



Institut supérieur du Génie Appliqué

Mémoire de Projet de Fin d'Etudes

Pour l'obtention du

Diplôme d'Ingénieur d'Etat

Spécialité

Développement Logiciel et Technologies de l'Information
(DLTI)

Réalisé par :

BENHMIDI El Mehdi

Sous le thème :

Conception et Développement d'une Application Web et Mobile pour la Digitalisation des Tickets

Encadrant :

M. JALAL Abdellah

(Encadrant de stage)

M. NADIR Youness

(Professeur d'Enseignement Supérieur)

Devant les jurys composés de :

M. MOHAMMADI Zakaria

M. AISSAOUI Khalid

M. RIYAMI Bouchaib

M. SAIDI Abdelali

Dédicace

A mes très chers Parents

Aucun mot ne pourrait exprimer pleinement l'amour immense que je ressens pour vous, ni la profonde gratitude que je vous porte pour tous les efforts et sacrifices que vous avez toujours faits pour mon éducation et mon bonheur.

C'est à travers vos encouragements que j'ai opté pour cet itinéraire, et c'est à travers vos conseils que je me suis réalisé. J'espère avoir été à la hauteur des attentes que vous avez placées en moi.

Je vous témoigne toute ma reconnaissance et mon amour infini à travers ce modeste travail.
Que Dieu tout-puissant vous accorde santé, bonheur et longue vie afin que vous restiez le phare illuminant le chemin de vos enfants.

A mon encadrant académique NADIR Youness.

A mes professeurs, à mes protecteurs, à mes confidents.

A mes Collègues de 5 DLTI sans exception.

A ceux que je souhaiterais toujours voir fiers de moi.

Je dédie ce travail

El Mehdi BENHMIDI

Remerciement

Avant tout, j'aimerais rendre grâce à **Dieu** tout puissant pour ses bienfaits trop souvent négligés.

J'adresse mes remerciements à **M. Abdellah JALAL**, directeur général de CORAIL Maroc, qui m'a permis d'effectuer ce stage de fin d'études au sein de l'organisme.

Mes remerciements sincères vont également à tout le personnel de CORAIL, pour leur accueil chaleureux, leur bonne humeur quotidienne, leur capacité d'équipe exemplaire ainsi que leur présence quand j'avais besoin de leur aide.

Un grand merci à **M. Abdellah JALAL** notre responsable, pour ses efforts fournis.

Un grand merci à **M. Youness NADIR** mon encadrant académique durant la période de stage, pour ses efforts fournis.

Que les **membres du jury** trouvent ici l'expression de ma profonde reconnaissance pour avoir accepté de juger mon travail.

Bien évidemment, je remercie **mes amis et ma famille** qui tout au long de ce stage mais aussi tout au long de mes études ont su me soutenir et m'accompagner dans tous ces moments de doute mais aussi de joie.

Merci à tous !

Résumé

Ce rapport présente le travail effectué lors de mon stage de fin d'études au sein de l'entreprise **CORAIL Maroc**, dont le sujet principal était « La Digitalisation des tickets ».

L'objectif principal de ce projet était de remplacer les tickets au format papier par des tickets numériques, afin de simplifier les procédures d'achat, d'analyse et de consultation des tickets sans risque de perte.

Ce projet visait à améliorer l'efficacité du traitement des tickets et à réduire le temps de réponse aux clients. La méthodologie adoptée inclut des analyses approfondies des besoins des utilisateurs, ainsi que la conception d'une architecture logicielle robuste.

Le projet se compose de trois parties :

- **Client** : Une application mobile qui facilite le scan des tickets, la consultation des tickets et la visualisation des statistiques.
- **Administrateur de la marque** : Une application web permettant de consulter les statistiques des magasins dédiés, d'ajouter les comptes des magasins et de les gérer.
- **Magasins** : Une application pour afficher le QR Code aux clients.

En conclusion, cette solution de digitalisation des tickets répond efficacement aux besoins de l'entreprise Corail Maroc et propose une solution scalable et adaptable au marché.

Abstract

This report presents the work carried out during my final year internship at **CORAIL Maroc**, with the main topic being "The Digitalization of Tickets". The primary objective of this project was to replace paper tickets with digital tickets to simplify the procedures for purchasing, analyzing, and consulting tickets without the risk of losing them.

This project aimed to improve the efficiency of ticket processing and reduce the response time to customers. The adopted methodology included in-depth analyses of user needs and the design of a robust software architecture.

The project is divided into three parts :

- **Client** : A mobile application facilitating ticket scanning, ticket consultation, and viewing statistics.
- **Brand Administrator**: A web application for viewing dedicated store statistics, adding store accounts, and managing them.
- **Stores** : An application for displaying QR codes to customers.

In conclusion, this ticket digitalization solution effectively meets the needs of Corail Maroc and offers a scalable and adaptable solution to the market.

Liste des tableaux

Tableau 1 : Fiche Technique de Corail Maroc.....	5
Tableau 2 : Etude comparative entre KANBAN & SCRUM	9
Tableau 3 : Description des phases de la planification	15
Tableau 4 : Gestion des risques	16
Tableau 5 : Analyse SWOT de la digitalisation des tickets.....	17
Tableau 6 : Les Exigences fonctionnelles pour l'application mobile (Client).....	16
Tableau 7 : Les Exigences fonctionnelles pour l'application web (Administrateur marque).....	17
Tableau 8 : Les Exigences fonctionnelles pour l'application web (magasin)	17
Tableau 9 : Les Exigences non fonctionnelle	17
Tableau 10 : Description des tables du diagramme de classe.....	19
Tableau 11 : Les scénarios de l'inscription de l'utilisateur	21
Tableau 12 : Les scénarios du scan de ticket	26
Tableau 13 : Les scénarios de l'authentification – application web magasin	26
Tableau 14: Les scénarios de l'authentification JWT	28
Tableau 15: Les scénarios de la sécurité entre les services.....	28
Tableau 16: Les APIs du service d'authentification	29
Tableau 17: Les APIs du service core	30
Tableau 18: Comparaison entre Flutter et React Native	33
Tableau 19: Comparaison entre Machine Virtuelle et Image Docker	39

Liste des figures

Figure 1: CORAIL	3
Figure 2: Les unités géographiques de Corail.....	4
Figure 3 : Emplacement Corail Maroc - Casablanca	4
Figure 4 : Organigramme de Corail Maroc.....	5
Figure 5 : Exemple de tableau KANBAN	8
Figure 6 : Tableau KANBAN dans JIRA	10
Figure 7 : Les Tâches de la Période du PFE	10
Figure 8 : Diagramme de GANTT	11
Figure 9 : Diagramme de classe.....	18
Figure 10 : Diagramme de cas d'utilisation – Utilisateur application mobile	19
Figure 11 : Diagramme de cas d'utilisation – Administrateur marque (application web).....	20
Figure 12: Diagramme de cas d'utilisation –Commercial (application web du magasin).....	20
Figure 13 : Diagramme d'activité – L'inscription de l'utilisateur (application mobile).....	21
Figure 14 : Diagramme d'activité – Scan ticket (application mobile).....	22
Figure 15 : Diagramme d'activité – Authentification (application web magasin)	23
Figure 16 : Diagramme de séquence – Authentification JWT (Sécurité)	24
Figure 17 : Diagramme de séquence – Vérification du token entre les services (Sécurité).....	25
Figure 18 : Flutter	28
Figure 19: Angular.....	29
Figure 20 : Spring Boot.....	29
Figure 21 : Spring Security	29
Figure 22 : Spring Cloud	29
Figure 23 : Minio	29
Figure 24 : MySQL.....	30
Figure 25 : IntelliJ IDEA	30
Figure 26 : VS Code	30
Figure 27 : WampServer.....	30
Figure 28 : Github.....	30
Figure 29 : Docker	31
Figure 30 : VPS Azure.....	31
Figure 31 : Postman	31
Figure 32 : Swagger.....	31
Figure 33 : Slack	32
Figure 34 : Jira	32
Figure 35 : SMTP	32
Figure 36 : Qodana	32
Figure 37 : Comparaison entre les langages de programmation (Frontend).....	33
Figure 38 : Comparaison entre les langages de programmation (Frontend).....	34
Figure 39 : Comparaison entre les Frameworks	34
Figure 40 : Comparaison entre les Bases de données	36
Figure 41 : Architecture Microservices du projet.....	37
Figure 42 : la structure du projet – partie Backend.....	40
Figure 43 : VPS – Déploiement.....	40
Figure 44 : Documentation des APIs Servcie Auth– Swagger	41
Figure 45 : Documentation des APIs Servcie Core– Swagger	41
Figure 46 : Test de l'API de génération de QR Code– Service CORE.....	41
Figure 47 : Fichier JSON du ticket	42
Figure 48 : L'authentification – Application mobile	42
Figure 49 : Les informations de l'utilisateur connecté– Application mobile et web.....	42
Figure 50 : Les informations de l'utilisateur connecté– Application mobile et web	43
Figure 51 : Les informations des tickets– Application mobile et web	43
Figure 52 : La déconnexion – Application mobile et web	43

Figure 53 : La déconnexion – Application mobile et web	44
Figure 54 : Interfaces d'accès et de présentations	44
Figure 55 : Interfaces d'inscription.....	45
Figure 56 : Message d'erreur + Code de vérification.....	45
Figure 57 : Interface d'authentification.....	46
Figure 58 : Page d'accueil.....	46
Figure 59 : Interface des notifications.....	47
Figure 60 : Interface historique + filtrage.....	47
Figure 61 : Interface des catégories	48
Figure 62 : Interface des statistiques.....	48
Figure 63 : Interfaces de scan.....	49
Figure 64 : Interfaces du profile	49
Figure 65 : Interfaces Où utiliser Corail.....	50
Figure 66 : Interface parrainer un ami.....	50
Figure 67: Interface paramètre	51
Figure 68 : Interface support	51
Figure 69 : Tableau de bord – Administrateur marque	51
Figure 70 : Message d'erreur	52
Figure 71 : L'authentification– application web magasin	52
Figure 72 : Page de QR Code– application web magasin	52
Figure 73: Test de qualité de code avec Qodana.....	52

Table des matières

1. Introduction Générale	1
Chapitre 1 : Cadrage du projet et Contexte général	2
1.1. Présentation de l'organisme d'accueil	3
1.1.1 Présentation de CORAIL	3
1.1.2 Organigramme	5
1.2. Contexte du projet	6
1.2.1 Problématique	6
1.2.2 Solution proposée	6
1.2.3 Portée du projet	7
1.3. Conduite et gestion de projet	8
1.3.1 Méthodologie suivie : méthode KANBAN	8
1.3.2 Diagramme de GANTT	10
1.3.3 Analyse SWOT.....	12
Conclusion	13
Chapitre 2 : Etude conceptuelle du projet	14
2.1. Etude fonctionnelle	15
2.2.1 Critique de l'existant	15
2.2.2 Les besoins fonctionnels	15
2.2.3 Les besoins non fonctionnels	17
2.2. Conception	18
Diagramme de classe	18
Description des tables du diagramme de classe	18
Diagramme de cas d'utilisation	19
Diagramme d'activité	20

Diagramme de séquence	23
Conception de l'API	25
Conclusion	26
Chapitre 3 : Outils, Choix des technologies et architecture	27
3.1. Choix techniques et Architecture	28
2.3.1 Besoins techniques	28
2.3.2 Choix technologiques	28
Les langages de programmation	28
Outil de stockage.....	29
Service de base de données.....	30
Outils de développement.....	30
Outil de gestion de version :	30
Outil de conteneurisation :	31
Outil de déploiement :	31
Outil de test d'API :	31
Outil de documentation d'API REST :.....	31
Outil de communication et collaboration :	32
Outil de messagerie :	32
Outil d'analyse statique du code source.....	32
2.3.3 Raison de choix des technologies et des outils.....	32
2.3.4 Architecture :	36
Conclusion	38
Chapitre 4 : Réalisation et implémentation du projet	39
4.1. Aperçu de structure et des interfaces développées	40
4.1.1 La structure du partie Backend	40
4.1.2 Capture d'écran du VPS	40

4.1.2 Capture d'écran de documentation Swagger.....	41
4.1.4 Capture d'écran des Tests APIs Postman	41
4.1.5 Capture d'écran de l'outil de stockage des images.....	44
4.1.6 Les interfaces réalisées pour l'application mobile.....	44
4.1.7 Les interfaces réalisées pour l'application web	51
4.1.8 Capture d'écran du test de qualité de code.....	52
Conclusion	54
Conclusion générale et perspective	55
Bibliographie	56

Liste des abréviations

Abréviation	Signification
POS	Point Of Sale
CNDP	Commission Nationale de contrôle de la protection des Données à caractère Personnel
SWOT	Strengths Weaknesses Opportunities Threats
UML	Unified Modeling Language
UX/UI	User Experience / User Interface
JSON	JavaScript Object Notation
JWT	Json Web Token
API	Application Programming Interface
REST	REpresentational State Transfer
IDE	Integrated Development Environment
VPS	Virtual Private Server
SMTP	Simple Mail Transfer Protocol

Introduction Générale

Dans un monde de plus en plus numérique, les entreprises explorent constamment les moyens de passer au numérique. Améliorer leurs processus internes et externes pour gagner en efficacité et en réactivité. Corail Maroc, une entreprise leader dans son secteur, a identifié un besoin crucial d'optimiser la gestion des tickets de service client. Les tickets au format papier présentent plusieurs inconvénients, notamment la perte possible de documents, la difficulté de suivi, et l'inefficacité dans l'analyse des données.

Pour répondre à ces défis, CORAIL Maroc a décidé de se lancer dans un projet ambitieux : la digitalisation des tickets. Ce projet vise à remplacer les tickets en format papier par des tickets numériques, offrant ainsi une solution moderne et efficace pour les procédures d'achat, d'analyse et de consultation des tickets. Les objectifs principaux sont de simplifier ces procédures, de réduire le temps de réponse aux clients et d'améliorer globalement l'efficacité du traitement des tickets.

La méthodologie adoptée pour ce projet comprend une analyse approfondie des besoins des utilisateurs, la conception d'une architecture logicielle robuste et le développement de plusieurs applications adaptées aux différents types d'utilisateurs : clients, administrateurs et magasins. Le projet se décompose en trois parties principales, qui seront détaillées dans ce rapport.

Ce rapport détaillera les étapes de développement de ce projet, les résultats obtenus, ainsi que les bénéfices apportés par cette solution de digitalisation. En conclusion, nous évaluerons l'impact de cette solution sur l'entreprise Corail Maroc et son potentiel d'adaptation à d'autres entreprises ayant des besoins similaires.

Chapitre 1

Cadrage du projet et Contexte général

Ce chapitre aborde, dans sa première section, la présentation de l'établissement qui nous a accueillis durant notre période de stage ainsi que la direction d'accueil. Ensuite, nous mettons l'accent sur le contexte général ainsi que les objectifs et la problématique du projet réaliser, avant de préciser sa conduite en spécifiant la démarche, en prévoyant un planning de travail et en déterminant les livrables du projet.

1. Cadrage du projet et Contexte général

Le stage s'est déroulé au sein de CORAIL Maroc, basée à Casablanca et a pour objectif de réaliser une application web et l'autre mobile pour la digitalisation des tickets. Ce chapitre sera consacré à la présentation de l'entreprise d'accueil CORAIL, ensuite la description du périmètre de projet afin de définir la problématique et les objectifs de ce dernier et enfin exposer la planification et la conduite du projet.

1.1 Présentation de l'organisme d'accueil

Cette partie comprend une brève présentation de la société d'accueil : son secteur d'activité, les différents départements ou services, ses références, son historique et enfin son organigramme.

1.1.1 Présentation de CORAIL

CORAIL Maroc est une entreprise spécialisée dans la conception de solutions innovantes pour améliorer les parcours clients. Aider les marques à se développer, mettant à profit une compréhension approfondie des besoins et des comportements des consommateurs. Exploiter des milliards de points de données pour créer des expériences client mémorables et impactantes.



Figure 1: CORAIL

Corail Maroc est une entreprise spécialisée dans la création de solutions innovantes destinées à améliorer les parcours client. Elle se concentre sur l'utilisation des technologies modernes et de l'intelligence artificielle pour offrir des expériences client intégrées et optimisées. Ses services incluent la numérisation des tickets de caisse, facilitant ainsi les retours et les échanges de produits pour les consommateurs. Corail Maroc aide les marques à se développer en exploitant des milliards de points de données pour concevoir des expériences qui répondent aux attentes des clients et influencent positivement les marchés. La société travaille avec des marques de toutes tailles, des petites entreprises locales aux grandes multinationales, en offrant un support multicanal et une présence mondiale.

Corail compose de deux autres solutions :

Corail Business : Corail Business est une plateforme digitale de pilotage et d'animation des opérations dans le retail qui vous aide à reprendre un total contrôle sur vos activités quotidiennes sur terrain et en points de vente, vous offrant une communication et une collaboration bidirectionnelle entre points de vente et sièges sociaux, tout-en-un. Les réseaux de franchise de toutes tailles, les distributeurs et les marques ambitieuses utilisent Corail Business pour

connecter, engager et impacter leurs équipes terrain afin de les aider à délivrer une expérience client optimisée en magasin.

Corail Pay : une application de paiement mobile qui offre une expérience pratique et agréable tout au long du parcours de paiement, tant pour les consommateurs que pour les commerçants. L'application permet aux consommateurs d'effectuer des paiements numériques à l'aide d'un QR code, réduisant ainsi le besoin d'argent liquide, de cartes physiques et de dispositifs de point de vente (POS). CorailPay permet également aux consommateurs de relier leur carte en toute sécurité pour l'utiliser dans toutes leurs transactions, de surveiller leurs dépenses grâce à des rapports de dépenses visuels, d'organiser leurs reçus de paiement, d'activer un compte virtuel pour les enfants et les membres de la famille et d'accéder aux bons et coupons des commerçants.

