Nouvelle plateforme Foosus

Déclaration de travail d'architecture

ARCHITECTE LOGICIEL			
PROJET		**DATE DÉPÔT**	**DATE FIN**
P5	Concevez une nouvelle architecture afin de soutenir le développement de votre entreprise	--/03/2023	--/03/2023
Livrable 1/4	Déclaration de Travail d'Architecture		

Marc LEFRANÇOIS
mlefrancois.contact@gmail.com

CONTRÔLE DES RÉVISIONS

DATE	STATUT	AUTEUR	RÔLE	ACTIONS
13/01/2023	Brouillon	Marc LEFRANÇOIS	Créateur	Création du document et mise en place du plan.
20/01/2023	Brouillon	Marc LEFRANÇOIS	Créateur	Enrichissement du document ➔ V1.
27/01/2023	Brouillon	Grégoire CATAN	Correcteur	Relecture du document et remarques.
27/01/2023	Brouillon	Marc LEFRANÇOIS	Créateur	Enrichissement du document ➔ V2.
03/02/2023	Brouillon	Grégoire CATAN	Correcteur	Relecture du document et remarques.
03/02/2023	Brouillon	Marc LEFRANÇOIS	Créateur	Enrichissement du document ➔ V3.
10/02/2023	Brouillon	Grégoire CATAN	Correcteur	Relecture du document et remarques.
10/02/2023	Brouillon	Marc LEFRANÇOIS	Créateur	Enrichissement du document ➔ V4.
16/02/2023	Brouillon	Marc LEFRANÇOIS	Créateur	Ajustement du contenu ➔ V5.
26/03/2023	Brouillon	Grégoire CATAN	Correcteur	Relecture du document et remarques.
28/03/2023	Brouillon	Marc LEFRANÇOIS	Créateur	Ajustement du contenu ➔ V6.
29/03/2023	Validé	Grégoire CATAN	Correcteur	Relecture du document et validation.
29/03/2023	Relu	Jérôme POCHELU	Correcteur	Relecture du document et remarques.
31/03/2023	Finalisé	Marc LEFRANÇOIS	Créateur	Ajustement du contenu ➔ V7.

Légendes :

Statut = Brouillon, Validé, Relu et Final.

Rôle = Créateur / Modificateur, relecteur/correcteur, validateur.

1. Objet

Le présent document concerne le framework d'architecture TOGAF dans le cadre du projet initialisant la définition de la nouvelle plateforme Foosus et notamment sa nouvelle architecture.
Son objectif est de décrire toute la méthodologie, en identifiant les outils, les pratiques et les méthodes pouvant être mis en œuvre pour mener à bien l'implémentation de l'architecture retenue pour le projet.

2. Mission d'architecture

2.1. Contexte

Foosus, une start-up de 3 ans positionnée dans le secteur de l'alimentation durable, s'est donnée pour objectif de soutenir l'alimentation locale en mettant en relation les consommateurs avec des producteurs et des artisans locaux via à la fois une application mobile, et un site Web d'e-commerce.
Cependant, elle rencontre des problèmes techniques et fonctionnelles qui respectivement écorne son image de marque auprès de ses utilisateurs, et l'empêche d'évoluer pour répondre aux exigences du marché.
Foosus a donc besoin de répondre rapidement et activement à ces problématiques, si elle souhaite garder la confiance de ses utilisateurs. Ce qu'elle souhaite faire avec une nouvelle plateforme basée sur une nouvelle architecture technique.

2.2. Objectifs

Les objectifs désignés ci-après ont été énoncés respectivement dans les courriels de **Ash CALLUM** cofondatrice et CEO de **Fossus** et ayant pour objet « **Fonction Architecture** », et de Natasha JARSON cofondatrice et CIO de **Foosus**, ainsi que dans les documents attenants (*voir la section « **2.7. Documents fournis** »*).
Il s'agit de définir une nouvelle architecture standardisée permettant :

1. d'effacer la dette technique et ne plus en avoir à l'avenir ;
2. d'évoluer avec l'entreprise, id. soutenir sa croissance de manière continue ;
3. de prendre en charge un nouvel emplacement géographique, afin de tirer parti de la géolocalisation pour relier fournisseurs et consommateurs, pour proposer des produits disponibles à proximité des lieux de résidence de ces derniers ;
4. de faciliter la maintenance et les développements futurs (*maitrise des coûts et des délais*) ;
5. de fournir la fiabilité nécessaire aux clients, fournisseurs et consommateurs de la Société.

L'état final pour la société est de créer une plateforme de commerce électronique polyvalente lui permettant de franchir le million d'utilisateurs, et de concurrencer ainsi les grandes entreprises mondiales de commerce électronique qui dominent le marché de l'alimentation durable. Aussi, la finalité est que cette nouvelle architecture soit capable de supporter les exigences de la nouvelle plateforme.

2.3. Démarche globale

Afin de répondre aux objectifs susmentionnés, cette phase d'initialisation va :

- tout d'abord s'inscrire dans la culture Lean de la société, afin de répondre à la contrainte projet définie en amont ;
- fournir une vision architecturale répondant aux besoins et contraintes définies en amont ;
- définir les contrats sur le partage de cette vision avec les parties prenantes ;
- spécifier les conditions requises pour une implémentation de l'architecture conforme aux exigences.

Ci-fait, cette phase d'initialisation pourra alors donner à la phase d'implémentation, afin d'atteindre l'état final recherché mentionné dans les objectifs vus plus haut.

2.4. Parties prenantes

- **Ash CALLUM**, cofondatrice et CEO (*Chief Executive Officer*) ou Directrice Général de la société « **Foosus** », en qualité de responsable de la stratégie de l'entreprise ;
- **Natasha JARSON**, cofondatrice et CIO (*Chief Information Officer*) ou Directrice des Systèmes d'Information (DSI) de la société « **Foosus** », en qualité d'encadrant supérieur et de pilote du processus SI ;
- **Jo KUMAR**, CFO (*Chief Financial Officer*) ou Directrice financier, en charge de la relation avec le consortium d'investisseurs ;
- **Daniel ANTHONY**, CPO (*Chief Product Officer*) ou Directeur de Produit ;
- **Christina ORGEGA**, CMO (*Chief Marketing Officer*) ou Directrice Marketing, en qualité de référente pour le bon positionnement de la plateforme vis-à-vis du marché ;
- **Jack HARKNESS / HARKNER**, DO ou Directeur des Opérations de la société « **Foosus** », en qualité d'évaluateur technique ;
- **Peter Parker**, responsable ingénierie, en qualité de référent technique pour la création du plan de transformation de l'architecture ;
- **Équipe de développement** (*15 développeurs*) en charge de la maintenance de l'actuelle plateforme et de la réalisation de la nouvelle.
- **Marc LEFRANÇOIS**, Architecte Logiciel de la société « **Foosus** », en charge de définir la nouvelle architecture conformément aux attentes de la société, dans le cadre de la phase d'initialisation.

2.5. Périmètre

Le périmètre porte sur la plateforme e-commerce de la société **Foosus**, représentant son cœur de métier, par la définition et la mise en place d'une nouvelle plateforme basée sur une nouvelle architecture.

2.6. Livrables

Les livrables découlent directement de la démarche vue plus haut, et comprend donc une présentation des artefacts suivants, accessibles à chacune des parties prenantes clés depuis le dépôt GitHub :

- Une déclaration de travail d'architecture TOGAF spécifiant un framework d'architecture sur mesure, et comprenant :
  - une articulation claire d'une vision et d'une direction d'architecture qui permettent à Foosus de développer les capacités nécessaires pour réussir sur le marché ;
  - un État Cible de l'Architecture vers lequel l'organisation doit itérer ;
  - un process et une approche d'architecture sur mesure, mais flexibles, qui conviennent aux structures de nos équipes et à la topologie de notre organisation.
- Un document de spécification des conditions requises pour l'architecture basé sur le modèle TOGAF, supprimant toute ambiguïté et spécifiant :
  - une description des conditions requises pour l'implémentation de l'architecture ;
  - une description des conditions requises pour la conformité de l'implémentation.
- les contrats suivants avec les partenaires internes pour nous assurer que nous collaborons sur un parcours architectural partagé, sous la forme d'un contrat architectural ouvert à la révision et à l'amélioration :
  - accords avec le business sur l'architecture proposée ;
  - accords avec les fonctions design et développement.

2.7. Documents fournis

- l'autorisation de projet de Ash CALLUM ;
- les conditions requises pour le business de haut niveau de Daniel ANTHONY et Natasha JARSONS ;
- La vision architecturale ;
- Les flux de valeurs organisationnelle ou VSM pour Value Stream Map ;
- Organigramme de FOOSUS ;
- Le document de référence de l'architecture actuelle « Legacy C4 FR ».

3. Cycle de vie du projet

3.1. Lignes directrices du projet

Ci-dessous les lignes directrices pour le projet approuvé :

- Les solutions open source sont préférables aux solutions payantes.
- Le support continu des composants doit être pris en compte lors de leur sélection ou lors des prises de décision de création ou d'achat.
- Toutes les solutions du commerce ou open source doivent, dans la mesure du possible, faire partie d'une même pile technologique afin de réduire les coûts de maintenance et de support continus.
- Respecter la culture du Lean de Foosus.

3.2. Contraintes projet

Cette section contient les contraintes concernant le projet, définies en amont et dans les différents documents fournis de référence vus plus haut à la section « **2.7. Documents fournis** », et regroupées dans le tableau ci-dessous :

IDENTIFIANT	INTITULÉ
CTR-PRJ-1	Le projet est approuvé pour un coût de 50 000 USD (45 190 €).
CTR-PRJ-2	Le projet est approuvé pour un délai de 6 mois.
CTR-PRJ-3	L'architecture doit permettre d'obtenir le meilleur rapport qualité-coût.
CTR-PRJ-4	Pas de mise à jour ou de reprise de la plateforme actuelle. Celle-ci est conservée en mode maintenance uniquement.
CTR-PRJ-5	L'architecture peut inclure de nouveaux composants personnalisés ou des composants du commerce pour favoriser la flexibilité, la stabilité et l'extensibilité.
CTR-PRJ-6	Les solutions open source sont préférables aux solutions payantes.
CTR-PRJ-7	Le support continu des composants doit être pris en compte lors de leur sélection ou lors des prises de décision de création ou d'achat.
CTR-PRJ-8	Toutes les solutions du commerce ou open source doivent, dans la mesure du possible, faire partie d'une même pile technologique, afin de réduire les coûts de maintenance et de support continus.
CTR-PRJ-9	Les livrables doivent pouvoir être fournis à intervalles réguliers pour que le nouveau système soit rapidement opérationnel et puisse être doté de nouvelles fonctionnalités au fil du temps.
CTR-PRJ-10	Dissocier les nouvelles livraisons de l'architecture et de l'infrastructure existantes afin de limiter les interruptions de service.
CTR-PRJ-11	Respecter la culture du Lean de Foosus.

