LABINVENT

(DOCUMENTATION TECHNIQUE)

URL officielle de ce doc : https://tinyurl.com/labinvent
Auteurs: E. Pallier, E. Bourrec
Version: 17/02/2020
⇒ Pour faire une copie Word, Libre/Open Office, ou PDF de ce doc : menu "Fichier/Télécharger…"

Un gros effort est fait continuellement pour que cette doc soit le plus à jour possible, merci de bien vouloir la lire ATTENTIVEMENT.

Les auteurs de cette doc eux-mêmes la suivent à la lettre, pourquoi donc feriez-vous autrement ?

N'hésitez pas aussi à la mettre à jour vous-mêmes quand c'est nécessaire, ou bien à soumettre vos suggestions à :

epallier AT irap POINT omp POINT eu

(ANCIENNE DOC qui n'est plus à jour: lien vers la doc d'origine)

Pour info seulement, notes sur l'installation du framework CakePhp : https://book.cakephp.org/3.0/en/installation.html

*Ce document est davantage une **documentation technique** qu'une documentation destinée aux utilisateurs.*
Il entre dans des détails techniques permettant de comprendre comment est organisé le logiciel LabInvent, comment il fonctionne, comment le configurer, l'adapter et le faire évoluer.

Pour la documentation (non technique) décrivant l'utilisation du logiciel, cliquer sur le lien ci-dessous :

⇒ LIEN VERS LA DOC UTILISATEURS

TABLE DES MATIÈRES

1. Version de démonstration **7**

2. Installation des pré-requis **8**

2.1. Plateformes testées 8

2.2. Installation (pré-requis) 9

2.2.1. Installation sur Mac OS (avec HomeBrew) 10

2.2.2. Installation sur Windows 10 10

2.2.3. Installation sur Linux 11

2.2.3.1. Exemple pour une distribution UBuntu 18.04.3 LTS (janvier 2020) 11

2.2.3.2. Exemple pour ScientificLinux (CentOS) 6.4 (fait en juillet 2017) 16

2.2.3.3. Exemple pour CentOS 6.7 (fait en 2016) 17

2.2.3.4. Exemple pour une distribution UBuntu 14.04.4 (fait en 2016) et 14.04.5 (Mai 2017) 19

2.2.3.5. Exemple pour une distribution Fedora 20+ (fait en avril 2017) 21

2.3. (Optionnel) Configuration post Installation (pré-requis) 23

2.3.1. Configuration de Php 23

2.4. (Optionnel) Installation d'un IDE (environnement de développement) 24

3. Installation du logiciel LabInvent **25**

3.1. Récupération du logiciel 25

3.1.1. (NON RECOMMANDÉ) Téléchargement anonyme (version statique, non synchronisée, jetable) 25

3.1.2. (RECOMMANDÉ) Récupérer le logiciel via GIT (version dynamique, synchronisée, nécessité d'un login) 26

3.2. Installation du logiciel 29

3.3. Contrôles rapides 30

3.4. Contenu du projet 31

4. Mise à jour du logiciel (update) **35**

5. Accès local à l'application web **37**

6. Vérification de la conformité du site web LabInvent **40**

7. Test de l'application **47**

8. Fin de la phase installation, première connexion **49**

9. Configuration (Adaptation du logiciel) **51**
9.1. Configuration technique via le fichier de configuration (config/app.php) 51
9.2. Configuration via les pages web de configuration 51
9.2.1. Configuration des paramètres techniques (LDAP, mail, debug…) 53
9.2.1.1. Gestion de l'authentification des utilisateurs via un annuaire LDAP 53
9.2.1.2. Configuration de l'envoi des emails 54
9.2.1.3. Configuration du mode DEBUG "local" 55
9.2.1.4. Configuration des utilisateurs 55
9.2.2. Configuration des paramètres métier (liés à l'inventaire) 55
9.3. Autres types de configurations 56
9.3.1. Changer les images (logos) du site web 56
9.3.2. Modifier le format des étiquettes 56
9.4. (dev only) Utilisation des paramètres de configuration dans le code (dynamiquement) 56
9.5. (dev only) Comment ajouter un nouveau paramètre de configuration 57

10. Etiquettes (optionnel) **61**
10.1. Etiqueteuse (moins de 100€ HT), Thermal Transfert, USB 61
10.2. RUBAN (D1) 64
10.3. Installation du logiciel d'impression (DCD) pour Win (7+) ou Mac (10.9+) 64
10.4. Activation de la fonction d'impression sur LabInvent 64
10.5. Adaptation des étiquettes au besoin du laboratoire 65
10.6. Etiqueteuses installées sur le laboratoire IRAP (IRAP ONLY) 68

11. Workflow général (circuit de saisie) **69**
11.1. Description générale 69
11.2. Procédure à suivre pour commander un matériel 70
11.3. Cycle de vie d'un matériel 71
11.4. Diagramme de séquences 73

12. Descriptions techniques **74**
12.1. Modèle de données (Data model) **75**
12.2. Page d'accueil (démarrage, home page) 76
12.3. Ce qui se passe avant l'affichage d'une page web (workflow) **77**
12.4. Gestion des utilisateurs (connexion et profils associés) **80**
12.4.1. Authentification des utilisateurs (avec ou sans LDAP) 80
12.4.2. Gestion des profils utilisateurs (Autorisations, ACL) 83
12.5. FAKE LDAP 83
12.6. LOG **84**
12.7. DEBUG **85**
12.7.1. Debug général 85
12.7.2. Debug local (applicatif) 86

13. HOWTO **87**
13.1. Version du framework CakePhp utilisé dans le projet 87
13.2. Contrôleur des pages web "simples" 87
13.3. Tout sur les QrCodes 88
13.4. Génération de la liste des utilisateurs (depuis le LDAP) 89
13.5. Structure générale d'une page web (TEMPLATE) = default.ctp 90
13.6. Ajouter une nouvelle action sur un contrôleur 91
13.7. Champ facultatif/obligatoire : Comment rendre facultatif ou obligatoire un champ, et définir les vérifications à faire sur ce champ ? 92
13.8. Où définir les éléments passés à une vue (c'est à dire à une action) ? 92
13.9. Adapter LabInvent au laboratoire utilisateur 93
13.10. Ajouter une nouvelle page web 93
13.11. Recherche de matériels 94
13.12. Autres HOWTO plus anciens 94

14. Cycle de développement à respecter **115**

15. Rendons LabInvent plus "responsive" **117**
15.1. Analyse de l'existant et choix d'une solution technique 117
15.2. Dépendances 118
15.3. Installation du plugin bootstrap-ui 118

15.4. Fichiers installés par bootstrap-ui 119
15.5. Utilisation 120

16. Migrations de la BD 123
16.1. Procédure proposée pour garder la BD à jour suite à nos modifs (pour qu'on ait tous la même version de la BD) 123
16.2. Présentation du concept 124
16.3. Exemples divers 126
16.4. Générer une Migration à partir d'une Base de Données Existante 127
16.5. Nom de Fichier des Migrations 128
16.6. Définition de Colonnes 128
16.7. Créer une Table 129
16.8. Ajouter des Colonnes à une Table Existante 130
16.9. Générer un diff entre deux états de base de données 131
16.10. Les Commandes 132
16.10.1. migrate : Appliquer les Migrations 132
16.10.2. status : Statuts de Migrations 132
16.10.3. seed : Remplir votre Base de Données (Seed) 133
16.11. Trucs et Astuces 137
16.11.1. Mettre à jour les Noms de Colonne et Utiliser les Objets Table 137
16.11.2. Migrations et déploiement¶ 137
16.11.3. Renommer une table 137
16.11.4. Ne pas générer le fichier schema.lock 137

17. Le framework CakePhp 139
17.1. Erreurs sur le framework (TODO) 139
17.2. Mise à jour du framework CakePhp 139
17.2.1. Upgrade de la version 3.6 à la version 3.7 (16/1/19 pm) 139
17.2.2. Upgrade de la version 3.5 à la version 3.6 (16/1/19 matin) 142
17.3. PLUGINS 143
17.3.1. "Composer", le gestionnaire de plugins 143
17.3.2. Plugins spécifiques à Cakephp 144
17.3.3. Les plugins installés pour LabInvent 145
17.3.3.1. Génération des QrCodes (plugin phpqrcode) 145

17.3.3.2. Génération des documents PDF 145
17.3.3.2.1. Le Plugin fpdf 145
17.3.3.2.2. Le plugin cakephp-dompdf pour générer des docs pdf 146
17.3.4. Mettre à jour ou Réinstaller tous les plugins (dossier vendor/) 146
17.3.5. Ajouter un plugin spécifique 147
17.3.6. Supprimer un plugin 149

18. Historique du logiciel 150

19. Roadmap (feuille de route) (TODO LIST) 156
19.1. TODO LIST temporaire à migrer sur le Redmine (notes urgentes) 156
19.2. ANCIENNE ROADMAP 158
GENERAL 158
USERS 159
ETIQUETTES 161
MATERIELS 161
SUIVIS 164
EMPRUNTS 164
19.3. VERY OLD TODO LIST A VIRER BIENTOT (en italique ce qui a déjà été migré dans le tableau “TODO LIST” ci-dessus) 166

20. Annexes 172
20.1. Installation des pré-requis sur Mac OS X (avec HomeBrew) 172
20.1.1. Cleanup BREW 172
20.1.2. APACHE 174
20.1.3. PHP 178
20.1.4. MYSQL => MARIADB 192
20.1.5. Apache Virtual Hosts 194
20.1.6. PHPMYADMIN 196
20.1.7. Installation de l’extension YAML 197
20.2. Installation des pré-requis sur Windows 10 199
20.2.1. Installation directement sur Windows 10 (TODO) 199
20.2.2. Installation sur une VM linux (avec VirtualBox) 199
20.3. Installation et configuration du logiciel depuis l’IDE Eclipse 204
20.4. MODE PANIQUE 215

20.5. MODE DEBUG (dev only) 216
20.6. UPDATE DE LA BRANCHE MASTER (dev assermenté only) 216
20.7. Auto-génération du code (avec “bake”) 217
20.8. METHODES ET CONCEPTS INTERESSANTS A UTILISER 217

1. Version de démonstration

Si vous désirez juste voir à quoi ressemble le logiciel LabInvent, nous mettons à votre disposition un site de démo.

Notamment, en s'y connectant avec le login "utilisateur", on peut voir le workflow par défaut qui s'affiche sur la page d'accueil (pour tout utilisateur lambda, non privilégié).

Logins disponibles (sans mdp), correspondant aux 3 profils principaux (excepté le profil "superadmin"):

"utilisateur", "responsable", et "admin"

⇒ http://planetoweb2.cesr.fr/labinvent2

2. Installation des pré-requis

(created 25/1/19 EP - updated 13/1/20 EP)

2.1. Plateformes testées

Ce logiciel est **multiplateforme** (Linux, MacOS, et presque possible sur Windows), mais la plateforme de prédilection est Linux car c'est ce qui est utilisé pour la production dans tous les laboratoires où ce logiciel a été diffusé. D'autre part, la distribution linux conseillée est CentOS (et ses dérivés).
Pour le développement, on utilise plutôt Mac OS X, mais ça peut très bien se faire aussi sur Linux. On pourrait aussi le faire sur Windows 10, mais il faudrait "juste" pour cela convertir le script d'installation bash en PHP. Quelqu'un veut s'y coller ?

- **LINUX :**
  - **Fedora 20** (version test Thibault Ajas, IRAP, avril 2017)
  - **Centos 6.6** (version de "production", SI IRAP) :
    - PHP : 5.6.22
    - Mysql : 5.1.73-3.el6_5
    - Apache : 2.2.15-39.el6.centos
  - **Debian GNU/Linux 8.5** (jessie) (version de "production", IAS) :
    - PHP : 5.6.22
    - Mysql : MariaDB 10.0.25
    - Apache : 2.4.10
  - **Scientific Linux (=Centos) 6.4** (version dev/test Etienne Pallier linux, IRAP) :
    - PHP : 5.6.30
    - MYSQL : 5.5.56
    - APACHE : 2.2.15
  - **UBuntu 14.04.4** (Ancienne version dev/test Alexandre Cases, IRAP) :
    - PHP : 5.5.9 (ne suffit plus)
    - MYSQL : 5.5.47

        - Apache : 2.4.7

- **<u>MAC OS X</u> :**
    - (25/1/19) **Mac OS 10.14 (Mojave) avec brew** (version dev/test Etienne Pallier, IRAP)
        - OS Darwin Kernel Version 18.2.0: Mon Nov 12 20:24:46 PST 2018; root:xnu-4903.231.4~2/RELEASE_X86_64 x86_64
        - Apache/2.4.37 (Unix)
        - Mysql (MariaDB) Ver 15.1 Distrib 10.3.12-MariaDB, for osx10.14 (x86_64) using readline 5.1
        - PHP 5.6.40 (cli) (built: Jan 16 2019 14:53:29) ET php 7.3.1, avec Zend Engine v2.6.0, Copyright (c) 1998-2016 Zend Technologies with Zend OPcache v7.0.6-dev, Copyright (c) 1999-2016, by Zend Technologies
    - **(17/1/18) Mac OS 10.13.2 avec brew** (version dev/test Etienne Pallier, IRAP) :
        - PHP 7.2.0 + MySQL 5.7.20 + Apache 2.4.28
    - **Mac OS 10.12.5 avec XAMPP** 5.6.3 et 7.1 (version dev/test Etienne Pallier, IRAP) :
        - PHP 5.6.3 + MySQL 5.6.21 + Apache 2.4.10
        - PHP7.1.6 + MariaDB 10.1.24 + Apache 2.4.25

- **<u>WINDOWS 10</u>** (TODO)

2.2. Installation (pré-requis)

Le logiciel nécessite une combinaison "AMP" pour fonctionner, soit les 3 pré-requis suivants :

- *un serveur web **Apache** (récent)*
- *un serveur de base de données **Mysql** (récent)*
- *le langage **Php en version 7.x** (php 5.6 est encore supporté mais plus pour longtemps)*
- *Optionnel mais très recommandé : **PhpMyAdmin***
- *le gestionnaire de versions **git***

- *(N'installez PAS le framework CakePhp, il sera installé automatiquement pour vous !)*

Si ce tiercé est déjà présent sur votre OS, vous pouvez passer à l'étape suivante (Installation du logiciel), et revenir ici seulement en cas de problème de configuration.

Sur Windows, vous pouvez utiliser Wampserver ou XAMPP qui regroupent ces 3 éléments (il n'y aura rien d'autre à faire ensuite).

Sur Mac, vous pouvez installer chacun des 3 éléments séparément via HomeBrew (ou MacPort), ou plus simplement utiliser XAMPP ou MAMP, ou encore télécharger le paquet binaire Mac correspondant à chaque élément.

2.2.1. Installation sur Mac OS (avec HomeBrew)

⇒ Aller à l'annexe correspondante : Installation des pré-requis sur Mac OS X (avec HomeBrew)

2.2.2. Installation sur Windows 10

⇒ Aller à l'annexe correspondante : Installation des pré-requis sur Windows 10

2.2.3. Installation sur Linux

2.2.3.1. Exemple pour une distribution UBuntu 18.04.3 LTS (janvier 2020)

(fait par stagiaire Jeanne Prugniel, et aussi par E. Pallier)

Version des composants LAMP :

- **Linux Ubuntu 18.04.3** : utilisateur “labinv” (avec sudo) avec password = **'labinv'**
- **Apache 2.4.29**
- **Mysql** 14.14 (**5.7.28**) : password root utilisé = **'PassWord,0'** *(chiffre zéro)*
- **Php 7.2.24**
- **PhpMyAdmin :** password utilisé (le même que mysql root)
- **Git 2.17.1**

Choix faits pour la BD inventaire sur cette VM :

- nom BD = **labinvent**
- nom utilisateur labinvent du logiciel = **labinventuser**
- password de cet utilisateur = ‘**PassWord,0’** *(chiffre zéro)*
- *Autres utilisateurs définis :*
    - *superadmin, pass=login*
    - *… (pass=login)*

Installation AMP (Apache, Mysql, Php) :
https://www.digitalocean.com/community/tutorials/comment-installer-la-pile-linux-apache-mysql-php-lamp-sur-un-serveur-ubuntu-18-04-fr
(voir aussi https://doc.ubuntu-fr.org/lamp)

Installation PhpMyAdmin :
https://www.digitalocean.com/community/tutorials/how-to-install-and-secure-phpmyadmin-on-ubuntu-18-04

(mdp mysql et phpmyadmin vm jeanne : PassWord,0)

En résumé :

- **GIT**

- **Installation:**

  $ sudo apt install git

  $ git --version

- **APACHE**
  - **Installation :**

    $ sudo apt update

    $ sudo apt install apache2
  - **Ajuster votre pare-feu afin d'autoriser le trafic web :**

    $ sudo ufw app list

    Si vous regardez sur le profil Apache Full, il devrait y être indiqué qu'il permet le trafic aux ports 80 et 443 :

    $ sudo ufw app info "Apache Full"

    Autoriser le trafic HTTP et HTTPS entrant pour ce profil :

    $ sudo ufw allow in "Apache Full"

    Vous pouvez immédiatement effectuer une vérification :

    Avec firefox, se connecter à "http://localhost"

    ⇒ On devrait voir une page web avec "It works !", ce qui indique que votre serveur web est maintenant bien installé et qu'il est accessible à travers votre pare-feu.

- **MYSQL**
  - **Installation :**

    $ sudo apt install mysql-server

    $ sudo mysql_secure_installation

  - **Configuration** :

    Veuillez noter que pour les systèmes Ubuntu fonctionnant avec MySQL 5.7 (et les versions ultérieures), l'utilisateur **root** MySQL est configuré par défaut pour authentifier en utilisant le plugin auth_socket, plutôt qu'avec un mot de passe. Cela permet d'avoir une meilleure sécurité et ergonomie dans de nombreux cas, mais il peut également compliquer les choses lorsque vous devez autoriser l'ouverture d'un programme externe (ex : phpMyAdmin) afin d'accéder au serveur. Si vous préférez utiliser un mot de passe lorsque vous vous connectez au MySQL en tant que **root**, vous aurez besoin de changer le mode d'authentification de auth_socket à mysql_native_password. Pour y parvenir, ouvrez le prompt MySQL à partir de votre terminal :

$ sudo mysql

mysql> SELECT user,authentication_string,plugin,host FROM mysql.user;

mysql> ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY **'PassWord,0'**;

mysql> FLUSH PRIVILEGES;

mysql> SELECT user,authentication_string,plugin,host FROM mysql.user;

mysql> exit

- Tester la connexion en tant que root (sans utiliser “sudo”) :

  $ mysql -u root -p

- **PHP**
  - **Installation** :

    $ sudo apt install php libapache2-mod-php php-mysql

  - **Configuration :**

    Actuellement, si un utilisateur demande un répertoire du serveur, Apache recherchera d’abord pour un fichier nommé index.html. Nous voulons dire au serveur web de donner priorité aux fichiers PHP, ainsi il faut exiger à Apache de regarder pour un fichier index.php en premier.

    Editer /etc/apache2/mods-enabled/dir.conf

    Cela va ressembler à cela :

    /etc/apache2/mods-enabled/dir.conf

    <IfModule mod_dir.c>

    DirectoryIndex **index.html** index.cgi index.pl index.php index.xhtml index.htm

    </IfModule>

Déplacer le fichier d'index PHP (surligner ci-dessous) à la première position après la spécification DirectoryIndex, de la manière suivante :

<IfModule mod_dir.c>

DirectoryIndex **index.php index.html** index.cgi index.pl index.xhtml index.htm

</IfModule>

Ensuite, redémarrer le serveur web Apache afin que vos modifications prennent effet.

$ sudo systemctl restart apache2

$ sudo systemctl status apache2

vous avez l'option d'installer de modules supplémentaires si besoin. Pour voir les options disponibles de modules PHP et de bibliothèques :

$ apt search php- | less

Pour en savoir plus sur la fonctionnalité d'un module :

$ apt show package_name

- **Test:**

Afin de tester si votre système est configuré correctement pour PHP, créer un script PHP de base appelé **info.php**. Afin qu'Apache puisse localiser ce fichier et le desservir correctement, il devra être sauvegardé dans un répertoire bien spécifique, qui se nomme le "web root" : /var/www/html/. Mettre dans ce fichier le contenu suivant :

<?php

phpinfo();

?>

Se connecter à "http://localhost/info.php" pour voir le résultat

- **PHPMYADMIN**
  - **Installation:**

    $ sudo apt install phpmyadmin php-mbstring php-gettext

    - For the server selection, choose apache2
    - Select Yes when asked whether to use dbconfig-common to set up the database
    - You will then be asked to choose and confirm a MySQL application password for phpMyAdmin

    The installation process adds the phpMyAdmin Apache configuration file into the **/etc/apache2/conf-enabled/** directory, where it is read automatically. The only thing you need to do is explicitly enable the mbstring PHP extension, which you can do by typing:

    $ sudo phpenmod mbstring

    Afterwards, restart Apache for your changes to be recognized:

    $ sudo systemctl restart apache2

    When you installed phpMyAdmin onto your server, it automatically created a database user called **phpmyadmin** which performs certain underlying processes for the program. Rather than logging in as this user with the administrative password you set during installation, it's recommended that you log in as either your **root** MySQL user or as a user dedicated to managing databases through the phpMyAdmin interface.

    You can now access the web interface by visiting your server's domain name or public IP address followed by /phpmyadmin:

    http://localhost/phpmyadmin

    Se connecter en tant que user "root" (c'est à dire le mysql root user)

    (on peut aussi utiliser le user "phpmyadmin", mais il est moins privilégié)

- **BUGFIXES**:

  Cette version de phpmyadmin n'est pas assez récente pour php7.2, il faut donc corriger 2 lignes du code source selon ce qui est indiqué ici : https://doc.ubuntu-fr.org/phpmyadmin#incompatibilite_avec_php_72

2.2.3.2. Exemple pour ScientificLinux (CentOS) 6.4 (fait en juillet 2017)

ATTENTION, SUR CENTOS (ou dérivé), il vaut mieux désactiver selinux

Pour mettre à jour PHP de la 5.6 à la 7.1 :
--> https://blog.remirepo.net/post/2016/12/05/Install-PHP-7.1-on-CentOS-RHEL-or-Fedora
$ sudo yum update kernel
$ sudo yum update

$ sudo yum install yum-utils
$ sudo yum-config-manager --enable remi-php71
$ sudo yum update
=> mais il y a un conflit à cause de phpmyadmin, donc je supprime ce package :
$ sudo yum erase phpmyadmin
$ sudo yum update

$ php -v => 7.1

Redémarrage Apache:
$ sudo /etc/init.d/httpd restart

Tentative de réinstaller phpmyadmin
$ sudo yum install phpmyadmin
=> toujours un conflit, je laisse tomber, dommage...

Tentative d'accélérer php 7:
(https://community.1and1.com/php-7)
$ sudo yum install php71-php-opcache
Creer un repertoire .opcache/ dans le webroot/ du projet:
$ cd webroot/
$ mkdir .opcache/
$ chmod 777 .opcache/

Modifier php.ini :
$ sudo vi /etc/php.ini
Ajout des lignes suivantes :

; EP added this for opcache (Jul 2017):
zend_extension=opcache.so;
opcache.enable=1;opcache.memory_consumption=32;
opcache.interned_strings_buffer=8;
opcache.max_accelerated_files=3000;
opcache.revalidate_freq=180;
opcache.fast_shutdown=0;
opcache.enable_cli=0;
opcache.revalidate_path=0;
opcache.validate_timestamps=2;
opcache.max_file_size=0;
opcache.file_cache=/projects/labinvent/labinvent2/webroot/.opcache;
opcache.file_cache_only=1;

Faire un lien vers l'extension opcache.so
$ cd /usr/lib64/php/modules/
$ sudo ln -s /opt/remi/php71/root/usr/lib64/php/modules/opcache.so

Verifier que .opcache/ contient bien des données (du cache)

Mais je ne vois pas vraiment d'accélération...

2.2.3.3. Exemple pour CentOS 6.7 (fait en 2016)

ATTENTION, SUR CENTOS, il vaut mieux désactiver selinux

Mettre à jour le serveur:
$ sudo yum kernel
(restart)
$ sudo yum update

Pour installer Apache, MySQL & PHP 5.3 :
--> https://www.zerostopbits.com/how-to-install-apache-mysql-and-php-on-centos-6-7/

Pour mettre à jour PHP de la 5.3 à la 5.6
--> https://www.zerostopbits.com/how-to-upgrade-php-5-3-to-php-5-6-on-centos-6-7/

Mettre à jour Mysql (version 5.1 à 5.5):
$ sudo yum update

2.2.3.4. Exemple pour une distribution UBuntu 14.04.4 (fait en 2016) et 14.04.5 (Mai 2017)

/!\ Par défaut, la version de php installée ici est php5.5 qui ne suffit plus. Si vous souhaitez installer la version 5.6, remplacez TOUS les "php5" par "php5.6", et si vous voulez la version 7.1, remplacez TOUS les "php5" par "php7.1" /!\

Pour commencer il faut mettre à jour les "repository" de apt :
$ sudo apt-get update && sudo apt-get upgrade

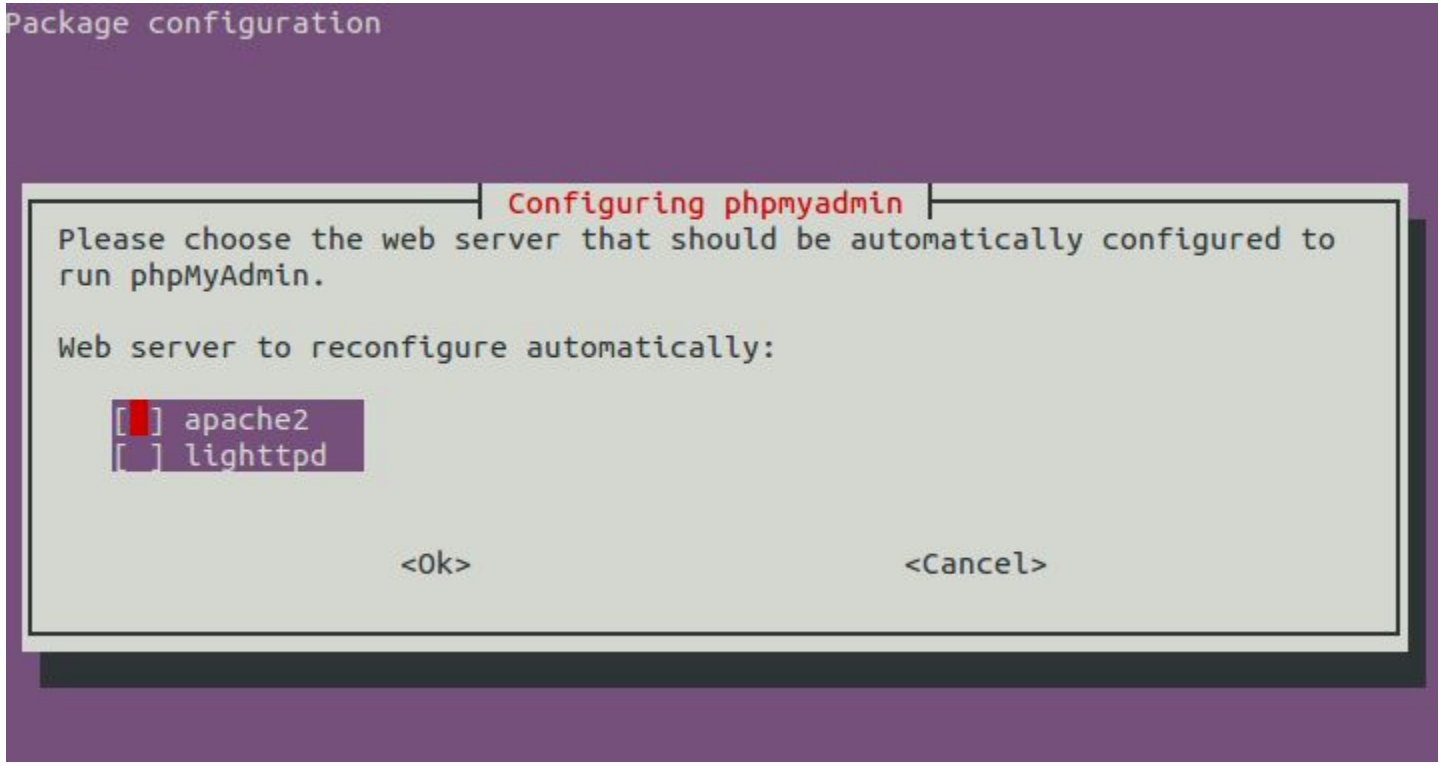
Installer un serveur web (Apache) :
$ sudo apt-get install apache2

Installer un serveur de base de données (MySQL):
$ sudo apt-get install mysql-server

Installer le langage PHP en version 5.5.9 minimum (5.6 recommandé)
$ sudo apt-get install php5 php-pear
$ sudo apt-get install php5-mysql

Installer phpmyadmin et le configurer
$ sudo apt-get install phpmyadmin
$ sudo dpkg-reconfigure -plow phpmyadmin

/!\ Lorsque vous aurez l'écran suivant, n'oubliez pas d'appuyer sur la touche "espace" avant la touche "entrée" /!\



Afin d'avoir cela :

Si, en visitant http://localhost/phpmyadmin/ vous avez l'erreur "The mcrypt extension is missing. Please check your PHP configuration.", executez les commades suivantes :

$ sudo apt-get install php5-mcrypt

```
$ sudo ln -s /etc/php5/conf.d/mcrypt.ini /etc/php5/mods-available
$ sudo php5enmod mcrypt
$ sudo service apache2 restart
```

2.2.3.5. Exemple pour une distribution Fedora 20+ (fait en avril 2017)

(https://www.digitalocean.com/community/tutorials/how-to-install-lamp-linux-apache-mysql-php-on-fedora)

Pour commencer il faut mettre à jour l'OS :

```
$ sudo dnf update dnf
$ sudo dnf update kernel
$ sudo dnf update
```

Installer git:

```
$ sudo dnf install git
```

Installer php :

```
$ sudo dnf install php
$ sudo dnf install php-mysql
```

Installer un serveur web (Apache) :

```
$ sudo dnf install httpd
$ sudo systemctl enable httpd
(ln -s '/usr/lib/systemd/system/httpd.service' '/etc/systemd/system/multi-user.target.wants/httpd.service')
$ sudo systemctl start httpd
```

Installer un serveur de base de données (MySQL):

$ sudo dnf install mariadb mariadb-server -y
$ sudo systemctl enable mariadb
(ln -s '/usr/lib/systemd/system/mariadb.service' '/etc/systemd/system/multi-user.target.wants/mariadb.service')
$ sudo systemctl start mariadb
$ sudo mysql_secure_installation

(OPTIONNEL) Installer phpmyadmin (par defaut accessible uniquement depuis localhost), utile pour gérer plus facilement la BD :

$ sudo dnf install phpmyadmin
$ sudo systemctl restart httpd

Pour Ubuntu:
sudo apt-get php5-mcrypt
sudo apt-get install phpmyadmin
Pensez à activer l'extension mcrypt : sudo php5enmod mcrypt

2.3. (Optionnel) Configuration post Installation (pré-requis)

2.3.1. Configuration de Php

Dossier de log (optionnel):
Dans le fichier /etc/php7/apache2/php.ini (ou /etc/php.ini sur CentOS), vous devez positionner votre répertoire de log :

error_reporting = E_ALL
error_log = /var/log/php/error.log
max_input_time = 30

Ensuite il vous faudra peut-être créer le dossier en question et donner à Apache les droits sur ce dossier (www-data pour Ubuntu, apache pour CentOS...):

sudo mkdir /var/log/php
sudo chown www-data /var/log/php

⇒ **Maintenant, recharger la configuration du serveur Web**

$ sudo service httpd reload
(CentOS: $ sudo systemctl reload httpd)
(MacOS: $ sudo apachectl restart)

2.4. (Optionnel) Installation d’un IDE (environnement de développement)

Ce pré-requis est optionnel et ne concerne que les développeurs qui souhaiteraient contribuer au développement du logiciel LabInvent avec un certain confort.
Nous vous conseillons d’installer l’IDE Eclipse avec le plugin PyDev.
Voici un lien qui peut vous aider pour cela :
https://www.linuxrouen.fr/wp/programmation/developpement-python-avec-eclipse-et-pydev-introduction-21165/

Bien sûr vous pouvez utiliser un autre IDE dédié à Php.
Si vous avez déjà l’habitude d’un environnement, gardez le, sinon installez Eclipse.

3. Installation du logiciel LabInvent

Une fois les pré-requis bien en place, on peut procéder à l'installation du logiciel lui-même.

NB: Pour faire l'installation directement depuis l'IDE Eclipse, aller à l'annexe 25.3. Installation et configuration du logiciel depuis l'IDE Eclipse

3.1. Récupération du logiciel

Avant toute chose, placez-vous dans le dossier où vous voulez installer le logiciel LabInvent.

Deux options se présentent à vous :

- a) **soit vous récupérez une version statique** du logiciel, en le téléchargeant (pas besoin de login, c'est anonyme) : considérez alors cette version comme une **version de test jetable car il vous faudra recommencer pour obtenir chaque nouvelle version**, pas très pratique donc, mais rapide
- b) (**méthode préférée**) **soit vous récupérez une version synchronisée**, avec git (nécessité d'avoir un login), ce qui vous permettra de rester constamment à jour (sans réinstallation), et même de contribuer à l'évolution du logiciel si vous le désirez

3.1.1. (NON RECOMMANDÉ) Téléchargement anonyme (version statique, non synchronisée, jetable)

Vous pouvez télécharger la version actuelle du logiciel.

Pour cela, aller sur : https://gitlab.irap.omp.eu/epallier/labinvent/tree/master

Cliquez sur : "Download zip" dans le coin en haut à droite.

Double-cliquez dessus ou dézippez-le (ou lancez la commande gzip -d labinvent.zip). Vous devriez avoir un dossier "labinvent.git".

Dans sa documentation, le logiciel sera désigné par "LABINVENT".

Vous pouvez renommer "labinvent.git" en "labinvent" si vous le souhaitez ("mv labinvent.git labinvent" ou clic droit->Renommer).

3.1.2. (RECOMMANDÉ) Récupérer le logiciel via GIT (version dynamique, synchronisée, nécessité d'un login)

Pour cette option, vous devez avoir un login. Si vous n'avez pas déjà un login, allez sur la page https://gitlab.irap.omp.eu/users/sign_in,

puis remplissez la section "Sign up". Ensuite, envoyez un email à epallier et à ebourrec (AT irap.omp.eu)

en demandant l'autorisation d'accéder au gitlab du projet labinvent. On vous donnera alors la procédure à suivre pour vous connecter.

(Si vous utilisez Windows, vous DEVEZ avant tout installer git pour windows, voir plus bas la section "Seulement pour windows")

En récupérant directement le logiciel via git, vous allez avoir une version dynamiquement synchronisée. Vous serez donc en mesure de la mettre à jour dès qu'une nouvelle version sera disponible avec la commande "git pull".

Aller dans le dossier où vous voulez installer LabInvent puis téléchargez-le via git :

> $ **git clone https://gitlab.irap.omp.eu/epallier/labinvent.git labinvent**

(Ou aussi depuis ssh, seulement au sein de l'IRAP : git clone git@gitlab.irap.omp.eu:epallier/labinvent.git labinvent)

(Si vous récupérez le projet pour la première fois, git vous demandera un login et un mot de passe)

Si vous obtenez ce message d'erreur ... :

> fatal: unable to access 'https://gitlab.irap.omp.eu/epallier/labinvent.git/': Peer's certificate issuer has been marked as not trusted by the user.

Essayez une de ces 2 solutions en tapant les commandes suivantes :

- solution la plus simple mais aussi radicale :
  $ git config --global http.sslVerify false
- solution un peu plus compliquée mais plus soft :
  $ mkdir -m 0700 ~/.gitcerts

$ echo | openssl s_client -connect gitlab.irap.omp.eu:443 -servername gitlab.irap.omp.eu 2>/dev/null | openssl x509 > ~/.gitcerts/gitlab.irap.omp.eu.crt
$ git config --global credential.helper 'cache --timeout=3600'
$ git config --global http.sslCAinfo ~/.gitcerts/gitlab.irap.omp.eu.crt

Puis essayez à nouveau de cloner le projet (commande “git clone” ci-dessus).

Enfin, ajoutez vos infos personnelles :
$ git config --global user.email "Vous@exemple.com"
$ git config --global user.name "Votre Nom"

Vérifiez que votre configuration est OK :
$ git config --list
(ou encore : cat ~/.gitconfig)

Cela devrait afficher quelque chose comme :
http.sslverify=true
http.sslcainfo=/home/labinv/.gitcerts/gitlab.irap.omp.eu.crt
credential.helper=cache --timeout=3600
user.email=...
user.name=Etienne Pallier
core.repositoryformatversion=0
core.filemode=false
core.bare=false
core.logallrefupdates=true
remote.origin.url=https://gitlab.irap.omp.eu/epallier/labinvent.git
remote.origin.fetch=+refs/heads/*:refs/remotes/origin/*
branch.master.remote=origin
branch.master.merge=refs/heads/master
branch.dev.remote=origin
branch.dev.merge=refs/heads/dev
...

Git a normalement créé un dossier "labinvent" qui contiendra votre projet (avec un sous dossier ".git" qui sert à la synchronisation avec le dépôt git).

ATTENTION:

Par défaut, vous êtes sur la **branche "master"** du git. Elle contient une version stable du logiciel. Si vous désirez seulement utiliser ce logiciel SANS LE MODIFIER, alors restez sur la branche "master", vous y serez très bien ;-). Attention seulement de ne faire AUCUNE modification sur la branche master !!!

Par contre, si vous souhaitez contribuer au développement de ce logiciel, et donc le modifier, **vous devez absolument changer de branche et vous placer sur la branche "dev"** (ou bien une sous-branche dédiée comme "dev-IRAP", ou "dev-LATMOS") :

```
$ cd labinvent/
$ git branch
$ git checkout dev
$ git branch
```

Si jamais votre dossier "labinvent" appartient à root (vous avez fait un "git clone" depuis root…, c'est pas bien !!), il serait préférable que vous en soyez vous-même (ou un autre user) le propriétaire :

```
sudo chown -R nom_utilisateur labinvent/
```

Seulement pour Windows :

Pour pouvoir faire les manipulations décrites ci-dessus sur Windows, il faut d'abord installer git.

- Téléchargez git sur https://git-scm.com/download/win
- Lancez l'installation (gardez la configuration par défaut)
- Une fois installé, lancer une invite de commande (Touche Windows+R, tapez cmd, touche entrer) pour commencer à taper des commandes git

Vous pouvez désormais utiliser git depuis une invite de commande ou depuis l'interface graphique de git.