L'emplacement géographique de Corail international :



Figure 2: Les unités géographiques de Corail

L'emplacement géographique de Corail Maroc :



Figure 3 : Emplacement Corail Maroc - Casablanca

Fiche Technique de Corail MAROC

Secteur	Service financiers
Siège social	Paris, Île -de-france
Spécialisations	Digital Retail Bank, Fintech et Digital Banking et smartphone
Site Internet	www.corail.co.ma
Adresse	Tour Jasmin, Casa finance city

Tableau 1 : Fiche Technique de Corail Maroc

1.1.2 Organigramme

L'organigramme de Corail Maroc a évolué depuis sa création en s'adaptant aux besoins du business pour un meilleur rendement et une meilleure évolution dans le temps.

Voici donc les organigrammes de Corail Maroc :

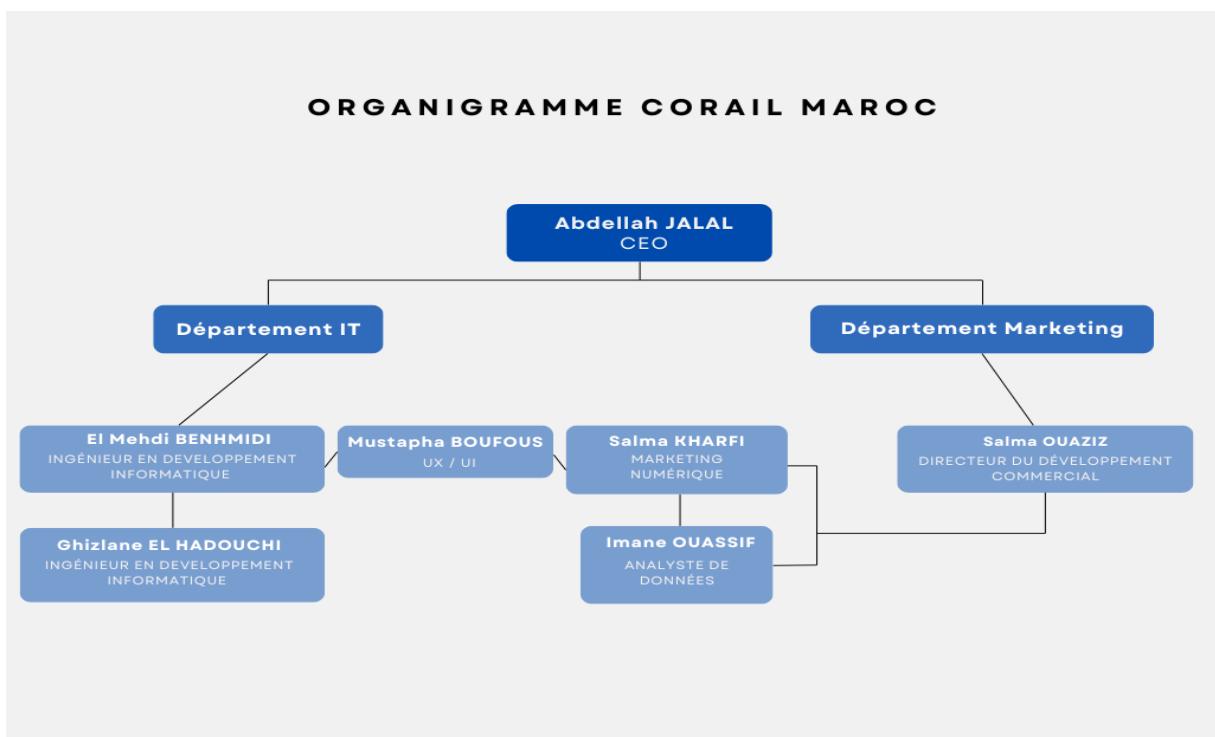


Figure 4 : Organigramme de Corail Maroc

1.2 Contexte du projet

1.2.1 Problématique

Le monde en 2024 est plus digitalisé que jamais, grâce à l'innovation des peuples et des avancées technologiques. Cependant, certains secteurs demeurent en retard dans cette transformation numérique, comme les grandes surfaces, les supermarchés et les marques. Ces entités continuent à utiliser des tickets de caisse au format papier. Bien que ces tickets permettent de réviser nos achats et de faciliter les échanges de produits, ils présentent plusieurs inconvénients majeurs.

Tout d'abord, la perte d'un ticket papier rend impossible le retour ou l'échange d'un produit, car ce ticket constitue une preuve d'achat indispensable. De plus, faire des statistiques de consommation personnelle à partir de tickets papier est laborieux : il faut rassembler tous les tickets, effectuer les calculs et classer les catégories manuellement. Le suivi manuel des tickets papier peut être laborieux et sujet à des erreurs humaines, ce qui impacte la réactivité et la satisfaction client. Il peut être difficile de retracer l'historique complet d'un ticket papier, y compris les interactions précédentes avec le client ou les actions prises.

En outre, beaucoup de gens jettent leurs tickets de caisse dans la rue immédiatement après leurs achats, ce qui pose un problème environnemental. Les tickets papier ne sont pas écologiques et contribuent à la pollution.

1.2.2 Solution proposée

Pour résoudre les problèmes liés à l'utilisation des tickets de caisse papier, une solution efficace consiste à adopter la digitalisation des tickets de caisse. Cette transformation présente de nombreux avantages, tant pour les clients que pour les entreprises et l'environnement.

1. Amélioration de l'expérience client :

- **Accessibilité et participé :** Les tickets de caisse numériques peuvent être envoyés directement via une application mobile. Les clients peuvent facilement retrouver et accéder à leurs tickets à tout moment, sans risque de les perdre ou de les endommager.
- **Gestion simplifiée des retours et garanties :** Avec les tickets numériques, il est plus facile pour les clients de gérer les retours et les garanties. Les preuves d'achat étant disponibles en ligne, le processus de retour ou d'échange devient plus fluide.
- **Centralisation des informations :** Les clients peuvent consulter l'historique de leurs achats en un seul endroit, ce qui simplifie la gestion de leurs finances personnelles et leur permet de suivre leurs dépenses plus efficacement.

2. Réduction de l'impact environnemental :

- **Diminution de l'utilisation du papier** : La suppression des tickets papier réduit la consommation de papier et les déchets associés. Cela contribue à la protection des forêts et à la réduction des déchets solides.
- **Moins de pollution** : Réduire la production et l'utilisation de papier diminue également les émissions de CO2 liées à la fabrication, au transport et à l'élimination des tickets papier.

3. Optimisation des coûts opérationnels :

- **Économies sur les fournitures** : La digitalisation permet aux entreprises de réduire considérablement les coûts liés à l'achat de papier et d'encre.
- **Efficacité opérationnelle** : Les employés n'ont plus besoin de gérer des tickets papier, ce qui leur permet de se concentrer sur des tâches à plus forte valeur ajoutée et d'améliorer leur productivité.

4. Sécurité et conformité :

- **Protection des données personnelles** : Les tickets numériques doivent être sécurisés et conformes aux réglementations en vigueur. Des mesures de sécurité appropriées peuvent garantir la protection des informations des clients.
- **Archivage sécurisé** : Les tickets de caisse numériques peuvent être archivés de manière sécurisée et facilement accessibles en cas de besoin pour des audits ou des contrôles fiscaux.

5. Analyse et utilisation des données :

- **Personnalisation de l'offre** : Les entreprises peuvent collecter des données précieuses sur les achats des clients à partir des tickets numériques. Ces données peuvent être utilisées pour offrir des promotions personnalisées et améliorer la fidélisation des clients.
- **Marketing** : Développer des fonctionnalités marketing dans l'application pour mieux comprendre les préférences des clients et adapter les offres en conséquence.

1.2.3 Portée du projet

- Développer une application mobile conviviale compatible avec les principaux systèmes d'exploitation (iOS et Android).
- Ajouter une interface utilisateur conviviale pour les administrateurs des marques sur l'application web, facilitant la collecte et l'analyse des données client.
- Intégrer une fonctionnalité de génération et d'envoi de tickets de caisse électroniques.

- Mettre en place un système de gestion des consentements clients pour la collecte des données personnelles.
- Assurer la conformité aux normes de sécurité des données et de protection de la vie privée (CNDP).
- Permettre aux administrateurs d'accéder à des rapports analytiques détaillés sur les comportements d'achat et les préférences des clients.

1.3 Conduite et gestion de projet

La conduite de projet est une démarche qui vise à structurer, assurer et optimiser le bon déroulement d'un projet complexe. Dans ce sens, il faudra tout d'abord choisir une méthode de développement et ensuite réaliser un planning à suivre respectant cette méthode.

1.3.1 Méthodologie suivie : méthode KANBAN

Kanban est un outil visuel de gestion de projet qui vous aide à suivre les tâches en temps réel, à gérer les priorités et à favoriser la collaboration entre les membres de l'équipe. La méthodologie Lean repose sur quatre principes clairs.

La méthode Kanban a pour but d'aider les équipes à répartir le volume de travail selon la disponibilité de chacun. La structure Kanban est basée sur une philosophie centrée sur le concept d'amélioration continue, où les activités sont « extraites » du backlog produit pour former un flux régulier.

Un tableau Kanban particulièrement simple pourrait simplement être constitué de colonnes intitulées « À faire », « En cours » et « Terminé ». Les tâches individuelles, représentées par des cartes visuelles, passent d'une colonne à l'autre jusqu'à ce qu'elles soient terminées.

Les avantages de la méthode Kanban :

- Une amélioration de la collaboration
- Réduire les coûts de production
- Eviter la surproduction
- Diminuer les délais
- Produire avec la meilleure qualité possible

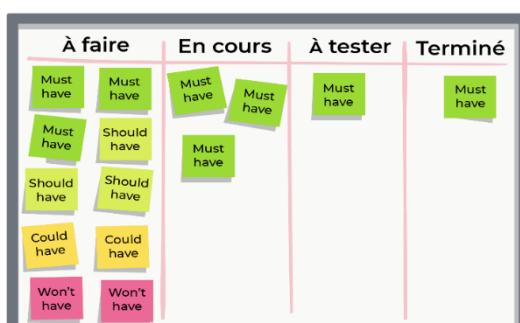


Figure 5 : Exemple de tableau KANBAN

Les idées principales de Kanban

Il y a 3 Caractéristiques principales sur Kanban :

- C'est un outil pour aider tout type d'équipe de projet à organiser le travail et à surveiller l'efficacité en visualisant l'état du flux de travail.
- Les éléments clés du tableau Kanban sont la portée du travail qui est décomposé en petits morceaux gérables.
- Il limite la quantité de travail autorisée dans chaque étape (colonnes) afin que l'équipe soit toujours de manière saine et durable à livrer.

Le principe de Kanban :

Le processus simplifié de la méthode de Kanban se déroule en six étapes sont :

- Répartissement le travail en user stories ou en tâches de plus petite taille.
- Il faut écrire ces user stories dans des cartes et placez-les sur la 1ère colonne du tableau (Backlog).
- Sur la base des user stories ou des tâches, il faut discuter avec l'équipe quelles sont les étapes/colonnes requises.
- Une fois que l'équipe commence à travailler sur les tâches du projet, il faut déplacer ces cartes vers les colonnes correspondantes.
- Création un rituel / routines (par exemple daily stand-up) pour discuter de la progression, les problèmes, la vitesse avec les membres de l'équipe.
- Il faut Réviser la limite de quantité de tâche dans chaque colonne en fonction de la vitesse de l'équipe.

La différence entre la méthode Scrum et la méthode Kanban :

KANBAN	SCRUM
Les rôles sont fluides. Gestionnaire de projet facultatif	Les rôles sont prédéfinis. Scrum master requis
Les tâches sont partagées par tous	Les tâches sont attribuées à certains propriétaires
Les échéanciers évoluent selon les besoins	Les chronologies sont timeboxed en sprints
Des modifications peuvent être apportées à mi-parcours, permettant d'itérations avant l'achèvement d'un projet	Le changement ne peut être effectué qu'à la fin d'un sprint
La productivité est mesurée par le temps de cycle du projet complet	La productivité est mesurée par le nombre de points de chaque sprint

Tableau 2 : Etude comparative entre KANBAN & SCRUM

Exemple d'un tableau Kanban dans Jira :

The screenshot shows a Jira Kanban board for the project 'Beyond Gravity'. The board is divided into four columns: 'TO DO', 'IN PROGRESS', 'IN REVIEW', and 'DONE'. Each column contains several tasks, each with a title, a color-coded label (e.g., ACCOUNTS, BILLING, FORMS), and a unique ID (e.g., NUC-342, NUC-367). The tasks are listed vertically within each column, and each task has a small circular icon representing the assignee.

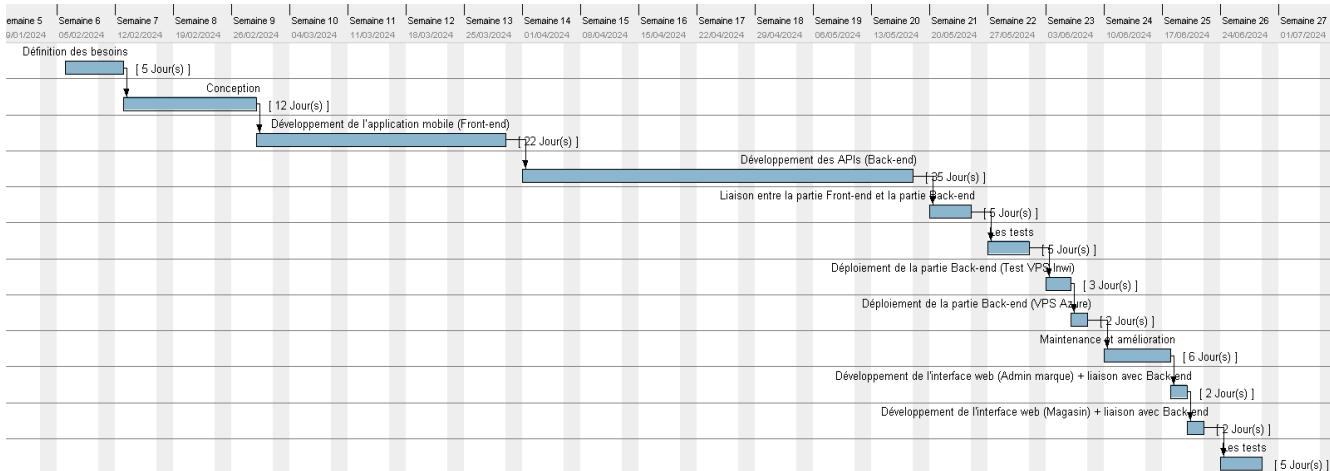
Figure 6 : Tableau KANBAN dans JIRA

1.3.2 Diagramme de GANTT

Avant de démarrer le projet, il est essentiel de prévoir une planification de sa mise en œuvre. L'objectif du planning d'un projet est, d'une part, de découper ce dernier en plusieurs phases intermédiaires afin de permettre une meilleure estimation de la durée totale du projet et des ressources nécessaires, et, d'autre part, de valider séquentiellement le projet pour assurer sa conformité avec les besoins exprimés. Grâce aux réunions tenues avec l'encadrant, nous avons été bien éclairés sur les différentes phases du projet ainsi que leur progression dans le temps. Cela consistait en douze phases, comme décrit dans le déroulement du projet. Ci-dessous se trouvent le diagramme de Gantt du projet ainsi qu'un tableau présentant la planification détaillée.

Nom	Date de début	Date de fin	Durée
Définition des besoins	06/02/2024	12/02/2024	5
Conception	13/02/2024	28/02/2024	12
Développement de l'application mobile (Front-end)	29/02/2024	29/03/2024	22
Développement des APIs (Back-end)	01/04/2024	17/05/2024	35
Liaison entre la partie Front-end et la partie Back-end	20/05/2024	24/05/2024	5
Les tests	27/05/2024	31/05/2024	5
Déploiement de la partie Back-end (Test VPS Inwi)	03/06/2024	05/06/2024	3
Déploiement de la partie Back-end (VPS Azure)	06/06/2024	07/06/2024	2
Maintenance et amélioration	10/06/2024	17/06/2024	6
Développement de l'interface web (Admin marque) + liaison avec Back-end	18/06/2024	19/06/2024	2
Développement de l'interface web (Magasin) + liaison avec Back-end	20/06/2024	21/06/2024	2
Les tests	24/06/2024	28/06/2024	5

Figure 7 : Les Tâches de la Période du PFE



Le tableau ci-dessous présente les objectifs réalisés concrètement dans chaque phase du projet.

Phase	Objectifs
Etude préliminaire de projet	Comprendre l'architecture du projet
	Comprendre le rôle de chaque fonctionnalité du projet.
	Comprendre le fonctionnement général de l'application mobile et web.
	Critique de l'existant.
	Déterminer les spécifications du projet.
Réalisation	Se familiariser avec les outils de développement : Flutter, Angular, Spring boot, Spring Security, Spring cloud
	Préparation des interfaces de l'application mobile et des interfaces web
	Génération des services pour les fonctionnalités de l'application : <ul style="list-style-type: none"> ➤ Scanner de QR Code ➤ Historique d'achats ➤ Statistique ➤ Les offres
	APIs pour les fonctionnalités de l'application.
	Gestion des permissions et sécurité des données
	Tester chaque composant de l'application
	Déployer les deux parties (Frontend de l'application mobile, et le Backend)
	Les tests de validation

Tableau 3 : Description des phases de la planification

Livrable :

- ✓ **Application mobile pour les utilisateurs** : Cette application permet aux clients de scanner leur ticket après l'achat, de consulter l'historique des tickets, de voir des statistiques détaillées de leurs achats, et de localiser les emplacements des supermarchés et des marques où ils peuvent utiliser l'application Corail.

