Préambule

Cette SAÉ (Situation d'Apprentissage et d'Évaluation) vise à installer et configurer des services réseaux permettant de développer et de déployer des applications informatiques communicantes. L'objectif étant de répondre à un besoin exprimé par un client. Cette SAÉ est relatives aux ressources R2.04 et R2.05.

Votre mission requiert l'installation et la configuration des services nécessaires au développement d'un site web. En effet, un site Web seul ne suffit pas, il lui faut à minima des modules supplémentaires tels qu'un langage de programmation et une base de données pour générer du contenu dynamique et personnalisé pour différents utilisateurs de différents profils. Souvent, lors de l'installation des services, il est question de repérer et de bien paramétrer les fichiers de configuration du serveur et de ses différents modules. Dans notre cas, nous allons manipuler tout cela sous un environnement Linux avec un serveur Web de type Apache, une base de données de type MySQL, et un langage (code serveur) qui est le PHP.

Consignes importantes

Lire **très attentivement** les consignes du fichier : « fiche de suivi ».

Déroulement et timing

Lire **très attentivement** les consignes du fichier : « fiche de suivi ».

Cahier de charges

Cette mission combine : installation, configuration et programmation Web. Concrètement, on demande la réalisation du **cahier de charge inclut dans le TP (obligatoire)**, **le reste des points (tout ce qui est en dehors du texte TP) est optionnel**. En résumé, on vise idéalement les points suivants :

- S'assurer de **la bonne installation** du serveur Web, du langage PHP côté serveur et de la base de données MySQL
- Mettre en place un ensemble de jolies pages **publiques** (accessibles depuis tout le monde) du serveur Web. Ces pages doivent être liées entre elles (navigation avec des liens) et ne doivent pas être complètement statiques. Elles doivent en effet contenir des informations dynamiques (tels que la date, le jour, la position géographique, le type de terminal [est ce que le client utilise un portable ou un PC], etc.)
- Mettre en place un espace/page Web pour un accès réservé à un « administrateur ». L'accès à cet espace demande **une authentification de l'utilisateur** par login/mot de

passe. Ces derniers sont **gérés par le serveur Apache directement (sans le recours à une base de données)**.

- Mettre en place un espace/page Web qui demande **une authentification de l'utilisateur** par login/mot de passe **sauvés dans votre propre base de données**. Cela veut dire que de n'importe où, un utilisateur a besoin de se connecter avec un login et un mot de passe. Les logins/mots de passes autorisés doivent être stocké dans votre propose base de données. Vous l'avez compris, il faut une base de données, une table de login/mot de passe et gérer l'authentification avec un code PHP qui examine le login et mot de passe de l'utilisateur et en cas de succès (existence et matching du login/passe et utilisateur « validé ») votre serveur doit donner le droit à l'utilisateur d'accéder à une partie du site Web. La notion d'utilisateur « validé » sera détaillée dans la suite. **3 comptes** différents doivent être crées dans votre propre base de données. Sur le plan PHP, attention aux sessions et au paramétrage pour par exemple ne pas demander une authentification à chaque fois. Aussi, toutes les pages nécessitant que l'utilisateur soit authentifié, doivent s'assurer (par code PHP) qu'au moment de l'accès, l'utilisateur est toujours bien authentifié, sinon l'utilisateur doit être redirigé vers la page d'authentification.
- Les pages de votre serveur Web doivent avoir **un minimum de style (CSS)**. Les pages d'erreur du site Web doivent être personnalisées (exemple, une erreur « 404 not found » ne doit pas délivrer la page par défaut d'Apache mais une page à vous)
- L'ajout d'utilisateurs dans la base de données doit se faire avec **une page d'inscription** où l'utilisateur renseigne : nom, prénom, mail, login, mot de passe choisi.
    - Votre code php doit s'assurer de l'unicité des logins dans la base de données ainsi que l'unicité des mails utilisateurs. En cas d'erreur, cette dernière doit être signalée à l'utilisateur (exemple : « login existe déjà »)
    - Lorsqu'un utilisateur s'inscrit, votre code php doit générer un lien personnalisé et aléatoire associé à cet utilisateur (le lien doit être complexe et non devinable)
    - L'utilisateur doit cliquer sur ce lien pour que son « statut » devient utilisateur validé, sinon tant qu'il n'a pas cliqué sur ce lien, il ne pourra pas se logger même si son login et mot de passe sont correctes
    - Le lien de validation est censé être envoyé à l'adresse email de l'utilisateur utilisée lors de l'inscription. Ce point n'est pas demandé dans notre cahier de charge
    - Votre serveur doit être programmé pour que tous les comptes utilisateurs inscrits mais pas validés doivent être effacés automatiquement chaque jour à 3h du matin (indication : utiliser « cron » sous Linux)

