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 a 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 assuré 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 le 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 pensé 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 razberry pi 3B, montée par nos soins, pour vérifier et identifié 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 attitré. La réflexion c'est porté, qu'un seul fournisseur ne pouvait produite 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 grossîtes) avec dates d'achats et dates de péremptions pour les produits périssables (Alimentaire)

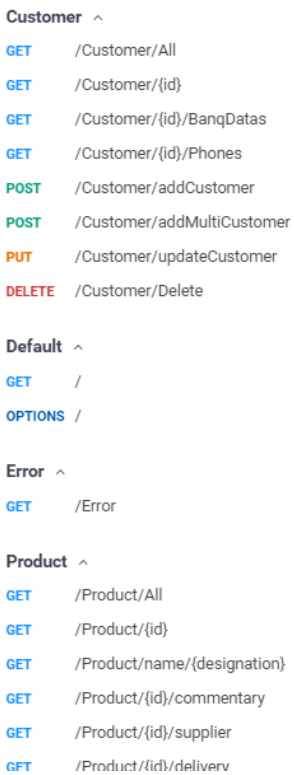
La table magasin quant à elle, dans ce POC rien n'était indiqué pour la présence de plusieurs magasins, donc nous avons préférer prendre de l'avance et anticiper des ouvertures futures. Elle comprend les données liées aux magasins ainsi que ses employées que nous affections à chaque magasin. Ces dernières 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

Voici un aperçu :



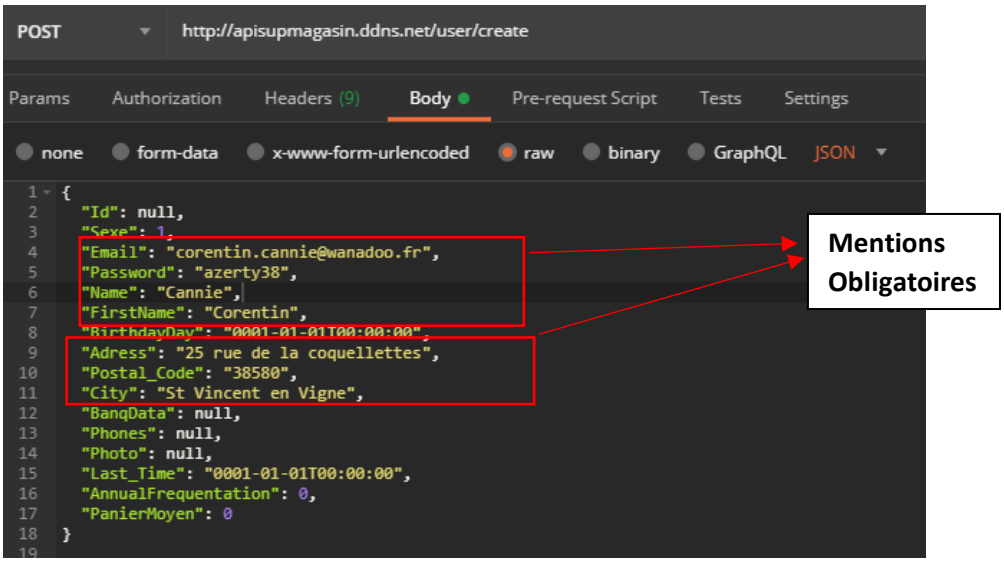
L'URL de l'api:

http://apisupmagasin.ddns.net

IV) L'authentification

L'authentification est enregistrée dans le Schéma Customer.

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



Code Erreur Spécifique :

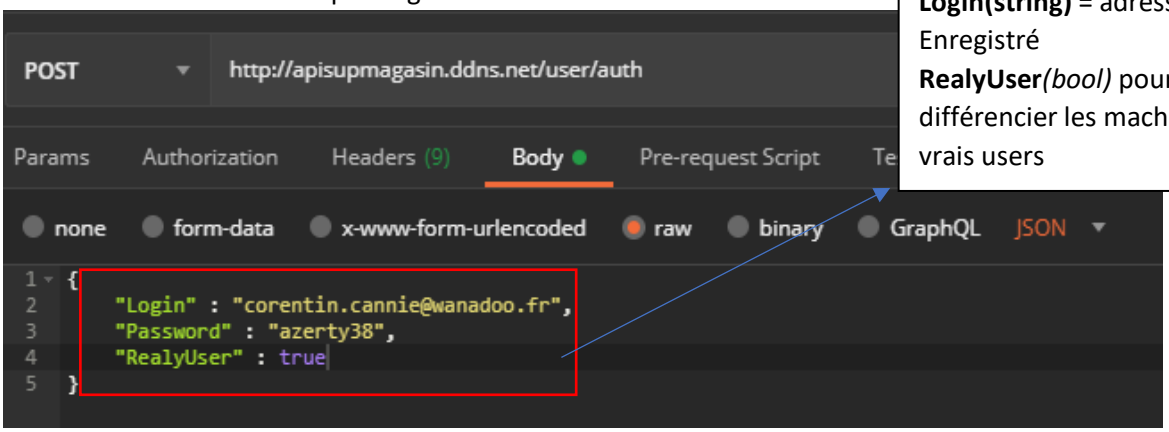
Code Erreur	Message	Raison
400	`Filds Missing : X où X est le champ manquant`	Champs obligatoires non remplis
550	`User exist Already`	Utilisateur déjà existant

