

Projets réalisés dans le cadre du passage du Titre Professionnel de
DÉVELOPPEUR WEB et WEB MOBILE

Présenté par *Jessy CHARLET*
La_Plateforme - Toulon 2024

Projet 2/2

GAME THRONE

Boutique en ligne de chaises gaming



SOMMAIRE

Introduction	... 3
Compétences du référentiel couvertes par le projet	... 4
Activité type n°2	... 4
Développer la partie back-end d'une application web ou web mobile sécurisée	... 4
- Créer une base de données	... 4
- Développer les composants d'accès aux données	... 4
- Développer la partie back-end d'une application web ou web mobile	... 4
Cahier des charges	... 5
Objectifs	... 5
Conception de l'application	... 5
- Composants du MVP :	... 5
Utilisateurs et User Stories	... 6
- Rôles	... 6
- User Stories	... 6
Spécifications techniques	... 8
Versioning	... 8
Choix technologiques	... 10
- Front-End	... 10
- Back-End	... 10
La base de données	... 11
Rendu	... 13

Le code	... 17
Altorouter	... 17
CSS	... 19
JavaScript / Jquery	... 20
PHP	... 23

MISE EN GARDE

Ce second projet vise à présenter l'activité type N° 2, il ne reviendra donc pas ou alors succinctement sur l'activité type N°1.

INTRODUCTION



Notre projet consiste à créer une boutique en ligne spécialisée dans la vente de chaises gaming haut de gamme. Cette boutique offrira une variété de modèles de chaises, conçues pour répondre aux besoins des gamers en termes de confort, d'ergonomie et de style. Chaque chaise est soigneusement sélectionnée pour offrir la meilleure expérience de jeu possible, avec des fonctionnalités et des designs adaptés à tous les types de joueurs.

Il a été réalisé en coopération avec [Alina Yefimova](#) et [Ismael Lebrun](#).

COMPÉTENCES DU RÉFÉRENTIEL COUVERTES PAR LE PROJET

Game Throne est un projet développé en PHP et en Javascript avec la bibliothèque Jquery.
Il a été réalisé pour un exercice lors de la formation.

Activité type n°2

Développer la partie back-end d'une application web ou web mobile sécurisée

- **Créer une base de données**

En utilisant les user stories et le MVP, nous avons ensuite réalisé le MCD (Modèle Conceptuel des Données) et généré le MLD (Modèle Logique des Données). Nous avons utilisé la méthode MERISE en distinguant clairement la modélisation des données (entités, relations) de celle des traitements (processus, événements).

MCD : Entités et leurs relations.

MCT : Processus et événements.

MLD : Tables dérivées du MCD.

MPD : Implémentation des tables dans un SGBD (Système de Gestion de Base de Données) en l'occurrence MySQL.

- **Développer les composants d'accès aux données**

Avec du PHP en POO (Programmation Orientée Objet), nous avons créé une classe Database qui permet de gérer tout ce qui utilise la base de données. Elle contient des méthodes telles que connect() ou getProductById() qui utilisent PDO (PHP Data Objects) afin de créer une connexion à la base de données et d'y préparer les requêtes SQL paramétrées. Ce qui est une bonne pratique en termes de sécurité pour se prémunir des injections SQL.

- **Développer la partie back-end d'une application web ou web mobile**

J'ai utilisé PHP pour la partie back-end et JavaScript avec Jquery pour les animations, mais également pour l'affichage des produits lors de la recherche et du tri par filtres.

Cahier des charges

Objectifs

Offrir une plateforme de vente spécialisée dans les chaises gaming :

- Proposer une large gamme de chaises gaming de haute qualité.
- Répondre aux besoins spécifiques des gamers en termes de confort, d'ergonomie et de style.

Améliorer l'expérience utilisateur :

- Faciliter la recherche et la comparaison de produits.
- Offrir des options de personnalisation pour répondre aux préférences individuelles des clients.

Augmenter la satisfaction client :

- Fournir des descriptions détaillées et des avis clients pour chaque produit.
- Offrir un suivi de commande transparent et en temps réel.

Conceptualisation de l'application

Le MVP pour notre boutique en ligne inclura les éléments essentiels permettant aux utilisateurs d'acheter des chaises gaming en ligne, avec les fonctionnalités de base pour une expérience utilisateur satisfaisante.

Composants du MVP :

-
- 1. Page d'accueil :**
 - Présentation des produits phares et des promotions.
 - 2. Catalogue de produits :**
 - Liste des chaises gaming disponibles avec des filtres de recherche par prix, couleur, et caractéristiques.
 - 3. Page produit :**
 - Description détaillée de chaque chaise (matériaux, dimensions, fonctionnalités).
 - Photos de haute qualité.
 - Avis et évaluations des clients.
 - 4. Panier et commande :**
 - Ajout de produits au panier.
 - Processus de commande simplifié.
 - Options de paiement sécurisées.
 - 5. Support client :**
 - Formulaire de contact.

UTILISATEURS ET USER STORIES

Rôles

Le User pourra consulter toute la boutique, même sans se connecter, il pourra également passer une commande en tant que visiteur.

Il pourra néanmoins créer un compte afin de pouvoir consulter son historique de commande ou poster des commentaires.

L'admin aura accès au back office, il pourra ajouter, modifier, supprimer des produits, des catégories et des commandes.

