



Projet personnel encadré 2 :



DragN'Receive



Dev'iliser

CLEMENT Pierre de Dev'iliser

01/03/2022

Pour PROMEO - BTS Services informatiques aux organisations - Session 2022



1 - Définition de la situation professionnelle	3
1.1 - Présentation de l'entreprise cliente	3
1.1.1 - Généralités	3
1.1.2 - Environnement technique	4
1.2 - Présentation du prestataire de service	4
2 - Conception graphique	4
2.1 - Brief créatif	4
2.2 - Aperçu de la charte graphique	5
2.2.1 - Logos	5
2.2.2 - Palette de couleurs	6
2.2.3 - Typographies	6
3 - Cahier des charges fonctionnel	6
3.1 - Présentation générale	6
3.2 - Analyse du besoin	7
3.3 - Solutions envisagées	7
3.4 - Solution retenue	7
3.4.1 - Présentation des choix	7
3.4.2 - Benchmark	8
3.4.2.1 - Langage de programmation	8
3.4.2.2 - Framework	9
3.4.3 - Explication des choix	10
4 - Analyses et estimations	11
4.1 - Planification	11
4.2 - Budgétisation	11
5 - Conception fonctionnelle	12
5.1 - Dictionnaire de données	12
5.1.1 - Utilisateur	12
5.1.2 - Rôle	13
5.1.3 - Boutique	13
5.1.4 - Produit	14
5.1.5 - Commande	15
5.1.6 - Statut Commande	15
5.1.7 - Ticket	16
5.1.8 - Statut Ticket	16



5.1.9 - Message	17
5.1.10 - Web Token	17
5.1.11 - Mobile Code	18
5.2 - Exigences	19
5.3 - Cas d'utilisations	20
5.4 - Maquettes	21
5.4.1 - Connexion	21
5.4.2 - Profil	22
5.4.2 - Boutique	23
5.4.3 - Commande	24
5.4.4 - Produit	25
5.4.5 - Ticket	26
5.4.6 - Inscription	27
5.4.7 - Récupération de mot de passe	27
5.4.8 - Changement de mot de passe	28
6 - Cahier des charges technique	29
6.1 - Conception technique	29
6.1.1 - Séquence	29
6.1.2 - Spécificités des droits d'Administration des fonctionnalités	30
6.2 - Environnement technologique	32
6.2.1 - Hardware	32
6.2.2 - Software	32
6.3 - Modélisation de données	33
7 - Réalisation technique	34
7.1 - Les points forts du projet	34
7.2 - Les problèmes rencontrés	34
8 - Tests	35
8.1 - Tests unitaires	35
8.2 - Tests de validation	36
8.2.1 - Connexion	36
8.2.2 - Réinitialisation de mot de passe	38
9 - Indicateurs de performance	40
9.1 - Analyse de la planification mise à jour	40
9.2 - Analyse du coût global	41
10 - Conclusion	42



1 - Définition de la situation professionnelle

Le contexte défini ci-après est une situation fictive mais réaliste d'un projet d'entreprise visant à mettre en place une solution logicielle.

1.1 - Présentation de l'entreprise cliente

L'entreprise cliente, nommée DragN'Receive, est une société fictive de service Click & Collect implantée à Compiègne.

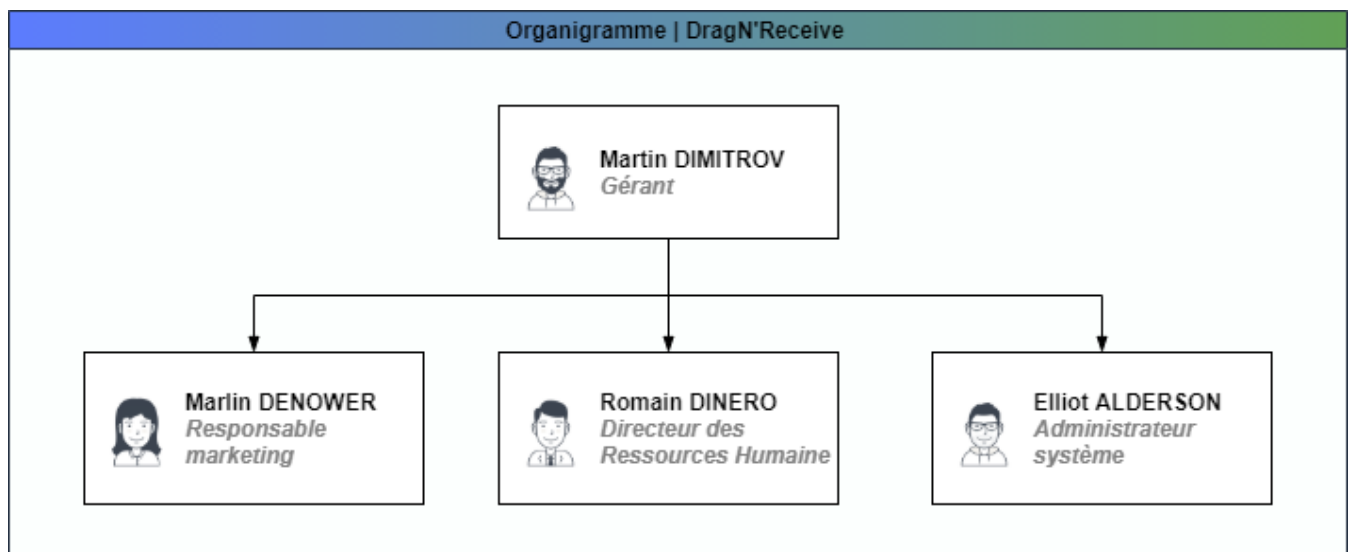
1.1.1 - Généralités

L'entreprise DragN'Receive est une SARL composée de 20 à 35 salariés, cette entreprise possède plusieurs contacts et partenaires dans les domaines de la restauration, de la distribution ainsi que des petits commerces.

La création de l'entreprise était d'abord à des fins opportunistes lors de la période de confinement puis s'est vu devenir un avantage primordial pour les personnes avec peu de disponibilités, les personnes à handicap et à mobilités réduites. La société se tourne également vers le développement durable avec des partenaires primant des produits possédant des impacts environnementaux faibles ainsi qu'un service de logistique verte, afin de réduire son impact sur l'environnement.

Leur capital social durant la période de 2020-2021 s'élève à 16 000 euros, pour un chiffre d'affaires de 245 000 euros.

Voici l'organigramme de l'entreprise DragN'Receive :





1.1.2 - Environnement technique

L'environnement technique de DragN'Receive est formé de plusieurs composants :

- Systèmes d'exploitation des appareils : Windows 10, Androids, IOS
- Serveurs : Dédiés
- Base de données : MySQL hébergé sur un serveur dédié
- Nombre d'utilisateurs simultanés potentiels : 500
- Site Web sous Vue (hébergé sur le serveur)
- API NodeJS (hébergé sur le serveur)

1.2 - Présentation du prestataire de service

Le prestataire pour ce projet sera l'entreprise Dev'iliser, étant une société spécialisée dans le développement de solution logicielle et applicative depuis plus de 16 années. Dans ce cadre d'activité, elle s'occupera du développement, suivi de la mise en service ainsi que du service technique de l'application mobile demandée par DragN'Receive. La principale personne chargée de la réalisation de ce projet ainsi que de la communication avec la société DragN'Receive sera Clément Pierre, ayant réalisé le précédent projet de site web pour DragN'Receive.

2 - Conception graphique

2.1 - Brief créatif

Le graphisme du site se doit de retranscrire l'identité de DragN'Receive tout en s'adaptant aux normes d'une application mobile. Pour ce faire nous conserverons les touches de couleurs représentatives des logos de la société que nous poserons sur un style graphique adapté aux smartphones.



