Préparation à la mise en situation professionnelle simulée individuelle de maintenance d’une application - DFS

L'objectif de ce document est de vous préparer au mieux à l'évaluation. Il contiendra donc des informations essentielles pour réussir l'examen.

Sujet

A partir d’une application web existante et comportant des bugs techniques, des failles de sécurité et reposant sur certains composants obsolètes ou nécessitant une amélioration, les candidats mettent en œuvre les compétences du bloc 4, en salle surveillée.

Prés-requis techniques

- HTML / CSS / Javascript
- Bootstrap
- PHP 8
- MySQL > 5.7

Environnement

L’environnement fourni comprend une application web existante. Elle se compose principalement :

- D’un front-end
- D’un back-end Rest
- D’une base de données relationnelle

Pour cette mise en situation, nous considérerons que l'environnement de PRODUCTION sera configuré de la manière suivante :

- Un VPS sous Debian réservé chez OVH avec accès SSH : https://ovhcloud.com/fr/vps/

L'offre Starter est suffisante pour de petits projets personnels

Compétences visées

Bloc 4 : Déployer et assurer le maintien en production d'une application web

- C27 : Produire la documentation technique d'une application web

- C28 : Administrer l'enregistrement de noms de domaines et de certificats de sécurités pour une application web
- C29 : Sélectionner une plateforme d'hébergement dans un environnement dédié
- C30 : Administrer des services d'hébergement dans un environnement dédié
- C31 : Mettre en œuvre un système de déploiement automatisé d'une application pour différents types d'hébergement, en respectant les bonnes pratiques DevOps, pour être en mesure d'assurer une livraison en continu.
- C32 : Mettre en œuvre un système de supervision d'une application et de services d'hébergement.
- Mesures correctives et amélioration de l'application web

C27 : Produire la documentation technique d'une application web

Le code source est documenté

Pour commenter le code source correctement en PHP il est nécessaire de comprendre et maîtriser l'utiliser de la "PHPDoc" :

- https://docs.phpdoc.org/guide/references/phpdoc/index.html
- https://phpstan.org/writing-php-code/phpdocs-basics

L'utilisation de l'outils PHP Stan permet de s'assurer que les bonnes pratiques en terme de syntaxe sont bien respectées.

La documentation de l'API est rédigée

Un des outils les plus utilisé pour rédiger la documentation d'une API est "Swagger" : https://swagger.io/

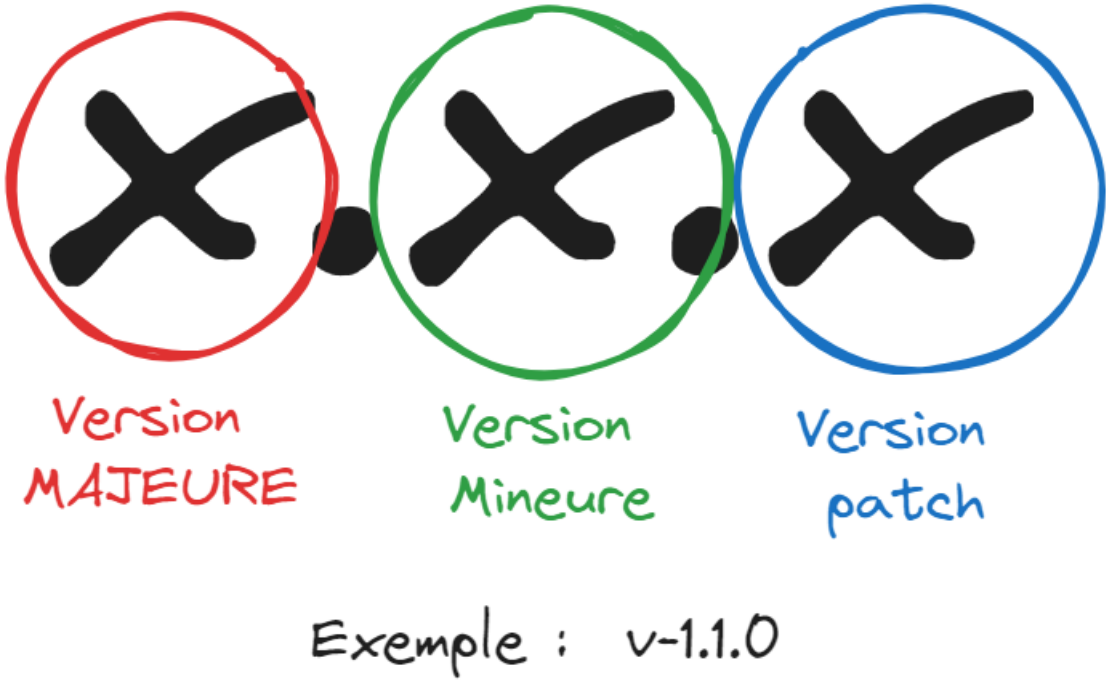
Il existe un projet sur GitHub permettant justement de générer une documentation de type Swagger pour un projet PHP via la PHPDoc : https://github.com/zircote/swagger-php (déjà installé sur le projet)

Les mises à jour relatives aux évaluations sont portées dans le journal des évolutions.

