

# **Dossier de projet**

## Création d'un site e-commerce: 3D Factory

*Titre: RNCP Niv. 5 - DWWM - Développeur Web et Web Mobile*

Alon Ben David



## Sommaire

<b>1. Compétences du référentiel couvertes par le projet</b>	3
<b>2. Résumé du projet</b>	3
<b>3. Cahier des charges - Spécifications fonctionnelles du projet</b>	4
• Contexte du besoin	4
• Cible	4
• Arborescence du site	4
• Description des fonctionnalités	5
- Accueil du site	5
- connexion classique / espace client	5
- Recherche de produits et fiche produit	5
- panier client	6
- Paiement et achat	6
- Panel admin	6
<b>4. Spécifications techniques du projet</b>	7
• Choix techniques et architecture du projet	7
4.1. Technologies front-end :	8
4.2. Technologies back-end :	8
4.3. Sécurité :	8
4.4. Web mobile :	8
4.5. Maquettage et conception :	9
4.6. Programmation orientée objet (POO) :	9
4.7. Architecture du projet:	9
4.8. Gestion des utilisateurs :	9
4.9. Fonctionnalités de la boutique en ligne :	10
4.10. Fonctionnalités administratives :	10
4.11. Simulation du processus de paiement :	10
<b>5. Réalisations:</b>	11
Maquette:	11
Base de données:	11
Code:	12
5.1 Inscription et authentification:	13
5.2 Affichage des produits:	15
5.3 Création d'une session de paiement avec Stripe:	17
5.4 Ajout d'un article au panier:	19
5.5 Autocomplétion:	20
<b>6. Vulnérabilités en termes de sécurité</b>	23
<b>7. Recherches à partir de sites anglophones</b>	24
<b>8. Extrait d'un site anglophone</b>	25



## **1. Compétences du référentiel couvertes par le projet**

Le projet couvre les compétences énoncées ci-dessous:

Pour l'activité 1, “Développer la partie front-end d'une application web et web mobile en intégrant les recommandations de sécurité”:

- Maquetter une application
- Réaliser une interface utilisateur web ou mobile statique et adaptable
- Développer une interface utilisateur web dynamique
- Réaliser une interface utilisateur avec une solution de gestion de contenu ou e-commerce

Pour l'activité 2, “Développer la partie back-end d'une application web ou web mobile en intégrant les recommandations de sécurité.”:

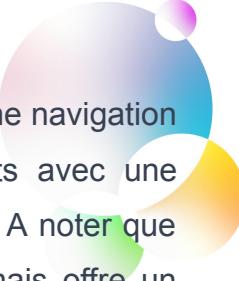
- Créer une base de données
- Développer les composants d'accès aux données
- Développer la partie back-end d'une application web ou web mobile
- Élaborer et mettre en œuvre des composants dans une application de gestion de contenu ou e-commerce.

## **2. Résumé du projet**

Le site e-commerce 3D Factory est un projet de boutique en ligne dédié à la vente d'impressions 3D. J'ai choisi de créer ce site afin d'allier ma passion pour l'impression 3D, le design et l'ingénierie avec mes compétences en développement web.

L'impression 3D offre des possibilités créatives infinies en transformant des modèles en objets physiques. En combinant cette technologie avec le développement web, j'ai conçu une expérience unique pour les utilisateurs, leur permettant de découvrir et d'acheter des produits imprimés en 3D directement sur le site.

Initialement, ce projet a été réalisé en groupe, puis, j'ai poursuivi cette création de site seul. Cela m'a permis de modifier le contenu du site et de regrouper deux sujets pour lesquels je porte un intérêt particulier: le développement web et le design / impression 3D.



Ce projet a impliqué la création d'un site web complet, convivial et sécurisé, offrant une navigation fluide. J'ai travaillé sur la conception du site en mettant en valeur les produits avec une présentation attrayante et en offrant aux utilisateurs des options de personnalisation. A noter que le contenu présenté dans la boutique en ligne est à des fins de démonstration, mais offre un aperçu des possibilités créatives offertes par l'impression 3D. À l'avenir, je pourrai personnaliser le catalogue de produits avec mes propres modèles imprimés en 3D.

Ce dossier de projet me permettra de détailler les différentes étapes de la création de ce site, les défis auxquels j'ai été confronté et les solutions que j'ai mises en place pour créer cette boutique.

### **3. Cahier des charges - Spécifications fonctionnelles du projet**

- **Contexte du besoin**

3D Factory a été imaginé et conçu pour un de mes futurs projets professionnels. L'objectif étant de comprendre comment créer un site e-commerce sans avoir besoin d'utiliser WordPress Shopify par exemple. Ainsi, je pourrai personnaliser ce site de la manière souhaitée et être libre dans ma conception. Pour cela, j'ai créé une boutique en ligne pour la vente de modèles imprimés en 3D, accessibles dans le monde entier, tout en rendant l'expérience utilisateur agréable. Le design graphique du site a été étudié pour cela également.

Ce site a été réalisé en anglais de manière responsive. A termes, plusieurs langues pourront être disponibles afin de cibler une clientèle plus large.

- **Cible**

Ce projet s'adresse premièrement à une clientèle de particuliers, hommes et femmes confondus. Selon les produits proposés, il se peut que ce site s'ouvre à une clientèle de professionnels (par exemple pour la vente de surfboard, foil board ou autre).