3.2. Installation du logiciel

Il suffit d’exécuter le script installation.sh depuis le dossier install/ :

```
$ cd install/
$ ./intallation.sh
```

Attention: sur **Mac OS X**, utiliser installation-macos.sh au lieu de installation.sh

(vous pourriez éventuellement le faire en tant qu'administrateur "root", mais ça n'est pas nécessaire, le script fera quelques petits "sudo" seulement au besoin).

(Conseil : à la plupart des questions, laissez les réponses par défaut)

3.3. Contrôles rapides

- **Verifier que la BD d'inventaire (par défaut "labinvent" sauf si vous lui avez donné un autre nom) à bien été créée (avec phpmyadmin par exemple)**

- **Tester le site web avec le serveur web de développement inclus**
  Attention, ne faites aucune action sur cette page, c'est juste pour vérifier qu'on accède bien au site
  En attendant de configurer votre serveur web (Apache), vous pouvez déjà voir à quoi ressemble le logiciel en utilisant le serveur web de dev inclus :

  ```
  $ chmod +x TEST_WEB
  $ ./TEST_WEB
  ```

  ⇒ CONNECTEZ-VOUS MAINTENANT A http://localhost:8765
  Attention, ne faites aucune action sur cette page, c'est juste pour vérifier qu'on accède bien au site
  CTRL-C pour stopper
  On ne peut malheureusement pas utiliser ce mode rapide de visualisation, c'est juste un aperçu.
  Il faut passer par une configuration du serveur web Apache que l'on verra plus loin.

3.4. Contenu du projet

(updated 18/12/18 - EP)

Voici la description des dossiers et fichiers de l'application

Fichier ou Dossier	Description
README.md	Le fichier README général du projet, contient l'historique des versions importantes
TESTS.sh	Script d'exécution des tests (à exécuter régulièrement, après chaque modif)
bin/	Quelques fichiers binaires du framework CakePhp (à utiliser uniquement pendant la phase de développement)
composer.json	Le fichier de description de tous les plugins utilisés par CakePhp et installés automatiquement par Composer
config/	Dossier des fichiers de configuration, notamment app.php qui décrit la configuration de la base de données, des mails, et des tests
database/	Dossier contenant le schéma de base de données pour l'installation ainsi que les scripts de mise à jour à exécuter lorsque la BD a été modifiée (sous-dossier "update")
doc/	Quelques documents, notamment celui-ci
index.php	Le point d'entrée du site web
install/	Le dossier pour l'installation contenant notamment le script général d'installation "installation.sh"
logs/	Les logs de l'application (utiles en pour le debug)
src/	C'est le dossier principal contenant tout le code source de l'application LabInvent pour CakePhp (il aurait aussi pu s'appeler "app"). Voir ci-dessous pour le contenu de ce dossier.
tests/	Le dossier contenant tous les tests exécutés par le script TESTS.sh (voir ci-dessus)

tmp/	Dossier utilisé par LabInvent comme cache des pages web et des tables de la BD
vendor/	Tous les plugins pour CakePhp installés automatiquement par Composer. Lors d'une première récupération du projet, ce dossier est vide. Une fois l'installation faite, ce dossier sera plein de sous-dossiers, un par plugin. Par exemple, vous y trouverez le plugin **composer**/ lui-même, ainsi que le plugin principal **cakephp**/ et le plugin **phpunit**/ qui sert à exécuter les tests (on y trouve même un peu des frameworks symfony et zend...).
webroot/	Le dossier contenant toutes les "ressources" ou éléments utilisés par les pages web, comme les images, les feuilles de style CSS pour la mise en page, les modules javascript pour l'animation des pages, les documents attachés, etc.

Description du dossier principal de l'application src/ :

Controller/	Dans un framework MVC (Modèle-Vue-Contrôleur) tel que CakePhp, le **Contrôleur (le C du MVC)** contient toute la logique métier, en gros presque tout le code de traitement des données. Il récupère une demande d'un utilisateur (qui clique sur une page web), réclame les données nécessaires pour cette demande à la couche Modèle (voir ci-dessous), et les passe à la couche Vue (voir ci-dessous) pour qu'elle les présente à l'utilisateur (une nouvelle page web par exemple). Ce dossier contient donc tous les contrôleurs, un par entité (matériel, utilisateur, fournisseur…), en gros un par table de la BD (mais pas toujours). Par exemple le contrôleur MaterielsController est responsable de traiter les demandes concernant l'entité "matériel". Comme vous pouvez vous en douter, c'est le plus gros contrôleur de LabInvent, étant donné qu'il est centré sur la ressource "matériel". C'est lui qui contient les méthodes CRUD (Create, Read, Update, Delete) permettant de gérer un matériel. Il contient aussi les méthodes permettant de changer le statut d'un matériel pour gérer son cycle de vie (CREATED => VALIDATED => TOBEARCHIVED => ARCHIVED). Tous les contrôleurs sont des classes qui héritent de la classe **AppController** qui définit un ensemble de comportements par défaut pour tout contrôleur. Notamment, tout comportement général d'un contrôleur doit être déplacé dans AppControlleur plutôt que d'être répété (recopié) dans chaque contrôleur. AppController lui-même hérite de **Controller** qui est une super-classe de CakePhp offrant elle-même beaucoup de comportements par défaut de tout contrôleur (comme les opérations standard CRUD). Contrairement à AppController, il vaut mieux ne pas modifier la classe Controller car elle risque d'évoluer avec chaque nouvelle version de CakePhp.
Model/ - Behavior/ - **Entity**/ - **Table**/	**C'est la couche "M" (Model)** du pattern MVC implémenté par le framework CakePhp. Cette couche est responsable de lire ou écrire les données demandées ou rendues par la couche C (Contrôleur), quelque soit le système de gestion de données sous-jacent (MySql, ou PostGreSql, ou Sqlite, ou simplement des fichiers, etc.). Elle assure ainsi un niveau d'abstraction par rapport à ce système sous-jacent, qui permet de s'en rendre indépendant et même de le remplacer facilement si besoin était. Le niveau d'abstraction est même poussé un peu plus loin en offrant un mapping relationnel-objet (ORM) qui permet de manipuler les données sous forme d'objets plutôt qu'avec des requêtes SQL plus difficiles à écrire et à maintenir. Ainsi, par exemple, $materiel->save() permettra tout simplement d'enregistrer un objet matériel (représentant une ligne de la table)

Template/ (et View/)	dans la table matériels, en le créant s'il n'existait pas déjà, ou en le mettant à jour sinon. Le dossier Model/ contient notamment les sous-dossiers Entity/ et Table/ : - le sous-dossier **Entity**/ contient un fichier par table qui est une représentation en Php de la structure de la table. Chacun de ces fichiers définit une classe qui hérite de la super-classe **Entity** de CakePhp. Une fois ce fichier écrit, on peut alors générer la table SQL correspondante dans la base de données. Toute modification de la table devrait donc être faite ici plutôt que directement dans la table elle-même. Cela permet la portabilité du projet d'un SGBD à un autre étant donné que cette représentation Php est indépendante de tout formalisme spécifique à un SGBD particulier. - le sous-dossier **Table**/ contient lui aussi un fichier par table, mais décrit plutôt la façon dont les champs (colonnes) de la table doivent être validées. Par exemple, les règles qui permettent de valider un "email" avant de l'enregistrer dans le champ "email" de la table. Chacun de ces fichiers définit une classe qui hérite de la classe **AppTable** (qui elle-même hérite de la super-classe **Table** de CakePhp, à ne pas toucher vu qu'elle peut évoluer avec les nouvelles versions du framework) qui définit quelques règles de validation générales utilisables pour toutes les tables (par exemple, la validation d'un email, ou d'une chaine de caractères pour définir les caractères interdits par défaut...). **C'est la couche "V" (View)** du pattern MVC implémenté par le framework CakePhp (pour plus d'infos, voir https://book.cakephp.org/3.0/fr/views.html). Elle est responsable de présenter les données la plupart du temps sous forme d'une page web. Ces données lui sont passées par la couche (C)ontrôleur. Le dossier View/ (normalement utilisé pour y placer des classes de vue, sous-classes de AppView, elle-même sous-classe de View ; par exemple, PdfView) n'étant pas utilisé pour le moment dans LabInvent, nous ne parlerons que du dossier **Template**/. Celui-ci contient **des sous-dossiers portant chacun le nom du contrôleur qui va l'utiliser**. Les fichiers templates (.ctp) qui sont dans ces sous-dossiers portent un nom correspondant à l'action effectuée. Par exemple, le fichier de vue pour l'action « view() » du controller Products devra normalement se trouver dans **src/Template/Products/view.ctp.** Un template (.ctp) est un fichier HTML qui peut en plus contenir des instructions particulières dans un format de template propre à CakePhp. Il construit la page web qui sera présentée à l'utilisateur. En général, il y a au moins un fichier par ACTION faite par un contrôleur. Par exemple, les actions CRUD (Create, Read, Update, Delete) sur une table donnée sont permises grâce à des vues dédiées (pages web) telles que (resp.) les templates **add.ctp, view.ctp/index.ctp, et edit.ctp**, l'action Delete n'ayant en général pas besoin d'une page web pour être exécutée (donc a priori pas de template delete.ctp). Les templates view.ctp et index.ctp représentent l'action Read respectivement sur UNE ligne de la table ou sur TOUTES les lignes. Ensuite, on peut avoir d'autres templates selon les besoins, tels que **find.ctp** par exemple pour afficher une page web de recherche des matériels. Dans le dossier **Template**/, on trouve quelques sous-dossiers particuliers : - **Element**/ : contient notamment des parties (blocs) de pages web réutilisés dans différentes pages - **Error**/ : contient les pages web qui sont affichées en cas d'erreur - **Fichemetrologiques/, Formules/, et Unites**/ : contiennent toutes les vues spécifiques pour le module Métrologie - **Layout**/ : contient notamment la page web **default.ctp** qui sert de page de base (template) à toutes les pages web du site ; c'est elle qui donne la structure générale des pages du site (menu à gauche, header, footer, …) ; cette page contient aussi le numéro de version et la date de LabInvent qui doit être mise à jour à chaque modification du projet

	- **Pages**/ : contient toutes les “pages” web simples (plutôt “statiques) du site, comme la page “**home**.ctp” (page d’accueil), la page “**about**.ctp”, la page “**tools**.ctp” (menu Outils), la page “**printers**.ctp” (présentation de l’étiqueteuse), ou la page **infos**.ctp (informations systèmes concernant le système hôte, accessible uniquement au profil SuperAdmin) - **QrCodes**/ : page creer.ctp permettant la création du QrCode associé à un matériel, sur chaque page web de visualisation d’une fiche matériel La couche vue de CakePHP peut être constituée d’un certain nombre de parties différentes. Chaque partie a différents usages qui seront présentés dans ce chapitre : ● **templates**: Les templates sont la partie de la page qui est unique pour l’action lancée. Elles sont la substance de la réponse de votre application. ● **elements** : morceaux de code de view plus petits, réutilisables. Les elements sont habituellement rendus dans les vues. ● **layouts** : fichiers de template contenant le code de présentation qui se retrouve dans plusieurs interfaces de votre application. La plupart des vues sont rendues à l’intérieur d’un layout. ● **helpers** : ces classes encapsulent la logique de vue qui est requise à de nombreux endroits de la couche view. Parmi d’autres choses, les helpers de CakePHP peuvent vous aider à créer des formulaires, des fonctionnalités AJAX, à paginer les données du model ou à délivrer des flux RSS. ● **cells**: Ces classes fournissent des fonctionnalités de type controller en miniature pour créer des components avec une UI indépendante. Regardez la documentation View Cells pour plus d’informations.

4. Mise à jour du logiciel (update)

(updated 14/01/19 - EP)

Si vous venez juste d'installer le logiciel, vous pouvez passer directement au chapitre suivant.

Une fois le logiciel installé, il est très facile de le mettre à jour à la dernière version disponible.

Pour information, le fichier README-LABINVENT.md contient la liste des mises à jour.

1) Noter votre version actuelle

Voir sur la page d'accueil en bas à droite

2) Mettre à jour le code source (et la base de données si nécessaire)

Il suffit d'exécuter le script "./**UPDATE**" depuis la racine du projet :

```
$ cd LABINVENT/
$ chmod +x UPDATE
$ ./UPDATE
```

Vérifiez que la version du logiciel a bien changé (en bas à droite de la page d'accueil).

Ce script fait deux choses pour vous :

- il met à jour le code source du logiciel (avec un simple "git pull")
- il met à jour la base de données (seulement si nécessaire)

*NB: Ensuite, il est préférable de **vider le cache de votre navigateur** (idem pour tous les utilisateurs du logiciel) afin qu'il utilise bien la dernière version des pages web et des scripts javascript :*

- *pour cela, quand vous êtes devant une page web de labinvent, taper la combinaison de touches suivantes pour obliger votre navigateur à vider son cache : **Maj + Ctrl + R** (on peut aussi essayer Ctrl + F5 ou Maj + F5) (sur Mac, remplacer la touche Ctrl par ⌘)*

- *pour **effacer complètement le cache** (méthode plus radicale et donc plus efficace), on peut tenter la combinaison de touches `Ctrl` + `Maj` + `Suppr` (sur Mac, remplacer la touche Ctrl par ⌘)*

Si nécessaire, vous pouvez aussi **mettre à jour MANUELLEMENT la base de données si vous le souhaitez. Attention toutefois,** ceci n'est utile que si la mise à jour AUTOMATIQUE (ci-dessus) n'a pas été suffisante.

Pour cela, aller dans le dossier database/update/

Exécuter TOUS les scripts qui sont d'une date postérieure à la date de votre version (notée à l'étape 1)

Attention, il faut les exécuter **dans l'ordre chronologique**, un par un

Exemple:

$ ./db-update-2016-06-03.sh

$ ./db-update-2016-06-06.sh

$ ./db-update-2016-06-07.sh

3) Tester que tout fonctionne bien

Aller à la racine du projet.

```
$ ./TESTS.sh
```

Si jamais les tests ne veulent pas s'exécuter, essayer ceci:

```
$ cd install/
$ ./plugins_update.sh
(équivalent à faire "php composer.phar update", ce qui met à jour les plugins, et notamment CakePhp)
(et si ça ne marche toujours pas, on peut aussi essayer ./plugins_install.sh)
$ ./TESTS.sh
```

Tous les tests doivent passer

NB: Si phpunit ne se met pas à jour, ne pas hésiter à supprimer tout le dossier vendor/phpunit/ avant de lancer la commande ./plugins_update.sh

5. Accès local à l'application web

Comment accéder à l'application web LabInvent depuis le navigateur web du poste d'installation ?

Maintenant que l'application est installée localement sur votre pc ou serveur, il y a 3 moyens d'accéder (localement) au site web de l'application LabInvent :

- (1) Soit classiquement via "http://localhost/labinvent"
- (2) Soit via un port dédié tel que par exemple "http://localhost:8081"
- (3) Soit via un nom de serveur dédié tel que par exemple "http://labinvent.test"

Voici comment faire dans chacun de ces 3 cas :

- **Cas (1) :**
  - soit vous déplacez le dossier "labinvent/" directement dans le repertoire des sites webs du serveur web local (DocumentRoot, en général c'est /var/www/html/ ou /var/www/html/localhost/public_html/, ...)
  - soit vous laissez ce dossier là où il est et :
    - vous faites un lien depuis votre DocumentRoot (/var/www/html/ ou autre) vers ce dossier labinvent/
    - ou bien vous indiquez au serveur web où le trouver via une directive dans le fichier de configuration du serveur web httpd.conf :

```
Alias /labinvent /Users/epallier/Documents/_W_more/PROJECTS/LABINVENT/SOURCE/labinvent2

<Directory /Users/epallier/Documents/_W_more/PROJECTS/LABINVENT/SOURCE/labinvent2/>
    Options FollowSymLinks
    AllowOverride All
    Require all granted
</Directory>
```

- **Cas (2)** : utiliser un Virtual Host avec un port dédié (ici 8081) :

```
Listen 8081
<VirtualHost *:8081>
<Directory /PROJECTS/LABINVENT/SOURCE/labinvent2>
   Options FollowSymLinks
   AllowOverride All
   Require all granted
</Directory>
      ServerAdmin epallier AT irap...
      DocumentRoot "/PROJECTS/LABINVENT/SOURCE/labinvent2"
      # logs are relative to <ServerRoot> :
      ErrorLog logs/labinvent2.error_log
      CustomLog logs/labinvent2.access_log combined
</VirtualHost>
```

- **Cas (3)** : utiliser un Virtual Host avec un nom de serveur (ici labinvent.test) :

  Il faut d'abord ajouter le nom du serveur (ici labinvent.test) à la fin de votre fichier **/etc/hosts** :

```
127.0.0.1       labinvent.test
```

  Puis créer un hôte virtuel de ce genre :

```
<VirtualHost *:80>
      ServerName labinvent.test
      ServerAdmin votre-email@toto.com
      # Mettre ici le chemin vers votre dossier labinvent :
      DocumentRoot "/home/labinv/PROJECTS/LABINVENT/labinvent"
```

```
        # Logs are relative to <ServerRoot> :
        #ErrorLog logs/labinvent2.error_log
        #CustomLog logs/labinvent2.access_log combined
        ErrorLog ${APACHE_LOG_DIR}/labinvent_error.log
        CustomLog ${APACHE_LOG_DIR}/labinvent_access.log combined
        # ATTENTION, mettre ici exactement le meme chemin que
        # dans DocumentRoot ci-dessus (vers votre dossier labinvent) :
        <Directory /home/labinv/PROJECTS/LABINVENT/labinvent>
                Options FollowSymLinks
                AllowOverride All
                Require all granted
        </Directory>
</VirtualHost>
```

(pour **Ubuntu**, ce texte doit être ajouté à la fin du fichier /etc/apache2/sites-enabled/000-default.conf)

Dans les 3 cas, redémarrer le serveur web suite à ces modifications :

Sur Ubuntu :
$ sudo systemctl restart apache2

Passez maintenant au chapitre suivant pour exploiter cette configuration.

6. Vérification de la conformité du site web LabInvent

Le site est installé, il vous faut maintenant le configurer.

Si vous voulez le conserver, le script de création de la BD est **database/build.sql** (s'il existait déjà avant votre installation, il a été sauvegardé dans build.sql.ORIG).

Votre fichier de configuration est **config/app.php** (s'il existait déjà avant votre installation, il a été sauvegardé dans app.php.ORIG).

Tester l'accès web à l'application :

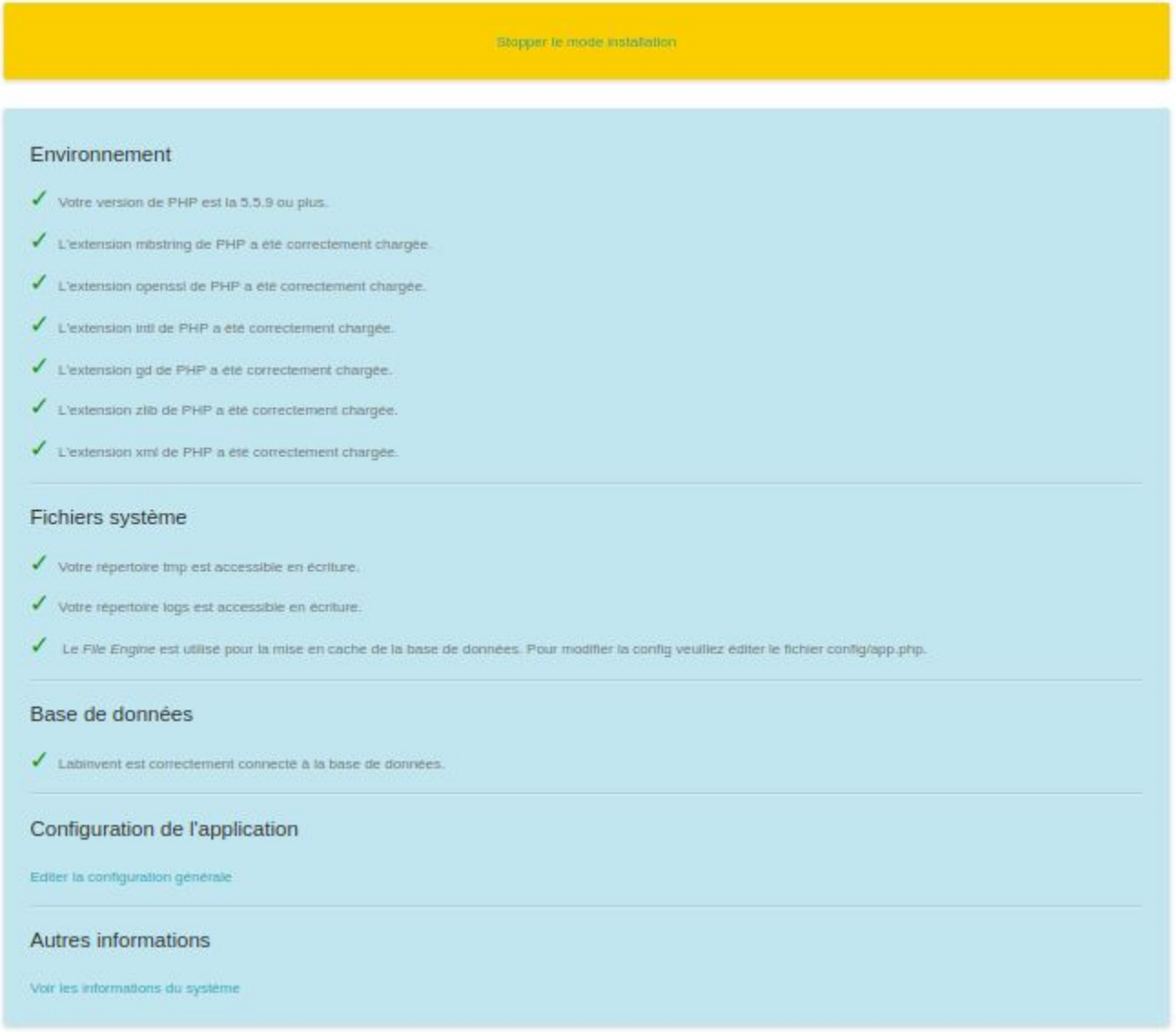
- http://localhost/labinvent (cas 1)

- ou bien http://localhost:8081 (cas 2)

- ou bien http://labinvent.test (cas 3, voir chapitre précédent)

Vous devriez arriver par défaut sur une page correspondant au mode install, elle vous indique la bonne configuration de l'application et de ses dépendances.

Vérifier simplement que **TOUS LES POINTS SONT AU VERT**, comme ci-dessous :



(La page affichera un message si l'URL-Rewriting ne fonctionne pas correctement).
Si tout est au vert, passez au chapitre suivant (Fin de l'installation).

Sinon (il y a des points rouges), reportez-vous ci-dessous à l' (ou les) extension(s) qui pose problème, pour l'installer ou la configurer :

- **Conformité de votre php.ini :**

  Taper la commande suivante pour voir si tout va bien avec votre fichier php.ini

  $ php -i > /dev/null

  Si aucun message ne s'affiche, tout va bien.

  Par contre, si vous obtenez ce genre de warning concernant votre "timezone" :

  *PHP Warning: Unknown: It is not safe to rely on the system's timezone settings. You are *required* to use the date.timezone setting or the date_default_timezone_set() function. In case you used any of those methods and you are still getting this warning, you most likely misspelled the timezone identifier. We selected the timezone 'UTC' for now, but please set date.timezone to select your timezone. in Unknown on line 0*

  Alors, vous devez définir votre fuseau horaire dans le fichier php.ini :

  date.timezone = Europe/Paris

  Pour savoir où est le fichier php.ini:

  $php -r "print phpinfo();" | grep ".ini"

  (sur XAMPP, c'est dans /Applications/XAMPP/xamppfiles/etc/php.ini)

  ⇒ Maintenant, recharger la configuration du serveur Web, et recharger la page web pour voir si le problème est réglé :

  $ sudo service httpd reload

  (Ubuntu: $ sudo systemctl restart apache2)

  (CentOS: $ sudo systemctl reload httpd)

  (MacOS: $ sudo apachectl restart)

  ($ sudo service apache2 restart

- **URL-Rewriting (autoriser les fichiers .htaccess)**

  Pour info, page dédiée pour Cakephp : https://book.cakephp.org/3/fr/installation.html#url-rewriting

  Activer le mode rewrite de Apache:

  (Ubuntu)

$ sudo a2enmod rewrite

Allez aussi dans le fichier /etc/apache2/apache2.conf (ou httpd.conf sur CentOS), ou dans votre configuration “virtual host”, et vérifier que la propriété 'AllowOverride' soit à la valeur 'All' pour le dossier Labinvent :

```
#<Directory votre-chemin-vers-labinvent>
<Directory /var/www/html/labinvent>
        Options FollowSymLinks
        AllowOverride All
</Directory>
```

⇒ Recharger la configuration du serveur Web, et recharger la page web pour voir si le problème est réglé

Si ça ne règle pas le problème, il est aussi possible que le mode rewrite ne fonctionne pas correctement si les fichiers .htaccess dans la racine et dans webroot disparaissent. Sur certains systèmes les fichiers commençant par un point peuvent disparaître lors d'une copie. Si c'est le cas pour vous, voici le contenu de ces deux fichiers et leurs emplacements :

Fichier ./.htaccess :

```
<IfModule mod_rewrite.c>
   RewriteEngine on
   RewriteRule    ^$    webroot/    [L]
   RewriteRule    (.*) webroot/$1    [L]
</IfModule>
```

Fichier ./webroot/.htaccess :

```
<IfModule mod_rewrite.c>
   RewriteEngine On
   RewriteCond %{REQUEST_FILENAME} !-f
   RewriteRule ^ index.php [L]
</IfModule>
```

- **Extension PHP "intl"**:

  Vérifier qu'elle est activée dans le php.ini:

  ```
  $ php -ini | grep intl
  ```

  Sinon, l'ajouter dans le php.ini:
  ```
  extension=intl.so
  ```

  Vérifier qu'elle est bien installée :

  $ php -m | grep intl
  (ça devrait retourner "intl")

  Sinon, l'installer:

  Ubuntu:
  $ sudo apt install php-intl
  (pour chercher un package, taper “apt search php-intl”)

  CentOS:
  $ sudo yum install php-intl

  MacOS avec brew:
  Installer l'extension
  Les extensions sont recherchées dans /usr/local/Cellar/php72/7.2.0_11/lib/php/
  $ brew install php72-intl
  Fichier créé /usr/local/etc/php/7.2/conf.d/ext-intl.ini donne chemin de l'extension : /usr/local/opt/php72-intl/intl.so
  (Pour info, le fichier php.ini est dans /usr/local/etc/php/7.2/php.ini, mais c'est inutile de le modifier)

  MacOS avec XAMPP:
  cf http://stackoverflow.com/questions/27886117/php-intl-installation-on-xampp
  Il se pourrait que vous ayez besoin de ré-exécuter l'installeur de XAMPP afin de cocher l'option "XAMPP Developer Files", si vous ne l'aviez pas déjà fait lors de votre installation de XAMPP
  cd /Applications/XAMPP/bin

sudo ./pecl install intl
(Vérifier qu'elle a bien été installée dans /Applications/XAMPP/xamppfiles/lib/php/extensions/no-debug-non-zts-20131226/)
Attention, cette méthode semble ne plus fonctionner avec XAMPP 7 (qui inclut php 7).
Sur Mac, j'ai dû installer l'extension intl via macport :
sudo port install php71-intl
Puis copie de cette extension dans le dossier de XAMPP :
sudo cp /opt/local/lib/php71/extensions/no-debug-non-zts-20160303/intl.so
XAMPP_716-0/xamppfiles/lib/php/extensions/no-debug-non-zts-20160303/

⇒ Recharger la configuration du serveur Web, et recharger la page web pour voir si le problème est réglé

- **Extension PHP "mbstring" et librairie "zlib"**

La collection de librairies est normalement déjà installée avec PHP en tant que dépendance.
Sinon, faire :
(Ubuntu) :
$ sudo apt-get install libapache2-mod-php5
⇒ Recharger la configuration du serveur Web, et recharger la page web pour voir si le problème est réglé

- **Librairie PHP "php-gd"**

$ sudo apt-get install php5-gd

⇒ Recharger la configuration du serveur Web, et recharger la page web pour voir si le problème est réglé
Si ce n'est toujours pas le cas, aller dans le fichier /etc/php5/apache2/php.ini et vérifier que la ligne suivante est présente et décommentée (absence de ";" devant la ligne), sinon il faut la rajouter :
extension=gd.so
⇒ Recharger la configuration du serveur Web, et recharger la page web pour voir si le problème est réglé

- **KeepAlive:**

Editer le fichier de configuration du serveur web (**/etc/apache2/apache2.conf** sur Ubuntu, /etc/httpd/conf/http.conf sur CentOS/Fedora) et Mettre le paramêtre "KeepAlive" à "Off" :

```
KeepAlive Off
```

- **<u>event et prefork modules:</u>**

  Par défaut Apache utilise un "event module" et PHP un "prefork module".
  Il faut désactiver le 1er module et activer le second :

```
$ sudo a2dismod mpm_event
$ sudo a2enmod mpm_prefork
```

7. Test de l’application

(updated 24/01/20 - EP)

Ces tests sont à exécuter après chaque modification (mais aussi après une nouvelle installation), et surtout avant tout “git commit”, pour éviter toute régression (vérifier que ce qui fonctionnait avant continue de fonctionner).

- **Verifier que la suite de tests passe complètement**

  Placez-vous à la racine du projet, exécutez la commande :

  $ ./TESTS.sh
  (équivalent à 'vendor/bin/phpunit' ; si ça ne marche pas, essayez 'vendor/phpunit/phpunit/phpunit')
  (Php7 utilise Phpunit 6 alors que Php 5 utilise Phpunit PHPUnit 5.7)

  A la fin, vous devez voir quelque chose du genre :

  Time: 28.5 seconds, Memory: 32.00MB

  OK (99 tests, 515 assertions)

- **Temps d’exécution constaté sur différentes plateforme**
  - Sur une VM Ubuntu18 installée sur un MacBook Pro (15-inch, 2017) 2.9GHz Core i7 (fait le 21/1/20)
    - Avec php7.2, les 99 tests s'exécutent en un peu moins de 50 secondes
  - Sur Mac OS 10.14 avec brew, sur un MacBook Pro (15-inch, 2017) 2.9GHz Core i7 (fait le 15/1/20)
    - Avec php7.2, les 99 tests s'exécutent en 29 secondes
  - Sur Mac OS 10.13 avec brew (fait fin 2018) :
    - Avec php5.6, les 94 tests s'exécutent en 55 secondes
    - Avec php7.2, les 94 tests s'exécutent en 49 secondes
  - Sur Mac OS 10.12 avec XAMPP (fait en 2017) :

        - En php5.6, les 53 tests s'exécutent en 25-26 secondes
        - En php7.1, les 53 tests s'exécutent en 9-10 secondes, soit 2 à 3 fois plus vite !!! (vive Php 7)
    - Sur ScientificLinux (CentOS) 6.4 (serveur de test pweb) :
        - En php5.6, les 53 tests s'exécutent en 20 secondes
        - En php7.1, les 53 tests s'exécutent en 16-17 secondes, soit pas beaucoup plus rapide, bizarre...
    - Sur CentOS 6.4 (serveur de production), dans une VM :
        - avec php5.6 avec phpunit 5.7, les 53 tests s'exécutent en 47 secondes !!!
        - avec php7.1, les 53 tests s'exécutent en ??? (Php7, c'est pour très bientôt...)

- **Remarque : si jamais vous voulez n'exécuter que certains tests**

  Marquer le(s) test voulu avec une annotation :

```
/**
 * @group failing
 * Tests the api edit form
 */
public function testEditAction()
```

  Ensuite, exécuter avec :

```
$ phpunit --group failing
```

  Attention, ne pas oublier d'enlever ces annotations avant de commiter avec git !

  On peut attribuer plusieurs groupes à un test :

```
/**
 * @group failing
 * @group bug2204
 */
public function testSomethingElse()
```

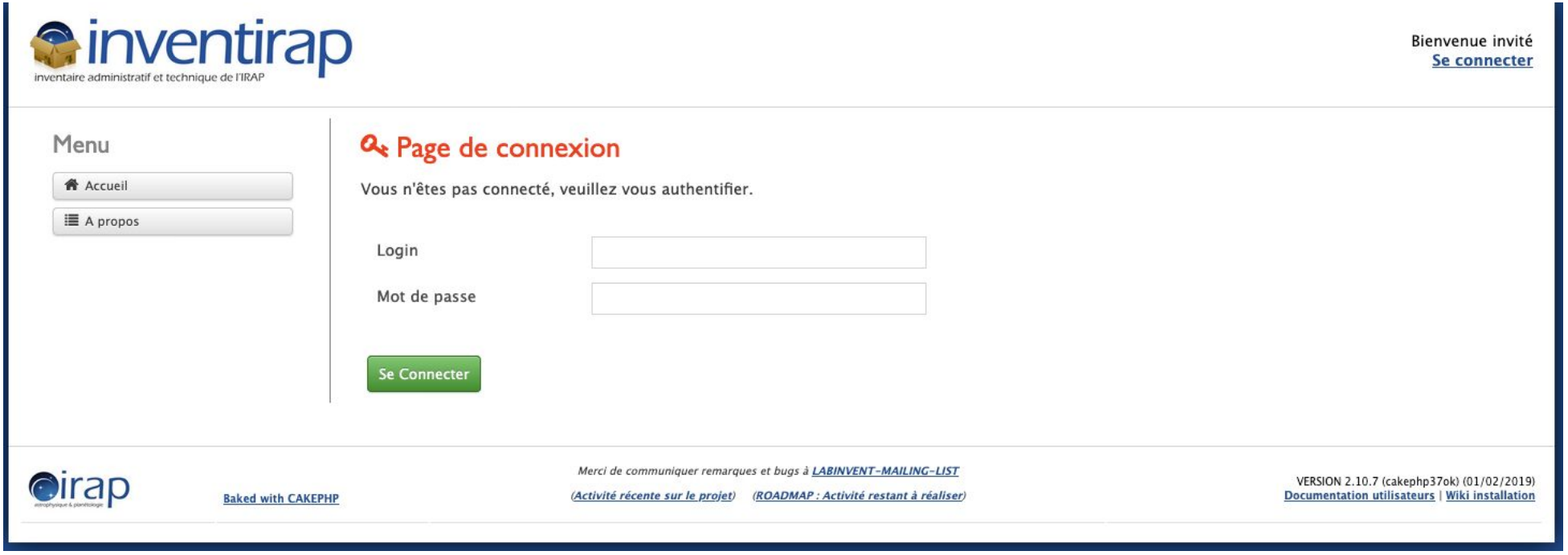
8. Fin de la phase installation, première connexion

(updated 22/01/20 - EP)

Si vous n'y êtes pas déjà, allez sur la page d'accueil de LabInvent : http://localhost/labinvent, ou bien http://localhost:8081, ou encore http://labinvent.test

Maintenant, sortez du mode "installation" en cliquant sur le bouton "Stopper le mode installation"
Si vous voulez revenir au mode "installation" plus tard, allez dans le dossier database/ et exécutez simplement ./mode_panique.sh (./mode_panique_macos.sh sur Mac OS) puis revenez sur la page d'accueil.

Vous arrivez alors sur la page de connexion suivante :



Si vous êtes en train d'installer une **version de "production"**, alors il vaut mieux enlever le mode DEBUG général.
Par contre, si vous installez une **version de "développement"**, alors vous pouvez laisser le logiciel en mode DEBUG général.
(le mode DEBUG général est différent de celui proposé dans le chapitre suivant sur la "configuration")

Pour désactiver (manuellement) le mode DEBUG général, allez à la racine du projet, et ouvrez votre fichier de configuration générale de labinvent (**config/app.php**).
Allez environ à la ligne 12 (au début du fichier):

```
'debug' => filter_var(env('DEBUG', true), FILTER_VALIDATE_BOOLEAN),
```

Et remplacez juste le **true** par **false**, comme ceci :

```
'debug' => filter_var(env('DEBUG', false), FILTER_VALIDATE_BOOLEAN),
```

Le logiciel est configuré par défaut en **mode "SANS LDAP"**, ce qui veut dire que les utilisateurs sont définis localement dans le logiciel, il n'y a pas de connexion LDAP. Vous pourrez changer ça plus tard si vous le souhaitez, dans la configuration, au chapitre suivant.

Donc pour l'instant, **connectez-vous à l'aide du login utilisateur créé lors de l'installation (par défaut "superadmin"),** avec le mot de passe "**login**". Il vous permet de vous connecter en tant que "super administrateur" **pour pouvoir configurer l'application**.
Vérifiez que c'est bien le cas : sur la page d'accueil, vous devriez avoir une ligne qui affiche "Vous êtes connecté en tant que *Nom Prénom* avec le niveau d'authentification Super Administrateur."

NB1 : Les autres comptes utilisateurs que le script d'installation aura éventuellement créés (si demandé lors de l'installation) ont aussi le même mot de passe par défaut : "login"
NB2 : Il y a toujours un utilisateur par défaut en mode "SANS LDAP", il s'agit de l'utilisateur "**_fake_ldap_user_**" avec le mot de passe "**_fake_ldap_user_pass**" (il a le profil minimum, c'est à dire "**Utilisateur**" ; il est utile notamment pour les tests, afin de tester le profil "Utilisateur" accordé par défaut à toute personne du ldap qui n'a pas de privilège particulier)

9. Configuration (Adaptation du logiciel)

(updated 24/01/20 - EP)

Ce chapitre explique comment adapter le logiciel à vos besoins (**seulement si nécessaire**).

9.1. Configuration technique via le fichier de configuration (config/app.php)

Cette configuration permet de modifier les paramètres techniques de base (BDD, emails, mode debug général, …).
Elle se fait via la modification du fichier Php "**config/app.php**".

Pour info, si vous désirez lire un paramètre de configuration dans le code source (dynamiquement), c'est très simple :

```
// Lecture de la valeur du paramètre "debug"
if ( Configure::read('debug') ) …
```

On pourrait ajouter d'autres paramètres dans ce fichier, selon le besoin, mais en général on préférera passer par la BDD qui permet de configurer les paramètres via une simple page web. C'est ce qui est présenté dans la section suivante.

9.2. Configuration via les pages web de configuration

La plus grosse partie de la configuration se fait directement **via des pages web** qui interagissent avec une **table de la base de données** (nommée "**configurations**").
Ces pages sont accessibles via le menu **Outils** :

- **Configuration générale** de l'application : pour configurer le LDAP, les emails, le nom du labo, etc...
- **Gérer le contenu variable** de l'application : pour gérer les catégories de matériels, les sites, les organismes, les types de suivis, les groupes thématiques et métier, les types de docs, les fournisseurs, les unités et les formules (module métrologie), ...
- **Gérer les utilisateurs :** pour ajouter, modifier, ou supprimer des utilisateurs
- etc.