➢ **Durant la première séance de TP**, constituez votre groupe de projet. Idéalement, chaque groupe doit comporter **4 membres mais pas plus et un minimum de 3 (pour répartir le travail)**. Les 3/4 membres doivent appartenir au même groupe TP (exemple A1).

➢ Élisez **votre chef de projet et son adjoint** (car on n'est pas à l'abri d'une absence). N'oubliez pas de l'indiquer dans la fiche de suivi. Le chef de projet doit coordonner le travail du groupe, veiller sur la bonne coordination entre membres et trancher lorsqu'il n'y a pas de consensus entre membres. Il est le contact privilégié de l'enseignant pour remonter les informations et préoccupations du groupe (mais pas au dernier moment).

➢ **Durant la première séance de TP**, déterminez, en discutant avec les membres, la manière dont vous allez :

- partager l'information, sauvegarder vos documents (fichiers de configuration, notes techniques, rapport du projet),
- répartir les tâches du projet entre membres,

- établir un diagramme prévisionnelle des tâches
- rédiger le rapport de projet

Résumé de ce qu'il faut rendre à la fin ?

- Sur Moodle, dans la section **« Remise du rapport de projet »**, sélectionnez votre demi groupe de TP et rendez votre rapport en PDF. Nomenclature : nomDeFamille1_nomDeFamille1_nomDeFamille1_nomDeFamille1_1C1.pdf
  (nomDeFamille1 : nom de famille du premier membre de votre groupe ; 1C1 : le demi groupe TP 1 du groupe 1C)
- La remise du PDF se fait **avant la fin de la dernière séance** dédiée sur l'emploi du temps à cette SAÉ (à l'issue des 15h réglementaire pour tout le monde)
- Votre rapport doit inclure au début la **« fiche de suivi »** remplie. Elle est fournie en version éditable sur Moodle
- Ensuite, votre rapport doit inclure **« la partie obligatoire »** qui est la réponse à toutes les manipulations indiquées dans le fichier TP (disponible également sur Moodle)
- Ensuite, votre rapport, peut inclure **(en bonus)**, la réalisation des points ouverts indiqué dans la section **« Cahier de charges » de ce TD (voir ci-dessous)**
- *Important : n'attendez pas la dernière séance pour demander des précisions sur le cahier de charge. Faîtes le tout le long de la SAÉ pour les éventuelles demandes d'éclaircissement des points à réaliser.*

Pour cette SAÉ, nous optons pour la mise en œuvre d'un serveur de type Apache. Sur le Moodle de la SAÉ, vous trouverez **une documentation complète** Apache, MySQL et PHP dans la section « Documentation ». Apache est conçu sur le principe de la **modularité** (utilisation de modules). Par conséquent, Apache permet à un administrateur de serveur Web d'ajouter ou de supprimer différents modules afin d'étendre -au besoin- les fonctionnalités primaires du serveur et également améliorer ses performances. Parmi les modules les plus connus et les plus utilisés par un serveur Apache se trouvent les **modules PHP et SSL**. Ces modules permettent d'augmenter les capacités du serveur Web par un langage de programmation dynamique (qui s'exécute côté serveur) et s'assurer que le trafic entre le serveur Web et le visiteur du site est crypté.
Les modules Apache sont activés en utilisant une directive (i.e. une ligne de configuration) nommée « LoadModule » indiquée dans un fichier de configuration. Voici l'algorithme général adopté lors de l'activation d'un module :