- **Arborescence du site**

Le site 3D Factory est présenté de la manière suivante:



- Page d'accueil
- Page de connexion / inscription
- Page catégories de produits et sous catégories de produits
- Page de détail de produit
- Page panier
- Page paiement
- Page de confirmation d'achat
- Page profil
- Page panel admin

- **Description des fonctionnalités**

- **Accueil du site**

La page d'accueil a été créée pour une expérience client intuitive. Elle comporte toutes les informations nécessaires pour une bonne compréhension du site. Ainsi depuis cette page, il est possible d'accéder à la barre de recherche de produits (avec les prix), de trier les produits en fonction de catégories et sous-catégories. Cette barre de recherche avec autocompletion a été conçue via Javascript et PHP. Une fois que le produit recherché est trouvé grâce à cet outil, l'utilisateur est redirigé vers la page de celui-ci. Tous les produits sont présentés sur la page d'accueil. Il y a également un accès à l'espace de connexion client et au panier client.

- **connexion classique / espace client**

Il est nécessaire de s'inscrire sur le site avant d'effectuer un achat, par le biais d'un formulaire d'inscription. Chaque client aura un identifiant et un mot de passe pour faciliter leur prochaine connexion et avoir la possibilité d'acheter de manière plus rapide. Toutes les informations renseignées par l'utilisateur dans son espace pourront être modifiées.

- **Recherche de produits et fiche produit**

Comme énoncé précédemment, la page d'accueil permet de recenser l'ensemble des produits disponibles, avec leur prix. Ainsi, il y a une photo pour illustrer chaque produit, avec la possibilité



de l'ajouter au panier. Lorsque l'utilisateur clique sur un produit, une nouvelle page s'ouvre avec la fiche produit. Dans celle-ci on y retrouve:

- le nom du produit,
- le code du produit,
- la photo
- la description
- le prix
- le bouton pour ajouter le produit au panier

- **panier client**

Le panier client recense les produits que l'utilisateur a choisis afin de passer commande sur le site. Les produits peuvent être ajoutés depuis différentes pages (accueil, fiche produit). Grâce au bouton "To order", il a la possibilité de commander les produits du panier et de procéder au paiement. Dans ce panier, le client pourra voir la/les photos du/des article(s) sélectionné(s), le prix de chaque produit, la somme totale des produits, le montant total à payer. Le client peut aussi supprimer les produits qu'il souhaite avant la validation du panier.

- **Paiement et achat**

Dès lors que le panier est validé par l'utilisateur, ce dernier est redirigé vers la page de paiement opérée par Stripe pour permettre la réalisation de la transaction. Si le paiement est accepté, le site renvoie l'utilisateur vers une page qui lui confirme son paiement en lui envoyant un mail, avec le récapitulatif de sa commande. Cette solution est sécurisée et permet un paiement par carte bancaire.

- **Panel admin**

En qualité de Développeur web, j'ai un accès sécurisé au panel admin afin de gérer le site et d'y apporter les modifications nécessaires si besoin. Ainsi, je peux ajouter ou supprimer des catégories, sous catégories, ou encore des produits. J'ai également la possibilité de modifier toutes les caractéristiques liées au produit (nom du produit, prix, description, photo). Aussi, je peux avoir accès à tous les éléments de commandes et aux informations des acheteurs (date de commande, identité du client, adresse, articles commandés).



## 4. Spécifications techniques du projet

- Choix techniques et architecture du projet

- Maquettage de projet / application : Création d'une maquette graphique du site, en prenant en compte les différentes interfaces pour les versions mobiles et desktop, afin d'assurer un design responsive.
- Conception de base de données MCD / MLD : Réflexion et structuration de la base de données en utilisant un modèle relationnel de données (MCD).
- Programmation Orientée Objet : Utilisation des classes pour organiser et gérer les différentes fonctionnalités et objets du site.
- Structurer un projet et penser son architecture .
- Faire de l'asynchrone avec JS : Utilisation de JavaScript pour gérer les interactions asynchrones avec le back-end, notamment pour la barre de recherche avec autocomplétion.
- Pitcher un projet : Présentation orale du projet en utilisant des slides pour mettre en valeur les fonctionnalités, le design et l'expérience utilisateur.

**Les fonctionnalités demandées pour le site de vente d'impressions 3D sont les suivantes :**

- Page d'accueil attrayante qui met en avant les produits phares et les derniers produits mis en ligne.
- Design attractif respectant la charte graphique de l'entreprise.
- Site responsive adapté aux différents appareils.
- Barre de recherche de produits avec autocomplétion en utilisant JavaScript de manière asynchrone.
- Accès à la boutique présentant tous les produits, avec la possibilité de filtrer par catégorie/sous-catégorie sans rechargement de page.
- Page de détails pour chaque produit, généré dynamiquement, affichant le nom, l'image, le prix, la description et un bouton "add to cart".
- Système de création de comptes d'utilisateurs, incluant une page d'inscription/connexion et un espace de gestion du profil utilisateur.
- Tableau de bord administrateur permettant la gestion des produits, des stocks, des catégories et sous-catégories.
- Système de validation du panier avec une simulation du processus de paiement.



En plus de ces fonctionnalités de base, plusieurs fonctionnalités "bonus" peuvent être envisagées pour enrichir le projet, telles que la possibilité pour les utilisateurs de laisser des commentaires/avis sur les produits avec une modération au niveau administrateur, l'implémentation d'une solution de paiement réelle, la gestion des stocks, la génération de numéros de commande/factures, la gestion de la TVA, des promotions et des tags pour les produits.