User Stories

En tant que	je souhaite	afin de
visiteur	parcourir le site	accéder à la page d'accueil
visiteur	ouvrir la page produit	accéder à tous nos produit et les trier par filtre
visiteur	utiliser la barre de recherche	accéder à un produit rapidement
visiteur	ouvrir le menu de connexion	accéder au formulaire de connexion
visiteur	ouvrir le menu de connexion	accéder au formulaire d'inscription
visiteur	parcourir les mentions légales	accéder au CGV, RGPD
visiteur	accéder au formulaire de contact	nous contacter
utilisateur connecté	parcourir mon profil	consulter mon historique de commande
utilisateur connecté	parcourir mon profil	modifier mes informations personnelles ou supprimer mon compte
utilisateur connecté	accéder au formulaire de commentaires	poster un commentaire sur un produit
administrateur	me connecter	accéder au back-office
administrateur	accéder au back-office	modifier un produit
administrateur	accéder au back-office	voir les détails d'un produit
administrateur	accéder au back-office	supprimer un produit
administrateur	accéder au back-office	ajouter une catégorie
administrateur	accéder au back-office	modifier une catégorie
administrateur	accéder au back-office	modifier un utilisateur
administrateur	accéder au back-office	supprimer un utilisateur

administrateur	accéder au back-office	attribuer un rôle à un utilisateur
----------------	------------------------	------------------------------------

SPECIFICATIONS TECHNIQUES

Versioning

La gestion de version permet de conserver l'historique du code source et de travailler plus facilement en équipe. Il était très important de garder les versions précédentes du code, notamment en cas de gros problème avec une fonctionnalité suite à des modifications dans le code. Pour cette gestion, nous avons utilisé Git.

J'ai au départ utilisé GitHub puis avec le temps je suis passé directement en ligne de commande. En début de projet ce fut un peu l'anarchie, mais après une grosse réunion, nous avons mis en place l'utilisation de Pull Request afin de contrôler les merges de branches. Nous avons également arrêté d'utiliser le trello au profit des Git Issues, plus pratiques. Pour chaque Git Issue nous avons créé une branche puis déposé une Pull Request avant de valider collégialement le merge.

La liste des git issues résolues :

<input type="checkbox"/>	<input checked="" type="radio"/> Formulaire CSS	<small>enhancement</small>	
	#36 by Jessy-Charlet was closed on May 14		
<input type="checkbox"/>	<input checked="" type="radio"/> Amélioration du panier	<small>enhancement</small>	
	#34 by Jessy-Charlet was closed on May 14		
<input type="checkbox"/>	<input checked="" type="radio"/> Résolution des bugs et amélioration de la page produit	<small>bug</small>	
	#32 by Jessy-Charlet was closed on May 14		
<input type="checkbox"/>	<input checked="" type="radio"/> Réparation le bouton "Ajouter au panier"	<small>bug</small>	
	#29 by alina-yefimova was closed on May 9		
<input type="checkbox"/>	<input checked="" type="radio"/> Réparation le bouton "Ajouter au panier"		
	#28 by alina-yefimova was closed on May 8		
<input type="checkbox"/>	<input checked="" type="radio"/> BUG si produit n'a pas de lien d'image.	<small>bug</small>	
	#27 by IsmaelLebrun was closed on May 7		
<input type="checkbox"/>	<input checked="" type="radio"/> Réparer la page produit après le changement de BDD	<small>bug</small>	
	#26 by Jessy-Charlet was closed on May 7		
<input type="checkbox"/>	<input checked="" type="radio"/> Création d'une méthode pour récupérer tous les produits	<small>enhancement</small>	
	#24 by Jessy-Charlet was closed on May 7		
<input type="checkbox"/>	<input checked="" type="radio"/> Création d'une méthode pour récupérer les images triées des produits	<small>enhancement</small>	
	#23 by Jessy-Charlet was closed on May 7		
<input type="checkbox"/>	<input checked="" type="radio"/> Menu mobile	<small>enhancement</small>	
	#21 by Jessy-Charlet was closed on May 7		
<input type="checkbox"/>	<input checked="" type="radio"/> Mettre à jour les requêtes SQL	<small>bug</small>	
	#20 by Jessy-Charlet was closed on May 6		
<input type="checkbox"/>	<input checked="" type="radio"/> Harmoniser l'affichage des produits	<small>bug</small>	
	#19 by Jessy-Charlet was closed on May 13		
<input type="checkbox"/>	<input checked="" type="radio"/> Ajouter un fil d'Ariane sur les produits	<small>enhancement</small>	
	#18 by Jessy-Charlet was closed on May 14		
<input type="checkbox"/>	<input checked="" type="radio"/> Amélioration de la navigation du site	<small>enhancement</small>	
	#17 by Jessy-Charlet was closed on Apr 29		
<input type="checkbox"/>	<input checked="" type="radio"/> Bug erreur SQL adresse show inside input	<small>bug</small>	
	#13 by Jessy-Charlet was closed on May 14		
<input type="checkbox"/>	<input checked="" type="radio"/> recommandation géographique à la création de compte	<small>enhancement</small>	
	#12 by Jessy-Charlet was closed on Apr 30		
<input type="checkbox"/>	<input checked="" type="radio"/> Panier fonctionnel	<small>enhancement</small>	
	#11 by Jessy-Charlet was closed on May 13		
<input type="checkbox"/>	<input checked="" type="radio"/> Ajouter une SQL constraint pour la liaison one to many	<small>bug</small>	
	#10 by Jessy-Charlet was closed on May 3		
<input type="checkbox"/>	<input checked="" type="radio"/> Valider le paiement dans la page panier Stripe	<small>enhancement</small>	
	#9 by Jessy-Charlet was closed on May 9 ➔ Version 1.0 le clie...		
<input type="checkbox"/>	<input checked="" type="radio"/> Pré remplissage des champs inscriptions	<small>bug</small>	
	#8 by Jessy-Charlet was closed on May 17		
<input type="checkbox"/>	<input checked="" type="radio"/> Filtres combinés pour la page boutique	<small>enhancement</small>	
	#7 by Jessy-Charlet was closed on Apr 26		
<input type="checkbox"/>	<input checked="" type="radio"/> NEW FEATURE : Autoremplissage de l'input city quand on remplit le code postal et inversement		
	#6 by IsmaelLebrun was closed on Apr 30		

CHOIX TECHNOLOGIQUES

- Front-End

Pour la structure nous avons utilisé le HTML5 (HyperText Markup Language) ainsi que le CSS3 (Cascading Style Sheets). Ayant réalisé une grande partie du design, je n'ai pas utilisé de bibliothèque ou framework CSS tels que Bootstrap ou Tailwind, mais plutôt du CSS vanilla. En revanche pour le Javascript j'ai utilisé la bibliothèque Jquery car je préfère sa syntaxe.

