Justine Tu

Rapport Web, BD
Cahier des charges et journal de bord

Objectif

Le principal objectif de mon site est d'apprendre le chinois à l'aide de mini-jeux. Je pensais récupérer des exercices et du vocabulaire provenant de mes manuels de licence de chinois, mais finalement je suis partie sur un niveau assez simple pour débuter. J'ai donc repris quelques éléments provenant de mes manuels de licence mais de manière générale les cours et les exercices proviennent de moi.
J'ai décidé d'appeler mon site chinois(pas)fun car j'espère rendre l'apprentissage un peu près amusant grâce à mon site.

Cibles

Mon site cible ainsi un public français, étant donné que mon site sera en français, souhaitant apprendre quelques bases du chinois.

Aspects, ergonomie et graphisme

- couleurs du site

Concernant les couleurs que l'on peut retrouver sur mon site, j'étais tout d'abord parti sur ces fonds là pour le mode clair / mode sombre :

J'ai finalement changé d'avis car je trouvais que les couleurs n'étaient pas assez sobres. Mon choix s'est donc porté sur ces 2 fonds, réalisés avec le site https://meshgradient.com/ :

J'ai décidé de partir sur ces couleurs-ci car je les trouvais assez neutres. En effet, comme c'est un site d'apprentissage, je me suis demandée si des couleurs un peu trop flash n'auraient pas eu tendance à déconcentrer l'utilisateur. J'ai également fait le choix de faire des divs en transparence et avec un effet de flou pour le footer.
Pour le texte dans le site, je suis restée sur une couleur sobre c'est-à-dire des nuances de gris et du noir.
Je suis également parti sur des formes arrondies afin que le site soit plus attractif et donc plus agréable à regarder.

- logo

Le logo de mon site est inspiré de plusieurs images que j'ai trouvées sur un site de conception de logo (Tailor Brands).

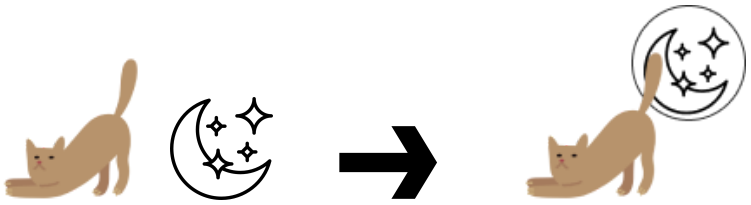
A partir de la première image, j'ai utilisé l'intelligence artificielle DALL•E 2 d'OpenAI pour créer une continuité au logo. Ma requête précisait qu'il fallait que le monsieur ait les mains sur les hanches (car il est là pour en découdre avec ses nouveaux élèves), et voici le résultat :

Une fois générée, j'ai décidé de le redessiner avec un fond qui a pour symbole la prospérité, la richesse en Chine et une écriture en français et en chinois du nom de mon site. Cette version est donc la première version de mon logo :

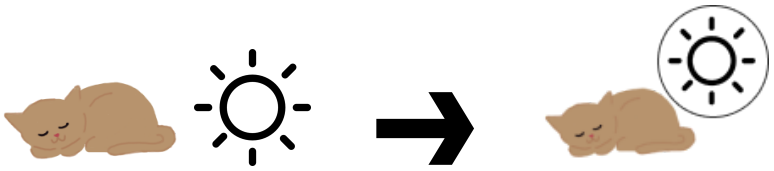
Après réflexion, j'ai décidé de changer la couleur de l'écriture de mon logo en noir car on ne voyait pas le jaune lorsque le mode clair était activé. Ceci est donc la version actuelle de mon logo :

- icône de changement de thème

Pour les icônes de changement de thème, j'ai repris ceux que j'avais réalisés pour le cours de techniques web. J'ai utilisé Figma pour associer ce dessin de chat au dessin de la lune pour passer au mode sombre.



Et n'ayant pas trouvé de dessin de chat me convenant pour passer au mode clair, j'ai décidé de le dessiner moi-même puis de l'associer au dessin du soleil.