Par ailleurs, les spécifications techniques du projet élaborées comprennent des aspects tels que la sécurité et l'optimisation pour le web mobile:

#### **4.1. Technologies front-end :**

- Utilisation des langages JavaScript, CSS et HTML pour le développement de la partie front-end de l'application, assurant une interface utilisateur interactive et conviviale.

#### **4.2. Technologies back-end :**

- Utilisation des technologies PHP, SQL et Composer pour la mise en place de la partie back-end, incluant la gestion des requêtes et de la base de données, ainsi que l'intégration des packages Stripe via Composer pour la gestion des paiements.

#### **4.3. Sécurité :**

- Les mots de passe des utilisateurs sont stockés de manière sécurisée en utilisant des techniques de hachage et de salage.
- Les formulaires d'inscription et de connexion sont protégés contre les attaques de type injection SQL et cross-site scripting (XSS).
- L'accès aux fonctionnalités administratives est restreint aux utilisateurs autorisés.

#### **4.4. Web mobile :**

- Le site est conçu en utilisant des techniques de développement responsive pour garantir une expérience utilisateur optimale sur les appareils mobiles.
- Les pages sont optimisées pour un chargement rapide, en compressant les ressources et en utilisant des images adaptées pour les écrans mobiles.
- Les fonctionnalités de la boutique en ligne seront adaptées aux écrans de petite taille pour une navigation conviviale sur les appareils mobiles.



#### **4.5. Maquettage et conception :**

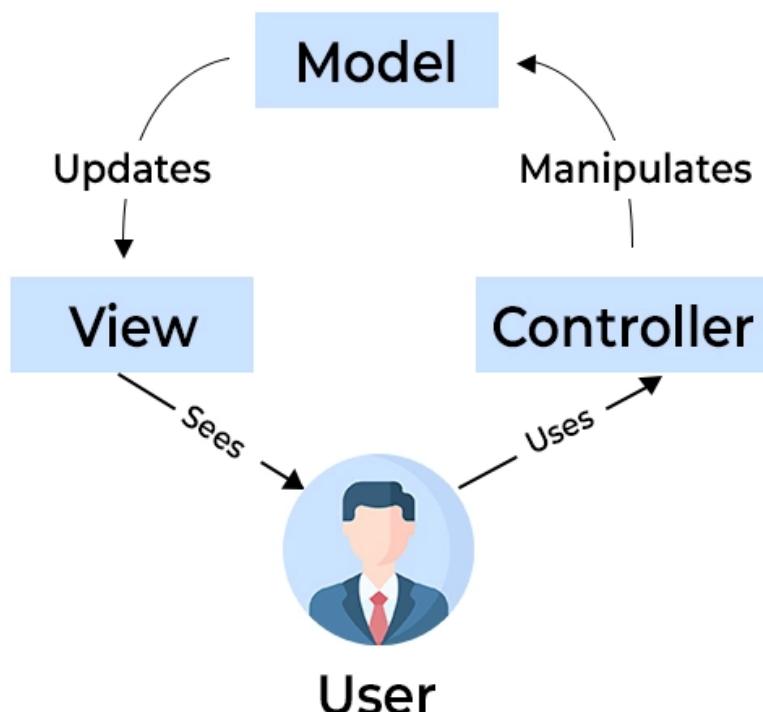
- Avant de commencer le développement, une maquette graphique a été réalisée, en mettant l'accent sur les versions mobiles et desktop pour garantir un design responsive.
- Un modèle conceptuel de données (MCD) a été créé pour structurer la base de données en tenant compte des différents éléments du système de vente en ligne.

#### **4.6. Programmation orientée objet (POO) :**

- Le développement du projet a été réalisé en utilisant des concepts de POO, en créant des classes pour organiser les fonctionnalités et les objets du site.
- Les principes de l'encapsulation, de l'héritage et du polymorphisme.

#### **4.7. Architecture du projet:**

- Utilisation du modèle MVC (Modèle-Vue-Contrôleur) pour la structuration et l'organisation du projet, garantissant une séparation claire des responsabilités entre le modèle de données, la logique métier et l'interface utilisateur.



#### **4.8. Gestion des utilisateurs :**

- Le site propose un système d'inscription et de connexion sécurisé pour les utilisateurs.



- Chaque utilisateur aura un profil personnel où il pourra consulter et modifier ses informations, ainsi que consulter son historique d'achats et son panier.

#### **4.9. Fonctionnalités de la boutique en ligne :**

- La page d'accueil présente des sections attractives mettant en avant les produits phares et les derniers produits mis en ligne.
- Un système de recherche avec autocomplétion a été mis en place, permettant aux utilisateurs de trouver rapidement les produits souhaités.
- Les produits sont classés par catégories et sous-catégories, avec la possibilité de les filtrer sans rechargement de page.
- Chaque produit dispose d'une page détaillée avec des informations complètes, une image, un prix et un bouton d'ajout au panier.

#### **4.10. Fonctionnalités administratives :**

- Un espace d'administration est disponible pour la gestion des produits, des stocks, des catégories et sous-catégories.
- Les administrateurs peuvent effectuer des opérations telles que l'ajout, la suppression et la modification de produits.

