BTS Solutions d'Infrastructure, Systèmes et Réseaux – 2ème année

Mettre en place une solution de gestion avec GLPI

Raphael SANCHEZ – BTS SIO

Cahier des charges :

Gestion de parc :

- Installer et configurer un logiciel de gestion de parc – Détailler la procédure d'installation du logiciel de gestions de parc, ici GLPI et d'incidents sur Linux
- Montrer l'installation des éventuels agents sur un poste client et vérifier que le poste est bien répertorié dans le logiciel de gestion de parc.

Gestion des incidents :

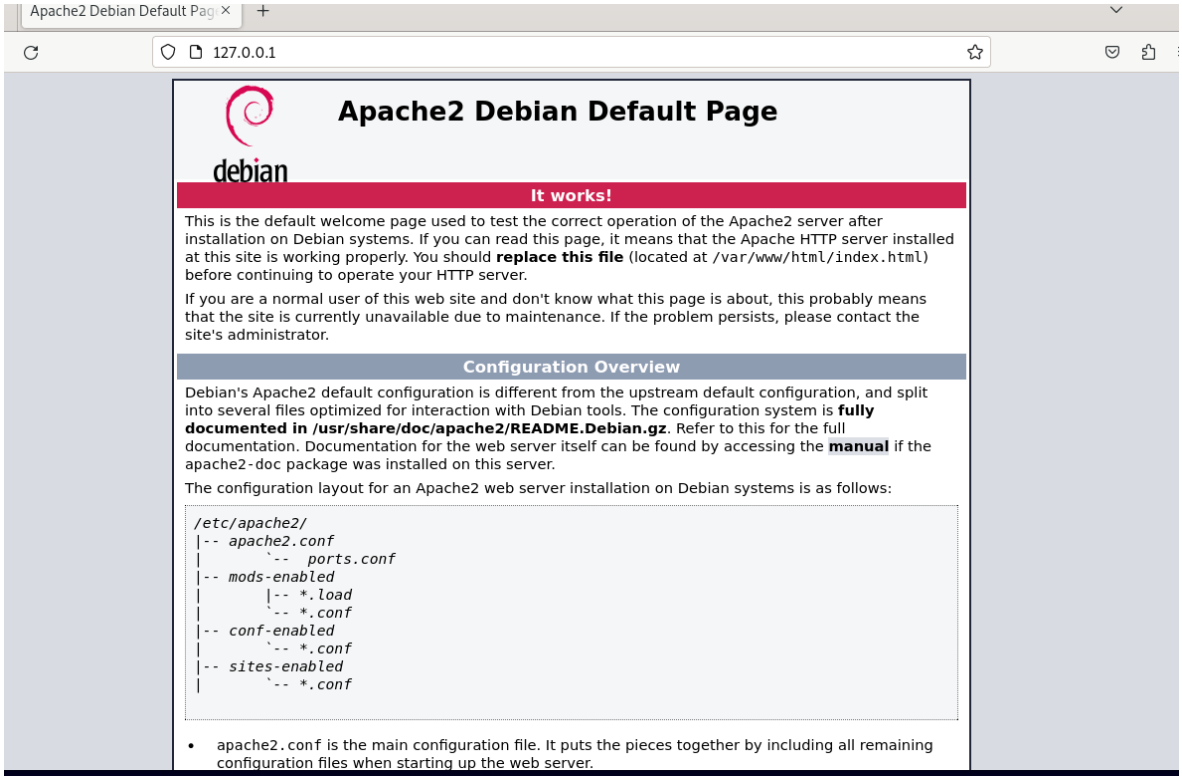
- Créer un/des utilisateurs, « tech » ayant un profil de technicien (notion de profil utilisateur)
- Créer 2 utilisateurs, Ilyan & Loan ayant un profil permettant uniquement de créer des tickets d'incidents.
- Se connecter avec un profil technicien (qui gère la résolution des tickets) à gérer les tickets et les affecter aux techniciens chargés de les résoudre.
- Résoudre un incident (gestion et suivi des incidents) et clore le ticket

1ère étape :

Commençons par mettre à jour tous les paquets du système debian 12. Pour cela on va utiliser la commande « sudo apt-get update & sudo apt-get upgrade ».

Une fois que tout est à jour, on peux commencer avec l'installation du paquet Apache2 qui nous permet de faire la configuration de nos sites en local.