⇨ 1. Installer le module Apache souhaité, s'il n'est pas déjà installé.
Exemple : *# apt install fail2ban*

Fail2ban est un module intéressant. C'est un module de prévention des intrusions qui protège contre les attaques (principalement par force brute) en interdisant les agents utilisateurs malveillants, en interdisant les scanneurs d'URL et bien plus encore. Si trop d'échec -d'une IP cliente- alors ce n'est pas normal et donc on la bannie (si c'est « fail » alors c'est « ban » !)

⇨ 2. Vérifier si le module est déjà chargé ou activé par Apache. Par exemple, avec la commande : *# apachectl -t -D DUMP_MODULES*

« apache2 -M » est un raccourcis de la commande précédente qui peut être utilisé. -M comme « modules » !

⇨ 3. S'assurer que dans le fichier de configuration d'Apache, il y a une directive qui demande à Apache de charger ce module. Cette directive s'appelle « LoadModule ». Exemple de syntaxe (pour le module SSL) :
LoadModule ssl_module /usr/lib/apache2/modules/mod_ssl.so (le fichier .so est là grâce à l'installation du module en étape 1.)

Les systèmes Ubuntu et d'autre dérivés Debian, openSUSE et SLES disposent de la commande « a2enmod » déjà installée et prête à charger facilement les modules Apache sans se soucier à utiliser manuellement la directive « LoadModule » et chercher les « .so ». De même, pour désactiver un module, il existe la commande « a2dismod ». "a2" comme Apache2, "en/dis" comme "enable" ou "disable" et "mod" comme module ! Exemple : # a2enmod ssl ou #a2dismod ssl, et c'est tout !

⇨ 4. Relancer Apache pour charger le module
⇨ 5. Vérifier si le module est chargé (avec #apache2 -M par exemple)

Le contenu du dossier Apache, fraichement installé sur Debian 11, ressemble à ce qui suit :

```
root@SAE:/root# ls /etc/apache2/
apache2.conf      conf-enabled  magic           mods-enabled          sites-
available conf-available      envvars          mods-available    ports.conf
sites-enabled
```

1. Repérez cette organisation du dossier, dans la documentation donnée en Annexe (i.e. le fichier vierge de configuration Apache 2).
   ⇨ À votre avis que contient les dossiers suivants ?
      - sites-available
      - sites-enabled
      - mods-available
      - mods-enabled
   ⇨ Sur notre machine virtuelle vierge, les dossiers sites-enabled et availables contiennent :

```
root@SAE:/root# ls /etc/apache2/sites-enabled/
000-default.conf
root@SAE:/root# ls /etc/apache2/sites-available/
000-default.conf  default-ssl.conf
```

   ⇨ Que peut ont déduire ?
2. Dans la documentation donnée en Annexe (fichier vierge d'Apache 2), repérez l'information suivante :
   ⇨ La valeur (en minutes) du délai pendant lequel Apache HTTP attendra une nouvelle entrée/sortie avant l'échec de la demande de connexion. Cet échec peut être dû à différentes circonstances telles que des paquets n'arrivant pas au serveur ou des données n'étant pas confirmées comme ayant été reçues par le client.
3. Dans les environnements où le service Internet est lent, cette valeur par défaut peut être raisonnable, mais il faut éviter une durée assez longue, en particulier si le serveur couvre une cible d'utilisateurs avec un service Internet plus rapide. Qu'en pensez-vous de la valeur par défaut en termes de sécurité ? Justifiez votre réponse.

➢ En fait, l'endroit et l'organisation des différents fichiers de apache diffèrent d'une distribution Linux à une autre. Comme rien ne garantit que vous allez travailler sous Debian

dans votre future vie professionnelle, il est intéressant de voir comment Apache est organisé dans les autres distribution (Table 1).