- ✓ **Application web pour les administrateurs de marque** : Une interface conviviale pour la collecte, l'analyse et la gestion des données clients. Cette plateforme permettra aux administrateurs de suivre les tendances d'achat, d'ajuster les offres en fonction des préférences des clients et de faciliter la communication avec la clientèle.
- ✓ **Application web pour les magasins** : Permettant d'afficher le QR code lors de l'achat d'un client et contenant les informations sur les produits achetés.
- ✓ **Rapports analytiques détaillés sur l'utilisation de l'application mobile et web** : Fournissant des statistiques approfondies sur les retours de produits, les préférences des clients et d'autres indicateurs clés de performance pour les deux plateformes.

Gestion des risques :

Afin de réussir le bon déroulement des différentes phases du projet, nous avons eu recours à l'anticipation des risques qui pourront affecter le projet.

L'identification initiale des risques s'est effectuée en fonction des objectifs et des exigences. Pour effectuer ce recensement, nous avons organisé une séance de brainstorming avec le responsable afin de clarifier les points essentiels et de discuter des risques potentiels. Cette réunion a permis de recueillir différentes perspectives et d'identifier les risques susceptibles de survenir tout au long du projet.

Ensuite, les risques sont analysés (classification selon une typologie et estimation de leur probabilité d'apparition) et leurs conséquences évaluées (en termes d'impacts).

Risque	Impact	Classement	Action préventive
La productivité diminue un peu dans le mois du RAMADAN	Ralentissement des travaux	Faible	- Doubler l'effort et travailler un temps extra après le mois sacré du RAMADAN
Événement GITEX	Ralentissement des travaux	Faible	- Doubler l'effort et travailler un temps extra.
La Fête de l'Aïd al-Adha	Stop des travaux	Moyen	- Doubler l'effort et travailler un temps extra

Tableau 4 : Gestion des risques

1.3.3 Analyse SWOT

L'analyse SWOT est un outil stratégique utilisé pour identifier les forces (Strengths), les faiblesses (Weaknesses), les opportunités (Opportunities) et les menaces (Threats) d'une organisation, d'un projet ou d'une situation. Voici un bref aperçu de chacun des éléments :

- **Forces** : Atouts internes qui donnent un avantage compétitif.

- **Faiblesses** : Points faibles internes qui peuvent nuire à la performance.
- **Opportunités** : Facteurs externes favorables à exploiter.
- **Menaces** : Facteurs externes défavorables pouvant poser des risques.

Le tableau suivant présente une analyse SWOT du projet de digitalisation des tickets :

Forces	Faiblesses
Efficacité accrue : La digitalisation des tickets permet une gestion plus rapide et plus efficace des demandes des utilisateurs.	Complexité initiale : La mise en place d'une architecture microservices peut être complexe et nécessite des compétences techniques élevées.
Scalabilité : Grâce à l'architecture microservices, le système peut facilement s'adapter à une augmentation du nombre d'utilisateurs.	Temps de développement : Le développement et le déploiement initial peuvent prendre du temps en raison de la complexité de l'architecture et des technologies utilisées.
Opportunités	Menaces
Adoption croissante de la digitalisation : La tendance mondiale vers la digitalisation des services crée un environnement favorable à l'adoption de ce système.	Concurrence : L'augmentation de la concurrence dans le domaine de la digitalisation des tickets peut exercer une pression sur les prix et les marges.
Amélioration continue : Les retours des utilisateurs et les avancées technologiques permettent une amélioration continue du système.	Évolutions technologiques rapides : La rapidité des changements technologiques peut rendre certaines parties du système obsolètes rapidement, nécessitant des mises à jour fréquentes.

Tableau 5 : Analyse SWOT de la digitalisation des tickets

Conclusion

Après une brève présentation de l'organisme d'accueil, CORAIL MAROC, nous avons donné une situation générale de notre projet de fin d'étude qui s'intègre dans le cadre de la « Digitalisation des tickets ». Nous avons également explicité la problématique objet de notre intervention. Ce chapitre détaille également les étapes et la méthodologie de réalisation de notre projet que nous avons adoptées afin de répondre aux normes et aux exigences de la qualité logicielle. Le contexte général du projet étant défini, il convient d'étudier les différents besoins des utilisateurs et d'aboutir à la conception du système. Le chapitre suivant traitera en détail de la phase conceptuelle du projet.

Chapitre 2

Etude conceptuelle du projet

La phase de conception est la phase initiale qui consiste à créer et à mettre en œuvre notre projet. En fait, cette phase est considérée comme importante car il s'agit d'une étape de réflexion dans le processus de développement logiciel après la phase de détection de la problématique, de proposition d'une solution et de spécification.

Dans ce chapitre, nous présenterons en détails la conception de notre projet à travers les diagrammes UML.

2. Etude conceptuelle du projet

Avant la réalisation d'un projet, il faut pratiquer une analyse informatique. Pour ça nous analysons et spécifions les besoins du projet au niveau des fonctionnalités requises et des exigences non fonctionnelles. Cette analyse consiste à comprendre et modéliser le projet sur lequel nous voulons travailler.

Ce chapitre donne une explication pour le déroulement de notre projet ainsi qu'assurer une bonne compréhension des besoins.

2.1 Etude fonctionnelle

Cette partie consiste à capturer et à formaliser la vision du produit (solution) que nous souhaitons développer, tout en définissant les interactions attendues entre le système et les utilisateurs.

2.2.1 Critique de l'existant

Actuellement, le système de gestion des tickets utilise principalement des supports papier et des processus manuels. Cette méthode présente plusieurs inconvénients significatifs :

- **Lenteur et inefficacité** : Le traitement manuel des tickets est chronophage et sujet aux erreurs humaines. La recherche et la gestion des tickets papier prennent du temps, ce qui ralentit les opérations et affecte la productivité.
- **Risque de perte et de détérioration** : Les tickets papier peuvent facilement être perdus, endommagés ou mal classés, ce qui entraîne des pertes de données et complique le suivi des incidents.
- **Accès limité aux informations** : L'accès aux tickets papier est souvent limité à un seul endroit physique, ce qui complique la collaboration entre les équipes et le partage des informations.
- **Manque de visibilité et de traçabilité** : Il est difficile de suivre l'état et l'historique des tickets de manière efficace avec un système papier, ce qui entraîne un manque de visibilité sur les performances et les tendances.

2.2.2 Les besoins fonctionnels

Les exigences fonctionnelles ou métier représentent les actions que le système doit effectuer. Il ne sera opérationnel que si ces conditions sont remplies. Cette candidature doit principalement couvrir les exigences fonctionnelles suivantes :

❖ Application mobile : Client

Les Besoins fonctionnels	Description
Processus d'inscription	<ul style="list-style-type: none"> ➤ Fournit des informations d'inscription : nom, prénom, ville, numéro de téléphone, adresse e-mail. ➤ Confirmation par e-mail avec un code de validation. ➤ Création d'un mot de passe. ➤ Connexion au compte avec le numéro de téléphone et le mot de passe.
Scan Code QR	<p>Les utilisateurs effectuent le paiement de leur ticket en scannant le code QR affiché sur le bord Corail lors de leur passage en caisse. Le système envoie automatiquement le ticket au client.</p>
Consultation des tickets	<ul style="list-style-type: none"> ➤ L'utilisateur reçoit une notification contenant le ticket lorsqu'il effectue un nouvel achat. ➤ L'utilisateur consulte les tickets qu'il achetés instantanément. ➤ L'utilisateur consulte les statistiques des achats par marque, date, montant
Politique de gestion des sessions inactives	<ul style="list-style-type: none"> ➤ Déconnexion automatique après 1 heure d'inactivité. ➤ Possibilité pour les utilisateurs de visualiser et de déconnecter leurs sessions actives à partir de différents appareils. ➤ Connexions multiples autorisées avec détection des activités suspectes. ➤ Réactivation de session par authentification.

Tableau 6 : Les Exigences fonctionnelles pour l'application mobile (Client)

❖ Application web : Administrateur de la marque

Les Besoins fonctionnels	Description
Gestion des utilisateurs magasins	<ul style="list-style-type: none"> ➤ Crédit de nouveaux comptes utilisateurs magasins pour étendre la communauté magasin associée à la marque. ➤ Modification des informations des comptes utilisateurs magasin existants pour maintenir des données précises.
Contrôle d'accès	<ul style="list-style-type: none"> ➤ Attribution ou révocation de rôles pour les utilisateurs magasins, offrant un contrôle fin sur les permissions d'accès. ➤ Gestion des droits d'accès pour ajuster les fonctionnalités spécifiques accessibles par les utilisateurs magasins

Accès complet aux fonctionnalités de l'application	L'administrateur marque à un accès complet aux fonctionnalités de l'application, incluant la possibilité de modifier les droits d'accès, et la consultation des statistiques d'achats globales.
---	---

Tableau 7 : Les Exigences fonctionnelles pour l'application web (Administrateur marque)

❖ Application web : Magasin

Les Besoins fonctionnels	Description
Authentification	Au niveau du magasin, le commerçant reçoit un code de la part de l'administrateur de la marque. Ce code lui permet de s'authentifier sur l'application web.
Affiche de QR code	Après que le client passe à la caisse pour payer, le commerçant affiche le QR code au client. Ce dernier scanne le QR code et recevoir son ticket.

Tableau 8 : Les Exigences fonctionnelles pour l'application web (magasin)

2.2.3 Les besoins non fonctionnels

Les spécifications non-fonctionnelles sont des exigences qui ne concernent pas directement les services spécifiques fournis par le système. Ils peuvent concermer des propriétés et des contraintes remplies par le système dans son intégralité.

Les principaux besoins non fonctionnels de notre application sont résumés dans le tableau suivant :

Exigences	Description
La rapidité	L'application doit exécuter les fonctions demandées dans un temps raisonnable. (Utilisation de Spring boot)
La scalabilité	L'application doit pouvoir gérer une augmentation du nombre d'utilisateurs et du volume de données sans dégradation des performances. (Utilisation de l'architecture microservices)
La convivialité	L'application doit être facile à utiliser avec une interface simple et attractive. (Utilisation de UX/UI Design + les tests en mode réel)
La disponibilité	Lorsqu'un utilisateur désire utiliser l'application, cette dernière doit être disponible. (Déploiement dans Play store)
La Maintenabilité	Facilité avec laquelle on peut localiser et corriger les pannes. (Utilisation de Docker logs)
La sécurité	L'application doit manipuler à terme une base de données qui peut s'avérer importante, il est important de sécuriser l'accès à cette dernière ainsi que les données manipulées. (Utilisation de Spring Security)
Contraintes techniques	L'accès à la base de données doit être souple et rapide. L'application doit être toujours fonctionnelle. (Déploiement dans le Cloud)
Compatibilité	Compatibilité avec une large gamme de dispositifs mobiles et de navigateurs web. Adaptabilité aux différentes résolutions d'écran et orientations. (Utilisation de Flutter et Angular)

Tableau 9 : Les Exigences non fonctionnelle

2.2 Conception

Pour une bonne étude analytique et conceptuelle du projet on s'est basé sur l'utilisation des diagrammes UML.

1. Diagramme de classe

On commence par le diagramme de classe pour définir la structure des objets dans le projet, les relations entre eux, ainsi que leurs attributs. Cela aide à visualiser et à concevoir l'architecture du projet avant de commencer le code.

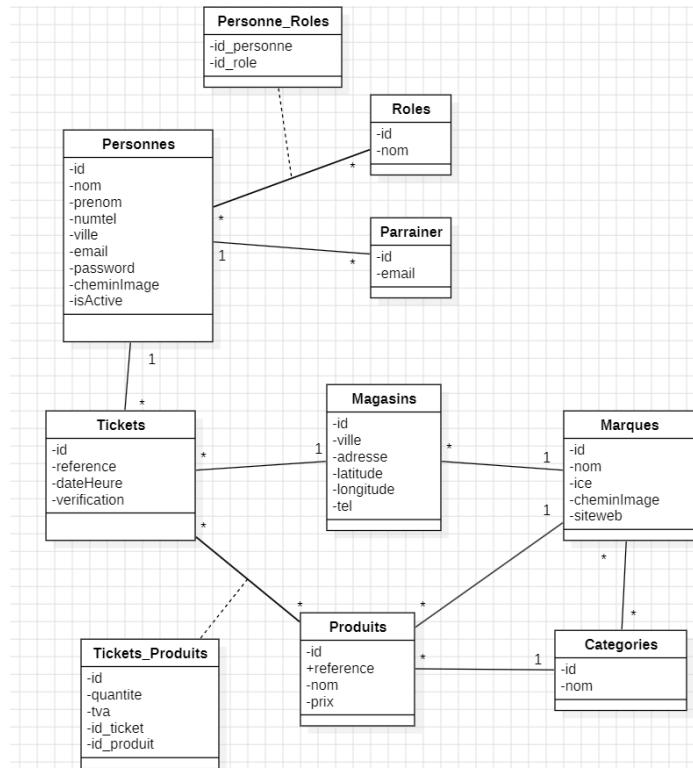


Figure 9 : Diagramme de classe

Le diagramme contient des éléments clés :

- Entités** : Les boîtes représentent les entités de la base de données, comme “Personnes” avec des attributs tels qu’id, nom, prénom et email.
- Relations** : Les lignes indiquent comment ces entités sont reliées, avec des symboles montrant la nature de leur relation, comme un-à-plusieurs ou plusieurs-à-un.
- Attributs** : Chaque entité contient une liste d’attributs qui définissent ses propriétés, comme “nom”, “prenom”, “numtel” (numéro de téléphone), et “ville” pour l’entité “Personnes”.

2. Description des tables du diagramme de classe

Table et Association	Description
Personnes	Pour ajouter une nouvelle personne avec son nom, prénom, numéro de téléphone, ville, email, mot de passe, image et isActive pour activer et désactiver le compte.

Roles	Pour ajouter les rôles avec un nom.
Personnes_Roles	Association permettant d'affecter à chaque personne un ou plusieurs rôles.
Parrainer	Pour enregistrer l'email de la personne invitée.
Tickets	Pour enregistrer les tickets des personnes avec une référence, une date et une heure, et vérifier si l'utilisateur consulte la notification du ticket ou non.
Produits	Pour enregistrer les produits avec une référence, un nom et un prix.
Tickets_Produits	Association pour enregistrer les produits de chaque ticket avec la quantité et la TVA.
Catesgories	Pour ajouter les catégories avec un nom.
Marques	Pour ajouter les marques avec un nom, un ICE, une image et le site web.
Magasins	Pour enregistrer les magasins avec la ville, l'adresse, la latitude, la longitude et le numéro de téléphone.

Tableau 10 : Description des tables du diagramme de classe

3. Diagramme de cas d'utilisation

Un diagramme de cas d'utilisation est un type de diagramme UML qui montre les interactions entre les acteurs (utilisateurs ou autres systèmes) et le système lui-même. Ce diagramme permet de visualiser les différents scénarios où les acteurs interagissent avec le système pour accomplir une tâche spécifique.

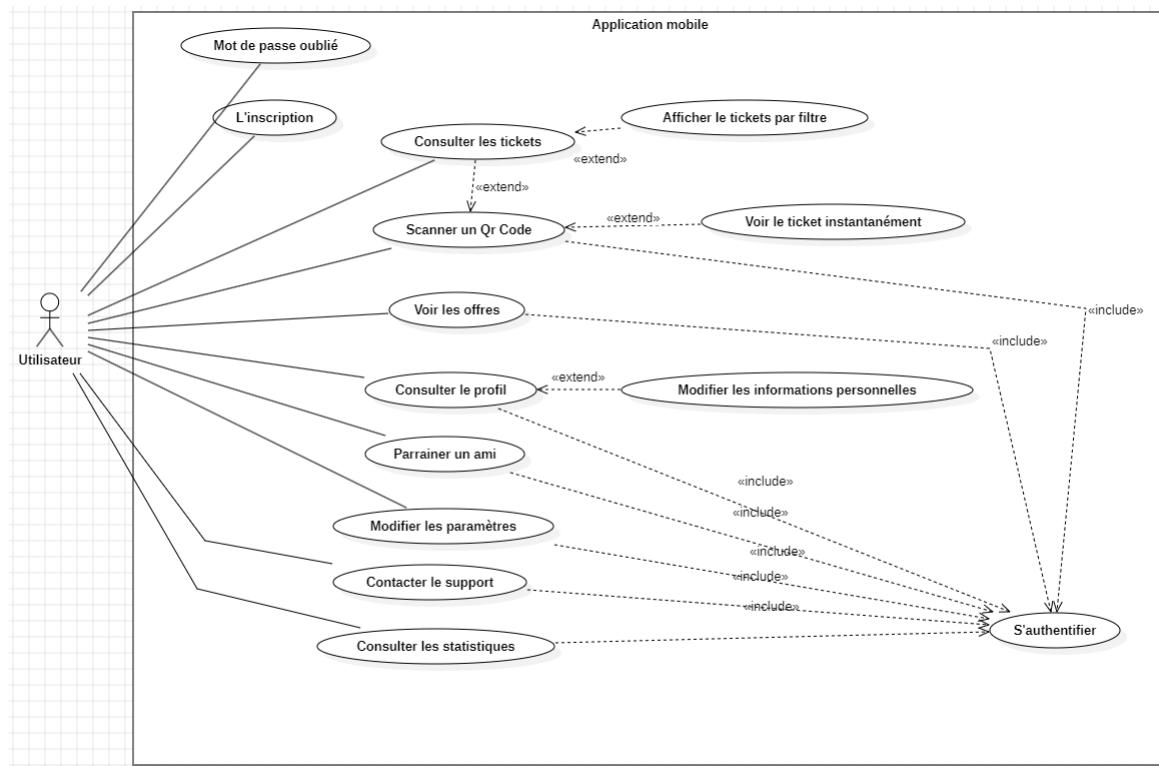


Figure 10 : Diagramme de cas d'utilisation – Utilisateur application mobile

Le diagramme présente les fonctionnalités offertes à l'utilisateur de l'application mobile, telles que le scan et la consultation des tickets, la consultation des statistiques, les offres, les informations personnelles, et d'autres fonctionnalités après l'authentification.