La commande « sudo apt install apache2 » ici sera utilisé pour pouvoir installer Apache2



Comme **GLPI** utilise principalement une base de données, nous allons préparer **MariaDB** pour qu'il puisse héberger la base de données de **GLPI**.

Pour vérifier que l'installation a bien fonctionné, sur le navigateur on entre l'adresse IP « 127.0.0.1 » et si la page d'**Apache2** par défaut comme si dessus apparait cela nous montre que l'installation s'est bien déroulée.

2ème étape :

Avant de pouvoir faire l'installation de **MariaDB**, installer les paquets du socle **LAMP**. Sous **Debian 12**, qui est la dernière version stable de Debian, **PHP 8.2** est distribué par défaut dans les dépôts officiels. (**PHP 7.4** n'est plus supporté donc il faut absolument mettre à jour vers **PHP 8.2**) Ensuite il faut installer les extensions de **PHP** et une fois cela fait, nous pouvons procéder à l'installation de **MariaDB**.

sudo apt-get install php-xml php-common php-json php-mysql php-mbstring php-curl php-gd php-intl php-zip php-bz2 php-imap php-apcu

Pour effectuer une installation sécurisée on tape cette commande « sudo mysql_secure_installation ».

```
Setting the root password or using the unix_socket ensures that nobody
can log into the MariaDB root user without the proper authorisation.

You already have your root account protected, so you can safely answer 'n'.

Switch to unix_socket authentication [Y/n] n
 ... skipping.

You already have your root account protected, so you can safely answer 'n'.

Change the root password? [Y/n] y
New password:
Re-enter new password:
Password updated successfully!
Reloading privilege tables..
 ... Success!


By default, a MariaDB installation has an anonymous user, allowing anyone
to log into MariaDB without having to have a user account created for
them.  This is intended only for testing, and to make the installation
go a bit smoother.  You should remove them before moving into a
production environment.

Remove anonymous users? [Y/n] y
 ... Success!

Normally, root should only be allowed to connect from 'localhost'.  This
ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] y
 ... Success!

By default, MariaDB comes with a database named 'test' that anyone can
access.  This is also intended only for testing, and should be removed
before moving into a production environment.

Remove test database and access to it? [Y/n] y
 - Dropping test database...
 ... Success!
 - Removing privileges on test database...
 ... Success!

Reloading the privilege tables will ensure that all changes made so far
will take effect immediately.

Reload privilege tables now? [Y/n] y
 ... Success!

Cleaning up...

All done!  If you've completed all of the above steps, your MariaDB
installation should now be secure.

Thanks for using MariaDB!
```

Une fois cela fait **MariaDB** est configuré.

Maintenant on peux se connecter a notre instance **MariaDB** avec cette commande.

“sudo mysql -u root -p”

Une fois cela fait, on peux passer a nos requêtes **SQL** pour crée la base de données **GLPI** avec la commande

CREATE DATABASE ;

GRANT ALL PRIVILEGES ON **PGLPI**.* TO **PGLPI_AD**@localhost IDENTIFIED BY "**NotreMotDePasse**";

FLUSH PRIVILEGES;