3.3. Méthodologie

Comme mentionné plus haut dans la section « **2.3. Démarche globale** », le projet d'initialisation doit s'inscrire dans la culture Lean de Foosus, conformément aux attentes de cette dernière. Pour rappel, d'une manière générale l'approche Lean est un processus itératif se basant sur la recherche de la performance par l'amélioration continue, et dont l'objectif est l'amélioration de la valeur globale pour le client. Autrement dit, on recherche et élimine tout ce qui n'a pas de plus-value, id. les pertes dans le travail qui n'apporte aucune « valeur métier » à un produit ou à un service, et ce afin d'apporter une réponse efficiente au besoin / attente client.
Aussi pour ce faire, le Lean IT repose sur cinq principes :

1. Identifier les processus IT (*métiers, maîtrise d'ouvrage, conseil d'administration, clients finaux des produits et services de l'entreprise, …*) et détermination de la valeur ajoutée du point de vue des clients : réponse aux attentes et besoins explicites et implicites, critères de performances et de disponibilité, différenciation avec les nouvelles technologies, respect des contraintes réglementaires et environnementales, etc.
2. Définir la chaîne de valeur ajoutée, appelée Value Stream Mapping (VSM) : la description, qualification et quantification en performance de chacune des étapes de ce processus, sous la forme de chaîne de valeur (temps de traitement de chaque étape, délai entre deux étapes, taux de transformation, etc.)
3. Assurer le flux continu avec un mouvement continu des produits, services et informations, de bout en bout en éliminant tout gaspillage.
4. Passer du « flux poussé » au « flux tiré » (ou Pull) : le client devient demandeur, rien n'est fait en amont du processus tant que le client ne montre pas ses besoins en aval.
5. Mettre en œuvre une dynamique d'amélioration continue pérenne afin de tendre vers la perfection.

Ceci étant, il serait également intéressant de porter notre regard sur la méthodologie agile Scrum. En effet, Scrum est également basé sur un processus itératif, avec une organisation et des rôles bien définis. De ce fait elle pourra s'intégrer au processus Lean IT facilement en y apportant spécifiquement la partie gestion de projet, aussi bien pour assurer la transformation architecturale que de l'implémentation de cette dernière.
De plus, Scrum offre un cadre de travail structuré (*holistique*) itératif, et facilement appréhendable, et permettant d'être réactif en cas de problème. Ce qui en fait un standard incontournable, et dont l'utilisation est la plus répandue des méthodes agiles.
Aussi, nous permettra-t-elle d'atteindre notre objectif dans les meilleures conditions.

3.4. Plan de gestion des intervenants

Cette phase d'avant-projet intervient avant tout lancement de projet, dans le but de s'assurer autant de sa faisabilité, que de son bon déroulement du début à la fin.
Celle-ci requiert les définitions suivantes :

1. Activités du projet.
2. Registre des intervenants.
3. Matrice des responsabilités.
4. Planification.
5. Gestion des risques.

3.4.1. Activités du projet

Les activités du projet constituent l'ensemble des tâches à réaliser pour concourir à la finalité de celui-ci en l'implémentation des outils de visioconférence et partage de fichiers comme composant de l'actuel système.

Pour pouvoir être réalisée, cette étape nécessite en prérequis des éléments telle que la définition des besoins (*fonctionnels et non-fonctionnels*), des contraintes, et ainsi que la solution retenue, sur lesquels elle s'appuie.

Pour définir les activités du projet, il convient de passer par les étapes suivantes :

1. Prendre connaissance des besoins et des contraintes susmentionnés.
2. Recenser les tâches à réaliser.
3. Ordonner les tâches à réaliser.

À noter que les besoins (*fonctionnels et non-fonctionnels*) et les contraintes ont été définis dans les documents énumérés à la section « **2 .7. Documents Fournis** » du présent document.

À noter également que les tâches à réaliser, dans le cadre d'un projet agile Scrum, représentent l'ensemble des fonctionnalités à réaliser et qui seront chacune par la suite subdivisées en tâches à réaliser au début de chaque sprint par l'équipe de développement ou dev. team.

3.4.2. Registre des intervenants

L'objectif du registre des intervenants est d'identifier l'ensemble des ressources humaines pour la réalisation du projet.
Pour ce faire, il faut identifier :

1. les rôles / métiers nécessaires ;
2. les personnes pouvant remplir chacun des rôles.

À date, les rôles suivants ont déjà identifié :

- Projet :
  - **Product Owner**, pour gérer le projet et l'avancement de la réalisation.
  - **Scrum Master**, pour s'assurer que les principes Scrum soient bien appliqués.
  - **Business Owner**, le client ou son représentant, pour valider les réalisations attendues.
  - **Directeur de projet**, pour superviser l'ensemble du projet.
  - **Testeur du projet** (recette), pour s'assurer que la réalisation corresponde bien au besoin fonctionnel.
- Implémentation :
  - **Architecte Logiciel**, pour la recherche et la conception de la solution, ainsi que la supervision de son implémentation.
  - **Référent technique**, pour superviser les développements, et débloquer d'éventuelles situations.
  - **Concepteur / développeur full stack**, pour la réalisation de la partie code.
  - **Concepteur / développeur base de données**, pour la réalisation de la partie éponyme.
  - **Concepteur / développeur Web**, pour la réalisation de la partie Web.
  - **Intégrateur Web**, pour s'occuper de tous les aspects charte graphique.
  - **Testeur fonctionnel**, pour s'assurer avant la livraison que les fonctionnalités réalisées soient exclues de tout bug.
  - **Intégrateur système**, pour gérer tous l'aspect implémentation dans le système actuel.
- Opérationnel :
  - **Administrateur système**, pour la supervision et l'optimisation de la solution en production.
  - **Administrateur base de données**, pour la supervision et l'optimisation du système de gestion de base de données en production de la solution.
  - **Support**, pour assurer le support auprès des utilisateurs.

Nous avons également à date déjà identifié les intervenants suivants :

- **Business Owner : Natasha JARSON**.
- **Product Owner : Jack HARKNESS / HARKNER**.
- **Directeur de projet : Natasha JARSON**.
- **Architecte Logiciel : Marc LEFRANÇOIS**.

À noter qu'en l'état actuel du projet, et donc des informations que nous disposons, ces deux listes seront bien entendu à parfaire par la suite.

3.4.3. Matrice des responsabilités

La matrice des responsabilités, ou matrice RACI, définie l'attribution des tâches pour chaque intervenant, ainsi que la nature de leur relation (***R**esponsable, **A**pprobateur, **C**onsulté et **I**nformé*). Comme le laisse supposer son nom, elle se présente sous la forme d'un tableau à double entrées, ou matrice, avec d'un côté les tâches à réaliser, et de l'autre les intervenants du projet, et la nature de leur relation.
L'objectif de cette matrice est de cadrer le périmètre de chaque intervenant, en évitant que tout à chacun intervienne sur le projet et les tâches à réaliser comme il l'entend, et donc de créer de la confusion. Ce qui aurait pour conséquence de compliquer la bonne exécution du projet, pouvant aller jusqu'à le mettre en péril.

3.4.4. Planification

Une fois définis les intervenants, leur(s) rôle(s) et responsabilité, il restera à planifier les tâches à réaliser, dans le but d'atteindre les objectifs du projet ; c'est sa feuille de route.
Pour ce faire, il faut définir :

1. l'ordre dans lequel réaliser les fonctionnalités ;
2. le planning de réalisation.

L'ordonnancement des fonctionnalités est effectué au début de projet par l'Architecte Logiciel, et constituera, dans le cadre de la gestion de projet Scrum, le product backlog, point d'entrée pour le Product Owner, à qui il appartiendra de planifier les sprints.
Le product backlog constituera également le point d'entrée pour l'équipe de développement au début de chaque Sprint et lors de la mêlée quotidienne Scrum (daily Scrum).

3.4.5. Gestion des risques

La gestion des risques a pour objectif d'assurer le bon déroulement du projet, par la gestion de tout risque potentiel pouvant survenir et impacter significativement ce dernier.
Elle se décompose comme suit :

1. Recenser les risques.
2. Déterminer la probabilité et le(s) impact(s) de chacun des risques recensé.
3. Prioriser les risques en fonction de leur probabilité et leur(s) impact(s).

Ci-dessous les risques déjà identifiés, il s'agit de cas classiques, aussi cette liste sera à compléter par la suite par le Product Owner.
Le premier tableau contient la pondération pour la probabilité et l'impact du risque.

PONDÉRATION	LIBELLÉ PROBABILITÉ	LIBELLÉ IMPACT
1	Peu probale	Négligeable
2	Possible	Marginal
3	Probable	Critique
4	Très probable	Catastrophique

Sur les deux pages suivantes, se trouvent deux autres tableaux listant les risques.
Le premier tableau contient la description des risques et leur probabilité, tandis que le second quant à lui contient la description des impacts et leur(s) contre-mesures.

ID	LIBELLÉ	DESCRIPTION	CATÉGORIE	PROBABILITÉ	RESPONSABLE
R1	Communication	Le prestataire de services peux ne pas être géographiquement proche de ses clients, ce qui peut engendrer des difficultés de communication.	OPÉRATIONNEL	PROBABLE	RESPONSABLE DE PROJET
R2	Définion du besoin	Le besoin du client change en cours de projet.	STRUCTURANT	PEU PROBABLE	RESPONSABLE DE PROJET
R3	Sécurité des données	Cela concerne la sécurité des données et des applications inhérentes. Il est important de veiller à ce que le système actuel soit conforme aux normes et règles de sécurité en vigueur.	SÉCURITAIRE	PEU PROBABLE	RÉFÉRENT TECHNIQUE
R4	Tâches imprévues.	Des tâches imprévues surviennent lors du déroulement du projet.	STRUCTURANT	POSSIBLE	CLIENT / RESPONSABLE DE PROJET
R5	Gestion des risques	Si certains risques sont mal voire pas gérés, les délais et les coûts peuvent augmenter, et mettre en danger le projet.	BUDGETAIRE	PEU PROBABLE	RESPONSABLE DE PROJET
R6	Blocage implémentation applicatif	Il peut y avoir des difficultés à implémenter certaines fonctionnalités, ex : problème technique ou de configuration, …	TECHNIQUE	POSSIBLE	RÉFÉRENT TECHNIQUE
R7	Blocage implémentation données	Il peut y avoir des difficultés à implémenter les structures et les données : revue de schéma interne des bases de données existantes, problème technique ou de configuration, ...	TECHNIQUE	POSSIBLE	RÉFÉRENT TECHNIQUE
R8	Revue des délais / coûts par le client	Un changement interne de politique ou d'orientation, peu modifier le budget consacré au projet, ainsi que les délais, en les réduisant.	STRUCTURANT	POSSIBLE	CLIENT
R9	Roadmap inadaptée	La roadmap proposée n'est pas réaliste.	STRUCTURANT	PEU PROBABLE	RESPONSABLE DE PROJET
R10	Solution inadaptée	La solution proposée n'est pas réaliste ou ne répond pas totalement aux besoins / attentes du client.	TECHNIQUE	PEU PROBABLE	RÉFÉRENT TECHNIQUE
R11	Absence non prévue équipe du projet.	Tout ou partie de l'équipe projet est fréquemment absente et/ou sur de longues périodes.	OPÉRATIONNEL	POSSIBLE	RESPONSABLE DE PROJET
R12	Perte de motivation équipe du projet.	Tout ou partie de l'équipe projet n'est plus motivée pour travailler sur le projet.	OPÉRATIONNEL	PEU PROBABLE	RESPONSABLE DE PROJET
R13	Conflits entre les membres de l'équipe.	Tout ou partie de l'équipe projet n'arrive pas ou plus à s'entendre, pour travailler ensemble.	OPÉRATIONNEL	PEU PROBABLE	RESPONSABLE DE PROJET
R14	Perte de membres de l'équipe du projet.	Dans le contexte actuel du marché de l'emploi, qui est très porteur, certains membres de l'équipe peuvent quitter le projet en cours de route.	STRUCTURANT	POSSIBLE	RESPONSABLE DE PROJET