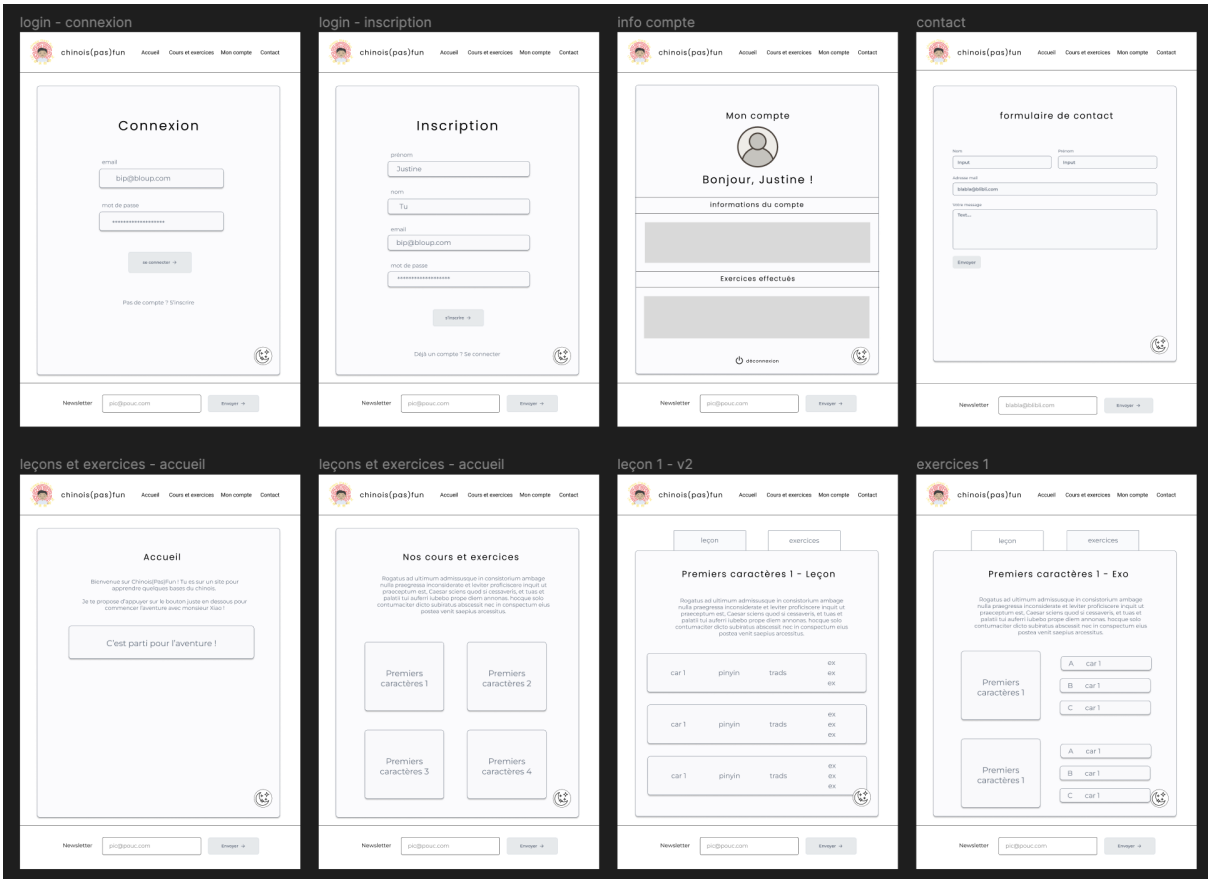
- polices

Mon site est composé de 3 polices :

- ➔ le nom de mon site est écrit avec la police ZCOOL KuaiLe disponible sur Google Fonts (ZCOOL KuaiLe)
- ➔ les titres de chaque page sont rédigés avec la police Reem Kufi Fun (Reem Kufi Fun)
- ➔ La navbar et le reste du contenu de mes pages sont rédigés avec la police Kanit (Kanit)