#### **4.11. Simulation du processus de paiement :**

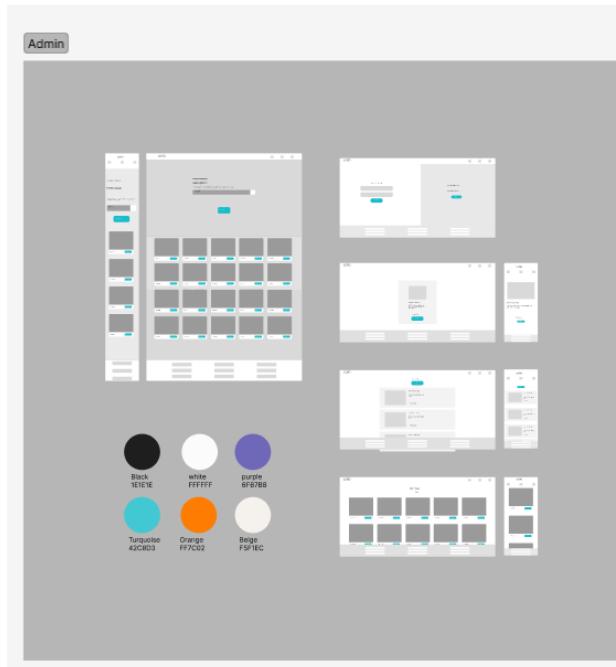
- Un système de validation du panier a été mis en place, permettant aux utilisateurs de simuler le processus de paiement.
- Une solution de paiement réelle peut être envisagée comme fonctionnalité future.



## 5. Réalisations:

### Maquette:

Figma a été utilisé comme outil de conception graphique pour créer la maquette complète du site. Au début du projet, nous avons travaillé en groupe sur la conception de l'interface de l'administrateur, et une partie du design avait déjà été réalisée. Cependant, il restait des pages manquantes à concevoir. J'ai pris en charge cette tâche et j'ai créé les designs nécessaires pour compléter l'ensemble du site. En utilisant Figma, j'ai pu assurer une cohérence visuelle avec le travail précédemment réalisé et garantir une expérience utilisateur fluide et attrayante sur l'ensemble du site.



### Base de données:

Ce projet a commencé initialement en tant que projet de groupe avec une base de données préexistante. Cependant, lorsque j'ai décidé de changer le concept du projet, j'ai effectué des modifications significatives sur la base de données pour mieux correspondre à la nouvelle identité du projet.

J'ai ainsi procédé à une analyse approfondie de la base de données existante, puis j'ai effectué les modifications nécessaires pour l'adapter aux nouvelles fonctionnalités et exigences du projet.



Après avoir réalisé les modélisations conceptuelles de données (MCD), logiques (MLD) et physiques (MPD), j'ai reconstruit la base de données en utilisant les nouvelles structures et relations. J'ai adapté les tables, les attributs et les relations existantes, puis, j'ai ajouté de nouvelles tables pour mieux répondre aux besoins du nouveau concept.

Cette nouvelle version de la base de données, créée après les modélisations MCD, MLD et MPD, reflète désormais de manière plus précise les exigences du projet et permet une meilleure gestion des données selon le nouveau concept.

### **Code:**

Les codes que je vais présenter se concentrent principalement sur les opérations CRUD (Create, Read, Update, Delete). Ces opérations sont essentielles pour permettre aux utilisateurs d'interagir avec les données du site de manière dynamique et personnalisée.

Chaque morceau de code que nous examinerons sera lié à l'une de ces opérations CRUD. Par exemple, nous verrons comment créer de nouvelles entrées dans la base de données pour répondre aux besoins spécifiques des utilisateurs, récupérer des données existantes pour les afficher de manière personnalisée, mettre à jour des informations en fonction des préférences des utilisateurs, et supprimer des enregistrements selon leurs souhaits.

En utilisant ces opérations CRUD, le site devient dynamique et permet aux utilisateurs de personnaliser leur expérience en fonction de leurs besoins. Ils peuvent créer, lire, mettre à jour et supprimer des informations selon leurs préférences, ce qui rend le site plus interactif et adapté à chaque utilisateur. Cela contribue à offrir une expérience utilisateur plus engageante et personnalisée.

## **5.1 Inscription et authentification:**

Le code fourni représente le processus d'inscription et d'authentification dans l'application. L'authentification est l'une des premières interactions de l'utilisateur sur le site, car il doit être connecté pour effectuer des actions telles que l'achat de produits. Voici une brève explication de ce processus :

Lorsque l'utilisateur remplit le formulaire d'inscription avec son adresse e-mail, son mot de passe, son prénom, son nom de famille et son sexe, le formulaire est soumis. Le code PHP associé vérifie que les champs ne sont pas vides et que le paramètre "inscription" dans la requête GET est défini sur "true".

```
***** INSRIPTION *****
if ($_POST != NULL && isset($_GET['inscription']) && $_GET['inscription'] === 'true') [
    $email = $_POST['email'];
    $password = $_POST['password'];
    $confPassword = $_POST['confpassword'];
    $prenom = $_POST['prenom'];
    $nom = $_POST['nom'];
    $sexe = $_POST['sexe'];

    $user->register($_POST);
    echo $user->getMsg();

    die();
}
```

Ensuite, la méthode `register` du contrôleur utilisateur est appelée. À l'intérieur de cette méthode, plusieurs vérifications sont effectuées, notamment la validation des champs du formulaire, la vérification de l'adresse e-mail et du mot de passe, ainsi que le hachage sécurisé du mot de passe. Les données sont ensuite insérées dans la base de données via la méthode `requestRegister` du modèle utilisateur.

```

public function register($post){
    if($this->checkFormNotEmpty($post)){
        if($this->checkEmail($post['email'])){

            if($this->checkPasswordRegex($post['password'], $post['confpassword'])){

                $post = $this->injectionSQL($post);

                $post['password'] = password_hash($post['password'], PASSWORD_BCRYPT);
                $post['avatar'] = "profil_pic_". $post['sexe'] .".svg";

                unset($post['confpassword']);

                $request = new UserModel();
                $request->requestRegister('insert', 'utilisateurs', $post);

                $this->msg = "<p id='succes'>Inscription réussie ! Vous pouvez maintenant vous connecter.</p>";
            }
        }
    }
}