ID	LIBELLÉ	IMPACT(S)	EFFET	ATTÉNUATION - CONTRE-MESURES
R1	Communication	Retard du projet : -> Augmentation en conséquence des délais. -> Augmention en conséquence des coûts.	CRITIQUE	-> S'assurer du bon fonctionnement du réseau pour les web-conf. -> Planifier suffisamment en avance les réunions, mettre en place une récurrence. -> Faire des points en présence physique, une fois chez le client, une fois chez le fournisseur. -> Soliciter le fournisseur sur son rôle de conseil.
R2	Définion du besoin	Risque de tâches imprévues : -> Augmentation en conséquence des délais. -> Augmention en conséquence des coûts.	CRITIQUE	-> Bien assurer un rôle de conseil. -> Ne pas hésiter à plannifier des points supplémentaires sur le besoin. -> Être transparent sur la compréhension du besoin avec le client et traiter le sujet avec lui.
R3	Sécurité des données	Surcoûts globale pour l'entreprise : -> Pertes de client(e)s. -> Pertes de CA.	CRITIQUE	-> Veiller à ce que l'ensemble des applications / données respectent les politiques / normes de sécurité en vigueur.
R4	Tâches imprévues.	Rallongement du projet : -> Augmentation en conséquence des délais. -> Augmention en conséquence des coûts.	MARGINAL	-> Intégrer au projet uniquement les fonctionnalités n'engendrant pas ou peu de coûts et de délais supplémentaires, négociées avec le client. -> Renvoyer les autres fonctionnalités à un projet ultérieure.
R5	Gestion des risques	Retard voire arrêt du projet. -> Augmentation en conséquence des délais. -> Augmention en conséquence des coûts.	CATASTROPHIQUE	-> Bien assurer le suivi lors des points. -> Augementer la fréquences des points de suivi. -> Avoir une communication transparente vis-à-vis du client et des membres de l'équipe. -> Induire la transparence à tous les membres de l'équipes pour une bonne communication. -> Ne pas hésiter à revoir la priorité des risques, pour s'adapter à la réalite lors de l'évolution du projet.
R6	Blocage implémentation applicatif	Retard du projet et augmentation en conséquence des délais.	CRITIQUE	-> Induire la transparence à tous les membres de l'équipe ; id. en cas de point dur il faut lever une alerte au Responsable de Projet. -> S'appuyer sur le référent technique pour aider à résoudre les points durs.
R7	Blocage implémentation données	Retard du projet et augmentation en conséquence des délais.	CRITIQUE	-> Induire la transparence à tous les membres de l'équipe ; id. en cas de point dur il faut lever une alerte au Responsable de Projet. -> S'appuyer sur le Référent Technique pour aider à résoudre les points durs.
R8	Revue des délais / coûts par le client	Rallongement ou arrêt du projet : -> Augmentation en conséquence des délais. -> Augmention en conséquence des coûts.	CATASTROPHIQUE	-> Le client devra en informer le Responsable de Projet.
R9	Roadmap inadaptée	Retard voire arrêt du projet : -> Augmentation en conséquence des délais. -> Augmention en conséquence des coûts.	CATASTROPHIQUE	-> Valider la roadmap avec toutes les parties prenantes. -> Redéfinir si nécessaire la roadmap avec toutes les parties prenantes.
R10	Solution inadaptée	Retard voire arrêt du projet : -> Augmentation en conséquence des délais. -> Augmention en conséquence des coûts.	CATASTROPHIQUE	-> Valider la solution avec toutes les parties prenantes. -> Redéfinir si nécessaire la solution avec toutes les parties prenantes.
R11	Absence non prévue équipe du projet.	Retard du projet : -> Augmentation en conséquence des délais.	CRITIQUE	-> Induire la transparence à tous les membres de l'équipes pour une bonne communication ; chaque membre de l'équipe avertira dans les meilleurs délai le Chef de Projet. -> Organiser dès le départ du projet, une politique de remplacement pour chacun des rôles, afin de pourvoir en cas de besoin à l'exécution des remplacements dans les meilleurs délais.
R12	Perte de motivation équipe du projet.	Retard du projet : -> Augmentation en conséquence des délais.	CRITIQUE	-> Induire la transparence à tous les membres de l'équipe ; id. il faut lever une alerte au Chef de Projet dès les premiers soupçons. -> Dès le départ du projet, bien définir et explicité les rôles et responsabilité de chacun, ainsi que les objectifs à atteindre, à tous les membres de l'équipe du projet. -> Impliquer tous les membres dans le projet, ne laisser personne isolée / exclue. -> Ne pas imposer des changements de cap, de rôle, de responsabilité sans en discuter avec les parties prenantes. -> Préférer la négociation et l'obtention de l'approbation avec les parties prenantes ; trouver des solutions ensembles.
R13	Conflits entre les membres de l'équipe.	Retard du projet : -> Augmentation en conséquence des délais.	CRITIQUE	-> Induire la transparence à tous les membres de l'équipe ; id. il faut lever une alerte au Responsable de Projet dès les premières friction. -> Ne pas hésiter à faire des points de conciliation. -> En cas de conflit avéré, réorganiser l'équipe, ou effectuer des changements de membres (*sortie du projet*).
R14	Perte de membres de l'équipe du projet.	Retard du projet : -> Augmentation en conséquence des délais.	CRITIQUE	-> Prévoir ou rendre prévisible les sorties ; éviter les sorties soudaines. -> Prévoir des membres remplaçant dans le "pool" des consultants en inter-mission. -> Prévoir des primes ou augmentation de salaire en fonction des profils.

3.5. Faisabilité du projet

Une étude de faisabilité devra être réalisée par le responsable du projet, dans le but de de définir les options du projet d'une part, et d'autre part de s'assurer de la viabilité de celui-ci.
Cette étude de faisabilité s'appuie sur la connaissance des éléments suivants :

- la prise en compte des besoins et de toutes les contraintes, et notamment le budget et le délai ;
- les conditions requises pour le business de haut niveau ;
- la vision architecturale ;
- l'identification et organisation des tâches à réaliser ;
- l'identification des ressources projet ;
- l'identification et la mesure des risques ;
- l'étude de la viabilité et faisabilité du projet.

4. Cycle de vie architectural

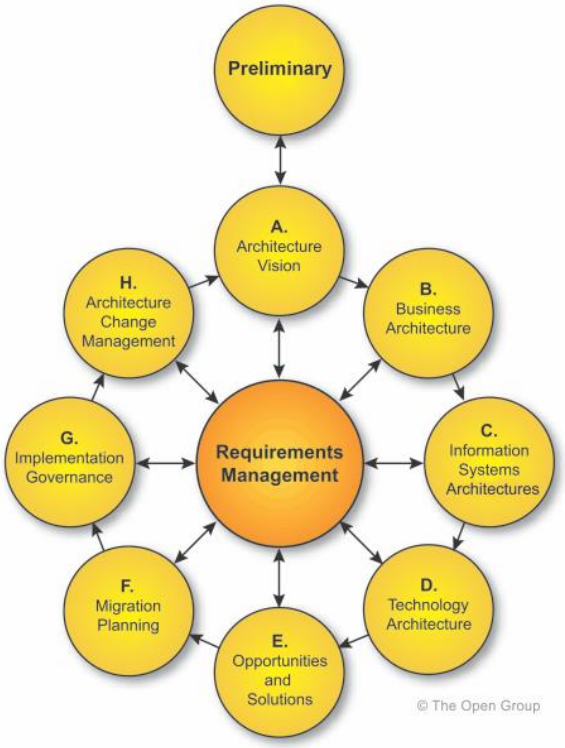
4.1. Méthodologie

Pour gérer la transformation architecturale de la nouvelle plateforme de e-commerce, nous allons nous appuyer sur le framework TOGAF.
En effet, celui-ci offre un cadre de progression organisé, et une méthode reconnue regroupant un ensemble de techniques centrées sur la transformation de l'architecture d'entreprise, et couvrant toutes les facettes de cette dernière. La méthodologie TOGAF est toute à la fois générique et pragmatique, c'est une boîte à outils au service de tous les acteurs de l'architecture d'entreprise.
Répondant à la préoccupation majeure de l'alignement entre le business et la technique, des directions métier et des DSI, toujours à la recherche de l'agilité SI, elle est très largement adoptée aujourd'hui pour la transformation des architectures d'entreprise. Aussi, nous permettra-t-elle d'atteindre notre objectif dans les meilleures conditions.
De plus à l'instar de Lean et Scrum, la méthode TOGAF est également basée sur un processus itératif. De ce fait, elle pourra facilement s'intégrer au processus Lean IT en y apportant spécifiquement la partie méthodologie pour assurer la transformation architecturale.

4.2. Cycle ADM

Le cycle ADM pour Architecture Development Method, ou méthode de développement d'architecture, est une démarche cyclique et itérative visant à transformer l'architecture d'entreprise, au cœur du framework TOGAF. Elle est découpée en huit phases séquentielles de A à H, auxquelles s'ajoutent deux autres particulières que sont les phases préliminaires et de gestion. La première initie le projet de transformation de l'architecture d'entreprise. Tandis que la seconde de par sa position centrale, vis-à-vis des huit phases, témoigne ainsi de son rôle pivot, et par conséquent de la préoccupations des exigences tout au long du cycle ADM.

Le cycle ADM est représenté par la « roue TOGAF » ci-dessous :



[1]

À noter que même si l'enchaînement est décrit de manière séquentielle, celui-ci peut être revu et adapté en fonction du contexte.

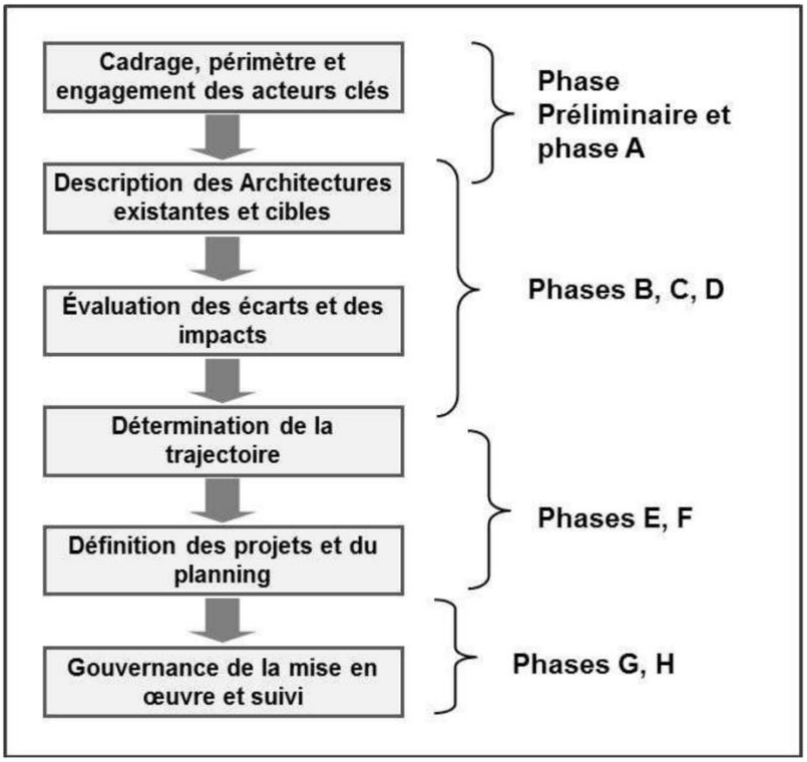
[1] Schéma tiré du document de référence TOGAF disponible sur le site Web « https://pubs.opengroup.org/architecture/togaf91-doc/arch/ ».