Option	Debian, Ubuntu	openSUSE and SLES	Fedora Core, CentOS, RHEL	macOS
Dossier principal	/etc/apache2/	/etc/apache2/	/etc/httpd/	/private/etc/apache2/
Fichier de conf principale	apache2.conf	httpd.conf	conf/httpd.conf	httpd.conf
ServerRoot	apache2.conf	n/a	conf/httpd.conf	httpd.conf
DocumentRoot	sites-enabled/*.conf	default-server.conf	conf/httpd.conf	httpd.conf
VirtualHost	sites-enabled/*.conf	vhosts.d/*.conf	conf/httpd.conf	httpd.conf, other/*.conf
LoadModule	mods-enabled/*.load	loadmodule.conf	conf.modules.d/*.conf	httpd.conf, extra/*.conf
Log	apache2.conf, sites-enabled/*.conf	httpd.conf, vhosts.d/*.conf	conf/httpd.conf	httpd.conf
User / Group	apache2.conf, envvars	uid.conf	conf/httpd.conf	

Table 1. Endroit des options pour Apache sous différentes distributions

Les distribution Linux utilisent aussi des valeurs différentes pour les options de Apache (Table 2).

Option	Debian, Ubuntu	openSUSE and SLES	Fedora Core, CentOS, RHEL	macOS
ServerRoot	/etc/apache2/	n/a	/etc/httpd	/usr/
DocumentRoot	/var/www/html/	/srv/www/	/var/www/html	/Library/WebServer/Documents/
Contenu des modules (eg. .so)	/usr/lib/apache2/modules/	/usr/lib64/apache2-prefork/	modules/	libexec/apache2/
Access / Error log	/var/log/apache2	/var/log/apache2/	logs/	/private/var/log/apache2/
User	www-data	wwwrun	apache	_www
Group	www-data	www	apache	_www
binaire	apache2	httpd	httpd	httpd

Table 2. Valeurs par défaut des options pour Apache sous différentes distributions

Notons que lorsqu'un chemin, indiqué dans les tableaux, n'est pas absolu (i.e. ne commence pas par /) alors ce chemin se réfère au chemin du dossier principal. Par exemple le dossier "modules/" pour le dossier des modules sous CentOS se lit /etc/httpd/modules/.

- ServerRoot : définit le répertoire dans lequel le serveur est installé ; DocumentRoot : est une directive qui permet de définir le répertoire à partir duquel Apache va servir les fichiers comme les pages HTML et images ; LoadModule : les fichiers qui permettent de charger des modules avec une directive nommée LoadModule ; User/Group : Apache se lance dans le système en tant que utilisateur et group indiqué (ex. sous Debian : apache s'exécute en tant que utilisateur www-data appartenant au groupe www-data) ; VirtualHost : les fichiers de configuration des « sites virtuels » i.e. des sites qui ont des noms de domaines différents et qui cohabitent avec d'autres sur le même serveur physique gérés par le même serveur Apache.

4. Un serveur Web vierge vient d'être installé et lancé sur une machine serveur qui n'est branchée à aucun réseau.
   - ⇨ A-t-on la possibilité d'accéder à sa page d'accueil par défaut ? Si oui, comment ?
   - ⇨ Sur quel port informatique le serveur est en écoute ?
   - ⇨ À votre avis, dans quel fichier on configure cette valeur (voir fichier donnée en annexe) ?

- ⇨ Peut-on écrire le port dans l'adresse Web (URL) pour accéder au site ? ou cela génèrerait une erreur ? Si oui, avec quelle syntaxe d'après vous ?
- ⇨ Dans quel cas il est intéressant de personnaliser le port d'écoute par défaut d'un serveur Web ?

