PROJET PROFESSIONNEL ENCADRÉ

Site Web de couverture avec GLPI

Juin 2026

FOSSEY Florent

IRIS Médiaschool

SOMMAIRE

INTRODUCTION

Dans le cadre de mon Projet Professionnel Encadré (PPE) en BTS SIO option SISR, j'ai réalisé le déploiement complet d'une infrastructure de ticketing accessible via le web. Ce projet s'inscrit dans un contexte professionnel réaliste : une entreprise souhaite permettre à ses employés de signaler des incidents informatiques via un formulaire web simple, sans avoir accès à l'interface d'administration GLPI.

Les objectifs techniques du projet sont les suivants :

- Déployer un serveur web Apache sur Ubuntu Server 24.04 LTS
- Créer un site vitrine avec une page de contact pour signaler des incidents
- Installer et configurer GLPI, logiciel open source de gestion de tickets
- Connecter le formulaire web à GLPI via son API REST pour créer automatiquement des tickets
- Gérer plusieurs sites sur le même serveur via les VirtualHosts Apache
- Mettre en place une supervision en temps réel avec Zabbix 7.2

Environnement technique

Machine virtuelle Ubuntu Server 24.04 LTS sur VMware Workstation — IP : 192.168.85.128
Serveur web : Apache 2.4.58 — Base de données : MariaDB 10.x — Langage : PHP 8.3
Ticketing : GLPI 10.0.10 — Supervision : Zabbix 7.2

ARCHITECTURE RÉSEAU

2.1 Présentation de l'infrastructure

L'ensemble du projet est hébergé sur une unique machine virtuelle VMware en mode NAT. Le PC hôte Windows communique avec la VM via le réseau virtuel VMnet8. La résolution des noms de domaine locaux est assurée par les fichiers hosts des deux machines.

Composant	Détail
Hyperviseur	VMware Workstation
OS de la VM	Ubuntu Server 24.04 LTS
IP de la VM	192.168.85.128 /24
Passerelle VMware NAT	192.168.85.2
Interface réseau	ens33
Serveur web	Apache 2.4.58
Base de données	MariaDB 10.x
Langage serveur	PHP 8.3
Outil de ticketing	GLPI 10.0.10
Supervision	Zabbix 7.2

2.2 Schéma réseau et flux de communication

Le schéma ci-dessous illustre le flux complet d'une demande : l'utilisateur remplit le formulaire sur le site vitrine, le script PHP contacte l'API GLPI en interne, et un ticket est créé automatiquement dans la base de données.

Trois noms de domaine locaux sont utilisés, définis dans les fichiers hosts :

- http://glpi.local → interface d'administration GLPI
- http://www.monsite.local → site vitrine avec formulaire de contact
- http://zabbix.local/zabbix → interface de supervision Zabbix

INSTALLATION D'APACHE, PHP ET MARIADB

3.1 Installation des paquets

Avant toute chose, le système est mis à jour. Ensuite, Apache, PHP avec toutes ses extensions requises par GLPI, et MariaDB sont installés :

```
sudo apt update && sudo apt upgrade -y
sudo apt install apache2 -y
sudo apt install php php-curl php-json php-mysqli php-mbstring \
php-xml php-gd php-intl php-zip php-bz2 php-ldap libapache2-mod-php -y
sudo apt install mariadb-server -y
```

Extension PHP	Rôle
php-mysqli	Communication avec MariaDB/MySQL
php-curl	Requêtes HTTP (API REST)
php-xml / php-mbstring	Traitement des données XML et chaînes multi-octets
php-gd	Génération d'images (graphiques GLPI)
php-intl	Internationalisation et formatage des dates
php-zip / php-bz2	Gestion des archives (plugins GLPI)
php-ldap	Authentification LDAP/Active Directory

3.2 Vérification des services

Les services Apache et MariaDB sont activés pour démarrer automatiquement au lancement du serveur :

```
sudo systemctl enable apache2 mariadb
sudo systemctl start apache2 mariadb
# Vérifier que tout est actif
sudo systemctl status apache2
sudo systemctl status mariadb
```

