

EASY2DRIVE

Documentation complète du projet — Développement & Infrastructure

Membres du Projet :

Équipe Infrastructure et Réseaux :

Ilyess Bouchnafa et Estaban Moussu

Équipe de Développement :

Ryan Coulon et Simond Marchand

Sommaire

Présentation du projet

- Vue d'ensemble
- Environnement technique & journal de développement

Partie 1 — Développement de l'application

- DevOps & Gouvernance — Cas d'utilisation et architecture
- Évolution de la base de données
- Sécurité et authentification
- Sécurisation des mots de passe — niveau base de données
- Sécurisation des accès à la base de données
- Sécurisation des mots de passe — niveau applicatif
- Traçage des événements (RGPD)

Partie 2 — Infrastructure et réseau

- Vue d'ensemble et schéma réseau
- Installation et configuration de Proxmox
- Configuration du serveur Web
- Configuration du serveur de base de données
- Conteneur de sauvegarde (SVG)
- Configuration de DokuWiki

Présentation du projet

Bienvenue sur la documentation du projet Easy2Drive

Ce document rassemble l'ensemble de la documentation du projet **Easy2Drive**, portant sur les avancées en réseau et en développement réalisées au cours du projet.

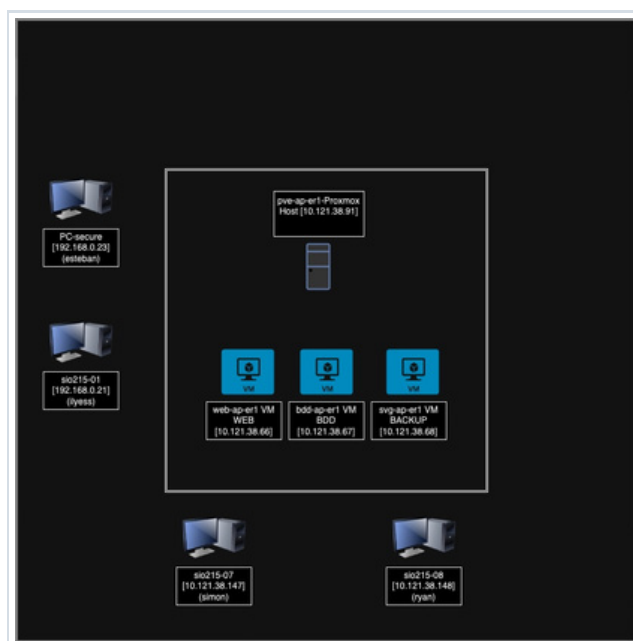


Schéma d'architecture générale du projet Easy2Drive

Environnement technique

- **Outils de collaboration** : Git (Gitea) pour le versioning, Trello pour la gestion de projet.
- **Stack** : PHP 8 / MySQL (MariaDB).
- **Accès base de données** : utilisateur `adminEasy` restreint à la base `bdeasy` uniquement.
- **Gestion des erreurs** : utilisation de blocs `try...catch` pour éviter les fuites d'informations techniques lors de plantages SQL.

Journal de développement

- **D0/D1** : installation de la base et mise en place du système d'authentification par rôles.
- **D2/D5** : développement des interfaces de gestion (consultation / modification / suppression).
- **D6** : création d'un formateur grâce à la fonction `add_formateur`.
- **D7 & D10** : affichage d'une liste d'élève ou formateur en fonction de l'auto-école du directeur grâce à la fonction `getUtilisateurAutoEcole()`.
- **D8** : modification et suppression d'un formateur possible grâce aux fonctions `update_formateur()` et `delete_user()`.
- **D9** : création d'un élève grâce à la fonction `add_eleve()`.
- **D11** : modification et suppression d'un élève possible grâce aux fonctions `update_user()` et `delete_user()`.
- **D12** : audit final de conformité RGPD et rédaction des mentions légales.

