

Département Mathématiques et Informatiques

Rapport de Stage de Fin d'Etudes

Pour l'obtention du titre
Master de l'ENSET de Mohammedia

Filière :
**Ingénierie Informatique Big Data et Cloud Computing
IIBDCC**

Développement d'une solution SaaS dédié au commerce

Lieu du stage: **WE ARE beebay**



Réalisé par :
+ **Zakaria El moumnaoui**

Encadré par :

Encadrants Entreprise :
+ **Mustapha Jabrane**
+ **Said EL Haddati**

Encadrant Académique:
+ **Pr. Mohammed Youssfi**

Année
Universitaire :
2023/2024
Table des matières

Introduction générale.....	1
I. Chapitre 1 : Contexte générale.....	3
1. Introduction	4
2. Présentation de l'organisme d'accueil	4
a) Présentation Générale	4
b) Secteur d'activité	4
c) Palette technologique.....	6
d) Domaines d'expertises.....	6
e) Client de l'entreprise	7
3. Contexte du projet.....	8
a) Vue d'ensemble de ERP SaaS.....	8
b) Contexte générale	8
c) Stack technique.....	8
d) Problématique.....	10
e) Objectif	11
4. Gestion du projet	11
5. Conclusion.....	14
II. Chapitre 2 : Analyse initiale et exigence.....	15
1. Introduction	16
2. Le BackLog produit	16
3. Objectif du projet	20
4. Tendance du marché.....	20
5. Identification des acteurs.....	21
6. Spécification des besoins.....	21
7. Développement Cross plateforme	23
a) Adoption de la solution Cloud SaaS.....	23
b) Adoption Flutter et Firebase dans le développement SaaS	24
8. Conclusion.....	26
III. Chapitre 3 : Analyse technique et conception	27
1. Introduction	28
2. Modélisation des besoins fonctionnels.....	28
a) Présentation UML	28
b) Diagramme de classe	28
c) Diagramme cas d'utilisation.....	31
d) Diagramme de séquence.....	36

3.	Structure de l'application	42
a)	Application modulaire	42
4.	Structure du Projet Flutter à développer.....	50
5.	Outils et technologies	52
6.	Conclusion.....	55
IV.	Chapitre 4 : Mise en œuvre.....	56
1.	Introduction	57
2.	Sprint 1 : Développement module authentification et internalisation	57
α)	Planification.....	57
b)	Conception.....	57
c)	Réalisation	59
3.	Sprint 2 : Développement module profile utilisateur et Dashboard admin	64
a)	Planification.....	64
b)	Conception.....	64
c)	Réalisation	64
4.	Sprint 3 : Développement module store des produits	64
a)	Planification.....	64
b)	Conception.....	64
c)	Réalisation	65
5.	Sprint 4 : Développement catalogue produit et module panier	66
a)	Planification.....	66
b)	Conception.....	67
c)	Réalisation	68
6.	Conclusion	72
	Conclusion générale et perspective	73
	Webographie :	75
	Bibliographie.....	76

Introduction générale

Dans le domaine de la technologie de l'information, les solutions ERP SaaS (Enterprise Resource Planning Software as a Service) occupent une place de plus en plus centrale dans la gestion des entreprises modernes. Ces solutions, hébergées dans le cloud, offrent une approche innovante pour intégrer et gérer efficacement les ressources d'une organisation, couvrant un large éventail de fonctions allant de la gestion financière à la logistique en passant par les ressources humaines.

L'utilité de cette dernière réside dans la capacité à centraliser et automatiser les processus métier, offrant ainsi une vue d'ensemble complète des opérations de l'entreprise. En simplifiant ainsi la gestion des ressources et en améliorant l'efficacité opérationnelle, ces solutions permettent aux entreprises de rester compétitives dans un environnement commercial en constante évolution.

Notamment, l'intégration de technologies Flutter et Firebase dans le développement d'applications ERP SaaS offre un avantage significatif. Flutter, avec son framework de développement d'interfaces utilisateur réactives et dynamiques, permet de créer des expériences utilisateur fluides et homogènes sur toutes les plateformes, qu'il s'agisse de mobiles, de desktop ou du web. D'autre part, Firebase fournit une infrastructure cloud robuste et évolutive, offrant une gamme de services tels que l'authentification des utilisateurs, la base de données en temps réel et le stockage cloud, qui sont essentiels pour le fonctionnement efficace des applications ERP SaaS. En combinant la puissance de Flutter pour la création d'interfaces utilisateur modernes et réactives avec les fonctionnalités riches de Firebase pour la gestion des données et des services cloud, les développeurs peuvent créer des applications ERP SaaS hautement performantes et évolutives.

Cette approche permet non seulement d'améliorer l'expérience utilisateur, mais également de garantir la fiabilité, la sécurité et l'évolutivité des applications, répondant ainsi aux besoins croissants des entreprises en matière de gestion des ressources et d'optimisation des processus métier.

Le rapport se compose de quatre chapitres :

- Le premier chapitre aborde le cadre global du projet commençant par une introduction de l'entité hôte, une explication détaillée et la méthodologie de la gestion de projet.
- Le deuxième chapitre concerne l'étude fonctionnelle, non fonctionnelle et l'exigence du projet, il traite également l'aspect SaaS dans le développement des logiciels, exhibant Flutter et Firebase comme des solutions utilisées pour développer les modules en utilisant ce dernier.
- Le troisième chapitre explore la conception et la modélisation de l'application ainsi développée, en expliquant comment aboutir ces deux points dans un contexte SaaS, incluant l'architecture du système et les différentes technologies qui sont adoptés.
- Le quatrième chapitre couvre la partie de la réalisation du projet en mentionnant les différents sprints de la mise en œuvre du système.

I. Chapitre 1 : Contexte générale

1. Introduction

Ce chapitre élabore une mise dans le contexte du projet, il met le point en premier lieu sur l'entreprise d'accueil, ensuite il expose la conduite du projet incluant la méthodologie de travail et la planification du stage.

2. Présentation de l'organisme d'accueil

a) Présentation Générale

We Are BEEBAY est une société spécialisée dans les technologies de l'information, se concentrant sur le conseil et la mise en œuvre des plateformes d'intégration, avec une expertise particulière dans les Hybrid Integration Platforms (HIP) avec des clients principalement basés en Europe et au Moyen Orient depuis plus de 10 ans d'après leurs bureaux à Paris et Casablanca.



Figure 1 : Region de travail de We Are beebay

b) Secteur d'activité



Figure 2 : Sécteur d'activité de We Are Beebay

L'activité de We Are Beebay se base sur 2 axes principaux :

- Conseil en architecture des systèmes distribués et technologies Cloud :

We Are Beebay offre des services de consultation spécialisés dans la conception et l'optimisation des architectures de systèmes distribués, en mettant l'accent sur les technologies cloud. Cela comprend l'évaluation des besoins de l'entreprise, la conception d'architectures adaptées, la sélection des technologies cloud appropriées et la mise en place de stratégies d'intégration efficaces. Leur expertise permet aux clients de bénéficier d'infrastructures robustes et évolutives, alignées sur leurs objectifs métier et optimisées pour une utilisation efficace des ressources cloud.

- Conception et implémentation de plateformes d'intégration et plateformes d'intégration hybrides :

We Are BEEBAY excelle dans la conception et l'implémentation de plateformes d'intégration, y compris les plateformes d'intégration hybrides. Cela comprend la mise en place de solutions telles que IPaaS (Integration Platform as a Service), la gestion d'API, les middlewares orientés messages, etc. Leur approche holistique englobe la compréhension des besoins spécifiques de l'entreprise, la sélection des technologies appropriées, le développement sur mesure et la mise en œuvre, ainsi que l'intégration avec les systèmes existants. En offrant des solutions d'intégration flexibles et robustes, BEEBAY aide les entreprises à optimiser leurs flux de données et à améliorer leur agilité opérationnelle.

Présentation de l'équipe

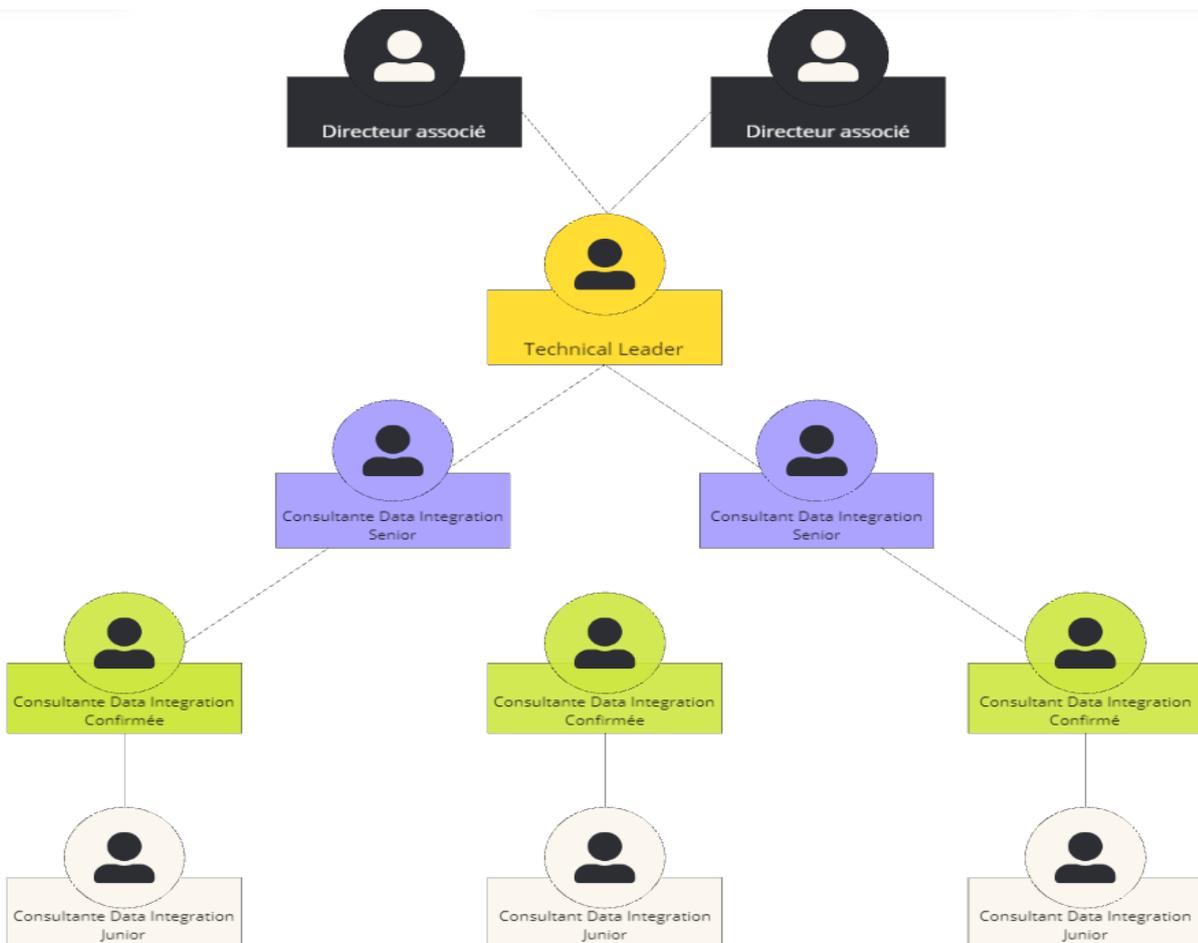


Figure 3 : Présentation de l'équipe

La présentation de l'équipe de We Are Beebay est l'occasion de mettre en lumière les talents qui constituent le cœur de l'entreprise. C'est une équipe diversifiée de professionnels passionnés, chacun apportant son expertise unique pour enrichir la qualité des services offerts. Ensemble, ils incarnent les valeurs de We Are Beebay et travaillent de concert pour délivrer des solutions innovantes en intégration de données, reflétant un engagement envers l'excellence et la satisfaction client.

c) Palette technologique



Figure 4 : Technologies utilisées

We Are BEEBAY excelle dans la maîtrise d'un éventail varié de technologies de pointe, ce qui lui permet de répondre efficacement aux besoins complexes de ses clients en matière de développement et d'architecture informatique ainsi qu'au niveau de l'intégration des données.

Parmi ces technologies figurent des solutions leaders du marché telles que Flutter, Firebase, et FlutterFlow, ainsi que des plateformes Cloud de renommée mondiale comme AWS, Google Cloud Platform et Azure.

De plus, BEEBAY possède une expertise avancée dans des Frameworks spécialisés tels que Spring, offrant ainsi à ses clients une gamme complète de solutions pour leurs défis de développement.

Cette diversité de compétences technologiques permet à We Are BEEBAY de concevoir et de mettre en œuvre des architectures informatiques sur mesure, répondant aux exigences spécifiques de chaque entreprise et favorisant un développement avancé des systèmes et des applications.

d) Domaines d'expertises

Voici les domaines d'expertise :

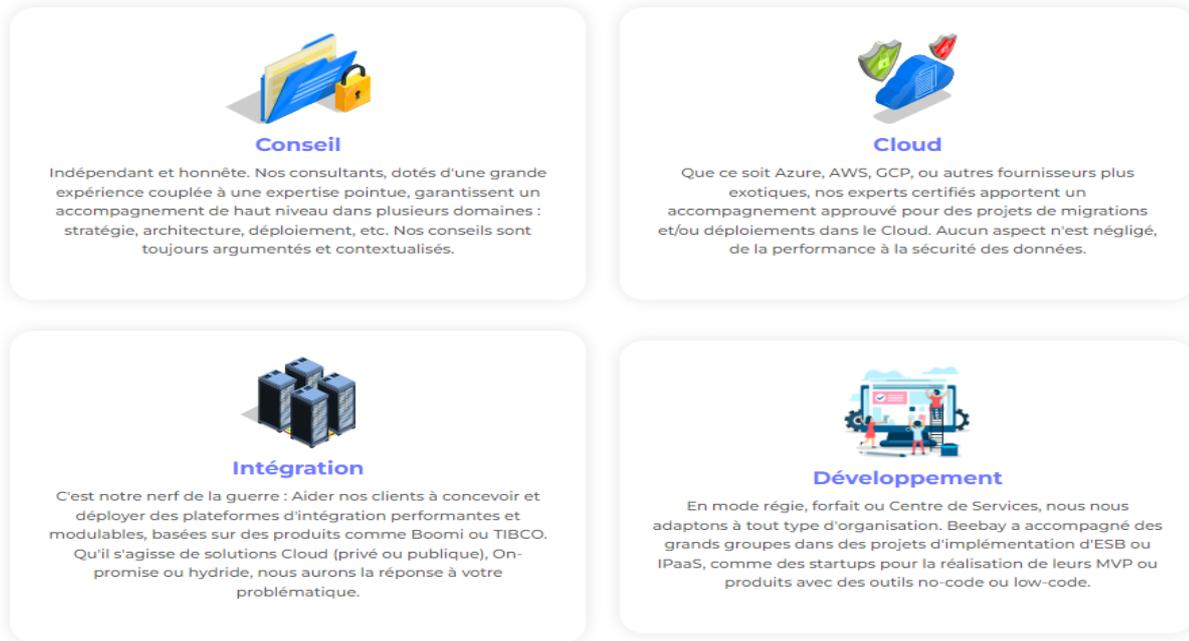


Figure 5 : Domaines d'expertise

We are beebay a accumuler des expériences qui lui permet d'être une référence au niveau de consulting, cloud, intégration et surtout en développement.

e) Client de l'entreprise



Figure 6 : Clients de l'entreprise

We Are Beebay coopère avec ses clients avec une des start-up innovantes aux grandes entreprises internationales, montrant leur capacité à fournir des solutions innovantes pour différents secteurs et besoins.

3. Contexte du projet

a) Vue d'ensemble de ERP SaaS

Dans le paysage dynamique de la gestion d'entreprise moderne, les solutions ERP SaaS (Enterprise Resource Planning Software as a Service) émergent comme des éléments fondamentaux pour les organisations cherchant à optimiser leurs opérations, à accroître leur efficacité et à rester compétitives sur un marché en constante évolution. Ces solutions, hébergées dans le cloud, offrent une approche novatrice pour la gestion intégrée des ressources, couvrant un large éventail de fonctions allant de la comptabilité à la gestion des ressources humaines.

En résumé, l'utilisation d'une solution ERP SaaS offre aux entreprises une approche complète pour la gestion efficace de leurs ressources, en offrant des fonctionnalités avancées pour automatiser les processus opérationnels et améliorer leur agilité.

b) Contexte générale

Les avancées technologiques et les besoins changeants des consommateurs ont conduit les entreprises de commerce en ligne à élargir leurs services et à proposer des solutions plus innovantes. Aujourd'hui, en plus de faciliter les achats en ligne, les plateformes de commerce électronique offrent une multitude de services pratiques et accessibles via Internet. Par exemple, les clients peuvent gérer leurs commandes, exécuter des paiements, suivre leurs livraisons, accéder à des recommandations personnalisées et bénéficier de promotions exclusives directement depuis leur appareil connecté. Cependant, suivant les besoins exponentiel des clients les développements restent toujours à considérer dans ce domaine de commerce électronique.

Dans le cas du projet ZITECH, leur solution standard comprend des modules dédiés à la gestion des produits, au panier d'achat et au paiement. Les modules de gestion des produits permettent aux vendeurs de gérer les stocks, de mettre à jour les descriptions de produits de même, pour les clients ils leur permettent de passer leurs commandes aisément.

Quant aux modules de paiement, ils regroupent des fonctionnalités permettant aux clients de payer par carte de crédit ou par virement bancaire. Ces modules facilitent le processus de paiement en offrant une plateforme centralisée où les clients peuvent saisir leurs informations de paiement et effectuer leurs transactions de manière pratique et sécurisée.

c) Stack technique

i. Flutter

Voici une image illustrative de la framework Flutter :



Figure 7 : Framework Flutter

Flutter est un framework open-source et polyvalent pour créer des applications mobiles et web de hautes performances. Développé par Google, le développement d'applications Flutter permet aux développeurs d'utiliser une seule base de code pour créer des applications pour iOS et Android, ainsi que des plateformes web et de bureau. Il utilise un modèle de programmation réactif pour construire des interfaces utilisateur, ce qui permet aux développeurs de créer facilement des interfaces belles et interactives.

Flutter framework se distingue par son SDK qui offre un style réactif, sans compter sur un pont JavaScript. Cette approche permet à Flutter d'atteindre des niveaux de performance comparables à ceux de son concurrent direct, React Native. Flutter s'intègre également de manière transparente à diverses plateformes, notamment les applications Android, iOS, Linux, macOS, Windows et Google Fuchsia.

Il dispose d'une communauté intéressante de développeurs qui contribuent à son développement et créent des paquets qui peuvent être facilement intégrés dans une application Flutter. Il s'agit d'un excellent choix pour les développeurs qui veulent créer des applications multi-plateformes visuellement superbes avec un processus de développement rapide.

ii. Firebase

Voici l'architecture Firebase:

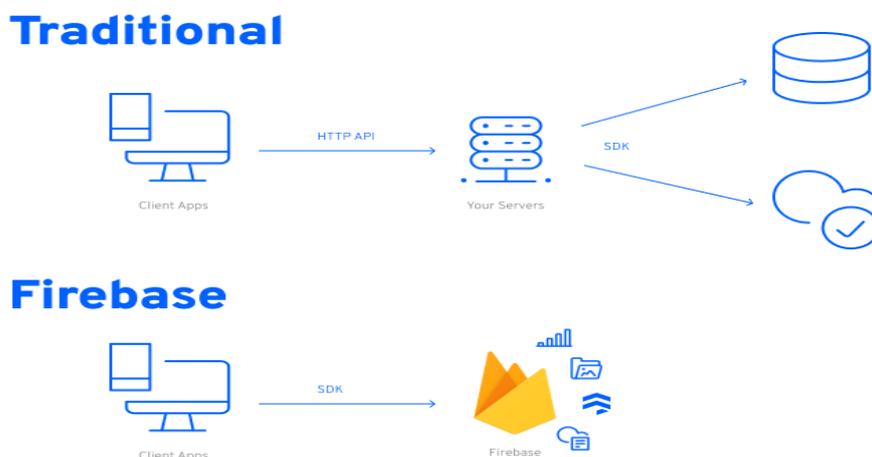


Figure 8 : Firebase backend

Firestore est une suite intégrée d'outils de développement qui permettent aux utilisateurs de créer de puissantes applications web et mobiles. Maintenu par Google et exécuté sur l'infrastructure cloud de Google, Firestore interagit directement avec les services backend, contournant le besoin d'un middleware entre l'application et le service.

En tant que sous-ensemble organisé de la plateforme Google Cloud (GCP), Firestore rend plusieurs services Google plus accessibles pour les développeurs d'applications. La console Firestore est beaucoup plus concentrée et intuitive que celle de GCP. Et le kit de développement logiciel (SDK) bien documenté, constitué d'un ensemble d'outils, de bibliothèques, de documentation, d'exemples de code et de guides que les développeurs téléchargent directement et suivent qui permet de connecter leurs applications à ces services avec un effort de développement minimal.

De plus, l'ensemble d'outils intègre une grande partie des services que les développeurs doivent normalement configurer manuellement, comme l'authentification, la configuration, le stockage de fichiers et les analyses. S'appuyer sur l'infrastructure Google permet aux utilisateurs (et à nos clients) de savoir avec certitude que les applications créées avec Firestore sont bien conçues, fiables et évolutives.

d) Problématique

À l'ère numérique, le développement de plateformes web et d'applications mobiles est devenu un élément clé de la stratégie commerciale de nombreuses entreprises. Ces plateformes doivent être à la fois performantes, sécurisées et évolutives pour répondre aux besoins changeants des utilisateurs et aux défis technologiques émergents.

Dans ce contexte dynamique, le besoin de rapidité de développement est crucial pour rester compétitif sur le marché. C'est là que des outils et des technologies modernes comme Flutter et Firestore entrent en jeu. Flutter, un Framework de développement d'interfaces utilisateur multiplateforme, offre une approche rapide et efficace pour la création d'applications web et mobiles. Grâce à son architecture basée sur des widgets réactifs, Flutter permet aux développeurs de concevoir des interfaces utilisateur fluides et attrayantes tout en réduisant le temps de développement.