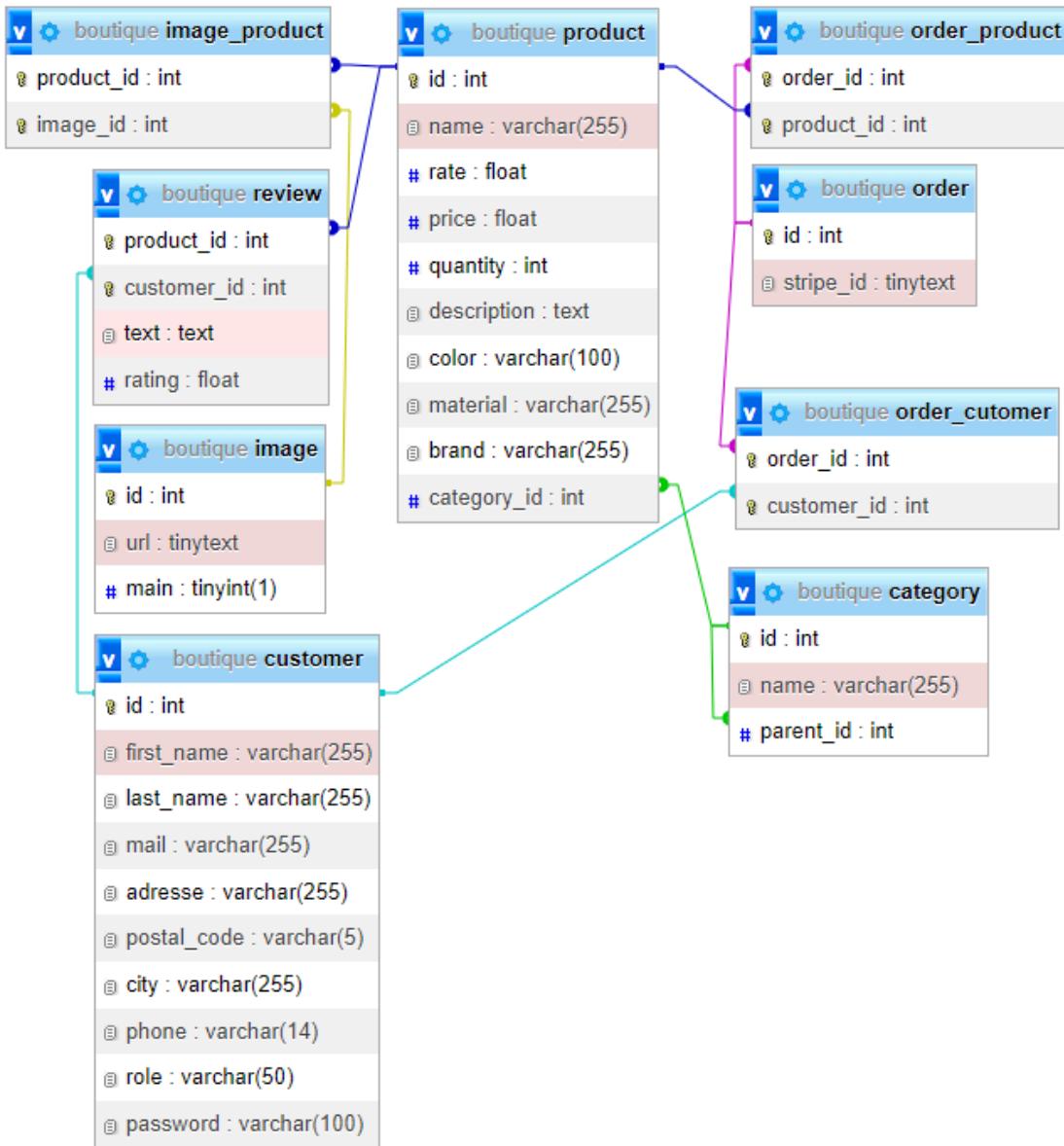
- Back-End

Nous avons utilisé une structure MVC (Modèle-Vue-Contrôleur) pour le projet Game Throne afin de séparer les préoccupations et améliorer la maintenabilité du code. Cette approche permet de découpler la logique métier (Modèle) de la présentation (Vue) et du flux de contrôle (Contrôleur), facilitant ainsi les tests, les mises à jour et l'évolutivité. En utilisant MVC, nous assurons également une meilleure organisation du projet.

Nous avons codé en PHP sans framework (car nous n'en avions pas encore vu), couplé à AltoRouter pour gérer les routes de manière efficace et flexible. AltoRouter facilite la définition et la gestion des URL, permettant de diriger les requêtes utilisateur vers les contrôleurs appropriés sans complexité excessive. Cette solution légère et performante améliore la maintenabilité et l'évolutivité de l'application en offrant une syntaxe claire et simple pour le routage. En utilisant AltoRouter, nous assurons une navigation fluide et une gestion cohérente des chemins d'accès au sein de notre boutique en ligne.

