Mise en place de l’interface de migration de données API

# Environnement de travail

## Installations

* Installation de VirtualBox dans sa version 5.2 en laissant les options par défauts
* Installation de Vagrant dans sa version actuelle, 2.2.19 en laissant les options par défauts
* Installation de Git Bash depuis Git for Windows dans sa version 2.35.1.2 avec MingW
* Installation de l’IDE PhpStorm dans sa version Trial de 30J

## Récupération des différents dossiers

Pour cette mission, plusieurs environnements ont du être reçupéré sur github et gitlab après avoir récupérer les droits d’accès

### Depuis gitlab

* Le dossier api contenant
* Le dossier web2 D6lts contenant la configuration par défaut des différents environnements

### Depuis github

* Web2.5 qui est une version ultérieure du dossier web2 D6lts sur lequel le travail va s’effectuer. Certaines configurations dépendent

Ces récupérations se sont effectuées depuis la console GitBash par git clone tels que :

|  |
| --- |
| git clone https://github.com/xxxx/nomdossierremote.git nomdossierlocal |

Modification des logins de connexion en local dans web 2.5

App, etc, modules, local.xml

Login et mdp sont dans keypass

## Développement du WS

Depuis laminas framework suivi tuto dotec API en lecture GET uniquement

Api, module, magento, V1, REST,

Creation du WS FT pour fiche technique FT

Cretaion de FTMAPPER

## Récupération des bases existantes

Dans local.php

'adapters' => array(  
 'AZ\_read' => array(  
 'driver' => 'Pdo\_Mysql',  
 'database' => 'aromazone\_prod',  
 'username' => 'nma',  
 'password' => 'kaOiUnGsIuiJ6wY4VeF1',  
 'hostname' => '192.168.1.115',  
 'charset' => 'utf8',  
 ),

[…]

'API' => array(  
 'driver' => 'Pdo\_Mysql',  
 'database' => 'api',  
 'username' => 'nma',  
 'password' => 'kaOiUnGsIuiJ6wY4VeF1',  
 'hostname' => '192.168.1.115',  
 'charset' => 'utf8',  
),

## Modification dans module.php

Permet de récupérer les configurations

Classe qui implémente implements ApiToolsProviderInterface

'magento\V1\Rest\Ft\FtMapper' => function ($sm) {  
 $config = $sm->get('config');  
 $adapter = $sm->get('AZ\_read');  
 return new \magento\V1\Rest\Ft\FtMapper($config, $adapter);  
},

## Modification local.php : api config autolooad local.php

En tant que configuration pour couch db dans local.php

'couchdbtest' => array(  
 'findproduct' => array(  
 'uri' => 'http://10.0.2.6:5984/aromazone\_prod',  
 'host' => '10.0.2.6',  
 'table' => 'aromazone\_prod',  
 'port' => '5984',  
 'username' => 'vagrant',  
 'password' => 'vagrant',  
 'headers' => array(  
 'Accept' => 'application/vnd.magento.v1+json',  
 'Content-Type' => 'application/json'  
 ),  
  
 )  
)

'authentication' => array(  
 'adapters' => array(  
 'couchdb' => array(  
 'adapter' => \Laminas\ApiTools\MvcAuth\Authentication\HttpAdapter::class,  
 'options' => array(  
 'accept\_schemes' => array(  
 0 => 'basic',  
 ),  
 'realm' => 'ft api',  
 'htpasswd' => 'data/preprod\_users.htpasswd',  
 ),  
 ),

## CReationde get() à partir Web Sevice WS dans Tools api mmangeto src REST TOOLS

public function get($url, $data,$headers = array())  
{  
 // Load config and get central WS URL  
 $magento = $this->config->getConfig('magento');  
 if (isset($magento['ws'])) {  
 $ws = $magento['ws'];  
 $client = new Client($ws['uri'].$url, array(  
 'adapter' => 'Laminas\Http\Client\Adapter\Curl',  
 'useragent' => 'PROTECTED AZ label'  
 ));  
 if(empty($headers)){  
 $headers = array(  
 'Accept' => 'application/vnd.pos.v2+json',  
 'Content-type' => 'application/vnd.pos.v2+json'  
 );  
 }  
 $headers['Authorization'] = $client->encodeAuthHeader($ws['username'], $ws['password']);  
 $client->setHeaders($headers);  
 $client->setMethod('GET');  
  
 // Prepare and POST field "data" only with increment\_id and so on...  
 $data2send = Json::*encode*($data);  
  
