BTS SNIR

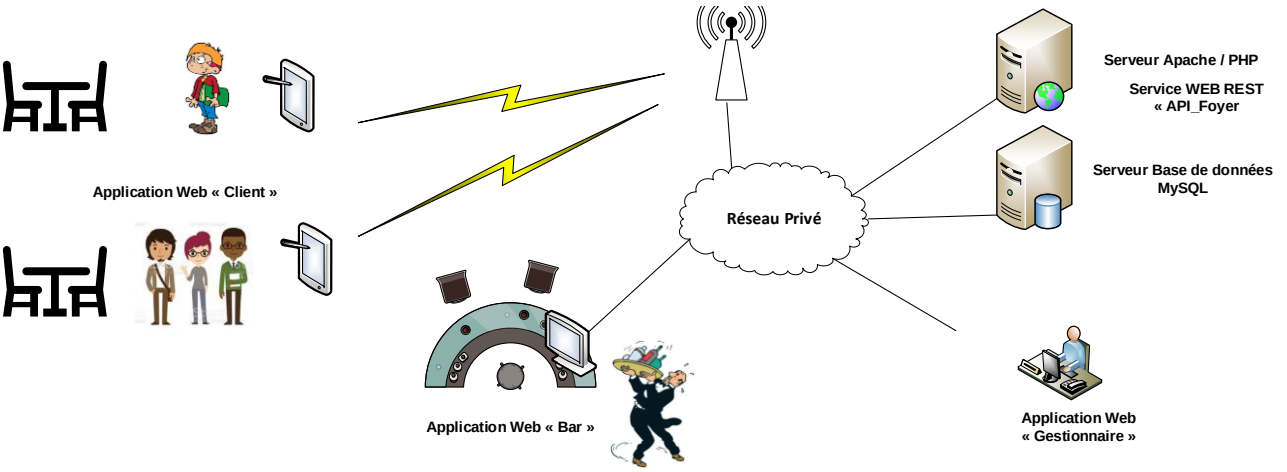
E6 – MINI-PROJET INFORMATIQUE

Dossier de présentation du sujet de mini projet

Lycée ou Centre de formation : Institut Lemonnier
Ville : Caen
Nom du mini projet : Bar foyer Institut Lemonnier

1. Présentation du mini projet

Configuration d'exploitation



Le projet consiste à développer trois applications web permettant la gestion de commandes au bar du foyer de l'Institut Lemonnier.

2. Expression du besoin

Afin d'éviter que les élèves qui fréquentent le foyer attendent devant le bar lorsqu'ils ont effectué une commande, le responsable du foyer a demandé à la section de BTS informatique d'imaginer une application permettant d'effectuer des commandes depuis les tables du foyer, sans déplacement.