9.2.1. Configuration des paramètres techniques (LDAP, mail, debug…)

9.2.1.1. Gestion de l'authentification des utilisateurs via un annuaire LDAP

Labinvent peut être utilisé avec un annuaire LDAP. Si c'est ce que vous voulez, voici comment faire.

ATTENTION : après être passé en mode LDAP, vous risquez de ne plus pouvoir vous connecter à l'application !!!

Assurez-vous donc d'abord que votre login super administrateur (par défaut "superadmin") porte le MEME NOM que votre login ldap. Si c'est actuellement "superadmin", remplacez-le par votre login ldap. Pour cela, allez dans le menu "Outils", puis "Gérer les utilisateurs privilégiés", éditez l'utilisateur correspondant au super administrateur actuel, et changez son login et son mot de passe pour qu'ils correspondent exactement à ceux du LDAP. Déconnectez-vous puis reconnectez-vous avec ce nouveau login et vérifiez que vous êtes bien "super administrateur".

Maintenant que c'est fait, vous pouvez passer en mode LDAP :

- Si ce n'est pas déjà fait, connectez-vous à l'aide de votre login super administrateur (qui doit maintenant correspondre à votre login LDAP)
- Dans le menu de gauche, cliquez sur le lien "Outils", puis sur "Configuration générale de l'application", puis sur "Editer la configuration"
- Cochez "Utilisation du LDAP". Les options de configuration du LDAP s'affichent alors. Modifiez-les selon vos besoins
- Cliquez sur "Valider"
- Une fois la connexion au LDAP configurée, toute personne enregistrée dans l'annuaire LDAP peut alors se connecter au logiciel. Vérifiez que vous arrivez à vous connecter avec votre login LDAP qui a le profil super administrateur
- Par défaut, un utilisateur provenant du LDAP a un statut de simple "utilisateur". Pour lui attribuer un rôle supérieur (privilégié), il faut aller dans "Gérer les utilisateurs privilégiés" du menu "Outils" pour l'y ajouter (liste déroulante des utilisateurs du LDAP), en lui attribuant un rôle.

Mode panique : Si vous ne pouvez plus vous connecter à l'application (après être passé en mode LDAP), vous pouvez toujours repasser en mode "installation" comme au début : allez dans le dossier database/ et exécutez simplement ./mode_panique.sh

(./mode_panique_macos.sh sur Mac OS) puis revenez sur la page d'accueil et connectez-vous avec votre login super administrateur ("superadmin" par défaut)

9.2.1.2. Configuration de l'envoi des emails

Si vous désirez activer l'envoi des emails, voici la procédure à suivre.

En tant que SuperAdministrateur, aller dans "Outils" -> "Configuration générale de l'application", cliquez sur "Editer la configuration".

Vous avez alors 2 options :

- "Activer l'envoi des mails général" : active l'envoi d'emails à la liste générale, c'est à dire aux responsables de groupes (thématique, métier, …), aux responsables d'un matériel, au personnel administratif, etc.
- "Activer l'envoi des mails pour la liste spécifique ci-dessous" : active l'envoi d'emails à cette liste spécifique

Les emails sont envoyés automatiquement à chaque opération importante, telle que création de matériel, validation, suppression, etc.

Si ces 2 options sont cochées, les emails sont envoyés aux 2 listes (liste générale et liste spécifique).

L'envoi de mail est réalisé grâce à une adresse qu'il faut créer pour labinvent. Le protocole d'envoi est à définir dans config/app.php selon le serveur choisi. Lors de l'envoi d'un mail il faut utiliser le transport 'dev' en local.

Pour changer de mail et de transport, il faut aller dans la section Email/Transport de config/app.php, et vérifier/adapter a votre convenance le bloc suivant :

```
'EmailTransport' => [
        'default' => [
            'className' => 'Mail',
            // The following keys are used in SMTP transports
            'host' => 'localhost',
            'port' => 25,
            'timeout' => 30,
            'username' => 'user',
            'password' => 'secret',
```

```
            'client' => null,
            'tls' => null,
            'url' => env('EMAIL_TRANSPORT_DEFAULT_URL', null),
        ],
...
```

9.2.1.3. Configuration du mode DEBUG “local”

Si vous désirez activer le mode DEBUG applicatif (ou “local”), voici la procédure à suivre.

En tant que SuperAdministrateur, aller dans le menu "Outils" -> “Passer en mode DEBUG”
Ce mode est différent du mode DEBUG “général” qui est configurable seulement via le fichier config/app.php
Le mode DEBUG “local” permet de mieux comprendre ce qui se passe en cas de problème. Il provoque l’affichage de quelques informations techniques utiles sur chaque page web.

9.2.1.4. Configuration des utilisateurs

Menu Outils / “Gérer les utilisateurs”

9.2.2. Configuration des paramètres métier (liés à l’inventaire)

Via les 2 premiers liens du **menu Outils** (“**Configuration générale** de l'application”, et “**Gérer le contenu variable** de l'application”), vous avez accès à de nombreux paramètres tels que le nom du laboratoire, les catégories de matériels, les sites, les organismes, les types de suivis, les groupes thématiques et métier, les types de docs, les fournisseurs, les unités et les formules (module métrologie), etc...

9.3. Autres types de configurations

9.3.1. Changer les images (logos) du site web

(TODO: éviter que ces images soient commitées...)

Vous pouvez facilement personnaliser les images logos qui s'affichent en haut à gauche et en bas à gauche du site web.
Pour cela, il suffit de remplacer les fichiers JPEG suivants dans webroot/img/ :

- logo_entity.jpg : le logo du laboratoire
- logo_software.jpg : le logo du logiciel

Attention toutefois à bien conserver exactement les mêmes noms.

9.3.2. Modifier le format des étiquettes

Voir le chapître suivant sur les "Etiquettes"

9.4. (dev only) Utilisation des paramètres de configuration dans le code (dynamiquement)

- **dans un contrôleur** (ex: src/Controller/Pages/PagesController.php) :
  - // La variable $this->**confLabinvent** est **définie** dans la classe mère **AppController** (pour que tous les contrôleurs en héritent), dans la fonction **initialize**(), comme ceci :
    $this->confLabinvent = **TableRegistry::get('Configurations')->find()->first();**
  - // Elle est donc utilisable dans n'importe quel contrôleur
    // ex: pour savoir si on est en mode INSTALL ou pas :
    $configuration = $this->**confLabinvent**
    if ($configuration->mode_install) …
  - // Elle est aussi **passée à TOUTES les vues** (templates) via la fonction **beforeRender**() de AppController, sous la forme d'une variable nommée $**configuration** :

```
$configuration = $this->confLabinvent;
$this->set('configuration', $configuration);
```

- **dans un template** (ex: src/Template/Pages/tools_sm.ctp), la variable $configuration (passée à toutes les vues par AppController comme expliqué ci-dessus) est disponible :

```
// ex: pour savoir si la variable de configuration “metrologie” est à True ou False :
if ($configuration->metrologie) ...
```

9.5. (dev only) Comment ajouter un nouveau paramètre de configuration

(TODO: utiliser le principe des “Migrations” cakephp, ça éviterait d’avoir à toucher la BD à la mano)

Si vous désirez ajouter un nouveau paramètre de configuration, voici la procédure à suivre.

Prenons l’exemple suivant : on veut ajouter une option “format d’étiquette” pour permettre à un laboratoire de choisir son format d’étiquette à imprimer. Pour cela, on va ajouter un nouveau champ nommé “label_format_num”.

a - (Optionnel) Ajouter une entrée pour valider le nouveau champ “label_format_num” de la table configurations

Editer le fichier src/Model/Table/ConfigurationsTable.php
Dans la fonction validationDefault(), ajouter une ligne :

```
$validator->allowEmpty(label_format_num);
```

b - Ajouter une colonne “label_format_num” dans la table configurations

On peut utiliser phpmyadmin pour ça, ou encore la commande “mysql -u” en mode terminal.
Exécuter cette requete SQL :

```
ALTER TABLE `configurations` ADD `label_format_num` INT(3) UNSIGNED NOT NULL DEFAULT '1' COMMENT 'numero de format etiquette' AFTER `hasPrinter`;
```

c - Ajouter cette requete SQL dans un fichier de mise à jour de la BD

Ceci, afin que les autres utilisateurs du logiciel puissent hériter (automatiquement) de cette nouvelle fonctionnalité lors de la prochaine exécution du script UPDATE.
Aller dans database/update/

Faire une copie du DERNIER script BASH présent, par exemple db-update-2019-01-09.sh, et le nommer à la date du jour : db-update-2019-03-14.sh
Pas besoin de modifier ce fichier, il faut juste qu'il soit présent.

Ce fichier exécutera sur la BD la requete SQL du même nom qui se trouve dans le sous-dossier script_sql/.
Donc, dans ce sous-dossier, de la même façon, faire une copie du DERNIER script SQL, par exemple db-update-2019-01-09.sql et le nommer à la date du jour : db-update-2019-03-14.sql
Attention, les 2 scripts (bash et sql) doivent être à la MEME DATE.
Dans votre script SQL db-update-2019-03-14.sql, ajouter le code SQL de l'étape précédente.
Ce script doit donc avoir le contenu suivant :

```
use database;

ALTER TABLE `configurations` ADD `label_format_num` INT(3) UNSIGNED NOT NULL DEFAULT '1' COMMENT 'numero de format etiquette' AFTER `hasPrinter`;
```

d - Ajouter le champ "label_format_num" dans la vue d'édition de la table "configurations"

Editer le fichier src/Template/Configurations/**edit**.ctp
Ajouter ces lignes sous la ligne qui concerne le champ "hasPrinter" :

```
echo $this->Form->control('label_format_num', [
    'options' => [
        '1' => 1,
        '2' => 2,
        '3' => 3,
        '4' => 4,
        '5' => 5,
        '6' => 6,
```

```
            ],
            'label' => 'Numéro Format Etiquette'
        ]);
```

e - Ajouter le champ “label_format_num” dans la vue détaillée de la table “configurations”

Editer le fichier src/Template/Configurations/**view**.ctp

Sous la ligne qui concerne l’imprimante (“Imprimante disponible”), ajouter la ligne suivante :

```
$displayElement(__('Numéro format étiquette'), h($configurationObj->label_format_num));
```

f - Utiliser cette nouvelle option dans le code qui en dépend

En l’occurence, c’est le contrôleur des matériels qui utilise cette option.
Editer le fichier src/Controller/MaterielsController.php
Aller dans la fonction printLabel()
Ajouter cette ligne pour récupérer la valeur actuelle de l’option (dans la table configurations) :

```
$label_format_num = $this->confLabinvent->label_format_num;
```

g - Documenter cette nouvelle option dans CE document

Si si, c’est important ça, vous êtes pas tout seul au monde, m’enfin, allez, au boulot !!!
https://tinyurl.com/labinvent

Puis, faire une copie PDF “Labinvent Documentation.pdf” de ce document (Google Doc) et la mettre dans le dossier doc/

h - Commiter tous ces changements sur le dépôt GIT central du projet

$ git pull

$ git add .
$ git commit -m “Ajout d’une nouvelle option pour choisir le format d’étiquette”
$ git push

Voilà, les laboratoires qui voudront récupérer cette nouvelle option auront seulement à exécuter le script “./UPDATE” (à la racine du projet).
Elle est pas belle la vie ?

10. Etiquettes (optionnel)

(updated 23/09/19 - EP)
\etiquette \etiqueteuse \titreuse

Si la fonction d'impression d'étiquettes (ruban) de LabInvent vous intéresse, voici ce que vous devez faire

10.1. Etiqueteuse (moins de 100€ HT), Thermal Transfert, USB

Vous devez acheter une étiqueteuse (titreuse) "Imprimante d'étiquettes - **Dymo** - **LabelManager PnP** - USB" et la brancher sur le port USB d'un poste (client) Windows ou Mac (pas de driver pour linux).

Description technique :

Pas de logiciel ni de pilote à installer. Le logiciel intégré s'ouvre à l'écran, prêt à l'emploi.

Garantie 2 ans.
Fonctionne avec les rubans D1 6, 9 et 12 mm.
Petite et compacte, elle trouve facilement sa place sur un bureau.
Batterie lithium-ion fournie, rechargeable par USB - pas d'adaptateur secteur ni de piles.
Personnalisez vos étiquettes avec les polices et graphiques de votre ordinateur.
Connectez-là à votre PC ou Mac et imprimez instantanément et très facilement des étiquettes professionnelles !
Connexion USB à votre PC ou Mac.

Lien chez Lyreco

Lien chez OfficeDepot

Liens chez constructeur DYMO :
- Lien direct vers la titreuse
- Etiqueteuse et etiquettes compatibles (D1)

ETIQUETEUSES PERMETTANT D'IMPRIMER SUR ETIQUETTES PLUS GROSSES (ruban 19mm ou 24mm)

⇒ **Tous les rubans compatibles** (D1 de largeur 6 à 24 mm)

Etiqueteuse	Prix (2019)	Connexion PC ?	Sans fil ?	Taille Ruban (D1)	Logiciel
MobileLabeler **SKU: 1978243** **(300 dpi/ppp)** Ref LYRECO	160€	USB	BlueTooth	6, 9, 12, 19, 24mm	DCD/DLS Il existe aussi un application mobile (compatible avec Smartphones iOS/Android et tablettes)
LabelManager 420P	135€	USB	NON	6, 9, 12, 19mm	

(180 dpi/ppp)					
LabelManager 500TS **(300 dpi/ppp)**	250€	USB	NON	6, 9, 12, 19, 24mm	

Etiqueteuse utilisée par le CRAL (mais déjà retirée du marché) : Dymo LabelManager PCII
Elle utilise exactement le même logiciel mais permet de mettre des **rubans plus larges de 19mm** (ce qui est le minimum acceptable pour l'impression correcte du QrCode par exemple) :
Si vous installez cette titreuse, il vous faudra sélectionner le format d'étiquette numéro 2 dans la configuration du logiciel LabInvent.
https://www.ldlc.com/fiche/PB00102244.html

10.2. RUBAN (D1)

La technologie à base de **ruban** (TT, Transfert Thermique) a été préférée à la technologie à base **d'étiquette** (TD, Transfert Direct) car leur durée de vie est bien plus longue. Voici une comparaison des 2 techniques (et une autre comparaison ici).
C'est dommage car il n'existe pas (en 2019) d'étiqueteuse ruban en wifi, alors qu'une étiqueteuse étiquette wifi existe, ce qui permettrait de mettre à disposition de tous une étiqueteuse sur le réseau, sans avoir besoin de la connecter à un PC dédié avec partage réseau via une connexion USB...

Il faut donc commander le **ruban** suivant : DYMO **D1 en 12mm** (ou 19mm, ou 24mm si votre étiqueteuse le permet)

10.3. Installation du logiciel d'impression (DCD) pour Win (7+) ou Mac (10.9+)

Pour pouvoir utiliser cette étiqueteuse (à ruban) avec le logiciel LabInvent, vous devez installer le logiciel **DCD** (**"**Dymo Connect for Desktop") **(**remplaçant du logiciel DLS **"**DYMO LABEL SOFTWARE"**)**

DLS est toujours utilisé sur MacOSX car DCD n'existe pour l'instant que pour Windows (oct 2019).

La dernière version testée (début 2019) sur MacOSX, Win7, et Win10 avec LabInvent est DLS v8.6.1 (DLS 8.5.4 aussi testé ok sur Win7)

Page de téléchargement du logiciel

Plus de versions

10.4. Activation de la fonction d'impression sur LabInvent

Enfin, pour pouvoir étiqueter vos matériels depuis LabInvent, vous devez cocher "Imprimante disponible" dans la section "Divers" de la page de configuration générale (Outils/Configuration générale de l'application).

10.5. Adaptation des étiquettes au besoin du laboratoire

a - Créer votre étiquette idéale avec le logiciel propriétaire DLS

Ouvrir le logiciel Dymo Label Software.
Choisir le format d'étiquette voulu dans le panneau de gauche (12mm étant celui utilisé par défaut pour LabInvent).
Créer votre étiquette comme souhaitée dans le panneau de droite.
Enregistrer cette étiquette dans un dossier quelconque avec "Fichier/Enregistrer sous…"
Ce fichier porte l'extension ".label" et est au format XML.
Voici un extrait du début d'un fichier de ce type :

```
<?xml version="1.0" encoding="utf-8"?>
<ContinuousLabel Version="8.0" Units="twips">
  <PaperOrientation>Landscape</PaperOrientation>
  <Id>Tape12mm</Id>
  <PaperName>12mm</PaperName>
  <LengthMode>Auto</LengthMode>
  <LabelLength>0</LabelLength>
  <RootCell>
        <Length>0</Length>
        <LengthMode>Auto</LengthMode>
```

b - Créer votre fonction étiquette dans le code source du logiciel LabInvent (copier le contenu XML de votre fichier étiquette)

Oui, je sais, c'est pas du tout l'idéal, mais je sais pas comment faire mieux pour l'instant.
Aller dans le contrôleur des matériels (src/Controller/**MaterielsController.php)**
Faire une copie de la fonction **etiquette_format1**() qui est la définition du format d'étiquette pour le **laboratoire IRAP**. Cette fonction sera automatiquement appelée par la fonction printLabel().
Nommer cette nouvelle fonction etiquette_format2() (ou etiquette_format3... si etiquette_format2 existe déjà).
Modifier les commentaires juste au-dessus de cette fonction pour décrire le format (ruban 12mm, 1, 2, ou 3 lignes, avec ou sans logo...)

Dans VOTRE nouvelle fonction **etiquette_format2**(), Il faut copier TOUT le contenu XML de votre fichier étiquette, à la place du code existant.
Attention toutefois, cette fonction utilise 4 paramètres qui permettent de faire varier le contenu de l'étiquette :

- $numeroLab
- $organisme
- $dateAcquisition
- $numeroInventaireOrganisme

Ce sont ces paramètres qui sont imprimés par défaut sur les étiquettes.
Si vous voulez imprimer d'autres paramètres, il faudra les changer, et vous assurer qu'ils sont bien passés à cette fonction lors de son appel dans la fonction printLabel()

Sur l'image ci-dessous de l'étiquette IRAP, on remarque qu'elle est organisée en 2 colonnes :

- une première colonne (à gauche sur l'étiquette) contenant le texte (formé des 4 paramètres mentionnés ci-dessus).
- une deuxième colonne (à droite sur l'étiquette) contenant une image, le logo IRAP

Voir ci-dessous le contenu XML correspondant à cette étiquette. On retrouve ce contenu dans la fonction **etiquette_format1**().
Dans ce code XML, les colonnes (vues sur l'étiquette ci-dessus) sont appelées cellules (<**Cell**>).
Il y a donc 2 sections <Cell> :

- une première (de type <**TextObject**>) qui décrit les lignes de texte
- une deuxième (de type <**ImageObject**>) qui décrit l'image (logo IRAP)

Le texte à imprimer est défini au début de la fonction **etiquette_format1() avec $text_line1 et $text_line2**
L'image à imprimer est définie avec $img_logo.
Ce sont ces variables que vous devrez adapter pour vos besoins.

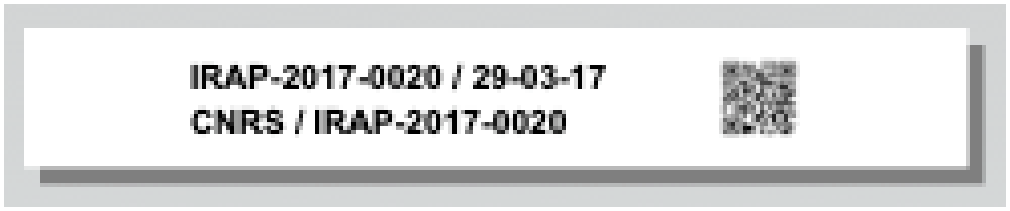
```
<Subcells>

    <Cell>
        <TextObject>
        …
            <StyledText>
                <Element>
                    <String>'
                        .$text_line1 . "\n"
                        .$text_line2 .
                    '</String>
                    <Attributes>
                        <Font Family="Arial" Size="24" Bold="True" Italic="False" Underline="False" Strikeout="False"/>
                        <ForeColor Alpha="255" Red="0" Green="0" Blue="0"/>
                    </Attributes>
                </Element>
            </StyledText>
        </TextObject>
        ...
    </Cell>

    <Cell>
        <ImageObject>
            ...
            <Image>'
                .$img_logo.
            '</Image>
            ...
        </ImageObject>
        …
    </Cell>

</Subcells>
```

Une autre fonction “etiquette” nommée etiquette_format_avec_QRCODE() fournit un exemple avec la même étiquette IRAP ci-dessus mais qui remplace le logo IRAP par le QRCode du matériel. Voici l’étiquette produite alors :



c - Sélectionner votre étiquette dans la partie configuration du logiciel

Une fois votre fonction étiquette **etiquette_format2()** bien définie, vous devez la sélectionner en allant dans le menu configuration du logiciel :

Aller dans le menu “Outils” puis “Configuration générale de l’application”.
Cliquer sur “Editer la configuration”.
Aller dans la section “Divers”.
S’assurer que l’option “Imprimante disponible” est bien cochée.
Dans l’option “**Numéro format étiquette**”, sélectionner le numéro “2” correspondant à votre fonction **etiquette_format2()** que vous avez créée (ou bien, sélectionner “3” pour la fonction etiquette_format3(), “4” pour la fonction etiquette_format4(), etc.)
Cliquer sur le bouton vert “Valider” tout en bas.

d - Tester

Aller sur la vue détaillée d’un matériel. Attention, on ne peut imprimer l’étiquette que d’un matériel **VALIDÉ**.
Cliquer sur le bouton “Impr. etiquette”. Voilà !

10.6. Etiqueteuses installées sur le laboratoire IRAP (IRAP ONLY)

https://projects.irap.omp.eu/projects/inventirap/wiki/Installation#I-Etiquettes-optionnel

11. Workflow général (circuit de saisie)

11.1. Description générale

(updated 12/12/18 - Elodie Bourrec) **(TODO: mettre à jour cette section, obsolète)**

Les materiels figurent dans Labinvent s'ils font plus de 1000 E (pour l'IRAP), ou s'il y a un interet technique a inventorier ce type de matériel.

Un usager veut commander un matériel :
Il va creer une fiche (avec toutes les caracteristiques et description appropri)ee) et avoir un munéro d'inventaire. Le matériel a un statut "Created". Il peut eventuellement etre supprimé.
Avec ce numéro (ou la fiche) il envoie sa demande (avec le devis) a son personnel de gestion qui va emettre le bon de commande, et eventuellement, sur Labinvent, remplir la partie administrative du matériel en question.

Qd le matériel arrive :
Avec le bon de livraison, le matériel peut etre validé (c'est la que certaines questions d'organisation, et de droit sur l'appli, se posent) Les gestionnaires peuvent valider les materiels mais souvent, ils n'en voient que le Bon de livraison. Pour completer la fiche matériel, avec num serie, date de garantie ... il va falloir que les personnes qui l'ont commandé se reconnectent.
Une fois le materiel "validated", il ne peut plus etre supprimé. Il ne peut qu'etre sorti de l'inventaire.
Tous les personnels du labo peuvent demander une sortie d'inventaire, il n'y a que les gestionnaires (profil "administration") qui peuvent faire la sortie effective.
Les materiels restent en base avec le statut "Archived" mais ne sont plus visibles sur les listes de l'appli. Les gestionnaires peuvent voir la liste du materiel archivé.

Il y a plusieurs profils dans l'appli :
User - utilisateur : Mr tout le monde. Il voit tous les materiels mais ne peut modifier que les siens.
Responsable : nous avions pense que les responsables de groupe pouvaient etre valideurs et avoir acces en modification a tous les materiels de leur groupe, mais chez nous, nous ne nous en servons pas. Il faudrait voir avec les autres labos s'ils se servent de ce profil et s'il est fonctionnel.
Administration : profil pour les gestionnaires. La gestion voit tous les materiels et peuvent tout modifier. Ils ont acces en plus a la partie administrative (bon de commande, EOTP, ...)
Super-administrateur: peut tout faire sauf sortir les materiels de l'inventaire.

11.2. Procédure à suivre pour commander un matériel

(updated 17/12/18 - Etienne Pallier) **(TODO: mettre à jour cette section, obsolète)**

Voici le message d'accueil affiché à la connexion d'un utilisateur de LabInvent ayant le profil "utilisateur" (c'est à dire le profil de plus bas niveau, correspondant à un utilisateur lambda, "non privilégié"). Ce message lui indique la marche à suivre (le "workflow" à respecter) pour commander un matériel. Les acteurs (le demandeur et le gestionnaire) sont en jaune, les actions en rouge.

*Voici la procédure (en 8 étapes) pour passer commande d'un matériel de plus de 800€ (matériel **inventoriable**) :*

*(je **peux** aussi, si je le désire, suivre cette procédure pour un matériel < 800€, afin qu'il soit référencé)*

*(je deviens **"demandeur/utilisateur/référent"** de ce matériel)*

1. *J'obtiens un devis*
2. *Je crée une fiche matériel (clic sur "Nouveau Matériel") avec les quelques informations obligatoires (notamment une **description précise** du matériel) (éventuellement, je peux y associer le devis en document attaché)*
3. *J'imprime ma fiche et l'amène (avec le devis) à un gestionnaire (ou bien je l'envoie par email)*
4. *Le gestionnaire retrouve cette fiche (en tapant son n° interne labo dans le champ "Recherche") et la complète avec les infos administratives*
5. *Le gestionnaire crée le bon de commande pour ce matériel et passe la commande (éventuellement, il peut associer le bon de commande à la fiche matériel en document attaché)*

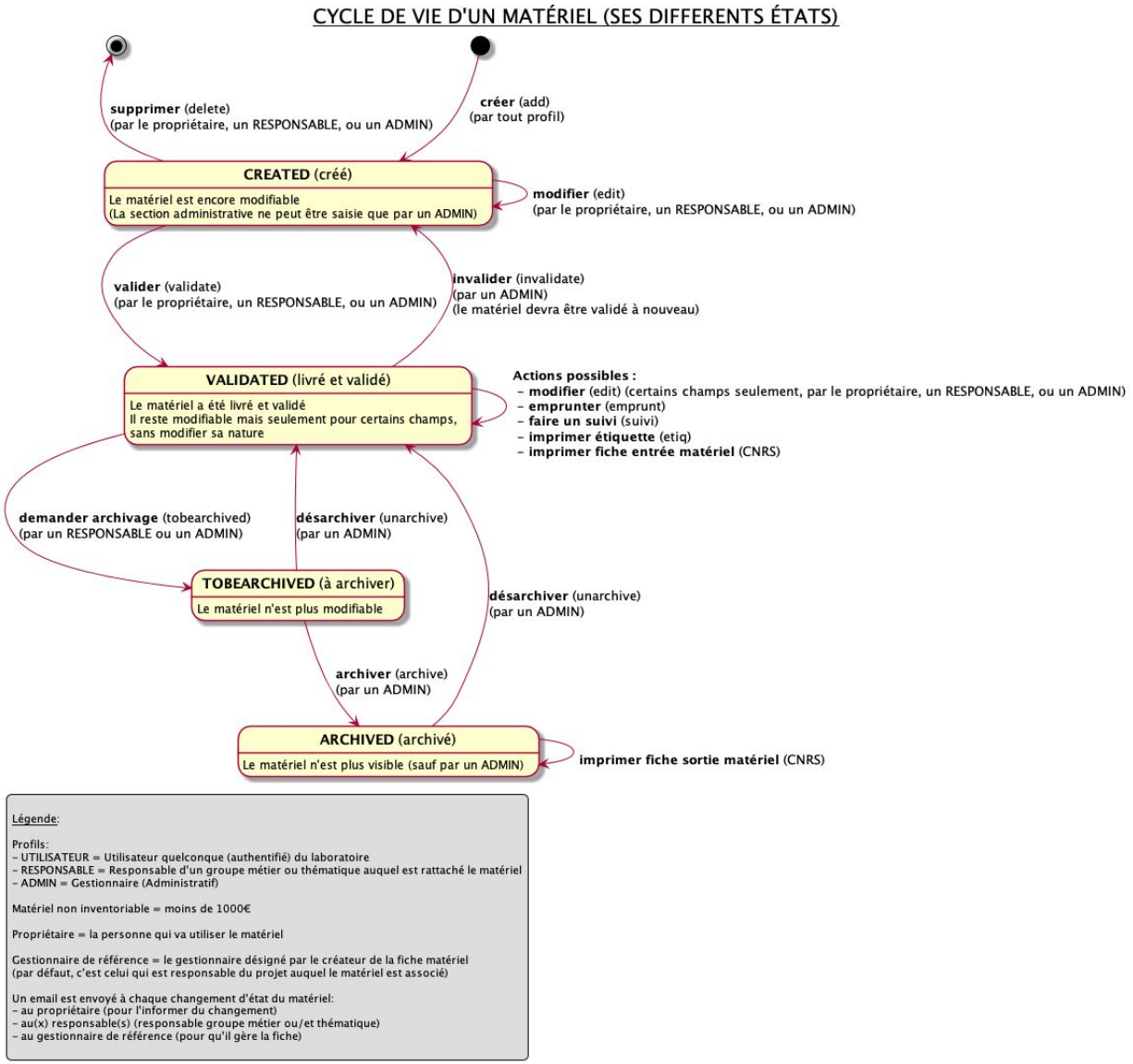
   ...ATTENTE DE LA LIVRAISON...

6. *A la **livraison** du matériel, le gestionnaire valide la fiche matériel (éventuellement, il peut y associer le bon de livraison en document attaché) => (je reçois un mail qui m'informe de l'arrivée du matériel et me demande de **vérifier** ma fiche)*
7. *Le gestionnaire imprime l'étiquette d'inventaire associée au matériel ainsi que la fiche complète du matériel et la joint au carton du matériel*
8. *Je viens chercher mon nouveau matériel et y colle l'étiquette d'inventaire*

11.3. Cycle de vie d'un matériel

(EP updated 15/10/19)

<u>Diagramme états-transitions</u> :



(TODO: mettre à jour cette description détaillée)

Le statut d'un materiel change selon le workflow suivant :
1) Un utilisateur lambda le cree (n'importe qui du labo) --> CREATED
2) L'Administration le valide (apres avoir eventuellement complete la fiche) --> VALIDATED
3) Un utilisateur lambda demande a l'archiver --> TOBEARCHIVED
4) L'Administration le sort de l'inventaire --> ARCHIVED

Notes :
- Dans l'ideal, le materiel est premierement cree par l'utilisateur concerne,
puis mis a jour par l'administration au moment de la commande (puis validé)
- L'administration peut toujours retrograder le statut d'un materiel (ce qui revient a annuler un changement de statut)

Créer un matériel ==> passe alors en statut **CREATED** ==> peut alors être éventuellement supprimé

Valider un matériel CREATED ==> passe alors en statut **VALIDATED** => ne peut plus être supprimé

Demander l'Archivage d'un materiel VALIDATED ==> passe alors en statut **TOBEARCHIVED**

Archiver (sortir de l'inventaire, en validant une demande d'archivage d'un matériel TOBEARCHIVED) ==> statut **ARCHIVED** (n'est alors plus visible, sauf pour admin)

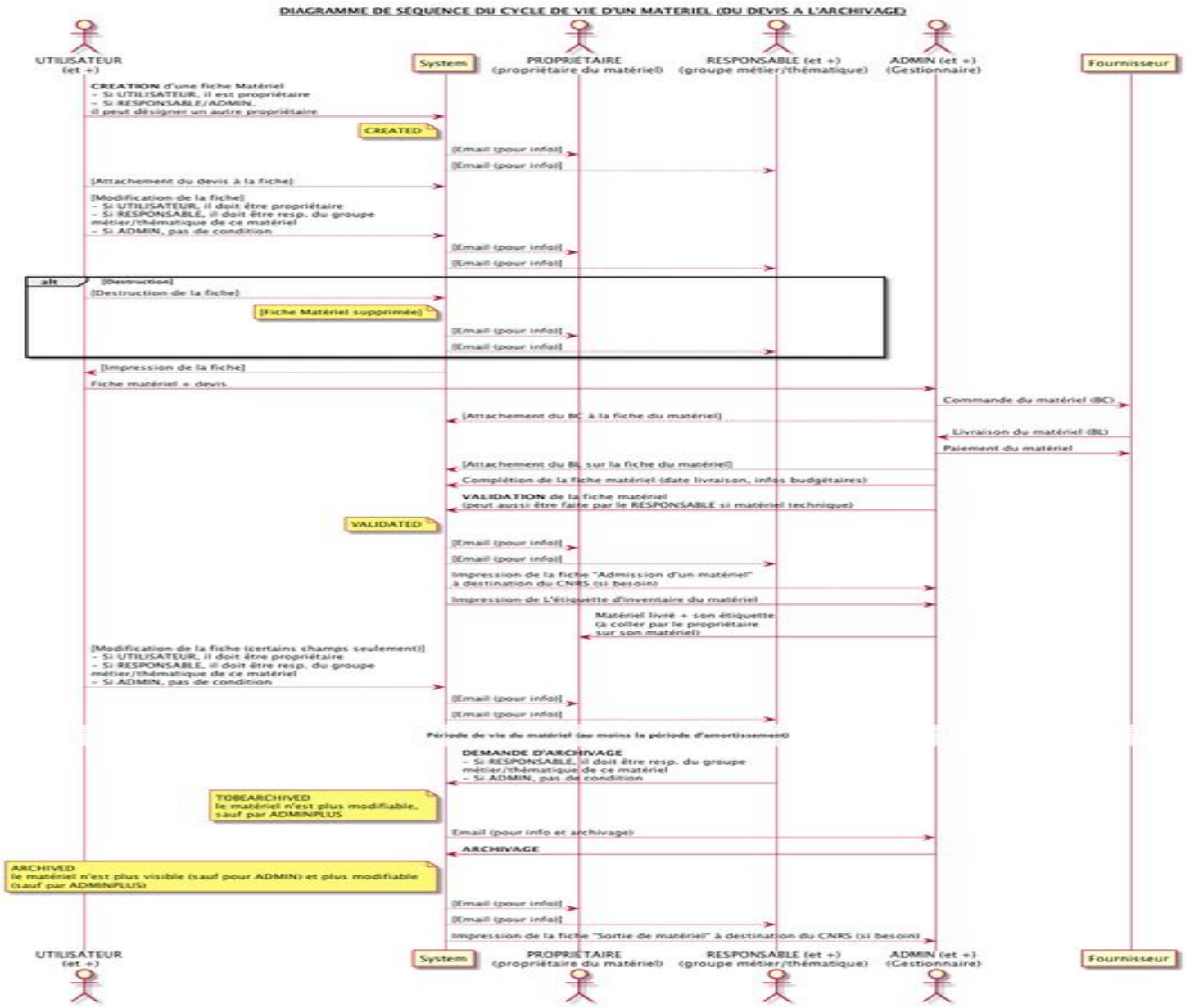
Désarchiver un matériel ==> repasse de TOBEARCHIVED ou ARCHIVED à VALIDATED

En résumé :

CREATED (fiche matériel créée) ==> VALIDATED (= matériel livré [et vérifié]) ==> TOBEARCHIVED ==> ARCHIVED
||
||
V
DELETED

Attention : **VALIDATED = matériel livré** [+ **fiche vérifiée** (dans l'idéal)]

11.4. Diagramme de séquences



12. Descriptions techniques

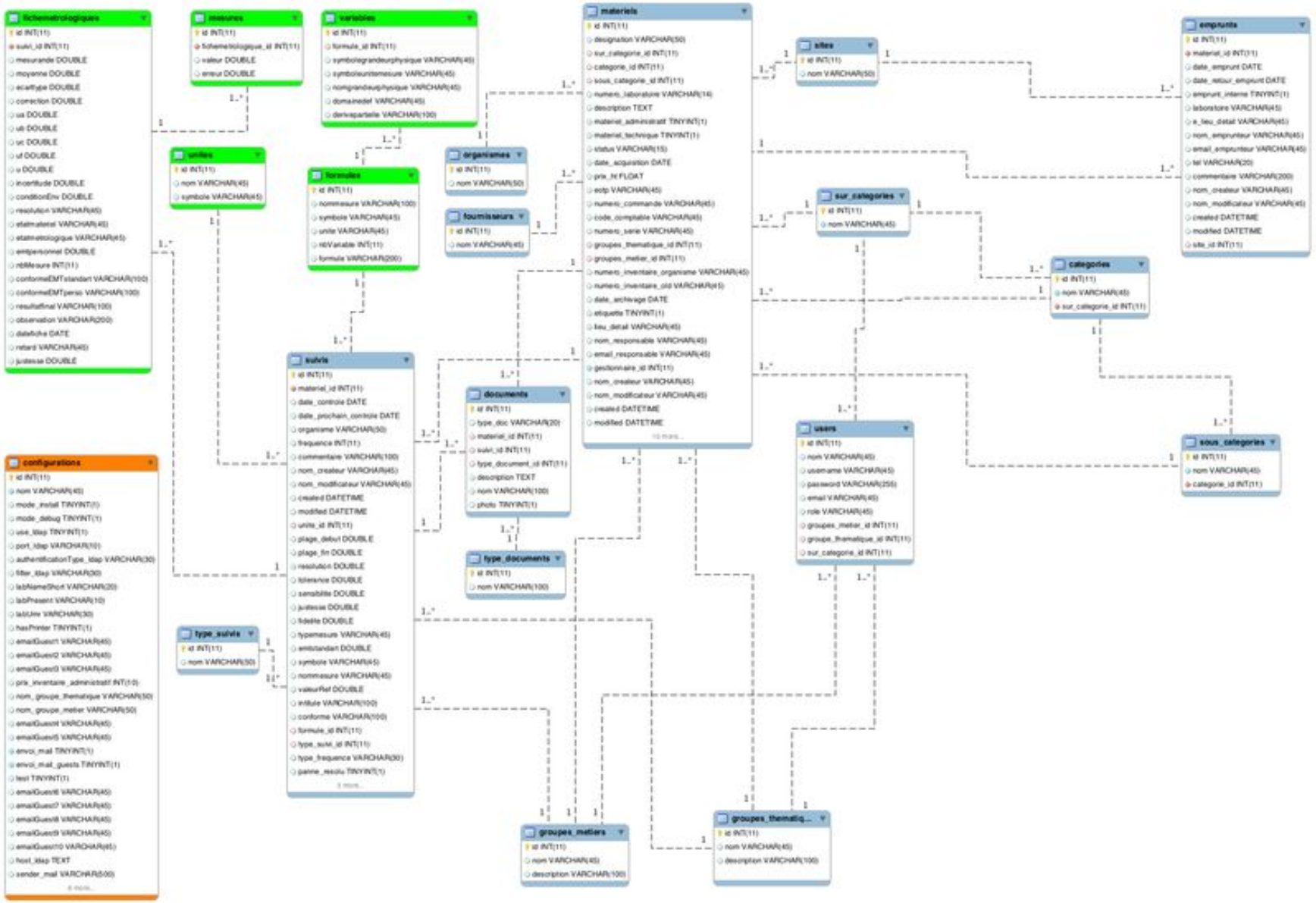
Cette section décrit différents aspects internes du fonctionnement du logiciel.