 $client->setRawBody($data2send);  
 // POST data to REST client and get the result  
 $response = $client->send();  
  
 // parse result and prepare output  
 if ($response->isSuccess()) {  
 $responseContent = Json::*decode*($response->getBody());  
 } else {  
 $responseContent = Json::*decode*($response->getBody());  
 print\_r($response->getBody());  
 die;  
 }  
 }  
 return $responseContent;  
}

Dans FTRessource :

Mise en place du constructeur et varaible ampper

*/\*\*  
 \* Mapper variables  
 \*/*protected $mapper;  
  
  
*/\*\*  
 \* Constructor FtMapper  
 \*  
 \** ***@param*** *$mapper  
 \*/*public function \_\_construct($mapper){  
 $this->mapper = $mapper;  
}

Modification des fonctions implémentés pour get alla et getone

*/\*\*  
 \* Fetch one resource  
 \*  
 \** ***@param*** *mixed $id  
 \** ***@return*** *ApiProblem|mixed  
 \*/*public function fetch($id)  
{  
 return $this->mapper->fetchOne($id);  
}  
  
*/\*\*  
 \* Fetch all or a subset of resources  
 \*  
 \** ***@param*** *array $params  
 \** ***@return*** *ApiProblem|mixed  
 \*/*public function fetchAll($params = [])  
{  
 return $this->mapper->fetchAll($params);  
}

Dans ftRessource deux méthodes nous intéresse

* Fetch(id) qui renvoir la valeur de Mapper fetcHONe (id)
* fetchAll(param = [])qui renvoie la valeur de via mapper fatchALL(param)

## Mise en place des connection pour test vers postman vers l’api produt déjà existante

<https://10.0.0.14/product>

generation de login et mot de passe via :

<https://www.infowebmaster.fr/outils/crypter-htpasswd.php>

pour le dossier data.htpassword

## Creation de fetchOne via requête GET

Dans ftMapper

Verification via postman que le ws mis en place recupere bien 1 objet selon son id

A partir de cela, il faut reçupprer la structure des fiches techniques qui servira de modele pour l’envoi de ces données selon id

## Récupération de la structure des fiches techniques

Dans heidsql, on recheche les fiches techniques

A base de jointures, il faut récupérer la structure de FT

type\_name = 'fiche\_technique'

table concernées

drupal\_content\_group\_fields

drupal\_content\_group

drupal\_content\_node\_field

drupal\_content\_node\_field\_instance

|  |
| --- |
| SELECT group\_type, gpF.group\_name, ndfi.field\_name, parent, gp.label AS 'gp.label', ndfi.label AS 'ndfi.label', module, widget\_module, 'type', widget\_type, description    FROM drupal\_content\_node\_field\_instance ndfi  JOIN drupal\_content\_node\_field ndf ON (ndfi.field\_name = ndf.field\_name)    LEFT JOIN drupal\_content\_group\_fields gpF ON (ndfi.field\_name = gpF.field\_name)  LEFT JOIN drupal\_content\_group gp ON (gpF.group\_name = gp.group\_name)    WHERE ndfi.type\_name = 'fiche\_technique'  ORDER BY ndfi.weight, gp.weight |

## CReation de methode de récupération de la structure dans FtMApper.php

<?php  
namespace magento\V1\Rest\Ft;  
use Application\Tools\Config;  
use Application\Tools\Ws;  
use Application\Tools\Html;  
  
use Zend\Db\Adapter\AdapterInterface;  
use Zend\Db\Adapter\Adapter;  
  
use ZF\ApiProblem\ApiProblem;  
  
  
  
//use Zend\Db\Sql\Sql;  
//use Zend\Db\Sql\Select;  
  
use Laminas\Json\Json;  
use Laminas\Http\Client;  
use Laminas\Db\Sql\Sql;  
use Laminas\Db\Sql\Select;  
use Laminas\Http\Reponse;  
  
class FtMapper  
{  
  
 protected $adapter\_azr,  
 $adapter\_api,  
 $config,  
 $result,  
 $DrupalStructure = array(),  
 $Etiquettes\_other,  
 $Etiquettes\_null,  
 $Etiquettes\_none,  
 $data\_gather,  
 $data\_gather\_data,  
 $data\_couchDb,  
 $data\_drupal\_term\_property,  
 $data\_drupal\_term\_inpratice;  
  
