Installation classique



Prérequis de GLPI

Avant d'évoquer l'installation, parlons des prérequis. GLPI a besoin d'**un serveur Web, de PHP et d'une base de données** pour fonctionner. Bien que l'installation de GLPI soit possible sur Windows Server via IIS, l'installation sur Linux est recommandée. D'une façon générale, GLPI supporte **plusieurs serveurs Web** : Apache2, Nginx, lighttpd et IIS.

Pour le reste, voici ce que vous devez savoir avant de procéder à l'**installation de GLPI 11** :

- **Version de PHP** : au minimum **PHP 8.2** pour GLPI 11.
- **Base de données**
  - MySQL 8.0 minimum
  - MariaDB 10.6 minimum

Il y aura également plusieurs **extensions PHP** à installer pour que GLPI puisse fonctionner. Pour cette démonstration, nous allons utiliser une machine sous **Debian 13** et nous allons installer dessus **Apache2, PHP 8.4 ainsi que MariaDB** (dernière version disponible dans les dépôts Debian).

Si vous avez besoin de précisions supplémentaires, vous pouvez consulter la documentation officielle :

- Les prérequis de GLPI

> **Note** : GLPI est aussi disponible sous la forme d'**images Docker**, ouvrant la porte à l'exécution sous la forme de conteneurs. Ces images représentent une nouveauté 2025 et elles sont disponibles pour les versions GLPI 10 et GLPI 11. À ce sujet, vous pouvez consulter le Docker Hub.

Préparer le serveur pour installer GLPI

Commençons par l'installation par une **mise à jour des paquets sur la machine Debian 13**. Pensez également à lui attribuer une adresse IP, un nom d'hôte (avec un enregistrement DNS pour votre application) et à effectuer la configuration du système.

La suite des opérations s'effectue directement depuis le Terminal :

```
sudo apt-get update && sudo apt-get upgrade
```

Installer le socle LAMP

La première grande étape consiste à installer les paquets du socle LAMP : **Linux Apache2 MariaDB PHP**. Sous **Debian 13 Trixie**, qui est la dernière version majeure stable de Debian, **PHP 8.4** est distribué par défaut dans les dépôts officiels. Il est à noter que l'intégration de PHP sera effectuée via PHP-FPM plutôt que l'extension PHP pour Apache, pour des raisons de performance.

Commençons par installer les trois paquets principaux :

```
sudo apt-get install apache2 php8.4-fpm mariadb-server
```

Puis, nous allons installer toutes les extensions nécessaires au bon fonctionnement de GLPI et qui ne sont pas intégrées au paquet `php8.4-common`.

```
sudo apt install php8.4-{curl,gd,intl,mysql,zip,bcmath,mbstring,xml,bz2}
```

En comparaison de GLPI 10, la version 11 impose des extensions PHP supplémentaires : `bcmath`, `mbstring` et `openssl`

Voici, à titre d'information, le rôle de chaque extension installée :

- `curl` : utilisée pour accéder à des ressources distantes (marketplace, flux RSS, etc.).

- `gd` : permet la manipulation et la génération d’images.
- `intl` : fournit les fonctions d’internationalisation (formats, locale, conversions…).
- `mysql` : gère la connexion et les opérations avec une base de données MySQL/MariaDB.
- `zlib` : nécessaire pour la compression/décompression, notamment pour les paquets gzip du marketplace et la génération de PDF.
- `bcmath` : utilisé pour générer des QR codes (calculs de précision arbitraire).
- `mbstring` : indispensable pour la gestion des chaînes de caractères multioctets (UTF-8, conversions, compatibilité internationale).
- `xml` (inclut `dom`, `simplexml`, `xmlreader`, `xmlwriter`) : fournit les outils nécessaires au traitement XML utilisés par diverses fonctions de l’application.
- `openssl` : permet la communication chiffrée (connexion HTTPS, authentification OAuth 2.0, etc.).
- `bz2` : pour le bon fonctionnement de la marketplace.

Ces commandes vont permettre de récupérer les versions de ces extensions pour PHP 8.4.

