Projet de base de donne

Tistea Stefan-Valentin

1220FB(iot)

Introduction:

Mon projet consiste en un système de gestion de boutique en ligne, élaboré dans le but de permettre aux utilisateurs d'explorer, d'ajouter des produits à leur panier, et de finaliser des commandes. Ce rapport présente une documentation détaillée sur les diverses fonctionnalités du système, ainsi que des informations approfondies concernant sa conception et sa mise en œuvre.

Langages et technologie:

HTML (HyperText Markup Language):

Utilisé pour structurer et définir le contenu des pages web. Intègre des balises pour créer des éléments tels que formulaires, images, liens, etc.

JavaScript:

Langage de programmation côté client qui améliore l'interactivité des pages web. Il permet de créer des fonctionnalités dynamiques et réactives, améliorant ainsi l'expérience utilisateur.

PHP:

Langage de script côté serveur principalement utilisé pour le développement web. Il facilite la gestion des formulaires, des sessions utilisateur, et la connexion à des bases de données, contribuant ainsi à la création d'applications web dynamiques.

SQL (Structured Query Language):

Langage de programmation utilisé pour gérer et manipuler des bases de données relationnelles. Permet d'effectuer des opérations telles que l'insertion, la mise à jour, la suppression et la récupération de données.

CSS (Cascading Style Sheets):

Utilisé pour styliser et formater le contenu des pages web. Permet de définir la présentation visuelle, y compris les couleurs, les polices, les marges, etc., pour assurer une expérience utilisateur esthétiquement agréable.

Structures du project:

-index.php (La page principale du site)

-db.php(Connexion à la base de données)

- cart.php (Ce fichier affiche le contenu du panier)

-login.php(La page de login)

-loginphp.php(Vérification de compte)

-new 1.php(Page de contact)

-insert.php(Insérez le message envoyé dans la base de données)

-result.php(Renvoie les messages de la base de données)

-js folder(le code javascript pour la mise en page)

-css folder(le code css pour la mise en page)

-images folder(le dossier images)

-database folder(le dossier de la base de données)

Structures de Base de Données :

Dans le cadre de nos bases de données pour le magasin en ligne, chaque base de données est conçue avec des tables distinctes, chacune jouant un rôle spécifique dans la gestion des informations. Examinons la structure de chaque base de données :

1. Base de Données 'contact' :

Table 'contact' :

Colonnes : 'name', 'email', 'subject', 'message'

Stocke les informations de contact des utilisateurs.

2. Base de Données 'login' :

Table 'login' :

Colonnes : 'id', 'username', 'password'

Stocke les informations d'identification des utilisateurs.

3. Base de Données 'store' :

Table 'products' :

Colonnes : 'id', 'Name', 'Price', 'Descript', 'image'

Stocke les détails des produits disponibles dans le magasin en ligne.

Relations entre les Bases de Données :

Aucune relation explicite n'est définie dans les données fournies. Chaque base de données semble être indépendante, traitant des aspects spécifiques du magasin en ligne.

Pourquoi Utiliser des Relations :

Réduction de la Redondance : En établissant des relations, on évite la duplication inutile des données. Par exemple, les informations du client ne sont stockées qu'une seule fois, même s'il passe plusieurs commandes.

Consistance des Données :

Les relations assurent la cohérence des données, facilitant la mise à jour en un seul endroit.

Optimisation des Requêtes :

Les relations facilitent la récupération d'informations via des requêtes qui joignent plusieurs tables, permettant des analyses plus complexes de manière efficace.

Maintenabilité et Évolutivité :

La structure relationnelle rend la base de données plus facile à comprendre et à maintenir, tout en offrant une flexibilité pour ajouter de nouvelles fonctionnalités sans altérer la structure existante.

Cette approche relationnelle contribue à une gestion efficace des données pour assurer la cohérence, la fiabilité et l'évolutivité du système.

1.db.php:

```php
<?php
$host = "localhost";
$username = "root";
$password = "";
$database = "store";

$conn = new mysqli($host, $username, $password, $database);

if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
?>
```

$host, $username, $password, $database : Ces variables stockent les informations nécessaires à la connexion à la base de données, telles que l'hôte MySQL, le nom d'utilisateur, le mot de passe, et le nom de la base de données.

new mysqli($host, $username, $password, $database) : Crée une nouvelle instance de la classe mysqli pour établir la connexion à la base de données.

$conn->connect_error : Vérifie si la connexion a échoué en inspectant l'erreur de connexion.

die("Échec de la connexion : " . $conn->connect_error) : Termine le script et affiche un message d'erreur en cas d'échec de la connexion.

Ce fichier est crucial pour garantir une connexion réussie à la base de données avant d'effectuer des opérations de lecture ou d'écriture. Vous pouvez l'inclure dans d'autres scripts PHP qui nécessitent un accès à la base de données.

