**Rapport du Projet : Boutique en Ligne Armurerie**

**Contexte et Utilité du Projet**

Le projet consiste en la création d'une boutique en ligne spécialisée dans la vente d'armes fictives, nommée **"Armurerie"**. L’objectif principal est de proposer une expérience utilisateur fluide, avec une gestion claire et efficace des produits, des paniers et des commandes. Le projet est divisé en deux parties principales : une interface **utilisateur** (clients) et une interface **administrateur**. Les fonctionnalités sont soutenues par une base de données MongoDB qui assure le stockage et la gestion des informations essentielles.

**Les Collections de la Base de Données**

La base de données MongoDB se nomme **"armurerie"** et contient les collections suivantes :

1. **Products** : Contient les produits disponibles à la vente.
   * Champs : \_id, name, description, price, stock, image, created\_at.
   * Exemple : Une arme avec un nom, une description, un prix, une image et des informations de stock.
2. **Users** : Contient les informations des administrateurs et des utilisateurs enregistrés.
   * Champs : \_id, username, password, role (admin ou user).
   * Exemple : L’administrateur a des droits spécifiques, tandis que les utilisateurs peuvent acheter des produits.
3. **Cart** : Gère les informations des paniers des utilisateurs.
   * Champs : \_id, username, products.
   * Exemple : Un utilisateur ajoute plusieurs produits à son panier avant de passer commande.
4. **Orders** : Enregistre les commandes validées.
   * Champs : \_id, user\_id, products, total\_price, status, created\_at.
   * Exemple : Après validation d’un panier, les informations sont enregistrées comme une commande.

**Fonctionnalités du Projet**

**1. Accès et Authentification**

* **Page de Connexion (LOGIN.PHP)** :
  + Permet à un utilisateur (admin ou client) de se connecter grâce à un formulaire.
  + Vérification des identifiants via LOGPROCESS.PHP.
  + Redirection :
    - Si admin : vers l’espace administrateur.
    - Si utilisateur : vers la boutique principale.
* **Création de Compte (REGISTER.PHP)** :
  + Permet aux nouveaux utilisateurs de s’enregistrer via un formulaire.
  + Les informations sont stockées dans la collection **Users** via REGISTERPROCESS.PHP.
* **Déconnexion (LOGOUT.PHP)** :
  + Termine la session active et redirige vers la page de connexion.

**2. Interface Utilisateur (Client)**

* **Page Principale (INDEX.PHP)** :
  + Affiche la liste des produits disponibles (récupérés depuis la collection **Products**).
  + Chaque produit inclut : nom, description, prix, stock restant, et un bouton "Ajouter au panier".
  + Gestion via ADD\_TO\_CART.PHP : Lorsqu’un produit est ajouté, les informations sont mises à jour dans la collection **Cart**.
* **Panier (CART.PHP)** :
  + Affiche les produits ajoutés au panier avec leurs quantités, prix et un total.
  + Gestion des produits :
    - Suppression via REMOVECART.PHP.
    - Validation via SAVE\_CART.PHP, qui transfère les données du panier vers la collection **Orders**.
* **Confirmation de Commande (CONFIRMATION.PHP)** :
  + Affiche un message de validation après le passage de commande.

**3. Interface Administrateur**

* **Tableau de Bord (ADMIN.PHP)** :
  + Liste tous les produits disponibles dans la boutique avec des options pour les gérer.
* **Gestion des Produits** :
  + **Ajout de Produits (ADDPRODUCT.PHP)** :
    - Formulaire permettant d’ajouter un nouveau produit à la collection **Products**.
  + **Modification de Produits (UPDATEPRODUCT.PHP)** :
    - Permet de mettre à jour les informations d’un produit (stock, description, etc.).
    - Les modifications sont immédiatement reflétées côté utilisateur et admin.
  + **Suppression de Produits (DELETEPRODUCT.PHP)** :
    - Supprime un produit de la base de données, rendant celui-ci indisponible partout.

**Processus Techniques**

**Gestion des Stocks**

* Lorsqu’un utilisateur ajoute un produit au panier, la quantité disponible diminue dynamiquement dans la boutique.
* Si un produit est entièrement vendu :
  + Il est marqué comme "Épuisé".
  + L’administrateur peut recharger le stock depuis son tableau de bord.

**Structure CSS**

Chaque page est associée à un fichier CSS spécifique pour un rendu adapté :

* **LOGIN.PHP** : STYLE.CSS
* **REGISTER.PHP** : STYLEREJ.CSS
* **ADMIN.PHP** : STYLEADMIN.CSS
* **INDEX.PHP** : STYLEINDEX.CSS
* **CART.PHP** : STYLECART.CSS
* **CONFIRMATION.PHP** : STYLECONFIRM.CSS
* **UPDATEPRODUCT.PHP** : STYLEUPDATE.CSS

**Installation et Lancement**

**Prérequis**

* **Serveur Web** : WampServer.
* **Base de Données** : MongoDB.

Après avoir installé mongo db, il faut exécuter une commande dans le terminal de votre ordinateur ensuite vous pourrez créer la base de donnée et les collections. Voici la commande a exécuter dans le terminal :

* **PHP** : Version 7.4 ou supérieure.

**Étapes d’Installation**

1. Clonez le projet :

git clone https://github.com/dabirebensada/nosql.git

cd nosql

1. Configurez la base de données MongoDB :
   * Importez les collections initiales (products, users, etc.).
2. Modifiez le fichier DB\_CONNECT.PHP avec vos informations de connexion MongoDB.
3. Placez les fichiers dans le dossier racine de votre serveur Web notamment WWW pour le localhost PHP ou par ex., htdocs pour XAMPP.

**Lancement du Projet**

* Démarrez le serveur web.
* Accédez à l’application via :

nosql/login.php

**Lancer les Tests**

**Tests Fonctionnels**

* **Connexion/Inscription** :
  + Testez la connexion avec différents utilisateurs (admin et client).
  + Vérifiez la création de compte.
* **Boutique Utilisateur** :
  + Ajoutez des produits au panier.
  + Supprimez des produits et validez une commande.
* **Gestion Admin** :
  + Ajoutez, modifiez et supprimez des produits.

**Validation des Données**

* Assurez-vous que les données saisies (produits, commandes, utilisateurs) apparaissent correctement dans MongoDB.