**Carnet de bord**

Lors de mon premier jour de stage, mon tuteur m’a expliqué la mission qu’il me confié. Il m’a tout d’abord présenté l’entreprise cliente qui leur a confié la gestion de leur site de vente. C’est une entreprise qui propose des carnets de chèques cadeau, utilisable dans de grandes enseignes commerciales et de la distribution. Mon tuteur m’explique que ces chèques cadeaux est un bon moyen de récompenser les salariés d’une entreprise en fin d’année par exemple, sans être imposable, a contrario du prime ou d’une augmentation de salaire. Il m’explique ensuite, que tous les matins en arrivant sur son lieu de travail, mon tuteur extrait toutes les commandes la veille, qui ont été payées. Plus précisément de 21 heures d’avant-hier à 21 heures hier. Il extrait ces commandes dans un fichier .csv grâce à phpmyadmin. Il va ensuite le convertir en un fichier Excel (format .xls). Il souhaite que cette manipulation soit automatisée. Il me fournit les identifiants pour phpmyadmin, au serveur ftp, et le site du client étant protégé par un « htaccess », il m’a donné les identifiants nécessaires. Il me fournit également la requête qu’il exécute tous les matins. Il me conseille l’éditeur de texte « NetBeans », pratique et efficace, notamment pour programmé avec des fichiers provenant d’un serveur ftp. Ce logiciel permet d’envoyer les fichiers modifiés, en écrivant par-dessus le fichier déjà existant sur le serveur. Mon tuteur m’a signalé qu’il reprenait la maintenance du site, et que le site embarqué une framework, « codeigniter ». Il s'agit d'une boite à outils d'aide à la construction de sites web. Son but est de permettre d'améliorer le temps de développement de projets en fournissant un ensemble complet de bibliothèques prenant en charge les tâches les plus répétitives. J’ai donc dans un premier temps, apprit à utiliser donc ce nouveau logiciel et cette framework.

Après quoi j’ai commencé le vif du travail. Mon tuteur stage m’a donc conseillé de copier/coller un fichier qui se nommait « cron.php ». Ce fichier m’a permis d’exemple et de base de travail. J’ai étudié la classe de base de données de codeigniter, pour l’exécution de requête et l’exploitation des données. J’ai réalisé des tests, avec de simple « echo », pour afficher le nom de champs ainsi que les données retournée par la requête.

Une fois mes tests réussis, je me suis mis à la recherche d’une solution pour la création de tableur. J’ai alors trouvé une solution qui utilisé simplement des fonctions php natives. Après quelques adaptions pour mon projet, j’ai réussi à exporté les données qui m’intéressé dans un tableur, qui se téléchargé.

Mais j’ai dû changer de solution, pour des raisons que j’explique dans un rapport.

Suite à la demande de mon tuteur, il souhaite recevoir les tableurs créés en pièce jointe de mail. Je me suis mis à la recherche d’information quant à l’envoie de mail avec php. J’ai alors trouvé un cours sur « Open Classroom », très détaillé, avec lequel j’ai compris comment envoyer un mail avec une pièce jointe. J’ai d’abord fait quelques tests, en m’envoyant des mails simples, puis j’ai ensuite envoyé des mails avec pièce jointe avec succès.

C’est alors que M.Petitfils a formulé la demande de journaliser les différentes étapes de l’exportation de ventes. Il m’a indiqué que la journalisation se faisait dans un simple fichier texte, mais qu’il devait y avoir une syntaxe similaire à un fichier .csv. Chaque ligne de ce fichier, correspond à une étape d’exportation. Cette ligne est composée de la date/heure d’exécution, du numéro d’étape, une information sur l’étape et une information complémentaire. Mon tuteur de stage, souhaitait journaliser la requête sql lorsque l’erreur de l’exportation provient de la base de données. Pour des raisons que j’explique dans un autre rapport, toutes les requêtes sont « logger » dans un autre fichier. Il souhaitait aussi journaliser les erreurs sql. « Codeigniter » le gère déjà en natif, il suffit juste changer la valeur d’une constante dans le fichier de configuration. Cela va également « logger » les erreurs PHP. A ce propos, « codeigniter » à plusieurs fichiers de configuration, mon tuteur souhaite centraliser ces configurations dans un même fichier, en l’occurrence « constant.php » en gardant tout de même tous les fichiers de la framework, de ce fait je crée une constante dans ce fichier, qui servira de configuration dans un autre fichier. Je fais ceci dès qu’une configuration ne figure pas dans le fichier « constant.php ».