4.2.1. Phases ADM

Les différentes phases susmentionnées du cycle ADM se décrivent comme suit :

PHASE	LIBELLÉ	DESCRIPTION
Pr	Preliminary Préliminaire	Cette phase, de nature transverse et ne faisant pas partie d'un cycle ADM, a pour but de préparer l'entreprise à la réalisation des travaux d'architecture : -> organisation et la gouvernance de l'architecture, -> les principes généraux, -> Les méthodes, -> les outils, -> le référentiel d'architecture, -> le déclenchement de cycle ADM.
RqtMgt	Requierments Management Gestion de exigences	Il s'agit de l'ensemble des exigences déterminant ce qui doit être réellement mis en oeuvre, et inversement ce que l'on ne retient pas. En effet, à partir d'objectifs métier donnés, les exigences concrètes traduisent en général la prise en compte de plusieurs facteurs, à la fois techniques, budgétaires ou organisationnels. Cette étape intervient tout au long du cycle ADM, et n'étant pas figée elle donc peut évoluer à tout moment.
A	Architecture Vision Vision architecturale	Déclenchée par la validation du document de demande de mise en chantier de l'architecture, elle a deux objectifs essentiels : -> préciser et enrichir les éléments issus de la phase préliminaire, comme les principes d'architecture, les indicateurs clé, l'organisation ou le planning des travaux d'élaboration ; -> préparer les phases suivantes sous la forme d'une représentation générale des architectures initiales et cibles.
B	Business Architecture Architecture métier	Au niveau de la couche métier, détaille l'architecture initiale et cible, en mesure l'écart, puis évalue les impacts des évolutions sur l'ensemble des facettes de l'entreprise. Ces éléments permettront d'établir la feuille de route de transition, qui s'élabore progressivement au cours des phases B, C et D, et ce afin de constituer les matériaux de base des phases E et F, chargées de définir le plan de transformation.
C	Information Systems Architectures Architecture du système d'information	Au niveau de la couche SI (*applicative + données*), détaille l'architecture initiale et cible, en mesure l'écart, puis évalue les impacts des évolutions sur l'ensemble des facettes de l'entreprise. Ces éléments permettront d'établir la feuille de route de transition, qui s'élabore progressivement au cours des phases B, C et D, et ce afin de constituer les matériaux de base des phases E et F, chargées de définir le plan de transformation.
D	Technology Architecture Architecture technique	Au niveau de la couche technique, détaille l'architecture initiale et cible, en mesure l'écart, puis évalue les impacts des évolutions sur l'ensemble des facettes de l'entreprise. Ces éléments permettront d'établir la feuille de route de transition, qui s'élabore progressivement au cours des phases B, C et D, et ce afin de constituer les matériaux de base des phases E et F, chargées de définir le plan de transformation.
E	Opportunities and Solutions Solutions et opportunités	Consolide les résultats des phases d'élaboration B, C et D pour la définition des architectures de transition en tenant comptes des capacités de changement de l'entreprise.
F	Migration Planning Planning de migration	Établit précisément le planning de migration, la constitution des projets de mise en oeuvre, leur organisation, les objectifs et les coûts.
G	Implementation Governance Gouvernance de la mise en œuvre	Établit la version définitive des contrats d'architecture avec les projets d'implémentation, qui inclut les recommandations du comité d'architecture. Ces contrats signés fourniront le matériel de base pour les revues de conformité, menés auprès des projets de mise en oeuvre.
H	Architecture Change Management Gestion de la maintenance et des évolutions	Prend en charge la gestion de l'architecture déployée : la gestion des modifications, qui comprend l'évaluation des demandes de changements impactant l'architecture.

Ci-dessous une représentation du parcours type d'un cycle ADM :



[2]

4.2.2. Livrables

Les livrables sont produits tout au long du cycle ADM, et leur approbation par les parties prenantes établit un consensus formel, et fixe un état des résultats à partir desquels les travaux suivants pourront s'exécuter. Pour autant, cela ne signifie pas que chaque livrable est fourni et validé par une phase unique. Certains d'entre eux sont produits au cours de plusieurs phases, c'est le cas par exemple du « Document de définition de l'architecture » dont les parties sont élaborées au cours des phases B, C et D.
Le standard TOGAF propose une base de référence constituée de 22 livrables.
Sur la page suivante, se trouve la liste des livrables majeurs correspondant à chacune des phases du cycle de l'ADM.

[2] Schéma tiré de l'ouvrage TOGAF, Archimate, UML et BPMN - 3e éd. (Management des systèmes d'information) de Philippe DESFRAY et Gilbert RAYMOND aux éditions Dunod.

ID	LIVRABLE	DESCRIPTION	PHASE(S) ADM
L01	Request for Architecture Work Demande de mise en chantier d'architecture	Issue de la phase préliminaire, il déclenche le démarrage d'un nouveau cycle ADM. Notons que la réponse peut être négative, et que l'entreprise peut décider de ne pas lancer le cycle ADM de transformation d'architecture (go no go).	Pr
L02	Architecture Principles Principes d'architecture	Définis en phase préliminaire, ils établissent les principes généraux d'architecture applicables à tout cycle ADM.	Pr
L03	Tailored Architecture Framework Cadre d'architecture contextualisé	Élaboré en phase préliminaire, il joue un rôle particulier. Typiquement, il permet d'adapter le framework TOGAF au contexte de l'entreprise. C'est un des résultats de la phase préliminaire, qui initie les éléments à mettre en oeuvre par les chantiers de transformation sur les différents plans : démarche, contenus, référentiel et gouvernance.	Pr
L04	Business Principles, Business Goals, and Business Drivers Principes, objectifs et moteurs du métier	Ils précisent le contexte et les buts poursuivis par un cycle ADM. Initialisés en phase préliminaire, ils sont enrichis et consolidés en phases A et B (vision et métier).	Pr, A, B
L05	Architecture vision Vision de l'architecture	Initie la suite des travaux en fournissant une vue macroscopique et transverse : -> des objectifs, -> des exigences, -> de l'architecture initiale, -> de la cible.	A
L06	Statement of Architecture Work Définition du chantier d'architecture	Détaille tous les éléments nécessaires à l'organisation du cycle ADM, à partir de la demande de mise en chantier d'architecture : le management, les procédures, le planning du cycle, le périmètre.	A
L07	Communication plan Plan de communication	Également produit en phase A, il fournit le cadre de communication interne : moyens, outils, procédures.	A
L08	Architecture Definition Document Document de définition de l'architecture	Il est le livrable majeur des phases d'élaboration de l'architecture : B (métier), C (système), D (technique). Il contient notamment le détail de l'architecture (initiale et cible), l'analyse des écarts et l'évaluation des impacts.	B, C, D
L09	Architecture Requirements Specification Spécification des exigences d'architecture	Elle consigne l'ensemble des exigences.	B, C, D, E, F
L10	Architecture Roadmap Feuille de route	Issue des phases d'élaboration B, C et D établit la progression de la transition : la définition de chaque palier, le planning macroscopique. Ces éléments seront précisés par les deux livrables suivants, en phase E et F.	B, C, D, E, F
L11	Transition Architecture Architecture de transition	Détaille les différents paliers, la répartition en lots de travail avec leurs contenus, dépendances, et précise l'architecture attendue à chaque palier.	E, F
L12	Implementation and Migration Plan Plan de migration et de déploiement	Fournit le planning détaillé, les projets d'implémentation, les ressources, le budget.	E, F
L13	Architecture Contrat Contrat d'architecture	Ils formalisent les engagements des projets d'implémentation vis-à-vis du comité d'architecture.	F
L14	Capability Assessment Évaluation des capacités	Consigne le résutat de l'évaluation démarrée en phase A, puis mise à jour en phase E.	A, E
L15	Compliance Assessment Évaluation de la conformité	Consigne le résultat des revues de conformité.	G
L16	Change Request Demande de modification	Une fois la nouvelle architecture déployée en phase H, des demandes de modification pourront être transmises et évaluées par le comité d'architecture.	H

Ci-dessous les livrables regroupés par thème / partie du cycle ADM :

THÈME	LIVRABLE(S)
Livrables liés à la gestion du chantier d'architecture	L01 : Demande de mise en chantier d'architecture L03 : Cadre d'architecture contextualisé L06 : Définition du chantier d'architecture L07 : Plan de communication
Livrables liés aux principes, objectifs et exigences.	L02 : Architecture Principles L04 : Principes, objectifs et moteurs du métier L09 : Spécification des exigences d'architecture
Livrables de description de l'architecture	L05 : Vision de l'architecture L08 : Document de définition de l'architecture
Livrables dédiés à la transition de l'architecture	L10 : Feuille de route L11 : Architecture de transition L12 : Plan de migration et de déploiement
Livrables liés à la mise en œuvre	L13 : Contrat d'architecture L14 : Évaluation des capacités L15 : Évaluation de la conformité L16 : Demande de modification

5. Bonnes pratiques

Cette section a pour but de décrire les principales bonnes pratiques à appliquer.
En effet, il est important d'avoir et de suivre de bonnes pratiques tant sur la gestion de projet, que la transformation de l'architecture d'entreprise, ainsi que la réalisation / implémentation de cette dernière, id. le développement en lui-même ; le code. Le tout sans oublier, bien entendu, les bonnes pratiques au niveau de l'entreprise elle-même, incluant les parties susmentionnées, et sans quoi la mise en place de bonnes pratiques n'aurait pas beaucoup de valeur, ni de sens d'ailleurs.
Le but premier est bien évidemment de garder sous contrôle toutes les activités, pour
les garder sur les rails afin d'aller dans la bonne direction jusqu'à l'atteinte des objectifs. Mais aussi de le faire dans les meilleures conditions, id. sans revenir de manière récurrente, sur le travail déjà effectué avec les conséquences temporelles et financières inhérentes.
Et enfin, de permettre d'assurer la compréhension et la maintenance de toute réalisation.