Si vous envisagez d'associer GLPI avec un annuaire LDAP comme l'Active Directory, vous devez installer l'extension LDAP de PHP. Sinon, ce n'est pas nécessaire et vous pouvez le faire par la suite, si besoin.

```
sudo apt install php8.4-ldap
```

Note : l'extension `exif` est également recommandée pour améliorer la sécurité du processus de validation des images. Elle est directement intégrée au paquet `php8.4-common`. Aucune action supplémentaire n'est requise.

Nous venons d'installer Apache2, MariaDB, PHP et un ensemble d'extensions.

Préparer une base de données pour GLPI

Nous allons préparer MariaDB pour qu'il puisse héberger la base de données de GLPI. La première action à effectuer, c'est d'exécuter la commande ci-dessous pour **effectuer le minimum syndical en matière de sécurisation de MariaDB**.

```
sudo mariadb-secure-installation


# Si vous utilisez MySQL :
sudo mysql_secure_installation
```

Vous serez invité à changer le mot de passe root, mais aussi à supprimer les utilisateurs anonymes, désactiver l'accès root à distance, etc... Tout est bien expliqué. Voici un exemple sur mon serveur pour vous guider :

```
adm_fb@SRV-GLPI11:~$ sudo mariadb-secure-installation

NOTE: MariaDB is secure by default in Debian. Running this script is
      useless at best, and misleading at worst. This script will be
      removed in a future MariaDB release in Debian. Please read
      mariadb-server.README.Debian for details.

Enter root user password or leave blank:

Enter current password for root (enter for none):
OK, successfully used password, moving on...

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
SQL executed without errors!
The operation might have been successful, or it might have not done anything.

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
SQL executed without errors!
The operation might have been successful, or it might have not done anything.

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...
SQL executed without errors!
The operation might have been successful, or it might have not done anything.
 - Removing privileges on test database...
SQL executed without errors!
The operation might have been successful, or it might have not done anything.

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
```

Ensuite, nous allons créer **une base de données dédiée pour GLPI** et celle-ci sera accessible par **un utilisateur dédié**. Hors de question d'utiliser le compte root de MariaDB : appliquons le principe de moindre privilège. Donc : une base de données = un utilisateur.

Connectez-vous à votre instance MariaDB :

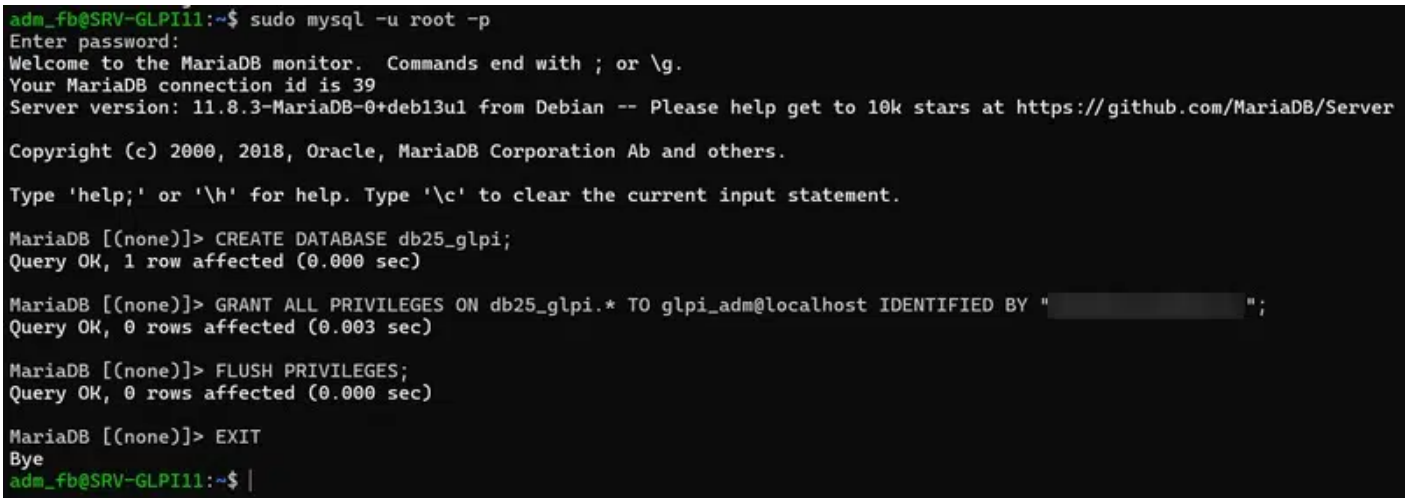
```
sudo mysql -u root -p
```

Saisissez le mot de passe root de MariaDB, que vous venez de définir à l'étape précédente.