 const *DRUPAL\_STRUCTURE\_WHERE\_NONE* = ['ndfi.type\_name'=>'fiche\_technique', 'parent' => ''];  
 const *DRUPAL\_STRUCTURE\_WHERE\_NULL* = ['ndfi.type\_name'=>'fiche\_technique', 'parent' => null];  
 const *DRUPAL\_STRUCTURE\_WHERE\_OTHER* = ['ndfi.type\_name'=>'fiche\_technique', 'parent != \'\''];  
  
 const *property\_bloc\_caract* = 'field\_ft\_caracteristique';  
 const *property\_bloc\_propr* = 'field\_ft\_propriete';  
 const *property\_bloc\_proprdesc* = 'field\_ft\_desc\_propriete';  
 const *property\_bloc\_proprWeight* = 'field\_ft\_prop\_weight';  
 const *STR\_FT\_PRESENTATIONS* = 'str\_ft\_presentation';  
  
  
 */\*\*  
 \* Constructeur du WS Ft Fiche Technique  
 \*/* public function \_\_construct($config, AdapterInterface $adapter\_azr, AdapterInterface $adapter\_api){  
  
 $getconfig = new Config($config);  
 $this->config = $getconfig->getConfig('couchdb');  
  
 $this->adapter\_azr = $adapter\_azr;  
 $this->adapter\_api = $adapter\_api;  
 }  
  
  
  
 public function fetchAll($param){  
 if(isset($param)){  
 $this->result = $this->get(null, $param);  
 }  
 return $this->result;  
 }  
  
  
 */\*\*  
 \* Methode Principal de gestion des données  
 \*  
 \** ***@param*** *$id  
 \** ***@return*** *mixed  
 \*/* public function fetchOne($id)  
 {die('e');  
 if(isset($id)){  
  
 ///  
 // Get Drupal structure like  
 $this->Etiquettes\_none = iterator\_to\_array($this->selectStructureDrupal(self::*DRUPAL\_STRUCTURE\_WHERE\_NONE*));  
 $this->Etiquettes\_null = iterator\_to\_array($this->selectStructureDrupal(self::*DRUPAL\_STRUCTURE\_WHERE\_NULL*));  
 $this->Etiquettes\_other = iterator\_to\_array($this->selectStructureDrupal(self::*DRUPAL\_STRUCTURE\_WHERE\_OTHER*));  
 //$this->DrupalStructure = $this->makeDrupalStructure();  
  
 // Get Drupal terms  
 $this->data\_drupal\_term\_property = iterator\_to\_array($this->selectDrupalTermData(4));  
 $this->data\_drupal\_term\_inpratice = iterator\_to\_array($this->selectDrupalTermData(3));  
  
 // Get data\_drupal\_term as associative array  
 $this->data\_drupal\_term\_property = $this->getAssociativeArray($this->data\_drupal\_term\_property);  
 $this->data\_drupal\_term\_inpratice = $this->getAssociativeArray($this->data\_drupal\_term\_inpratice);  
  
  
 ///  
 // Get Data from CouchDb product  
 $this->data\_couchDb = $this->get($id, null);  
 if($this->data\_couchDb instanceof ApiProblem){  
 return $this->data\_couchDb;  
 }  
  
 $this->data\_couchDb= $this->get($id, null);  
  
 $data\_drupal\_sku = $this->data\_couchDb['sku'];  
 $this->combine\_structure\_presentation();  
 $this->combine\_structure\_property();  
 $this->combine\_structure\_inpratice();  
  
  
 // IF HE  
 // Get one Data from API gather with sku  
 $this->data\_gather = $this->selectOneDataGather($data\_drupal\_sku);  
 $this->data\_gather = $this->selectOneDataGather($data\_drupal\_sku);  
 if( $this->data\_gather != null){  
 $this->data\_gather = iterator\_to\_array($this->data\_gather);  
 foreach($this->data\_gather as $item){  
 $gather\_ar = $item;  
 }  
 $this->data\_gather\_data = Json::*decode*($gather\_ar['data'], true);  
  
 $this->toFix\_presentation();  
 $this->toFix\_property();  
 $this->toFix\_inpratice();  
 }  
  
  
 // Combine Couchdb with Structure  
 //$combine\_structure\_couchdb = $this->combine\_structure\_couchdb();  
  