```

```

// REQUETE POUR AJOUTER UN UTILISATEUR EN BASE DE DONNEE.
public function requestRegister($action, $table, $columns){
    $strRequest = $this->requestGenerator($action, $table, $columns);

    $request = $this->connect->prepare($strRequest);
    $request->execute($columns);
}

```

Ce processus permet aux utilisateurs de s'inscrire et de créer un compte dans l'application, ce qui leur donne accès à des fonctionnalités supplémentaires telles que l'achat de produits.



## 5.2 Affichage des produits:

Le code présenté affiche tous les produits sur la page d'accueil du site. Voici comment cela fonctionne :

Dans le fichier "index", nous utilisons la méthode "getAllArticles" du contrôleur pour récupérer tous les articles de la base de données. Ces articles sont stockés dans une variable appelée "\$articles".

```
$articles = $article->getAllArticles();
// var_dump($articles);
foreach ($articles as $article) {
    echo '<div class="product-box">';
    if (!isset($_SESSION['user'])) {
        // echo '<a class="link-product-box" href="/3dfactory/librotopia/src/view/index.php?msg=not-connected!>';
        echo '<a class="link-product-box" href="/3dfactory/librotopia/src/view/connexionRegister.php?msg=empty">';
    } else {
        echo '<a class="link-product-box" href="/3dfactory/librotopia/src/view/detail.php?id= ' . $article['id'] . '">';
    }
    echo " <div class='details'>  ";
    echo "</div>";
    echo "<div class='img-box'><img class='index_pro_img' src='./images/article/" . $article['id'] . ".jpg' alt=''"></div>";
    // echo "Subcategory ID: " . $article['idsubcategory1'] . "<br>";
    echo '<div class="price_buy">';
    echo "Price: €" . $article['prix'] . "<br>";
    // echo " <button>Buy!</button> ";
    if (isset($_SESSION['user'])) {

        echo "<form action=' ' method='POST' class='add_item_form' >
<input type='hidden' name='userId' value='" . $user->getID() . "'>
<input type='hidden' name='itemPrice' value='" . $article['prix'] . "'>
<input type='hidden' name='itemId' value='" . $article['id'] . "'>

<button type='submit' name='addToCart'>Add To Cart</button>
</form>";
    }
    echo " </div> ";
    echo '</div>';
    echo '</a>';
    echo "<br>";
}
```

Ensuite, nous parcourons chaque article à l'aide d'une boucle "foreach". Pour chaque article, nous affichons les détails du produit, tels que le nom, le prix et l'image. Nous utilisons les données de l'article pour construire les éléments HTML nécessaires, tels que les liens vers les pages de détail du produit et les boutons "Ajouter au panier".



Si l'utilisateur n'est pas connecté, le lien vers les détails du produit redirige vers la page de connexion/inscription. Si l'utilisateur est connecté, le lien pointe vers la page de détail du produit correspondant à l'ID de l'article.

Dans le contrôleur, la méthode "getAllArticles" fait appel à la méthode "requestSelectAllArticlesInfos" du modèle. Cette méthode exécute une requête SQL pour sélectionner tous les articles de la table "articles" dans la base de données. Les données récupérées sont renvoyées sous forme d'ensemble de résultats.

contrôleur:

```
public function getAllArticles()
{
    $request = new ArticleModel();
    $data = $request->requestSelectAllArticlesInfos();
    return $data;
}
```

MODEL:

```
public function requestSelectAllArticlesInfos()
{
    $request = $this->connect->prepare("SELECT * FROM articles");
    $request->execute();
    $data = $request->fetchAll(\PDO::FETCH_DEFAULT);

    return $data;
}

// REQUETE POUR COMPTER LE NOMBRE D'ARTICLES TOTAL
```

En résumé, ce code récupère tous les articles de la base de données et les affiche sur la page d'accueil du site. Les utilisateurs peuvent voir les détails des produits et, s'ils sont connectés, ajouter des articles à leur panier en cliquant sur le bouton "Ajouter au panier".



### 5.3 Crédation d'une session de paiement avec Stripe:

Le code présenté concerne la page du panier (cart) et la création d'une session de paiement avec Stripe. Voici comment cela fonctionne :