Voici d'autres éléments sur le wiki (à mettre à jour et intégrer dans cette présente doc) :

- Doc technique
- Doc de développement

12.1. Modèle de données (Data model)

Schéma de la base de données (v2.6) :



Légende:

- en orange : la partie "configuration" de labinvent ; c'est cette table qui contient tous les paramètres configurables du logiciel
- en vert : la partie "metrologie" qui est une extension de labinvent ajoutée par le laboratoire LATMOS ; cette extension est désactivée par défaut dans la configuration
- en bleu : tout le reste, c'est à dire la partie principale : les matériels, les suivis et les emprunts

12.2. Page d'accueil (démarrage, home page)

(updated 29/1/19 - EP)

LabInvent démarre sur la page **src/Template/Pages/home.ctp**

Cette page est la page d'accueil par défaut produite par le framework CakePhp (et est affichée tant que le mode debug est actif).

Elle a été modifiée pour LabInvent avec le contenu suivant :

```
<?php
if ($configuration->mode_install) {
	include ("home_install.ctp");
} else {
	include ("home_app.ctp");
}
?>
```

Comme on peut le voir, si on est en mode installation (lors de la phase d'installation du projet), on va sur la page **home_install**.ctp qui aide à vérifier si l'installation s'est bien passée ou bien s'il subsiste des problèmes à régler.

Sinon (situation par défaut, une fois l'installation terminée), on va sur la page **home_app**.ctp qui est la page d'accueil par défaut.

12.3. Ce qui se passe avant l’affichage d’une page web (workflow)

(updated 23/5/19 - EP)

Cette section est un peu technique mais très utile pour comprendre dans quel ordre les méthodes des contrôleurs sont appelées, depuis l’appel d’une action, jusqu’à arriver finalement à l’affichage d’une vue (la page web qu’on voit au final sur le site).

Le mieux pour comprendre est de passer en mode DEBUG (Outils/Configuration générale de l’application/Editer/Cocher “Mode DEBUG”).

Une fois ce mode DEBUG activé, vous verrez s’afficher les différentes étapes.

1 - Si vous n’êtes pas encore connecté, vous verrez s’afficher, juste au-dessus de la page d’accueil (login), dans l’ordre :

'step AVANT 0: AppController.**initialize**()'

'step 0: AppController.**beforeFilter**()'

'step 2: **UsersController.login**()'

'step 3: AppController.**beforeRender**()'

2 - Une fois connecté (logué), c’est comme si vous aviez appelé l’action “**display**” du controleur “**PagesController”** (controleur responsable de l’affichage des pages webs classiques), et donc vous verrez s’afficher, juste au-dessus de la page d’accueil (après authentification), dans l’ordre :

'step AVANT 0: AppController.**initialize**()'

'step 0: AppController.**beforeFilter**()'

'step 2: **PagesController**.**display**()'

'step 3: AppController.**beforeRender**()'

3 - Si maintenant vous affichez la liste des matériels (action **“index”** du controleur **“MaterielsController**”), vous verrez s’afficher, juste au-dessus de cette liste, la séquence complète, dans l’ordre :

'step AVANT 0: AppController.**initialize**()'

'step 0: MaterielsController.**beforeFilter**()' ⇒ **ce qui va appeller** ⇒ 'step 0: AppController.beforeFilter()'

'step 1: MaterielsController.**isAuthorized**()'

'step 2: **MaterielsController**.**index**()'

'step 3: MaterielsController.**beforeRender**()' ⇒ **ce qui va appeller** ⇒ 'step 3: AppController.beforeRender()'

Quelques explications s’imposent. Les méthodes suivantes sont appelées dans cet ordre :

- src/Controller/**AppController.initialize**() :
    - Cette **méthode est appelée en tout premier lieu**
    - on va notamment y lire (une fois pour toutes) la **configuration** et la mettre dans une variable $**this->confLabinvent**, utilisée partout dans le code ensuite

- src/Controller/**AppController.beforeFilter**() :
    - cette méthode est appelée (après initialize()) par défaut quand on ne sait pas quel controleur est responsable de l’action, ou bien APRES l’appel de la méthode éponyme d’un controleur spécifique (par exemple, MaterielsController.beforeFilter() est appelée, ce qui va appeler ensuite AppController.beforeFilter())
    - on y initialise (une fois pour toutes) des variables utilisées partout dans le code, telles que le profil de l’utilisateur connecté (ce qui va déterminer ses droits ensuite) :
        - $this->USER_IS_UTILISATEUR
        - $this->USER_IS_RESPONSABLE
        - $this->USER_IS_ADMIN
        - $this->USER_IS_ADMINPLUS
        - $this->USER_IS_SUPERADMIN
        - $this->USER_IS_RESPONSABLE_OR_MORE

        - $this->USER_IS_ADMIN_OR_MORE
        - $this->USER_IS_ADMINPLUS_OR_MORE
    - on passe aussi ces variables à la VUE grâce à la méthode $this->set() ; toutes les VUES vont donc hériter automatiquement de ces variables et pourront les utiliser directement ainsi :
        - if ($USER_IS_UTILISATEUR) …
        - if ($USER_IS_ADMIN) ...

- src/Controller/MaterielsController.**isAuthorized**() :
    - Si une méthode isAuthorized() existe dans le controleur spécifique responsable de l'action demandée (par exemple MaterielsController si l'action est "afficher la liste des materiels", c'est à dire, materiels/index), elle est appelée pour savoir si l'action demandée est autorisée ou pas pour l'utilisateur courant. Cette méthode retourne VRAI si l'action est autorisée ou FAUX sinon. Si FAUX, alors on ne va pas plus loin et un message du genre "cette action n'est pas autorisée" va s'afficher en haut de page.

- src/Controller/MaterielsController.**index**() :
    - C'est maintenant la méthode portant le nom de l'action demandée qui est appelée sur le controleur compétent. Ici, par exemple, on appelle la méthode index() du controleur MaterielsController si on a demandé la liste des matériels (action "materiels/index").
    - Dans cette méthode (correspondant à une action), on va initialiser des variables qui seront passées à la vue correspondant à cette action (ici la vue src/Template/Materiels/index.ctp), grâce à la méthode $this->set(). Ces variables pourront être directement utilisées dans le fichier de la vue (voir ci-après). Par exemple, dans l'action "index()" du controleur MaterielsController, on va initialiser une variable $**materiels** qui contient la liste des matériels à afficher.

- src/Controller/MaterielsController.**beforeRender() :**
    - Juste avant l'affichage de la vue (ici la vue index, voir ci-dessous), il y a encore la méthode beforeRender() qui est appelée, et qu'on peut utiliser pour définir quelques variables générales pour la vue, mais il est préférable de le faire dans **beforeFilter**() (voir ci-dessus).
    - Actuellement, on utilise cette méthode pour initialiser des FONCTIONS utilitaires dont TOUTES les vues vont héritées (encore grâce à la méthode $this->set()) ; On définit ainsi les fonctions $**displayElement**(), $**dateProchainControleVerif**(), et $**echoActionButton**(), très utilisées par les vues pour afficher des boutons, ou autres éléments...

- src/Template/Materiels/**index.ctp** :
    - On est maintenant à la fin du workflow, sur la dernière page exécutée, la vue. Ici, il s'agit de la vue "index.ctp" (du template Materiels) qui affiche la liste des materiels. Comme cette vue a hérité de la variable $materiels (voir ci-dessus), elle n'a plus qu'à boucler sur cette liste pour afficher les materiels un par un. Elle hérite aussi des variables générales initialisées dans **AppController.beforeFilter()** telles que $USER_IS_UTILISATEUR… (voir ci-dessus) pour décider de ce qu'elle peut afficher ou pas.
    - De manière générale, **la vue doit être la plus BETE possible**, et se contenter d'afficher des éléments un par un, sans avoir trop à réfléchir car presque toute la réflexion doit avoir été faite en amont, dans le controleur. **Ca doit donc être du style** "Si l'utilisateur

courant est un ADMIN, alors je peux afficher les informations administratives", **mais pas du style** "Si l'utilisateur courant est un ADMIN, et si le matériel est VALIDATED ou alors Si l'utilisateur est un RESPONSABLE et qu'il est lié à la thématique du matériel en cours… etc., alors je peux afficher les informations suivantes..."

12.4. Gestion des utilisateurs (connexion et profils associés)

(updated 14/10/19 - EP)

\user

12.4.1. Authentification des utilisateurs (avec ou sans LDAP)

(updated 12/02/19 - EP)

\auth

Nous décrivons ici le processus de CONNEXION d'un utilisateur avec un login et mot de passe. Il est possible avec ou sans LDAP.

Ce processus se nomme en anglais "**authentication**". Pour plus d'information à ce sujet, lire la documentation de CakePhp (https://book.cakephp.org/3.0/en/controllers/components/authentication.html)

1) BOOTSTRAP

(Cette étape est décrite pour être exhaustif, mais vous pouvez aller directement à l'étape suivante, "2- LOGIN")

On décrit ici le chemin parcouru (dans l'ordre) depuis le tout début, avant d'arriver à la page de login.

Enchaînement des fichiers, depuis la racine du projet LABINVENT/ :

index.php

webroot/**index.php** :
require config/requirements.php

require vendor/autoload.php
new Application('config')

src/**Application.php**
class Application extends BaseApplication
function bootstrap() {
parent::bootstrap()
addPlugin(Migrations)
addPlugin(DebugKit)

vendor/cakephp/.../BaseApplication.php
function bootstrap() {
require_once config/bootstrap.php

config/**bootstrap.php**
require config/**paths.php**
require vendor/autoload.php
require vendor/cakephp/.../config/bootstrap.php
load config/**app.php** *(ce fichier contient ma config perso)*
set config (cache, db, email, log, …)

src/Controller/PagesController/display()

src/Controller/UsersController/login() sans POST

src/Template/Users/login.ctp => formulaire de login => POST

2) LOGIN (avec ou sans LDAP)

Une fois le formulaire de LOGIN "posté" (validé par l'utilisateur), on arrive à l'étape de login proprement dite.

Pour info, si ça bogue *(par exemple, à l'époque d'un fameux stagiaire..., on retournait "YOLO" au lieu de FALSE en cas d'échec de connection ldap, et ça plantait lamentablement, je m'en souviens très très bien...),*
*⇒ c'est ****src/Template/Error/error400.ctp***** qui prend le relais...*

Voici la description du processus de login (authentification) :

*a) src/Controller/UsersController.php, fonction **login**() :*

```
$user = $this->LdapAuth->connection();

// Utilisateur identifié
if ($user != FALSE) {
        $this->LdapAuth->setUser($user);
        // On va maintenant à la page qui était demandée
        return $this->redirect($this->LdapAuth->redirectUrl());
}

// Utilisateur non reconnu
$this->Flash->error(__('Login ou mot de passe invalide, réessayez'));
```

b) *src/Controller/Component/LdapAuthComponent.php, fonction **connection**() :*

```
// Get login and password entered by the current user (who is trying to connect)
$login = $this->request->getData('ldap');
$password = $this->request->getData('password');
$return TableRegistry::get('LdapConnections')->ldapAuthentication($login, $password);
```

c) *src/Model/Table/LdapConnectionsTable.php, fonction **ldapAuthentication**(**$user_login, $user_password**) :*

```
if (strlen(trim($user_password)) == 0) return FALSE;        // No connexion allowed without password
$filter = "(&".$this->filter."(".$this->authenticationType . '=' . $user_login."))";

// Connection
$ldapConnection = ldap_connect($this->host, $this->port) or die("Could not connect to $this->host (port $this->port)");
if ($ldapConnection) {
        ldap_set_option($ldapConnection, LDAP_OPT_PROTOCOL_VERSION, 3);
        // Binding optionnel
        if ($this->ldap_authentified) $ldapbind = ldap_bind($ldapConnection, $this->bindDn, $this->bindPass)
                or die("Could not bind to LDAP server.". ldap_error($ldapConnection) );
        $search = $this->getUserAttributes($user_login, $ldapConnection, $filter, $just_these);
        if ($search === false) die("Could not get user attributes from LDAP server, response was: " . ldap_error($ldapConnection) );
        return $search[0];
```

```
}
```

d) *src/Model/Table/LdapConnectionsTable.php, fonction* ***getUserAttributes****($userName, $ldapConnection='', $filter='', $just_these=[])) :*

```
$results = ldap_search($ldapConnection, $this->baseDn, $filter, $just_these)
        or die("Could not search to LDAP server  response was: " . ldap_error($ldapConnection) );
$info = ldap_get_entries($ldapConnection, $results);
return $info;
```

e) On retourne maintenant à l'étape a) ci-dessus, à la suite de la ligne "$user = $this->LdapAuth->connection()"

12.4.2. Gestion des profils utilisateurs (Autorisations, ACL)

Cette partie fait l'objet d'un document séparé

12.5. FAKE LDAP

(updated 27/05/19 - EP)

Comment simuler un LDAP quand on est sur une machine de développement, et qu'on n'a pas de LDAP sous la main ?

1 - Créer une table fakeldap qui représentera le LDAP et contiendra quelques utilisateurs.

```
CREATE TABLE `fakeldap` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `sn` varchar(45) DEFAULT NULL,
  `givenname` varchar(45) DEFAULT NULL,
  `uid` varchar(45) DEFAULT NULL,
```

```
  `mail` varchar(45) DEFAULT NULL,
  `userpassword` varchar(255) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ALTER TABLE `fakeldapusers`
-- MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=73;
```

(...TO BE CONTINUED...)

12.6. LOG

(updated 19/12/18 - EP)

Des messages de debug et d'erreur peuvent être loggés grâce au système de logging de CakePhp qui est configuré dans le fichier de configuration générale **config/app.php** :

```
/**
 * Configures logging options
 */
'Log' => [
        'debug' => [
                'className' => 'Cake\Log\Engine\FileLog',
                'path' => LOGS,
                'file' => 'debug',
                'levels' => ['notice', 'info', 'debug'],
                'url' => env('LOG_DEBUG_URL', null),
        ],
        'error' => [
                'className' => 'Cake\Log\Engine\FileLog',
                'path' => LOGS,
                'file' => 'error',
                'levels' => ['warning', 'error', 'critical', 'alert', 'emergency'],
                'url' => env('LOG_ERROR_URL', null),
```

```
        ],
],
```

Remarque :

error_log(“bla bla bla”);
==> écrit dans **logs**/error.log

Dans une ancienne version de LabInvent, on pouvait faire ceci (il faudrait remettre ça en place) :

$this->log('Something broke');
==> écrit dans cakephp/app/tmp/logs/**labinvent.log**

12.7. DEBUG

(updated 17/01/19 - EP)

BUT : CORRIGER UNE ERREUR, RESOUDRE UN PROBLEME, COMMENT DEBOGUER (DEBUG)...

Il y a 2 niveaux de debug dans LabInvent.

12.7.1. Debug général

COMMENT VOIR LES REQUETES SQL FAITES PAR CAKEPHP ?

Vous devez d’abord vous assurer que vous êtes bien en mode **DEBUG=true** dans le fichier de configuration config/app.php :

'debug' => filter_var(env('DEBUG', **true**), FILTER_VALIDATE_BOOLEAN),

⇒ vous verrez alors une icone CakePhp (en forme de gateau blanc sur fond rouge) en bas à droite de chaque page web, il suffit alors de cliquer dessus pour avoir accès à plein d’informations utiles et notamment “Sql Log” qui donne le contenu des dernières requêtes SQL effectuées, très très utile !

Dans le code, on peut aussi lire la variable de configuration “debug” ainsi :

if (Configure::read('debug')) ...

Autre possibilité supplémentaire:
Dans le layout general (src/Template/Layout/default.php), ajouter cette ligne :
    <?php echo $this->element('SQL_DUMP'); ?>

Comment pister les erreurs ?

Cakephp vous affiche une erreur, mais vous ne savez pas d'ou vient le probleme.
Don't panic.
En general, un bon moyen de le savoir est de lire le fichier de log des erreurs
dans **logs/error.log**

Dans le dossier **logs/**, vous trouverez d'autres logs utiles :
- **labinvent.log**
- **debug.log** (pas sur que ce fichier soit encore utilisé...)

Remarque :
error_log(“bla bla bla”);
==> écrit dans **logs**/error.log

12.7.2. Debug local (applicatif)

C’est un mode “debug” pour l’application LabInvent qui se met alors à afficher des informations techniques en haut de page (header).
Pour y passer, il suffit d’aller dans le menu **Outils** puis “**Passer en mode DEBUG**”.
Pour revenir au mode normal, menu **Outils** puis “**Stopper le mode DEBUG**”.

13. HOWTO

(EP updated 4/2/19)

Ce document est un HOWTO, c'est à dire un guide technique pour savoir comment faire quoi.

Il donne des solutions à différents types de besoins ou problèmes, et explique aussi où trouver quoi.

13.1. Version du framework CakePhp utilisé dans le projet

(EP updated 4/2/19)

L'information est dans le fichier **vendor/cakephp/cakephp/VERSION.txt**
Ou bien, exécuter cette ligne php :

```
Configure::version();          // 3.5.11
```

13.2. Contrôleur des pages web "simples"

(EP updated 4/2/19)

Controleur des pages simples = Controller/PagesController.php)

Page d'accueil = src/Template/Pages/home.ctp

13.3. Tout sur les QrCodes

Le **QrCode** représente l'URL de la vue détaillée d'un matériel
Il est généré à la volée à chaque fois qu'on visualise un matériel en allant sur sa vue détaillée (materiels/view/<materiel_id>).
C'est donc le fichier "vue détaillée" du matériel qui appelle le code de génération du QrCode, soit le fichier src/Template/Materiels/view.ctp
Ce fichier contient le code source suivant qui crée une section html "image" contenant le nom du fichier image PNG du QrCode, généré à la volée par l'action creer() du controleur des QrCodes (src/Controller/QrCodesController.php) :

```
$this->request->getSession()->write("qrUrl", $this->request->env('SERVER_NAME') . $this->request->env('REQUEST_URI'));
$this->requestAction('/QrCodes/creer/');
echo $this->Html->image('qrcodes/' . $this->request->getSession()
->read("filename"), [
'alt' => 'QrCode : ' . $materiel->numero_laboratoire,
'style' => 'float: right'
]);
```

Voici le code source de la fonction **creer**() du controleur des QrCodes ;

```
use \PHPQRCode\QRcode;
...

// Le fichier QrCode porte simplement le nom de l'ID de la session suivi de l'extension ".png".
// Par exemple, "0fga4e9e6osa97iunfsvk8m8m8.png".
$fileName = $this->request->getSession()->id() . '.png';

// L'image QrCode est créée dans le dossier webroot/img/qrcodes/
$cakephpPath = str_replace('webroot/index.php', '', $_SERVER['SCRIPT_FILENAME']);
```

```
$qrCodePath = $cakephpPath . 'webroot/img/qrcodes/' . $fileName;

$this->request->getSession()->write('filename', $fileName);
$this->request->getSession()->write('qrCodePath', $qrCodePath);

// Creation du QrCode avec la methode png() de PHPQRCode\QRcode
if ($message == null) {
        return QRcode::png($this->request->getSession()->read('qrUrl'), $qrCodePath);
} else {
        return QRcode::png($message, $qrCodePath);
}
```

13.4. Génération de la liste des utilisateurs (depuis le LDAP)

(EP updated 4/2/19)

Où se trouve le code qui affiche la liste des utilisateurs dans les vues ajout/édition de matériel ?
Il suffit de suivre cette logique :

- Ajout de materiels = src/Template/Materiels/add.ctp
- Edition de materiels = src/Template/Materiels/edit.ctp

Dans la vue "add.ctp" (et “edit.ctp”) tu trouves le champ "Nom de l\'utilisateur"
Il contient une variable 'options' => $utilisateurs
C'est donc cette variable $utilisateurs qui est utilisée.

Là ca se complique un peu :
Cette variable n'est pas définie dans la vue, mais dans le controlleur de matériels qui l'a créée et passée à la vue :
On cherche donc "$utilisateurs" dans src/Controller/MaterielsController.php, on le trouve à la ligne 811 (elle est ensuite passée à la vue à la ligne 830 avec set(..., 'utilisateurs', ...))
Elle vient de "$users" qui est définit juste au-dessus, ligne 807 :

$users = TableRegistry::get('LdapConnections')->getListUsers();
getListUsers() est définie dans src/Model/Table/LdapConnectionsTable.php

Attention, une fois qu'on choisit un utilisateur dans la liste, son email est cherché via le ldap pour l'afficher dans le champ suivant nommé "Email de l'utilisateur"
En plus, c'est un code javascript (ajax) qui fait ça à la fin du fichier add.ctp :

```
$("#nom-responsable").bind("change", function(event) {
        var url = document.URL;
        var reg = new RegExp("(materiels).*$", "g");
        var emailUrl = url.replace(reg, "Users/getLdapEmail/");
        $.ajax({
                url: emailUrl + $("#nom-responsable").val()
        }).done(function(data) {
                $("#email-responsable").val(data)
        });
});
```

Ce code appelle la fonction **getLdapEmail**() de **src/Controller/UsersController**.php

13.5. Structure générale d'une page web (TEMPLATE) = default.ctp

src/Template/Layouts/**default.ctp** contient la structure suivante :

```
<div id="container">
    <div id="header">
        LE HEADER AVEC SON LOGO
        <div class="user">
```

```
                BIENVENUE $userName (ou invité)
            </div>
        </div>
        <div id="content">
            <?php echo $this->Session->flash(); ?>
            <?php echo $this->fetch('content'); ?>
        </div>
        <div id="footer">
            LE FOOTER
        </div>
    </div>
```

La DIV "content" insère 2 contenus :
- le message flash éventuel (qui dit si une opération demandée s'est bien passée ou pas...)
- la section "content", qui n'est autre que le coeur de la page

La page d'ACCUEIL n'est qu'un "content" parmi d'autres.
Elle se trouve dans src/Template/Pages/home.ctp
(elle est affichée par le controleur de pages nommé PagesController)

13.6. Ajouter une nouvelle action sur un contrôleur

Ex: ajout de l'action statusCreated() dans le controleur Materiel

Il faut l'ajouter à 2 endroits dans MaterielsController.php :

a) ajouter une methode statusCreated() :

```
    public function statusCreated($id = null, $from = 'index') { ... }
```

b) ajouter un cas 'statusCreated' dans la methode isAuthorized() qui doit retourner true pour autoriser cette action :

case 'statusCreated': ... return true; ...

La voici en détail :

```
    case 'statusCreated': // de-validation d'un materiel (repasse à CREATED)
                    $id = (int) $this->request->getParam('pass.0');
                    // Admin + ok
                    if ($this->USER_IS_ADMIN_AT_LEAST()) return true;
                    // Resp ok if same group
                    if ($this->USER_IS_RESP() && $this->isRespGroup($id, $user[$ldapAuthType])) return true;
                    // User not ok
                    break;
```

13.7. Champ facultatif/obligatoire : Comment rendre facultatif ou obligatoire un champ, et définir les vérifications à faire sur ce champ ?

=> src/Model/Table/<Model>Table.php
ex : src/Model/Table/MaterielsTable.php pour définir les règles sur les champs de la table "materiels"

13.8. Où définir les éléments passés à une vue (c'est à dire à une action) ?

=> src/Controller/<Model>Controller/nom_de_la_vue_ou_action()

ex : src/Controller/MaterielsController/edit() pour définir (avec set()) les éléments passés à la vue "edit" des materiels

13.9. Adapter LabInvent au laboratoire utilisateur

(updated 19/12/18 - EP) ⇒ **A compléter**

Vues (Pages) concernées :
- src/Template/Layout/default.ctp (template)
- src/Template/Pages/home.ctp (Accueil)
- src/Template/Pages/about.ctp (A propos)
- src/Template/Pages/printers (Etiqueteuses)

Controleurs concernées :
- src/Controller/MaterielsController.php (fonction getLabNumber())

Documents concernés :
- Etiquette (**fait**)
- src/Template/Documents/admission.ctp (Document d'admission UPS et CNRS)

13.10. Ajouter une nouvelle page web

(updated 19/12/18 - EP)

Par exemple, voici ce que j'ai fait pour ajouter la page "about", qui est accessible via l'url https://inventirap.irap.omp.eu/pages/about
1) Aller dans src/Template/Pages/
1) Faire un copier/coller d'une page existante, par exemple infos.ctp, et l'appeler about.ctp
2) Remplir cette page avec le contenu souhaité
3) Tester que cette page est bien accessible via l'url http://.../pages/about

4) Ajouter un lien vers cette page, soit dans le menu outils (src/Template/Pages/tools.ctp), soit dans le menu général (src/Template/Element/menu.ctp)
5) Si cette page ne doit pas être accessible à tout le monde, définir le profil minimum exigé dans src/Controller/PagesController.php, dans la fonction display() :

```
if ($page == about) {
        // Autoriser seulement à partir du role ADMIN
        if (! $this->USER_IS_ADMIN_AT_LEAST()) return $this->redirect('/');
}
```

(ou bien ajouter une ligne dans le tableau $minProfileAllowedForPage)

13.11. Recherche de matériels

(updated 19/12/18 - EP)

3 methodes :
- Recherche générale (champ “Recherche” sous le menu général à gauche) : src/Template/Element/menu.ctp
    ⇒ appelle l’action “materiels/find”
- VUE de recherche : src/Template/Materiels/find.ctp
- ACTION de recherche : src/Controller/MaterielsController (methode find())

13.12. Autres HOWTO plus anciens

Le contenu de ce chapître est à utiliser avec précaution car, étant assez ancien, il risque de ne pas être complètement utilisable (bien qu’il puisse encore l’être dans la plupart des cas). Cependant il reste utile pour information.

INSTALLATION
****
La procedure d'installation est decrite dans le fichier INSTALLATION.txt qui se trouve dans le dossier install/
Pour une installation à partir d'Eclipse, voir le document install/manual_install/INSTALLATION_MANUELLE_mode_expert.txt

OU EST QUOI (WHERE IS WHAT) ?
****

- OU mettre a jour la VERSION du soft (AVANT CHAQUE COMMIT) ?
En attendant mieux, on fait ça dans le fichier README.md (ça sera automatiquement affiché par src/Template/Layout/default.ctp)

- OU EST LA PAGE GENERALE (qui contient tous les elements) ? : cakephp/app/View/Layouts/default.ctp

- OU EST LA PAGE D'ACCUEIL ? : app/View/Pages/home.ctp

- OU SONT LES MENUS ? : app/View/Elements/
  - menu general + Recherche generale : menu.ctp
  - sous le menu general, et sous le champ "Recherche", titre du modele a ajouter : menu_index.ctp

- OU EST LA FEUILLE DE STYLE CSS ? : cakephp/app/webroot/css/inventirap.css

- OU EST DEFINIE LA PAGINATION ?
Dans le Controleur
Ex : la pagination des materiels est definie dans app/Controller/MaterielsController.php

```
    public $paginate = array(
            'limit' => 50,
            'order' => array('Materiel.id' => 'desc'));
```

- OU SONT LES INFOS CONCERNANT UNE ENTITE quelconque (Materiel, Emprunt, Suivi, Utilisateur) ?
par exemple, ou trouver les infos sur l'entite "Materiel" ? : Aller dans cakephp/app/
- le Modele : Model/Materiel.php
- le Controleur : Controller/MaterielsController.php
- les Vues (templates) : View/Materiels
  - Vue de consultation : scaffold.view.ctp
  - Vue d'ajout (add) et edition (edit) : scaffold.form.ctp
  - vue de liste : index.ctp

- OU SONT DEFINIS LES ROLES (ACL) ?

Dans app/Model/Utilisateur.php :

```
    private $acceptedRoles = array ('Utilisateur', 'Responsable', 'Administration', 'Super Administrateur');
    public function getAuthenticationLevelFromRole($role) {
            if ($role == 'Utilisateur')
                    return 1;
            elseif ($role == 'Responsable')
                    return 2;
            elseif ($role == 'Administration')
                    return 3;
            elseif ($role == 'Super Administrateur')
                    return 4;
            return 0;
    }
```

ANCIEN FICHIER DE CONFIG labinvent.php

****

(updated 19/12/18 - EP)

(cf http://book.cakephp.org/2.0/fr/development/configuration.html#loading-configuration-files)
Ce fichier n'est plus utilisé car maintenant la configuration est mise dans une table "configurations".
Je laisse quand même cette section pour information sur la manière de gérer la configuration via un fichier php.

1) J'ai créé le nouveau fichier de config dans app/Config/ (par exemple labinvent.php)
Ce fichier doit au minimum contenir un tableau nommé $config :

```
    $config = array(
                    'labName' => 'IRAP',
                    ...
    );
```

2) Charger ce nouveau fichier de config au démarrage
Pour cela, ajouter cette ligne dans app/Config/bootstrap.php :

```
    Configure::load('labinvent');
```

3) Lire (ou même modifier) les paramètres de ce fichier de config, depuis N'IMPORTE OU (controleur, modèle, vue) :

```
debug(Configure::version()); // pour afficher la version de Cakephp
debug(Configure::read()); // tout lire
debug(Configure::read('localisation')); // le tableau $localisation
$labName = strtoupper(Configure::read('localisation.labNameShort'));
debug(Configure::read('localisation.labName'));
if (Configure::read('USE_LDAP')) ...
if (Configure::read('debug') > 0 ) ...
```

On peut même modifier la valeur d'un paramètre dynamiquement comme ceci :

```
Configure::write('debug',2)
```

C'est d'ailleurs ce qui est fait dans app/Config/core.php

4) Penser à modifier le script d'installation install/installation.sh pour qu'il prenne en compte ce nouveau fichier

GOOGLE ANALYTICS INTEGRATION
****
adapté de http://blog.janjonas.net/2010-01-31/cakephp-google-analytics-integration

1) Copier ce code dans src/Template/Element/google-analytics.ctp :

```
<?php
$gaCode = Configure::read('google-analytics.tracker-code');
if ($gaCode) {
$googleAnalytics = <<<EOD
<script type="text/javascript">
  (function(i,s,o,g,r,a,m){i['GoogleAnalyticsObject']=r;i[r]=i[r]||function(){
  (i[r].q=i[r].q||[]).push(arguments)},i[r].l=1*new Date();a=s.createElement(o),
  m=s.getElementsByTagName(o)[0];a.async=1;a.src=g;m.parentNode.insertBefore(a,m)
  })(window,document,'script','//www.google-analytics.com/analytics.js','ga');

  ga('create', 'UA-45668893-3', 'omp.eu');
  ga('send', 'pageview');

</script>
```

```
EOD;
echo $googleAnalytics;
}
?>

/* ANCIEN CODE JAVASCRIPT :
<script type="text/javascript">
var gaJsHost = (("https:" == document.location.protocol) ? "https://ssl." : "http://www.");
document.write(unescape("%3Cscript src='" + gaJsHost + "google-analytics.com/ga.js' type='text/javascript'%3E%3C/script%3E"));
</script>
<script type="text/javascript">
try {
var pageTracker = _gat._getTracker("$gaCode");
pageTracker._trackPageview();
} catch(err) {}</script>
*/
```

2) Inclure le view element dans src/Template/Layout/default.ctp (juste avant </body>):

```
<?php echo $this->element('google-analytics'); ?>
```

3) Définir le tracker code dans la configuration (/app/Config/core.php):

```
Configure::write('google-analytics.tracker-code', false); // disables Google Analytics
Configure::write('google-analytics.tracker-code', 'YOUR-TRACKING-CODE'); // enables Google Analytics
```

GENERALITES
****

- URL : http://localhost/inventirapsvn/cakephp/

- BD admin : http://localhost/phpmyadmin

- LOG : cakephp/app/tmp/logs/inventirap.log

- (UPDATE : cd Inventirap/ ; chmod -R 777 ./cakephp/app/tmp/)

- Eclipse config :
Setup syntax highlighting for .ctp files:
Window > Preferences > General > Appearance > content Types > Text > PHP Content type > Add.. , then put in *.ctp.

- Cakephp config : cakephp/app/Config/
Mode debug ON (pour voir les erreurs et pour vider le cache en cas de changement sur la BD)
core.php : Configure::write('debug', 1);

- Fichier de demarrage :
cakephp/index.php (qui pointe sur cakephp/app/webroot/index.php)
(Attention, il y a aussi un fichier cakephp/app/index.php qui pointe sur webroot/index.php)
(ROOT doit pointer sur le dossier cakephp/)
Le fichier execute ensuite est cakephp/lib/Cake/bootstrap.php qui lance ./Core/App.php
and so on...

ACL (Controle d'acces aux ressources)
****
NB: Cette section n'est sans doute plus très à jour ; sur ce sujet, il est préférable de lire le document docs/userguide/ACL.doc (ou .pdf)

Synthese sur les droits selon le profil

Profils (roles), dans le sens du pouvoir croissant :
Qq = Utilisateur Quelconque (lambda)
Rp = Responsable
Ad = Administration
Sa = Superadmin

Actions :
C = Create
R = Read (voir, consulter)

U = Update (mettre a jour)
D = Delete (supprimer)
V = Valider un materiel CREATED --> passe alors en statut VALIDATED
A = Demander l'Archivage d'un materiel VALIDATED --> passe alors en statut TOBEARCHIVED
S = Sortir de l'inventaire (Valider une demande d'archivage d'un materiel TOBEARCHIVED) --> passe alors en statut ARCHIVED
E = Exporter

Par defaut, le superadmin a acces a TOUT

Materiels :
- Qq a les droits C, R (sauf champs admin), U (si createur et sauf champs admin), A, D (si CREATED et owner)
- Rp a les droits C, R (sauf champs admin), U (sauf champs admin), D (si CREATED), V, A, E
- Ad a les droits C, R, U (ssi NOT ARCHIVED), D (si CREATED), V, (A mais inutile car fait directement S sans passer par A), S, E

Suivis et Emprunts :
- Dans tous les cas, on ne doit pas pouvoir emprunter ou suivre un materiel non valide (CREATED)
- Qq a les droits C, R, U (si createur), D (si createur)
- Rp a les droits C, R, U, D
- Ad a les memes droits que Rp

VUES specifiques :
- Acces aux Outils : reserve a Rp et Ad (vue contenant des liens vers differentes ressources telles que utilisateurs, materiels, categories...)
- L'administration a une vue resumee sur la page d'accueil (liens directs vers actions a faire)
- L'administration a une vue enrichie de la liste des materiels :
  - filtres (y-compris sur ARCHIVED)
  - export en CSV (pour tableur Excel)
  - upgrade du statut (validation et sortie)

Autres regles (de gestion) importantes :
- un materiel non valide (CREATED) peut ���tre supprime uniquement par le createur de la fiche, le responsable (Roger), et l'administration
Idem pour la mise ��� jour de cette fiche
- un materiel valide (VALIDATED) n'est pas supprimable
- les champs ADMINISTRATIFS d'un materiel ne sont visibles et modifiables que par l'administration
- par defaut, la liste des materiels affiche tous les materiels SAUF ceux qui sont sortis de l'inventaire (ARCHIVED) qui sont donc masques.

Il est de toutes facons toujours possible (pour l'administration seulement) de les voir, grace au nouveau filtre "Archives" present sur la liste des materiels
- la recherche doit s'effectuer dans TOUS les materiels (y-compris ARCHIVED)

COMMENT CONTOURNER LE LDAP officiel
****
(ATTENTION: CONTENU OBSOLETE A METTRE A JOUR (EP))

(cf Controller/UtilisateursController.php : methode login())

1) Si on veut tester le LDAP, on peut utiliser le LDAP de la Virtual Machine Upsilon (CentOS 6)
Dans ce cas, il faut ajouter dans la BD les utilisateurs specifiques suivants :
Ajout d'utilisateurs de TEST (LDAP upsilon)
INSERT INTO utilisateurs (nom, login, email, role, groupes_metier_id) VALUES
('Hillembrand Cedric', 'Cedric', 'Cedric.Hillembrand@irap.omp.eu', 'Super Administrateur', 1),
('Turner Daniel', 'Daniel', 'Daniel.Turner@irap.omp.eu', 'Administration', 1),
('Sky Gin', 'Gin', 'Gin.Sky@irap.omp.eu', 'Responsable', 1),
('Robert Henri', 'Henri', 'Henri.Robert', 'Utilisateur', 1);

2) Sinon, le plus simple est d'utiliser un FAUX (fake) LDAP interne a l'application
Dans ce cas, il faut decommenter la variable $fakeLDAP dans le fichier config.php et ajouter dans la BD les utilisateurs specifiques suivants :
Ajout d'utilisateurs de TEST (hors LDAP)
INSERT INTO utilisateurs (nom, login, email, role, groupes_metier_id) VALUES
('Hubert SuperAdmin', 'superadmin', 'hubert.superadmin@irap.omp.eu', 'Super Administrateur', 1),
('Pierre Responsable', 'responsable', 'pierre.resp@irap.omp.eu', 'Responsable', 2),
('Jean Administration', 'admin', 'Jean.Administration@irap.omp.eu', 'Administration', 5),
('Jacques Utilisateur', 'user', 'Jacques.Utilisateur@irap.omp.eu', 'Utilisateur', 7);

Informations sur le LDAP :

Classe LdapAuthComponent (extends AuthComponent), definie dans : app/Controller/Component/LdapAuthComponent.php

Definition des roles (ACL) : app/Model/Utilisateur.php :

```
    private $acceptedRoles = array ('Utilisateur', 'Responsable', 'Administration', 'Super Administrateur');
    public function getAuthenticationLevelFromRole($role) {
            if ($role == 'Utilisateur')
                    return 1;
            elseif ($role == 'Responsable')
                    return 2;
            elseif ($role == 'Administration')
                    return 3;
            elseif ($role == 'Super Administrateur')
                    return 4;
            return 0;
    }
```

Lecture du fichier de config : app/Model/LdapConnection.php :

```
            private function checkConfiguration()
```

Workflow du LDAP :
1) Page d'accueil : j'entre mon login et pwd
2) clic sur bouton "Se Connecter" --> execute l'action Utilisateurs/login (dans app/Controller/UtilisateursController.php)

COMMENT AJOUTER UNE NOUVELLE TABLE DANS LA BD
****

On explique ici ce qu'il faut faire quand on veut ajouter une nouvelle table dans la base de données,
table qui doit être prise en compte dans l'application.
Cette explication est donnée avec 2 exemples.