CONFIGURATION DE LA BASE DE DONNÉES SQL

4.1 Sécurisation de MariaDB

Après l'installation, MariaDB est sécurisé avec le script fourni. Ce script supprime les utilisateurs anonymes, désactive la connexion root distante et supprime la base de test :

```
sudo mysql_secure_installation
# Répondre : n (pas de plugin mot de passe)
# Puis : y, y, y, y pour tout sécuriser
```

4.2 Création de la base et de l'utilisateur GLPI

Une base de données dédiée est créée pour GLPI avec un utilisateur MariaDB appliquant le principe du moindre privilège. Le mot de passe utilisé est **Glpi2024!** (le mot de passe initial ayant échoué lors de la première tentative d'installation).

```
sudo mysql

CREATE DATABASE glpi CHARACTER SET utf8 COLLATE utf8_unicode_ci;
CREATE USER 'glpiuser'@'localhost' IDENTIFIED BY 'Glpi2024!';
GRANT ALL PRIVILEGES ON glpi.* TO 'glpiuser'@'localhost';
FLUSH PRIVILEGES;
EXIT;
```

Sécurité

L'utilisateur 'glpiuser' n'a accès qu'à la base 'glpi' et uniquement depuis localhost. Il n'a aucun droit sur les autres bases de données du serveur.

4.3 Vérification de la base

```
sudo mysql
SHOW DATABASES; -- Doit afficher 'glpi' dans la liste
SELECT user, host FROM mysql.user WHERE user = 'glpiuser';
-- Doit afficher : glpiuser | localhost
EXIT;
```

INSTALLATION ET CONFIGURATION DE GLPI

5.1 Téléchargement et déploiement

GLPI 10.0.10 est téléchargé depuis le dépôt officiel GitHub, extrait et déplacé dans le répertoire web d'Apache. Les droits sont ensuite attribués à l'utilisateur www-data :

```
cd /tmp
wget https://github.com/glpi-project/glpi/releases/download/10.0.10/glpi-10.0.10.tgz
tar -xzf glpi-10.0.10.tgz
sudo mv glpi /var/www/glpi
sudo chown -R www-data:www-data /var/www/glpi
sudo chmod -R 755 /var/www/glpi
```

5.2 Installation via la console CLI

Contrairement aux versions antérieures, GLPI 10 ne fournit plus de page install.php accessible via navigateur. L'installation se fait obligatoirement via la console PHP. Les options **--force** et **--reconfigure** ont été nécessaires pour écraser une configuration existante :

```
sudo php /var/www/glpi/bin/console db:install \
--db-host=localhost \
--db-name=glpi \
--db-user=glpiuser \
--db-password=Glpi2024! \
--no-interaction \
--force \
--reconfigure
```

Configuration des timezones pour éviter des erreurs dans GLPI :

```
sudo mysql_tzinfo_to_sql /usr/share/zoneinfo | sudo mysql -u root mysql
sudo mysql -u root -e "GRANT SELECT ON mysql.time_zone_name TO
'glpiuser'@'localhost'; FLUSH PRIVILEGES;"
sudo php /var/www/glpi/bin/console database:enable_timezones
```

5.3 Première connexion à GLPI

Les identifiants par défaut sont à modifier immédiatement pour des raisons de sécurité :

Compte	Login	Mot de passe par défaut	Action
Administrateur	glpi	glpi	Changer immédiatement
Technicien	tech	tech	Changer ou désactiver
Utilisateur	normal	normal	Changer ou désactiver
Post-only	post-only	postonly	Changer ou désactiver

CONFIGURATION APACHE — VIRTUALHOSTS ET DNS

Apache permet d'héberger plusieurs sites sur un même serveur grâce aux VirtualHosts. Chaque VirtualHost associe un nom de domaine à un dossier et une configuration spécifique. Dans ce projet, deux VirtualHosts principaux sont créés.