2.2 - Aperçu de la charte graphique

2.2.1 - Logos

Durant le cadre du précédent projet, la réalisation de divers logos à l'effigie de l'entreprise ont été réalisés :





2.2.2 - Palette de couleurs

Concernant les couleurs, nous utiliserons ces références de couleurs basés sur la charte graphique de DragN'Receive, ainsi que d'autres tonalités comme réalisé pour le site web précédemment, celles-ci seront les principales :



2.2.3 - Typographies

Nous avons choisi comme typographie Roboto, utilisée sur le site web, elle est très sobre et facile à lire, critères majeurs pour une application sur un téléphone.

3 - Cahier des charges fonctionnel

3.1 - Présentation générale

La société DragN'Receive souhaite créer un système de Click & Collect à l'échelle de toute l'agglomération de Compiègne, en ayant pour partenaires des marchés et commerces centrés principalement sur la vente de denrées alimentaires et de produits de grande consommation.

Un site web a donc été réalisé, permettant la gestion des boutiques partenaires, permettant ainsi à ces dernières de proposer leurs produits, d'ajouter de nouveaux produits, gérant stocks et commandes clientes. Une API a également été réalisée et sera donc à faire évoluer.

DragN'Receive envisage désormais la conception d'une application mobile permettant aux clients de réaliser des commandes dans les différentes boutiques affiliées situées dans l'agglomération de Compiègne. La méthode de distribution de cette



application serait par l'affichage ou la distribution d'un QR code dans les boutiques partenaires permettant l'installation de l'application pour les clients.

3.2 - Analyse du besoin

Pour la réalisation de l'application mobile, il sera nécessaire d'utiliser un serveur, permettant l'hébergement du client mais aussi une éventuelle base de données. L'utilisation d'un langage de programmation compatible au développement mobile sera également nécessaire, il faudra également mettre en place une interface graphique, se basant sur la charte graphique définie avec le client.

Le but serait donc de réaliser une application mobile disponible sur Android et IOS, focalisée sur la vente de produits des boutiques partenaires pour les clients, qui géreront et suivront leurs commandes.

3.3 - Solutions envisagées

Les solutions envisagées permettant la réponse des divers besoins du projet sont :

- Utiliser le serveur dédié de l'entreprise ou bien utiliser un serveur OVH loué par cette dernière.
- L'utilisation d'un de ces langages de programmation : JavaScript, Java, Kotlin, C++ étant les langages les plus utilisés pour la réalisation d'applications mobiles.
- La réalisation d'une interface de programmation (API) communiquant avec (et entre) l'application mobile, la base de données. Les langages les plus répandus pour sa réalisation sont le Python, le JavaScript ainsi que le Java.
- L'utilisation d'un outil permettant la conversion / compatibilité du développement vers une solution mobile tels que : Ionic, NativeScript, Flutter.

3.4 - Solution retenue

3.4.1 - Présentation des choix

La solution optimale permettant de répondre à chacune des problématiques serait donc, selon notre expertise :

- D'utiliser le serveur dédié, permettant la mise en place de l'application mobile, de plus la base de données et l'API créé pour le site web y sont stockées et seront utilisées pour ce projet.
- Nous avons retenu comme langage de programmation le JavaScript avec l'utilisation du framework Vue, également utilisé dans le projet précédent.
- Pour ce qui est de l'API, nous allons utiliser l'API réalisée durant le dernier projet permettant de la faire évoluer durant le développement mobile, fait en JavaScript



avec l'utilisation de l'environnement d'exécution NodeJS accompagné du framework ExpressJS.

- Pour ce qui est du framework mobile, nous avons sélectionné Ionic, permettant d'utiliser le framework Vue, simplifiant donc grandement la vitesse de développement du projet et permettant une ressemblance avec le site web étant tout deux réalisés en Vue.

3.4.2 - Benchmark

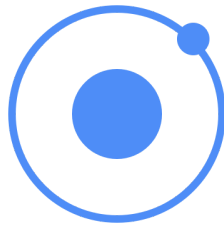
3.4.2.1 - Langage de programmation



	JavaScript	Java	Kotlin	C++
Utilisation	Généraliste	Généraliste	Généraliste	Client Lourd, Mobile
Typage	Fort	Fort	Fort	Fort
Avis	Langage très simple de compréhension, orienté objet, en plein essor ces dernières années dans de multiples domaines par ses performances de dynamisation et de polyvalence.	Langage de programmation orienté objet, demande un code rigoureux et long, compilation rapide, permet de créer divers types de clients. 100% compatible Android.	Langage de programmation orienté objet, compilé pour JVM et JavaScript. Concurrent de Java sur le marché du développement mobile Android.	Langage de programmation orienté objet compilé, langage très pointilleux permettant l'optimisation des performances et la précision pour des applications variées.
Date de lancement	1996	1995	2011	1985
Outils Mobile	Ionic, NativeScript, React Native	Java Natif	Kotlin Multiplatform Mobile	C++ Natif



3.4.2.2 - Framework



	Ionic	NativeScript	Flutter
UI	HTML / CSS / Composants	XML / CSS / Composants	Dart / Widgets
Apprentissage	Apprentissage facile et rapide, documentation complète, l'utilisation d'un framework web permet de réaliser rapidement une application mobile. Très intéressant pour sa rapidité si le balisage HTML est maîtrisé. Très similaire au développement web.	Le plus performant des 3, cependant c'est également celui qui nécessite le plus d'apprentissage. Offrant beaucoup de libertés, cela à un coût. Il fonctionne également avec du HTML et donc similaire au développement web. La documentation est très riche.	Apprentissage rapide et simple par l'utilisation des widgets à l'intérieur des scripts, il ne requiert pas la connaissance de l'HTML. La documentation ainsi que la chaîne YouTube officiel permettent d'avoir une compréhension accrue.
Écosystème	Gros écosystème, regroupant ceux de Vue Angular et React.	Communauté impliquée, moins importante mais encore en plein essor.	Croissance importante, moins importante que Ionic mais reste conséquente.
Langage	JavaScript, TypeScript	JavaScript, TypeScript	Dart
Compatibilités	Android et IOS	Android et IOS	Android et IOS
Performances	La moins performante, mais reste puissante.	Bonne performance, au coude à coude avec Flutter.	Bonne performance, au coude à coude avec NativeScript.
Taille	Léger	Plus lourd que Ionic et Flutter	Moins léger que Ionic
Date de lancement	2013	2015	2017
Utilisé par	Electronic Arts, Airbus Helicopters	Raiffeisen Bank, Sennheiser, PUMA	BMW, Alibaba, eBay, Toyota, Google Pay



3.4.3 - Explication des choix

Notre équipe a donc choisi d'utiliser le langage JavaScript qui, pour la réalisation du projet, est le plus approprié, utilisable avec de multiples frameworks, notre choix se justifiant par ses performances mais également car il va nous permettre de réaliser aussi bien l'application que le côté permettant les interactions avec la base de données. De plus, il a déjà fait ses preuves pour la réalisation du site web de DragN'Receive.

Nous avons donc choisi trois frameworks, l'un pour la réalisation du **Back-End***, l'autre pour le **Front-End**** et le dernier pour rendre compatible le logiciel sous forme d'une application mobile.

Nous avons choisi en Back-End le framework ExpressJS basé sur du NodeJS puisque nous allons utiliser l'API développée durant les travaux du site web.

Pour ce qui est du Front-End nous avons décidé d'utiliser Vue. Le choix de ce dernier, n'aura aucun impact sur ses performances mais aura un impact positif sur la rapidité, qualité et compréhension du code de notre équipe.

Pour l'outil mobile, nous avons choisi le framework Ionic car ce dernier est compatible avec Vue et plus globalement le JavaScript HTML et CSS, possédant des environnements de configuration qui permettent donc des libertés visuelles. Ionic permet également d'utiliser divers composants permettant un développement rapide et une interface soignée et homogène. Il permet aussi d'utiliser l'application sur Android ainsi que IOS permettant de cibler la majorité des utilisateurs mobile.