D'autre part, Firestore, une plateforme de développement d'applications mobiles et web de Google, fournit une suite complète de services cloud, y compris l'authentification des utilisateurs, la base de données en temps réel, le stockage de fichiers, et bien plus encore. L'intégration de Firestore dans le processus de développement permet aux développeurs de gagner du temps en évitant la mise en place d'une infrastructure backend complexe et en se concentrant plutôt sur la création de fonctionnalités clés pour l'application.

Cependant, malgré les avantages de rapidité offerts par Flutter et Firestore, les développeurs sont toujours confrontés à des défis complexes liés à la conception et à l'implémentation d'une architecture logicielle robuste et flexible. Ils doivent trouver des stratégies efficaces pour intégrer harmonieusement ces technologies tout en garantissant la réutilisabilité du code et la maintenabilité du système. De plus, assurer des performances optimales et la sécurité des applications reste une priorité majeure pour garantir une expérience utilisateur fluide et sécurisée.

Dans ce contexte, les questions clés qui se posent sont les suivantes :

- ❖ Comment concevoir une architecture logicielle flexible et évolutive pour une application web, en tirant parti des avantages de Flutter et Firebase tout en respectant les principes de conception logicielle et les meilleures pratiques de développement ?
- ❖ Quelles sont les stratégies efficaces pour intégrer de manière modulaire Flutter et Firebase dans une application web, en favorisant la factorisation du code et la maintenabilité du système ?
- ❖ Comment assurer des performances pour l'application, en mettant en œuvre des techniques d'optimisation du code, de gestion des ressources et de mise en cache des données ?
- ❖ Quels sont les enjeux majeurs liés à la sécurité des logiciels et à la protection des données, et quelles approches peuvent être adoptées pour identifier les risques de sécurité potentiels, tout en garantissant la conformité aux normes de sécurité et de confidentialité ?

e) Objectif

Ce projet permet de répondre aux objectifs ci-dessous liés aux problématiques mentionnées :

1. Mettre en place une architecture logicielle flexible et intégrée pour les applications web et mobiles, en adaptant les technologies Flutter et Firebase en fonction des besoins spécifiques de chaque plateforme.
2. Utiliser des méthodes de développement efficaces qui utilisent les capacités de Flutter et Firebase, tout en garantissant la qualité du code, la continuité du système et la réutilisabilité des composants.
3. Améliorer les performances de l'application en utilisant des techniques avancées d'optimisation, de gestion des ressources et de mise en cache des données, pour garantir une expérience utilisateur fluide et réactive.
4. Implémenter des mesures de sécurité robustes pour préserver la confidentialité des données utilisateurs, en respectant les normes et réglementations en vigueur, et en anticipant les risques de piratage ou de violation de la vie privée.

4. Gestion du projet

La sélection des méthodes de travail joue un rôle déterminant dans la réussite de votre projet de développement informatique. Chez We Are Beebay, les méthodes agiles sont souvent privilégiées. En effet, ces méthodes impliquent activement les clients tout au long du processus de développement, permettant une meilleure compréhension du projet et des risques potentiels. Cette implication favorise la transparence, la collaboration et une meilleure adaptabilité aux besoins changeants. En conséquence, les méthodes agiles contribuent à la création d'un produit final qui répond mieux aux attentes des clients et augmente leur satisfaction.

a) Ressources Humaines

Dans l'ensemble, la puissance de l'équipe de projet réside dans l'alliance stratégique entre les encadrants internes de l'ENSET Mohammédia et les encadrants externes de We Are Beebay, comme en témoigne le tableau de référence. Cette collaboration dynamique et polyvalente est essentielle pour mener à bien et réussir le projet.

Pr. Youssfi Mohamed	Prof a ENSET Mohammédia	Encadrant interne
Mr. El haddati Said	Directeur Technique	Encadrant externe
Mr. Jabrane Mustapha	Technical Leader	Encadrant externe

Tableau 1 : Equipe du projet

b) Méthode Scrum

La méthode **Scrum** est une approche **agile** de gestion de projet, particulièrement utilisée dans le développement logiciel. Elle se caractérise par des itérations courtes et régulières appelées "sprints", généralement d'une durée de deux à quatre semaines. Chaque sprint commence par une planification détaillée où l'équipe décide des tâches à accomplir. Cette planification inclut la conception des fonctionnalités à développer, leur réalisation et leur intégration.

Scrum repose sur trois rôles clés : le **Product Owner**, le **Scrum Master** et l'**équipe de développement**. Le Product Owner est responsable de la vision du produit et de la gestion du backlog, en priorisant les fonctionnalités en fonction de la valeur pour l'utilisateur. Le Scrum Master facilite le processus Scrum, élimine les obstacles et aide l'équipe à s'améliorer continuellement. L'équipe de développement est auto-organisé et interdisciplinaire, capable de transformer les éléments du backlog en fonctionnalités utilisables.

Les cérémonies Scrum incluent les réunions quotidiennes ("daily stand-ups"), où chaque membre de l'équipe partage ses progrès et ses obstacles, ainsi que les planifications de sprint, les revues de sprint et les rétrospectives. Ces événements favorisent la transparence, l'inspection et l'adaptation, permettant à l'équipe de réagir rapidement aux changements et de livrer un produit de haute qualité de manière itérative et incrémentale.

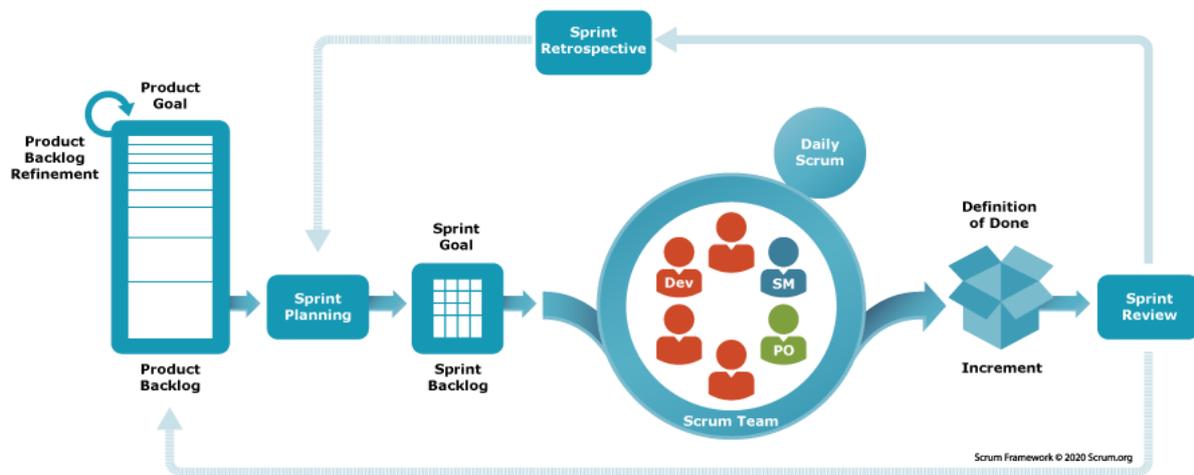
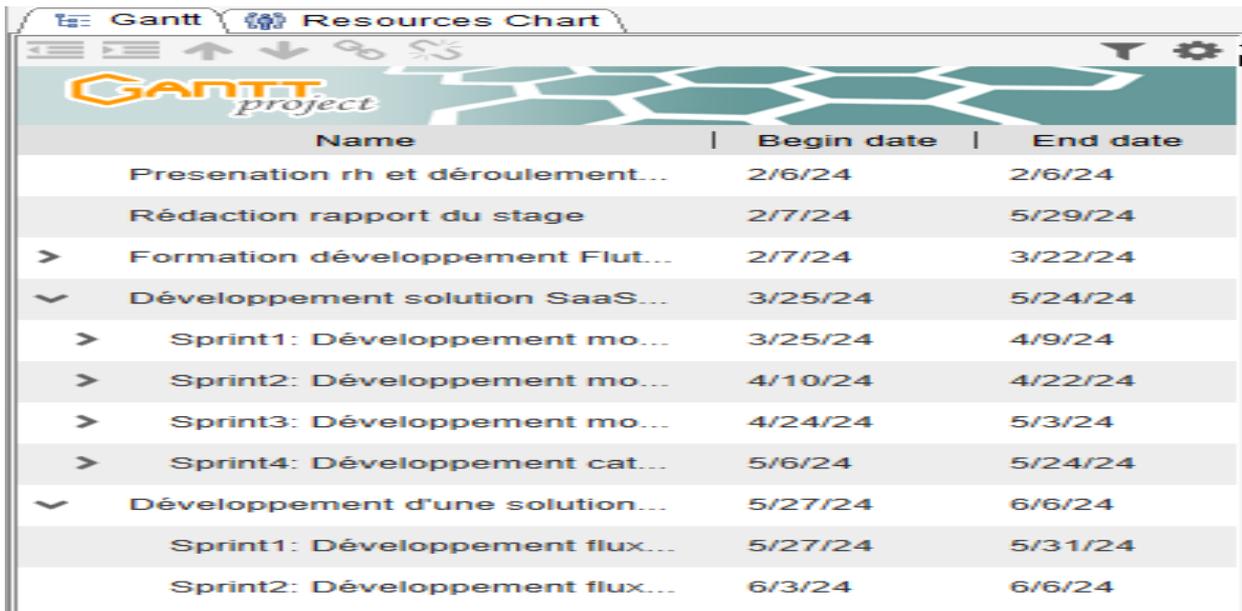


Figure 9 : Methode scrum

c) Diagramme de Gantt

Le diagramme est un outil essentiel en gestion de projet, offrant une vue d'ensemble des diverses tâches, de leur ordre d'exécution et de leur période d'accomplissement. Cet outil se présente sous la forme d'une chronologie horizontale où les tâches sont illustrées par des segments. Chaque segment sur le diagramme de Gantt symbolise une tâche distincte et sa longueur reflète le temps estimé pour son achèvement.

Le tableau ci-dessous représente les tâches et leurs dates de nos projets :



Name	Begin date	End date
Presenation rh et déroulement...	2/6/24	2/6/24
Rédaction rapport du stage	2/7/24	5/29/24
> Formation développement Flut...	2/7/24	3/22/24
✓ Développement solution SaaS...	3/25/24	5/24/24
> Sprint1: Développement mo...	3/25/24	4/9/24
> Sprint2: Développement mo...	4/10/24	4/22/24
> Sprint3: Développement mo...	4/24/24	5/3/24
> Sprint4: Développement cat...	5/6/24	5/24/24
✓ Développement d'une solution...	5/27/24	6/6/24
Sprint1: Développement flux...	5/27/24	5/31/24
Sprint2: Développement flux...	6/3/24	6/6/24

Figure 10 : Partition du projet

Le diagramme ci-dessous représente la planification suivant l'approche de Gantt:

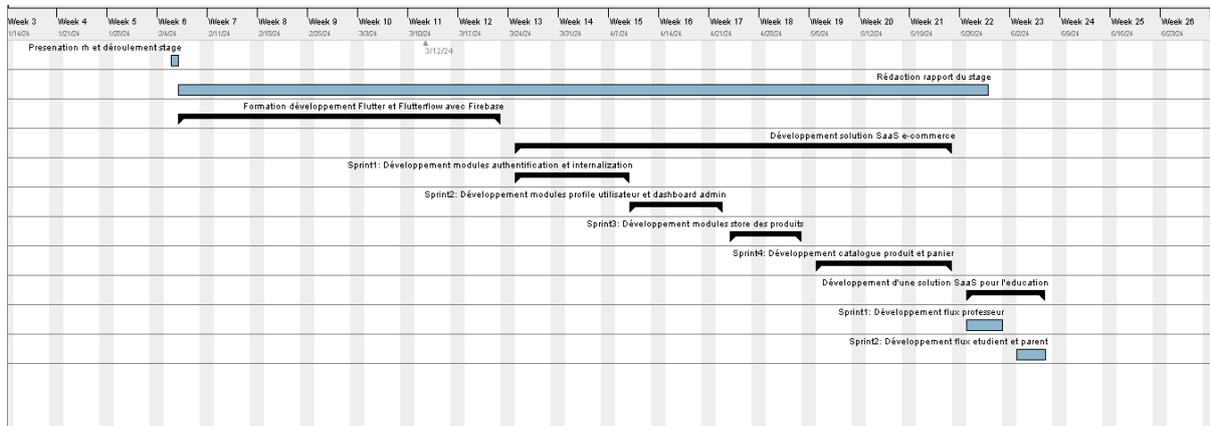


Figure 11 : Planification du projet

5. Conclusion

En conclusion de ce chapitre, une vue d'ensemble de l'entreprise a été présentée, incluant une fiche technique de son expertise, les technologies adoptées, une description de la clientèle et le contexte du stage. Le chapitre a également introduit le projet de stage, la problématique à résoudre, le stack technique utilisé, ainsi que la méthodologie et la planification suivies ont également été introduites. Le prochain chapitre se concentrera sur une description approfondie du projet de stage afin de clarifier le problème et la solution avant sa mise en œuvre.

II. Chapitre 2 : Analyse initiale et exigence

1. Introduction

Afin d'atteindre les objectifs de notre projet, l'étude fonctionnelle est une étape primordiale. Dans ce chapitre, nous nous intéresserons à l'étude et la spécification des besoins. Il a pour principal objectif de présenter un repérage initial des besoins et exigences fonctionnelles que doit satisfaire la solution.

2. Le BackLog produit

Le besoin du produit (ou product Backlog) est une liste qui contient toutes les fonctionnalités et les tâches à réaliser dans un projet. Il est créé par l'équipe de développement en collaboration avec les parties prenantes du projet, comme les clients ou les utilisateurs. Il est constamment mis à jour et évolue tout au long du projet. Les éléments peuvent être ajoutés, modifiés ou supprimés en fonction des besoins et des retours des utilisateurs.

Les sections suivantes présentent le BackLog produit du projet à réaliser :

Voici user story pour vendeurs :

Voici User Story vendeur :

User story	Taches
Enregistrement du Vendeur <ul style="list-style-type: none"> En tant que vendeur, je veux m'enregistrer sur la plateforme afin de pouvoir vendre mes produits. 	<ul style="list-style-type: none"> Concevoir le formulaire d'inscription. Mettre en œuvre l'authentification des utilisateurs (vérification par e-mail, cryptage des mots de passe). Développer le schéma de base de données pour stocker les informations des vendeurs.
Listage des Produits <ul style="list-style-type: none"> En tant que vendeur, je veux lister mes produits afin que les clients puissent les voir et les acheter. 	<ul style="list-style-type: none"> Créer la page de listage des produits. Mettre en œuvre les catégories et les tags de produits. Développer la fonctionnalité de téléchargement d'images. Implémenter les champs de prix, promotions, gestion des stocks et description des produits.
Gestion des Commandes <ul style="list-style-type: none"> En tant que vendeur, je veux gérer mes commandes afin de pouvoir les traiter efficacement. 	<ul style="list-style-type: none"> Créer le tableau de bord de gestion des commandes. Développer la fonctionnalité de mise à jour du statut des commandes. Mettre en œuvre l'historique des commandes et les options de filtrage.

<p>Analyse et Rapports</p> <ul style="list-style-type: none"> ▪ En tant que vendeur, je veux voir les analyses de ventes afin de pouvoir suivre mes performances. 	<ul style="list-style-type: none"> ▪ Concevoir le tableau de bord des analyses de ventes. ▪ Mettre en œuvre la visualisation des données de vente (graphiques, tableaux). ▪ Développer la fonctionnalité d'exportation des rapports.
<p>Gestion du Profil du Vendeur</p> <ul style="list-style-type: none"> ▪ En tant que vendeur, je veux gérer mon profil afin que les clients puissent en savoir plus sur ma boutique électronique. 	<ul style="list-style-type: none"> ▪ Créer la page d'édition de profil. ▪ Mettre en œuvre le téléchargement de photo de profil. ▪ Développer les champs de bio et d'informations de contact du vendeur.

Tableau 2 : User story vendeur

Voici user story pour admin

User story	Taches
<p>Tableau de Bord Admin</p> <ul style="list-style-type: none"> ▪ En tant qu'admin, je veux accéder à un tableau de bord afin de pouvoir gérer la plateforme efficacement. 	<ul style="list-style-type: none"> ▪ Concevoir l'interface du tableau de bord admin. ▪ Mettre en œuvre la gestion des utilisateurs (vendeurs, clients). ▪ Développer les notifications et alertes du système.
<p>Approbation des Produits</p> <ul style="list-style-type: none"> ▪ En tant qu'admin, je veux approuver les produits avant qu'ils ne soient en ligne afin de maintenir des standards de qualité. 	<ul style="list-style-type: none"> ▪ Créer le workflow d'approbation des produits. ▪ Mettre en œuvre le système de notification pour les approbations en attente. ▪ Développer la fonctionnalité de retour de rejet.
<p>Gestion des Utilisateurs</p> <ul style="list-style-type: none"> ▪ En tant qu'admin, je veux gérer les comptes utilisateurs afin de pouvoir traiter les inscriptions, suspensions et suppressions. 	<ul style="list-style-type: none"> ▪ Mettre en œuvre la suspension et la réactivation des comptes utilisateurs. ▪ Développer la fonctionnalité de suppression des comptes utilisateurs. ▪ Créer le système de gestion des rôles et des permissions des utilisateurs.
<p>Gestion de Contenu</p> <ul style="list-style-type: none"> ▪ En tant qu'admin, je veux gérer le contenu du site afin de pouvoir mettre à jour les informations facilement. 	<ul style="list-style-type: none"> ▪ Créer l'interface de gestion de contenu. ▪ Mettre en œuvre un éditeur WYSIWYG pour les pages. ▪ Développer la bibliothèque de médias pour les images et les documents.
<p>Rapports et Analyses</p> <ul style="list-style-type: none"> ▪ En tant qu'admin, je veux voir les analyses de la plateforme afin de pouvoir suivre les performances et l'activité des utilisateurs. 	<ul style="list-style-type: none"> ▪ Concevoir le tableau de bord des analyses admin. ▪ Mettre en œuvre le suivi de l'activité des utilisateurs. ▪ Développer les outils de rapport financier.

Tableau 3 : User story admin

Voici user story pour client

User story	Taches
<p>Enregistrement et Connexion du Client</p> <ul style="list-style-type: none"> En tant que client, je veux m'inscrire et me connecter afin de pouvoir effectuer des achats et suivre mes commandes. 	<ul style="list-style-type: none"> Concevoir les formulaires d'inscription et de connexion. Mettre en œuvre l'intégration de connexion via les réseaux sociaux. Développer la fonctionnalité de récupération de mot de passe.
<p>Recherche et Filtrage de Produits</p> <ul style="list-style-type: none"> En tant que client, je veux rechercher et filtrer des produits afin de trouver ce dont j'ai besoin rapidement. 	<ul style="list-style-type: none"> Créer la barre de recherche et les options de filtrage avancées. Mettre en œuvre le tri par prix, popularité et notation. Développer la navigation par catégories et sous-catégories.
<p>Détails et Avis des Produits</p> <ul style="list-style-type: none"> En tant que client, je veux voir les informations détaillées des produits et les avis afin de pouvoir prendre des décisions d'achat éclairées. 	<ul style="list-style-type: none"> Concevoir la mise en page de la page de détails du produit. Mettre en œuvre le système d'avis et de notation des clients. Développer la section des produits associés et des recommandations.
<p>Panier et Paiement</p> <ul style="list-style-type: none"> En tant que client, je veux ajouter des produits à mon panier et passer à la caisse afin de pouvoir les acheter. 	<ul style="list-style-type: none"> Créer la fonctionnalité de panier d'achat. Mettre en œuvre le processus de paiement sécurisé. Développer l'intégration de la passerelle de paiement (carte de crédit, PayPal, etc.).
<p>Suivi des Commandes et Historique</p> <ul style="list-style-type: none"> En tant que client, je veux suivre mes commandes et voir l'historique des commandes afin de connaître le statut de mes achats. 	<ul style="list-style-type: none"> Créer l'interface de suivi des commandes. Mettre en œuvre la page d'historique des commandes. Développer le système de notification pour les mises à jour du statut des commandes.
<p>Support Client</p>	<ul style="list-style-type: none"> Créer le formulaire de contact du support client.

<ul style="list-style-type: none"> ▪ En tant que client, je veux accéder au support client afin de pouvoir résoudre tout problème avec mes commandes. 	<ul style="list-style-type: none"> ▪ Mettre en œuvre la fonctionnalité de chat en direct. ▪ Développer la section FAQ et centre d'aide.
--	---

Tableau 4 : User story client

En abordant ces user stories et tâches, la plateforme e-commerce peut offrir une expérience robuste et conviviale pour les vendeurs, les admins et les clients.

3. Objectif du projet

L'objectif principal d'une application de commerce électronique est de faciliter la vente de produits et services en ligne, offrant accessibilité et commodité aux clients. Elle vise à augmenter les ventes en atteignant un public plus large et en réduisant les coûts de gestion. Grâce à une architecture modulaire, l'application permet une personnalisation flexible de l'expérience d'achat et optimise la gestion des stocks et la logistique. Cette modularité permet d'intégrer facilement diverses options de paiement sécurisées et des services de support client en ligne.

4. Tendances du marché

a) Introduction

Les évolutions technologiques et les attentes croissantes des consommateurs obligent les entreprises de commerce en ligne à innover constamment pour rester compétitive et répondre aux besoins nombreux ainsi générer.

b) Tendances du Marché

Les tendances actuelles dans le domaine du e-commerce révèlent une évolution rapide et significative, notamment avec l'avènement du Mobile Commerce. L'utilisation des smartphones et les appareils ordinateurs pour les achats en ligne connaît une augmentation continue, propulsant ainsi les entreprises à adapter leurs plateformes pour une expérience utilisateur optimisée sur tout appareil connectés possible.

Parallèlement, l'**Intelligence Artificielle (IA)** et le **Machine Learning (ML)** prennent une place de plus en plus importante dans le paysage du e-commerce. Ces technologies sont largement utilisées pour fournir des **recommandations personnalisées** aux clients, améliorer la personnalisation des services et optimiser le support client, renforçant ainsi l'engagement et la fidélité des clients.