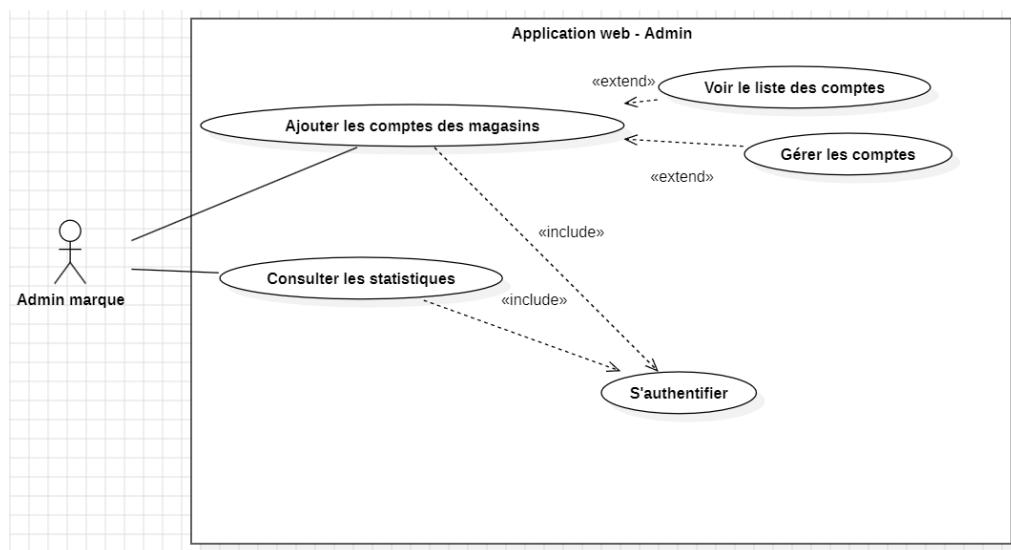


Figure 11 : Diagramme de cas d'utilisation – Administrateur marque (application web)

Le diagramme présente les fonctionnalités de l'administrateur d'une marque telles que la consultation des tickets, l'ajout des comptes pour les magasins avec la possibilité de consulter les comptes et de les gérer, le tout après l'authentification.

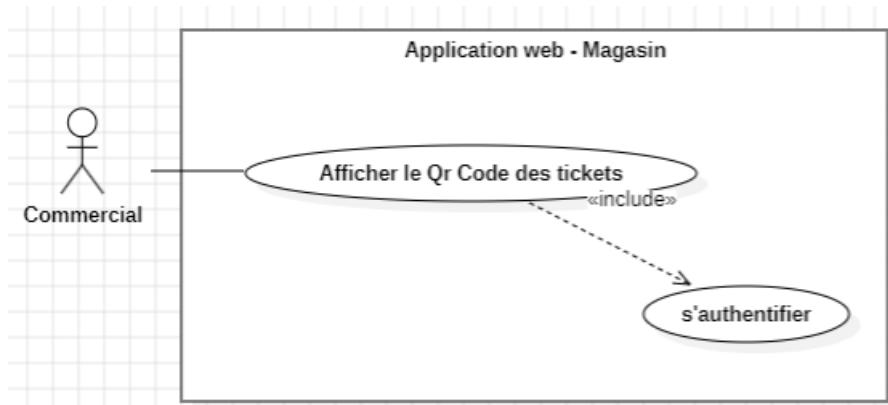


Figure 12: Diagramme de cas d'utilisation –Commercial (application web du magasin)

Le diagramme présente la seule fonctionnalité du commercial : affichage de QR Code des tickets, après l'authentification bien sûr.

4. Diagramme d'activité

Le diagramme d'activité est un diagramme comportemental d'UML, permettant de représenter le déclenchement d'événements en fonction des états du système et de modéliser des comportements parallélisables.

Les scénarios nominales	Description
Remplir le formulaire d'inscription correctement	L'utilisateur commence par cette étape

Cliquer sur enregistrer	Si le formulaire est rempli correctement, l'utilisateur doit cliquer sur le bouton d'enregistrement.
Formulaire valide	Le système vérifie la validité du formulaire.
Envoyer le code de vérification à l'adresse email	Si le formulaire est valide, un code de vérification est envoyé à l'email de l'utilisateur.
Saisie du code	L'utilisateur entre le code de vérification reçu.
Correct	Si le code est correct, le processus se poursuit normalement
Les scénarios alternatifs	Description
Formulaire invalide	Si le formulaire n'est pas valide après avoir cliqué sur enregistrer, un message d'erreur s'affiche.
Code incorrect	Si le code de vérification saisi est incorrect, un message d'erreur s'affiche.
Ressaisir	L'utilisateur à la possibilité de ressaisir le code.
Utilisateur quitte l'application	Si l'utilisateur décide de quitter l'application à n'importe quel moment.
Rester sur la page du code de vérification	L'utilisateur peut choisir de rester sur la page pour essayer de saisir à nouveau le code.
Contacter le support	Si l'utilisateur rencontre des problèmes, il peut contacter le support pour obtenir de l'aide.

Tableau 11 : Les scénarios de l'inscription de l'utilisateur

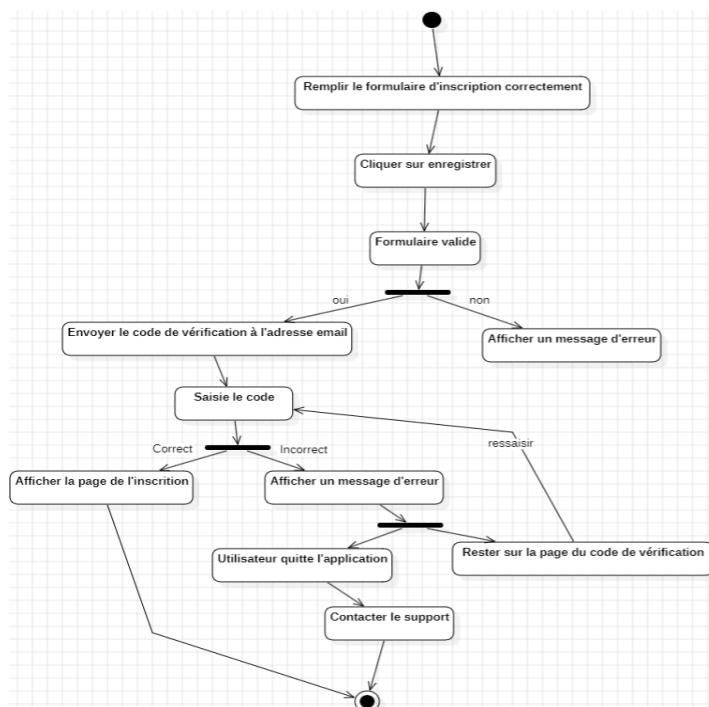


Figure 13 : Diagramme d'activité – L'inscription de l'utilisateur (application mobile)

Les scénarios nominales	Description
Recevoir le fichier JSON	Le processus commence par la réception d'un fichier au format JSON.
Extraction des données	Les données sont extraites du fichier JSON.
Générer un QR code	Un QR code est généré à partir des données extraites.
Scanner le QR code	Le client scanne le QR code.
Enregistrer les données	Si le QR code est correctement scanné, les données sont enregistrées dans la base de données.
Afficher le ticket et masquer le QR code	Enfin, le ticket est affiché et le QR code est masqué.
Les scénarios alternatifs	Description
QR code non scanné	Si le QR code n'est pas correctement scanné, le processus revient à l'étape de scan.

Tableau 12 : Les scénarios du scan de ticket

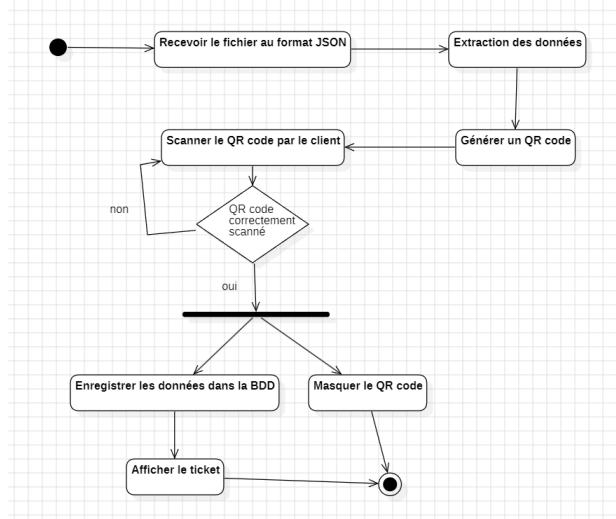


Figure 14 : Diagramme d'activité – Scan ticket (application mobile)

Les scénarios nominales	Description
Recevoir un code de l'admin	Le processus commence par la réception d'un code de la part de l'administrateur.
Saisir le code	L'utilisateur entre le code reçu.
Correct	Si le code est correct, l'utilisateur accède à la page d'affichage des QR Codes.
Les scénarios alternatifs	Description
Incorrect	Si le code saisi est incorrect, un message "Code incorrect" s'affiche.
Ressaisir	L'utilisateur a la possibilité de ressaisir le code.

Tableau 13 : Les scénarios de l'authentification – application web magasin

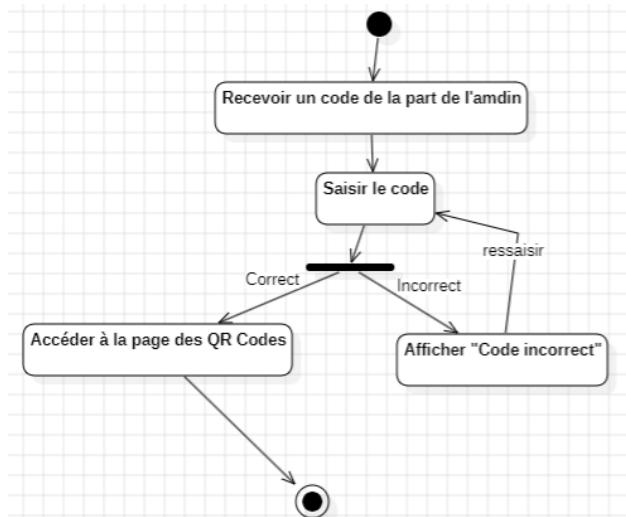


Figure 15 : Diagramme d'activité – Authentification (application web magasin)

5. Diagramme de séquence

Un diagramme de séquence est une représentation visuelle qui montre comment différents objets interagissent dans un système à travers le temps, en utilisant des lignes de vie pour chaque objet et des messages pour montrer les interactions entre eux.

Les scénarios nominales	Description
Téléphone/email & mot de passe	L'utilisateur entre son identifiant (téléphone ou email) et son mot de passe dans l'interface de connexion.
Spring Security FilterChain	Le système utilise Spring Security pour filtrer la requête de connexion.
JWT Authentication Filter	Un filtre d'authentification JWT vérifie les informations et génère un token si elles sont correctes.
UserDetailsServiceImp	Ce service charge les détails de l'utilisateur à partir de la base de données.
PersonneServiceImp	Ce service peut être une implémentation personnalisée pour gérer les informations spécifiques de l'utilisateur.
findUserByUsername	Une requête SQL est exécutée pour trouver l'utilisateur par son nom d'utilisateur.
Vérifier le mot de pass	Le système vérifie si le mot de passe entré correspond à celui stocké dans la base de données.
Générer JWT	Si l'authentification est réussie, un token JWT est généré pour l'utilisateur.
Autorisation ; Bearer JWT	Le token JWT est utilisé pour autoriser l'utilisateur à accéder aux ressources protégées.
Les scénarios alternatifs	Description
Mot de passe incorrect	Si le mot de passe entré est incorrect, le système

	affiche un message d'erreur.
Utilisateur non trouvé	Si le nom d'utilisateur n'est pas trouvé dans la base de données, un message d'erreur est également affiché.

Tableau 14: Les scénarios de l'authentification JWT

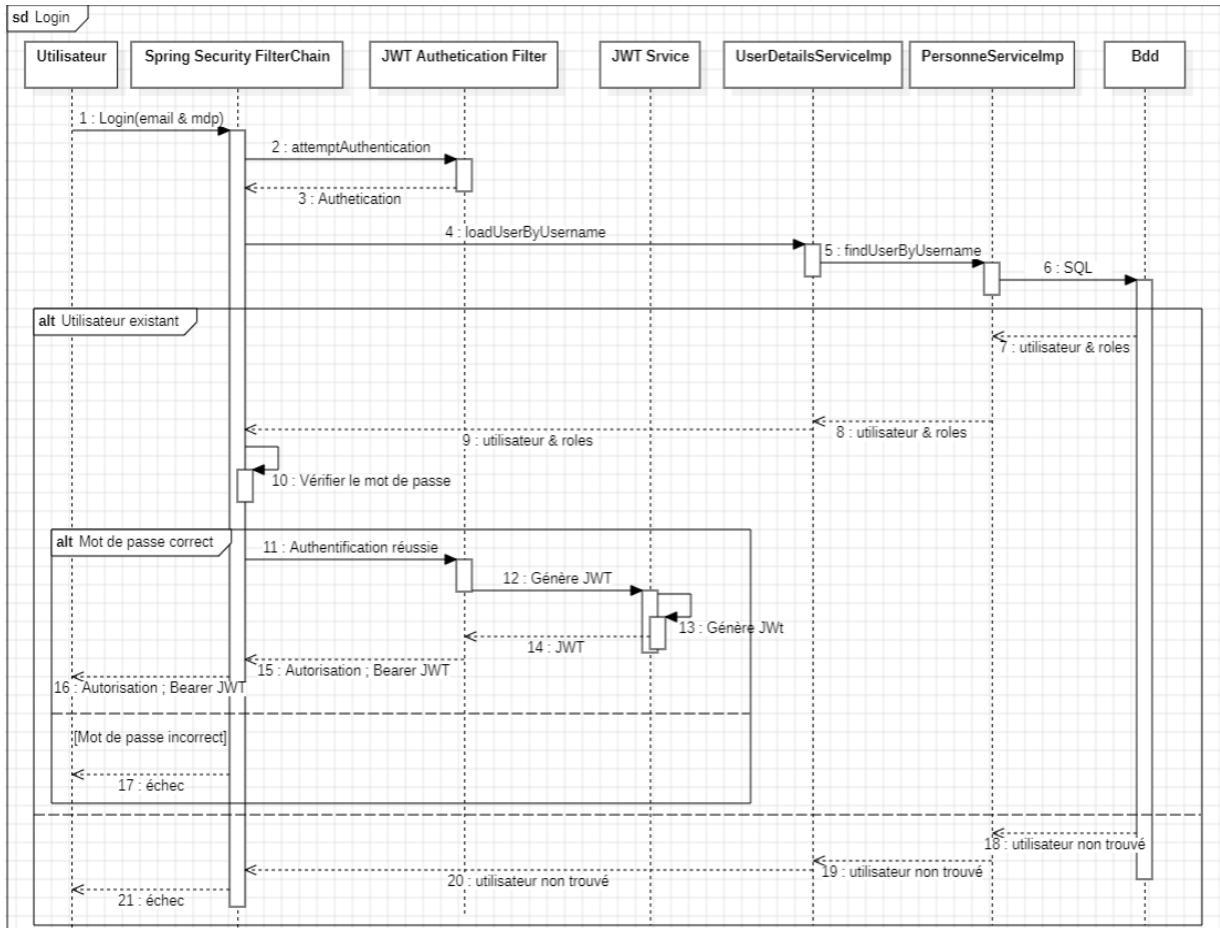


Figure 16 : Diagramme de séquence – Authentification JWT (Sécurité)

Ce diagramme illustre le processus d'authentification d'un utilisateur dans un système informatique en utilisant Spring Security, qui est un cadre de sécurité robuste pour les applications Java.

Les scénarios nominales	Description
Authentifier	L'utilisateur envoie une demande d'authentification au Service Authentication.
Envoyer le token	Le Service Authentication transmet le token à l'utilisateur.
Récupérer les tickets	Après une authentification réussie, l'utilisateur récupère des tickets (Autorisation/Bearer JWT).
Vérifier le JWT	Le Service Core vérifie le JWT en le comparant avec la clé secrète du Service Authentication.
Les scénarios alternatifs	Description
JWT invalide	Si une réponse avec un JWT invalide est reçue, il y a un retour un avis d'échec.