BASE DE DONNÉES

Il a tout d'abord fallu analyser les besoins et créer un MCD (Modèle Conceptuel des Données) afin de définir les entités et les relations de manière claire et compréhensible. Puis de le transformer en MLD (Modèle Logique des Données) plus proche de la structure de la base de données relationnelle avec des clés primaires et étrangères



On a ici, réalisé sous phpMyAdmin les relations entre les tables qui sont matérialisées par les traits colorés avec suivant le type de relation des accroches différentes. Par exemple un customer peut être l'auteur de plusieurs review de produit, mais une review ne peut avoir qu'un seul customer comme auteur. Il s'agit donc d'une relation oneToMany. En revanche, un produit

peut avoir plusieurs catégories et une catégorie peut avoir plusieurs produits. Dans ce cas, il s'agit d'une relation manyToMany. On peut également voir une liaison particulière au niveau des catégories. En effet, celle-ci possède une relation récursive, c'est-à-dire qu'elle entretient une relation avec elle-même. Une catégorie peut être liée à une autre catégorie ce qui en ferait une sous-catégorie. C'est assez rare mais utile dans ce cas de figure car on pourra utiliser l'instruction SQL "WITH RECURSIVE" afin de récupérer toutes les sous-catégories d'une catégorie par exemple. On peut également noter la présence de tables de jointures. Elles sont reconnaissables par leur nom qui comporte celui des deux tables jointes. Par exemple, produit_order qui fait la jointure entre product et order. Sa clé primaire est la combinaison des clés étrangères liées aux tables qu'elle joint. Ces tables permettent d'éviter la redondance des données, de modéliser facilement les relations manyToMany et elles maintiennent l'intégrité référentielle entre les tables associées.

RENDU

Nouveau Throne

Disponible maintenant

Nouveau Throne

Disponible maintenant

Filter par prix
1 € - 1000 €

Filter par couleurs

Black	White	Red	Yellow	Green	Cyan	Purple	Grey
-------	-------	-----	--------	-------	------	--------	------

Chaise de jeu racing style avec support lombaire 179.99 € 0.0 ★

Chaise de gaming XL pour les joueurs sérieux 299.99 € 0.0 ★

Filter par prix
1 € - 1000 €

Filter par couleurs

Black	White	Red	Yellow	Green	Cyan	Purple	Grey
-------	-------	-----	--------	-------	------	--------	------

Chaise de jeu racing style avec support lombaire 179.99 € 0.0 ★

Chaise de gaming XL pour les joueurs sérieux 299.99 € 0.0 ★

Chaise de jeu convertible en lit 349.99 € 0.0 ★

Chaise de jeu camouflage 209.99 € 0.0 ★

Chaise de jeu design futuriste 399.99 € 0.0 ★

Chaise de gaming rétro avec design vintage 189.99 € 0.0 ★

Chercher un Thron...

Mon panier

Chaise de jeu racing style avec support lombaire	- 1 +
Bleu	179.99 €
Total	579.97 €
Chaise de gaming rétro avec design vintage	- 1 +
Rouge	189.99 €
Chaise de gaming camouflage	- 1 +
Vert	209.99 €

Passer la commande

SHOP INFO

Mon panier

Chaise de jeu racing style avec support lombaire	179.99 €
	179.99 €
Chaise de gaming rétro avec design vintage	189.99 €
	189.99 €
Chaise de gaming camouflage	209.99 €
	209.99 €
Total	579.97 €
Passer la commande	

RGPD Nous contacter

f i Instagram

← Alina YEFIMOVA TEST Détails ▾

Payer Alina YEFIMOVA **579,97 €**

Afficher les détails ▾

Informations de livraison

E-mail

Adresse de livraison

Nom complet

France

Adresse

Saisir l'adresse manuellement

Données de paiement

Informations de la carte

1234 1234 1234 1234 VISA

MM / AA CVC

Les informations de facturation sont identiques aux informations de livraison

Enregistrer mes informations en toute sécurité pour le paiement en un clic

Saisissez votre numéro de téléphone pour créer un compte Link et régler vos achats plus rapidement auprès de Alina YEFIMOVA et partout où Link est accepté.

Payer Alina YEFIMOVA **579,97 €**

Payer par carte

Informations de livraison

E-mail

Adresse de livraison

Nom complet

France

Adresse

Saisir l'adresse manuellement

Données de paiement

Informations de la carte

1234 1234 1234 1234 VISA

MM / AA CVC

Les informations de facturation sont identiques aux informations de livraison

Enregistrer mes informations en toute sécurité pour le paiement en un clic

Saisissez votre numéro de téléphone pour créer un compte Link et régler vos achats plus rapidement auprès de Alina YEFIMOVA et partout où Link est accepté.

Enregistrer mes informations en toute sécurité pour le paiement en un clic

Saisissez votre numéro de téléphone pour créer un compte Link et régler vos achats plus rapidement auprès de Alina YEFIMOVA et partout où Link est accepté.

Propulsé par stripe | Conditions d'utilisation - Confidentialité

Chercher un Throne...

Erreur 404

Oups ...

Nous venons de faire une petite boulette lors de la construction de ce Throne...

[Retourner à l'accueil](#)



[SHOP](#) [INFO](#)



Erreur 404

Oups ...

Nous venons de faire une petite boulette lors de la construction de ce Throne...