Une autre tendance majeure est l'intégration de la **Réalité Augmentée (RA)** dans les plateformes de commerce électronique. La RA permet aux consommateurs de visualiser les produits dans leur environnement réel avant de les acheter, offrant ainsi une expérience d'achat plus immersive et augmentant la confiance dans la qualité des produits.

Par ailleurs, la question de la durabilité et de la responsabilité sociale gagne en importance dans le e-commerce. Les consommateurs sont de plus en plus sensibles aux pratiques commerciales durables et éthiques, ce qui incite les entreprises à adopter des initiatives responsables, telles que la réduction des déchets, l'utilisation de matériaux recyclés et le commerce équitable, afin de répondre aux attentes de leurs clients et de contribuer positivement à la société.

5. Identification des acteurs

Les acteurs clés sont les utilisateurs (clients), les vendeurs, les administrateurs, le système de paiement, le service de notification, et module IA (recommandation).

Utilisateur (Client)

Les utilisateurs recherchent et consultent des produits, gèrent leur panier, passent des commandes, reçoivent des notifications et obtiennent des recommandations personnalisées.

Vendeur (Seller)

Les vendeurs gèrent leurs produits, traitent les commandes, suivent les ventes, gèrent l'inventaire et communiquent avec les clients.

Administrateur

Les administrateurs gèrent les comptes, supervisent les produits, suivent les commandes, envoient des notifications et génèrent des rapports.

Système de Paiement

Le système de paiement traite les transactions financières, assure la sécurité des paiements et envoie des confirmations.

Service de Notification et messagerie

Le service de notification et messagerie envoie des notifications push sur les statuts des commandes, les promotions, les annonces importantes, des e-mails de confirmation et permet un échange de messages entre les acteurs suivant leurs besoins de communication.

Module IA

Le module IA offre des recommandations personnalisées, répond aux questions fréquentes et améliore l'expérience utilisateur.

En fin, l'application facilite les ventes en ligne, optimise les transactions et communications, et améliore l'expérience utilisateur grâce aux notifications et recommandations personnalisées.

6. Spécification des besoins

Dans le paysage concurrentiel du commerce électronique, fournir une expérience utilisateur exceptionnelle est essentiel pour attirer et fidéliser les clients. Une application e-commerce doit donc intégrer une série de fonctionnalités fondamentales et avancées pour répondre aux besoins variés des utilisateurs. De la création de compte à la réception de notifications post-achat, chaque fonctionnalité joue un rôle crucial pour assurer une expérience fluide et satisfaisante. En outre, l'incorporation de recommandations personnalisées via le module IA peut enrichir l'expérience utilisateur en offrant des suggestions pertinentes en temps réel. Cette partie explore en détail ces fonctionnalités essentielles et sophistiquées, démontrant comment elles peuvent transformer une application e-commerce en une plateforme conviviale et efficace.

a) Spécifications Fonctionnelles

Les spécifications fonctionnelles définissent les capacités essentielles de l'application e-commerce, assurant une expérience utilisateur fluide.

1. Gestion des Produits et de l'Inventaire

Permettre l'ajout, la mise à jour, la suppression des produits, la surveillance des niveaux de stock, et l'affichage des détails des produits.

2. Recherche, Consultation et Recommandations de Produits

Faciliter la recherche de produits et offrir des recommandations personnalisées basées sur les préférences et l'historique d'achat des utilisateurs.

3. Gestion du Panier et Traitement des Commandes

Permettre aux utilisateurs de gérer leur panier, entrer les informations de livraison et paiement, confirmer les commandes, et suivre leur statut.

4. Authentification et Gestion des Comptes

Assurer l'authentification des utilisateurs pour accéder aux fonctionnalités, créer, modifier et supprimer des comptes, et gérer les rôles et autorisations.

5. Paiement et Notifications

Gérer les paiements en ligne en toute sécurité, envoyer des notifications push et e-mails, et faciliter la communication entre les utilisateurs, vendeurs et administrateurs.

Ces besoins fonctionnels assurent une gestion efficace des produits, des commandes, et des comptes, tout en garantissant la sécurité des transactions et des communications. Ils fournissent une expérience utilisateur optimale et sécurisée pour les clients, vendeurs, et administrateurs.

b) Spécifications non Fonctionnelles

Les spécifications non fonctionnelles sont essentielles pour garantir la performance, la sécurité, la scalabilité, la modularité, la fiabilité et l'accessibilité d'une application e-commerce. Ces aspects assurent une expérience utilisateur fluide et sécurisée.

Performance :

L'application doit être réactive avec des temps de chargement minimaux. Pour garantir cette réactivité, les pratiques d'optimisation des performances doivent être suivies, telles que l'utilisation d'images compressées, la mise en cache des données fréquemment consultées, et l'optimisation du code pour une exécution rapide. Des outils comme Firebase Performance Monitoring peuvent être utilisés pour surveiller et améliorer les temps de réponse et l'expérience utilisateur globale.

Sécurité :

Les données des utilisateurs et les transactions doivent être sécurisées avec des protocoles appropriés tels que SSL/TLS pour la transmission sécurisée des données. De plus, les données sensibles doivent être chiffrées, tant au repos qu'en transit, pour protéger contre les accès non autorisés. Des mesures

supplémentaires comme l'authentification à deux facteurs (2FA) peuvent être implémentées pour renforcer la sécurité des comptes utilisateurs.

Scalabilité :

L'application doit pouvoir gérer une augmentation du nombre d'utilisateurs et de transactions sans dégradation des performances. L'utilisation de services cloud tels que Firebase permet une mise à l'échelle automatique des ressources en fonction de la demande. Cette scalabilité garantit que l'application peut continuer à fonctionner de manière fluide même en période de forte activité, comme lors de promotions ou de périodes de soldes.

Fiabilité :

L'application doit fonctionner de manière fiable, sans bugs ni erreurs. Les fonctionnalités doivent être testées et validées afin d'assurer un bon fonctionnement global du système.

Extensibilité :

L'application doit être conçue de manière à faciliter l'ajout de nouvelles fonctionnalités ultérieurement, sans entraîner des coûts de développement excessifs. Elle doit être flexible et adaptable pour répondre aux évolutions des besoins.

Modularité :

L'application doit être développée de manière modulaire, en séparant les différents modules fonctionnels. Cela permet une meilleure gestion du code, une facilité de maintenance et une plus grande flexibilité pour ajouter, modifier ou supprimer des fonctionnalités.

Ergonomie :

Les interfaces de l'application doivent être conviviales et faciles à utiliser. L'objectif est de garantir une expérience utilisateur agréable et intuitive.

Accessibilité :

L'application doit être accessible aux utilisateurs avec des handicaps. Cela inclut la compatibilité avec les lecteurs d'écran, une navigation facile, et des éléments interactifs clairement identifiables. L'interface utilisateur doit respecter les directives d'accessibilité, comme les WCAG (Web Content Accessibility Guidelines), pour s'assurer que tous les utilisateurs, y compris ceux ayant des limitations physiques ou sensorielles, peuvent utiliser l'application efficacement.

En se concentrant sur la performance, la sécurité, la scalabilité, la modularité, la fiabilité et l'accessibilité, l'application e-commerce est bien équipée pour offrir une expérience utilisateur optimale et s'adapter aux besoins futurs.

7. Développement Cross plateforme

a) Adoption de la solution Cloud SaaS

Le logiciel en tant que service (SaaS) est une offre de cloud computing qui donne aux utilisateurs l'accès à un logiciel basé sur le cloud d'un fournisseur. Quels sont les principaux avantages de l'utilisation du SaaS ?

Chapitre 2 : Analyse initiale et exigence

Ce tableau résume les principaux avantages de l'utilisation des solutions SaaS dans le développement et la gestion des applications.

Avantage	Description
Réduction du délai de rentabilisation	Contrairement au modèle traditionnel, Software as a Service (SaaS) est déjà installé et configuré. Il suffit de provisionner un serveur pour une instance dans le cloud, et en quelques heures, l'application est prête à l'emploi. Cela réduit le temps consacré à l'installation et à la configuration, et peut minimiser les problèmes qui entravent le déploiement du logiciel.
Coûts réduits	Le SaaS peut offrir des économies de coûts intéressantes car il réside généralement dans un environnement partagé ou multi-tenant, où les coûts des licences de matériel et de logiciels sont faibles par rapport au modèle traditionnel. Un autre avantage est que vous pouvez rapidement étendre votre clientèle puisque le SaaS permet aux petites et moyennes entreprises d'utiliser un logiciel qu'elles n'utiliseraient pas autrement en raison du coût élevé des licences. Les coûts de maintenance sont également réduits, car l'environnement appartient au fournisseur SaaS et est réparti entre tous les clients qui utilisent cette solution.
Évolutivité et intégration	Les solutions SaaS résident généralement dans des environnements cloud évolutifs et s'intègrent à d'autres offres SaaS. Par rapport au modèle traditionnel, vous n'avez pas besoin d'acheter un autre serveur ou logiciel. Il vous suffit d'activer une nouvelle offre SaaS et le fournisseur SaaS s'occupe de la planification de la capacité du serveur. Vous avez également la flexibilité d'augmenter ou de diminuer votre utilisation du SaaS en fonction de vos besoins spécifiques.
Nouvelles versions (mises à niveau)	Avec le SaaS, le fournisseur met à niveau la solution et elle devient disponible pour ses clients. Les coûts et les efforts associés aux mises à niveau et aux nouvelles versions sont inférieurs à ceux du modèle traditionnel qui vous oblige généralement à acheter un package de mise à niveau et à l'installer (ou à payer des services spécialisés pour mettre à niveau l'environnement).
Facilité d'utilisation et réalisation de preuves de concept	Les offres SaaS sont faciles à utiliser car elles intègrent déjà les meilleures pratiques et des exemples. Les utilisateurs peuvent réaliser des preuves de concept et tester la fonctionnalité du logiciel ou une nouvelle fonctionnalité avant son lancement. De plus, vous pouvez avoir plusieurs instances avec des versions différentes et effectuer une migration en douceur. Même pour les environnements importants, vous pouvez utiliser des offres SaaS pour tester le logiciel avant de l'acheter.

Tableau 5 : Solution SaaS

b) Adoption Flutter et Firebase dans le développement SaaS

L'utilisation de Flutter et Firebase dans le développement de solutions SaaS (Software as a Service) présente plusieurs avantages notables. Voici un aperçu des deux technologies et de leurs bénéfices dans ce contexte :

Flutter

Flutter est un framework open-source développé par Google pour créer des applications multiplateformes (iOS, Android, web, et desktop) à partir d'un seul code source.

Avantages de Flutter dans le développement SaaS :

1. **Développement multiplateforme** : Écriture d'un seul code source pour plusieurs plateformes, réduisant ainsi le temps et les coûts de développement.
2. **Performance native** : Flutter compile le code en langage natif, offrant une performance proche des applications natives.
3. **Hot Reload** : Permet aux développeurs de voir les changements instantanément sans redémarrer l'application, accélérant ainsi le cycle de développement.
4. **UI riche et personnalisable** : Grâce à son moteur de rendu graphique, Flutter permet de créer des interfaces utilisateur attrayantes et fluides.
5. **Large écosystème de widgets** : Offre une grande variété de widgets pré-construits et personnalisables.

Firebase

Firebase est une plateforme de développement d'applications web et mobiles proposée par Google, qui offre une gamme de services backend tels que l'authentification, la base de données en temps réel, le stockage, l'hébergement, et les notifications push. Utiliser Firebase comme backend pour le développement d'une application web présente plusieurs avantages, particulièrement en conjonction avec une architecture SaaS. Voici les principales raisons pour lesquelles Firebase est une excellente option :

Raison	Description
Facilité d'utilisation	<p>Firebase propose une gamme de services prêts à l'emploi qui simplifient le développement, comme Firestore pour la gestion de base de données en temps réel, Authentication pour la gestion des utilisateurs, et Hosting pour le déploiement d'applications web. Cela permet de démarrer rapidement sans avoir à configurer une infrastructure complexe.</p>
Scalabilité	<p>Firebase est conçu pour évoluer automatiquement avec la demande. Que vous ayez quelques utilisateurs ou des millions, Firebase s'adapte pour gérer le trafic sans nécessiter d'interventions manuelles pour la mise à l'échelle.</p>
Intégration avec Flutter	<p>Firebase offre une excellente intégration avec Flutter, facilitant la création d'applications cross-platforms avec une seule base de code. Cela permet de partager des fonctionnalités entre les versions web, iOS et Android de l'application.</p>
Sécurité	<p>Firebase propose des fonctionnalités de sécurité robustes comme Firebase Authentication pour sécuriser l'accès utilisateur et Firebase Security Rules pour protéger les données dans Firestore et Realtime Database. Cela assure une protection élevée des données des utilisateurs.</p>
Services en Temps Réel	<p>Firestore et Realtime Database de Firebase permettent de synchroniser les données en temps réel, offrant une expérience utilisateur fluide et réactive. Cela est particulièrement utile pour les applications nécessitant des mises à jour en temps réel comme les messageries instantanées ou les tableaux de bord collaboratifs.</p>

Analytique et Surveillance	<p> Firebase Analytics fournit des insights détaillés sur l'utilisation de l'application, permettant d'optimiser l'expérience utilisateur. De plus, Firebase Performance Monitoring et Crashlytics aident à surveiller les performances et à identifier les problèmes rapidement. </p>
Infrastructure sans serveur (Serverless)	<p> Avec Firebase Functions, vous pouvez exécuter du code backend en réponse à des événements déclenchés par Firebase ou d'autres services Google Cloud, sans gérer de serveurs. Cela simplifie le déploiement et la gestion des services backend. </p>
Déploiement et Hébergement	<p> Firebase Hosting offre un déploiement rapide et sécurisé pour les applications web, avec des fonctionnalités comme le SSL automatique et la gestion des versions. Cela simplifie la mise en ligne et la gestion continue des applications. </p>

Tableau 6 : Firebase solution

Intégration de Flutter et Firebase :

L'intégration de Flutter et Firebase est relativement simple grâce aux plugins disponibles. Voici comment ils peuvent être combinés efficacement dans une solution SaaS :

1. **Développement rapide** : Utiliser Flutter pour créer rapidement une interface utilisateur multiplateforme et Firebase pour gérer les services backend permet de réduire considérablement le temps de mise sur le marché.
2. **Synchronisation en temps réel** : Utiliser Firebase Realtime Database ou Firestore pour synchroniser les données entre les utilisateurs en temps réel.
3. **Authentification sécurisée** : Intégrer les services d'authentification Firebase pour gérer les connexions utilisateurs de manière sécurisée.
4. **Notifications et engagement** : Utiliser Firebase Cloud Messaging pour envoyer des notifications et améliorer l'engagement des utilisateurs.
5. **Scalabilité et maintenance** : Utiliser les services de Firebase pour assurer que l'application peut évoluer et rester maintenable facilement.

En combinant Flutter et Firebase, les développeurs peuvent créer des applications SaaS performantes, attrayantes et évolutives tout en réduisant les efforts et les coûts de développement.

8. Conclusion

Ce chapitre a structuré l'approche de développement en définissant un backlog clair, des objectifs mesurables, et des tendances du marché pertinentes. Il a permis de comprendre le contexte et les dynamiques influençant le projet à créer. En précisant les spécifications des besoins, nous avons établi un cadre rigoureux pour aligner nos actions avec les attentes des parties prenantes et les réalités du marché, assurant ainsi une planification et une exécution efficaces.

III. Chapitre 3 : Analyse technique et conception

1. Introduction

Ce chapitre présente une analyse technique approfondie et la conception de notre système. Nous examinerons les aspects technologiques, les choix d'architecture et les différentes étapes de la conception, en nous assurant que chaque composant du système est bien défini et répond aux exigences fonctionnelles et non fonctionnelles.

2. Modélisation des besoins fonctionnels

a) Présentation UML

Les modèles et les diagrammes sont des représentations d'une application réelle. Les modèles fournissent une vue abstraite du système, tandis que différents diagrammes fournissent des représentations concrètes du système.

Les modèles de langage de modélisation unifié (UML) représentent des systèmes à différents niveaux de détail. Certains modèles décrivent un système à partir d'un niveau plus élevé, plus abstrait, tandis que d'autres modèles fournissent plus de détails. Les modèles UML contiennent des éléments de modèle, tels que des acteurs, des cas d'utilisation, des classes et des paquets, et un ou plusieurs diagrammes qui montrent une perspective spécifique d'un système.

b) Diagramme de classe

Dans UML, les diagrammes de classe sont l'un des six types de diagramme structurel. Les diagrammes de classes sont fondamentaux pour le processus de modélisation des objets et modélisent la structure statique d'un système.

Voici le diagramme de classe d'application e-commerce à développer :

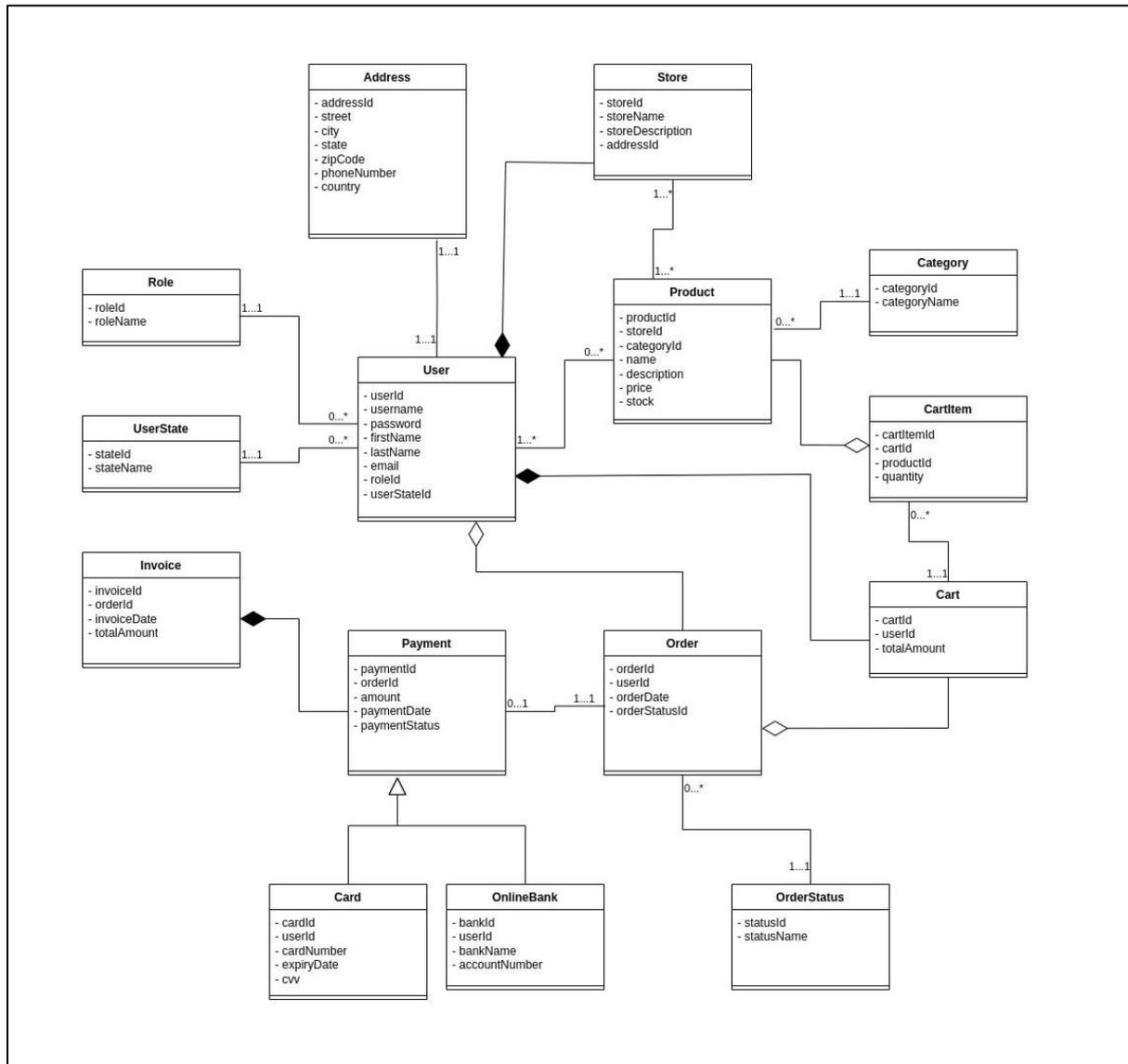


Figure 12 : Conception Uml

Le diagramme de classe présenté illustre les différentes tables et leurs relations dans un système de gestion de commandes en ligne. Chaque entité joue un rôle clé dans la gestion des commandes, des produits, des utilisateurs et des paiements.

✓ **Address**

L'entité Address est associée à une seule entité Store. Cela signifie que chaque adresse

appartient à un magasin spécifique, fournissant les détails nécessaires pour localiser ce magasin.

✓ **Store**

L'entité Store a une relation avec l'entité Address, indiquant qu'un magasin a une adresse unique. De plus, un magasin peut avoir plusieurs produits associés, créant ainsi un lien avec l'entité Product.

✓ **Product**

L'entité Product est reliée à deux autres entités : Store et Category. Chaque produit appartient à une catégorie spécifique et est associé à un magasin particulier. De plus, un produit peut apparaître dans plusieurs éléments de panier, établissant une connexion avec l'entité CartItem.

✓ **Category**

L'entité Category est en relation avec l'entité Product. Une catégorie peut contenir plusieurs produits, ce qui permet une organisation structurée des produits dans différentes catégories.

✓ **User**

L'entité User possède des relations avec plusieurs autres entités : Role, UserState, Cart, Order, Card, et OnlineBank. Chaque utilisateur a un rôle et un état utilisateur spécifiques. De plus, un utilisateur peut posséder un panier, passer des commandes, et avoir des cartes de crédit et des comptes bancaires en ligne.

✓ **Role**

L'entité Role est en relation avec l'entité User, indiquant que chaque rôle peut être assigné à plusieurs utilisateurs. Cela définit les permissions et les niveaux d'accès des utilisateurs.