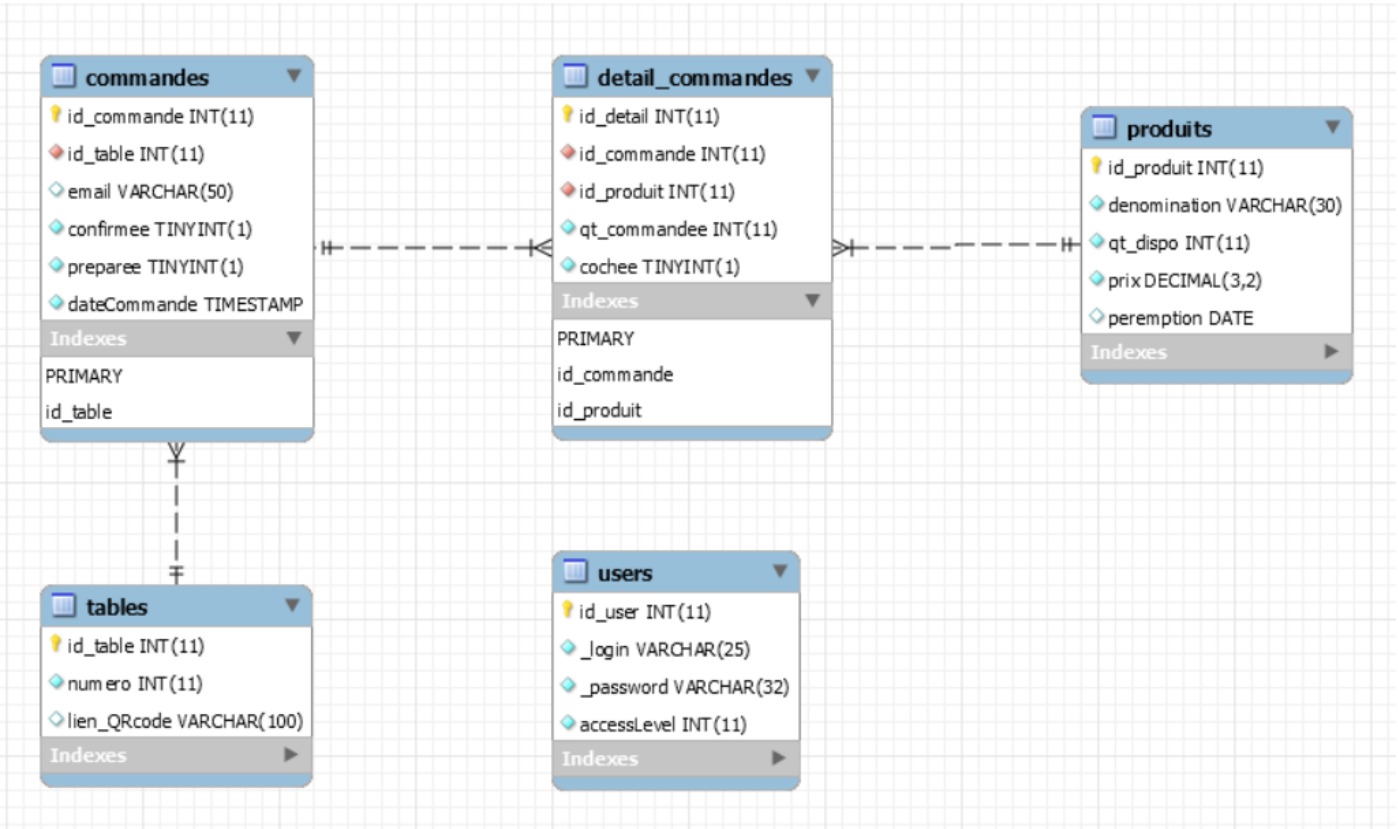
L'idée retenue consiste à étiqueter chaque table avec un QRcode permettant d'accéder à une page de commande en ligne, tout en authentifiant la table. Le consommateur, après avoir scanné le QRCode avec son téléphone, devra sélectionner les produits et les quantités qu'il désire commander et renseigner son mail (xxx @yyy.institutlemonnier.fr). Il recevra alors un lien pour valider sa commande.

Côté bar, les commandes s'afficheront au fur et à mesure. Chaque produit préparé sera coché sur la commande en cours et lorsque tous les produits seront cochés, le barman pourra valider la commande qui disparaîtra alors de l'affichage.

Le gestionnaire quant à lui pourra essentiellement gérer les stocks.

Le but de ce mini projet va donc être de développer :

- Une application **Web Client**, qui devra permettre à un consommateur de visionner la liste des produits disponibles au bar (boissons, confiseries) et d'effectuer une commande sans quitter sa place.
- Une application **Web Bar**, qui affichera au niveau du bar, toutes les commandes Clients non satisfaites. Une fois préparées, les commandes disparaitront de l'application.
- Une application **Web Gestionnaire** qui permettra de gérer les stocks (ajout/suppression de produits, quantités disponibles, date de péremption) et de gérer les QRcodes en cas de modification de l'agencement du foyer (ajout / suppression de tables)
- Une **API Web** qui fera le lien entre les applications Web à développer et une base de données mySql dont le schéma est fourni ci-dessous :



La base sera accessible à deux utilisateurs :

- "Barman", qui dispose du droit "select" sur toutes les tables, et du droit "update" sur les tables "commandes" et "detail_commandes",
- "Gestionnaire" qui dispose de tous les droits sur toutes les tables.

Les mots de passe respectifs enregistrés dans la BDD pour ces deux utilisateurs sont les Hash MD5 de Password1234 et Password12345.

