Mettre en place projet NoSQL

Nous avons maintenant mis en place tous les éléments pour la mise en route du projet NoSQL, nous allons voir la technique maintenant

vsCode

Dans VSCode, nous allons déjà créer deux répertoires (src, vendor), src sera l'endroit où notre code sera placé alors que vendor servira de stockage pour les dépendances que composer installera sur notre projet.

Puis nous allons créer un fichier qui sera frère des deux répertoires (au même niveau),

composer.json est le fichier où composer va stocker toutes les informations sur les dépendances installées (leur nom, version etc), il utilisera ce fichier dès qu'il sera importé dans un autre projet avec la commande composer install ou composer update

Premieres commandes

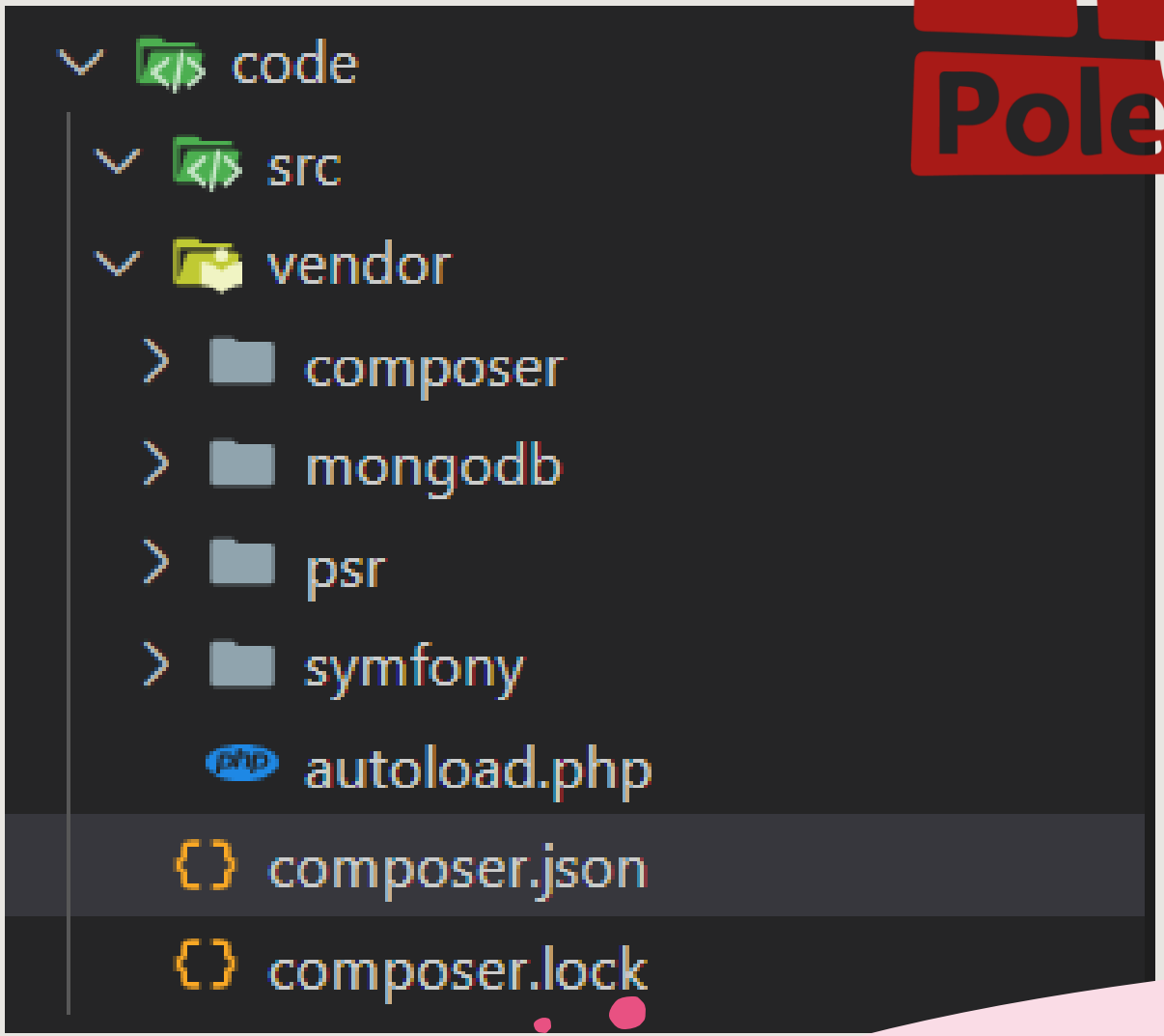
Nous allons installer la bibliothèque de mongoDB pour PHP avec la commande suivante : mongoDB/mongoDB --ignore-platform-reqs

Puis ext-mongodb –ignore-platform-reqs (ext-mongodb est une extension php de mongoDB qui lui permettra d'intéragir directement avec la BDD), que nous avons téléchargée sur pecl et mis le DLL dans php

--ignore-platform-reqs (ignore les obligations de versions)

Où en sommes-nous ?

Voici à quoi devrait ressembler votre projet à ce moment



Premiere requête

Nous allons créer dans src
un fichier index.php pour nos
premières requêtes (insert et find)

(voir le fichier index.php fourni
directement)

Resultat ---------------->

Voici les informations des employés : Mitra La Meilleure

Les filtres de requêtes

- Egalité : $result = $collection->find([nom => 'mitra']);
- $gt/$lt/$gte/$lte (plus grand/plus petit/ plus grand ou egal/ plus petit ou egal) : $result = $collection->find(['age' => ['$gt' => 25])
- $and/$or (ET/OU) : $result = $collection->find(['$and' => [['age' => ['$gt' => 25]],['sexe' => 'F']
- $not (pas) : $result = $collection->find(['age' => ['$not' => ['$gt' => 30]]]) (retourne tous les resultats dont l'âge n'est pas supérieur à 30)
- $nor (ni ni) : $result = $collection->find(['$nor' => [['age' => ['$lt' => 20]],['age' => ['$gt' => 60]]]]) (retourne tous les résultats dont l'âge n'est ni inférieur à 20 ni supérieur à 60;

Suite

- $in (soit) : $result = $collection->find(['sexe' => ['$in' => ['M','F']]]) (retourne les résultats où 'sexe' est soit M soit F)
- $nin (not in/ pas dans) : $result = $collection->find(['sexe' => ['$nin' => ['M']]]) (retourne les résultats où 'sexe" n'est pas M
- $regex : $result = $collection->find(['prenom' => ['$regex' => '^L']]) (Retourne tous les prénoms qui commencent par L
- $exists (existe) : $result = $collection->find(['surnom' => ['$exists' => true]]) (Retourne les résultats où le champ surnom existe)
- Champs imbriqués : $result = $collection->find(['adresse.ville' => 'Paris']) (Retourne les résultats où le champ ville dans l'objet adresse est paris