✓ **UserState**

L'entité UserState est liée à l'entité User, chaque état utilisateur pouvant être assigné à plusieurs utilisateurs. Cela indique le statut actuel de l'utilisateur dans le système.

✓ **Cart**

L'entité Cart est en relation avec l'entité User et l'entité CartItem. Chaque panier appartient à un utilisateur spécifique et peut contenir plusieurs éléments de panier. Cela facilite la gestion des articles ajoutés pour l'achat.

✓ **CartItem**

L'entité CartItem est reliée à deux autres entités : Cart et Product. Chaque élément de panier appartient à un panier et référence un produit, permettant de suivre les quantités et les produits spécifiques dans chaque panier.

✓ **Order**

L'entité Order a des relations avec User, Payment et OrderStatus. Chaque commande est faite par un utilisateur et peut avoir un paiement associé. De plus, chaque commande a un statut, indiquant son état de traitement.

✓ **OrderStatus**

L'entité OrderStatus est liée à l'entité Order. Chaque statut de commande peut être assigné à plusieurs commandes, facilitant le suivi des étapes de traitement des commandes.

✓ **Invoice**

L'entité Invoice est en relation avec l'entité Order. Chaque facture est associée à une commande spécifique, documentant les transactions financières.

✓ **Payment**

L'entité Payment a des relations avec Order, Card, et OnlineBank. Chaque paiement est lié à une commande. Les paiements peuvent être effectués par carte de crédit ou par banque en ligne, offrant ainsi plusieurs options de paiement.

✓ **Card**

L'entité Card est liée à l'entité User. Chaque carte de crédit appartient à un utilisateur, facilitant les transactions par carte de crédit.

✓ **OnlineBank**

L'entité OnlineBank est en relation avec l'entité User. Chaque compte bancaire en ligne appartient à un utilisateur, permettant les paiements en ligne via des comptes bancaires.

Ce diagramme de classe montre comment les différentes entités sont interconnectées pour gérer efficacement les commandes en ligne, les produits, les utilisateurs et les paiements dans un système de commerce électronique. Les relations entre les entités assurent une cohérence et une intégrité des données tout en facilitant la gestion et le traitement des commandes.

c) Diagramme cas d'utilisation

Dans UML, les diagrammes de cas d'utilisation modélisent le comportement d'un système et aident à capturer les exigences du système.

Voici les principaux cas d'utilisation de l'application e-commerce pour différents acteurs:

 Vendeur :

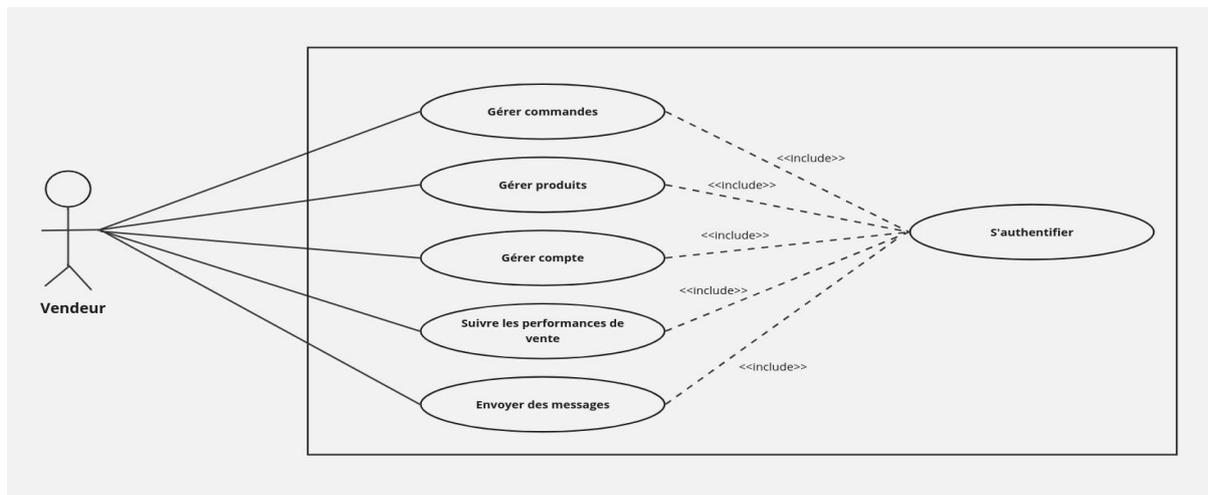


Figure 13 : Usecase vendeur

Ce diagramme est un diagramme de cas d'utilisation montrant les différentes interactions possibles entre un vendeur et le système. Voici une description des cas d'utilisation représentés dans le diagramme ci-dessous :

Ce dernier représente les fonctionnalités qu'un vendeur doit avoir :

✓ **Vendeur :**

L'acteur principal du système, qui interagit avec diverses fonctionnalités.

✓ **Gérer commandes :**

Le vendeur peut gérer les commandes passées par les clients.

✓ **Gérer produits :**

Le vendeur peut gérer les produits qu'il vend, c'est-à-dire ajouter, modifier ou supprimer des produits.

✓ **Gérer compte :**

Le vendeur peut gérer les informations de son compte.

✓ **Suivre les performances de vente :**

Le vendeur peut suivre et analyser les performances de ses ventes, par exemple via des rapports et des statistiques.

✓ **Envoyer des messages :**

Le vendeur peut envoyer des messages, par exemple des notifications ou des communications aux clients.

✓ **S'authentifier :**

- Pour accéder à certaines fonctionnalités comme gérer les commandes, les produits, le compte, et suivre les performances de vente, le vendeur doit d'abord s'authentifier.
- Les cas d'utilisation suivants incluent l'authentification :
 - Gérer commandes
 - Gérer produits
 - Gérer compte
 - Suivre les performances de vente
 - Envoyer des messages

Ce diagramme met en évidence les principales fonctionnalités offertes aux vendeurs et montre les relations entre ces fonctionnalités et l'action d'authentification nécessaire pour certaines d'entre elles.

 Admin:

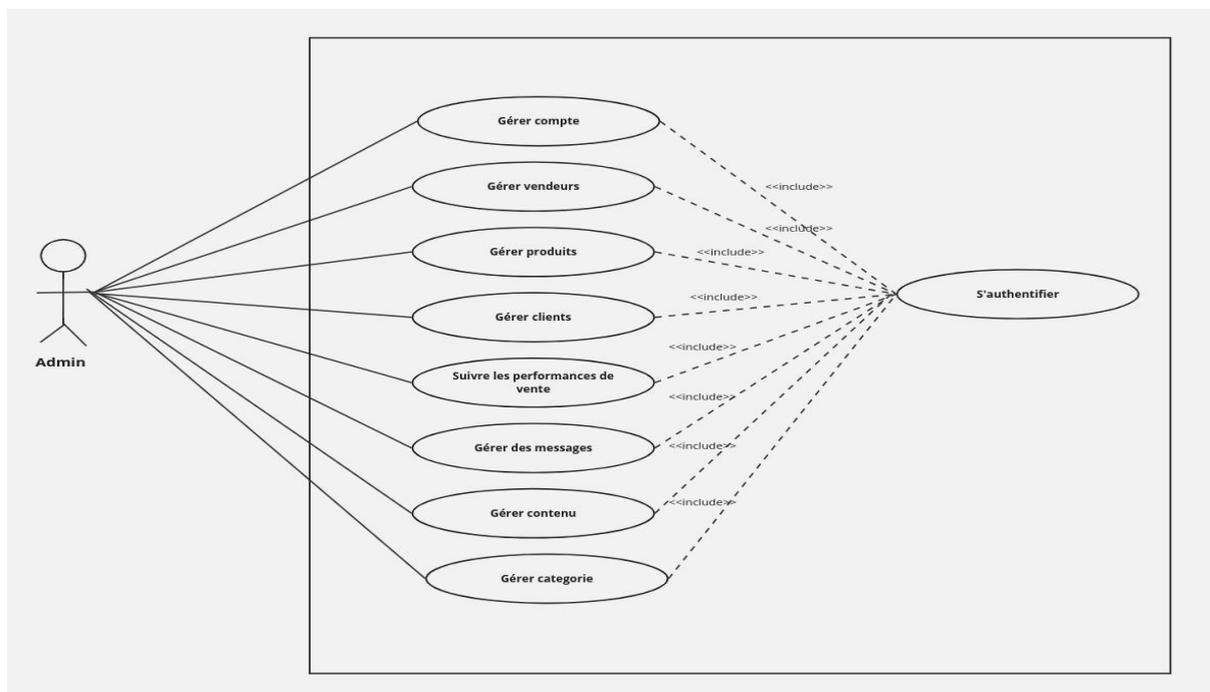


Figure 14 : Usecase admin

Ce diagramme est un diagramme de cas d'utilisation montrant les différentes interactions possibles entre un administrateur (Admin) et le système. Voici une description des cas d'utilisation représentés dans le diagramme ci-dessous :

Ce dernier représente les fonctionnalités qu'un admin doit avoir :

✓ **Gérer compte :**

L'admin peut gérer les informations de son compte.

✓ **Gérer vendeurs :**

L'admin peut gérer les informations et les activités des vendeurs, c'est-à-dire ajouter, modifier ou supprimer des vendeurs.

✓ **Gérer produits :**

L'admin peut gérer les produits proposés sur la plateforme, c'est-à-dire ajouter, modifier ou supprimer des produits.

✓ **Gérer clients :**

L'admin peut gérer les informations et les activités des clients inscrits sur la plateforme.

✓ **Suivre les performances de vente :**

L'admin peut suivre et analyser les performances des ventes, par exemple via des rapports et des statistiques.

✓ **Gérer des messages :**

L'admin peut gérer les messages, par exemple les notifications ou les communications envoyées aux utilisateurs.

✓ **Gérer contenu :**

L'admin peut gérer le contenu disponible sur la plateforme, y compris les articles, les descriptions de produits, et autres informations.

✓ **Gérer catégorie :**

L'admin peut gérer les catégories de produits, c'est-à-dire ajouter, modifier ou supprimer des catégories.

✓ **S'authentifier :**

Pour accéder à certaines fonctionnalités comme gérer le compte, les vendeurs, les produits, les clients, suivre les performances de vente, gérer les messages, le contenu et les catégories, l'admin doit d'abord s'authentifier.

Les cas d'utilisation suivants incluent l'authentification :

- Gérer compte
- Gérer vendeurs
- Gérer produits
- Gérer clients
- Suivre les performances de vente
- Gérer des messages
- Gérer contenu
- Gérer catégorie

Ce diagramme met en évidence les principales fonctionnalités offertes aux administrateurs et montre les relations entre ces fonctionnalités et l'action d'authentification nécessaire pour certaines d'entre elles

 Client:

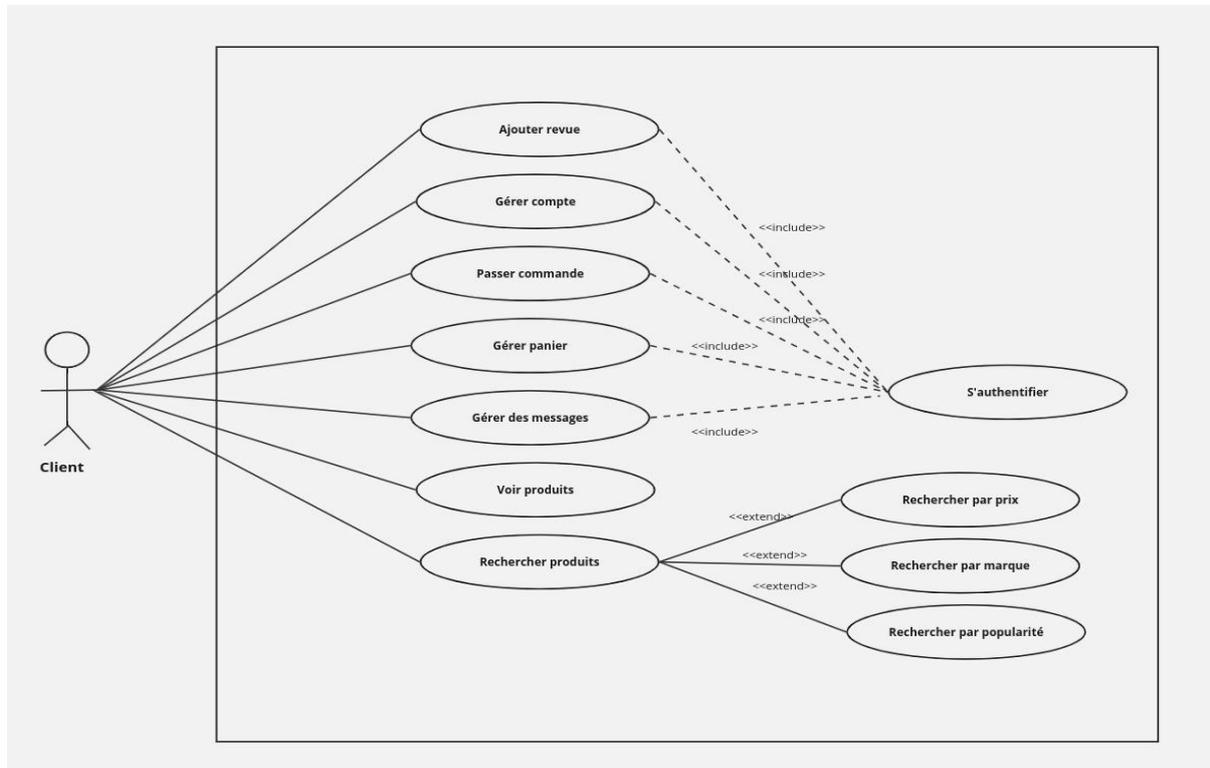


Figure 15 : Usecase client

Ce diagramme est un diagramme de cas d'utilisation qui montre les différentes interactions possibles entre un client et le système. Voici une description des cas d'utilisation représentés dans le diagramme ci-dessous.

Ce dernier représente les fonctionnalités qu'un client doit avoir :

✓ **Ajouter revue :**

Le client peut ajouter une revue sur un produit.

✓ **Gérer compte :**

Le client peut gérer les informations de son compte.

✓ **Passer commande :**

Le client peut passer une commande pour les produits sélectionnés.

✓ **Gérer panier :**

Le client peut gérer son panier, c'est-à-dire ajouter, supprimer ou modifier les produits dans le panier.

✓ **Gérer des messages :**

Le client peut gérer ses messages, par exemple les communications avec le support.

✓ **Voir produits :**

Le client peut voir les produits disponibles dans le système.

✓ **Rechercher produits :**

Le client peut rechercher des produits selon différents critères.

➤ Ce cas d'utilisation est étendu par :

1. Rechercher par prix : Rechercher des produits en fonction de leur prix.
2. Rechercher par marque : Rechercher des produits en fonction de leur marque.
3. Rechercher par popularité : Rechercher des produits en fonction de leur popularité.

✓ **S'authentifier :**

Pour accéder à certaines fonctionnalités comme gérer le compte, passer une commande, gérer le panier, et gérer des messages, le client doit d'abord s'authentifier.

➤ Les cas d'utilisation suivants incluent l'authentification :

1. Ajouter revue
2. Gérer compte
3. Passer commande
4. Gérer panier
5. Gérer des messages

Ce diagramme met en évidence les principales fonctionnalités offertes aux clients et montre les relations entre ces fonctionnalités et les actions d'authentification nécessaires pour certaines d'entre elles.

d) Diagramme de séquence

Un diagramme de séquence est un diagramme UML qui illustre la séquence des messages entre les objets dans une interaction. Un diagramme de séquence se compose d'un groupe d'objets qui sont représentés par des lignes de vie, et les messages qu'ils échangent au fil du temps pendant l'interaction.

Voici les différentes implémentations pour le diagramme de séquence pour l'application e-commerce :

Passer commande

Ce diagramme détaille les étapes nécessaires pour passer une commande :

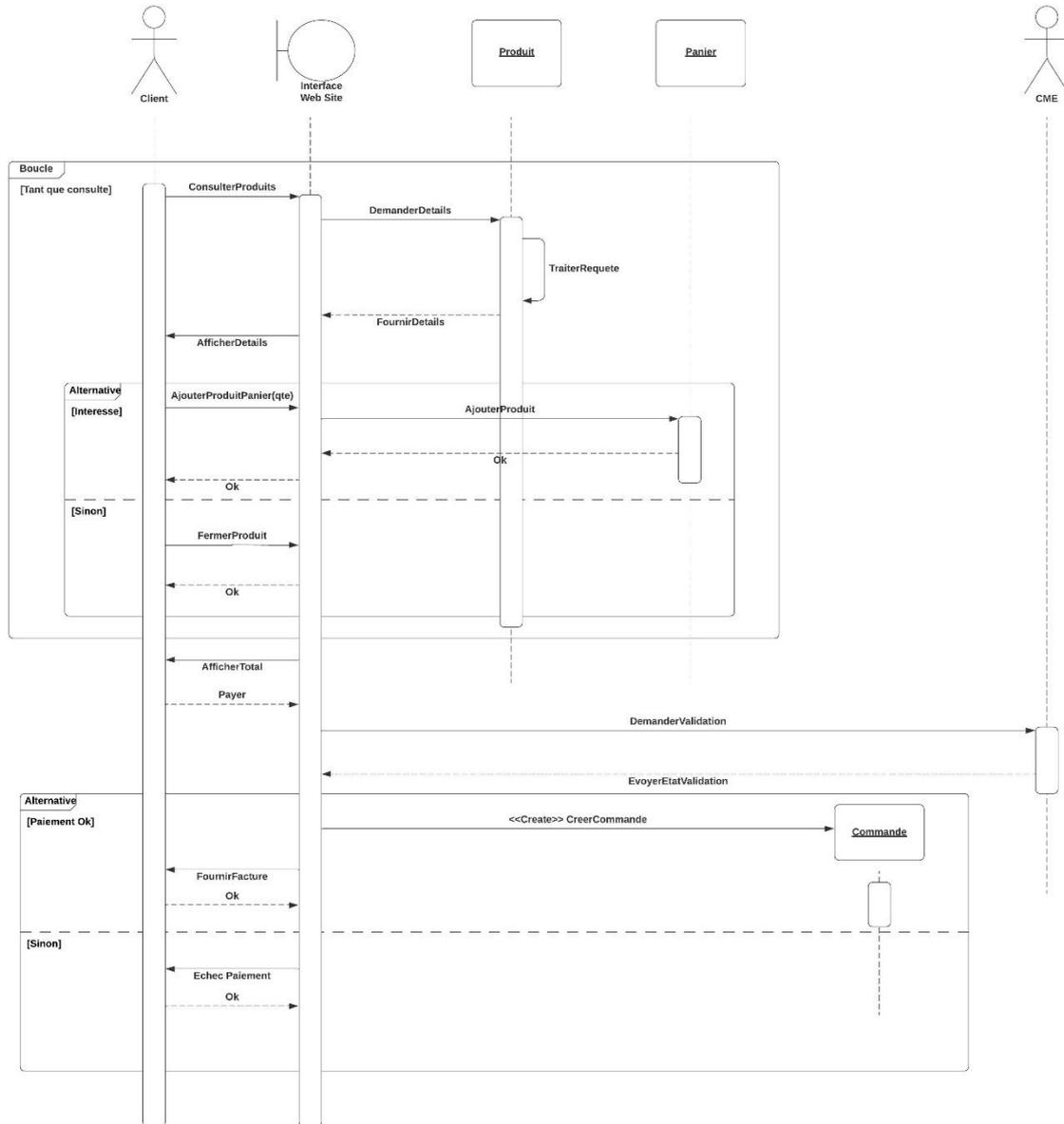


Figure 16 : Diagramme de séquence passer commande

Le diagramme présente un processus détaillé d'interaction entre un client, une interface de site web, des produits, un panier et un Centre Monétique Interbancaire (CME).

✓ Consultation des Produits

- Le client commence par consulter les produits sur l'interface du site web.
- Le client peut demander les détails d'un produit, déclenchant une requête pour obtenir ces informations.
- Une fois les détails reçus, ils sont affichés à l'utilisateur.

✓ **Ajout au Panier**

- Si le client est intéressé par un produit, il peut l'ajouter au panier.
- Cette action envoie une requête pour ajouter le produit, et une confirmation est retournée pour indiquer que le produit a été ajouté avec succès.
- Le client a également la possibilité de fermer la vue du produit après avoir consulté les détails.

✓ **Affichage du Panier et Paiement**

- Après avoir ajouté des produits au panier, le client peut afficher le total de son panier.
- Le client peut ensuite procéder au paiement.

✓ **Validation du Paiement**

- Une fois le paiement initié, une demande de validation est envoyée au CME.
- Si le paiement est validé, une commande est créée.
- Une facture est alors fournie au client.

✓ **Gestion des Échecs**

- En cas d'échec du paiement, le processus de commande est interrompu.
- Un message d'échec est retourné au client.

En résumé, le diagramme illustre un flux d'achat typique sur un site web, depuis la consultation des produits jusqu'à la création et la validation d'une commande via le CME, en passant par l'ajout de produits au panier et le paiement.