Wireframe et arborescence du site

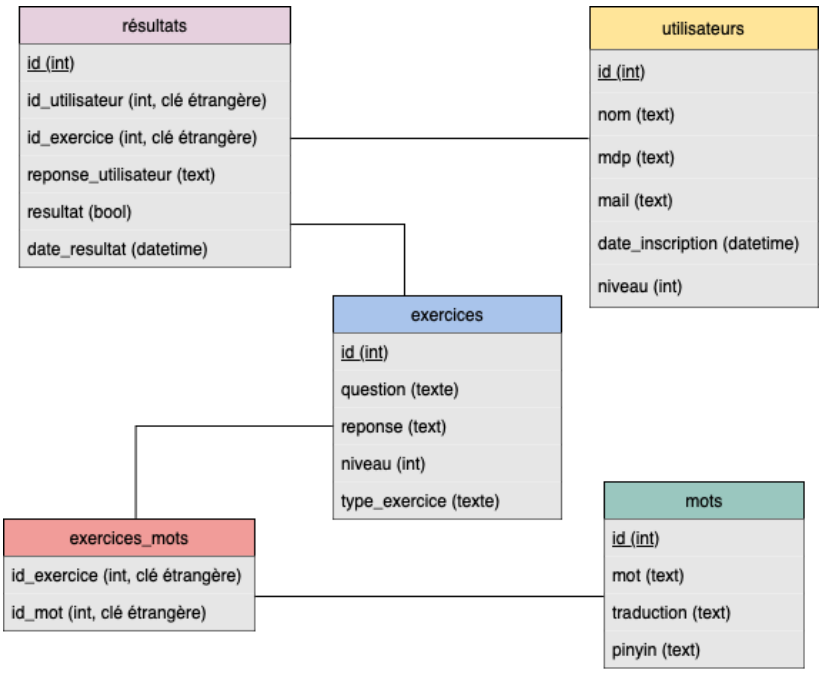
Pour me faire une idée précise de ce qu'il fallait que je réalise pour mon site, j'ai pris l'initiative de réaliser un wireframe avec Figma. Cela m'a donc permis d'y voir plus clair et de me faire une idée de la manière dont il fallait que je construise mon site.



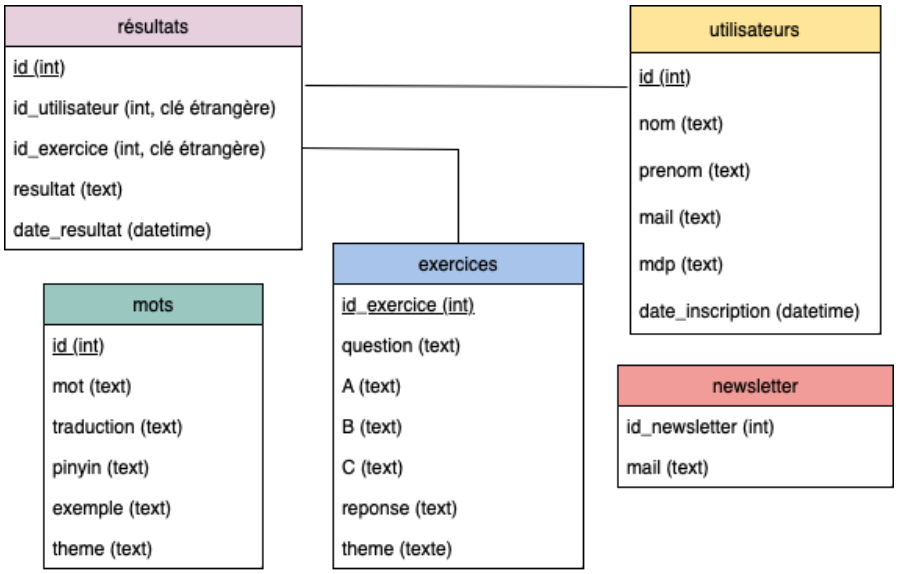
J'ai pu donc faire ressortir un squelette pour mon site. Chaque page est donc construit avec :