5.1. Entreprise

Respecter la démarche de type Lean de Foosus, et notamment par :

- le respect des cinq principes Lean, et notamment la mise en œuvre d'un dynamique d'amélioration continue pérenne ;
- l'assurance qualité, afin de répondre aux objectifs de la société, et ayant pour finalité la satisfaction client ;
- la définition d'objectifs SMART pour : Spécifique, Mesurable, Atteignable, Réaliste et Temporel ;
- la compréhension des objectifs de la société et leur adhésion.
- l'implication, sur chacun des sujets, de toutes les parties prenantes ;
- la mise en place de processus d'entreprise sur toutes les chaines de valeur de celle-ci, et ce afin de supporter l'amélioration continue.
- le suivi et la mesure de la performance desdits processus par mise en place d'indicateurs inhérents ;

5.2. Métier

Pour que l'architecture SI (*applicative & données*) puisse être optimale, le métier doit avoir un référentiel complet, correspondant à son architecture à date. Ce référentiel doit contenir l'ensemble des objets manipulés, par qui, et dans quel cadre, et ce pour chacun des processus métier. Sans quoi, il sera d'une part, difficile de répondre de manière optimale aux besoins du métier. Et d'autre part, de mesurer la pertinence des changements demandés.

5.3. Projet

La gestion de projet permet de cadrer et d'assurer l'avancement de la réalisation jusqu'à l'atteinte des objectifs. Aussi est-il important de respecter les bonnes pratiques suivantes :

- Les responsabilités de chaque intervenant doivent être claires, et par conséquent de bien définir les rôles et leur répartition entre :
  - le Product Owner, qui doit assurer le suivi du projet ;
  - l'équipe de développement (DevTeam) devant assurer la réalisation ;
  - le Scrum Master, comme référent et arbitre, garant des bonnes pratiques Scrum ;
  - le Business Owner, représentant le client, les aspects business.
- Adhérer aux valeurs Scrum :
  - Focus
  - Ouverture
  - Respect
  - Transparence
  - Engagement
- Comprendre et respecter les 12 principes généraux du manifeste agile :
  1. Satisfaire le client en priorité.
  2. Accueillir favorablement les demandes de changement.
  3. Livrer le plus souvent possibles des versions opérationnelles de l'application.
  4. Assurer une coopération permanente entre le client et l'équipe projet.
  5. Construire des projets autour d'individus motivés.
  6. Privilégier la conversation en face à face.
  7. Mesurer l'avancement du projet en termes de fonctionnalités de l'application.
  8. Faire avancer le projet à un rythme soutenable et constant.
  9. Porter une attention continue à l'excellence technique et à la conception.
  10. Faire simple.
  11. Responsabiliser les équipes.
  12. Ajuster à intervalles réguliers son comportement et ses processus pour être plus efficace.
- Comprendre et respecter les piliers Srcum :
  - Transparence
  - Adaptation
  - Inspection
- Comprendre et respecter les principes Srcum :
  - Processus empirique (*observation et expérimentation*).
  - Auto-organisation.
  - Collaboration.
  - Priorisation basée sur la valeur (*le résultat apporté au client*).
  - Délimité dans le temps.
  - Développement itératif.

5.4. Architecture

Ci-dessous les bonnes pratiques pour l'architecture SI (*applicative & données*).

5.4.1. Critères de référence

Ci-dessous les critères définissant une architecture de référence, et sur lesquels l'évaluation architecturale a été basée :

- Évolutivité
- Simplicité
- Maintenabilité
- Compatibilité
- Interconnectivité

À ces critères s'ajoutent, dans le cadre des bonnes pratiques, la documentation architecturale de référence correspondant à l'architecture à date.

À noter qu'à l'instar du codage, on ne documente pas pour soit, mais bien pour communiquer aux autres parties prenantes présentes et futures.

5.4.2. Production du logiciel

Pour rappel, l'Architecte Logiciel n'est pas seulement garant de la vision du système, de la définition des standards, des bonnes pratiques, … mais également de l'implémentation technique. De facto, il est important d'une part qu'il ne s'enferme pas dans la conception, on parle dans ce cas de syndrome de la tour d'ivoire, mais qu'il tienne bien compte de la technique, id. la capacité à pouvoir implémenter la solution qu'il aura conçu.

Et d'autre part, il lui est important de considérer également la ligne de production du logiciel, afin que celle-ci soit en adéquation avec les attentes implicites du métier, ici en l'occurrence la capacité de faire évoluer rapidement et régulièrement la plateforme, en somme le "leitmotiv" de beaucoup d'entreprises.

Aussi pour ce faire, la ligne de production doit posséder les caractéristiques suivantes ayant la capacité de :

1. Versionner chaque évolution de code tout en tenant compte de la charte de codage définie, pour éviter d'intégrer du code dit « pourri » qui ne respecterait ce référentiel, et notamment la documentation du code par les commentaires.
2. produire des tests unitaires automatisés **TU** ou **UT** pour **Unit Tests** ;
3. produire des tests d'intégration / non-régression automatisés : **build continu** ou **CI** pour **Continue Integration** ;
4. déployer à la demande une nouvelle version en production de manière automatisée : **déploiement continu** ou **CD** pour **Continue Deployment**.

Le marché et notamment l'Open source propose des solutions telles que :

- Git pour versionner du code.
- GitLab pour l'intégration et la livraison continue, ainsi que le suivi des bugs.
- Jenkins pour également l'intégration et la livraison continue.
- JUnit, NUnit et TestNG (Java) pour les tests unitaires.
  À noter que les tests unitaires sont intimement liés au langage de programmation utilisé, et qu'il en existe également pour les langages les plus courant / connu : Python, PHP, .NET, C, C++, Delphi, Ada, PL/SQL, etc.

5.5. Développement

5.5.1. Conception

Dans le cadre de tout développement informatique, afin de faciliter la maintenance et les projets futurs, il est important de respecter les principes :

- **KISS** pour Keep It Stupid Simple ou gardez ça super simple, préconisant la simplicité dans la conception, et d'éviter toute complexité inutile.
- **DRY** pour Don't Repeat Yourself ou ne vous répétez pas, philosophie à la fois de conception et de codage consistant à éviter la répétition de code.

En ce qui concerne le développement informatique objet, il est important de respecter les principes SOLID :

- **S** pour Single responsibility principle ou responsabilité unique.
- **O** pour Open / closed principle ou ouvert / fermé.
- **L** pour Liskov substitution principle ou principe de substitution de Liskov.
- **I** pour Interface segregation principle ou principe de ségrégation des interfaces.
- **D** pour Dependency inversion principle ou principe d'inversion des dépendances.

En complément des principes **DRY** et **SOLID**, la règle de trois de **Martin Flower**, qui consiste en une refactorisation / réorganisation d'une même instance de code au bout de la troisième duplication. Cela a pour but d'éviter une refactorisation, prématurée et incessante, qui n'est pas toujours pertinente / nécessaire et pouvant mener à des problèmes de fiabilité et de maintenabilité. Aussi vient-elle contrebalancer la philosophie DRY susmentionnée, qui mal utilisée pourrait laisser croire à une nécessité de refactoriser en permanence des instances de codes.

5.5.2. Réalisation

En plus de la conception et pour les mêmes raisons, il est important de respecter les principes de codage suivants lors de la réalisation du développement :

- Organiser
- Simplifier
- Réutiliser
- Réorganiser / optimiser
- Commenter & documenter
- Versionner

De plus, il est important de choisir une charte de codage pertinente, et à laquelle se tenir / respecter. On doit y retrouver entre autres les conventions pour :

- le nommage des variables / constantes, fonctions / procédures, fichiers, classes, … ;
- les commentaires pour les éléments susmentionnés ;
- l'organisation rationnelle de la réalisation, id. code, répertoires, fichiers, packages, …

Et pour finir, il est important de rappeler que l'on ne code pas pour soit mais bien pour l'équipe / l'entreprise.

5.6. Opérationnel

Dans le cadre de la bonne exploitation du système, mais aussi à chaque modification de celui-ci, il est important de prévoir :

- la supervision et la métrologie ;
- la sauvegarde des données ;
- le plan de reprise de sur panne.

5.7. Documentation

Que ce soit pour la conception ou la réalisation, il est important de documenter pour :

- gagner du temps et créer de la cohérence ;
- donner de la lisibilité ;
- assurer la maintenabilité ;
- diminuer le coût de ladite documentation ;
- éviter la dette technique et documentaire.

Pour ce faire, il est important de garder à l'esprit plusieurs choses comme :

1. À quelle(s) partie(s) prenante(s) s'adresse la documentation ;
2. Trouver un compromis en termes de qualité et de quantité.
3. Maintenir continuellement à jour la documentation, id. en même temps qu'évolue le code et le système ;
4. Utiliser / formaliser via des outils standards (ex : UML).

6. Mission architecturale

6.1. Situation à date

À l'heure actuelle l'architecture logicielle de l'entreprise est composée d'un ensemble hétérogène de technologies ne lui permettant pas d'évoluer avec la société en fonction de ses besoins, d'une fait une dette technique. Aussi, l'architecture actuelle entrave la croissance de la société, qui ne peut innover dans ses produits, et ainsi attirer de nouveaux utilisateurs.
En effet, le système, dans sa partie logicielle, est en incapacité d'évoluer au même rythme que la base clientèle de la société, id. d'absorber les pics d'utilisation, et ce faisant, cela crée des pannes, et donc des interruptions de service.

6.2. Problématique & impacts

6.2.1. Mesure

La mesure des impacts vise à déterminer les causes à l'origine de celles-ci. Aussi pour ce faire, la méthode des « 5 pourquoi » a été utilisée.

Taux d'inscriptions utilisateurs #1 :

1. La société ne peut attirer de nouveaux utilisateurs ! Pourquoi ?
2. La société ne peut innover ! Pourquoi ?
3. L'architecture entrave la croissance de la société ! Pourquoi ?
4. L'architecture logicielle est technologiquement hétérogène ! Pourquoi ?
5. L'ancienne DSI a laissé l'équipe technique expérimenter de nouvelles approches techniques sans cadre standard ni vision à long terme. ➔ **cause**.

Taux d'inscriptions utilisateurs #2 :

1. La société ne peut attirer de nouveaux utilisateurs ! Pourquoi ?
2. La correspondance de la société avec le marché a été éclipsée ! Pourquoi ?
3. L'image de marque auprès des clients / utilisateurs est écornée ! Pourquoi ?
4. Des pannes créent des interruptions de service de la plateforme ! Pourquoi ?
5. L'architecture logicielle est technologiquement hétérogène ! Pourquoi ?
6. L'ancienne DSI a laissé l'équipe technique expérimenter de nouvelles approches techniques sans cadre standard ni vision à long terme. ➔ **cause**.

Alignement commercial & technique :

1. Pas d'alignement entre la partie commerciale et la partie technique ! Pourquoi ?
2. La plateforme a naturellement évolué vers la complexité avec une architecture hétérogène, induisant de facto un manque de clarté et donc de compréhension de cette dernière ! Pourquoi ?
3. L'ancienne DSI a laissé l'équipe technique expérimenter de nouvelles approches techniques sans cadre standard ni vision à long terme. ➔ **cause**.

Compétitivité concurrentielle :

1. Les concurrents directs prennent l'avantage, des parts de marché, sur la société ! Pourquoi ?
2. Contrairement à ses concurrents, la société n'arrive pas à s'adapter à la demande du marché, aux attentes des clients / utilisateurs ! Pourquoi ?
3. L'architecture peut difficilement évoluée / s'étendre ! Pourquoi ?
4. L'architecture est verrouillée par des solutions passé, accusant de fait une dette technique importante ! Pourquoi ?
5. L'ancienne DSI a laissé l'équipe technique expérimenter de nouvelles approches techniques sans cadre standard ni vision à long terme. ➔ **cause**.