Tableau 15: Les scénarios de la sécurité entre les services

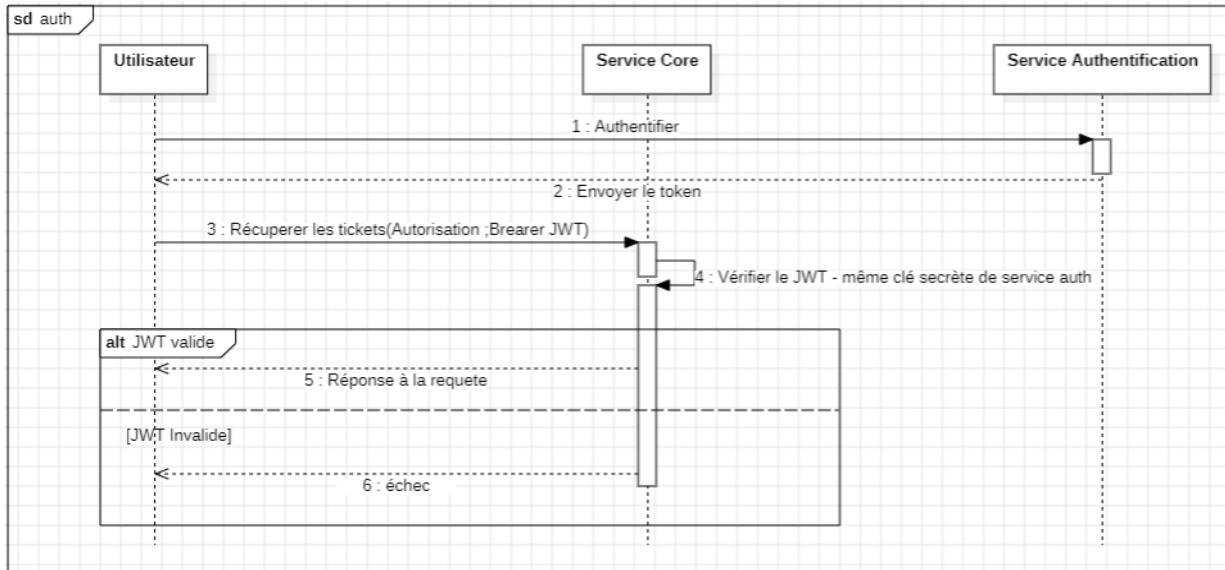


Figure 17 : Diagramme de séquence – Vérification du token entre les services (Sécurité)

Ce diagramme représente le processus de vérification du token entre deux services. L'utilisateur s'authentifie et envoie une requête vers l'autre serveur pour afficher ses tickets. La procédure commence au niveau du service core, où il vérifie si la même clé secrète est utilisée et si le token est valide, afin d'envoyer la réponse.

6. Conception de l'API

Le tableau suivant représente les différentes APIs du service authentication :

Méthode	Ressource	Description
POST	<code>{url}/auth/user-verify</code>	Saisir le code envoyé à l'email de l'utilisateur pour activer son compte.
POST	<code>{url}/auth/parrainer</code>	Inviter un ami en envoyant une demande à son email.
POST	<code>{url}/auth/mdp_oublie</code>	Récupération du mot de passe en cas de perte par email.
POST	<code>{url}/auth/logout</code>	Déconnexion
POST	<code>{url}/auth/login_client</code>	Authentification de l'utilisateur de l'application mobile.
POST	<code>{url}/auth/login</code>	Authentification des utilisateurs des applications web.
POST	<code>{url}/auth/addImage</code>	Ajouter et modifier une image, que ce soit pour l'utilisateur de l'application mobile ou pour les photos des marques.
POST	<code>{url}/auth/UpdateUser</code>	Modifier les informations de l'utilisateur de l'application mobile.
POST	<code>{url}/auth/UpdatePass</code>	Changer le mot de passe de l'utilisateur de l'application mobile.
POST	<code>{url}/auth/SousRetail-verify</code>	La connexion au compte du magasin (application web).
POST	<code>{url}/auth/Ajouter_Role</code>	Ajouter des rôles.
POST	<code>{url}/auth/Ajouter_Personne</code>	Ajouter des personnes (utilisateur de l'application mobile, administrateur de marque, magasin).
GET	<code>{url}/auth/user</code>	Afficher les informations des utilisateurs.
GET	<code>{url}/auth/roles</code>	Afficher tous les rôles.

Tableau 16: Les APIs du service d'authentification

Le tableau suivant représente les différentes APIs du service core :

Méthode	Ressource	Description
POST	{{url}}/core/upload	Génération d'un QR Code (transformation du fichier JSON contenant le ticket en un QR Code).
POST	{{url}}/core/date/{id}	Ajouter le ticket à la base de données après le scan.
GET	{{url}}/core/Tickets	Afficher tous les tickets.
GET	{{url}}/core/Tickets-Filter	Afficher les tickets en filtrant par catégorie et date.
GET	{{url}}/core/Tickets-Brand	Afficher les tickets en filtrant par marque.
GET	{{url}}/core/TicketId	Afficher le ticket par son ID.
GET	{{url}}/core/AllNotif	Afficher toutes les notifications.
POST	{{url}}/core/NotNotif	Afficher les notifications non lues.
GET	{{url}}/core/CheckedNotif	Afficher les notifications lues.
GET	{{url}}/core/CountClient	Afficher le nombre de clients.
POST	{{url}}/core/AjouterCategorie	Ajouter une catégorie.
POST	{{url}}/core/AjouterMarque	Ajouter une marque.

Tableau 17: Les APIs du service core

Conclusion

Ce Dans ce chapitre, nous avons abordé l'analyse et l'étude conceptuelle du projet. Nous avons d'abord détaillé les besoins fonctionnels et non fonctionnels. Ensuite, nous avons présenté le diagramme de classes, le diagramme de cas d'utilisation, le diagramme d'activités, et le diagramme de séquence des différents services. Nous avons terminé avec la conception des APIs. Le chapitre suivant sera consacré à la phase de choix des technologies et des outils, ainsi qu'à l'architecture du projet.

Chapitre 3

Outils, Choix des technologies et architecture

Ce chapitre se concentre sur les outils, le choix des technologies et l'architecture de notre projet. Nous y identifions les technologies appropriées, sélectionnons les outils nécessaires et définissons l'architecture du système. Cette analyse nous permet de concevoir une infrastructure robuste et efficace qui répond aux normes de qualité établies. Nous détaillons également les critères de sélection des technologies et la manière dont elles s'intègrent dans notre méthodologie de développement.

3. Etude et analyse de projet

L'analyse des besoins est une étape primordiale dans chaque projet. En effet, nous analysons et spécifions l'architecture du projet et les technologies et outils de développement utilisés pour sa réalisation.

3.1 Choix techniques et Architecture

L'architecture et le choix technique sont des étapes essentielles dans la construction de tout projet. Ce chapitre nous permet de sélectionner les outils appropriés aux exigences techniques de notre solution et de notre architecture, nous permettant ainsi d'implémenter efficacement les besoins fonctionnels et non fonctionnels.

2.3.1 Besoins techniques

Pour appliquer l'idée de digitalisation des tickets de caisse, plusieurs critères sont requis :

- Développer une application mobile robuste et simple à utiliser.
- Développer deux applications web, l'un pour administrateur des marques, et l'autre pour les magasins.
- Utiliser des technologies modernes.
- Utiliser des web services REST.
- Réduire le temps de réponses des services.
- S'appuyer sur une architecture évolutive, maintenable et robuste.

Pour satisfaire ces contraintes, nous avons décidé, suite aux réunions effectuées entre les membres de développement de l'équipe Soin, d'adopter les choix techniques et technologiques décrits dans la suite de cette partie.

2.3.2 Choix technologiques

Dans cette section, nous présentons le choix des langages de programmation et des outils utilisés pour le projet ainsi que l'architecture de ce dernier.

❖ Les langages de programmation

Flutter :



Figure 18 : Flutter

Flutter est un framework open-source de Google qui permet de créer des applications mobiles, web et desktop à partir d'un seul code source. Il utilise le langage de programmation Dart et se distingue par ses widgets personnalisables, ses performances proches du natif, et son "Hot Reload" qui permet de voir instantanément les modifications de code.

Angular :



Figure 19: Angular

Angular est un framework open-source de Google pour développer des applications web. Utilisant TypeScript comme langage principal, Angular permet de créer des applications dynamiques et réactives grâce à ses fonctionnalités avancées comme la gestion des composants, le data binding bidirectionnel, et l'injection de dépendances.

Spring boot :



Figure 20 : Spring Boot

Spring Boot est un framework open-source de Java qui simplifie la création d'applications web et de services microservices. Il fait partie de l'écosystème Spring et permet de démarrer rapidement des applications en fournissant des configurations par défaut et des starters prédefinis pour intégrer diverses technologies.

Spring Security :



Figure 21 : Spring Security

Spring Security est un framework de sécurité pour les applications Java, faisant partie de l'écosystème Spring. Il fournit des fonctionnalités complètes de sécurité, telles que l'authentification, l'autorisation, la protection contre les attaques CSRF (Cross-Site Request Forgery), et la gestion des sessions.

Spring Cloud :



Figure 22 : Spring Cloud

Spring Cloud est un ensemble de projets de l'écosystème Spring conçu pour faciliter le développement de systèmes distribués et de microservices. Il offre des outils pour gérer des problèmes courants dans les architectures cloud, tels que la configuration distribuée, la découverte de services, le routage, les appels à des API REST, la tolérance aux pannes, la gestion des sessions, et la sécurité.

❖ **Outil de stockage :**

Minio:



Figure 23 : Minio

MinIO est une solution de stockage d'objets open-source compatible avec l'API S3 d'Amazon. Conçue pour stocker de grandes quantités de données non structurées, MinIO est hautement performante, évolutive, et adaptée aux environnements cloud natifs. Elle permet de déployer des systèmes de stockage distribués sur des infrastructures standard.

❖ Service de base de données :

MySQL :



Figure 24 : MySQL

MySQL est un système de gestion de base de données relationnelle (SGBDR) open-source. Utilisé pour stocker, organiser et gérer des données, MySQL est connu pour sa fiabilité, ses performances élevées, et sa facilité d'utilisation. de nombreux systèmes d'exploitation et plateformes.

❖ Outils de développement

IntelliJ IDEA :



Figure 25 : IntelliJ IDEA

IntelliJ IDEA est un environnement de développement intégré (IDE) pour les langages de programmation Java et autres, développé par JetBrains. Il offre des fonctionnalités avancées comme l'auto-complétion de code, la refactorisation, le débogage, l'intégration avec les systèmes de contrôle de version (comme Git).

VS Code :



Figure 26 : VS Code

Visual Studio Code, souvent abrégé en VS Code, est un éditeur de code source développé par Microsoft. Il est largement utilisé par les développeurs pour écrire et déboguer du code dans une variété de langages de programmation.

WampServer :



Figure 27 : WampServer

WampServer est un logiciel pour Windows qui regroupe Apache, MySQL (ou MariaDB) et PHP dans un environnement de développement web local. Il permet aux développeurs de créer, tester et déployer des sites web sans avoir besoin d'une connexion internet constante, ce qui accélère le processus de développement.

❖ Outil de gestion de version :

Github :



Figure 28 : Github

GitHub est une plateforme web utilisée pour l'hébergement et la gestion de projets de développement logiciel. Elle utilise Git pour le contrôle de version et offre des fonctionnalités robustes de collaboration, telles que la gestion des pull requests et des tickets.

❖ Outil de conteneurisation :

Docker :



Figure 29 : Docker

Docker est une plateforme permettant de créer des conteneurs virtuels légers pour exécuter des applications de manière isolée et portable. Cela simplifie le déploiement d'applications en assurant la cohérence et l'efficacité sur différents environnements.

❖ Outil de déploiement :

VPS :



Figure 30 : VPS Azure

VPS (Virtual Private Server) est une solution d'hébergement web qui utilise la virtualisation pour créer un serveur privé isolé à partir d'un serveur physique plus grand. Il permet aux utilisateurs d'avoir un contrôle complet sur leur environnement, y compris le système d'exploitation et les logiciels installés, tout en offrant une flexibilité et des performances comparables à un serveur dédié, mais à un coût souvent plus abordable.

❖ Outil de test d'API :

Postman :



Figure 31 : Postman

Postman est une application utilisée par les développeurs pour tester, déboguer et documenter des API. Elle permet de créer facilement des requêtes HTTP pour interagir avec des services web, automatiser les tests d'API, et faciliter la collaboration au sein des équipes de développement.

❖ Outil de documentation d'API REST :

Swagger :



Figure 32 : Swagger

Swagger est un outil open source qui permet de concevoir, documenter et tester des API RESTful. Il utilise la spécification OpenAPI pour décrire la structure des API de manière standardisée. Swagger simplifie le processus de développement d'API en fournissant des outils pour la documentation, le design, le test et la génération de code.

❖ Outil de communication et collaboration :

Slack :



Figure 33 : Slack

Slack est une plateforme de messagerie instantanée et de collaboration conçue pour les équipes de travail. Elle permet des communications en temps réel, le partage de fichiers, l'intégration avec d'autres outils et facilite la gestion des projets au sein des organisations.

Jira :



Figure 34 : Jira

Jira est un outil de gestion de projet utilisé principalement pour suivre les tâches et organiser les projets au sein des équipes de développement. Il offre des fonctionnalités pour la gestion agile, comme les tableaux Kanban et Scrum, ainsi que des rapports pour analyser les performances des projets.

❖ Outil de messagerie :

SMTP :



Figure 35 : SMTP

SMTP (Simple Mail Transfer Protocol) est un protocole de communication utilisé pour l'envoi de courriels sur internet. Il fonctionne en transférant des messages d'un serveur de messagerie à un autre, ou d'un client de messagerie à un serveur de messagerie.

❖ Outil d'analyse statique du code source :

Qodana:



Figure 36 : Qodana

Qodana est un outil conçu pour améliorer la qualité du code et la productivité des développeurs. Son rôle principal est d'effectuer une analyse statique du code source, de détecter les problèmes potentiels tels que les bugs, les violations de bonnes pratiques de programmation, les vulnérabilités de sécurité, et de fournir des recommandations pour les améliorations.

2.3.3 Raison de choix des technologies et des outils :

Dans cette partie, nous allons justifier les choix de langages et d'outils utilisés, ainsi que faire une comparaison avec d'autres langages et outils existants.

- **Langage de programmation : Frontend mobile - Flutter**

Flutter : Framework open source développé, permet de développer des applications pour iOS, Android, web et desktop avec une seule base de code. Cela réduit le temps et les coûts de développement.

React native : Framework open source développé par Facebook qui permet de créer des applications mobiles pour iOS et Android en utilisant JavaScript et React.

Comparaison :

	Flutter	React Native
Langage de programmation	Dart	JavaScript
Performance	Excellent	Faible
Rapidité et performance	Rapide	Lent
Interface utilisateur	Compatibilité avec plusieurs devices	Compatible avec les outils standards
Documentation	Clair et structuré	Pas structure

Tableau 18: Comparaison entre Flutter et React Native

Dans l'édition 2023 de l'enquête Stack Overflow, Flutter est toujours en tête de React Native, selon les développeurs (tous les répondants) :

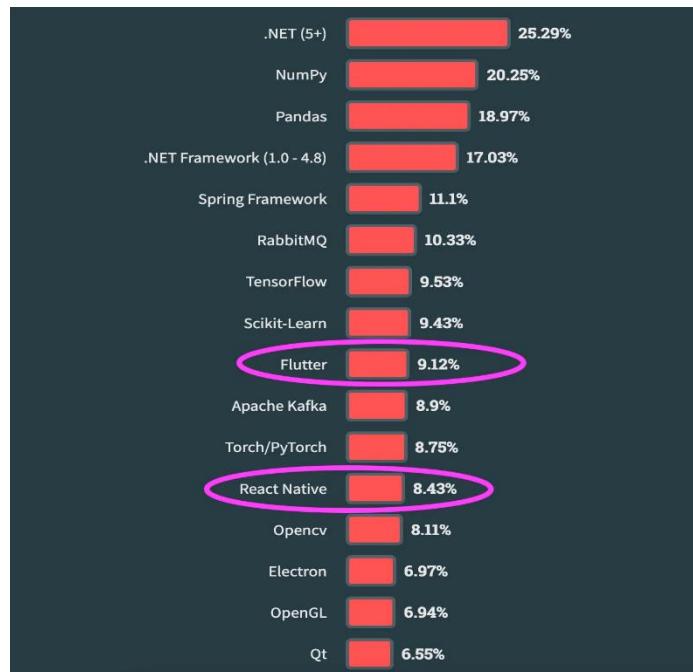


Figure 37 : Comparaison entre les langages de programmation (Frontend)

D'après les comparaisons et en lien avec nos besoins pour l'application mobile, le meilleur choix est Flutter en raison de sa rapidité, de sa compatibilité pour rendre l'application simple

d'utilisation, ainsi que de sa documentation complète et claire.

- **Langage de programmation : Frontend web - Angular**

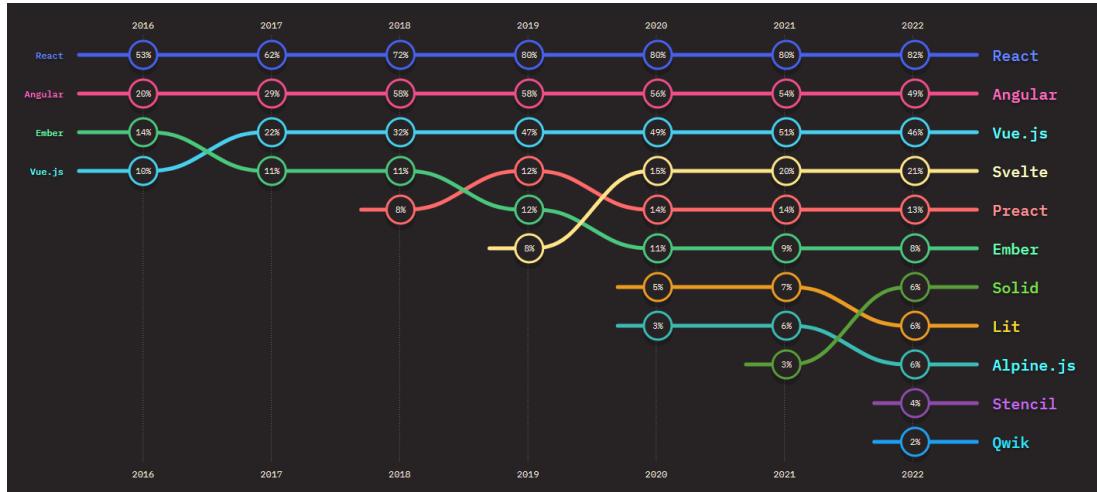


Figure 38 : Comparaison entre les langages de programmation (Frontend)

D'après les analyses de Stackdiary, les deux meilleurs langages de programmation pour la partie frontend sont React et Angular. Notre choix cette fois est basé sur l'équipe : certains membres maîtrisent Angular, c'est pourquoi nous l'avons choisi pour permettre aux autres de développer leurs compétences.