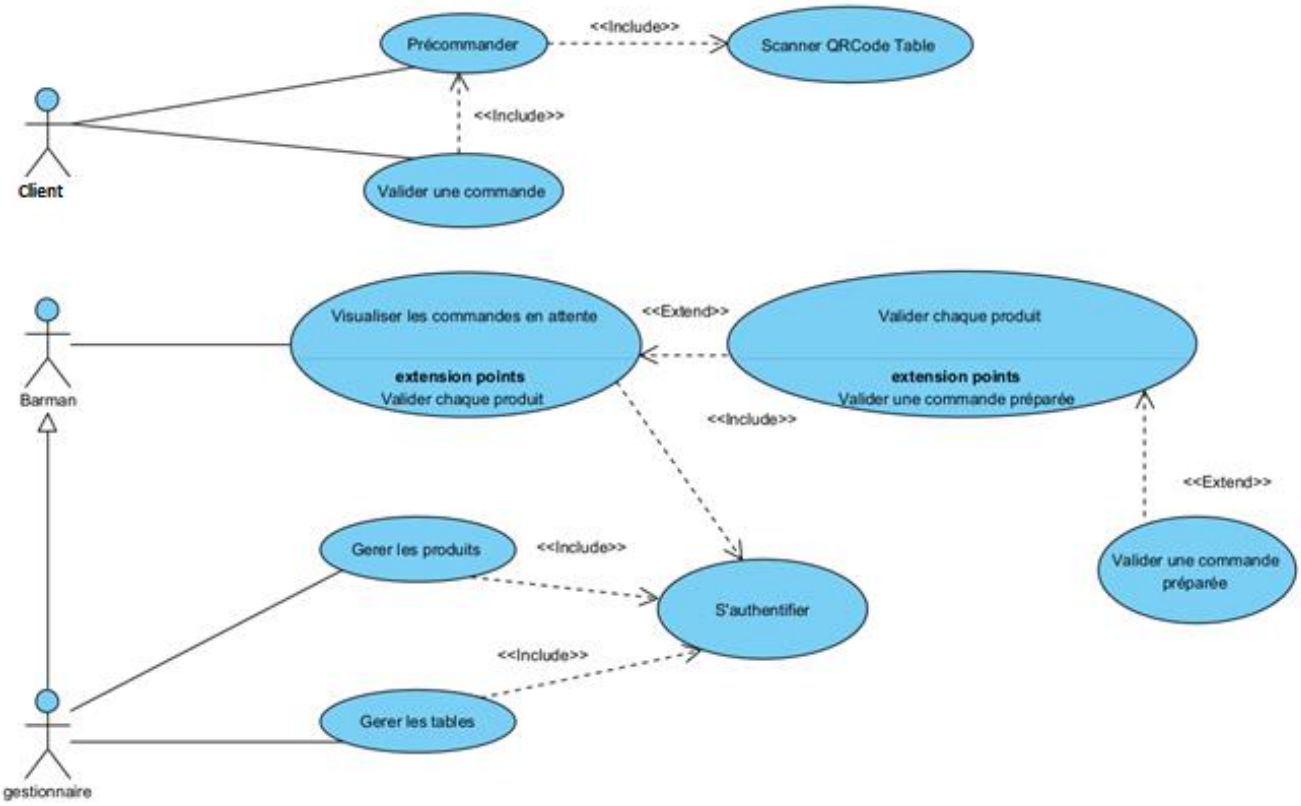
A remarquer que dans le cadre de ce mini-projet, chaque groupe disposera de sa propre base de données qui sera nommée "foyerbdd_gN" où N représente le numéro de groupe. Pour chaque groupe, l'URL d'accès à sa BDD sera de la forme :

http://localhost/foyerbdd_gN

Descriptif des tables :

- ➔ Table "users" : table indépendante qui contient les 2 utilisateurs présentés ci-dessus. Pour chacun de ces utilisateurs, un niveau d'accès est enregistré. Ce niveau pourra être utilisé par le service web pour déterminer si une méthode est bien accessible au demandeur.
- ➔ Table "produits" : contient les produits en stocks et leurs caractéristiques.
- ➔ Table "tables" : contient les informations sur les tables situées dans le foyer (numéro et lien servant à générer un QRcode)
- ➔ Table "commandes" : contient la commande et l'état de son traitement.
- ➔ Table "detail_commande" : contient, pour chaque commande, le détail des produits commandés.

Cas d'utilisations à satisfaire :



Cas d'utilisation " Scanner QRCode Table" :

Description : Chaque table dispose d'un QRcode fixé sur celle-ci. Ce QRCode contient l'identifiant de la table ainsi que l'url de la page permettant d'effectuer une commande. Le consommateur doit donc commencer par scanner ce QRCode avant de pouvoir précommander.

Acteur concerné : le client.

Cas d'utilisation " Précommander" :

Description : Après avoir scanné un QRCode sur sa table, le consommateur reçoit une page web affichant l'ensemble des produits disponibles au bar. Après avoir sélectionné ce qu'il veut commander, et la quantité, il renseigne son email et envoie sa commande. A ce stade, la commande est à l'état de précommande

Acteur concerné : le client.