 // Combine Structure with gather  
 if(isset($gather\_ar['data'])){  
  
 }  
  
 //die('end');  
  
 unset($this->data\_couchDb['\_id'],$this->data\_couchDb['\_rev']);  
 return $this->data\_couchDb;  
 }  
 return "id not found" ;  
 }  
  
 protected function combine\_structure\_presentation(){  
 $this->data\_couchDb['custom\_index\_data']['node'][self::*STR\_FT\_PRESENTATIONS*] = array(  
 'field\_ft\_odeur' => !empty($this->data\_couchDb['custom\_index\_data']['node']['field\_ft\_odeur']) ? implode($this->data\_couchDb['custom\_index\_data']['node']['field\_ft\_odeur']['value']): '',  
 'field\_ft\_chemotype' => !empty($this->data\_couchDb['custom\_index\_data']['node']['field\_ft\_chemotype']) ? implode($this->data\_couchDb['custom\_index\_data']['node']['field\_ft\_chemotype']['value']): '',  
 'field\_ft\_accroche' => ''  
 ) ;  
 $this->data\_couchDb['custom\_index\_data']['node'][self::*STR\_FT\_PRESENTATIONS*]['field\_ft\_accroche\_gather'] = '';  
 unset($this->data\_couchDb['custom\_index\_data']['node']['field\_ft\_odeur'],$this->data\_couchDb['custom\_index\_data']['node']['field\_ft\_chemotype'] );  
 }  
 protected function toFix\_presentation(){  
  
 foreach($this->data\_couchDb['custom\_index\_data']['node'][self::*STR\_FT\_PRESENTATIONS*] as $key => $value) {  
  
 $this->data\_couchDb['custom\_index\_data']['node'][self::*STR\_FT\_PRESENTATIONS*]['field\_ft\_accroche'] = $this->data\_gather\_data['Intro']['Accroche'];  
 if(empty($this->data\_couchDb['custom\_index\_data']['node'][self::*STR\_FT\_PRESENTATIONS*]['field\_ft\_odeur'])||isset($this->data\_couchDb['custom\_index\_data']['node'][self::*STR\_FT\_PRESENTATIONS*]['field\_ft\_odeur'])){  
 $this->data\_couchDb['custom\_index\_data']['node'][self::*STR\_FT\_PRESENTATIONS*]['field\_ft\_odeur'] = $this->data\_gather\_data['Intro']['Odeur'];  
 }  
 if(empty($this->data\_couchDb['custom\_index\_data']['node'][self::*STR\_FT\_PRESENTATIONS*]['field\_ft\_odeur'])|| isset($this->data\_couchDb['custom\_index\_data']['node'][self::*STR\_FT\_PRESENTATIONS*]['Chémotype'])){  
 $this->data\_couchDb['custom\_index\_data']['node'][self::*STR\_FT\_PRESENTATIONS*]['field\_ft\_chemotype']= $this->data\_gather\_data['Intro']['Chémotype'];  
 }  
  
 }  
 }  
  
 protected function combine\_structure\_property(){  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'] = array();  
  
 foreach ($this->data\_couchDb['custom\_index\_data']['node'][self::*property\_bloc\_proprdesc*] as $key => $value){  
 if(!empty($value)){  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key] = array(  
 'property\_bloc\_caract' => array(  
 !empty($this->data\_couchDb['custom\_index\_data']['node'][self::*property\_bloc\_caract*][$key]) ? $this->data\_couchDb['custom\_index\_data']['node'][self::*property\_bloc\_caract*][$key]: '',  
 'property\_bloc\_couleur' => '',  
 'property\_bloc\_title' => '',  
 'property\_bloc\_proprietes' => '',  
 'property\_bloc\_utilisation' => '',  
 ),  
 'property\_bloc\_propr' => !empty($this->data\_couchDb['custom\_index\_data']['node'][self::*property\_bloc\_propr*][$key]) ? $this->data\_couchDb['custom\_index\_data']['node'][self::*property\_bloc\_propr*][$key]: '',  
 'property\_bloc\_proprdesc' => !empty($this->data\_couchDb['custom\_index\_data']['node'][self::*property\_bloc\_proprdesc*][$key]) ? $this->data\_couchDb['custom\_index\_data']['node'][self::*property\_bloc\_proprdesc*][$key]: '',  
 'property\_bloc\_proprWeight' => !empty($this->data\_couchDb['custom\_index\_data']['node'][self::*property\_bloc\_proprWeight*][$key]) ? $this->data\_couchDb['custom\_index\_data']['node'][self::*property\_bloc\_proprWeight*][$key]: '',  
  
 ) ;  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['label']='';  
  