- une navbar réalisée avec bootstrap et composée du menu (accueil, mon compte, leçons et exercices et contact) et du logo avec le nom du site
- une div principale dans le body auquelle sera ajouté tous les éléments nécessaires au bon fonctionnement des pages
- un bouton pour passer d'un mode à l'autre (clair / sombre)
- un footer pour s'inscrire à la newsletter qui a également été crée avec bootstrap

Structure de la base de données



Ceci était donc la première version de la structure de ma base de données mais en réalisant le site, je me suis rendue compte que certaines tables et certaines colonnes n'étaient pas nécessaires. Ou inversement, que certaines colonnes étaient à ajouter.



Après remaniement, ma base de données était plus claire et plus simple.

- Dans la **table exercices** j'ai donc fait le choix de retirer le niveau surtout pour une raison de praticité. Mais ce point a été rajouter dans les améliorations à apporter. J'ai ajouté à cette table les colonnes A, B et C qui correspondent en fait aux propositions de réponses et aussi la colonne theme qui correspond au thème du cours en cours (Les nombres, Les pronoms ou Les dates). Cette colonne m'a permis de ne faire afficher que ce qui m'intéressait. En effet, lorsque l'on arrive sur mon site, il faut passer par la page d'accueil des cours et exercices avant d'accéder au contenu de chaque thème. Cette page est composée de boutons pour accéder au contenu d'un

cours en particulier, nous ne voulons pas de pronoms dans la partie nombres par exemple. Avec une requête SQL, j'ai donc dû préciser que je souhaitais afficher seulement les exercices avec le même thème que celui sélectionné dans la page d'accueil des cours et des exercices.
Nous retrouvons également les colonnes question qui contient en fait le mot à deviner et reponse qui a le même résultat qu'une des 3 propositions (A, B ou C). J'utilise cette colonne afin de comparer la réponse de l'utilisateur et la réponse attendue.

- Dans la **table mots**, j'ai seulement ajouté une colonne exemple et une colonne theme pour la même raison que pour la table exercices. Cette table est seulement composée des éléments que j'affiche dans la partie cours.
- Dans la **table utilisateurs**, j'ai ajouté le prénom que j'avais oublié et j'ai retiré le niveau étant donné que je ne le prenais plus en compte. Cette table était surtout composée des informations utiles à la connexion et de ce qui était affichable sur la page compte.
- La **table exercices_mots** n'existe plus car je n'ai pas eu l'occasion de l'utiliser et elle a été remplacée par une **table newsletter** pour les personnes qui souhaitent s'inscrire à la newsletter sans s'inscrire sur le site.
- Pour la **table resultats**, je n'ai pas eu le temps de l'exploiter mais elle aurait été utile pour enregistrer les résultats des utilisateurs pour chaque exercice puis ensuite d'afficher chaque score obtenu dans la page compte de l'utilisateur. La requête pour insérer les données était prête mais je n'ai pas compris comment récupérer la valeur du score depuis le fichier javascript `exercices.js` et de la réutiliser dans mon fichier php `cours_exo.php`.
    - id_utilisateur, aurait donc été l'id de l'utilisateur connecté
    - id_exercice, correspond à l'id de la question mais dans mon cas elle aurait seulement retourné l'id de la dernière question car je voulais m'en servir pour faire afficher le thème de l'exercice dans la page de compte (dans le cas où les variables de sessions n'existaient pas)
    - resultat, correspond au nombre de bonnes réponses / le nombre total de questions
    - date_resultat, la date d'obtention du score

J'ai fait le choix de transformer tous mes fichiers en fichier php et donc de ne pas les laisser en html car dans le fichier `connect_db.php`, en plus de la fonction permettant de se connecter à la base de données, j'ai déclaré 4 autres fonctions qui m'ont facilité la vie :

- `entete()`, contenant le début du code html, c'est-à-dire le head
- `navbar()`, contenant le header de mon body. Je n'ai donc pas eu besoin de changer les liens de chacune de mes pages lorsque j'ai passé mes fichiers html en php
- `switch_theme()`, contenant la div et le bouton pour changer de thème (sombre / clair)
- `footer()`, contenant mon footer et la fin de mon code html

Ces 4 fonctions sont donc appelées dans tous mes fichiers php sauf `deconnexion.php`.