- **Langage de programmation : Backend – Spring boot**

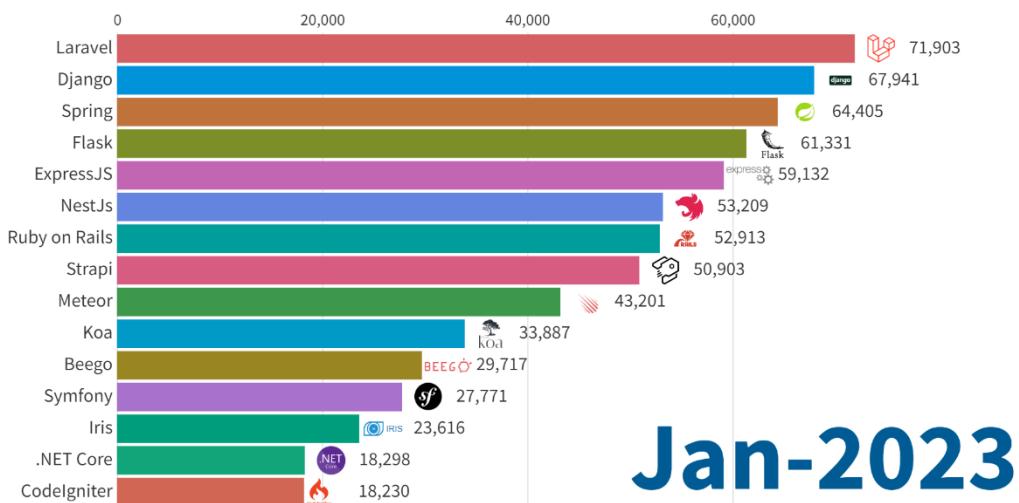


Figure 39 : Comparaison entre les Frameworks

D'après les analyses de Statisticsanddata, les trois meilleurs frameworks de développement sont Laravel, Django et Spring.

Laravel : Framework PHP moderne et expressif pour le développement web rapide avec une sécurité intégrée et un écosystème robuste de packages.

Django : Framework web Python qui favorise la productivité grâce à une configuration par défaut complète, idéal pour les applications web complexes et les systèmes de gestion de contenu.

Spring Framework : Framework d'application Java offrant une intégration IoC puissante, avec des modules comme Spring MVC pour le web, Spring Data pour l'accès aux données, et Spring Security pour la gestion de la sécurité.

Notre projet est basé sur une architecture microservices, ce qui fait la différence entre les trois options. Le meilleur choix est le framework Spring, car il est généralement considéré comme le plus adapté pour l'architecture de microservices en raison de :

- **Modularité et extensibilité** : Spring Framework offre une architecture modulaire qui permet de découpler et de développer des microservices indépendamment les uns des autres.
- **Support pour Spring Boot** : Spring Boot simplifie le développement, la configuration et le déploiement des microservices, ce qui est essentiel dans une architecture basée sur les services.
- **Spring Cloud** : Fournit des outils et des bibliothèques pour la création de systèmes distribués, la gestion de la configuration, la découverte de services, et plus encore.
- **Sécurité** : Spring Security offre des fonctionnalités robustes pour sécuriser les microservices et gérer les autorisations.

D'un autre côté, Django et Laravel sont davantage orientés vers les monolithes ou les applications web traditionnelles. Bien qu'ils puissent être utilisés pour des microservices, ils ne sont pas aussi optimisés que Spring pour cette architecture spécifique.

- **Conteneurisation : Docker**

La conteneurisation est un concept de plus en plus populaire dans le monde du développement logiciel. Bien plus léger que les traditionnelles machines virtuelles, cette technique permet de regrouper et de gérer les applications et leurs dépendances dans des conteneurs séparés et isolés les uns des autres.

Voici un comparatif des différences entre les machines virtuelles et les conteneurs :

Machine Virtuelle (VM)	Conteneur (Image Docker)
Lourde	Léger
Performance limitée	Performance native
Temps de démarrage en minutes	Temps de démarrage en millisecondes
Nécessite beaucoup de mémoire	Nécessite moins de mémoire

Tableau 19: Comparaison entre Machine Virtuelle et Image Docker

De plus, en stockant notre image Docker dans Docker Hub, cela simplifie le déploiement et la gestion des conteneurs. Donc Docker reste le meilleur choix.

- **Service de base de données : MySQL**

D'après les analyses de Appinventiv, MySQL est l'une des bases de données les plus populaires à utiliser dans le monde informatique.

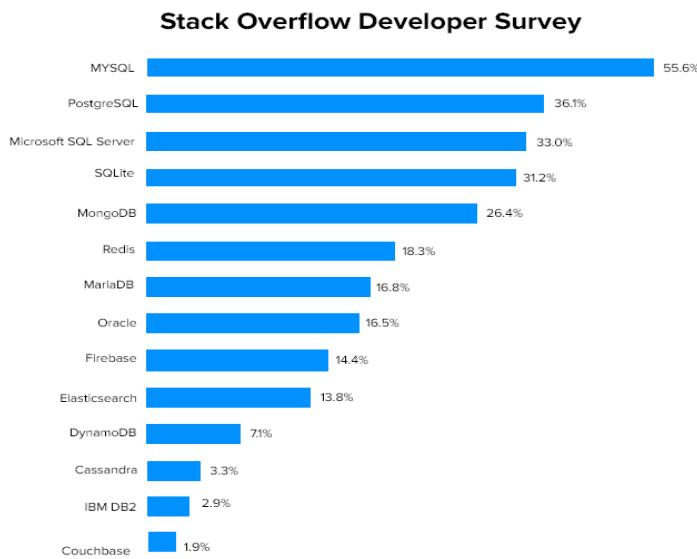


Figure 40 : Comparaison entre les Bases de données

- **Déploiement : Azure VPS**

Pour le choix de l'outil de déploiement, l'entreprise utilise déjà un compte Azure, donc nous restons dans le même cloud pour le déploiement de notre code backend.

2.3.4 Architecture :

L'architecture microservices est une approche de développement logiciel où une application est structurée en une collection de services indépendants et faiblement couplés. Chaque service est conçu pour accomplir une fonction métier spécifique et peut être développé, déployé et mis à l'échelle indépendamment des autres. Voici quelques caractéristiques clés des architectures microservices :

- **Modularité** : Chaque service est un module autonome, ce qui facilite le développement, la maintenance et l'évolution de l'application.
- **Indépendance** : Les microservices sont indépendants les uns des autres, ce qui permet de déployer et de mettre à jour un service sans affecter les autres.
- **Technologies diversifiées** : Chaque microservice peut être développé avec des technologies et des langages différents, en fonction des besoins spécifiques du service.
- **Communication par API** : Les microservices communiquent entre eux via des API, souvent en utilisant des protocoles HTTP/REST ou des messages de file d'attente.
- **Scalabilité** : Les microservices permettent une mise à l'échelle granulaire, où seul le service nécessitant plus de ressources est mis à l'échelle, plutôt que l'application entière.
- **Déploiement continu** : Grâce à l'indépendance des services, les équipes peuvent adopter

des pratiques de déploiement continu, améliorant ainsi l'agilité et la rapidité des mises à jour.

Avantages des architectures microservices :

- **Flexibilité de développement** : Les équipes peuvent travailler sur différents services en parallèle, ce qui accélère le cycle de développement.
- **Maintenance facilitée** : La modification ou la mise à jour d'un microservice est plus simple car elle n'impacte pas directement les autres services.
- **Adaptation aux nouvelles technologies** : Les nouveaux services peuvent être créés en utilisant les technologies les plus récentes sans affecter les anciens services.

Après le choix des outils et des technologies utilisés pour le développement, nous présenterons ensuite l'architecture de notre projet.

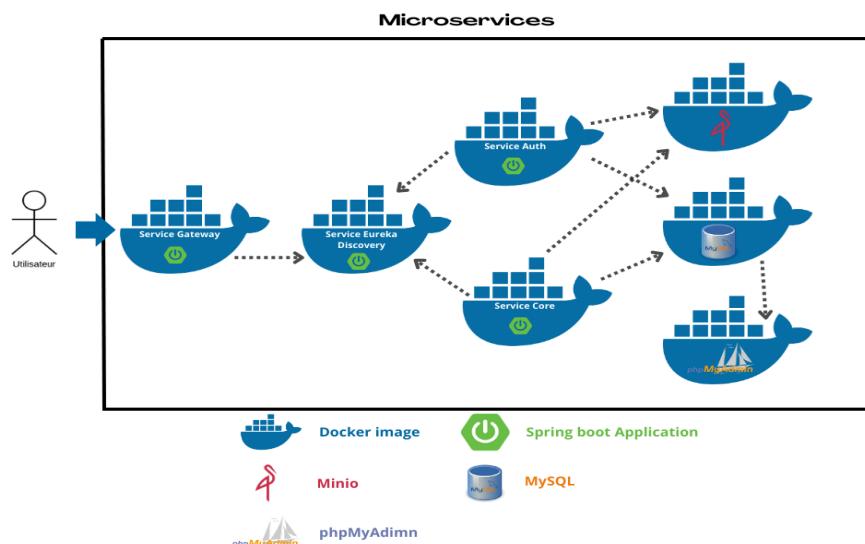


Figure 41 : Architecture Microservices du projet

En bas de l'architecture, on trouve cinq éléments clés. Au niveau de l'architecture, nous avons quatre services (quatre applications Spring Boot), chaque service étant déployé dans une image Docker. À droite, nous avons trois images Docker :

- **Minio** pour le stockage des images.
- **MySQL** comme système de gestion de bases de données relationnelles.
- **PhpMyAdmin** une application Web de gestion pour les systèmes de gestion de base de données MySQL.

L'utilisateur utilise le service Gateway pour accéder aux deux autres services, Auth et Core. Le rôle du Gateway est d'acheminer les requêtes vers le service correspondant. Le service Eureka Discovery permet aux services de se découvrir et de communiquer sans avoir besoin de

connaître les adresses exactes des autres services.

L'utilisateur Les relations existantes sont les suivantes :

- Les services Auth et Core utilisent Minio et MySQL.
- MySQL est en relation avec phpMyAdmin.
- Le service Eureka Discovery est en relation avec les services Auth et Core pour stocker leurs adresses.
- Le service Gateway est en relation avec le service Eureka Discovery pour connaître l'adresse des services.

Conclusion

Dans les chapitres précédents, nous avons d'abord détaillé les besoins fonctionnels et non fonctionnels du projet. Ensuite, nous avons élaboré différents diagrammes tels que les diagrammes de classes, de cas d'utilisation, d'activités et de séquence pour décrire les divers services du système. Par la suite, nous avons progressé vers la conception des APIs qui serviront à interconnecter les différentes parties de l'application.

Toutes ces étapes ont jeté les bases nécessaires pour le chapitre actuel, où nous nous concentrerons sur le choix des technologies, des outils et sur l'architecture du projet. L'objectif est de garantir que le projet soit réalisé selon les normes de qualité élevées et en utilisant les meilleures pratiques disponibles.

Chapitre 4

Réalisation et implémentation du projet

Après avoir entamé les phases précédentes qui ont inclus la spécification, l'analyse, la conception, l'étude technique et le choix des outils et des technologies, nous abordons dans ce chapitre la réalisation. Cette étape consiste à concrétiser le projet, et nous présenterons finalement quelques captures d'écran.

4. Réalisation et implémentation du projet

Ce chapitre présente les interfaces réalisées au niveau de l'application mobile et des deux applications web, les tests des APIs, ainsi qu'un test de qualité de code.

4.1 Aperçu de structure et des interfaces développées

Durant cette partie on va représenter des captures d'écrans du structure partie Backend avec les interfaces de l'application mobile, web et des exemples de test.

4.1.1 La structure du partie Backend

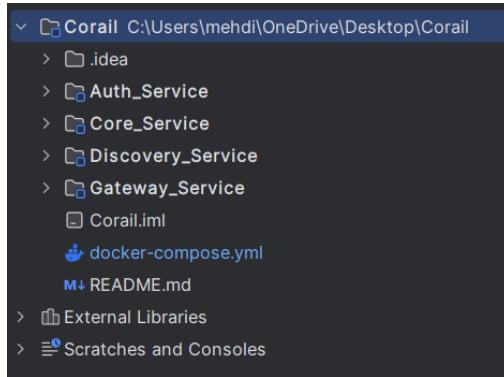


Figure 42 : la structure du projet – partie Backend

- **Corail** : représente le groupe des services.
- **Auth_Service** : le service de l'authentification.
- **Core_Service** : contient la plupart des fonctionnalité tel que : le scan, affichage des tickets, statistique, catégorie, marque et magasin.
- **Discovery_Service** : le service de découverte qui assure la communication efficace entre les microservices.
- **Gateway_Service** : Le service de passerelle qui permet de centraliser et simplifier l'accès aux microservices.

4.1.2 Capture d'écran du VPS

```
azreuser@CORAIL: ~
System load: 0.0          Processes:           136
Usage of /: 30.9% of 28.89GB  Users logged in: 0
Memory usage: 25%          IPv4 address for eth0: 10.0.0.5
Swap usage: 0%
* Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s just raised the bar for easy, resilient and secure K8s cluster deployment.
https://ubuntu.com/engage/secure-kubernetes-at-the-edge
Expanded Security Maintenance for Applications is not enabled.
0 updates can be applied immediately.
6 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm
New release '22.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

*** System restart required ***
Last login: Fri Jun 28 14:06:22 2024 from 105.71.4.203
azreuser@CORAIL:~
```

Figure 43 : VPS – Déploiement

Nous utilisons un VPS Azure pour le déploiement du code backend. Nous avons installé Docker et Docker Compose sur ce VPS, puis transféré le fichier Docker Compose de l'ordinateur vers ce VPS, et ensuite exécuté ce fichier.

4.1.3 Capture d'écran du documentation Swagger

The screenshot shows the Swagger UI for the 'personne-controller'. It lists 14 endpoints:

- POST /auth/user-verify
- POST /auth/parrainer
- POST /auth/mdp_publie
- POST /auth/logout
- POST /auth/login_client
- POST /auth/login
- POST /auth/addImage
- POST /auth/UpdateUser
- POST /auth/UpdatePass
- POST /auth/SousRetail-verify
- POST /auth/Ajouter_Role
- POST /auth/Ajouter_Personne
- GET /auth/user
- GET /auth/roles

Figure 44 : Documentation des APIs Service Auth– Swagger

La documentation présente 14 endpoints, 12 pour la méthode POST et 2 pour la méthode GET.

The screenshot shows the Swagger UI for the 'ticket-controller' and 'produit-controller'. It lists 12 endpoints:

- POST /core/upload
- POST /core/data/{id}
- POST /core/NotNotif
- GET /core/Tickets
- GET /core/Tickets-Filter
- GET /core/Tickets-Brand
- GET /core/TicketId
- GET /core/CountClient
- GET /core/Checkednotif
- GET /core/AllNotif
- produit-controller
- POST /core/AjouterMarque
- POST /core/AjouterCategorie

Figure 45 : Documentation des APIs Service Core– Swagger

La documentation présente 10 endpoints pour le ticket-controller : 3 pour la méthode POST et 7 pour la méthode GET. Pour le produit-controller, il y a 2 endpoints pour la méthode POST.

4.1.4 Capture d'écran des Tests APIs Postman

The screenshot shows a Postman API test for the 'CORE-SERVICE/core/upload' endpoint. The 'Body' tab is selected, showing a 'form-data' key named 'file' with a value of 'ticket1.json'. The 'Headers' tab shows 'Content-Type: application/json'. The 'Test Results' tab displays a QR code generated by the API.

Figure 46 : Test de l'API de génération de QR Code– Service CORE

L'endpoint permet de transformer le fichier JSON contenant le ticket du client en un QR Code à scanner.

```

1  {
2   "logo_marque": "nike.jpg",
3   "Nom_marque": ,
4   "ice_marque": "2",
5   "website_marque": "www.nike.ma",
6   "date_ticket": "24/04/2024",
7   "time_ticket": "16:18:26",
8   "ticket_number": "12",
9   "produits": [
10    {
11      "category": 2,
12      "reference": "TR",
13      "description": "Tee",
14      "quantity": 1,
15      "prix": 20,
16      "prix_total": 10,
17      "HT": 52.3,
18      "TVA": 10.46,
19      "TTC": 62.76
20    }
21  ],
22  "total_HT": "220.00",
23  "TVA": "43.00",
24  "totale_TTC": "264.00",
25  "magasin_address": "hay oulfa bd 45",
26  "magasin_phone_number": "05 38 90 04 61 - 65"
27 }

```

Figure 47 : Fichier JSON du ticket

Le fichier JSON contient les informations des produits achetés par le client, ainsi que la date et les informations du magasin.

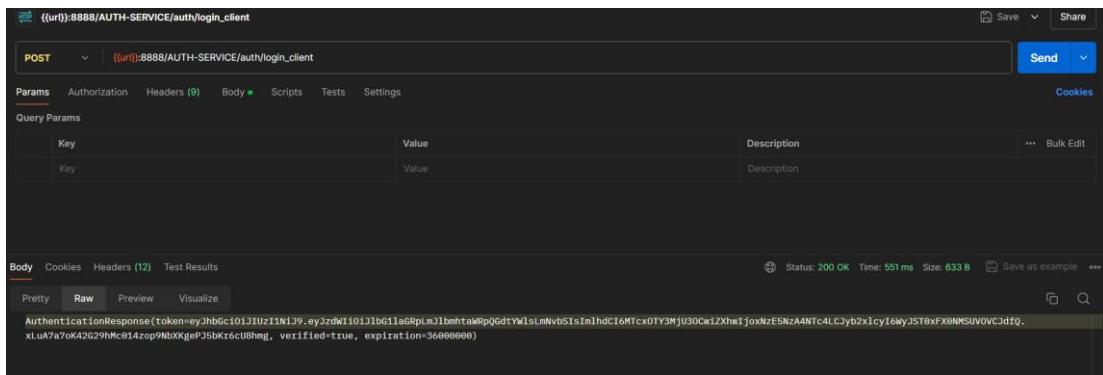


Figure 48 : L'authentification – Application mobile

L'endpoint permet à l'utilisateur de saisir l'email et le mot de passe pour accéder à l'application mobile. Comme vous pouvez le voir dans la réponse, elle contient le token permettant à l'utilisateur de naviguer dans l'application, la vérification si le compte est vérifié ou non, et le temps d'expiration.