Dans le fichier de la page du panier, nous parcourons les éléments du panier (panier\_items) à l'aide d'une boucle foreach. Pour chaque élément, nous affichons les détails de l'article, tels que l'image, le titre et le prix. Nous fournissons également un bouton pour supprimer l'article du panier. Les informations de chaque article sont ensuite utilisées pour créer un tableau d'articles (items) qui sera utilisé pour créer la session de paiement avec Stripe.

```
echo ' <div class="cart" > ';
$items = [];
// var_dump($panier_items);
foreach ($panier_items as $item_obj) {
    $articleTitre = $article->showArticleById($item_obj['idArticle']);
    echo ' <div class="article_box_in_cart" > ';
    echo ' <div class="article_img_cart" >  </div>';
    echo ' <div class="article_info" > ';
    echo ' <h5>' . $articleTitre[0]['titre'] . ' </h5><br>';

    echo ' Price : €' . $item_obj['prix'];
    echo " <form action=' ' method='POST'> <input type='hidden' value=" . $item_obj['idArticle'] . " name='item_to_delete'><button type='submit' class=' > ";
    echo ' </div> ';

    echo ' </div> ';
    // echo $item_obj['prix'];
    $item = [
        'name' => $articleTitre[0]['titre'],
        'price' => intval($item_obj['prix'] * 100), // Multiply by 100 to convert to cents
        'currency' => 'eur',
        'quantity' => 1
        // 'images' => ['view/images/article/16.jpg'],
    ];
    $items[] = $item;
}
// var_dump($items);

if (isset($_POST['delete_item_from_cart'])) {
    $panier->deleteItemFromCart($_POST['item_to_delete']);
    header("Location: cart.php");
}
if (isset($_POST['send_to_stripe'])) {
    createCheckout($items);
}
```

Lorsque l'utilisateur clique sur le bouton "send\_to\_stripe" pour passer à la caisse, la fonction createCheckout() est appelée. Cette fonction prend le tableau d'articles (items) en argument et crée une liste d'articles (lineItems) conforme à la structure requise par Stripe. Chaque article du panier est ajouté à cette liste en spécifiant les détails tels que le nom, le prix, la devise et la quantité.

Ensuite, une session de paiement est créée avec Stripe en utilisant la méthode \Stripe\Checkout\Session::create(). Les détails de la session de paiement incluent les méthodes de paiement acceptées, la liste des articles (lineItems), le mode de paiement (mode), les URL de succès et d'annulation. L'URL de succès est l'adresse vers laquelle l'utilisateur sera redirigé après un paiement réussi, tandis que l'URL d'annulation est l'adresse vers laquelle l'utilisateur sera redirigé en cas d'annulation du paiement.

```
\Stripe\Stripe::setApiKey($stripeSecretKey);

// Set your Stripe API key

function createCheckout($items)
{
    // Create an array of line items
    $lineItems = [];
    foreach ($items as $item) {
        $lineItems[] = [
            'price_data' => [
                'currency' => $item['currency'],
                'unit_amount' => $item['price'],
                'product_data' => [
                    'name' => $item['name'],
                    // 'images' => $item['images'], // Replace with your product image URLs
                ],
            ],
            'quantity' => $item['quantity'],
        ];
    }
}

// Create a Stripe Checkout Session
$session = \Stripe\Checkout\Session::create([
    'payment_method_types' => ['card'],
    'line_items' => $lineItems,
    'mode' => 'payment',
    'success_url' => 'http://localhost/3dfactory/librotopia/src/view/succses', // Replace with your success URL
    'cancel_url' => 'https://example.com/cancel', // Replace with your cancel URL
]);
header('Location: ' . $session->url);
exit;
}
```

Enfin, l'utilisateur est redirigé vers la page de paiement Stripe en utilisant l'URL de la session de paiement. Là, il pourra finaliser le paiement en utilisant les méthodes de paiement acceptées.



## **5.4 Ajout d'un article au panier:**

La fonction `addItemToCart` permet d'ajouter un article au panier d'un utilisateur. Le code effectue les étapes suivantes :

```
public function addItemToCart($userId, $prix, $idArticle)
{
    // Check if the article already exists in the cart
    $checkQuery = $this->connect->prepare("SELECT * FROM panier WHERE idUser = :user_id AND idArticle = :article_id");
    $checkQuery->bindParam(':user_id', $userId, \PDO::PARAM_INT);
    $checkQuery->bindParam(':article_id', $idArticle, \PDO::PARAM_INT);
    $checkQuery->execute();

    if ($checkQuery->rowCount() > 0) {
        return false; // Article already exists in the cart
    }

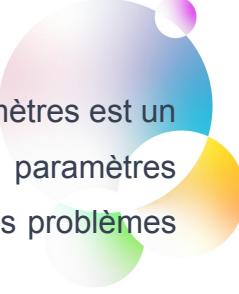
    // Insert the article into the cart
    $insertQuery = $this->connect->prepare("INSERT INTO panier (idUser, prix, idArticle) VALUES (:user_id, :prix, :article_id)");
    $insertQuery->bindParam(':user_id', $userId, \PDO::PARAM_INT);
    $insertQuery->bindParam(':prix', $prix, \PDO::PARAM_STR);
    $insertQuery->bindParam(':article_id', $idArticle, \PDO::PARAM_INT);
    $insertQuery->execute();

    // Check if the article was successfully added
    if ($insertQuery->rowCount() > 0) {
        return true; // Article added to the cart successfully
    } else {
        return false; // Failed to add the article to the cart
    }
}
```

1. Vérification de l'existence de l'article dans le panier : Le code exécute une requête de sélection pour vérifier si l'article que l'utilisateur souhaite ajouter existe déjà dans son panier.
2. Insertion de l'article dans le panier : Si l'article n'existe pas déjà dans le panier, le code exécute une requête d'insertion pour ajouter l'article au panier de l'utilisateur.
3. Vérification de l'ajout réussi : Le code vérifie si l'opération d'insertion a réussi en vérifiant le nombre de lignes affectées par la requête. Si au moins une ligne a été affectée, cela signifie que l'article a été ajouté avec succès au panier.

En termes de sécurité, ce code utilise des requêtes préparées avec des liaisons de paramètres pour se prémunir contre les attaques par injection SQL. Cela garantit que les valeurs des variables sont correctement traitées et empêche l'exécution de code SQL malveillant.