6.1 VirtualHost pour GLPI

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

<VirtualHost *:80>
ServerName glpi.local
DocumentRoot /var/www/glpi/public
<Directory /var/www/glpi/public>
Options Indexes FollowSymLinks
AllowOverride All
Require all granted
</Directory>
ErrorLog ${APACHE_LOG_DIR}/glpi_error.log
CustomLog ${APACHE_LOG_DIR}/glpi_access.log combined
</VirtualHost>
```

Le DocumentRoot pointe vers /var/www/glpi/public (et non /var/www/glpi) car GLPI 10 sépare les fichiers publics des fichiers privés pour des raisons de sécurité.

6.2 VirtualHost pour le site vitrine

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

<VirtualHost *:80>
ServerName www.monsite.local
DocumentRoot /var/www/site
<Directory /var/www/site>
AllowOverride All
Require all granted
</Directory>
</VirtualHost>
```

Activation des VirtualHosts et rechargement d'Apache :

```
sudo a2ensite glpi.conf
sudo a2ensite site.conf
sudo a2enmod rewrite status
sudo a2dissite 000-default.conf
sudo systemctl reload apache2
sudo apache2ctl -S # Vérifier les deux VirtualHosts actifs
```

6.3 Fichier .htaccess pour GLPI

GLPI 10 nécessite un fichier .htaccess dans son dossier public pour que la réécriture des URLs fonctionne correctement. Sans ce fichier, toutes les pages de GLPI retournent une erreur 404.

```
sudo nano /var/www/glpi/public/.htaccess

RewriteEngine On
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^(.*)$ index.php [QSA,L]
```

6.4 Configuration DNS local (fichiers hosts)

Sur le PC hôte Windows — C:\Windows\System32\drivers\etc\hosts :

```
192.168.85.128 glpi.local
192.168.85.128 www.monsite.local
192.168.85.128 zabbix.local
```

Sur la VM Ubuntu — /etc/hosts :

```
127.0.0.1 glpi.local
127.0.0.1 www.monsite.local
127.0.0.1 zabbix.local
```

CONFIGURATION DE L'API REST GLPI

L'API REST de GLPI permet à des applications externes (comme notre formulaire PHP) de créer des tickets, consulter le parc informatique, ou gérer des utilisateurs. Elle utilise le protocole HTTP avec des tokens d'authentification.

8.1 Activation de l'API REST

Dans l'interface GLPI : Configuration → Général → onglet API :

- Activer l'API REST → Oui
- Activer la connexion avec les tokens externes → Oui
- Cliquer sur Sauvegarder

8.2 Création du client API et génération du token applicatif

Dans Configuration → API → Ajouter un client API :

Paramètre	Valeur
Nom	monsite (nom du client)
Actif	Oui
Adresse IP début	Vide (pas de restriction)
Adresse IP fin	Vide (pas de restriction)
App-Token	Généré automatiquement — à noter

App-Token
Le App-Token identifie l'application cliente. Il est généré une seule fois et doit être conservé. Dans ce projet, il est intégré dans le script PHP submit_ticket.php.

8.3 Génération du token utilisateur

Le token utilisateur est généré dans Administration → Utilisateurs → glpi → onglet Préférences → section Jeton d'API → Régénérer.

Token	Rôle	Où l'utiliser
App-Token	Identifie l'application cliente	Header HTTP : App-Token
User-Token	Identifie l'utilisateur GLPI	Header HTTP : Authorization: user_token

8.4 Test de l'API en ligne de commande

Avant d'intégrer les tokens dans le script PHP, l'API a été testée directement en ligne de commande avec curl pour vérifier le bon fonctionnement de l'authentification :

```
curl -X GET "http://glpi.local/apirest.php/initSession" \
-H "Content-Type: application/json" \
-H "App-Token: votre_app_token" \
-H "Authorization: user_token votre_user_token"

# Réponse valide attendue :
{"session_token":"abc123...xyz"}
```

07

DÉVELOPPEMENT DU SCRIPT PHP — INTÉGRATION GLPI

9.1 Fonctionnement du script

Le script submit_ticket.php est le cœur de l'intégration. Lorsqu'un utilisateur soumet le formulaire, le script effectue les opérations suivantes :

Étape	Action
1	Récupération et nettoyage des données du formulaire (protection XSS)
2	Ouverture d'une session API GLPI avec les deux tokens
3	Création du ticket via une requête HTTP POST vers l'API
4	Fermeture de la session API
5	Affichage du résultat à l'utilisateur (succès ou erreur)