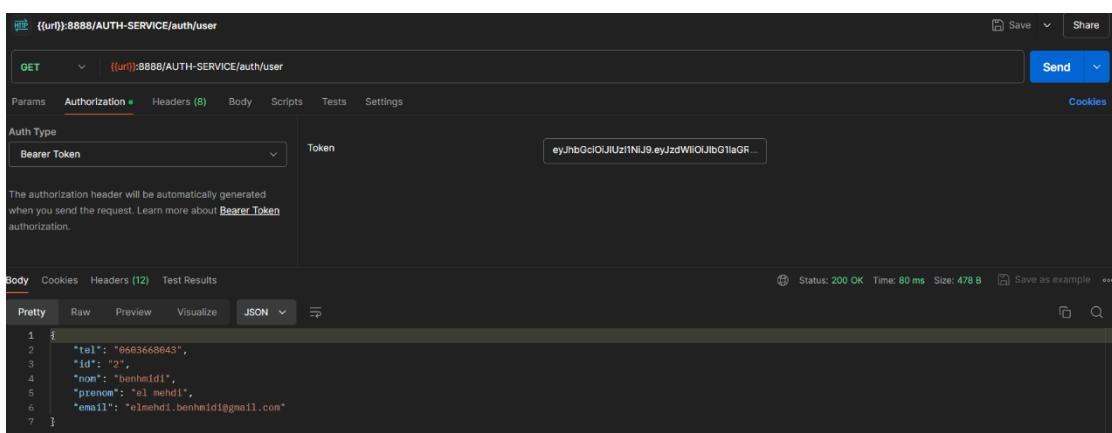


Figure 49 : Les informations de l'utilisateur connecté– Application mobile et web

L'endpoint permet d'afficher les informations de l'utilisateur connecté telles que le

téléphone, l'identifiant, le nom, le prénom et l'email.

Figure 50 : Les informations de l'utilisateur connecté– Application mobile et web

L'endpoint permet à l'utilisateur d'ajouter une image à son profil après s'être authentifié. L'utilisateur choisit l'image et l'email est envoyé dans l'en-tête. Un message de succès s'affiche après l'ajout.

Figure 51 : Les informations des tickets– Application mobile et web

L'endpoint permet de retourner les informations du ticket de l'utilisateur connecté, telles que la marque, le montant, la date, etc.

Figure 52 : La déconnexion – Application mobile et web

L'endpoint permet à l'utilisateur de quitter l'application et d'expirer son propre token, dans le cadre des mesures de sécurité.

4.1.5 Capture d'écran de l'outil de stockage des images

The screenshot shows the MinIO Object Store interface. On the left, there's a sidebar with sections for User (Object Browser, Access Keys, Documentation), Administrator (Buckets, Policies, Identity, Monitoring, Events, Tiering, Site Replication, Configuration), and Subnet. The main area is titled 'Object Browser' with a search bar 'Filter Buckets'. It lists a single bucket named 'corail' with 2 objects, a size of 39.7 kB, and R/W access.

Figure 53 : La déconnexion – Application mobile et web

Au lieu de stocker les images des marques et des utilisateurs dans le projet, nous utilisons l'outil MinIO qui simplifie le stockage. Nous utilisons une clé privée pour l'accès avec un bucket nommé "corail", qui contient à l'intérieur un dossier pour stocker les images. Au niveau de la base de données, nous stockons uniquement le chemin vers l'image.

4.1.6 Les interfaces réalisées pour l'application mobile



Figure 54 : Interfaces d'accès et de présentations

Les 4 interfaces présentent l'accès à l'application après l'ouverture, avec une présentation de l'application dans les 3 dernières interfaces.

The figure consists of four screenshots of a mobile application interface titled "Créer un compte".

- Screenshot 1 (14:05):** Shows the initial form with empty fields for Nom, Prenom, Date de naissance, Civilité, Ville, and a password section.
- Screenshot 2 (14:05):** Shows the form partially filled with "mehdi" in Nom and "benhmid" in Prenom.
- Screenshot 3 (14:11):** Shows the form fully filled with "mehdi" in Nom, "benhmid" in Prenom, "31/01/2001" in Date de naissance, "Monsieur" in Civilité, and "Casablanca" in Ville.
- Screenshot 4 (14:07):** Shows the final step where "Sauvegarder" (Save) is highlighted. The password field contains "*****" and the confirmation field also contains "*****". A note at the bottom right states: "Mentions obligatoires: Gorail collecte et traite vos données conformément à la réglementation applicable en matière de protection des données personnelles. Pour en savoir plus, consultez notre charte des données personnelles."

Figure 55 : Interfaces d'inscription

Les 4 interfaces sont pour l'inscription, et il est obligatoire pour l'utilisateur de remplir tous les champs pour s'authentifier correctement. Les champs comprennent : nom, prénom, date de naissance, civilité, ville, email, téléphone, mot de passe avec confirmation.

Après que l'utilisateur clique sur sauvegarder, s'il existe déjà un compte, un message en rouge s'affiche dans l'application pour indiquer que le compte existe déjà. Sinon, l'inscription se termine et l'interface suivante s'affiche.

The figure consists of two screenshots of a mobile application interface titled "Créer un compte".

- Screenshot 1 (18:04):** Shows an error message "Utilisateur déjà existant" above the form. The form fields are identical to Figure 55, but the "Sauvegarder" button is disabled.
- Screenshot 2 (14:13):** Shows a modal dialog titled "Verification code" asking for a 4-digit OTP sent via email. The email address "mehdibenhmidi51@gmail.com" is listed. Below the modal is a "Se connecter à mon compte" (Connect to my account) button.

Figure 56 : Message d'erreur + Code de vérification

L'utilisateur reçoit un code de 4 chiffres dans son email. Il doit le saisir ici pour terminer l'inscription. Si le code est correct, l'application redirige vers la page d'authentification. Sinon, un message d'erreur s'affiche pour demander de saisir le code correctement.

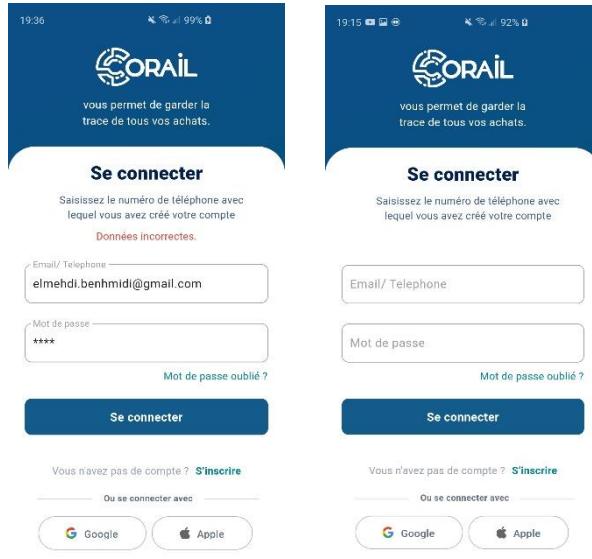


Figure 57 : Interface d'authentification

L'interface d'authentification contient deux champs : un pour l'email/téléphone et l'autre pour le mot de passe, afin d'accéder à l'application. Après avoir rempli ces champs et cliqué sur le bouton "Se connecter", si l'utilisateur saisit un email/téléphone ou un mot de passe incorrect, un message d'erreur s'affiche. L'utilisateur peut également se connecter avec son compte Google ou via l'App Store. En cas de perte du mot de passe, il y a un bouton "Mot de passe oublié" pour le récupérer.

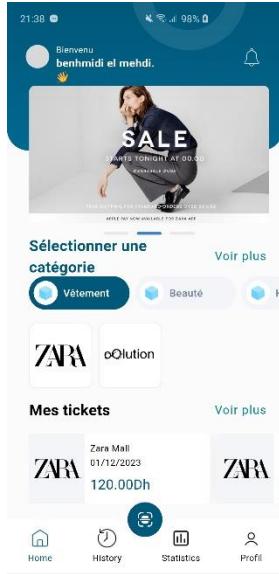


Figure 58 : Page d'accueil

L'interface d'accueil, la première page affichée après l'authentification, contient les éléments suivants :

- Un texte de bienvenu avec le nom et le prénom de l'utilisateur.
- Un bouton pour afficher les notifications.
- Une barre pour consulter les offres disponibles.

- Les catégories avec un bouton "Voir plus".
- Les tickets avec un bouton "Voir plus".
- Navigation dans la barre inférieure.

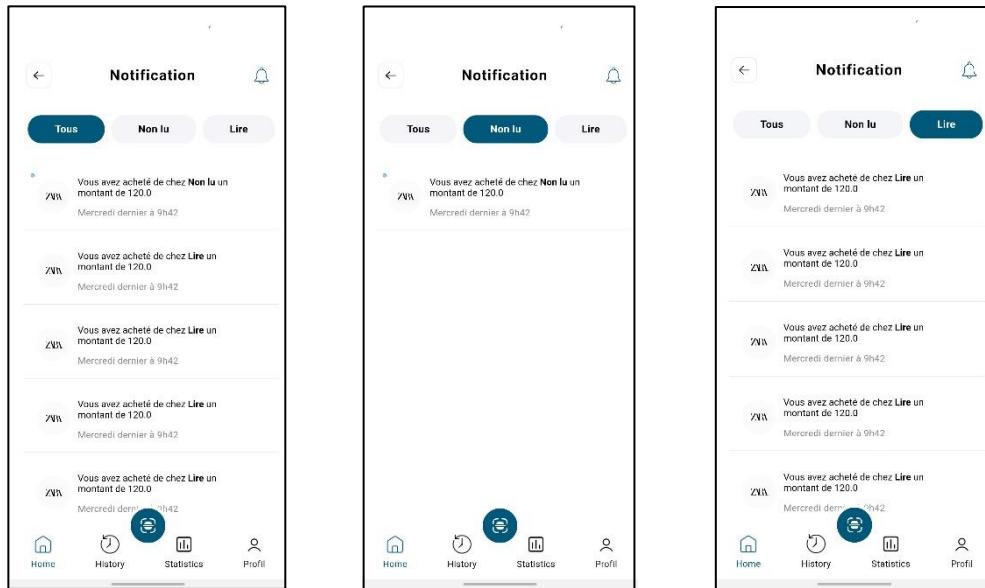


Figure 59 : Interface des notifications

L'interface des notifications est divisée en 3 parties : Toutes, Non lues, et Lues, chacune ayant sa propre interface dédiée. Les notifications non lues sont affichées avec un petit bouton bleu à côté de l'image de la marque. Les notifications sont envoyées automatiquement après que le client ait scanné le ticket.

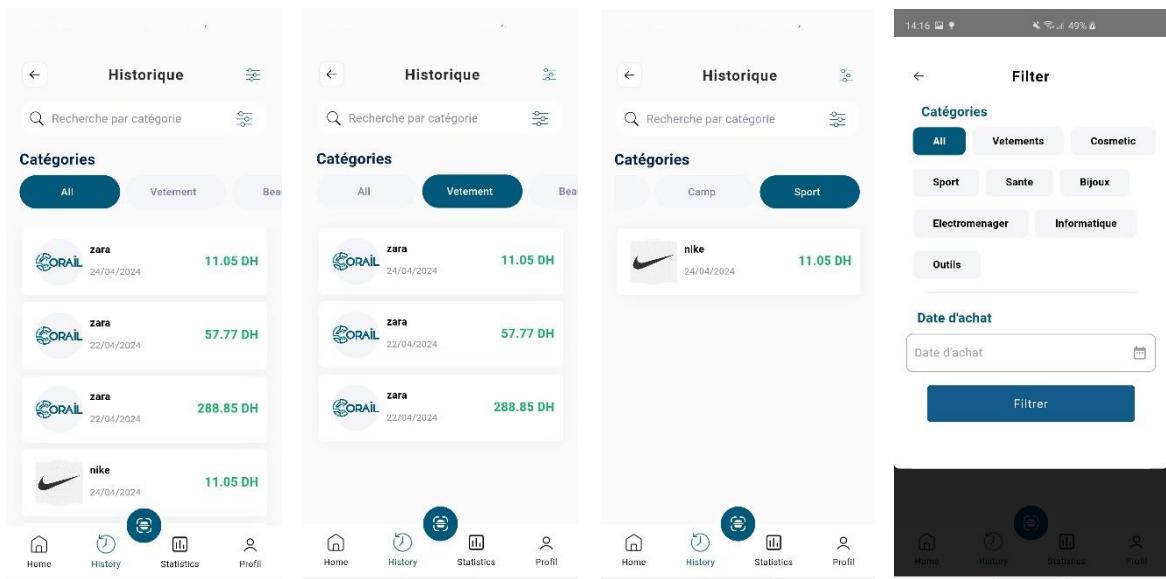


Figure 60 : Interface historique + filtrage

L'interface historique des tickets contient les tickets scannés par l'utilisateur avec le nom et l'image de la marque, la date du ticket et le prix total. Il y a une barre de recherche et un bouton pour le filtrage. Ce dernier se fait par la catégorie et la date d'achat. En bas de la barre de recherche, il y a des boutons contenant les noms des catégories, permettant un filtrage en temps réel.

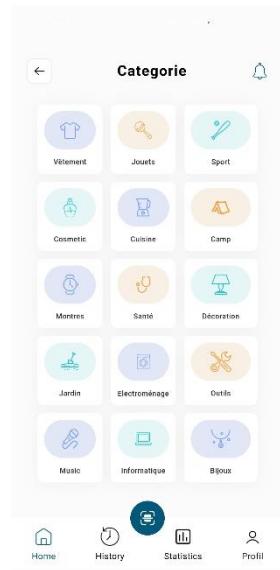


Figure 61 : Interface des catégories

L'interface permet d'afficher les catégories disponibles, parmi lesquelles on trouve 15 catégories. Il y a la possibilité de naviguer vers les pages d'historique, de statistiques ou de profil, ainsi que d'afficher l'interface des notifications.

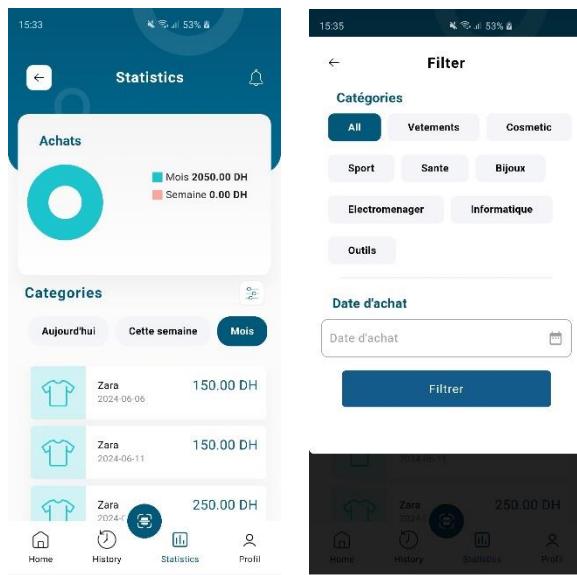


Figure 62 : Interface des statistiques

L'interface statistique affiche les analyses d'achat de l'utilisateur dans un diagramme circulaire pour la somme des mois et des semaines. En bas, il y a un filtrage par jour, semaine et mois, ainsi qu'un filtrage avancé par catégorie et date d'achat.

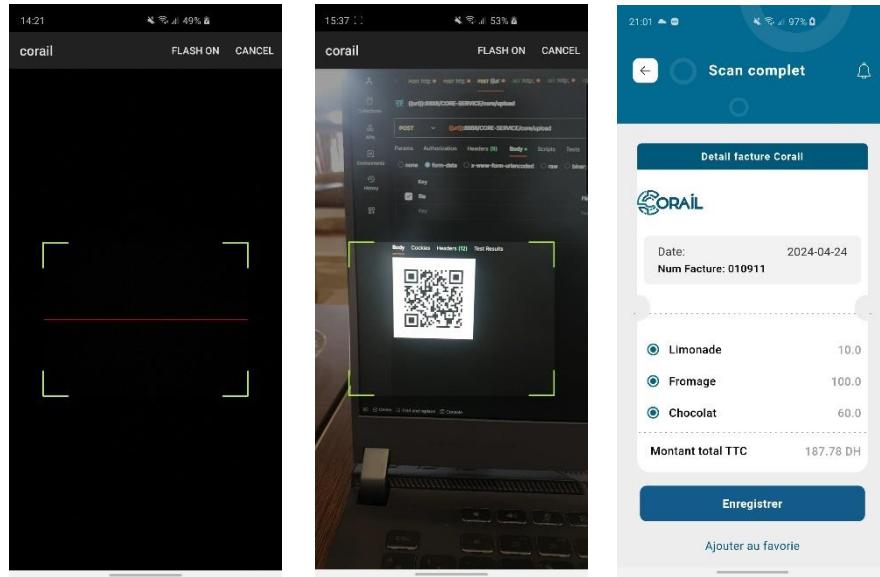


Figure 63 : Interfaces de scan

L'interface de scan présente l'une des fonctionnalités essentielles de l'application. L'utilisateur clique sur le bouton "Scan" pour activer la caméra, scanner le QR Code, puis recevoir son ticket qui contient l'image et le nom de la marque, ainsi que les produits avec les prix correspondants.