PARTIE 1

Développement de l'application

Cas d'utilisation

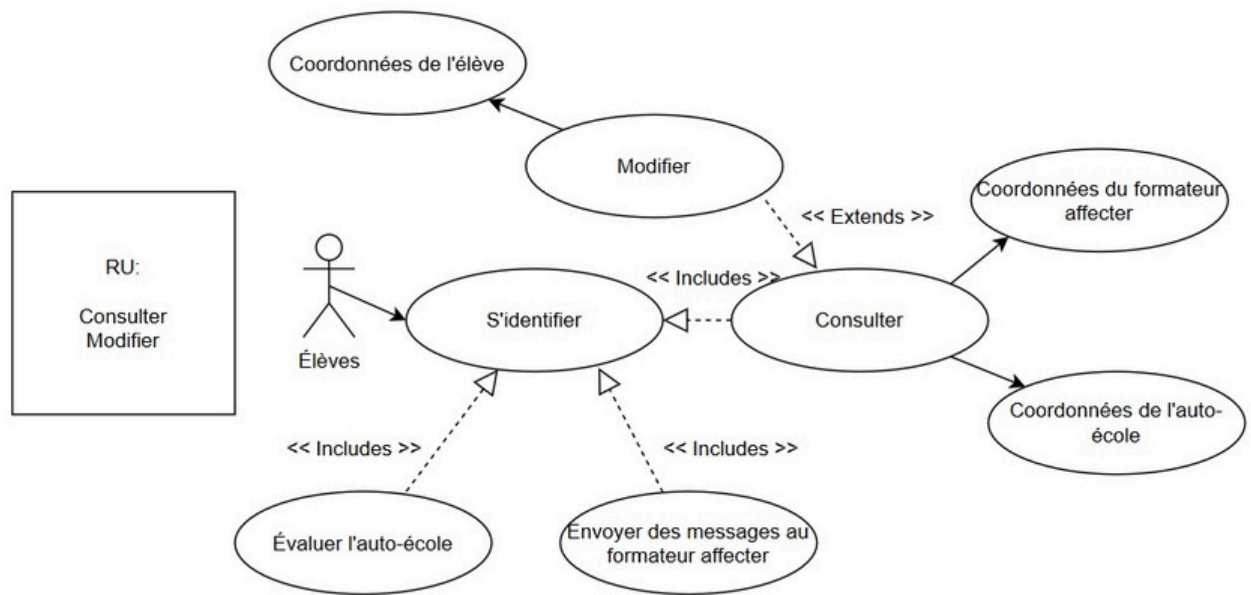


Schéma cas d'utilisation — Élève

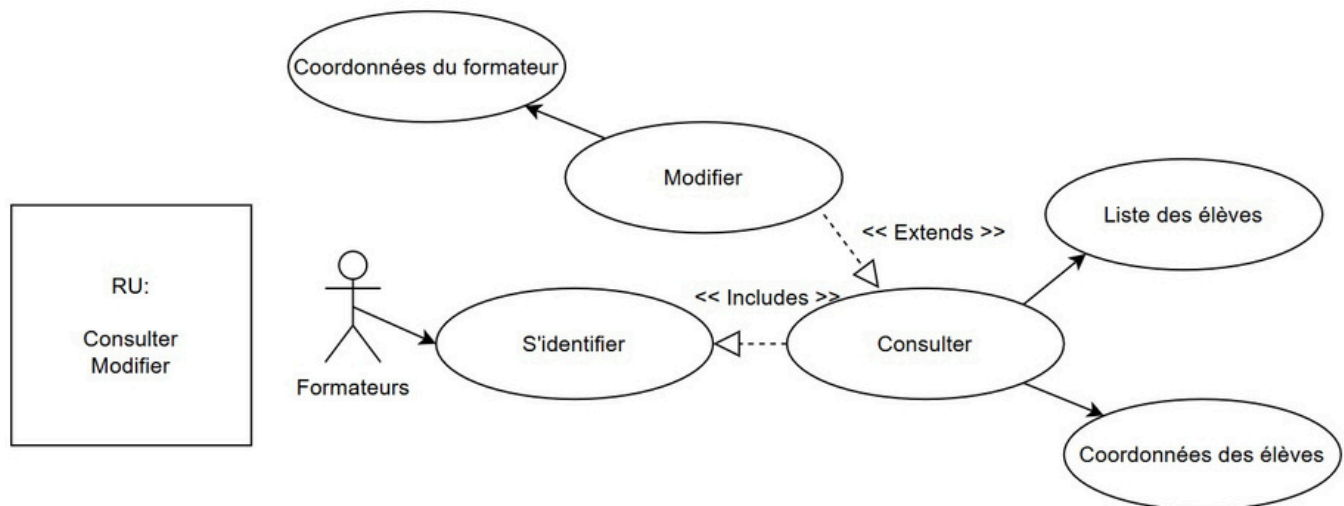


Schéma cas d'utilisation — Formateur

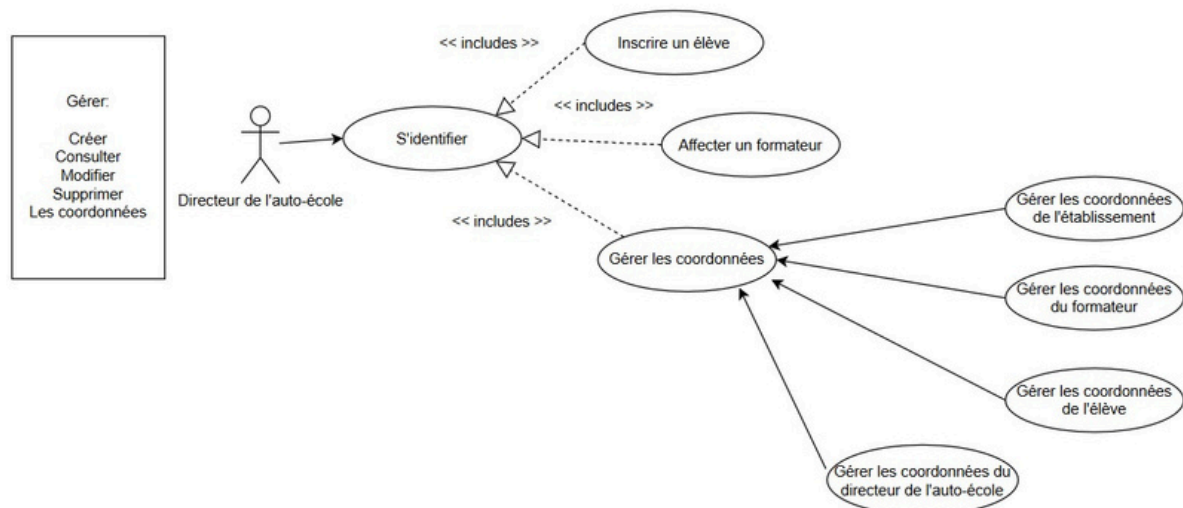


Schéma cas d'utilisation — Directeur

Architecture et normes de développement

Le projet a été restructuré pour passer d'un code monolithique (« stagiaire ») à une architecture robuste.

Modèle MVC

Séparation stricte entre :

les données (Modèle),
la logique (Contrôleur),
l'affichage (Vue).

-
-

Modèle objet (classes métiers)

- **Utilisateur.php** : classe parente (id, nom, prenom, email).
- **Eleve.php** : hérite d'Utilisateur (+ NEPH, date de naissance, adresse).
- **Formateur.php** : hérite d'Utilisateur (+ numéro d'autorisation).

Accessibilité

Utilisation de balises sémantiques HTML (`<header>` , `<footer>`) et design responsive avec Bootstrap.

Évolution de la base de données

- Modification des noms, pour éviter toute erreur dans les scripts, ainsi qu'ajout d'un aspect sécurité.
- Séparation dans une autre table des mots de passe, pour une future amélioration en matière de sécurité.
- Modification afin de pouvoir affecter un élève à un formateur.