[Retourner à l'accueil](#)

    Jessy

Back Office

GAMES THRONES

Ajouter un produit				
4	Chaise de jeu racing style avec support lombaire	179.99 €	20 □	Modifier X
5	Chaise de gaming XL pour les joueurs sérieux	299.99 €	20 □	Modifier X
7	Chaise de jeu convertible en lit	349.99 €	20 □	Modifier X
8	Chaise de gaming rétro avec design vintage	189.99 €	20 □	Modifier X
9	Chaise de gaming camouflage	209.99 €	20 □	Modifier X
10	Chaise de jeu design futuriste	399.99 €	20 □	Modifier X
11	Chaise de gaming avec système de massage intégré	299.99 €	20 □	Modifier X

Back Office

GAMES THRONES

Ajouter un produit

4	Chaise de jeu racing style avec support lombaire	179.99 €	20	Modifiez	X
5	Chaise de gaming XL pour les joueurs sérieux	299.99 €	20	Modifiez	X
NOM		PRIX EN €	STOCK		
Chaise de jeu convertible en lit		349.99	20		
COULEUR	MATIÈRE	MARQUE	CATÉGORIE		
Blanc	Tissu	DreamGamer	Chaise gaming		
RATE					
0					
DESCRIPTION	<p>Confort similaire au canapé, le dossier de cette chaise de jeu est composé de ressorts ensachés et de mousse moulée qui assurent la même elasticité et le même confort qu'un canapé. Les ressorts permettent mieux la pression sur le coussin d'assise et rendent la conduite plus confortable et ergonomique. Tissu hautement respirant : le matériau de surface de la chaise de jeu est composé d'un tissu en maille hautement respirant avec une bonne dissipation de la chaleur. Même en été chaud, vous ne vous sentirez pas étouffé lorsque vous êtes assis sur la chaise. En outre, ce tissu est plus élastique et le corps est soutenu lorsque vous êtes assis, ce qui permet un travail plus efficace. [Garantie de haute qualité] Nous avons amélioré la fixation du dossier de cette table de jeu et de cette chaise. Le dossier est fixé avec des plaques en acier et résiste aux chocs de plus de 300 livres. Les vis de fixation ont une forme triangulaire, ce qui améliore considérablement la stabilité du dossier.</p>				
ID IMAGE PRINCIPALE	CHEMIN IMAGE PRINCIPALE	LIEN IMAGE SECONDAIRE	LIEN IMAGE SECONDAIRE		
11	product_6_main_image.jpg	20	product_6_image_1.jpg		
Ajouter une image		Annuler	valider		
8	Chaise de gaming rétro avec design vintage	189.99 €	20	Modifiez	X
9	Chaise de camion camouflage	209.99 €	20	Modifiez	X

localhost:8080

Back Office

GAMES THRONES

Recherche

Produits

Catégories

Commandes

Clients

Commentaires

Notes

1	test@test.test	Voir plus	X
2	admin@admin.admin	Voir plus	X
PRÉNOM	NOM	EMAIL	ADRESSE
Jessy	Charlet	1@mail.fr	452 rue blabla
CODE POSTAL	VILLE	TELEPHONE	Annuler
83100	Toulon	11 22 33 44 55	
4	a@a.a	Voir plus	X
5	b@b.b	Voir plus	X
6	m@m.m	Voir plus	X
7	h@h.h	Voir plus	X

localhost:8080

LE CODE

ALTOROUTER

```
1  <?php
2  require '../vendor/autoload.php';
3
4  $uri = $_SERVER['REQUEST_URI'];
5  $router = new AltoRouter();
6
7  // Cr ation des routes
8  // route de page affich e
9  // $router->map('METHOD', '/url/to/page', '/path/to/file_name', 'name_route');
10 $router->map('GET', '/', 'home', 'accueil');
11 $router->map('GET', '/connexion', 'signIn', 'connexion');
12 $router->map('GET', '/inscription', 'signUp', 'inscription');
13 $router->map('GET', '/profil', 'profil', 'profil');
14 $router->map('GET', '/cgv', 'cgv', 'cgv');
15 $router->map('GET', '/rgpd', 'rgpd', 'rgpd');
16 $router->map('GET', '/mention', 'mention', 'mention');
17 $router->map('GET', '/contact', 'contact', 'contact');
18 $router->map('GET', '/produit', 'product', 'produit');
19 $router->map('GET', '/404', '404', '404');
20 $router->map('GET', '/panier', 'basket', 'panier');
21 $router->map('GET', '/filtre', 'filters', 'filtre');
22 $router->map('GET', '/checkout', 'checkout_backend', 'checkout_backend');
23 $router->map('GET', '/success', 'success', 'success');
24 $router->map('GET', '/cancel', 'cancel', 'cancel');
25
26 // route de page de traitement
27 $router->map('POST', '/signUpControllerphp', '../public/controller/php/signUpController', 'signUpControllerphp');
28 $router->map('POST', '/signInControllerphp', '../public/controller/php/signInController', 'signInControllerphp');
29 $router->map('GET', '/deconnexion', '../public/controller/php/deconnexion', 'deconnexion');
30 $router->map('POST', '/profilControllerphp', '../public/controller/php/profilController', 'profilControllerphp');
31 $router->map('POST', '/backOfficeControllerphp', '../public/controller/php/backOffice/backOfficeController', 'backOfficeControllerphp');
32
33 // Routes to AJAX files
34 $router->map('GET', '/api/panier', 'basket_json', 'Ajax/panier');
35 $router->map('GET', '/addProductToBasketAjaxController', '../public/controller/php/ajax/addProductToBasketAjaxController', 'addProductToBasketAjaxController');
36 $router->map('GET', '/getProductDataByIdAjaxController', '../public/controller/php/ajax/getProductDataByIdAjaxController', 'getProductDataByIdAjaxController');
37 $router->map('GET', '/getCartContentsAjaxController', '../public/controller/php/ajax/getCartContentsAjaxController', 'getCartContentsAjaxController');
38
39 // Routes Back Office
40 $router->map('GET', '/gt-admin', '../public/backOffice/backOffice', 'backOffice');
41 $router->map('GET', '/bot_backend', 'bot_backend', 'bot_backend');
```

AltoRouter fonctionne en mappant les URL aux fonctions ou contrôleur spécifiques. Nous définissons des routes en spécifiant les chemins d'URL et les méthodes HTTP associées, puis nous associons ces routes à des fonctions de traitement. Lorsqu'une requête est reçue, AltoRouter compare l'URL de la requête aux routes définies pour trouver une correspondance. Si une route correspond, le contrôleur ou la fonction associée est appelé. Cela simplifie la gestion des chemins d'accès et améliore l'organisation du code.