5. Les services réseaux mettent en place des journaux d'accès (des logs). Donnez quelques utilités à l'utilisation des logs d'un serveur Web.
6. Sous Linux, les logs des services se trouvent souvent dans le dossier /var/log/. Ci-dessous ce que contient le dossier pour notre serveur Apache sous notre machine virtuelle de la SAÉ :

```
root@SAE:/root# ls /var/log/apache2/
access.log      access.log.1      error.log      error.log.1      error.log.2.gz
other_vhosts_access.log
```

- Quels sont les différents types de log qui sont présents ?
- À votre avis, à quoi sert chaque type ?
- À votre avis, quelle est la signification des fichier *.n (avec *n* un numéro)
- Le format des journaux d'accès est hautement configurable. Il est défini à l'aide d'une chaîne de format qui ressemble sensiblement à la chaîne de format de style langage C de printf. La liste exhaustive de ce que peut contenir une chaîne de format est fournie par le module natif d'Apache : le module mod_log_config. Ci-dessous un extrait du fichier access.log obtenu grâce à la commande
  # tail -f /var/log/apache2/access.log après un accès local au serveur Web. Le format de la ligne suivante est le fameux format appelé « *Combined Log Format* ». Ce format est très similaire au format standard appelé « Common Log Format » mais il contient quelques champs en plus utile pour le débogage et l'analyse des logs :

```
127.0.0.1 - - [01/May/2022:17:00:17 +0200] "GET / HTTP/1.1" 200 3384 "-"
"Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0"
```

- Interprétez les différents champs de cette ligne de journal. À votre avis quel serait le changement notable dans la ligne précédente si la page demandée n'existait pas sur le serveur ?

- <u>Contrôle d'accès par Apache au niveau utilisateur</u>
- Le mécanisme de contrôle d'accès au niveau utilisateur par Apache est différent du système d'utilisateurs (login Linux) au niveau système. Le mécanisme de contrôle d'accès est contrôlé par Apache en utilisant deux fichiers : un fichier des mots de passe et le fichier de de configuration du serveur.
- <u>*Le fichier des mots de passe*</u> : peut porter n'importe quel nom et doit se situer de préférence dans le répertoire /etc/apache2/. (Faire commencer le nom du fichier par un point afin de le caché). Les mots de passe dans ce fichier sont cryptés. Afin de rajouter une ligne dans le fichier de mot de passe contenant le nom d'utilisateur en clair et le mot de passe en crypté, on utilise la commande htpasswd. Exemple :
  - `root@SAE:~# htpasswd -c /etc/apache2/pass testeur`
    `New password:`
    `Re-type new password:`
    `Adding password for user testeur`
  - `root@SAE:~# more /etc/apache2/pass`
  - `admin:$apr1$EO2/RisZ$TnmBjFf4zzRZTZNn0AdTw1`

- <u>*Le fichier de configuration*</u> : afin que la protection par mot de passe s'applique à toutes les pages Web contenues dans un sous-répertoire du serveur, il faut définir une section « Directory ». Par exemple si la partie qu'on veut protéger est accessible par

http://www.monsite.dom/dossier1/privee/, il faut créer la section suivante dans le fichier de configuration principal de apache :

```
<Directory "/var/www/html/dossier1/privee">
      AuthType Basic
      AuthName "Veuillez saisir votre mot de login/passe"
      AuthUserFile "/etc/apache2/pass"
      Require valid-user
</Directory>
```

- La directive "AuthName" (lorsqu'elle est supportée par le navigateur) permet de préciser un texte qui sera affiché lors de la demande de mot de passe.
- La directive « Require » précise que pour avoir accès à ce dossier il faut s'authentifier. Cette approche est appelée authentification de type basique (`AuthType Basic`)

- Sous Debian, le fichier /etc/mime.types est le fichier qui contient les règles applicables à l'ensemble du système pour la mise en correspondance des suffixes des noms de fichier avec les types de media (types MIME). Il est fourni par le paquet media-types. Chaque ligne du fichier est une déclaration de type de media. Pour un service comme le Web, cela permet de traiter les fichiers d'extension « .gif » comme des images et les envoyer au navigateur du client en l'informant qu'il s'agit bien d'image (i.e. contenu binaire à décoder avec un code qui peut afficher les pixels de l'image dans le navigateur). Par exemple, voici des suffixes de fichier mis en correspondance avec des types de media application :

```
application/java-archive          jar
application/msword                doc dot
application/pdf             pdf
image/gif                   gif
```