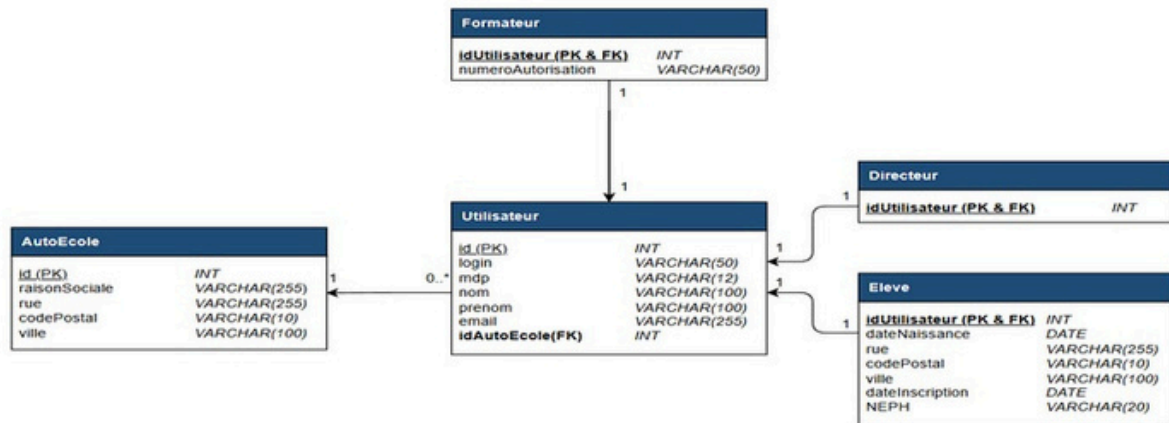


Schéma de la base de données — Avant évolution

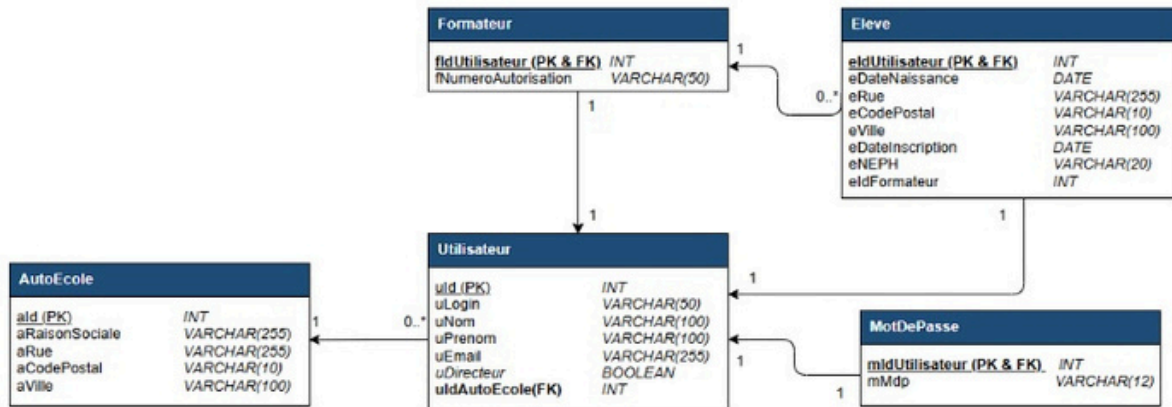


Schéma de la base de données — Après évolution

Sécurité et authentification

Stratégie de connexion

- **Nommage des logins** : format cohérent `prenom.nom` , justifié pour l'unicité et la simplicité.
- **Mots de passe** : sécurisation par hachage via `password_hash()` (algorithme BCrypt). Aucun mot de passe n'est stocké en clair (exemple : « test » devient `$2y$10$gX2wtz.H20AoFrT8QNmsXurQecEug3caL7F49sb767KKHoSdFCdF`) .
- **Sessions** : utilisation de `$_SESSION['user']` pour maintenir l'état de connexion et l'ID de l'auto-école d'appartenance.
- **Vérification du couple identifiant / mot de passe** : grâce à la fonction `verify_credentials(identifiant, mot de passe)` .

Matrice des droits (CRUD)

Conformément au cahier des charges, les droits sont répartis ainsi :

Rôle	Droits	Description
Directeur	CRUD	Gestion complète : ses infos, son école, ses formateurs, ses élèves et les affectations.
Formateur	RU RU	Consulter / modifier son profil + consulter la liste de ses élèves.
Élève		Consulter / modifier son profil + consulter les infos de son école et de son formateur.

Conformité RGPD

Justification des données :

- **NEPH / Autorisation** : obligations légales pour la formation au permis.
- **Date de naissance** : vérification de l'âge légal pour l'examen.

Filtrage des données : un directeur ne voit que les élèves et formateurs de sa propre auto-école, grâce à la sécurisation des requêtes SQL par l'ID de session.

Mentions légales : accessibles sur l'interface pour informer sur l'usage et la durée de conservation des données.

Sécurisation des mots de passe — niveau base de données

Nous avons entièrement repensé la manière dont les mots de passe sont stockés dans la base.

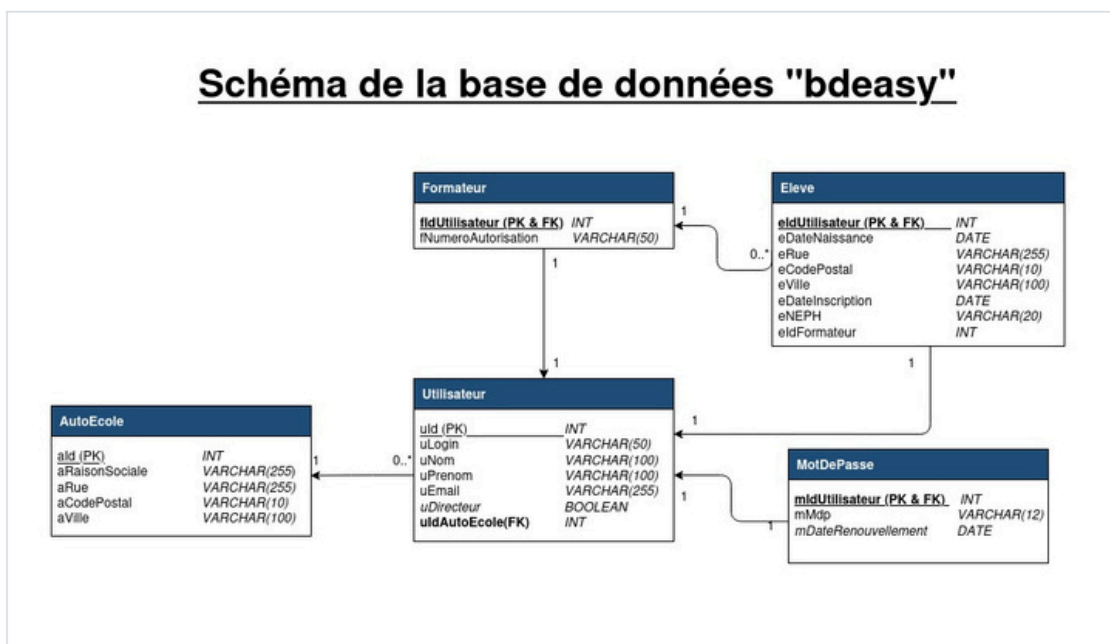
Nos actions :

- Nous avons créé une **table séparée** pour les mots de passe (héritage direct de la mission 1). Cette table n'est pas directement liée aux informations personnelles des utilisateurs, ce qui limite les risques en cas de faille.
- Nous avons mis en place un **hachage systématique** de chaque mot de passe avec un algorithme robuste (bcrypt, via `password_hash()` en PHP).
- Nous avons ajouté un **salage unique** par utilisateur, ce qui rend inopérantes les attaques par tables arc-en-ciel (rainbow tables).
- Aucun mot de passe n'est jamais stocké ou transmis en clair dans la base de données.