Publier produit

Ce diagramme montre les étapes nécessaires pour publier un produit :

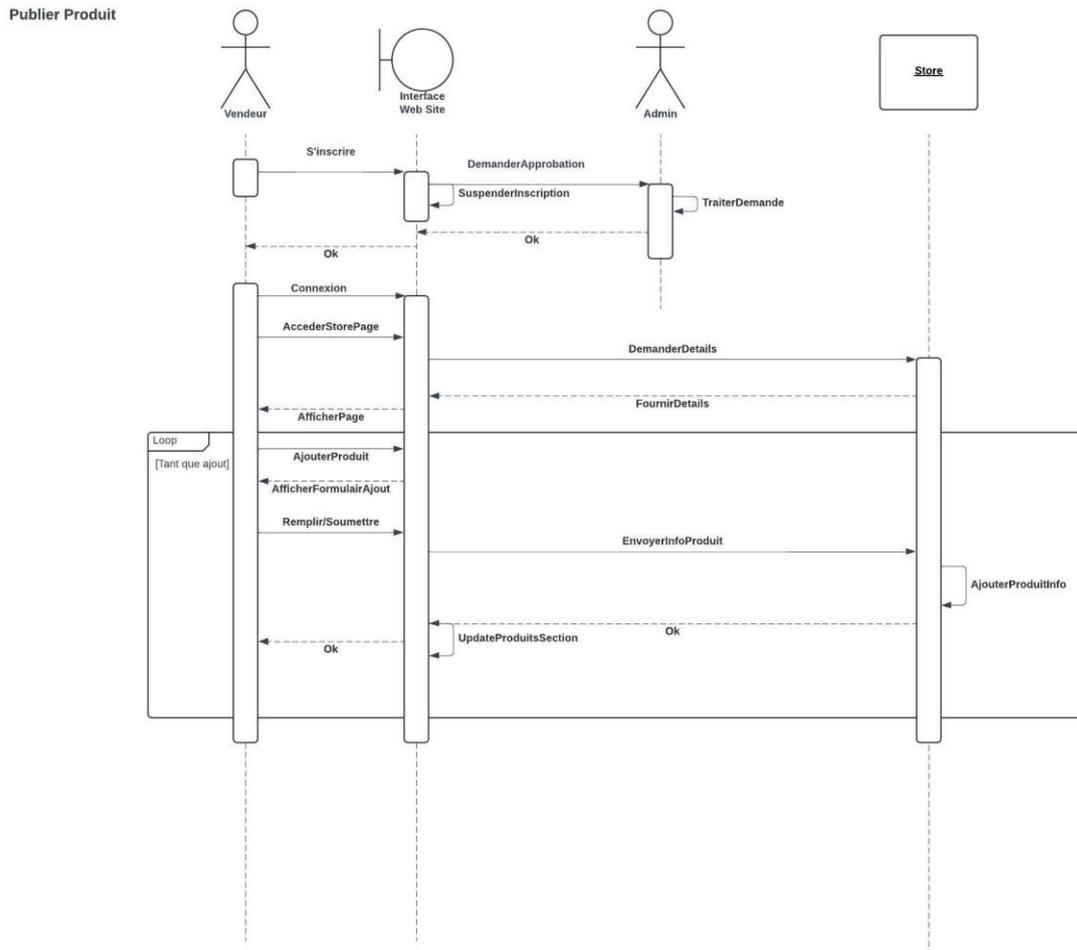


Figure 17 : Diagramme de séquence publier produit

Le diagramme présente le processus de publication d'un produit par un vendeur sur une interface de site web, impliquant également un administrateur et un magasin.

✓ Inscription et Approbation

- Le vendeur commence par s'inscrire sur l'interface du site web.
- Une demande d'approbation est envoyée à l'administrateur.
- L'administrateur traite la demande et peut approuver ou suspendre l'inscription.
- Une confirmation de la décision est renvoyée au vendeur.

✓ Connexion et Accès à la Page de Vente

- Une fois inscrit et approuvé, le vendeur se connecte à l'interface du site web.
- Il accède à la page du magasin pour gérer ses produits.

✓ **Ajout de Produits**

- Le vendeur affiche la page de gestion des produits.
- Il choisit d'ajouter un produit, ce qui affiche un formulaire d'ajout.
- Le vendeur remplit et soumet le formulaire avec les informations du produit.
- Les informations du produit sont envoyées au système pour mise à jour.

✓ **Mise à Jour des détails**

- Les informations du produit sont ajoutées à la base de données du magasin.
- La section des produits est mise à jour pour refléter l'ajout du nouveau produit.
- Une confirmation de la mise à jour réussie est renvoyée au vendeur.

✓ **Boucle d'Ajout de Produits**

- Le processus d'ajout de produits peut être répété tant que le vendeur souhaite ajouter de nouveaux produits.
- Chaque cycle suit les mêmes étapes d'affichage, remplissage, soumission, envoi et mise à jour.

En résumé, le diagramme décrit un flux de publication de produit où un vendeur s'inscrit et est approuvé par un administrateur, se connecte pour accéder à la page de gestion des produits, ajoute de nouveaux produits via un formulaire, et voit les informations de ces produits mises à jour dans le système du magasin. Ce processus peut être répété pour chaque nouveau produit ajouté.

Gérer utilisateurs

Ce diagramme illustre les étapes nécessaires pour gérer un utilisateur :

Gerer Utilisateur

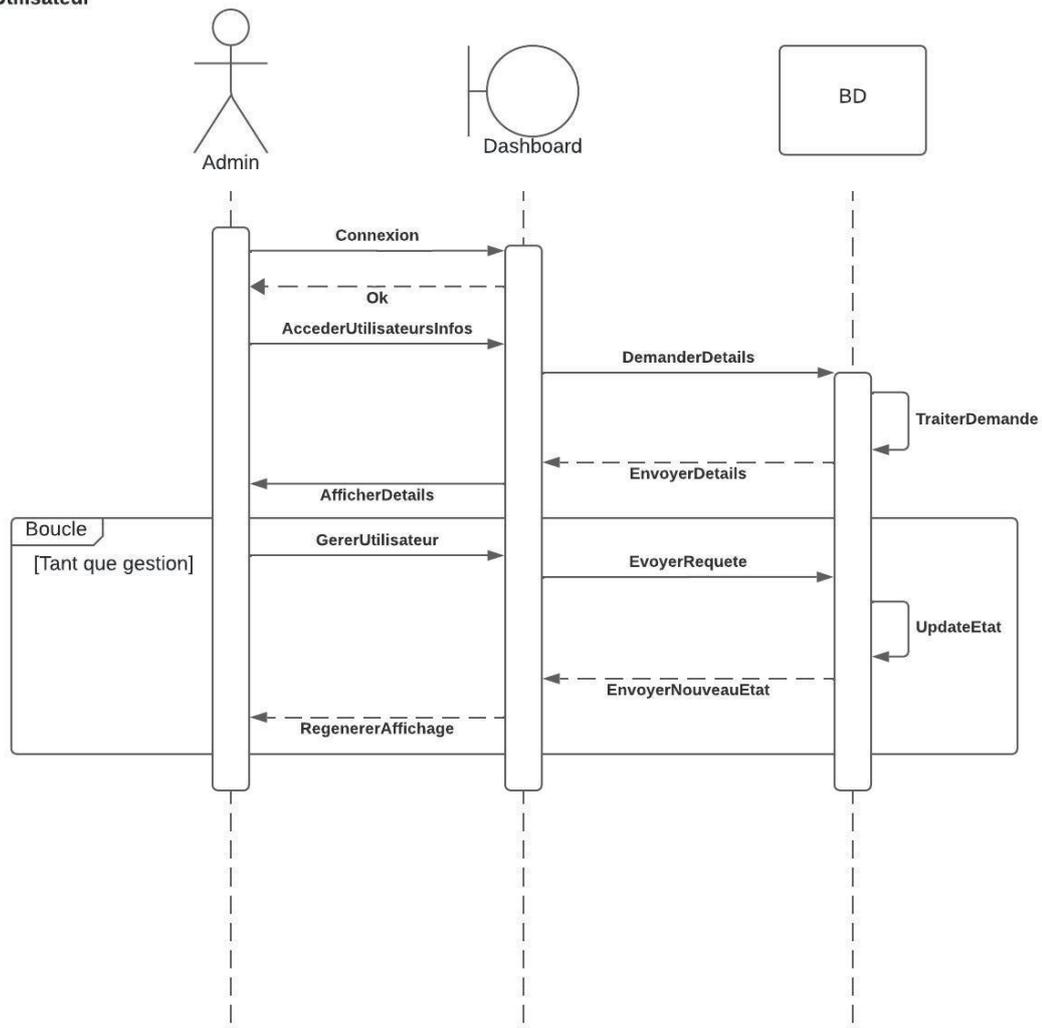


Figure 18 : Diagramme de séquence gérer utilisateur

Le diagramme de séquence ci-joint illustre le processus de gestion des utilisateurs par un administrateur via un tableau de bord, incluant les interactions avec une base de données.

✓ **Connexion et Accès aux Informations :**

- L'administrateur se connecte au tableau de bord (Connexion).
- Une fois connecté, l'administrateur accède aux informations des utilisateurs (AccederUtilisateursInfos), et une confirmation de connexion est envoyée (OK).

- ✓ **Demande de Détails Utilisateurs:**
 - L'administrateur peut demander des détails spécifiques sur les utilisateurs (DemanderDetails).
 - Cette demande est envoyée à la base de données (TraiterDemande), qui renvoie les détails demandés (EnvoyerDetails).
- ✓ **Affichage et Gestion des Utilisateurs :**
 - Les détails des utilisateurs sont affichés (AfficherDetails).
 - L'administrateur procède à la gestion des utilisateurs (GererUtilisateur), ce qui peut impliquer des modifications dans les états des utilisateurs.
- ✓ **Mise à Jour de l'État des Utilisateurs :**
 - Une requête est envoyée à la base de données pour mettre à jour l'état des utilisateurs (EnvoyerRequete).
 - La base de données traite la requête et met à jour l'état (UpdateEtat), puis envoie le nouvel état mis à jour au tableau de bord (EnvoyerNouveauEtat).
- ✓ **Rafraîchissement de l’Affichage:**
 - Le tableau de bord régénère l'affichage pour refléter les modifications apportées aux états des utilisateurs (RegenererAffichage).
- ✓ **Boucle de Gestion:**
 - Le processus de gestion des utilisateurs peut être répété tant que l'administrateur le souhaite (Boucle : Tant que gestion).

Ce diagramme de séquence décrit les interactions essentielles entre l'administrateur, le tableau de bord et la base de données pour la gestion efficace des utilisateurs, incluant la récupération, la modification et l'affichage des informations utilisateur en temps réel.

3. Structure de l’application

a) Application modulaire

Voici une organisation modulaire pour une application e-commerce, en utilisant les acteurs décrits, afin de structurer les fonctionnalités en modules distincts :



Figure 19 : Modularité d'application

La structuration modulaire de l'application e-commerce garantit une organisation claire et une maintenance simplifiée. Chaque module peut être développé, mis à jour, ou remplacé indépendamment, favorisant une évolutivité aisée et une expérience utilisateur optimisée.

a) Architecture de l'application

Ce rapport détaille l'architecture et la mise en œuvre d'une application SaaS de commerce électronique utilisant Flutter Web pour le frontend et Firebase pour le backend. Il couvre la structure du projet, les modèles, les référentiels, les services, les BLoCs, les écrans et la structure de la base de données Firestore. Le rapport inclut également la mise en œuvre des rôles d'utilisateur pour le contrôle d'accès.

b) Aperçu de l'Architecture

Frontend avec Flutter :

1. Dart

Dart est un langage de programmation moderne polyvalent et de haut niveau développé à l'origine par Google. Il est apparu en 2011 comme nouveau langage de programmation, mais sa version stable n'a été publiée qu'en juin 2017. À ses débuts, Dart n'était pas très populaire, mais il a gagné en popularité lorsqu'il a été utilisé avec Flutter.

Dart est un langage de programmation dynamique, orienté objet et basé sur les classes, avec des fermetures et une portée lexicale. Sur le plan syntaxique, il est assez similaire à Java, C et JavaScript. Si vous connaissez l'un de ces langages de programmation, vous pouvez facilement apprendre le langage de programmation Dart.

Dart est un langage de programmation open source largement utilisé pour développer des applications mobiles, des applications web modernes, des applications de bureau et l'Internet des objets (IoT) en utilisant le framework Flutter. Il prend également en charge quelques concepts avancés tels que les interfaces, les mixins, les classes abstraites, les generics réfléchis et l'inférence de types. C'est un langage compilé qui prend en charge deux types de techniques de compilation.

AOT (Ahead of Time) - Il convertit le code Dart en code JavaScript optimisé à l'aide du compilateur dar2js et s'exécute sur tous les navigateurs web modernes. Il compile le code au moment de la construction.

JIT (Just-In-Time) - Il convertit le bytecode en code machine (code natif), mais seulement le code qui est nécessaire.

Voici une image qui resume la différence :

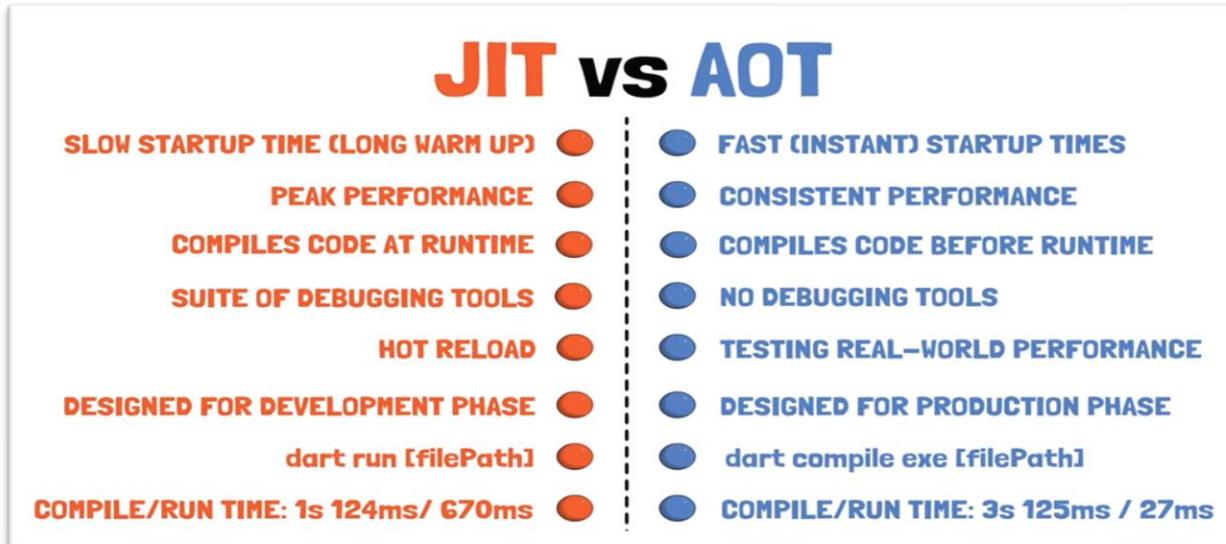


Figure 20 : Ahead of time vs Just in time

Voici des fondamentaux concepts en Dart:

1. Tout en Dart est traité comme un objet, y compris les nombres, les booléens, les fonctions, etc. Tous les objets héritent de la classe Object.
2. Les outils Dart peuvent signaler deux types de problèmes lors de la programmation : les avertissements et les erreurs. Les avertissements indiquent qu'il peut y avoir un problème dans votre code, mais cela n'interrompt pas l'exécution du code, tandis que les erreurs peuvent empêcher l'exécution du code.
3. Dart prend en charge le typage fort.
4. Dart prend en charge les types génériques, comme `List<int>` (une liste d'entiers) ou `List<dynamic>` (une liste d'objets de n'importe quel type).

1. Flutter

Flutter est un kit d'outils UI permettant de créer des applications compilées de manière native pour mobile, web et bureau à partir d'une seule base de code. Conçu comme un système extensible et en couches, Flutter se compose de plusieurs bibliothèques indépendantes, où chaque bibliothèque dépend de la couche sous-jacente. Aucune couche n'a un accès privilégié à la couche inférieure, et chaque partie du Framework est conçue pour être optionnelle et remplaçable.

Cette figure illustre les éléments essentiels de flutter :

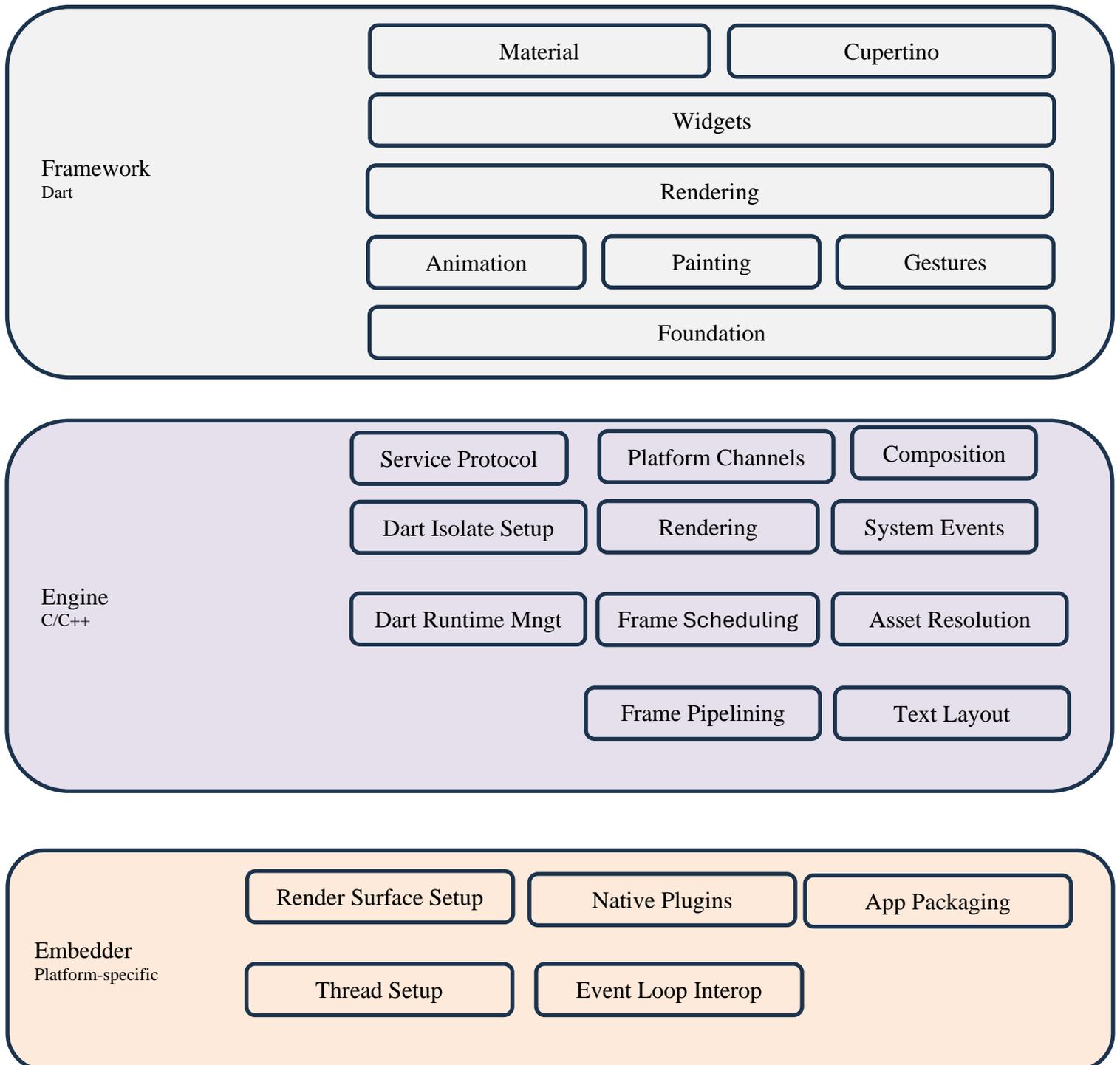
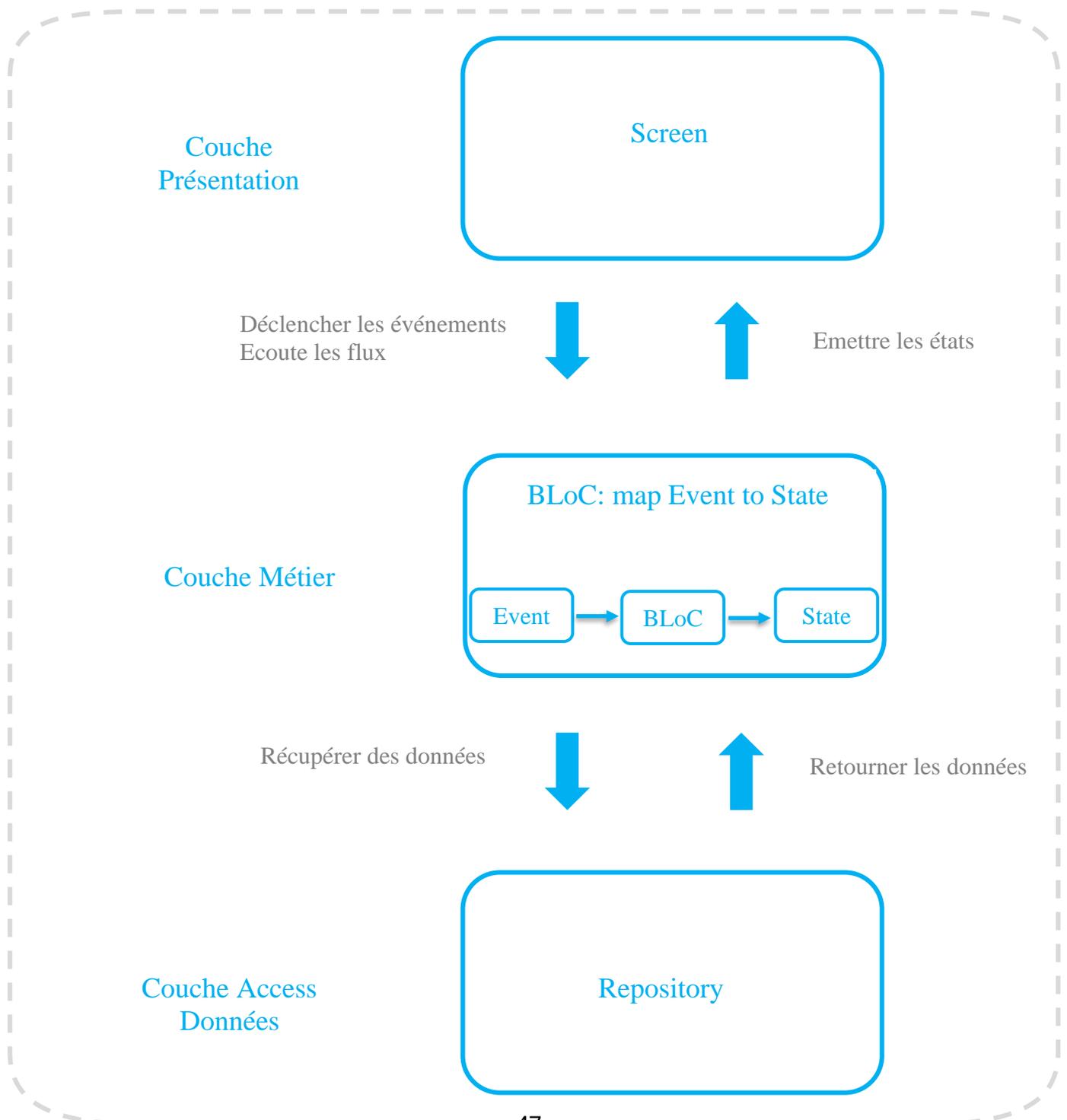


Figure 21 : Architecture Flutter

Gestion des états avec BLoC :

Pour la gestion des états, Flutter propose le pattern BLoC (Business Logic Component). BloC fournit un moyen prévisible et structuré de gérer les états et de séparer la présentation de la logique métier.