Réponse :


```
{
    "login": "corentin.cannie@wanadoo.fr",
    "password": null,
    "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE1ODk5MjIwNzQsIm1zcyI6Il1N1cF9fTWFnYXppbl9fMjAyMCIsImF1ZCI6ImZyb250SW9uaWMifQ.gI7
    "realyUser": true,
    "id": "25XCC58"
}
```

b) L'authentification (http://apisupmagasin.ddns.net/user/auth) *POST*

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

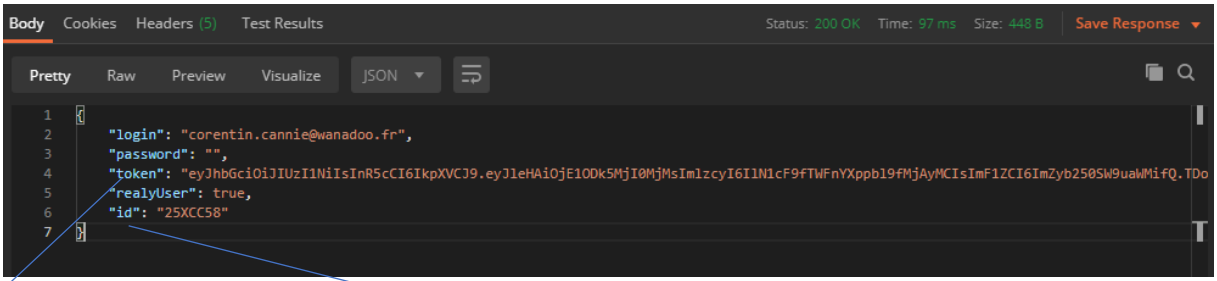


Login(string) = adresseEmail Enregistré
RealyUser*(bool)* pour différencier les machines et les vrais users

Code Erreur Spécifiques :

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. ヽ(ಠ益ಠ)ノ`	Erreur d'authentification

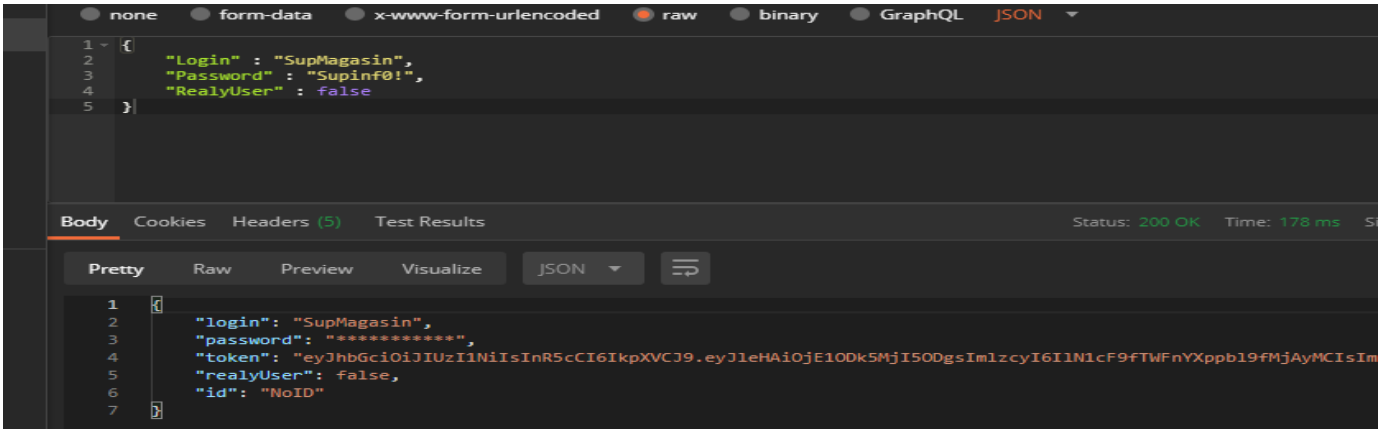
Réponse :



Token à utiliser pour les requêtes

IDCustomer dans la base

Utilisateur par default :



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

V) Connection Bluetooth