On observe donc bien que la table mot de passe est séparée ; nous avons également ajouté une date de renouvellement dans la base de données, ce qui permettra le développement d'une future fonctionnalité.

Pourquoi c'est important ? Même si un attaquant parvenait à voler la table des mots de passe, il ne pourrait pas les utiliser directement. Le hachage + salage rend leur passage extrêmement long et coûteux.

Schéma de la base de données "bdeasy"



Modèle conceptuel de données — bdeasy

Sécurisation des accès à la base de données

Nous avons mis en place une **gestion fine des privilèges** au niveau de la base elle-même, en créant plusieurs comptes MySQL/PostgreSQL avec des droits différenciés.

Les comptes créés et leurs droits :

Compte	Rôle	Droits (CRUD) sur les tables principales
cpt_dir	Directeur	CRUD complet sur Utilisateur, Élève, Formateur, AutoÉcole
cpt_for	Formateur	R + U partiel (lecture et mise à jour de ses données)
cpt_ele	Élève	R + U très limité (profil personnel)
cpt_auth	Authentification	R uniquement sur la table des mots de passe et quelques champs

Légende : C = Create (INSERT), R = Read (SELECT), U = Update (UPDATE), D = Delete (DELETE)

Notre application utilise le compte approprié en fonction du rôle de l'utilisateur connecté :

- Un élève ne peut pas modifier les informations d'une auto-école.
- Exemple : le compte `cpt_auth` ne sert qu'à vérifier les identifiants et ne peut pas lire d'autres tables.

Amélioration identifiée (post-mission) : nous avons noté qu'il faudrait encore restreindre les permissions de lecture de `cpt_auth` sur les autres tables pour une sécurité maximale. Modification des droits prévue pour que les utilisateurs aient accès à une vue spécifique.

Sécurisation des mots de passe — niveau applicatif

Nous ne nous sommes pas contentés du stockage sécurisé. Nous avons également imposé des **règles de gestion** fortes dans l'application web.

Nos règles applicatives :

- **Renouvellement obligatoire à la première connexion** — lorsqu'un compte est créé (par un directeur ou un administrateur), l'utilisateur doit immédiatement choisir un nouveau mot de passe. Le mot de passe temporaire ne fonctionne qu'une seule fois.
- **Expiration tous les 90 jours** — au bout de 90 jours, l'application bloque l'accès et force l'utilisateur à changer son mot de passe avant de pouvoir continuer.
- **Renouvellement volontaire** — chaque utilisateur peut changer son mot de passe à tout moment depuis sa page profil, après avoir saisi son mot de passe actuel.
- **Validation de complexité :**
 - Longueur minimale : 12 caractères
 - Au moins une majuscule, une minuscule, un chiffre et un caractère spécial

Nous avons codé ces règles en PHP côté serveur pour garantir qu'elles soient appliquées.

Traçage des événements (RGPD)

Pour respecter le **RGPD** et permettre une traçabilité complète, nous avons créé une **base de données dédiée** : `dbRgpdLog` .

Schéma de la base de données "bdRgpdLog"

Evenement	
id (PK)	INT
loginUtilisateur	VARCHAR(50)
ipUtilisateur	VARCHAR(16)
typeEvent	VARCHAR(20)
tableCible	VARCHAR(20)
date	DATE

Schéma de la base de données bdRgpdLog

Que contient cette base ?

Une table **Evenement** avec les colonnes suivantes (exemple) :

- `id` (auto-incrément)
- `loginUtilisateur` (lien vers l'utilisateur concerné)
- `ipUtilisateur` (de l'utilisateur)
- `typeEvent` (connexion, échec_connexion, modification_profil, suppression, consultation_donnees_sensibles, etc.)
- `tableCible` (table impactée par l'action de l'utilisateur)
- `date` (date de l'action)

Comment nous traçons automatiquement ?

À chaque action importante dans l'application web, notre code PHP appelle une fonction `gererLog($type, $details)` qui insère silencieusement une ligne dans `bdRgpdLog` . Cela ne bloque jamais l'utilisateur et n'impacte pas les performances.

Ce que nous traçons systématiquement

- Connexion réussie / échouée
- Ajout d'utilisateur élève ou formateur
- Changement de mot de passe
- Modification ou suppression d'un élève ou d'un formateur
- Consultation de la liste des élèves (pour tracer les accès abusifs)

Pourquoi cette base séparée ?

- Pour assurer une sécurité complémentaire, car elle n'est pas sur le même réseau.
- Pour faciliter la gestion des durées de conservation (les logs peuvent être purgés après 6 mois ou 1 an sans toucher aux données métier).
- Pour répondre plus facilement à une demande de la CNIL ou d'un utilisateur qui souhaite savoir qui a consulté ses données.