Puis, nous allons exécuter les **requêtes SQL** ci-dessous pour **créer la base de données** `db25_glpi` ainsi que **l'utilisateur** `glpi_adm` avec le **mot de passe** `MotDePasseRobuste` (que vous personnalisez, bien sûr). Cet utilisateur aura tous les droits sur cette base de données (et uniquement sur celle-ci).

```
CREATE DATABASE db25_glpi;
GRANT ALL PRIVILEGES ON db25_glpi.* TO glpi_adm@localhost IDENTIFIED BY "MotDePasseRobuste";
FLUSH PRIVILEGES;
EXIT
```

Ce qui donne :



Voilà, la base de données est prête.

Télécharger GLPI

La prochaine étape consiste à **télécharger l'archive ".tgz"** qui contient les sources d'installation de GLPI. À partir du **GitHub de GLPI**, récupérez le lien vers la dernière version. Ici, c'est la version **GLPI 11.0.4** qui est installée.

- GitHub de GLPI

Note : je vous encourage vivement à prendre systématiquement la dernière version, et donc à consulter le GitHub de GLPI pour récupérer le lien de la dernière version, et ainsi, adapter la commande `wget` spécifiée ci-dessous (URL).

L'archive sera téléchargée dans le répertoire `/tmp` :

```
cd /tmp
wget https://github.com/glpi-project/glpi/releases/download/11.0.4/glpi-11.0.4.tgz
```

Puis, nous allons exécuter la commande ci-dessous pour **décompresser l'archive .tgz dans le répertoire** `/var/www/`, ce qui donnera le chemin d'accès `/var/www/glpi` pour GLPI.

```
sudo tar -xzvf glpi-11.0.4.tgz -C /var/www/
```

Préparer l'installation

Désormais, nous allons préparer l'**installation de GLPI 11** via la création de plusieurs répertoires et la personnalisation des permissions.

Tout d'abord, nous allons définir l'utilisateur `www-data` correspondant à **Apache2** (sur Debian/Ubuntu), en tant que **propriétaire** sur les fichiers GLPI.

```
sudo chown www-data /var/www/glpi/ -R
```

Ensuite, nous allons devoir **créer plusieurs dossiers** et sortir des données de la racine Web (`/var/www/glpi`) de manière à les stocker dans les nouveaux dossiers que nous allons créer. Ceci va permettre de faire une **installation sécurisée de GLPI, qui suit les recommandations de l'éditeur**.

- **Le répertoire** `/etc/glpi`

Commencez par **créer le répertoire** `/etc/glpi` qui va recevoir les fichiers de configuration de GLPI. Nous donnons des autorisations à `www-data` sur ce répertoire car il a besoin de pouvoir y accéder.

```
sudo mkdir /etc/glpi
sudo chown www-data /etc/glpi/
```

Puis, nous allons déplacer le répertoire sensible `config` de GLPI vers ce nouveau dossier :

```
sudo mv /var/www/glpi/config /etc/glpi
```

- **Le répertoire** `/var/lib/glpi`

Répétons la même opération avec la création du répertoire `/var/lib/glpi` :

```
sudo mkdir /var/lib/glpi
sudo chown www-data /var/lib/glpi/
```

Dans lequel nous déplaçons également le dossier `files` qui contient la majorité des fichiers de GLPI : CSS, plugins, etc.

```
sudo mv /var/www/glpi/files /var/lib/glpi
```

- **Le répertoire `/var/log/glpi`**

Terminons par la création du répertoire `/var/log/glpi` destiné à stocker les journaux de GLPI. Toujours sur le même principe :

```
sudo mkdir /var/log/glpi
sudo chown www-data /var/log/glpi
```

Nous n'avons rien à déplacer dans ce répertoire.