***Back-End** : Partie cachée de l'utilisateur, permettant la gestion de la base de données et les interactions de cette dernière permettant la communication avec le Front-End

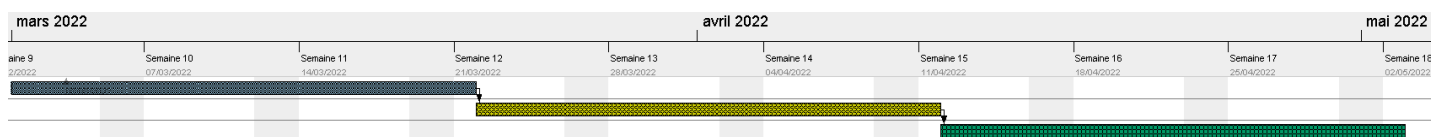
****Front-End** : Partie d'interfaces graphiques visible par l'utilisateur qui vient interagir avec le Back-End pour retourner/envoyer des éléments à l'utilisateur/la base de données.



4 - Analyses et estimations

4.1 - Planification

Nom	Date de début	Date de fin
● Rédaction du Cahier des charges	01/03/2022	21/03/2022
● Evolution de l'API	22/03/2022	11/04/2022
● Développement mobile	12/04/2022	02/05/2022



4.2 - Budgétisation

Quantité	Description	Réduction	Coût*
1	Création d'une application mobile permettant l'achat par des clients des produits partenaires ainsi que la gestion et suivis de leurs commandes.	-	6 000 €
1	Évolution de l'API pour le Back-End		6 000 €
1	Adaptation de la charte graphique sur mobile	-	800 €
1	Référencement	-	350 €
1	Service technique	-	350 €

Total :

	13 500 €
--	-----------------

* Les coûts horaires sont de 50 euros.



5 - Conception fonctionnelle

5.1 - Dictionnaire de données

5.1.1 - Utilisateur

Champs	Types	Tailles	Contraintes	Descriptions
id	varchar	36	PK, Obligatoire	Chaîne de caractère unique générée aléatoirement.
lastname	varchar	50	Obligatoire	nom de famille
firstname	varchar	50	Obligatoire	prénom
email	varchar	100	Obligatoire, format d'email	adresse mail
password	varchar	60	Obligatoire, comporte une majuscule, une minuscule, un nombre et un caractère spécial au minimum	mot de passe
path	varchar	255	Non-Obligatoire	chemin vers la photo
createdAt	datetime	yyyy-mm-ddThh:mm:ss.000Z	Obligatoire	date de création
updatedAt	datetime	yyyy-mm-ddThh:mm:ss.000Z	Obligatoire	date de la dernière modification
roleId	varchar	1	FK: Rôle, Obligatoire	permet de séparer, admins, clients et partenaires
shopId	varchar	36	FK: Boutique, Non-Obligatoire	relie la boutique du partenaire



5.1.2 - Rôle

Champs	Types	Tailles	Contraintes	Descriptions
id	varchar	1	PK, Obligatoire	Chaîne de caractère entre 1 et 4
label	varchar	20	Obligatoire	nom du rôle
createdAt	datetime	yyyy-mm-ddThh:mm:ss.000Z	Obligatoire	date de création
updatedAt	datetime	yyyy-mm-ddThh:mm:ss.000Z	Obligatoire	date de la dernière modification

5.1.3 - Boutique

Champs	Types	Tailles	Contraintes	Descriptions
id	varchar	36	PK, Obligatoire	Chaîne de caractère unique générée aléatoirement.
name	varchar	50	Obligatoire	nom de la boutique
email	varchar	100	Obligatoire	adresse mail
phone	varchar	10	Non-Obligatoire	numéro de téléphone
city	varchar	60	Non-Obligatoire	ville
street	varchar	100	Non-Obligatoire	rue
postal	int	5	Non-Obligatoire	code postal
path	varchar	255	Non-Obligatoire	chemin vers le logo
deleted	tinyint (bool)	1	Obligatoire	Booléen pour savoir si la boutique est supprimée
createdAt	datetime	yyyy-mm-ddThh:mm:ss.000Z	Obligatoire	date de création
updatedAt	datetime	yyyy-mm-ddThh:mm:ss.000Z	Obligatoire	date de la dernière modification



5.1.4 - Produit

Champs	Types	Tailles	Contraintes	Descriptions
id	varchar	36	PK, Obligatoire	Chaîne de caractère unique générée aléatoirement.
name	varchar	50	Obligatoire	nom du rôle
description	varchar	255	Obligatoire	description du produit
price	float	9,2	Obligatoire	prix
stock	int	11	Obligatoire	stock restant
path	varchar	255	Obligatoire	chemin vers l'image du produit
deleted	tinyint (bool)	1	Obligatoire	entier 1 ou 0 pour savoir si le produit est supprimé
createdAt	datetime	yyyy-mm-ddThh:mm:ss.000Z	Obligatoire	date de création
updatedAt	datetime	yyyy-mm-ddThh:mm:ss.000Z	Obligatoire	date de la dernière modification
shopId	varchar	36	FK: Boutique, Obligatoire	relie le produit à sa boutique



5.1.5 - Commande

Champs	Types	Tailles	Contraintes	Descriptions
id	varchar	36	PK, Obligatoire	Chaîne de caractère unique générée aléatoirement.
quantities	int	11	Obligatoire	quantité de produit
price	float	9,2	Obligatoire	prix total de la commande
number	varchar	100	Obligatoire	numéro de commande
createdAt	datetime	yyyy-mm-ddThh:mm:ss.000Z	Obligatoire	date de création
updatedAt	datetime	yyyy-mm-ddThh:mm:ss.000Z	Obligatoire	date de la dernière modification
userId	varchar	36	Obligatoire	relie l'utilisateur à la commande
productId	varchar	36	Obligatoire	relie le produit à la commande
shopId	varchar	36	Obligatoire	relie la boutique à la commande
orderId	varchar	1	FK: Statut Commande, Obligatoire	relie un statut à la commande

5.1.6 - Statut Commande

Champs	Types	Tailles	Contraintes	Descriptions
id	varchar	1	PK, Obligatoire	Chaîne de caractère entre 1 et 5
label	varchar	20	Obligatoire	nom du statut
createdAt	datetime	yyyy-mm-ddThh:mm:ss.000Z	Obligatoire	date de création
updatedAt	datetime	yyyy-mm-ddThh:mm:ss.000Z	Obligatoire	date de la dernière modification



5.1.7 - Ticket

Champs	Types	Tailles	Contraintes	Descriptions
id	varchar	36	PK, Obligatoire	Chaîne de caractère unique générée aléatoirement.
title	varchar	60	Obligatoire	titre du ticket
content	varchar	255	Obligatoire	contenu
createdAt	datetime	yyyy-mm-ddThh:mm:ss.000Z	Obligatoire	date de création
updatedAt	datetime	yyyy-mm-ddThh:mm:ss.000Z	Obligatoire	date de la dernière modification
userId	varchar	36	Obligatoire	relie l'utilisateur au ticket
ticketStatusId	varchar	1	FK: Statut Ticket, Obligatoire	relie un statut au ticket

5.1.8 - Statut Ticket

Champs	Types	Tailles	Contraintes	Descriptions
id	varchar	1	PK, Obligatoire	Chaîne de caractère entre 1 et 2
label	varchar	20	Obligatoire	nom du statut
createdAt	datetime	yyyy-mm-ddThh:mm:ss.000Z	Obligatoire	date de création
updatedAt	datetime	yyyy-mm-ddThh:mm:ss.000Z	Obligatoire	date de la dernière modification



5.1.9 - Message

Champs	Types	Tailles	Contraintes	Descriptions
id	varchar	36	PK, Obligatoire	Chaîne de caractère unique générée aléatoirement.
content	varchar	255	Obligatoire	contenu
createdAt	datetime	yyyy-mm-ddThh:mm:ss.000Z	Obligatoire	date de création
updatedAt	datetime	yyyy-mm-ddThh:mm:ss.000Z	Obligatoire	date de la dernière modification
userId	varchar	36	Obligatoire	relie l'utilisateur au message
ticketId	varchar	36	FK: Statut Ticket, Obligatoire	relie un message au ticket