1) Premier exemple : ajout de la table sur-categorie (EP)

Avant cet ajout, il n'y avait que la table categorie et la table sous-categorie.

Vous trouverez plus de détails sur ce sujet dans la section suivante nommée "COMMENT J'AI FAIT POUR AJOUTER UN 3eme niveau de categorie appele sur_categorie (ou domaine)"

a) impact sur la BD
- ajout d'une nouvelle table sur_categories(id,nom) :

```
CREATE TABLE IF NOT EXISTS sur_categories (
  id int(11) NOT NULL AUTO_INCREMENT,
  nom varchar(45) DEFAULT NULL,
  PRIMARY KEY (id),
  UNIQUE KEY nom_UNIQUE (nom)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1;
```

- ajout dans la table categories d'une cle etrangere sur_categorie_id :

```
ALTER TABLE categories
    ADD sur_categorie_id INT( 11 ) NULL DEFAULT NULL,
      ADD CONSTRAINT fk_sur_categorie_id FOREIGN KEY (sur_categorie_id) REFERENCES sur_categories (id) ON DELETE NO ACTION ON UPDATE NO ACTION;
```

- ajout dans la table materiels d'une cle etrangere sur_categorie_id :

```
ALTER TABLE materiels
    ADD sur_categorie_id INT( 11 ) NOT NULL after designation,
    ADD CONSTRAINT fk_materiels_sur_categorie_id FOREIGN KEY (sur_categorie_id) REFERENCES sur_categories (id) ON DELETE NO ACTION ON UPDATE NO ACTION;
```

(attention, il faut d'abord vider les lignes des tables suivis, emprunts, et materiels)

```
delete from suivis;
delete from emprunts;
delete from materiels;
```

Il faut aussi ajouter des sur-categories :

```
# Ajout de quelques sur-categories
insert into sur_categories (id,nom) values
(1,'Electronique'),
(2,'Informatique'),
(3,'Instrumentation')
```

(4, 'Logistique'),
(5, 'Mecanique'),
(6, 'Optique')
;

Relier toutes les categories a une sur-categorie (la 1)
update categories set sur_categorie_id=1 where id<10;
update categories set sur_categorie_id=2 where id>=10 and id<20;
update categories set sur_categorie_id=3 where id>=20;

b) impact sur les modeles (cakephp/app/Model)
- Ajout du nouveau modèle SurCategorie.php : copie de Categorie.php avec "hasMany categorie"
- Modif du modèle Categorie.php : + belongsTo="SurCategorie"

c) impact sur les controleurs (cakephp/app/Controller)
- Ajout du nouveau controleur SurCategoriesController.php : minimaliste (comme CategoriesController)
- Modif du controleur CategoriesController.php : +getBySurCategorie()

d) impact sur les vues (cakephp/app/View)
- SurCategorie : no view (scaffold ?)
- Categorie : actuellement no view (scaffold ?)
--> creer une vue (comme SousCategorie)
+ get_by_surcategorie.ctp
+ scaffold.form.ctp

e) View/Pages/tools.ctp : ajouter une entree au menu pour les Sur-Categories

f) impact sur TOUTES les vues de Materiel
add/edit/view/index
dans la vue index, remplacer la categorie par la sur-categorie
GROS BOULOT sur la vue ADD/EDIT (+ javascript)

2) Deuxième exemple : ajout de la table type_suivis (VM)

Un peu plus haut est expliqué comment créer une table, je vais ici expliquer comment créer et remplir tout ce qui est basiquement nécessaire pour l'utiliser, en prenant pour exemple la table type_suivis.

Après avoir créé la table, on lui crée :
- Un controller : Héritant de AppController, avec une variable $scaffold sui se chargera de presque tout et une variable $name avec son nom.
- Un model : Avec la même variable $name et une variable $displayField qui correspond a la designation dans la BDD (ex: "nom");
  Il doit contenir la fonction typeSuiviNameIsUnik($name) {} (qu'on retrouve dans sites ou organisme) ainsi que le $validate comprenant les validations necessaires.
- Une vue, selon l'utilité, n'est pas forcément nécessaire grâce au scaffold.

Par la suite, pour lister son contenu via un lien dans "outils" par exemple, il suffit de créer le chemin dans view/pages/tools.ctp.

CREER UN CHAMP DATE (VM)
****

Pour expliquer comment créer un champ date fonctionnel, je vais prendre l'exemple de la Date de reception.
Une fois votre colonne créée dans la table materiel, vous devrez faire des modifications dans :

MaterielController : //Passer la date au format français

```
if(isset($this->request->data['Materiel']['date_reception'])){
$this->request->data['Materiel']['date_reception'] = date("d-m-Y", strtotime($this->request->data['Materiel']['date_reception'])); }
```

le Model Materiel : Créer un champ de validation adapté à la date, avec un regex. Et surtout remplir le formatage de date à l'américaine :

```
if (isset($this->data['Materiel']['date_reception'])) {
        $originalDate = $this->data['Materiel']['date_reception'];
        $this->data['Materiel']['date_reception'] = date("Y-m-d", strtotime($originalDate));
}
```

La vue Materiel : - Scaffold.view : Repasser la date en français et l'afficher.
- Scaffold.form : Créer le champ date du formulaire

Puis il faut adapter le champ date reception presque partout ou il y a le champ date acquisition.

COMMENT J'AI FAIT POUR AJOUTER UN 3eme niveau de categorie appele sur_categorie (ou domaine) (EP) ?
****

je me rends compte d'une incoherence de stockage dans la table Materiels.
En effet, quand on saisit un materiel, on doit choisir une categorie ET une sous-categorie.
Du coup, les 2 id (categorie + sous-categorie) sont stockes dans la table Materiels...
Or, c'est inutile car la sous-categorie suffit a determiner la categorie...
Cette redondance pourrait meme amener des incoherences dans la table Materiels si par exemple on fait des modifications
dans les tables categories et sous_categories sans les repercuter dans la table materiels !!
Je suppose que c'est un choix de facilite qui a ete fait par upsilon.

Donc, si vous etes ok, et a moins que Upsilon me dise qu'il y avait une bonne raison de faire ce choix (j'en doute),
je propose de modifier le code pour que seule la sous-categorie soit stockee (on ne stocke plus la categorie).

En plus, cela simplifiera l'ajout du 3eme niveau "sur-categorie" puisque lui-meme n'aura pas besoin d'etre stocke etant donne qu'il est
automatiquement determine par la categorie.
On aura alors :
sous_categorie_id -> categorie_id -> sur_categorie_id
Avec seulement la sous_categorie, on pourra determiner la categorie, et donc la sur_categorie.

Du coup, si je fais cette modif, on pourrait meme se permettre de demarrer officiellement INVENTIRAP rapidement.
En effet, l'ajout de la sur-categorie ne change rien au contenu des tables "administratives" (materiels, emprunts, suivis),
et donc on pourra le faire plus tard, meme apres que l'administration ait commence a remplir la BD.
Ca sera totalement transparent.

1) suppression de la redondance (categorie_id) dans la table materiels

a) impact sur la vue index de Materiel
add/edit/view/index

2) ajout d'une sur-categorie

a) impact sur la BD
- ajout d'une nouvelle table sur_categories(id,nom) :
```
CREATE TABLE IF NOT EXISTS sur_categories (
  id int(11) NOT NULL AUTO_INCREMENT,
  nom varchar(45) DEFAULT NULL,
  PRIMARY KEY (id),
  UNIQUE KEY nom_UNIQUE (nom)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1;
```

- ajout dans la table categories d'une cle etrangere sur_categorie_id :
```
ALTER TABLE categories
    ADD sur_categorie_id INT( 11 ) NULL DEFAULT NULL,
      ADD CONSTRAINT fk_sur_categorie_id FOREIGN KEY (sur_categorie_id) REFERENCES sur_categories (id) ON DELETE NO ACTION ON
UPDATE NO ACTION;
```

- ajout dans la table materiels d'une cle etrangere sur_categorie_id :
```
ALTER TABLE materiels
    ADD sur_categorie_id INT( 11 ) NOT NULL after designation,
    ADD CONSTRAINT fk_materiels_sur_categorie_id FOREIGN KEY (sur_categorie_id) REFERENCES sur_categories (id) ON DELETE NO
ACTION ON UPDATE NO ACTION;
```

(attention, il faut d'abord vider les lignes des tables suivis, emprunts, et materiels)
```
delete from suivis;
delete from emprunts;
delete from materiels;
```

Il faut aussi ajouter des sur-categories :
```
# Ajout de quelques sur-categories
insert into sur_categories (id,nom) values
(1,'Electronique'),
(2,'Informatique'),
(3,'Instrumentation')
```

```
(4, 'Logistique'),
(5, 'Mecanique'),
(6, 'Optique')
;

# Relier toutes les categories a une sur-categorie (la 1)
update categories set sur_categorie_id=1 where id<10;
update categories set sur_categorie_id=2 where id>=10 and id<20;
update categories set sur_categorie_id=3 where id>=20;
```

b) impact sur les modeles (cakephp/app/Model)
- SurCategorie.php : copie de Categorie.php avec "hasMany categorie"
- Categorie.php : + belongsTo="SurCategorie"

c) impact sur les controleurs (cakephp/app/Controller)
- SurCategoriesController.php : minimaliste (comme CategoriesController)
- CategoriesController.php : +getBySurCategorie()

d) impact sur les vues (cakephp/app/View)
- SurCategorie : no view (scaffold ?)
- Categorie : actuellement no view (scaffold ?)
--> creer une vue (comme SousCategorie)
+ get_by_surcategorie.ctp
+ scaffold.form.ctp

e) View/Pages/tools.ctp : ajouter une entree au menu pour les Sur-Categories

f) impact sur TOUTES les vues de Materiel
add/edit/view/index
dans la vue index, remplacer la categorie par la sur-categorie
GROS BOULOT sur la vue ADD/EDIT (+ javascript)

TESTS
****

A - EXECUTION
****

Prérequis : PHPUNIT 3 doit être déjà installé et accessible depuis la console (phpunit -version) via le fichier php.ini
(ATTENTION : cakephp 2.x n'est pas compatible avec PhpUnit 4)

Avec XAMPP (conseillé) : PhpUnit 3 est déjà inclus
Avec MAMP (+ difficile) : pour installer PhpUnit, suivre cette documentation :
http://www.dolinaj.net/software-installation/mac/how-to-install-phpunit-with-mamp-on-mac/

Procédure à suivre pour pouvoir exécuter les tests unitaires et fonctionnels livrés avec le logiciel :

1) Ajouter une nouvelle configuration de base de données dans votre fichier app/Config/database.php pour la BD de test

Exemple de configuration :

```
    public $test = array(
            'datasource' => 'Database/Mysql',
            'persistent' => false,
            'host' => 'localhost',
            'database' => 'test_labinvent',
            'login' => 'root',
            'password' => '',
    );
```

2) Créer la BD de test (vide)
A l'aide de phpmyadmin ou bien avec un client mysql quelconque,
créer une BD de test vide que vous nommerez avec le même nom que celui donné dans la configuration ci-dessus,
soit pour l'exemple, "test_labinvent"
Syntaxe SQL : "create database test_labinvent"

3) Passer en mode "debug"
Dans votre fichier de configuration labinvent.php, mettez votre paramètre "debug" à 1 ou 2 (mais pas à 0) :
	'debug' => 2,

4) Se connecter à l'application
Les test ne passeront pas si vous n'êtes pas connectés

5) Exécuter les tests

a) Exécution depuis l'application (conseillé) :
ATTENTION : vous devez être logué pour que les tests passent !!!
Il suffit d'aller à l'URL /test.php de votre installation du logiciel LabInvent
(vous pouvez aussi essayer l'URL "/app/webroot/test.php", ou encore "/cakephp/test.php")
Cette page de tests se trouve dans cakephp/app/webroot/
Vous avez alors accès à 2 types de tests :
- App : Tests ==> les tests écris pour tester l'application Labinvent
- Core : Tests ==> les tests fournis avec cakephp pour tester le framework
Pour exécuter tous les tests liés à l'application Labinvent (à faire systématiquement avant de commiter tout changement) :
cliquer sur "Tests" sous "App", puis sur "AllTests"
("AllController" exécute tous les tests de controleurs ; "AllModel" exécute tous les tests de modèles)

b) Execution depuis la console :
Aller dans le repertoire app/
Pour tester le controleur MaterielsController :
./Console/cake test app Controller/MaterielsController

B - ECRITURE DE NOUVEAUX TESTS
****

cf http://book.cakephp.org/2.0/en/development/testing.html

Aller dans app/Test/

Ce dossier contient l'arborescence suivante :

- Case/ : les tests
    - Controller/ : les tests de controleurs
    - Model/ : les tests de modèles
    - View/ (peu ou pas utilisé) : les tests de vues (ou de helpers)
    - AllControllerTest.php : exécution de tous les tests de controleurs
    - AllModelTest.php : exécution de tous les tests de modèles
    - AllTestsTest.php : éxécution de TOUS les tests

- Fixture/ : les différentes initialisations nécessaires dans la BD de test pour pouvoir executer les tests
Ces "fixtures" sont automatiquement executees AU DEBUT de chaque test.
Ce dossier contient un fichier pour chaque table pour laquelle on a besoin d'une "fixture".

1) Les "fixtures"

La façon la plus basique de creer une fixture pour une table donnee est de la realiser automatiquement a partir d'une copie de la table de la vraie BD.
Par exemple, pour que la table "categories" de la BD de test contienne la meme chose que la table de la vraie BD, il suffit de creer un fichier CategorieFixture.php contenant ceci :

```
class CategorieFixture extends CakeTestFixture {
    public $import = array('model' => 'Categorie', 'records' => true);
}
```

Au demarrage des tests, cette table sera chargee automatiquement avec les vraies donnees.
A la fin des tests, cette table sera vidée.

Dans le cas particulier de la table "materiels", on prefere l'initialiser nous-memes avec des valeurs choisies.
Exemple d'une fixture avec 2 materiels (dans le fichier MaterielFixture.php) :

```
class MaterielFixture extends CakeTestFixture {

    public $import = 'Materiel'; // import only structure, no record

    public $records = array(
```

```
                array(
                        'designation' => 'matos1',
                        'sur_categorie_id' => 1,
                        'categorie_id' => 11,
                        'materiel_administratif' => 0,
                        'materiel_technique' => 1,
                        'status' => 'CREATED',
                        'nom_createur' => 'Pallier Etienne',
                        'nom_modificateur' => 'Jean Administration',
                        'nom_responsable' => 'Jacques Utilisateur',
                        'email_responsable' => 'Jacques.Utilisateur@irap.omp.eu',
                ),
                array(
                        'designation' => 'matos2',
                        'sur_categorie_id' => 1,
                        'categorie_id' => 11,
                        'materiel_administratif' => 0,
                        'materiel_technique' => 1,
                        'status' => 'CREATED',
                        'nom_createur' => 'Pallier Etienne',
                        'nom_modificateur' => 'Jean Administration',
                        'nom_responsable' => 'Jacques Utilisateur',
                        'email_responsable' => 'Jacques.Utilisateur@irap.omp.eu',
                ),
    );
}
```

2) Les tests

Prenons l'exemple des tests ecrits pour le controleur des materiels (MaterielsController). Il devra s'appeler MaterielsControllerTest et aura la structure suivante :

```
class MaterielsControllerTest extends ControllerTestCase {
```

```
    // Liste des fixtures à charger avant l'execution des tests
    public $fixtures = array('app.materiel', 'app.sur_categorie', 'app.categorie', 'app.sous_categorie',
                    'app.groupes_thematique', 'app.groupes_metier', 'app.suivi', 'app.emprunt'
    );

    // Initialisations diverses a faire avant chaque test
    public function setUp() {
        parent::setUp();
    }

    // un 1er test
    public function testMonPremier() {
        $result = $this->testAction(...);
        $this->assert...('resultat attendu', $result);
    }

    // un 2eme test
    public function testMonDeuxieme() {
        $result = $this->testAction(...);
        $this->assert...('resultat attendu', $result);
    }

    ...

}
```

Voir le vrai fichier Test/Case/Controller/MaterielsControllerTest.php

3) Execution

Exemple avec l'execution du test MaterielsControllerTest

a) Execution depuis le site web :
/test.php?case=Controller%2FMaterielsController

Ajouter &debug=1 à l'url pour voir tous les messages de debug

b) Execution depuis la console :
Dans le repertoire app : ./Console/cake test app Controller/MaterielsController
Ajouter --debug pour voir tous les messages de debug

DATE PICKERS
****

Pour fontionner, le datePicker fait appel dans la page "View/Layout/default" à 3 scripts (jquery-1.5.2.js, jquery-1.8.12.js, DatepickerConfig.js) présents dans le repertoire "webroot/js/" et à un fichier "Theme" (smoothness.css) présent dans le repertoire "webroot/css/"

Le thème global peut très facilement être changé en téléchargeant le fichier css de son choix, à cette adresse: "http://jqueryui.com/themeroller/" et en remplaçant celui se trouvant dans "webroot/css/"

Les options du datePicker sont modifiables dans le fichier "webroot/js/DatepickerConfig.js" et sont assez explicites.
Malgré cela, pour plus de précisions, la doc est facilement consultable à cette adresse: "http://jqueryui.com/datepicker/"

14. Cycle de développement à respecter

(11 commandements)

Je veux apporter un changement (correction ou evolution) au projet, comment dois-je faire ?

1) Mettre à jour la version du projet et la date dans le fichier README.md (ça sera automatiquement affiché par src/Template/Layout/default.ctp)

2) Selectionner le changement a apporter (une demande) dans le Redmine du projet (https://projects.irap.omp.eu/projects/inventirap) :
- soit depuis la liste des demandes : onglet Demandes
- soit depuis la roadmap : onglet Roadmap, cocher la case "Anomalie", cliquer sur Appliquer, puis "version 1.3" (la version en cours depuis fin 2012)
Commencer de préférence par les "anomalies" parmi celles qui ont la plus haute priorité (et faire les "évolutions" dans un 2ème temps).
Choisir une demande et cliquer dessus pour aller sur sa fiche detaillee et voir le travail a faire.
Cliquer sur "mettre a jour", selectionner le statut "En cours", modifier eventuellement d'autres champs..., puis cliquer sur "Soumettre".
Noter l'URL de cette fiche (par exemple : https://projects.irap.omp.eu/issues/1050)
NB : Si la demande n'existe pas encore, la creer :
- cliquer sur l'onglet "Nouvelle demande"
- Selectionner "Anomalie" ou "Evolution"
- Positionner le statut a "En cours" si on veut travailler aussitot dessus (sinon, laisser a "Nouveau")
- Completer le reste de la fiche de demande (mettre "Assigne a" a "<<moi>>", choisir la version cible "version 1.3" ou "version 1.4", ...)
- cliquer sur le bouton "Creer"

3) Verifier que cette nouvelle demande apparait bien dans la roadmap (cliquer sur onglet "Roadmap", ..., puis version 1.3)

ÉTAPE OPTIONNELLE MAIS FORTEMENT CONSEILLÉE :
4) Ecrire un test qui vérifie que cette fonctionnalité ne marche pas encore (bug) ou bien n'est pas encore implémentée
On utilise ici l'approche TDD (Test Driven Development)
Ce test ne devrait pas passer (il est au rouge) ; il passera plus tard, quand on aura écrit le code nécessaire.
Bien sur, c'est dans la mesure du possible, car on ne peut pas TOUT tester.
Dans tous les cas, il faut ecrire un test qui s'approche le plus possible de la réalité à tester.
Voir pour cela la section "TESTS" (à la fin de ce document)

5) Faire les changements necessaires dans le code Php
Si ce changement implique aussi un changement dans la base de donnees,
copier le script SQL correspondant à ce changement dans un fichier database/update/db-update-YYYY-MM-DD.sql
portant la date du jour (voir des exemples dans database/update/old/).
Plus tard, il faudra penser à intégrer ce changement dans le script general de creation de la BDD database/BDD_IRAP.sql
(et/ou éventuellement les fichiers Insert_TablesFunct.sql, Insert_Users.sql, et Upd_TableConstraints.sql)
et donc deplacer le script de modification database/update/db-update-YYYY-MM-DD.sql dans database/update/old/db-update-YYYY-MM-DD.sql

6) Tester manuellement ce changement jusqu'a ce qu'il soit totalement OK
a) faire quelques tests manuels
b) Le test écrit à l'étape (4) doit maintenant passer (il est au vert).
Il doit être inclus dans l'ensemble des tests accessibles par le lien "AllTests".
c) Test de non régression : afin de s'assurer que cette modification du code n'entraîne aucune régression sur le reste du code,
tous les autres tests écrits avant doivent aussi passer (vert) : pour cela, exécuter l'url /test.php?case=AllTests

7) Compléter le fichier /README.txt, section HISTORIQUE DES VERSIONS (fichier situé à la racine du projet)
Y mettre le même commentaire que ce que tu mettras lors du commit

8) Mettre a jour TON code (en mode console, "svn update" depuis la racine du projet, ou bien depuis Eclipse, clic droit sur le projet, "Team/Update")
C'est important, au cas ou quelqu'un d'autre aurait fait des modifs (avant ou en meme temps que toi), et pour etre bien sur d'avoir la derniere version

9) Faire un "commit" du code, en collant dans le commentaire l'URL de la demande realisee (exemple : https://projects.irap.omp.eu/issues/1050)

10) Fermer la demande sur redmine
Sur la fiche detaillee, cliquer sur "Mettre a jour", changer le statut a "ferme", changer "% realise" a 100%, (copier l'URL de la fiche), cliquer sur Soumettre

11) Si c'est un changement important, le répercuter sur le site officiel
Le changement est important si c'est une anomalie ou bien si c'est une évolution attendue.
Demander alors au service informatique de le répercuter sur l'installation officielle (faire un "svn update", mail à loic.jahan@irap.omp.eu).
Si ce changement implique aussi la base de données, donner la directive que le fichier database/update/db-update-YYYY-MM-DD.sql doit être exécuté sur leur BDD.
Si ce changement impliquer une modification du fichier de configuration labinvent.php, donner la procédure à suivre dans le mail.

15. Rendons LabInvent plus “responsive”

(updated 12/2/20 - EP)

Comment ? En utilisant **bootstrap**.

15.1. Analyse de l’existant et choix d’une solution technique

Il y a environ 4 “plugins” (parfois “helpers”) bootstrap disponibles pour cakephp, dont 2 qui semblent vraiment obsolètes.

Obsolètes :
- https://github.com/WebAndCow/CakePHP-BsHelpers
- ...

Actifs :
- https://holt59.github.io/cakephp3-bootstrap-helpers/ (et https://github.com/Holt59/cakephp3-bootstrap-helpers)
- https://github.com/FriendsOfCake/bootstrap-ui/tree/develop
- https://twitter-bootstrap-plugin-for-cakephp-3.readthedocs.io/en/latest/readme/ qui se base sur **bootstrap-ui** ci-dessus

On a choisi "**bootstrap-ui**" car géré par une communauté d’utilisateurs FOC (contrairement à holt59 qui est seul à maintenir son projet bien qu’actif...), dans sa version de développement qui utilise bootstrap 4 (ça serait un peu bête d'utiliser bootstrap 3 quand bootstrap4 est proposé par défaut sur le site officiel) :

https://github.com/FriendsOfCake/bootstrap-ui/tree/develop (dernier commit novembre 2019 donc c'est actif...)

Bootstrap est proposé par défaut en version 4 (et déjà en 4.4) : https://getbootstrap.com
Doc :
- https://getbootstrap.com/docs/4.4/getting-started/introduction/
- https://getbootstrap.com/docs/4.4/examples/

Plus d’infos sur les “**helpers**” cakephp : https://book.cakephp.org/3/en/views/helpers.html

15.2. Dépendances

Apparemment (à confirmer), bootstrap utiliserait :

- jQuery 3.2+ : https://jquery.com
- Popper.js 1.x : https://github.com/popperjs/popper-core (j'ai téléchargé directement : https://unpkg.com/@popperjs/core@2.0.3/dist/umd/popper.min.js)
- Fontawesome 5.x (https://fontawesome.com)

15.3. Installation du plugin bootstrap-ui

https://github.com/FriendsOfCake/bootstrap-ui/tree/develop

1 - installation du plugin

```
$ cd labinvent/
$ php composer.phar require friendsofcake/bootstrap-ui
```

=> a ajouté la ligne suivante dans composer.json :
"friendsofcake/bootstrap-ui": "^1.4"
=> a modifié composer.lock en conséquence

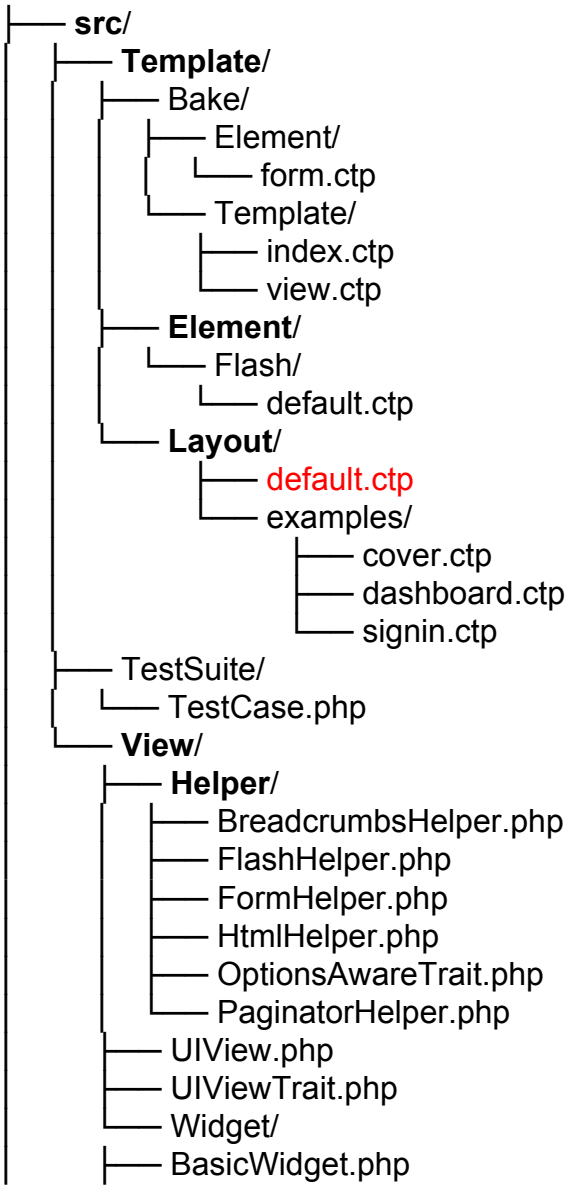
Warnings :
Package aferrandini/phpqrcode is abandoned, you should avoid using it. Use endroid/qr-code instead.
Package phpunit/phpunit-mock-objects is abandoned, you should avoid using it. No replacement was suggested.
Package zendframework/zend-diactoros is abandoned, you should avoid using it. Use laminas/laminas-diactoros instead.

2 - Load
Add the following to your app's **config/boostrap.php**:

```
Plugin::load('BootstrapUI');
```

15.4. Fichiers installés par bootstrap-ui

Contenu de vendor/friendsofcake/bootstrap-ui/



```
│       ├── ButtonWidget.php
│       ├── CheckboxWidget.php
│       ├── InputgroupTrait.php
│       ├── RadioWidget.php
│       ├── SelectBoxWidget.php
│       └── TextareaWidget.php
├── tests/
└── webroot/
        └── css/
                ├── cover.css
                ├── dashboard.css
                └── signin.css
```

15.5. Utilisation

The **src\View\AppView.php** should look something like the following:

```
namespace App\View;
use BootstrapUI\View\UIView;
class AppView extends UIView
{

    /**
     * Initialization hook method.
     */
    public function initialize()
    {
        //Don't forget to call the parent::initialize()
        parent::initialize();
    }
}
```

or

```
namespace App\View;
use BootstrapUI\View\UIViewTrait;
use Cake\View\View;
class AppView extends View
{
    use UIViewTrait;

    /**
     * Initialization hook method.
     */
    public function initialize()
    {
        //render the initializeUI method from the UIViewTrait
        $this->initializeUI();
    }
}
```

Layout

When no layout for the view is defined the **BootstrapUI\View\UIViewTrait** will load its own **default.ctp** layout file

Helper Usage

(voir la section du même nom sur la page de bootstrap-ui pour avoir quelques exemples)

At the core of BootstrapUI is a collection of enhancements for CakePHP core helpers. These helpers replace the HTML templates used to render elements for the views. This allows you to create forms and components that use the Bootstrap styles.

The current list of enhanced helpers are:

- BootstrapUI\View\Helper\FlashHelper
- BootstrapUI\View\Helper\FormHelper
- BootstrapUI\View\Helper\HtmlHelper

- BootstrapUI\View\Helper\PaginatorHelper
- BootstrapUI\View\Helper\BreadcrumbsHelper

When the BootstrapUI\View\UIViewTrait is initialized it loads the above helpers with the same aliases as the CakePHP core helpers. That means that when you use $**this->Form->create**() in your views, the helper being used is from the BootstrapUI plugin.
⇒ voir dans src/Template/ (par exemple src/Template/Materiels/add.ctp)

16. Migrations de la BD

https://book.cakephp.org/3.0/fr/migrations.html
et
https://book.cakephp.org/3.0/fr/phinx.html
et
https://blog.osmosys.asia/2017/04/17/schema-migration-in-cakephp-3-x/
et
https://www.sanisoft.com/blog/2014/10/20/migrations-cakephp-3-quickstart/

IMPORTANT:
Ce chapître est dans une **forme temporaire**. C'est une **réflexion en cours** sur l'opportunité ou pas d'utiliser le plugin "migrations" pour gérer les modifications faites sur la BD. Pour l'instant, on n'utilise pas ce plugin, on gère les modifs manuellement en créant des scripts db-update.sh (dans le dossier database/update/). Il est donc **inutile de lire ce chapître pour l'instant**, sauf si vous êtes intéressés par le sujet.

16.1. Procédure proposée pour garder la BD à jour suite à nos modifs (pour qu'on ait tous la même version de la BD)

1 - Création d'un fichier de migration initial contenant tout le schéma de la BD actuelle, qui sera le schéma de référence
Sans doute pas nécessaire, on peut passer directement à l'étape suivante
$ bin/cake **bake migration_snapshot Initial**
*Creating file config/Migrations/**20190111105729_Initial.php***

2 - Dump initial de la BD pour avoir une version de référence (version initiale)
$ bin/cake **migrations dump**
using migration paths config/Migrations and seed paths config/Seeds
Writing dump file config/Migrations/schema-dump-default.lock

3 - Modification de la BD
On peut imaginer tout un tas de modifs telles que ajout ou suppression de tables ou colonnes, mais attention **CakePhp ne sait pas gérer le "renommage" de colonnes**. Dans ce cas, il faudra ajouter manuellement ces renommages dans les fichiers de migration générés.

4 - Générer un fichier de migration qui fait un DIFF entre la BD actuelle et la BD de référence
$ bin/cake **bake migration_diff NameOfTheMigrations**
Eventuellement, modifier ce fichier généré pour y ajouter manuellement les "renommages" de colonnes (car pas gérés automatiquement)

5 - Commiter (push) ce fichier NameOfTheMigrations sur le dépot de référence pour que tout le monde y ait accès
$ git commit
$ git push

6 - Application de la migration pour etre synchro

- Pour appliquer toutes les migrations :

$ bin/cake migrations migrate
(Pour annuler ces migrations (rollback, retour en arrière): $ bin/cake migrations rollback)
(# You can also pass a migration version number to rollback to a specific version:: $ bin/cake migrations rollback -t 20150103081132)
(optionally up to a specific version : migrate -e development)

- Pour appliquer une migration particulière (la dernière par exemple) :

# Migrate to a specific version using the ``--target`` option or ``-t`` for short.
# The value is the timestamp that is prefixed to the migrations file name::
$ bin/cake migrations migrate -t 20150103081132

16.2. Présentation du concept

Migrations est un plugin pour aider à **gérer les changements dans la base de données** en écrivant des fichiers PHP (qui peuvent d'ailleurs être versionnés par le système de gestion de version). Il permet de faire évoluer les tables au fil du temps. Au lieu d'écrire les modifications de schéma en SQL, ce plugin permet d'utiliser un ensemble intuitif de méthodes qui facilite la mise en œuvre des modifications au sein de la base de données.

Une migration est simplement un fichier PHP qui décrit les changements à effectuer sur la base de données. Un fichier de migration peut créer ou supprimer des tables, ajouter ou supprimer des colonnes, créer des index et même insérer des données dans votre base de données.

Ci-dessous un exemple de migration:

```
<?php
use Migrations\AbstractMigration;

class CreateProducts extends AbstractMigration
{
    /**
     * Change Method.
     *
     * More information on this method is available here:
     * http://docs.phinx.org/en/latest/migrations.html#the-change-method
     * @return void
     */
    public function change()
    {
        $table = $this->table('products');
        $table->addColumn('name', 'string', [
            'default' => null,
            'limit' => 255,
            'null' => false,
        ]);
        $table->addColumn('description', 'text', [
            'default' => null,
            'null' => false,
        ]);
        $table->addColumn('created', 'datetime', [
            'default' => null,
            'null' => false,
        ]);
        $table->addColumn('modified', 'datetime', [
            'default' => null,
            'null' => false,
```

```
        ]);
        $table->create();
    }
}
```

Cette migration va ajouter une table à votre base de données nommée products avec les définitions de colonne suivantes:

- id colonne de type integer comme clé primaire
- name colonne de type string
- description colonne de type text
- created colonne de type datetime

*La colonne avec clé primaire nommée id sera ajoutée **implicitement**.*
*Notez que ce fichier décrit ce à quoi la base de données devrait ressembler **après** l'application de la migration. À ce stade la table products n'existe pas dans votre base de données, nous avons simplement créé un fichier qui est à la fois capable de créer la table products avec les bonnes colonnes mais aussi de supprimer la table quand une opération de rollback (retour en arrière) de la migration est effectuée.*

Une fois que le fichier a été créé dans le dossier **config/Migrations**, vous pourrez exécuter la commande migrations suivante pour créer la table dans votre base de données:
$ bin/cake migrations migrate

La commande migrations suivante va effectuer un rollback (retour en arrière) et supprimer la table de votre base de données:
$ bin/cake migrations rollback

Les fichiers de migrations sont stockés dans le répertoire **config/Migrations** de votre application. Le nom des fichiers de migration est précédé de la date/heure du jour de création, dans le format **YYYYMMDDHHMMSS_MigrationName.php**

16.3. Exemples divers

Synchroniser la BD à partir des classes du Model (src/Model/) :
$ bin/cake **migrations migrate**

Rollback:

$ bin/cake **migrations rollback**

Avec bake:
$ bin/cake **bake migration** CreateProducts name:string description:text created modified
avec :

- le nom de la migration que vous allez générer (CreateProducts dans notre exemple)
- les colonnes de la table qui seront ajoutées ou retirées dans la migration (name:string description:text created modified dans notre exemple)

Créer un fichier de migration vide pour avoir un contrôle total sur ce qui doit être exécuté, en ne spécifiant pas de définition de colonnes:
$ bin/cake **migrations create** MyCustomMigration

16.4. Générer une Migration à partir d'une Base de Données Existante

https://book.cakephp.org/3.0/fr/migrations.html#generer-une-migration-a-partir-d-une-base-de-donnees-existante

Si on a affaire à une base de données pré-existante (labinvent) et qu'on veut commencer à utiliser migrations, ou qu'on souhaite versionner le schéma initial de la base de données, on peut exécuter la commande migration_snapshot:
$ bin/cake **bake migration_snapshot** Initial

Elle va générer un fichier de migration appelé Initial contenant toutes les déclarations pour toutes les tables de la base de données. Par défaut, le snapshot va être créé en se connectant à la base de données définie dans la configuration de la connection **default** (si vous devez créer un snapshot à partir d'une autre source de données, vous pouvez utiliser l'option **--connection**). Vous pouvez aussi vous assurer que le snapshot inclut **seulement les tables pour lesquelles vous avez défini les classes de model correspondantes** en utilisant le flag **--require-table**. Quand vous utilisez ce flag, le shell va chercher les classes Table de votre application et va seulement ajouter les tables de model dans le snapshot.

Notez que quand vous créez un snapshot, il est automatiquement **marqué** (dans la table de log de phinx) **comme migré.**

⇒ Exécution faite le 11/1/19 (par EP) :
$ bin/cake bake migration_snapshot DB_complete
Creating file config/Migrations/20190111105729_DBComplete.php
*Marking the migration 20190111105729_DBComplete as **migrated**… (using migration paths config/Migrations/, and seed paths config/Seeds/)*
Creating a dump of the new database state...

Writing dump file config/Migrations/schema-dump-default.lock`...

16.5. Nom de Fichier des Migrations

Les noms des migrations peuvent suivre l'une des structures suivantes:

- (/^(Create)(.*)/) Crée la table spécifiée.
- (/^(Drop)(.*)/) Supprime la table spécifiée. Ignore les arguments de champ spécifié.
- (/^(Add).*(?:To)(.*)/) Ajoute les champs à la table spécifiée.
- (/^(Remove).*(?:From)(.*)/) Supprime les champs de la table spécifiée.
- (/^(Alter)(.*)/) Modifie la table spécifiée. Un alias pour CreateTable et AddField.

Vous pouvez aussi utiliser la_forme_avec_underscores comme nom pour vos migrations par exemple create_products.

16.6. Définition de Colonnes

Quand vous définissez des colonnes avec la ligne de commande, il peut être pratique de se souvenir qu'elles suivent le modèle suivant:
fieldName:fieldType?[length]:indexType:indexName

Par exemple, les façons suivantes sont toutes des façons valides pour spécifier un champ d'email:

- email:string?
- email:string:unique
- email:string?[50]
- email:string:unique:EMAIL_INDEX
- email:string[120]:unique:EMAIL_INDEX

Le point d'interrogation qui suit le type du champ entrainera que la colonne peut être null.
Le paramètre length pour fieldType est optionnel et doit toujours être écrit entre crochets.