2.login.php:

Fichier PHP pour la Page de Connexion

J'ai conservé toutes les parties HTML du fichier, mais j'ai retiré le contenu pour simplifier la présentation.

J'ai conservé les balises PHP et commenté celles qui semblent être en commentaire dans le fichier original.

Notez que les parties commentées commencent par // et sont donc ignorées dans l'exécution du code PHP. Vous pouvez les décommenter si nécessaire.

3.loginphp.php

```php
<?php
$conn = mysqli_connect("localhost", "root", "", "login");
if ($conn === false) {
    die("ERROR: Could not connect. " . mysqli_connect_error());
}

$name = $_REQUEST['username'];
$password = md5($_REQUEST['parola']);

$query = "SELECT * FROM login";
$OK = 0;

$mysqli = new mysqli("localhost", "root", "", "login");

if ($result = $mysqli->query($query)) {
    while ($row = $result->fetch_assoc()) {
        if ($name == $row['username']) {
            if ($password == $row['password']) {
                $OK = 1;
                header("Location: index.php");
            }
        }
    }

    if ($OK == 0) {
          echo '<script type="text/javascript">
            alert("passwrod error");
            window.location.href= "login.php";
        </script>
        ';
```

Utilise $_REQUEST pour récupérer les valeurs du formulaire (username et parola) soumises via la méthode POST.

Exécute une requête SQL pour récupérer toutes les lignes de la table "login".

Utilise une boucle pour parcourir les résultats de la requête et compare le nom d'utilisateur et le mot de passe avec ceux soumis dans le formulaire.

Si une correspondance est trouvée, redirige l'utilisateur vers "index.php". Sinon, affiche une alerte indiquant une erreur de mot de passe et redirige l'utilisateur vers "login.php".

4.index.php

```php
                    </h2>
<?php
$result = $conn->query("SELECT * FROM products");

                    if ($result->num_rows > 0) {
                        while ($row = $result->fetch_assoc()) {
                            echo '<div class="col-md-4">';
                            echo '<div class="card mb-4">';
                            echo '<img src="images/'. $row['Name'] .'.jpg" class="card-img-top" alt="Product Image">';
                            echo '<div class="card-body">';
                            echo '<h5 class="card-title">' . $row['Name'] . '</h5>';
                            echo '<p class="card-text">' . $row['Descript'] . '</p>';
                            echo '<p class="card-text">Price: $' . $row['Price'] . '</p>';
                            echo '<form method="post" action="index.php">';
                            echo '<input type="hidden" name="product_id" value="' . $row['id'] . '">';
                            echo '<label for="quantity">Quantity:</label>';
                            echo '<input type="number" name="quantity" value="1" min="1">';
                            echo '<button type="submit" name="buy" class="btn btn-primary">Add to Cart</button>';
                            echo '</form>';
                            echo '</div>';
                            echo '</div>';
                            echo '</div>';
                        }
                    } else {
                        echo "No products found";
                    }
                    $conn->close();
```

Ce code récupère les produits à partir de la base de données, puis génère des cartes Bootstrap pour afficher chaque produit sur la page. Il inclut également un formulaire pour permettre aux utilisateurs d'ajouter des produits à leur panier.

5.cart.php

```php
<?php
$totalPrice = 0;
if (!empty($_SESSION['cart'])) {
foreach ($_SESSION['cart'] as $productId => $quantity) {
    $productResult = $conn->query("SELECT * FROM products WHERE id = $productId");
    if ($productResult->num_rows > 0) {
        $product = $productResult->fetch_assoc();
        echo '<div class="col-md-4">';
        echo '<div class="card mb-4">';
        echo '<div class="card-body">';
         echo '<img src="'. $product['Name'] .'.jpg" class="card-img-top" alt="Product Image">';
        echo '<h5 class="card-title">' . $product['Name'] . '</h5>';
        echo '<p class="card-text">Quantity: ' . $quantity . '</p>';
        echo '<p class="card-text">Price: $' . $product['Price'] * $quantity . '</p>';
         $productTotal = $product['Price'] * $quantity;
            echo '<p class="card-text">Price: $' . $productTotal . '</p>';
            $totalPrice += $productTotal;
        echo '<form method="post" action="cart.php">';
        echo '<input type="hidden" name="remove_product_id" value="' . $productId . '">';
        echo '<button type="submit" name="remove" class="btn btn-danger">Remove</button>';
        echo '</form>';
        echo '</div>';
        echo '</div>';
        echo '</div>';
    }
}
     echo '<div class="col-lg-12 text-center">';
    echo '<h4>Total Price: $' . $totalPrice . '</h4>';
    echo '</div>';
else {
  echo '<div class="col-lg-12 text-center">';
```

Inclut les détails de connexion à la base de données.