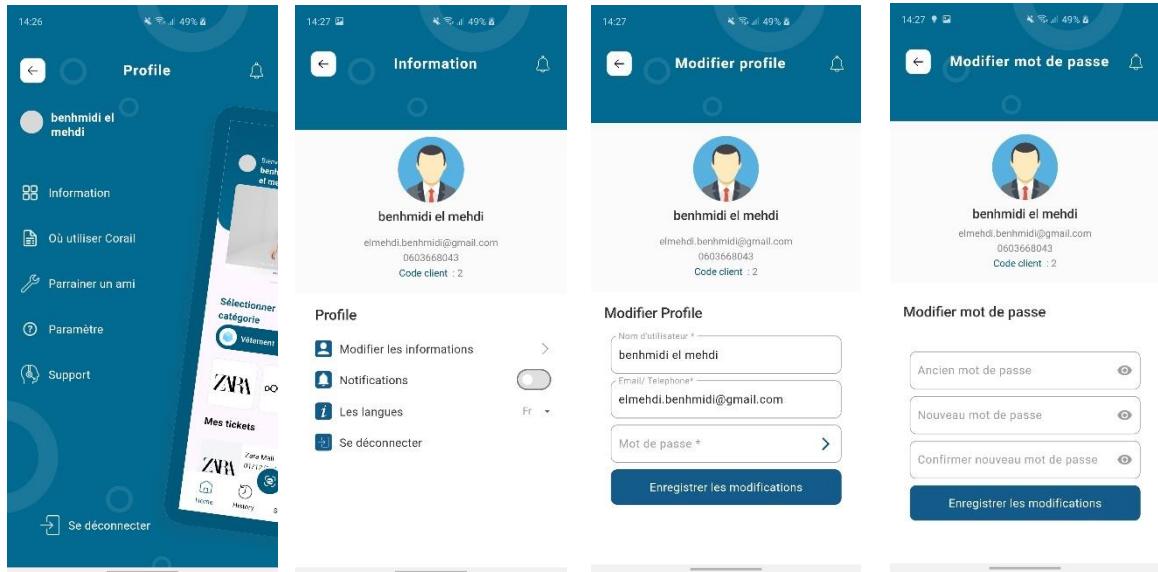


Figure 64 : Interfaces du profile

La première interface présente un tiroir de navigation avec 5 sections : Information, Où utiliser Corail, Parrainer, Paramètre et Support, avec un bouton pour la déconnexion.

Les 3 autres interfaces permettent la modification des informations de l'utilisateur telles que le nom et prénom, l'email, le mot de passe et l'image. Elles offrent la possibilité d'activer ou désactiver les notifications, de changer la langue et de se déconnecter une autre fois.

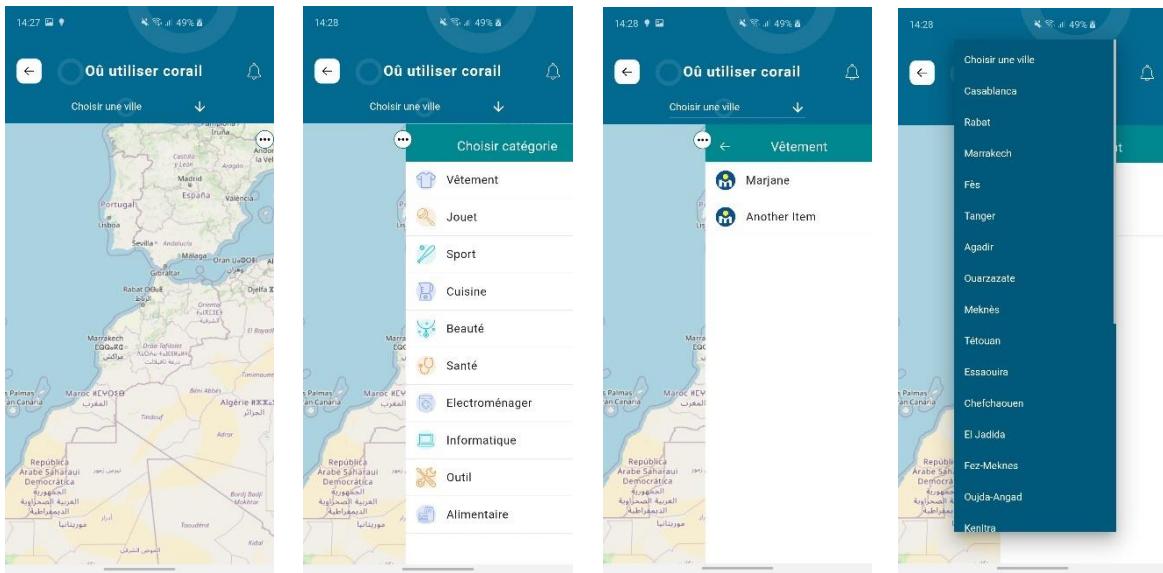


Figure 65 : Interfaces Où utiliser Corail

L'interface "Où utiliser Corail" contient une carte pour afficher la localisation des magasins qui supportent l'application Corail pour scanner le ticket. On peut sélectionner le magasin en filtrant par catégorie et ville.



Figure 66 : Interface parrainer un ami

L'interface "Parrainer un ami" permet à l'utilisateur d'inviter un ami en saisissant son email. Ensuite, le système envoie automatiquement un email à l'adresse saisie pour l'inviter.

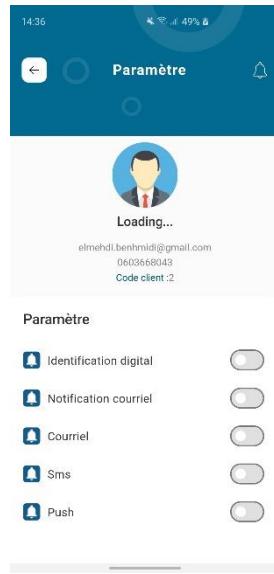


Figure 67: Interface paramètre

L'interface Paramètre permet de régler quelques fonctionnalités au niveau de l'application, comme l'activation et la désactivation de l'empreinte digitale, des notifications par courriel ou SMS.



Figure 68 : Interface support

L'interface Support permet de simplifier le contact entre l'utilisateur et le support de l'application.

4.1.7 Les interfaces réalisées pour l'application web

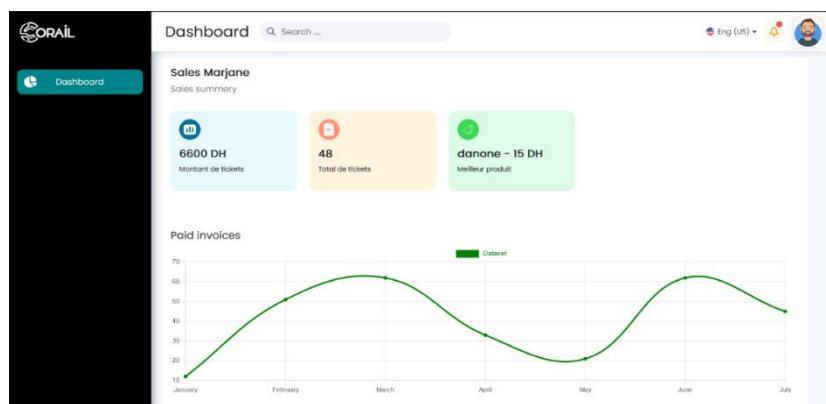


Figure 69 : Tableau de bord – Administrateur marque

L'interface tableau de bord de l'administrateur de marque est conçue pour afficher les statistiques de la marque, notamment le montant des tickets, leur nombre, ainsi que le meilleur produit vendu. Elle comprend également un graphique pour visualiser le total des ventes de chaque mois.

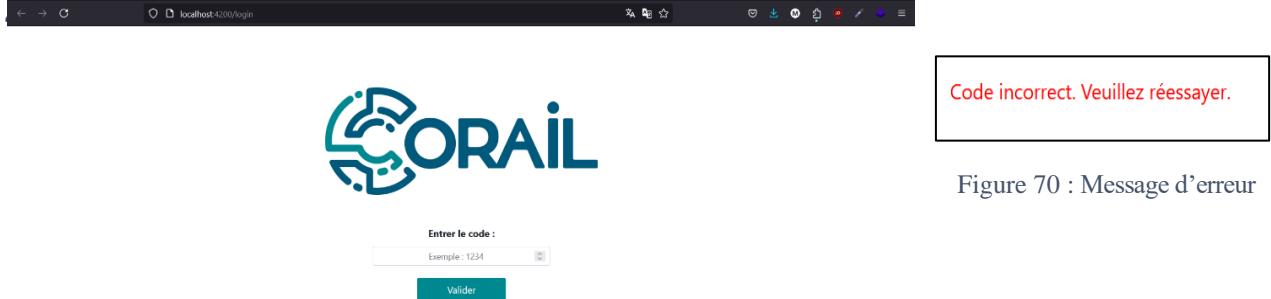


Figure 70 : Message d'erreur

Figure 71 : L'authentification– application web magasin

L'interface d'authentification du magasin contient un champ pour saisir le code reçu de la part de l'administrateur de la marque. Si le code est correct, la page du QR Code s'affiche ; sinon, un message d'erreur s'affiche.

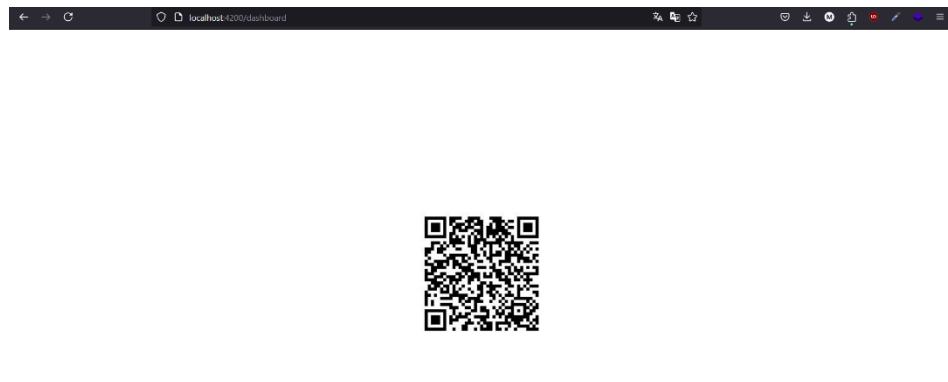


Figure 72 : Page de QR Code– application web magasin

L'interface du QR Code s'affiche après l'authentification dans le but d'afficher le QR après l'achat d'un client.

4.1.8 Capture d'écran du test de qualité de code

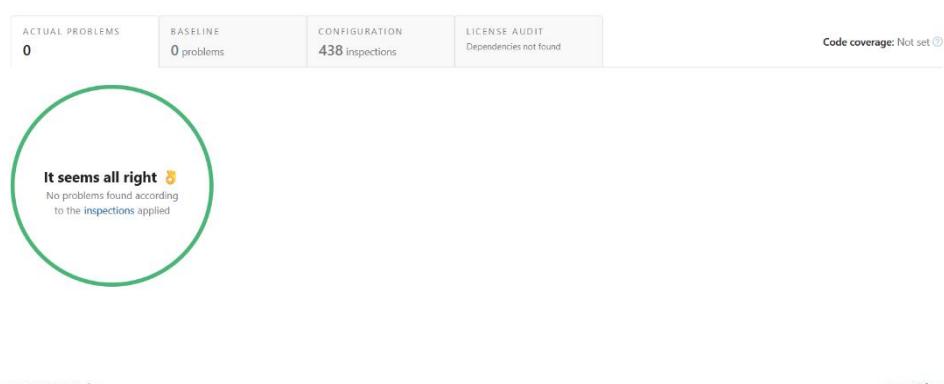


Figure 73: Test de qualité de code avec Qodana

Le test de Qodana mentionne "Tout semble aller bien 🌟 . Aucun problème constaté selon les inspections appliquées." Cela signifie que, d'après les analyses effectuées par Qodana, aucun problème significatif n'a été identifié dans le code selon les critères d'inspection configurés. C'est une indication positive que le code semble être de bonne qualité, sans erreurs majeures ou violations de bonnes pratiques détectées par l'outil.

4.2 Problématiques rencontrés

En tant que stagiaire, je ne maîtrise pas tout parfaitement, ce qui fait que je rencontre des difficultés de temps en temps. Parmi les problématiques que j'ai rencontrées :

Pour le front :

- **Animation au niveau de Flutter :** Il y a pas mal d'interfaces qui contiennent des animations lors du changement de page, ce qui peut être complexe à gérer.
- **Problématique du responsive :** L'un des points essentiels en développement mobile est de rendre l'application adaptative à n'importe quel écran. Cela représente également un défi pour moi.
- **Version APK :** Lors du passage de l'application en version APK (version installable), j'ai rencontré un problème où certaines pages apparaissent correctement en mode débogage, mais deviennent grises en version APK.

Pour le backend :

- **Sécurité entre les services :** J'ai trouvé des difficultés à lier correctement la sécurité entre les différents services.

4.3 Solutions appliquées

Bien sûr, pour chaque problème, il y a une solution, et voici les solutions que j'ai appliquées :

Pour le frontend :

Pour l'animation et le problème du responsive, j'ai suivi des vidéos pour apprendre les règles. Pour le problème de la version APK, j'ai effectué des recherches sur Google et découvert que beaucoup de personnes rencontrent ce problème, bien que chacun ait des causes différentes. Cela m'a permis de vérifier chaque partie du code pour détecter les problèmes, et j'ai finalement réussi.

Pour le backend :

Concernant la problématique de la sécurité entre les services, j'ai trouvé une solution dans

une vidéo de **M. Youssfi** où il explique verbalement à ses étudiants. J'ai appliqué cette solution, qui fonctionne bien.

Grâce à ces problématiques et aux efforts déployés pour les résoudre, j'ai acquis beaucoup d'expérience qui m'aidera dans le futur.

Conclusion

Dans ce chapitre, nous avons présenté le fruit du travail réalisé sous forme de captures d'écran de l'application développée. Nous avons expliqué les fonctionnalités de chaque écran de notre projet, étant donné que le stage n'est pas encore terminé.

Conclusion générale et perspective

La digitalisation des tickets représente une avancée significative dans la gestion des services, offrant des solutions plus efficaces et modernes par rapport aux méthodes traditionnelles. Ce projet a permis de démontrer comment une architecture basée sur des microservices, couplée avec des outils et technologies robustes, peut répondre aux besoins de gestion et de traitement des tickets de manière plus fluide et scalable.

Nous avons exploré en détail l'architecture du système, composée de plusieurs services distincts déployés dans des conteneurs Docker. Cette approche permet une modularité et une flexibilité accrues, facilitant la maintenance et les mises à jour des composants individuels sans perturber l'ensemble du système. L'utilisation de Minio pour le stockage des images, MySQL pour la gestion des bases de données relationnelles, et phpMyAdmin pour l'administration de MySQL, assure une infrastructure fiable et performante.

Le service Gateway joue un rôle crucial en acheminant les requêtes des utilisateurs vers les services appropriés, tandis que le service Eureka Discovery permet une communication fluide et dynamique entre les services, simplifiant ainsi la gestion des adresses et des points d'entrée des différents services.

Dans le dernier chapitre, nous avons exposé le fruit de travail réalisé sous forme des prises d'écrans de l'application développée, en expliquant les fonctionnalités de chaque écran de notre projet puisque le stage n'est pas encore terminé.

L'application mobile et les sites web développés jusqu'à présent sont en première version, conçus pour tester en conditions réelles et identifier les problèmes rencontrés. La deuxième version est déjà prête et sera lancée une fois les dernières étapes finalisées. Par la suite, nous commencerons la réalisation de cette version, en intégrant les modifications nécessaires.

Pour la scalabilité, nous utiliserons des solutions horizontales pour gérer l'augmentation du nombre d'utilisateurs. L'intelligence artificielle sera intégrée, notamment pour le scan des tickets, apportant des solutions innovantes. L'interopérabilité sera facilitée par des APIs ouvertes, et l'analytique avancée fournira des insights précieux pour optimiser les ressources et prévoir les besoins futurs. Ces perspectives permettront au projet de s'adapter aux évolutions futures, assurant une gestion optimisée et durable des tickets.

Bibliographie

- Flutter : <https://docs.flutter.dev/>
- Angular : <https://angular.dev/overview>
- Spring boot : <https://docs.spring.io/spring-boot/documentation.html>
- Spring cloud : <https://spring.io/projects/spring-cloud>
- Spring security : <https://docs.spring.io/spring-security/reference/index.html>
- Minio: <https://min.io/docs/kes/>
- Kanban : <https://asana.com/fr/resources/what-is-kanban#:~:text=Kanban%20est%20un%20outil%20de,selon%20la%20disponibilit%C3%A9%20de%20chacun.>
- Jira : <https://confluence.atlassian.com/jira>
- Comparaison frameworks pour microservices : [10 popular REST frameworks For your MicroService - DEV Community](#) , [Best Microservices Frameworks for Development \(back4app.com\)](#) , [6 Best Microservices Frameworks to Create Scalable Applications \(cmarix.com\)](#)
- Comparaison frameworks backend : [Top 10 Backend Frameworks in 2024: A Comprehensive Guide \(turing.com\)](#) , <https://statisticsanddata.org/data/most-popular-backend-frameworks-2012-2023/>
- Comparaison frameworks frontend : <https://www.imaginarycloud.com/blog/best-frontend-frameworks/>
- Comparaison docker et virtualisation : <https://www.bluesoft-group.com/la-conteneurisation-dans-le-developpement-et-le-deploiement/>