La page index contient la liste de nos routes.

```

43     function my_autoloader($class)
44     {
45         include 'controller/php/classes/' . $class . '.class.php';
46     }
47
48 // Enregistrement de la fonction d'autoload
49 spl_autoload_register('my_autoloader');
50
51 $match = $router->match();
52
53 if (is_array($match)) {
54     // Handle routes that send JSON
55     if (str_contains($match["name"], "Ajax")) {
56         if (is_callable($match['target'])) {
57             call_user_func_array($match['target'], $match['params']);
58         } else {
59             $params = $match['params'];
60             require "../src/{$match['target']}.php";
61         }
62     } else if (str_contains($match["name"], "_backend")) {
63         if (is_callable($match['target'])) {
64             call_user_func_array($match['target'], $match['params']);
65         } else {
66             $params = $match['params'];
67             require "../src/{$match['target']}.php";
68         }
69     } else {
70         // Handle routes that send HTML
71         require '../templates/header.php';
72         if (is_callable($match['target'])) {
73             call_user_func_array($match['target'], $match['params']);
74         } else {
75             $params = $match['params'];
76             require "../src/{$match['target']}.php";
77         }
78         require '../templates/footer.php';
79         echo "</html>";
80     }
81 } else {
82     // 404 error
83     header("location:" . $router->generate('404') . "");
84 }

```

Puis en fonction du contenu de l'URL, on envoie soit le contenu sur une page html qui inclut le header et le footer. Soit on retourne du Json (qu'on récupère par exemple en JS avec un fetch() pour l'affichage des articles par filtres)

CSS

```
201  /***-MOBILE-*-
202  @media (orientation: portrait) {
203    .productsGrid {
204      grid-template-columns: repeat(2, 1fr);
205    }
206
207    .filters {
208      grid-template-columns: 1fr;
209      grid-template-rows: auto 1fr;
210      padding: 2vh 0 2vh 0;
211
212      .filtersConditions {
213        display: grid;
214        grid-template-columns: repeat(2, 1fr);
215        gap: 5vw;
216
217        .filtersContainer {
218          margin: 0;
219
220          .filtersColors {
221            grid-template-columns: repeat(8, 1fr);
222            grid-template-rows: 1fr;
223          }
224
225          .filtersContainer:nth-child(2) {
226            order: 3;
227            grid-column: 1/3;
228          }
229        }
230      }
231
232      .oups {
233        margin-top: 10vh;
234      }
235    }
236  }
237
238
239 }
```

J'ai utilisé les media queries pour la version mobile (principalement orientation: portrait) qui me permet de ne prévoir que 2 possibilités d'affichages:
une version portrait (pour les mobiles et tablettes)
et une version paysage (pour les ordinateurs, tablettes / mobiles en mode paysage).
J'utilise principalement des unitées vh/vw, les % afin d'adapter au mieux mon design aux résolutions d'écrans.
et pour les mises en forme j'utilise beaucoup Flex box et les grids. ce qui permet une grande souplesse et un plus grand contrôle sur l'affichage.

JavaScript / Jquery

Le fichier JS que j'ai codé pour faire le slider de la page d'accueil. Il affiche / cache simplement les conteneurs #slide1/2/3 en fonction du bouton cliqué.

La "difficulté" ici à surtout été le défilement automatique géré dans la fonction SlideAuto()

```
1  $(document).ready(function () {
2      $("#slide2").hide();
3      $("#slide3").hide();
4
5      function sleep(ms) {
6          return new Promise(resolve => setTimeout(resolve, ms));
7      }
8
9      function Slide(slide) {
10         $(".slide").hide();
11         $("#slide" + slide).fadeIn("slow");
12         $(".slideButton button").removeClass();
13         $("#bannerButton" + slide).addClass("buttonActif");
14     }
15
16     async function SlideAuto(nbrSlide) {
17         let i = 1;
18         $("#bannerButton1").on("click", function () {
19             i = 1;
20         })
21         $("#bannerButton2").on("click", function () {
22             i = 2;
23         })
24         $("#bannerButton3").on("click", function () {
25             i = 3;
26         })
27         for (i = i; i <= nbrSlide; i++) {
28             Slide(i);
29             if (i >= nbrSlide) {
30                 i = 0;
31             }
32             await sleep(5000);
33         }
34     }
35
36     $("#bannerButton1").on("click", function () {
37         Slide("1");
38     })
39     $("#bannerButton2").on("click", function () {
40         Slide("2");
41     })
42     $("#bannerButton3").on("click", function () {
43         Slide("3");
44     })
45
46     SlideAuto(3);
47 })
```

J'ai également codé l'affichage des produits par filtre, ce qui à été le plus compliqué, c'est de rendre possible l'addition de ses filtres. Par exemple, je veux une chaise rouge, mais qui coûte entre 200€ et 300€.