Fonctionnalités

Au niveau des fonctionnalités de mon site, j'avais 3 principaux objectifs :
- faire afficher les exercices à partir de ma base de données

- faire une fonctionnalité de login
- faire un site avec un mode clair et sombre (avec jquery) et/ou afficher du vert quand la réponse à un exercice est bon et rouge dans le cas contraire

➔ Affichage des exercices à partir de ma base de données

L'affichage des exercices à partir de ma base de données était plutôt simple. En effet, après avoir récupéré le thème envoyé par la page `cours_exo_accueil.php`, il restait simplement à faire une requête SQL pour ne retourner que les données ayant le même thème que celui sur lequel nous avions cliqué au préalable dans la page précédente et une boucle while pour afficher toutes les lignes avec une structure identique pour chacune.
C'est plutôt sur la mise en forme des exercices où j'ai rencontré quelques difficultés. En effet, je ne savais pas comment organiser la page. L'organisation que je voulais réaliser à l'origine (celle du wireframe) ne convenait pas car toutes les questions étaient affichées d'un coup sur la page. Cela provoquait donc une incohérence au niveau de la sélection de la réponse car nous ne pouvions sélectionner qu'une seule réponse parmi toutes celles proposées sur la page. Je suis finalement parti sur un affichage à l'aide des propriétés css display:none; et display:flex; pour afficher les questions une à une. Pour passer à la question suivante, il faut appuyer sur le bouton "question suivante". Cette action est possible grâce à mon code js `exercices.js` où la fonction `nextQuestion()` permet d'afficher la question courante et de cacher les autres. Ce bouton se change en bouton "Terminer" lorsque l'indice de la question courante correspond au nombre total de questions, c'est-à-dire à la dernière question de l'exercice. Dans ce cas, lorsque l'on appuie sur le bouton "Terminer", une alerte s'affiche pour annoncer le score obtenu par l'utilisateur à l'issue de l'exercice. Cet affichage est possible grâce à la fonction `terminerExercice()`.
J'ai également restreint le nombre de boutons qu'il est possible de sélectionner pour une question afin que l'utilisateur ne puisse pas changer sa réponse au cours de l'exercice et ainsi éviter de fausser le compteur de bonnes réponses. Cette contrainte a été définie avec la variable answered définit par défaut à false, c'est-à-dire que l'utilisateur n'a pas encore répondu. Cette variable est définie à true dans la fonction `verifReponse()` qui vérifie la réponse sélectionnée par l'utilisateur afin de soit colorer la div en verte si c'est la bonne réponse, soit en rouge si c'est une mauvaise réponse. Cette fonction prend comme argument la réponse sélectionnée par l'utilisateur et la réponse attendue qui est disponible dans l'attribut data-reponse pour chaque proposition de réponse. Cette fonction m'a également permise de tenir un compteur de bonnes réponses qui sera réutilisé dans la fonction `terminerExercice()`.

➔ Connexion

Pour cette partie de gestion du compte d'un utilisateur, je n'étais pas sûre de comment organiser la page. Je voulais que les informations du compte ne soient accessibles que si l'utilisateur est connecté ou inscrit. J'ai donc décidé d'écrire un code (`compte.php`) comprenant 2 parties : il y a une première condition if…else… dans le cas où l'utilisateur est connecté. Si c'est le cas, celui-ci peut accéder à son compte, si ce n'est pas le cas, on se retrouve dans le else, c'est-à-dire que l'utilisateur doit soit se connecter, soit s'inscrire et on se retrouve donc dans un second if…else….