 }  
 unset($this->data\_couchDb['custom\_index\_data']['node'][self::*property\_bloc\_caract*],$this->data\_couchDb['custom\_index\_data']['node'][self::*property\_bloc\_propr*], $this->data\_couchDb['custom\_index\_data']['node'][self::*property\_bloc\_proprdesc*],$this->data\_couchDb['custom\_index\_data']['node'][self::*property\_bloc\_proprWeight*]);  
 }  
  
 }  
 protected function combine\_structure\_propertyold(){  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'] = array();  
 $key\_foreach = self::*property\_bloc\_proprdesc*;  
  
 foreach ($this->data\_couchDb['custom\_index\_data']['node'][$key\_foreach] as $key => $value ){  
 // var\_dump($value);  
 if(!empty($value)){  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key] = array(  
 'property\_bloc\_caract' => !empty($this->data\_couchDb['custom\_index\_data']['node'][self::*property\_bloc\_caract*][$key]) ? $this->data\_couchDb['custom\_index\_data']['node'][self::*property\_bloc\_caract*][$key]: '',  
 'property\_bloc\_propr' => !empty($this->data\_couchDb['custom\_index\_data']['node'][self::*property\_bloc\_propr*][$key]) ? $this->data\_couchDb['custom\_index\_data']['node'][self::*property\_bloc\_propr*][$key]: '',  
 'property\_bloc\_proprdesc' => !empty($this->data\_couchDb['custom\_index\_data']['node'][self::*property\_bloc\_proprdesc*][$key]) ? $this->data\_couchDb['custom\_index\_data']['node'][self::*property\_bloc\_proprdesc*][$key]: '',  
 'property\_bloc\_proprWeight' => !empty($this->data\_couchDb['custom\_index\_data']['node'][self::*property\_bloc\_proprWeight*][$key]) ? $this->data\_couchDb['custom\_index\_data']['node'][self::*property\_bloc\_proprWeight*][$key]: '',  
 ) ;  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['label']='';  
 }  
  
 }  
 unset($this->data\_couchDb['custom\_index\_data']['node'][self::*property\_bloc\_caract*],$this->data\_couchDb['custom\_index\_data']['node'][self::*property\_bloc\_propr*], $this->data\_couchDb['custom\_index\_data']['node'][self::*property\_bloc\_proprdesc*],$this->data\_couchDb['custom\_index\_data']['node'][self::*property\_bloc\_proprWeight*] );  
  
 }  
 protected function toFix\_property(){  
  
 foreach($this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'] as $key => $value) {  
 foreach ($this->data\_gather\_data as $\_key => $\_value) {  
 }  
  
 if ($\_key != 'Intro' && $\_key != 'Pictos' && $\_key != 'Précautions') {  
  
 $name = $this->data\_drupal\_term\_property[$this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['value']];  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['label'] = $name;  
  
  
 // BLOC SANTE  
 if ($this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['label'] == 'Santé' && isset($this->data\_gather\_data['Bloc Santé'])) {  
 if ($this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_propr']['value'] == 'proprietes') {  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_proprdesc']['value'] = $this->data\_gather\_data['Bloc Santé']['Propriétés Santé'];  
 }  
  
 if ($this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_propr']['value'] == 'utilisations') {  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_proprdesc']['value'] = $this->data\_gather\_data['Bloc Santé']['Utilisations Santé'];  
 }  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['Couleur'] = $this->data\_gather\_data['Bloc Santé']['Couleur'];  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['Titre Santé'] = $this->data\_gather\_data['Bloc Santé']['Titre Santé'];  
 }  
  
 // BLOC MAISON  
 if ($this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['label'] == 'Maison' && isset($this->data\_gather\_data['Bloc Maison'])) {  
 if (!empty($this->data\_gather\_data['Bloc Maison'])) {  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['label'] = $name;  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['Couleur'] = $this->data\_gather\_data['Bloc Maison']['Couleur'];  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['Titre Maison'] = $this->data\_gather\_data['Bloc Maison']['Titre Maison'];  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['Propriétés Maison'] = $this->data\_gather\_data['Bloc Maison']['Propriétés Maison'];  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['Utilisations Maison'] = $this->data\_gather\_data['Bloc Maison']['Utilisations Maison'];  
 }  
  
 }  
 //BLOC Beaute  
 if ($this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['label'] == 'Beauté' && isset($this->data\_gather\_data['Bloc Beauté'])) {  
 if (isset($this->data\_gather\_data['Bloc Beauté'])) {  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['label'] = $name;  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['Couleur'] = $this->data\_gather\_data['Bloc Beauté']['Couleur'];  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['Propriétés Beauté'] = $this->data\_gather\_data['Bloc Beauté']['Propriétés Beauté'];  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['Utilisations Beauté'] = $this->data\_gather\_data['Bloc Beauté']['Utilisations Beauté'];  
 }  
  