Arborescence du projet

```
EASY2DRIVE/
├── controleur/
│   ├── databaseFonctions.php
│   └── sessionFonctions.php
├── donnees/
│   ├── images/
│   │   ├── autoecole.jpg
│   │   ├── Logo.png
│   │   └── LogoE2D.png
│   └── modele/
│       ├── classesMetiers/
│       │   ├── Eleve.php
│       │   ├── Formateur.php
│       │   └── Utilisateur.php
│       ├── database.cfg
│       ├── Database.php
│       └── partielles/
└── bootstrap.js
```

```
|
|   |   | footer.php
|   |   | header.php
|   |   | introPage.php
|   |   | profilPage.php
| style/
|   |   | css.css
|   |   | style.css
|
|   |   | vue/
|   |   |   | accueil.php
|   |   |   | delete.php
|   |   |   | gestion.php
|   |   |   | login.php
|   |   |   | logout.php
|   |   |   | profil.php
|   |   | renouvellement.php
|   |   |   | test.php
|   |   | updateFormateur.php
|   |   |   | updateProfil.php
|   |   |   | updateUser.php
|   |   | .gitignore
```

PARTIE 2

Infrastructure et réseau

Projet Easy2Drive — dirigé par Ilyess Bouchnafa et Esteban Moussu.

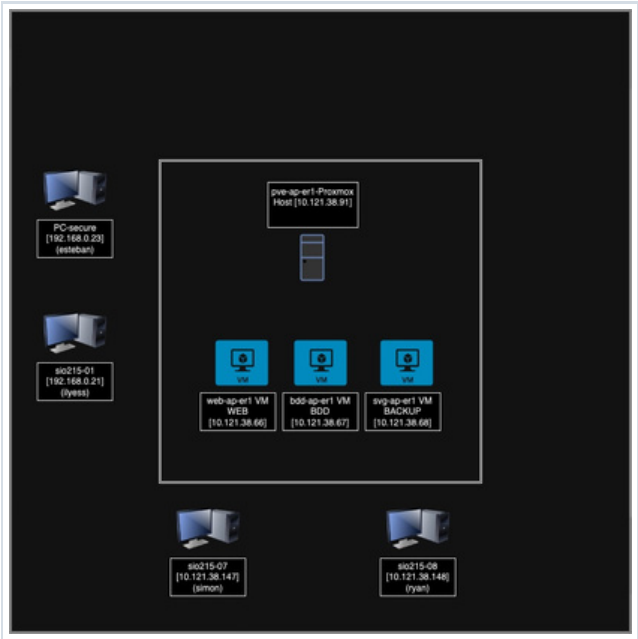


Schéma d'architecture générale du réseau

Schéma de la base de données "bdRgpdLog"

Evenement	
id (PK)	INT
loginUtilisateur	VARCHAR(50)
ipUtilisateur	VARCHAR(16)
typeEvent	VARCHAR(20)
tableCible	VARCHAR(20)
date	DATE

Installation de Proxmox sur la machine

(R0.1) Installation de Proxmox

Nous avons installé Proxmox sur notre machine grâce aux étapes suivantes : nous disposons d'une clé USB fournie avec Ventoy et l'ISO de Proxmox. Lors du boot de la machine (touche F12), nous démarrons sur la clé USB puis sélectionnons l'installation normale, puis l'installation graphique. Dès que l'installation graphique est choisie, on nous demande sur quel disque installer Proxmox, puis on choisit la langue, le pays et le mot de passe. Nous choisissons ensuite comme nom `pve-ap-er1.sio.lan`, puis l'adresse IP où sera installé le serveur Proxmox.

(R0.2) Mise à jour de la distribution et configuration

Une fois Proxmox installé, nous nous connectons sur le compte root via le site web indiqué sur l'écran du serveur au redémarrage. Nous créons un groupe Administrateur dans **Centre de donnée > Permissions > Groupes**, auquel nous attribuons la permission administrateur dans **Centre de donnée > Permissions** (Ajouter, chemin d'accès `/`, groupe créé, rôle « Administrator »).

Les comptes sont ensuite créés dans **Centre de donnée > Permissions > Utilisateurs** (nos prénoms), rattachés au groupe Administrateur. Pour mettre à jour la distribution, nous nous connectons via **Centre de donnée > pve-ap-er1 > Shell** avec les informations root, puis :

```
apt update && apt upgrade -y
```

Changement du fuseau horaire :

```
dpkg-reconfigure tzdata
```

puis choix de Europe puis Paris.

(R0.3) Connexion SSH à l'hyperviseur par couple de clés

Création d'un compte pour chaque administrateur réseau :

```
apt install sudo
useradd -m -s /bin/bash nomutilisateur
usermod -aG sudo nomutilisateur
passwd nomutilisateur
```

Une fois les comptes sudo créés, installation de SSH via `apt install ssh` puis création d'un couple de clés SSH sur le poste avec **keygen**, déposée avec `ssh-copy-id nomutilisateur@ip`

Une fois cela fait, l'identification au PVE se fait sans mot de passe.