Cas d'utilisation " Valider une commande" :

Description : Le service renvoie un lien cliquable à l'adresse indiquée lors de la précommande. Ce lien pointe vers la méthode du service web permettant de valider la précommande et contient un token avec une date de validité. Ce token est encodé par le serveur avec un chiffrement symétrique AES-256-GCM. Le client doit valider ce lien avant la fin de validité du token pour voir sa précommande transformée en commande.

Acteur concerné : le client.

Cas d'utilisation " S'authentifier " :

Description : Toutes les fonctionnalités Barman et Gestionnaire ne sont accessibles que sous réserve d'être authentifié. Après authentification, un token est délivré par le service. Ce token contient le niveau d'accessibilité (0 pour le gestionnaire, 1 pour le barman et -1 en cas de mauvaise authentification) et une date de validité. Ce token est encodé avec un chiffrement symétrique AES-256-GCM.

Acteurs concernés : le barman et le gestionnaire.

Cas d'utilisation " Visualiser les commandes en attente" :

Description : Dès qu'un consommateur à validé une commande, cette dernière apparaît dans l'application bar afin que le barman puisse la préparer

Acteur concerné : le barman.

Cas d'utilisation " Valider chaque produit" :

Description : Au fur et à mesure que le barman prépare une commande, il coche chaque produit prêt à être livré.

Acteur concerné : le barman.

Cas d'utilisation " Valider une commande préparée" :

Description : Lorsque tous les produits d'une commande sont cochés, le barman peut valider la commande qui disparait alors de l"application Bar

Acteur concerné : le barman.

Cas d'utilisation " Gérer les tables " :

Description : L'aménagement du foyer pouvant être appelé à changer, le nombre de tables ainsi que leurs numérotations doivent pouvoir être modifiés.

Acteurs concernés : Le Gestionnaire.

Cas d'utilisation " Gérer les produits " :

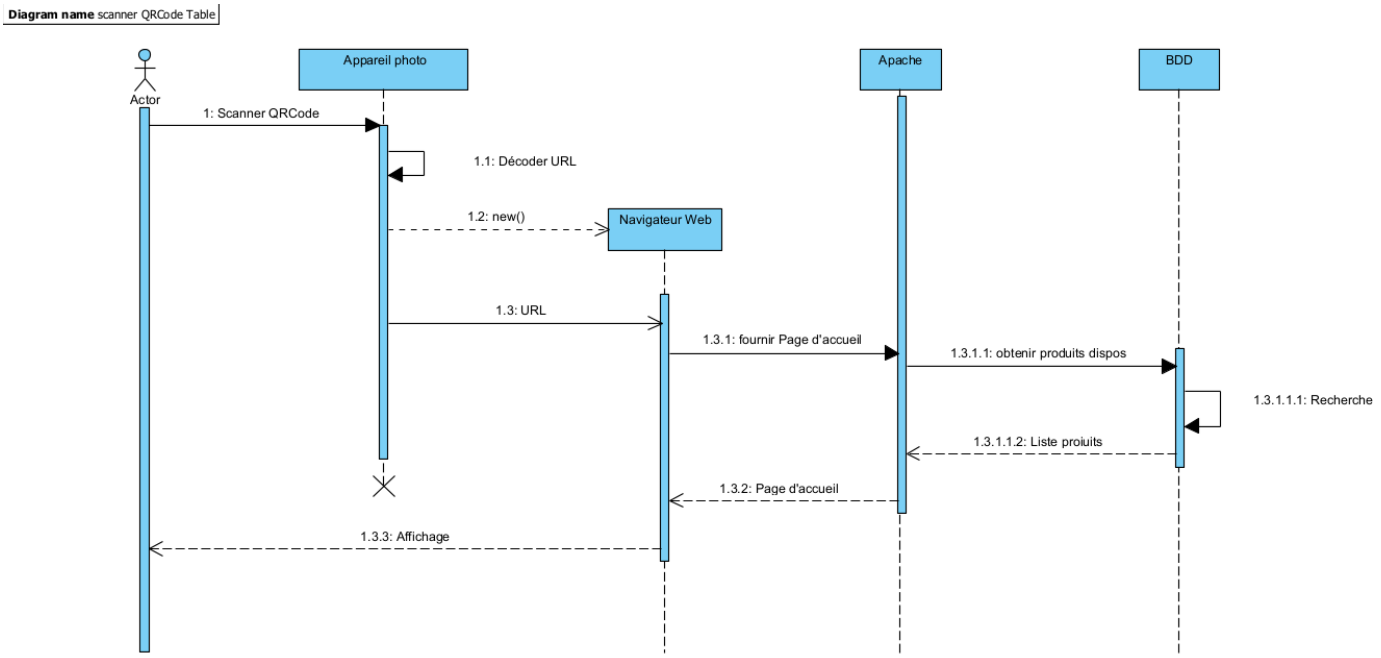
Description : Lorsque le gestionnaire refait les stocks, il doit pouvoir mettre les données à jour dans la BDD. Il en va de même s'il sort des produits périmés des stocks.