Les attaques par injection SQL sont une forme courante d'attaque de sécurité où un attaquant insère des instructions SQL malveillantes dans une requête afin de compromettre la base de données ou d'accéder à des informations sensibles. Pour prévenir ces attaques, il est crucial de mettre en place des mesures de sécurité appropriées, telles que l'utilisation de requêtes préparées avec des liaisons de paramètres.



Dans le code présenté, l'utilisation de requêtes préparées avec des liaisons de paramètres est un bon moyen de se protéger contre les attaques par injection SQL. Les liaisons de paramètres permettent de séparer clairement les données des instructions SQL, évitant ainsi les problèmes liés à l'interprétation incorrecte des caractères spéciaux.

## 5.5 Autocomplétion:

Le code présenté implémente la fonctionnalité d'autocomplétion dans le champ de recherche du site. Lorsque l'utilisateur saisit une valeur dans le champ de recherche, cet événement déclenche une fonction qui effectue les actions suivantes :

```
// alert("check");
let search = document.getElementById('searchInput');
const results = document.getElementById('search_results');
search.addEventListener('input',async()=>{
  //clean the results before new search or when deleted
  results.innerHTML = '';

  // get the value from search input as key
  // NOTE: 'search' was selected by its class name ('searchTxt') outside this function
  key = search.value;
  // console.log(key);
  // use this key and our own 'getJsonByKey()' function to get the json object
  // example :
  //   jsonObj = [{"id": "1", "name": "Three Valleys", "km": "600"}...]
  let jsonObj = await getJsonByKey(key);
  // display the result in our html
  displayResult(jsonObj);

  // console.log(jsonObj);
})
```

1. Nettoyage des résultats précédents : Avant de procéder à une nouvelle recherche ou en cas de suppression du contenu du champ de recherche, les résultats précédents sont effacés en vidant le contenu de l'élément HTML ayant pour ID "search\_results".
2. Récupération de la valeur saisie dans le champ de recherche : La valeur saisie par l'utilisateur est stockée dans une variable appelée "key".
3. Appel à la fonction "getJsonByKey()" : Cette fonction asynchrone est utilisée pour récupérer un objet JSON correspondant à la clé de recherche. Elle envoie une requête vers le script PHP "autocomplete.php" en incluant la clé de recherche comme paramètre dans l'URL de la requête.

```

//fetch the key and send it to PHP
let getJsonByKey = async (key) => [
  // create an empty array as result
  let result = [];
  // fetch the php data using the given 'key' as keyword
  // let response = await fetch('/librotopia/src/view/admin_article.php?keyWord=' + key);
  let response = await fetch('/librotopia/src/model/autocomplition.php?keyWord=' + key);

  // if the response is ok / no errors
  if (response.ok) {
    // update the result with the json data
    result = await response.json();
  }
  if(search.value===''){
    result = [];
  }

  // return the result
  // the result will be empty if no response was found
  return result;
];
// present the results of the search

```

4. Affichage des résultats : La fonction "displayResult()" est appelée pour afficher les résultats de la recherche dans le HTML de la page. Cette fonction parcourt l'objet JSON obtenu et crée des éléments d'ancre (<a>) contenant les informations pertinentes des résultats de recherche. Ces éléments sont ensuite ajoutés à l'élément HTML ayant pour ID "search\_results".

```

// present the results of the search
let displayResult = (obj) => {
  // remove all list items or children of '<ul id="result">' element
  results.innerHTML = '';

  // for each item in the given obj
  obj.forEach((item) => {

    let a = document.createElement("a");

    a.textContent = item.titre;
    a.setAttribute('href', "/3dfactory/librotopia/src/view/detail.php?article="+item.titre);
    // a.setAttribute('class', "article_link");

    results.appendChild(a);
  });
};


```

Le script PHP "autocomplete.php" est responsable de récupérer les données correspondant à la clé de recherche fournie. Il effectue une vérification de sécurité pour s'assurer que l'utilisateur est connecté et qu'il possède les droits d'accès appropriés. Ensuite, il utilise le contrôleur et le modèle associés pour effectuer une recherche d'articles correspondant à la clé de recherche et renvoie les résultats sous forme de tableau JSON.

```
1 <?php
2
3 namespace App\view;
4
5 require_once("../autoload.php");
6 session_start();
7
8 use App\controller\ArticleController;
9 use App\model\ArticleModel;
10
11 if (!isset($_SESSION['user']) || $_SESSION['user']->getRole() != "admin") {
12     header('location: index.php');
13 }
14 $search = new ArticleModel;
15
16 $article = new ArticleController;
17
18 // -----
19 $key = $_GET['keyWord'];
20
21 |
22 $search->searchArticle($key);
23
```

## 6. Vulnérabilités en termes de sécurité



```
135 public function searchArticle($key)
136 {
137     // $request = $this->connect->prepare("SELECT titre FROM articles WHERE titre LIKE '%$key%'");
138     $request = $this->connect->prepare("SELECT * FROM articles WHERE titre LIKE '%$key%'");
139
140     $request->execute();
141     $data = $request->fetchAll(\PDO::FETCH_DEFAULT);
142     echo json_encode($data);
143     // return $data;
144     // var_dump($data);
145 }
```

Le code présenté comporte une vulnérabilité potentielle qui pourrait permettre une attaque connue sous le nom d'injection de code malveillant. Cette vulnérabilité se produit lorsqu'une entrée utilisateur n'est pas correctement filtrée ou échappée avant d'être utilisée dans une requête SQL.