- **Créer les fichiers de configuration**

Nous devons configurer GLPI pour qu'il sache où aller chercher les données. Autrement dit, nous allons déclarer les nouveaux répertoires fraichement créés. Nous allons créer ce premier fichier :

```
sudo nano /var/www/glpi/inc/downstream.php
```

Afin d'ajouter le contenu ci-dessous qui indique le chemin vers le **répertoire de configuration** :

```
<?php
define('GLPI_CONFIG_DIR', '/etc/glpi/');
if (file_exists(GLPI_CONFIG_DIR . '/local_define.php')) {
    require_once GLPI_CONFIG_DIR . '/local_define.php';
}
```

Ensuite, nous allons créer ce second fichier :

```
sudo nano /etc/glpi/local_define.php
```

Afin d'ajouter le contenu ci-dessous permettant de **déclarer deux variables** permettant de préciser les chemins vers **les répertoires `files` et `log`**. Pour rappel, ces deux répertoires ont été préparés précédemment.

```
<?php
define('GLPI_VAR_DIR', '/var/lib/glpi/files');
define('GLPI_LOG_DIR', '/var/log/glpi');
```

GLPI prend en charge de nombreuses variables pour personnaliser l'emplacement des répertoires, bien au-delà des deux variables spécifiées ci-dessus. Par exemple, la variable `GLPI_CACHE_DIR` sert à spécifier un emplacement personnalisé pour le stockage du cache.

Voilà, cette étape est terminée.

Configurer Apache2 pour GLPI

Passons à la configuration du serveur web Apache2. Nous allons créer un nouveau fichier de configuration qui va permettre de configurer le VirtualHost dédié à GLPI. Dans mon cas, le fichier s'appelle `support.it-connectlab.fr.conf` en référence au nom de domaine choisi pour accéder à GLPI : **support.it-connectlab.fr**. L'idéal étant d'avoir un nom de domaine (même interne) pour accéder à GLPI afin de pouvoir positionner un certificat TLS/SSL par la suite.

```
sudo nano /etc/apache2/sites-available/support.conf
```

Ce qui donne la configuration suivante (selon le modèle officiel de la documentation) :

```
<VirtualHost *:80>
    ServerName support.domain.tld

    DocumentRoot /var/www/glpi/public

    # If you want to place GLPI in a subfolder of your site (e.g. your virtual host is serving multiple applications),
    # you can use an Alias directive. If you do this, the DocumentRoot directive MUST NOT target the GLPI directory itself.
    # Alias "/glpi" "/var/www/glpi/public"

    <Directory /var/www/glpi/public>
        Require all granted

        RewriteEngine On

        # Ensure authorization headers are passed to PHP.
        # Some Apache configurations may filter them and break usage of API, CalDAV, ...
        RewriteCond %{HTTP:Authorization} ^(.+)$
```

```
        RewriteRule .* - [E=HTTP_AUTHORIZATION:%{HTTP:Authorization}]

        # Redirect all requests to GLPI router, unless file exists.
        RewriteCond %{REQUEST_FILENAME} !-f
        RewriteRule ^(.*)$ index.php [QSA,L]

    </Directory>
</VirtualHost>
```

Quand la configuration est prête, enregistrez le fichier.

Puis, nous allons **activer ce nouveau site dans Apache2** :

```
sudo a2ensite support.conf
```

Nous en profitons également pour **désactiver le site par défaut** car il est inutile :

```
sudo a2dissite 000-default.conf
```

Nous allons aussi **activer le module** `rewrite` (nécessaire pour activer la prise en charge des règles de réécriture) car nous l'avons utilisé dans le fichier de configuration du VirtualHost (via les directives `RewriteCond` / `RewriteRule`).

```
sudo a2enmod rewrite
```

Il ne reste plus qu'à **redémarrer le service Apache2** :

```
sudo systemctl restart apache2
```

Utiliser PHP8.4-FPM avec Apache2

Pour utiliser PHP en tant que moteur de scripts avec Apache2, il y a deux possibilités : **utiliser le module PHP pour Apache2 (libapache2-mod-php8.4) ou utiliser PHP-FPM**. Il est **recommandé d'utiliser PHP-FPM** car il est plus performant et se présente comme un service indépendant. Dans l'autre mode, chaque processus Apache2 exécute son propre moteur de scripts PHP.

Pour rappel, nous avons déjà fait le choix d'utiliser PHP-FPM : le paquet `php8.4-fpm` a été installé précédemment, au même moment que ceux pour Apache2 et MariaDB. Néanmoins, nous devons configurer l'intégration de PHP-FPM avec Apache.