9.2 Code source — submit_ticket.php

```
<?php
define('GLPI_URL', 'http://glpi.local/apirest.php');
define('APP_TOKEN', 'votre_app_token');
define('USER_TOKEN', 'votre_user_token');

$nom = htmlspecialchars($_POST['nom'] ?? '');
$email = htmlspecialchars($_POST['email'] ?? '');
$urgence = intval($_POST['urgence'] ?? 3);
$description = htmlspecialchars($_POST['description'] ?? '');
if (empty($nom) || empty($description)) die('Champs obligatoires manquants.');

// Étape 1 : Ouverture de session
$ch = curl_init(GLPI_URL . '/initSession');
curl_setopt_array($ch, [CURLOPT_RETURNTRANSFER => true,
CURLOPT_HTTPHEADER => ['Content-Type: application/json',
'App-Token: '.APP_TOKEN, 'Authorization: user_token '.USER_TOKEN]]);
$session = json_decode(curl_exec($ch), true);
curl_close($ch);
$session_token = $session['session_token'];

// Étape 2 : Création du ticket
$ticket = ['input' => ['name' => "Incident signalé par $nom",
'content' => "Nom : $nom\nEmail : $email\n\nDescription :\n$description",
'type' => 1, 'status' => 1, 'urgency' => $urgence,
'impact' => 3, 'priority' => $urgence]];
$ch = curl_init(GLPI_URL . '/Ticket');
curl_setopt_array($ch, [CURLOPT_RETURNTRANSFER => true,
CURLOPT_CUSTOMREQUEST => 'POST', CURLOPT_POSTFIELDS => json_encode($ticket),
CURLOPT_HTTPHEADER => ['Content-Type: application/json',
'App-Token: '.APP_TOKEN, 'Session-Token: '.$session_token]]);
$result = json_decode(curl_exec($ch), true);
curl_close($ch);

// Étape 3 : Fermeture + affichage
$ch = curl_init(GLPI_URL . '/killSession');
curl_setopt_array($ch, [CURLOPT_RETURNTRANSFER => true,
CURLOPT_HTTPHEADER => ['App-Token: '.APP_TOKEN, 'Session-Token: '.$session_token]]);
curl_exec($ch); curl_close($ch);
if (isset($result['id'])) echo '<h2>Ticket #'.$result['id'].' créé !</h2>';
else echo '<h2>Erreur</h2><pre>'.print_r($result, true).'</pre>';
?>
```

08

MISE EN PLACE DE LA SUPERVISION AVEC ZABBIX

15.1 Installation de Zabbix 7.2

```
wget https://repo.zabbix.com/zabbix/7.2/release/ubuntu/pool/main/z/zabbix-release/
zabbix-release_latest_7.2+ubuntu24.04_all.deb
sudo dpkg -i zabbix-release_latest_7.2+ubuntu24.04_all.deb && sudo apt update
sudo apt install -y zabbix-server-mysql zabbix-frontend-php zabbix-apache-conf \
zabbix-sql-scripts zabbix-agent
```

15.2 Création de la base de données Zabbix

```
sudo mysql
CREATE DATABASE zabbix CHARACTER SET utf8mb4 COLLATE utf8mb4_bin;
CREATE USER 'zabbix'@'localhost' IDENTIFIED BY 'zabbix123';
GRANT ALL PRIVILEGES ON zabbix.* TO 'zabbix'@'localhost';
FLUSH PRIVILEGES; EXIT;
```

15.3 Import du schéma SQL (chemin corrigé)

Attention : le chemin du script SQL diffère de la documentation officielle sur cette installation. La commande find a permis de localiser le bon chemin :

```
# Localiser le fichier :
find /usr/share/zabbix* -name '*.sql.gz' 2>/dev/null
# Résultat : /usr/share/zabbix/sql-scripts/mysql/server.sql.gz

# Import du schéma :
zcat /usr/share/zabbix/sql-scripts/mysql/server.sql.gz | mysql -u zabbix -p'zabbix123' zabbix
```