La liste des modifications est correctement structurée (présentée par ordre décroissant de commit, et par regroupement par version).

Pour ce faire, il est conseiller de créer un fichier CHANGELOG.md (synthaxe markdown) à mettre à jour à chaque évolution (Monté de version, Mise à jour, Patch).

Il est nécessaire de s'imposer une règle pour les numéros de version, comme ci-dessous :



Des outils comme "Git Cliff" permettent la génération de fichiers CHANGELOG.md en se basant sur l'historique du dépôt Git : https://git-cliff.org/

C28 : Administrer l'enregistrement de noms de domaines et de certificats de sécurités pour une application web

Un nom de domaine a été réservé. Les déclarations administratives ont été effectuées.

Réservez un nom de domaine en ligne est possible sur plusieurs plateforme.
La plus utilisée en France est sans doute OVHCloud.

La procédure pour réserver un nom de domaine se trouve ici :
https://www.ovhcloud.com/fr/domains/

Les serveurs de noms correspondants sont configurés et fonctionnels. Les bonnes pratiques de sécurité ont été appliquées.

Une fois le nom de domaine enregistré il sera nécessaire de le rediriger vers notre serveur en modifiant la valeur des champs A (IPv4) et AAAA (IPv6) dans la zone DNS.

Cette modification se fait via l'interface admin du compte OVHCloud qui a réservé le nom de domaine.

Ressource : https://help.ovhcloud.com/csm/fr-dns-edit-dns-zone?id=kb_article_view&sysparm_article=KB0051684

Les certificats de sécurités ont été créés. Ils sont installés, configurés et fonctionnels pour les différents services.

En ce qui concerne les certificats de sécurité, nous verrons comment paramétrer un système d'administration dédié sur notre VPS.

C29 : Sélectionner une plateforme d'hébergement dans un environnement dédié

L’architecture technique d’hébergement proposée est adaptée à l’application.

Pour des projets d'application web de ce type, le plus simple est de recourir à un VPS sous distribution Linux (Debian, CentOS, Ubuntu).
Le VPS doit absolument être accessible via SSH afin de pouvoir paramétrer correctement l'environnement de production.

Pour accéder en SSH au serveur il est nécessaire de connaître la méthode en fonction de son OS :

- Windows : l'utilisation du logiciel Putty se montre très pratique ici
- Linux : https://www.digitalocean.com/community/tutorials/how-to-use-ssh-to-connect-to-a-remote-server-fr / https://doc.ubuntu-fr.org/ssh
- Mac OS : https://blog.lws-hosting.com/serveur-dedie/comment-se-connecter-en-ssh-avec-mac-os-x-et-windows/

C30 : Administrer des services d’hébergement dans un environnement dédié

Les services d’hébergement sont installés, configurés et fonctionnels.

Un gestionnaire de configuration a été utilisé.

Afin de simplifier la configuration de notre VPS, nous recommandons l'utilisation de l'outils d'administration open-source aaPanel : https://www.aapanel.com/new/index.html

Celui est très simple a installé et se montre très complet !

Voici comment l'installer : https://www.aapanel.com/new/download.html#install

Cet outils permet notamment de se passer de l'installation d'un LAMP from-scratch car il contient déjà tous les éléments suivants :

- Apache / Nginx
- PHP
- MySql / MariaDB
- Génération de certificat SSL Let's Encrypt

VU EN COURS COMMENT L'INSTALLER ET LE PARAMETER POUR UN PROJET PHP

C31 : Mettre en œuvre un système de déploiement automatisé d’une application pour différents types d’hébergement, en respectant les bonnes pratiques DevOps, pour être en mesure d’assurer une livraison en continu.

Le meilleur moyen de déployer son code sur un VPS avec la configuration donnée est d'utiliser GIT.

Pour ce faire :

- Git doit être installé sur le VPS
- Un dépôt Git doit être créer sur le serveur (git init --bare)
- Les permissions de fichiers doivent correctement être paramétrée (autorisation du groupe WWW)
- Un script de déploiement doit être rédigé afin d'optimiser le déploiement continu

Notes : il est toujours possible de paramétrer des actions CI/CD via GitHub afin d'automatiser d'avantage le déploiement sur le serveur.

VU EN COURS COMMENT EFECTUER CE PARAMETRAGE

C32 : Mettre en oeuvre un système de supervision d’une application et de services d’hébergement, définir des sondes et des alertes (état des services et

sécurité), pour détecter, diagnostiquer et analyser l’origine de bugs, problèmes techniques, et failles de sécurité puis déployer des mesures correctives dans des délais adaptés.

IMPORTANT : L'utilisation de aaPanel permet de couvrir cette compétence.

Mesures correctives et amélioration de l'application web

Traitement des failles de sécurités

Faille XSS

Pour se prémunir d'une faille XSS en PHP, il est important de ne jamis faire confiance aux données transmises à l'application.
Cela se fait en général par l'url en passant un paramètre (/?param=) ou via les champs d'un formulaire.

