DOCUMENTATION TECHNIQUE

ESPORIFY

SOMMAIRE

Réflexions initiales technologiques....2

HTML/CSS SASS avec Bootstrap....2

Javascript....2

PHP....2

PHP Mailer....2

Base de données....3

SEO et moteurs de recherche....3

Requêtes base de données....3

Environnement et sécurité du site....4

GIT....4

Configuration de l'environnement de travail....5

Mise en place de Trello....5

GitHub....5

Création de ma base de données....6

Création de mon environnement de travail en local....6

MCD, diagramme d'utilisation et diagramme de séquence....7

Réflexions initiales technologiques

HTML/CSS SASS avec Bootstrap

J'ai utilisé HTML pour réaliser la structure de base de mon site, étant le seul langage que je connaisse à ce jour pour réaliser le squelette d'un site web en front. J'ai ajouté à ceci le CSS pour styliser et habiller mes pages, puis Sass pour une gestion plus efficace des variables, des fonctions, des mixins et autres éléments à réaliser, Sass facilitera la maintenance des styles. J'ai en parallèle de cela décidé d'utiliser Bootstrap pour ses composants réactifs, ce qui m'a grandement facilité la mise en place de la partie responsive de mon site sur les différents breakpoints, grâce à Sass j'ai pu modifier facilement les composants pour qu'ils correspondent plus à mes besoins.

Javascript

JavaScript est utilisé pour les interactions dynamiques et ainsi améliorer l'expérience utilisateur sur la plateforme, étant donné que JavaScript permet de modifier le contenu d'une page dynamiquement sans effectuer de rechargement de celle-ci. Il est utilisé entre autres pour les filtres et tout ce qui a besoin d'être dynamique sur le site.

PHP

PHP est mon langage de back-end, car pour le moment je n'ai pas encore étudié d'autres langage pouvant réaliser les mêmes tâches que PHP. Et pour un projet comme celui-ci je me suis concentré sur l'apprentissage, la compréhension et la mise en pratique de ce langage.

PHP traitera toute la logique serveur, interagira avec les bases de données et fera le traitement des données des formulaires.

PHP Mailer

Je me suis tourné vers PHP Mailer pour la gestion des envois de mails de manière plus sécurisé et plus configurable que la fonction mail de PHP.

Base de données

Pour ce qui est de la base de données je me suis tourné vers MariaDB, ayant commencé la gestion de mon projet en local via Wampserver dans lequel il est intégré. Pour la base de données relationnelle j'ai utilisé MySQL, puis me suis servis de PhpMyAdmin comme interface visuelle pour afficher les différentes tables de la base de données de mon projet.

Pour la base de données non relationnelle NoSQL je suis passé par MongoDb, j'ai fait usage de Compass là aussi pour une interface visuelle de mes données.

SEO et moteurs de recherche

Afin d'améliorer le SEO du site ainsi que pour les moteurs de recherche je décide d'effectuer un routage des pages en PHP, via ce que l'on nomme un MVC (Model Vue Controller), ainsi les pages seront directement visibles par les moteurs de recherche, alors qu'en JavaScript les pages ne seront visibles qu'après le chargement par l'utilisateur étant donné qu'elles ne seront pas chargées sur le serveur mais directement sur le navigateur.

Le CSR (Client-Side Rendering) est certes plus lent au chargement mais la navigation est plus fluide et il y a une meilleure persistance au niveau des données mais pour un site de commerce, ce n'est pas forcément ce que nous recherchons. Nous recherchons à être vus par les moteurs de recherche et un bon référencement. Un bon référencement est donc dans notre cas plus important qu'une meilleure expérience utilisateur et c'est pour cela que j'ai choisi le SSR (Server-Side Rendering). Les parties qui ne sont pas importantes pour les moteurs de recherche pourront être effectuée en JS.

Requêtes base de données

Les requêtes vers la base de données récupèreront toutes les informations nécessaires en une fois afin de minimiser la connexion à la base de données et tout récupérer en une seule fois afin que les utilisateurs sur mobile avec une petite connexion ne doivent pas patienter à chaque chargement dû à une requête à la base de données.

Environnement et sécurité du site

Toutes les données sensibles seront placées dans un fichier .env qui sera placé à la racine du site déployé et qui sera sécurisé par un fichier .htaccess. Les droits d'accès à ce fichier seront modifiés et il sera exclu du public (owner uniquement en read et en write).

J'ai mis en place un token CSRF lors d'envoi de formulaire pour augmenter la sécurité. Ce qui permet de s'assurer que l'utilisateur voulant envoyer les données se trouve bien en Session sur la plateforme. Dans le cas où le CSRF est manquant la requête ne sera pas envoyé vers le serveur et stoppera immédiatement toute exécution de l'envoi des données.