Alignement technico-commercial de la performance :

1. La vision de l'intelligence business, id. l'alignement entre la performance technique et commerciale n'est pas clair ! Pourquoi ?
2. La recherche de l'intelligence business nécessite au préalable une lourde étape d'analyse de données ! Pourquoi ?
3. Les données à analyser nécessitent de passer par des outils tiers (*registres et feuilles de calcul*) ! Pourquoi ?
4. La plateforme n'offre pas de moyens de supervision à la fois sur sa performance technique que sur sa performance business ! ➔ **cause**.

La réputation de la marque #1 :

1. Les livraisons créent des interruptions de service de la plateforme, impactant les utilisateurs ! Pourquoi ?
2. Incapacité de livrer de nouvelle versions / évolutions de la plateforme sans interruptions de service ! Pourquoi ?
3. Pas de processus qualité quant auxdites livraisons ! ➔ **cause**.

La réputation de la marque #2 :

1. Des pannes créent des interruptions de service de la plateforme, impactant les utilisateurs ! Pourquoi ?
2. Le système logiciel est en incapacité d'absorber les pics d'utilisation ! Pourquoi ?
3. L'architecture logicielle est technologiquement hétérogène ! Pourquoi ?
4. L'ancienne DSI a laissé l'équipe technique expérimenter de nouvelles approches techniques sans cadre standard ni vision à long terme ! ➔ **cause**.

6.2.2. Causes

Organisationnelles :

- Pas de cadrage quant à libre expérimentation de nouvelles approches techniques, par l'équipe de la précédente DSI.
- Pas de vision à long terme aussi bien sur les aspects techniques que business, id. pas d'alignement entre ces deux parties indissociables.

Techniques :

- Architecture logicielle composée d'un ensemble hétérogène de technologies ; id. pas de standardisation permettant de déterminer une architecture de référence (*voir plus haut les bonnes pratiques à la section « **5.4. Architecture** »*).
- Incapacité du système logiciel à absorber les pics d'utilisation.

Opérationnelles :

- Pas de processus qualité quant aux livraisons logicielles de la plateforme.
- Pas de plan de reprise sur panne.
- Pas de supervision de la plateforme.
- Pas d'outils d'analyse et de mesure business.

6.2.3. Conséquences

- L'architecture actuelle entrave la croissance de la société, qui ne peut innover dans ses produits, et ainsi attirer de nouveaux utilisateurs.
- Pannes créant des interruptions de service, écornant l'image de la société et donc la confiance des utilisateurs.

6.3. Objectifs

Standardiser l'architecture par la mise en place d'une nouvelle permettant :

1. d'effacer la dette technique et ne plus en avoir à l'avenir ;
2. d'évoluer avec l'entreprise, id. soutenir sa croissance de manière continue ;
3. de prendre en charge un nouvel emplacement géographique, afin de tirer parti de la géolocalisation pour relier fournisseurs et consommateurs, pour proposer des produits disponibles à proximité des lieux de résidence de ces derniers ;
4. de faciliter la maintenance et les développements futurs (maitrise des coûts et des délais) ;
5. de fournir la fiabilité nécessaire aux clients, fournisseurs et consommateurs de la Société.

L'objectif à terme pour la Société est de créer une plateforme de commerce électronique polyvalente lui permettant de franchir le million d'utilisateurs, et de concurrencer ainsi les grandes entreprises mondiales de commerce électronique qui dominent le marché de l'alimentation durable.

6.4. Contraintes

Le besoin s'accompagne bien évidemment de différents types de contraintes définies ci-dessous pour l'atteinte des objectifs, et donc de la réalisation la nouvelle plateforme e-commerce de Foosus, et à commencer par son architecture.

6.4.1. Fonctionnelles

IDENTIFIANT	INTITULÉ
CTR-FCT-1	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.
CTR-FCT-2	Prise en compte des catégories d'utilisateurs (fournisseurs, consommateurs, …) et des fonctionnalités et services associés (spécifiques).
CTR-FCT-3	La future plateforme doit inclure la possibilité pour les uilisateurs internes de mettre en place des indicateurs : -> de suivi pour le pilotage des processus ; -> de performance pour s'assurer que les processus fournissent bien les résultats attendus.

6.4.2. Techniques

IDENTIFIANT	INTITULÉ
CTR-TECH-1	La nouvelle architecture, découlant de la nouvelle plateforme de commerce électronique, doit être évolutive (scalable).
CTR-TECH-2	Prise en charge des ordinateurs fixes et portables.
CTR-TECH-3	Prise en charge des terminaux mobiles IOS (Apple) : smartphones et tablettes.
CTR-TECH-4	Prise en charge des terminaux mobiles Android (Google) : smartphones et tablettes.
CTR-TECH-5	Prise en charge des environnements OS (ordinateurs) : -> Windows, -> Mac, -> Linux.
CTR-TECH-6	Prise en charge des environnements OS (terminaux) : -> Android, -> IOS.
CTR-TECH-7	Prise en charge des navigateurs Web : -> Edge (Microsoft, -> Chrome (Google), -> Firefox (Mozilla), -> Safari (Apple).
CTR-TECH-8	Prise en compte des bandes passantes : réseaux fixe et mobile.
CTR-TECH-9	Inclure un calculateur de distance, via la géolocalisation, pour permettre aux consommateurs de trouver les fournisseurs les plus proches d'eux
CTR-TECH-10	Possibilité d'inverser les décisions d'architecture tant que cela reste peu onéreux.
CTR-TECH-11	Toute nouvelle conception de backlog doit cependant tenir compte des éléments suivants : -> Emplacement des offres alimentaires proposées par les fournisseurs. -> Proximité de l'utilisateur effectuant la recherche en cours. -> Visualisation des informations statistiques secondaires et sectorielles relatives au produit alimentaire concerné. Par exemple, détails sur son indice glycémique.
CTR-TECH-12	La future plateforme doit supporter quotidiennement une augmentation d'adhésions utilisateurs de 10%.
CTR-TECH-13	La future plateforme doit s'exécuter dans un environnement Cloud.

6.4.3. Opérationnelles

IDENTIFIANT	INTITULÉ
CTR-TECH-1	La nouvelle architecture, découlant de la nouvelle plateforme de commerce électronique, doit être évolutive (scalable).
CTR-TECH-2	Prise en charge des ordinateurs fixes et portables.
CTR-TECH-3	Prise en charge des terminaux mobiles IOS (Apple) : smartphones et tablettes.
CTR-TECH-4	Prise en charge des terminaux mobiles Android (Google) : smartphones et tablettes.
CTR-TECH-5	Prise en charge des environnements OS (ordinateurs) : -> Windows, -> Mac, -> Linux.
CTR-TECH-6	Prise en charge des environnements OS (terminaux) : -> Android, -> IOS.
CTR-TECH-7	Prise en charge des navigateurs Web : -> Edge (Microsoft, -> Chrome (Google), -> Firefox (Mozilla), -> Safari (Apple).
CTR-TECH-8	Prise en compte des bandes passantes : réseaux fixe et mobile.
CTR-TECH-9	Inclure un calculateur de distance, via la géolocalisation, pour permettre aux consommateurs de trouver les fournisseurs les plus proches d'eux
CTR-TECH-10	Possibilité d'inverser les décisions d'architecture tant que cela reste peu onéreux.
CTR-TECH-11	Toute nouvelle conception de backlog doit cependant tenir compte des éléments suivants : -> Emplacement des offres alimentaires proposées par les fournisseurs. -> Proximité de l'utilisateur effectuant la recherche en cours. -> Visualisation des informations statistiques secondaires et sectorielles relatives au produit alimentaire concerné. Par exemple, détails sur son indice glycémique.
CTR-TECH-12	La future plateforme doit supporter quotidiennement une augmentation d'adhésions utilisateurs de 10%.
CTR-TECH-13	La future plateforme doit s'exécuter dans un environnement Cloud.

6.5. Hypothèses de solutions

Au regard des problématiques de la section « **2.2. Problématiques & impacts** » vue plus haut, des pistes de travail et attentes à considérer.

6.5.1. Taux d'inscriptions utilisateurs

Pour pallier à la chute d'inscriptions, nécessaires pour rassurer les investisseurs de la société :

- Ajouter une fonctionnalité de géociblage pour toucher une gamme plus large d'utilisateurs.
- Définir une architecture se mettant à l'échelle en fonction de la base clients.

6.5.2. Alignement commercial & technique

Pour pallier à la complexité de la plateforme et soutenir la croissance :

- Garder impliqué les équipes techniques et les guider vers une standardisation architecturale.
- Définir un périmètre architectural clair et concis.

6.5.3. Compétitivité concurrentiel

Pour pallier la dette technique, le manque d'extensibilité (verrouillage) et de l'instabilité de l'architecture :

- Définir une architecture évolutive / extensible, et avec un degré de granularité pertinent / justifié.
- Mettre en place des bonnes pratiques à la fois architecturales et techniques (conception & développement).

6.5.4. Alignement technico-commercial de la performance

Pour pallier à la problématique de visibilité et du manque de clarté de l'intelligence business, id. l'alignement entre la performance technique et commercial :

- La nouvelle architecture doit permettre de fournir les informations utiles à l'analyse de l'intelligence business.
- La plateforme quant à elle doit offrir des outils permettant l'analyse de l'intelligence business, tel que par exemple la possibilité de produire à la demande des tableaux de bord, ou bien encore des KPI ou Key Performance Indicator (indicateurs clés de performance).

6.5.5. La réputation de la marque

Pour pallier au manque de qualité des livraisons, les interruptions dues à ces dernières ainsi qu'aux pannes :

- Mettre en place un processus de qualité des livraisons pour éviter les pannes et donc les interruptions de service.
- Définir une architecture permettant lors des livraisons :
  - d'éviter lesdites interruptions de service, ou tout du moins les minimiser ;
  - de revenir en arrière en cas de problème.
- Définir une architecture se mettant à l'échelle en fonction de la base clients.

7. Architecture cible

Cette section décrit la vision architecturale pour satisfaire l'état terminal recherché, au travers de quatre couches : métier, applicative, infrastructure et opérationnelle.

7.1. Couche Métier

La couche Métier représente la partie business et donc le cœur de métier, et la finalité que doit satisfaire le système informatique.

7.1.1. Périmètre du système

Ci-dessous ce qui est attendu du futur système (*la plateforme*) et ce qu'il ne fait pas.

Attendu du système :

- Référencer les fournisseurs.
- Enregistrer les clients.
- Gérer le catalogue fournisseurs et produits (*inventaire*).
- Rechercher :
  - des fournisseurs alimentaires ;
  - des produits alimentaires.
- Gérer les facturations :
  - des consommateurs ;
  - des fournisseurs alimentaires (*commissions*) ;
- Gérer le suivi de commande et de livraison.

Non-attendu du système :

- Gérer les paiements, ce qui est attendu du fournisseur alimentaire.
- Gérer les préparations de commandes, ce qui est attendu du fournisseur alimentaire.
- Gérer les livraisons, ce qui est attendu du fournisseur alimentaire, et ce qu'il le fasse lui-même ou qu'il, selon les cas, passe par un transporteur tiers.

