Module Drupal

Mon tuteur de stage me confie un nouveau projet. Ce projet consiste à créer un module permettant de vérifier si les sites client que Archriss ont en charge. Si un dès site ne fonctionne plus, une alerte est déclenchée, qui avertit les responsables qu’un problème est survenu sur l’un des sites clients. Ce besoins survient, suite à l’appel de différents clients qui signalés un problème sur leur site. Le but étant d’être averti avant le client, qu’un problème est survenu et non l’inverse. L’équipe d’Archriss décide de monter un site Drupal, sur l’un de leur serveur, sur lequel j’intégrais mon script permettant de vérifier si les sites fonctionnent. Du fait de l’utilisation du CMS, il y a une base de données qui également mit en place. Il me fournisse un accès administrateur pour le site, et les identifiants FTP au serveur.

Pour vérifier que le site fonctionne, il va être stocké dans la base de données l’url du site, pointant vers une page précise du site. Dans une autre table de la base de données, va être stocké un mot, un élément se trouvant sur la page. Ce mot ou élément devra provenir de la base de données du site vérifié. Ainsi, il sera vérifié en même temps le site et le service web, et la base de données. Mon tuteur de stage, m’explique que tout le contenu d’un site utilisant un CMS est stocké dans la base de données, et non dans des pages au format html ou php, comme je le croyais. Ainsi la plupart des éléments d’une page provient de la base de données.

Dans ce script il y a deux étapes. La première qui permet de vérifier si la connexion au serveur, et à la page qui l’héberge s’est réalisée correctement. Et la seconde qui est de s’assurer que le mot de vérification est bien dans le contenu de la page.

Une fois le cahier des charges rédigé, et mon planning établit, je commande la programmation du module. Je cherche tout d’abord à utiliser les fonctions SQL intégrées avec Drupal. Je réalise quelques recherches sur internet, dans lesquelles j’apprends que les fonctions SQL ont connus une modification majeure avec la version 7. En effet, il u l’apparition de la fonction « db\_select ». Cette fonction permet une extraction des données permettant de ne pas connaitre la syntaxe SQL. En effet, cette fonction permet au néophyte de ne pas être confronté à la rugosité du langage SQL. Cependant il faut quand même comprendre le fonctionnement, et les rouages d’une base de données. Mais avant d’utiliser cette fonction, il faut d’abord inclure les fichiers nécessaires pour l’utilisation des fonctions SQL. Il faut faire appel au fichier « bootstrap.inc », qui le fichier central de configuration de Drupal. Il faut ensuite utiliser la fonction « drupal\_bootstrap », qui permet d’amorcer la config de Drupal et charge les fichiers php. En passant en paramètre la phase « [DRUPAL\_BOOTSTRAP\_FULL](https://api.drupal.org/api/drupal/includes%21bootstrap.inc/constant/DRUPAL_BOOTSTRAP_FULL/7)**», cela me permet de charger l’intégrité des fonctionnalités offerte par Drupal.**

Une fois cela fait, j’extrais de la base les URLs qui peuvent y être stockées. Et je les affiche pour réaliser des tests. Mon tuteur de stage m’a signalé qu’il fallait créer des fonctions, pour diviser un maximum le travail.

Une fois les URLs récupérées, je les passe en paramètre d’une fonction qui récupère le contenu de la page, en utilisant le CURL. Cette méthode permet une grande configuration de connexions à un serveur en utilisant au choix un large panel de protocoles, dont http. L’utilisation du CURL permet de connaître les raisons en cas d’échec de connexion au serveur. Grâce à cela, il sera plus facile de connaître du dysfonctionnement du serveur, ce qui permettra une résolution du problème plus rapide. C’est donc la première étape qui est accomplie ici. Mais je suis confronté à une authentification htaccess. Je signale donc ce problème à la personne responsable du site, et lui propose une évolution future qui permettra de stocker dans la base de données les identifiants htaccess. D’autant plus qu’avec CURL, il est simple de s’authentifier. Il suffit de définir l’option CURL « CURLOPT\_USERPWD ». Une fois ce problème réglé, il faut vérifier si l’élément stocké dans la base en corrélation avec la page est bien présent sur celle-ci. Pour cela, je passe en paramètre d’une fonction, l’identifiant de la page stockée dans la base pour récupérer le mot rattaché à la page. Une fois le mot récupéré, un seconde paramètre qui est le contenu de la page, va me permettre de vérifier si le mot est stocké dans la base est présent dans le contenu de la page. Pour faire ceci j’utilise la fonction « substr\_count », qui compte le nombre de fois qu’apparaît le mot dans le contenu de la page. Le script accomplie la deuxième étape importante du script. Je me rends compte qu’un mot apparaît 41 fois, alors que sur le site il n’apparaît qu’une seule fois. En analysant le code source grâce à mon navigateur, je me rends compte qu’il y a beaucoup de balise « href », contenant le mot de vérification. Alors je décide d’utiliser la fonction native à php, qui se nomme « strip\_tags », qui supprime les balises HTML et PHP d'une chaîne.

