

# Documentation Technique Projet Sup Magasin

## I) Stack Technologique

L'architecture du projet sera de type back/font end.

Constitué d'une api en C# ASP .NETCORE dans sa version 3.1. Ce choix est dû principalement aux connaissances avancées dans ce langage de nos développeurs. Elle aura le rôle central dans la solution, car elle servira d'interface avec la base de donnée(BDD), et les différents éléments nécessaires à son utilisation.

Le front, web et mobile se fera en Ionic version ??, Framework javascript frontend, couplé à du typescript par le fait de notre développeur frontend plus à l'aise avec cet outil.

L'API et le front communique par le biais d'appel front et de renvois de données sous format Json. La sécurité est assurée par des JWT(javascript web token), dérivé de l'auth2.0. Les tokens sont valables 24H et nécessaire pour chaque requête sur l'api sauf ( route User).

L'Api est une interface entre la base de données, mis en place en Mongo DB dans sa version 5.2?. Pour sa souplesse et son économie de ressource par le stockage de données dans des fichiers JSON.

L'Architecture a été pensée pour être le plus souple possible. Nous avons décidé d'utiliser Windows Sever 2016 pour l'hébergement et la création du réseau au sein de l'application.

Les bornes Bluetooth sont pour ce POC, des raspberry pi 3B, montée par nos soins, pour vérifier et identifier les usagers dans le magasin.

## **II) La Base de Données (annexe 1)**

La base de données a été pensée pour être la plus compact possible en rassemblant les éléments de Type 1/N dans un seul schéma.

Par cette méthode, nous avons pu dégager 4 schémas, le Customer qui représente la pierre angulaire, comprenant les différents éléments liés aux clients, de ses moyens d'identification à ses données personnelles.

De plus nous l'avons liée au schéma vente afin que celui-ci soit liée aux produits qu'il achète et que nous utiliserons dans la recherche de promotion en temps réels dans le schéma produit, tous en conservant une historique des ventes.

Ce dernier schéma cité comprend toutes les informations de produits ainsi que ces stocks et son fournisseurs attiré. La réflexion c'est porté, qu'un seul fournisseur ne pouvait produire qu'un unique produit et que nous ne passions que par lui pour les commandes.

Les stocks sont pris en comptes avec une alerte signalée en cas d'atteinte d'un stock d'alerte et d'une menace de rupture de stock.

De plus nous assurons la traçabilité des produits par l'enregistrements des lots, (nous achetons à des grossistes) avec dates d'achats et dates de péremptions pour les produits périssables (Alimentaire)

La table magasin quant à elle, dans ce POC rien n'était indiqué pour la présence de plusieurs magasins, donc nous avons préféré prendre de l'avance et anticiper des ouvertures futures. Elle comprend les données liées aux magasins ainsi que ses employés que nous affectons à chaque magasin. Ces dernières données comprennent, les informations liées à chaque employé ainsi que ces logins pour ce connecter à la partie caché du magasin et comprend différentes fonctions de gestions (contrôle en temps réels des stocks, alertes, gestions des rayons et commandes, monitorings, etc.)

### III) Les Routes

L'ensemble des routes se trouvent sur la documentation swagger à cette adresse, ainsi que ses entrées valides :

[https://app.swaggerhub.com/apis/Zamiquie/api-sup\\_magasin/v1](https://app.swaggerhub.com/apis/Zamiquie/api-sup_magasin/v1)

Voici un aperçu :

L'URL de l'api:

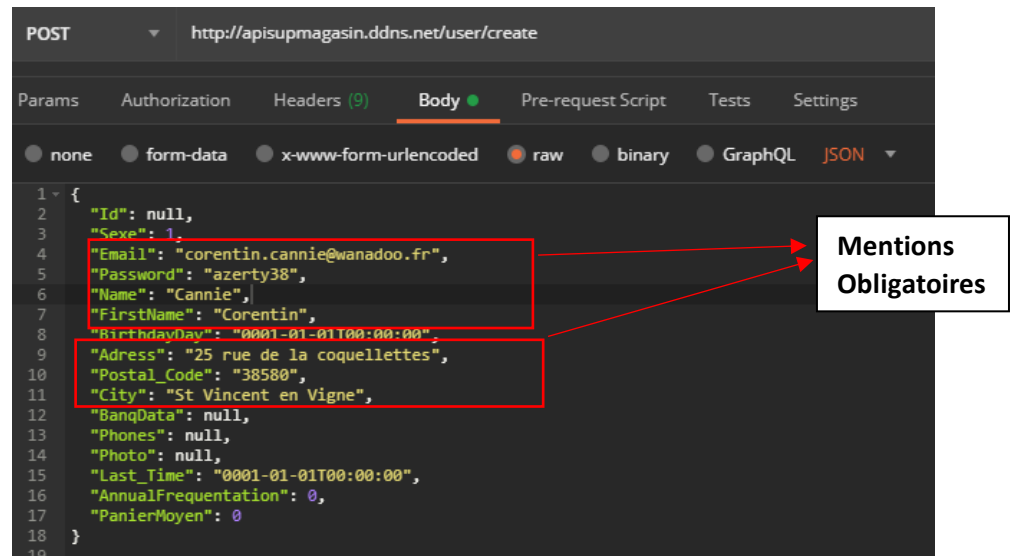
<http://apisupmagasin.ddns.net>

<b>Customer</b> ^	
GET	/Customer/All
GET	/Customer/{id}
GET	/Customer/{id}/BanqDatas
GET	/Customer/{id}/Phones
POST	/Customer/addCustomer
POST	/Customer/addMultiCustomer
PUT	/Customer/updateCustomer
DELETE	/Customer/Delete
<b>Default</b> ^	
GET	/
OPTIONS	/
<b>Error</b> ^	
GET	/Error
<b>Product</b> ^	
GET	/Product/All
GET	/Product/{id}
GET	/Product/name/{designation}
GET	/Product/{id}/commentary
GET	/Product/{id}/supplier
GET	/Product/{id}/deliverv

## IV) L'authentification

L'authentification est enregistrée dans le Schéma Customer.

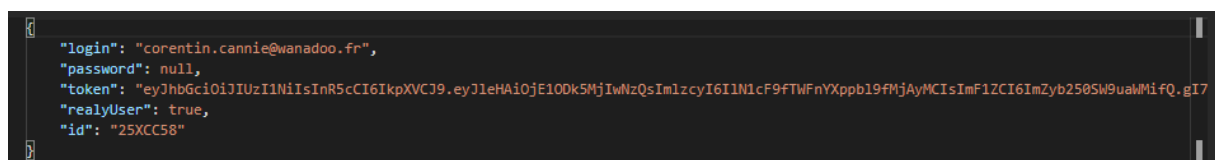
a) L'enregistrement (<http://apisupmagasin.ddns.net/user/create>) : POST



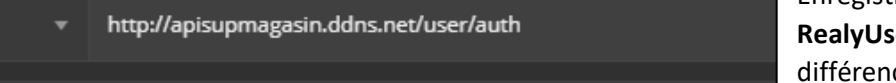
Code Erreur Spécifique :

<u>Code Erreur</u>	<u>Message</u>	<u>Raison</u>
400	Filds Missing : X où X est le champ manquant	Champs obligatoires non remplis
550	User exist Already	Utilisateur déjà existant

Réponse :



L'authentification se fait par l'adresse email et le mdp enregistré en MD5 :



The screenshot shows a REST client interface. The method is POST and the URL is http://apisupmagasin.ddns.net/user/auth. The 'Body' tab is selected, showing a JSON payload: {"Login": "corentin.cannie@wanadoo.fr", "Password": "azerty38", "RealyUser": true}. A red box highlights the JSON body, and a blue arrow points from the 'RealyUser' field to the 'RealUser' field in the adjacent code snippet.

```
POST http://apisupmagasin.ddns.net/user/auth
```