7.1.2. Utilisateurs et rôles

Les utilisateurs s'organisent au travers des rôles comme suit :

- Utilisateur interne (*collaborateur*) :
  - Relation clientèle,
  - Facturation & demandes d'avoir,
  - Validateur (*données partenaires*),
  - Commercial,
  - Administrateur.
- Utilisateur externe (*client*) :
  - Fournisseur alimentaire (*partenaire*),
  - Consommateur alimentaire.

Ci-dessous le détail de chacun des rôles susmentionnés précisant les fonctions desdits rôles :

RÔLE	CATÉGORIE	FONCTION
Relation clientèle	INTERNE	Dans le cadre du suivi client, l'objectif est de prendre en charge la relation clientèle, aussi bien avec les fournisseurs que les consommateurs.
Facturation	INTERNE	En charge de la facturation des clients, et de la comission du site auprès des fournisseurs.
Commercial	INTERNE	Ce rôle défini les politiques commerciales, leur suivi et l'analyse de leur performance.
Validateur	INTERNE	En charge de valider les données de chaque fournisseur à leur entrée dans l'annuaire fournisseur de Foosus.
Administrateur	INTERNE	Référent en charge de l'administration de la plateforme e-commerce de Foosus.
Fournisseur	EXTERNE	Le fournisseur alimentaire, qui peut être également le producteur alimentaire, et qui va proposer ses produits dans le catalogue et répondre aux commandes des clients de la plateforme.
Consommateur	EXTERNE	Le consommateur alimentaire, qui va sur la plateforme e-commerce : s'enregistrer, rechercher des produits locaux dans le catalogue et commander des produits alimentaires.

7.1.3. Processus métier

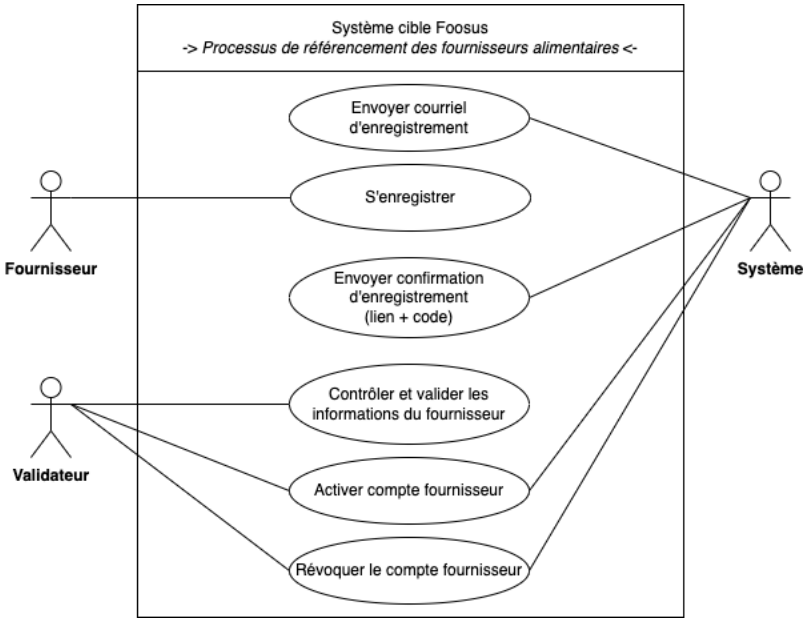
En complément des utilisateurs et rôles, et afin d'être complet, cette section précise **les processus métiers** inhérents aux fonctionnalités attendues du système, id. les aspects non-fonctionnels de celui-ci, permettant d'implémenter les règles de gestion métier. Autrement dit, les processus métiers indiquent au système ce qu'il doit faire et comment pour répondre aux fonctionnalités attendues par le métier.

Aussi pour ce faire, cette section se base notamment sur les document « **Flux de valeurs organisationnelle** » et « **Legacy C4 FR** », cités plus haut à la section « **2.7. Documents Fournis** », décrivant respectivement les différents flux et leur organisation, et l'architecture de référence. Aussi ce document a permis de déterminer les processus suivants et le rôle joué par les différents acteurs, id. comment ces derniers s'intègrent dans les processus :

- Processus de référencement des fournisseurs alimentaires.
- Processus de référencement des produits alimentaires.
- Processus de commande et de facturation.
- Processus de suivi commercial.

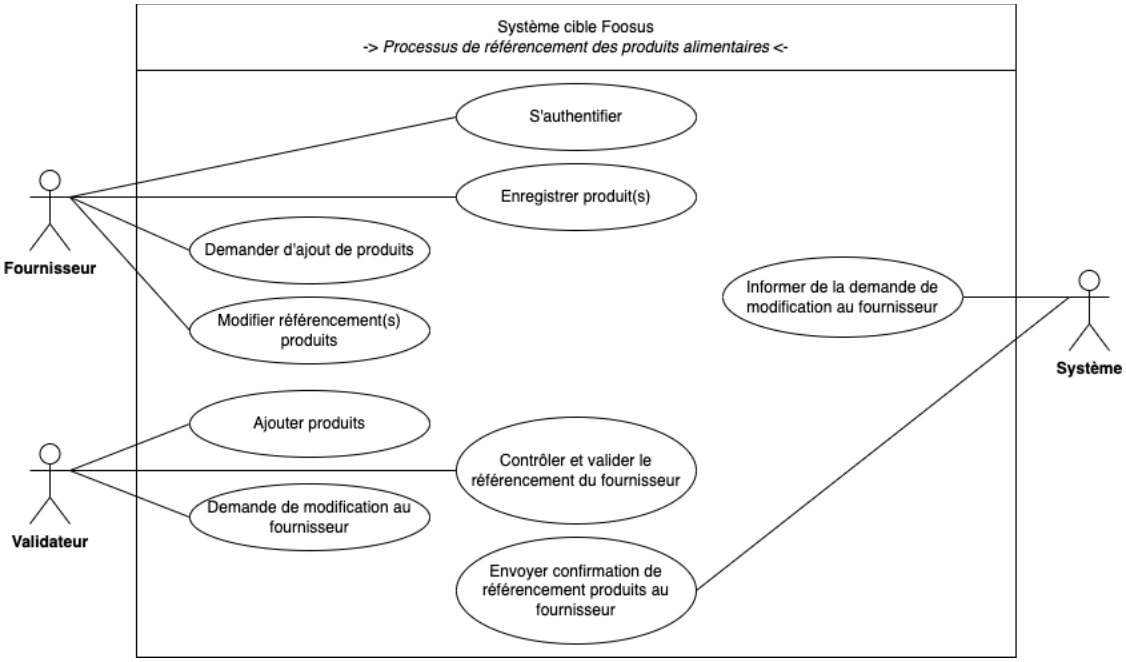
7.1.4. Processus de référencement des fournisseurs alimentaires

Ci-dessous le diagramme UML des cas d'utilisation déterminant les fonctionnalités attendues du système pour le processus de référencement des fournisseurs alimentaires :



7.1.5. Processus de référencement des produits alimentaires

Ci-dessous le diagramme UML des cas d'utilisation déterminant les fonctionnalités attendues du système pour le processus de référencement des produits alimentaires :

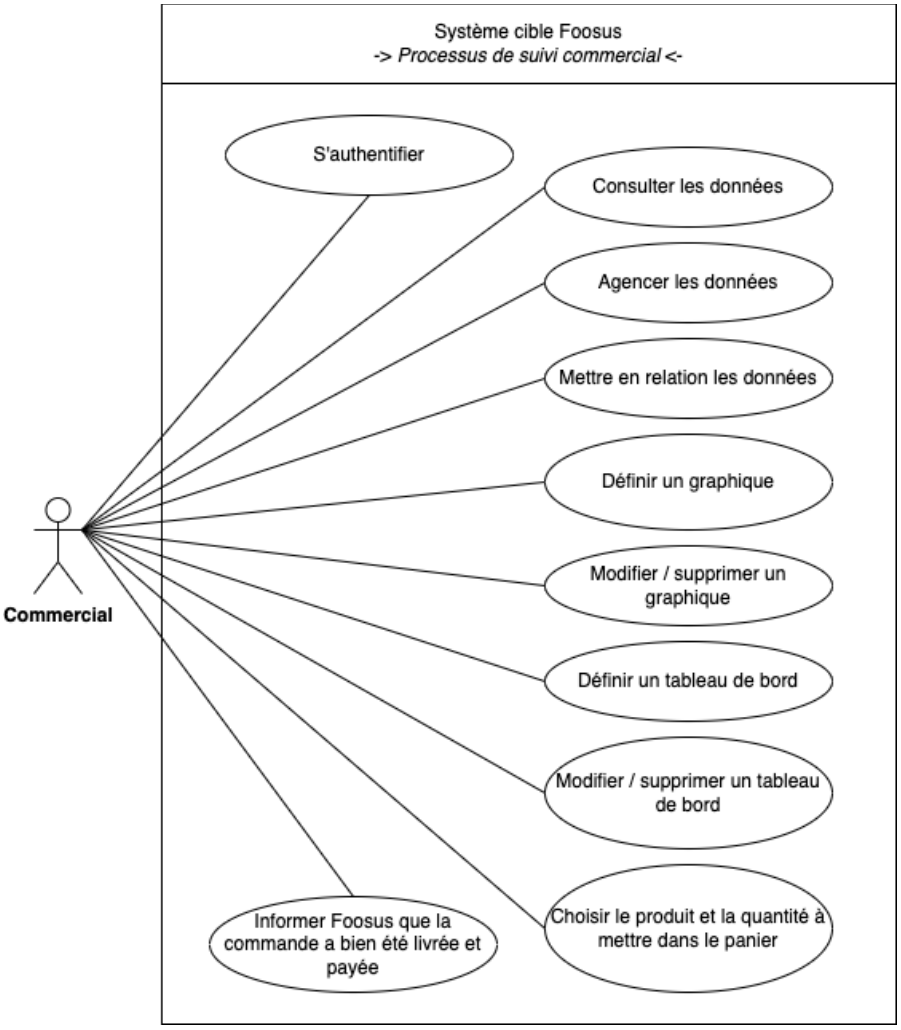


7.1.6. Processus de commande et de facturation

Ci-dessous le diagramme UML des cas d'utilisation déterminant les fonctionnalités attendues du système pour le processus de commande et de facturation :

7.1.7. Processus de suivi commercial

Ci-dessous le diagramme UML des cas d'utilisation déterminant les fonctionnalités attendues du système pour le processus de suivi commercial :