Les champs nommés created et modified, tout comme les champs ayant pour suffixe _at, vont automatiquement être définis avec le type datetime.
Les types de champ sont ceux qui sont disponibles avec la librairie Phinx. Ce sont les suivants:

- string
- text
- integer
- biginteger
- float
- decimal
- datetime
- timestamp
- time
- date
- binary
- boolean
- uuid

Il existe quelques heuristiques pour choisir les types de champ quand ils ne sont pas spécifiés ou définis avec une valeur invalide. Par défaut, le type est string:

- id: integer
- created, modified, updated: datetime

16.7. Créer une Table

Vous pouvez utiliser bake pour créer une table:
$ bin/cake **bake migration CreateProducts name:string description:text created modified**

La ligne de commande ci-dessus va générer un fichier de migration qui ressemble à:

```
<?php
use Migrations\AbstractMigration;

class CreateProducts extends AbstractMigration
```

```
{
    /**
     * Change Method.
     *
     * More information on this method is available here:
     * http://docs.phinx.org/en/latest/migrations.html#the-change-method
     * @return void
     */
    public function change()
    {
        $table = $this->table('products');
        $table->addColumn('name', 'string', [
            'default' => null,
            'limit' => 255,
            'null' => false,
        ]);
        $table->addColumn('description', 'text', [
            'default' => null,
            'null' => false,
        ]);
        $table->addColumn('created', 'datetime', [
            'default' => null,
            'null' => false,
        ]);
        $table->addColumn('modified', 'datetime', [
            'default' => null,
            'null' => false,
        ]);
        $table->create();
    }
}
```

16.8. Ajouter des Colonnes à une Table Existante

Si le nom de la migration dans la ligne de commande est de la forme « AddXXXToYYY » et est suivie d'une liste de noms de colonnes et de types alors un fichier de migration contenant le code pour la création des colonnes sera généré:
$ bin/cake **bake migration AddPriceToProducts price:decimal**

L'exécution de la ligne de commande ci-dessus va générer:

```
<?php
use Migrations\AbstractMigration;

class AddPriceToProducts extends AbstractMigration
{
    public function change()
    {
        $table = $this->table('products');
        $table->addColumn('price', 'decimal')
              ->update();
    }
}
```

16.9. Générer un diff entre deux états de base de données

Nouveau dans la version cakephp/migrations: 1.6.0
Vous pouvez générer un fichier de migrations qui regroupera toutes les différences entre deux états de base de données en utilisant le template bake migration_diff. Pour cela, vous pouvez utiliser la commande suivante:
$ bin/cake bake migration_diff NameOfTheMigrations

Pour avoir un point de comparaison avec l'état actuel de votre base de données, le **shell migrations va générer, après chaque appel de migrate ou rollback un fichier « dump »**. **Ce fichier dump est un fichier qui contient l'ensemble de l'état de votre base de données à un point précis dans le temps**.
Quand un fichier dump a été généré, toutes les modifications que vous ferez directement dans votre SGBD seront ajoutées au fichier de migration qui sera généré quand vous appelerez la commande bake migration_diff.

Si vous souhaitez utiliser la fonctionnalité de diff sur une application qui possède déjà un historique de migrations, vous allez avoir besoin de créer le fichier dump manuellement pour qu'il puisse être utilisé comme point de comparaison:

$ bin/cake migrations dump

L'état de votre base de données devra être le même que si vous aviez migré tous vos fichiers de migrations avant de créer le fichier dump. Une fois que le fichier dump est créé, vous pouvez opérer des changements dans votre base de données et utiliser la commande bake migration_diff quand vous voulez.

ATTENTION: le système n'est pas capable de détecter les colonnes renommées.

16.10. Les Commandes

16.10.1. migrate : Appliquer les Migrations

Une fois que vous avez généré ou écrit votre fichier de migration, vous devez exécuter la commande suivante pour appliquer les modifications à votre base de données:
# Exécuter toutes les migrations
$ bin/cake migrations migrate

16.10.2. status : Statuts de Migrations

La commande status affiche une liste de toutes les migrations, ainsi que leur état actuel. Vous pouvez utiliser cette commande pour déterminer les migrations qui ont été exécutées:
$ bin/cake **migrations status**

16.10.3. seed : Remplir votre Base de Données (Seed)

Depuis la version 1.5.5, vous pouvez utiliser le shell migrations pour remplir votre base de données.
Par défaut, les fichiers de seed vont être recherchés dans le répertoire config/Seeds de votre application.

Une interface bake est fournie pour les fichiers de seed:
Ceci va créer un fichier ArticlesSeed.php dans le répertoire config/Seeds
de votre application
Par défaut, la table que le seed va essayer de modifier est la version
"tableized" du nom de fichier du seed
$ bin/cake bake seed Articles
⇒ Ca ne copie pas les données, ça crée seulement un "seeder" que l'on peut customiser pour remplir la BD, mais par défaut, il n'insèrera aucune donnée dans la base

Ex: table "configurations"
$ bin/cake bake seed Configurations
Creating file ***config/Seeds/ConfigurationsSeed.php***

```
<?php
use Migrations\AbstractSeed;

/**
 * Configurations seed.
 */
class ConfigurationsSeed extends AbstractSeed
{
        /**
        * Run Method.
        *
        * Write your database seeder using this method.
        *
        * More information on writing seeds is available here:
        * http://docs.phinx.org/en/latest/seeding.html
        *
        * @return void
        */
        public function run()
```

```
        {
        $data = [];

        $table = $this->table('configurations');
        $table->insert($data)->save();
        }
}
```

Les options **--data**, --limit and --fields ont été ajoutées pour permettre d'exporter des données extraites depuis votre base de données.
A partir de 1.6.4, la commande bake seed vous permet de créer des fichiers de seed avec des lignes exportées de votre base de données en utilisant l'option --data:
$ bin/cake bake seed --data Articles
Par défaut, cela exportera toutes les lignes trouvées dans la table.

Ex: table "configurations"
$ bin/cake bake seed --data Configurations
*Creating file **config/Seeds/ConfigurationsSeed.php***

```
<?php
use Migrations\AbstractSeed;

/**
 * Configurations seed.
 */
class ConfigurationsSeed extends AbstractSeed
{
        /**
        * Run Method.
        *
        * Write your database seeder using this method.
        *
        * More information on writing seeds is available here:
        * http://docs.phinx.org/en/latest/seeding.html
        *
        * @return void
        */
```

```
public function run()
{
$data = [
[
        'id' => '1',
        'nom' => 'default',
        'mode_install' => '0',
        'mode_debug' => '0',
        'labNameShort' => 'IRAP',
        'labPresent' => 'de l\'',
        'labUmr' => 'UMR 5277',
        'hasPrinter' => '1',
        'nom_groupe_thematique' => 'Groupe thematique',
        'nom_groupe_metier' => 'Groupe metier',
        'envoi_mail' => '1',
        'envoi_mail_guests' => '1',
        'emailGuest1' => 'inventirap@labo',
        'emailGuest2' => '',
        'emailGuest3' => '',
        'emailGuest4' => '',
        'emailGuest5' => 'etienne.pallier@labo’,
        'test' => NULL,
        'prix_inventaire_administratif' => '800',
        'emailGuest6' => 'epallier@labo',
        'emailGuest7' => 'etienne.pallier@perso',
        'emailGuest8' => '',
        'emailGuest9' => '',
        'emailGuest10' => '',
        'sender_mail' => 'inventirap@labo',
        'labName' => 'Institut de Recherche en Astrophysique et Planétologie',
        'date_commande_facultative' => '0',
        'numero_labo_sans_annee' => '0',
        'taille_max_doc' => '8000000',
        'metrologie' => '1',
        'aff_par_defaut' => '20',
        'procedure_sur_accueil' => '1',
        'ldap_used' => '0',
```

```
                'ldap_authentified' => '1',
                'ldap_bindDn' => '',
                'ldap_bindPass' => '',
                'ldap_host' => 'ldaps://ldap2.labo',
                'ldap_port' => 'port'',
                'ldap_authenticationType' => 'uid',
                'ldap_baseDn' => '...',
                'ldap_filter' => '...',
        ],
        ];

        $table = $this->table('configurations');
        $table->insert($data)->save();
        }
}
```

Appliquer un seed pour alimenter la BD

Pour faire un seed de votre base de données, vous pouvez utiliser la sous-commande **seed**:
Sans paramètres, la sous-commande seed va exécuter tous les seeders
disponibles du répertoire cible, dans l'ordre alphabétique.
$ bin/cake migrations seed

Vous pouvez **spécifier seulement un seeder** à exécuter en utilisant l'option `--seed`
$ bin/cake migrations seed --seed ArticlesSeed

Ex: on supprime les données de la table "configurations", puis on les remet avec "migrations seed" en utilisant le seed **ConfigurationsSeed** créé ci-dessus, easy !!! :
(from PhpMyadmin) delete from configurations;
$ bin/cake **migrations seed** --seed ConfigurationsSeed

Notez que, **à l'opposé des migrations, les seeds ne sont pas suivis**, ce qui signifie que le même seeder peut être appliqué plusieurs fois.

16.11. Trucs et Astuces

16.11.1. Mettre à jour les Noms de Colonne et Utiliser les Objets Table

Si vous utilisez un objet Table de l'ORM de CakePHP pour manipuler des valeurs de votre base de données, comme renommer ou retirer une colonne, assurez-vous de créer une nouvelle instance de votre objet Table après l'appel à update(). Le registre de l'objet Table est nettoyé après un appel à update() afin de rafraîchir le schéma qui est reflèté et stocké dans l'objet Table lors de l'instanciation.

16.11.2. Migrations et déploiement¶

Si vous utilisez le plugin dans vos processus de déploiement, assurez-vous de vider le cache de l'ORM pour qu'il renouvelle les _metadata_ des colonnes de vos tables. Autrement, vous pourrez rencontrer des erreurs de colonnes inexistantes quand vous effectuerez des opérations sur vos nouvelles colonnes. Le Core de CakePHP inclut un Shell de Cache du Schéma que vous pouvez utilisez pour vider le cache:
// Avant 3.6, utilisez orm_cache
$ bin/cake schema_cache clear

Veuillez vous référer à la section du cookbook à propos du Shell du Cache du Schéma si vous voulez plus de détails à propos de ce shell.

16.11.3. Renommer une table

Le plugin vous donne la possibilité de renommer une table en utilisant la méthode rename(). Dans votre fichier de migration, vous pouvez utiliser la syntaxe suivante:

```
public function up()
{
    $this->table('old_table_name')
        ->rename('new_table_name');
}
```

16.11.4. Ne pas générer le fichier schema.lock

Nouveau dans la version cakephp/migrations: 1.6.5
Pour que la fonctionnalité de « diff » fonctionne, un fichier **.lock** est généré à chaque que vous faites un migrate, un rollback ou que vous générez un snapshot via bake pour permettre de suivre l'état de votre base de données à n'importe quel moment. Vous pouvez empêcher que ce fichier ne soit généré, comme par exemple lors d'un déploiement sur votre environnement de production, en utilisant l'option --no-lock sur les commandes mentionnées ci-dessus:

```
$ bin/cake migrations migrate --no-lock

$ bin/cake migrations rollback --no-lock

$ bin/cake bake migration_snapshot MyMigration --no-lock
```

17. Le framework CakePhp

Communautés : https://plus.google.com/communities/108328920558088369819

17.1. Erreurs sur le framework (TODO)

- **Deprecated Error**: strpos(): Non-string needles will be interpreted as strings in the future.

Use an explicit chr() call to preserve the current behavior in **vendor/cakephp/cakephp/src/Routing/RequestActionTrait.php, line 119**
Solution: remplacer

```
//if (is_string($url) && strpos($url, $baseUrl) === 0) {
```

par

```
if (is_string($url) && strpos($url, chr($baseUrl)) === 0) {
```

- **Correction d'une petite erreur sur MessageTrait.php, ligne 126**, qui provoque un warning à l'intérieur des pdf (doc de sortie notamment), ce qui empêche de l'ouvrir avec Acrobat Reader.

Ouvre le fichier vendor/zendframework/zend-diactoros/src/**MessageTrait.php**
et changer la ligne 126 en ceci :

```
//return isset($this->headerNames[strtolower($header)]);
return isset($this->headerNames[strtolower($header[0])]);
```

17.2. Mise à jour du framework CakePhp

17.2.1. Upgrade de la version 3.6 à la version 3.7 (16/1/19 pm)

cf https://book.cakephp.org/3.next/en/appendices/3-7-migration-guide.html

Version courante:
$ **bin/cake version**
=> 3.6

Upgrade:
$ rm composer.lock (c'est mieux)
$ php composer.phar **require** --update-with-dependencies "**cakephp/cakephp:3.7**.*"

./composer.json has been updated
Loading composer repositories with package information
Updating dependencies (including require-dev)
  - Removing zendframework/zend-diactoros (1.7.0)
  - Installing zendframework/zend-diactoros (1.8.6)
        Loading from cache

  - Installing psr/simple-cache (1.0.1)
        Loading from cache

  - Removing psr/log (1.0.2)
  - Installing psr/log (1.1.0)
        Loading from cache

  - Removing cakephp/chronos (1.1.4)
  - Installing cakephp/chronos (1.2.3)
        Loading from cache

  - Removing cakephp/cakephp (3.5.11)
  - Installing cakephp/cakephp (3.7.2)
        Downloading: 100%

Writing lock file
Generating autoload files
> Cake\Composer\Installer\PluginInstaller::postAutoloadDump

Nouvelle version:

$ **bin/cake version**
=> 3.7.2
(avec plein de “Deprecated Error”)
ex:
Deprecated Error: EventDispatcherTrait::eventManager() is deprecated. Use EventDispatcherTrait::setEventManager()/getEventManager() instead. - vendor/cakephp/debug_kit/src/Panel/PanelRegistry.php, line: 35
You can disable deprecation warnings by setting `Error.errorLevel` to `E_ALL & ~E_USER_DEPRECATED` in your config/app.php. *[vendor/cakephp/cakephp/src/Core/functions.php, line 311]*

Mais pb sur le site web : “Property _transportConfig does not exist“
Cette page me donne la solution : https://github.com/cakephp/cakephp/issues/12797
⇒ You have to update debug kit also, not documented yet.
"cakephp/debug_kit": "^3.17.0",

Je mets donc à jour la ligne debug_kit dans mon composer.json et je réinstalle les dépendances de composer :
$ rm composer.lock (c’est mieux)
$ **php composer.phar install**
Loading composer repositories with package information
Updating dependencies (including require-dev)
- Removing cakephp/debug_kit (3.16.7)
- **Installing cakephp/debug_kit (3.17.0)**
    Loading from cache

- Removing cakephp/bake (1.8.7)
- **Installing cakephp/bake (1.9.2)**
    Downloading: 100%

Writing lock file
Generating autoload files
> Cake\Composer\Installer\PluginInstaller::postAutoloadDump

Et là, c’est nickel !

Bon, il y a encore énormément de “deprecated” errors dans vendor/ (et cela aussi avec cakephp 3.6)

17.2.2. Upgrade de la version 3.5 à la version 3.6 (16/1/19 matin)

https://book.cakephp.org/3.next/en/appendices/3-6-migration-guide.html
et
https://www.dereuromark.de/2018/03/14/cakephp-3-6-is-coming/#migration-tooling

Version courante:
$ **bin/cake version**
=> 3.5.11

Upgrade:
$ php composer.phar **require** --update-with-dependencies "**cakephp/cakephp:3.6**.*"

./composer.json has been updated
Loading composer repositories with package information
Updating dependencies (including require-dev)
  - Removing psr/simple-cache (1.0.1)
  - Removing cakephp/cakephp (3.7.2)
  - Installing cakephp/cakephp (3.6.14)
        Downloading: 100%

Writing lock file
Generating autoload files
> Cake\Composer\Installer\PluginInstaller::postAutoloadDump

Nouvelle version:
$ bin/cake version
3.6.14
⇒ Mais, il y a plein de “deprecated” errors !!!

Remettre au propre les dépendances de Composer :

$ rm composer.lock
$ php composer.phar install
⇒ ce qui crée un nouveau composer.lock plus à jour
(notamment, phpunit est passé de la version 5 à la version 6)

Check cakephp version à nouveau:
=> il n'y a plus que 1 erreur "deprecated" contre 5 avant !!!

$ bin/cake version
3.6.14
Deprecated Error: Use Cake\Http\ServerRequest instead of Cake\Network\Request. - **vendor/composer/ClassLoader.php**, line: 412
You can disable deprecation warnings by setting `Error.errorLevel` to `E_ALL & ~E_USER_DEPRECATED` in your config/app.php

Bugfix quelques deprecated errors avec l'outil rector :
https://www.dereuromark.de/2018/03/14/cakephp-3-6-is-coming/
⇒ mais pb de dépendance pour installer rector, donc je laisse tomber

17.3. PLUGINS

17.3.1. "Composer", le gestionnaire de plugins

Pour pouvoir gérer les plugins, il faut d'abord installer **composer**.
Cet exécutable est **déjà installé dans LabInvent**, mais voici comment le ré-installer si besoin :
Depuis la racine du projet, exécuter les 4 lignes de php qui sont données au début de cette page web :
https://getcomposer.org/download/
Cela crée un fichier composer.phar qui va nous permettre de gérer les plugins.

17.3.2. Plugins spécifiques à Cakephp

https://book.cakephp.org/3.0/fr/plugins.html

On peut facilement étendre les capacités de CakePhp à l'aide de plugins (à tester).

Les meilleurs plugins : https://github.com/FriendsOfCake/awesome-cakephp
- plugin PDF: https://github.com/FriendsOfCake/awesome-cakephp#pdf
- search : https://github.com/FriendsOfCake/awesome-cakephp#search

Tous les plugins officiels : https://plugins.cakephp.org/

17.3.3. Les plugins installés pour LabInvent

(updated 13/5/19 - EP)

TODO: cette liste n'est pas exhaustive, il faut la compléter

17.3.3.1. Génération des QrCodes (plugin phpqrcode)

TODO: Attention, ce plugin est obsolète, il faudrait le remplacer dès que possible

"Package aferrandini/phpqrcode is abandoned, you should avoid using it. Use **endroid/qr-code** instead"

17.3.3.2. Génération des documents PDF

Historiquement on a utilisé le plugin fpdf (depuis cakphp v2), puis en mai 2019 on a installé le plugin cakephp-dompdf pour le CRAL (car fpdf ne marchait pas bien sur cakephp v3 chez eux en php7).

Autres solutions possibles (mais pas testées) :

- Avec le plugin Cakepdf :
  - https://github.com/FriendsOfCake/CakePdf
  - https://pritomkumar.blogspot.com/2017/04/generate-pdf-in-cakephp-3x-with-cakepdf.html
- Sans plugin:
  - http://caketuts.key-conseil.fr/index.php/2015/05/18/generer-du-pdf-sous-cakephp-v3/

17.3.3.2.1. Le Plugin fpdf

Voir la faq de fpdf: http://www.fpdf.org/fr/FAQ.php

17.3.3.2.2. Le plugin cakephp-dompdf pour générer des docs pdf

(installé le 19/5/2019)

(Test démo : http://labinvent.test/demo/view/test.pdf)

cf http://www.flavienbeninca.fr/2016/05/04/generer-des-pdf-avec-dompdf-et-cakephp.html

(voir aussi https://github.com/DaoAndCo/cakephp-dompdf)

Installation en 2 instructions :

$ php composer.phar require daoandco/cakephp-dompdf

$ bin/cake plugin assets symlink
(cette dernière instruction est sans doute à faire après installation de tout plugin)
Cela a créé 2 liens à la racine du dossier webroot/ :

- debug_kit@ -> vendor/cakephp/debug_kit/webroot
- dompdf@ -> vendor/daoandco/cakephp-dompdf/webroot

Ensuite, il faut faire quelques changements dans les fichiers config/bootstrap.php et config/routes.php

… etc (voir lien ci-dessus)

17.3.4. Mettre à jour ou Réinstaller tous les plugins (dossier vendor/)

(Réinstallation faite le 15/4/19 avec php 7)

D'abord mettre à jour le fichier /composer.phar :

$ php composer.phar **self-update**

Ensuite, mettre à jour ou réinstaller tous les plugins:

- Pour une simple **mise à jour** :
  $ php composer.phar **update**

- Pour une **réinstallation complète** :
  (Pour être bien sûr, on peut carrément supprimer le dossier vendor/, il sera re-créé à la réinstallation) : $ rm -rf vendor/
  $ php composer.phar **install**
  Warnings après installation:
  - Package aferrandini/**phpqrcode** is abandoned, you should avoid using it. Use endroid/qr-code instead.
  - Package phpunit/**phpunit-mock-objects** is abandoned, you should avoid using it. No replacement was suggested

Attention, dans le cas d'une réinstallation, les dossiers tmp/cache/views, tmp/sessions, et tmp/tests sont re-créés et leurs permissions mises à jour :
Loading composer repositories with package information
Installing dependencies (including require-dev) from lock file
Nothing to install or update
Generating autoload files
> Cake\Composer\Installer\PluginInstaller::postAutoloadDump
> App\Console\Installer::postInstall
Created `/Users/epallier/Documents/_W_more/PROJECTS/LABINVENT/SOURCE/labinvent2/tmp/cache/views` directory
Created `/Users/epallier/Documents/_W_more/PROJECTS/LABINVENT/SOURCE/labinvent2/tmp/sessions` directory
Created `/Users/epallier/Documents/_W_more/PROJECTS/LABINVENT/SOURCE/labinvent2/tmp/tests` directory
Set Folder Permissions *? (Default to Y) [Y,n]?* ***Y***
Permissions set on /Users/epallier/Documents/_W_more/PROJECTS/LABINVENT/SOURCE/labinvent2/tmp/cache/views
Permissions set on /Users/epallier/Documents/_W_more/PROJECTS/LABINVENT/SOURCE/labinvent2/tmp/sessions
Permissions set on /Users/epallier/Documents/_W_more/PROJECTS/LABINVENT/SOURCE/labinvent2/tmp/tests
No Security.salt placeholder to replace.

17.3.5. Ajouter un plugin spécifique

https://book.cakephp.org/3.0/fr/plugins.html#installer-un-plugin-avec-composer

Exemple: on veut ajouter le plugin **fpdf de setasign** (pour générer des fichiers pdf)
Ne le faites pas, car c'est déjà fait pour LabInvent. C'est juste pour expliquer comment faire.

Installation automatique (méthode conseillée) :

Il suffit de taper cette commande :

```
$ php composer.phar require setasign/fpdf
```

Ceci installe la dernière version de fpdf, et met à jour vos fichiers **composer.json**, **composer.lock**, met à jour **vendor/cakephp-plugins.php** et met à jour votre autoloader.

Installation manuelle :

Il suffit d'ajouter le nom du plugin dans le fichier composer.json dans la clé "require", à la fin, comme ceci :

```
"require" : {
        "php" : ">=5.6",
        "cakephp/cakephp" : "3.7.*",
        "cakephp/migrations" : "^2.0.0",
        "cakephp/plugin-installer" : "^1.0",
        "mobiledetect/mobiledetectlib" : "2.*",
        "aferrandini/phpqrcode" : "1.0.1",
        "setasign/fpdf" : "1.8.1"
},
```

Ensuite, il faut relancer l'installation des plugins (ils sont tous installés dans le dossier **vendor**/), comme ceci :

```
$ php composer.phar update
```

Si jamais il y a un conflit avec d'autres plugins, on peut essayer de supprimer le fichier composer.lock afin de laisser composer libre de mettre à jour certains plugins. Dans ce cas, attention toutefois à relancer les tests (anti-régression) ensuite afin de s'assurer que tout fonctionne comme avant.

```
$ mv composer.lock composer.lock.ORIG
$ php composer.phar install
```

On devrait maintenant voir un sous-dossier **setasign/fpdf/** dans le dossier **vendor/**
De plus, un nouveau fichier **composer.lock** doit avoir été créé.

Si ça ne marche toujours pas, on peut aussi essayer de ré-installer tous les plugins. Pour cela, il suffit de taper "install" au lieu de "update". Cela re-créera entièrement le dossier vendor/. Pour en être bien sûr on peut même supprimer complètement ce dossier (rm -rf vendor/) :

```
$ php composer.phar install
```

17.3.6. Supprimer un plugin

$ php composer.phar **remove** vendor/package

ex: php composer.phar **remove setasign/fpdf**

Cela va supprimer le package (plugin) du dossier vendor/, et des fichiers composer.json et composer.lock...

(Il faudra aussi supprimer toute référence à ce plugin dans LabInvent).

18. Historique du logiciel

Année 2019

DATE VERSION	CHANGEMENTS VISIBLES	CHANGEMENTS NON VISIBLE (STRUCTURE INTERNE)
15/5/19 **Version ???**	Bugfix, simplification, et amélioration de la fonctionnalité “Copier ce matériel”	
15/04/19 **Version 2.12.4** **(EP)**		Mise à jour du fichier /**composer**.phar (php composer.phar self-update, avec php version 7) Warnings après installation: - Package aferrandini/phpqrcode is abandoned, you should avoid using it. Use endroid/qr-code instead. - Package phpunit/phpunit-mock-objects is abandoned, you should avoid using it. No replacement was suggested ~~- Installation du plugin **tcpdf** pour remplacer fpdf :~~ ~~$ php composer.phar require tecnickcom/tcpdf~~ ~~*Using version ^6.2 for tecnickcom/tcpdf*~~ ~~*./composer.json has been updated*~~ ~~*Loading composer repositories with package information*~~ ~~*Updating dependencies (including require-dev)*~~ ~~*Package operations: 1 install, 0 updates, 0 removals*~~ ~~*- Installing tecnickcom/tcpdf (6.2.26): Downloading (100%)*~~

02/04/19 version 2.12.3 (EP)	Recherche matériels sur numéro interne labo en plein texte Affichage du prix dans le résultat de recherche	
26/03/19 version 2.12 (EP)		Bugfix Gestionnaire de reference : maintenant bien sauvegardé (avant il était perdu...) => modif de la BD Gros cleanup du code LDAP dans LdapConnectionsTable.php Refactorisation des actions add() et edit() de MaterielsController en une seule fonction add_or_edit() car elles étaient très semblables
14/03/19 version 2.11.00 (EP)		Format étiquette configurable (pour que CRAL puisse faire sa version), et QrCode peut être imprimé dessus Doc du logiciel grandement mise à jour
26/02/19 version 2.10.21 (EP)		Simplification des interactions domaine<->categorie (materiel) : la selection de la categorie ne met plus à jour le domaine, ça ne sert à rien Bugfix "Categorie sélectionnée ne met plus à jour le domaine"
20/02/19 version 2.10.15-19 (EP)		Retour de la nouvelle version du **LDAP avec 2 modes** (**anonyme** pour IRAP, et **authentifié** pour le CRAL), refactorisé Ajout d'un lien "**What's New**" en bas de page web, qui pointe vers cette page "Historique du logiciel" LDAP refactorisation && optimisation : LdapConnectionsTable.php/getAllLdapUsers() remplacé par getAllLdapUsersNEW LDAP refactor progressif pour remettre le mode LDAP authentifié (pour CRAL)

15-20/02/19 version 2.10.7-14 (EP)		(**IRAP only**) **Gros bugfix du LDAP** : en mode LDAP anonyme, n'importe qui pouvait se connecter avec n'importe quel compte (il n'y avait pas de vérification du mot de passe !!!)
23/01/19 version 2.10.1-6 (EP)	**Recherche Matériels, Bugfixes et améliorations** : - Bugfix Recherche générale sur plusieurs mots : fait un AND de ces mots et non pas un OR !!! - Bugfix Recherche générale d'une date - **Ajout recherche générale d'un fournisseur et d'une catégorie** - Bugfix recherche sur un montant à virgule... - Complète refactorisation de MaterielsController::find(), y avait besoin ! - Ajout de tests - ...	**Mise à jour du framework cakephp** à la version courante : - passage de v3.5 à v3.7 - adaptations pour cakephp 3.7 - ajout du dossier vendor/ dans git (désormais versionné pour faciliter la mise à jour du framework cakephp) **Nouveaux fichiers** VERSION, UPDATE.sh, et mise à jour de TESTS.sh **LDAP authentifié bugfix**
15/01/2019 version 2.9.2 (EP)		**Nouveau script UPDATE** ("install/update.sh") pour mettre à jour automatiquement le logiciel (git pull) ET la BD (si besoin)
10/01/19 version 2.9.1.11 (EP)		**Nouveau mode LDAP AUTHENTIFIÉ** opérationnel (pour le CRAL): - renommé tous les champs et variables *_ldap en ldap_* pour meilleure lisibilité - adapté la section LDAP de la config - update fichier database/labinvent_last_version.sql contenant la BD COMPLETE - cleanup

---

Année 2018

21/12/2018 - version: 2.9.1.4 (EP)

Version et date affichés automatiquement et merge vues listes... :

- Version et date affichés automatiquement à partir du README.md
- Les vues "Voir les autres listes" et "Outils/Gérer le contenu variable..." sont désormais les mêmes

14/12/2018 - version: 2.9.1.0 (EP)

Ajout en cours du mode LDAP authentifié & Nouvelle documentation (dossier doc/)

ETE 2018 - version: 2.9.0 (Malik Imelhaine, stagiaire)

Nombreux petits bugfixes et petites améliorations (a priori sans changement structure BD)

02/02/2018 - version: 2.8.1 (EP)

Nombreux bugfixes (saisie dates, etiquette, doc admission...)

Année 2017

15/09/2017 Version: 2.8.0 (EP)

- Nouveau système ACL simplifié basé sur les tableaux $easyACL (en cours)
- Tests paramétrés
- Bugfix emails

30/08/2017 Version: 2.7.9 (EP)

- fonction intelligente AppController::getUserRole() qui donne le role "Utilisateur" par défaut pour un utilisateur non privilégié
- Refactorisation des ACL (authorizations) dans isAuthorized() et beforeFilter()

28/08/2017 Version: 2.7.6 (EP)

- renforcement important des TESTS : généralisation, refactorisation, simplification + numérotation systématique (cf doc ACL)
- nouvelle classe General dont héritent tous les tests
- nouvelle philo mise en place : 1 fichier tests par Controleur (c'était déjà le cas), puis pour un controleur donné, tri des tests par ACTION, puis pour chaque action, tests systématique de tous les ROLES (profils) avec les cas particuliers de chacun

06/07/2017 Version: 2.7.3 (EP)
- suppression du contenu de vendor/ => désormais ignoré car autogénéré
- update des plugins phpqrcode et fpdf => désormais via composer.json
- amélioration script install => pour générer automatiquement le contenu de vendor/

30/06/2017 Version: 2.7.0 (EP)
Passage de Cakephp3.2 à 3.4, Phpunit 5 à 6, et compatibilité avec Php5.6 et 7

12/05/2017 Version: 2.6.2 (Thibaud Ajas?)
L'ajout d'un gestionnaire du matériel est désormais obligatoire à l'ajout d'un matériel

09/05/2017 Version: 2.6.0 (Thibaud Ajas?)
Les noms des matériels sont en rouge dans la liste lorsque la date de garantie à été dépassée,
ainsi que la date elle-même dans leur fiche détaillée
Petits bugfixes divers

25/04/2017 Version: 2.5.6 (Thibaud Ajas?)
On peut maintenant configurer et choisir sur la page le nombre de materiels affichés (20 par défaut)
Petits bugfixes divers
Amélioration du script et de la doc d'installation

Année 2016

24/06/2016 Version: 2.4.7.2 (Version 2.4 finale)

Implémentation des ACL (droits) & Autres demandes
https://projects.irap.omp.eu/versions/107

30/05/2016 Version: 2.3.2.1 (Version 2.3 finale)
Implémentation du LDAP (vrai et fake)
https://projects.irap.omp.eu/versions/108

23/05/2016 Version: 2.2.5.4 (Version 2.2 finale)
Implémentation de toutes les autres actions
https://projects.irap.omp.eu/versions/106

12/05/2016 Version: 2.1.10 (Version 2.1 finale)
Implémentation complète du CRUD
https://projects.irap.omp.eu/versions/101

04/05/16 Version 2.0.8 (Version 2.0 finale)
Version de base (from bake) : php5 + cakephp3
https://projects.irap.omp.eu/versions/105

21/01/16 Version 1.3.636 (Version 1.3 finale)
php5 + cakephp2.1
https://projects.irap.omp.eu/versions/17

19. Roadmap (feuille de route) (TODO LIST)

\todo

La ROADMAP officielle est gérée sur le REDMINE du laboratoire IRAP.

⇒ Suivre ce lien pour la roadmap de la **version en cours (v2.13)**

19.1. TODO LIST temporaire à migrer sur le Redmine (notes urgentes)

Cette section contient des actions notées en urgence en attendant d'être ajoutées à la roadmap officielle (lien redmine ci-dessus).

(27/1/20) TODO au moins 1 fois sur chaque client git (CRAL, IRAP, …) :
$ cd labinvent/
$ rm -rf vendor/*
$ git pull
(maintenant, il faut encore modifier composer.json et faire un push)
$ ./UPDATE (=> va voir que composer.json a changé et donc provoquer une install des plugins)

(Ajouts de Etienne, fin janvier 2020) :

- **TODO: installer proprement la v3.7 (et même passer à 3.8) :**
  - **install : https://book.cakephp.org/3/en/installation.html**
  - **Config : https://book.cakephp.org/3/en/development/configuration.html**
  - **Application : (pas encore activé pour l'instant, ça craint) ajouter src/Application.php :** https://book.cakephp.org/3/en/development/application.html
  - … (voir section **Getting Started** de la doc cakephp3)
- migrer wiki pages dev et techniques dans ce gdoc

- bugfix fakeldap : table fakeldapusers à gérer
- superadmin login marche pas
- supprimer double login superadmin (1 créé par défaut, 2eme créé avec les logins bidons proposés à l'install)
- _fake_ldap_user_ : on garde ? que devient-il ?
- que deviennent les logins bidons après passage à LDAP ?
- Outils/Page/Etiqueteuse, et ACL : lien vers gdoc et non pas wiki
- "Gérer les users ~~privilégiés~~"
- vendor/ : sortir de git !!!
- https://github.com/abedputra/Inventory-Stock-With-Scanner
- https://discourse.cakephp.org/t/how-to-integrate-html-bootstrap-template-in-cakephp-3-5/3635
- mail général envoyé à responsables, admin, users, … : paramétrer les actions pour lesquelles on envoie des mails, et à qui

(suite à réunion avec CRAL du 15-16/01/20 :

SUIVI:
- rappel automatique par mail qq jours avant : user, resp, et superadmin
- bug : impossible modifier la périodicité (6 mois), même superadmin
- bug : la périodicité n'est pas affichée sur la fiche détail
- groupe thématique par défaut = qualité, possible ?
- afficher la date de la prochaine interv (fonction de la périodicité)

- valider l'intervention périodique si elle a été faite (garder l'historique ?)

- type interv = Réparation : ne pas afficher le champ « fréquence »…

19.2. ANCIENNE ROADMAP

Cette roadmap a déjà été transférée sur le Redmine officiel le 7/6/19.
Elle est gardée ici juste pour info.