Configuration du serveur Web

(R1.1) Installation et configuration de base du serveur web

Création de la première machine virtuelle du réseau, nommée « web-ap-erx », via une image ISO netinst sans interface graphique, renseignement des caractéristiques (RAM, stockage...). Une fois l'installation terminée, mise à jour des paquets installés :

```
apt update && apt dist-upgrade
```

Paramétrage de la time zone sur Europe/Paris :

```
timedatectl set-timezone Europe/Paris
```

Modification de l'adresse IP dans `/etc/network/interfaces` avec nano ou vim. Pour finir l'installation, paramétrage du SSH depuis les machines physiques via `ssh-keygen` puis `ssh-copy-id root@web-ap-erx`.

(R1.2) Configuration de base du serveur de base de données

Clonage de la VM du serveur de base de données, modification de `/etc/hosts` et `/etc/hostname` de « web-ap-erx » vers « bdd-ap-erx » puis modification de l'adresse IP statique dans `/etc/network/interfaces`.

Mise à jour de la timezone avec `dpkg-reconfigure tzdata` (Europe puis Paris).

(R1.3) Serveur web et PHP fonctionnels

Installation des paquets nécessaires au bon fonctionnement du serveur (apache2, php, php-gd, php-mbstring, php-mysql) :

```
apt install apache2 php php-gd php-mbstring php-mysql
```

Installation de Git :

```
apt install git gpg -y
```

Tests réalisés en se connectant depuis un navigateur sur le serveur web.

(R1.5) Hébergement de la solution web développée

Une fois Git installé, clonage à la racine du service web (`/var/www/html`) du dépôt Git de l'équipe de développement via `git clone` suivi de l'URL du dépôt. Vérification de l'accessibilité du site via le navigateur et de la présence du dépôt cloné dans l'index du serveur web.

(R1.7) Script de mise à jour du dépôt Git de production

Mise en place des accès SSH des développeurs vers le serveur web par clé publique : création de comptes personnels sur la machine web-ap-er1 » :

```
sudo useradd -m -s /bin/bash nomutilisateur
```

Les options `-m` et `-s` permettent de créer le répertoire personnel en même temps que le profil, et d'attribuer `/bin/bash` comme shell opérationnel. Sur les machines physiques des développeurs :

```
ssh-keygen
ssh-copy-id nomutilisateur@web-ap-er1
```

La connexion par clé SSH est alors active (mot de passe de session requis la première fois).

Script de mise à jour automatique du dépôt de production :

```
#!/bin/bash
export http_proxy=http://10.121.38.1:8080
export https_proxy=http://10.121.38.1:8080
cd /var/www/html/Easy2Drive---G1
git pull
```

Ce script est copié à la racine du répertoire personnel des deux développeurs :

```
sudo cp /var/www/html/majcode.sh /home/ryan/
sudo cp /var/www/html/majcode.sh /home/simon/
```

Paramétrage des droits d'accès : création d'un groupe « développeurs » avec `groupadd développeurs` , ajout des deux développeurs avec `usermod -aG développeurs nomutilisateur` . Attribution du dépôt Git (Easy2Drive---G1) avec droit d'écriture récursif :

```
sudo chmod -R g+rwX /var/www/html/Easy2Drive---G1
sudo chown -R root:développeurs /var/www/html/Easy2Drive---G1
```

Configuration du serveur de base de données

(R1.4) Serveur de base de données opérationnel

Installation des packages

`mariadb-server` et `ssh` :

```
apt install mariadb-server ssh
```

Configuration de mariadb-server avec la commande `mariadb-secure-installation`, puis changement du mot de passe root en acceptant la connexion avec unix.

Connexion puis création des comptes des développeurs :

```
mysql -u nomdutilisateur -pmotdepasse
```

```
CREATE USER 'developpeur1'@'adresseip1' IDENTIFIED BY 'motdepasse';  
CREATE USER 'developpeur2'@'adresseip2' IDENTIFIED BY 'motdepasse';
```

Création de la base de données puis attribution des accès aux développeurs :

```
CREATE DATABASE dbeasy;  
GRANT ALL PRIVILEGES ON dbeasy.* TO 'developpeur1'@'adresseip1';  
GRANT ALL PRIVILEGES ON dbeasy.* TO 'developpeur2'@'adresseip2';  
FLUSH PRIVILEGES;
```

À ce stade, les développeurs n'ont pourtant toujours pas accès à la base de données, ce qui est normal : il faut modifier la configuration de mariadb-server pour qu'il accepte toute requête sur son port 3306 depuis toutes les IP qui pourraient le joindre. Pour cela, modification de `/etc/mysql/mariadb.conf.d/50-server.cnf` en commentant la ligne `bind-address = 127.0.0.1`.

Les développeurs peuvent ensuite utiliser la base de données en se connectant avec leurs identifiants sur l'IP et le port de la VM (3306), avec toutes les permissions.