15.4 Configuration et démarrage des services

```
# Éditer /etc/zabbix/zabbix_server.conf
# Trouver DBPassword= avec Ctrl+W, décommenter et mettre : DBPassword=zabbix123

sudo systemctl restart zabbix-server zabbix-agent apache2
sudo systemctl enable zabbix-server zabbix-agent
sudo systemctl status zabbix-server
```

15.5 Interface web et modèle Apache

Accéder à http://zabbix.local/zabbix — Identifiants par défaut : Admin / zabbix. Puis dans Data collection → Hosts → Zabbix server → Templates → rechercher 'Apache by Zabbix agent' → Add → Update.

```
sudo a2enmod status
sudo systemctl restart apache2
```

15.6 Test de détection d'incident

Pour valider la supervision, on simule une panne en arrêtant manuellement Apache. Zabbix détecte l'incident automatiquement en moins de 2 minutes :

```
sudo systemctl stop apache2 # Simuler la panne
# Monitoring → Problems → alertes apparaissent :
# Apache: Process is not running (sévérité Haut)
# Apache: Service is down (sévérité Moyen)

sudo systemctl start apache2 # Relancer Apache
# → Les alertes passent automatiquement en statut Résolu
```

TESTS ET VALIDATION END-TO-END

10.1 Scénario de test complet

Le test de bout en bout consiste à simuler le parcours complet d'un utilisateur, depuis le formulaire web jusqu'à l'apparition du ticket dans GLPI.

Étape	Action	Résultat attendu
1	Accéder à http://www.monsite.local	Page d'accueil du portail s'affiche
2	Cliquer sur 'Signaler un problème'	Formulaire de contact s'affiche
3	Remplir et envoyer le formulaire	Message de confirmation avec numéro de ticket
4	Se connecter à http://glpi.local	Tableau de bord GLPI s'affiche
5	Aller dans Assistance → Tickets	Le nouveau ticket apparaît dans la liste

10.2 Vérification du ticket dans GLPI

Dans l'interface GLPI, sous Assistance → Tickets, le ticket créé par le formulaire est visible avec toutes les informations saisies : nom de l'utilisateur, email, niveau d'urgence et description du problème.

10.3 Validation de la supervision Zabbix

La supervision a été validée en simulant une panne Apache. Zabbix a détecté automatiquement les alertes suivantes en moins de 2 minutes, puis les a résolues dès le redémarrage du service :

Alerte	Sévérité	Statut final
Apache: Process is not running	Haut (High)	RÉSOLU après redémarrage Apache
Apache: Service is down	Moyen (Average)	RÉSOLU en 1 seconde
Apache: Service has been restarted	Information	PROBLEM (normal après redémarrage)

DIFFICULTÉS RENCONTRÉES

Ce projet a présenté plusieurs difficultés techniques. Voici un bilan détaillé de chaque problème rencontré et la solution appliquée.

Difficulté 1 — Mot de passe MariaDB glpiuser incorrect

	Détail
Symptôme	Access denied for user glpiuser@localhost (using password: YES)
Cause identifiée	Le mot de passe MotDePasse123! avait échoué silencieusement lors de la création
Solution appliquée	Connexion root via sudo mysql, puis ALTER USER avec le nouveau mot de passe Glpi2024!
Leçon	Toujours tester la connexion avec mysql -u user -p avant de lancer l'installation GLPI

Difficulté 2 — Fichier install.php absent dans GLPI 10

	Détail
Symptôme	http://glpi.local/install/install.php → erreur 404 Not Found
Cause identifiée	GLPI 10 a supprimé l'installateur web au profit d'une installation CLI
Solution appliquée	Utilisation : php bin/console db:install avec --force --reconfigure
Leçon	Toujours lire les notes de version lors d'un changement de version majeure

Difficulté 3 — Erreur 404 sur toutes les pages GLPI

	Détail
Symptôme	http://glpi.local affiche 404 Not Found même après installation
Cause identifiée	Le fichier .htaccess était absent du dossier /var/www/glpi/public
Solution appliquée	Création manuelle du .htaccess avec les règles RewriteEngine
Explication	GLPI 10 utilise le pattern Front Controller : toutes les URLs passent par index.php