Voici une figure illustrative d'architecture de l'application suivant BloC :



Ci-dessous les principales caractéristiques du pattern BloC :

Séparation des préoccupations : Le pattern BloC aide à séparer la logique métier de la présentation, ce qui rend le code plus propre et maintenable.

Gestion réactive des états : Utilise des Streams pour gérer les événements et les états de manière réactive.

Testabilité améliorée : Grâce à la séparation claire entre la logique métier et l'interface utilisateur, il est plus facile de tester chaque composant indépendamment.

Scalabilité : BloC est particulièrement adapté pour les applications de grande envergure où la gestion complexe des états est nécessaire.

En résumé, le pattern BloC dans Flutter permet de maintenir une interface utilisateur cohérente et réactive tout en gérant les états de manière claire et structurée, facilitant ainsi le développement et la maintenance des applications.

Conception d'architecture d'application en utilisation Bloc, l'architecture suivante comporte trois couches :

Couche de données et domaine :

Cette couche est celle chargée d'interagir avec les API. C'est elle qui est chargée de transformer les données qui proviennent des API en des modèles. Et enfin, nous voulons gérer l'état de ces données et les présenter sur notre interface utilisateur, c'est pourquoi nous divisons la couche de présentation en deux :

Couche de logique métier :

Cette couche gère l'état (généralement à l'aide de flutter_bloc).

Couche de présentation :

Rend les composants de l'interface utilisateur en fonction de l'état.

Backend avec Firebase

Firestore est un ensemble d'outils sans serveur, également de Google comme **Flutter**, et il nous aidera à créer et à développer notre **application** dans de nombreux termes différents. L'image suivante est très descriptive et nous aidera à comprendre que travailler avec Firebase nous aide à supprimer une partie assez complexe de notre application (le serveur) de l'équation dans un premier temps, en plus d'ajouter de la sécurité et de l'agilité.

Voici une image illustrative de l'architecture Backend avec Firebase :

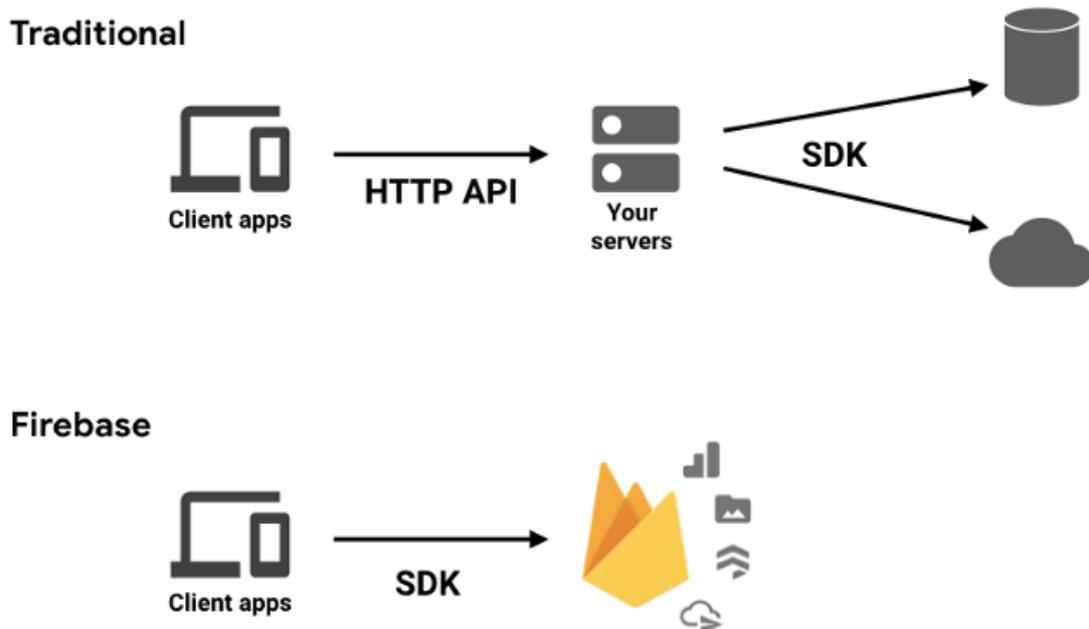


Figure 23 : Architecture traditionnelle vs Firebase

- **Firestore** : Une base de données flexible et évolutive pour le développement mobile, web et serveur. Firestore permet de stocker, synchroniser et interroger des données pour les applications en temps réel, tout en offrant une structure de données hiérarchique.
- **Authentication** : Fournit des services backend pour authentifier les utilisateurs via divers moyens, tels que les adresses e-mail/mots de passe, les fournisseurs tiers comme Google, Facebook, et autres. Authentication gère les processus de connexion, d'inscription et de gestion des sessions utilisateur de manière sécurisée et simple.
- **Storage** : Un service de stockage d'objets qui permet de stocker et servir du contenu généré par les utilisateurs, comme des images, des vidéos et d'autres fichiers volumineux. Storage est intégré avec Authentication pour contrôler l'accès et les permissions des utilisateurs pour charger et télécharger des fichiers.
- **Security Rules** : Utilisé pour appliquer des contrôles d'accès et garantir l'intégrité des données dans Firestore et Storage. Ces règles permettent de définir qui peut lire ou écrire dans la base de données ou le stockage, et

sous quelles conditions. Elles assurent que seules les opérations autorisées sont effectuées par les utilisateurs authentifiés ou anonymes.

4. Structure du Projet Flutter à développer

a) Structure du Projet

Voici la structure de l'application ainsi développée :

```

lib/
├─ main.dart
├─ models/
├─ repositories/
├─ services/
├─ blocs/
├─ screens/
├─ widgets/
├─ i10n/
├─ router/
├─ theme/
└─ utils/
  
```

Figure 24 : Structure du projet

Explication Détaillée

- **main.dart** : Le point d'entrée de l'application.
- **models/** : Contient les modèles de données représentant les entités dans l'application.
- **repositories/** : Gère la récupération et la persistance des données.
- **services/** : Contient la logique métier et interagit avec les référentiels.
- **blocs/** : Contient les classes BLoC pour la gestion des états.
- **screens/** : Contient les écrans UI de l'application.
- **widgets/** : Contient des composants UI réutilisables.
- **i10n/** : Contient les fichiers de internalisation.
- **router/** : Gère le routage de l'application.
- **utils/** : Contient des fonctions utilitaires et des constantes.

b) Structure de Firestore Firebase

Firestore est un service de base de données NoSQL entièrement géré par Google Cloud Platform. Voici les principaux éléments de Firestore :

1. **Collections** : Ce sont des groupes de documents, similaires à des tables dans une base de données relationnelle.
2. **Documents** : Chaque document est une unité de données dans une collection. Il est stocké au format JSON et contient des champs et leurs valeurs.
3. **Champs** : Ce sont les différentes clés et valeurs dans un document. Chaque champ peut contenir différents types de données comme des chaînes de caractères, des nombres, des tableaux, etc.
4. **Index** : Firestore crée automatiquement des index pour chaque champ simple dans votre base de données pour faciliter les requêtes rapides.

5. **Requêtes** : Vous pouvez interroger Firestore pour récupérer des documents basés sur des critères spécifiques en utilisant une syntaxe de requête flexible.

c) Stockage Firebase

Pour les images associées aux produits et aux profils utilisateur, Firebase Storage est utilisé pour stocker ces fichiers.

Voici comment vous pouvez organiser votre structure de stockage :

- **Répertoire racine:** /
 - **Sous-répertoire:** users/
 - profile.jpg (exemple de nom de fichier pour l'image de profil utilisateur)
 - **Sous-répertoire:** products/
 - product.jpg (exemple de nom de fichier pour l'image du produit)

d) Règles de Sécurité Firestore

Les règles de sécurité Firestore définissent les autorisations d'accès aux données et sont cruciales pour garantir la sécurité et l'intégrité de votre base de données. Voici les points principaux à comprendre à propos des règles de sécurité Firestore :

1. **Lecture (read)** : Ces règles déterminent qui peut lire les données dans une collection ou un document. Vous pouvez spécifier des conditions basées sur l'authentification de l'utilisateur, le contenu des données ou d'autres paramètres. Par exemple, restreindre l'accès aux documents basés sur des champs spécifiques ou sur l'identité de l'utilisateur.
2. **Écriture (write)** : Ces règles contrôlent qui peut écrire (ajouter, modifier ou supprimer) des données dans une collection ou un document. Comme pour les règles de lecture, vous pouvez spécifier des conditions complexes pour sécuriser l'écriture des données.
3. **Conditions** : Les règles peuvent utiliser des conditions booléennes pour vérifier des attributs tels que l'authentification de l'utilisateur (request.auth), les données envoyées (request.resource.data), et les données actuelles dans Firestore (resource.data). Cela permet de définir des politiques d'accès précises.
4. **Stratégies de sécurité** : Il est recommandé de définir des règles de sécurité aussi restrictives que possible tout en répondant aux besoins fonctionnels de votre application. Par exemple, limiter l'accès en lecture et en écriture aux utilisateurs authentifiés ou restreindre l'accès à certains documents basés sur des critères spécifiques.

e) Implémentation des Rôles

Pour gérer les rôles dans une application Flutter avec Firestore et Firebase Authentication :

1. **Authentification avec Firebase** : Utilisez firebase_auth pour gérer l'authentification des utilisateurs.
2. **Attribution des rôles** : Après l'authentification, attribuez des rôles à chaque utilisateur en utilisant Firestore ou une base de données supplémentaire.
3. **Utilisation de BLoC (Business Logic Component)** : Utilisez le pattern BLoC pour encapsuler la logique métier de votre application, y compris la gestion des rôles et des autorisations.

Exemple de structure : Création d'un AuthRepository pour gérer l'authentification et récupérer les informations utilisateur, telles que les rôles. Modélisation des données utilisateur avec une classe comme UserModel pour inclure les rôles.

4. **Sécurité et autorisations** : Configuration des règles de sécurité dans Firestore en fonction des rôles attribués pour restreindre l'accès aux données sensibles en toute sécurité.

5. Outils et technologies

a) Langages :

Dart :



Figure 25 : Logo Dart

Est un langage de programmation open source et évolutif, avec des bibliothèques et des environnements d'exécution robustes, pour la création d'applications Web, serveur et mobiles.

Accessible : Développez avec un langage de programmation fortement typé, cohérent, concis et offrant des fonctionnalités de langage moderne comme sécurité nulle et design patterns.

Portable : compilez au format ARM, x64 ou RISC-V pour mobile, ordinateur de bureau et backend. Compilez en JavaScript ou WebAssembly pour le web.

Productif : apportez des modifications de manière itérative : utilisez hot reload pour voir le résultat instantanément dans votre application en cours d'exécution. Diagnostiquer les problèmes d'application à l'aide de DevTools.

Dart Native : Pour les programmes ciblant les appareils (mobile, web, ordinateur de bureau, serveur, etc.), Dart Native comprend à la fois une machine virtuelle Dart avec une compilation JIT (just-in-time) et un compilateur AOT (ahead-of-time) pour produire du code machine.

Dart Web : Pour les programmes ciblant le Web, Dart Web inclut à la fois un compilateur de temps de développement (dartdevc) et un compilateur de temps de production (dart2js).

b) Framworks

Flutter :



Figure 26 : Logo Flutter

Est la boîte à outils d'interface utilisateur de Google pour créer de belles applications compilées nativement pour les appareils mobiles, Web, de bureau et embarqués à partir d'une seule base de code.

Performances proches du natif : Flutter utilise le langage de programmation Dart et se compile en code machine. Les appareils hôtes comprennent ce code, ce qui garantit des performances rapides et efficaces.

Rendu rapide, cohérent et personnalisable : Au lieu de s'appuyer sur des outils de rendu spécifiques à la plate-forme, Flutter utilise la bibliothèque graphique open source Skia de Google pour rendre l'interface utilisateur. Cela fournit aux utilisateurs des visuels cohérents, quelle que soit la plate-forme qu'ils utilisent pour accéder à une application.

Outils conviviaux pour les développeurs : Google a construit Flutter en mettant l'accent sur la facilité d'utilisation. Avec des outils tels que hot reload, les développeurs peuvent prévisualiser à quoi ressembleront les modifications de code sans perdre l'état. D'autres outils comme l'inspecteur de widgets facilitent la visualisation et la résolution des problèmes liés aux mises en page de l'interface utilisateur.

c) Outils de développement

Firebase :



Figure 27 : Logo Firebase

Est une plateforme de développement d'applications Backend-as-a-Service (BaaS) qui fournit des services backend hébergés tels qu'une base de données en temps réel, le stockage dans le cloud, l'authentification, les rapports de plantage, l'apprentissage automatique, la configuration à distance et l'hébergement de vos fichiers statiques.

Authentication

Ajoutez facilement l'authentification des utilisateurs pour la connexion et l'intégration à partir de divers fournisseurs.

Cloud Firestore

Créez des applications réactives avec une base de données de documents NoSQL qui vous permet de structurer les données comme vous le souhaitez et de les récupérer avec des requêtes expressives.

Cloud Storage

Stockez et diffusez du contenu généré par les utilisateurs au fur et à mesure que votre application passe du prototype à la production.

IntelliJ IDEA :



Figure 28 : Logo IntelliJ

Est un environnement de développement intégré (IDE) puissant pour la création d'applications Java, mais aussi compatible avec d'autres langages de programmation telle que Dart. Il offre des fonctionnalités avancées telles que la complétion automatique du code, le débogage la refactoring et l'intégration avec des outils de construction et de gestion de versions. IntelliJ IDEA facilite le développement en offrant des fonctionnalités intelligentes et en mettant l'accent sur la productivité des développeurs.

d) Outils de collaboration

Git :



Figure 29 : Logo git

Est un système de contrôle de version distribué largement utilisé dans le développement logiciel. Il permet de gérer les différentes versions d'un projet en enregistrant les modifications apportées aux fichiers et en facilitant la collaboration entre les membres d'une équipe de développement.

e) Outils de planification et modélisation

GanttProject :



Figure 30 : Logo GanttProject

Est un logiciel de gestion de projet open-source qui permet aux utilisateurs de créer et de gérer des diagrammes de Gantt. Il aide à la planification, à l'ordonnancement et au suivi des tâches et des ressources dans un projet. GanttProject offre des fonctionnalités telles que les dépendances de tâches, les jalons, l'allocation des ressources et l'analyse du chemin critique, ce qui le rend utile pour la planification et l'exécution de projets.

StarUML :



Figure 31 : Logo StarUML

Est un outil de modélisation et de diagramme utilisé pour concevoir et visualiser des systèmes logiciels. Il prend en charge différentes notations de modélisation, notamment UML (Unified Modeling Language), et propose une large gamme de diagrammes tels que les diagrammes de classes, les diagrammes de séquence et les diagrammes d'activités. StarUML aide les développeurs et les concepteurs de logiciels à créer des représentations claires et complètes de leurs systèmes.

6. Conclusion

En conclusion, ce chapitre a fourni une analyse technique détaillée et une conception méticuleuse de notre système. Nous avons exploré les aspects technologiques, justifié les choix d'architecture et décrit les différentes étapes de la conception. Chaque composant du système a été soigneusement défini pour s'assurer qu'il répond aux exigences fonctionnelles et non fonctionnelles identifiées. Grâce à cette analyse et à cette conception rigoureuse, nous avons posé les bases solides nécessaires pour le développement et la mise en œuvre efficace de notre système.

IV. Chapitre 4 : Mise en œuvre

1. Introduction

Après avoir analysé et spécifié les besoins globaux des fonctionnalités, maintenant c'est le moment d'implémentation de la solution proposée. Cette section comporte les détails du processus de développement qui est divisé en sprints. Les quatre premiers sprints concernent le développement d'une solution SaaS appliqué au commerce, tandis que les deux derniers sprints sont dédiés au développement d'une solution SaaS dans le domaine d'éducation.

2. Sprint 1 : Développement module authentification et internalisation

a) Planification

▼ Sprint1: Développement modules authentification et internalization	3/25/24	4/9/24
Analyse et conception	3/25/24	3/25/24
Création module authentification	3/25/24	3/27/24
Création module internalization	3/27/24	4/9/24

Figure 32 : Partition Sprint 1

Le premier sprint s'est concentré sur le développement des modules d'authentification et d'internalisation.

La première étape a été l'analyse et la conception, définissant les spécifications techniques et fonctionnelles. Ensuite, nous avons créé le module d'authentification, incluant les fonctionnalités d'inscription et de connexion. Enfin, nous avons développé le module d'internalisation pour adapter l'application à plusieurs langues et régions, améliorant ainsi l'accessibilité et l'expérience utilisateur.

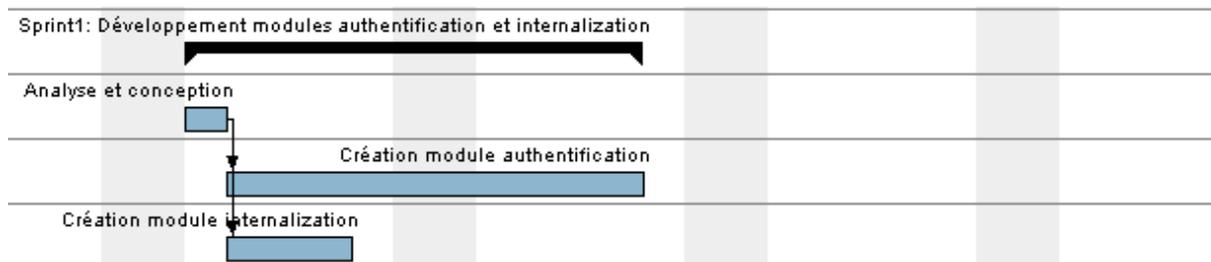


Figure 33 : Planification Sprint 1

b) Conception

Au cours du premier sprint, nous avons développé les modules d'authentification et d'internalisation.

Module d'authentification : Ce module a été conçu pour sécuriser l'accès à l'application. Il inclut des fonctionnalités telles que l'inscription des utilisateurs, la connexion sécurisée, la gestion des mots de passe.

Module d'internalisation : Ce module a permis de rendre l'application accessible à un public international. Dans un premier temps, il comprend le support multilingue, par la suite il est prévu l'adaptation culturelle des formats de date et de monnaie.

Ces développements assurent une application sécurisée et conviviale pour les utilisateurs du monde entier et du coup, elle s'inscrit dans le contexte du développement SaaS.

Module authentification :

Se connecter avec e-mail et mot de passe :

Cas d'utilisation	S'authentifier
Acteur	Utilisateur
Scénario minimale	<p>L'utilisateur entre son nom d'utilisateur et son mot de passe.</p> <p>Le système vérifie si les informations sont valides et correspondantes dans la base de données.</p> <p>Si les informations sont correctes, l'utilisateur est connecté.</p>
Scénario alternatif	<p>Si les informations ne sont pas valides, un message d'erreur est affiché indiquant que l'authentification a échoué.</p>

Tableau 7 : Scénario d'authentification

S'inscrire avec e-mail et mot de passe :

Cas d'utilisation	S'authentifier
Acteur	Utilisateur
Scénario minimale	<p>L'utilisateur accède à la page d'inscription de l'application.</p> <p>Il remplit un formulaire avec les informations requises, telles que nom d'utilisateur, adresse e-mail, et mot de passe.</p> <p>Il soumet le formulaire pour créer un nouveau compte.</p> <p>Le système vérifie que l'adresse e-mail est unique et valide.</p> <p>Il vérifie également que le nom d'utilisateur n'est pas déjà utilisé.</p> <p>Si toutes les données sont valides, un nouveau compte est créé dans la base de données avec les informations fournies par l'utilisateur.</p> <p>Une fois le nouveau compte créé avec succès, l'utilisateur est redirigé vers la page de connexion pour utiliser ses nouveaux identifiants.</p>
Scénario alternatif	<p>Si les informations ne sont pas valides, un message d'erreur est affiché indiquant que l'authentification a échoué.</p>

Tableau 8 : Scénario de souscription

Authentification avec Google :

L'application possède une authentification OAuth 2.0 avec la service Google, voila les scenarios possibles :

Cas d'utilisation	S'authentifier
Acteur	Utilisateur
Scenarion minimale	L'utilisateur choisit de s'authentifier via Google. Le système le redirige alors vers la page de connexion de Google, où il peut entrer ses identifiants s'il possède déjà un compte, ou créer un compte Google s'il n'en a pas encore. Une fois authentifié, Google renvoie au système les informations de l'utilisateur, telles que l'ID unique de Google. Le système vérifie si cet ID est déjà associé à un compte dans sa base de données. Si l'utilisateur existe déjà, il est connecté. Sinon, un nouveau compte est créé pour lui, qu'il pourra utiliser ultérieurement.
Scenarion alternatif	Si les informations ne sont pas valides, un message d'erreur est affiché indiquant que l'authentification a échoué.