Acteurs concernés : Le Gestionnaire.

3. Conditions de réalisation, contraintes

3.1 Analyse

A titre d'exemple, on fournit le diagramme de séquence associé au cas d'utilisation "scanner le QRCode". Dans ce diagramme, les objets représentés s'entendent au sens large et non au sens de la POO. Un diagramme plus détaillé, mettant en évidences tous les objets impliqués dans la réalisation de ce cas d'utilisation, sera à fournir par l'étudiant en charge de cette partie. Les autres diagrammes associés aux autres cas d'utilisations seront eux aussi à fournir par les étudiants en charge de ces différents cas.



3.2 API

3.2.1 Appel des méthodes :

Les méthodes à développer dans l'API seront appelées par la page index.php dont la base est fournie et qu'il faudra compléter au fils des développements. Index.php fait appel à une classe Route intégralement fournie et codée dans le fichier route.php. Ainsi, l'appel à une méthode X se présentera sous la forme :

http://serveurlinux/~nomprenom/projetFoyer/index.php/X

Si l'URL existe, la méthode correspondante dans l'API sera appelée avec la méthode du protocole http indiquée (POST OU GET). A titre d'exemple, index.php contient le code permettant d'invoquer, via la méthode POST, la méthode authentication() d'un objet API:

```
Route::add('/authentication',function(){
    global $api;
    header('Content-Type:application/json');
    echo $api->authentication() ;
},'post');
```

Quelques explications :

Route :: add (…) : la page index.php fait appel à une classe "Route" de type "static". Il n'y a donc pas d'obet d'instancié. Ici on appelle simplement la méthode "add()" de cette classe. "add()" attend 3 paramètres: une url partielle , une fonction et une méthode http:

-1- URL

```
'/authentication'
```

L'URL permet de différencier les méthodes à appeler. Dans l'exemple donné, l'URL devrait donc être http://serveurlinux/~nomprenom/projetFoyer/index.php/authentication, mais la classe Route "parse" cette URL de sorte que seule la fin de l'url est à préciser (c'est pour cette raison que vous trouverez en dernière ligne du fichier index.php la ligne) :

Route::run('/~nomprenom/projetFoyer/index.php);

-2- Fonction

```
function()
{
	global $api;
	header('Content-Type:application/json');
	echo $api->authentication();
}
```

C'est l'appel à la méthode de l'API souhaitée. Dans l'exemple donné, la fonction (anonyme) commence par appeler la variable $api qui est en fait un objet (instance de API_Foyer). Or, en PHP on ne déclare pas une variable globale pour qu'elle soit visible dans les fonctions, mais c'est à la fonction d'indiquer qu'elle veut accéder à une variable "invisible". C'est le rôle du mot clé "global". On trouve ensuite un header qui indiquera au protocole http que les données transmises seront encodées au format JSON. Enfin, on appelle la méthode authentication dont le retour sera affiché via la fonction echo.

-3- Méthode

```
'post'
```

"post" est méthode du protocole http que l'on souhaite utiliser pour appeler notre méthode.

3.2.2 Retour des méthodes :

Tous les retours seront encodés en JSON. Il appartient aux équipes de concevoir leur API et d'en définir les contours : données fournies en entrée, méthode http utilisée, nom des variables, données récupérées en sortie.

3.2.3 Exemples :

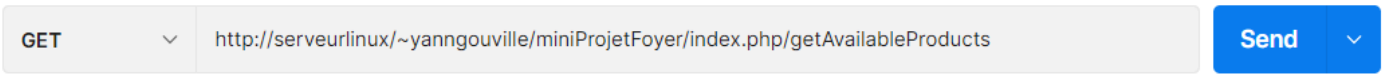
Un service web, accessible à l'adresse http://serveurlinux/~yanngouville/MiniProjetFoyer/index.php, est disponible pour que chaque étudiant puisse tester une requête liée à la partie qu'il devra traiter.