Commençons par activer deux modules dans Apache et la configuration de PHP-FPM, avant de recharger Apache2 :

```
sudo a2enmod proxy_fcgi setenvif
sudo a2enconf php8.4-fpm
sudo systemctl reload apache2
```

Pour **configurer PHP-FPM pour Apache2**, nous n'allons pas éditer le fichier `/etc/php/8.4/apache2/php.ini`. À la place, le fichier que vous devez éditer est celui-ci :

```
sudo nano /etc/php/8.4/fpm/php.ini
```

Dans ce fichier, recherchez l'option `session.cookie_httponly` (CTRL+W avec nano) et indiquez la valeur `on` pour l'activer, afin de protéger les cookies de GLPI.

```
; Whether or not to add the httpOnly flag to the cookie, which makes it
; inaccessible to browser scripting languages such as JavaScript.
; https://php.net/session.cookie-httponly
session.cookie_httponly = on
```

Toujours pour renforcer la sécurité, configurez la directive `session.cookie_samesite` avec la valeur `Lax` conseillée par la documentation de GLPI. Elle contrôle la façon dont le navigateur envoie le cookie de session, ce qui bloque notamment certaines **attaques CSRF** (Cross-Site Request Forgery).

```
; Add SameSite attribute to cookie to help mitigate Cross-Site Request Forgery (CSRF/XSRF)
; Current valid values are "Strict", "Lax" or "None". When using "None",
; make sure to include the quotes, as `none` is interpreted like `false` in ini files.
; https://tools.ietf.org/html/draft-west-first-party-cookies-07
session.cookie_samesite = Lax
```

Enregistrez le fichier quand c'est fait. Par la suite, vous pourriez être amené à effectuer d'autres modifications, notamment pour augmenter la taille des uploads sur GLPI (`upload_max_filesize` limité à 2 Mo par défaut), etc. Je pense aussi à la directive `session.cookie_secure` que nous passerons sur `on` une fois GLPI accessible en HTTPS.

Pour appliquer les modifications, nous devons redémarrer PHP-FPM :

```
sudo systemctl restart php8.4-fpm.service
```

Pour finir, nous devons **modifier notre VirtualHost** pour préciser à Apache2 que PHP-FPM doit être utilisé pour les fichiers PHP. Cette configuration vise à pointer vers le socket de PHP-FPM pour traiter les fichiers avec l'extension `.php`. Editez le fichier `support.it-connectlab.fr.conf` pour

ajouter ceci :

```
<FilesMatch \.php$>
    SetHandler "proxy:unix:/run/php/php8.4-fpm.sock|fcgi://localhost/"
</FilesMatch>
```