5.1.10 - Web Token

Champs	Types	Tailles	Contraintes	Descriptions
id	varchar	36	PK, Obligatoire	Chaîne de caractère unique générée aléatoirement.
token	varchar	255	Obligatoire	token généré aléatoirement
createdAt	datetime	yyyy-mm-ddThh:mm:ss.000Z	Obligatoire	date de création
updatedAt	datetime	yyyy-mm-ddThh:mm:ss.000Z	Obligatoire	date de la dernière modification
userId	varchar	36	Obligatoire	relie l'utilisateur au ticket

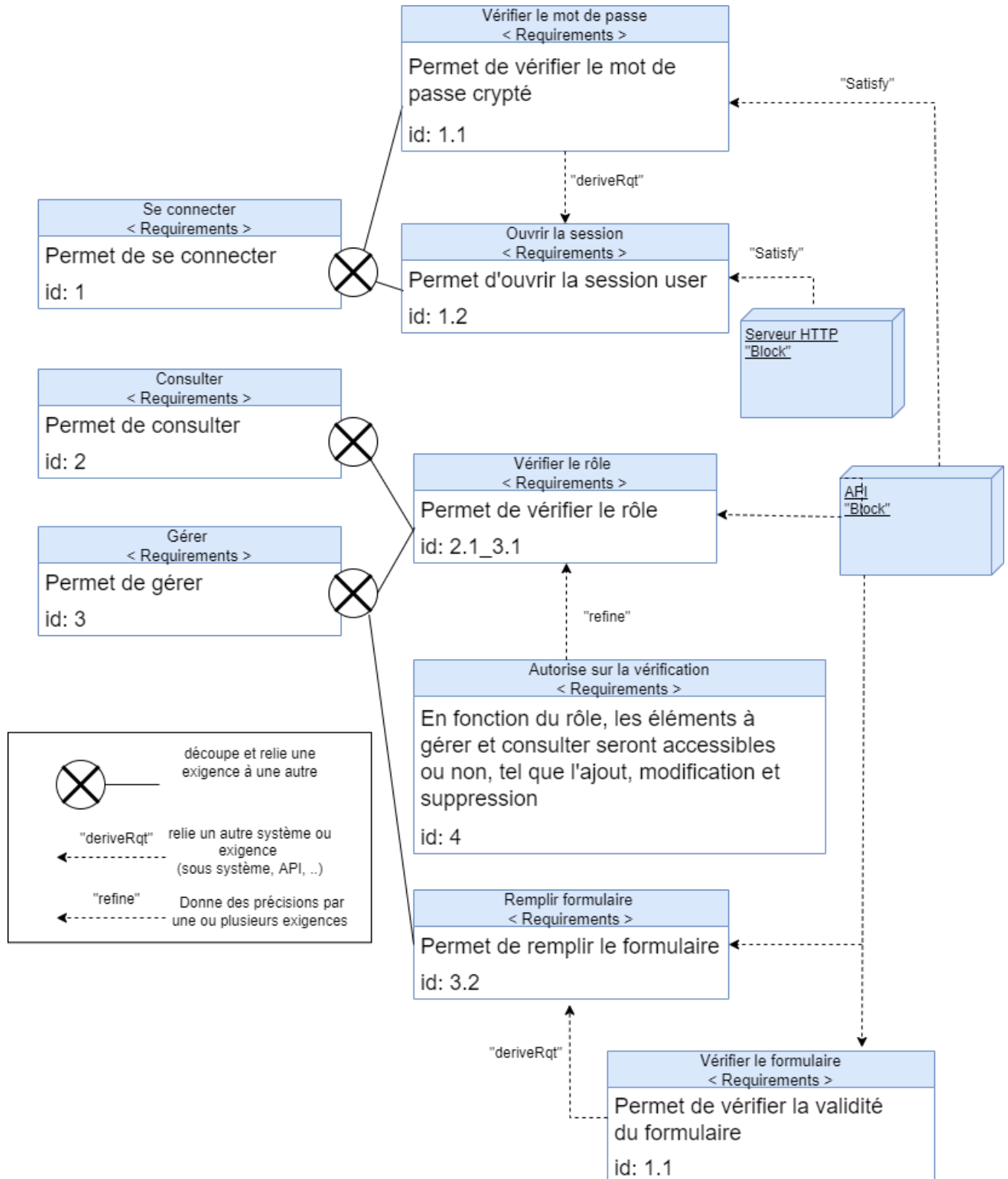


5.1.11 - Mobile Code

Champs	Types	Tailles	Contraintes	Descriptions
id	varchar	36	PK, Obligatoire	Chaîne de caractère unique générée aléatoirement.
code	varchar	4	Obligatoire	code généré aléatoirement (nombre)
createdAt	datetime	yyyy-mm-ddThh:mm:ss.000Z	Obligatoire	date de création
updatedAt	datetime	yyyy-mm-ddThh:mm:ss.000Z	Obligatoire	date de la dernière modification
userId	varchar	36	Obligatoire	relie l'utilisateur au ticket