- Si un utilisateur est déjà connecté, il peut donc directement accéder aux informations de son compte. Les informations le concernant et présentes dans la base de données sont récupérées grâce aux variables de session. Celles-ci sont affichées sur la page.
  Celui-ci peut également se déconnecter en appuyant sur le bouton de déconnexion qui renvoie vers le fichier `deconnexion.php`. Ce fichier supprime toutes les variables de session, détruit la session et redirige l'utilisateur vers `compte.php` où l'utilisateur fera face à la page de connexion.
  Le design de ce bouton a été récupéré sur le site uiverse (Button by kennyotsu-monochromia). J'ai choisi de récupérer un bouton déjà tout fait pour donner un peu de dynamique au site. Ce choix est purement tourné design car je ne souhaitais pas mettre seulement une image cliquable et j'avoue m'être fait envoûté par l'effet rendu par ce bouton.

- Si aucun utilisateur n'est connecté, celui-ci se retrouve devant un formulaire de connexion. Il peut aussi s'inscrire en appuyant sur le lien pour s'inscrire. Ce lien ne va pas l'emmener sur une nouvelle page mais juste faire disparaître la div de connexion et faire apparaître la div d'inscription. L'action inverse est également possible grâce au fichier js `log-affichage.js` et la propriété css display. Concernant le design de ces 2 formulaires, je les ai également récupérés sur le site uiverse (formulaire de connexion : Form by JkHuger / formulaire d'inscription : Form by JkHuger). C'était encore un choix purement design et je ne voulais pas faire de formulaire simple étant donné que j'en avais déjà fait un pour la page `contact.php`. De plus, je trouvais que le design plutôt arrondi et gris se prêtait bien à mon site et cette fois-ci j'ai été envoûté par l'effet des titres d'input lorsque l'on appuie sur un champ. Certains éléments du formulaire ne me convenaient pas alors j'ai modifié le css selon mes préférences.

Pour la partie connexion, si l'utilisateur rentre bien son mail et son mot de passe, une alerte bootstrap (https://creersonsiteweb.net/page-bootstrap-alert) apparaît pour dire que c'est en cours de connexion. Les informations le concernant sont sauvegardées dans les variables de session et l'utilisateur est redirigé vers le premier if, c'est-à-dire vers l'accès aux informations de son compte. Si le mot de passe est incorrect ou si l'utilisateur n'existe tout simplement pas dans la base de données, alors une alerte bootstrap apparaît pour prévenir l'utilisateur.

Pour la partie inscription, si tous les champs pour l'inscription sont remplis, alors on peut procéder à l'insertion des données dans la base de données. A ce moment j'ai rencontré un problème pour insérer la colonne de date d'inscription (type datetime). Je pensais que l'insertion de cette colonne se ferait automatiquement comme pour l'id mais aucune donnée ne s'insérait dans ma base de données si je ne remplissais pas cette colonne. Donc après quelques recherches (How to insert date and time Functions in PHP), je me suis rendu compte que pour insérer un élément de type datetime, il fallait le déclarer comme ceci : date("Y-m-d H:i:s") où la méthode date prend en paramètre le format que l'on souhaite :

- Y = l'année
- m = mois

- d = jour du mois
- H = heure
- i = minute
- s = seconde

Une fois les données insérées, une alerte bootstrap apparaît pour signaler que l'inscription est bien faite. Cela ne connecte pas automatiquement l'utilisateur, il faut qu'il se connecte de lui-même.

➔ **Mode clair et sombre**

Cette fonctionnalité est développée avec Javascript / jQuery dans le fichier `switch-mode.js`.

Pour cette fonctionnalité, j'avais eu l'occasion de commencer à la développer pour le projet de techniques web mais 2 problèmes persistaient :

- des fois l'image du chat qui dort (celui pour changer en mode clair) restait sur le mode clair. Il fallait rappuyer sur le bouton pour que le bon chat apparaisse
- un certain enchaînement de mode clair / mode sombre provoquait un problème : le background-color ne se chargeait pas donc le fond était blanc. Il fallait changer de page ou la recharger pour faire apparaître un fond

La solution à ces 2 problèmes a été l'ajout de la propriété localStorage qui permet d'enregistrer des données dans un navigateur et que le système se "souvienne" du choix réalisé. Je me suis surtout servie de 2 méthodes qui sont setItem() pour ajouter une clé et une valeur au localStorage et getItem() pour obtenir un élément du localStorage à l'aide de la clé. La méthode getItem() permet de sauvegarder l'état du site même si l'utilisateur quitte la page. Si on quitte la page en étant en mode clair et qu'on ne supprime pas les caches entre temps, quand on retournera sur le site, celui-ci sera en mode clair même si le mode par défaut est le mode sombre.