7.2. Couche Applicative

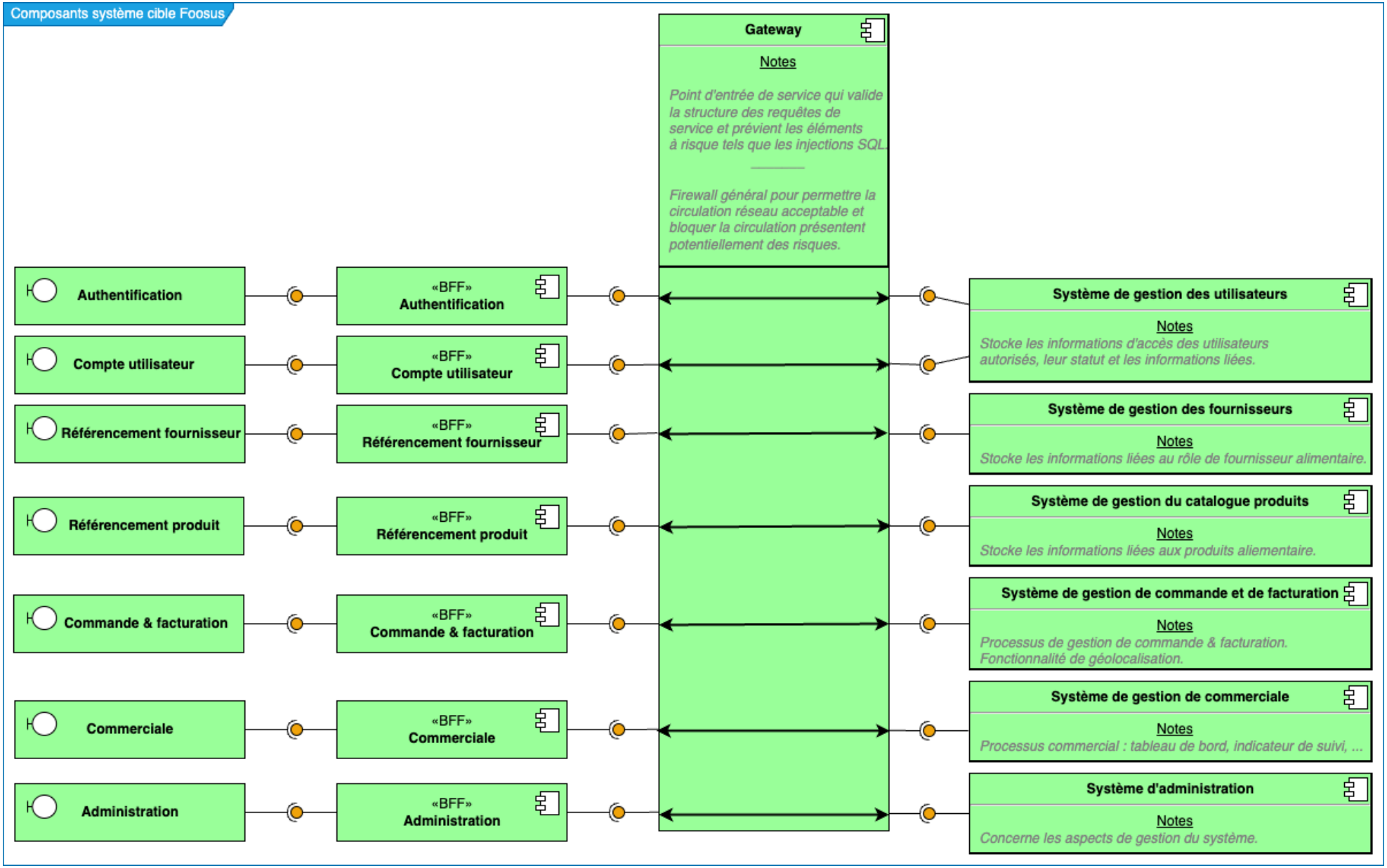
La couche Applicative, nommée également Architecture SI dans TOGAF et qui comprend aussi bien les fonctionnalités que les données informatiques, définit la réponse aux besoins à satisfaire de la couche métier vue plus haut.
Aussi afin de répondre aux besoins, et notamment d'extensibilité et de mise à l'échelle architecturale, de disponibilité et de faciliter à la fois la maintenance et les livraisons, le choix de l'architecture s'est porté sur une architecture Microservices.
Une architecture Microservices est une variante d'une architecture orientée services ou SOA pour Service-Oriented Architecture, qui pour rappel centralise des services Web aussi bien internes au site (*Web, Extranet et Intranet*), qu'externes, (*transporteurs pour la livraison des colis, les banques pour la gestion des paiements Web, etc.*), dans le but de mettre lesdits services Web à disposition de clients en toute transparence, ce qui par là-même facilite leur expérience. Aussi se différentie-t-elle de l'architecture SOA par la particularité d'avoir :

- des composants indépendants possédant leur propre base de données ;
- une granularité plus fine par la mise en service de plus petits composants.

Aussi ces particularités rendent de facto les composants autonomes, faiblement couplés et beaucoup plus appréhendables, dû à l'encapsulation de chaque composant qui est responsable du rendu de service qui lui a été assigné.
Au final, l'architecture Microservices rend la couche applicative plus :

- appréhendable,
- maintenable,
- adaptable,
- extensible,
- robuste / résiliente,
- opérationnellement efficiente.

De plus, il est à noter qu'une architecture Microservices ne s'arrête pas à la couche applicative mais se propage jusqu'aux couches infrastructure et opérationnelle.
Aussi le recours à une telle architecture permet de répondre parfaitement à la fois à la nécessité de service représentant le cœur de métier de Foosus, mais aussi aux besoins fonctionnels, techniques et opérationnels définis plus haut.
En conséquence de quoi, elle a permis de définir la couche applicative, représentant la vision de la nouvelle architecture de la plateforme e-commerce de Foosus, dans le diagramme UML de composant à la page suivante dudit système cible, en cartographiant les processus métiers définis plus haut dans la couche métier, sous la forme d'un bus de service pour chacun d'eux.



Rajouter une note sur la fonction de géolocalisation dans le composant « Système de gestion de commande et de facturation ».

7.3. Couche Infrastructure

La couche infrastructure traite de l'ensemble des mécanismes sous-jacents à la couche Applicative en vue de la supporter, en y apportant les ressources informatiques nécessaires et tout en garantissant :

- la performance ;
- la disponibilité ;
- la protection des données.

Aussi au regard de la couche Applicative définie plus haut et du choix d'architecture, la couche infrastructure va supporter celle-ci en la découpant selon deux axes :

- **vertical**, avec un découpage en trois tiers :
  - **Frontend**, représentant le point d'entrée pour chaque processus / fonctionnalité ;
  - **Gateway** ;
  - **Backend**.
- **horizontal**, afin de respecter le découpage en bus de la couche Applicative.

Pour ce faire, ce découpage va s'appuyer sur l'utilisation de machines virtuelles ou VM (*Virtual Machine*) permettant d'encapsuler le bus de service et les composants inhérents, et de les séparer les uns des autres. Le but étant la robustesse dans le rendu du service aux utilisateurs en leur évitant, autant que faire se peut, de subir des interruptions dudit service.

De plus, le découpage via des machines virtuelles va permettre de faciliter la maintenance et la livraison de nouvelles fonctionnalités / services.

L'ensemble sera implémenté dans un système Cloud en PaaS pour Platform as a Service (AWS[3], OVHcloud[4], …). Quant aux machines virtuelles, elles seront containerisées (Docker[5], Podman[6]) et orchestrées (Docker Swarm[7], Kubernetes[8], OpenShift[9], AWS ECS / EKS[10], …) ; les choix d'implémentation seront à parfaire au besoin et à définir par la suite.

La finalité de ce choix d'implémentation vise à rejoindre, renforcer et compléter celle de la couche Applicative et celle sur l'utilisation de machines virtuelles pour avoir une infrastructure :

- appréhendable,
- maintenable,
- adaptable,
- extensible,
- robuste / résiliente,
- opérationnellement efficiente.

[3] AWS pour Amazon Web Services, est une division du groupe Amazon spécialisée dans le cloud computing à la demande.
[4] OVHcloud, anciennement OVH, société spécialisée entre autres dans le cloud computing, et concurrent d'AWS.
[5] Docker est une solution logicielle de conteneurisation d'applications ayant pour objectif de faciliter et fiabiliser leur déploiement.
[6] Podman, à l'instar de Docker dont il est à la fois le concurrent et l'alternative, est également une solution de conteneurisation d'applications.
[7] Docker Swarm est une solution logicielle d'orchestration native à Docker dont le but est le déploiement d'applications conteneurisées.
[8] Kubernetes, à l'instar de Docker Swarm dont il est le concurrent, est également une solution logicielle d'orchestration, native à Podman.
[9] OpenShift est une solution complète de conteneurisation de la société Red Hat, basée sur Docker et Kubernetes.
[10] AWS ECS / EB pour Amazon Web Services Elastic Container Service / Elastic Kubernetes Service, sont deux solutions d'orchestration de conteneurs de AWS.


Serveur Frontend
Machine virtuelle Authentification
Authentification
«BFF»
Authentification
Machine virtuelle Compte utilisateur
Compte utilisateur
«BFF»
Compte utilisateur
Machine virtuelle Référencement fournisseur
Référencement fournisseur
«BFF»
Référencement fournisseur
Machine virtuelle Référencement produit
Référencement produit
«BFF»
Référencement produit
Machine virtuelle Commande et facturation
Commande & facturation
«BFF»
Commande & facturation
Machine virtuelle Commerciale
Commerciale
«BFF»
Commerciale
Machine virtuelle Administration
Administration
«BFF»
Administration
Serveur d'entrée et d'équilibrage
Machine virtuelle
Firewall réseau
Machine virtuelle
Firewall réseau
Machine virtuelle
Firewall réseau
Machine virtuelle
Firewall réseau
Machine virtuelle
Firewall réseau
Machine virtuelle
Firewall réseau
Machine virtuelle
Firewall réseau
Serveur Backend
Machine virtuelle
Système de gestion des utilisateurs
Notes
Stocke les informations d'accès des utilisateurs autorisés, leur statut et les informations liées.
Machine virtuelle
Système de gestion des fournisseurs
Notes
Stocke les informations liées au rôle de fournisseur alimentaire.
Machine virtuelle
Système de gestion du catalogue produits
Notes
Stocke les informations liées aux produits aliementaire.
Machine virtuelle
Système de gestion de commande et de facturation
Notes
Processus de gestion de commande & facturation.
Fonctionnalité de géolocalisation.
Machine virtuelle
Système de gestion de commerciale
Notes
Processus commercial : tableau de bord, indicateur de suivi, ...
Machine virtuelle
Système d'administration
Notes
Concerne les aspects de gestion du système.

7.4. Couche Opérationnelle

En ce qui concerne l'aspect opérationnel, il faudra avant la mise en service de la solution :

- un plan de reprise sur panne ;
- un mode d'utilisation dégradée de la plateforme ;
- un processus SI de suivi et de mesure de la performance de la plateforme ;
- un support technique et une supervision dédiée ;
- un processus métier de suivi et de mesure de la performance business ;
- une supervision dédiée à la performance business ;
- un processus qualité pour la livraison logicielle de la plateforme ;
- une documentation spécifique de référence à l'architecture, au regard des bonnes pratiques, et permettant de capitaliser au fur-et-à-mesure sur l'expérience acquise ;
- une mise à jour de la documentation IT ;
- un plan de formation rendant les équipes pluridisciplinaires, afin d'éviter la perte de savoir-faire et optimiser les interventions techniques.

8. Bibliographie

- Documentation TOGAF : https://pubs.opengroup.org/architecture/togaf91-doc/arch/
- TOGAF, Archimate, UML et BPMN - 3e éd. (Management des systèmes d'information) de Philippe DESFRAY et Gilbert RAYMOND aux éditions Dunod
- Le Standard TOGAF, version 9.2 - Guide de poche de Andrew Josey et al. aux éditions Van Haren Publishing