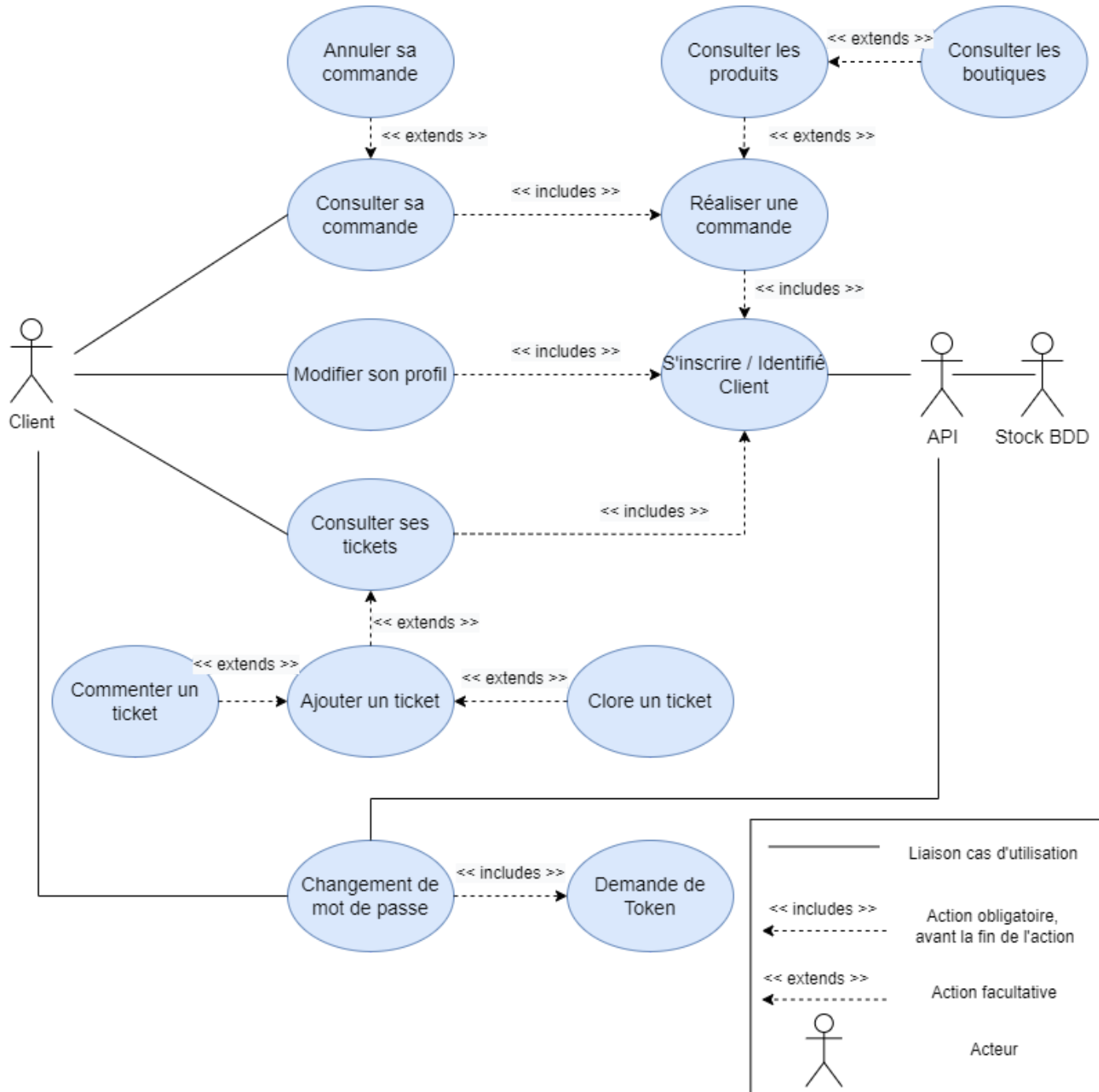


5.2 - Exigences





5.3 - Cas d'utilisations





5.4 - Maquettes

⚠ Les maquettes suivantes sont des aperçus de la basse fidélité de l'application mobile. Elles sont donc volontairement dénuées de toutes formes de graphisme. Ces aperçus servent donc à lister et placer les éléments fonctionnels sur les différentes pages du site plus qu'à donner une idée de son esthétique, qui seront voués à changer en fonction des besoins exprimés.

5.4.1 - Connexion

Logo

Identifiant

Mot de passe

Connexion



5.4.2 - Profil

Logo

NomPrénom

Email
Nom

Modifier mot de passe

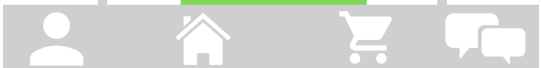
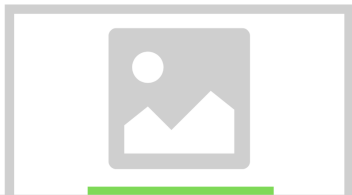
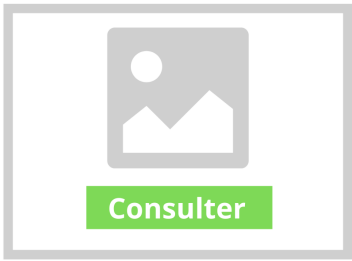
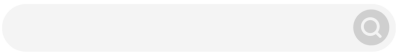
Supprimer le compte

Vos commandes

Voir



5.4.2 - Boutique





5.4.3 - Commande

Logo

Numéro de commande

Statut de commande | Nom de la boutique



Nom
x Quantité

Annuler

Statut de commande | Nom de la boutique



Nom
x Quantité









Logo

Numéro de commande





...

Voir

Statut Date

Numéro de commande



Voir

Statut Date

Numéro de commande





Voir











5.4.4 - Produit





5.4.5 - Ticket

Logo

Titre

Contenu

Auteur

Contenu

Date

Auteur

Contenu

Date

Auteur

Contenu

Date

Auteur

Contenu

Date

Auteur

Contenu

Date

Auteur

Contenu

Date

Logo

Titre

Date

Statut

Voir

Clore

Titre

Date

Statut

Voir

Clore

Titre

Date

Statut

Voir

Clore

Titre

Date

Statut

Voir

Clos



5.4.6 - Inscription

Logo

The registration form is enclosed in a light gray border. On the left side, there are two icons: a person icon at the top and a lock icon below it. To the right of these icons are several input fields and buttons. The input fields are labeled 'Nom', 'Prénom', 'Email', 'Mot de passe', and 'Confirmer'. Below these fields is a green button labeled 'Créer'.

5.4.7 - Récupération de mot de passe

Logo

The password recovery form is enclosed in a light gray border. On the left side, there is a person icon. To the right of the icon is a single input field labeled 'Email'. Below the input field is a green button labeled 'Demander'.



5.4.8 - Changement de mot de passe

Logo

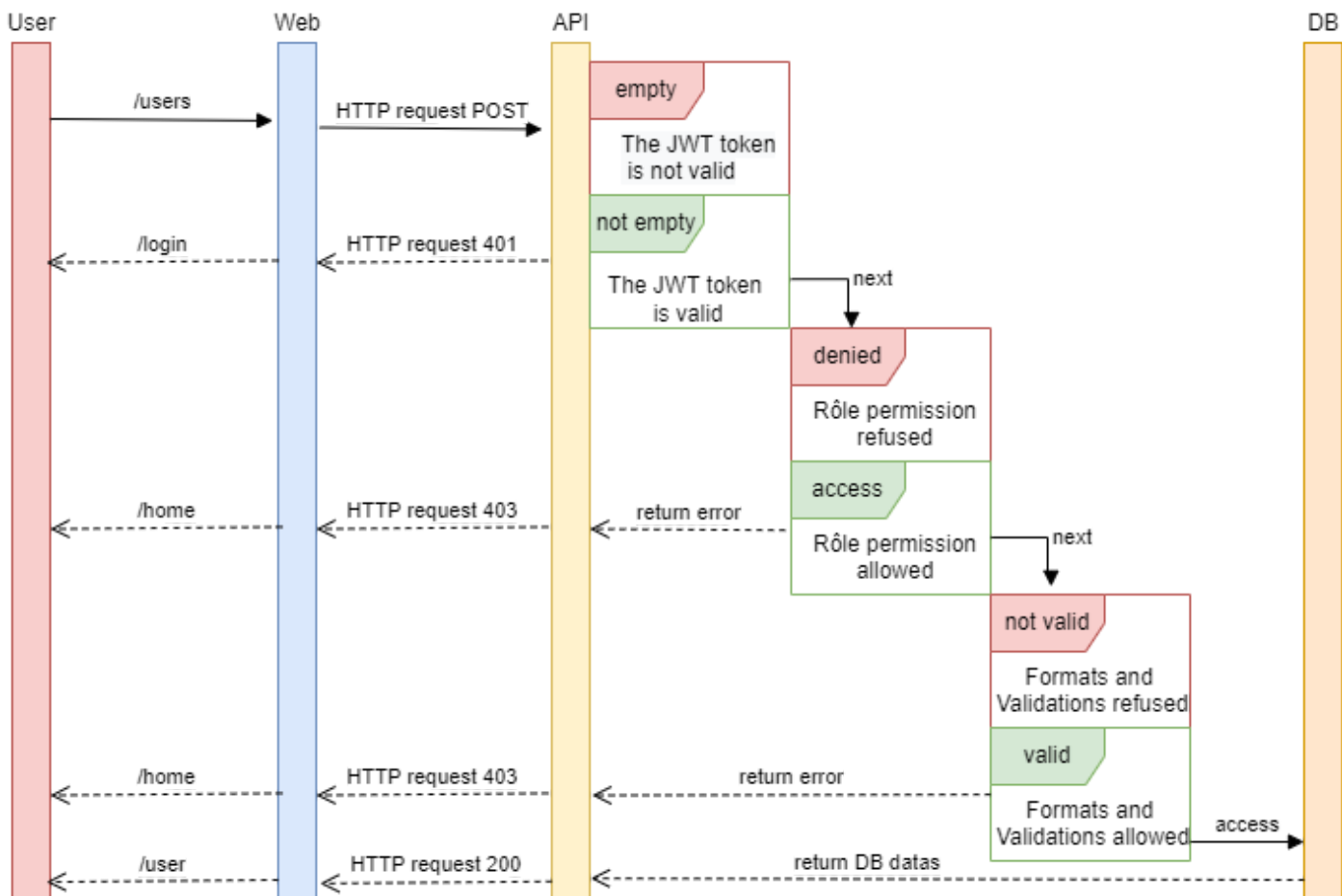
The diagram illustrates the password change interface. It features a central area with a lock icon on the left and four input fields on the right: 'Code', 'Mot de passe', 'Confirmer', and 'Changer'. The 'Changer' button is highlighted in green.