7. Le fichier de configuration du module php (/etc/apache2/mods-available/phpX.Y.conf) contient par défaut la section :

```
<FilesMatch ".+\.ph(ar|p|tml)$">
    SetHandler application/x-httpd-php
</FilesMatch>
```

   ⇨ À votre avis à quoi correspond la chaîne de caractères : « .+\.ph(ar|p|tml)$ » ?
   ⇨ Même question pour la chaîne « ^[^.]+$ »
   ⇨ Essayez de donner une interprétation au reste du contenu de la section précédente ligne par ligne

La directive ForceType : (comme SetHandler) force l'identification du type MIME des fichiers spécifiés à la valeur de l'argument donné. Exemple : ForceType image/gif.
SetHandler : peut être utilisé comme ForceType et avec la même syntaxe. Mais elle est plus générique que ForceType car elle eut être utilisée pour d'autre fonctionnalité comme gérer des liens, etc.

8. Pour utiliser les bases de données, nous avons MySQL : un système de gestion de base de données relationnelle (SGBDR) open source basé sur un langage de requête structuré (SQL). MySQL peut être exécuté dans les environnements UNIX, Linux et Windows avec une compatibilité totale.

Il est relativement facile d'installer un serveur MySQL, que ce soit pour faire un environnement de développement ou de production. En effet, il suffit d'installer « mysql-server » avec « apt-get install » qui vous l'installe en quelques secondes avec une configuration par défaut. Pourtant, un serveur MySQL n'est pas exploitable tout de suite : le mot de passe root n'est pas initialisé, des privilèges anonymes existent, etc. Il est préférable

d'accorder de l'importance à la sécurité au début d'un projet, que d'essayer d'y revenir plus tard.

La commande « mysql_secure_installation » (/usr/bin/mysql_secure_installation) va apporter un minimum de sécurité pour vos nouvelles installations. Voici les questions qu'elle demande une fois lancée :

```
Securing the MySQL server deployment.
Enter password for user root:
VALIDATE PASSWORD COMPONENT can be used to test passwords and improve security. It
checks the strength of password and allows the users to set only those passwords
which are secure enough. Would you like to setup VALIDATE PASSWORD component?
Press y|Y for Yes, any other key for No: ???
Change the password for root ? ((Press y|Y for Yes, any other key for No) : ???
By default, a MySQL installation has an anonymous user, allowing anyone to log into
MySQL 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? (Press y|Y for Yes, any other key for No) : ???
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? (Press y|Y for Yes, any other key for No) : ???
By default, MySQL 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? (Press y|Y for Yes, any other key for No) :
???
Reloading the privilege tables will ensure that all changes made so far will take
effect immediately.
Reload privilege tables now? (Press y|Y for Yes, any other key for No) : ???
All done!
```

9. Récapitulez ce qu'elle permet de faire cette commande ? Quelle sont les réponses que vous donnerez aux différentes questions ?
   1. Voici quelques requêtes SQL utiles pour la manipulation des bases de données. Attention, ceci n'est qu'un exemple, ce n'est pas l'unique ni la seule manière de le faire :
      - CREATE DATABASE networkSae; : créer nouvelle base de donnée nommée « networkSae »
      - CREATE TABLE : permet de créer une table en SQL. Une table peut être vue comme un tableau ou une entité appartenant à une base de données pour stocker des données ordonnées dans des colonnes. La création d'une table sert à définir les colonnes et le type de données qui seront contenus dans chacun des colonne (entier, chaîne de caractères, date, valeur binaire …). Exemple :

```
CREATE TABLE salles
(   numero INT PRIMARY KEY NOT NULL,
    commentaire VARCHAR(100),
    construction DATE
)
```

   numero : identifiant unique qui est utilisé comme clé primaire et qui n'est pas nulle
   construction : date au format AAAA-MM-JJ
      - DROP TABLE nom_table ; : effacer une table de la base de données courante
      - SHOW DATABASES ; : afficher les bases de données existantes
      - use <nom_de_base_de_données> ; : sélectionner une base de donnée par son nom (naviguez entre bases en changeant de nom de la base dans « use »)
      - show tables ; : affiche les tables de la base de données courante
      - SHOW TABLES FROM database_name ; : affiche les tables de la base de données avec son nom (sans être obligé à choisir la base de données avec « use »)