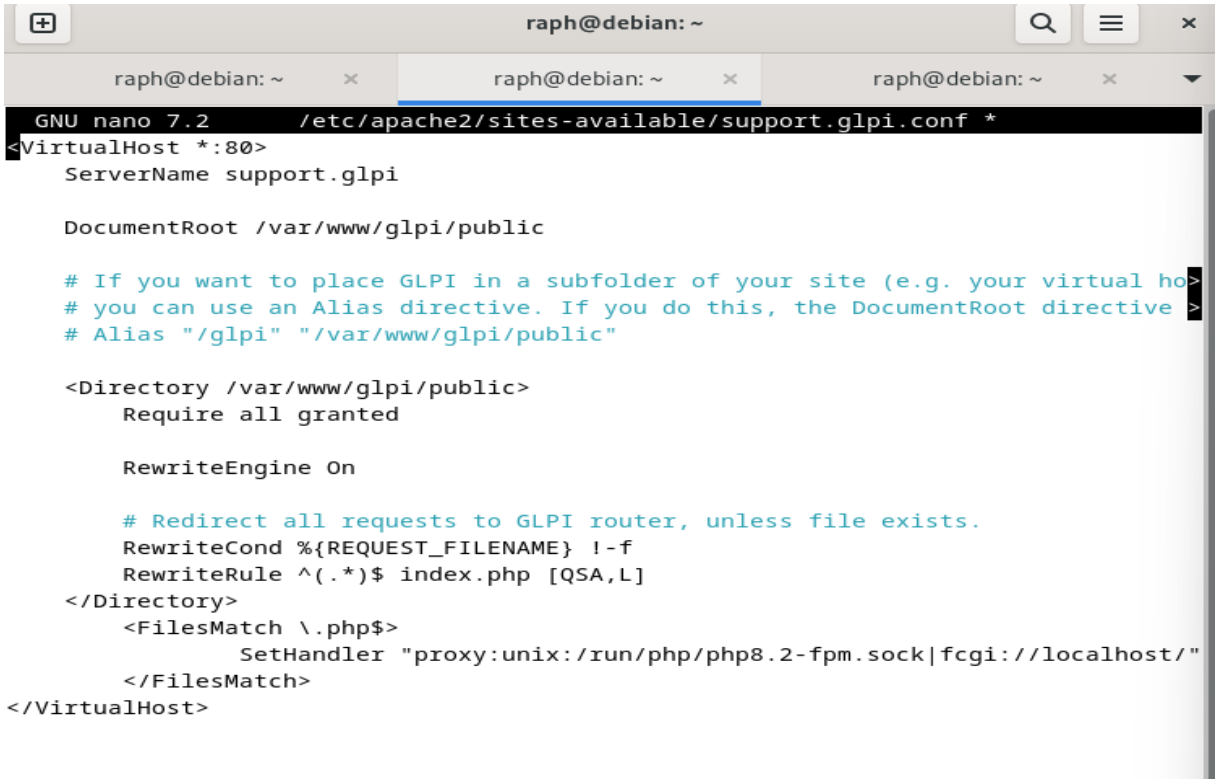
Après cela notre de base de données est prête.

3ème Etape :

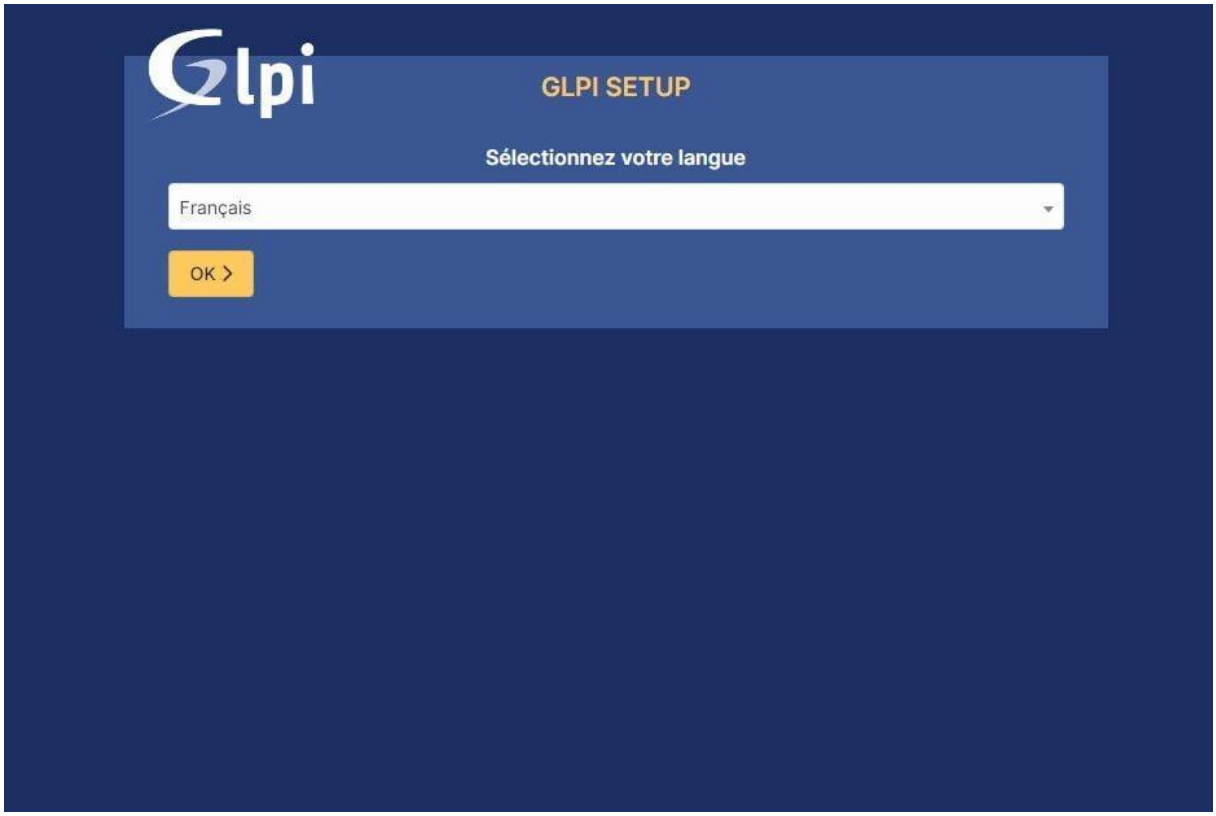
Récupérer l’archive de **GLPI** pour cela on va utiliser la commande : « wget https://github.com/glpi-project/glpi/releases/download/10.0.10/glpi-10.0.10.tgz »

```
root@debian:/tmp# wget https://github.com/glpi-project/glpi/releases/download/10
.0.10/glpi-10.0.10.tgz
```