Difficultés

A part avec les fonctionnalités que j'avais à mettre en place, j'ai pu rencontrer d'autres difficultés :

- D'une manière générale, je voulais faire en sorte que mon site soit le plus responsive possible donc c'est pour cela que j'ai utilisé bootstrap à chaque fois que j'en avais l'occasion
- L'utilisation de jQuery m'a demandé énormément de temps car il fallait trouver la bonne méthode à utiliser à chaque fois
- J'ai perdu pas mal de temps à cause des caches aussi. En effet il fallait les vider assez régulièrement afin de voir les changements sur le site

Plus spécifiquement maintenant :

- **Page `contact.php`** : C'est un formulaire de contact assez simple réalisé grâce à bootstrap de la même manière que nous avions déjà fait en classe. Je me suis dis

qu'au lieu de stocker les messages dans la base de données, il serait plus judicieux d'envoyer directement un mail à la personne en charge du service client du site (c'est-à-dire moi). J'ai donc découvert que php fournissait la méthode mail (Envoi du contenu d'un formulaire vers un email – Les Docs). Après avoir attendu quelques heures sans recevoir de mail, je me suis rendue compte que cette méthode permettait juste de vérifier que la demande d'envoi de courrier est soumise au serveur (Comment envoyer un mail en PHP avec mail(), SMTP, Phpmailer - Letecode), donc j'aurai pu attendre très longtemps. J'ai quand même décidé de le laisser comme trace d'une nouvelle connaissance.

Pour cette même page, j'ai eu des difficultés à faire afficher le message de confirmation quand l'utilisateur envoyait le formulaire. Lorsque j'envoyais le formulaire, le contenu de la div disparaissait bien pour faire apparaître le message de confirmation mais celui-ci ne restait que quelques secondes avant que la page se recharge. J'ai donc utilisé la méthode preventDefault dans le fichier javascript `confirmation.js` qui permet d'empêcher le rafraichissement de la page.
Après relecture des consignes, je me suis rendue compte qu'il fallait en fait utiliser l'attribut action de la balise form pour renvoyer vers un fichier php qui contiendrait la réponse. Ce code est donc stocké dans le fichier `confirmation.php` et il n'y avait donc plus de problème concernant la page qui se rafraîchit.

- **Page `cours_exo_accueil.php`** : Cette page sert de page d'accueil aux cours et exercices. L'utilisateur peut donc sélectionner le thème qu'il souhaite étudier. J'ai eu des difficultés par rapport au fait qu'il fallait que je fasse en sorte de renvoyer le thème choisi vers la page `cours_exo.php`. En cherchant un peu, je voyais des tutoriels pour utiliser windows.location.href qui m'aurait permis de récupérer le thème directement dans l'URL grâce à la méthode GET. Cette technique a effectivement fonctionné mais j'ai trouvé la manière de faire un peu compliquée par rapport à ce que nous avions appris. Finalement, j'ai simplement assigné une valeur à chaque bouton et j'ai entouré le bouton de la balise form avec comme attribut action="cours_exo.php". Une fois que l'utilisateur a appuyé sur le thème de son choix, je vérifie si le thème est bien présent dans la requête GET et si oui, on récupère la valeur et l'utilisateur est redirigé vers `cours_exo.php` dans lequel on pourra récupérer le thème de la même manière qu'avec windows.location.href et ainsi n'afficher que le cours et les exercices propres au thème choisi.