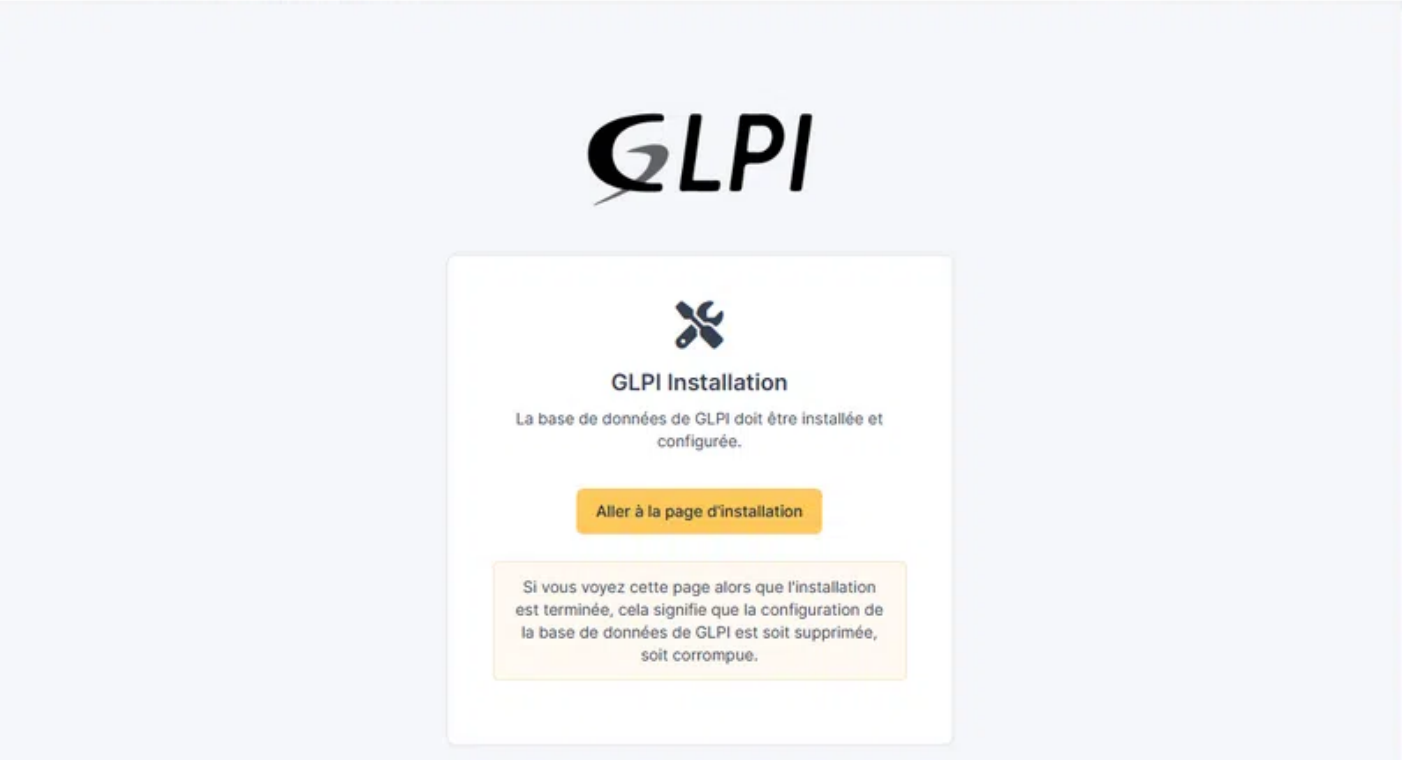
Quand c'est fait, relancer Apache2 :

```
sudo systemctl restart apache2
```

Voilà, tout est prêt ! Il ne reste plus qu'à installer GLPI !

Installation de GLPI

Pour effectuer l'installation de GLPI, nous devons utiliser un navigateur Web afin d'accéder à l'adresse du GLPI. Il s'agit de l'adresse déclarée dans le fichier de configuration Apache2 (`ServerName`).



Si vous avez suivi toutes les étapes correctement, vous devriez obtenir la page visible ci-dessous. Cliquez simplement sur le bouton au centre.

Nous allons commencer par choisir la langue.



Puisqu'il s'agit d'une nouvelle installation, nous cliquons sur "**Installer**".



Étape importante : **GLPI vérifie la configuration de notre serveur** pour déterminer si tous les prérequis sont respectés. Tout est bon, donc nous pouvons continuer. Référez-vous à la colonne des résultats.



À l'étape suivante, nous devons renseigner les informations pour se connecter à la base de données. Nous indiquons `localhost` en tant que serveur SQL puisque MariaDB est installé en local, sur le même serveur que GLPI. Puis, nous indiquons notre utilisateur `glpi_adm` et le mot de passe associé.



Après avoir cliqué sur "**Continuer**", nous devons choisir la base de données `db25_glpi` créée précédemment.



Patientez...



Si l'initialisation s'est correctement déroulée, vous devriez obtenir ce résultat :



Suivez les dernières étapes qui n'ont pas de réel impact. Le plus dur est fait !