Afin de visualiser les retours, les méthodes ont été appelées avec le logiciel Postman (en version "free") que vous pourrez également utiliser pour vous aider à développer votre API. Ce logiciel permet notamment de choisir la méthode à utiliser pour la requête (GET, POST, etc.) et d'indiquer les paramètres passés.

Exemple 1 : Récupérer la liste de tous les produits disponibles à la vente

Méthode de l'API : getAvailableProducts()

Requête :



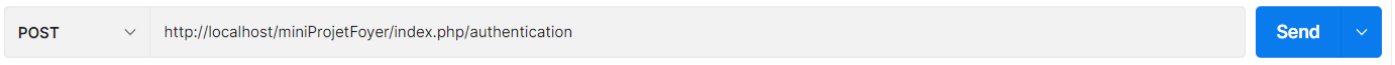
Réponse :


```
{
   "produitsDispos":
      [
        {"id_produit":"1","denomination":"café expresso","prix":"0.40","qt_dispo":"90"},
        {"id_produit":"3","denomination":"coca-cola 33cl","prix":"0.90","qt_dispo":"90"},
        {"id_produit":"4","denomination":"orangina 33 cl","prix":"0.90","qt_dispo":"70"},
        {"id_produit":"5","denomination":"kinder bueno","prix":"0.60","qt_dispo":"30"},
        {"id_produit":"6","denomination":"chocolat chaud","prix":"0.50","qt_dispo":"88"},
        {"id_produit":"7","denomination":"ice tea","prix":"0.90","qt_dispo":"72"},
        {"id_produit":"8","denomination":"snickers","prix":"0.60","qt_dispo":"50"}
      ]
}
```

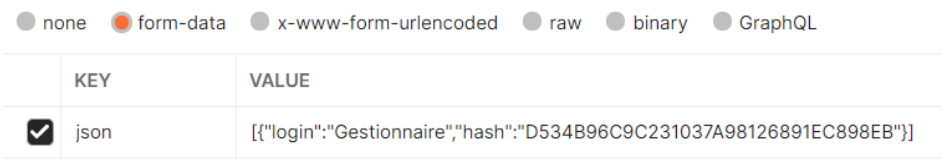

Exemple 2 : S'authentifier en tant que Gestionnaire

Méthode de l'API : authentication

Requête :



Paramètres:



Réponse:


```
{
 "identification":
   {
        "accessLevel": "0",
         "token": "TpyLCaE+DnC6y9HZd7bdyRYSULVq1xABk8vySMGUIKu+IY9jYUS9JLCSba8H9pH
cAomgv0smI2KqXrwUnjQMtuRI5P3B+kYwoFL40poIgf3YGCJ+RvDQWu5BlDdohEAi0R6IYHlPRm+3h
7lFjNAE7VUbYTtIfJlqpcRKa5Ms3lY="
   }
}
```