Ci dessous la fonction qui permet l'affichage des produits en remplaçant le contenu html de la balise #filtersProducts.

```
1  $(document).ready(function () {
2    $("#filtersSup").hide();
3    sessionStorage.clear('color');
4    function afficherProduits(products) {
5      const liste = document.createElement('div');
6      liste.classList.add("productsGrid");
7      if ((0 < products.length)) {
8        for (var i = 0; i < products.length; i++) {
9          // Container
10         const listItem = document.createElement('div');
11         listItem.classList.add("product");
12         // Nom du produit
13         const titleElement = document.createElement('div');
14         titleElement.classList.add("productName");
15         titleElement.textContent = products[i]["name"];
16         // Lien du produit
17         const urlElement = document.createElement('a');
18         urlElement.href = `/produit?id=${products[i]["id"]}`;
19         // Image du produit
20         const imageElement = document.createElement('img');
21         const imageUrl = "./assets/img/products/" + products[i]["img"];
22         imageElement.src = imageUrl;
23         imageElement.alt = products[i]["name"];
24         // Prix du produit
25         const divElement = document.createElement('div');
26         divElement.classList.add("productDiv");
27         const priceElement = document.createElement('div');
28         priceElement.classList.add("productPrice");
29         priceElement.textContent = products[i]["price"] + " €";
30         // Note du produit
31         const ratingElement = document.createElement('div');
32         ratingElement.classList.add("productRate");
33         ratingElement.innerHTML = `parseFloat(products[i]["rate"]).toFixed(1) + "<img src='./assets/img/star.png'>"`;
34         // Ajouter les éléments à la liste
35         listItem.appendChild(urlElement);
36         urlElement.appendChild(imageElement);
37         urlElement.appendChild(titleElement);
38         urlElement.appendChild(divElement);
39         divElement.appendChild(priceElement);
40         divElement.appendChild(ratingElement);
41         liste.appendChild(listItem);
42     };
43     $("#filtersProducts").html(liste);
44   } else {
45     $("#filtersProducts").html("<div class='oops'>Oups... Aucun produit ne correspond à cette recherche :(</div>\"");
46   }
47 }
```

la fonction asynchrone recherche() va fetch() le controller recherche.php qui va effectuer une requête SQL avec les paramètres passés en argument puis va récupérer la réponse sous format json et ainsi appeler la fonction afficherProduits() que l'on a vu juste au dessus

avec le résultat en argument.

Nous avons ensuite plusieurs fonctions anonymes.

Une qui se déclenche lors du choix de couleur et qui va la passer dans le local.storage afin de pouvoir l'utiliser plus tard, et lancer une la fonction recherche() avec le resultat en argument.

Une autre qui se déclenche lors de la modification du slider de prix et qui va gérer celui ci ainsi que ses valeurs puis lancer une recherche() avec ,si une couleur est stockée en local storage, la couleur + la fourchette de prix, dans le cas contraire la recherche() avec uniquement la fourchette de prix.

```
48 ↴  async function recherche(mini = 1, maxi = 1000, color = "all") {
49    const reponse = await fetch("../controller/php/recherche.php?color=" + color + "&mini=" + mini + "&maxi=" + maxi);
50    const products = await reponse.json();
51    afficherProduits(products);
52  }
53 ↵  $("#noir, #blanc, #rouge, #jaune, #vert, #bleu, #violet, #gris").on("click", function () {
54    sessionStorage.setItem('color', this.id);
55    recherche($("#slider-range").slider("values", 0), $("#slider-range").slider("values", 1), sessionStorage.getItem('color'));
56    $("#filtersSup").html("X Supprimer le filtre <span class='bold'>" + this.id + "</span>");
57    $("#filtersSup").slideDown();
58  });
59 ↵  $(function () {
60    $("#slider-range").slider({
61      range: true,
62      min: 1,
63      max: 1000,
64      values: [1, 1000],
65      slide: function (event, ui) {
66        $("#amount").val(ui.values[0] + " € - " + ui.values[1] + " €");
67        if (sessionStorage.getItem('color')) {
68          recherche($("#slider-range").slider("values", 0), $("#slider-range").slider("values", 1), sessionStorage.getItem('color'));
69        } else {
70          recherche($("#slider-range").slider("values", 0), $("#slider-range").slider("values", 1));
71        }
72      }
73    });
74    $("#amount").val($("#slider-range").slider("values", 0) +
75    " € - " + $("#slider-range").slider("values", 1) + " €");
76  });
77 ↵  $(function () {
78    if (sessionStorage.getItem('color')) {
79      $("#filtersSup").slideDown();
80    }
81    $("#filtersSup").on("click", function () {
82      sessionStorage.clear('color');
83      recherche($("#slider-range").slider("values", 0), $("#slider-range").slider("values", 1));
84      $(this).slideUp();
85    });
86  }
87  })
88  recherche();
89 })
```

PHP

Nous utilisons la Programmation Orientée Objet (POO) et PDO dans le projet Game Throne pour améliorer la modularité, la réutilisabilité et la maintenabilité du code. La POO permet de structurer le code en objets représentant des entités du domaine, facilitant ainsi la gestion et l'évolution du projet. PDO, en tant qu'interface d'accès aux bases de données, sécurise les interactions avec la base de données grâce aux requêtes paramétrées et offre une portabilité entre différents systèmes de gestion de bases de données. Ensemble, POO et PDO contribuent à une application plus robuste, sécurisée et évolutive.

```
2 class Database
3 {
4     private static $servername = 'localhost';
5     private static $username = 'root';
6     private static $password = '';
7     private static $BDD = 'boutique';
8     private static $conn = null;
9
10    public function __construct()
11    {
12        die('Init function is not allowed');
13    }
14    public static function connect()
15    { //fonction de connexion à la BDD
16        if (null == self::$conn) { //si la connexion est nulle
17            try { //on essaie de se connecter
18                self::$conn = new PDO("mysql:host=" . self::$servername . ";" . "dbname=" . self::$BDD, self::$username, self::$password);
19                self::$conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
20            } catch (PDOException $e) {
21                die($e->getMessage());
22            }
23        }
24        return self::$conn;
25    }
26    public static function disconnect()
27    {
28        self::$conn = null;
29    }
}
```

La classe Database gère tout ce qui touche à la base de données, elle contient par exemple la fonction connect, qui permet de se connecter à la base de données avec PDO afin de faire des requêtes par la suite.