Tableau 9 : Scenarion d'authentification avec Google

c) Réalisation

Authentification avec username et mot de passe :

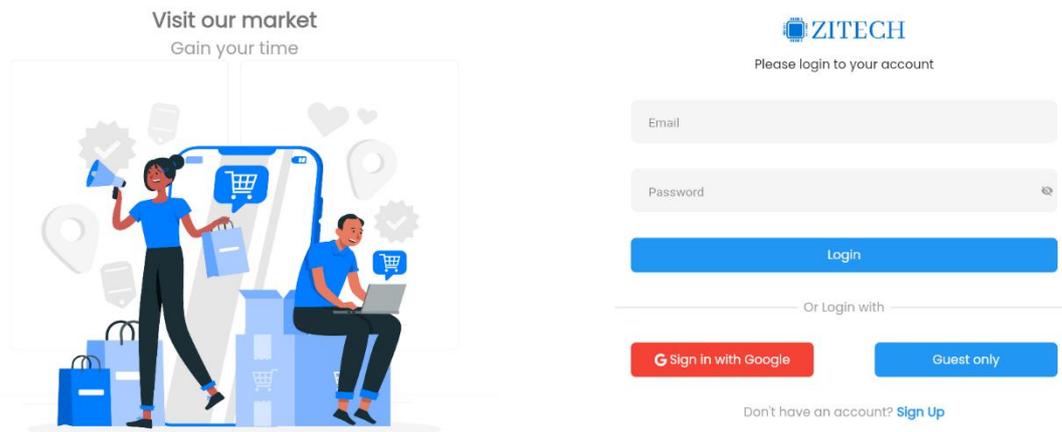


Figure 34 : Authentification avec username et mot de passe

La page de connexion de l'application de marché en ligne ZITECH présente les fonctionnalités suivantes :

1. Formulaire de Connexion Standard :

- ✓ **Email** : Champ de saisie pour entrer l'adresse email de l'utilisateur.
- a. **Mot de passe** : Champ de saisie pour entrer le mot de passe, avec une option pour afficher ou masquer le mot de passe.

2. Bouton de Connexion :

- a) **Login** : Bouton pour soumettre les informations d'identification et accéder au compte de l'utilisateur.

3. Options de Connexion Alternatives :

- a) **Connexion avec Google** : Bouton pour se connecter via un compte Google, facilitant l'accès pour les utilisateurs ayant un compte Google.
- b) **Accès Invité** : Bouton "Guest only" permettant d'accéder à l'application en tant qu'invité, sans avoir besoin de créer un compte.

4. Lien d'Inscription :

- a) **Sign Up** : Lien pour les nouveaux utilisateurs souhaitant créer un compte.

Ces fonctionnalités assurent une expérience utilisateur fluide et multiple, offrant plusieurs moyens d'accès à l'application selon les préférences et besoins des utilisateurs.

Souscription avec username et mot de passe :

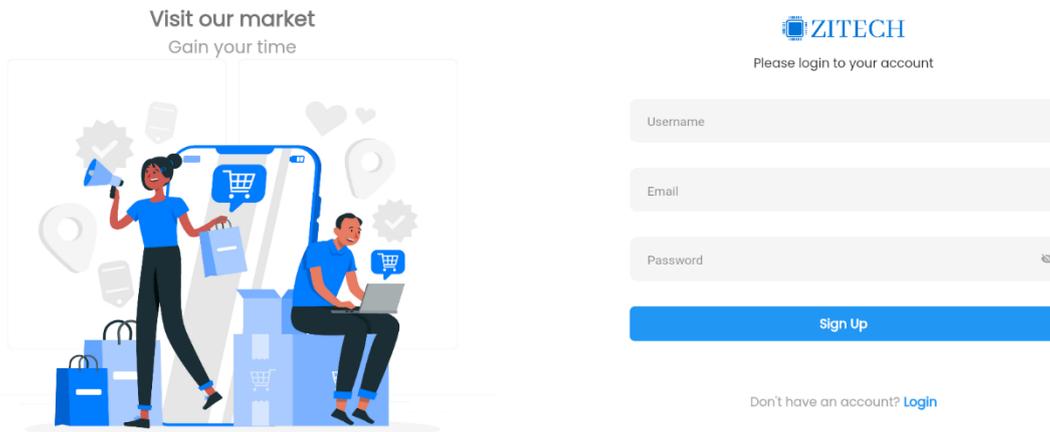


Figure 35 : Souscription avec username et mot de passe

La page d'inscription de l'application de marché en ligne ZITECH comporte les fonctionnalités suivantes :

1. **Formulaire d'Inscription :**

- Nom d'utilisateur : Champ de saisie pour entrer le nom d'utilisateur.
- Email : Champ de saisie pour entrer l'adresse email de l'utilisateur.
- Mot de passe : Champ de saisie pour entrer le mot de passe, avec une option pour afficher ou masquer le mot de passe.

2. **Bouton d'Inscription :**

- Sign Up : Bouton pour soumettre les informations et créer un nouveau compte utilisateur.

3. **Lien de Connexion :**

- Login : Lien pour les utilisateurs existants permettant de se rendre à la page de connexion.

Ces fonctionnalités permettent aux nouveaux utilisateurs de créer facilement un compte et aux utilisateurs existants de se connecter s'ils ont déjà un compte

Authentification avec Google :

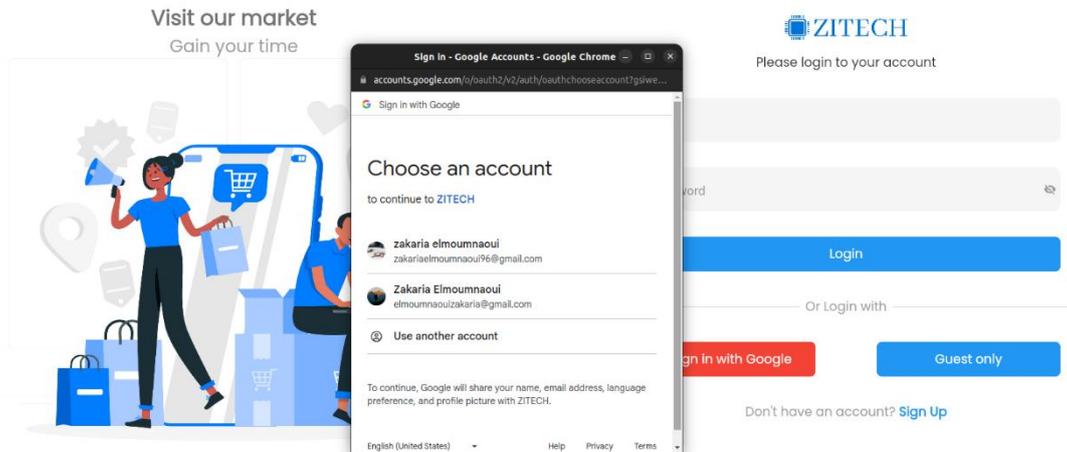


Figure 36 : Authentification avec Google

La page de connexion de l'application de marché en ligne ZITECH, incluant la connexion avec Google, propose les fonctionnalités suivantes :

1. Connexion avec Google :

- a) **Sign in with Google** : Bouton rouge permettant de se connecter via un compte Google.
- b) **Choix du Compte Google** : Une fenêtre pop-up s'ouvre pour permettre à l'utilisateur de choisir parmi les comptes Google disponibles ou d'utiliser un autre compte.
- c) **Informations Partagées** : La fenêtre indique que Google partagera le nom, l'adresse email, les préférences linguistiques et l'image de profil de l'utilisateur avec ZITECH.

La connexion via Google simplifie le processus en utilisant des informations déjà existantes, ce qui peut encourager plus d'utilisateurs à s'inscrire et à se connecter

Module internalisation :

Cette version de l'application offre un paquet de deux langues largement utilisé, la langue française et la langue anglaise.

Voici la version française de l’application pour la page ‘A propos’:

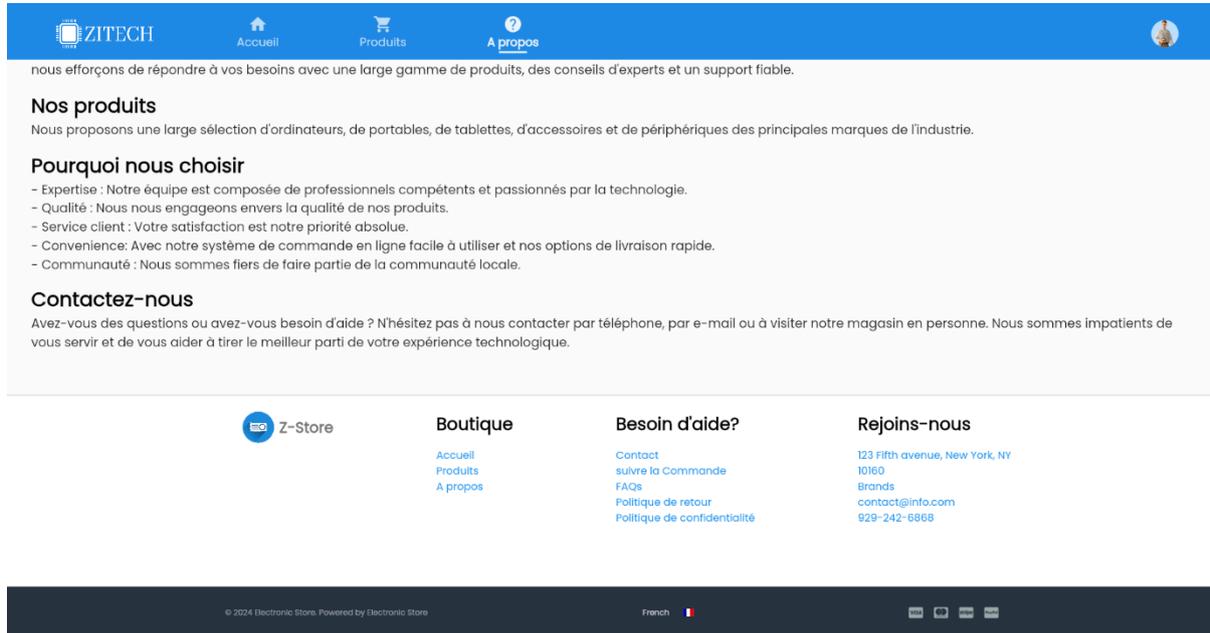


Figure 37 : Version française

Ci-joint la version anglaise de l’application pour la même page ‘About’:

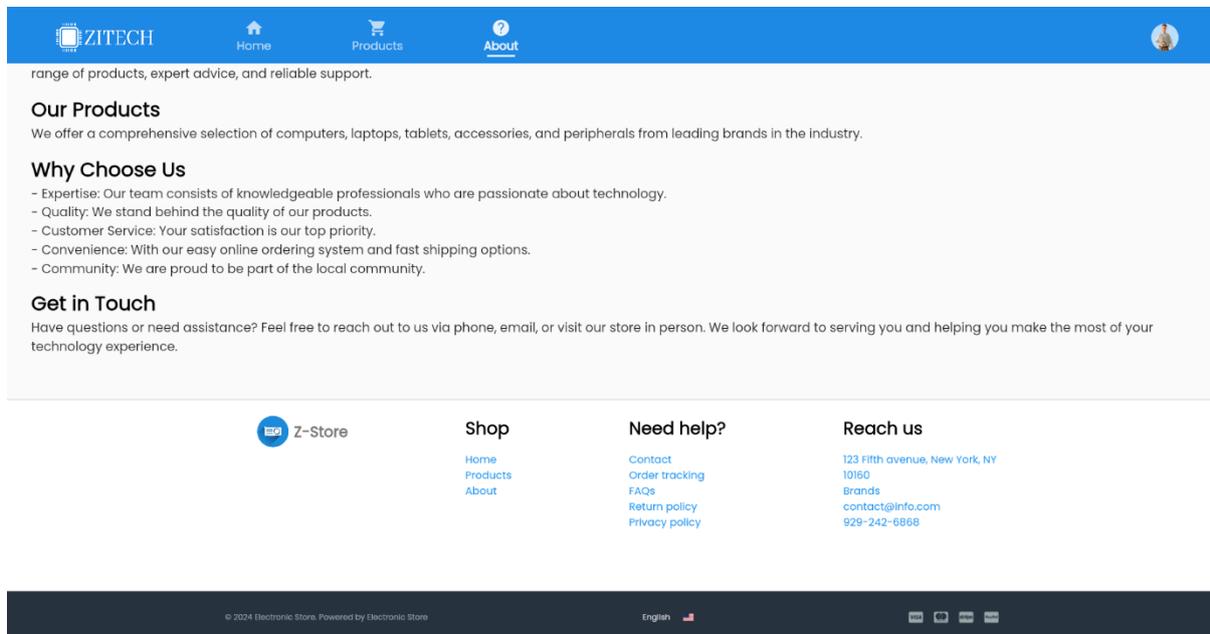


Figure 38 : Version anglaise

Cette fonctionnalité d'internalisation rend l'application responsive suivant les préférences linguistiques de l'utilisateur ce qui projette l'application dans un contexte de développement SaaS encore une fois.

3. Sprint 2 : Développement module profile utilisateur et Dashboard admin

4. Sprint 3 : Développement module store des produits

a) Planification

Le module développé dans ce sprint est la boutique des produits. Ce module est principalement destiné aux vendeurs pour gérer leurs produits, aux administrateurs pour superviser les activités des vendeurs, et aux clients pour consulter les détails des produits sélectionnés.

▼ Sprint3: Développement modules store des produits	4/24/24	5/3/24
Analyse et conception	4/24/24	4/24/24
Développement interface Store	4/24/24	4/29/24
Développement fonctionnalités gestion store	5/1/24	5/3/24

Figure 43 : Partition sprint 3

Voici la planification sprint 3

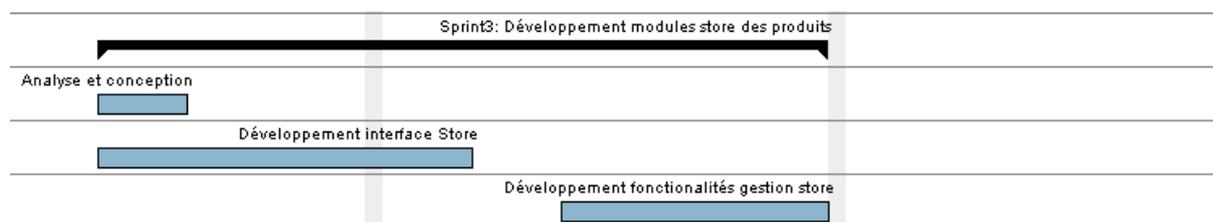


Figure 44 :Planification sprint 3

b) Conception

Le module de la boutique des produits contient les éléments suivants :

1. Interface Vendeur

Gestion des Produits :

- Ajouter un Produit : Formulaire pour ajouter de nouveaux produits.
- Modifier un Produit : Mettre à jour les informations des produits existants.
- Supprimer un Produit : Option pour retirer des produits de la liste.

2. Interface Administrateur

Supervision des Vendeurs :

- Liste des vendeurs avec leurs informations détaillées.
- Capacité de suspendre ou de bannir des vendeurs.

- Suivi des produits ajoutés par chaque vendeur.
- Intervenir dans la gestion des produits.

3. Interface Client

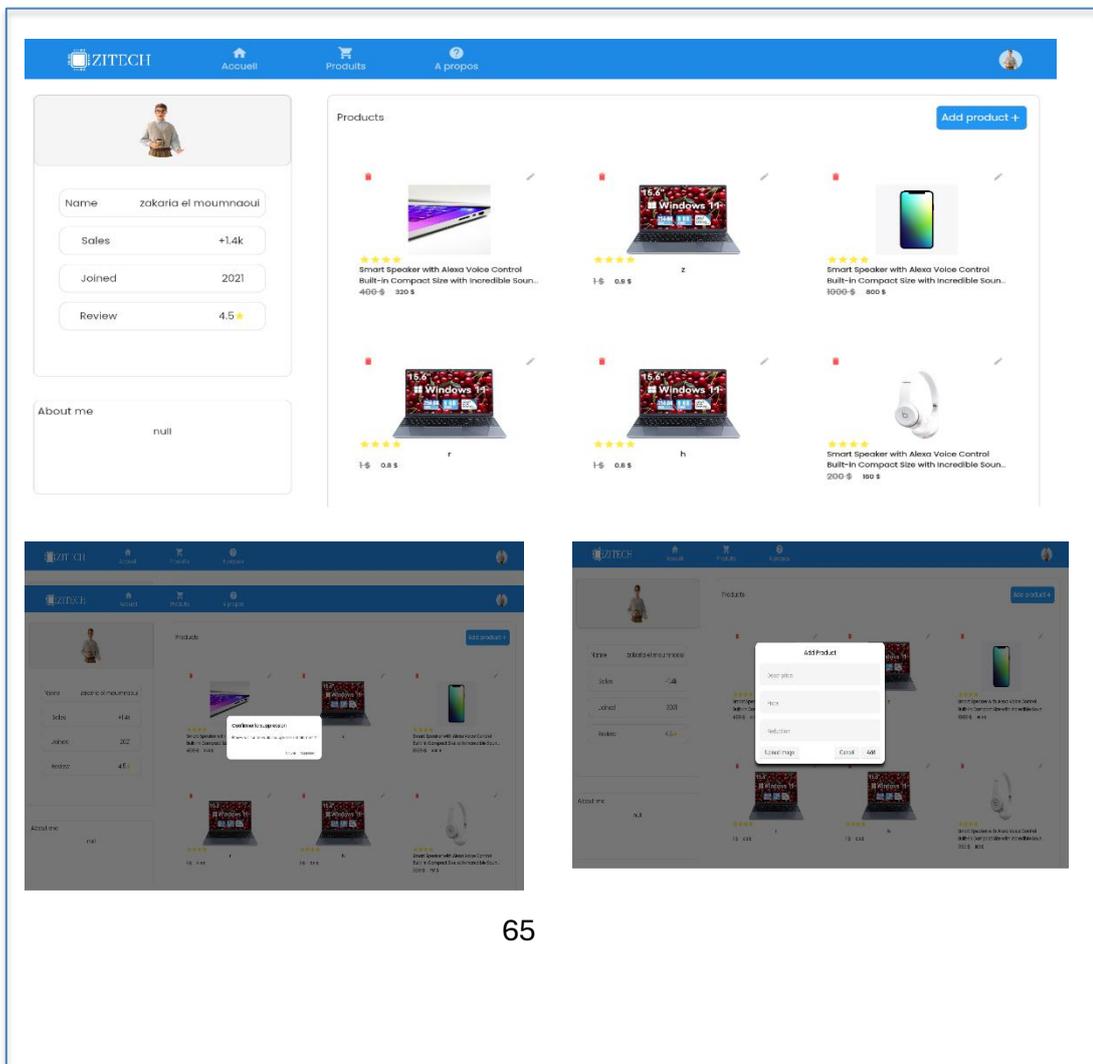
Consultation des Produits :

- Affichage des détails du produit (nom, description, prix, images, etc.).
- Affichage de la Boutique : Afficher la boutique associée à chaque produit.

Cette conception se concentre sur la gestion des produits par les vendeurs, la supervision par les administrateurs, et l'affichage des boutiques parmi les détails des produits pour les clients.

c) Réalisation

Voici la réalisation du module store



Les interfaces suivantes montrent les différents éléments une boutique d'un vendeur et les différentes actions peuvent être effectuées, ces points sont les suivants :

- Listage des produits : Le vendeur peut voir les différents produits disponibles au niveau du boutique.
- Ajout d'un produit : Le vendeur peut ajouter un produit avec les éléments qui définissent ce dernier tels que ; description, prix, image, réduction.
- Suppression d'un produit : Le vendeur peut supprimer un produit
- Modification d'un produit : Le vendeur peut modifier les attributs d'un produit déjà existant.

En plus le vendeur peut consulter des indicateurs de performance concernant sa boutique ou bien un produit spécifique.

Ces éléments rendent la plateforme responsive aux besoins d'un vendeur.

5. Sprint 4 : Développement catalogue produit et module panier

a) Planification

Les éléments développés dans ce sprint consistent d'un catalogue produit publié par différents vendeurs, un panier, formulaire de paiement.

▼	Sprint4: Développement catalogue produit et panier	5/6/24	5/23/24
	Analyse et conception	5/6/24	5/6/24
	Développement interface catalogue	5/6/24	5/10/24
	Développement fonctionnalités gestion catalogue	5/13/24	5/16/24
	Développement détails produit et panier	5/17/24	5/23/24

Figure 46 : Partition sprint 4

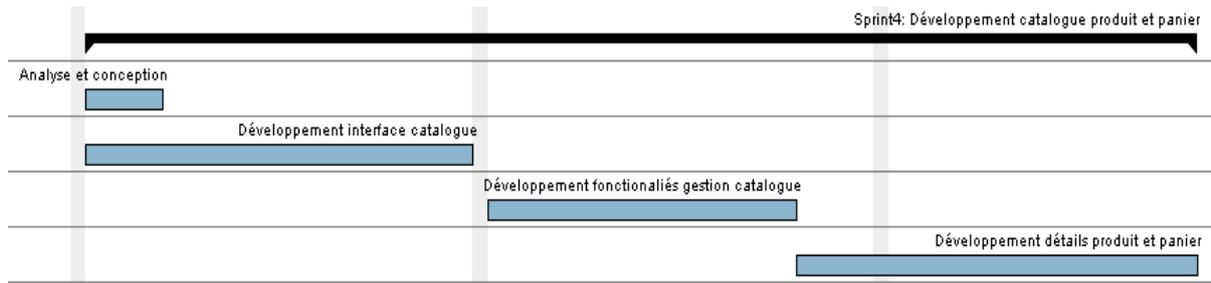


Figure 47 : Planification sprint 4

b) Conception

Au niveau de ce sprint, la plateforme offerte des points essentiels à propos les produits et l'interaction client avec la plateforme au niveau d'achat ; ce sprint couvre l'étape de développement du :

- Catalogue produit : ce catalogue contient les différents produits des différents vendeurs du la plateforme, en plus des sections dédiées à des offerts spéciaux.
- Détails produit : cette partie contient les détails essentiels d'un produit ainsi consulter par un client de plus d'une partie des avis à propos le produit par d'autres clients.
- Produits similaires : cette section constitue des produits similaires au produit ainsi sélectionné
- Buttons d'interaction avec le panier : cet emplacement contient des buttons d'interaction avec le panier, ajout produit, réduire la quantité d'un produit ou bien consulter panier.
- Gestion panier : à ce niveau le client peut gérer les produits ajoutés au panier, il peut également voir les quantités demandées, les différents prix.