- **Page `cours_exo.php`** : Pour cette page, je voulais afficher les cours et exercices sur la même page. Pour cela, je me suis dis que cela pourrait être sympa d'avoir des onglets et de pouvoir naviguer entre le cours d'un thème précis et les exercices de ce thème. Il fallait donc que je trouve le moyen de faire apparaître la partie cours quand j'appuie sur le bouton cours et ainsi faire disparaître la partie exercices et inversement. Comme pour la partie affichage des exercices expliquée précédemment dans la partie fonctionnalité, je me suis tournée vers la propriété css display:none; (d-none) qui permet d'afficher ou non un élément (code `affichage.js`). Donc par défaut, lorsque nous arrivons sur la page `cours_exo.php` c'est la partie cours qui est affichée et la partie exercices est cachée.

Je me suis également demandée s'il était possible d'utiliser les méthodes hide() et show() mais d-none me posait problème. En effet, si j'enlève d-none de la div contenant les exercices, alors cette partie s'affiche à la suite de la partie cours mais si je laissais d-none alors quand j'appuyais sur l'onglet exercices, rien ne s'affichait. J'ai donc préféré rester sur la méthode avec le code js.

Points d'amélioration

Bien que je sois satisfaite de ce que j'ai pu réaliser pour mon site, il y a encore et il y aura toujours de nombreux points à améliorer :

D'une manière générale, je n'ai découvert les variables de sessions que trop tard étant donné que je n'ai fait les sessions php qu'à la fin de mon projet. Mais j'ai beaucoup apprécié la praticité pour faire passer des informations d'une page à l'autre et j'aurais aimé m'en servir plus.

Partie exercices :

- diversifier les exercices et ajouter plus de questions
- ajouter du contenu audio pour remplir des textes à trous par exemple ou tout simplement pour prononcer les caractères dans la partie cours
- faire une barre de progression en affichant le numéro des questions / le nombre total de questions
- mettre en place des niveaux afin de donner un objectif à l'utilisateur

Partie formulaire de contact :

- trouver un moyen de réellement m'envoyer un mail quand l'utilisateur envoie un formulaire de contact car pour l'instant je n'ai rien reçu

Partie compte :

- mettre en place un "remember me" pour ne pas avoir à se reconnecter à chaque fois
- mettre en place un "mot de passe oublié" parce que ça arrive
- pouvoir supprimer son compte et modifier ses informations
- sécuriser l'enregistrement des mots de passe dans la base de données. Pour ce point j'avais commencé à me renseigner mais je me suis finalement rendu compte que ce n'était pas une priorité étant donné que ce n'est pas un sujet que l'on a abordé en cours (MYSQL BIEN STOCKER MOT DE PASSE PASSWORD UTILISATEUR MD5 SALT | Créer son site web | Apprendre HTML CSS PHP Javascript JQuery MySQL Bootstrap Twitter)

Conclusion et critiques

Pour conclure, j'ai vraiment apprécié ce cours et j'ai appris énormément de choses étant donné qu'à l'origine je n'avais que des connaissances en html/css. Je me suis toujours demandé comment on utilisait d'autres langages de front-end et de back-end et, bien qu'il existe beaucoup d'autres langages, je suis satisfaite des connaissances que j'ai pu acquérir et cela me permet d'avoir une base solide pour continuer à apprendre.

Justine Tu

J'ai trouvé que ce cours était un bon complément au cours de techniques web que nous avions le lundi. Même si certains sujets se répétaient, cela nous permettait de mieux assimiler l'information voire de comprendre si nous n'avions pas compris au préalable.
Pour ma part, je pense que l'idée de ce projet est vraiment bien mais avec les deadlines de tous les autres projets et le début du stage, ce n'était pas forcément évident de trouver du temps pour faire quelque chose de bien. Peut-être faudrait-il aborder le sujet du projet plus tôt dans l'année afin de pouvoir y réfléchir tout au long du semestre.
En tout cas un grand merci pour ce semestre de front, c'était très intéressant et je suis vraiment contente d'avoir approfondi mes connaissances dans ce domaine.