Difficulté 4 — API REST désactivée

	Détail
Symptôme	curl retourne ["ERROR","API disabled"]
Cause identifiée	L'API REST n'était pas activée dans la configuration GLPI
Solution appliquée	Configuration → Général → API → Activer l'API REST : Oui + Jeton externe : Oui
Leçon	Toujours vérifier l'activation de l'API avant de tester les tokens

Difficulté 5 — glpi.local non résolu depuis la VM

	Détail
Symptôme	curl: (6) Could not resolve host: glpi.local
Cause identifiée	Le fichier /etc/hosts de la VM ne contenait pas les entrées DNS locales

Solution appliquée	Ajout de 127.0.0.1 glpi.local et 127.0.0.1 www.monsite.local dans /etc/hosts
Leçon	La VM ne lit pas le fichier hosts Windows, il faut le configurer séparément

Difficulté 6 — Chemin SQL Zabbix incorrect

	Détail
Symptôme	gzip: /usr/share/zabbix-sql-scripts/mysql/server.sql.gz: No such file or directory
Cause identifiée	Le chemin réel est /usr/share/zabbix/sql-scripts/ sur cette installation
Solution appliquée	find /usr/share/zabbix* -name "*.sql.gz" pour localiser le bon chemin
Leçon	Toujours vérifier les chemins avec find avant d'exécuter les commandes d'import SQL

COMPÉTENCES MOBILISÉES

12.1 Compétences techniques

Domaine	Compétence mise en œuvre
Virtualisation	Création et paramétrage d'une VM VMware (CPU, RAM, réseau NAT)
Administration Linux	Gestion des paquets apt, services systemctl, droits chmod/chown
Serveur web Apache	VirtualHosts, module rewrite, .htaccess, journaux d'erreurs
Base de données SQL	Création de BDD et utilisateur MariaDB, gestion des privilèges
Installation GLPI	Déploiement, installation CLI, configuration initiale
API REST	Authentification par tokens, requêtes HTTP GET/POST avec cURL
PHP	Développement d'un script de création de ticket via API REST
HTML/CSS	Création de formulaire web responsive et page d'accueil
Réseau	Configuration IP, table de routage, résolution DNS locale
Supervision	Installation Zabbix 7.2, templates Apache, déclencheurs d'alertes

12.2 Lien avec le référentiel BTS SIO — SISR

Activité type	Ce qui a été réalisé
A1.1 — Analyse de l'existant	Étude de l'architecture nécessaire avant déploiement
A1.2 — Déploiement de services	Installation et configuration Apache, MariaDB, GLPI, Zabbix
A2.1 — Exploitation	Administration du serveur, gestion des logs, tests
A2.3 — Sécurité	Gestion des tokens API, droits utilisateurs SQL et système
A4.1 — Services réseaux	Configuration DNS local, VirtualHosts, architecture NAT

CONCLUSION

Ce PPE m'a permis de déployer de A à Z une infrastructure complète et fonctionnelle, en partant de la création de la machine virtuelle jusqu'à la validation du flux de création automatique de tickets et de la supervision en temps réel. Chaque étape a nécessité de comprendre le rôle de chaque composant et leurs interactions.

Résultat final

L'objectif est pleinement atteint : un utilisateur accède à http://www.monsite.local, remplit le formulaire, et son incident est automatiquement enregistré dans GLPI sous forme de ticket, sans aucune action manuelle de la part de l'équipe informatique.

En complément, une infrastructure de supervision avec Zabbix 7.2 a été déployée sur la même VM Ubuntu Server. Zabbix surveille en temps réel les services Apache et MariaDB, la charge CPU, la mémoire et le disque. Des déclencheurs ont été configurés pour alerter automatiquement en cas de panne détectée. Ce dispositif complète parfaitement GLPI : Zabbix détecte les incidents côté système de manière proactive, tandis que GLPI reçoit les signalements côté utilisateur.

Plusieurs améliorations sont envisageables : sécuriser les communications avec HTTPS, remplacer les fichiers hosts par un vrai serveur DNS, ajouter un DHCP dédié, envoyer des emails de confirmation aux utilisateurs, mettre en place un pare-feu UFW, et automatiser les sauvegardes de la base GLPI.