Mais l’activation de la journalisation des erreurs sql, active également l’affichage d’erreur pour le client. Mon tuteur ne souhaite pas que les erreurs soient apparentes pour le client. J’ai alors modifié cette fonction. J’ai ajouté une constante dans « constant.php », et dans la fonction une structure conditionnelle vérifie la valeur de cette constante. Si elle est égale à 1 alors les erreurs s’afficheront sinon non.

Travaillant sur une version de test du site, et non pas sur celui de « prod », mon tuteur de stage veut que je consigne tous les modifications, notamment au niveau des fichiers de configuration dans une documentation détaillé que vous pouvez retrouver [ici](https://docs.google.com/document/d/1Qe9f2SZ-mNpCu-3veahbqd5dQs8orvWsG0f0B2EvCtg/edit?usp=sharing).

Le projet a été réalisé dans les temps prévu, malgré le fait que quelques problèmes sont survenus. Une réactivité et un investissement personnel a permis de respecter les prévisions.

Après relecture de la documentation, mon tuteur souhaite que je la réorganise de de manière plus logique, et que je supprimer le verbiage superflus afin de ne garder que l’essentiel. Cela ayant pour but de ne pas faire une documentation à rallonge et que les modifications à apporter soient faites rapidement.

Mon tuteur de stage souhaite ne pas afficher les erreurs PHP sur le navigateur afin que le client ne soit pas exposé à des informations qui ne le concerne pas, et qui gâche les pages du site. Il m’a alors confié la tâche de désactiver l’affichage des erreurs PHP. J’ai alors étudié l’affichage des erreurs, et j’ai effectué une recherche sur tous les fichiers (grâce à « NetBeans ») du message qui s’affiche dans le navigateur. Cette recherche m’a conduit à un fichier ne contenant que du html, n’était qu’en fait que la mise en forme des informations et qui était appelé par d’autre fichier. J’ai donc réalisé une autre recherche sur tous les fichiers, mais cette fois-ci avec le nom du fichier contenant le html, afin de savoir dans quel autre fichier il était appelé. Ce qui m’a conduit à un fichier « Exception.php » avec plusieurs fonctions PHP. Ces fonctions étaient l’affichage d’erreur quel qu’en soit la nature, et une seconde l’affichage des erreurs PHP spécifiquement. J’ai alors réutilisé une des constantes (« DISPLAY\_ERROR\_ENABLED ») du fichier « constants.php », afin d’activer ou non l’affichage des erreurs. C’est la même constante que l’affichage des erreurs SQL. J’ai donc mit tout le contenu de la fonction dans une structure conditionnel, qui vérifie que la valeur de la constante soit bien à 1. Si c’est exact, alors les erreurs php sont affichées, sinon non.

Sur le site de vente en ligne, j’ai réalisé du « wording ». C’est-à-dire de la modification de contenu de certaines pages. Le client souhaitait que certaines informations contenues dans certaines pages soient changées. Comme par exemple la hausse de la tva, ou le changement du montant maximum qu’un salarié peut recevoir en chèque cadeau. Le client a demandé également la suppression de lien hypertexte, ou la modification de certains d’entre eux. Mais aussi le changement de style d’un titre. Le client, lors de ses demandes de modifications, stipulé les pages concernées et les modifications a apportées.