- Finaliser la commande : l'interface panier contient un bouton de finalisation de commande qui redirige vers une interface qui contient un formulaire de paiement. Ces éléments rendent l'application bien structurée et offrent des fonctionnalités pour répondre aux besoins client.

c) Réalisation

Voici les réalisations du catalogue produit :

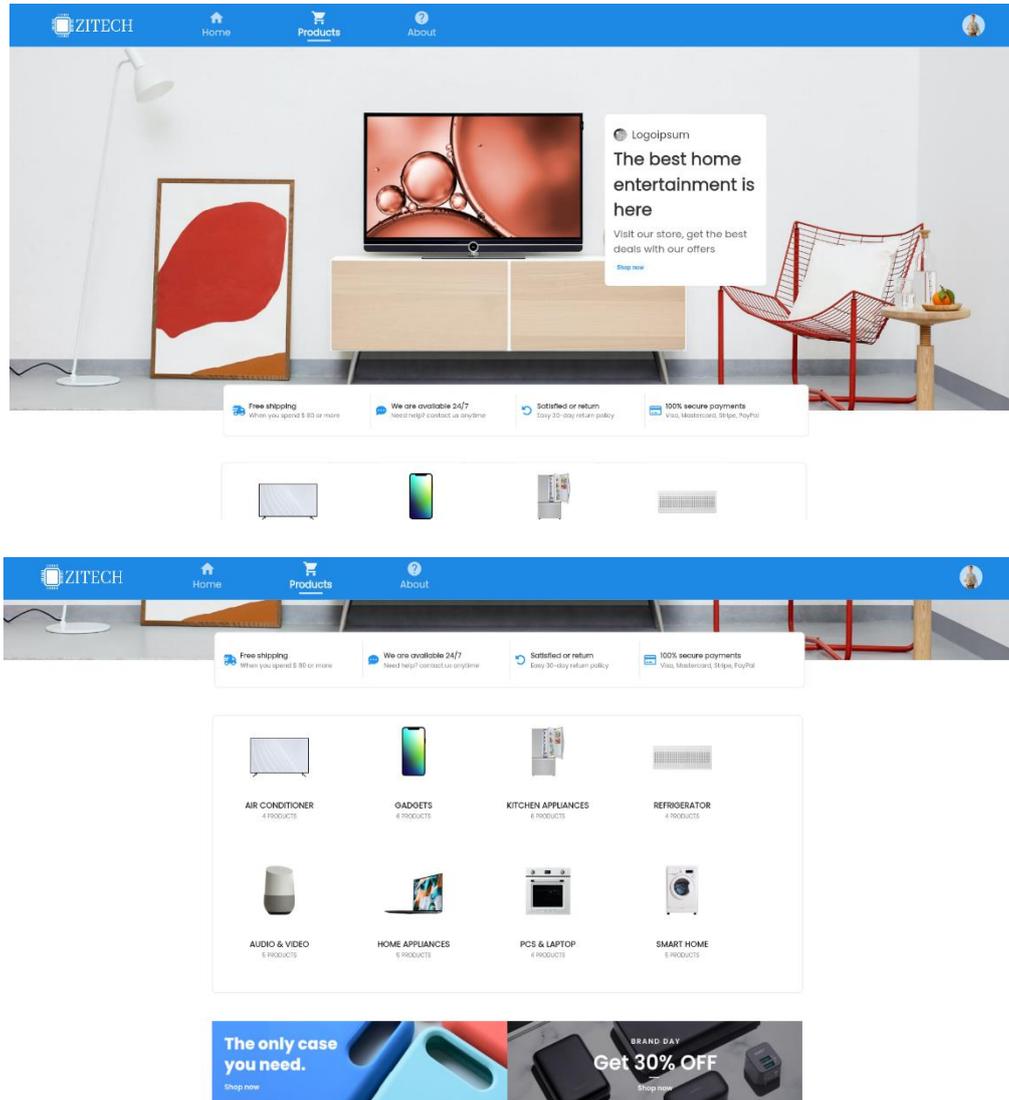


Figure 48 : Interfaces catalogue produit

Suite des interfaces :

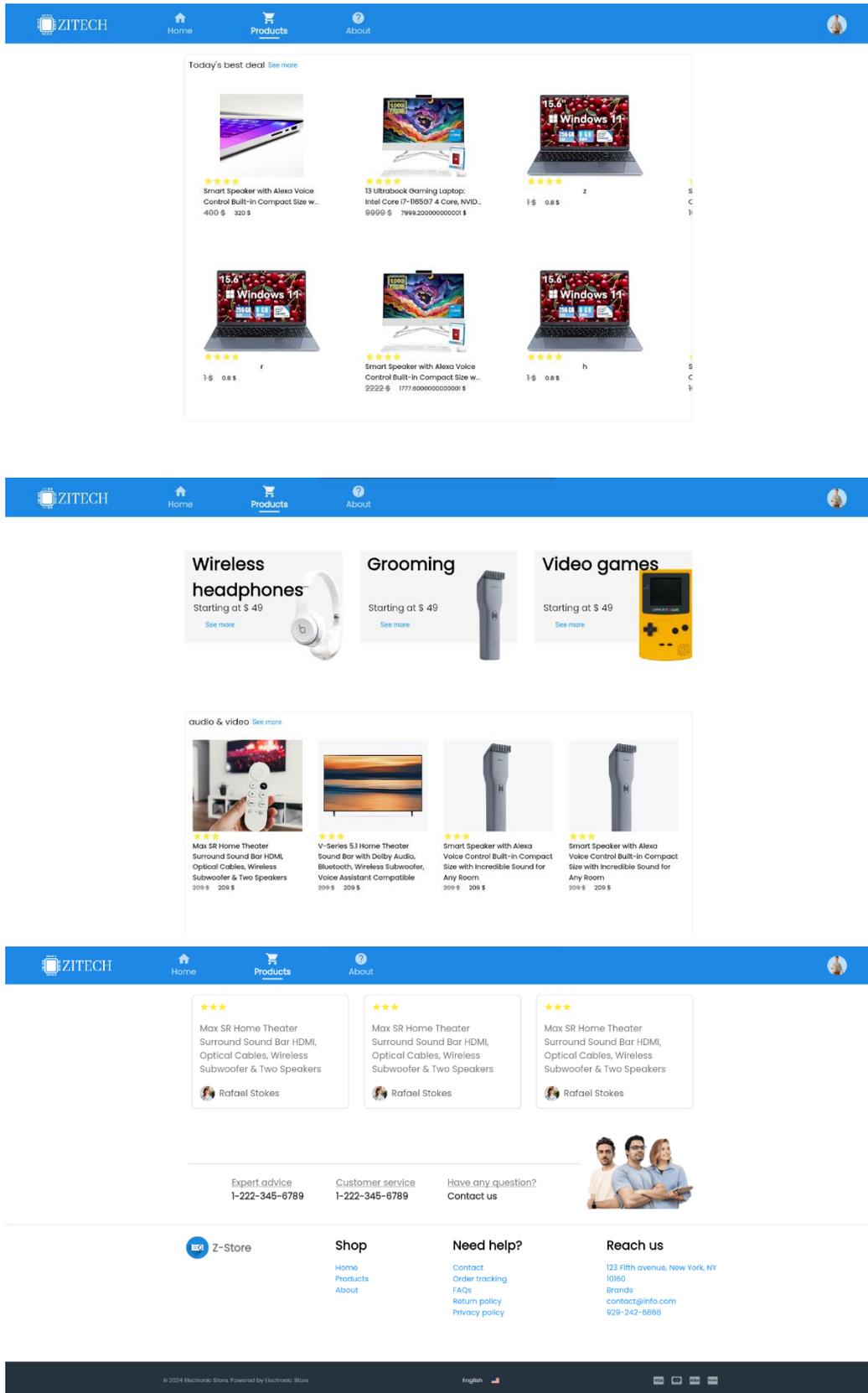
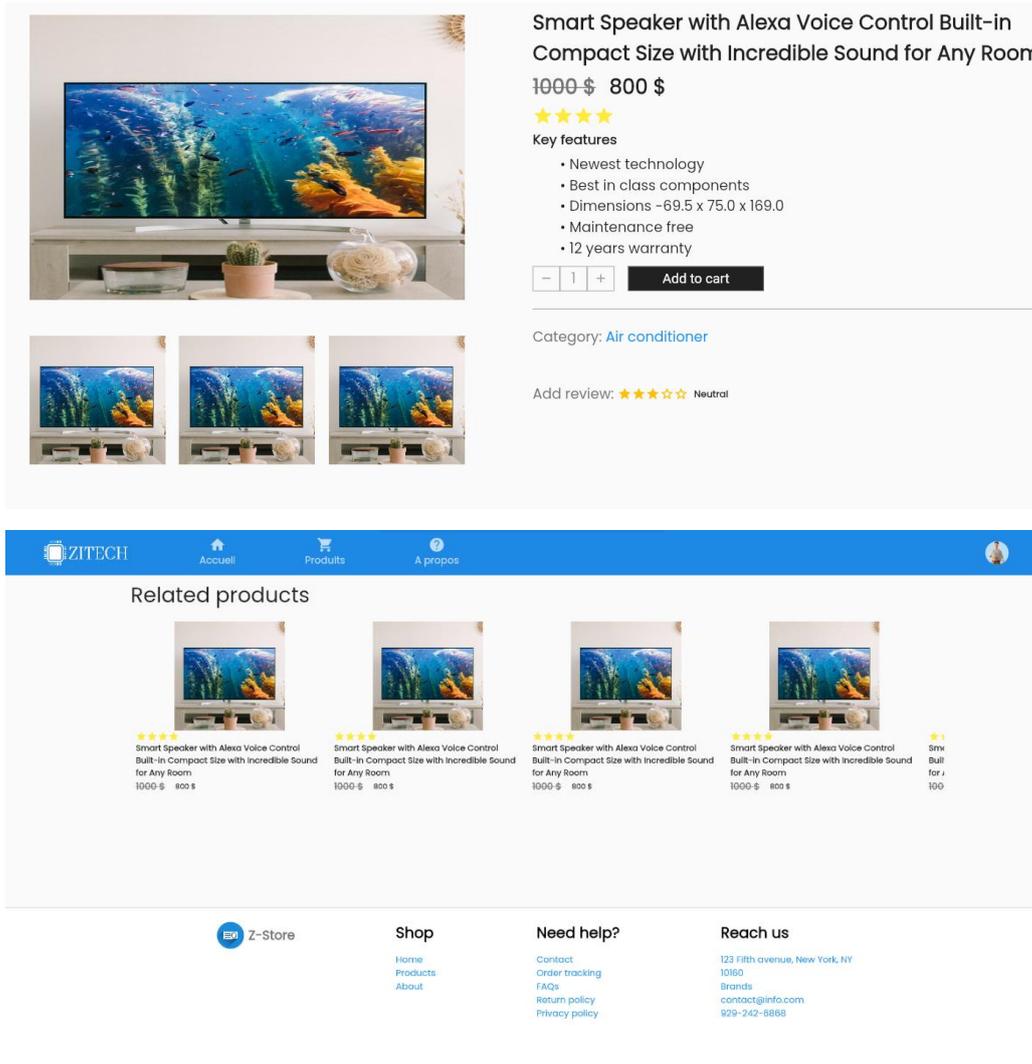


Figure 49 : Suite des interfaces catalogue produit

Les interfaces suivantes montrent les éléments suivants :

- **Catégories des produits** : les différentes catégories sont en relations avec les produits électronique.
- **Catalogue des produits** : les produits des différents vendeurs sont listés dans cette section, de plus ce catalogue contient des sections des offerts spéciaux.
- **Section des avis et support** : cette partie contient les avis des clients à propos les différents produits publiés, une partie contient les informations de support client.

Ces fonctionnalités offrent une expérience utilisateur fluide est optimisée concernant l'achat des produits.



Smart Speaker with Alexa Voice Control Built-in Compact Size with Incredible Sound for Any Room

1000\$ 800 \$

★★★★★

Key features

- Newest technology
- Best in class components
- Dimensions -69.5 x 75.0 x 169.0
- Maintenance free
- 12 years warranty

- 1 + **Add to cart**

Category: [Air conditioner](#)

Add review: ★★★★★ Neutral

Related products

[Smart Speaker with Alexa Voice Control Built-in Compact Size with Incredible Sound for Any Room](#)
 1000\$ 800 \$

[Smart Speaker with Alexa Voice Control Built-in Compact Size with Incredible Sound for Any Room](#)
 1000\$ 800 \$

[Smart Speaker with Alexa Voice Control Built-in Compact Size with Incredible Sound for Any Room](#)
 1000\$ 800 \$

[Smart Speaker with Alexa Voice Control Built-in Compact Size with Incredible Sound for Any Room](#)
 1000\$ 800 \$

[Smart Speaker with Alexa Voice Control Built-in Compact Size with Incredible Sound for Any Room](#)
 1000\$ 800 \$

Z-Store
 Home
 Products
 About

Shop
 Home
 Products
 About

Need help?
 Contact
 Order tracking
 FAQs
 Return policy
 Privacy policy

Reach us
 123 Fifth avenue, New York, NY
 10160
 Brands
 contact@info.com
 929-242-8888

Figure 50 : Interfaces détails produit

Voici la partie panier et formulaire de paiement :

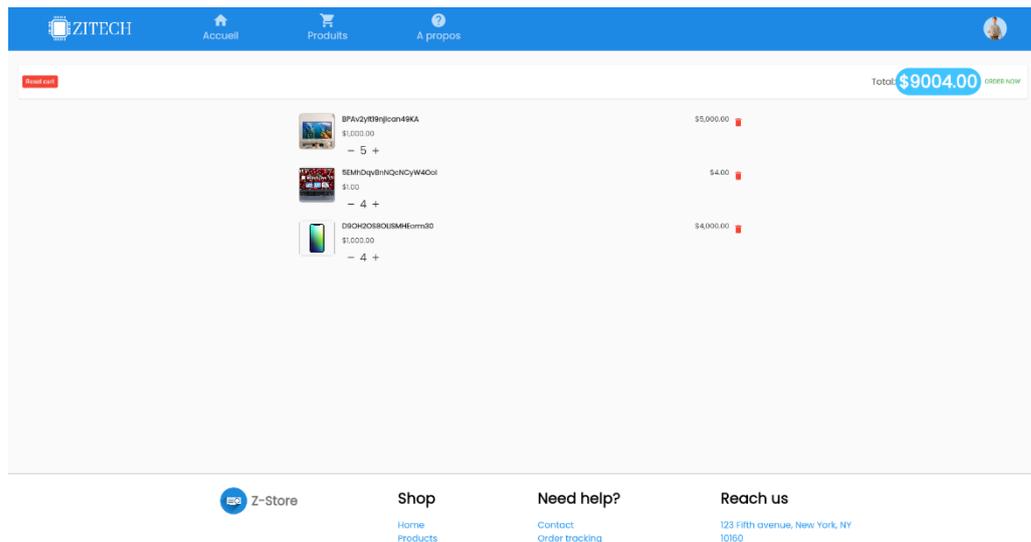


Figure 52 : Panier détails

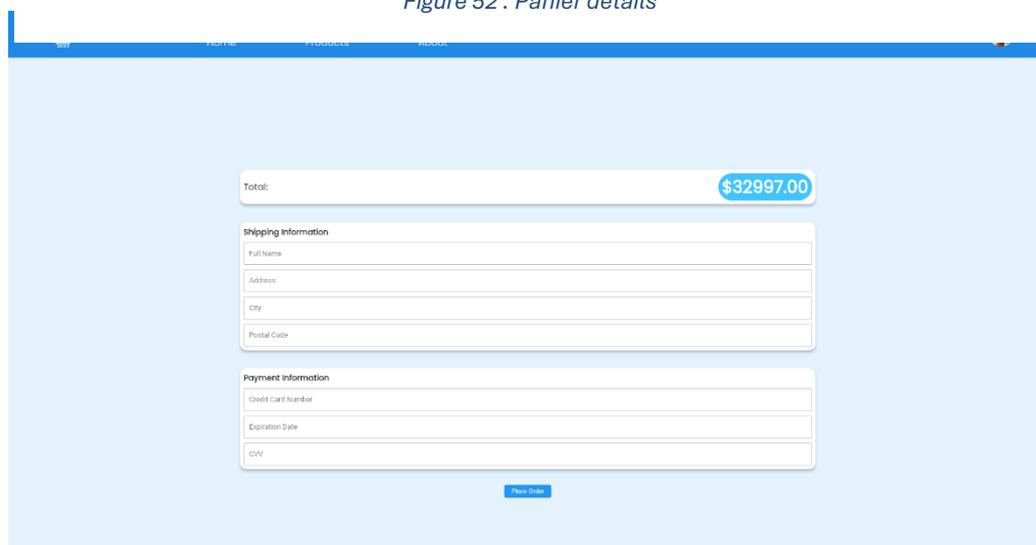


Figure 51 : Formulaire de paiement

Les interfaces au-dessus contiennent les éléments suivants :

- **Détails produits** : cette partie dédiée aux détails des produits tels que ; description, images, prix, avis, réduction, catégorie.
- **Interaction avec panier** : ajout, suppression des produits, accès au panier.
- **Section des produits similaires** : cette section contient les produits similaires avec le produit ainsi consulté.
- **Interface panier** : cet endroit contient les éléments du panier, l'information sur les produits sélectionnés, leurs quantités, leurs prix totaux, le total des prix, cette interface contient aussi des boutons d'interaction avec le panier tels que : vider panier, supprimer produit, réduire ou augmenter quantité et bouton de passer commande.
- **Interface de paiement** : cette interface contient un formulaire de paiement avec les informations sur le client et des informations de compte bancaire.

Ces fonctionnalités rendent l'application plus opérationnelle par le fait de finaliser la commande client.

6. Conclusion

La mise en œuvre de notre système ERP SaaS s'est déroulée en plusieurs sprints, chacun axé sur le développement de modules spécifiques tels que l'authentification, le profil utilisateur, le tableau de bord administratif, le magasin de produits et le catalogue produit. Chaque sprint a suivi une méthodologie rigoureuse incluant la planification, la conception et la réalisation. Grâce à cette approche, nous avons pu créer un système modulaire et scalable répondant aux besoins fonctionnels et non fonctionnels identifiés. Cette phase de mise en œuvre a non seulement validé nos choix techniques mais a également démontré la viabilité de notre architecture pour des déploiements futurs.

Conclusion générale et perspective

Ce rapport retrace l'ensemble du processus de développement d'un système ERP SaaS, depuis l'analyse initiale jusqu'à la mise en œuvre finale. L'approche a commencé par une étude approfondie du contexte général, ce qui a permis de comprendre les besoins et les problèmes du projet en détail. Cette étape a été la base pour définir les exigences fonctionnelles et non fonctionnelles qui guideront la conception et le développement.

L'analyse technique et la conception sont au cœur du travail. Ce travail se projette dans une architecture modulaire, basée sur des technologies modernes comme Flutter pour le développement multi-plateforme et Firebase pour la gestion de données et les services backend. De cette façon, l'architecture est conçue pour être évolutive, flexible et sécurisée, répondant aux besoins uniques du projet et donc prévue pour répondre à ses besoins futurs.

La mise en œuvre a été organisée en plusieurs sprints pour développer les modules prévus. Il s'agit d'une approche agile pour fournir progressivement et itérativement des fonctionnalités, en validant continuellement les choix techniques et en s'adaptant rapidement aux besoins changeants. Chaque sprint a été suivi d'un processus de planification, de conception et de réalisation afin d'assurer une qualité homogène et une intégration harmonieuse des différents modules.

Plusieurs facteurs ont joué un rôle clé dans la réussite de ce projet :

Choix de technologies avancées : Flutter et Firebase ont joué un rôle essentiel dans l'adoption d'un développement rapide, une gestion légère des données et une intégration native du service backend.

Méthodologie agile : Les sprints de développement activent une approche itérative pour faciliter des ajustements rapides en fonction du feedback de l'utilisateur et des nouvelles exigences.

Collaboration et communication : Une étroite collaboration entre les équipes de développement, de conception et de gestion de projet a été essentielle pour assurer la cohérence et l'efficacité pendant le développement.

L'architecture a été très bien conçue dans un souci de sécurité et d'évolutivité, Des règles de sécurité strictes ont été mises en place pour protéger les données et faciliter la croissance évolutive du système.

Les étapes suivantes du projet se décrivent comme suit :

- ✓ l'intégration de l'IA et de l'apprentissage machine va rendre le système capable de faire des analyses prédictives, de formuler des recommandations automatisées, et améliorer encore la prise de décision dans le système.
- ✓ Développement de nouveaux modules pour répondre à des besoins supplémentaires tels que la gestion avancée des ressources humaines, la comptabilité intégrée et la gestion de la chaîne d'approvisionnement.
- ✓ Optimisation des performances du système et affinement des requêtes de base de données tout en minimisant les temps de latence pour une interface utilisateur réactive.

Ce projet a permis de mettre les pieds sur le terrain pour la construction d'un système ERP SaaS innovant et performant, et a ouvert la voie à de nombreuses futures améliorations. Alliant une architecture solide, une méthodologie agile et une vision orientée vers l'innovation, le travail est bien placé pour continuer à évoluer et relever les nouveaux défis posés par les utilisateurs.

Webographie :

<https://www.javatpoint.com/dart-programming>

<https://docs.flutter.dev/resources/architectural-overview>

<https://www.ibm.com/think/insights/software-as-a-service-advantages#:~:text=SaaS%20can%20provide%20beneficial%20cost,compared%20with%20the%20traditional%20model.>

<https://www.ibm.com/docs/en/rsm/7.5.0?topic=models-managing-uml>

<https://aws.amazon.com/what-is/flutter/>

<https://github.com/dart-lang>

<https://firebase.google.com/products-build>

Bibliographie

BEGINNING Flutter® A HANDS ON GUIDE TO APP DEVELOPMENT

by Marco L. Napoli

APPRENEZ firebase-database

by contributeurs de Stackoverflow