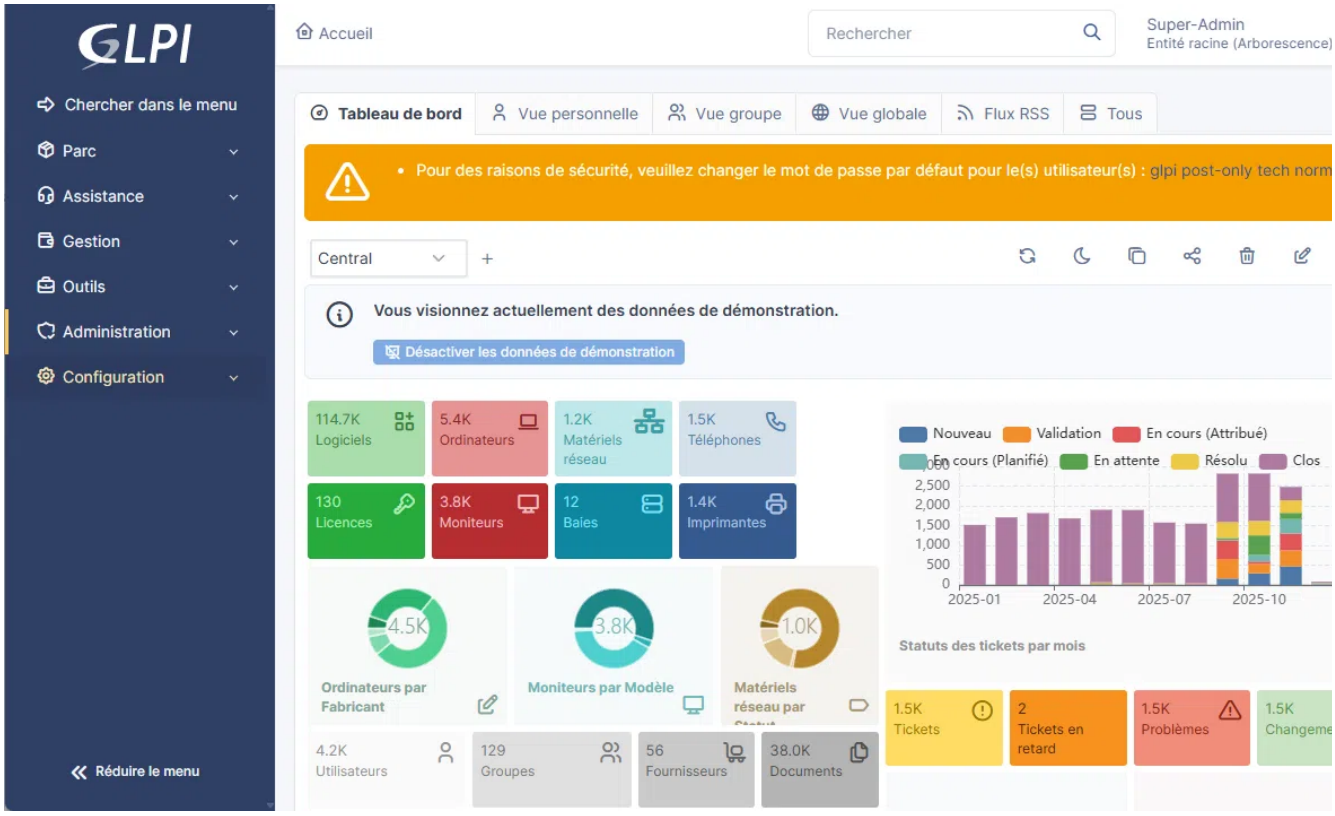


Félicitations, vous venez d'installer GLPI ! Comme le précise la dernière étape, le compte **administrateur** par défaut est "**glpi/glpi**" !



Nous allons donc nous connecter avec le compte "**glpi**" et le mot de passe "**glpi**".

Bienvenue sur votre nouveau serveur GLPI ! Il y a déjà des informations issues des données de démonstration, vous n'avez qu'à cliquer sur le bouton bleu nommé "**Désactiver les données de démonstration**" pour que toutes les sections soient vierges comme attendu.



Même si l'installation est terminée, nous avons encore quelques **actions indispensables** à réaliser pour la finaliser :

- Changer le mot de passe de tous les comptes par défaut (cliquez sur les liens situés dans l'encadré orange)
- Supprimer le fichier `install.php` puisqu'il n'est plus nécessaire et représente un risque (possibilité de relancer l'installation)

```
sudo rm /var/www/glpi/install/install.php
```

Voilà, c'est fait. Désormais, votre GLPI est prêt à être utilisé et configuré (création d'utilisateurs, de catégories, de tickets, etc...).

Revision #2
Created 2026-06-20 12:40:55 UTC by clement-derouet
Updated 2026-06-20 12:52:15 UTC by clement-derouet