Conteneur de sauvegarde (SVG)

(3.0) Installation de la sauvegarde

Pour le serveur de sauvegarde, utilisation d'un conteneur LXC dont l'installation diffère d'une VM classique : depuis Proxmox, installation du modèle de conteneur (Debian 12 standard), puis création du conteneur en renseignant toutes les informations :

- IP : 10.121.38.68
- Passerelle : 10.121.38.254
- DNS Domain : sio.lan
- Serveur DNS : 10.121.38.7

Une attention particulière est portée lors de l'installation, car les paramètres renseignés ne pourront pas être modifiés par la suite. Une fois tous les paramètres renseignés, le conteneur est opérationnel.

(3.1 et 3.2) Sauvegarde et compression de la base opérationnelle

Mise en place d'une base de données de jeu d'essai contenant des données utilisateurs pour les tests, évitant l'exécution de commandes potentiellement destructives. Script de sauvegarde et compression de la base avec le jeu d'essai :

```
#!/bin/bash
DAT=$(date +%F-%Hh-%Mm-%Ss")
ssh utilisateur@adresseip "mysqldump -u utilisateurdb -p'motdepasse' basededonnée" | gzip > basededonnee-$DAT-dump.sql.gz
```

(3.3 et 3.4) Planification automatisée des sauvegardes

Utilisation de l'outil **cron** pour planifier les scripts précédemment créés : deux sauvegardes par jour (8h et 18h), du lundi au vendredi.

```
( crontab -l 2>/dev/null; echo "0 8,18 * * 1-5 /home/svg/compression.sh" ) | crontab -
```

(3.5) Rotation des sauvegardes

Modification du script de sauvegarde afin de ne conserver que les 10 dernières sauvegardes les plus récentes :

```
#!/bin/bash
set -e

BACKUP_DIR="/home/svg/sauvegardes"
DAT=$(date +%F-%Hh-%Mm-%Ss")

mkdir -p "$BACKUP_DIR"

ssh svg@10.121.38.67 "mysqldump -u svg -p'root' dbtest" | gzip > "$BACKUP_DIR/dbeasy-$DAT-dump.sql.gz"

ls -lt "$BACKUP_DIR"/dbeasy-*-dump.sql.gz 2>/dev/null | tail -n +11 | xargs -r rm --
```

Configuration de DokuWiki

(2.1) Application documentaire installée

Récupération de l'URL de l'archive depuis dokuwiki.org, installation sur le serveur :

```
sudo wget https://download.dokuwiki.org/out/dokuwiki-63487079d8919ad20087d39beea025a9.tgz
```

Extraction de l'archive :

```
sudo tar xvzf dokuwiki-63487079d8919ad20087d39beea025a9.tgz
```

Suppression de l'archive puis attribution de la propriété à `www-data` pour les répertoires `data/`, `lib/` et `conf/` :

```
rm dokuwiki-63487079d8919ad20087d39beea025a9.tgz
sudo chown -R www-data data/
sudo chown -R www-data lib/
sudo chown -R www-data conf/
```

Lors de l'accès à l'installation, le message d'erreur suivant est apparu : « PHP function xml_parser_create is not available. Maybe your hosting provider disabled it for some reason? ». Installation de la fonction manquante :

```
sudo apt install php-xml
```

(2.2) Paramétrage de l'espace documentaire

Une fois `php-xml` installé, actualisation de la page wiki puis passage par le menu d'installation : nom du wiki, choix d'un « superuser » (compte admin) avec mot de passe.

Sur la page « Welcome to your new DokuWiki », accès via l'engrenage « Administrator » à la page d'administration, permettant de gérer les ACL (Access Control List), les plugins utilisés par DokuWiki, les comptes utilisateurs, les paramètres de thème du site web, ainsi que d'autres configurations.

Création des utilisateurs et attribution de groupes définissant leurs permissions, groupes créés dans « Gestion de la liste des contrôles d'accès » (recherche du groupe puis sélection des permissions par page).

(2.3) Organisation en pages de la documentation

Modification de la page d'accueil « start » (page par défaut) via le crayon d'édition.

Ajout de titres :

```
===== Titre =====
```

Création de pages :

```
[[Nomdelapage || Aliasdelapage]]
```

Le texte redirige vers la page Nomdelapage mais affiche Aliasdelapage.

Création de catégories, elles-mêmes contenant des pages :

```
[[Nomdelacatégorie:Nomdelapage | Aliasdelapage]]
```

Le texte redirige vers la catégorie Nomdelacatégorie, vers la page Nomdelapage, mais affiche Aliasdelapage.

Deux catégories ont été créées : **IT** et **Développeurs**, chacune dotée d'une page d'accueil avec redirections vers des pages ressources de documentation.

(2.4) Accès en lecture et écriture réglementé

Pour limiter les accès depuis l'extérieur et en interne, des groupes ont été configurés avec les permissions suivantes :

Rôle	Aucune	Lecture	Écriture
Page d'accueil			
ALL	x		
user		x	

admin			x
it		x	
developpeurs		x	
Catégorie IT			
ALL	x		
user	x		
admin			x
it		x	
developpeurs	x		
Catégorie Développeurs			
ALL	x		
user	x		
admin			x
it	x		
developpeurs		x	