J'ai mis en place un système de gestion des erreurs qui permet d'attraper toutes les exceptions qui pourraient survenir que ce soit en développement ou en production.

En ce qui concerne l'envoi de fichier image, elles seront remaniées, tout d'abord pour vérifier leur intégrité, puis en les renommant ce qui permettra une gestion plus facile des images en BDD, l'ajout d'un identifiant unique dans le nom pour éviter que les images en double s'écrasent, et un redimensionnement autant pour l'affichage sur le navigateur que pour l'espace de stockage.

À chaque début de session des paramètres de cookies sont mis en place.

GIT

J'utilise GitHub et Trello pour la gestion de mon projet. GitHub me permettant de gérer les différentes versions de mon projet lors des phases de développement, et Trello me permettant de suivre l'avancement des différentes tâches et fonctionnalités à réaliser dans le projet.

Cela me permet aussi de voir les différents outils de travail qui facilite la collaboration au sein d'un groupe pour des projet de plus grande envergure. Surtout cela offre un historique qui donne une vision globale de l'avancement mais surtout apporte une aide non négligeable lorsqu'il s'agit de débugger le système, on peut ainsi remonter dans les différentes versions pour récupérer un état avant le bug et comprendre la source du problème.

Configuration de l'environnement de travail

Mise en place de Trello

J'ai créé mon espace de travail sur Trello en prenant soin de le rendre public.

Le lien vers le Trello du projet : https://trello.com/b/XrhgQl3s/ecf-esportify

GitHub

Après la création de mon compte sur la plateforme GitHub, j'ai créé le repository de mon projet directement via l'interface web avec l'ajout du fichier README.md et du .gitignore :

Clic sur "New", qui redirige vers la page de création d'un nouveau repository.

Je passe par le terminal GitBash que j'ai téléchargé en amont pour lancer les différentes commandes en me positionnant dans le dossier où je veux copier mon repository.

J'entre la commande suivante :

git clone https://github.com/Alkgorth/ECF-Esportify.git

Ce qui clonera mon projet sur mon poste de travail, ainsi je peux directement travailler et envoyer mes changements sur mon repository.

J'effectue mon premier commit pour finaliser d'initialiser mon repo distant avec mon projet en local.

git status

git commit –m "Initial commit"

git push –u origin main

Par la suite je créé différentes branches qui me permettront de mieux gérer mon travail : main sera toujours la copie de secours qui sera celle publiée à l'état final et testée, dev sera la branche de travail et de développement, l'ajout d'autres branches pour isoler la création de fonctionnalités qui seront merge une fois les tests réalisés.

Création de ma base de données

Je crée une base de données sur mon espace. Je réalise le fichier de création sur mon IDE que j'envoie ensuite à ma base de données via PhpMyAdmin pour qu'il importe les données et crée les différentes tables. L'utilisation de PhpMyAdmin me permet de me signaler si une erreur est présente dans la réalisation de mes tables ce que je peux plus facilement corriger.

Création de mon environnement de travail en local

J'ai tout d'abord utilisé Xamp mais ai finis par basculer sur Wamp, plus adapté à mon environnement de travail. Cela m'a permis de disposer d'un serveur Apache pour tester mon site en local.

J'ai positionné mon projet dans le fichier www de Wamp64 afin de pouvoir le lancer directement via mon localhost, j'ai nommé le fichier esportify. Il me suffira d'entrer dans mon URL http://esportify/ et j'arriverai sur mon application.

Je peux dès à présent commencer à travailler sur la création de mon index.php, la porte d'entrée de mon application.

Mon second push vers mon repository contiendra les fichiers sur lesquels j'ai déjà avancé, comme l'arborescence du site, les différents diagrammes et quelques images qui permettent d'habiller un minimum la page d'accueil.

Je crée la une sous branche Dev sur laquelle publier mon avancement de façon régulière sans venir écraser les données de la branche main.

Je télécharge en local les données de Bootstrap qui me sont nécessaires pour la création de mon application et j'ajoute Sass avec la commande dans le terminal de mon IDE "npm install bootstrap@5.3.3".

Je mets en place PHP Mailer grâce à la commande "composer require phpmailer/phpmailer".

Je mets en place Docker Desktop pour la gestion de mes conteneurs, je prépare les fichiers Dockerfile et docker-compose.yml pour la création de mon environnement de travail local

sous conteneurs. Et j’installe également MongoDb Compass pour la gestion de ma base de données non relationnel, j’intègre MongoDb à mes conteneurs en local par la suite.

MCD, diagramme d’utilisation et diagramme de séquence

Ces documents sont disponibles dans le répertoire Assets/Documentation de mon projet.

Détail du déploiement