Je me débutai alors dans programmation de la fonction d’envoi de mail en cas de problème rencontré. Je commençais par faire des recherches, pour utiliser l’api de Drupal. Il y a en effet, une fonction permettant d’envoyer des mails. Après plusieurs recherches, et des tests infructueux (je recevais bien les mails, mais il y avait aucun contenu, ni d’objet), je me demande alors un peu d’aide à un des membres de l’équipe, possédant une solide expérience sur Drupal. Il me fournit alors un extrait de code qu’il a réalisé, en me l’expliquant. Je réadapte ce code pour mes besoins, en changeant l’adresse de l’expéditeur, et l’objet du mail en autre. Et réalise des tests, concluant cette fois-ci. Étant donné qu’il peut y avoir soit une erreur est due au serveur web, soit au serveur de base de données. Il faut alors qu’il est des messages différents, dans les mails. Alors la fonction prend un paramètre un entier qui selon la valeur passée, écrit un message différent. Pour cela, j’utilise une structure « switch ».

Je souhaitais aussi, dans le message, intégré le nom du projet sur lequel porte le problème. Je me renseigne donc auprès de la personne qui a monté le site, pour lui demander où se trouve l’information il me communique le nom de la table, mais surtout, il me dit que grâce à une fonction, il est possible de récupérer soit une occurrence de cette table, en passant paramètre l’identifiant de cette dernière (exemple : taxonomy\_term\_load($nid)), soit utilisé une fonction quasi-similaire qui récupère toutes les occurrences (exemple : taxonomy\_term\_load\_multiple()). Il me dit que pour chaque table, il existe des fonctions permettant de faire la même extraction des données.

Cette information fait que je dois revoir l’intégralité de mon code. Une des fonctions que j’avais créées, permet d’extraire des données. Avec cette découverte, cette fonction devient inutile. La valeur retournée par ces fonctions est un objet de la « stdClass ». Les données sont organisées en tableau. Cependant, j’ai eu un problème pour extraire les informations de ce tableau. En effet, au lieu d’utiliser les crochets, comme habituellement, j’utilise un tiret-chevron (comme ceci ->). C’est suite à une recherche sur internet que j’ai pu trouver cette solution.

C’est alors que j’ai pu intégrer le nom du projet dans mon mail. Il me faut à présent récupèrer l’adresse mail du destinataire. Comme ces données sont stockées dans la base de données, il existe une fonction pour récupérer toutes les données de la base, comme il expliqué plus haut. Il me faut juste l’identifiant de la personne en charge du projet. Comme ayant récupèrer toutes les informations du « node », j’ai donc l’ « uid » du reponsable. Je passe donc en paramètre l’ « uid » dans la fonction « mailAlert », et dans cette fonction je fais appel à « user\_load » qui me permet de récupérer toutes les informations, dont l’adresse mail.

Ayan était confronté aux problèmes d’une authentification htaccess, la personne en charge du site a mis en place le stockage des login et mot de passe. De plus je les récupère déjà grâce à l’appel de la fonction « taxonomy\_term\_load ». Il me suffit juste de les trouver dans l’objet renvoyé, vérifié qu’ils ne sont pas vides, et si ce n’est pas le cas, d’utiliser l’authentification CURL, avec les identifiants retournées par la fonction.

Mon projet commence à bien être complet, et de plus j’ai une légère avance sur mes prévisions, car j’ai pu réaliser la documentation un jour plus tôt.