Exemple avec la fonction ci-dessous : getProductById() qui prend en paramètre l'id du produit que l'on souhaite récupérer.

On utilise ici le try / catch afin de pouvoir récupérer l'erreur s'il y a un problème avec la connexion à la base de données.

Puis on prépare la requête SQL avec des attributs que l'on va hydrater grâce à binParam() et enfin exécuter la requête.

On récupère les données qu'on place dans un tableau à l'intérieur de la variable \$product, on ferme la connexion et on retourne \$product.

```

112     // Récupération des produits par ID
113     public static function getProductById($id)
114     {
115         try {
116             // Connection à La BDD
117             $conn = Database::connect();
118
119             // Préparation de La requête SQL pour récupérer L'ensemble produits
120             $stmt = $conn->prepare("SELECT * FROM product WHERE id=:id");
121             $stmt->bindParam(':id', $id);
122             $stmt->execute();
123
124             // Execution de La requête SQL
125             $product = $stmt->fetch(PDO::FETCH_ASSOC);
126
127             // Fermeture de La connection
128             $conn = null;
129
130             // return["id"] pour L'ID du produit
131             // return["name"] pour le nom du produit
132             // return["rate"] pour la note du produit
133             // return["price"] pour le prix du produit
134             // return["quantity"] pour la quantité du produit
135             // return["description"] pour la description complète du produit
136             // return["color"] pour la couleur du produit
137             // return["material"] pour la matière du produit
138             // return["brand"] pour la marque du produit
139             // return["category_id"] pour la catégorie du produit
140             return $product;
141         } catch (PDOException $e) {
142             return $e;
143         }
144     }

```

voici le contrôleur dont on parlait un peu plus haut, celui qui est appelé par le fetch du JS pour afficher les produits selon les différents filtres.

On peut voir au require qu'il à besoin de la class database pour fonctionner

On déclare \$selec qui est un tableau et qui sera la valeur renournée à la résolution du code, puis si il n'y a pas color dans l'URL on effectu une requête SQL via PDO pour récupérer tous les produits. Par contre si il y a color dans l'URL:

Si sa valeur est all, alors la requête ne prend en compte que la fourchette de prix.

Dans tous les autres cas, la requête prend en compte sa valeur en plus de la fourchette de prix.

Une fois cette requête préparée, on l'exécute et on place tout dans la \$product. Pour finir on appelle la fonction getImg() qui est tout en haut du code afin d'effectuer une requête SQL pour chaque produit dans le but d'y récupérer toutes ses images dans la table image qui est liée par une table de jointure image_product à la table product comme nous l'avons vu plus haut.

On termine par push \$product dans le tableau \$selec, on ferme la connexion et on echo le \$selec en format Json. Ce qui sera récupéré par le JavaScript et affiché dans le DOM.

```

3  require('../classes/Database.class.php');
4  $conn = Database::connect();
5  $selec = [];
6  function getImg($products)
7  {
8      global $selec;
9      global $conn;
10     foreach ($products as $product) {
11         try {
12             $sql = $conn->prepare("SELECT url FROM image
13             INNER JOIN image_product
14             WHERE image.id = image_product.image_id AND image_product.product_id = " . $product['id'] . " and image.main = 1");
15             $sql->execute();
16             $img = $sql->fetch(PDO::FETCH_ASSOC);
17             $product["img"] = $img["url"];
18             array_push($selec, $product);
19         } catch (PDOException $e) {
20             http_response_code(500);
21             echo json_encode(["msg" => "Erreur de connexion à la base de données: " . $e->getMessage()]);
22             exit();
23         }
24     }
25 }
26 if (isset($_GET["color"])) {
27     try {
28         switch ($_GET["color"]) {
29             case "all":
30                 $stmt = $conn->prepare("SELECT * FROM product WHERE price >= :mini AND price <= :maxi");
31                 break;
32             default:
33                 $stmt = $conn->prepare("SELECT * FROM product WHERE price >= :mini AND price <= :maxi AND color= :color");
34                 $stmt->bindParam(':color', $_GET['color']);
35                 break;
36         }
37         $stmt->bindParam(':mini', $_GET['mini']);
38         $stmt->bindParam(':maxi', $_GET['maxi']);
39         $stmt->execute();
40         $products = $stmt->fetchAll(PDO::FETCH_ASSOC);
41         getImg($products);
42     } catch (PDOException $e) {
43         http_response_code(500);
44         echo json_encode(["msg" => "Erreur de connexion à la base de données: " . $e->getMessage()]);
45         exit();
46     }
47 } else {
48     try {
49         $stmt = $conn->prepare("SELECT * FROM product");
50         $stmt->execute();
51         $products = $stmt->fetchAll(PDO::FETCH_ASSOC);
52         getImg($products);
53     } catch (PDOException $e) {
54         http_response_code(500);
55         echo json_encode(["msg" => "Erreur de connexion à la base de données: " . $e->getMessage()]);
56         exit();
57     }
58 }
59 $conn = null;
60 header('Content-Type: application/json');
61 echo json_encode($selec);

```