Voici la préparation du fichier de configuration du serveur web **Apache2** qui va nous permettre de créer notre **VirtualHost** dédié à **GLPI**.



Une fois cela fait, on peut passer à la configutation de **GLPI**

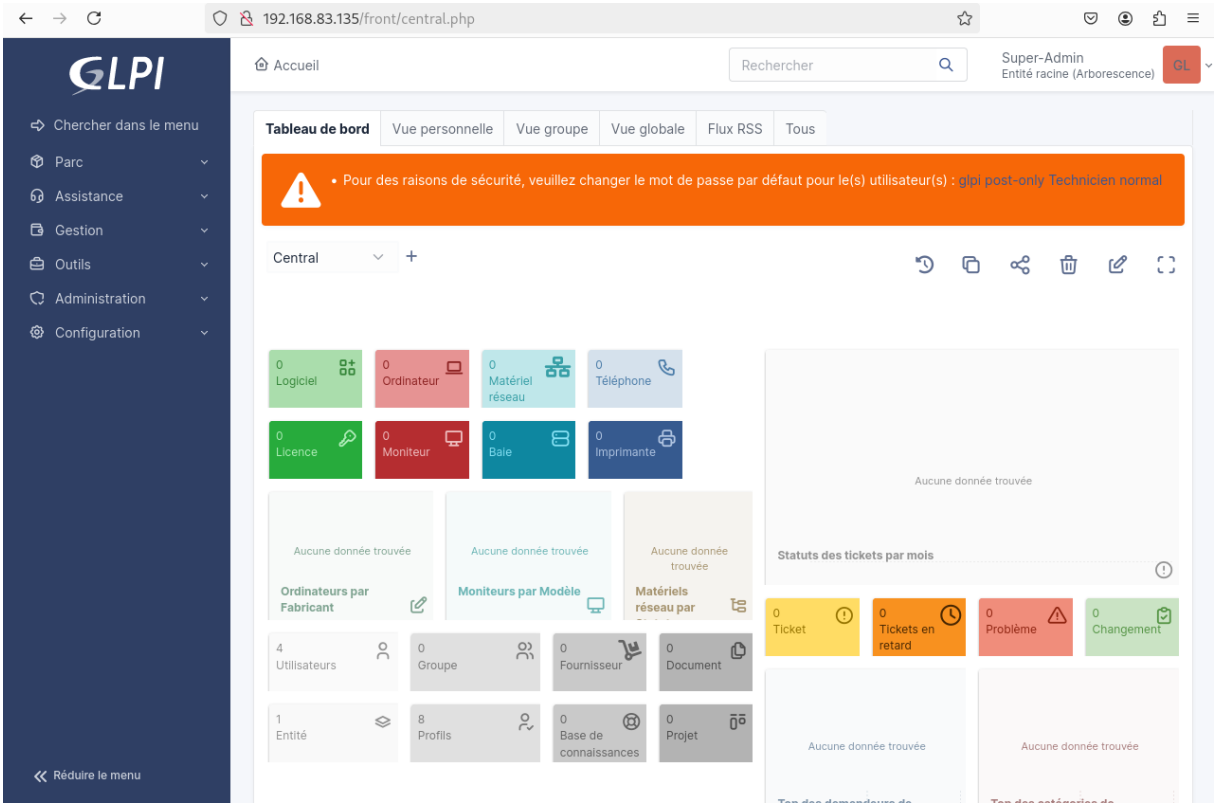


Ensuite suivre l'installation :

Localhost comme indiqué dans le nom est fait pour travailler en local.