⇒ Accès direct aux thèmes:

- GENERAL
- USERS
- ETIQUETTES
- MATERIELS
- SUIVIS
- EMPRUNTS

migré dans redmine	THEME	ITEM	FAIT
		LÉGENDE: - ***Urgence**: Très Urgent, Assez Urgent, par défaut = peut attendre* - ***Type**: (**B**) = Bug ou anomalie ; (**I**) = Improvement (amélioration) ; (**N**) = Nouvelle fonctionnalité ; (**M**) = Modification (changement)*	
	GENERAL		
X	**Version mobile**	**??? (oui, mais par un stagiaire)**	
X	**Database**	**(I) Changer toutes les contraintes** pour remplacer “on delete no action” par “**on delete SET NULL**”. (ou CASCADE selon les cas) Déjà fait pour materiels.fk_materiels_gestionnaire_id, en 2 temps : 1) supprimer contrainte actuelle ALTER TABLE materiels DROP FOREIGN KEY fk_materiels_gestionnaire_id;	

		2) Recréer contrainte ALTER TABLE materiels ADD CONSTRAINT fk_materiels_gestionnaire_id FOREIGN KEY (gestionnaire_id) REFERENCES users (id) ON DELETE SET NULL ON UPDATE NO ACTION;	
	CONFIG	~~**(M) Passer seuil de 800€ à 1000€**~~	26/03/19 (EP)
X	**Website**	**(I) (13/3/19) Fichiers logo ne doivent pas être versionnés (git)** Il s'agit des 2 fichiers logo_entity.jpg et logo_software.jpg Ils ne doivent pas être versionnés directement depuis le **dossier "destination"** webroot/img/ Ils doivent plutôt être copiés lors de la phase d'installation, depuis un **dossier "source"** (versionné) vers le dossier "destination" (non versionné) Cela afin d'éviter que le labo qui change ces logos les change pour tout le monde !!!	
X	**Plugins update**	**(I) (31/1/19) Changer le plugin de création des QrCodes :** *"Package **aferrandini/phpqrcode is abandoned**, you should avoid using it.* *Use **endroid/qr-code** instead."* **(I) (15/4/19) Installer mpdf à la place de fpdf** (qui n'a pas de mise à jour depuis 2015 !!!) https://mpdf.github.io *https://mpdf.github.io/installation-setup/installation-v7-x.html*	
	USERS **(Gestion des utilisateurs)**		
inutile ?	**DIVERS**	**(I) (12/2/19) Plugin Users** pour mieux gérer les users ○ https://github.com/CakeDC/users ○ Exemple app: https://github.com/CakeDC/users-example	

		~~(B) (25/4/19)~~ ~~**ajouter un utilisateur avec privilège**~~ ~~Dans le menu, on sélectionne la personne dans la liste fournie par LDAP, la champ "mail" se remplit automatiquement.~~ ~~Le "login" est grisé.~~ ~~Quand on valide => message d'erreur (le champ login reste inaccessible)~~ ~~**=> voir mail**~~ ~~Du coup j'ai rentré son utilisateur à la main dans la base mais ça n'est pas une solution.~~	26/4/19
X	**EMAIL**	**(B) (19/4/19) Bugfix mail reçu par l'"utilisateur" d'un matériel :** Jean-louis Lefort a ajouté le matériel "Optiplex 7060" (id=12111). Veuillez vérifier et compléter si besoin la fiche correspondante. ~~Vous recevez ce message car vous etes dans la liste spécifique des emails de LabInvent.~~ ~~Pour faire retirer votre mail de cette liste, veuillez contacter un SuperAdmin.~~ **(I) (1/2/19) Améliorer email sur materiel/add :** - **ajouter un lien direct vers le matériel (plutôt que son id qui ne sert à rien)** - **changer "contacter un SuperAdmin" en "contacter un administrateur du logiciel (email1, email2...)** *"Etienne Pallier a ajouté le matériel 'new201901' (id=11964).* *Veuillez vérifier et compléter si besoin la fiche corespondante.* *Vous recevez ce message car vous etes dans la liste spécifique des emails de LabInvent.* *Pour faire retirer votre mail de cette liste, veuillez contacter un SuperAdmin."* **(I) (2/4/19) Améliorer email sur materiel/delete !!! :** *Marjorie Cloup a supprimé le matériel "LATT - Software For Sidecar ASIC Development Kit" (id=10421).* ***Domaine : 2*** ***Catégorie : 19*** *Vous recevez ce message car vous etes dans la liste spécifique des emails de LabInvent.* *Pour faire retirer votre mail de cette liste, veuillez contacter un SuperAdmin.*	
X (fakelda pusers)	**LDAP**	~~**(I) (EP 4/2/19) Optimisation : optimiser la recherche LDAP**~~ ~~Pb : Eviter de rechercher plusieurs fois les utilisateurs, ça peut prendre bcp de temps à chaque fois… (surtout au CRAL).~~	6/6/19

		~~**Solution** : utiliser la table Utilisateurs de la BD comme CACHE du LDAP ; à la première recherche ldap, mettre dans la table Utilisateurs de la BD les utilisateurs qui n'y sont pas encore (s'ils sont "non privilégié", les mettre avec un profil "utilisateur") ainsi que tous leurs attributs nécessaires au traitement (nom, pnom, email, ..., mais aussi ldap PASSWORD) ; Mais, nouveau pb : quand on modifie ou supprime un utilisateur du ldap, il faut penser à le modifier aussi dans cette table... le risque d'incohérences est fort avec le temps… => optimisation: avec une date de création, comme ça au bout de 3 mois par exemple, on peut rechercher à nouveau dans le ldap et mettre à jour éventuellement (sinon, les utilisateurs modifiés ou supprimés dans le ldap ne seront pas mis à jour dans la BD… => incohérence) Plus de détails encore à la fin de ce tableau (cf (*))~~	
	ETIQUETTES		
X	**Etiqueteuse**	**(B) Imprimantes ne fonctionnent plus sur Win10, why ?...** - ok sur Mac - ok sur PC win10 Etienne - pas ok sur pc win10 Elodie - pas ok sur pc win10 des gestionnaires	
	MATERIELS		
X	**RECHERCHE**	~~**(I) (28/3/19) Recherche numéro IRAP plein texte**~~ ~~Ex: retrouver les materiels de + de 10.000 E .~~ ~~Dans Labinvent, j'aimerais pouvoir chercher sur une partie du Num Irap et la, j'ai une liste deroulante. Pas top. C'est possible de remettre un champ texte ?~~ ~~**(I) (28/3/19) avoir le prix sur la liste des materiels suite a une recherche**~~ **(I) (12/2/19) Search Plugin :** https://github.com/CakeDC/search	2/4/19 2/4/19

X	**ADD (création) et EDIT (modification) et ADMINISTRER**	~~**(B) (25/4/19) Copie de matériel (cf Yann Parot)**~~ ~~Certains champs ne sont pas modifiables…~~ ~~**(I) (5/2/19) Optimisation : refactoriser les vues add et edit en UNE SEULE VUE**~~ ~~Car elles se ressemblent énormément ! Eviter cette redondance !~~ **(B) Après erreur de saisie, on perd la liste des categ et sous-categ !!!** **(B) Vue "administrer" perdue ?** Pour la voir, taper "administrer" à la place de "view" dans une URL Si cette fonction est toujours utile, il faudrait factoriser cette action avec add_or_edit() de MaterielsController **(B) (29/1/19) Edition de la section "Partie administrative" d'un matériel en profil superadmin (et autres profils ?)** Cette section devrait être masquée par défaut et ouverte quand on clique dessus, mais ça ne marche plus **(B) (29/1/19) Ajouter champ manquant nom_ancien_responsable dans table materiels** Puis activer le code commenté de la fonction beforeSave() de MaterielsTable.php pour garder trace du responsable précédent ou d'un responsable parti qui n'est plus dans le ldap ~~**(B) (31/1/19) Bugfix** SurCategoriesController::getFromCategorie()~~ ~~○ Request URL: /**SurCategories/getFromCategorie**~~ ~~○ Referer URL: http://labinvent.test/materiels/add~~ ~~○ Client IP: 127.0.0.1~~ ~~○ => reprogrammer différemment : obliger à choisir dans l'ordre domaine puis categ puis sous-categ :~~ ~~■ **NEW** si domaine=null => mettre categ à null aussi~~ ~~■ CHECK si categ null => mettre souscateg à null aussi~~ ~~■ si domaine selected => changer liste categ~~	25/3/19 (EP) 20/3/19 (EP)

		~~■ **NEW** si categ choisie => changer liste sous-categ, mais ne pas toucher à domaine !~~	
X	**index (liste matos)**	**(I) (21/5/19) Demandes Yann Parot :** · **Serait-il possible dans la liste de matériel d'afficher le numéro de série ?** *possible, mais ya déjà bcp de trucs, ça risque de pas tenir sur des petits écrans pc portables (12 à 14") et le num série est rarement entré, mais s'il était entré plus souvent oui ça serait bien utile...* · **Serait-il possible dans la liste de matériel d'afficher si il est emprunté ou non ?** *Là aussi, très utile mais trop rarement utilisé..., mais bon, à la limite c'est juste une case à cocher de plus (en mettant pour titre de colonne juste "E")* *Ou alors, mettre toute la ligne du matériel concerné en couleur (orange ?)* *On pourrait aussi se contenter de le préciser quand on va sur la fiche détaillée du matériel (style: "Attention, ce matériel est emprunté")* *Au même titre, il faudrait aussi signaler un suivi ou une réparation en cours (style: "Attention, ce matériel est en cours d'opération de maintenance")* · **Serait-il possible dans le cas de création multiple de pouvoir lui faire avaler un fichier excel par exemple ?** *Oui dans la version 145..., ça c'est plutôt du rêve, mais je le note quand même dans la TODO list, on sait jamais si on trouve un super stagiaire...*	
X	**Documents**	~~**(B) (15/4/19) Bugfix Doc admission:**~~ - ~~Plantage quand on clique sur le bouton "Doc. admission" d'un matériel pour lequel on n'a pas précisé l'organisme ou/et le fournisseur~~ - Si l'organisme ou le fournisseur n'est pas precisé, on ne doit pas pouvoir obtenir un doc admission. Mais la fiche materiel oui	15/4/19

		(I) (21/5/19, CRAL) Adapter les logos des documents pdf (notamment les admission*.ctp) : Il faudra que j'adapte en passant la liste des organismes en paramètre, via le tableau $data de la fonction admission() du controleur src/Controller/DocumentsController.php Ce tableau $data est lu dans admission_pdf.ctp et il suffira alors d'adapter le code de la ligne 30 et suivantes. Il faut aussi que tout ça soit réglable via le menu de configuration... (upload des logos)	
	SUIVIS		
	...		
	EMPRUNTS		
	...		

⇒ Retour au début du tableau

(*) Détail sur l'action "Optimisation du LDAP" ci-dessus :

Où se trouve le code qui affiche la liste des utilisateurs dans les vues ajout/édition de matériel ?
Il suffit de suivre cette logique :

- Ajout de materiels = src/Template/Materiels/add.ctp
- Edition de materiels = src/Template/Materiels/edit.ctp

Dans la vue "add.ctp" (et "edit.ctp") tu trouves le champ "Nom de l\'utilisateur"
Il contient une variable 'options' => $utilisateurs
C'est donc cette variable $utilisateurs qui est utilisée.
Là ca se complique un peu :
Cette variable n'est pas définie dans la vue, mais dans le controlleur de matériels qui l'a créée et passée à la vue :

Tu cherches donc "$utilisateurs" dans src/Controller/MaterielsController.php, on le trouve à la ligne 811 (elle est ensuite passée à la vue à la ligne 830 avec set(..., 'utilisateurs', ...))
Elle vient de "$users" qui est définit juste au-dessus, ligne 807 :
$users = TableRegistry::get('LdapConnections')->getListUsers();
getListUsers() est définie dans src/Model/Table/LdapConnectionsTable.php

Si je peux modifier ce code et entrer les personnes à la main ça pourrait tourner en attendant une vraie correction?
oui, ça pourrait suffire, mais non, car une fois que tu choisis un utilisateur dans la liste, son email est cherché via le ldap pour l'afficher dans le champ suivant nommé "Email de l'utilisateur"
En plus, c'est un code javascript (ajax) qui fait ça à la fin du fichier add.ctp :

```
$("#nom-responsable").bind("change", function(event) {
	var url = document.URL;
	var reg = new RegExp("(materiels).*$", "g");
	var emailUrl = url.replace(reg, "Users/getLdapEmail/");
	$.ajax({
		url: emailUrl + $("#nom-responsable").val()
	}).done(function(data) {
		$("#email-responsable").val(data)
	});
});
```

Mais tu peux feinter, car ce code appelle la fonction getLdapEmail() de src/Controller/UsersController.php
Il te suffit donc de modifier aussi cette fonction... (ou alors modifier le code javascript)
Ca n'est pas bien optimal, mais comme chez nous ça prend 1 seconde, pas besoin de mieux...

ALGO**:**

- De manière générale, utiliser la DB comme un “LDAP CACHE” :
  - remplacer toutes les recherches ldap par : search DB 1st then “search ldap + update DB”

Cas possibles:

- **un user se logue** => search ldap pour valider son pass
- **matos/add/edit** => search ldap pour liste des users
- matos/index or view => ouf, **pas de recherche dans le ldap** car le nom et email sont écrits en dur dans la table matériels

19.3. VERY OLD TODO LIST A VIRER BIENTOT (en italique ce qui a déjà été migré dans le tableau “TODO LIST” ci-dessus)

- ***(EP 26/3/19)***
  - ***Recherche numéro IRAP plein texte***
    - *ex: retrouver les materiels de + de 10.000 E .*
    - *Dans Labinvent, j'aimerai pouvoir chercher sur une partie du Num Irap et la, j'ai une liste deroulante. Pas top.*
    - *C'est possible de remettre un champ texte ?*
  - ***Vue “administrer” perdue ?***
    - *pour la voir taper “administrer” à la place de “view” dans une URL*
    - *si cette fonction est toujours utile, il faudrait factoriser cette action avec add_or_edit() de MaterielsController*
  - ***Changer toutes les contraintes** pour remplacer “on delete no action” par “**on delete SET NULL**”.*

Déjà fait pour materiels.fk_materiels_gestionnaire_id, en 2 temps :

3) supprimer contrainte actuelle

```
ALTER TABLE materiels
    DROP FOREIGN KEY fk_materiels_gestionnaire_id;
```

4) Recréer contrainte

```
ALTER TABLE materiels
    ADD CONSTRAINT fk_materiels_gestionnaire_id
    FOREIGN KEY (gestionnaire_id)
    REFERENCES users (id)
    ON DELETE SET NULL ON UPDATE NO ACTION;
```

  - ~~*Passer seuil de 800€ à 1000€*~~
  - *Imprimantes ne fonctionnent plus sur Win10, why ?...*
  - *(matos add) Après erreur de saisie, on perd la liste des categ et sous-categ !!!*

- ***(EP 13/3/19) Fichiers logo ne doivent pas être versionnés (git)***

- *Il s'agit des 2 fichiers logo_entity.jpg et logo_software.jpg*
- *Ils ne doivent pas être versionnés directement depuis le **dossier "destination"** webroot/img/*
- *Ils doivent plutôt être copiés lors de la phase d'installation, depuis un **dossier "source"** (versionné) vers le dossier "destination" (non versionné)*
- *Cela afin d'éviter que le labo qui change ces logos les change pour tout le monde !!!*

- ***(EP 12/2/19) Plugin Users** pour mieux gérer les users*
  - *https://github.com/CakeDC/users*
  - *Exemple app: https://github.com/CakeDC/users-example*

- ***(EP 12/2/19) Search Plugin :** https://github.com/CakeDC/search*

- ***(EP 5/2/19) Optimisation : refactoriser les vues add et edit en UNE SEULE VUE, car elles se ressemblent énormément ! Eviter cette redondance facheuse !***

- ***(EP 4/2/19) Optimisation : optimiser la recherche LDAP***

Pb : Eviter de rechercher plusieurs fois les utilisateurs, ça peut prendre bcp de temps à chaque fois… (surtout au CRAL).
Solution : utiliser la table Utilisateurs de la BD comme CACHE du LDAP ; à la première recherche ldap, mettre dans la table Utilisateurs de la BD les utilisateurs qui n'y sont pas encore (s'ils sont "non privilégié", les mettre avec un profil "utilisateur") ainsi que tous leurs attributs nécessaires au traitement (nom, pnom, email, ..., mais aussi ldap PASSWORD) ;
Mais, nouveau pb : quand on modifie ou supprime un utilisateur du ldap, il faut penser à le modifier aussi dans cette table... le risque d'incohérences est fort avec le temps…
=> optimisation: avec une date de création, comme ça au bout de 3 mois par exemple, on peut rechercher à nouveau dans le ldap et mettre à jour éventuellement (sinon, les utilisateurs modifiés ou supprimés dans le ldap ne seront pas mis à jour dans la BD… => incohérence)

Où se trouve le code qui affiche la liste des utilisateurs dans les vues ajout/édition de matériel ?
Il suffit de suivre cette logique :
- *Ajout de materiels = src/Template/Materiels/add.ctp*
- *Edition de materiels = src/Template/Materiels/edit.ctp*

Dans la vue "add.ctp" (et “edit.ctp”) tu trouves le champ "Nom de l\'utilisateur"
Il contient une variable 'options' => $utilisateurs
C'est donc cette variable $utilisateurs qui est utilisée.
Là ca se complique un peu :
Cette variable n'est pas définie dans la vue, mais dans le controlleur de matériels qui l'a créée et passée à la vue :
Tu cherches donc "$utilisateurs" dans src/Controller/MaterielsController.php, on le trouve à la ligne 811 (elle est ensuite passée à la vue à la ligne 830 avec set(..., 'utilisateurs', ...))
Elle vient de "$users" qui est définit juste au-dessus, ligne 807 :
$users = TableRegistry::get('LdapConnections')->getListUsers();
getListUsers() est définie dans src/Model/Table/LdapConnectionsTable.php

Si je peux modifier ce code et entrer les personnes à la main ça pourrait tourner en attendant une vraie correction?
oui, ça pourrait suffire, mais non, car une fois que tu choisis un utilisateur dans la liste, son email est cherché via le ldap pour l'afficher dans le champ suivant nommé "Email de l'utilisateur"
En plus, c'est un code javascript (ajax) qui fait ça à la fin du fichier add.ctp :

```
$("#nom-responsable").bind("change", function(event) {
        var url = document.URL;
        var reg = new RegExp("(materiels).*$", "g");
        var emailUrl = url.replace(reg, "Users/getLdapEmail/");
        $.ajax({
                url: emailUrl + $("#nom-responsable").val()
        }).done(function(data) {
                $("#email-responsable").val(data)
        });
});
```

Mais tu peux feinter, car ce code appelle la fonction getLdapEmail() de src/Controller/UsersController.php
Il te suffit donc de modifier aussi cette fonction... (ou alors modifier le code javascript)
Ca n'est pas bien optimal, mais comme chez nous ça prend 1 seconde, pas besoin de mieux...

ALGO:

- *De manière générale, utiliser la DB comme un "LDAP CACHE" :*
  - *remplacer toutes les recherches ldap par : search DB 1st then "search ldap + update DB"*

Cas possibles:

- ***un user se logue** => search ldap pour valider son pass*
- ***matos/add/edit** => search ldap pour liste des users*
- *matos/index or view => ouf, **pas de recherche dans le ldap** car le nom et email sont écrits en dur dans la table matériels*

- ***(EP 1/2/19) Améliorer email sur materiel/add => ajouter un lien direct vers le matériel (plutôt que son id qui ne sert à rien) :***
  - *"Etienne Pallier a ajouté le matériel 'new201901' (id=11964).*
    Veuillez vérifier et compléter si besoin la fiche corespondante.
    Vous recevez ce message car vous etes dans la liste spécifique des emails de LabInvent. Pour faire retirer votre mail de cette liste, veuillez contacter un SuperAdmin."

- ~~***(EP 31/1/19) Bugfix** SurCategoriesController::getFromCategorie()~~
  - ~~*Request URL: /**SurCategories/getFromCategorie***~~
  - ~~*Referer URL: http://labinvent.test/materiels/add*~~
  - ~~*Client IP: 127.0.0.1*~~
  - ~~*=> reprogrammer différemment : obliger à choisir dans l'ordre domaine puis categ puis sous-categ :*~~
    - ~~***NEW** si domaine=null => mettre categ à null aussi*~~
    - ~~*CHECK si categ null => mettre souscateg à null aussi*~~
    - ~~*si domaine selected => changer liste categ*~~
    - ~~***NEW** si categ choisie => changer liste sous-categ, mais ne pas toucher à domaine !*~~

- ***(EP 31/1/19) Changer le plugin de création des QrCodes :***
  - *"Package aferrandini/phpqrcode is abandoned, you should avoid using it. Use endroid/qr-code instead."*

- ***(EP 29/1/19) Bugfix Ajouter champ manquant nom_ancien_responsable dans table materiels***
  - *puis activer le code commenté de la fonction beforeSave() de MaterielsTable.php pour garder trace du responsable précédent ou d'un responsable parti qui n'est plus dans le ldap*

- ***(EP 29/1/19) Bugfix Edition de la section "Partie administrative" d'un matériel en profil superadmin (et autres profils ?)***
    - *Cette section devrait être masquée par défaut et ouverte quand on clique dessus, mais ça ne marche plus*

20. Annexes

20.1. Installation des pré-requis sur Mac OS X (avec HomeBrew)

Cette installation a été faite sur la version Mojave (10.14.2)

20.1.1. Cleanup BREW

Pour plus de détail :
https://getgrav.org/blog/macos-mojave-apache-multiple-php-versions

$ brew uninstall --force php56 php56-apcu php56-opcache php56-xdebug php56-yaml php56-intl
Ignoring bigdecimal-1.3.4 because its extensions are not built. Try: gem pristine bigdecimal --version 1.3.4
Uninstalling php56-intl... (7 files, 895.5KB)

$ brew uninstall --force php72 php72-apcu php72-opcache php72-xdebug php72-yaml php72-intl
Uninstalling php... (522 files, 76.4MB)
Uninstalling php72-xdebug... (12 files, 679.3KB)
Uninstalling php72-intl... (7 files, 474.5KB)

$ brew cleanup
Pruned 1 symbolic links from /usr/local

$ brew list | grep php
phpmyadmin

$ brew uninstall --force phpmyadmin

```
Uninstalling phpmyadmin... (2,434 files, 32.3MB)

$ brew list | grep php
RIEN


$ rm -Rf /usr/local/etc/php/*

$ brew --version
Homebrew 1.9.2
Homebrew/homebrew-core (git revision d4872; last commit 2019-01-21)

$ brew doctor
Your system is ready to brew.

$ brew update
$ brew upgrade

$ brew install openldap libiconv
Warning: openldap 2.4.47 is already installed and up-to-date
To reinstall 2.4.47, run `brew reinstall openldap`
==> Downloading https://homebrew.bintray.com/bottles/libiconv-1.15.mojave.bottle.tar.gz
######################################################################## 100.0%
==> Pouring libiconv-1.15.mojave.bottle.tar.gz
==> Caveats
libiconv is keg-only, which means it was not symlinked into /usr/local,
because macOS already provides this software and installing another version in
parallel can cause all kinds of trouble.

If you need to have libiconv first in your PATH run:
  echo 'export PATH="/usr/local/opt/libiconv/bin:$PATH"' >> ~/.bash_profile

For compilers to find libiconv you may need to set:
  export LDFLAGS="-L/usr/local/opt/libiconv/lib"
  export CPPFLAGS="-I/usr/local/opt/libiconv/include"
```

==> Summary
🍺 /usr/local/Cellar/libiconv/1.15: 30 files, 2.4MB

20.1.2. APACHE

****
1.1 - Apache INSTALL
****

The latest macOS 10.14 Mojave comes with Apache 2.4 pre-installed, however, it is no longer a simple task to use this version with Homebrew because Apple has removed some required scripts in this release. However, the solution is to install Apache 2.4 via Homebrew and then configure it to run on the standard ports (80/443).
If you already have the built-in Apache running, it will need to be shutdown first, and any auto-loading scripts removed

$ sudo apachectl stop
Password:
httpd: Syntax error on line 191 of /usr/local/etc/httpd/httpd.conf: Cannot load /usr/local/lib/libphp5.so into server: dlopen(/usr/local/lib/libphp5.so, 10): image not found

==> J'ai Commenté ces lignes dans httpd.conf :
#LoadModule php5_module /usr/local/lib/libphp5.so
#LoadModule php7_module /usr/local/lib/libphp7.so

$ sudo apachectl stop
(2)No such file or directory: AH02291: Cannot access directory '/usr/local/opt/httpd/logs/' for error log of vhost defined at /usr/local/etc/httpd/extra/httpd-vhosts.conf:56
AH00014: Configuration check failed

==> J'ai Commenté ces lignes dans httpd-vhosts.conf :

```
    ####ErrorLog logs/labinvent2.error_log
    ####CustomLog logs/labinvent2.access_log combined



$ sudo apachectl stop
httpd (pid 6600?) not running


$ sudo launchctl unload -w /System/Library/LaunchDaemons/org.apache.httpd.plist 2>/dev/null


Now we need to install the new version provided by Brew:

$ brew install httpd
Warning: httpd 2.4.37_1 is already installed and up-to-date
To reinstall 2.4.37_1, run `brew reinstall httpd`

$ brew reinstall httpd
==> Reinstalling httpd
==> Downloading https://homebrew.bintray.com/bottles/httpd-2.4.37_1.mojave.bottle.tar.gz
Already downloaded:
/Users/epallier/Library/Caches/Homebrew/downloads/4459c998a0822638fecfe41352bbfd0755d17a65e188bff0b3adbffc61dd5e4c--httpd-2.4.37_1.moj
ave.bottle.tar.gz
==> Pouring httpd-2.4.37_1.mojave.bottle.tar.gz
==> Caveats
DocumentRoot is /usr/local/var/www.

The default ports have been set in /usr/local/etc/httpd/httpd.conf to 8080 and in
/usr/local/etc/httpd/extra/httpd-ssl.conf to 8443 so that httpd can run without sudo.

To have launchd start httpd now and restart at login:
  brew services start httpd
Or, if you don't want/need a background service you can just run:
  apachectl start
```

==> Summary
🍺 /usr/local/Cellar/httpd/2.4.37_1: 1,648 files, 26.9MB

$ sudo brew services start httpd

To ensure the server is up:
$ ps -aef | grep httpd

```
    0 91454     1   0 10:59   ??         0:00.07 /usr/local/opt/httpd/bin/httpd -D FOREGROUND
   70 91455 91454   0 10:59   ??         0:00.00 /usr/local/opt/httpd/bin/httpd -D FOREGROUND
   70 91456 91454   0 10:59   ??         0:00.00 /usr/local/opt/httpd/bin/httpd -D FOREGROUND
   70 91457 91454   0 10:59   ??         0:00.00 /usr/local/opt/httpd/bin/httpd -D FOREGROUND
   70 91458 91454   0 10:59   ??         0:00.00 /usr/local/opt/httpd/bin/httpd -D FOREGROUND
   70 91459 91454   0 10:59   ??         0:00.00 /usr/local/opt/httpd/bin/httpd -D FOREGROUND
   70 91460 91454   0 10:59   ??         0:00.00 /usr/local/opt/httpd/bin/httpd -D FOREGROUND
  501 91463   453   0 11:01   ttys002    0:00.00 grep httpd
```

=> se connecter à localhost[:8080]

Try to restart Apache with:
$ sudo apachectl -k restart

You can watch the Apache error log in a new Terminal tab/window during a restart to see if anything is invalid or causing a problem:
$ tail -f /usr/local/var/log/httpd/error_log
Apache is controlled via the apachectl command so some useful commands to use are:
$ sudo apachectl start
$ sudo apachectl stop
$ sudo apachectl -k restart
The -k will force a restart immediately rather than asking politely to restart when apache is good and ready

****
1.2 - Apache CONFIG
****

In the latest version of Brew, you have to manually set the listen port from the default of 8080 to 80, so we will need to edit Apache's configuration file.
vi /usr/local/etc/httpd/httpd.conf

Change « Listen 8080 » to « Listen 80 »

(pas fait:)
Next we'll configure it to use the to change the document root for Apache. This is the folder where Apache looks to serve file from. By default, the document root is configured as /usr/local/var/www. As this is a development machine, let's assume we want to change the document root to point to a folder in our own home directory.
Search for the term DocumentRoot, and you should see the following line:
DocumentRoot "/usr/local/var/www"
Change this to point to your user directory where your_user is the name of your user account:
DocumentRoot /Users/your_user/Sites
You also need to change the <Directory> tag reference right below the DocumentRoot line. This should also be changed to point to your new document root also:
<Directory /Users/your_user/Sites>

In that same <Directory> block you will find an AllowOverride setting, this should be changed as follows:
AllowOverride controls what directives may be placed in .htaccess files.
It can be "All", "None", or any combination of the keywords:
AllowOverride FileInfo AuthConfig Limit
#
AllowOverride All
Also we should now enable mod_rewrite which is commented out by default. Search for mod_rewrite.so and uncomment the line by removing the leading #:
LoadModule rewrite_module lib/httpd/modules/mod_rewrite.so

User & Group (pas fait)
Now we have the Apache configuration pointing to a Sites folder in our home directory. One problem still exists, however. By default, apache runs as the user daemon and group daemon. This will cause permission problems when trying to access files in our home directory. About a third of the way

down the httpd.conf file there are two settings to set the User and Group Apache will run under. Change these to match your user account (replace your_user with your real username), with a group of staff:
User your_user
Group staff
(je l'ai pas fait, j'ai laissé comme c'est:
User _www
Group _www
)

Servername
Apache likes to have a server name in the configuration, but this is disabled by default, so search for:
#ServerName www.example.com:8080
and replace it with:
ServerName localhost

Sites Folder (pas fait)
Now, you need to create a Sites folder in the root of your home directory. You can do this in your terminal, or in Finder. In this new Sites folder create a simple index.html and put some dummy content in it like: <h1>My User Web Root</h1>.
$ mkdir ~/Sites
$ echo "<h1>My User Web Root</h1>" > ~/Sites/index.html

$ sudo apachectl -k restart

20.1.3. PHP

****
2.1 - Php INSTALL
****

Up until the end of March 2018, all PHP related brews were handled by Homebrew/php tab, but that has been deprecated, so now we use what's available in the Homebrew/core package. This should be a better maintained, but is a much less complete, set of packages.

Both PHP 5.6 and PHP 7.0 has been deprecated and removed from Brew because they are out of support, and while it's not recommended for production, there are legitimate reasons to test these unsupported versions in a development environment.
Remember only PHP 7.1 through 7.3 are officially supported by Brew so if you want to install PHP 5.6 or PHP 7.0 you will need to add this tap:
$ brew tap exolnet/homebrew-deprecated

Both PHP 5.6 and PHP 7.0 has been deprecated and removed from Brew because they are out of support, and while it's not recommended for production, there are legitimate reasons to test these unsupported versions in a development environment.
Remember only PHP 7.1 through 7.3 are officially supported by Brew so if you want to install PHP 5.6 or PHP 7.0 you will need to add this tap:
$ brew tap exolnet/homebrew-deprecated
==> Tapping exolnet/deprecated
Cloning into '/usr/local/Homebrew/Library/Taps/exolnet/homebrew-deprecated'...
remote: Enumerating objects: 11, done.
remote: Counting objects: 100% (11/11), done.
remote: Compressing objects: 100% (11/11), done.
remote: Total 11 (delta 1), reused 7 (delta 0), pack-reused 0
Unpacking objects: 100% (11/11), done.
Tapped 2 formulae (42 files, 78.3KB).

We will proceed by installing various versions of PHP and using a simple script to switch between them as we need. Feel free to exclude any versions you don't want to install.
$ brew install php@5.6
(pas fait)$ brew install php@7.0
$ brew install php@7.1
$ brew install php@7.2
$ brew install php@7.3

COOL : You no longer have to unlink each version between installing PHP versions as they are not linked by default

$ brew install php@5.6
==> Installing php@5.6 from exolnet/deprecated

```
==> Installing dependencies for exolnet/deprecated/php@5.6: mhash and mcrypt
==> Installing exolnet/deprecated/php@5.6 dependency: mhash
==> Downloading https://homebrew.bintray.com/bottles/mhash-0.9.9.9.mojave.bottle.tar.gz
######################################################################## 100.0%
==> Pouring mhash-0.9.9.9.mojave.bottle.tar.gz
🍺  /usr/local/Cellar/mhash/0.9.9.9: 20 files, 455.4KB
==> Installing exolnet/deprecated/php@5.6 dependency: mcrypt
==> Downloading https://homebrew.bintray.com/bottles/mcrypt-2.6.8.mojave.bottle.1.tar.gz
######################################################################## 100.0%
==> Pouring mcrypt-2.6.8.mojave.bottle.1.tar.gz
🍺  /usr/local/Cellar/mcrypt/2.6.8: 19 files, 397.8KB
==> Installing exolnet/deprecated/php@5.6
==> Downloading https://dl.bintray.com/exolnet/bottles-deprecated/php@5.6-5.6.40.mojave.bottle.tar.gz
######################################################################## 100.0%
==> Pouring php@5.6-5.6.40.mojave.bottle.tar.gz
==> /usr/local/Cellar/php@5.6/5.6.40/bin/pear config-set php_ini /usr/local/etc/php/5.6/php.ini system
==> /usr/local/Cellar/php@5.6/5.6.40/bin/pear config-set php_dir /usr/local/share/pear@5.6 system
==> /usr/local/Cellar/php@5.6/5.6.40/bin/pear config-set doc_dir /usr/local/share/pear@5.6/doc system
==> /usr/local/Cellar/php@5.6/5.6.40/bin/pear config-set ext_dir /usr/local/lib/php/pecl/20131226 system
==> /usr/local/Cellar/php@5.6/5.6.40/bin/pear config-set bin_dir /usr/local/opt/php@5.6/bin system
==> /usr/local/Cellar/php@5.6/5.6.40/bin/pear config-set data_dir /usr/local/share/pear@5.6/data system
==> /usr/local/Cellar/php@5.6/5.6.40/bin/pear config-set cfg_dir /usr/local/share/pear@5.6/cfg system
==> /usr/local/Cellar/php@5.6/5.6.40/bin/pear config-set www_dir /usr/local/share/pear@5.6/htdocs system
==> /usr/local/Cellar/php@5.6/5.6.40/bin/pear config-set man_dir /usr/local/share/man system
==> /usr/local/Cellar/php@5.6/5.6.40/bin/pear config-set test_dir /usr/local/share/pear@5.6/test system
==> /usr/local/Cellar/php@5.6/5.6.40/bin/pear config-set php_bin /usr/local/opt/php@5.6/bin/php system
==> /usr/local/Cellar/php@5.6/5.6.40/bin/pear update-channels
==> Caveats
To enable PHP in Apache add the following to httpd.conf and restart Apache:
    LoadModule php5_module /usr/local/opt/php@5.6/lib/httpd/modules/libphp5.so

    <FilesMatch \.php$>
    SetHandler application/x-httpd-php
    </FilesMatch>
```

Finally, check DirectoryIndex includes index.php
	DirectoryIndex index.php index.html

The php.ini and php-fpm.ini file can be found in:
	/usr/local/etc/php/5.6/

php@5.6 is keg-only, which means it was not symlinked into /usr/local,
because this is an alternate version of another formula.

If you need to have php@5.6 first in your PATH run:
  echo 'export PATH="/usr/local/opt/php@5.6/bin:$PATH"' >> ~/.bash_profile
  echo 'export PATH="/usr/local/opt/php@5.6/sbin:$PATH"' >> ~/.bash_profile

For compilers to find php@5.6 you may need to set:
  export LDFLAGS="-L/usr/local/opt/php@5.6/lib"
  export CPPFLAGS="-I/usr/local/opt/php@5.6/include"

To have launchd start exolnet/deprecated/php@5.6 now and restart at login:
  brew services start exolnet/deprecated/php@5.6
Or, if you don't want/need a background service you can just run:
  php-fpm
==> Summary
🍺 /usr/local/Cellar/php@5.6/5.6.40: 494 files, 60.5MB
==> Caveats
==> php@5.6
To enable PHP in Apache add the following to httpd.conf and restart Apache:
	LoadModule php5_module /usr/local/opt/php@5.6/lib/httpd/modules/libphp5.so

	<FilesMatch \.php$>
	SetHandler application/x-httpd-php
	</FilesMatch>

Finally, check DirectoryIndex includes index.php
	DirectoryIndex index.php index.html

```
The php.ini and php-fpm.ini file can be found in:
        /usr/local/etc/php/5.6/

php@5.6 is keg-only, which means it was not symlinked into /usr/local,
because this is an alternate version of another formula.

If you need to have php@5.6 first in your PATH run:
  echo 'export PATH="/usr/local/opt/php@5.6/bin:$PATH"' >> ~/.bash_profile
  echo 'export PATH="/usr/local/opt/php@5.6/sbin:$PATH"' >> ~/.bash_profile

For compilers to find php@5.6 you may need to set:
  export LDFLAGS="-L/usr/local/opt/php@5.6/lib"
  export CPPFLAGS="-I/usr/local/opt/php@5.6/include"


To have launchd start exolnet/deprecated/php@5.6 now and restart at login:
  brew services start exolnet/deprecated/php@5.6
Or, if you don't want/need a background service you can just run:
  php-fpm


$ brew install php@7.1
==> Downloading https://homebrew.bintray.com/bottles/php@7.1-7.1.25.mojave.bottle.tar.gz
######################################################################## 100.0%
==> Pouring php@7.1-7.1.25.mojave.bottle.tar.gz
==> /usr/local/Cellar/php@7.1/7.1.25/bin/pear config-set php_ini /usr/local/etc/php/7.1/php.ini system
==> /usr/local/Cellar/php@7.1/7.1.25/bin/pear config-set php_dir /usr/local/share/pear@7.1 system
==> /usr/local/Cellar/php@7.1/7.1.25/bin/pear config-set doc_dir /usr/local/share/pear@7.1/doc system
==> /usr/local/Cellar/php@7.1/7.1.25/bin/pear config-set ext_dir /usr/local/lib/php/pecl/20160303 system
==> /usr/local/Cellar/php@7.1/7.1.25/bin/pear config-set bin_dir /usr/local/opt/php@7.1/bin system
==> /usr/local/Cellar/php@7.1/7.1.25/bin/pear config-set data_dir /usr/local/share/pear@7.1/data system
==> /usr/local/Cellar/php@7.1/7.1.25/bin/pear config-set cfg_dir /usr/local/share/pear@7.1/cfg system
==> /usr/local/Cellar/php@7.1/7.1.25/bin/pear config-set www_dir /usr/local/share/pear@7.1/htdocs system
==> /usr/local/Cellar/php@7.1/7.1.25/bin/pear config-set man_dir /usr/local/share/man system
```

```
==> /usr/local/Cellar/php@7.1/7.1.25/bin/pear config-set test_dir /usr/local/share/pear@7.1/test system
==> /usr/local/Cellar/php@7.1/7.1.25/bin/pear config-set php_bin /usr/local/opt/php@7.1/bin/php system
==> /usr/local/Cellar/php@7.1/7.1.25/bin/pear update-channels
==> Caveats
To enable PHP in Apache add the following to httpd.conf and restart Apache:
    LoadModule php7_module /usr/local/opt/php@7.1/lib/httpd/modules/libphp7.so

    <FilesMatch \.php$>
    SetHandler application/x-httpd-php
    </FilesMatch>

Finally, check DirectoryIndex includes index.php
    DirectoryIndex index.php index.html

The php.ini and php-fpm.ini file can be found in:
    /usr/local/etc/php/7.1/

php@7.1 is keg-only, which means it was not symlinked into /usr/local,
because this is an alternate version of another formula.

If you need to have php@7.1 first in your PATH run:
  echo 'export PATH="/usr/local/opt/php@7.1/bin:$PATH"' >> ~/.bash_profile
  echo 'export PATH="/usr/local/opt/php@7.1/sbin:$PATH"' >> ~/.bash_profile

For compilers to find php@7.1 you may need to set:
  export LDFLAGS="-L/usr/local/opt/php@7.1/lib"
  export CPPFLAGS="-I/usr/local/opt/php@7.1/include"


To have launchd start php@7.1 now and restart at login:
  brew services start php@7.1
Or, if you don't want/need a background service you can just run:
  php-fpm
==> Summary
🍺  /usr/local/Cellar/php@7.1/7.1.25: 513 files, 63.2MB
```

```
$ brew install php@7.2
==> Downloading https://homebrew.bintray.com/bottles/php@7.2-7.2.14.mojave.bottle.tar.gz
######################################################################## 100.0%
==> Pouring php@7.2-7.2.14.mojave.bottle.tar.gz
==> /usr/local/Cellar/php@7.2/7.2.14/bin/pear config-set php_ini /usr/local/etc/php/7.2/php.ini system
==> /usr/local/Cellar/php@7.2/7.2.14/bin/pear config-set php_dir /usr/local/share/pear@7.2 system
==> /usr/local/Cellar/php@7.2/7.2.14/bin/pear config-set doc_dir /usr/local/share/pear@7.2/doc system
==> /usr/local/Cellar/php@7.2/7.2.14/bin/pear config-set ext_dir /usr/local/lib/php/pecl/20170718 system
==> /usr/local/Cellar/php@7.2/7.2.14/bin/pear config-set bin_dir /usr/local/opt/php@7.2/bin system
==> /usr/local/Cellar/php@7.2/7.2.14/bin/pear config-set data_dir /usr/local/share/pear@7.2/data system
==> /usr/local/Cellar/php@7.2/7.2.14/bin/pear config-set cfg_dir /usr/local/share/pear@7.2/cfg system
==> /usr/local/Cellar/php@7.2/7.2.14/bin/pear config-set www_dir /usr/local/share/pear@7.2/htdocs system
==> /usr/local/Cellar/php@7.2/7.2.14/bin/pear config-set man_dir /usr/local/share/man system
==> /usr/local/Cellar/php@7.2/7.2.14/bin/pear config-set test_dir /usr/local/share/pear@7.2/test system
==> /usr/local/Cellar/php@7.2/7.2.14/bin/pear config-set php_bin /usr/local/opt/php@7.2/bin/php system
==> /usr/local/Cellar/php@7.2/7.2.14/bin/pear update-channels
==> Caveats
To enable PHP in Apache add the following to httpd.conf and restart Apache:
    LoadModule php7_module /usr/local/opt/php@7.2/lib/httpd/modules/libphp7.so

    <FilesMatch \.php$>
    SetHandler application/x-httpd-php
    </FilesMatch>

Finally, check DirectoryIndex includes index.php
    DirectoryIndex index.php index.html

The php.ini and php-fpm.ini file can be found in:
    /usr/local/etc/php/7.2/

php@7.2 is keg-only, which means it was not symlinked into /usr/local,
```

```
because this is an alternate version of another formula.

If you need to have php@7.2 first in your PATH run:
  echo 'export PATH="/usr/local/opt/php@7.2/bin:$PATH"' >> ~/.bash_profile
  echo 'export PATH="/usr/local/opt/php@7.2/sbin:$PATH"' >> ~/.bash_profile

For compilers to find php@7.2 you may need to set:
  export LDFLAGS="-L/usr/local/opt/php@7.2/lib"
  export CPPFLAGS="-I/usr/local/opt/php@7.2/include"

To have launchd start php@7.2 now and restart at login:
  brew services start php@7.2
Or, if you don't want/need a background service you can just run:
  php-fpm
==> Summary
🍺  /usr/local/Cellar/php@7.2/7.2.14: 514 files, 75.0MB

$ brew install php@7.3
==> Downloading https://homebrew.bintray.com/bottles/php-7.3.1.mojave.bottle.tar.gz
Already downloaded:
/Users/epallier/Library/Caches/Homebrew/downloads/012e37ae0f9b4dc0cbd5faa3e3fabb9b71347dde138d38947819fe332b11fff0--php-7.3.1.mojave.
bottle.tar.gz
==> Pouring php-7.3.1.mojave.bottle.tar.gz
==> /usr/local/Cellar/php/7.3.1/bin/pear config-set php_ini /usr/local/etc/php/7.3/php.ini system
==> /usr/local/Cellar/php/7.3.1/bin/pear config-set php_dir /usr/local/share/pear system
==> /usr/local/Cellar/php/7.3.1/bin/pear config-set doc_dir /usr/local/share/pear/doc system
==> /usr/local/Cellar/php/7.3.1/bin/pear config-set ext_dir /usr/local/lib/php/pecl/20180731 system
==> /usr/local/Cellar/php/7.3.1/bin/pear config-set bin_dir /usr/local/opt/php/bin system
==> /usr/local/Cellar/php/7.3.1/bin/pear config-set data_dir /usr/local/share/pear/data system
==> /usr/local/Cellar/php/7.3.1/bin/pear config-set cfg_dir /usr/local/share/pear/cfg system
==> /usr/local/Cellar/php/7.3.1/bin/pear config-set www_dir /usr/local/share/pear/htdocs system
==> /usr/local/Cellar/php/7.3.1/bin/pear config-set man_dir /usr/local/share/man system
==> /usr/local/Cellar/php/7.3.1/bin/pear config-set test_dir /usr/local/share/pear/test system
==> /usr/local/Cellar/php/7.3.1/bin/pear config-set php_bin /usr/local/opt/php/bin/php system
==> /usr/local/Cellar/php/7.3.1/bin/pear update-channels
```

==> Caveats
To enable PHP in Apache add the following to httpd.conf and restart Apache:
    LoadModule php7_module /usr/local/opt/php/lib/httpd/modules/libphp7.so

    <FilesMatch \.php$>
    SetHandler application/x-httpd-php
    </FilesMatch>

Finally, check DirectoryIndex includes index.php
    DirectoryIndex index.php index.html

The php.ini and php-fpm.ini file can be found in:
    /usr/local/etc/php/7.3/

To have launchd start php now and restart at login:
  brew services start php
Or, if you don't want/need a background service you can just run:
  php-fpm
==> Summary
🍺 /usr/local/Cellar/php/7.3.1: 521 files, 76.4MB

Also, you may have the need to tweak configuration settings of PHP to your needs. A common thing to change is the memory setting, or the date.timezone configuration. The php.ini files for each version of PHP are located in the following directories:
/usr/local/etc/php/5.6/php.ini
/usr/local/etc/php/7.0/php.ini
/usr/local/etc/php/7.1/php.ini
/usr/local/etc/php/7.2/php.ini
/usr/local/etc/php/7.3/php.ini

Recopie de mes anciens php.ini :
cp -p /tmp/oldbrew_php/5.6/php.ini /usr/local/etc/php/5.6/
Pour toutes les versions 7, j’ai laissé le php.ini par défaut intact, sans modif

Let's switch back to the first PHP version now:
$ brew unlink php@7.3 && brew link --force --overwrite php@5.6
Unlinking /usr/local/Cellar/php/7.3.1... 24 symlinks removed
Linking /usr/local/Cellar/php@5.6/5.6.40... 25 symlinks created

If you need to have this software first in your PATH instead consider running:
  echo 'export PATH="/usr/local/opt/php@5.6/bin:$PATH"' >> ~/.bash_profile
  echo 'export PATH="/usr/local/opt/php@5.6/sbin:$PATH"' >> ~/.bash_profile

At this point, I strongly recommend closing ALL your terminal tabs and windows. This will mean opening a new terminal to continue with the next step. This is strongly recommended because some really strange path issues can arise with existing terminals (trust me, I have seen it!).

