceptation

Étape	Description	Responsables / Participants	Outils / Documents associés
1. Pré-validation interne	Relecture du livrable par l'Architecte + Lead Dev pour vérifier la conformité aux standards	Architecte, Lead Dev	Draft Git, checklist de validation
2. Revue collaborative	Présentation en atelier aux parties prenantes concernées (tech, produit, sécurité, UX)	Architecte, PO, Dev Team, Sécurité	Réunion, commentaires Notion / Git
3. Validation formelle	Approbation par le Comité d'Architecture ou le Comité de Pilotage selon le niveau	Architecte, CIO, CPO, éventuellement CEO	Signature dans Git (merge / tag) ou validation PDF
4. Archivage et diffusion	Dépôt du livrable dans le référentiel partagé, annonce sur Slack / mail	Architecte, PM	Notion, GitHub, Slack, email
5. Historisation	Enregistrement du statut, de la date et du versioning de l'artefact	Architecte	Journal des livrables (changelog)

Types de livrables concernés

- Vision d'architecture (as-is, to-be)
- Modèles TOGAF (cartographies, référentiels, principes)
- Roadmap et feuille de route technique
- Standards et bonnes pratiques
- Architecture Decision Records (ADRs)
- Stratégie de migration
- Indicateurs et rapport d'avancement

Modalités de signature

- Les livrables formels (versions validées) sont signés via :
  - **Pull Request approuvée + tag Git signé**, OU
  - **Approbation numérique via DocuSign / PDF avec initiales électroniques**, selon le niveau de sensibilité
- La liste des signataires est consignée dans la section **9. Approbations signées** du présent document.
- Toute modification post-validation fait l'objet d'un **changelog et d'une revalidation partielle ou complète** selon l'impact (niveau mineur / majeur).

Remarques complémentaires

- Les **critiques et ajustements** sont recueillis pendant la phase de revue collaborative.
- En cas de **désaccord bloquant**, une réunion exceptionnelle du Comité d'Architecture est convoquée pour arbitrage sous 3 jours ouvrés.
- La **définition des critères de succès (KPIs)** associés à chaque livrable est fournie en annexe ou dans les user stories correspondantes.

9. Approbations signées

a. Date de signature

Le tableau ci-dessous consigne les **signatures formelles** des parties prenantes ayant approuvé la Déclaration de Travail d'Architecture. Ces signatures valident le **contenu, le périmètre, la vision, les rôles, les processus et le plan de travail** associés à l'initiative de transformation architecturale.

Modalités de signature :

- Les signatures peuvent être effectuées **électroniquement** via DocuSign, GitHub (Pull Request + tag), ou **manuellement** en version PDF.

- Le fichier signé est **archivé dans le dépôt Git** du projet et partagé dans l'espace documentaire (Notion ou Drive).

Nom	Fonction	Organisation	Signature	Date
Hedi Dhib	Architecte Logiciel	Foosus		19/08/2025
Ash Callum	Chief Executive Officer (CEO)	Foosus		
Natasha Jarson	Chief Information Officer (CIO)	Foosus		
Daniel Anthony	Chief Product Officer (CPO)	Foosus		
Christina Orgega	Chief Marketing Officer (CMO)	Foosus		
Jo Kumar	Chief Financial Officer (CFO)	Foosus		
Pete Parker	Engineering Owner	Foosus		
Jack Harkner	Operations Lead	Foosus		

Annexe A – Glossaire des acronymes :

Acronyme	Signification	Définition
TOGAF	The Open Group Architecture Framework	Cadre de référence pour l'architecture d'entreprise
ADM	Architecture Development Method	Méthodologie TOGAF en 8 phases pour construire une architecture
ACF	Architecture Content Framework	Cadre TOGAF définissant les artefacts d'architecture
CI/CD	Continuous Integration / Continuous Delivery	Ensemble de pratiques d'automatisation des tests, intégration et déploiement
EDA	Event-Driven Architecture	Architecture pilotée par événements, favorisant la découplage et l'asynchronisme
DDD	Domain-Driven Design	Approche de conception centrée sur les domaines métier
C4 Model	Context, Container, Component, Code	Méthodologie de modélisation logicielle multi-niveaux
ADR	Architecture Decision Record	Document décrivant une décision structurante avec justification
MVP	Minimum Viable Product	Version minimale viable d'un produit ou service
IAM	Identity and Access Management	Gestion des identités et des accès (SSO, rôles, permissions)
RBAC	Role-Based Access Control	Contrôle d'accès basé sur les rôles
OWASP	Open Web Application Security Project	Référentiel de sécurité pour les applications web
SSO	Single Sign-On	Mécanisme d'authentification unique
OpenAPI	OpenAPI Specification	Standard pour la description d'interfaces RESTful (anciennement Swagger)
GitOps	Git + Operations	Pratique d'infrastructure-as-code pilotée par Git
Strangler Pattern	Modèle de migration incrémentale	Stratégie de remplacement progressif d'un système legacy

Annexe B – Référentiel de standards et bonnes pratiques :

Domaine	Standard / Référence	Lien ou description
Architecture d'entreprise	TOGAF 10	https://www.opengroup.org/togaf
Modélisation logicielle	C4 Model	https://c4model.com
Sécurité applicative	OWASP Top 10	https://owasp.org/www-project-top-ten
APIs	OpenAPI 3.x	https://swagger.io/specification
Microservices	12-Factor App	https://12factor.net
Documentation technique	Docs-as-code (Markdown, ADRs, versioning Git)	Pratique recommandée pour la traçabilité et la collaboration
DevOps	GitOps / CI/CD	https://opengitops.dev, pipelines GitLab / GitHub Actions
Design de services	Domain-Driven Design	https://domainlanguage.com/ddd/
Gestion des identités	Zero Trust / OAuth2	https://oauth.net/2/ & principes Zero Trust de NIST
Observabilité	Logs, traces, métriques via ELK / Prometheus / Grafana	Mise en place obligatoire pour la supervision des microservices