Dans ce cas spécifique, la vulnérabilité réside dans le fait que la clé de recherche fournie par l'utilisateur est directement intégrée dans la requête SQL sans être correctement traitée. Cela signifie qu'un attaquant malveillant pourrait potentiellement exploiter cette vulnérabilité en fournissant une clé de recherche contenant des caractères spéciaux ou des instructions SQL malveillantes.

Pour atténuer cette vulnérabilité, il est recommandé d'utiliser des techniques appropriées pour filtrer et échapper les entrées utilisateur avant de les utiliser dans des requêtes SQL. Cela peut inclure l'utilisation de fonctions de filtrage ou de paramètres liés dans les requêtes préparées, qui permettent de traiter les entrées utilisateur de manière sécurisée.



## 7. Recherches à partir de sites anglophones

Au cours de la réalisation du projet, j'ai rencontré des difficultés pour rendre le site responsive. Pour résoudre cette problématique, j'ai visionné des vidéos explicatives sur YouTube, qui m'ont permis de comprendre les principes fondamentaux et les meilleures pratiques pour créer un design responsive. J'ai également consulté des forums de programmation où j'ai pu trouver des réponses à mes questions spécifiques et bénéficier des conseils de la communauté des développeurs.

En approfondissant mes recherches, j'ai trouvé des informations précieuses sur l'utilisation des requêtes media queries. Grâce à ces requêtes, j'ai pu créer des règles CSS conditionnelles qui s'adaptent dynamiquement à différents environnements et garantissent ainsi une expérience utilisateur fluide.

Avant ce projet, j'ai toujours commencé la conception d'un site en me focalisant sur la version pour les écrans de bureau, puis j'ai essayé d'ajuster le design pour les appareils mobiles en utilisant de nombreuses requêtes media queries. Cependant, mes recherches m'ont permis de comprendre qu'il est préférable de prendre une approche différente. Selon les conseils que j'ai trouvés, il est recommandé de commencer par concevoir la version pour les appareils mobiles, puis d'apporter de petites modifications en utilisant les media queries pour adapter le design aux écrans plus grands.

Cette approche, appelée "mobile first", permet de s'assurer que la version mobile du site a une mise en page simple et fonctionnelle, puis de développer progressivement des mises en page plus complexes pour les écrans plus grands. En suivant cette méthodologie, on s'assure que le site est parfaitement adapté aux appareils mobiles dès le départ, puis on l'améliore pour les écrans plus grands.

Grâce à cette recherche et à l'adoption de l'approche "mobile first", j'ai pu constater une amélioration significative de la qualité et de l'efficacité de mes conceptions responsives. En commençant par les appareils mobiles, j'ai pu concevoir des mises en page qui fonctionnent bien sur tous les types d'appareils.

Un autre point important que j'ai appris au cours de mes recherches, notamment grâce à une vidéo YouTube, est que avant même d'ajouter du CSS à un site, celui-ci est déjà responsive. Il peut ne pas être esthétiquement plaisant, mais il s'adapte déjà correctement à différents appareils et tailles d'écran. Cette notion m'a fait prendre conscience qu'un problème de responsivité dans la conception d'un site est souvent dû à une ligne de CSS inutile qui entre en conflit avec une autre propriété.

Cette prise de conscience m'a rendu beaucoup plus attentif aux propriétés CSS que j'ajoute à une classe. J'ai adopté une approche plus rigoureuse en n'ajoutant que les propriétés nécessaires pour obtenir le résultat souhaité, tout en évitant les redondances et les conflits potentiels. Cela m'a permis de préserver la



responsivité du site et de garantir que les éléments s'ajustent correctement sur tous les appareils, sans causer de problèmes d'affichage ou de mise en page inattendus.

## Responsive design made easy



Kevin Powell

781K subscribers

Subscribe

Source: [Responsive design made easy](#)

### 8. Extrait d'un site anglophone

Voici l'extrait d'un des sites qui m'ont aidé à mieux comprendre comment rendre un site responsive.

## Active learning: mobile first responsive design

Broadly, you can take two approaches to a responsive design. You can start with your desktop or widest view and then add breakpoints to move things around as the viewport becomes smaller, or you can start with the smallest view and add layout as the viewport becomes larger. This second approach is described as **mobile first** responsive design and is quite often the best approach to follow.

The view for the very smallest devices is quite often a simple single column of content, much as it appears in normal flow. This means that you probably don't need to do a lot of layout for small devices — order your source well and you will have a readable layout by default.

The below walkthrough takes you through this approach with a very simple layout. In a production site you are likely to have more things to adjust within your media queries, however the approach would be exactly the same.

Conception responsive mobile-first :

Il existe deux approches principales pour concevoir un site responsive. La première consiste à commencer par la version de bureau et à ajouter des points de rupture pour les appareils plus petits. La deuxième approche, connue sous le nom de mobile-first, consiste à concevoir d'abord pour les appareils mobiles, puis à ajouter des mises en page pour les écrans plus grands. Cette approche est souvent considérée comme la plus efficace.



Pour les petits appareils, la mise en page est généralement une simple colonne de contenu, ce qui garantit une lisibilité par défaut. Cette approche permet de minimiser les ajustements de mise en page nécessaires pour les appareils plus petits.

Il est recommandé de suivre cette approche lors de la conception responsive pour garantir une expérience utilisateur optimale sur tous les appareils.

Source: [https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS\\_layout/Media\\_queries](https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout/Media_queries)