 }  
  
 //BLOC PRECAUTIONS  
  
 // Cuisine  
 if ($this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['label'] == 'Cuisine' && isset($this->data\_gather\_data['Bloc Cuisine'])) {  
 if (isset($this->data\_gather\_data['Bloc Cuisine'])) {  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['label'] = $name;  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['Couleur'] = $this->data\_gather\_data['Bloc Cuisine']['Couleur'];  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['Titre Cuisine'] = $this->data\_gather\_data['Bloc Cuisine']['Titre Cuisine'];  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['Propriétés Cuisine'] = $this->data\_gather\_data['Bloc Cuisine']['Propriétés Cuisine'];  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['Utilisations Cuisine'] = $this->data\_gather\_data['Bloc Cuisine']['Utilisations Cuisine'];  
 }  
  
 }  
 // BIEN ETRE  
 if ($this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['label'] == 'Bien-être' && isset($this->data\_gather\_data['Bloc Bien-être'])) {  
 if (isset($this->data\_gather\_data['Bloc Maison'])) {  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['label'] = $name;  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['Couleur'] = $this->data\_gather\_data['Bloc Bien-être']['Couleur'];  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['Titre Bien-être'] = $this->data\_gather\_data['Bloc Bien-être']['Titre Bien-être'];  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['Propriétés Bien-être'] = $this->data\_gather\_data['Bloc Bien-être']['Propriétés Bien-être'];  
 $this->data\_couchDb['custom\_index\_data']['node']['ft\_proprietes\_multi'][$key]['property\_bloc\_caract']['Utilisations Bien-être'] = $this->data\_gather\_data['Bloc Bien-être']['Utilisations Bien-être'];  
 }  
  
 }  
 }  
 }  
 }  
  
 protected function combine\_structure\_inpratice(){  
 $this->data\_couchDb['custom\_index\_data']['node']['field\_ft\_pictogramme'] = array();  
  
 foreach ($this->data\_couchDb['custom\_index\_data']['node']['field\_ft\_picto\_mode\_util'] as $key => $value ){  
  
 //var\_dump($key);  
 if(!empty($value)){  
 $this->data\_couchDb['custom\_index\_data']['node']['field\_ft\_pictogramme'][$key] = array(  
 'field\_ft\_picto\_pratice' => array(  
 'field\_ft\_picto\_mode\_util' => !empty($this->data\_couchDb['custom\_index\_data']['node']['field\_ft\_picto\_mode\_util'][$key]) ? implode($this->data\_couchDb['custom\_index\_data']['node']['field\_ft\_picto\_mode\_util'][$key]): '',  
 'field\_ft\_ponderation' => !empty($this->data\_couchDb['custom\_index\_data']['node']['field\_ft\_ponderation'][$key]) ? implode($this->data\_couchDb['custom\_index\_data']['node']['field\_ft\_ponderation'][$key]): ''  
 ),  
 'field\_ft\_picto\_gather' => array()  
 ) ;  
 }  
  
 }  
 unset($this->data\_couchDb['custom\_index\_data']['node']['field\_ft\_picto\_mode\_util'],$this->data\_couchDb['custom\_index\_data']['node']['field\_ft\_ponderation'] );  
  
 }  
 protected function toFix\_inpratice(){  
 foreach ($this->data\_couchDb['custom\_index\_data']['node']['field\_ft\_pictogramme'] as $key => $value) {  
 $name = $this->data\_drupal\_term\_inpratice[$this->data\_couchDb['custom\_index\_data']['node']['field\_ft\_pictogramme'][$key]['field\_ft\_picto\_pratice']['field\_ft\_picto\_mode\_util']];  
 $this->data\_couchDb['custom\_index\_data']['node']['field\_ft\_pictogramme'][$key]['field\_ft\_picto\_pratice']['label']= $name ;  
  
 foreach ($this->data\_gather\_data['Pictos'] as $key\_p => $value\_p){  
  
 if($key\_p == $name ){  
 $this->data\_couchDb['custom\_index\_data']['node']['field\_ft\_pictogramme'][$key]['field\_ft\_picto\_gather']['label']= $name;  
 $this->data\_couchDb['custom\_index\_data']['node']['field\_ft\_pictogramme'][$key]['field\_ft\_picto\_gather']['ponderation']= $this->data\_gather\_data['Pictos'][$name];  
 }  
  