Exemple :

En PHP, voici une manière simple de s'en prémuni :

Considérons que nous avons un script PHP permettant d'afficher les message "Bonjour {prenom}" où "prenom" est une valeur passée via l'url : http://monscript.php/?prenom=John

```
// PAS BIEN !
<p>Hello <?= $_GET['prenom'] ?></p>
```

```
// BIEN !
<p>Hello <?= htmlspecialchars($_GET['prenom']) ?></p>
```

L'utilisation de la fonction htmlspecialchars permet de convertir les données afinde ne pas exécuter le code HTML.

Documentation : https://www.php.net/manual/fr/function.htmlspecialchars.php

Injection SQL

Pour se protéger d'une injection SQL en PHP, il faut respecter le même principe que pour les injections de scripts (XSS).
L'idée est de toujours traiter les données entrantes avant de les utiliser dans une requête SQL.

Exemple:

Considérons un script permettant de traiter un formulaire de connexion (en POST) et qui va récupérer les informations de l'utilisateur à partir de l'adresse email saisie.

```php
<?php
// PAS BIEN
$sql = "SELECT * FROM user WHERE email = '". $_POST['email'] ."'";
$stmt = $pdo->query($sql);
```

Dans l'exemple ci-dessus, un visiteur malveillant pourrait utiliser cette injection dans le champs "email" :

```
' OR 1=1;//
```

``

Ce qui donnera l'interprétation du la requête suivante :

```
SELECT * FROM user WHERE email = '' OR 1=1;//'
// Retournera toujours VRAIE
```

Pour résoudre ce problème, l'utilisation de la fonction htmlspecialchars peut être une bonne solution car celle-ci va retirer les apostrophes (quotes) de la chaîne traitée.

```php
<?php
// BIEN !!!
$sql = "SELECT * FROM user WHERE email = '".
htmlspecialchars($_POST['email']) ."'";
$stmt = $pdo->query($sql);
```

OU alors, utiliser les requête préparée avec PDO :

```php
<?php
// BIEN !!!
$sql = "SELECT * FROM user WHERE email = :email";
$stmt = $pdo->prepare($sql);
$stmt->execute([
        'email' => $_POST['email']
]);
```

Pour en savoir plus : https://www.php.net/manual/fr/mysqli.quickstart.prepared-statements.php

Parcours d'arborescence

Cette faille vient plutôt de la configuration du serveur et non du code. En effet celle-ci repose sur le fait que l'arborescence des répertoires du projet est accessible et consultable (cf. exemple ci-dessous).

Il existe plusieurs solutions l'éviter.

Solution 1 : créer des fichiers index.php vides dans chaque sous-dossier du projets
En effet, les serveurs web comme Apache ou NGinx sont paramétrer pour servir par défaut les fichiers "index" lorsqu'ils sont présents. Ceci évite que l'arborescence ne s'affiche dans le navigateur.

Solution 2 : mettre les ressources publiques dans un dossier dedié et paramétrer le serveur afin que le projet pointe vers celui-ci.

Solution 3 : paramétrer le serveur web afin de bloquer le parcours de répertoires

Apache : création d'un fichier .htaccess à la racine du projet

```
//Prevent directory listings
Options All -Indexes
```

Outil utile : https://www.htaccessredirect.net/

Nginx : https://www.digitalocean.com/community/tools/nginx?global.app.lang=fr

Accès via l'URL

Cette faille consiste à accéder à une url qui normalement devrait être protégée. Dans le parcours utilisateur, cette url est accessible lorsque l'on est identifié mais que se passe-t-il si

un visiteur lambda essaie d'accéder à celle-ci. Ce type d'url est en général réservé aux interfaces admin et commence souvent pas "/admin".

Pour s'en protéger, il faut donc s'assurer qu'un utilisateur est bien authentifié et qu'il possède les bon droits pour accéder à ces url.

Les mécanismes d'authentification simples utilisent la session pour stocker l'information que l'utilisateur est bien connecté. Il faut donc vérifier que la session existe et qu'un utilisateur est réellement connecté.

Exemple :

Supposons que notre script d'authentification, stock l'id de l'utilisateur en session après avoir vérifié ses identifiants (email + mot de passe).

Voici comment vérifier que l'utilisateur est bien connecté :

```php
<?php

session_start(); // Indispensable pour accéder aux informations de la session

// On vérifie que l'id de l'utilsateur existe en session et qu'il n'est pas vide
if(empty($_SESSION('user_id')))
{
        // S'il est vide on redirige vers la page de login et on quitte le programme
        header('Location: /login.php');
        exit;
}

// Utilisateur autorisé
```

Faille CSRF

Cette faille est un peu plus complexe à expliquer donc voici quelques ressources utiles :

Grafikart : https://youtu.be/58y6772MHKw?feature=shared
Blog du hacker : https://www.leblogduhacker.fr/faille-csrf-explications-contre-mesures/

Ressources complémentaires :

- PHP - Les failles et attaques courantes - Comment se protéger ? https://youtu.be/87IjAi7qvv0?feature=shared

#dfs