$ php -v
PHP 5.6.40 (cli) (built: Jan 16 2019 14:53:29)
Copyright (c) 1997-2016 The PHP Group
Zend Engine v2.6.0, Copyright (c) 1998-2016 Zend Technologies
        with Zend OPcache v7.0.6-dev, Copyright (c) 1999-2016, by Zend Technologies

****
2.2 - Apache PHP Setup - Part 1
****

If you have been following this guide correctly, the last entry should be your mod_rewrite module:
LoadModule rewrite_module lib/httpd/modules/mod_rewrite.so
Below this add the following libphp modules:
LoadModule php5_module /usr/local/opt/php@5.6/lib/httpd/modules/libphp5.so
#LoadModule php7_module /usr/local/opt/php@7.0/lib/httpd/modules/libphp7.so
#LoadModule php7_module /usr/local/opt/php@7.1/lib/httpd/modules/libphp7.so
#LoadModule php7_module /usr/local/opt/php@7.2/lib/httpd/modules/libphp7.so
#LoadModule php7_module /usr/local/opt/php@7.3/lib/httpd/modules/libphp7.so

Also you must set the Directory Indexes for PHP explicitly, so search for this block:

```
<IfModule dir_module>
        DirectoryIndex index.html
</IfModule>
```

and replace it with this:

```
<IfModule dir_module>
        DirectoryIndex index.php index.html
</IfModule>

<FilesMatch \.php$>
        SetHandler application/x-httpd-php
</FilesMatch>
```

```
$ sudo apachectl -k stop
$ sudo apachectl start
```

Simply create a file called info.php in your Sites/ folder you created earlier with this one-liner.

```
echo "<?php phpinfo();" > /usr/local/var/www
(ou echo "<?php phpinfo();" > ~/Sites/info.php)
```

Point your browser to http://localhost/info.php and you should see a shiny PHP information page:

You can test the other PHP versions by commenting the LoadModule ... php@5.6 ... entry and uncommenting one of the other ones. Then simply restart apache and reload the same page.

We hard-coded Apache to use PHP 5.6, but we really want to be able to switch between versions. Luckily, some industrious individuals have already done the hard work for us and written a very handy little PHP switcher script.
We will install the sphp script into brew's standard /usr/local/bin:

```
$ curl -L https://gist.githubusercontent.com/rhukster/f4c04f1bf59e0b74e335ee5d186a98e2/raw > /usr/local/bin/sphp
$ chmod +x /usr/local/bin/sphp
```

Homebrew should have added its preferred /usr/local/bin and /usr/local/sbin to your path as part of its installation process. Quickly test this by typing:

```
$ echo $PATH
/usr/local/bin:/usr/bin:/bin:/usr/sbin:/sbin
```

After you have completed these steps, you should be able to switch your PHP version by using the command sphp followed by a two digit value for the PHP version:

```
$ sphp 7.1
Switching to php@7.1
Switching your shell
Unlinking /usr/local/Cellar/php@5.6/5.6.40... 25 symlinks removed
Unlinking /usr/local/Cellar/php@7.1/7.1.25... 0 symlinks removed
Unlinking /usr/local/Cellar/php@7.2/7.2.14... 0 symlinks removed
Unlinking /usr/local/Cellar/php/7.3.1... 0 symlinks removed
Linking /usr/local/Cellar/php@7.1/7.1.25... 25 symlinks created

If you need to have this software first in your PATH instead consider running:
  echo 'export PATH="/usr/local/opt/php@7.1/bin:$PATH"' >> ~/.bash_profile
  echo 'export PATH="/usr/local/opt/php@7.1/sbin:$PATH"' >> ~/.bash_profile
You will need sudo power from now on
Switching your apache conf
Restarting apache

PHP 7.1.25 (cli) (built: Dec  7 2018 08:20:45) ( NTS )
Copyright (c) 1997-2018 The PHP Group
Zend Engine v3.1.0, Copyright (c) 1998-2018 Zend Technologies
        with Zend OPcache v7.1.25, Copyright (c) 1999-2018, by Zend Technologies

All done!
```

You will need to switch to each of your installed PHP versions and run update again to get updates for each PHP version and ensure you are running the version of PHP you intend.
Je l’ai fait pour 5.6 et 7.3 :

```
$ sphp 5.6
$ brew update
```

This will spit out a list of available updates, and any deleted formulas.
To upgrade the packages simply type:

```
$ brew upgrade
```

```
$ sphp 7.3
Switching to php@7.3
Switching your shell
Unlinking /usr/local/Cellar/php@5.6/5.6.40... 25 symlinks removed
Unlinking /usr/local/Cellar/php@7.1/7.1.25... 0 symlinks removed
Unlinking /usr/local/Cellar/php@7.2/7.2.14... 0 symlinks removed
Unlinking /usr/local/Cellar/php/7.3.1... 0 symlinks removed
Linking /usr/local/Cellar/php/7.3.1... 24 symlinks created
You will need sudo power from now on
Switching your apache conf
Restarting apache

PHP 7.3.1 (cli) (built: Jan 10 2019 13:15:37) ( NTS )
Copyright (c) 1997-2018 The PHP Group
Zend Engine v3.3.1, Copyright (c) 1998-2018 Zend Technologies
        with Zend OPcache v7.3.1, Copyright (c) 1999-2018, by Zend Technologies
$ brew update
$ brew upgrade
```

You can see the specific versions of a PHP package by typing:

```
$ brew info php@7.3

php: stable 7.3.1 (bottled)
General-purpose scripting language
https://secure.php.net/
/usr/local/Cellar/php/7.3.1 (521 files, 76.4MB) *
  Poured from bottle on 2019-01-22 at 12:25:39
From: https://github.com/Homebrew/homebrew-core/blob/master/Formula/php.rb
==> Dependencies
Build: httpd ✔, pkg-config ✔
```

Required: apr ✔, apr-util ✔, argon2 ✔, aspell ✔, autoconf ✔, curl-openssl ✔, freetds ✔, freetype ✔, gettext ✔, glib ✔, gmp ✔, icu4c ✔, jpeg ✔, libpng ✔, libpq ✔, libsodium ✔, libzip ✔, openldap ✔, openssl ✔, pcre2 ✔, sqlite ✔, tidy-html5 ✔, unixodbc ✔, webp ✔
==> Caveats
To enable PHP in Apache add the following to httpd.conf and restart Apache:
    LoadModule php7_module /usr/local/opt/php/lib/httpd/modules/libphp7.so

    <FilesMatch \.php$>
    SetHandler application/x-httpd-php
    </FilesMatch>

Finally, check DirectoryIndex includes index.php
    DirectoryIndex index.php index.html

The php.ini and php-fpm.ini file can be found in:
    /usr/local/etc/php/7.3/

To have launchd start php now and restart at login:
  brew services start php
Or, if you don't want/need a background service you can just run:
  php-fpm
==> Analytics
install: 37,504 (30 days), 123,299 (90 days), 417,557 (365 days)
install_on_request: 33,685 (30 days), 111,773 (90 days), 376,762 (365 days)
build_error: 0 (30 days)

$ brew services list
Name Status User   Plist
httpd   started root   /Library/LaunchDaemons/homebrew.mxcl.httpd.plist
mysql started epallier /Users/epallier/Library/LaunchAgents/homebrew.mxcl.mysql.plist
php   stopped
php@5.6 stopped
php@7.1 stopped
php@7.2 stopped

$ mysql --version
mysql Ver 8.0.13 for osx10.14 on x86_64 (Homebrew)

****

Part 2: macOS 10.14 Mojave Web Development Environment

****

https://getgrav.org/blog/macos-mojave-apache-mysql-vhost-apc

****

20.1.4. MYSQL => MARIADB

****

(mysql ne semble plus marcher sur mon mac, donc j'installe mariadb)

In the original guide, we used the Oracle MySQL installation package. However, we now have switched to MariaDB which is a drop-in replacement for MySQL and is easily installed and updated with Brew. Detailed information on the HomeBrew installation process can be found on the mariadb.org site but the essentials are as follows.

$ brew uninstall --force mysql
$ brew doctor
$ brew update

$ brew install mariadb
==> Downloading https://homebrew.bintray.com/bottles/mariadb-10.3.12.mojave.bottle.tar.gz
100.0%
==> Pouring mariadb-10.3.12.mojave.bottle.tar.gz
==> Caveats
A "/etc/my.cnf" from another install may interfere with a Homebrew-built
server starting up correctly.

MySQL is configured to only allow connections from localhost by default

To connect:
    mysql -uroot

To have launchd start mariadb now and restart at login:
  brew services start mariadb
Or, if you don't want/need a background service you can just run:
  mysql.server start
==> Summary
🍺 /usr/local/Cellar/mariadb/10.3.12: 658 files, 174.4MB
macp1219:install epallier$ brew services start mariadb
==> Successfully started `mariadb` (label: homebrew.mxcl.mariadb)
macp1219:install epallier$ mysql -uroot

After a successful installation, you can start the server to ensure it autostarts in the future with:
$ brew services start mariadb
You should get some positive feedback on that action:
==> Successfully started `mariadb` (label: homebrew.mxcl.mariadb)

(PAS FAIT)
Download SequelPro and install it. (it's awesome and free!). You should be able to automatically create a new connection via the Socket option without changing any settings.

(PAS FAIT, je garde “no password”)
It is advisable to change the **MySQL server password** and secure your installation. The simplest way to do this is to use the provided script:
$ /usr/local/bin/mysql_secure_installation
Just answer the questions and fill them in as is appropriate for your environment.
If you need to stop the server, you can use the simple command:

$ brew services stop mariadb

20.1.5. Apache Virtual Hosts

Apache already comes preconfigured to support this behavior but it is not enabled. First you will need to uncomment the following lines in your /usr/local/etc/httpd/httpd.conf file:
LoadModule vhost_alias_module lib/httpd/modules/mod_vhost_alias.so
and:
Virtual hosts
Include /usr/local/etc/httpd/extra/httpd-vhosts.conf
Then you can edit this referenced file and configure it to your needs:
$ vi /usr/local/etc/httpd/extra/httpd-vhosts.conf

When you set up virtual hosts, you will lose your older document root, so you will need to add back support for that first as a virtual host.

```
<VirtualHost *:80>
        DocumentRoot "/Users/your_user/Sites"
        ServerName localhost
</VirtualHost>

<VirtualHost *:80>
        DocumentRoot "/Users/your_user/Sites/grav-admin"
        ServerName grav-admin.test
</VirtualHost>
```

Dnsmasq (PAS FAIT car j'utilise plutot /etc/hosts)
We used to recommend using .dev domain name, but since Chrome 63 forces all .dev domains to use SSL, this guide has been updated to use .test
In the example virtualhost we setup above, we defined a ServerName of grav-admin.test. This by default will not resolve to your local machine, but it's often very useful to be able to setup various virtual hosts for development purposes. You can do this by manually adding entries to /etc/hosts ever time, or you can install and configure Dnsmasq to automatically handle wildcard *.test names and forward all of them to localhost (127.0.0.1).

```
$ cat /etc/hosts
##
# Host Database
#
```

```
# localhost is used to configure the loopback interface
# when the system is booting.  Do not change this entry.
##
127.0.0.1    localhost
255.255.255.255    broadcasthost
::1                localhost
127.0.0.1          labinvent.test

$ ping labinvent.test
```

APC Cache (pas fait)
Caching in PHP is a big part of the performance equation. There are two types of caching typically available, and both have a big impact on speed and performance.
The first type of cache is called an opcode cache, and this is what takes your PHP script and compiles it for faster execution. This alone can typically result in a 3X speed increase!.
The second type of cache is a user cache, and this is a data-store that PHP can use to quickly store and retrieve data from. These typically run in memory which means they are transient, but very fast.
All PHP packages now come pre-built with Zend OPcache by default, but you can still install APCu Cache as a data store.

Xdebug (pas fait)
One of the most important aspects of any kind of development is the ability to debug and fix your code. PHP comes with limited support to dump variables or log to a file, but for more complex situations you need something more powerful.
Xdebug provides is a debugging and profiling extension for PHP that provides an HTML-friendly output for the var_dump() method that improves the readability of the default version. It also provides other useful dumping methods as well as displaying stack traces. One of the best features however, is the ability to remote debug your code. This means you can set breakpoints, and step through your PHP code inspecting as you go. Full documentation on Xdebug contains extensive information about all the functionality available. Let's switch back to PHP 5.6 and get started.

NOTE: The brew installation actually creates configuration files in /usr/local/etc/php/5.6/conf.d, /usr/local/etc/php/7.0/conf.d, /usr/local/etc/php/7.1/conf.d, /usr/local/etc/php/7.2/conf.d, and /usr/local/etc/php/7.3/conf.d respectively. If you want to uninstall a PHP extension, simply rename the .ini file to .ini.bak and restart apache. Alternatively, you can simply use brew to uninstall it, and reinstall it again when you need it.

20.1.6. PHPMYADMIN

```
$ brew install phpmyadmin
Updating Homebrew...
==> Auto-updated Homebrew!
Updated 1 tap (homebrew/core).
==> Updated Formulae
graph-tool                                librsvg                                siril

==> Downloading https://files.phpmyadmin.net/phpMyAdmin/4.8.4/phpMyAdmin-4.8.4-all-languages.tar.gz
Already downloaded:
/Users/epallier/Library/Caches/Homebrew/downloads/d0266b8cf3a58a0c144a4b2fe8723d6fe8e854e1b6c81f6e4667236ee5875e76--phpMyAdmin-4.
8.4-all-languages.tar.gz
==> Caveats
To enable phpMyAdmin in Apache, add the following to httpd.conf and
restart Apache:
    Alias /phpmyadmin /usr/local/share/phpmyadmin
    <Directory /usr/local/share/phpmyadmin/>
    Options Indexes FollowSymLinks MultiViews
    AllowOverride All
    <IfModule mod_authz_core.c>
    Require all granted
    </IfModule>
    <IfModule !mod_authz_core.c>
    Order allow,deny
    Allow from all
    </IfModule>
    </Directory>
Then open http://localhost/phpmyadmin
The configuration file is /usr/local/etc/phpmyadmin.config.inc.php
==> Summary
🍺  /usr/local/Cellar/phpmyadmin/4.8.4: 2,434 files, 32.3MB, built in 7 seconds
```

Config dans /usr/local/etc/phpmyadmin.config.inc.php pour connexion avec “root” sans pass :
//$cfg['Servers'][$i]['host'] = 'localhost';
$cfg['Servers'][$i]['host'] = '127.0.0.1';

//$cfg['Servers'][$i]['AllowNoPassword'] = false;
$cfg['Servers'][$i]['AllowNoPassword'] = true;

$cfg['Servers'][$i]['controluser'] = 'root';
$cfg['Servers'][$i]['controlpass'] = '';

20.1.7. Installation de l’extension YAML

$ php -v
=> 7.3

$ brew install libyaml
$ pecl install yaml
=> ça plante

Enlever l’extension du fichier php.ini et la mettre dans conf.d/ :
$ vi /usr/local/etc/php/7.3/php.ini
;extension="yaml.so"
$ vi /usr/local/etc/php/7.3/conf.d/ext-yaml.ini
[yaml]
;extension=/usr/local/Cellar/php/7.3.1/pecl/20180731/yaml.so
extension=yaml.so

J’ai supprimé un vieux lien qui ne servait à rien et gênait l’installation qui voulait créer un dossier pecl/ à la place :
$ rm /usr/local/Cellar/php/7.3.1/pecl

Je recommence l’install :
$ pecl install yaml

=> OK

$ php -v
=> l'extension n'est pas trouvée

Il manque pour ça un lien vers pecl/ :
$ cd /usr/local/lib/php
$ ln -s ../../Cellar/php/7.3.1/pecl/

$ php -v
=> OK

Pour que ça marche depuis le web :
$ sudo apachectl -k restart

Pour l'installer sur une autre version de php, par exemple 5.6 :
$ sphp 5.6
$ pecl uninstall -r yaml
$ pecl install yaml

$ cd /usr/local/etc/php/7.3/conf.d/
cp ext-yaml.ini ../../5.6/conf.d
cp ext-yaml.ini ../../7.2/conf.d

20.2. Installation des pré-requis sur Windows 10

20.2.1. Installation directement sur Windows 10 (TODO)

TODO:
Installer WAMP server, ou bien XAMPP.
Il faudra aussi convertir le script d'installation bash install/installation.sh en PHP.
Un volontaire ?

20.2.2. Installation sur une VM linux (avec VirtualBox)

(updated 17/1/20 - EP)
(testé le 17/1/20 sur Mac OS X 10.14 avec VB 6.1.2 et une VM Ubuntu 18.04.3 LTS)

(

- *Autre option possible : installer linux en dual-boot :*

https://lecrabeinfo.net/installer-ubuntu-18-04-lts-dual-boot-windows-10.html#etape-4-installer-ubuntu-1804-lts

- *Autre VM disponible, CentOS:*

https://pixelabs.fr/machine-virtuelle-centos-7-virtualbox/

)

Dans le détail, suivre ce tuto:

https://lecrabeinfo.net/virtualbox-installer-windows-linux-dans-une-machine-virtuelle.html

(autre méthode possible : https://openclassrooms.com/fr/courses/43538-reprenez-le-controle-a-laide-de-linux/37630-installez-linux-dans-une-machine-virtuelle)

En résumé (avec quelques remarques pertinentes absentes du tuto) :

- **Télécharger une image iso Linux (ici Ubuntu 18 LTS)**
  https://ubuntu.com/#download

- **Installer VirtualBox (VB)**
  https://www.virtualbox.org/wiki/Downloads

- **Ajouter le pack extension de VirtualBox pour bénéficier du plein écran et du copier-coller**
  Télécharger ce pack (on trouve le lien de téléchargement sur la même page)
  Démarrer VB
  Cliquer sur Paramètres
  Cliquer sur Extensions
  Cliquer sur “+” et sélectionner le fichier téléchargé, puis sur Installation

- **Créer une machine virtuelle (VM) à partir de VB**
  Cliquez sur le bouton **Nouvelle**

  Donnez un **nom** à la machine virtuelle (ex. : ubuntu18), puis sélectionnez le **type** “linux” et la **version** “ubuntu” du système d’exploitation.
  Sélectionnez la **quantité de mémoire vive** à attribuer la VM
  Si vous avez assez de mémoire vive et pour plus de confort, je vous conseille de choisir **2048 Mo** (2 Go). Faites attention à laisser assez de mémoire vive à la machine hôte.

Ajoutez un **disque virtuel**. Il sera enregistré sous la forme d'un fichier disque (ex. : « ubuntu18.vdi ») sur votre PC.
VirtualBox affiche la **taille recommandée** pour le disque virtuel en fonction du système d'exploitation choisi précédemment. Sélectionnez **Créer un disque dur virtuel maintenant** puis cliquez sur **Créer.**
Type de fichier pour le disque virtuel : si vous n'utilisez que VirtualBox comme logiciel de virtualisation, laissez le choix sur VDI (sélectionnez un autre format si vous souhaitez une compatibilité avec un autre logiciel de virtualisation, VHD est le format utilisé par Windows Virtual PC, le format VMDK par VMWare).
Type de stockage sur le disque de votre PC : **Dynamiquement alloué** (le fichier .vdi représentant le disque virtuel grossira automatiquement en fonction de la quantité de données que vous enregistrerez dessus dans la machine virtuelle. Si vous choisissez une taille de disque de 50 Go, le fichier .vdi représentant le disque virtuel occupera 0 Go sur votre PC, puis 10 Go après l'installation de Windows, puis 11 Go après l'installation de plusieurs logiciels, etc).
Pour terminer, choisissez la **taille** du disque virtuel puis cliquez sur **Créer.**
(pour moi, c'est créé dans ~/VirtualBox VMs/ubuntu18/ubuntu18.vdi)
Pour démarrer la VM, sélectionnez-la puis cliquez sur le bouton **Démarrer.**
Disque de démarrage : pour pouvoir installer notre système d'exploitation dans la machine virtuelle, il faut amorcer celle-ci à partir d'une disque de démarrage, l'image ISO du système d'exploitation Ubuntu18 à installer, puis cliquez sur **Démarrer**.
Votre machine virtuelle démarre **à partir de l'image ISO**, il ne vous reste plus qu'à installer le système d'exploitation en suivant les instructions de l'installeur.

- **Installer Ubuntu dans la VM**
  Cliquer sur "Install Ubuntu"
  Cliquer sur "Français" et "Installer Ubuntu"
  Cliquer sur "Installation normale" et "Installer un logiciel tiers pour le matériel graphique et wifi…"
  Cliquer sur "Effacer le disque et installer Ubuntu"
  J'ai nommé la machine "vm-labinvent-ub18", user "Etienne Pallier" (login="labinv", pass="labinv")

- **Configurer la VM**
  Réseau : vérifier qu'il est bien configuré en **NAT** (c'est ce qui est fait par défaut)
  Pour le copier-coller : Menu Périphériques→ Presse-Papier Partagé, choisir Bidirectionnel

(ou Devices/Shared Clibboard/Bidirectional)

Pour le Plein écran :

- menu périphériques, insérer CD extension => fait le lien vers le package extension VirtualBox (valider par oui…)
- menu View/Virtual screen 1/ choisir une résolution
- Sur Win10 : CTRL-F pour passer en plein écran, CTRL-C pour revenir en petit écran (touche CTRL de droite)
- Sur Mac : Cmd-F et Cmd-C (touche Cmd de gauche)

- **Maintenant, il ne reste qu'à procéder à l'installation de LabInvent** comme si vous étiez sur un pc linux !
  Aller au début de cette doc, dans le chapitre "Installation des pré-requis", dans la section "Installation sur linux", suivre l'exemple pour Ubuntu "2.2.3.1. Exemple pour une distribution UBuntu 18.04.3 LTS (fait en 2020 01) (stagiaire Jeanne Prugniel)".

20.3. Installation et configuration du logiciel depuis l'IDE Eclipse

0) Installer Eclipse (Si nécessaire), voire Java

En effet, même si la version que vous allez installer est une version pour PHP, Eclipse à besoin de Java pour pouvoir s'exécuter.

Vérifiez que Java soit bien installé sur votre système :

java -version

Si ce n'est pas le cas, exécutez les lignes suivantes :
sudo apt-add-repository ppa:webupd8team/java
sudo apt-get update
sudo apt-get install oracle-java8-installer

Pour installer Eclipse, allez sur : http://www.eclipse.org/downloads/packages/release/Neon/3

Selectionnez la version que vous désirez (Neon, Oxygen, Mars ...) puis sélectionnez "Eclipse for PHP Developers".

Téléchargez la version correspondant à votre système d'exploitation.

Placez-vous dans le dossier ou vou voulez installer Eclipse avec l'archive précédement téléchargée (renommez-la en eclipse.tar.gz), puis exécutez la commande suivante :

tar -zxvf eclipse.tar.gz

Si vous voulez Eclipse dans le menu des applications sous Ubuntu, il vous faudra crééer le fichier eclipse.desktop :

gksudo gedit /usr/share/applications/eclipse.desktop

puis collez-y ce qui suit, en modifiant le chemin des lignes Exec et Icon au besoin :

```
[Desktop Entry]
Name=Eclipse
Type=Application
Exec=/opt/eclipse/eclipse
Terminal=false
Icon=/opt/eclipse/icon.xpm
Comment=Integrated Development Environment
NoDisplay=false
Categories=Development;IDE
Name[en]=eclipse.desktop
```

Puis donnez les droits à tous les utilisateurs sur ce fichier :

sudo chmod a+r /usr/share/applications/eclipse.desktop

1) Désactiver la vérification du certificat

Window -> Preferences -> Team -> git -> configuration -> Add entry

Key = http.sslVerify

Value = false

2) Récupérer le projet

(Si le projet git n'existe pas déjà sur votre machine)

File/Import project from git

Select repository source: Clone URI: https://gitlab.irap.omp.eu/epallier/labinvent.git

Directory:

- par défaut, il propose : /Users/epallier/git/labinvent

- mais on peut le mettre n'importe où ailleurs,

par exemple, on pourrait le mettre directement dans le repertoire web de Apache:

/Applications/XAMPP/xamppfiles/htdocs

(si on veut que le projet s'execute directement dans le dossier web apache htdocs, mais ca n'est pas obligatoire...)

initial branch: master

remote name: origin

Import as general project

Project name: LABINVENT

(si le projet git existe déjà sur votre machine)

File/Import project from git

Existing local repository

Directory

ADD => Selectionnez le dossier contenant le projet git => Finish => Next

Import existing Eclipse project

Selectionner le projet => Search for nested project => finish

3) Configurer le projet

a) S'assurer que le projet est bien reconnu comme un projet PHP (il doit y avoir un petit "P" sur le dossier racine du projet)

Si ça n'est pas le cas, vérifier que le fichier .project (à la racine) contient bien

<natures>

<nature>org.eclipse.php.core.PHPNature</nature>

</natures>

NB : Le fichier .project est normalement versionné et donc le projet labinvent devrait être reconnu automatiquement comme projet PHP

b) S'assurer que les fichiers de vue de cakephp ("*.ctp") sont bien reconnus comme des fichiers PHP.

Pour tester cela, ouvrir le fichier de vue cakephp/app/View/Categories/get_all.ctp

Si ce fichier s'ouvre comme un simple fichier texte, c'est qu'il n'est pas reconnu par Eclipse comme un fichier Php.

Il faut donc associer l'editeur Php a l'extension de fichier "*.ctp" :

- Preferences/General/Content types

- Dans la liste "Content types", ouvrir la section "Text", selectionner PHP

- Ajouter l'extension "*.ctp"

c) Vérifier la version de php utilisée (il serait préférable d'utiliser la meme version que celle officiellement utilisée par le logiciel, c'est à dire php 5.6, mais attention, le serveur IRAP utilise toujours une version 5.3 pour inventirap) :

- Clic-droit sur le projet, Propriétés

- PHP

- Interpreter

- Enable project specific settings, PHP Version : "PHP 5.6"

d) S'assurer que le texte est bien encodé en UTF-8 par défaut :

clic-droit sur le dossier racine du projet (dans PHP Explorer), Properties, Resource : dans la zone "Text file encoding" cocher "Other" et sélectionner UTF-8

(

Il faudrait commiter ça mais je ne sais pas trop si c'est risqué ou pas.

Les fichiers concernés sont :

- .project (déjà versionné) : car il commence par la ligne ""

- mais c'est surtout celui-ci qui compte (actuellement ignoré de git) : .settings/org.eclipse.core.resources.prefs : car sa 2eme ligne est "encoding/<project>=UTF-8"

)

Les éléments suivants sont normalement DEJA ignorés par git, à vérifier :

- .settings/

- cakephp/app/tmp/ : tout sauf

- documents/

- cakephp/app/Config/ :

- database.php

- labinvent.php

****

REMARQUES INTERRESSANTES (MAIS VOUS POUVEZ LES IGNORER)

// DEBUT DES REMARQUES

A la racine du projet, j'ai plusieurs éléments cachés de configuration Eclipse :

1) fichier .buildpath
Il est versionné puisque "svn status .buildpath" (depuis la console) ne donne rien
Il contient :

```
<?xml version="1.0" encoding="UTF-8"?>
<buildpath>
	<buildpathentry kind="con" path="org.eclipse.php.core.LANGUAGE"/>
	<buildpathentry kind="lib" path="docs/mockup/mockup_html.zip"/>
	<buildpathentry kind="src" path="cakephp"/>
</buildpath>
```

2) fichier .project
Il est déjà versionné
Il contient :

```
<?xml version="1.0" encoding="UTF-8"?>
<projectDescription>
	<name>invirap</name>
	<comment></comment>
	<projects>
	</projects>
	<buildSpec>
		<buildCommand>
			<name>org.eclipse.wst.common.project.facet.core.builder</name>
			<arguments>
			</arguments>
		</buildCommand>
		<buildCommand>
```

```
            <name>org.eclipse.wst.validation.validationbuilder</name>
            <arguments>
            </arguments>
        </buildCommand>
        <buildCommand>
            <name>org.eclipse.dltk.core.scriptbuilder</name>
            <arguments>
            </arguments>
        </buildCommand>
    </buildSpec>
    <natures>
        <nature>org.eclipse.php.core.PHPNature</nature>
        <nature>org.eclipse.wst.common.project.facet.core.nature</nature>
    </natures>
</projectDescription>
```

3) dossier .settings/ (exclus de svn)
Il contient 3 fichiers :

- org.eclipse.core.resources.prefs : bizarrement, il ne contient que quelques références seulement :

```
eclipse.preferences.version=1
encoding//cakephp/app/Controller/MaterielsController.php=UTF-8
encoding//cakephp/app/View/Elements/menu_view.ctp=UTF-8
encoding//cakephp/app/View/Layouts/default.ctp=UTF-8
encoding//cakephp/app/View/Materiels/index.ctp=UTF-8
encoding//cakephp/app/View/Materiels/scaffold.view.ctp=UTF-8
encoding//database/Upd_TableConstraints.sql=UTF-8
encoding//database/update/README.txt=UTF-8
encoding//docs/HOWTO.txt=UTF-8
```

encoding//install/HOWTO.txt=UTF-8
encoding/<project>=UTF-8

- org.eclipse.php.core.prefs
eclipse.preferences.version=1
include_path=0;/invirap\u00051;/invirap/docs/mockup/mockup_html.zip

- org.eclipse.wst.common.project.facet.core.xml : sans doute inutile ? (lié à "Faceted Project Validation Builder" dans Properties/Builders)
<?xml version="1.0" encoding="UTF-8"?>
<faceted-project>
  <fixed facet="php.core.component"/>
  <fixed facet="php.component"/>
  <installed facet="php.core.component" version="1"/>
  <installed facet="php.component" version="5.4"/>
</faceted-project>

// FIN DES REMARQUES
****

4) (TODO:) Set Code style

Window/Preferences : PHP / Editor

...

5) (TODO:) virtualenv

Now, once the PHP5 virtual environment is installed (see above),

set it in Eclipse as the project interpreter:

(cf http://virtphp.org)

...

6) (TODO:) Test

7) (TODO:) Run

check http://localhost:8080/

20.4. MODE PANIQUE

Vous avez changé la configuration du LDAP et elle ne fonctionne plus !

Du coup, vous ne pouvez plus vous connecter au site !

Pas de panique, le mode panique est là !

Il suffit de repasser le logiciel en mode INSTALL pour pouvoir y accéder sans connexion.

Pour cela, exécuter le script mode_panique.sh qui se trouve dans le dossier database/ :

```
$ ./mode_panique.sh
```

(

Si vous préférez, vous pouvez aussi exécuter manuellement la requête sql set_mode_install.sql qui se trouve dans le même dossier:

mysql -u login -p labinvent2 < set_mode_install.sql

Remplacer 'login' par le 'username' défini dans votre fichier config/app.php dans la section Datasources (default)

Remplacer 'labinvent2' par le 'database' défini au même endroit

Ou bien exécuter cette requête depuis phpmyadmin

)

20.5. MODE DEBUG (dev only)

Pour les développeurs, il y a un outil très pratique à utiliser qui est DebugKit. Il aide à voir ce qui se passe en interne au sein du framework cakephp (valeur des variables, dernières requetes sql exécutées...).

Il est installé par défaut. Donc, pour l'activer, rien de plus simple, il suffit de passer config/app.php en mode debug (ligne 12 qui définit la variable debug, passer à "true")

20.6. UPDATE DE LA BRANCHE MASTER (dev assermenté only)

Attention, cette opération ne doit être réalisée que par le responsable du projet LabInvent.

Pour mettre à jour la branche "master" à partir de la branche "dev-IRAP" (ou "dev") :

Aller sur la branche master :

$ git checkout master

$ git branch

Mettre à jour la branche master :

$ git merge --no-ff dev-IRAP

$ git push

Revenir sur la branche dev-IRAP (ou dev)

$ git checkout dev-IRAP

20.7. Auto-génération du code (avec “bake”)

(fait début 2019)

Tentative d’auto-génération du code avec “cake bake”, pour voir ce qui pourrait manquer au code actuel, pour y ajouter plus de cohérence et peut-être des nouveautés...

$ bin/**cake bake model** Materiels

Notice Error: Undefined index: Gestionnaires in vendor/cakephp/bake/src/View/Helper/DocBlockHelper.php, line 234
Notice Error: Undefined index: Photos in vendor/cakephp/bake/src/View/Helper/DocBlockHelper.php, line 234

Baking **table** class for Materiels...
Wrote src/Model/Table/**MaterielsTable**.php

Baking **entity** class for Materiel...
Wrote src/Model/Entity/**Materiel**.php

Baking **test fixture** for Materiels...
Wrote tests/Fixture/**MaterielsFixture**.php`

Baking **test case** for App\Model\Table\MaterielsTable ...
Wrote tests/TestCase/Model/Table/**MaterielsTableTest**.php`

20.8. METHODES ET CONCEPTS INTERESSANTS A UTILISER

Dans la vue Materiels/view, find, et index :

$materiel->**has**('designation') ? $materiel->designation : ‘’

Dans le controleur AppController, définition de fonctions passées aux vues :

```
$displayElement = function ($nom, $valeur, $params = "") {
        $balise = ($params != "") ? '<td ' . $params . '>' : '<td>';
        // Ca c'est parce que sinon y'a au moins deux tests qui passent pas, a cause de l'espace dans la balise ...
        if ($valeur != "")
        echo '<tr><td><strong>' . $nom . ' </strong></td>' . $balise . $valeur . '</td></tr>';
};
$this->set('displayElement', $displayElement);

$dateProchainControleVerif = function ($t) {
        $time = Time::now(); // On récupère la date et l'heure actuelles
        $today = new \DateTime((new date("$time->year-$time->month-$time->day"))->format('Y-m-d'));
        $time1 = new time($t);
        $dateTime1 = new \DateTime((new date("$time1->year-$time1->month-$time1->day"))->format('y-m-d'));
        $interval = ($today->diff($dateTime1));
        $strInterval = $interval->format('%a');
        return (int) $strInterval;
};
$this->set('dateProchainControleVerif', $dateProchainControleVerif);
```

⇒ utilisées dans la vue Suivis/view :

```
if ($dateProchainControleVerif($suivi->date_prochain_controle) < 0)
        $style = '"color: red"';
elseif ($dateProchainControleVerif($suivi->date_prochain_controle) <= 15)
        $style = '"color: orange"';
else $style = '"color: green"';
```

```
$displayElement(__('Materiel'), $suivi->has('materiel') ?
        $this->Html->link($suivi->materiel->designation, [
                'controller' => 'Materiels',
                'action' => 'view',
                $suivi->materiel->id
        ])
        :
        ''
);
```