6 - Cahier des charges technique

6.1 - Conception technique

6.1.1 - Séquence





6.1.2 - Spécificités des droits d'Administration des fonctionnalités

Connexion sur l'application mobile : Vérifie que l'utilisateur existe via son email, Vérifie que l'utilisateur est un client. Comparer le mot de passe avec celui en base de données. Si tout est valide, retourne le Token JWT de l'utilisateur ainsi que ses informations personnelles.

Inscription : Vérifie que les formats du nom, prénom, email et mot de passe soient valides, vérification que l'email n'existe pas. Si valide, hash le mot de passe et envoie par email à l'utilisateur et création de son dossier sur le dossier **"Store"**.

Génération de code changement de mot de passe mobile : Vérifie la validité de l'email, vérifie que l'utilisateur existe et qu'il soit un client. Si valide, cela crée un code de 4 chiffres valable pendant 1h et envoie par email du lien de réinitialisation.

Réinitialisation de mot de passe mobile : Vérifie la validité du mot de passe, celle du Code (existe et non expiré), que l'utilisateur existe et que l'utilisateur soit un client. Si valide, supprime le Code et change le mot de passe du client et envoie par email pour lui confirmer le changement.

Voir son profil : Vérification du Token, vérifie les permissions de l'utilisateur, retourne ses infos.

Modifier son profil : Vérification du Token, vérifie les permissions de l'utilisateur. L'utilisateur peut modifier son mot de passe et sa photo profil, vérifier le nouveau mot de passe et comparer le mot de passe actuel avec celui en base de données. Pour la photo de profil, vérifie la validité de son format et stock l'image dans le dossier de l'utilisateur dans le "Store". Si valide, modifie les informations de l'utilisateur.

Supprimer son compte : Vérification du Token, vérifie les permissions de l'utilisateur. Si valide et qu'il n'a aucune commande en cours, supprime l'utilisateur en base de données, ainsi que son dossier dans le "Store" puis envoie un email à ce dernier pour l'en informer.

Voir une boutique : Vérification du Token, vérifie les permissions de l'utilisateur, retourne la ou les boutique(s) avec les produits ainsi que ses partenaires.

Voir un produit : Vérification du Token, vérifie les permissions de l'utilisateur, vérifie que le produit existe bien ainsi que la boutique du produit, si valide, retourne le ou les produit(s) ainsi que sa boutique.

Créer un ticket : Vérification du Token, vérifie les permissions de l'utilisateur, vérifie que les formats du titre et du contenu soient valides. Si valide, crée un ticket avec pour statut "Ouvert".



Voir un ticket : Vérification du Token, vérifie les permissions de l'utilisateur, retourne ensuite le ou les ticket(s) contenant les messages qui eux mêmes contiennent un utilisateur.

Clore un ticket : Vérification du Token, vérifie les permissions de l'utilisateur, vérifie que le ticket existe et que son utilisateur existe également. Vérifie que le ticket ne soit pas de statut "Clos". Si valide, passe le statut en "Clos" et avertit l'utilisateur possédant le ticket par email.

Répondre à un ticket : Vérification du Token, vérifie les permissions de l'utilisateur, vérifie que le ticket existe et que l'utilisateur existe également. Vérifie le format du contenu. Si valide, crée un nouveau message au ticket et envoie par email à la personne possédant le ticket si c'est une réponse venant d'un Admin ou Super-Admin.

Faire une commande : Vérification du Token, vérifie les permissions de l'utilisateur, vérifie que les produits existent, vérifie que les quantités sont valides en fonction des stocks et au bon format. Si valide, crée les commandes en définissant le numéro de commande, tout en supprimant les quantités des stocks. Pour finir un envoi par mail vient confirmer la commande à l'utilisateur, ainsi qu'aux boutiques partenaires, joignant les factures en formats pdf, les pdf sont stockés dans le dossier de facture de l'utilisateur dans le "Store".

Voir une commande : Vérification du Token, vérifie les permissions de l'utilisateur, vérifie que la commande existe, retournant ainsi la ou les commande(s) contenant les produits en question.

Annuler une commande : Vérification du Token, vérifie les permissions de l'utilisateur, vérifie que les boutiques et les commandes attribuées aux numéros et boutiques existent. Vérifie également que chaque commande possède le statut "Validée", si tout est valide, modifie le statut des commandes, réattribue les stocks aux produits et envoie des mails de confirmation d'annulation des commandes.

*** Le dossier du nom de "Store" est le lieu stockant les fichiers des différents acteurs de l'application.**



6.2 - Environnement technologique

6.2.1 - Hardware

Pour la réalisation de ce projet l'environnement technologique utilisé a été :

- Ordinateur fixe et portable
- Clavier, souris, écran
- Box Wifi

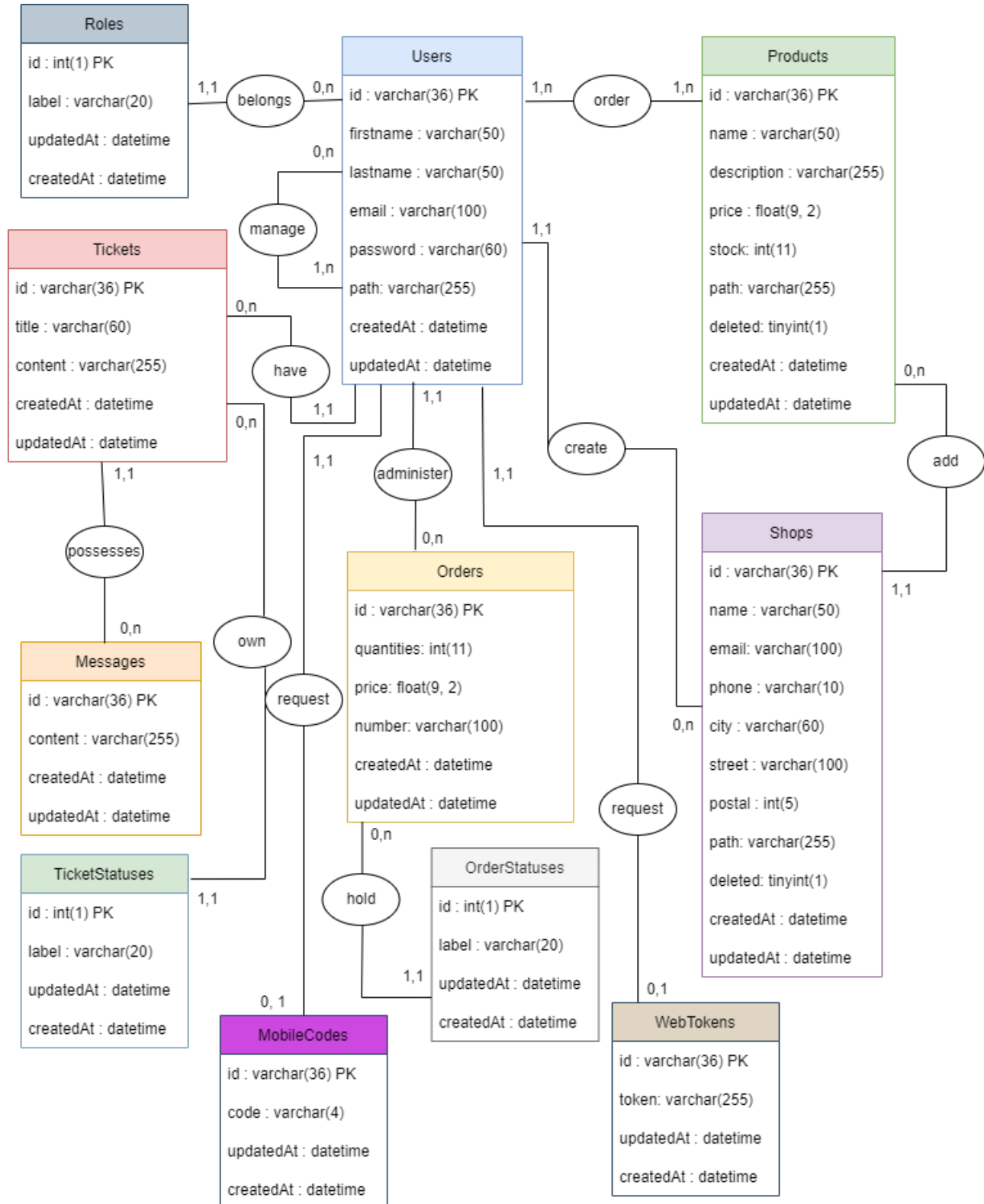
6.2.2 - Software

La conception de ce projet nous a fait utiliser les outils suivants :