- SELECT * FROM table_name; : affiche le contenu d'une table
- \c (cou contrôle + c): interrompre sous console MySQL
- \q : quitter la console MySQL

2. Et voici quelques fonction PHP utiles pour la manipulation des bases de données :
   - mysqli_connect : se connecter au serveur de base de donnée MySQL
   - mysqli_select_db (mysqli $link , string $dbname) : bool : sélectionner une base de donnée existante sur le serveur. $link est la connexion renvoyée par la commande mysqli_connect
   - mysqli_query ($link,$sql) : exécuter la requête SQL $sql (exemple "SELECT * FROM table_name") sur la base de donnée à laquelle on est connecté avec $link
   - mysqli_error($link) : affiche l'erreur relative à la connexion $link
   - mysqli_fetch_row : récupère une ligne de résultat d'une requête sous forme de tableau indexé. Chaque case du résultat (indexe 0 par exemple) correspond à une colonne de résultat. Par exemple, à la requête *"SELECT Name, CountryCode FROM City ORDER BY ID DESC"*, la case d'indexe 0 de la réponse correspond à la colonne « Name ».

10. Voici un code php. Attention, ceci n'est qu'un exemple, ce n'est pas l'unique ni la meilleure ni la plus sécure manière de le faire :

```
<?php
        $sql="SHOW DATABASES";
        $link = mysqli_connect('10.1.2.3', 'root', 'password') or die ('Error
        connecting to mysql: ' . mysqli_error($link).'\r\n');

        if (!($result=mysqli_query($link,$sql))) {
                printf("Error: %s\n", mysqli_error($link));
        }

        while( $row = mysqli_fetch_row( $result ) ){
                if (($row[0]!="information_schema") && ($row[0]!="mysql")) {
                    echo $row[0]."<br/>\r\n";
                }
        }
?>
```

   - Essayez d'interprétez ce qu'il fait le code.
   - Quelle requête SQL est utilisé ?
   - Modifiez le code pour que l'affichage soit inconditionnel

Sous certaines conditions, lorsqu'on essaie d'utiliser le script php précédent, le journal d'erreur du serveur Web Apache (le /var/log/apache/error.log) peut nous signaler l'erreur suivante :

```
[Sun May 01 17:00:10.816875 2022] [php7:error] [pid 6142] [client
127.0.0.1:47936] PHP Fatal error:  Uncaught Error: Call to undefined
function mysqli_connect() in /var/www/html/showdb.php:XX\nStack
trace:\n#0 {main}\n  thrown in /var/www/html/showdb.php on line XX
```

11. Donnez une explication possible à cette erreur
12. L'installation du package php7.4-mysql a donné ce qui suit :

```
root@SAE:~# apt install php7.4-mysql
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Les NOUVEAUX paquets suivants seront installés :
  php7.4-mysql
```

```
0 mis à jour, 1 nouvellement installés, 0 à enlever et 0 non mis à jour.
Il est nécessaire de prendre 121 ko dans les archives.
Après cette opération, 470 ko d'espace disque supplémentaires seront utilisés.
Réception de :1 http://deb.debian.org/debian bullseye/main amd64 php7.4-mysql amd64 7.4.28-1+deb11u1 [121 kB]
121 ko réceptionnés en 0s (689 ko/s)
Sélection du paquet php7.4-mysql précédemment désélectionné.
(Lecture de la base de données... 110640 fichiers et répertoires déjà installés.)
Préparation du dépaquetage de .../php7.4-mysql_7.4.28-1+deb11u1_amd64.deb ...
Dépaquetage de php7.4-mysql (7.4.28-1+deb11u1) ...
Paramétrage de php7.4-mysql (7.4.28-1+deb11u1) ...

Creating config file /etc/php/7.4/mods-available/mysqlnd.ini with new version
Creating config file /etc/php/7.4/mods-available/mysqli.ini with new version
Creating config file /etc/php/7.4/mods-available/pdo_mysql.ini with new version
Traitement des actions différées (« triggers ») pour libapache2-mod-php7.4 (7.4.28-1+deb11u1) ...
Traitement des actions différées (« triggers ») pour php7.4-cli (7.4.28-1+deb11u1) ...
root@SAE:~#
```

⇨ Quel dossier a été modifié ? À votre avis c'est quoi son rôle ? Faut-il redémarrer Apache ?