 }  
  
 }  
}  
  
  
  
 protected function combine\_structure\_couchdb(){  
 $this->data\_couchDb;  
 $this->DrupalStructure;  
  
 foreach($this->DrupalStructure as $key\_stru){  
 foreach ($this->data\_couchDb as $key\_db){  
  
 if(array\_search('description',$key\_db) != null || array\_search('description',$key\_db)!= false){  
 $find = array\_search('Riche en actifs anti-inflammatoires et cicatrisants, cette huile est',$key\_db);  
 var\_dump($find);  
 }  
 }  
 }  
 //var\_dump($find);  
 }  
  
 */\*\*  
 \* Method to make Drupal like structure with selected arrays  
 \*  
 \** ***@return*** *array  
 \*/* protected function makeDrupalStructure(){  
  
 $tree = array();  
 $\_tree = array();  
  
 $remove\_other = $this->Etiquettes\_other;  
 $index\_weight\_grp = 0;  
 $index\_weight\_node = 0;  
  
 $old\_weight=$this->Etiquettes\_none['ndfiWeight'] ;  
  
 */\*\* Make not null parent structure \*/* foreach($this->Etiquettes\_none as $key){  
 if($old\_weight != $key['gpWeight']){  
 $index\_weight\_node = $key['ndfiWeight'];  
 $old\_weight = $key['gpWeight'] ;  
 }  
  
 if($key['ndfiWeight']!= $index\_weight\_node ){  
  
 foreach ($this->Etiquettes\_other as $keya ){  
 if($keya['parent'] == $key['group\_name'] && $keya['gpWeight'] == $index\_weight\_node)  
 {  
 array\_push($tree, $keya);  
 unset($remove\_other[array\_search($keya,$remove\_other)]);  
 }  
 }  
 $index\_weight\_node++;  
 }  
  
 array\_push($tree, $key);  
 $index\_weight\_node++;  
 }  
  
 */\*\* Get other not find \*/* foreach($tree as $key){  
 foreach ($remove\_other as $keyo) {  
 if ($keyo['parent'] == $key['group\_name'] ) {  
 array\_push($\_tree, $keyo);  
 unset($remove\_other[array\_search($keyo,$remove\_other)]);  
 }  
 }  
 }  
  
  
 */\*\* Make null parent structure\*/* $\_\_tree=array();  
 foreach($this->Etiquettes\_null as $key){  
 // initialisation field  
 // nome de la fiche  
 // ingrédient  
  
 // visuel ambiance  
  
 // taxonomie  
 // Paramètres du menu  
 // information de publication  
  
 //Informations sur les révisions  
 // option de publication  
 //paramètres des commentaires  
 array\_push($\_\_tree, $key);  
 // insert meta tag  
 // insert path settings  
 // insert printer, email and PDF versions  
 }  
  
  
 */\*\* Combine Drupal Structure \*/* array\_push( $\_\_tree, $tree, $\_tree);  
 //foreach ($\_\_tree as $i){  
 // var\_dump($i);  
 //}  
 //print\_r($\_\_tree);die('end');  
 return $\_\_tree;  
  
 }  
  
 protected function selectDrupalTermData($vid){  
 $db = $this->adapter\_azr;  
  
 $sql = new Sql($db);  
  
 $select = new Select(array('drtd'=>'drupal\_term\_data'));  
 $select->columns(array(  
 'tid',  
 'name'  
 )  
 )  
 ->where('vid ='.$vid)  
 ->order(['weight ASC']);  
  
  
 $statement = $sql->prepareStatementForSqlObject($select);  
 $results = $statement->execute();  
  
  
 if ($results->isQueryResult()){  
 return $results;  
 }  
 else{  
 return new ApiProblem(404, 'Not found');  
 }  
 }  
  
 */\*\*  
 \* Method to get structure from drupal  
 \* MySQL select  
 \*  
 \** ***@param*** *$DrupalStructure\_Where  
 \** ***@return*** *ApiProblem  
 \*/* protected function selectStructureDrupal($DrupalStructure\_Where){  
 $db = $this->adapter\_azr;  
  
 $sql = new Sql($db);  
  