- Windows comme système d'exploitation
- Visual Studio Code comme éditeur de texte et invite de commandes
- Android Studio pour l'émulation d'un Google Pixel 5 (393 x 851)
- GitHub pour gestion de code et de version mais aussi son stockage
- Postman pour faire des requêtes HTTP sur l'API
- NPM, gestionnaire de paquets / librairies JavaScript
- NodeJS pour JavaScript
- Ionic comme framework mobile, ainsi que Capacitor pour construire le projet sur Android et IOS
- Google Chrome pour tester l'application via navigateur web
- Draw.io pour la réalisation des diagrammes
- Jimdo pour la conception des logos
- Canva pour la conception des maquettes
- Gantt pour la planification des travaux
- Google doc pour la rédaction du cahier des charges, son stockage et sa gestion de versions



6.3 - Modélisation de données





7 - Réalisation technique

7.1 - Les points forts du projet

Le plus grand avantage de ce projet est qu'il m'a permis de réaliser une approche de développement et de gestion de projet autant au back-end qu'au front-end, tout en apprenant à adapter mon front à un format mobile.

J'ai également progressé dans le développement front, notamment avec l'utilisation de Vue.JS, me permettant d'approfondir l'utilisation des composants.

Un des autres avantages serait également personnel, le projet me permettant de réaliser plus rapidement et facilement l'application mobile de gestion de trésorerie et suivis des matchs de mon association de basket-ball.

7.2 - Les problèmes rencontrés

Les problèmes rencontrés se sont vus être moindres par rapport à mon application web, cette application mobile étant également basée sur le framework Vue.JS.

J'ai rencontré des difficultés dans l'envoi de formats de données et la gestion des commandes lors de l'amélioration de mon API.

Le client Mobile et l'utilisation de balise et composant de Ionic dédiés ont été compliqués à prendre en main au départ, j'ai cependant au fil du temps réussi à les appréhender.



8 - Tests

8.1 - Tests unitaires

Pour le passage des tests unitaires de mon API, j'ai choisi d'utiliser Mocha et Chai, des bibliothèques JavaScript permettant d'effectuer des tests sur différentes fonctionnalités et vérification de l'API, notamment via requêtes HTTP.

Exists

Email exist

- ✓ Email exist in Database (59ms)
- ✓ Email dont exist in Database

Shop exist

- ✓ Shop exist in Database
- ✓ Shop dont exist in Database

Formats

Email format

- ✓ Email format is valid
- ✓ Email should be a string
- ✓ Email should be "string@string.string" format
- ✓ Email length should be less or equal to 100

Password format

- ✓ Password format is valid
- ✓ Password should be a string
- ✓ Password should have one upper case character
- ✓ Password should have one lower case character
- ✓ Password should have one number character
- ✓ Password should less or equal to 60 characters
- ✓ Password should be more than 7 characters

Requests API

GET /api/currentUser

- ✓ Response is an object
- ✓ Response is null because no token in header

POST /api/loginClient

- ✓ Sadmin, Admin or Partner cant be loggedin
- ✓ Wrong password cannot loggedin
- ✓ Wrong email cannot loggedin
- ✓ Valid Client can be logged in



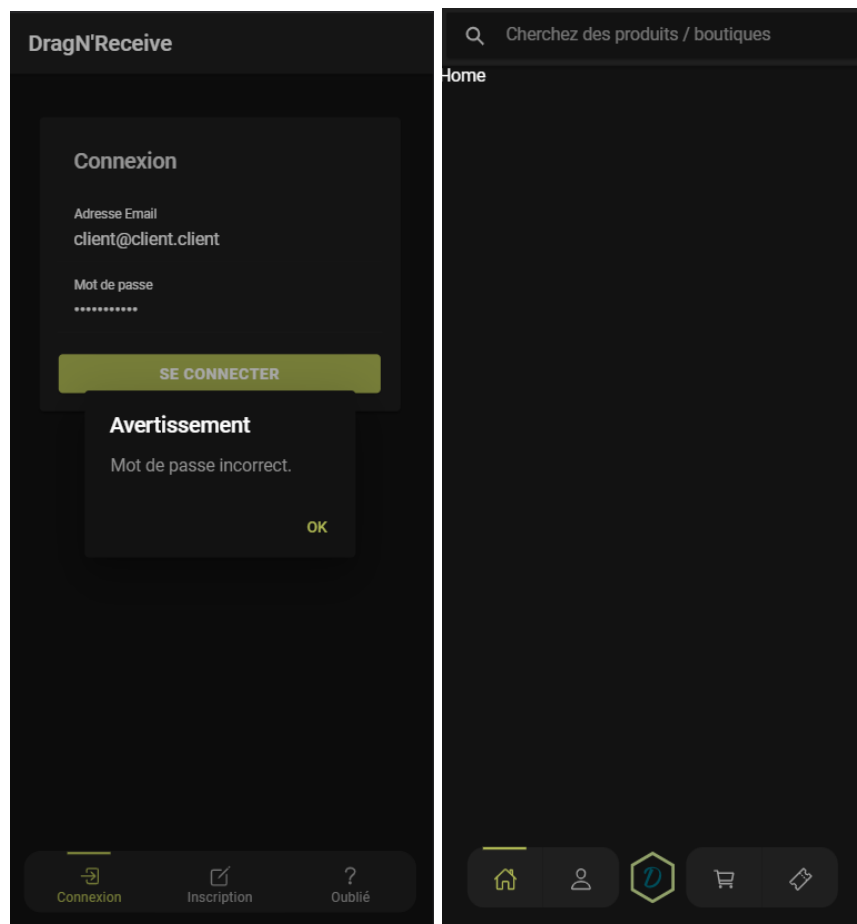
8.2 - Tests de validation

8.2.1 - Connexion

The following table summarizes the content of the three screenshots shown in the image:

Écran	Adresse Email	Mot de passe	Message d'erreur / Validation
1. État initial			Le-mail est requis
2. Email invalide	c		Le format de l'e-mail doit être valide
3. Mot de passe invalide	client@client.cliente	*****	Le mot de passe doit contenir : une majuscule, une minuscule, un chiffre, un caractère spécial

Each screenshot also displays a bottom navigation bar with three options: Connexion (selected), Inscription, and Oublié.





8.2.2 - Réinitialisation de mot de passe

DragN'Receive

Réinitialisation de mot de passe

Adresse Email

L'e-mail est requis

ENVOYER

Connexion
Inscription
Oublié

DragN'Receive

Réinitialisation de mot de passe

Adresse Email

client@client.client

ENVOYER

Souhaitez vous recevoir votre code de réinitialisation ?

Vous recevrez un code par email vous permettant de changer de mot de passe

ANNULER
DEMANDER

Connexion
Inscription
Oublié

DragN'Receive

Annuler

Réinitialisation de mot de passe

Un code a été envoyé à votre adresse email ! Saisissez le code :

Mot de passe

Le mot de passe est requis

Confirmez le mot de passe

Veillez confirmer le mot de passe

CHANGER

Connexion
Inscription
Oublié

Search...

DragN'Receive - Demande de réinitialisation de mot de passe !

to: <client@client.client> a few seconds ago

DragN'Receive - Demande de réinitialisation de mot de passe !