Une partie du fichier de configuration (par défaut) de Apache 2 :

```
# This is the main Apache server configuration file.  It contains the
# configuration directives that give the server its instructions.
# See http://httpd.apache.org/docs/2.4/ for detailed information about
# the directives and /usr/share/doc/apache2/README.Debian about Debian specific
# hints.
# It is split into several files forming the configuration hierarchy outlined
# below, all located in the /etc/apache2/ directory:
#
#	/etc/apache2/
#	|-- apache2.conf
#	|	`--  ports.conf
#	|-- mods-enabled
#	|	|-- *.load
#	|	`-- *.conf
#	|-- conf-enabled
#	|	`-- *.conf
#	`-- sites-enabled
#		`-- *.conf
# Global configuration
# ServerRoot: The top of the directory tree under which the server's
# configuration, error, and log files are kept.
#
# NOTE!  If you intend to place this on an NFS (or otherwise network)
# mounted filesystem then please read the Mutex documentation (available
# at <URL:http://httpd.apache.org/docs/2.4/mod/core.html#mutex>);
# you will save yourself a lot of trouble.
# Do NOT add a slash at the end of the directory path.
#ServerRoot "/etc/apache2"
#
# Timeout: The number of seconds before receives and sends time out.
Timeout 300
# KeepAlive: Whether or not to allow persistent connections (more than
# one request per connection). Set to "Off" to deactivate.
#
KeepAlive On
```

```
#
# MaxKeepAliveRequests: The maximum number of requests to allow
# during a persistent connection. Set to 0 to allow an unlimited amount.
# We recommend you leave this number high, for maximum performance.
#
MaxKeepAliveRequests 100
# KeepAliveTimeout: Number of seconds to wait for the next request from the
# same client on the same connection.
#
KeepAliveTimeout 5
# These need to be set in /etc/apache2/envvars
User ${APACHE_RUN_USER}
Group ${APACHE_RUN_GROUP}
#
# HostnameLookups: Log the names of clients or just their IP addresses
# e.g., www.apache.org (on) or 204.62.129.132 (off).
# The default is off because it'd be overall better for the net if people
# had to knowingly turn this feature on, since enabling it means that
# each client request will result in AT LEAST one lookup request to the
# nameserver.
#
HostnameLookups Off
# ErrorLog: The location of the error log file.
# If you do not specify an ErrorLog directive within a <VirtualHost>
# container, error messages relating to that virtual host will be
# logged here.  If you *do* define an error logfile for a <VirtualHost>
# container, that host's errors will be logged there and not here.
#
ErrorLog ${APACHE_LOG_DIR}/error.log
#
# LogLevel: Control the severity of messages logged to the error_log.
# Available values: trace8, ..., trace1, debug, info, notice, warn,
# error, crit, alert, emerg.
# It is also possible to configure the log level for particular modules, e.g.
# "LogLevel info ssl:warn"
#
LogLevel warn
# Include module configuration:
IncludeOptional mods-enabled/*.load
IncludeOptional mods-enabled/*.conf

# Include list of ports to listen on
Include ports.conf
# Sets the default security model of the Apache2 HTTPD server. It does
# not allow access to the root filesystem outside of /usr/share and /var/www.
# The former is used by web applications packaged in Debian,
# the latter may be used for local directories served by the web server. If
# your system is serving content from a sub-directory in /srv you must allow
# access here, or in any related virtual host.
<Directory />
        Options FollowSymLinks
        AllowOverride None
        Require all denied
```

```
</Directory>
```