 $select = new Select(array('ndfi'=>'drupal\_content\_node\_field\_instance'));  
 $select->columns(array(  
 'field\_name',  
 'type\_name',  
 'ndfilabel'=>'label',  
 'widget\_type',  
 'description',  
 'widget\_module',  
 'ndfiWeight'=> 'weight'  
 )  
 )  
 ->join(  
 array('ndf'=>'drupal\_content\_node\_field'),  
 'ndfi.field\_name = ndf.field\_name',  
 array('type', 'module')  
 )  
 ->join(  
 array('gpF'=> 'drupal\_content\_group\_fields'),  
 'ndfi.field\_name = gpF.field\_name',  
 array('group\_name'),  
 Select::*JOIN\_LEFT* )  
 ->join(  
 array('gp'=> 'drupal\_content\_group'),  
 'gpF.group\_name = gp.group\_name',  
 array('group\_type',  
 'parent',  
 'gpLabel'=>'label' ,  
 'gpWeight'=>'weight'  
 ),  
 Select::*JOIN\_LEFT* )  
 ->where($DrupalStructure\_Where)  
 ->order(['gp.weight ASC','ndfi.weight ASC', 'gp.label ASC', 'parent ASC']);  
  
  
 $statement = $sql->prepareStatementForSqlObject($select);  
 $results = $statement->execute();  
  
 if ($results->isQueryResult()){  
 return $results;  
 }  
 else{  
 return new ApiProblem(404, 'Not found');  
 }  
 }  
  
  
 */\*\*  
 \* Method to get structure from API gather  
 \* MySQL select  
 \*  
 \** ***@param*** *$DrupalStructure\_Where  
 \** ***@return*** *ApiProblem  
 \*/* protected function selectOneDataGather($sku){  
 $db = $this->adapter\_api;  
  
 $sql = new Sql($db);  
  
 $select = new Select(array('gather'=>'app\_he\_gather'));  
 $select->columns(array(  
 'id',  
 'project\_id',  
 'item\_id',  
 'name',  
 'sku',  
 'created\_at',  
 'updated\_at',  
 'data',  
 'status'  
 )  
 )  
 ->where(['sku' => $sku])  
 ->order('id');  
  
  
 $statement = $sql->prepareStatementForSqlObject($select);  
 $results = $statement->execute();  
  
  
 if(count($results)== 0){  
 return null;  
 }  
 else if ($results->isQueryResult()){  
 return $results;  
 }  
 else{  
 return new ApiProblem(404, 'Not found');  
 }  
 }  
  
  
 */\*\*  
 \* Methode to get datas from couchDB  
 \* Get send method  
 \*  
 \** ***@param*** *$id  
 \** ***@param*** *$param  
 \** ***@return*** *bool|string|void  
 \*/* protected function get($id, $param)  
 {  
 $path = '';  
  
 // Load config and get central WS URL  
 $config = $this->config;  
  
  
 if (isset($config['findproduct'])) {  
 if (isset($id)) {  
 $path = '/'.'/store\_1\_catalog\_product\_'.$id;  
 }  
  
 if (isset($param)) {  
  
 }  
  
 $ws = $config['findproduct'];  
 $client = new Client($config['findproduct']['uri'] . $path, array(  
 'adapter' => 'Laminas\Http\Client\Adapter\Curl',  
 ));  
 //die("ma".$config['findproduct']['uri'].$path);  
 if (isset($ws['headers'])) {  
 $headers = $ws['headers'];  
 if (isset($ws['username']) && isset($ws['password'])) {  
 $client->setAuth($ws['username'], $ws['password'], Client::*AUTH\_BASIC*);  
 }  
 $client->setHeaders($headers);  
 }  
  
 $client->setMethod('GET');  
 $result = $client->send();  
 //$data = json\_decode(json\_encode($result->getBody()), true);  
 //$data = Json::decode($result->getBody());  
 $data = Json::*decode*(Html::*filterText*($result->getBody()), true);  
  
 if ($result->isSuccess()) {  
 //die($data);  
 return $data;  
 } else {  
 //die("requete get invalide");  
 $error = json\_decode($result->getBody());  
 $httpCode = ($error->error == 'not\_found') ? '404':'503';  
 return new ApiProblem($httpCode, $error);  
  
 return print\_r($result->getBody());  
  
 }  
 }  
 }  
  
 */\*\*  
 \* Method to make array as associative  
 \*  
 \** ***@param*** *$array  
 \** ***@return*** *array  
 \*/* protected function getAssociativeArray($array){  
 $tid ='';  
 $name='';  
  
 foreach($array as $item ){  
 foreach ( $item as $keyn => $value){  
 if($keyn == 'tid'){$tid = $value;}  
 if($keyn == 'name'){ $name= $value; }  
 }  
 $data[$tid] = $name;  
 }  
  
 return $data;  
 }  
}