From: Service DragN'Receive <service@dragndreceive.fr>
To: <client@client.client>

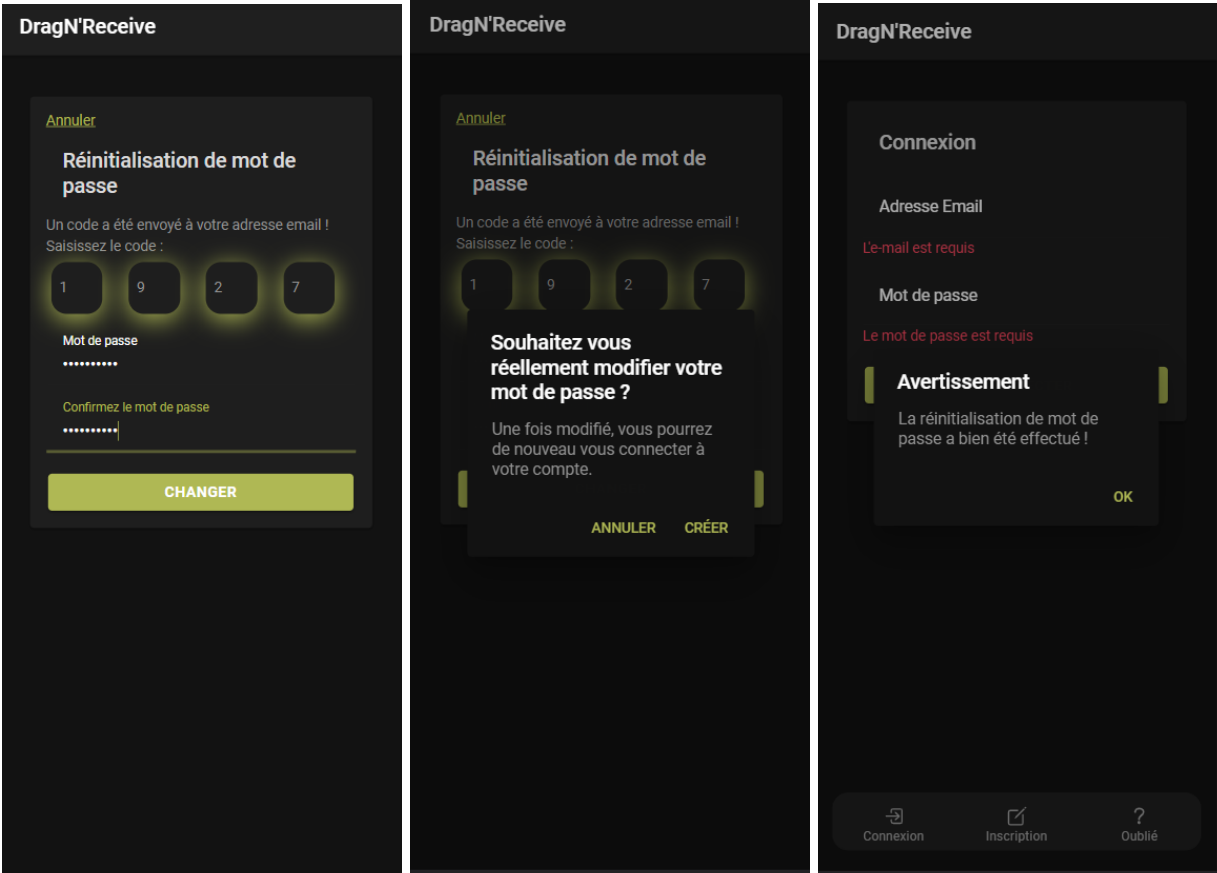
2022-04-06 08:12, 641 Bytes

Show Headers

HTML HTML Source Text Raw Spam Analysis HTML Check Tech Info



Bonjour Client Client,
Vous avez demandé une réinitialisation de mot de passe :
Votre code : 1927
Ce n'est pas vous ? Veuillez changer de mot de passe par sécurité.

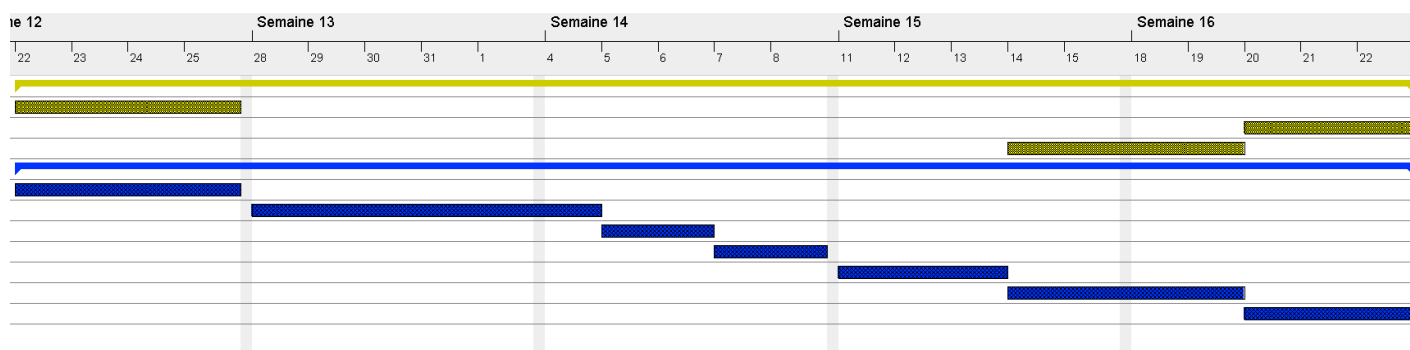




9 - Indicateurs de performance

9.1 - Analyse de la planification mise à jour

Nom	Date de début	Date de fin
☐ ● API	22/03/2022	22/04/2022
● amélioration et modificat.	22/03/2022	25/03/2022
● annulation de commande	20/04/2022	22/04/2022
● affichages de commandes	14/04/2022	19/04/2022
☐ ● MOBILE	22/03/2022	22/04/2022
● Initialisation	22/03/2022	25/03/2022
● Authentification	28/03/2022	04/04/2022
● Profile	05/04/2022	06/04/2022
● Ticket	07/04/2022	08/04/2022
● Boutique / Produit	11/04/2022	13/04/2022
● Panier / Commander	14/04/2022	19/04/2022
● Commandes	20/04/2022	22/04/2022





9.2 - Analyse du coût global

Quantité	Description	Réduction	Coût
1	Création d'une application mobile permettant l'achat par des clients des produits partenaires ainsi que la gestion et suivis de leurs commandes.	-	10 000 €
1	Évolution de l'API pour le Back-End		5 200 €
1	Adaptation de la charte graphique sur mobile	-	800 €
1	Référencement	-	350 €
1	Service technique	-	350 €

Total :

	16 700 €
--	-----------------



10 - Conclusion

Le projet s'est trouvé être très enrichissant que ce soit dans la gestion du projet, la rédaction du cahier des charges mais aussi dans le développement back-end et front-end d'applications mobiles.

D'un point de vue personnel il m'a également appris, grâce à l'autonomie réalisé sur le projet, à chercher les réponses par moi même et prendre conscience des compétences que je possède.

Le projet m'a aussi appris à réaliser de la veille et me renseigner sur les différents langages et frameworks de programmation, me permettant de choisir un environnement de développement me permettant de m'épanouir dans le domaine du développement Mobile.

Suite à la deadline de plus en plus proche, j'ai eu peur de ne pas avoir le temps de terminer, cependant, grâce aux compétences acquises sur le premier PPE, j'ai pu réaliser ce projet beaucoup plus rapidement.

Pour conclure, la réalisation de ce projet m'a permis de réaliser quelque chose de concret sur une longue période ainsi qu'apprendre à me tenir aux deadlines et définir des endpoints. Je suis satisfait de mon travail et ces heures passées à le réaliser ont confirmé ma détermination à poursuivre dans le développement.