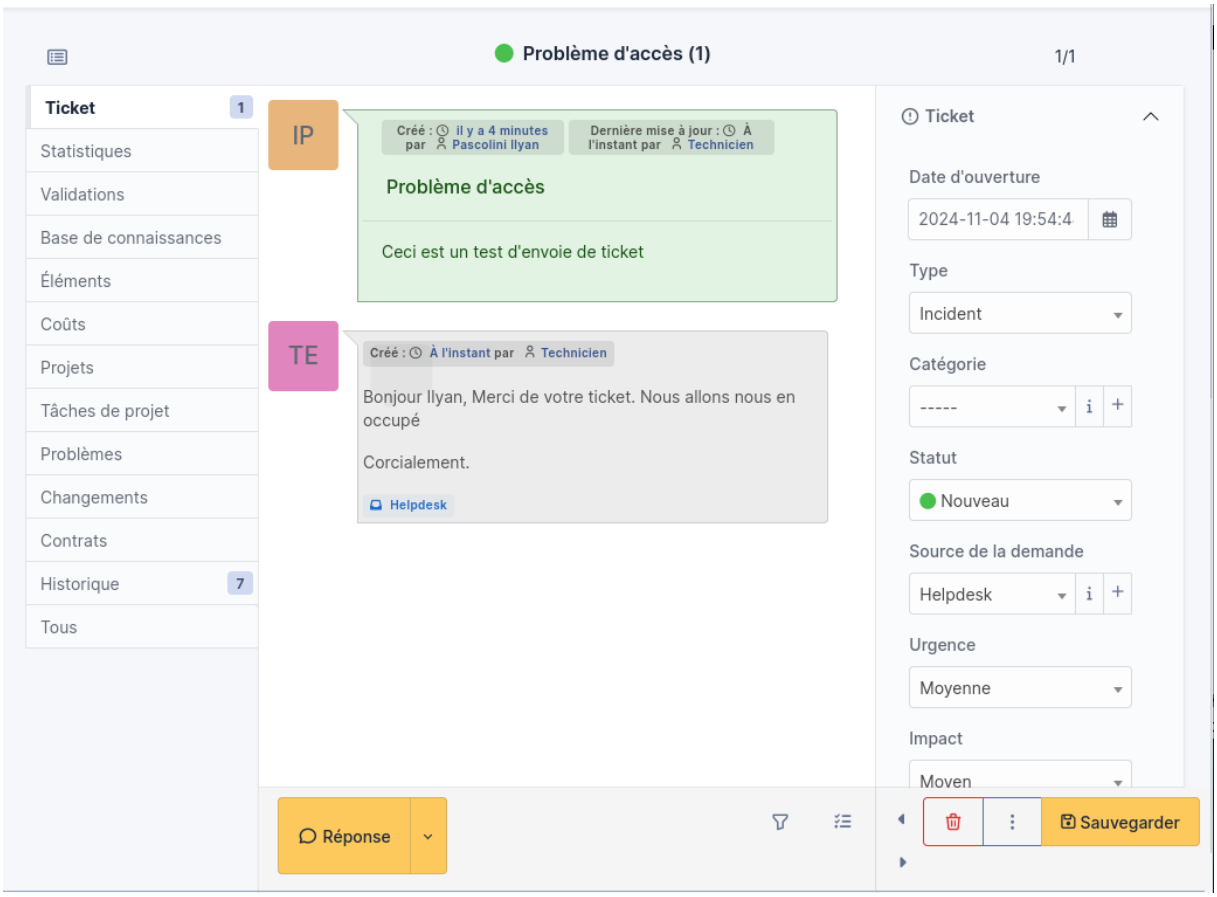
Ensuite on peut se connecter sur l'interface web de **GLPI** :

Ensuite on arrive sur le tableau de bord de **GLPI** :



GLPI est prêt à l'emploi.

Gestion de ticket :



Ici nous avons Ilyan qui nous a formulé un ticket de test pour vérifier que le système de ticket fonctionne et un technicien va s'occuper de son problème.

Avec tout cela, nous avons installé GLPI et l'avoir rendu utilisable.