Params Authorization Headers (9) **Body** Pre-request Script Test

none form-data x-www-form-urlencoded **raw** binary GraphQL JSON

```
1 {
2   "Login" : "corentin.cannie@wanadoo.fr",
3   "Password" : "azerty38",
4   "RealyUser" : true
5 }
```

Login(String) = adresse  
Enregistré  
**RealUser**(bool) pour  
différencier les macs  
vrais users

Code	Message	Raison
400	Login Null. Stop it little pervers - _-#	Le champs LOGIN est vide
400	login not resolved. Try Again band of little green hacker. \ / 0益0 \ /	Erreur d'authentification

```
Body Cookies Headers (5) Test Results Status: 200 OK Time: 97 ms Size: 448 B Save Response
```

```
Pretty Raw Preview Visualize JSON
```

```
1 {  
2   "login": "corentin.cannie@wanadoo.fr",  
3   "password": "",  
4   "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjEwK5MjI0MjMsIm1zcyI6ImlnClF9fWFmYXppb19FMjAyMCIsImZlZC6ImZybzI5OW5uaW1kLmF0dC58",  
5   "reallyUser": true,  
6   "id": "25XCC58"  
7 }
```

## IDCustomer dans la base

The screenshot shows the Chrome DevTools Network tab with a selected POST request to `/api/auth/login`. The status bar indicates a `200 OK` response with a time of `178 ms`. The 'Body' tab is active, displaying the response in 'JSON' format:

```
{
  "Login": "SupMagasin",
  "Password": "Supinf0!",
  "ReallyUser": false
}
```

Toutes authentifications réussis ou pas est mis dans des logs classés par jour.  
Afin de repérer des éventuelles attaques ou break-force.

## V) Connexion Bluetooth

Dans le cahier des charges une localisation par Bluetooth était demandée à la suite de tests et comme expliqué plus haut, le test ne fut pas concluant.

### La connexion Bluetooth s'effectue en 2 temps :

- La reconnaissance Bluetooth
- L'identification au sein du magasin

#### a) *La reconnaissance Bluetooth*

Chaque téléphone des clients est enregistré dans la base de données dans le model phone comprenant le model du téléphone, sa marque, son numéro et surtout son adresse mac.

Cette adresse est unique pour chaque téléphone car elle permet d'identifier la puce réseau utilisé par le téléphone. Cette information est enregistrée lors de l'inscription par le biais du téléphone.

Tous les 15 secs, des bornes Bluetooth, dans notre projet des raspberry pi, scanne l'ensemble des téléphones pris dans le magasin et l'envois sur l'api via la route (/bluetooth).

Cette route demande un mot de passe, afin de sécuriser la connexion en cas d'interception de la route par un tiers non autorisé. Le choix de cette route est dû à une facilité à changer les bornes sans pour autant l'enregistrer dans le système.

Lorsque le scan est fait, il est envoyé dans sur l'api. Cette dernière l'inscrit dans un fichier temp.

Cette mise à jour est effectuée toutes les 15 secondes, d'une part par contrainte matérielle, d'autre part car 15 secs laissent le temps, en cas de défaillance de connexion de pouvoir réagir.

Code erreur spécifique :

Code erreur	Message	Raison
421	"Key not correspond. Are you Porcinet ?	Mauvais code
420	"Phones attributs is empty"	Listes vides

*b) L'identification au sein du magasin*

Lors de la connexion par l'appli, web ou mobile, une demande est envoyée sur la route /bluetooth/{Mac} qui va renvoyer un boolean indiquant la présence du téléphone dans le magasin. Si oui, elle ouvrira les portes de la partie panier, et achat. Sinon l'utilisateur ne pourra que consulter son compte ou son profil.

Code erreur spécifique :

Code Erreur	Message	Raison
425	"MAC Adress not found"	Le mac présenté n'est pas dans la liste