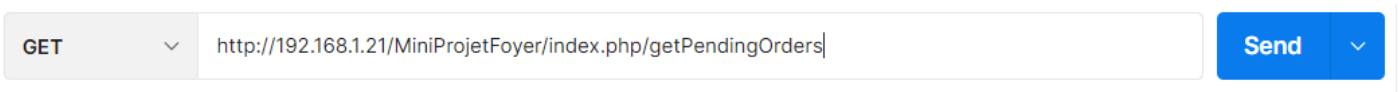

En cas de mauvaise identification :

```
{
   "identification": "-1"
}
```

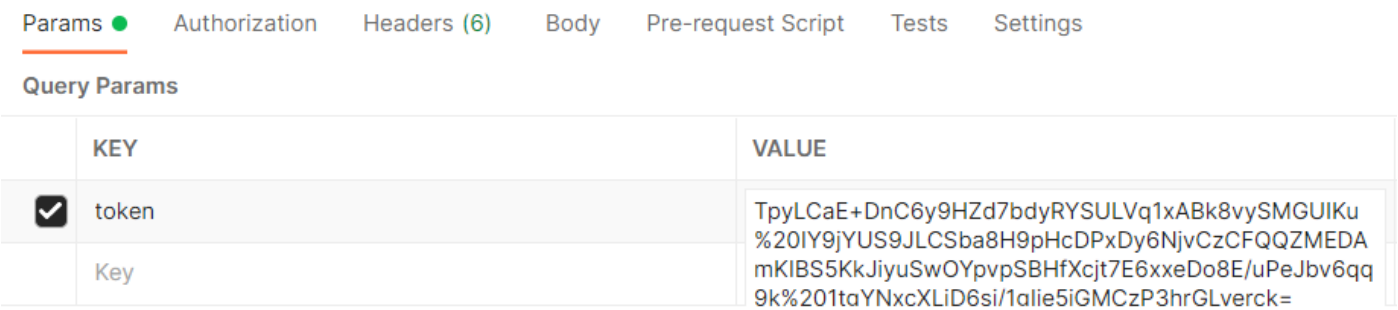

Exemple 3 : Voir les commandes en attentes

Méthode de l'API : getPendingOrders()

Requête :



Paramètres:



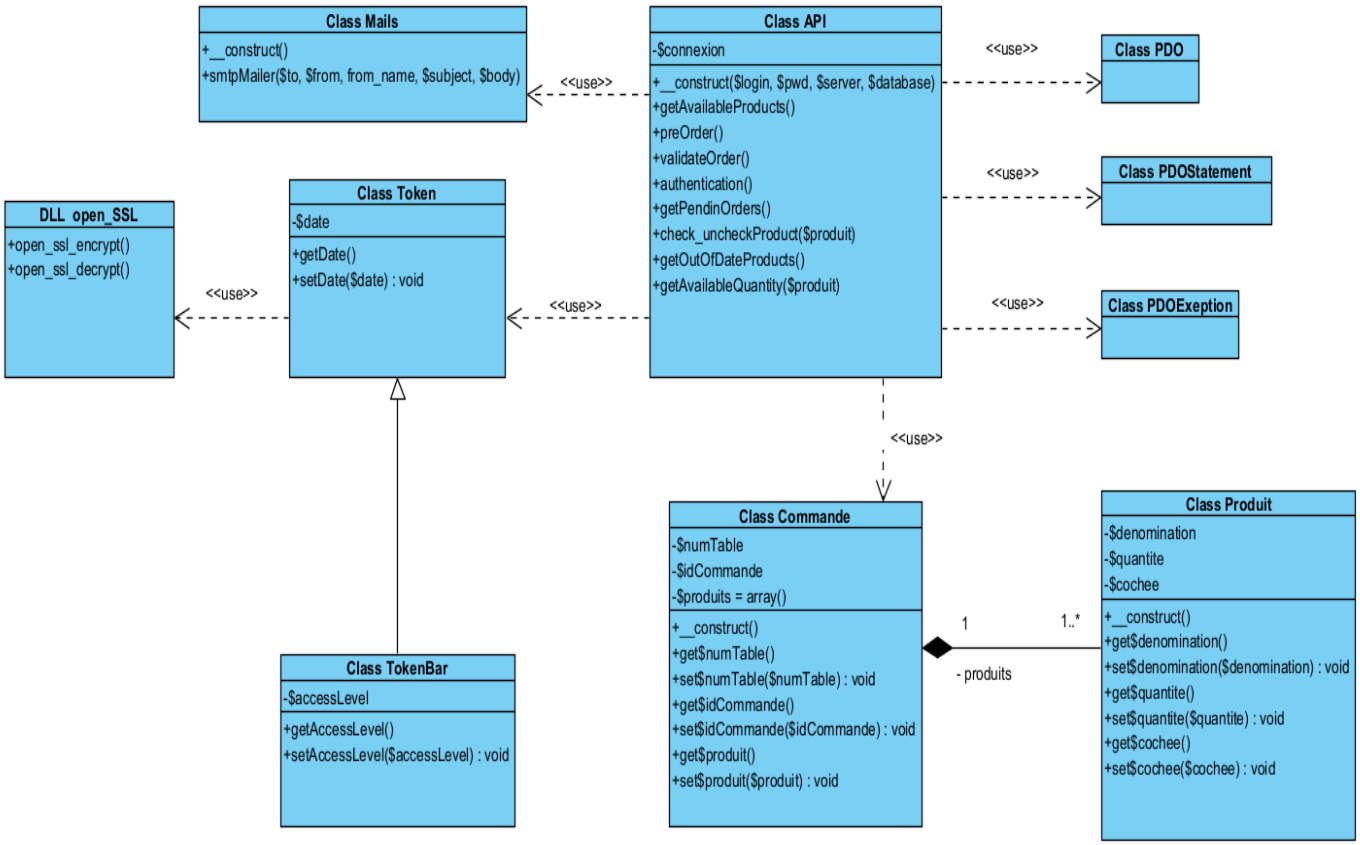
Réponse:


```
{
"cmdEnCours":
   [
      {"numTable":"8","idCommande":"44","produit":
         [
            {"denomination":"café expresso","quantite":"4","cochee":"0"},
            {"denomination":"chocolat chaud","quantite":"1","cochee":"0"}
         ]
      },
      {"numTable":"6","idCommande":"45","produit":
         [
            {"denomination":"kinderbueno","quantite":"2","cochee":"0"},
            {"denomination":"snickers","quantite":"2","cochee":"0"}
         ]
      },
      {"numTable":"2","idCommande":"46","produit":
         [
            {"denomination":"coca-cola 33cl","quantite":"1","cochee":"0"},
            {"denomination":"orangina 33 cl","quantite":"2","cochee":"0"},
            {"denomination":"ice tea","quantite":"2","cochee":"0"}
         ]
      }
   ]
}
```