Démarrage de la session PHP :

Initialise ou restaure une session enregistrée sur le serveur.

Gestion de la suppression de produit lorsque le formulaire est soumis :

Si une requête POST est reçue et qu'elle contient une clé 'remove', cela signifie qu'un produit doit être supprimé du panier. L'ID du produit à supprimer est récupéré à partir des données du formulaire, et le produit correspondant est retiré du panier.

Affichage des produits dans le panier :

Cette boucle parcourt chaque produit dans le panier, récupère ses détails depuis la base de données, et effectue l'affichage des informations du produit.

Affichage du prix total du panier :

Calcule et affiche le prix total de tous les produits dans le panier.

Gestion de la suppression de produit (encore) :

Cette partie gère à nouveau la suppression de produit lorsque le formulaire est soumis, au cas où il y aurait une autre occurrence après le contenu principal.

En résumé, le fichier cart.php gère la logique côté serveur pour l'affichage des produits dans le panier, la suppression de produits et le calcul du prix total du panier.

5.insert.php

```php
<?php
$conn = mysqli_connect("localhost", "root", "", "contact");
if($conn === false){
    die("ERROR: Could not connect. "
        . mysqli_connect_error());
}
$name = $_REQUEST['name'];
$email = $_REQUEST['email'];
$subject = $_REQUEST['subject'];
$message = $_REQUEST['message'];
$sql = "INSERT INTO contact VALUES ('$name',
    '$email','$subject', '$message')";
if(mysqli_query($conn, $sql)){

    echo "<br><br><br><br><h3>Your message was added!<br>";
    echo " <br>You will get a notice on your email when we accept it!<br>";
    echo nl2br("<b>Your name and email :</b> \n$name $email\n ");
    $headers = "From: webmaster@example.com" . "\r\n" .
"CC: somebodyelse@example.com";
//mail("stefantastea4@gmail.com", $subject, $message, $headers);
//mail($email,'Contact istoria arithecturii', 'Mesajul a fost trimis');
} else{
    echo "ERROR: Hush! Sorry $sql. "
        . mysqli_error($conn);
}
mysqli_close($conn);
?>
```

Non inclus dans le code, mais nécessaire pour se connecter à la base de données. Vérifiez que le fichier existe et inclut les informations de connexion.

Démarrage de la session PHP :

Non inclus dans le code HTML/PHP, mais essentiel si vous avez l'intention d'utiliser les sessions PHP. Assurez-vous que cela est géré dans d'autres parties de votre application si nécessaire.

Connexion à la base de données :

Établit une connexion à la base de données MySQL avec les détails fournis (localhost, root, mot de passe vide, base de données "contact").

Récupération des données du formulaire :

Récupère les valeurs des champs de formulaire (nom, email, sujet, message) à partir des données de la requête.

Insertion des données dans la base de données :

Exécute une requête SQL pour insérer les données dans la table "contact".

Affichage du résultat de l'opération :

Affiche un message indiquant si l'opération d'insertion a réussi ou échoué.

Envoi d'un e-mail (commenté) :

Le code pour envoyer des e-mails est commenté. Si décommenté, il enverrait des e-mails aux adresses spécifiées.

Fermeture de la connexion à la base de données :

Ferme la connexion à la base de données après l'exécution des requêtes.

En résumé, ce fichier gère la réception des données du formulaire, les insère dans une base de données, affiche un message de réussite/échec, et inclut la possibilité d'envoyer des e-mails (actuellement commenté).

6.result.php

```php
<?php
$username = "root";
$password = "";
$database = "contact";
$mysqli = new mysqli("localhost", $username, $password, $database);

$query = "SELECT * FROM contact";
echo '<center>';
echo "<br><br><br><br><b> <center><h2>Message list</h2><hr></center> </b> <br>";
if ($result = $mysqli->query($query)) {

    while ($row = $result->fetch_assoc()) {
        $name = $row["name"];
        $email = $row["email"];
        $subject=$row["subject"];
        $message=$row["message"];
        echo 'The name:';
        echo '<br>';
        echo '<b>'.$name.'</b><br />';
        echo '<br>';
        echo 'The mail';
        echo '<br>';
        echo '<b>'.$email.'</b><br />';
        echo '<br>';
        echo '<hr>';
    }
echo '</center>';
$result->free();
}
?>
                </div>
```

Query Execution and Data Retrieval:

Executes a SELECT query to fetch all records from the "contact" table.

Data Display:

Iterates through the result set and displays information for each record, including name, email, subject, and message.

HTML Output:

Embeds the PHP code within the HTML structure, displaying a header, navigation, a parallax section, a list of messages, and a footer.

This script fetches records from the "contact" table and displays relevant information in a formatted manner. The displayed information includes the

name and email of each contact record. It seems to be designed to present a list of messages received through a contact form.