A noter qu'en cas d'erreur dans l'appel de chacune des méthodes (mauvais nom de variable par exemple), ces dernières retournent un message d'erreur sous la forme

```
{
    "Erreur": "message d'erreur"
}
```

3.2.4 Diagramme "minimum" des classes à mettre en jeu dans l'API :



4. Exigences qualité à respecter

4.1. Exigences qualité sur le produit à réaliser

Les pages web devront être conviviales et ergonomiques. Les formulaires seront tous proposés au client pour accord.

L'application devra être :

- Robuste, en assurant le contrôle de la validité des données de part et d'autre du système,
- Structurée en favorisant le développement modulaire afin de faciliter la réutilisation des modules,

4.2. Exigences qualité sur le développement

Le développement reposera sur la modélisation UML 2 en utilisant l'outil d'analyse "Visual Paradigm".

Le codage sous Visual Studio Code sera réalisé à l'aide des langages html 5, CSS 3, javaScript (toutes fonctionnalités compatibles ECMAScript 10) et PHP 7.4 mini. L'utilisation des "Grid CSS" est fortement recommandée.

Chaque groupe enregistrera son projet sur un dépôt GitHub privé et invitera le professeur en charge de ce groupe. Chaque étudiant travaillera sur sa propre branche (et sous branches si nécessaire). La branche principale intégrera l'ensemble du projet et sera complétée chaque fois qu'une fonctionnalité stable sera développée par l'un ou l'autre des étudiants.

4.3. Exigences qualité sur la livraison

Les produits livrables du mini projet sont :

- Les applications **Web API, Web Client, Web Bar, Web Gestionnaire**

- La documentation du mini projet qui doit être composée :
  - ✓ D'un dossier technique comprenant les dossiers de spécification, de conception préliminaire, de conception détaillée.
  - ✓ D'annexes séparées comprenant le code et les tests.

5. Répartition des fonctions ou des cas d'utilisation

Le mini projet est à réaliser par équipes de 3 étudiants, le tableau suivant donne la répartition du travail à réaliser pour chaque étudiant :

Étudiant	Module ou cas d'utilisation
Étudiant 1 (E1) :	**Application Web Client** ✓ Cas d'utilisation **" Scanner QRCode Table "** ✓ Cas d'utilisation "**Précommander"** ✓ Cas d'utilisation "**Valider une commande"** **Service WebRest** Toutes les requêtes permettant de satisfaire les cas d'utilisations de l'application web Client
Étudiant 2 (E2) :	**Application Web Barman** ✓ Cas d'utilisation "**Visualiser les commandes en attente"** ✓ Cas d'utilisation "**Valider chaque produit"** ✓ Cas d'utilisation "**Valider une commande préparée"** ✓ Cas d'utilisation **" S'authentifier "** (en lien avec l'étudiant E3 mais développé par ce dernier) **Service WebRest** Toutes les requêtes permettant de satisfaire les cas d'utilisations de l'application web Barman

Étudiant 3 (E3) :	**Application Web Gestionnaire** ✓ Cas d'utilisation **" S'authentifier "** (en lien avec l'étudiant E2) ✓ Cas d'utilisation "**Gérer les produits"** ✓ Cas d'utilisation **" Gérer les tables"** **Service WebRest** Toutes les requêtes permettant de satisfaire les cas d'utilisations de l'application web Gestionnaire